Optimation OptiLogic Driver

Table of Contents

Optimation OptiLogic Driver

Help version 1.018

CONTENTS

Overview

The Optimation OptiLogic Driver provides a reliable way to connect Optimation OptiLogic devices to OPC client applications; including HMI, SCADA, Historian, MES, ERP and countless custom applications. It is intended for use with Optimation OptiLogic Series Remote Terminal Units, I/O Modules and Operator Panels.

Setup

Supported RTUs

OL4054
OL4058

Supported Modules

OL2208, OL2211, OL2201
OL2108, OL2109, OL2111
OL2408, OL2418
OL2252
OL3406, OL3420, OL3440, OL3850

Communication Protocol

Ethernet using Winsock V1.1 or higher.

Channel and Device Limits

The maximum number of channels supported by this driver is 100. The maximum number of devices supported by this driver is 1024 per channel.

Device IDs

The Device ID is used to specify the device IP address along with an OptiLogic Node Number on the Ethernet network. Device IDs are specified as either YYY.YYY.YYY.YYY:XXX. The YYY designates the device IP address: each YYY byte should be in the range of 0 to 255. The XXX designates the OptiLogic node number of the RTU and can be in the range of 0 to 99.

Note: Each device on the channel must be uniquely identified by its own IP address and Node ID. Users can use the OptiLogic Update Tool software supplied by Optimation to configure an RTU's IP address. Users can set the RTU Node ID using the address switches under the base unit cover.

Precision

The default precision will be used when writing float data to the device.

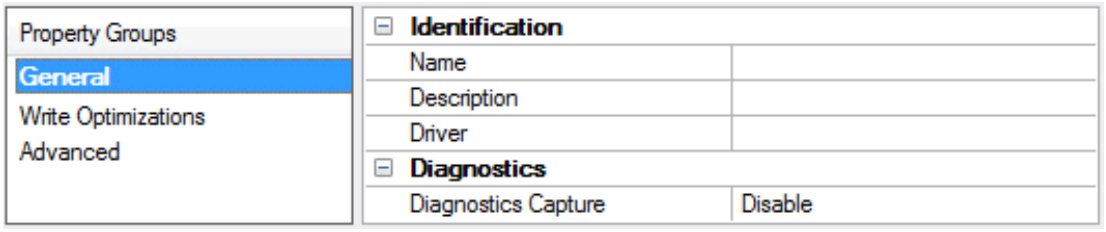
For example, if 50.02 is written from the client, the float value received by the server could be 50.019999. The driver needs to know the precision that should be used for this value. Thus, a bit number (representing the precision) should be appended to the address. A bit number of three would send the previous example as 50.019 with a precision of 3.

Configuration

The Optimation OptiLogic Driver automatically detects the location of each module in the RTU. While certain I/O modules are required to use Slot 0, placement of others is flexible. *For more information, refer to the specific module's owner manual.* The number of slots varies with RTU: 4 for OL4054 and 8 for OL4058. Only one operator panel can be connected per RTU.

Channel Properties — General

This server supports the use of simultaneous multiple communications drivers. Each protocol or driver used in a server project is called a channel. A server project may consist of many channels with the same communications driver or with unique communications drivers. A channel acts as the basic building block of an OPC link. This group is used to specify general channel properties, such as the identification attributes and operating mode.



Identification

Name: User-defined identity of this channel. In each server project, each channel name must be unique. Although names can be up to 256 characters, some client applications have a limited display window when browsing the OPC server's tag space. The channel name is part of the OPC browser information. The property is required for creating a channel.

For information on reserved characters, refer to "How To... Properly Name a Channel, Device, Tag, and Tag Group" in the server help.

Description: User-defined information about this channel.

Many of these properties, including Description, have an associated system tag.

Driver: Selected protocol / driver for this channel. This property specifies the device driver that was selected during channel creation. It is a disabled setting in the channel properties. The property is required for creating a channel.

Note: With the server's online full-time operation, these properties can be changed at any time. This includes changing the channel name to prevent clients from registering data with the server. If a client has already acquired an item from the server before the channel name is changed, the items are unaffected. If, after the channel name has been changed, the client application releases the item and attempts to re-acquire using the old channel name, the item is not accepted. With this in mind, changes to the properties should not be made once a large client application has been developed. Utilize the User Manager to prevent operators from changing properties and restrict access rights to server features.

Diagnostics

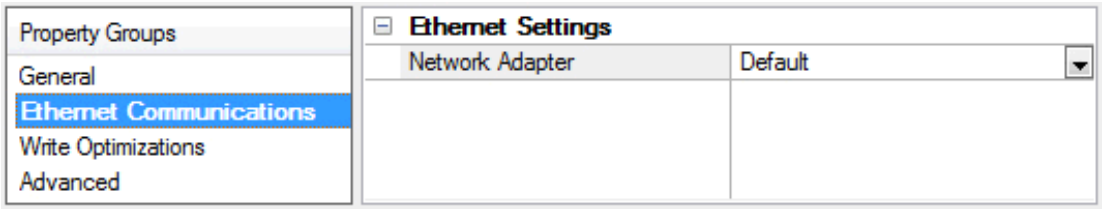
Diagnostics Capture: When enabled, this option makes the channel's diagnostic information available to OPC applications allows the usage of statistics tags that provide feedback to client applications regarding the operation of the channel. Because the server's diagnostic features require a minimal amount of overhead processing, it is recommended that they be utilized when needed and disabled when not. The default is disabled.

Note: This property is not available if the driver does not support diagnostics.

For more information, refer to "Communication Diagnostics" and "Statistics Tags" in the server help.

Channel Properties — Ethernet Communications

Ethernet Communication can be used to communicate with devices.

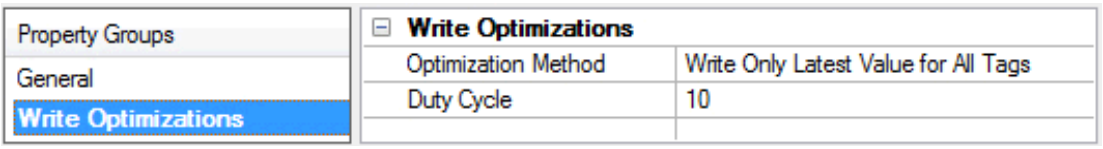


Ethernet Settings

Network Adapter: Specify the network adapter to bind. When left blank or Default is selected, the operating system selects the default adapter.

Channel Properties — Write Optimizations

As with any server, writing data to the device may be the application's most important aspect. The server intends to ensure that the data written from the client application gets to the device on time. Given this goal, the server provides optimization properties that can be used to meet specific needs or improve application responsiveness.



Write Optimizations

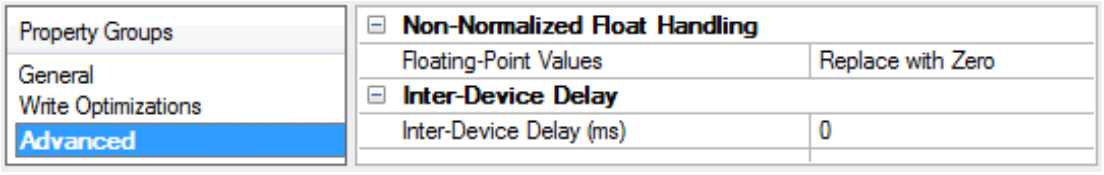
Optimization Method: Controls how write data is passed to the underlying communications driver. The options are:

- **Write All Values for All Tags**: This option forces the server to attempt to write every value to the controller. In this mode, the server continues to gather write requests and add them to the server's internal write queue. The server processes the write queue and attempts to empty it by writing data to the device as quickly as possible. This mode ensures that everything written from the client applications is sent to the target device. This mode should be selected if the write operation order or the write item's content must uniquely be seen at the target device.
- **Write Only Latest Value for Non-Boolean Tags**: Many consecutive writes to the same value can accumulate in the write queue due to the time required to actually send the data to the device. If the server updates a write value that has already been placed in the write queue, far fewer writes are needed to reach the same final output value. In this way, no extra writes accumulate in the server's queue. When the user stops moving the slide switch, the value in the device is at the correct value at virtually the same time. As the mode states, any value that is not a Boolean value is updated in the server's internal write queue and sent to the device at the next possible opportunity. This can greatly improve the application performance.
  Note: This option does not attempt to optimize writes to Boolean values. It allows users to optimize the operation of HMI data without causing problems with Boolean operations, such as a momentary push button.
- **Write Only Latest Value for All Tags**: This option takes the theory behind the second optimization mode and applies it to all tags. It is especially useful if the application only needs to send the latest value to the device. This mode optimizes all writes by updating the tags currently in the write queue before they are sent. This is the default mode.

Duty Cycle: is used to control the ratio of write to read operations. The ratio is always based on one read for every one to ten writes. The duty cycle is set to ten by default, meaning that ten writes occur for each read operation. Although the application is performing a large number of continuous writes, it must be ensured that read data is still given time to process. A setting of one results in one read operation for every write operation. If there are no write operations to perform, reads are processed continuously. This allows optimization for applications with continuous writes versus a more balanced back and forth data flow.
Note: It is recommended that the application be characterized for compatibility with the write optimization enhancements before being used in a production environment.

Channel Properties — Advanced

This group is used to specify advanced channel properties. Not all drivers support all properties; so the Advanced group does not appear for those devices.



Non-Normalized Float Handling: A non-normalized value is defined as Infinity, Not-a-Number (NaN), or as a Denormalized Number. The default is Replace with Zero. Drivers that have native float handling may default to Unmodified. Non-normalized float handling allows users to specify how a driver handles non-normalized IEEE-754 floating point data. Descriptions of the options are as follows:

- **Replace with Zero**: This option allows a driver to replace non-normalized IEEE-754 floating point values with zero before being transferred to clients.

- **Unmodified**: This option allows a driver to transfer IEEE-754 denormalized, normalized, non-number, and infinity values to clients without any conversion or changes.

Note: This property is not available if the driver does not support floating point values or if it only supports the option that is displayed. According to the channel's float normalization setting, only real-time driver tags (such as values and arrays) are subject to float normalization. For example, EFM data is not affected by this setting.

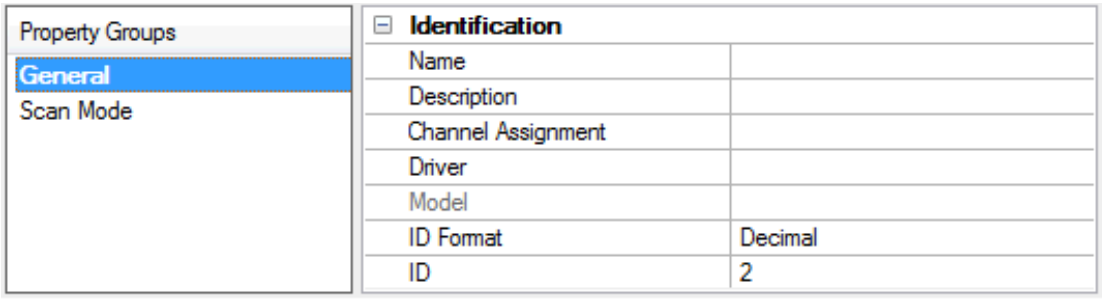
For more information on the floating point values, refer to "How To ... Work with Non-Normalized Floating Point Values" in the server help.

Inter-Device Delay: Specify the amount of time the communications channel waits to send new requests to the next device after data is received from the current device on the same channel. Zero (0) disables the delay.

Note: This property is not available for all drivers, models, and dependent settings.

Device Properties — General

A device represents a single target on a communications channel. If the driver supports multiple controllers, users must enter a device ID for each controller.



Identification

Name: This property specifies the name of the device. It is a logical user-defined name that can be up to 256 characters long, and may be used on multiple channels.

Note: Although descriptive names are generally a good idea, some OPC client applications may have a limited display window when browsing the OPC server's tag space. The device name and channel name become part of the browse tree information as well. Within an OPC client, the combination of channel name and device name would appear as "ChannelName.DeviceName".
For more information, refer to "How To... Properly Name a Channel, Device, Tag, and Tag Group" in server help.

Description: User-defined information about this device.
Many of these properties, including Description, have an associated system tag.

Channel Assignment: User-defined name of the channel to which this device currently belongs.

Driver: Selected protocol driver for this device.

Model: This property specifies the specific type of device that is associated with this ID. The contents of the drop-down menu depends on the type of communications driver being used. Models that are not supported

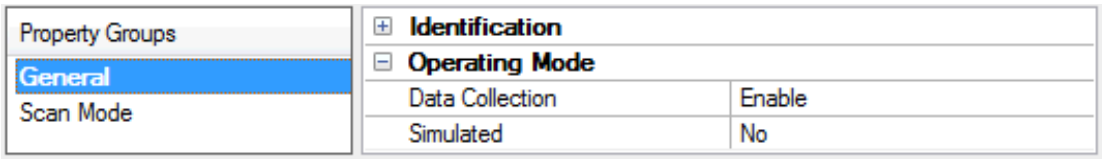
by a driver are disabled. If the communications driver supports multiple device models, the model selection can only be changed when there are no client applications connected to the device.

Note: If the communication driver supports multiple models, users should try to match the model selection to the physical device. If the device is not represented in the drop-down menu, select a model that conforms closest to the target device. Some drivers support a model selection called "Open," which allows users to communicate without knowing the specific details of the target device. For more information, refer to the driver help documentation.

ID: This property specifies the device's driver-specific station or node. The type of ID entered depends on the communications driver being used. For many communication drivers, the ID is a numeric value. Drivers that support a Numeric ID provide users with the option to enter a numeric value whose format can be changed to suit the needs of the application or the characteristics of the selected communications driver. The format is set by the driver by default. Options include Decimal, Octal, and Hexadecimal.

Note: If the driver is Ethernet-based or supports an unconventional station or node name, the device's TCP/IP address may be used as the device ID. TCP/IP addresses consist of four values that are separated by periods, with each value in the range of 0 to 255. Some device IDs are string based. There may be additional properties to configure within the ID field, depending on the driver. *For more information, refer to the driver's help documentation.*

Operating Mode



Data Collection: This property controls the device's active state. Although device communications are enabled by default, this property can be used to disable a physical device. Communications are not attempted when a device is disabled. From a client standpoint, the data is marked as invalid and write operations are not accepted. This property can be changed at any time through this property or the device system tags.

Simulated: This option places the device into Simulation Mode. In this mode, the driver does not attempt to communicate with the physical device, but the server continues to return valid OPC data. Simulated stops physical communications with the device, but allows OPC data to be returned to the OPC client as valid data. While in Simulation Mode, the server treats all device data as reflective: whatever is written to the simulated device is read back and each OPC item is treated individually. The item's memory map is based on the group Update Rate. The data is not saved if the server removes the item (such as when the server is reinitialized). The default is No.

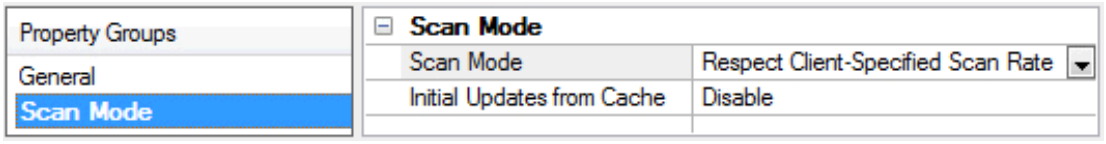
Notes:

1. This System tag (_Simulated) is read only and cannot be written to for runtime protection. The System tag allows this property to be monitored from the client.
2. In Simulation mode, the item's memory map is based on client update rate(s) (Group Update Rate for OPC clients or Scan Rate for native and DDE interfaces). This means that two clients that reference the same item with different update rates return different data.

Simulation Mode is for test and simulation purposes only. It should never be used in a production environment.

Device Properties — Scan Mode

The Scan Mode specifies the subscribed-client requested scan rate for tags that require device communications. Synchronous and asynchronous device reads and writes are processed as soon as possible; unaffected by the Scan Mode properties.



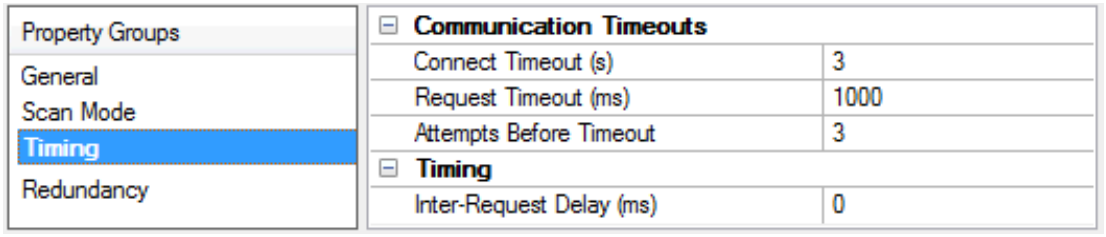
Scan Mode: Specifies how tags in the device are scanned for updates sent to subscribing clients. Descriptions of the options are:

- **Respect Client-Specified Scan Rate**: This mode uses the scan rate requested by the client.
- **Request Data No Faster than Scan Rate**: This mode specifies the value set as the maximum scan rate. The valid range is 10 to 99999990 milliseconds. The default is 1000 milliseconds.
  Note: When the server has an active client and items for the device and the scan rate value is increased, the changes take effect immediately. When the scan rate value is decreased, the changes do not take effect until all client applications have been disconnected.
- **Request All Data at Scan Rate**: This mode forces tags to be scanned at the specified rate for subscribed clients. The valid range is 10 to 99999990 milliseconds. The default is 1000 milliseconds.
- **Do Not Scan, Demand Poll Only**: This mode does not periodically poll tags that belong to the device nor perform a read to get an item's initial value once it becomes active. It is the client's responsibility to poll for updates, either by writing to the _DemandPoll tag or by issuing explicit device reads for individual items. *For more information, refer to "Device Demand Poll" in server help*.
- **Respect Tag-Specified Scan Rate**: This mode forces static tags to be scanned at the rate specified in their static configuration tag properties. Dynamic tags are scanned at the client-specified scan rate.

Initial Updates from Cache: When enabled, this option allows the server to provide the first updates for newly activated tag references from stored (cached) data. Cache updates can only be provided when the new item reference shares the same address, scan rate, data type, client access, and scaling properties. A device read is used for the initial update for the first client reference only. The default is disabled; any time a client activates a tag reference the server attempts to read the initial value from the device.

Device Properties — Timing

The device Timing properties allow the driver's response to error conditions to be tailored to fit the application's needs. In many cases, the environment requires changes to these properties for optimum performance. Factors such as electrically generated noise, modem delays, and poor physical connections can influence how many errors or timeouts a communications driver encounters. Timing properties are specific to each configured device.



Communications Timeouts

Connect Timeout: This property (which is used primarily by Ethernet based drivers) controls the amount of time required to establish a socket connection to a remote device. The device's connection time often takes longer than normal communications requests to that same device. The valid range is 1 to 30 seconds. The default is typically 3 seconds, but can vary depending on the driver's specific nature. If this setting is not supported by the driver, it is disabled.

Note: Due to the nature of UDP connections, the connection timeout setting is not applicable when communicating via UDP.

Request Timeout: This property specifies an interval used by all drivers to determine how long the driver waits for a response from the target device to complete. The valid range is 50 to 9,999,999 milliseconds (167.6667 minutes). The default is usually 1000 milliseconds, but can vary depending on the driver. The default timeout for most serial drivers is based on a baud rate of 9600 baud or better. When using a driver at lower baud rates, increase the timeout to compensate for the increased time required to acquire data.

Attempts Before Timeout: This property specifies how many times the driver issues a communications request before considering the request to have failed and the device to be in error. The valid range is 1 to 10. The default is typically 3, but can vary depending on the driver's specific nature. The number of attempts configured for an application depends largely on the communications environment. This property applies to both connection attempts and request attempts.

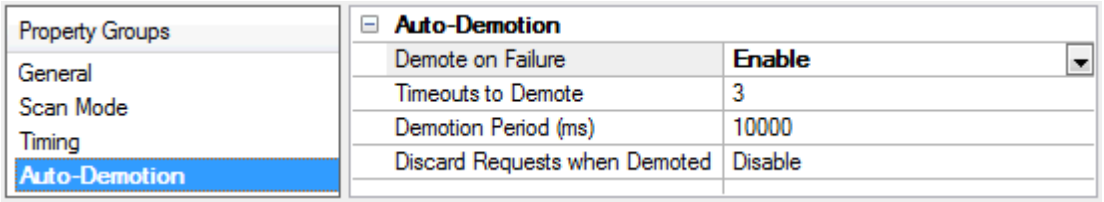
Timing

Inter-Request Delay: This property specifies how long the driver waits before sending the next request to the target device. It overrides the normal polling frequency of tags associated with the device, as well as one-time reads and writes. This delay can be useful when dealing with devices with slow turnaround times and in cases where network load is a concern. Configuring a delay for a device affects communications with all other devices on the channel. It is recommended that users separate any device that requires an inter-request delay to a separate channel if possible. Other communications properties (such as communication serialization) can extend this delay. The valid range is 0 to 300,000 milliseconds; however, some drivers may limit the maximum value due to a function of their particular design. The default is 0, which indicates no delay between requests with the target device.

Note: Not all drivers support Inter-Request Delay. This setting does not appear if it is not available.

Device Properties — Auto-Demotion

The Auto-Demotion properties can temporarily place a device off-scan in the event that a device is not responding. By placing a non-responsive device offline for a specific time period, the driver can continue to optimize its communications with other devices on the same channel. After the time period has been reached, the driver re-attempts to communicate with the non-responsive device. If the device is responsive, the device is placed on-scan; otherwise, it restarts its off-scan time period.



Demote on Failure: When enabled, the device is automatically taken off-scan until it is responding again.

Tip: Determine when a device is off-scan by monitoring its demoted state using the _AutoDemoted system tag.

Timeouts to Demote: Specify how many successive cycles of request timeouts and retries occur before the device is placed off-scan. The valid range is 1 to 30 successive failures. The default is 3.

Demotion Period: Indicate how long the device should be placed off-scan when the timeouts value is reached. During this period, no read requests are sent to the device and all data associated with the read requests are set to bad quality. When this period expires, the driver places the device on-scan and allows for another attempt at communications. The valid range is 100 to 3600000 milliseconds. The default is 10000 milliseconds.

Discard Requests when Demoted: Select whether or not write requests should be attempted during the off-scan period. Disable to always send write requests regardless of the demotion period. Enable to discard writes; the server automatically fails any write request received from a client and does not post a message to the Event Log.

Device Properties — Tag Generation

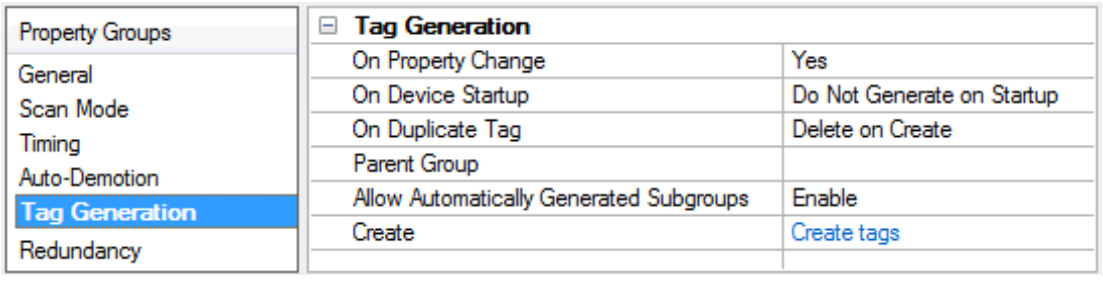
The automatic tag database generation features make setting up an application a plug-and-play operation. Select communications drivers can be configured to automatically build a list of tags that correspond to device-specific data. These automatically generated tags (which depend on the nature of the supporting driver) can be browsed from the clients.

Not all devices and drivers support full automatic tag database generation and not all support the same data types. Consult the data types descriptions or the supported data type lists for each driver for specifics.

If the target device supports its own local tag database, the driver reads the device's tag information and uses the data to generate tags within the server. If the device does not natively support named tags, the driver creates a list of tags based on driver-specific information. An example of these two conditions is as follows:

1. If a data acquisition system supports its own local tag database, the communications driver uses the tag names found in the device to build the server's tags.
2. If an Ethernet I/O system supports detection of its own available I/O module types, the communications driver automatically generates tags in the server that are based on the types of I/O modules plugged into the Ethernet I/O rack.

Note: Automatic tag database generation's mode of operation is completely configurable. *For more information, refer to the property descriptions below.*



On Property Change: If the device supports automatic tag generation when certain properties change, the **On Property Change** option is shown. It is set to **Yes** by default, but it can be set to **No** to control over when tag generation is performed. In this case, the **Create tags** action must be manually invoked to perform tag generation.

On Device Startup: This property specifies when OPC tags are automatically generated. Descriptions of the options are as follows:

- **Do Not Generate on Startup**: This option prevents the driver from adding any OPC tags to the tag space of the server. This is the default setting.
- **Always Generate on Startup**: This option causes the driver to evaluate the device for tag information. It also adds tags to the tag space of the server every time the server is launched.
- **Generate on First Startup**: This option causes the driver to evaluate the target device for tag information the first time the project is run. It also adds any OPC tags to the server tag space as needed.

Note: When the option to automatically generate OPC tags is selected, any tags that are added to the server's tag space must be saved with the project. Users can configure the project to automatically save from the **Tools** | **Options** menu.

On Duplicate Tag: When automatic tag database generation is enabled, the server needs to know what to do with the tags that it may have previously added or with tags that have been added or modified after the communications driver since their original creation. This setting controls how the server handles OPC tags that were automatically generated and currently exist in the project. It also prevents automatically generated tags from accumulating in the server.

For example, if a user changes the I/O modules in the rack with the server configured to **Always Generate on Startup**, new tags would be added to the server every time the communications driver detected a new I/O module. If the old tags were not removed, many unused tags could accumulate in the server's tag space. The options are:

- **Delete on Create**: This option deletes any tags that were previously added to the tag space before any new tags are added. This is the default setting.
- **Overwrite as Necessary**: This option instructs the server to only remove the tags that the communications driver is replacing with new tags. Any tags that are not being overwritten remain in the server's tag space.
- **Do not Overwrite**: This option prevents the server from removing any tags that were previously generated or already existed in the server. The communications driver can only add tags that are completely new.
- **Do not Overwrite, Log Error**: This option has the same effect as the prior option, and also posts an error message to the server's Event Log when a tag overwrite would have occurred.

Note: Removing OPC tags affects tags that have been automatically generated by the communications driver as well as any tags that have been added using names that match generated tags. Users should avoid adding tags to the server using names that may match tags that are automatically generated by the driver.

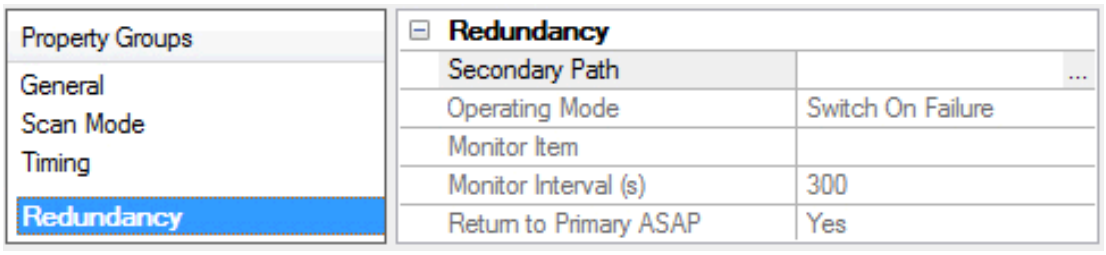
Parent Group: This property keeps automatically generated tags from mixing with tags that have been entered manually by specifying a group to be used for automatically generated tags. The name of the group can be up to 256 characters. This parent group provides a root branch to which all automatically generated tags are added.

Allow Automatically Generated Subgroups: This property controls whether the server automatically creates subgroups for the automatically generated tags. This is the default setting. If disabled, the server generates the device's tags in a flat list without any grouping. In the server project, the resulting tags are named with the address value. For example, the tag names are not retained during the generation process.

Note: If, as the server is generating tags, a tag is assigned the same name as an existing tag, the system automatically increments to the next highest number so that the tag name is not duplicated. For example, if the generation process creates a tag named "AI22" that already exists, it creates the tag as "AI23" instead.

Create: Initiates the creation of automatically generated OPC tags. If the device's configuration has been modified, **Create tags** forces the driver to reevaluate the device for possible tag changes. Its ability to be accessed from the System tags allows a client application to initiate tag database creation.
Note: **Create tags** is disabled if the Configuration edits a project offline.

Device Properties — Redundancy

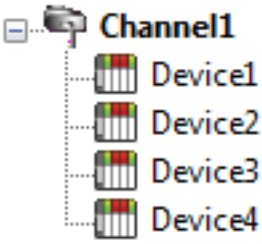


Redundancy is available with the Media-Level Redundancy Plug-In.
Consult the website, a sales representative, or the user manual for more information.

Optimizing Communications

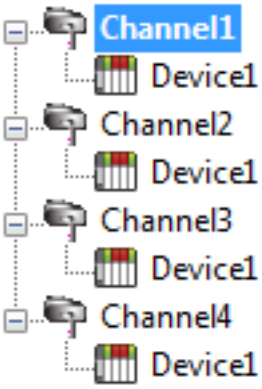
The Optimation OptiLogic Driver has been designed to provide the best performance with the least amount of impact on the system's overall performance. While the Optimation OptiLogic Driver is fast, there are a couple of guidelines that can be used in order to control and optimize the application and gain maximum performance.

Our server refers to communications protocols like Optimation OptiLogic as a channel. Each channel defined in the application represents a separate path of execution in the server. Once a channel has been defined, a series of devices must then be defined under that channel. Each of these devices represents a single OptiLogic RTU from which data will be collected. While this approach to defining the application will provide a high level of performance, it won't take full advantage of the Optimation OptiLogic Driver or the network. An example of how the application may appear when configured using a single channel is shown below.



Each device appears under a single channel. In this configuration, the driver must move from one device to the next as quickly as possible in order to gather information at an effective rate. As more devices are added or more information is requested from a single device, the overall update rate begins to suffer.

If the Optimation OptiLogic Driver could only define one single channel, then the example shown above would be the only option available; however, the Optimation OptiLogic Driver can define up to 100 channels. Using multiple channels distributes the data collection workload by simultaneously issuing multiple requests to the network. An example of how the same application may appear when configured using multiple channels to improve performance is shown below.



Each device has now been defined under its own channel. In this new configuration, a single path of execution is dedicated to the task of gathering data from each device. If the application has 100 or fewer channels, it can be optimized exactly how it is shown here.

The performance will improve even if the application has more channels. While fewer channels may be ideal, the application will still benefit from additional channels. Although spreading the device load across all channels will cause the server to move from device to device again, it can now do so with far less devices to process on a single channel.

Data Types Description

Data Type	Description
Boolean	Single bit
Byte	Unsigned 8-bit value bit 0 is the low bit bit 7 is the high bit
Char	Signed 8-bit value bit 0 is the low bit bit 6 is the high bit bit 7 is the sign bit
Word	Unsigned 16-bit value bit 0 is the low bit bit 15 is the high bit
Short	Signed 16-bit value bit 0 is the low bit bit 14 is the high bit bit 15 is the sign bit
DWord	Unsigned 32-bit value bit 0 is the low bit bit 31 is the high bit
Long	Signed 32-bit value bit 0 is the low bit bit 30 is the high bit bit 31 is the sign bit
Float	32-bit floating point value
Double	64-bit floating point value
String	Null-terminated character array

Address Descriptions

Address specifications vary depending on the type of module in use. Select a link from the following list to obtain specific address information for the module type of interest.

Select By SubType

OL2104
OL2205
OL2304
OL3420
OL2108
OL2208
OL2408
OL3440
OL2109
OL2211
OL2418
OL3850
OL2111
OL2252
OL2602
OL4054
OL2201
OL2258
OL3406

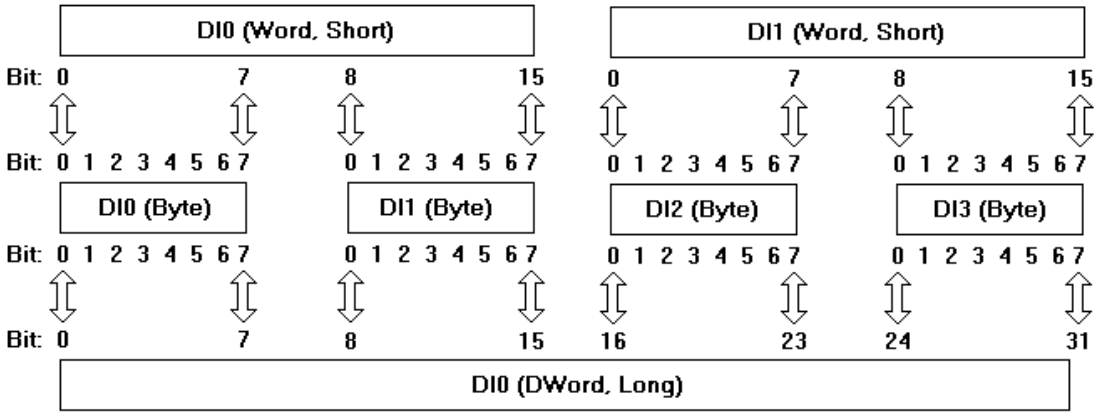
Select By Classification

4 Digital Input
2-Channel Analog Input
Base RS232 Port
8 Digital Input
4-Channel Analog Input
Dual RS232 Port
16 Digital Input
8 Channel Analog Input
OL3406 Operator Panel
24 Digital Input
16-Channel Analog Input
OL3420 Operator Panel
32 Digital Input
2-Channel Analog Output
OL3440 Operator Panel
4 Digital Output
4-Channel Analog Output
OL3850 Operator Panel

This device requires a client connection: tag addresses cannot be validated completely until successful communication with the device is made. Since the driver is unaware of the device configuration until this communication is made, the tag addressing parser can only notify the user of improper addresses within the driver.

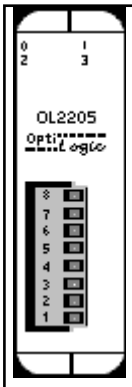
Digital I/O Bit Mapping

Points (bits) can be referenced using the optional Byte, Word, Short, DWord and Long data types. The following diagram illustrates how the driver maps bits within these data types.



All digital inputs and outputs can be addressed as Byte, Word, Short, DWord or Long data types. When addressing a module (such as an 8 bit input module) as a Word data type, the upper 8 bits will be fixed at zero while the lower bits will contain the status of the 8 inputs.

4 Digital Input Module



Digital input modules are used to monitor the state of discrete field devices. Typical digital inputs are connected to switches, buttons, digital outputs from other equipment, discrete level sensors, thermostats and other On/Off sensing devices.

Digital status is sensed by a controller such as an OptiLogic system, by passing current through an input sensor. When the current is on, the input state is active. When there is no current, the state is inactive.

Subtypes
OL2205

4 Digital Input Addressing Specifications

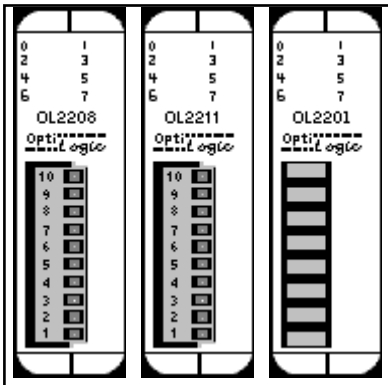
Syntax	Data Type	Range	Access
S<slot>:DI<point>	**Boolean**	0-3	Read Only
S<slot>:DI<offset>*	Byte	0	Read Only
S<slot>:DI<offset>*	Word, Short	0	Read Only
S<slot>:DI<offset>*	DWord, Long	0	Read Only

* Multiple points can be referenced through the use of this optional syntax. For more information, refer to **Digital I/O Bit Mapping**.

Examples

Address	Value	Description
S1:DI0	1	Slot 1, point 0 is on.
S1:DI0 (as Byte)	15	Slot 1, byte 0 (points 0-3 are on).
S1:DI0 (as Word)	0	Slot 1, word 0 (points 0-3 are off).
S1:DI0 (as DWORD)	15	Slot 1, DWord 0 (points 0-3 are on).

8 Digital Input Module



Digital input modules are used to monitor the state of discrete field devices. Typical digital inputs are connected to switches, buttons, digital outputs from other equipment, discrete level sensors, thermostats and other On/Off sensing devices.

Digital status is sensed by a controller such as an OptiLogic system, by passing current through an input sensor. When the current is on, the input state is active. When there is no current, the state is inactive.

Subtypes
OL2208, OL2211, OL2201

8 Digital Input Addressing Specifications

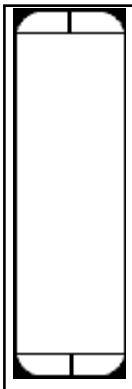
Syntax	Data Type	Range	Access
S<slot>:DI<point>	**Boolean**	0-7	Read Only
S<slot>:DI<offset>*	Byte	0	Read Only
S<slot>:DI<offset>*	Word, Short	0	Read Only
S<slot>:DI<offset>*	DWord, Long	0	Read Only

* Multiple points can be referenced through the use of this optional syntax. For more information, refer to **Digital I/O Bit Mapping**.

Examples

Address	Value	Description
S1:DI0	1	Slot 1, point 0 is on.
S1:DI0 (as Byte)	255	Slot 1, byte 0 (points 0-7 are on).
S1:DI0 (as Word)	0	Slot 1, word 0 (points 0-7 are off).
S1:DI0 (as DWORD)	255	Slot 1, DWord 0 (points 0-7 are on).

16 Digital Input Module



Digital input modules are used to monitor the state of discrete field devices. Typical digital inputs are connected to switches, buttons, digital outputs from other equipment, discrete level sensors, thermostats and other On/Off sensing devices.

Digital status is sensed by a controller such as an OptiLogic system, by passing current through an input sensor. When the current is on, the input state is active. When there is no current, the state is inactive.

Subtypes

N/A

16 Digital Input Addressing Specifications

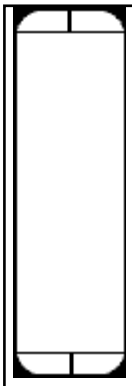
Syntax	Data Type	Range	Access
S<slot>:DI<point>	**Boolean**	0-7	Read Only
S<slot>:DI<offset>*	Byte	0-1	Read Only
S<slot>:DI<offset>*	Word, Short	0	Read Only
S<slot>:DI<offset>*	DWord, Long	0	Read Only

* Multiple points can be referenced through the use of this optional syntax. For more information, refer to **Digital I/O Bit Mapping**.

Examples

Address	Value	Description
S1:DI0	1	Slot 1, point 0 is on.
S1:DI1 (as Byte)	255	Slot 1, byte 1 (points 8-15 are on).
S1:DI0 (as Word)	0	Slot 1, word 0 (points 0-15 are off).
S1:DI0 (as DWORD)	65535	Slot 1, DWord 0 (points 0-15 are on).

24 Digital Input Module



Digital input modules are used to monitor the state of discrete field devices. Typical digital inputs are connected to switches, buttons, digital outputs from other equipment, discrete level sensors, thermostats and other On/Off sensing devices.

Digital status is sensed by a controller such as an OptiLogic system, by passing current through an input sensor. When the current is on, the input state is active. When there is no current, the state is inactive.

Subtypes

N/A

24 Digital Input Addressing Specifications

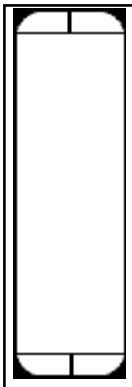
Syntax	Data Type	Range	Access
S<slot>:DI<point>	**Boolean**	0-7	Read Only
S<slot>:DI<offset>*	Byte	0-2	Read Only
S<slot>:DI<offset>*	Word, Short	0-1	Read Only
S<slot>:DI<offset>*	DWord, Long	0	Read Only

* Multiple points can be referenced through the use of this optional syntax. For more information, refer to **Digital I/O Bit Mapping**.

Examples

Address	Value	Description
S1:DI0	1	Slot 1, point 0 is on.
S1:DI2 (as Byte)	255	Slot 1, byte 2 (points 16-23 are on).
S1:DI0 (as Word)	0	Slot 1, word 0 (points 0-15 are off).
S1:DI0 (as DWORD)	16777216	Slot 1, DWord 0 (points 0-23 are on).

32 Digital Input Module



Digital input modules are used to monitor the state of discrete field devices. Typical digital inputs are connected to switches, buttons, digital outputs from other equipment, discrete level sensors, thermostats and other On/Off sensing devices.

Digital status is sensed by a controller such as an OptiLogic system, by passing current through an input sensor. When the currnet is on, the input state is active. When there is no current, the state is inactive.

Subtypes

N/A

32 Digital Input Addressing Specifications

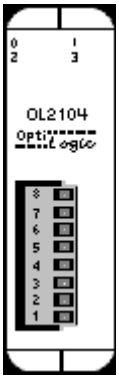
Syntax	Data Type	Range	Access
S<slot>:DI<point>	**Boolean**	0-7	Read Only
S<slot>:DI<offset>*	Byte	0-3	Read Only
S<slot>:DI<offset>*	Word, Short	0-1	Read Only
S<slot>:DI<offset>*	DWord, Long	0	Read Only

* Multiple points can be referenced through the use of this optional syntax. For more information, refer to **Digital I/O Bit Mapping**.

Examples

Syntax	Data Type	Range	Access
S<slot>:DI<point>	**Boolean**	0-7	Read Only
S<slot>:DI<offset>*	Byte	0-3	Read Only
S<slot>:DI<offset>*	Word, Short	0-1	Read Only
S<slot>:DI<offset>*	DWord, Long	0	Read Only

4 Digital Output Module



Digital outputs are used to turn loads on and off. Loads may be lights, motors, solenoids or any type of On/Off device.

Digital outputs in the OptiLogic series come in three types: relay, transistor and solid state relay. Each type has applications it is best suited for. For more information, refer to the module manual.

If there is a loss of communication with the host, the output module will enter a fail safe state. There are three types of states: fail safe to all outputs off, fail safe to a pattern and fail safe to last state. The time in which a fail safe state is entered after loss of communication can be delayed up to 25.5 seconds. If certain points need to be treated differently in fail safe mode, a pattern can be used to signify which points are turned on and which ones are turned off in fail safe mode.

Subtypes
OL2104

Address Types
4 Digital Output
Fail Safe Type
Fail Safe Pattern
Fail Safe Time
Fail Safe Set

4 Digital Output Addressing Specifications

Syntax	Data Type	Range	Access
S<slot>:DO<point>	**Boolean**	0-3	Read/Write
S<slot>:DO<offset>*	Byte**	0	Read/Write
S<slot>:DO<offset>*	Word, Short**	0	Read/Write
S<slot>:DO<offset>*	DWord, Long**	0	Read/Write

* Multiple points can be referenced through the use of this optional syntax. For more information, refer to Digital I/O Bit Mapping.
** Since only the lower nibble of the least significant byte is being used, any value entered above 15 will be cropped.

Examples

Address	Value	Description
S1:DO0	1	Slot 1, point 0 (turn point 0 on).
S1:DO0 (as Byte)	15	Slot 1, byte 0 (turn points 0-3 on).
S1:DO0 (as Word)	0	Slot 1, word 0 (turn all points off).
S1:DO0 (as DWORD)	15	Slot 1, DWord 0 (turn points 0-3 on).

Fail Safe Type Addressing
There are three types of fail safe states: fail safe to all outputs off, fail safe to a pattern and fail safe to last state. These types may be specified by referencing FS.TYPE.

Values
1 = Fail safe to all outputs off
2 = Fail safe to the pattern contained in FS<point>.PATTERN
3 = Fail safe to last state

Requirements
None

Important: The value specified in FS.TYPE is not sent to the device until FS.SET is set.

Specifications

Syntax	Data Type	Range	Access
S<slot>:FS.TYPE	**Byte**, Word, Short, DWord, Long	N/A	Write Only

Fail Safe Pattern Addressing

Using a fail safe pattern, users can specify which points are set and which ones are cleared when in fail safe mode.

Values

True = turn point on
False = turn point off

Requirements

FS.TYPE must be set to 2 for device to fail safe to a pattern.

Note: If using Byte, Word, Short, DWord or Long data types, the bit pattern determines which points are turned on and which ones are turned off.

Important: The pattern specified in FS.PATTERN is not sent to the device until FS.SET is set.

Specifications

Syntax	Data Type	Range	Access
S<slot>FS<point>.PATTERN	**Boolean**	0-3	Write Only
S<slot>:FS<offset>.PATTERN*	Byte**	0	Write Only
S<slot>:FS<offset>.PATTERN*	Word, Short**	0	Write Only
S<slot>:FS<offset>.PATTERN*	DWord, Long**	0	Write Only

* Multiple points can be referenced through the use of this optional syntax. For more information, refer to **Digital I/O Bit Mapping**.
** Since only the lower nibble of the least significant byte is being used, any value entered above 15 will be cropped.

Examples

Address	Value	Description
S1:FS0.PATTERN	1	Slot 1, turn point 0 on in fail safe mode.*
S1:FS0.PATTERN (as Byte)	15	Slot 1, turn points 0-3 on in fail safe mode.*
S1:FS0.PATTERN (as Word)	0	Slot 1, turn points 0-3 off in fail safe mode.*
S1:FS0.PATTERN (as DWORD)	15	Slot 1, turn points 0-3 on in fail safe mode.*

* See Notes and Requirements above.

Fail Safe Time Addressing

After communication between the host and the RTU has been lost, a time delay may be introduced before fail safe mode is engaged for the output module. This time delay can be entered by referencing FS.TIME and is specified in tenths of seconds.

Requirements

None

Important: The value specified in FS.TYPE is not sent to the device until FS.SET is set.

Specifications

Syntax	Data Type	Range	Access
S<slot>:FS.TIME	**Byte**, Word, Short, DWord, Long	N/A	Write Only

Examples

Address	Value	Description
S1:FS.TIME	255	Slot 1, fail safe time delay = 25.5 seconds.
S1:FS.TIME	0	Slot 1, no delay.
S1:FS.TIME	1	Slot 1, fail safe time delay = .1 seconds.

Fail Safe Set Addressing

For any of the fail safe configuration properties (type, time and pattern) to be sent to the device, FS.SET must be set. Immediately after the properties are sent, FS.SET will be reset.

Values

True = Send fail safe configurations to device
False = No action

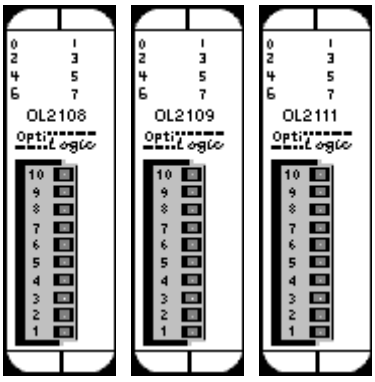
Requirements

None

Specifications

Syntax	Data Type	Range	Access
S<slot>:FS.SET	**Boolean**	N/A	Write Only

8 Digital Output Module



Digital outputs are used to turn loads on and off. Loads may be lights, motors, solenoids or any type of On/Off device.

Digital outputs in the OptiLogic series come in three types: relay, transistor and solid state relay. Each type has applications it is best suited for. For more information, refer to the module manual.

If there is a loss of communication with the host, the output module will enter a fail safe state. There are three types of states: fail safe to all outputs off, fail safe to a pattern and fail safe to last state. The time in which a fail safe state is entered after loss of communication can be delayed up to 25.5 seconds. If certain points need to be treated differently in fail safe mode, a pattern can be used to signify which points are turned on and which ones are turned off in fail safe mode.

Subtypes

OL2108, OL2109, OL2111

Address Types

8 Digital Output
Fail Safe Type
Fail Safe Pattern
Fail Safe Time
Fail Safe Set

8 Digital Output Addressing Specifications

Syntax	Data Type	Range	Access
S<slot>:DO<point>	**Boolean**	0-7	Read/Write
S<slot>:DO<offset>*	Byte	0	Read/Write
S<slot>:DO<offset>*	Word, Short	0	Read/Write
S<slot>:DO<offset>*	DWord, Long	0	Read/Write

* Multiple points can be referenced through the use of this optional syntax. For more information, refer to **Digital I/O Bit Mapping**.

Examples

Address	Value	Description
S1:DO0	1	Slot 1, point 0 (turn point 0 on).
S1:DO0 (as Byte)	255	Slot 1, byte 0 (turn points 0-7 on).
S1:DO0 (as Word)	0	Slot 1, word 0 (turn points 0-7 off).
S1:DO0 (as DWORD)	255	Slot 1, DWord 0 (turn points 0-7 on).

Fail Safe Type Addressing

There are three types of fail safe states: fail safe to all outputs off, fail safe to a pattern and fail safe to last state. These types may be specified by referencing FS.TYPE.

Values

1 = Fail safe to all outputs off
2 = Fail safe to the pattern contained in FS<point>.PATTERN
3 = Fail safe to last state

Requirements

None

Important: The value specified in FS.TYPE is not sent to the device until FS.SET is set.

Specifications

Syntax	Data Type	Range	Access
S<slot>:FS.TYPE	**Byte**, Word, Short, DWord, Long	N/A	Write Only

Fail Safe Pattern Addressing

Using a fail safe pattern, users can specify which points are set and which ones are cleared when in fail safe mode.

Values
True = turn point on
False = turn point off

Requirements
FS.TYPE must be set to 2 for device to fail safe to a pattern.

Note: If using Byte, Word, Short, DWord or Long data types, the bit pattern determines which points are turned on and which ones are turned off.

Important: The pattern specified in FS.PATTERN is not sent to the device until FS.SET is set.

Specifications

Syntax	Data Type	Range	Access
S<slot>FS<point>.PATTERN	**Boolean**	0-7	Write Only
S<slot>:FS<offset>.PATTERN*	Byte	0	Write Only
S<slot>:FS<offset>.PATTERN*	Word, Short	0	Write Only
S<slot>:FS<offset>.PATTERN*	DWord, Long	0	Write Only

* Multiple points can be referenced through the use of this optional syntax. For more information, refer to **Digital I/O Bit Mapping**.

Examples

Address	Value	Description
S1:FS0.PATTERN	1	Slot 1, turn point 0 on in fail safe mode.*
S1:FS0.PATTERN (as Byte)	255	Slot 1, turn points 0-7 on in fail safe mode.*
S1:FS0.PATTERN (as Word)	0	Slot 1, turn point 0-7 off in fail safe mode.*
S1:FS0.PATTERN (as DWORD)	255	Slot 1, turn points 0-7 on in fail safe mode.*

* See Notes and Requirements above.

Fail Safe Time Addressing
After communication between the host and the RTU has been lost, a time delay may be introduced before fail safe mode is engaged for the output module. This time delay can be entered by referencing FS.TIME and is specified in tenths of seconds.

Requirements
None

Important: The value specified in FS.TYPE is not sent to the device until FS.SET is set.

Specifications

Syntax	Data Type	Range	Access
S<slot>:FS.TIME	**Byte**, Word, Short, DWord, Long	N/A	Write Only

Examples

Address	Value	Description
S1:FS.TIME	255	Slot 1, fail safe time delay = 25.5 seconds.
S1:FS.TIME	0	Slot 1, no delay.
S1:FS.TIME	1	Slot 1, fail safe time delay = .1 seconds.

Fail Safe Set Addressing

For any of the fail safe configuration properties (type, time and pattern) to be sent to the device, FS.SET must be set. Immediately after the properties are sent, FS.SET will be reset.

Values

True = Send fail safe configurations to device
False = No action

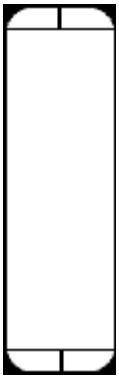
Requirements

None

Specifications

Syntax	Data Type	Range	Access
S<slot>:FS.SET	**Boolean**	N/A	Write Only

16 Digital Output Module



Digital outputs are used to turn loads on and off. Loads may be lights, motors, solenoids or any type of On/Off device.

Digital outputs in the OptiLogic series come in three types: relay, transistor and solid state relay. Each type has applications it is best suited for. For more information, refer to the module manual.

If there is a loss of communication with the host, the output module will enter a fail safe state. There are three types of states: fail safe to all outputs off, fail safe to a pattern and fail safe to last state. The time in which a fail safe state is entered after loss of communication can be delayed up to 25.5 seconds. If certain points need to be treated differently in fail safe mode, a pattern can be used to signify which points are turned on and which ones are turned off in fail safe mode.

Subtypes

N/A

Address Types

16 Digital Output
Fail Safe Type
Fail Safe Pattern
Fail Safe Time

Fail Safe Set

16 Digital Output Addressing Specifications

Syntax	Data Type	Range	Access
S<slot>:DO<point>	**Boolean**	0-7	Read/Write
S<slot>:DO<offset>*	Byte	0-1	Read/Write
S<slot>:DO<offset>*	Word, Short	0	Read/Write
S<slot>:DO<offset>*	DWord, Long	0	Read/Write

* Multiple points can be referenced through the use of this optional syntax. For more information, refer to Digital I/O Bit Mapping.

Examples

Address	Value	Description
S1:DO0	1	Slot 1, point 0 (turn point 0 on).
S1:DO1 (as Byte)	255	Slot 1, byte 1 (turn points 8-15 on).
S1:DO0 (as Word)	0	Slot 1, word 0 (turn points 0-15 off).
S1:DO0 (as DWORD)	65535	Slot 1, DWord 0 (turn points 0-15 on).

Fail Safe Type Addressing

There are three types of fail safe states: fail safe to all outputs off, fail safe to a pattern and fail safe to last state. These types may be specified by referencing FS.TYPE.

Values

1 = Fail safe to all outputs off
2 = Fail safe to the pattern contained in FS<point>.PATTERN
3 = Fail safe to last state

Requirements

None

Important: The value specified in FS.TYPE is not sent to the device until FS.SET is set.

Specifications

Syntax	Data Type	Range	Access
S<slot>:FS.TYPE	**Byte**, Word, Short, DWord, Long*	N/A	Write Only

* The default data type is shown in **bold**.

Fail Safe Pattern Addressing

Using a fail safe pattern, users can specify which points are set and which ones are cleared when in fail safe mode.

Values

True = turn point on
False = turn point off

Requirements

FS.TYPE must be set to 2 for device to fail safe to a pattern.

Note: If using Byte, Word, Short, DWord or Long data types, the bit pattern determines which points are turned on and which ones are turned off.

Important: The pattern specified in FS.PATTERN is not sent to the device until FS.SET is set.

Specifications

Syntax	Data Type	Range	Access
S<slot>FS<point>.PATTERN	**Boolean**	0-7	Write Only
S<slot>:FS<offset>.PATTERN*	Byte	0-1	Write Only
S<slot>:FS<offset>.PATTERN*	Word, Short	0	Write Only
S<slot>:FS<offset>.PATTERN*	DWord, Long	0	Write Only

* Multiple points can be referenced through the use of this optional syntax. For more information, refer to **Digital I/O Bit Mapping**.

Examples

Address	Value	Description
S1:FS0.PATTERN	1	Slot 1, turn point 0 on in fail safe mode.*
S1:FS1.PATTERN (as Byte)	255	Slot 1, turn points 8-15 on in fail safe mode.*
S1:FS0.PATTERN (as Word)	0	Slot 1, turn points 0-15on in fail safe mode.*
S1:FS0.PATTERN (as DWORD)	65535	Slot 1, turn points 0-15 on in fail safe mode.*

* See Notes and Requirements above.

Fail Safe Time Addressing

After communication between the host and the RTU has been lost, a time delay may be introduced before fail safe mode is engaged for the output module. This time delay can be entered by referencing FS.TIME and is specified in tenths of seconds.

Requirements

None

Important: The value specified in FS.TYPE is not sent to the device until FS.SET is set.

Specifications

Syntax	Data Type	Range	Access
S<slot>:FS.TIME	**Byte**, Word, Short, DWord, Long	N/A	Write Only

Examples

Address	Value	Description
S1:FS.TIME	255	Slot 1, fail safe time delay = 25.5 seconds.
S1:FS.TIME	0	Slot 1, no delay.

S1:FS.TIME	1	Slot 1, fail safe time delay = .1 seconds.

Fail Safe Set Addressing

For any of the fail safe configuration properties (type, time and pattern) to be sent to the device, FS.SET must be set. Immediately after the properties are sent, FS.SET will be reset.

Values

True = Send fail safe configurations to device
False = No action

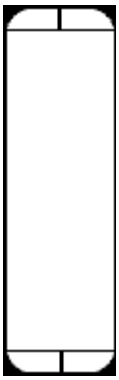
Requirements

None

Specifications

Syntax	Data Type	Range	Access
S<slot>:FS.SET	**Boolean**	N/A	Write Only

24 Digital Output Module



Digital outputs are used to turn loads on and off. Loads may be lights, motors, solenoids or any type of On/Off device.

Digital outputs in the OptiLogic series come in three types: relay, transistor and solid state relay. Each type has applications it is best suited for. For more information, refer to the module manual.

If there is a loss of communication with the host, the output module will enter a fail safe state. There are three types of states: fail safe to all outputs off, fail safe to a pattern and fail safe to last state. The time in which a fail safe state is entered after loss of communication can be delayed up to 25.5 seconds. If certain points need to be treated differently in fail safe mode, a pattern can be used to signify which points are turned on and which ones are turned off in fail safe mode.

Subtypes

N/A

Address Types

24 Digital Output
Fail Safe Type
Fail Safe Pattern
Fail Safe Time
Fail Safe Set

24 Digital Output Addressing Specifications

Syntax	Data Type	Range	Access
S<slot>:DO<point>	**Boolean**	0-7	Read/Write

Syntax	Data Type	Range	Access
S<slot>:DO<offset>*	Byte	0-2	Read/Write
S<slot>:DO<offset>*	Word, Short	0-1	Read/Write
S<slot>:DO<offset>*	DWord, Long	0	Read/Write

* Multiple points can be referenced through the use of this optional syntax. For more information, refer to **Digital I/O Bit Mapping**.

Examples

Address	Value	Description
S1:DO0	1	Slot 1, point 0 (turn point 0 on).
S1:DO2 (as Byte)	255	Slot 1, byte 2 (turn points 16-23 on).
S1:DO1 (as Word)	1	Slot 1, word 1 (turn point 16 on).
S1:DO0 (as DWORD)	16777216	Slot 1, DWord 0 (turn points 0-23 on).

Fail Safe Type Addressing

There are three types of fail safe states: fail safe to all outputs off, fail safe to a pattern and fail safe to last state. These types may be specified by referencing FS.TYPE.

Values

1 = Fail safe to all outputs off
2 = Fail safe to the pattern contained in FS<point>.PATTERN
3 = Fail safe to last state

Requirements

None

Important: The value specified in FS.TYPE is not sent to the device until FS.SET is set.

Specifications

Syntax	Data Type	Range	Access
S<slot>:FS.TYPE	**Byte**, Word, Short, DWord, Long	N/A	Write Only

Fail Safe Pattern Addressing

Using a fail safe pattern, users can specify which points are set and which ones are cleared when in fail safe mode.

Values

True = turn point on
False = turn point off

Requirements

FS.TYPE must be set to 2 for device to fail safe to a pattern.

Note: If using Byte, Word, Short, DWord or Long data types, the bit pattern determines which points are turned on and which ones are turned off.

Important: The pattern specified in FS.PATTERN is not sent to the device until FS.SET is set.

Specifications

Syntax	Data Type	Range	Access
S<slot>FS<point>.PATTERN	**Boolean**	0-7	Write Only
S<slot>:FS<offset>.PATTERN*	Byte	0-2	Write Only
S<slot>:FS<offset>.PATTERN*	Word, Short	0-1	Write Only
S<slot>:FS<offset>.PATTERN*	DWord, Long	0	Write Only

* Multiple points can be referenced through the use of this optional syntax. For more information, refer to **Digital I/O Bit Mapping**

Examples

Address	Value	Description
S1:FS0.PATTERN	1	Slot 1, turn point 0 on in fail safe mode.*
S1:FS2.PATTERN (as Byte)	255	Slot 1, turn points 16-23 on in fail safe mode.*
S1:FS1.PATTERN (as Word)	1	Slot 1, turn point 16 on in fail safe mode.*
S1:FS0.PATTERN (as DWORD)	16777216	Slot 1, turn points 0-23 on in fail safe mode.*

* See Notes and Requirements above.

Fail Safe Time Addressing

After communication between the host and the RTU has been lost, a time delay may be introduced before fail safe mode is engaged for the output module. This time delay can be entered by referencing FS.TIME and is specified in tenths of seconds.

Requirements

None

Important: The value specified in FS.TYPE is not sent to the device until FS.SET is set.

Specifications

Syntax	Data Type	Range	Access
S<slot>:FS.TIME	**Byte**, Word, Short, DWord, Long	N/A	Write Only

Examples

Address	Value	Description
S1:FS.TIME	255	Slot 1, fail safe time delay = 25.5 seconds.
S1:FS.TIME	0	Slot 1, no delay.
S1:FS.TIME	1	Slot 1, fail safe time delay = .1 seconds.

Fail Safe Set Addressing

For any of the fail safe configuration properties (type, time and pattern) to be sent to the device, FS.SET must be set. Immediately after the properties are sent, FS.SET will be reset.

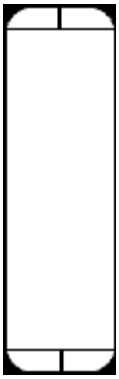
Values
True = Send fail safe configurations to device
False = No action

Requirements
None

Specifications

Syntax	Data Type	Range	Access
S<slot>:FS.SET	**Boolean**	N/A	Write Only

32 Digital Output Module



Digital outputs are used to turn loads on and off. Loads may be lights, motors, solenoids or any type of On/Off device.

Digital outputs in the OptiLogic series come in three types: relay, transistor and solid state relay. Each type has applications it is best suited for. For more information, refer to the module manual.

If there is a loss of communication with the host, the output module will enter a fail safe state. There are three types of states: fail safe to all outputs off, fail safe to a pattern and fail safe to last state. The time in which a fail safe state is entered after loss of communication can be delayed up to 25.5 seconds. If certain points need to be treated differently in fail safe mode, a pattern can be used to signify which points are turned on and which ones are turned off in fail safe mode.

Subtypes
N/A

Address Types
32 Digital Output
Fail Safe Type
Fail Safe Pattern
Fail Safe Time
Fail Safe Set

32 Digital Output Addressing Specifications

Syntax	Data Type	Range	Access
S<slot>:DO<point>	**Boolean**	0-7	Read/Write
S<slot>:DO<offset>*	Byte	0-3	Read/Write
S<slot>:DO<offset>*	Word, Short	0-1	Read/Write
S<slot>:DO<offset>*	DWord, Long	0	Read/Write

* Multiple points can be referenced through the use of this optional syntax. For more information, refer to **Digital I/O Bit Mapping**.

Examples

Address	Value	Description
S1:DO0	1	Slot 1, point 0 (turn point 0 on).
S1:DO0 (as Byte)	255	Slot 1, byte 0 (turn points 0-7 on).
S1:DO1 (as Word)	1	Slot 1, word 1 (turn point 16 on).
S1:DO0 (as DWORD)	4294967295	Slot 1, DWord 0 (turn points 0-31 on).

Fail Safe Type Addressing

There are three types of fail safe states: fail safe to all outputs off, fail safe to a pattern and fail safe to last state. These types may be specified by referencing FS.TYPE.

Values

1 = Fail safe to all outputs off
2 = Fail safe to the pattern contained in FS<point>.PATTERN
3 = Fail safe to last state

Requirements

None

Important: The value specified in FS.TYPE is not sent to the device until FS.SET is set.

Specifications

Syntax	Data Type	Range	Access
S<slot>:FS.TYPE	**Byte**, Word, Short, DWord, Long	N/A	Write Only

Fail Safe Pattern Addressing

Using a fail safe pattern, users can specify which points are set and which ones are cleared when in fail safe mode.

Values

True = turn point on
False = turn point off

Requirements

FS.TYPE must be set to 2 for device to fail safe to a pattern.

Note: If using Byte, Word, Short, DWord or Long data types, the bit pattern determines which points are turned on and which ones are turned off.

Important: The pattern specified in FS.PATTERN is not sent to the device until FS.SET is set.

Specifications

Syntax	Data Type	Range	Access
S<slot>FS<point>.PATTERN	**Boolean**	0-7	Write Only
S<slot>:FS<offset>.PATTERN*	Byte	0-3	Write Only

Syntax	Data Type	Range	Access
S<slot>:FS<offset>.PATTERN*	Word, Short	0-1	Write Only
S<slot>:FS<offset>.PATTERN*	DWord, Long	0	Write Only

* Multiple points can be referenced through the use of this optional syntax. For more information, refer to **Digital I/O Bit Mapping**.

Examples

Address	Value	Description
S1:FS0.PATTERN	1	Slot 1, turn point 0 on in fail safe mode.*
S1:FS0.PATTERN (as Byte)	255	Slot 1, turn points 0-7 on in fail safe mode.*
S1:FS1.PATTERN (as Word)	1	Slot 1, turn point 16 on in fail safe mode.*
S1:FS0.PATTERN (as DWORD)	4294967295	Slot 1, turn points 0-31 on in fail safe mode.*

* See Notes and Requirements above

Fail Safe Time Addressing

After communication between the host and the RTU has been lost, a time delay may be introduced before fail safe mode is engaged for the output module. This time delay can be entered by referencing FS.TIME and is specified in tenths of seconds.

Requirements

None

Important: The value specified in FS.TYPE is not sent to the device until FS.SET is set.

Specifications

Syntax	Data Type	Range	Access
S<slot>:FS.TIME	**Byte**, Word, Short, DWord, Long	N/A	Write Only

Examples

Address	Value	Description
S1:FS.TIME	255	Slot 1, fail safe time delay = 25.5 seconds.
S1:FS.TIME	0	Slot 1, no delay.
S1:FS.TIME	1	Slot 1, fail safe time delay = .1 seconds.

Fail Safe Set Addressing

For any of the fail safe configuration properties (type, time and pattern) to be sent to the device, FS.SET must be set. Immediately after the properties are sent, FS.SET will be reset.

Values

True = Send fail safe configurations to device
False = No action

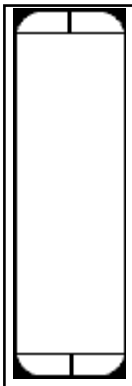
Requirements

None

Specifications

Syntax	Data Type	Range	Access
S<slot>:FS.SET	**Boolean**	N/A	Write Only

2-Channel Analog Input Module



Analog inputs are used to monitor the value of continuously variable field measurements. Typical analog inputs are measurements of temperature, pressure, weight, liquid level, pH, flow rate and many other "real world" parameters.

Each channel of an analog input module takes an analog signal as input and uses a 12-bit A/D converter to put that analog value in digital form.

Subtypes

N/A

2-Channel Analog Input Addressing Specifications

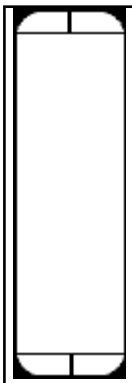
Syntax	Data Type	Range	Access
S<slot>:AI<channel>	**Word**, Short, DWord, Long	1-2	Read Only

Examples

Address	Value	Description
S1:AI1	*	Slot 1, channel 1.
S1:AI2	*	Slot 1, channel 2.

* Value depends on input level, voltage/current ranges, accuracy and so forth.

4-Channel Analog Input Module



Analog inputs are used to monitor the value of continuously variable field measurements. Typical analog inputs are measurements of temperature, pressure, weight, liquid level, pH, flow rate and many other "real world" parameters.

Each channel of an analog input module takes an analog signal as input and uses a 12 bit A/D converter to put that analog value in digital form.

Subtypes
N/A

4-Channel Analog Input Addressing Specifications

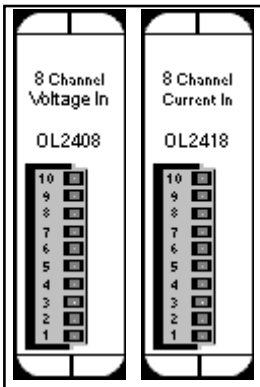
Syntax	Data Type	Range	Access
S<slot>:AI<channel>	**Word**, Short, DWord, Long	1-4	Read Only

Examples

Address	Value	Description
S1:AI2	*	Slot 1, channel 2.
S1:AI4	*	Slot 1, channel 4.

* Value depends on input level, voltage/current ranges, accuracy and so forth.

8-Channel Analog Input Module



Analog inputs are used to monitor the value of continuously variable field measurements. Typical analog inputs are measurements of temperature, pressure, weight, liquid level, pH, flow rate and many other "real world" parameters.

Each channel of an analog input module takes an analog signal as input and uses a 12-bit A/D converter to put that analog value in digital form.

Subtypes
OL2408, OL2418

8-Channel Analog Input Addressing Specifications

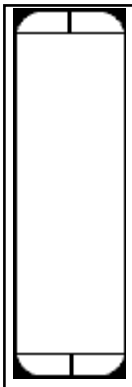
Syntax	Data Type	Range	Access
S<slot>:AI<channel>	**Word**, Short, DWord, Long	1-8	Read Only

Examples

Address	Value	Description
S1:AI2	*	Slot 1, channel 2.
S1:AI8	*	Slot 1, channel 8.

* Values depend on input level, voltage/current ranges, accuracy and so forth.

16-Channel Analog Input Module



Analog inputs are used to monitor the value of continuously variable field measurements. Typical analog inputs are measurements of temperature, pressure, weight, liquid level, pH, flow rate and many other "real world" parameters.

Each channel of an analog input module takes an analog signal as input and uses a 12-bit A/D converter to put that analog value in digital form.

Subtypes

N/A

16-Channel Analog Input Addressing Specifications

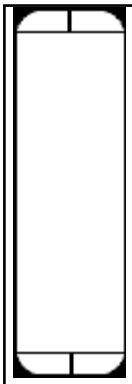
Syntax	Data Type	Range	Access
S<slot>:AI<channel>	**Word**, Short, DWord, Long	1-16	Read Only

Examples

Address	Value	Description
S1:AI2	*	Slot 1, channel 2.
S1:AI16	*	Slot 1, channel 16.

* Values depend on input level, voltage/current ranges, accuracy and so forth.

2-Channel Analog Output Module



Each channel of an analog input module uses a 12 bit D/A converter to produce an analog output signal.

Subtypes

N/A

2-Channel Analog Output Addressing Specifications

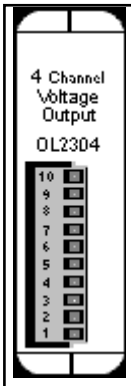
Syntax	Data Type	Range	Access
S<slot>:AO<channel>	**Word**, Short, DWord, Long	1-2	Write Only

Examples

Address	Value	Description
S1:AO1	1024 *	Slot 1, channel 1.
S1:AO2	0 *	Slot 1, channel 2.

* The exact analog output value depends on voltage range, accuracy and so forth.

4-Channel Analog Output Module



Each channel of an analog input module uses a 12 bit D/A converter to produce an analog output signal.

Subtypes

OL2304

Address Types

4-Channel Analog Output

4-Channel Analog Output Range

4-Channel Analog Output Addressing Requirements

As a precaution, the analog output range should be set prior to writing to an analog output channel.

Specifications

Syntax	Data Type	Range	Access
S<slot>:AO<channel>	**Word**, Short, DWord, Long	1-4	Write Only

Examples

Address	Value	Description
S1:AO2	1024 *	Slot 1, channel 2.
S1:AO4	0 *	Slot 1, channel 4.

* The exact analog output value depends on voltage range, accuracy and so forth.

4-Channel Analog Output Range Addressing

Each channel must be configured for a specific analog output range. This is done by setting AO<channel>.RANGE.

Values

0 = 0-5 V
1 = 0-10 V
2 = +/- 5 V
3 = +/- 10 V

Requirements

None.

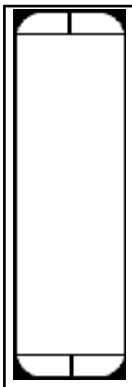
Specifications

Syntax	Data Type	Range	Access
S<slot>:AO<channel>.RANGE	**Word**, Short, DWord, Long	1-4	Write Only

Examples

Address	Value	Description
S1:AO2.RANGE	2	Slot 1, channel 2, set voltage range to +/- 5V.
S1:AO4.RANGE	0	Slot 1, channel 4, set voltage range to 0-5V.

8-Channel Analog Output Module



Each channel of an analog input module uses a 12 bit D/A converter to produce an analog output signal.

Subtypes

N/A

8-Channel Analog Output Addressing Specifications

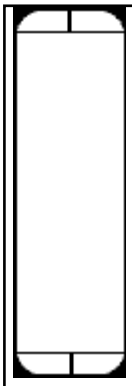
Syntax	Data Type	Range	Access
S<slot>:AO<channel>	**Word**, Short, DWord, Long	1-8	Write Only

Examples

Address	Value	Description
S1:AO2	1024 *	Slot 1, channel 2.
S1:AO8	0 *	Slot 1, channel 8.

* The exact analog output value depends on voltage range, accuracy and so forth.

16-Channel Analog Output Module



Each channel of an analog input module uses a 12-bit D/A converter to produce an analog output signal.

Subtypes

N/A

16-Channel Analog Output Addressing Specifications

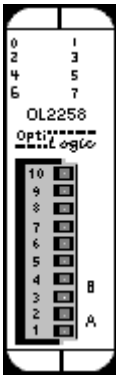
Syntax	Data Type	Range	Access
S<slot>:AO<channel>	**Word**, Short, DWord, Long	1-16	Write Only

Examples

Address	Value	Description
S1:AO2	1024 *	Slot 1, channel 2.
S1:AO16	0 *	Slot 1, channel 16.

* The exact analog output value depends on voltage range, accuracy and so forth.

High-Speed Counter Module



The OL2258 High-Speed Pulse Counter module provides for direct pulse counting for a variety of high-speed pulse interface applications. Typical applications include motion control, metering and velocity measurement.

The OL2258 can be configured to operate in one of three modes:

1. Pulse & Direction (up to 80 kHz input).
2. Up/Down Count (up to 80 kHz input).
3. Quadrature (up to 160 kHz input).

Besides the counter's current count value, the pulse count in the most recent frequency period (user configurable) is also accessible. The OL2258 also contains 2 digital outputs, triggered on when the pulse count reaches the output's minimum range and triggered off when it reaches the output's maximum range.

Subtypes

OL2258

Address Types

Channel Pulse Count

The pulse count for each channel can be referenced using address type C<channel>.COUNT.

Requirements

None

Specifications

Syntax	Data Type	Range	Access
S<slot>:C<channel>.COUNT	**DWord**, Long	1	Read Only

Examples

Address	Value	Description
S1:C1.COUNT	0	Slot 1, channel 1 pulse count is 0.
S1:C1.COUNT	1000	Slot 1, channel 1 pulse count is 1000.

Channel Frequency Data

A pulse count over a preset period of time (C<channel>.COUNT @ (start + period)-C<channel>.COUNT @ start) can be accessed through C<channel>.FREQDATA.

Requirements

C<channel>.FREQPER determines the period and should be set before accessing C<channel>.FREQDATA.

Specifications

Syntax	Data Type	Range	Access
S<slot>:C<channel>.FREQDATA	**Word**, Short, DWord, Long	1	Read Only

Examples

Address	Value	Description
S1:C1.FREQDATA	60000	Slot 1, ch1, if FREQPER = 1 sec, pulse is 60kHz.
S1:C1.FREQDATA	16000	Slot 1, ch1, if FREQPER = 200ms, pulse is 80kHz.

Channel Status: A

Pulse input A can be referenced through CSTS<channel>.A. Count type defines the meaning of this input as follows:

Count Type	Input Definition
Pulse & Direction	Pulse Input
Up/Down Count	Up Pulse Input
Quadrature Encoder Input	Pulse Input

For more information, refer to OptiLogic I/O Modules Manual.

Values

True = Pulse level high
False = Pulse level low

Requirements

CCFG<channel>.COUNTTYPE determines the meaning of this input and should be set before accessing CSTS<channel>.A

Specifications

Syntax	Data Type	Range	Access
S<slot>:CSTS<channel>.A	**Boolean**	1	Read Only

Channel Status: B

Pulse input B can be referenced through CSTS<channel>.B. Count type defines the meaning of this input as follows:

Count Type	Input Definition
Pulse & Direction	Direction Input
Up/Down Count	Down Pulse Input
Quadrature Encoder Input	Pulse Input

For more information, refer to OptiLogic I/O Modules Manual.

Values

True = Pulse level high
False = Pulse level low

Requirements

CCFG<channel>.COUNTTYPE determines the meaning of this input and should be set before accessing CSTS<channel>.B

Specifications

Syntax	Data Type	Range	Access
S<slot>:CSTS<channel>.B	**Boolean**	1	Read Only

Channel Status: Z

Optional input Z provides a means of automatically resetting the count value to a user-defined preset value. Reference Z through CSTS<channel>.Z.

Values

True = If ZPRESETEN set, send PRESET to device. Otherwise, take no action.
False = No action

Requirements

CCFG<channel>.ZPRESETEN must be set in order to enable use of Z in presetting counter

Specifications

Syntax	Data Type	Range	Access
S<slot>:CSTS<channel>.Z	**Boolean**	1	Read Only

Channel Status: LS

Optional limit switch input LS provides a means of automatically resetting the count value to a user-defined preset value. Reference LS through CSTS<channel>.LS.

Values

True = If LSPRESETEN set, send PRESET to device. Otherwise, take no action.
False = No action

Requirements

CCFG<channel>.LSPRESETEN must be set in order to enable use of LS in presetting counter

Specifications

Syntax	Data Type	Range	Access
S<slot>:CSTS<channel>.LS	**Boolean**	1	Read Only

Channel Configuration: Count Type (Mode)

The mode of counter operation is configured via CCFG<channel>.COUNTTYPE.

Values

0 = Pulse & Direction
1 = Up/Down Count
2 = Quadrature

Requirements

None

Specifications

Syntax	Data Type	Range	Access
S<slot>:CCFG<channel>.COUNTTYPE	**Byte**, Word, Short, DWord, Long	1	Read/Write

For more information, refer to OptiLogic I/O Modules Manual.

Examples

Address	Value	Description
S1:CCFG1.COUNTTYPE	1	Slot 1, ch1 count mode set for pulse & dir.
S1:CCFG1.COUNTTYPE	3	Slot 1, ch1 count mode set for quadrature.

Channel Configuration: Frequency Period

Recall that a pulse count over a preset period of time can be accessed through C<channel>.FREQDATA. The period in the FREQDATA formula C<channel>.COUNT @ (start + period)-C<channel>.COUNT @ start is configured through CCFG<channel>.FREQPER.

Values

0 = 1 second count
1 = 200 msec count

Requirements

None

Specifications

Syntax	Data Type	Range	Access
S<slot>:CCFG<channel>.FREQPER	**Byte**, Word, Short, DWord, Long	1	Read/Write

Examples

Address	Value	Description
S1:CCFG1.FREQPER	0	Slot 1, ch1, FREQDATA reflects count in 1 second time window.
S1:CCFG1.FREQPER	1	Slot 1, ch1, FREQDATA reflects count in 200 ms time window.

Channel Configuration: Preset

Establish a preset count value by setting CCFG<channel>.PRESET.

Requirements

For preset to take effect, at least one of the following must be true:
CCFG<channel>.FORCEPRESET be set to TRUE
CCFG<channel>.ZPRESETEN be set to TRUE and Z input be TRUE
CCFG<channel>.LSPRESETEN be set to TRUE and LS input be TRUE

Specifications

Syntax	Data Type	Range	Access
S<slot>:CCFG<channel>.PRESET	**DWord**, Long	1	Write Only

Examples

Address	Value	Description
S1:CCFG1.PRESET	1000	Slot 1, ch1, set preset to 1000.
S1:CCFG1.PRESET	0	Slot 1, ch1, set preset to 0 (default).

Channel Configuration: Force Preset

Set the counter's count value to the preset given in CCFG<channel>PRESET.

Values

True = Send PRESET to device
False = No action

Requirements

None

Specifications

Syntax	Data Type	Range	Access
S<slot>:CCFG<channel>.FORCEPRESET	**Boolean**	1	Write Only

Channel Configuration: Hold Count

Hold the current count value by referencing CCFG<channel>.HOLDCOUNT.

Values

True = Hold count value
False = No action

Requirements

None.

Specifications

Syntax	Data Type	Range	Access
S<slot>:CCFG<channel>.HOLDCOUNT	**Boolean**	1	Write Only

Channel Configuration: Z Preset Enable

Enables Z input to control the presetting of the counter.

Values

True = Z input state configured to preset counter when applicable
False = Z input state has no control over presetting counter

Requirements

None.

Specifications

Syntax	Data Type	Range	Access
S<slot>:CCFG<channel>.ZPRESETEN	**Boolean**	1	Write Only

Channel Configuration: LS Preset Enable

Enables LS input to control the presetting of the counter.

Values

True = LS input state configured to preset counter when applicable
False = LS input state has no control over presetting counter

Requirements

None.

Specifications

Syntax	Data Type	Range	Access
S<slot>:CCFG<channel>.LSPRESETEN	**Boolean**	1	Write Only

Triggered Digital Outputs

There are 2 digital outputs that can be configured to turn on when the counter count is within a specific range. The status of whether the digital output is on or off can be accessed through CTRIGDO<output>.

Requirements

CTRIGDO outputs must have a range specified and enabled, otherwise outputs will remain off.

Specifications

Syntax	Data Type	Range	Access
S<slot>:CTRIGDO<output>	**Boolean**	1-2	Read Only

Examples

Address	Value	Description
S1:CTRIGDO1	1	Slot 1, output 1 is on.
S1:CTRIGDO2	0	Slot 1, output 2 is off.

Triggered Digital Outputs: Minimum Range

Set the range's minimum value to be the counter's count value at which users desire the CTRIGDO<output> to turn from off to on. This minimum range value is referenced as CTRIGDO<output>.MINRANGE.

Requirements

CTRIGDO range usage must be enabled.

Specifications

Syntax	Data Type	Range	Access
S<slot>:CTRIGDO<output>.MINRANGE	**DWord**, Long	1-2	Write Only

Examples

Address	Value	Description
S1:CTRIGDO1.MINRANGE	100	Slot 1, output 1, turn on when count = 100.
S1:CTRIGDO2.MINRANGE	65536	Slot 1, output 2, turn on when count = 65536.

Triggered Digital Outputs: Maximum Range

Set the range's maximum value to be the counter's count value at which users desire the CTRIGDO<output> to turn from on to off. This maximum range value is referenced as CTRIGDO<output>.MAXRANGE.

Requirements

CTRIGDO usage range must be enabled.

Specifications

Syntax	Data Type	Range	Access
S<slot>:CTRIGDO<output>.MAXRANGE	**DWord**, Long	1-2	Write Only

Examples

Address	Value	Description
S1:CTRIGDO1.MAXRANGE	1000	Slot 1, output 1, turn off when count = 1000.
S1:CTRIGDO2.MAXRANGE	0	Slot 1, output 2, turn off when count = 0 (from rollover).

Triggered Digital Outputs: Range Enable

Enable the minimum and maximum range values for CTRIGDO (enable usage of a CTRIGDO output) by referencing CTRIGDO<output>.RANGEEN.

Requirements

None.

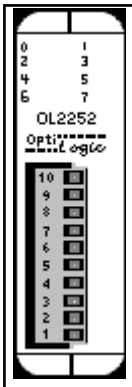
Specifications

Syntax	Data Type	Range	Access
S<slot>:CTRIGDO<output>.RANGEEN	**Boolean**	1-2	Write Only

Examples

Address	Value	Description
S1:CTRIGDO1.RANGEEN	1	Slot 1, output 1 enabled.
S1:CTRIGDO2.RANGEEN	0	Slot 1, output 2 disabled.

2-Channel High-Speed Counter Module



The OL2252 Dual High-Speed Pulse Counter module has two 0-15 kHz pulse counter inputs. Each input is independent of the other. There are a number of configuration options available. See OL2252 manual for details. Of the 10 inputs, 6 can be configured for general-purpose input and is referred to in this driver as CDI (counter digital input). Both channels have a software reset and an optional hardware reset as well as a software enable and optional hardware enable.

Subtypes
OL2252

Address Types
Channel Pulse Count
Channel Enable (Software)
Channel Reset (Software)
Channel Enable (Hardware)
Channel Reset (Hardware)
Channel Debounce Count
8 Digital Input

Channel Pulse Count Addressing
The pulse count for each channel can be referenced using address type C<channel>.COUNT.

Requirements
Channel counter must be enabled either by software or hardware for counter to pulse count.

Specifications

Syntax	Data Type	Range	Access
S<slot>:C<channel>.COUNT	**DWord**, Long	1-2	Read Only

Examples

Address	Value	Description
S1:C1.COUNT	0	Slot 1, channel 1 pulse count is 0.*
S1:C2.COUNT	1000	Slot 1, channel 2 pulse count is 1000.*

* See Requirements above.

Channel Enable (Software) Addressing
Channels can be enabled/disabled through software by referencing address type C<channel>.EN.

Values
True = channel enabled
False = channel disabled*

* Provided channel is not currently hardware enabled.

Requirements
None

Specifications

Address	Value	Description
S1:C1.COUNT	0	Slot 1, channel 1 pulse count is 0.*
S1:C2.COUNT	1000	Slot 1, channel 2 pulse count is 1000.*

Examples

Address	Value	Description
S1:C1.EN	0	Slot 1, ch1 not software enabled.
S1:C2.EN	1	Slot 1, ch2 software enabled.

Channel Reset (Software) Addressing
A channel's pulse count can be reset through software by referencing address type C<channel>.RES.

Values
True = channel pulse count reset
False = channel pulse counting*

* Provided channel is not currently in a hardware reset.

Requirements
None

Specifications

Syntax	Data Type	Range	Access
S<slot>:C<channel>.RES	**Boolean**	1-2	Write Only

Examples

Address	Value	Description
S1:C1.RES	0	Slot 1, ch1 not software reset.
S1:C2.RES	1	Slot 1, ch2 software reset.

Channel Enable (Hardware) Addressing
Channels can be enabled by an external enable signal. Enable/disable this capability by referencing address type CCFG<channel>.HEN.

Values
True = allow hardware enable of channel
False = don't allow hardware enable of channel

Requirements

None

Specifications

Syntax	Data Type	Range	Access
S<slot>:CCFG<channel>.HEN	**Boolean**	1-2	Write Only

Examples

Address	Value	Description
S1:CCFG1.HEN	0	Slot 1, ch1 hardware enable not allowed.
S1:CCFG2.HEN	1	Slot 1, ch2 hardware enabled allowed.

Channel Reset (Hardware) Addressing

A channel's pulse count can be reset by an external reset signal. Enable/disable this capability by referencing address type CCFG<channel>.HRES.

Values

True = allow hardware reset of pulse count
False = don't allow hardware reset of pulse count

Requirements

None

Specifications

Syntax	Data Type	Range	Access
S<slot>:CCFG<channel>.HRES	**Boolean**	1-2	Write Only

Examples

Address	Value	Description
S1:CCFG1.HRES	0	Slot 1, ch1 hardware reset not allowed.
S1:CCFG2.HRES	1	Slot 1, ch2 hardware reset allowed.

Channel Debounce Count Addressing

CCFG<channel>.DBNC establishes the maximum pulse frequency that a channel will count.

Values

2 = 15 kHz
4 = 10 kHz
8 = 5 kHz
16 = 2.5 kHz
40 = 1 kHz

Note: The default value is 15 kHz.

Requirements

None

Specifications

Syntax	Data Type	Range	Access
S<slot>:CCFG<channel>.DBNC	**Byte**, Word, Short, DWord, Long	1-2	Write Only

Examples

Address	Value	Description
S1:CCFG1.DBNC	40	Slot 1, ch1 max pulse freq set to 1 kHz.
S1:CCFG2.DBNC	2	Slot 1, ch2 max pulse freq set to 15 kHz.

8 Digital Input Addressing

The counter's general-purpose inputs are referenced using address type CDI.

Requirements

None

Specifications

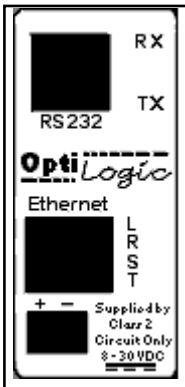
Syntax	Data Type	Range	Access
S<slot>:CDI<point>	**Boolean**	0-7	Read Only
S<slot>:CDI<offset>*	Byte, Word, Short, DWord, Long	0	Read Only

* Multiple points can be referenced through the use of this optional syntax. For more information, refer to **Digital I/O Bit Mapping**.

Examples

Address	Value	Description
S1:CDI0	1	Slot 1, point 0 (terminal 10) is on.
S1:CDI0 (as Byte)	0	Slot 1, byte 0 (points 0-7 are all off).

Base RS232 Port



The Ethernet Base contains an RS232 serial port. Both the transmit buffer and receive buffer of the driver are 48 bytes in size. Likewise, the corresponding tags can be a maximum of 48 bytes. Incoming bytes are appended to the receive buffer as long as they are received in proper time. This time period

depends on the baud rate and is based on a 20-character delay using a 20 ms resolution.

300 Baud => 660 ms

1200 Baud => 160 ms

2400 Baud => 80 ms

4800 Baud => 40 ms

9600 Baud => 20 ms

19200 Baud => 20 ms

For a 4800 baud link, bytes would be appended to the receive buffer as long as they were received within 40 ms of the last byte sent. After 40 ms, any incoming bytes are treated as a new stream. The receive buffer would clear and only contain these new bytes.

If the receive buffer is full and additional bytes are received within the proper time frame, the buffer will reset with these additional bytes. The first 48 bytes will be lost.

Below is a list of possible configurations:

1. Baud rates: 300, 1200, 2400, 4800, 9600 and 19200.

2. Data bits: 7 or 8

3. Parity: none, odd or even

4. Stop bits: 1, 1.5 or 2

An RJ45 connector is required.

Subtypes

OL4054, OL4058

Address Types

Serial Input: Data
Serial Input: Number of Received Bytes
Serial Input: Parity Error
Serial Output: Data
Serial Output: Number of Bytes Sent
Serial Port Configuration: Baud Rate
Serial Port Configuration: #Data Bits
Serial Port Configuration: Parity
Serial Port Configuration: #Stop Bits
Serial Port Configuration: Set

Serial Input: Data Addressing

To receive serial data, reference address type SI<port>.DATA.

Note: The default configuration parameters are 300, n, 8 and 1.

Specifications

Syntax	Data Type	Range	Access
B:SI<port>.DATA [r][c]*	**Byte Array**, Char Array	0	Read Only
B:SI<port>.DATA	String	0	Read Only

* To access as an array, [row][column] form is required. For example, DATA [1][24] would display 24 ASCII bytes in array notation: [x1, x2, x3..x24].

Examples

Address	Value	Description
B:SI0.DATA	"hello"	port 0 input data viewed as a string
B:SI0.DATA [2] [2]	[105, 105][105, 105]	port 0 input data in array form. In string form this would equate to "iiii"

Serial Input: Number of Received Bytes Addressing

The number of received serial bytes (number of bytes in SI<port>.DATA. can be accessed by referencing address type SI<port>.NUMBYTES. If bytes are received within the timeout period mentioned above and are therefore appended to the input DATA buffer, NUMBYTES will reflect the total number of bytes in the input DATA buffer and not the number of bytes received on an individual block read. NUMBYTES will reset upon receiving a new stream.

Requirements

None

Specifications

Syntax	Data Type	Range	Access
B:SI<port>:NUMBYTES	**Byte**, Word, Short, DWord, Long	0	Read Only

Examples

Address	Value	Description
B:SI0.NUMBYTES	0	port 0 input, no bytes in input DATA buffer
B:SI0.NUMBYTES	5	port 0 input, 5 bytes in input DATA buffer

Serial Input: Parity Error Addressing

Reference SI<port>.PARITYERR to determine whether a parity error occurred on the last block read of the serial input port.

Values

True = Parity error occurred
False = No parity error occurred

Requirements
None

Specifications

Syntax	Data Type	Range	Access
B:SI<port>.PARITYERR	**Boolean**	0	Read Only

Examples

Address	Value	Description
B:.SI0.PARITYERR	0	port 0 input, no error
B:SI0.PARITYERR	1	port 0 input, parity error occurred

Serial Output: Data Addressing
To transmit serial data, reference address type SO<port>.DATA.

Note: The default configuration parameters are 300, n, 8 and 1.

Specifications

Syntax	Data Type	Range	Access
B:SO<port>.DATA [r][c]*	**Byte Array**, Char Array	0	Write Only
B:SO<port>.DATA	String	0	Write Only

* To access as an array, [row][column] form is required. For example, DATA [1][24] would send 24 ASCII bytes in array notation: [x1, x2, x3..x24].

Examples

Address	Value	Description
B:SO0.DATA	"hello"	port 0 output, transmit "hello"
B:SO0.DATA [2][2]	[105, 105][105, 105]	port 0 output data in array form. In string form this would equate to transmitting "iiii"

Serial Output: Number of Bytes Sent
Reference SO<port>.BYTESSENT to determine how many bytes were sent on the last transmission. This value is available after transmission of serial data.

Requirements
None

Specifications

Syntax	Data Type	Range	Access
B:SO<port>.BYTESSENT	**Byte**, Word, Short, DWord, Long	0	Read Only

Examples

Address	Value	Description
B:SO0.BYTESSENT	0	port 0 output, no bytes sent on last transmission
B:SO0.BYTESSENT	47	port 0 output, 47 bytes sent

Serial Port Configuration: Baud Rate

To configure the baud rate for a serial port, reference SCFG<port>.BAUD

Values

1 = 300
2 = 1200
3 = 2400
4 = 4800
5 = 9600
6 = 19200

Note: The default value is 300 baud.

Requirements

None

Specifications

Syntax	Data Type	Range	Access
B:SCFG<port>.BAUD	**Byte**, Word, Short, DWord, Long	0	Write Only

Examples

Address	Value	Description
B:SCFG0.BAUD	4	port 0, baud = 4800
B:SCFG0.BAUD	6	port 0, baud = 19200

Serial Port Configuration: #Data Bits

To configure the number of data bits for a serial port, reference SCFG<port>.DATABITS

Values

7 or 8

Note: The default value is 8 data bits.

Requirements

None

Specifications

Syntax	Data Type	Range	Access
B:SCFG<port>.DATABITS	**Byte**, Word, Short, DWord, Long	0	Write Only

Examples

Address	Value	Description
B:SCFG0.DATABITS	7	port 0 configured for 7 data bits
B:SCFG0.DATABITS	8	port 0 configured for 8 data bits

Serial Port Configuration: Parity

To configure the parity for a serial port, reference SCFG<port>.BAUD

Values

0 = none
1 = odd
2 = even

Note: The default value is none.

Requirements:

None

Specifications

Syntax	Data Type	Range	Access
B:SCFG<port>.PARITY	**Byte**, Word, Short, DWord, Long	0	Write Only

Examples

Address	Value	Description
B:SCFG0.PARITY	0	port 0 configured for no parity
B:SCFG0.PARITY	2	port 0 configured for even parity

Serial Port Configuration: #Stop Bits

To configure the number of stop bits for a serial port, reference SCFG<port>.STOPBITS

Values

1 = 1
2 = 2
3 = 1.5

Note: The default value is 1.

Requirements

None

Specifications

Syntax	Data Type	Range	Access
B:SCFG<port>.STOPBITS	**Byte**, Word, Short, DWord, Long	0	Write Only

Examples

Address	Value	Description
B:SCFG0.STOPBITS	1	port 0 configured for 1 stop bit
B:SCFG0.STOPBITS	3	port 0 configured for 1.5 stop bits

Serial Port Configuration: Set

For any of the serial port configuration properties (baud, parity and so forth) to be sent to the device, SCFG.SET must be set. Immediately after the properties are sent, SCFG.SET will be reset.

Values

True = Send serial port configurations to device
False = No action

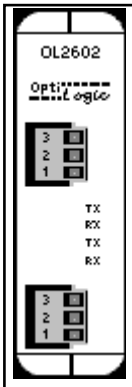
Requirements

None

Specifications

Syntax	Data Type	Range	Access
B:SCFG<port>.SET	**Boolean**	N/A	Write Only

Dual RS232 Port Module



The OL2602 is a dual RS232 port serial module. Both the transmit buffer and receive buffer of the driver are 48 bytes in size. Likewise, the corresponding tags can be a maximum of 48 bytes. Incoming bytes are appended to the receive buffer as long as they are received in proper time. This time period depends on the baud rate and is based on a 20-character delay using a 20 ms resolution.

1200 Baud => 160 ms

2400 Baud => 80 ms

4800 Baud => 40 ms

9600 Baud => 20 ms

19200 Baud => 20 ms

For a 4800 baud link, bytes would be appended to the receive buffer as long as they were received within 40 ms of the last byte sent. After 40 ms, any incoming bytes are treated as a new stream. The receive buffer would clear and only contain these new bytes.

If the receive buffer is full and additional bytes are received within the proper time frame, the buffer will reset with these additional bytes. The first 48 bytes will be lost.

Below is a list of possible configurations:

1. Baud rates: 1200, 2400, 4800, 9600 and 19200.

2. Data bits: 7 or 8

3. Parity: none, odd or even

4. Stop bits: 1, 1.5 or 2

If using the OL4058 RTU, a maximum of one OL2602 modules may be used and it must be placed in slot 0. For the OL4054 RTU, two OL2602 may be used and they must be in either slot 0 or 1.

Port 1 is the top terminal strip, Port 2 is the bottom terminal strip. Below is the pin assignments:

1: Signal ground

2: TX

3: RX

Subtypes
OL2602

Address Types
Serial Input: Data
Serial Input: Number of Received Bytes
Serial Input: Parity Error
Serial Output: Data
Serial Output: Number of Bytes Sent
Serial Port Configuration: Baud Rate
Serial Port Configuration: #Data Bits
Serial Port Configuration: Parity
Serial Port Configuration: #Stop Bits
Serial Port Configuration: Set

Serial Input: Data Addressing
To receive serial data, reference address type SI<port>.DATA.

Note: The default configuration parameters are 9600, n, 8 and 1.

Specifications

Syntax	Data Type	Range	Access
S<slot>:SI<port>.DATA [r][c]*	**Byte Array**, Char Array	1-2	Read Only
S<slot>:SI<port>.DATA	String	1-2	Read Only

* To access as an array, [row][column] form is required. For example, DATA [1][24] would display 24 ASCII bytes in array notation: [x1, x2, x3..x24])

Examples

Address	Value	Description
S0:SI1.DATA	"hello"	Slot 0, port 1input data viewed as a string.
S0:SI2.DATA [2] [2]	[105, 105] [105, 105]	Slot 0, port 2 input data in array form. In string form this would equate to "iiii".

Serial Input: Number of Received Bytes Addressing

The number of received serial bytes (number of bytes in SI<port>.DATA) can be accessed by referencing address type SI<port>.NUMBYTES. If bytes are received within the timeout period mentioned above and are therefore appended to the input DATA buffer, NUMBYTES will reflect the total number of bytes in the input DATA buffer and not the number of bytes received on an individual block read. NUMBYTES will reset upon receiving a new stream.

Requirements

None

Specifications

Syntax	Data Type	Range	Access
S<slot>:SI<port>.NUMBYTES	**Byte**, Word, Short, DWord, Long	1-2	Read Only

Examples

Address	Value	Description
S1:SI1.NUMBYTES	0	Slot 1, port 1 input, no bytes in input DATA buffer
S1:SI2.NUMBYTES	5	Slot1, port2 input, 5 bytes in input DATA buffer

Serial Input: Parity Error Addressing

Reference SI<port>.PARITYERR to determine whether a parity error occurred on the last block read of the serial input port.

Values

True = Parity error occurred
False = No parity error occurred

Requirements

None

Specifications

Syntax	Data Type	Range	Access
S<slot>:SI<port>.PARITYERR	**Boolean**	1-2	Read Only

Examples

Address	Value	Description
S1:SI1.PARITYERR	0	Slot 1, port 1 input, no error.
S1:SI2.PARITYERR	1	Slot 1, port 2 input, parity error occurred.

Serial Output: Data Addressing

To transmit serial data, reference address type SO<port>.DATA.

Note: The default configuration parameters are 9600, n, 8 and 1.

Specifications

Syntax	Data Type	Range	Access
S<slot>:SO<port>.BYTESSENT	**Byte**, Word, Short, DWord, Long	1-2	Read Only

* To access as an array, [row][column] form is required. For example, DATA [1][24] would send 24 ASCII bytes in array notation: [x1, x2, x3..x24])

Examples

Address	Value	Description
S0:SO1.DATA	"hello"	Slot 0, port 1output, transmit "hello".
S0:SO2.DATA [2][2]	[105, 105][105, 105]	Slot 0, port 2 output data in array form. In string form this would equate to transmitting "iiii".

Serial Output: Number of Bytes Sent

Reference SO<port>.BYTESSENT to determine how many bytes were sent on the last transmission. This value is available after transmission of serial data.

Requirements

None

Specifications

Syntax	Data Type	Range	Access
S<slot>:SO<port>.BYTESSENT	**Byte**, Word, Short, DWord, Long	1-2	Read Only

Examples

Address	Value	Description
S1:SO1.BYTESSENT	0	Slot 1, port 1 output, no bytes sent on last transmission.
S1:SO1.BYTESSENT	47	Slot 1, port 1 output, 47 bytes sent.

Serial Port Configuration: Baud Rate

To configure the baud rate for a serial port, reference SCFG<port>.BAUD

Values

2 = 1200
3 = 2400
4 = 4800
5 = 9600
6 = 19200

Note: The default value is 9600 baud.

Requirements

None

Specifications

Syntax	Data Type	Range	Access
S<slot>:SCFG<port>.BAUD	**Byte**, Word, Short, DWord, Long	1-2	Write Only

Examples

Address	Value	Description
S1:SCFG1.BAUD	4	Slot 1, port 1, baud = 4800.
S1:SCFG2.BAUD	6	Slot 1, port 2, baud = 19200.

Serial Port Configuration: #Data Bits

To configure the number of data bits for a serial port, reference SCFG<port>.DATABITS

Values

7 or 8

Note: The default value is 8 data bits.

Requirements

None

Specifications

Syntax	Data Type	Range	Access
S<slot>:SCFG<port>.DATABITS	**Byte**, Word, Short, DWord, Long	1-2	Write Only

Examples

Address	Value	Description
S1:SCFG1.DATABITS	7	Slot 1, port 1 configured for 7 data bits.
S1:SCFG2.DATABITS	8	Slot 1, port 2 configured for 8 data bits.

Serial Port Configuration: Parity

To configure the parity for a serial port, reference SCFG<port>.BAUD

Values

0 = none
1 = odd
2 = even

Note: The default value is none.

Requirements
None

Specifications

Syntax	Data Type	Range	Access
S<slot>:SCFG<port>.PARITY	**Byte**, Word, Short, DWord, Long	1-2	Write Only

Examples

Address	Value	Description
S1:SCFG1.PARITY	0	Slot 1, port 1 configured for no parity.
S1:SCFG2.PARITY	2	Slot 1, port 2 configured for even parity.

Serial Port Configuration: #Stop Bits
To configure the number of stop bits for a serial port, reference SCFG<port>.STOPBITS

Values
1 = 1
2 = 2
3 = 1.5

Note: The default value is 1.

Requirements
None

Specifications

Syntax	Data Type	Range	Access
S<slot>:SCFG<port>.STOPBITS	**Byte**, Word, Short, DWord, Long	1-2	Write Only

Examples

Address	Value	Description
S1:SCFG1.STOPBITS	1	Slot 1, port 1 configured for 1 stop bit.
S1:SCFG2.STOPBITS	3	Slot 1, port 2 configured for 1.5 stop bits.

Serial Port Configuration: Set
For any of the serial port configuration properties (baud, parity and so forth) to be sent to the device, SCFG.SET must be set. Immediately after the properties are sent, SCFG.SET will be reset.

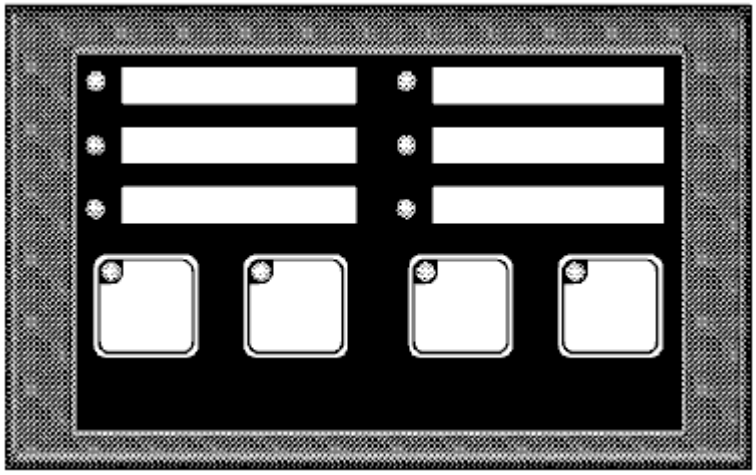
Values
True = Send serial port configurations to device
False = No action

Requirements
None

Specifications

Syntax	Data Type	Range	Access
S<slot>:SCFG<port>.SET	**Boolean**	N/A	Write Only

OL3406 Operator Panel



The OL3406 Operator Panel consists of 4 user definable pushbuttons with built-in LEDs and 6 indicator LEDs.

Subtypes
OL3406

Address Types
Pushbutton LED On State
Pushbutton LED Flash State
Pushbutton LED Separation State
Pushbutton State
Pushbutton Configuration
Force Pushbutton State
Indicator LED On State
Indicator LED Flash State

Pushbutton LED On State Addressing
Each button LED can be forced On/Off without the button being pressed. This can be achieved by referencing address type BTNLED<button>.ON.

Values
True = on

False = off

Requirements

Button LED Separation state must be set. Button(s) must also be configured for momentary action. For more information, refer to **PushButton Configuration**.

Specifications

Syntax	Data Type	Range	Access
P:BTNLED<button number>.ON	**Boolean**	0-3	Write Only

Examples

Address	Value	Description
P:BTNLED0.ON	1	Turn button 0 LED on (left most button).*
P:BTNLED3.ON	1	Turn button 3 LED on (right most button).*

* See Requirements above.

Pushbutton LED Flash State Addressing

Each button LED can be forced to flash On/Off without the button being pressed. This can be achieved by referencing address type BTNLED<button>.FLASH.

Values

True = flash on
False = flash off

Requirements

Button LED ON state must be set for the corresponding button and button LED Separation state must also be set. Button(s) must be configured for momentary action. For more information, refer to **PushButton Configuration Addressing**.

Specifications

Syntax	Data Type	Range	Access
P:BTNLED<button number>.FLASH	**Boolean**	0-3	Write Only

Examples

Address	Value	Description
P:BTNLED0.FLASH	1	Flash button 0 LED (left most button).*
P:BTNLED3.FLASH	1	Flash button 3 LED (right most button).*

* See Requirements above.

Pushbutton LED Separation State Addressing

When LED Separation is set, one is capable of controlling the on and flash state of individual button LEDs (see Pushbutton LED On and Flash State Addressing above).

Values

True = separation on
False = separation off

Requirements
None.

Specifications

Syntax	Data Type	Range	Access
P:LEDSEP	**Boolean**	N/A	Read/Write

Pushbutton State Addressing
Button state (pressed/not pressed) can be monitored by referencing address type BTNSTATUS<button>.

Values
True = pressed
False = not pressed

Requirements
None.

Note: Button state depends on the button configuration. For more information, refer to **Pushbutton Configuration Addressing**.

Specifications

Syntax	Data Type	Range	Access
P:BTNSTATUS<button number>	**Boolean**	0-3	Read Only

Examples

Address	Value	Description
P:BTNSTATUS0	1	Button 0 is pressed (left most button).*
P:BTNSTATUS3	0	Button 3 is not pressed (right most button).

* See Note above.

Pushbutton Configuration Addressing
Each button can be configured to either latch their state (alternate action) or hold it momentarily while its being pressed/not pressed.

Values
True = alternate action
False = momentary action

Requirements
None.

Specifications

Syntax	Data Type	Range	Access
P:BTNCFG<button number>	**Boolean**	0-3	Read/Write

Examples

Address	Value	Description
P:BTNCFG0	1	Button 0 is configured for alternate action.
P:BTNCFG3	0	Button 3 is configured for momentary action.

* See Note above.

Force Pushbutton State Addressing

Each button can be forced to a desired state. There are three means of forcing button state.

EQUALS:	Desired button state = specified state
OR:	Desired button state = (current state BITWISE OR specified state)
AND:	Desired button state = NOT(current state BITWISE AND specified state)

Values (Desired button states)

True = button on ("pressed")
False = button off ("not pressed")

Requirements

Button must be configured for alternate action.

Specifications

Below are the three means of forcing button state in detail:

Syntax	Data Type	Range	Access
P:BTNFORCE<button number>.EQUALS	**Boolean**	0-3	Write Only
P:BTNFORCE<button number>.OR	**Boolean**	0-3	Write Only
P:BTNFORCE<button number>.AND	**Boolean**	0-3	Write Only

Examples

Address	Value	Description
P:BTNFORCE0.EQUALS	1	Force button 0 state to be on.
P:BTNFORCE0.OR	1	OR current state with 1. Force button 0 state to be on.
P:BTNFORCE0.AND	1	If button 0 state is currently on, set state to off. Otherwise, do nothing.

* See Requirements above.

Indicator LED On State Addressing

Each indicator LED can be forced On/Off by referencing address type INDLED<led>.ON.

Values

True = on

False = off

Requirements
None.

Specifications

Syntax	Data Type	Range	Access
P:INDLED<led number>.ON	**Boolean**	0-5	Write Only

Examples

Address	Value	Description
P:INDLED0.ON	1	Turn indicator LED 0 on (upper left LED).
P:INDLED5.ON	1	Turn indicator LED 5 on (bottom right LED).

Indicator LED Flash State Addressing
Each indicator LED can be forced to flash On/Off by referencing address type INDLED<led>.FLASH.

Values
True = flash on
False = flash off

Requirements
Indicator LED ON state must be set for the corresponding LED.

Specifications

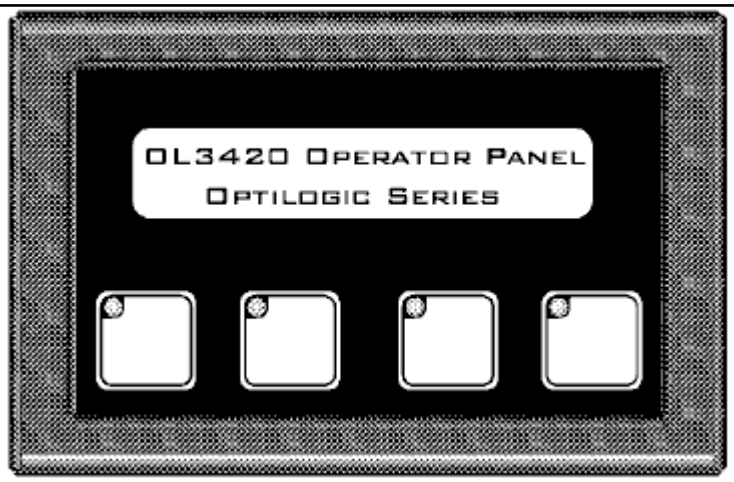
Syntax	Data Type	Range	Access
P:INDLED<led number>.FLASH	**Boolean**	0-5	Write Only

Examples

Address	Value	Description
P:INDLED1.FLASH	1	Flash indicator LED 1 (upper right LED).*
P:INDLED4.FLASH	1	Flash indicator LED 4 (lower left LED).*

* See Requirements above.

OL3420 Operator Panel



The OL3420 Operator Panel consists of 4 user definable pushbuttons with built-in LEDs and a 2 line x 20 character backlit LCD display

Subtypes
OL3420

Address Types
Pushbutton LED On State
Pushbutton LED Flash State
Pushbutton LED Separation State
Pushbutton State
Pushbutton Configuration
Force Pushbutton State
Alphanumeric Display

Pushbutton LED On State Addressing
Each button LED can be forced On/Off without the button being pressed. This can be achieved by referencing address type BTNLED<button>.ON.

Values
True = on
False = off

Requirements
Button LED Separation state must be set. Button(s) must also be configured for momentary action. For more information, refer to **PushButton Configuration**.

Specifications

Syntax	Data Type	Range	Access
P:BTNLED<button number>.ON	**Boolean**	0-3	Write Only

Examples

Address	Value	Description
P:BTNLED0.ON	1	Turn button 0 LED on (left most button).*
P:BTNLED3.ON	1	Turn button 3 LED on (right most button).*

* See Requirements above.

Pushbutton LED Flash State Addressing

Each button LED can be forced to flash On/Off without the button being pressed. This can be achieved by referencing address type BTNLED<button>.FLASH.

Values

True = flash on
False = flash off

Requirements

Button LED ON state must be set for the corresponding button and button LED Separation state must be set. Button(s) must also be configured for momentary action. For more information, refer to **PushButton Configuration**.

Specifications

Syntax	Data Type	Range	Access
P:BTNLED<button number>.FLASH	**Boolean**	0-3	Write Only

Examples

Address	Value	Description
P:BTNLED0.FLASH	1	Flash button 0 LED (left most button).*
P:BTNLED3.FLASH	1	Flash button 3 LED (right most button).*

* See Requirements above.

Pushbutton LED Separation State Addressing

When LED Separation is set, one is capable of controlling the on and flash state of individual button LEDs (see Pushbutton LED On and Flash State Addressing above).

Values

True = separation on
False = separation off

Requirements

None.

Syntax	Data Type	Range	Access
P:LEDSEP	**Boolean**	N/A	Read/Write

Pushbutton State Addressing

Button state (pressed/not pressed) can be monitored by referencing address type BTNSTATUS<button>.

Values
True = pressed
False = not pressed

Requirements
None.

Note: Button state depends on the button configuration. See **Pushbutton Configuration Addressing** below.

Specifications

Syntax	Data Type	Range	Access
P:BTNSTATUS<button number>	**Boolean**	0-3	Read Only

Examples

Address	Value	Description
P:BTNSTATUS0	1	Button 0 is pressed (left most button).*
P:BTNSTATUS3	0	Button 3 is not pressed (right most button).

* See Note above.

Pushbutton Configuration Addressing
Each button can be configured to either latch their state (alternate action) or hold it momentarily while its being pressed/not pressed.

Values
True = alternate action
False = momentary action

Requirements
None.

Specifications

Syntax	Data Type	Range	Access
P:BTNCFG<button number>	**Boolean**	0-3	Read/Write

Examples

Address	Value	Description
P:BTNCFG0	1	Button 0 is configured for alternate action.
P:BTNCFG3	0	Button 3 is configured for momentary action.

* See Note above.

Force Pushbutton State Addressing
Each button can be forced to a desired state. There are three ways to force button state.

EQUALS:	Desired button state = specified state
OR:	Desired button state = (current state BITWISE OR specified state)
AND:	Desired button state = NOT(current state BITWISE AND specified state)

Values (Desired button states)

True = button on ("pressed")
False = button off ("not pressed")

Requirements

Button must be configured for alternate action.

Specifications

Below are the three means of forcing button state in detail.

Syntax	Data Type	Range	Access
P:BTNFORCE<button number>.EQUALS	**Boolean**	0-3	Write Only
P:BTNFORCE<button number>.OR	**Boolean**	0-3	Write Only
P:BTNFORCE<button number>.AND	**Boolean**	0-3	Write Only

Examples

Address	Value	Description
P:BTNFORCE0.EQUALS	1	Force button 0 state to be on.
P:BTNFORCE0.OR	1	OR current state with 1. Force button 0 state to be on.
P:BTNFORCE0.AND	1	If button 0 state is currently on, set state to off. Otherwise, do nothing.

* See Requirements above.

Alphanumeric Display Addressing

There are two lines of display for alphanumeric strings of length 20 characters or less.

Requirements

None.

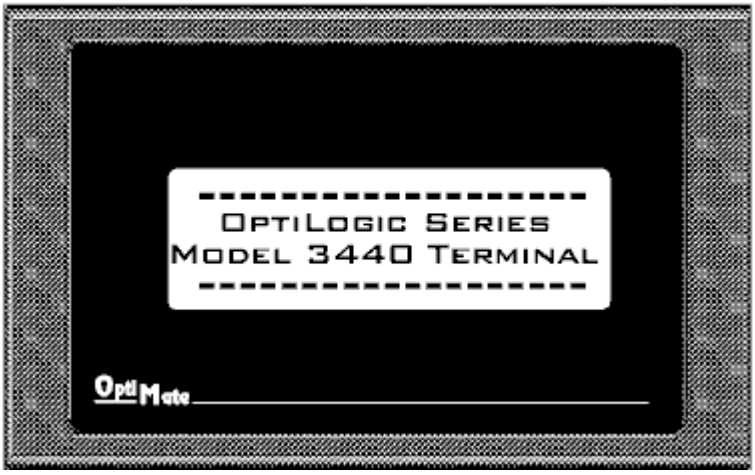
Specifications

Syntax	Data Type	Range	Access
P:LINE<line number>	**String**	0-1	Write Only

Examples

Address	Value	Description	
P:LINE<line number>	**String**	0-1	Write Only

OL3440 Operator Panel



The OL3440 Operator Panel consists of a 4 line x 20 character backlit LCD display

Subtypes
OL3440

Alphanumeric Display Addressing
There are four lines of display for alphanumeric strings of length 20 characters or less.

Requirements
None.

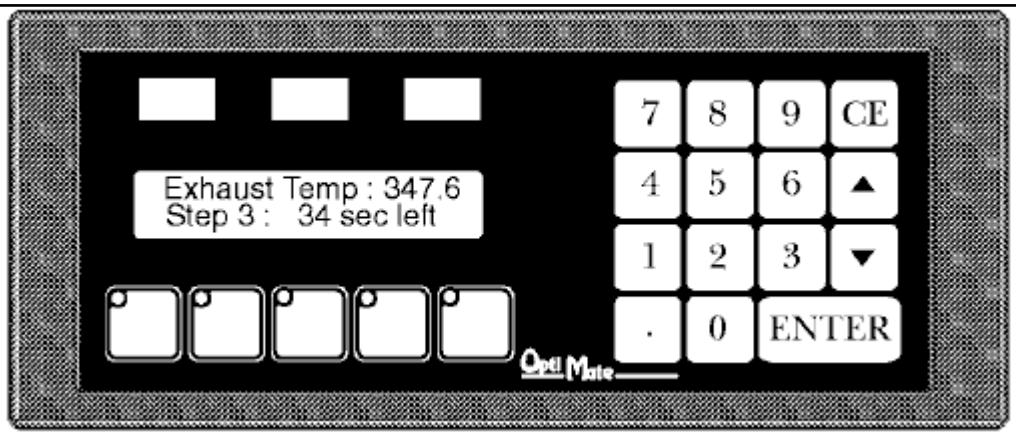
Specifications

Syntax	Data Type	Range	Access
P:LINE<line number>	**String**	0-3	Write Only

Examples

Address	Value	Description
P:LINE0	"hello"	Set line 0 (top line) string to "hello".
P:LINE3	"world"	Set line 3 (bottom line) string to "world".

OL3850 Operator Panel



The OL3850 Operator Panel consists of the following: 2 line x 20 character backlit LCD display, 5 user definable pushbuttons with built-in LEDs, 3 color coded indicator light bars and a numeric keypad. This panel can be used to display messages, accept numeric data input, display status indications and select functions.

Subtypes
OL3850

Address Types
Pushbutton LED On State
Pushbutton LED Flash State
Pushbutton LED Separation State
Pushbutton State
Pushbutton Configuration
Force Pushbutton State
Light Bar On State
Light Bar Flash State
Alphanumeric Display
Keypad Data
Keypad Data Available
Keypad Arrow Max
Keypad Arrow Min

Pushbutton LED On State Addressing
Each button LED can be forced On/Off without the button being pressed. This can be achieved by referencing address type BTNLED<button>.ON.

Values
True = on
False = off

Requirements
Button LED Separation state must be set. Button(s) must also be configured for momentary action. For more information, refer to PushButton Configuration Addressing .

Specifications

Syntax	Data Type	Range	Access
P:BTNLED<button number>.ON	**Boolean**	0-4	Write Only

Examples

Address	Value	Description
P:BTNLED0.ON	1	Turn button 0 LED on (left most button).*
P:BTNLED4.ON	1	Turn button 4 LED on (right most button).*

* See Requirements above.

Pushbutton LED Flash State Addressing

Each button LED can be forced to flash On/Off without the button being pressed. This can be achieved by referencing address type BTNLED<button>.FLASH.

Values

True = flash on
False = flash off

Requirements

Button LED ON state must be set for the corresponding button and button LED Separation state must be set. Button(s) must also be configured for momentary action. For more information, refer to **PushButton Configuration Addressing**.

Syntax	Data Type	Range	Access
P:BTNLED<button number>.FLASH	**Boolean**	0-4	Write Only

Examples

Address	Value	Description
P:BTNLED0.FLASH	1	Flash button 0 LED (left most button).*
P:BTNLED4.FLASH	1	Flash button 4 LED (right most button).*

* See Requirements above.

Pushbutton LED Separation State Addressing

When LED Separation is set, one is capable of controlling the on and flash state of individual button LEDs. For more information, refer to "Pushbutton LED On and Flash State Addressing" above.

Values

True = separation on
False = separation off

Requirements

None.

Specifications

Syntax	Data Type	Range	Access
P:LEDSEP	**Boolean**	N/A	Read/Write

Pushbutton State Addressing

Button state (pressed/not pressed) can be monitored by referencing address type BTNSTATUS<button>.

Values

True = pressed
False = not pressed

Requirements

None.

Note: Button state depends on the button configuration. For more information, refer to **Pushbutton Configuration Addressing** below.

Specifications

Syntax	Data Type	Range	Access
P:BTNSTATUS<button number>	**Boolean**	0-4	Read Only

Examples

Address	Value	Description
P:BTNSTATUS0	1	Button 0 is pressed (left most button).*
P:BTNSTATUS4	0	Button 4 is not pressed (right most button).

* See Note above.

Pushbutton Configuration Addressing

Each button can be configured to either latch their state (alternate action) or hold it momentarily while its being pressed/not pressed.

Values

True = alternate action
False = momentary action

Requirements

None.

Specifications

Syntax	Data Type	Range	Access
P:BTNCFG<button number>	**Boolean**	0-4	Read/Write

Examples

Address	Value	Description
P:BTNCFG0	1	Button 0 is configured for alternate action.
P:BTNCFG4	0	Button 4 is configured for momentary action.

Force Pushbutton State Addressing

Each button can be forced to a desired state. There are three ways to force button state.

EQUALS:	Desired button state = specified state
OR:	Desired button state = (current state BITWISE OR specified state)
AND:	Desired button state = NOT(current state BITWISE AND specified state)

Values (Desired button states)

True = button on ("pressed")
False = button off ("not pressed")

Requirements

Button must be configured for alternate action.

Specifications

Below are the three means of forcing button state in detail:

Syntax	Data Type	Range	Access
P:BTNFORCE<button number>.EQUALS	**Boolean**	0-4	Write Only
P:BTNFORCE<button number>.OR	**Boolean**	0-4	Write Only
P:BTNFORCE<button number>.AND	**Boolean**	0-4	Write Only

Examples

Address	Value	Description
P:BTNFORCE0.EQUALS	1	Force button 0 state to be on.*
P:BTNFORCE0.OR	1	OR current state with 1. Force button 0 state to be on.*
P:BTNFORCE0.AND	1	If button 0 state is currently on, set state to off. otherwise, do nothing.*

* See Requirements above.

Light Bar On State Addressing

Each light bar can be forced On/Off by referencing address type LITEBAR<bar>.ON.

Values

True = on
False = off

Requirements

None.

Specifications

Syntax	Data Type	Range	Access
P:LITEBAR<light bar>.ON	**Boolean**	0-2	Write Only

Examples

Address	Value	Description
P:LITEBAR0.ON	1	Turn light bar 0 on (left most bar).
P:LITEBAR2.ON	1	Turn light bar 2 on (right most bar).

Light Bar Flash State Addressing

Each light bar can be forced to flash On/Off by referencing address type LITEBAR<bar>.FLASH.

Values

True = flash on
False = flash off

Requirements

Light Bar ON state must be set for the corresponding light bar.

Specifications

Syntax	Data Type	Range	Access
P:LITEBAR<light bar>.FLASH	**Boolean**	0-2	Write Only

Examples

Address	Value	Description
P:LITEBAR0.FLASH	1	Flash light bar 0 LED (left most bar).*
P:LITEBAR2.FLASH	1	Flash light bar 2 LED (right most bar).*

* See Requirements above.

Alphanumeric Display Addressing

There are two lines of display for alphanumeric strings of length 20 characters or less. Keypad data may also be inserted into the text string by using the caret (^) as a placeholder for each digit (including the decimal point) of the keypad data.

Requirements

None.

Specifications

Syntax	Data Type	Range	Access
P:LINE<line number>	**String**	0-1	Write Only

Examples

Address	Value	Description
P:LINE0	"hello"	Set line 0 (top line) string to "hello".

Address	Value	Description
P:LINE1	"world"	Set line 1 (bottom line) string to "world".
P:LINE0	"^^^^^^^^"	If keypad data = 1234.56, then line 0 string will be "1234.56".*

* For more information, refer to "Keypad Data Addressing" below.

Keypad Data Addressing

Numeric data can be read from (entered via keypad or this address) and written to (via this address) the panel keypad. Float precision can be specified by appending a bit (0 to 10 allowed) to the KDATA address. This will represent the floating point precision on any writes to the device. If no precision is specified (the default case), the precision will be set such that the number of digits in the integer and fractional parts sums up to 10. If a precision is specified such that the 10-digit limit is exceeded, the precision will be set to the default value previously discussed. Note that the value written to the device may differ from the value displayed in the client. This may be due to floating point round off and truncation errors from the driver, client or both. For more information on floating point precision, refer to **Device Setup**.

Note: Keypad data may be altered using the arrows located on the keypad. Upper and lower bounds set by Arrow Max and Arrow Min respectively, will only limit the data set by the arrows, not the data set in KDATA. Only the data type can place constraints on the upper and lower limits of the keypad data when set using KDATA.

Specifications

Syntax	Data Type	Range	Access
P:KDATA	**Double**, Float, DWord	N/A	Read/Write
P:KDATA.<precision>	**Double**, Float, DWord	N/A	Read/Write

Examples (Writes)

Address	Value	Description
P.KDATA	10.123456	10.123456 will be written to the device.
P:KDATA.4 as a float	10.123456	10.1235 will be written to the device.
P:KDATA	9999999999	9999999999 will be written to the device.

Keypad Data Available Addressing

To determine if new keypad data has been entered at the panel, reference KDATAREADY. This flag can be cleared by writing to KDATA.

Values

True = new data has been entered
False = all data has been read from driver, no new data

Syntax	Data Type	Range	Access
P:KDATAREADY	**Boolean**	N/A	Read Only

Keypad Arrow Max Addressing

Panel keypad data entered via the arrows, can be upper bounded by referencing ARROW.MAX. Any keypad data entered above this max will automatically get set to this max value.

Note: Initially, before ARROW.MAX is set, the upper limit internally to the device is 9999999999. When the limit is set, keypad data cannot exceed the size of ARROW.MAX which is 32 bits (DWord).

Specifications

Syntax	Data Type	Range	Access
P:ARROW.MAX	**DWord**	N/A	Write Only

Keypad Arrow Min Addressing

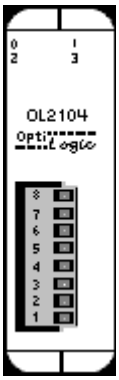
Panel keypad data entered via the arrows, can be lower bounded by referencing ARROW.MIN. Any keypad data entered below this min will automatically get set to this min value.

Note: Initially, before ARROW.MIN is set, the lower limit internally to the device is 0.

Specifications

Syntax	Data Type	Range	Access
P:ARROW.MIN	**DWord**	N/A	Write Only

4 Digital Output Module



Digital outputs are used to turn loads on and off. Loads may be lights, motors, solenoids or any type of On/Off device.

Digital outputs in the OptiLogic series come in three types: relay, transistor and solid state relay. Each type has applications it is best suited for. For more information, refer to the module manual.

If there is a loss of communication with the host, the output module will enter a fail safe state. There are three types of states: fail safe to all outputs off, fail safe to a pattern and fail safe to last state. The time in which a fail safe state is entered after loss of communication can be delayed up to 25.5 seconds. If certain points need to be treated differently in fail safe mode, a pattern can be used to signify which points are turned on and which ones are turned off in fail safe mode.

Subtypes

OL2104

Address Types

4 Digital Output
Fail Safe Type
Fail Safe Pattern
Fail Safe Time
Fail Safe Set

4 Digital Output Addressing Specifications

Syntax	Data Type	Range	Access
S<slot>:DO<point>	**Boolean**	0-3	Read/Write
S<slot>:DO<offset>*	Byte**	0	Read/Write
S<slot>:DO<offset>*	Word, Short**	0	Read/Write
S<slot>:DO<offset>*	DWord, Long**	0	Read/Write

* Multiple points can be referenced through the use of this optional syntax. For more information, refer to **Digital I/O Bit Mapping**.
** Since only the lower nibble of the least significant byte is being used, any value entered above 15 will be cropped.

Examples

Address	Value	Description
S1:DO0	1	Slot 1, point 0 (turn point 0 on).
S1:DO0 (as Byte)	15	Slot 1, byte 0 (turn points 0-3 on).
S1:DO0 (as Word)	0	Slot 1, word 0 (turn all points off).
S1:DO0 (as DWORD)	15	Slot 1, DWord 0 (turn points 0-3 on).

Fail Safe Type Addressing

There are three types of fail safe states: fail safe to all outputs off, fail safe to a pattern and fail safe to last state. These types may be specified by referencing FS.TYPE.

Values

1 = Fail safe to all outputs off
2 = Fail safe to the pattern contained in FS<point>.PATTERN
3 = Fail safe to last state

Requirements

None

Important: The value specified in FS.TYPE is not sent to the device until FS.SET is set.

Specifications

Syntax	Data Type	Range	Access
S<slot>:FS.TYPE	**Byte**, Word, Short, DWord, Long	N/A	Write Only

Fail Safe Pattern Addressing

Using a fail safe pattern, users can specify which points are set and which ones are cleared when in fail safe mode.

Values

True = turn point on
False = turn point off

Requirements

FS.TYPE must be set to 2 for device to fail safe to a pattern.

Note: If using Byte, Word, Short, DWord or Long data types, the bit pattern determines which points are turned on and which ones are turned off.

Important: The pattern specified in FS.PATTERN is not sent to the device until FS.SET is set.

Specifications

Syntax	Data Type	Range	Access
S<slot>FS<point>.PATTERN	**Boolean**	0-3	Write Only
S<slot>:FS<offset>.PATTERN*	Byte**	0	Write Only
S<slot>:FS<offset>.PATTERN*	Word, Short**	0	Write Only
S<slot>:FS<offset>.PATTERN*	DWord, Long**	0	Write Only

* Multiple points can be referenced through the use of this optional syntax. For more information, refer to Digital I/O Bit Mapping.
** Since only the lower nibble of the least significant byte is being used, any value entered above 15 will be cropped.

Examples

Address	Value	Description
S1:FS0.PATTERN	1	Slot 1, turn point 0 on in fail safe mode.*
S1:FS0.PATTERN (as Byte)	15	Slot 1, turn points 0-3 on in fail safe mode.*
S1:FS0.PATTERN (as Word)	0	Slot 1, turn points 0-3 off in fail safe mode.*
S1:FS0.PATTERN (as DWORD)	15	Slot 1, turn points 0-3 on in fail safe mode.*

* See Notes and Requirements above.

Fail Safe Time Addressing

After communication between the host and the RTU has been lost, a time delay may be introduced before fail safe mode is engaged for the output module. This time delay can be entered by referencing FS.TIME and is specified in tenths of seconds.

Requirements

None

Important: The value specified in FS.TYPE is not sent to the device until FS.SET is set.

Specifications

Syntax	Data Type	Range	Access
S<slot>:FS.TIME	**Byte**, Word, Short, DWord, Long	N/A	Write Only

Examples

Address	Value	Description
S1:FS.TIME	255	Slot 1, fail safe time delay = 25.5 seconds.
S1:FS.TIME	0	Slot 1, no delay.
S1:FS.TIME	1	Slot 1, fail safe time delay = .1 seconds.

Fail Safe Set Addressing

For any of the fail safe configuration properties (type, time and pattern) to be sent to the device, FS.SET must be set. Immediately after the properties are sent, FS.SET will be reset.

Values

True = Send fail safe configurations to device
False = No action

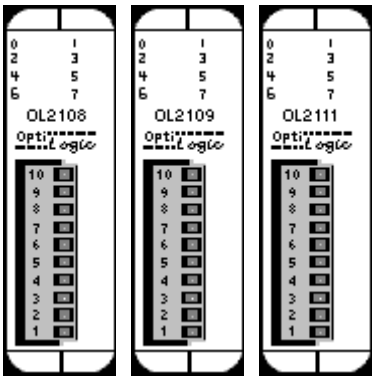
Requirements

None

Specifications

Syntax	Data Type	Range	Access
S<slot>:FS.SET	**Boolean**	N/A	Write Only

8 Digital Output Module



Digital outputs are used to turn loads on and off. Loads may be lights, motors, solenoids or any type of On/Off device.

Digital outputs in the OptiLogic series come in three types: relay, transistor and solid state relay. Each type has applications it is best suited for. For more information, refer to the module manual.

If there is a loss of communication with the host, the output module will enter a fail safe state. There are three types of states: fail safe to all outputs off, fail safe to a pattern and fail safe to last state. The time in which a fail safe state is entered after loss of communication can be delayed up to 25.5 seconds. If certain points need to be treated differently in fail safe mode, a pattern can be used to signify which points are turned on and which ones are turned off in fail safe mode.

Subtypes

OL2108, OL2109, OL2111

Address Types

8 Digital Output
Fail Safe Type
Fail Safe Pattern
Fail Safe Time
Fail Safe Set

8 Digital Output Addressing Specifications

Syntax	Data Type	Range	Access
S<slot>:DO<point>	**Boolean**	0-7	Read/Write
S<slot>:DO<offset>*	Byte	0	Read/Write

Syntax	Data Type	Range	Access
S<slot>:DO<offset>*	Word, Short	0	Read/Write
S<slot>:DO<offset>*	DWord, Long	0	Read/Write

* Multiple points can be referenced through the use of this optional syntax. For more information, refer to **Digital I/O Bit Mapping**.

Examples

Address	Value	Description
S1:DO0	1	Slot 1, point 0 (turn point 0 on).
S1:DO0 (as Byte)	255	Slot 1, byte 0 (turn points 0-7 on).
S1:DO0 (as Word)	0	Slot 1, word 0 (turn points 0-7 off).
S1:DO0 (as DWORD)	255	Slot 1, DWord 0 (turn points 0-7 on).

Fail Safe Type Addressing

There are three types of fail safe states: fail safe to all outputs off, fail safe to a pattern and fail safe to last state. These types may be specified by referencing FS.TYPE.

Values

1 = Fail safe to all outputs off
2 = Fail safe to the pattern contained in FS<point>.PATTERN
3 = Fail safe to last state

Requirements

None

Important: The value specified in FS.TYPE is not sent to the device until FS.SET is set.

Specifications

Syntax	Data Type	Range	Access
S<slot>:FS.TYPE	**Byte**, Word, Short, DWord, Long	N/A	Write Only

Fail Safe Pattern Addressing

Using a fail safe pattern, users can specify which points are set and which ones are cleared when in fail safe mode.

Values

True = turn point on
False = turn point off

Requirements

FS.TYPE must be set to 2 for device to fail safe to a pattern.

Note: If using Byte, Word, Short, DWord or Long data types, the bit pattern determines which points are turned on and which ones are turned off.

Important: The pattern specified in FS.PATTERN is not sent to the device until FS.SET is set.

Specifications

Syntax	Data Type	Range	Access
S<slot>FS<point>.PATTERN	**Boolean**	0-7	Write Only
S<slot>:FS<offset>.PATTERN*	Byte	0	Write Only
S<slot>:FS<offset>.PATTERN*	Word, Short	0	Write Only
S<slot>:FS<offset>.PATTERN*	DWord, Long	0	Write Only

* Multiple points can be referenced through the use of this optional syntax. For more information, refer to **Digital I/O Bit Mapping**.

Examples

Address	Value	Description
S1:FS0.PATTERN	1	Slot 1, turn point 0 on in fail safe mode.*
S1:FS0.PATTERN (as Byte)	255	Slot 1, turn points 0-7 on in fail safe mode.*
S1:FS0.PATTERN (as Word)	0	Slot 1, turn point 0-7 off in fail safe mode.*
S1:FS0.PATTERN (as DWORD)	255	Slot 1, turn points 0-7 on in fail safe mode.*

* See Notes and Requirements above.

Fail Safe Time Addressing

After communication between the host and the RTU has been lost, a time delay may be introduced before fail safe mode is engaged for the output module. This time delay can be entered by referencing FS.TIME and is specified in tenths of seconds.

Requirements

None

Important: The value specified in FS.TYPE is not sent to the device until FS.SET is set.

Specifications

Syntax	Data Type	Range	Access
S<slot>:FS.TIME	**Byte**, Word, Short, DWord, Long	N/A	Write Only

Examples

Address	Value	Description
S1:FS.TIME	255	Slot 1, fail safe time delay = 25.5 seconds.
S1:FS.TIME	0	Slot 1, no delay.
S1:FS.TIME	1	Slot 1, fail safe time delay = .1 seconds.

Fail Safe Set Addressing

For any of the fail safe configuration properties (type, time and pattern) to be sent to the device, FS.SET must be set. Immediately after the properties are sent, FS.SET will be reset.

Values

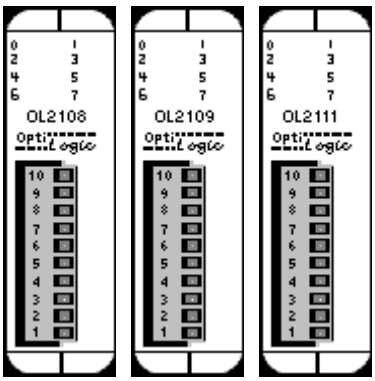
True = Send fail safe configurations to device
False = No action

Requirements
None

Specifications

Syntax	Data Type	Range	Access
S<slot>:FS.SET	**Boolean**	N/A	Write Only

8 Digital Output Module



Digital outputs are used to turn loads on and off. Loads may be lights, motors, solenoids or any type of On/Off device.

Digital outputs in the OptiLogic series come in three types: relay, transistor and solid state relay. Each type has applications it is best suited for. For more information, refer to the module manual.

If there is a loss of communication with the host, the output module will enter a fail safe state. There are three types of states: fail safe to all outputs off, fail safe to a pattern and fail safe to last state. The time in which a fail safe state is entered after loss of communication can be delayed up to 25.5 seconds. If certain points need to be treated differently in fail safe mode, a pattern can be used to signify which points are turned on and which ones are turned off in fail safe mode.

Subtypes
OL2108, OL2109, OL2111

Address Types
8 Digital Output
Fail Safe Type
Fail Safe Pattern
Fail Safe Time
Fail Safe Set

8 Digital Output Addressing Specifications

Syntax	Data Type	Range	Access
S<slot>:DO<point>	**Boolean**	0-7	Read/Write
S<slot>:DO<offset>*	Byte	0	Read/Write
S<slot>:DO<offset>*	Word, Short	0	Read/Write
S<slot>:DO<offset>*	DWord, Long	0	Read/Write

* Multiple points can be referenced through the use of this optional syntax. For more information, refer to **Digital I/O Bit Mapping**.

Examples

Address	Value	Description
S1:DO0	1	Slot 1, point 0 (turn point 0 on).
S1:DO0 (as Byte)	255	Slot 1, byte 0 (turn points 0-7 on).
S1:DO0 (as Word)	0	Slot 1, word 0 (turn points 0-7 off).
S1:DO0 (as DWORD)	255	Slot 1, DWord 0 (turn points 0-7 on).

Fail Safe Type Addressing

There are three types of fail safe states: fail safe to all outputs off, fail safe to a pattern and fail safe to last state. These types may be specified by referencing FS.TYPE.

Values

1 = Fail safe to all outputs off
2 = Fail safe to the pattern contained in FS<point>.PATTERN
3 = Fail safe to last state

Requirements

None

Important: The value specified in FS.TYPE is not sent to the device until FS.SET is set.

Specifications

Syntax	Data Type	Range	Access
S<slot>:FS.TYPE	**Byte**, Word, Short, DWord, Long	N/A	Write Only

Fail Safe Pattern Addressing

Using a fail safe pattern, users can specify which points are set and which ones are cleared when in fail safe mode.

Values

True = turn point on
False = turn point off

Requirements

FS.TYPE must be set to 2 for device to fail safe to a pattern.

Note: If using Byte, Word, Short, DWord or Long data types, the bit pattern determines which points are turned on and which ones are turned off.

Important: The pattern specified in FS.PATTERN is not sent to the device until FS.SET is set.

Specifications

Syntax	Data Type	Range	Access
S<slot>FS<point>.PATTERN	**Boolean**	0-7	Write Only
S<slot>:FS<offset>.PATTERN*	Byte	0	Write Only

Syntax	Data Type	Range	Access
S<slot>:FS<offset>.PATTERN*	Word, Short	0	Write Only
S<slot>:FS<offset>.PATTERN*	DWord, Long	0	Write Only

* Multiple points can be referenced through the use of this optional syntax. For more information, refer to **Digital I/O Bit Mapping**.

Examples

Address	Value	Description
S1:FS0.PATTERN	1	Slot 1, turn point 0 on in fail safe mode.*
S1:FS0.PATTERN (as Byte)	255	Slot 1, turn points 0-7 on in fail safe mode.*
S1:FS0.PATTERN (as Word)	0	Slot 1, turn point 0-7 off in fail safe mode.*
S1:FS0.PATTERN (as DWORD)	255	Slot 1, turn points 0-7 on in fail safe mode.*

* See Notes and Requirements above.

Fail Safe Time Addressing

After communication between the host and the RTU has been lost, a time delay may be introduced before fail safe mode is engaged for the output module. This time delay can be entered by referencing FS.TIME and is specified in tenths of seconds.

Requirements

None

Important: The value specified in FS.TYPE is not sent to the device until FS.SET is set.

Specifications

Syntax	Data Type	Range	Access
S<slot>:FS.TIME	**Byte**, Word, Short, DWord, Long	N/A	Write Only

Examples

Address	Value	Description
S1:FS.TIME	255	Slot 1, fail safe time delay = 25.5 seconds.
S1:FS.TIME	0	Slot 1, no delay.
S1:FS.TIME	1	Slot 1, fail safe time delay = .1 seconds.

Fail Safe Set Addressing

For any of the fail safe configuration properties (type, time and pattern) to be sent to the device, FS.SET must be set. Immediately after the properties are sent, FS.SET will be reset.

Values

True = Send fail safe configurations to device
False = No action

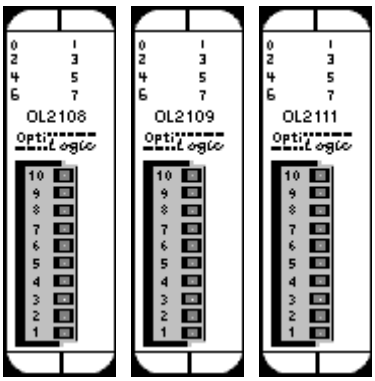
Requirements

None

Specifications

Syntax	Data Type	Range	Access
S<slot>:FS.SET	**Boolean**	N/A	Write Only

8 Digital Output Module



Digital outputs are used to turn loads on and off. Loads may be lights, motors, solenoids or any type of On/Off device.

Digital outputs in the OptiLogic series come in three types: relay, transistor and solid state relay. Each type has applications it is best suited for. For more information, refer to the module manual.

If there is a loss of communication with the host, the output module will enter a fail safe state. There are three types of states: fail safe to all outputs off, fail safe to a pattern and fail safe to last state. The time in which a fail safe state is entered after loss of communication can be delayed up to 25.5 seconds. If certain points need to be treated differently in fail safe mode, a pattern can be used to signify which points are turned on and which ones are turned off in fail safe mode.

Subtypes

OL2108, OL2109, OL2111

Address Types

8 Digital Output
Fail Safe Type
Fail Safe Pattern
Fail Safe Time
Fail Safe Set

8 Digital Output Addressing Specifications

Syntax	Data Type	Range	Access
S<slot>:DO<point>	**Boolean**	0-7	Read/Write
S<slot>:DO<offset>*	Byte	0	Read/Write
S<slot>:DO<offset>*	Word, Short	0	Read/Write
S<slot>:DO<offset>*	DWord, Long	0	Read/Write

* Multiple points can be referenced through the use of this optional syntax. For more information, refer to **Digital I/O Bit Mapping**.

Examples

Address	Value	Description
S1:DO0	1	Slot 1, point 0 (turn point 0 on).

S1:DO0 (as Byte)	255	Slot 1, byte 0 (turn points 0-7 on).
S1:DO0 (as Word)	0	Slot 1, word 0 (turn points 0-7 off).
S1:DO0 (as DWORD)	255	Slot 1, DWord 0 (turn points 0-7 on).

Fail Safe Type Addressing

There are three types of fail safe states: fail safe to all outputs off, fail safe to a pattern and fail safe to last state. These types may be specified by referencing FS.TYPE.

Values

1 = Fail safe to all outputs off
2 = Fail safe to the pattern contained in FS<point>.PATTERN
3 = Fail safe to last state

Requirements

None

Important: The value specified in FS.TYPE is not sent to the device until FS.SET is set.

Specifications

Syntax	Data Type	Range	Access
S<slot>:FS.TYPE	**Byte**, Word, Short, DWord, Long	N/A	Write Only

Fail Safe Pattern Addressing

Using a fail safe pattern, users can specify which points are set and which ones are cleared when in fail safe mode.

Values

True = turn point on
False = turn point off

Requirements

FS.TYPE must be set to 2 for device to fail safe to a pattern.

Note: If using Byte, Word, Short, DWord or Long data types, the bit pattern determines which points are turned on and which ones are turned off.

Important: The pattern specified in FS.PATTERN is not sent to the device until FS.SET is set.

Specifications

Syntax	Data Type	Range	Access
S<slot>FS<point>.PATTERN	**Boolean**	0-7	Write Only
S<slot>:FS<offset>.PATTERN*	Byte	0	Write Only
S<slot>:FS<offset>.PATTERN*	Word, Short	0	Write Only
S<slot>:FS<offset>.PATTERN*	DWord, Long	0	Write Only

* Multiple points can be referenced through the use of this optional syntax. For more information, refer to **Digital I/O Bit Mapping**.

Examples

Address	Value	Description
S1:FS0.PATTERN	1	Slot 1, turn point 0 on in fail safe mode.*
S1:FS0.PATTERN (as Byte)	255	Slot 1, turn points 0-7 on in fail safe mode.*
S1:FS0.PATTERN (as Word)	0	Slot 1, turn point 0-7 off in fail safe mode.*
S1:FS0.PATTERN (as DWORD)	255	Slot 1, turn points 0-7 on in fail safe mode.*

* See Notes and Requirements above.

Fail Safe Time Addressing

After communication between the host and the RTU has been lost, a time delay may be introduced before fail safe mode is engaged for the output module. This time delay can be entered by referencing FS.TIME and is specified in tenths of seconds.

Requirements

None

Important: The value specified in FS.TYPE is not sent to the device until FS.SET is set.

Specifications

Syntax	Data Type	Range	Access
S<slot>:FS.TIME	**Byte**, Word, Short, DWord, Long	N/A	Write Only

Examples

Address	Value	Description
S1:FS.TIME	255	Slot 1, fail safe time delay = 25.5 seconds.
S1:FS.TIME	0	Slot 1, no delay.
S1:FS.TIME	1	Slot 1, fail safe time delay = .1 seconds.

Fail Safe Set Addressing

For any of the fail safe configuration properties (type, time and pattern) to be sent to the device, FS.SET must be set. Immediately after the properties are sent, FS.SET will be reset.

Values

True = Send fail safe configurations to device
False = No action

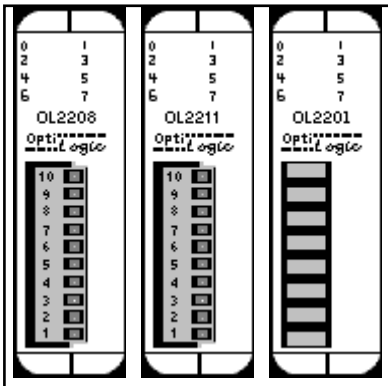
Requirements

None

Specifications

Syntax	Data Type	Range	Access
S<slot>:FS.SET	**Boolean**	N/A	Write Only

8 Digital Input Module



Digital input modules are used to monitor the state of discrete field devices. Typical digital inputs are connected to switches, buttons, digital outputs from other equipment, discrete level sensors, thermostats and other On/Off sensing devices.

Digital status is sensed by a controller such as an OptiLogic system, by passing current through an input sensor. When the current is on, the input state is active. When there is no current, the state is inactive.

Subtypes

OL2208, OL2211, OL2201

8 Digital Input Addressing Specifications

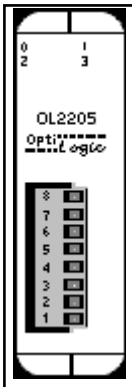
Syntax	Data Type	Range	Access
S<slot>:DI<point>	**Boolean**	0-7	Read Only
S<slot>:DI<offset>*	Byte	0	Read Only
S<slot>:DI<offset>*	Word, Short	0	Read Only
S<slot>:DI<offset>*	DWord, Long	0	Read Only

* Multiple points can be referenced through the use of this optional syntax. For more information, refer to **Digital I/O Bit Mapping**.

Examples

Address	Value	Description
S1:DI0	1	Slot 1, point 0 is on.
S1:DI0 (as Byte)	255	Slot 1, byte 0 (points 0-7 are on).
S1:DI0 (as Word)	0	Slot 1, word 0 (points 0-7 are off).
S1:DI0 (as DWORD)	255	Slot 1, DWord 0 (points 0-7 are on).

4 Digital Input Module



Digital input modules are used to monitor the state of discrete field devices. Typical digital inputs are connected to switches, buttons, digital outputs from other equipment, discrete level sensors, thermostats and other On/Off sensing devices.

Digital status is sensed by a controller such as an OptiLogic system, by passing current through an input sensor. When the current is on, the input state is active. When there is no current, the state is inactive.

Subtypes

OL2205

4 Digital Input Addressing Specifications

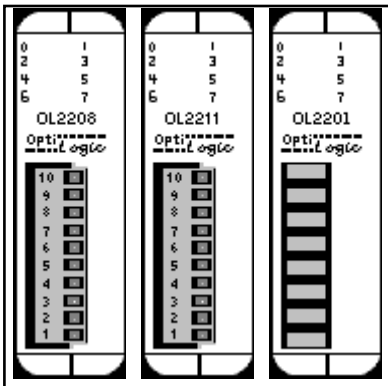
Syntax	Data Type	Range	Access
S<slot>:DI<point>	**Boolean**	0-3	Read Only
S<slot>:DI<offset>*	Byte	0	Read Only
S<slot>:DI<offset>*	Word, Short	0	Read Only
S<slot>:DI<offset>*	DWord, Long	0	Read Only

* Multiple points can be referenced through the use of this optional syntax. For more information, refer to **Digital I/O Bit Mapping**.

Examples

Address	Value	Description
S1:DI0	1	Slot 1, point 0 is on.
S1:DI0 (as Byte)	15	Slot 1, byte 0 (points 0-3 are on).
S1:DI0 (as Word)	0	Slot 1, word 0 (points 0-3 are off).
S1:DI0 (as DWORD)	15	Slot 1, DWord 0 (points 0-3 are on).

8 Digital Input Module



Digital input modules are used to monitor the state of discrete field devices. Typical digital inputs are connected to switches, buttons, digital outputs from other equipment, discrete level sensors, thermostats and other On/Off sensing devices.

Digital status is sensed by a controller such as an OptiLogic system, by passing current through an input sensor. When the current is on, the input state is active. When there is no current, the state is inactive.

Subtypes

OL2208, OL2211, OL2201

8 Digital Input Addressing Specifications

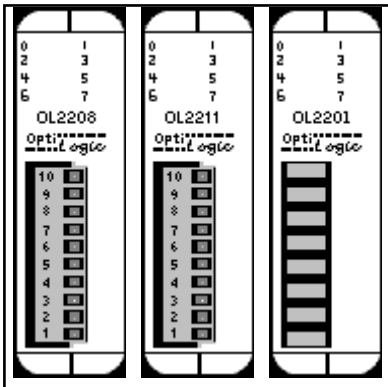
Syntax	Data Type	Range	Access
S<slot>:DI<point>	**Boolean**	0-7	Read Only
S<slot>:DI<offset>*	Byte	0	Read Only
S<slot>:DI<offset>*	Word, Short	0	Read Only
S<slot>:DI<offset>*	DWord, Long	0	Read Only

* Multiple points can be referenced through the use of this optional syntax. For more information, refer to **Digital I/O Bit Mapping**.

Examples

Address	Value	Description
S1:DI0	1	Slot 1, point 0 is on.
S1:DI0 (as Byte)	255	Slot 1, byte 0 (points 0-7 are on).
S1:DI0 (as Word)	0	Slot 1, word 0 (points 0-7 are off).
S1:DI0 (as DWORD)	255	Slot 1, DWord 0 (points 0-7 are on).

8 Digital Input Module



Digital input modules are used to monitor the state of discrete field devices. Typical digital inputs are connected to switches, buttons, digital outputs from other equipment, discrete level sensors, thermostats and other On/Off sensing devices.

Digital status is sensed by a controller such as an OptiLogic system, by passing current through an input sensor. When the current is on, the input state is active. When there is no current, the state is inactive.

Subtypes

OL2208, OL2211, OL2201

8 Digital Input Addressing Specifications

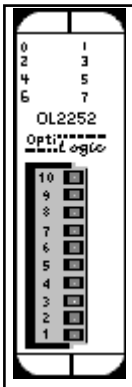
Syntax	Data Type	Range	Access
S<slot>:DI<point>	**Boolean**	0-7	Read Only
S<slot>:DI<offset>*	Byte	0	Read Only
S<slot>:DI<offset>*	Word, Short	0	Read Only
S<slot>:DI<offset>*	DWord, Long	0	Read Only

* Multiple points can be referenced through the use of this optional syntax. For more information, refer to **Digital I/O Bit Mapping**.

Examples

Address	Value	Description
S1:DI0	1	Slot 1, point 0 is on.
S1:DI0 (as Byte)	255	Slot 1, byte 0 (points 0-7 are on).
S1:DI0 (as Word)	0	Slot 1, word 0 (points 0-7 are off).
S1:DI0 (as DWORD)	255	Slot 1, DWord 0 (points 0-7 are on).

2-Channel High-Speed Counter Module



The OL2252 Dual High-Speed Pulse Counter module has two 0-15 kHz pulse counter inputs. Each input is independent of the other. There are a number of configuration options available. See OL2252 manual for details. Of the 10 inputs, 6 can be configured for general-purpose input and is referred to in this driver as CDI (counter digital input). Both channels have a software reset and an optional hardware reset as well as a software enable and optional hardware enable.

Subtypes

OL2252

Address Types

Channel Pulse Count
Channel Enable (Software)
Channel Reset (Software)
Channel Enable (Hardware)
Channel Reset (Hardware)
Channel Debounce Count
8 Digital Input

Channel Pulse Count Addressing

The pulse count for each channel can be referenced using address type C<channel>.COUNT.

Requirements

Channel counter must be enabled either by software or hardware for counter to pulse count.

Specifications

Syntax	Data Type	Range	Access
S<slot>:C<channel>.COUNT	**DWord**, Long	1-2	Read Only

Examples

Address	Value	Description

S1:C1.COUNT	0	Slot 1, channel 1 pulse count is 0.*
S1:C2.COUNT	1000	Slot 1, channel 2 pulse count is 1000.*

* See Requirements above.

Channel Enable (Software) Addressing

Channels can be enabled/disabled through software by referencing address type C<channel>.EN.

Values

True = channel enabled
False = channel disabled*

* Provided channel is not currently hardware enabled.

Requirements

None

Specifications

Address	Value	Description
S1:C1.COUNT	0	Slot 1, channel 1 pulse count is 0.*
S1:C2.COUNT	1000	Slot 1, channel 2 pulse count is 1000.*

Examples

Address	Value	Description
S1:C1.EN	0	Slot 1, ch1 not software enabled.
S1:C2.EN	1	Slot 1, ch2 software enabled.

Channel Reset (Software) Addressing

A channel's pulse count can be reset through software by referencing address type C<channel>.RES.

Values

True = channel pulse count reset
False = channel pulse counting*

* Provided channel is not currently in a hardware reset.

Requirements

None

Specifications

Syntax	Data Type	Range	Access
S<slot>:C<channel>.RES	**Boolean**	1-2	Write Only

Examples

Address	Value	Description

S1:C1.RES	0	Slot 1, ch1 not software reset.
S1:C2.RES	1	Slot 1, ch2 software reset.

Channel Enable (Hardware) Addressing

Channels can be enabled by an external enable signal. Enable/disable this capability by referencing address type CCFG<channel>.HEN.

Values

True = allow hardware enable of channel
False = don't allow hardware enable of channel

Requirements

None

Specifications

Syntax	Data Type	Range	Access
S<slot>:CCFG<channel>.HEN	**Boolean**	1-2	Write Only

Examples

Address	Value	Description
S1:CCFG1.HEN	0	Slot 1, ch1 hardware enable not allowed.
S1:CCFG2.HEN	1	Slot 1, ch2 hardware enabled allowed.

Channel Reset (Hardware) Addressing

A channel's pulse count can be reset by an external reset signal. Enable/disable this capability by referencing address type CCFG<channel>.HRES.

Values

True = allow hardware reset of pulse count
False = don't allow hardware reset of pulse count

Requirements

None

Specifications

Syntax	Data Type	Range	Access
S<slot>:CCFG<channel>.HRES	**Boolean**	1-2	Write Only

Examples

Address	Value	Description
S1:CCFG1.HRES	0	Slot 1, ch1 hardware reset not allowed.
S1:CCFG2.HRES	1	Slot 1, ch2 hardware reset allowed.

Channel Debounce Count Addressing

CCFG<channel>.DBNC establishes the maximum pulse frequency that a channel will count.

Values

2 = 15 kHz
4 = 10 kHz
8 = 5 kHz
16 = 2.5 kHz
40 = 1 kHz

Note: The default value is 15 kHz.

Requirements

None

Specifications

Syntax	Data Type	Range	Access
S<slot>:CCFG<channel>.DBNC	**Byte**, Word, Short, DWord, Long	1-2	Write Only

Examples

Address	Value	Description
S1:CCFG1.DBNC	40	Slot 1, ch1 max pulse freq set to 1 kHz.
S1:CCFG2.DBNC	2	Slot 1, ch2 max pulse freq set to 15 kHz.

8 Digital Input Addressing

The counter's general-purpose inputs are referenced using address type CDI.

Requirements

None

Specifications

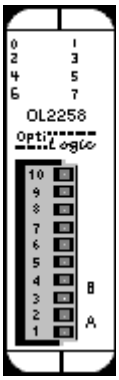
Syntax	Data Type	Range	Access
S<slot>:CDI<point>	**Boolean**	0-7	Read Only
S<slot>:CDI<offset>*	Byte, Word, Short, DWord, Long	0	Read Only

* Multiple points can be referenced through the use of this optional syntax. For more information, refer to **Digital I/O Bit Mapping**.

Examples

Address	Value	Description
S1:CDI0	1	Slot 1, point 0 (terminal 10) is on.
S1:CDI0 (as Byte)	0	Slot 1, byte 0 (points 0-7 are all off).

High-Speed Counter Module



The OL2258 High-Speed Pulse Counter module provides for direct pulse counting for a variety of high-speed pulse interface applications. Typical applications include motion control, metering and velocity measurement.

The OL2258 can be configured to operate in one of three modes:

1. Pulse & Direction (up to 80 kHz input).
2. Up/Down Count (up to 80 kHz input).
3. Quadrature (up to 160 kHz input).

Besides the counter's current count value, the pulse count in the most recent frequency period (user configurable) is also accessible. The OL2258 also contains 2 digital outputs, triggered on when the pulse count reaches the output's minimum range and triggered off when it reaches the output's maximum range.

Subtypes
OL2258

Address Types
Channel Pulse Count
Channel Frequency Data
Channel Status: A
Channel Status: B
Channel Status: Z
Channel Status: LS
Channel Configuration: Count Type
Channel Configuration: Frequency Period
Channel Configuration: Preset
Channel Configuration: Force Preset
Channel Configuration: Hold Count
Channel Configuration: Z Preset Enable
Channel Configuration: LS Preset Enable
Triggered Digital Outputs
Triggered Digital Outputs: Minimum Range
Triggered Digital Outputs: Maximum Range
Triggered Digital Outputs: Range Enable

Channel Pulse Count
The pulse count for each channel can be referenced using address type C<channel>.COUNT.

Requirements
None

Specifications

Syntax	Data Type	Range	Access
S<slot>:C<channel>.COUNT	**DWord**, Long	1	Read Only

Examples

Address	Value	Description
S1:C1.COUNT	0	Slot 1, channel 1 pulse count is 0.
S1:C1.COUNT	1000	Slot 1, channel 1 pulse count is 1000.

Channel Frequency Data

A pulse count over a preset period of time (C<channel>.COUNT @ (start + period)-C<channel>.COUNT @ start) can be accessed through C<channel>.FREQDATA.

Requirements

C<channel>.FREQPER determines the period and should be set before accessing C<channel>.FREQDATA.

Specifications

Syntax	Data Type	Range	Access
S<slot>:C<channel>.FREQDATA	**Word**, Short, DWord, Long	1	Read Only

Examples

Address	Value	Description
S1:C1.FREQDATA	60000	Slot 1, ch1, if FREQPER = 1 sec, pulse is 60kHz.
S1:C1.FREQDATA	16000	Slot 1, ch1, if FREQPER = 200ms, pulse is 80kHz.

Channel Status: A

Pulse input A can be referenced through CSTS<channel>.A. Count type defines the meaning of this input as follows:

Count Type	Input Definition
Pulse & Direction	Pulse Input
Up/Down Count	Up Pulse Input
Quadrature Encoder Input	Pulse Input

For more information, refer to OptiLogic I/O Modules Manual.

Values

True = Pulse level high
False = Pulse level low

Requirements

CCFG<channel>.COUNTTYPE determines the meaning of this input and should be set before accessing CSTS<channel>.A

Specifications

Syntax	Data Type	Range	Access
S<slot>:CSTS<channel>.A	**Boolean**	1	Read Only

Channel Status: B

Pulse input B can be referenced through CSTS<channel>.B. Count type defines the meaning of this input as follows:

Count Type	Input Definition
Pulse & Direction	Direction Input
Up/Down Count	Down Pulse Input
Quadrature Encoder Input	Pulse Input

For more information, refer to OptiLogic I/O Modules Manual.

Values

True = Pulse level high
False = Pulse level low

Requirements

CCFG<channel>.COUNTTYPE determines the meaning of this input and should be set before accessing CSTS<channel>.B

Specifications

Syntax	Data Type	Range	Access
S<slot>:CSTS<channel>.B	**Boolean**	1	Read Only

Channel Status: Z

Optional input Z provides a means of automatically resetting the count value to a user-defined preset value. Reference Z through CSTS<channel>.Z.

Values

True = If ZPRESETEN set, send PRESET to device. Otherwise, take no action.
False = No action

Requirements

CCFG<channel>.ZPRESETEN must be set in order to enable use of Z in presetting counter

Specifications

Syntax	Data Type	Range	Access
S<slot>:CSTS<channel>.Z	**Boolean**	1	Read Only

Channel Status: LS

Optional limit switch input LS provides a means of automatically resetting the count value to a user-defined preset value. Reference LS through CSTS<channel>.LS.

Values

True = If LSPRESETEN set, send PRESET to device. Otherwise, take no action.
False = No action

Requirements

CCFG<channel>.LSPRESETEN must be set in order to enable use of LS in presetting counter

Specifications

Syntax	Data Type	Range	Access
S<slot>:CSTS<channel>.LS	**Boolean**	1	Read Only

Channel Configuration: Count Type (Mode)

The mode of counter operation is configured via CCFG<channel>.COUNTTYPE.

Values

0 = Pulse & Direction
1 = Up/Down Count
2 = Quadrature

Requirements

None

Specifications

Syntax	Data Type	Range	Access
S<slot>:CCFG<channel>.COUNTTYPE	**Byte**, Word, Short, DWord, Long	1	Read/Write

For more information, refer to OptiLogic I/O Modules Manual.

Examples

Address	Value	Description
S1:CCFG1.COUNTTYPE	1	Slot 1, ch1 count mode set for pulse & dir.
S1:CCFG1.COUNTTYPE	3	Slot 1, ch1 count mode set for quadrature.

Channel Configuration: Frequency Period

Recall that a pulse count over a preset period of time can be accessed through C<channel>.FREQDATA. The period in the FREQDATA formula C<channel>.COUNT @ (start + period)-C<channel>.COUNT @ start is configured through CCFG<channel>.FREQPER.

Values

0 = 1 second count
1 = 200 msec count

Requirements

None

Specifications

Syntax	Data Type	Range	Access
S<slot>:CCFG<channel>.FREQPER	**Byte**, Word, Short, DWord, Long	1	Read/Write

Examples

Address	Value	Description
S1:CCFG1.FREQPER	0	Slot 1, ch1, FREQDATA reflects count in 1 second time window.
S1:CCFG1.FREQPER	1	Slot 1, ch1, FREQDATA reflects count in 200 ms time window.

Channel Configuration: Preset

Establish a preset count value by setting CCFG<channel>.PRESET.

Requirements

For preset to take effect, at least one of the following must be true:
CCFG<channel>.FORCEPRESET be set to TRUE
CCFG<channel>.ZPRESETEN be set to TRUE and Z input be TRUE
CCFG<channel>.LSPRESETEN be set to TRUE and LS input be TRUE

Specifications

Syntax	Data Type	Range	Access
S<slot>:CCFG<channel>.PRESET	**DWord**, Long	1	Write Only

Examples

Address	Value	Description
S1:CCFG1.PRESET	1000	Slot 1, ch1, set preset to 1000.
S1:CCFG1.PRESET	0	Slot 1, ch1, set preset to 0 (default).

Channel Configuration: Force Preset

Set the counter's count value to the preset given in CCFG<channel>PRESET.

Values

True = Send PRESET to device
False = No action

Requirements

None

Specifications

Syntax	Data Type	Range	Access
S<slot>:CCFG<channel>.FORCEPRESET	**Boolean**	1	Write Only

Channel Configuration: Hold Count

Hold the current count value by referencing CCFG<channel>.HOLDCOUNT.

Values

True = Hold count value
False = No action

Requirements

None.

Specifications

Syntax	Data Type	Range	Access
S<slot>:CCFG<channel>.HOLDCOUNT	**Boolean**	1	Write Only

Channel Configuration: Z Preset Enable

Enables Z input to control the presetting of the counter.

Values

True = Z input state configured to preset counter when applicable
False = Z input state has no control over presetting counter

Requirements

None.

Specifications

Syntax	Data Type	Range	Access
S<slot>:CCFG<channel>.ZPRESETEN	**Boolean**	1	Write Only

Channel Configuration: LS Preset Enable

Enables LS input to control the presetting of the counter.

Values

True = LS input state configured to preset counter when applicable
False = LS input state has no control over presetting counter

Requirements

None.

Specifications

Syntax	Data Type	Range	Access
S<slot>:CCFG<channel>.LSPRESETEN	**Boolean**	1	Write Only

Triggered Digital Outputs

There are 2 digital outputs that can be configured to turn on when the counter count is within a specific range. The status of whether the digital output is on or off can be accessed through CTRIGDO<output>.

Requirements

CTRIGDO outputs must have a range specified and enabled, otherwise outputs will remain off.

Specifications

Syntax	Data Type	Range	Access
S<slot>:CTRIGDO<output>	**Boolean**	1-2	Read Only

Examples

Address	Value	Description
S1:CTRIGDO1	1	Slot 1, output 1 is on.
S1:CTRIGDO2	0	Slot 1, output 2 is off.

Triggered Digital Outputs: Minimum Range

Set the range's minimum value to be the counter's count value at which users desire the CTRIGDO<output> to turn from off to on. This minimum range value is referenced as CTRIGDO<output>.MINRANGE.

Requirements

CTRIGDO range usage must be enabled.

Specifications

Syntax	Data Type	Range	Access
S<slot>:CTRIGDO<output>.MINRANGE	**DWord**, Long	1-2	Write Only

Examples

Address	Value	Description
S1:CTRIGDO1.MINRANGE	100	Slot 1, output 1, turn on when count = 100.
S1:CTRIGDO2.MINRANGE	65536	Slot 1, output 2, turn on when count = 65536.

Triggered Digital Outputs: Maximum Range

Set the range's maximum value to be the counter's count value at which users desire the CTRIGDO<output> to turn from on to off. This maximum range value is referenced as CTRIGDO<output>.MAXRANGE.

Requirements

CTRIGDO usage range must be enabled.

Specifications

Syntax	Data Type	Range	Access
S<slot>:CTRIGDO<output>.MAXRANGE	**DWord**, Long	1-2	Write Only

Examples

Address	Value	Description
S1:CTRIGDO1.MAXRANGE	1000	Slot 1, output 1, turn off when count = 1000.
S1:CTRIGDO2.MAXRANGE	0	Slot 1, output 2, turn off when count = 0 (from rollover).

Triggered Digital Outputs: Range Enable

Enable the minimum and maximum range values for CTRIGDO (enable usage of a CTRIGDO output) by referencing CTRIGDO<output>.RANGEEN.

Requirements

None.

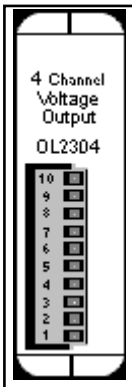
Specifications

Syntax	Data Type	Range	Access
S<slot>:CTRIGDO<output>.RANGEEN	**Boolean**	1-2	Write Only

Examples

Address	Value	Description
S1:CTRIGDO1.RANGEEN	1	Slot 1, output 1 enabled.
S1:CTRIGDO2.RANGEEN	0	Slot 1, output 2 disabled.

4-Channel Analog Output Module



Each channel of an analog input module uses a 12 bit D/A converter to produce an analog output signal.

Subtypes

OL2304

Address Types

4-Channel Analog Output

4-Channel Analog Output Range

4-Channel Analog Output Addressing Requirements

As a precaution, the analog output range should be set prior to writing to an analog output channel.

Specifications

Syntax	Data Type	Range	Access
S<slot>:AO<channel>	**Word**, Short, DWord, Long	1-4	Write Only

Examples

Address	Value	Description
S1:AO2	1024 *	Slot 1, channel 2.
S1:AO4	0 *	Slot 1, channel 4.

* The exact analog output value depends on voltage range, accuracy and so forth.

4-Channel Analog Output Range Addressing

Each channel must be configured for a specific analog output range. This is done by setting AO<channel>.RANGE.

Values
0 = 0-5 V
1 = 0-10 V
2 = +/- 5 V
3 = +/- 10 V

Requirements
None.

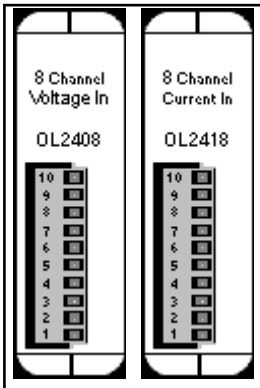
Specifications

Syntax	Data Type	Range	Access
S<slot>:AO<channel>.RANGE	**Word**, Short, DWord, Long	1-4	Write Only

Examples

Address	Value	Description
S1:AO2.RANGE	2	Slot 1, channel 2, set voltage range to +/- 5V.
S1:AO4.RANGE	0	Slot 1, channel 4, set voltage range to 0-5V.

8-Channel Analog Input Module



Analog inputs are used to monitor the value of continuously variable field measurements. Typical analog inputs are measurements of temperature, pressure, weight, liquid level, pH, flow rate and many other "real world" parameters.

Each channel of an analog input module takes an analog signal as input and uses a 12-bit A/D converter to put that analog value in digital form.

Subtypes
OL2408, OL2418

8-Channel Analog Input Addressing Specifications

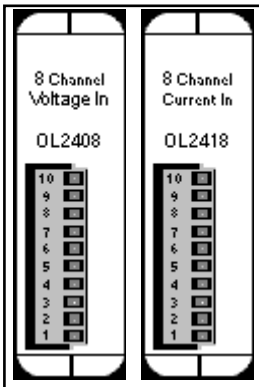
Syntax	Data Type	Range	Access
S<slot>:AI<channel>	**Word**, Short, DWord, Long	1-8	Read Only

Examples

Address	Value	Description
S1:AI2	*	Slot 1, channel 2.
S1:AI8	*	Slot 1, channel 8.

* Values depend on input level, voltage/current ranges, accuracy and so forth.

8-Channel Analog Input Module



Analog inputs are used to monitor the value of continuously variable field measurements. Typical analog inputs are measurements of temperature, pressure, weight, liquid level, pH, flow rate and many other "real world" parameters.

Each channel of an analog input module takes an analog signal as input and uses a 12-bit A/D converter to put that analog value in digital form.

Subtypes

OL2408, OL2418

8-Channel Analog Input Addressing Specifications

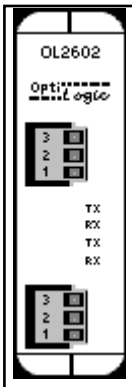
Syntax	Data Type	Range	Access
S<slot>:AI<channel>	**Word**, Short, DWord, Long	1-8	Read Only

Examples

Address	Value	Description
S1:AI2	*	Slot 1, channel 2.
S1:AI8	*	Slot 1, channel 8.

* Values depend on input level, voltage/current ranges, accuracy and so forth.

Dual RS232 Port Module



The OL2602 is a dual RS232 port serial module. Both the transmit buffer and receive buffer of the driver

are 48 bytes in size. Likewise, the corresponding tags can be a maximum of 48 bytes. Incoming bytes are appended to the receive buffer as long as they are received in proper time. This time period depends on the baud rate and is based on a 20-character delay using a 20 ms resolution.

1200 Baud => 160 ms

2400 Baud => 80 ms

4800 Baud => 40 ms

9600 Baud => 20 ms

19200 Baud => 20 ms

For a 4800 baud link, bytes would be appended to the receive buffer as long as they were received within 40 ms of the last byte sent. After 40 ms, any incoming bytes are treated as a new stream. The receive buffer would clear and only contain these new bytes.

If the receive buffer is full and additional bytes are received within the proper time frame, the buffer will reset with these additional bytes. The first 48 bytes will be lost.

Below is a list of possible configurations:

1. Baud rates: 1200, 2400, 4800, 9600 and 19200.

2. Data bits: 7 or 8

3. Parity: none, odd or even

4. Stop bits: 1, 1.5 or 2

If using the OL4058 RTU, a maximum of one OL2602 modules may be used and it must be placed in slot 0. For the OL4054 RTU, two OL2602 may be used and they must be in either slot 0 or 1.

Port 1 is the top terminal strip, Port 2 is the bottom terminal strip. Below is the pin assignments:

1: Signal ground

2: TX

3: RX

Subtypes

OL2602

Address Types

Serial Input: Data

Serial Input: Number of Received Bytes

Serial Input: Parity Error

Serial Output: Data

Serial Input: Data Addressing

To receive serial data, reference address type SI<port>.DATA.

Note: The default configuration parameters are 9600, n, 8 and 1.

Specifications

Syntax	Data Type	Range	Access
S<slot>:SI<port>.DATA [r][c]*	**Byte Array**, Char Array	1-2	Read Only
S<slot>:SI<port>.DATA	String	1-2	Read Only

* To access as an array, [row][column] form is required. For example, DATA [1][24] would display 24 ASCII bytes in array notation: [x1, x2, x3..x24])

Examples

Address	Value	Description
S0:SI1.DATA	"hello"	Slot 0, port 1input data viewed as a string.
S0:SI2.DATA [2] [2]	[105, 105] [105, 105]	Slot 0, port 2 input data in array form. In string form this would equate to "iiii".

Serial Input: Number of Received Bytes Addressing

The number of received serial bytes (number of bytes in SI<port>.DATA) can be accessed by referencing address type SI<port>.NUMBYTES. If bytes are received within the timeout period mentioned above and are therefore appended to the input DATA buffer, NUMBYTES will reflect the total number of bytes in the input DATA buffer and not the number of bytes received on an individual block read. NUMBYTES will reset upon receiving a new stream.

Requirements

None

Specifications

Syntax	Data Type	Range	Access
S<slot>:SI<port>.NUMBYTES	**Byte**, Word, Short, DWord, Long	1-2	Read Only

Examples

Address	Value	Description
S1:SI1.NUMBYTES	0	Slot 1, port 1 input, no bytes in input DATA buffer

Address	Value	Description
S1:SI2.NUMBYTES	5	Slot1, port2 input, 5 bytes in input DATA buffer

Serial Input: Parity Error Addressing

Reference SI<port>.PARITYERR to determine whether a parity error occurred on the last block read of the serial input port.

Values

True = Parity error occurred
False = No parity error occurred

Requirements

None

Specifications

Syntax	Data Type	Range	Access
S<slot>:SI<port>.PARITYERR	**Boolean**	1-2	Read Only

Examples

Address	Value	Description
S1:SI1.PARITYERR	0	Slot 1, port 1 input, no error.
S1:SI2.PARITYERR	1	Slot 1, port 2 input, parity error occurred.

Serial Output: Data Addressing

To transmit serial data, reference address type SO<port>.DATA.

Note: The default configuration parameters are 9600, n, 8 and 1.

Specifications

Syntax	Data Type	Range	Access
S<slot>:SO<port>.BYTESSENT	**Byte**, Word, Short, DWord, Long	1-2	Read Only

* To access as an array, [row][column] form is required. For example, DATA [1][24] would send 24 ASCII bytes in array notation: [x1, x2, x3..x24])

Examples

Address	Value	Description
S0:SO1.DATA	"hello"	Slot 0, port 1output, transmit "hello".
S0:SO2.DATA [2][2]	[105, 105][105, 105]	Slot 0, port 2 output data in array form. In string form this would equate to transmitting "iiii".

Serial Output: Number of Bytes Sent

Reference SO<port>.BYTESSENT to determine how many bytes were sent on the last transmission. This value is available after transmission of serial data.

Requirements

None

Specifications

Syntax	Data Type	Range	Access
S<slot>:SO<port>.BYTESSENT	**Byte**, Word, Short, DWord, Long	1-2	Read Only

Examples

Address	Value	Description
S1:SO1.BYTESSENT	0	Slot 1, port 1 output, no bytes sent on last transmission.
S1:SO1.BYTESSENT	47	Slot 1, port 1 output, 47 bytes sent.

Serial Port Configuration: Baud Rate

To configure the baud rate for a serial port, reference SCFG<port>.BAUD

Values

2 = 1200
3 = 2400
4 = 4800
5 = 9600
6 = 19200

Note: The default value is 9600 baud.

Requirements

None

Specifications

Syntax	Data Type	Range	Access
S<slot>:SCFG<port>.BAUD	**Byte**, Word, Short, DWord, Long	1-2	Write Only

Examples

Address	Value	Description
S1:SCFG1.BAUD	4	Slot 1, port 1, baud = 4800.
S1:SCFG2.BAUD	6	Slot 1, port 2, baud = 19200.

Serial Port Configuration: #Data Bits

To configure the number of data bits for a serial port, reference SCFG<port>.DATABITS

Values

7 or 8

Note: The default value is 8 data bits.

Requirements

None

Specifications

Syntax	Data Type	Range	Access
S<slot>:SCFG<port>.DATABITS	**Byte**, Word, Short, DWord, Long	1-2	Write Only

Examples

Address	Value	Description
S1:SCFG1.DATABITS	7	Slot 1, port 1 configured for 7 data bits.
S1:SCFG2.DATABITS	8	Slot 1, port 2 configured for 8 data bits.

Serial Port Configuration: Parity

To configure the parity for a serial port, reference SCFG<port>.BAUD

Values

0 = none
1 = odd
2 = even

Note: The default value is none.

Requirements

None

Specifications

Syntax	Data Type	Range	Access
S<slot>:SCFG<port>.PARITY	**Byte**, Word, Short, DWord, Long	1-2	Write Only

Examples

Address	Value	Description
S1:SCFG1.PARITY	0	Slot 1, port 1 configured for no parity.
S1:SCFG2.PARITY	2	Slot 1, port 2 configured for even parity.

Serial Port Configuration: #Stop Bits

To configure the number of stop bits for a serial port, reference SCFG<port>.STOPBITS

Values

1 = 1
2 = 2
3 = 1.5

Note: The default value is 1.

Requirements

None

Specifications

Syntax	Data Type	Range	Access
S<slot>:SCFG<port>.STOPBITS	**Byte**, Word, Short, DWord, Long	1-2	Write Only

Examples

Address	Value	Description
S1:SCFG1.STOPBITS	1	Slot 1, port 1 configured for 1 stop bit.
S1:SCFG2.STOPBITS	3	Slot 1, port 2 configured for 1.5 stop bits.

Serial Port Configuration: Set

For any of the serial port configuration properties (baud, parity and so forth) to be sent to the device, SCFG.SET must be set. Immediately after the properties are sent, SCFG.SET will be reset.

Values

True = Send serial port configurations to device
False = No action

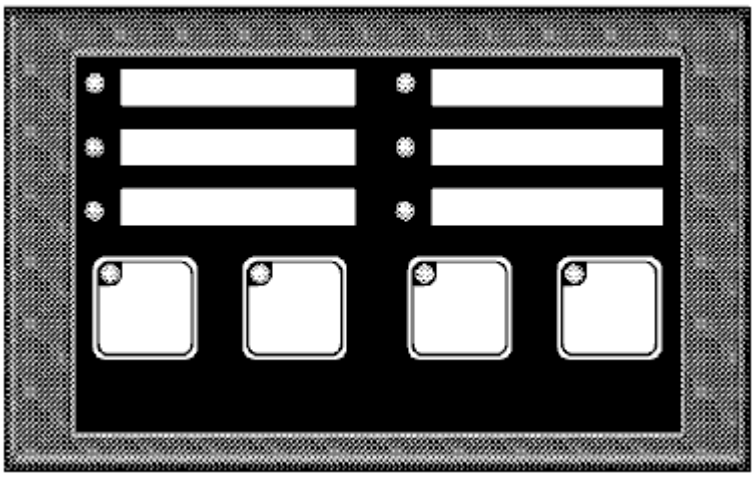
Requirements

None

Specifications

Syntax	Data Type	Range	Access
S<slot>:SCFG<port>.SET	**Boolean**	N/A	Write Only

OL3406 Operator Panel



The OL3406 Operator Panel consists of 4 user definable pushbuttons with built-in LEDs and 6 indicator LEDs.

Subtypes

OL3406

Address Types

Pushbutton LED On State
Pushbutton LED Flash State

Pushbutton LED Separation State
Pushbutton State
Pushbutton Configuration
Force Pushbutton State
Indicator LED On State
Indicator LED Flash State

Pushbutton LED On State Addressing

Each button LED can be forced On/Off without the button being pressed. This can be achieved by referencing address type BTNLED<button>.ON.

Values

True = on
False = off

Requirements

Button LED Separation state must be set. Button(s) must also be configured for momentary action. For more information, refer to PushButton Configuration.

Specifications

Syntax	Data Type	Range	Access
P:BTNLED<button number>.ON	**Boolean**	0-3	Write Only

Examples

Address	Value	Description
P:BTNLED0.ON	1	Turn button 0 LED on (left most button).*
P:BTNLED3.ON	1	Turn button 3 LED on (right most button).*

* See Requirements above.

Pushbutton LED Flash State Addressing

Each button LED can be forced to flash On/Off without the button being pressed. This can be achieved by referencing address type BTNLED<button>.FLASH.

Values

True = flash on
False = flash off

Requirements

Button LED ON state must be set for the corresponding button and button LED Separation state must also be set. Button(s) must be configured for momentary action. For more information, refer to PushButton Configuration Addressing.

Specifications

Syntax	Data Type	Range	Access
P:BTNLED<button number>.FLASH	**Boolean**	0-3	Write Only

Examples

Address	Value	Description
P:BTNLED0.FLASH	1	Flash button 0 LED (left most button).*
P:BTNLED3.FLASH	1	Flash button 3 LED (right most button).*

* See Requirements above.

Pushbutton LED Separation State Addressing

When LED Separation is set, one is capable of controlling the on and flash state of individual button LEDs (see Pushbutton LED On and Flash State Addressing above).

Values

True = separation on
False = separation off

Requirements

None.

Specifications

Syntax	Data Type	Range	Access
P:LEDSEP	**Boolean**	N/A	Read/Write

Pushbutton State Addressing

Button state (pressed/not pressed) can be monitored by referencing address type BTNSTATUS<button>.

Values

True = pressed
False = not pressed

Requirements

None.

Note: Button state depends on the button configuration. For more information, refer to **Pushbutton Configuration Addressing**.

Specifications

Syntax	Data Type	Range	Access
P:BTNSTATUS<button number>	**Boolean**	0-3	Read Only

Examples

Address	Value	Description
P:BTNSTATUS0	1	Button 0 is pressed (left most button).*
P:BTNSTATUS3	0	Button 3 is not pressed (right most button).

* See Note above.

Pushbutton Configuration Addressing

Each button can be configured to either latch their state (alternate action) or hold it momentarily while its being pressed/not pressed.

Values

True = alternate action
False = momentary action

Requirements

None.

Specifications

Syntax	Data Type	Range	Access
P:BTNCFG<button number>	**Boolean**	0-3	Read/Write

Examples

Address	Value	Description
P:BTNCFG0	1	Button 0 is configured for alternate action.
P:BTNCFG3	0	Button 3 is configured for momentary action.

* See Note above.

Force Pushbutton State Addressing

Each button can be forced to a desired state. There are three means of forcing button state.

EQUALS:	Desired button state = specified state
OR:	Desired button state = (current state BITWISE OR specified state)
AND:	Desired button state = NOT(current state BITWISE AND specified state)

Values (Desired button states)

True = button on ("pressed")
False = button off ("not pressed")

Requirements

Button must be configured for alternate action.

Specifications

Below are the three means of forcing button state in detail:

Syntax	Data Type	Range	Access
P:BTNFORCE<button number>.EQUALS	**Boolean**	0-3	Write Only
P:BTNFORCE<button number>.OR	**Boolean**	0-3	Write Only
P:BTNFORCE<button number>.AND	**Boolean**	0-3	Write Only

Examples

Address	Value	Description
P:BTNFORCE0.EQUALS	1	Force button 0 state to be on.
P:BTNFORCE0.OR	1	OR current state with 1. Force button 0 state to be on.
P:BTNFORCE0.AND	1	If button 0 state is currently on, set state to off. Otherwise, do nothing.

* See Requirements above.

Indicator LED On State Addressing

Each indicator LED can be forced On/Off by referencing address type INDLED<led>.ON.

Values

True = on
False = off

Requirements

None.

Specifications

Syntax	Data Type	Range	Access
P:INDLED<led number>.ON	**Boolean**	0-5	Write Only

Examples

Address	Value	Description
P:INDLED0.ON	1	Turn indicator LED 0 on (upper left LED).
P:INDLED5.ON	1	Turn indicator LED 5 on (bottom right LED).

Indicator LED Flash State Addressing

Each indicator LED can be forced to flash On/Off by referencing address type INDLED<led>.FLASH.

Values

True = flash on
False = flash off

Requirements

Indicator LED ON state must be set for the corresponding LED.

Specifications

Syntax	Data Type	Range	Access
P:INDLED<led number>.FLASH	**Boolean**	0-5	Write Only

Examples

Address	Value	Description
P:INDLED1.FLASH	1	Flash indicator LED 1 (upper right LED).*
P:INDLED4.FLASH	1	Flash indicator LED 4 (lower left LED).*

* See Requirements above.

OL3420 Operator Panel

The OL3420 Operator Panel consists of 4 user definable pushbuttons with built-in LEDs and a 2 line x 20 character backlit LCD display

Subtypes

OL3420

Address Types

Pushbutton LED On State
Pushbutton LED Flash State
Pushbutton LED Separation State
Pushbutton State
Pushbutton Configuration
Force Pushbutton State
Alphanumeric Display

Pushbutton LED On State Addressing

Each button LED can be forced On/Off without the button being pressed. This can be achieved by referencing address type BTNLED<button>.ON.

Values

True = on
False = off

Requirements

Button LED Separation state must be set. Button(s) must also be configured for momentary action. For more information, refer to PushButton Configuration.

Specifications

Syntax	Data Type	Range	Access
P:BTNLED<button number>.ON	**Boolean**	0-3	Write Only

Examples

Address	Value	Description
P:BTNLED0.ON	1	Turn button 0 LED on (left most button).*
P:BTNLED3.ON	1	Turn button 3 LED on (right most button).*

* See Requirements above.

Pushbutton LED Flash State Addressing

Each button LED can be forced to flash On/Off without the button being pressed. This can be achieved by referencing address type BTNLED<button>.FLASH.

Values

True = flash on
False = flash off

Requirements

Button LED ON state must be set for the corresponding button and button LED Separation state must be set. Button(s) must also be configured for momentary action. For more information, refer to **PushButton Configuration**.

Specifications

Syntax	Data Type	Range	Access
P:BTNLED<button number>.FLASH	**Boolean**	0-3	Write Only

Examples

Address	Value	Description
P:BTNLED0.FLASH	1	Flash button 0 LED (left most button).*
P:BTNLED3.FLASH	1	Flash button 3 LED (right most button).*

* See Requirements above.

Pushbutton LED Separation State Addressing

When LED Separation is set, one is capable of controlling the on and flash state of individual button LEDs (see Pushbutton LED On and Flash State Addressing above).

Values

True = separation on
False = separation off

Requirements

None.

Syntax	Data Type	Range	Access
P:LEDSEP	**Boolean**	N/A	Read/Write

Pushbutton State Addressing

Button state (pressed/not pressed) can be monitored by referencing address type BTNSTATUS<button>.

Values

True = pressed
False = not pressed

Requirements

None.

Note: Button state depends on the button configuration. See **Pushbutton Configuration Addressing** below.

Specifications

Syntax	Data Type	Range	Access
P:BTNSTATUS<button number>	**Boolean**	0-3	Read Only

Examples

Address	Value	Description
P:BTNSTATUS0	1	Button 0 is pressed (left most button).*
P:BTNSTATUS3	0	Button 3 is not pressed (right most button).

* See Note above.

Pushbutton Configuration Addressing

Each button can be configured to either latch their state (alternate action) or hold it momentarily while its being pressed/not pressed.

Values

True = alternate action
False = momentary action

Requirements

None.

Specifications

Syntax	Data Type	Range	Access
P:BTNCFG<button number>	**Boolean**	0-3	Read/Write

Examples

Address	Value	Description
P:BTNCFG0	1	Button 0 is configured for alternate action.
P:BTNCFG3	0	Button 3 is configured for momentary action.

* See Note above.

Force Pushbutton State Addressing

Each button can be forced to a desired state. There are three ways to force button state.

EQUALS:	Desired button state = specified state
OR:	Desired button state = (current state BITWISE OR specified state)
AND:	Desired button state = NOT(current state BITWISE AND specified state)

Values (Desired button states)

True = button on ("pressed")
False = button off ("not pressed")

Requirements

Button must be configured for alternate action.

Specifications

Below are the three means of forcing button state in detail.

Syntax	Data Type	Range	Access
P:BTNFORCE<button number>.EQUALS	**Boolean**	0-3	Write Only
P:BTNFORCE<button number>.OR	**Boolean**	0-3	Write Only
P:BTNFORCE<button number>.AND	**Boolean**	0-3	Write Only

Examples

Address	Value	Description
P:BTNFORCE0.EQUALS	1	Force button 0 state to be on.
P:BTNFORCE0.OR	1	OR current state with 1. Force button 0 state to be on.
P:BTNFORCE0.AND	1	If button 0 state is currently on, set state to off. Otherwise, do nothing.

* See Requirements above.

Alphanumeric Display Addressing

There are two lines of display for alphanumeric strings of length 20 characters or less.

Requirements

None.

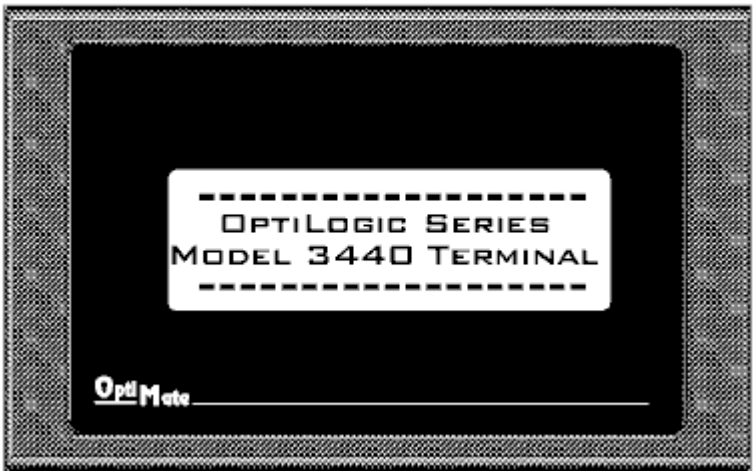
Specifications

Syntax	Data Type	Range	Access
P:LINE<line number>	**String**	0-1	Write Only

Examples

Address	Value	Description	
P:LINE<line number>	**String**	0-1	Write Only

OL3440 Operator Panel



The OL3440 Operator Panel consists of a 4 line x 20 character backlit LCD display

Subtypes

OL3440

Alphanumeric Display Addressing

There are four lines of display for alphanumeric strings of length 20 characters or less.

Requirements

None.

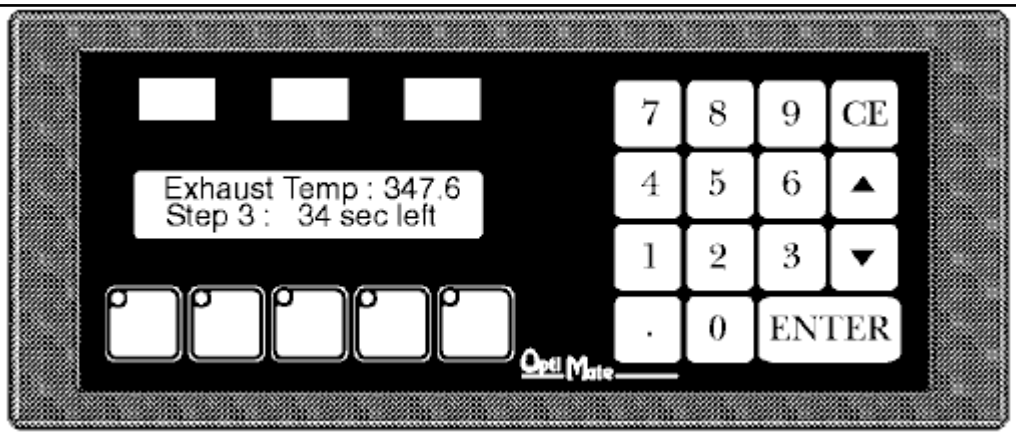
Specifications

Syntax	Data Type	Range	Access
P:LINE<line number>	**String**	0-3	Write Only

Examples

Address	Value	Description
P:LINE0	"hello"	Set line 0 (top line) string to "hello".
P:LINE3	"world"	Set line 3 (bottom line) string to "world".

OL3850 Operator Panel



The OL3850 Operator Panel consists of the following: 2 line x 20 character backlit LCD display, 5 user definable pushbuttons with built-in LEDs, 3 color coded indicator light bars and a numeric keypad. This panel can be used to display messages, accept numeric data input, display status indications and select functions.

Subtypes
OL3850

Address Types
Pushbutton LED On State
Pushbutton LED Flash State
Pushbutton LED Separation State
Pushbutton State
Pushbutton Configuration
Force Pushbutton State
Light Bar On State
Light Bar Flash State
Alphanumeric Display
Keypad Data
Keypad Data Available
Keypad Arrow Max
Keypad Arrow Min

Pushbutton LED On State Addressing
Each button LED can be forced On/Off without the button being pressed. This can be achieved by referencing address type BTNLED<button>.ON.

Values
True = on
False = off

Requirements
Button LED Separation state must be set. Button(s) must also be configured for momentary action. For more information, refer to PushButton Configuration Addressing .

Specifications

Syntax	Data Type	Range	Access
P:BTNLED<button number>.ON	**Boolean**	0-4	Write Only

Examples

Address	Value	Description
P:BTNLED0.ON	1	Turn button 0 LED on (left most button).*
P:BTNLED4.ON	1	Turn button 4 LED on (right most button).*

* See Requirements above.

Pushbutton LED Flash State Addressing

Each button LED can be forced to flash On/Off without the button being pressed. This can be achieved by referencing address type BTNLED<button>.FLASH.

Values

True = flash on
False = flash off

Requirements

Button LED ON state must be set for the corresponding button and button LED Separation state must be set. Button(s) must also be configured for momentary action. For more information, refer to **PushButton Configuration Addressing**.

Syntax	Data Type	Range	Access
P:BTNLED<button number>.FLASH	**Boolean**	0-4	Write Only

Examples

Address	Value	Description
P:BTNLED0.FLASH	1	Flash button 0 LED (left most button).*
P:BTNLED4.FLASH	1	Flash button 4 LED (right most button).*

* See Requirements above.

Pushbutton LED Separation State Addressing

When LED Separation is set, one is capable of controlling the on and flash state of individual button LEDs. For more information, refer to "Pushbutton LED On and Flash State Addressing" above.

Values

True = separation on
False = separation off

Requirements

None.

Specifications

Syntax	Data Type	Range	Access
P:LEDSEP	**Boolean**	N/A	Read/Write

Pushbutton State Addressing

Button state (pressed/not pressed) can be monitored by referencing address type BTNSTATUS<button>.

Values

True = pressed
False = not pressed

Requirements

None.

Note: Button state depends on the button configuration. For more information, refer to **Pushbutton Configuration Addressing** below.

Specifications

Syntax	Data Type	Range	Access
P:BTNSTATUS<button number>	**Boolean**	0-4	Read Only

Examples

Address	Value	Description
P:BTNSTATUS0	1	Button 0 is pressed (left most button).*
P:BTNSTATUS4	0	Button 4 is not pressed (right most button).

* See Note above.

Pushbutton Configuration Addressing

Each button can be configured to either latch their state (alternate action) or hold it momentarily while its being pressed/not pressed.

Values

True = alternate action
False = momentary action

Requirements

None.

Specifications

Syntax	Data Type	Range	Access
P:BTNCFG<button number>	**Boolean**	0-4	Read/Write

Examples

Address	Value	Description
P:BTNCFG0	1	Button 0 is configured for alternate action.
P:BTNCFG4	0	Button 4 is configured for momentary action.

Force Pushbutton State Addressing

Each button can be forced to a desired state. There are three ways to force button state.

EQUALS:	Desired button state = specified state
OR:	Desired button state = (current state BITWISE OR specified state)
AND:	Desired button state = NOT(current state BITWISE AND specified state)

Values (Desired button states)

True = button on ("pressed")
False = button off ("not pressed")

Requirements

Button must be configured for alternate action.

Specifications

Below are the three means of forcing button state in detail:

Syntax	Data Type	Range	Access
P:BTNFORCE<button number>.EQUALS	**Boolean**	0-4	Write Only
P:BTNFORCE<button number>.OR	**Boolean**	0-4	Write Only
P:BTNFORCE<button number>.AND	**Boolean**	0-4	Write Only

Examples

Address	Value	Description
P:BTNFORCE0.EQUALS	1	Force button 0 state to be on.*
P:BTNFORCE0.OR	1	OR current state with 1. Force button 0 state to be on.*
P:BTNFORCE0.AND	1	If button 0 state is currently on, set state to off. otherwise, do nothing.*

* See Requirements above.

Light Bar On State Addressing

Each light bar can be forced On/Off by referencing address type LITEBAR<bar>.ON.

Values

True = on
False = off

Requirements

None.

Specifications

Syntax	Data Type	Range	Access
P:LITEBAR<light bar>.ON	**Boolean**	0-2	Write Only

Examples

Address	Value	Description
P:LITEBAR0.ON	1	Turn light bar 0 on (left most bar).
P:LITEBAR2.ON	1	Turn light bar 2 on (right most bar).

Light Bar Flash State Addressing

Each light bar can be forced to flash On/Off by referencing address type LITEBAR<bar>.FLASH.

Values

True = flash on
False = flash off

Requirements

Light Bar ON state must be set for the corresponding light bar.

Specifications

Syntax	Data Type	Range	Access
P:LITEBAR<light bar>.FLASH	**Boolean**	0-2	Write Only

Examples

Address	Value	Description
P:LITEBAR0.FLASH	1	Flash light bar 0 LED (left most bar).*
P:LITEBAR2.FLASH	1	Flash light bar 2 LED (right most bar).*

* See Requirements above.

Alphanumeric Display Addressing

There are two lines of display for alphanumeric strings of length 20 characters or less. Keypad data may also be inserted into the text string by using the caret (^) as a placeholder for each digit (including the decimal point) of the keypad data.

Requirements

None.

Specifications

Syntax	Data Type	Range	Access
P:LINE<line number>	**String**	0-1	Write Only

Examples

Address	Value	Description
P:LINE0	"hello"	Set line 0 (top line) string to "hello".

Address	Value	Description
P:LINE1	"world"	Set line 1 (bottom line) string to "world".
P:LINE0	"^^^^^^^^"	If keypad data = 1234.56, then line 0 string will be "1234.56".*

* For more information, refer to "Keypad Data Addressing" below.

Keypad Data Addressing

Numeric data can be read from (entered via keypad or this address) and written to (via this address) the panel keypad. Float precision can be specified by appending a bit (0 to 10 allowed) to the KDATA address. This will represent the floating point precision on any writes to the device. If no precision is specified (the default case), the precision will be set such that the number of digits in the integer and fractional parts sums up to 10. If a precision is specified such that the 10-digit limit is exceeded, the precision will be set to the default value previously discussed. Note that the value written to the device may differ from the value displayed in the client. This may be due to floating point round off and truncation errors from the driver, client or both. For more information on floating point precision, refer to **Device Setup**.

Note: Keypad data may be altered using the arrows located on the keypad. Upper and lower bounds set by Arrow Max and Arrow Min respectively, will only limit the data set by the arrows, not the data set in KDATA. Only the data type can place constraints on the upper and lower limits of the keypad data when set using KDATA.

Specifications

Syntax	Data Type	Range	Access
P:KDATA	**Double**, Float, DWord	N/A	Read/Write
P:KDATA.<precision>	**Double**, Float, DWord	N/A	Read/Write

Examples (Writes)

Address	Value	Description
P.KDATA	10.123456	10.123456 will be written to the device.
P:KDATA.4 as a float	10.123456	10.1235 will be written to the device.
P:KDATA	9999999999	9999999999 will be written to the device.

Keypad Data Available Addressing

To determine if new keypad data has been entered at the panel, reference KDATAREADY. This flag can be cleared by writing to KDATA.

Values

True = new data has been entered
False = all data has been read from driver, no new data

Syntax	Data Type	Range	Access
P:KDATAREADY	**Boolean**	N/A	Read Only

Keypad Arrow Max Addressing

Panel keypad data entered via the arrows, can be upper bounded by referencing ARROW.MAX. Any keypad data entered above this max will automatically get set to this max value.

Note: Initially, before ARROW.MAX is set, the upper limit internally to the device is 9999999999. When the limit is set, keypad data cannot exceed the size of ARROW.MAX which is 32 bits (DWord).

Specifications

Syntax	Data Type	Range	Access
P:ARROW.MAX	**DWord**	N/A	Write Only

Keypad Arrow Min Addressing

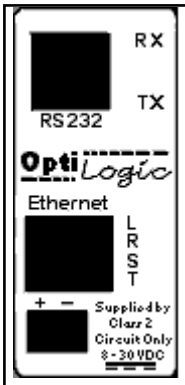
Panel keypad data entered via the arrows, can be lower bounded by referencing ARROW.MIN. Any keypad data entered below this min will automatically get set to this min value.

Note: Initially, before ARROW.MIN is set, the lower limit internally to the device is 0.

Specifications

Syntax	Data Type	Range	Access
P:ARROW.MIN	**DWord**	N/A	Write Only

Base RS232 Port



The Ethernet Base contains an RS232 serial port. Both the transmit buffer and receive buffer of the driver are 48 bytes in size. Likewise, the corresponding tags can be a maximum of 48 bytes. Incoming bytes are appended to the receive buffer as long as they are received in proper time. This time period depends on the baud rate and is based on a 20-character delay using a 20 ms resolution.

300 Baud => 660 ms

1200 Baud => 160 ms

2400 Baud => 80 ms

4800 Baud => 40 ms

9600 Baud => 20 ms

19200 Baud => 20 ms

For a 4800 baud link, bytes would be appended to the receive buffer as long as they were received within 40 ms of the last byte sent. After 40 ms, any incoming bytes are treated as a new stream. The receive buffer would clear and only contain these new bytes.

If the receive buffer is full and additional bytes are received within the proper time frame, the buffer will reset with these additional bytes. The first 48 bytes will be lost.

Below is a list of possible configurations:

1. Baud rates: 300, 1200, 2400, 4800, 9600 and 19200.

2. Data bits: 7 or 8

3. Parity: none, odd or even

4. Stop bits: 1, 1.5 or 2

An RJ45 connector is required.

Subtypes
OL4054, OL4058

Address Types
Serial Input: Data
Serial Input: Number of Received Bytes
Serial Input: Parity Error
Serial Output: Data
Serial Output: Number of Bytes Sent
Serial Port Configuration: Baud Rate
Serial Port Configuration: #Data Bits
Serial Port Configuration: Parity
Serial Port Configuration: #Stop Bits
Serial Port Configuration: Set

Serial Input: Data Addressing
To receive serial data, reference address type SI<port>.DATA.

Note: The default configuration parameters are 300, n, 8 and 1.

Specifications

Syntax	Data Type	Range	Access
B:SI<port>.DATA [r][c]*	**Byte Array**, Char Array	0	Read Only
B:SI<port>.DATA	String	0	Read Only

* To access as an array, [row][column] form is required. For example, DATA [1][24] would display 24 ASCII bytes in array notation: [x1, x2, x3..x24].

Examples

Address	Value	Description
B:SI0.DATA	"hello"	port 0 input data viewed as a string
B:SI0.DATA [2] [2]	[105, 105][105, 105]	port 0 input data in array form. In string form this would equate to "iiii"

Serial Input: Number of Received Bytes Addressing

The number of received serial bytes (number of bytes in SI<port>.DATA. can be accessed by referencing address type SI<port>.NUMBYTES. If bytes are received within the timeout period mentioned above and are therefore appended to the input DATA buffer, NUMBYTES will reflect the total number of bytes in the input DATA buffer and not the number of bytes received on an individual block read. NUMBYTES will reset upon receiving a new stream.

Requirements

None

Specifications

Syntax	Data Type	Range	Access
B:SI<port>:NUMBYTES	**Byte**, Word, Short, DWord, Long	0	Read Only

Examples

Address	Value	Description
B:SI0.NUMBYTES	0	port 0 input, no bytes in input DATA buffer
B:SI0.NUMBYTES	5	port 0 input, 5 bytes in input DATA buffer

Serial Input: Parity Error Addressing

Reference SI<port>.PARITYERR to determine whether a parity error occurred on the last block read of the serial input port.

Values

True = Parity error occurred
False = No parity error occurred

Requirements

None

Specifications

Syntax	Data Type	Range	Access
B:SI<port>.PARITYERR	**Boolean**	0	Read Only

Examples

Address	Value	Description
B:.SI0.PARITYERR	0	port 0 input, no error
B:SI0.PARITYERR	1	port 0 input, parity error occurred

Serial Output: Data Addressing

To transmit serial data, reference address type SO<port>.DATA.

Note: The default configuration parameters are 300, n, 8 and 1.

Specifications

Syntax	Data Type	Range	Access
B:SO<port>.DATA [r][c]*	**Byte Array**, Char Array	0	Write Only
B:SO<port>.DATA	String	0	Write Only

* To access as an array, [row][column] form is required. For example, DATA [1][24] would send 24 ASCII bytes in array notation: [x1, x2, x3..x24].

Examples

Address	Value	Description
B:SO0.DATA	"hello"	port 0 output, transmit "hello"
B:SO0.DATA [2][2]	[105, 105][105, 105]	port 0 output data in array form. In string form this would equate to transmitting "iiii"

Serial Output: Number of Bytes Sent

Reference SO<port>.BYTESSENT to determine how many bytes were sent on the last transmission. This value is available after transmission of serial data.

Requirements

None

Specifications

Syntax	Data Type	Range	Access
B:SO<port>.BYTESSENT	**Byte**, Word, Short, DWord, Long	0	Read Only

Examples

Address	Value	Description
B:SO0.BYTESSENT	0	port 0 output, no bytes sent on last transmission
B:SO0.BYTESSENT	47	port 0 output, 47 bytes sent

Serial Port Configuration: Baud Rate

To configure the baud rate for a serial port, reference SCFG<port>.BAUD

Values

1 = 300
2 = 1200
3 = 2400
4 = 4800
5 = 9600
6 = 19200

Note: The default value is 300 baud.

Requirements
None

Specifications

Syntax	Data Type	Range	Access
B:SCFG<port>.BAUD	**Byte**, Word, Short, DWord, Long	0	Write Only

Examples

Address	Value	Description
B:SCFG0.BAUD	4	port 0, baud = 4800
B:SCFG0.BAUD	6	port 0, baud = 19200

Serial Port Configuration: #Data Bits
To configure the number of data bits for a serial port, reference SCFG<port>.DATABITS

Values
7 or 8

Note: The default value is 8 data bits.

Requirements
None

Specifications

Syntax	Data Type	Range	Access
B:SCFG<port>.DATABITS	**Byte**, Word, Short, DWord, Long	0	Write Only

Examples

Address	Value	Description
B:SCFG0.DATABITS	7	port 0 configured for 7 data bits
B:SCFG0.DATABITS	8	port 0 configured for 8 data bits

Serial Port Configuration: Parity
To configure the parity for a serial port, reference SCFG<port>.BAUD

Values
0 = none
1 = odd
2 = even

Note: The default value is none.

Requirements:

None

Specifications

Syntax	Data Type	Range	Access
B:SCFG<port>.PARITY	**Byte**, Word, Short, DWord, Long	0	Write Only

Examples

Address	Value	Description
B:SCFG0.PARITY	0	port 0 configured for no parity
B:SCFG0.PARITY	2	port 0 configured for even parity

Serial Port Configuration: #Stop Bits

To configure the number of stop bits for a serial port, reference SCFG<port>.STOPBITS

Values

1 = 1
2 = 2
3 = 1.5

Note: The default value is 1.

Requirements

None

Specifications

Syntax	Data Type	Range	Access
B:SCFG<port>.STOPBITS	**Byte**, Word, Short, DWord, Long	0	Write Only

Examples

Address	Value	Description
B:SCFG0.STOPBITS	1	port 0 configured for 1 stop bit
B:SCFG0.STOPBITS	3	port 0 configured for 1.5 stop bits

Serial Port Configuration: Set

For any of the serial port configuration properties (baud, parity and so forth) to be sent to the device, SCFG.SET must be set. Immediately after the properties are sent, SCFG.SET will be reset.

Values

True = Send serial port configurations to device
False = No action

Requirements

None

Specifications

Syntax	Data Type	Range	Access
B:SCFG<port>.SET	**Boolean**	N/A	Write Only

Error Descriptions

The following error/warning messages may be generated. Click on the link for a description of the message.

Driver Error Messages

Winsock initialization failed (OS Error = n)
Winsock V1.1 or higher must be installed to use the Optimation OptiLogic Driver
Memory allocation error

Driver Warning Messages

Device '<device name>' is not responding
Device address <address> contains a syntax error
Address <address> is out of range for the specified device or register
Device address <address> is not supported by model <model name>
Device address <address> is Read Only
Array size is out of range for address <address>
Array support is not available for the specified address: <address>
Data type <type> is not valid for device address <address>
Base w/ type=<type>, Major ver.=<major>, Minor ver.=<minor> is currently not supported. Contact Technical Support
Module in slot <slot> w/ type=<type>, subtype=<subtype> is currently not supported. Contact Technical Support
Panel w/ type=<type>, subtype=<subtype> is currently not supported. Contact Technical Support

Read Errors

Frame received from device <device name> contains errors
Base module referenced in address <address> on device <device name> does not exist
Slot module referenced in address <address> on device <device name> does not exist
Panel module referenced in address <address> on device <device name> does not exist
Address <address> contains an invalid address type for RTU module, device <device name>
Address <address> contains an invalid address type for module <subtype>, slot <slot>, device <device name>
Address <address> contains an invalid address type for panel module <subtype>, device <device name>
Address <address> is out of range for module <subtype> in slot <slot>, device <device name>
Address <address> is out of range for panel module <subtype>, device <device name>
Port <port> of the base module returned an error with value = <error value>
Module <subtype> in slot <slot> returned an error with value = <error value>
Port <port> of module <subtype> in slot <slot> returned an error with value = <error value>
Panel module <subtype> returned an error with value = <error value>

Write Errors

Unable to write to <address> on device <device name>

Write failed. Frame received from device <device name> contains errors

Write rejected. Base module referenced in address <address> on device <device name> does not exist

Write rejected. Slot module referenced in address <address> on device <device name> does not exist

Write rejected. Panel module referenced in address <address> on device <device name> does not exist

Write rejected. Address <address> contains an invalid address type for RTU module, device <device name>

Write rejected. Address <address> contains an invalid address type for module <sub-type>, slot <slot>, device <device name>

Write rejected. Address <address> contains an invalid address type for panel module <subtype>, device <device name>

Write rejected. Address <address> is out of range for module <subtype> in slot <slot>, device <device name>

Write rejected. Address <address> is out of range for panel module <subtype>, device <device name>

Write failed. Port <port> of the base module returned an error with value = <error value>

Write failed. Module <subtype> in slot <slot> returned an error with value = <error value>

Write failed. Port <port> of module <subtype> in slot <slot> returned an error with value = <error value>

Write failed. Panel module <subtype> returned an error with value = <error value>

OptiLogic Device Error Codes

Error codes returned from the OptiLogic RTU, as well as their equivalent error string, are as follows.

Error Code	Description
0	No Error
1	Module message was the improper length
2	Improper module command
3	Module not present

Winsock initialization failed (OS Error = n)

Error Type:

Fatal

OS Error	Indication	Possible Solution
10091	Indicates that the underlying network subsystem is not ready for network communication.	Wait a few seconds and restart the driver.

OS Error	Indication	Possible Solution
10067	Limit on the number of tasks supported by the Windows Sockets implementation has been reached.	Close one or more applications that may be using Winsock and restart the driver.

Winsock V1.1 or higher must be installed to use the OptiLogic device driver

Error Type:
Fatal

Possible Cause:
The version number of the Winsock DLL found on the system is less than 1.1.

Solution:
Upgrade Winsock to version 1.1 or higher.

Memory allocation error

Error Type:
Fatal

Possible Cause:
Insufficient system RAM to support the number of tags the driver is being asked to scan.

Solution:
Increase the amount of system memory or reduce the number tags being scanned.

Device '<device name>' is not responding

Error Type:
Serious

Possible Cause:
1. The Ethernet connection between the device and the Host PC is broken.
2. The named device may have been assigned an incorrect IP address.
3. The requested address is not available in the device.
4. The response from the device took longer to receive than the amount of time specified in the "Request Timeout" device setting.

Solution:
1. Verify the cabling between the PC and the OptiLogic RTU device network.
2. Verify the IP address given to the named device matches that of the actual device.

3. Verify that the device supports the requested address.
4. Increase the Request Timeout setting so that the entire response can be handled.

Device address <address> contains a syntax error

Error Type:
Warning

Possible Cause:
1. A tag address contains one or more invalid characters.
2. Bit addressing notation conflicts with the assigned data type.

Solution:
Re-enter the address in the client application.

Address <address> is out of range for the specified device or register

Error Type:
Warning

Possible Cause:
A tag address that has been specified dynamically references a location that is beyond the range of supported locations for this address type on the specified device. This range is generic and is designed to set a hard upper limit on locations for the specified address type. Each module will impose their own upper limit which is less than or equal to the generic upper limit.

Solution:
Verify that the address is correct; if it is not, re-enter in the client application.

Device address <address> is not supported by model <model name>

Error Type:
Warning

Possible Cause:
The target device does not support a tag address that has been specified dynamically.

Solution:
Verify that the address is correct; if it is not, re-enter it in the client application.

Device address <address> is Read Only

Error Type:
Warning

Possible Cause:
A tag address that has been specified dynamically has a requested access mode that is not compatible with what the device supports for that address.

Solution:
Change the access mode in the client application.

Array size is out of range for address <address>

Error Type:
Warning

Possible Cause:
A tag address that has been specified dynamically is requesting an array size that is too large for the address type of the driver

Solution:
Re-enter the address in the client application to specify a smaller value for the array or a different starting point.

Array support is not available for the specified address: <address>

Error Type:
Warning

Possible Cause:
A tag address that has been specified dynamically contains an array reference for an address type that doesn't support arrays.

Solution:
Re-enter the address in the client application to remove the array reference or correct the address type.

Data type <type> is not valid for device address <address>

Error Type:
Warning

Possible Cause:
A tag address that has been specified dynamically has been assigned an invalid data type.

Solution:
Modify the requested data type in the client application.

Base with type=<type> Major ver.=<major> Minor ver.=<minor> is currently not supported. Contact Technical Support

Error Type:
Warning

Possible Cause:
An OptiLogic RTU with type '<type>', version '<major>' and '<minor>' is currently not supported in this driver.

Solution:

New RTUs may be released that are not currently handled in the driver. Contact Technical Support so that support can be added for this new RTU.

Module in slot <slot> w/ type=<type>, subtype=<subtype> is currently not supported. Contact Technical Support

Error Type:
Warning

Possible Cause:
An OptiLogic module with type '<type>', version '<major>' and '<minor>' is currently not supported in this driver.

Solution:
New modules may be released that are not currently handled in the driver. Contact Technical Support so that support can be added for this new module.

Panel with type=<type>, subtype=<subtype> is currently not supported. Contact Technical Support

Error Type:
Warning

Possible Cause:
An OptiLogic panel with type '<type>', version '<major>' and '<minor>' is currently not supported in this driver.

Solution:
New modules may be released that are not currently handled in the driver. Contact Technical Support so that support can be added for this new panel.

Frame received from device <device name> contains errors

Error Type:
Warning

Possible Cause:
The OptiLogic device <device name> responded with incorrect data possibly due to transmission errors or device malfunction.

Solution:
1. Place device on less noisy network if that is the case.
2. Increase the request timeout.

Base module referenced in address <address> on device <device name> does not exist

Error Type:
Warning

Possible Cause:
A tag address references an RTU that is currently not supported.

Solution:

1. Verify that the address is correct; if it is not, re-enter it in the client application.
2. Contact Technical Support regarding the RTU that is not supported.

Slot module referenced in address <address> on device <device name> does not exist

Error Type:
Warning

Possible Cause:
A tag address references a slot that that is not occupied by an I/O module.

Solution:

1. Verify that the address is correct; if it is not, re-enter it in the client application.
2. Verify that the module fits securely into its mating connector on the motherboard.

Panel module referenced in address <address> on device <device name> does not exist

Error Type:
Warning

Possible Cause:
A tag address references a panel that is not connected to the RTU.

Solution:

1. Verify that the address is correct; if it is not, re-enter it in the client application.
2. Verify that both ends of the interconnect cable are securely connected.

Address <address> contains an invalid address type for RTU module, device <device name>

Error Type:
Warning

Possible Cause:
A tag references an address using an address type that is not valid for this RTU module.

Solution:
Verify that the address type is correct; if it is not, re-enter it in the client application.

Address <address> contains an invalid address type for module <subtype>, slot <slot>, device <device name>

Error Type:
Warning

Possible Cause:
A tag references an address using an address type that is not valid for module '<subtype>' in slot '<slot>'.

Solution:
Verify that the address type is correct; if it is not, re-enter it in the client application.

Address <address> contains an invalid address type for panel module <subtype>, device <device name>

Error Type:
Warning

Possible Cause:
A tag references an address using an address type that is not valid for panel module '<subtype>'.

Solution:
Verify that the address type is correct; if it is not, re-enter it in the client application.

Address <address> is out of range for module <subtype> in slot <slot>, device <device name>

Error Type:
Warning

Possible Cause:
A tag address references a location that is beyond the range of supported locations for module '<subtype>' in slot '<slot>'.

Solution:
Verify that the address is correct; if it is not, re-enter it in the client application.

Address <address> is out of range for panel module <subtype>, device <device name>

Error Type:
Warning

Possible Cause:
A tag address references a location that is beyond the range of supported locations for panel module '<subtype>'.

Solution:
Verify that the address is correct; if it is not, re-enter it in the client application.

Port <port> of the base module returned an error with value = <error value>

Error Type:
Warning

Possible Cause:
The base module's serial port <port> generated an error during its operation. All tags referencing this slot will be invalidated.

Solution:
Check the error list for a detailed description of this error value. If is not included in the list, contact device manufacturer for assistance. Upon a successful block read, this error state will clear and all tags referencing this slot will be validated.

See Also:
Optilogic Device Error Codes

Module <subtype> in slot <slot> returned an error with value = <error value>

Error Type:
Warning

Possible Cause:
Module '<subtype>' in slot <slot>, generated an error during its operation. All tags referencing this slot will be invalidated.

Solution:
Check the error list for a detailed description of this error value. If is not included in the list, contact the device manufacturer for assistance. Upon a successful block read, this error state will clear and all tags referencing this slot will be validated.

See Also:
Optilogic Device Error Codes

Port <port> of module <subtype> in slot <slot> returned an error with value = <error value>

Error Type:
Warning

Possible Cause:
Serial port <port> of module '<subtype>' in slot <slot>, generated an error during its operation. All tags referencing this slot will be invalidated.

Solution:

Check the error list for a detailed description of this error value. If is not included in the list, contact device manufacturer for assistance. Upon a successful block read, this error state will clear and all tags referencing this slot will be validated.

See Also:
Optilogic Device Error Codes

Panel module <subtype> returned an error with value = <error value>

Error Type:
Warning

Possible Cause:
Panel module '<subtype>', generated an error during its operation. All tags referencing this panel will be invalidated.

Solution:
Check the error list for a detailed description of this error value. If is not included in the list, contact device manufacturer for assistance. Upon a successful block read, this error state will clear and all tags referencing this panel will be validated.

See Also:
Optilogic Device Error Codes

Unable to write to <address> on device <device name>

Error Type:
Serious

Possible Cause:

1. The Ethernet connection between the device and the Host PC is broken.
2. The named device may have been assigned an incorrect IP address.
3. The requested address is not available in the device.

Solution:

1. Verify the cabling between the PC and the OptiLogic RTU device network.
2. Verify the IP address given to the named device matches that of the actual device.
3. Verify that the device supports the requested address.

Write failed. Frame received from device <device name> contains errors

Error Type:
Warning

Possible Cause:

A write operation was retried the preset number of times and failed each time. This is possibly due to transmission errors or device malfunction.

Solution:

1. Place device on less noisy network if that is the case.
2. Increase the request timeout.

Write rejected. Base module referenced in address <address> on device <device name> does not exist

Error Type:
Warning

Possible Cause:
A write tag address references an RTU that is currently not supported.

Solution:

1. Verify that the address is correct; if it is not, re-enter it in the client application.
2. Contact Technical Support regarding the RTU that is not supported.

Write rejected. Slot module referenced in address <address> on device <device name> does not exist

Error Type:
Warning

Possible Cause:
A write tag address references a slot that that is not occupied by an I/O module.

Solution:

1. Verify that the address is correct; if it is not, re-enter it in the client application.
2. Verify that the module fits securely into its mating connector on the motherboard.

Write rejected. Panel module referenced in address <address> on device <device name> does not exist

Error Type:
Warning

Possible Cause:
A write tag address references a panel that is not connected to the RTU.

Solution:

1. Verify that the address is correct; if it is not, re-enter it in the client application.
2. Verify that both ends of the interconnect cable are securely connected.

Write rejected. Address <address> contains an invalid address type for RTU module, device <device name>

Error Type:
Warning

Possible Cause:
A write tag references an address using an address type that is not valid for this RTU module.

Solution:
Verify that the address type is correct; if it is not, re-enter it in the client application.

Write rejected. Address <address> contains an invalid address type for module <subtype>, slot <slot>, device <device name>

Error Type:
Warning

Possible Cause:
A write tag references an address using an address type that is not valid for module '<subtype>' in slot '<slot>'.

Solution:
Verify that the address type is correct; if it is not, re-enter it in the client application.

Write rejected. Address <address> contains an invalid address type for panel module <subtype>, device <device name>

Error Type:
Warning

Possible Cause:
A write tag references an address using an address type that is not valid for panel module '<subtype>'.

Solution:
Verify that the address type is correct; if it is not, re-enter it in the client application.

Write rejected. Address <address> is out of range for module <subtype> in slot <slot>, device <device name>

Error Type:
Warning

Possible Cause:

A write tag address references a location that is beyond the range of supported locations for module '<subtype>' in slot '<slot>'.

Solution:
Verify that the address is correct; if it is not, re-enter it in the client application.

Write rejected. Address <address> is out of range for panel module <subtype>, device <device name>

Error Type:
Warning

Possible Cause:
A write tag address references a location that is beyond the range of supported locations for panel module '<subtype>'.

Solution:
Verify that the address is correct; if it is not, re-enter it in the client application.

Write failed. Port <port> of the base module returned an error with value = <error value>

Error Type:
Warning

Possible Cause:
The base module's serial port <port> generated an error during a write operation. Write failed.

Solution:
Check the error list for a detailed description of this error value. If is not included in the list, contact device manufacturer for assistance. Upon a successful block read, this error state will clear and all tags referencing this slot will be validated.

See Also:
Optilogic Device Error Codes

Write failed. Module <subtype> in slot <slot> returned an error with value = <error value>

Error Type:
Warning

Possible Cause:
Module '<subtype>' in slot <slot>, generated an error during a write operation. Write failed.

Solution:
Check the error list for a detailed description of this error value. If is not included in the list, contact device manufacturer for assistance. Upon a successful block read, this error state will clear and all tags referencing this slot will be validated.

See Also:
Optilogic Device Error Codes

Write failed. Port <port> of module <subtype> in slot <slot> returned an error with value = <error value>

Error Type:
Warning

Possible Cause:
Serial port <port> of module '<subtype>' in slot '<slot>' generated an error during a write operation. Write failed.

Solution:
Check the error list for a detailed description of this error value. If is not included in the list, contact device manufacturer for assistance. Upon a successful block read, this error state will clear and all tags referencing this slot will be validated.

See Also:
Optilogic Device Error Codes

Write failed. Panel module <subtype> returned an error with value = <error value>

Error Type:
Warning

Possible Cause:
Panel module '<subtype>', generated an error during a write operation. Write failed.

Solution:
Check the error list for a detailed description of this error value. If is not included in the list, contact device manufacturer for assistance. Upon a successful block read, this error state will clear and all tags referencing this panel will be validated.

See Also:
Optilogic Device Error Codes

Index

A

B

C

D

E

F

G

H

I

M

N

O

P

R

S

T

U

W

Write rejected. Address <address> contains an invalid address type for RTU module device <device name> 151

Write rejected. Address <address> contains an invalid address type for module <subtype>, slot <slot>_ device name 151

Write rejected. Address <address> is out of range for module <subtype> in slot <slot> device <device name> 151

Write rejected. Address <address> is out of range for panel module <subtype> device <device name> 152

Write rejected. Base module referenced in address <address> on device <device name> does not exist 150

Write rejected. Panel module referenced in address <address> on device <device name> does not exist 150

Write rejected. Slot module referenced in address <address> on device <device name> does not exist 150