Allen-Bradley DF1 Driver

Table of Contents

Allen-Bradley DF1 Driver

Help version 1.055

CONTENTS

Overview

The Allen-Bradley DF1 Driver provides a reliable way to connect Allen-Bradley DF1 devices to client applications; including HMI, SCADA, Historian, MES, ERP, and countless custom applications. This driver supports Allen-Bradley Micrologix, SLC500, and PLC5 series PLCs.

Cable Connections

This driver is an Allen-Bradley DF1 RS232 serial driver. It does not handle the token ring passing of a DH-485 or DH+ network. When testing cable connections for this driver, first verify communications using the Allen Bradley programming software.

- For Micrologix and SLC500 series PLCs, if the APS driver selected is KF3/KE, Full-Duplex, Half-Duplex Slave or Full Duplex (Micro) and communication is available with the PLC, this driver can communicate using the same cabling and communication settings.
- For PLC5 series PLCs, if the PLC-5 programming software driver selected is Serial to PLC or KE/KF and communication is available with the PLC, this driver can communicate using the same cabling and communication settings.

Micrologix Series

Use the same cable as when using the Allen Bradley APS software.

SLC500 Series - Direct connection

If the PLC has an RS232 port it can be connected directly using a standard RS232 null modem cable (which is the same cable used in Allen Bradley APS software). The PLC port must be configured for Allen-Bradley DF1 communications, not DH-485 Master.

Note: This driver does not work in a direct connection to the DH-485 port of a SLC500 series PLC using a 1747-PIC converter the way the APS software does.

PLC5 Series - Direct connection

A direct connection can be made to the CH0 port of enhanced PLC5 processors using a standard RS232 null modem cable. The port must be configured for Allen-Bradley DF1 communications.

DH-485 Networks

An Allen Bradley KF3 or compatible device is needed to connect the driver to the DH-485 network. A standard null modem cable is used to connect the PC to the KF3 device.

DH+ Networks

There are three options for communicating to a device on DH+ using the Allen-Bradley DF1 Driver:

- Allen Bradley KF2 or compatible device. A standard null modem cable is used to connect the PC to the KF2 device.
- DataLink DL Interface Cards (PCI/ISA/PC104). Consult AB documentation for DH+ wiring.
- DataLink DL4500 Ethernet-to-DH+ Converter. Consult DL4500 documentation for wiring.

Channel Setup

Supported Link Protocols

Allen-Bradley DF1 Full-Duplex (point-to-point communication)
Allen-Bradley DF1 Half-Duplex Master (multi-drop communication) also known as Allen-Bradley DF1 Polled-Mode.*
Allen-Bradley DF1 Radio Modem (point-to-point and multi-drop communication).**

*Slave-to-slave communication is not supported.
**Store and forward feature is not supported.

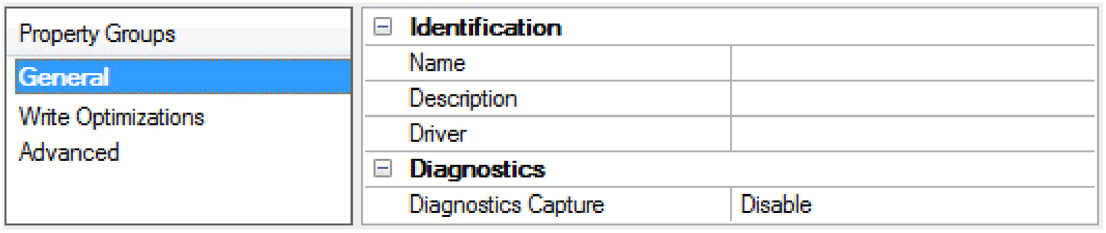
Note: For required firmware versions for Allen-Bradley DF1 Radio Modem support, refer to **Device Setup**.

Channel Properties

General
Serial Communications
Write Optimization
Advanced
Communication Serialization
Link Settings

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	
Description	
Driver	
Diagnostics	
Diagnostics Capture	Disable

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.
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.

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. 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 disabled if the driver does not support diagnostics.

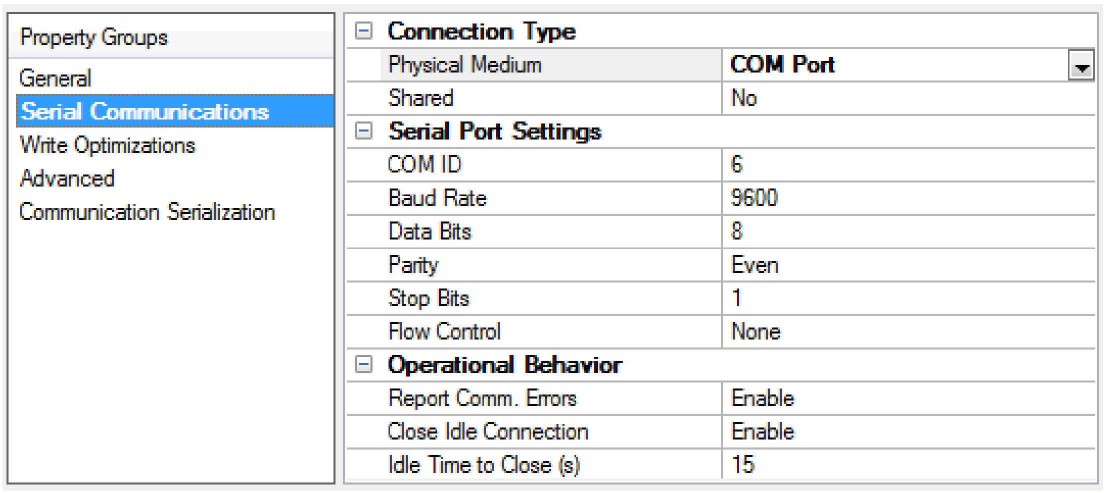
For more information, refer to "Communication Diagnostics" in the server help.

Channel Properties — Serial Communications

Serial communication properties are available to serial drivers and vary depending on the driver, connection type, and options selected. Below is a superset of the possible properties.

Click to jump to one of the sections: **Connection Type**, **Serial Port Settings** or **Ethernet Settings**, and **Operational Behavior**.

Note: With the server's online full-time operation, these properties can be changed at any time. Utilize the User Manager to restrict access rights to server features, as changes made to these properties can temporarily disrupt communications.



Connection Type	
Physical Medium	COM Port
Shared	No
Serial Port Settings	
COM ID	6
Baud Rate	9600
Data Bits	8
Parity	Even
Stop Bits	1
Flow Control	None
Operational Behavior	
Report Comm. Errors	Enable
Close Idle Connection	Enable
Idle Time to Close (s)	15

Connection Type

Physical Medium: Choose the type of hardware device for data communications. Options include COM Port, None, Modem, and Ethernet Encapsulation. The default is COM Port.

- **None**: Select None to indicate there is no physical connection, which displays the **Operation with no Communications** section.
- **COM Port**: Select Com Port to display and configure the **Serial Port Settings** section.

- **Modem**: Select Modem if phone lines are used for communications, which are configured in the Modem Settings section.
- **Ethernet Encap.**: Select if Ethernet Encapsulation is used for communications, which displays the Ethernet Settings section.
- **Shared**: Verify the connection is correctly identified as sharing the current configuration with another channel. This is a read-only property.

Serial Port Settings

COM ID: Specify the Communications ID to be used when communicating with devices assigned to the channel. The valid range is 1 to 9991 to 16. The default is 1.

Baud Rate: Specify the baud rate to be used to configure the selected communications port.

Data Bits: Specify the number of data bits per data word. Options include 5, 6, 7, or 8.

Parity: Specify the type of parity for the data. Options include Odd, Even, or None.

Stop Bits: Specify the number of stop bits per data word. Options include 1 or 2.

Flow Control: Select how the RTS and DTR control lines are utilized. Flow control is required to communicate with some serial devices. Options are:

- **None**: This option does not toggle or assert control lines.
- **DTR**: This option asserts the DTR line when the communications port is opened and remains on.
- **RTS**: This option specifies that the RTS line is high if bytes are available for transmission. After all buffered bytes have been sent, the RTS line is low. This is normally used with RS232/RS485 converter hardware.
- **RTS, DTR**: This option is a combination of DTR and RTS.
- **RTS Always**: This option asserts the RTS line when the communication port is opened and remains on.
- **RTS Manual**: This option asserts the RTS line based on the timing properties entered for RTS Line Control. It is only available when the driver supports manual RTS line control (or when the properties are shared and at least one of the channels belongs to a driver that provides this support).
  RTS Manual adds an **RTS Line Control** property with options as follows:
  - **Raise**: This property specifies the amount of time that the RTS line is raised prior to data transmission. The valid range is 0 to 9999 milliseconds. The default is 10 milliseconds.
  - **Drop**: This property specifies the amount of time that the RTS line remains high after data transmission. The valid range is 0 to 9999 milliseconds. The default is 10 milliseconds.
  - **Poll Delay**: This property specifies the amount of time that polling for communications is delayed. The valid range is 0 to 9999. The default is 10 milliseconds.

Tip: When using two-wire RS-485, "echoes" may occur on the communication lines. Since this communication does not support echo suppression, it is recommended that echoes be disabled or a RS-485 converter be used.

Operational Behavior

- **Report Comm. Errors**: Enable or disable reporting of low-level communications errors. When enabled, low-level errors are posted to the Event Log as they occur. When disabled, these same errors are not posted even though normal request failures are. The default is Enable.
- **Close Idle Connection**: Choose to close the connection when there are no longer any tags being referenced by a client on the channel. The default is Enable.
- **Idle Time to Close**: Specify the amount of time that the server waits once all tags have been removed before closing the COM port. The default is 15 seconds.

Ethernet Settings

Note: Not all serial drivers support Ethernet Encapsulation. If this group does not appear, the functionality is not supported.

Ethernet Encapsulation provides communication with serial devices connected to terminal servers on the Ethernet network. A terminal server is essentially a virtual serial port that converts TCP/IP messages on the Ethernet network to serial data. Once the message has been converted, users can connect standard devices that support serial communications to the terminal server. The terminal server's serial port must be properly configured to match the requirements of the serial device to which it is attached. *For more information, refer to "How To... Use Ethernet Encapsulation" in the server help.*

- **Network Adapter**: Indicate a network adapter to bind for Ethernet devices in this channel. Choose a network adapter to bind to or allow the OS to select the default.
  Specific drivers may display additional Ethernet Encapsulation properties. For more information, refer to Channel Properties — Ethernet Encapsulation.

Modem Settings

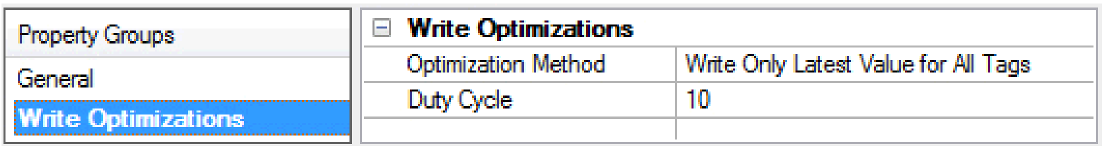
- **Modem**: Specify the installed modem to be used for communications.
- **Connect Timeout**: Specify the amount of time to wait for connections to be established before failing a read or write. The default is 60 seconds.
- **Modem Properties**: Configure the modem hardware. When clicked, it opens vendor-specific modem properties.
- **Auto-Dial**: Enables the automatic dialing of entries in the Phonebook. The default is Disable. *For more information, refer to "Modem Auto-Dial" in the server help.*
- **Report Comm. Errors**: Enable or disable reporting of low-level communications errors. When enabled, low-level errors are posted to the Event Log as they occur. When disabled, these same errors are not posted even though normal request failures are. The default is Enable.
- **Close Idle Connection**: Choose to close the modem connection when there are no longer any tags being referenced by a client on the channel. The default is Enable.
- **Idle Time to Close**: Specify the amount of time that the server waits once all tags have been removed before closing the modem connection. The default is 15 seconds.

Operation with no Communications

- **Read Processing**: Select the action to be taken when an explicit device read is requested. Options include Ignore and Fail. Ignore does nothing; Fail provides the client with an update that indicates failure. The default setting is Ignore.

Channel Properties — Write Optimizations

As with any OPC 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	Write Only Latest Value for All Tags
Duty Cycle	10

Write Optimizations

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

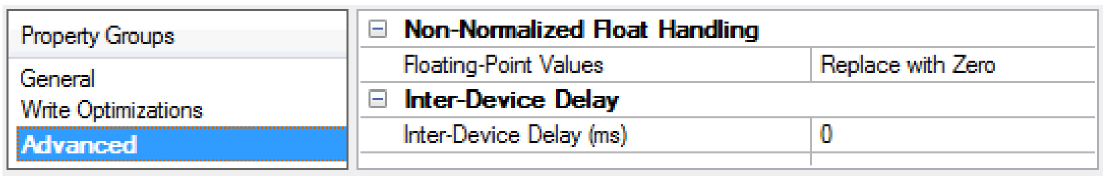
- **Write All Values for All Tags**: This option forces the server to attempt to write every value to the controller. In this mode, the server continues to gather write requests and add them to the server's internal write queue. The server processes the write queue and attempts to empty it by writing 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.



Property Groups
General
Write Optimizations
Advanced

Non-Normalized Float Handling	
Floating-Point Values	Replace with Zero
Inter-Device Delay	
Inter-Device Delay (ms)	0

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 disabled 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.

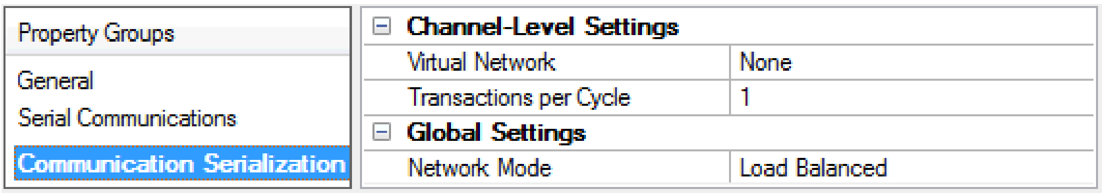
Inter-Device Delay: Specify the amount of time the communications channel waits to send new requests to the next device after data is received from the current device on the same channel. Zero (0) disables the delay.

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

Channel Properties — Communication Serialization

The server's multi-threading architecture allows channels to communicate with devices in parallel. Although this is efficient, communication can be serialized in cases with physical network restrictions (such as Ethernet radios). Communication serialization limits communication to one channel at a time within a virtual network.

The term "virtual network" describes a collection of channels and associated devices that use the same pipeline for communications. For example, the pipeline of an Ethernet radio is the master radio. All channels using the same master radio associate with the same virtual network. Channels are allowed to communicate each in turn, in a “round-robin” manner. By default, a channel can process one transaction before handing communications off to another channel. A transaction can include one or more tags. If the controlling channel contains a device that is not responding to a request, the channel cannot release control until the transaction times out. This results in data update delays for the other channels in the virtual network.



Channel-Level Settings	
Virtual Network	None
Transactions per Cycle	1
Global Settings	
Network Mode	Load Balanced

Channel-Level Settings

Virtual Network This property specifies the channel's mode of communication serialization. Options include None and Network 1 - Network 50. The default is None. Descriptions of the options are as follows:

- **None**: This option disables communication serialization for the channel.
- **Network 1 - Network 50**: This option specifies the virtual network to which the channel is assigned.

Transactions per Cycle This property specifies the number of single blocked/non-blocked read/write transactions that can occur on the channel. When a channel is given the opportunity to communicate, this number of transactions attempted. The valid range is 1 to 99. The default is 1.

Global Settings

- **Network Mode**: This property is used to control how channel communication is delegated. In **Load Balanced** mode, each channel is given the opportunity to communicate in turn, one at a time. In **Priority** mode, channels are given the opportunity to communicate according to the following rules (highest to lowest priority):
  - Channels with pending writes have the highest priority.
  - Channels with pending explicit reads (through internal plug-ins or external client interfaces) are prioritized based on the read’s priority.
  - Scanned reads and other periodic events (driver specific).

  The default is Load Balanced and affects *all* virtual networks and channels.

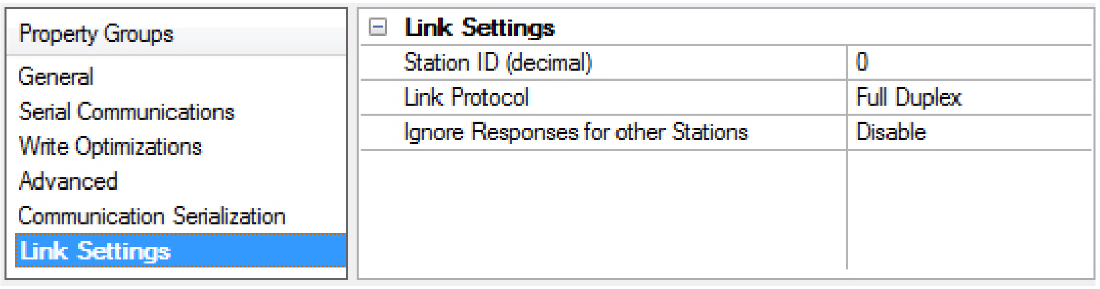
Devices that rely on unsolicited responses should not be placed in a virtual network. In situations where communications must be serialized, it is recommended that Auto-Demotion be enabled.

Due to differences in the way that drivers read and write data (such as in single, blocked, or non-blocked transactions); the application's Transactions per cycle property may need to be adjusted. When doing so, consider the following factors:

- How many tags must be read from each channel?
- How often is data written to each channel?
- Is the channel using a serial or Ethernet driver?
- Does the driver read tags in separate requests, or are multiple tags read in a block?
- Have the device's Timing properties (such as Request timeout and Fail after *x* successive timeouts) been optimized for the virtual network's communication medium?

Channel Properties - Link Settings

Access Link Settings under channel properties.



Link Settings	
Station ID (decimal)	0
Link Protocol	Full Duplex
Ignore Responses for other Stations	Disable

Station ID: The Station identity number is based on the device being communicated with (excluding radio modems):

- If the destination device is on a DH+ or DH-485 network, communication must go through a Serial-to-DH+/DH-485 converter (i.e. KF2/KF3 module). In this case, the device being communicated with is the converter, not the destination device itself (which is a Micrologix, SLC500, or PLC-5). The station number for this configuration should be set to the converter's node address.
- If the destination device is not on a DH+ or DH-485 network, the device being communicated with is a Micrologix, SLC500, or PLC-5 PLC. The station number for this configuration can be set to an arbitrary unique address.
- If the destination device is on a DH+ or DH-485 converter configuration, Station Number = Converter's node address (e.g. KF2/KF3 node address).
- If the destination device is in a standard serial configuration, the station ID is an rbitrary unique address on the network for the local PC. The range for DH-485 is 1 to 63. Otherwise, the range is 0 to 255.

Link Protocol: The following are the supported link protocols:

- **Full-Duplex**: (a.k.a Allen-Bradley DF1) used over a point-to-point link, allowing for high performance two-way communications between peers.
- **Half-Duplex Master**: (a.k.a Polled-Mode) is a multi-drop protocol with one master and one or more slaves. *For details, see* ***Half-Duplex Master***.
- **KF2/KF3 Half-Duplex Master** is a multi-drop protocol with one master and one or more slaves that provides lower data throughput than Full Duplex, but adds the ability to communicate with multiple KF2/KF3 modules from a single COM port. *For details, see* ***KF2/KF3 Half-Duplex Master***.
- **Radio Modem**: a command/reply protocol with no ACKs or NAKs during the request/response procedure. This reduces the number of bytes the radio modems transmit and receive to complete a transaction. This protocol supports full-duplex communications over a point-to-point link, allowing for high performance two-way communications between peers. It also supports master/slave communications, allowing for multi-drop configurations. Performance exceeds Full-Duplex and Half-Duplex Protocols.

Slave Poll Delay (ms): The driver is optimized to send master messages and polls as quickly as possible to increase data throughput. The initial slave poll is not delayed because a delay is unnecessary. If the slave needs time to process requests, it is apparent in the initial poll response, so the driver introduces a delay and re-polls the slave. The delay allows the slave time to process the request before the next poll.

Ignore Responses for other Stations: Enable to limit acceptance of responses to those destined for the station indicated in the Station ID field.

Note: Properties and options available vary with options selected.

Half Duplex Master

Half-Duplex Protocol is a multi-drop protocol with one master and one or more slaves. Generally, Half-Duplex provides lower data throughput than Full Duplex, but it adds the flexibility of being able to communicate with multiple devices from a single COM port. Half-Duplex is a master/slave protocol. In Half-Duplex Master mode, the driver is the master and all devices on the network are slaves. It is necessary that all the devices on the network be configured as Half-Duplex Slave since only one master is allowed on the network.

For more information on configuring the Micrologix/SLC500/PLC5 device using RSLogix, refer to the Rockwell documentation.

Note: If the destination device is on a DH-485 or DH+ network, communication must go through a KF2/KF3 module respectively. If the KF2/KF3 module is configured as a Half-Duplex Slave, the **KF2/KF3 Half-Duplex Master** Link Protocol must be chosen.

Master Responsibilities and Update Rates

The driver (master) is responsible for polling the slaves for data. In general, slaves would be polled in a round-robin manner. Due to the nature of OPC, how often a slave gets polled depends on the slave tags' update rate. In this manner, slaves are only polled when a Read/Write operation is requested from them. This reduces the traffic on the network and prevents unnecessary requests from taking place. In essence, the design of the client project (specifically update rates assigned) determines the traffic on the network. The faster the update rate, the more often a slave is polled.

Messages, Sink and Source

There are three messages exchanged between master and slave in a Read/Write operation. The first is the master message requesting the slave to perform a Read/Write operation. The slave does not respond immediately with data as in full-duplex mode. The second message is a poll message from the master to the slave requesting the data gathered from the last master message operation. The third is the slave response with the data requested in the master message. Incoming requests to the slave are placed in what is termed a "sink". Once the slave performs the operation requested, it places the result in what is termed a "source".

Number of Attempts

The number of attempts for master messages and polls share the same number of attempts as configured in the device Retry Attempts. This attempt count is misleading in Half-Duplex mode since there are multiple messages sent from the master in a single data request. For all intents and purposes:

Let cnAttempts = Retry Attempts
# attempts for master message timeout = cnAttempts
# attempts for poll timeout = cnAttempts
# attempts for request timeout = # attempts for master message timeout + # attempts for poll timeout == cnAttempts X 2

Sink and Source Full

Both sink and source are essentially buffers and buffers have limitations. More importantly, it is possible for the sink to fill up with requests. If this occurs, the slave does not acknowledge any master messages it receives. If after cnAttempts the slave does not ACK the master, the slave sink is likely full. The driver then polls the slave emptying any responses the slave may have, making room for responses of the requests that were in the full sink. This polling action takes place until the slave source is empty after cnAttempts. On the next slave request, the sink is likely empty. If it is not, it may mean the driver is polling the slave too quickly. If this is the case, increase the Slave Re-Poll Delay. Likewise, the slave source may also become full and the driver polls the slave until the source is empty after cnAttempts.

Accepted and Discarded Slave Responses

In the above sections, it has been mentioned that the slave is polled until the emptied. This is possible if the slave source is full of queued up responses. On any given poll, only the response to the last master message is accepted, all others are discarded.

Note: Slave-to-slave communication is not supported.

KF2/KF3 Half Duplex Master

Half-Duplex Protocol is a multi-drop protocol with one master and one or more slaves. Generally, Half-Duplex provides lower data throughput than Full Duplex, but it adds the flexibility of being able to communicate with multiple KF2/KF3 modules from a single COM port. Half-Duplex is a master/slave protocol. In Half-Duplex Master mode, the driver is the master and all KF2/KF3 modules on the network are slaves. It is necessary that all the devices on the network be configured as Half-Duplex Slave since only one master is allowed on the network.

For more information on configuring the KF2/KF3 module for Half-Duplex slave operation, refer to the Rockwell documentation.

Master Responsibilities and Update Rates

The driver (master) is responsible for polling the slaves for data. In general, slaves would be polled in a round-robin manner. Due to the nature of OPC, how often a slave gets polled depends on the update rate of the slave’s tags. In this manner, slaves are only polled when a Read/Write operation is requested from them. This reduces the traffic on the network and prevents unnecessary requests from taking place. In essence, the design of the client project (specifically update rates assigned) determines the traffic on the network. The faster the update rate, the more often a slave is polled.

Messages, Sink and Source

There are three messages exchanged between master and slave in a Read/Write operation. The first is the master message requesting the slave to perform a Read/Write operation. The slave does not respond immediately with data as in full-duplex mode. The second message is a poll message from the master to the slave requesting the data gathered from the last master message operation. The third is the slave response with the data requested in the master message. Incoming requests to the slave are placed in what is termed a "sink". Once the slave performs the operation requested, it places the result in what is termed a "source".

Number of Attempts

The number of attempts for master messages and polls share the same number of attempts as configured in the device Retry Attempts. This attempt count is misleading in Half-Duplex mode since there are multiple messages sent from the master in a single data request. For all intents and purposes:

Let cnAttempts = Retry Attempts
# attempts for master message timeout = cnAttempts
# attempts for poll timeout = cnAttempts
# attempts for request timeout = # attempts for master message timeout + # attempts for poll timeout == cnAttempts X 2

Sink and Source Full

Both sink and source are essentially buffers and buffers have limitations. More importantly, it is possible for the sink to fill up with requests. If this occurs, the slave does not acknowledge any master messages it receives. If after cnAttempts the slave does not ACK the master, it is most likely the case that the slave sink is full. The driver then polls the slave emptying any responses the slave may have, making room for responses of the requests that were in the full sink. This polling action takes place until the slave source is empty after cnAttempts. On the next slave request, the sink is likely empty. If it is not, it may mean the driver is polling the slave too quickly. If this is the case, increase the Slave Re-Poll Delay. Likewise, the slave source may also become full and the driver again polls the slave until the source is empty after cnAttempts.

Accepted and Discarded Slave Responses

In the above sections, it has been mentioned that the slave is polled until the emptied. This is possible if the slave source is full of queued up responses. On any given poll, only the response to the last master message is accepted: all others are discarded.

Note: Slave-to-slave communication is not supported.

Device Setup

Device Properties
Identification
Operating Mode
Scan Mode
Ethernet Encapsulation
Timing
Auto-Demotion
Protocol Settings
Slot Configuration
Function File Options
Redundancy

Supported Devices
Micrologix Series*
SLC500 Series*
PLC-5 Series (excluding the PLC-5/250 and PLC-5/VME series)
RSLogix5000 Controllers with DF1 Port

*Radio Modem link protocol requires the following firmware upgrades:

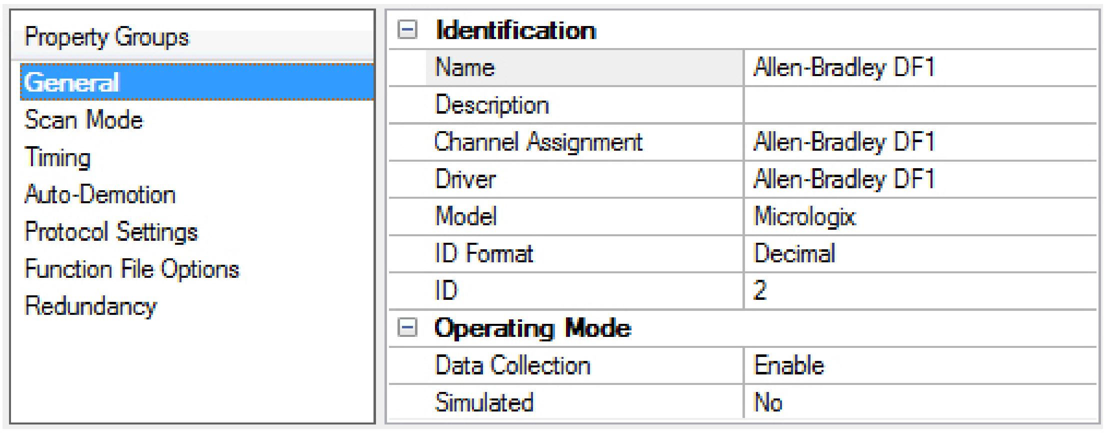
SLC 5/03, SLC 5/04 and SLC 5/05: Series C FRN6
MicroLogix 1200: Series C FRN7
MicroLogix 1500: Series C FRN8

DH-485 and DH+ Support
An Allen Bradley KF3 or compatible device is needed to connect the driver to the DH-485 network. There are four options for communicating to a device on DH+:

- Allen Bradley KF2 or compatible device.
- 1784-U2DHP USB converter. This converter appears as a new serial port to the system.
- DataLink DL Interface Cards (PCI/ISA/PC104). These cards add virtual serial ports for seamless configuration.
- DataLink DL4500 Ethernet-to-DH+ Converter. Configure the device for Ethernet Encapsulation. NIC is required.

See Also:
Cable Connections

Device Properties - General


Property Groups
General
Scan Mode
Timing
Auto-Demotion
Protocol Settings
Function File Options
Redundancy

Identification	
Name	Allen-Bradley DF1
Description	
Channel Assignment	Allen-Bradley DF1
Driver	Allen-Bradley DF1
Model	Micrologix
ID Format	Decimal
ID	2
Operating Mode	
Data Collection	Enable
Simulated	No

Identification

Name: User-defined identity of this device.

Description: User-defined information about this device.

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

Driver: Selected protocol driver for this device.

Model: The specific version of the device.

ID Format: Select how the device identity is formatted. Options include Decimal, Octal, and Hex.

ID: The device ID is the Allen-Bradley DF1 network address of the PLC. For PLCs on a DH-485 or DH+ network, the range is 1-63. Otherwise, the range is 0-255.

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:

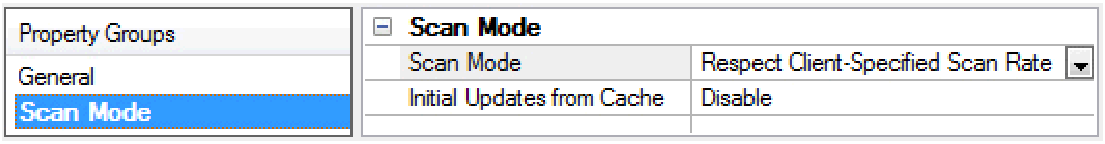
1. This System tag (_Simulated) is read only and cannot be written to for runtime protection. The System tag allows this property to be monitored from the client.

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

Scan Mode	Respect Client-Specified Scan Rate
Initial Updates from Cache	Disable

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 maximum scan rate to be used. 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.

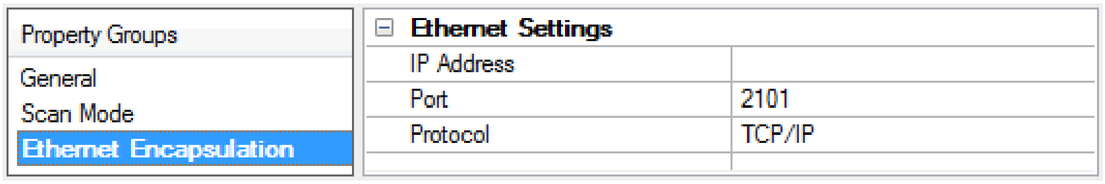
Initial Updates from Cache: When enabled, this option allows the server to provide the first updates for newly activated tag references from stored (cached) data. Cache updates can only be provided when the new item reference shares the same address, scan rate, data type, client access, and scaling properties. 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 — Ethernet Encapsulation

Ethernet Encapsulation is designed to provide communication with serial devices connected to terminal servers on the Ethernet network. A terminal server is essentially a virtual serial port. The terminal server converts TCP/IP messages on the Ethernet network to serial data. Once the message has been converted to a serial form, users can connect standard devices that support serial communications to the terminal server.

For more information, refer to "How to... Use Ethernet Encapsulation" in server help.

Ethernet Encapsulation is transparent to the driver; configure the remaining properties as if connecting to the device directly on a local serial port.


Property Groups
- General
- Scan Mode
- Ethernet Encapsulation

Ethernet Settings	
IP Address	
Port	2101
Protocol	TCP/IP

IP Address: This property is used to enter the four-field IP address of the terminal server to which the device is attached. IPs are specified as YYY.YYY.YYY.YYY. The YYY designates the IP address: each YYY byte should be in the range of 0 to 255. Each serial device may have its own IP address; however, devices may have the same IP address if there are multiple devices multi-dropped from a single terminal server.

Port: This property is used to configure the Ethernet port to be used when connecting to a remote terminal server.

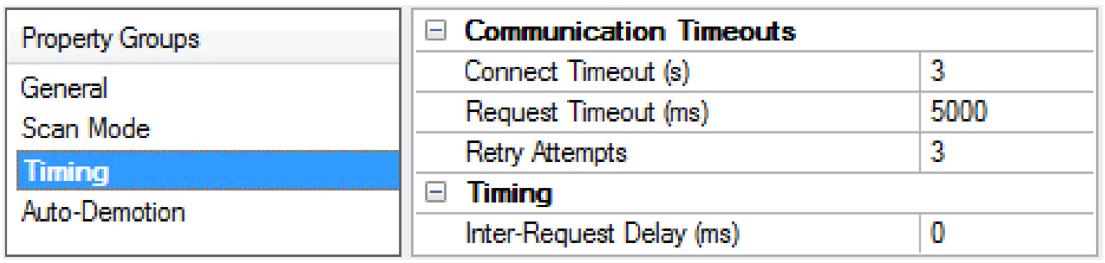
Protocol: This property is used to select either TCP/IP or UDP communications. The selection depends on the nature of the terminal server being used. The default protocol selection is TCP/IP. For more information on available protocols, refer to the terminal server's help documentation.

Notes

1. With the server's online full-time operation, these properties can be changed at any time. Utilize the User Manager to restrict access rights to server features and prevent operators from changing the properties.
2. The valid IP Address range is greater than (>) 0.0.0.0 to less than (<) 255.255.255.255.

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.


Property Groups
- General
- Scan Mode
- Timing
- Auto-Demotion

Communication Timeouts	
Connect Timeout (s)	3
Request Timeout (ms)	5000
Retry Attempts	3
Timing	
Inter-Request Delay (ms)	0

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.

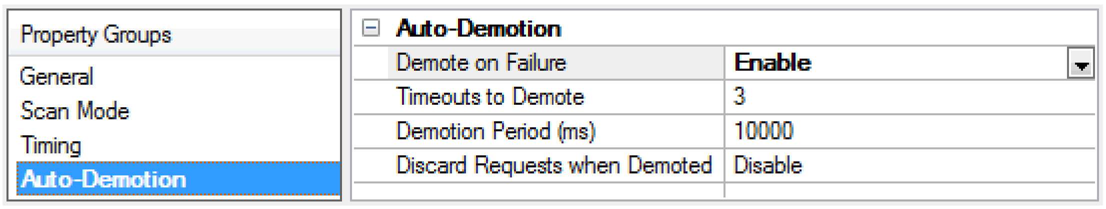
Attempts Before Timeout: This property specifies how many times the driver issues a communications request before considering the request to have failed and the device to be in error. The valid range is 1 to 10. The default is typically 3, but can vary depending on the driver's specific nature. The number 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.



Auto-Demotion	
Demote on Failure	Enable
Timeouts to Demote	3
Demotion Period (ms)	10000
Discard Requests when Demoted	Disable

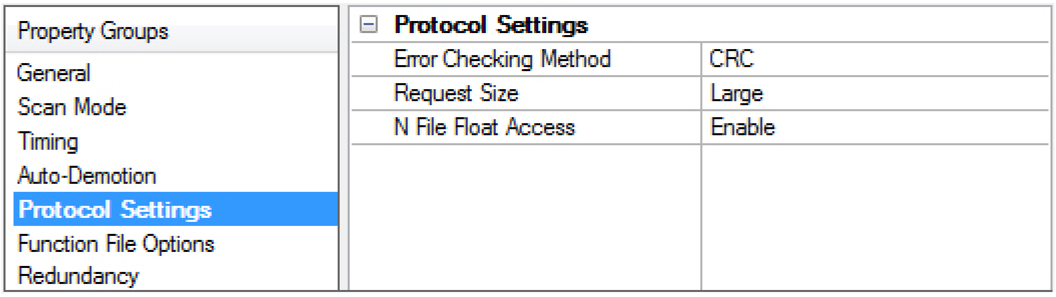
Demote on Failure: When enabled, the device is automatically taken off-scan until it is responding again.
Tip: Determine when a device is off-scan by monitoring its demoted state using the _AutoDemoted system tag.

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 - Protocol Settings



Protocol Settings	
Error Checking Method	CRC
Request Size	Large
N File Float Access	Enable

Error Checking Method: There are two methods of error checking available: Block Check Character (BCC) and 16-bit Cyclic Redundancy Check (CRC). Choose the checksum method expected by the device or the device does not respond.

Swap Float Words: Allen-Bradley PLC-5 devices always transfer the upper word first followed by the lower word, so the Float words must be swapped. This is the default setting. If the device transfers the lower word first, the upper word in the packet on the serial link does not require word swapping.*For more information, see **Float Words**.*

Request Size: Define or change the amount of data requested, which is important in refining application performance. If the application accesses large areas of PLC memory consecutively, a large request size may be beneficial. If the data is spread throughout the PLC, a small request size may be beneficial. The default setting is Large.

N File Float Access: Select whether the driver natively supports Float access to Integer Files. The default setting is Enable.

Float Words

PLC-5 Floats follow the IEEE 754 standard. They contain a sign bit S, an exponent E and a mantissa M. The 32-bit layout of this IEEE 754 Float is as shown below.

Upper Word Lower Word
SEEEEEEE EMMMMMMM MMMMMMMM MMMMMMMM
Byte 3 Byte 2 Byte 1 Byte 0

Allen-Bradley PLC-5 devices transfer binary Floating-point data on the serial link in the following order:
Upper Word Lower Word
Byte 2 Byte 3 Byte 0 Byte 1

This means the upper word is received first, followed by the lower word. Due to this ordering, a swap of the words is required, providing:
Lower Word Upper Word
Byte 0 Byte 1 Byte 2 Byte 3

The result passed on to the client is as follows:
Byte 3 Byte 2 Byte 1 Byte 0

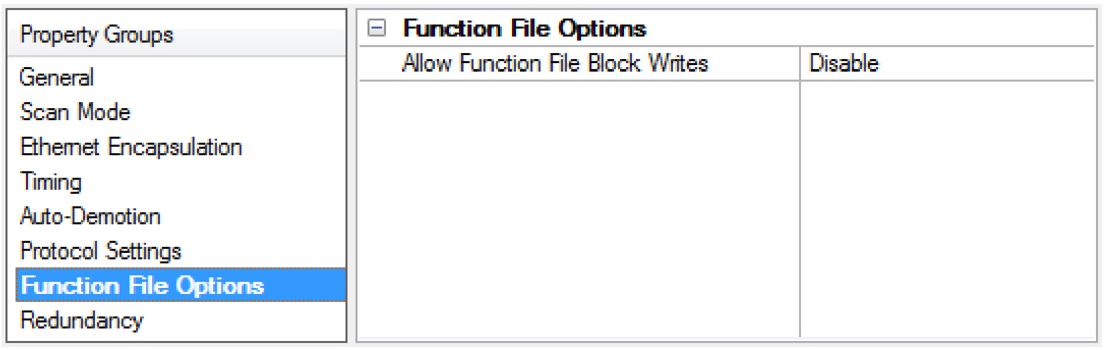
Some PLC-5 emulated devices (such as the Avtron ADDvantage-32) already transfer binary Floating point data on the serial link with the lower word first.

Lower Word Upper Word
Byte 0 Byte 1 Byte 2 Byte 3

In this case, no order swap is required. The result passed on to the client is as follows:
Byte 3 Byte 2 Byte 1 Byte 0

Generally is if the device transfers the lower word first, the upper word in the packet on the serial link does not require word swapping. This only applies to PLC-5 emulated devices; that is, devices that use the Allen-Bradley DF1 protocol with PLC-5 commands. Allen-Bradley PLC-5 devices always transfer the upper word first followed by the lower word so the Float words must be swapped.

Function File Options



Allow Function File Block Writes	Disable

For applicable function files, data can be written to the device in a single operation. By default, when data is written to a function file sub element (field within the function file structure), a write operation occurs immediately for that tag. For such files as the RTC file, whose sub elements include hour (HR), minute (MIN) and second (SEC), individual writes are not always acceptable. With such sub elements relying solely on time, values must be written in one operation to avoid time elapsing between sub elements writes. For this reason, there is the option to "block write" these sub elements.

Applicable Function Files/Sub Elements

RTC	
Year	YR
Month	MON
Day	DAY
Day of Week	DOW

RTC	
Hour	HR
Minute	MIN
Second	SEC

How Block Writes Work

Block writing involves writing to the device the values of every Read/Write sub element in the function file in a single write operation. It is not necessary to write to every sub element prior to performing a block write. Sub elements not affected (written to) have the current value written back to them. For example, if the current (last read) date and time is 1/1/2001, 12:00.00, DOW = 3, and the hour is changed to 1 o'clock, then the values written to the device would be 1/1/2001, 1:00.00, DOW = 3.

Instructions

1. Go to the Function File Options in the device properties. Enable **Allow Function File Block Writes**. This notifies the driver to utilize block writes on function files supporting block writes. The changes are effective immediately after clicking **OK** or **Apply**.
2. Write the desired value to the sub element tag(s) in question. The sub element tag(s) immediately takes on the value(s) written to it.

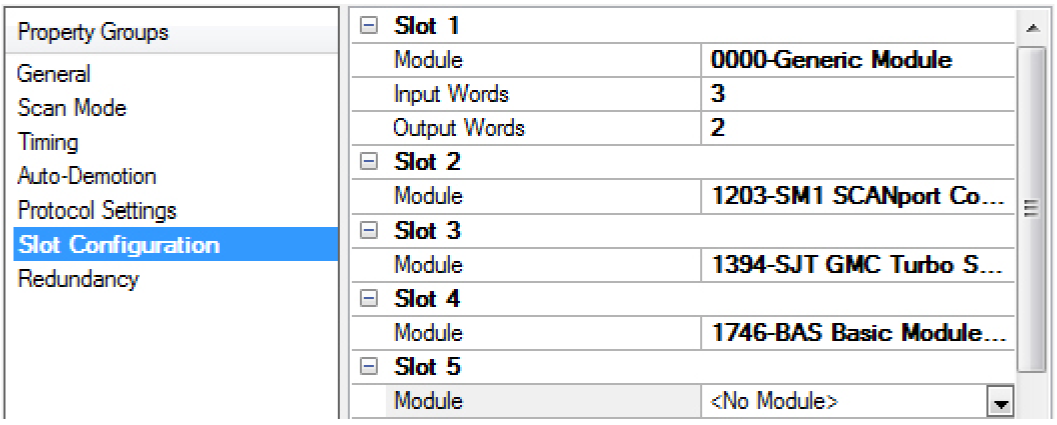
   Note: After a sub element is written to at least once in block write mode, the tag's value does not originate from the controller, but instead from the driver's write cache. After the block write is done, all sub element tag values originate from the controller.
3. Once the entire desired sub elements are written to, the block write that sends these values to the controller may be performed. To instantiate a block write, reference tag address **RTC:<element>._SET**. Setting this tag's value to "True" causes a block write to occur based on the current (last read) sub elements and the sub elements affected (written to). The _SET tag is treated as a Write Only tag; meaning, a write to this tag is not reflected in subsequent reads on it. Setting this tag's value to "False" performs no action.

Function files are structure-based files, similar to PD and MG data files, and are unique to the Micrologix 1200 and 1500.
For more information on a specific function file are supported in the Allen-Bradley DF1 Driver, select a link from the list below.

High-Speed Counter File (HSC)
Real-Time Clock File (RTC)
Channel 0 Communication Status File (CS0)
Channel 1 Communication Status File (CS1)
I/O Module Status File (IOS)

Slot Configuration

SLC500 models (with modular I/O racks) must be configured for use with this driver if the I/O is to be accessed by the driver. Up to 30 slots can be configured per device.



Slot 1	
Module	0000-Generic Module
Input Words	3
Output Words	2
Slot 2	
Module	1203-SM1 SCANport Co...
Slot 3	
Module	1394-SJT GMC Turbo S...
Slot 4	
Module	1746-BAS Basic Module...
Slot 5	
Module	<No Module>

To use the slot configuration, follow the instructions below:

1. Select the slot to be configured by clicking on the row in the module list box.
2. To select a module, click on it from the available modules drop-down list.
3. Configure the Input Words and Output Words if necessary.
4. To remove a slot/module, select **No Module** from the available modules drop-down list.
5. When complete, click **OK**.

Tip: Use the 0000-Generic Module to configure I/O that is not contained in the list of Available Modules.

Note: It is common to have open slots in the rack that do not contain a physical module. To correctly access data for the various slots that do contain a module, the preceding module(s) must have the correct number of words mapped. For example, if only interested in the I/O in slot 3, but slots 1 and 2 contain I/O modules, the correct modules must be selected for slots 1, 2, and 3 from this slot configuration group.

0000-Generic Module
Use the Generic Module to map Input and Output words for modules that are not represented in the list of available modules. To correctly use the Generic Module, users must know the number of Input and Output words required for each module.
Consult Allen-Bradley I/O user manual documentation to confirm Input and Output requirements and be aware that requirements may be different based on Class 1 or Class 3 operation.
For information on the number of input and output words available for each I/O module, refer to ***Modular I/O Selection Guide***.

Modular I/O Selection Guide

The following table lists the number of input and output words available for each I/O module in the Slot Configuration list.

Tip: Use the Generic Module to map input and output words for modules that are not represented in the list of available modules. The range of accepted values is shown in the table below. Consult the Allen-Bradley user manual for the specific I/O module to configure to confirm input and output requirements. Requirements may be different based on Class 1 or Class 3 operation.

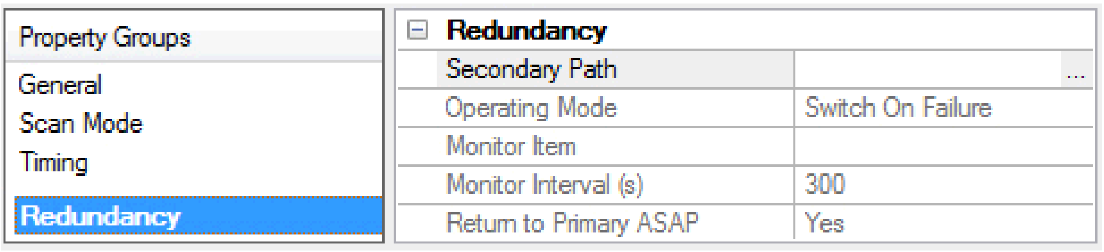
Module Type	Input Words	Output Words
0000-Generic Module	0-255	0-255

Module Type	Input Words	Output Words
1203-SM1 SCANport Comm Module - Basic	8	8
1203-SM1 SCANport Comm Module - Enhanced	32	32
1394-SJT GMC Turbo System	32	32
1746-BAS Basic Module 500 5/01 Configuration	8	8
1746-BAS Basic Module 5/02 Configuration	8	8
1746-HS Single Axis Motion Controller	4	4
1746-HSCE High-Speed Counter/Encoder	8	1
1746-HSRV Motion Control Module	12	8
1746-HSTP1 Stepper Controller Module	8	8
1746-I*16 Any 16 pt Discrete Input Module	1	0
1746-I*32 Any 32 pt Discrete Input Module	2	0
1746-I*8 Any 8 pt Discrete Input Module	1	0
1746-IA16 16 Input 100/120 VAC	1	0
1746-IA4 4 Input 100/120 VAC	1	0
1746-IA8 8 Input 100/120 VAC	1	0
1746-IB16 16 Input (Sink) 24 VDC	1	0
1746-IB32 32 Input (Sink) 24 VDC	2	0
1746-IB8 8 Input (Sink) 24 VDC	1	0
1746-IC16 16 Input (Sink) 48 VDC	1	0
1746-IG16 16 Input [TTL] (Source) 5 VDC	1	0
1746-IH16 16 Input [Trans] (Sink) 125 VDC	1	0
1746-IM16 16 Input 200/240 VAC	1	0
1746-IM4 4 Input 200/240 VAC	1	0
1746-IM8 8 Input 200/240 VAC	1	0
1746-IN16 16 Input 24 VAC/VDC	1	0
1746-INI4I Analog 4 Ch. Isol. Current Input	8	8
1746-INI4VI Analog 4 Ch. Isol. Volt./Current Input	8	8
1746-INO4I Analog 4 Ch. Isol. Current Input	8	8
1746-INO4VI Analog 4 Ch. Isol. Volt./Current Input	8	8
1746-INT4 4 Ch. Isolated Thermocouple Input	8	8
1746-IO12 6 In 100/120 VAC 6 Out [Rly] VAC/VDC	1	1
1746-IO12DC 6 Input 12 VDC, 6 Output [Rly	1	1
1746-IO4 2 In 100/120 VAC 2 Out [Rly] VAC/VDC3	1	1
1746-IO8 4 In 100/120 VAC 4 Out [Rly] VAC/VDC4	1	1
1746-ITB16 16 Input [Fast] (Sink) 24 VDC	1	0
1746-ITV16 16 Input [Fast] (Source) 24 VDC	1	0
1746-IV16 16 Input (Source) 24 VDC	1	0
1746-IV32 32 Input (Source) 24 VDC	2	0
1746-IV8 8 Input (Source) 24 VDC	1	0

Module Type	Input Words	Output Words
1746-NI4 4 Ch Analog Input	4	0
1746-NI8 8 Ch Analog Input, Class 1	8	8
1746-NI8 8 Ch Analog Input, Class 3	16	12
1746-NIO4I Analog Comb 2 in & 2 Current Out	2	2
1746-NIO4V Analog Comb 2 in & 2 Voltage Out	2	2
1746-NO4I 4 Ch Analog Current Output	0	4
1746-NO4V 4 Ch Analog Voltage Output	0	4
1746-NR4 4 Ch Rtd/Resistance Input Module	8	8
1746-NT4 4 Ch Thermocouple Input Module	8	8
1746-NT8 Analog 8 Ch Thermocouple Input	8	8
1746-O*16 Any 16 pt Discrete Output Module	0	1
1746-O*32 Any 32 pt Discrete Output Module	0	2
1746-O*8 Any 8 pt Discrete Output Module	0	1
1746-OA16 16 Output (Triac) 100/240 VAC	0	1
1746-OA8 8 Output (Triac) 100/240 VAC	0	1
1746-OAP12 12 Output [Triac] 120/240 VDC	0	1
1746-OB16 16 Output [Trans] (Source) 10/50 VDC	0	1
1746-OB16E 16 Output [Trans] (Source) Protected	0	1
1746-OB32 32 Output [Trans] (Source) 10/50 VDC	0	2
1746-OB32E 32 Output [Trans] (Source) 10/50 VDC	0	2
1746-OB6EI 6 Output [Trans] (Source) 24 VDC	0	1
1746-OB8 8 Output [Trans] (Source) 10/50 VDC	0	1
1746-OBP16 16 Output [Trans 1 amp] (SRC) 24 VDC	0	1
1746-OBP8 8 Output [Trans 2 amp] (Source) 24 VDC	0	1
1746-OG16 16 Output [TLL] (SINK) 5 VDC	0	1
1746-OV16 16 Output [Trans] (Sink) 10/50 VDC	0	1
1746-OV32 32 Output [Trans] (Sink) 10/50 VDC	0	2
1746-OV8 8 Output [Trans] (Sink) 10/50 VDC	0	1
1746-OVP16 16 Output [Trans 1 amp] (Sink) 24VDC3	0	1
1746-OW16 16 Output [Relay] VAC/VDC	0	1
1746-OW4 4 Output [Relay] VAC/VDC	0	1
1746-OW8 8 Output [Relay] VAC/VDC	0	1
1746-OX8 8 Output [Isolated Relay] VAC/VDC	0	1
1747-DCM Direct Communication Module (1/2 Rack)	4	4
1747-DCM Direct Communication Module (1/4 Rack)	2	2
1747-DCM Direct Communication Module (3/4 Rack)	6	6
1747-DCM Direct Communication Module (Full Rack)	8	8
1747-DSN Distributed I/O Scanner 30 Blocks	32	32
1747-DSN Distributed I/O Scanner 7 Blocks	8	8

Module Type	Input Words	Output Words
1747-KE Interface Module, Series A	1	0
1747-KE Interface Module, Series B	8	8
1747-MNET MNET Network Comm Module	0	0
1746-QS Synchronized Axes Control Module	32	32
1747-QV Open Loop Velocity Control	8	8
1747-RCIF Robot Control Interface Module	32	32
1747-SCNR ControlNet SLC Scanner	32	32
1747-SDN DeviceNet Scanner Module	32	32
1747-SN Remote I/O Scanner	32	32
AMCI-1561 AMCI Series 1561 Resolver Module	8	8

Device Properties — Redundancy



Redundancy

Secondary Path	
Operating Mode	Switch On Failure
Monitor Item	
Monitor Interval (s)	300
Return to Primary ASAP	Yes

Redundancy is available with the Media-Level Redundancy Plug-In.

Consult the website, a sales representative, or the user manual for more information.

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
Long	Signed 32-bit value
BCD	Two byte packed BCD, four decimal digits
LBCD	Four byte packed BCD, eight decimal digits
Float	32-bit IEEE Floating point
String	Null terminated character array

Note: The DWord, Long, and LBCD data types are not native to any of the PLC models.

When referencing a 16-bit location as a 32-bit value, the location referenced is the low word, and the next successive location is the high word. For example, if N7:10 selected as a DWord data type, N7:10 would be the low word and N7:11 the high word.

Address Descriptions

Address specifications vary with the model. Select a link from the following table to obtain specific address information for the model of interest.

Model	Output	Input	Status	Binary	Timer	Counter	Control	Integer	Float	ASCII	String	BCD	Long	PID	Message	Block Transfer	Function
Micrologix	X	X	X	X	X	X	X	X	X		X		X	X	X		X
Micro800	X	X	X	X	X	X	X	X	X		X		X	X	X		X
SLC500*	X	X	X	X	X	X	X	X									
SLC5/01	X	X	X	X	X	X	X	X									
SLC5/02	X	X	X	X	X	X	X	X									
SLC5/03	X	X	X	X	X	X	X	X	X	X	X						
SLC5/04	X	X	X	X	X	X	X	X	X	X	X						
SLC5/05	X	X	X	X	X	X	X	X	X	X	X						
PLC5	X	X	X	X	X	X	X	X	X	X	X	X		X	X	X	

* Fixed I/O processor

Output Files

The syntax for accessing data in an Output file differs depending on the PLC model. Arrays are not supported for output files. The default data types are shown in **bold**.

PLC-5 Syntax

Syntax	Data Type	Access
O:<word>	Short, **Word**, BCD	Read/Write
O:<word>/<bit>	**Boolean**	Read/Write
O/bit	**Boolean**	Read/Write

Note: Word and bit address information is in octal for PLC-5 models. This follows the convention of the programming software.

Micrologix Syntax

Syntax	Data Type	Access
O:<word>	Short, **Word**, BCD	Read/Write
O:<word>/<bit>	**Boolean**	Read/Write
O/bit	**Boolean**	Read/Write

Micrologix models have two types of I/O: embedded I/O and expansion I/O (not applicable for Micrologix 1000). Embedded I/O resides with the CPU base unit while Expansion I/O plugs into the CPU base unit. The table below lists the I/O capabilities of each Micrologix model.

Micrologix Model	Embedded I/O	Expansion I/O
1000	Slot 0	N/A
1100	Slot 0	Slots 1-4
1200	Slot 0	Slots 1-6
1400	Slot 0	Slots 1-7
1500	Slot 0	Slots 1-16

The address syntax for Micrologix I/O references a zero-based word offset, not a slot. Calculations must be done to determine the word offset to a particular slot. This requires knowledge of the modules and their respective size in words. The table below specifies the size of some available modules; however, it is recommended that the Micrologix documentation and controller project be consulted to determine the true word size of a module. Instructions and examples in calculating word offset follow the table below.

Micrologix Embedded I/O Word Sizes

Micrologix Model	# Input Words	# Output Words
1000	2	1
1100	6	4
1200	4	4
1400	8	6
1500	4	4

Micrologix Expansion I/O Word Sizes

Modules	# Input Words	# Output Words
1769-HSC	35	34
1769-IA8I	1	0
1769-IA16	1	0
1769-IF4	6	0
1769-IF4XOF2	8	2
1769-IF8	12	1
1769-IM12	1	0
1769-IQ16	1	0
1769-IQ6XOW4	1	1
1769-IQ16F	1	0
1769-IQ32	2	0
1769-IR6	8	0
1769-IT6	8	0
1769-OA8	0	1
1769-OA16	0	1
1769-OB8	0	1
1769-OB16	0	1
1769-OB16P	0	1

Modules	# Input Words	# Output Words
1769-OB32	0	2
1769-OF2	2	2
1769-OF8C	11	9
1769-OF8V	11	9
1769-OV16	0	1
1769-OW8	0	1
1769-OW16	0	1
1769-OW8I	0	1
1769-SDN	66	2
1769-SM1	12	12
1769-SM2	7	7
1769-ASCII	108	108
1762-IA8	1	0
1762-IF2OF2	6	2
1762-IF4	7	0
1762-IQ8	1	0
1762-IQ8OW6	1	1
1762-IQ16	1	0
1762-OA8	0	1
1762-OB8	0	1
1762-OB16	0	1
1762-OW8	0	1
1762-OW16	0	1
1762-IT4	6	0
1762-IR4	6	0
1762-OF4	2	4
1762-OX6I	0	1

Calculation

Output Word Offset for Slot x = # Output Words in Slot 0 through Slot (x-1).

Notes:

1. The Embedded I/O needs to be taken into account when offsetting to Expansion I/O.
2. The number of Input words does not factor into the calculation for Output Word Offset.

I/O Example

Let
Slot 0 = Micrologix 1500 LRP Series C = 4 Output Words
Slot 1 = 1769-OF2 = 2 Output Words
Slot 2 = 1769-OW8 = 1 Output Word
Slot 3 = 1769-IA16 = 0 Output Word

Slot 4 = 1769-OF8V = 9 Output Word
Bit 5 of Slot 4 = 4 + 2 + 1 = 7 words = O:7/5

SLC500 Syntax

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

Syntax	Data Type	Access
O:<slot>	Short, **Word**, BCD	Read Only
O:<slot>.<word>	Short, **Word**, BCD	Read Only
O:<slot>/<bit>	**Boolean**	Read Only
O:<slot>.<word>/<bit>	**Boolean**	Read Only

Ranges

PLC Model	Min. Slot	Max. Slot	Max. Word
Micrologix	NA	NA	2047
SLC500 Open	NA	NA	1
SLC 5/01	1	30	*
SLC 5/02	1	30	*
SLC 5/03	1	30	*
SLC 5/04	1	30	*
SLC 5/05	1	30	*
PLC-5 Family	NA	NA	277**

*The number of input or output words available for each I/O module can be found in the **Modular I/O Selection Guide**. For slot configuration help, refer to **Device Setup**.
**Octal.

Examples

Micrologix	Address
O:0	Word 0
O/2	Bit 2
O:0/5	Bit 5

SLC500 Fixed I/O	Address
O:0	Word 0
O:1	Word 1
O/16	Bit 16
O:1/0	Bit 0 word 1 (same as O/16)

PLC5	Address*
O:0	Word 0
O:37	Word 31 (37 octal = 31 decimal)

PLC5	Address*
O/42	Bit 34 (42 octal = 34 decimal)
O:2/2	Bit 2 Word 2 (same as O/42)

*Addresses are shown in Octal.

SLC500 Modular I/O	Address
O:1	Word 0 slot 1
O:1.0	Word 0 slot 1 (same as O:1)
O:12	Word 0 slot 12
O:12.2	Word 2 slot 12
O:4.0/0	Bit 0 word 0 slot 4
O:4/0	Bit 0 slot 4 (same as O:4.0/0)
O:4.2/0	Bit 0 word 2 slot 4
O:4/32	Bit 32 slot 4 (same as O:4.2/0)

Input Files

The syntax for accessing data in an Input file differs depending on the PLC model. Arrays are not supported for Input files. The default data types are shown in **bold**.

PLC-5 Syntax

Syntax	Data Type	Access
I:<word>	Short, **Word**, BCD	Read/Write
I:<word>/<bit>	**Boolean**	Read/Write
I/bit	**Boolean**	Read/Write

Note: Word and bit address information is in octal for PLC-5 models. This follows the convention of the programming software.

Micrologix Syntax

Syntax	Data Type	Access
I:<word>	Short, **Word**, BCD	Read/Write
I:<word>/<bit>	**Boolean**	Read/Write
I/bit	**Boolean**	Read/Write

Micrologix models have two types of I/O: embedded I/O and expansion I/O (not applicable for Micrologix 1000). Embedded I/O resides with the CPU base unit while Expansion I/O plugs into the CPU base unit. The table below lists the I/O capabilities of each Micrologix model.

Micrologix Model	Embedded I/O	Expansion I/O
1000	Slot 0	N/A
1100	Slot 0	Slots 1-4
1200	Slot 0	Slots 1-6

Micrologix Model	Embedded I/O	Expansion I/O
1400	Slot 0	Slots 1-7
1500	Slot 0	Slots 1-16

The address syntax for Micrologix I/O references a zero-based word offset, not a slot. Calculations must be done to determine the word offset to a particular slot. This requires knowledge of the modules and their respective size in words. The table below specifies the size of some available modules; however, it is recommended that the Micrologix documentation and controller project be consulted to determine the true word size of a module. Instructions and examples in calculating word offset follow the table below.

Micrologix Embedded I/O Word Sizes

Micrologix Model	# Input Words	# Output Words
1000	2	1
1100	6	4
1200	4	4
1400	8	6
1500	4	4

Micrologix Expansion I/O Word Sizes

Modules	# Input Words	# Output Words
1769-HSC	35	34
1769-IA8I	1	0
1769-IA16	1	0
1769-IF4	6	0
1769-IF4XOF2	8	2
1769-IF8	12	1
1769-IM12	1	0
1769-IQ16	1	0
1769-IQ6XOW4	1	1
1769-IQ16F	1	0
1769-IQ32	2	0
1769-IR6	8	0
1769-IT6	8	0
1769-OA8	0	1
1769-OA16	0	1
1769-OB8	0	1
1769-OB16	0	1
1769-OB16P	0	1
1769-OB32	0	2
1769-OF2	2	2
1769-OF8C	11	9

Modules	# Input Words	# Output Words
1769-OF8V	11	9
1769-OV16	0	1
1769-OW8	0	1
1769-OW16	0	1
1769-OW8I	0	1
1769-SDN	66	2
1769-SM1	12	12
1769-SM2	7	7
1769-ASCII	108	108
1762-IA8	1	0
1762-IF2OF2	6	2
1762-IF4	7	0
1762-IQ8	1	0
1762-IQ8OW6	1	1
1762-IQ16	1	0
1762-OA8	0	1
1762-OB8	0	1
1762-OB16	0	1
1762-OW8	0	1
1762-OW16	0	1
1762-IT4	6	0
1762-IR4	6	0
1762-OF4	2	4
1762-OX6I	0	1

Calculation

Input Word Offset for Slot x = # Input Words in Slot 0 through Slot (x-1).

Notes:

1. The Embedded I/O needs to be taken into account when offsetting to Expansion I/O.
2. The number of Output words does not factor into the calculation for Input Word Offset.

I/O Example

Let
Slot 0 = Micrologix 1500 LRP Series C = 4 Input Words
Slot 1 = 1769-OF2 = 2 Input Words
Slot 2 = 1769-OW8 = 0 Input Word
Slot 3 = 1769-IA16 = 1 Input Word
Slot 4 = 1769-OF8V = 11 Input Word
Bit 5 of Slot 3 = 4 + 2 = 6 words = I:6/5

SLC500 Syntax

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

Syntax	Data Type	Access
I:<slot>	Short, **Word**, BCD	Read Only
I:<slot>.<word>	Short, **Word**, BCD	Read Only
I:<slot>/<bit>	**Boolean**	Read Only
I:<slot>.<word>/<bit>	**Boolean**	Read Only

Ranges

PLC Model	Min. Slot	Max. Slot	Max. Word
Micrologix	NA	NA	2047
SLC500 Open	NA	NA	1
SLC 5/01	1	30	*
SLC 5/02	1	30	*
SLC 5/03	1	30	*
SLC 5/04	1	30	*
SLC 5/05	1	30	*
PLC-5 Family	NA	NA	277

*The number of Input or Output words available for each I/O module can be found in the **Modular I/O Selection Guide**. For more information, refer to **Device Setup**.
**Octal.

Examples

Micrologix	Address
I:0	Word 0
I/2	Bit 2
I:1/5	Bit 5 word 1

SLC500 Fixed I/O	Address
I:0	Word 0
I:1	Word 1
I/16	Bit 16
I:1/0	Bit 0 word 1 (same as I/16)

PLC5	Address
I:0	Word 0
I:10	Word 8 (10 octal = 8 decimal)
I/20	Bit 16 (20 octal = 16 decimal)
I:1/0	Bit 0 word 1 (same as I/20)

*Addresses are shown in Octal.

SLC500 Modular I/O	Address
I:1	Word 0 slot 1
I:1.0	Word 0 slot 1 (same as I:1)
I:12	Word 0 slot 12
I:12.2	Word 2 slot 12
I:4.0/0	Bit 0 word 0 slot 4
I:4/0	Bit 0 slot 4 (same as I:4.0/0)
I:4.2/0	Bit 0 word 2 slot 4
I:4/32	Bit 32 slot 4 (same as I:4.2/0)

Status Files

To access Status files, specify a word and an optional bit in the word. The default data types are shown in **bold**.

Syntax	Data Type	Access
S:<word>	Short, **Word**, BCD, DWord, Long, LBCD	Read/Write
S:<word> [rows][cols]	Short, **Word**, BCD, DWord, Long, LBCD*	Read/Write
S:<word> [cols]	Short, **Word**, BCD, DWord, Long, LBCD*	Read/Write
S:<word>/<bit>	**Boolean**	Read/Write
S/bit	**Boolean**	Read/Write

*Array types.

The number of array elements (in bytes) cannot exceed the block request size specified. This means that array size cannot exceed 16 words given a block request size of 32 bytes.

Ranges

PLC Model	Max. Word
Micrologix	96
All SLC	96
PLC-5	127

The maximum word location is one less when accessing as a 32-bit data type (Long, DWord or Long BCD).

Examples

Example	Description
S:0	Word 0.
S/26	Bit 26.
S:4/15	Bit 15 word 4.
S:10 [16]	16 Element array starting at word 10.
S:0 [4] [8]	4 by 8 element array starting at word 0.

Binary Files

To access Binary files, specify a file number, a word and optional bit in the word. The default data types are shown in **bold**.

Syntax	Data Type	Access
B<file>:<word>	Short, **Word**, BCD, DWord, Long, LBCD	Read/Write
B<file>:<word> [rows][cols]	Short, **Word**, BCD, DWord, Long, LBCD*	Read/Write
B<file>:<word> [cols]	Short, **Word**, BCD, DWord, Long, LBCD*	Read/Write
B<file>:<word>/<bit>	**Boolean**	Read/Write
B<file>/bit	**Boolean**	Read/Write

*Array types.

The number of array elements (in bytes) cannot exceed the block request size specified. This means that the array size cannot exceed 16 words given a block request size of 32 bytes.

Ranges

PLC Model	File Number	Max. Word
Micrologix	3-255	255
All SLC	3-255	255
PLC-5	3-999	1999

The maximum word location is one less when accessing as a 32-bit data type (Long, DWord or Long BCD).

Examples

Example	Description
B3:0	Word 0.
B3/26	Bit 26.
B12:4/15	Bit 15 word 4.
B3:10 [20]	20 Element array starting at word 10.
B15:0 [6] [6]	6 by 6 element array starting at word 0.

Timer Files

Timer files are a structured type whose data is accessed by specifying a file number, an element and a field. The default data types are shown in **bold**.

Syntax	Data Type	Access
T<file>:<element>.<field>	Depends on field	Depends on field

The following fields are allowed for each element. For the usage of each field, refer to the PLC's documentation.

Element Field	Data Type	Access
ACC	**Short**, Word	Read/Write
PRE	**Short**, Word	Read/Write
DN	**Boolean**	Read Only
TT	**Boolean**	Read Only
EN	**Boolean**	Read Only

Ranges

PLC Model	File Number	Max. Element
Micrologix	3-255	255
All SLC	3–255	255
PLC-5	3-999	1999

Examples

Example	Description
T4:0.ACC	Accumulator of timer 0 file 4.
T4:10.DN	Done bit of timer 10 file 4.
T15:0.PRE	Preset of timer 0 file 15.

Counter Files

Counter files are a structured type whose data is accessed by specifying a file number, an element and a field. The default data types are shown in **bold**.

Syntax	Data Type	Access
C<file>:<element>.<field>	Depends on field	Depends on field

The following fields are allowed for each element. For the meaning of each field, refer to the PLC's documentation.

Element Field	Data Type	Access
ACC	**Word**, Short	Read/Write
PRE	**Word**, Short	Read/Write
UA	**Boolean**	Read Only
UN	**Boolean**	Read Only
OV	**Boolean**	Read Only
DN	**Boolean**	Read Only
CD	**Boolean**	Read Only
CU	**Boolean**	Read Only

Ranges

PLC Model	File Number	Max. Element
Micrologix	3–255	255
All SLC	3-255	255
PLC-5	3-999	1999

Examples

Example	Description
C5:0.ACC	Accumulator of counter 0 file 5.
C5:10.DN	Done bit of counter 10 file 5.
C15:0.PRE	Preset of counter 0 file 15.

Control Files

Control files are a structured type whose data is accessed by specifying a file number, an element and a field. The default data types are shown in **bold**.

Syntax	Data Type	Access
R<file>:<element>.<field>	Depends on field	Depends on field

The following fields are allowed for each element. For the meaning of each field, refer to the PLC's documentation.

Element Field	Data Type	Access
LEN	**Word**, Short	Read/Write
POS	**Word**, Short	Read/Write
FD	**Boolean**	Read Only
IN	**Boolean**	Read Only
UL	**Boolean**	Read Only
ER	**Boolean**	Read Only
EM	**Boolean**	Read Only
DN	**Boolean**	Read Only
EU	**Boolean**	Read Only
EN	**Boolean**	Read Only

Ranges

PLC Model	File Number	Max. Element
Micrologix	3–255	255
All SLC	3–255	255
PLC-5	3-999	1999

Examples

Example	Description
R6:0.LEN	Length field of control 0 file 6.
R6:10.DN	Done bit of control 10 file 6.
R15:18.POS	Position field of control 18 file 15.

Integer Files

To access Integer files, specify a file number, a word and an optional bit in the word. The default data types are shown in **bold**.

Syntax	Data Type	Access
N<file>:<word>	Short, **Word**, BCD, DWord, Long, LBCD	Read/Write
N<file>:<word> [rows][cols]	Short, **Word**, BCD, DWord, Long, LBCD*	Read/Write
N<file>:<word> [cols]	Short, **Word**, BCD, DWord, Long, LBCD*	Read/Write
N<file>:<word>/<bit>	**Boolean**	Read/Write
N<file>/bit	**Boolean**	Read/Write

*Array types.

The number of array elements (in bytes) cannot exceed the block request size specified. This means that array size cannot exceed 16 words given a block request size of 32 bytes.

Ranges

PLC Model	File Number	Max. Word
Micrologix	3-255	255
All SLC	3-255	255
PLC-5	3-999	1999

The maximum word location is one less when accessing as a 32-bit data type (Long, DWord or Long BCD).

Examples

Example	Description
N7:0	Word 0.
N7/26	Bit 26.
N12:4/15	Bit 15 word 4.
N7:10 [8]	8 Element array starting at word 10.
N15:0 [4] [5]	4 by 5 element array starting at word 0.

Float Files

To access Float files, specify a file number and an element. The default data types are shown in **bold**.

Syntax	Data Type	Access
F<file>:<element>	**Float**	Read/Write

Syntax	Data Type	Access
F<file>:<element> [rows][cols]	**Float***	Read/Write
F<file>:<element> [cols]	**Float***	Read/Write

*Array type.

The number of array elements (in bytes) cannot exceed the block request size specified. This means that the array size cannot exceed 8 Floats given a block request size of 32 bytes.

Ranges

PLC Model	File Number	Max. Word
Micrologix	3-255	255
All SLC	3-255	255
PLC-5	3-999	1999

Examples

Example	Description
F8:0	Float 0.
F8:10 [16]	16 Element array starting at word 10.
F15:0 [4] [4]	16 Element array starting at word 0.

ASCII Files

To access ASCII file data, specify a file number and a character location. The default data types are shown in **bold**.

Syntax	Data Type	Access
A<file>:<char>	**Char**, Byte*	Read/Write
A<file>:<char> [rows][cols]	**Char**, Byte*	Read/Write
A<file>:<char> [cols]	**Char**, Byte*	Read/Write
A<file>:<word offset>/length	String**	Read/Write

*The number of array elements cannot exceed the block request size specified. Internally, the PLC packs two characters per word in the file, with the high byte containing the first character and the low byte containing the second character. The PLC programming software allows access at the word level or two-character level. The Allen-Bradley DF1 Driver allows accessing to the character level.

Using the programming software, **A10:0 = AB**, would result in 'A' being stored in the high byte of A10:0 and 'B' being stored in the low byte. Using the Allen-Bradley DF1 Driver, the two assignments **A10:0 = A** and **A10:1 = B** would result in the same data being stored in the PLC memory.

**Referencing this file as string data allows access to data at word boundaries like the programming software. The length can be up to 236 characters. If a string that is sent to the device is smaller in length than the length specified by the address, the driver null terminates the string before sending it down to the controller.

Ranges

PLC Model	File Number	Max. Character
Micrologix	3-255	511
All SLC	3-255	511
PLC-5	3-999	1999

Not all Micrologix and SLC500 PLC devices support ASCII file types. For more information, refer to the PLC's documentation.

Examples

Example	Description
A9:0	Character 0 (high byte of word 0).
A27:10 [80]	80 Character array starting at character 10.
A15:0 [4] [16]	4 By 16 character array starting at character 0.
A62:0/32	32 Character string starting at word offset 0.

String Files

To access data in a String file, specify a file number and an element. Strings are 82 character null terminated arrays. The driver places the null terminator based on the string length returned by the PLC. The default data types are shown in **bold**.

Note: Arrays are not supported for String files.

Syntax	Data Type	Access
ST<file>:<element>.<field>	**String**	Read/Write

Ranges

PLC Model	File Number	Max. Word
Micrologix	3-255	255
All SLC	3-255	255
PLC-5	3-999	999

Examples

Example	Description
ST9:0	String 0.
ST18:10	String 10.

BCD Files

To access BCD files, specify a file number and a word. The default data types are shown in **bold**.

Syntax	Data Type	Access
D<file>:<word>	**BCD**, LBCD	Read/Write
D<file>:<word> [rows][cols]	**BCD**, LBCD*	Read/Write

Syntax	Data Type	Access
D<file>:<word> [cols]	**BCD**, LBCD*	Read/Write

*Array types.

The number of array elements (in bytes) cannot exceed the block request size specified. This means that the array size cannot exceed 16 BCD, given a block request size of 32 bytes.

Ranges

PLC Model	File Number	Max Word
Micrologix	NA	NA
All SLC	NA	NA
PLC-5	3-999	999

Examples

Example	Description
D9:0	Word 0.
D27:10 [16]	16 Element array starting at word 10.
D15:0 [4] [8]	32 Element array starting at word 0.

Long Files

To access Long files, specify a file number and a DWord. The default data types are shown in **bold**.

Syntax	Data Type	Access
L<file>:<DWord>	Long, **DWord**, LBCD	Read/Write
L<file>:<DWord> [rows][cols]	Long, **DWord**, LBCD*	Read/Write
L<file>:<DWord> [cols]	Long, **DWord**, LBCD*	Read/Write

*Array types.

The number of array elements cannot exceed 16.

Ranges

PLC Model	File Number	Max. Word
Micrologix	3-255	255
All SLC	NA	NA
PLC5	NA	NA

Examples

Example	Description
L9:0	Word 0.
L9:10 [8]	8 Element array starting at word 10.
L15:0 [4] [5]	4 by 5 element array starting at word 0.

Micrologix PID Files

PID files are a structured type whose data is accessed by specifying a file number, an element and a field. The default data types are shown in **bold**.

Syntax	Data Type	Access
PD<file>:<element>.<field>	Depends on field	Depends on field

The following fields are allowed for each element. For the meaning of each field, refer to the PLC's documentation.

Element Field	Data Type	Access
SPS	**Word**, Short	Read/Write
KC	**Word**, Short	Read/Write
TI	**Word**, Short	Read/Write
TD	**Word**, Short	Read/Write
MAXS	**Word**, Short	Read/Write
MINS	**Word**, Short	Read/Write
ZCD	**Word**, Short	Read/Write
CVH	**Word**, Short	Read/Write
CVL	**Word**, Short	Read/Write
LUT	**Word**, Short	Read/Write
SPV	**Word**, Short	Read/Write
CVP	**Word**, Short	Read/Write
TM	**Boolean**	Read/Write
AM	**Boolean**	Read/Write
CM	**Boolean**	Read/Write
OL	**Boolean**	Read/Write
RG	**Boolean**	Read/Write
SC	**Boolean**	Read/Write
TF	**Boolean**	Read/Write
DA	**Boolean**	Read/Write
DB	**Boolean**	Read/Write
UL	**Boolean**	Read/Write
LL	**Boolean**	Read/Write
SP	**Boolean**	Read/Write
PV	**Boolean**	Read/Write
DN	**Boolean**	Read/Write
EN	**Boolean**	Read/Write

Ranges

PLC Model	File Number	Max. Element
Micrologix	3-255	255
All SLC	NA	NA
PLC-5	*	*

For more information, refer to **PLC5 PID Files.*

Examples

Example	Description
PD14:0.KC	Proportional gain of PD 0 file 14.
PD18:6.EN	PID enable bit of PD 6 file 18.

PLC5 PID Files

PID files are a structured type whose data is accessed by specifying a file number, an element and a field. The default data types are shown in **bold**.

Syntax	Data Type	Access
PD<file>:<element>.<field>	Depends on field	Depends on field

The following fields are allowed for each element. For the meaning of each field, refer to the PLC's documentation.

Element Field	Data Type	Access
SP	**Real**	Read/Write
KP	**Real**	Read/Write
KI	**Real**	Read/Write
KD	**Real**	Read/Write
BIAS	**Real**	Read/Write
MAXS	**Real**	Read/Write
MINS	**Real**	Read/Write
DB	**Real**	Read/Write
SO	**Real**	Read/Write
MAXO	**Real**	Read/Write
MINO	**Real**	Read/Write
UPD	**Real**	Read/Write
PV	**Real**	Read/Write
ERR	**Real**	Read/Write
OUT	**Real**	Read/Write
PVH	**Real**	Read/Write
PVL	**Real**	Read/Write
DVP	**Real**	Read/Write
DVN	**Real**	Read/Write

Element Field	Data Type	Access
PVDB	**Real**	Read/Write
DVDB	**Real**	Read/Write
MAXI	**Real**	Read/Write
MINI	**Real**	Read/Write
TIE	**Real**	Read/Write
FILE	**Word,** Short	Read/Write
ELEM	**Word,** Short	Read/Write
EN	**Boolean**	Read/Write
CT	**Boolean**	Read/Write
CL	**Boolean**	Read/Write
PVT	**Boolean**	Read/Write
DO	**Boolean**	Read/Write
SWM	**Boolean**	Read/Write
CA	**Boolean**	Read/Write
MO	**Boolean**	Read/Write
PE,	**Boolean**	Read/Write
INI	**Boolean**	Read/Write
SPOR	**Boolean**	Read/Write
OLL	**Boolean**	Read/Write
OLH	**Boolean**	Read/Write
EWD	**Boolean**	Read/Write
DVNA	**Boolean**	Read/Write
DVHA	**Boolean**	Read/Write
PVLA	**Boolean**	Read/Write
PVHA	**Boolean**	Read/Write

Ranges

PLC Model	File Number	Max. Element
Micrologix	*	*
All SLC	NA	NA
PLC-5	3-999	999

For more information, refer to* ***Micrologix PID Files**.*

Examples

Example	Description
PD14:0.SP	Set point field of PD 0 file 14.
PD18:6.EN	Status enable bit of PD 6 file 18.

Micrologix Message Files

Message files are a structured type whose data is accessed by specifying a file number, an element and a field. The default data types are shown in **bold**.

Syntax	Data Type	Access
MG<file>:<element>.<field>	Depends on field	Depends on field

The following fields are allowed for each element. For the meaning of each field, refer to the PLC's documentation.

Element Field	Data Type	Access
IA	**Word**, Short	Read/Write
RBL	**Word**, Short	Read/Write
LBN	**Word**, Short	Read/Write
RBN	**Word**, Short	Read/Write
CHN	**Word**, Short	Read/Write
NOD	**Word**, Short	Read/Write
MTO	**Word**, Short	Read/Write
NB	**Word**, Short	Read/Write
TFT	**Word**, Short	Read/Write
TFN	**Word**, Short	Read/Write
ELE	**Word**, Short	Read/Write
SEL	**Word**, Short	Read/Write
TO	**Boolean**	Read/Write
CO	**Boolean**	Read/Write
EN	**Boolean**	Read/Write
RN	**Boolean**	Read/Write
EW	**Boolean**	Read/Write
ER	**Boolean**	Read/Write
DN	**Boolean**	Read/Write
ST	**Boolean**	Read/Write
BK	**Boolean**	Read/Write

The following file numbers and maximum element are allowed for each model.

Ranges

PLC Model	File Number	Max. Element
Micrologix	3-255	255
All SLC	NA	NA
PLC5	*	*

For more information, refer to **PLC5 Message.*

Examples

Example	Description
MG14:0.TO	Ignore if timed out bit of MG 0 file 14.
MG18:6.CO	Continue bit of MG 6 file 18.

PLC5 Message Files

Message files are a structured type whose data is accessed by specifying a file number, an element and a field. The default data types are shown in **bold**.

Syntax	Data Type	Access
MG<file>:<element>.<field>	Depends on field	Depends on field

The following fields are allowed for each element. For the meaning of each field, refer to the PLC's documentation.

Element Field	Data Type	Access
ERR	**Short,** Word	Read/Write
RLEN	**Short,** Word	Read/Write
DLEN	**Short,** Word	Read/Write
EN	**Boolean**	Read/Write
ST	**Boolean**	Read/Write
DN	**Boolean**	Read/Write
ER	**Boolean**	Read/Write
CO	**Boolean**	Read/Write
EW	**Boolean**	Read/Write
NR	**Boolean**	Read/Write
TO	**Boolean**	Read/Write

Ranges

PLC Model	File Number	Max. Element
Micrologix	*	*
All SLC	NA	NA
PLC-5	3-999	999

**For more information, refer to Micrologix Message Files.*

Examples

Example	Description
MG14:0.RLEN	Requested length field of MG 0 file 14.
MG18:6.CO	Continue bit of MG 6 file 18.

Block Transfer Files

Block transfer files are a structured type whose data is accessed by specifying a file number, an element and a field. The default data types are shown in **bold**.

Syntax	Data Type	Access
BT<file>:<element>.<field>	Depends on field	Depends on field

The following fields are allowed for each element. For the meaning of each field, refer to the PLC's documentation.

Element Field	Data Type	Access
RLEN	**Word**, Short	Read/Write
DLEN	**Word**, Short	Read/Write
FILE	**Word**, Short	Read/Write
ELEM	**Word**, Short	Read/Write
RW	**Boolean**	Read/Write
ST	**Boolean**	Read/Write
DN	**Boolean**	Read/Write
ER	**Boolean**	Read/Write
CO	**Boolean**	Read/Write
EW	**Boolean**	Read/Write
NR	**Boolean**	Read/Write
TO	**Boolean**	Read/Write

Ranges

PLC Model	File Number	Max. Element
Micrologix	NA	NA
All SLC	NA	NA
PLC-5	3-999	1999

Examples

Example	Description
BT14:0.RLEN	Requested length field of BT 0 file 14.
BT18:6.CO	Continue bit of BT 6 file 18.

High-Speed Counter File (HSC)

The HSC files are a structured type whose data is accessed by specifying an element and a field. The default data types are shown in **bold**.

See Also: ***Function File Options***

Syntax	**Data Type**	**Access**
HSC:<element>.<field>	Depends on field	Depends on field

The following fields are allowed for each element. For the meaning of each field, refer to the PLC's documentation.

Element Field	**Default Type**	**Access**
ACC	**DWord**, Long	Read Only
HIP	**DWord**, Long	Read/Write
LOP	**DWord**, Long	Read/Write
OVF	**DWord**, Long	Read/Write
UNF	**DWord**, Long	Read/Write
PFN	**Word**, Short	Read Only
ER	**Word**, Short	Read Only
MOD	**Word**, Short	Read Only
OMB	**Word**, Short	Read Only
HPO	**Word**, Short	Read/Write
LPO	**Word**, Short	Read/Write
UIX	**Boolean**	Read Only
UIP	**Boolean**	Read Only
AS	**Boolean**	Read Only
ED	**Boolean**	Read Only
SP	**Boolean**	Read Only
LPR	**Boolean**	Read Only
HPR	**Boolean**	Read Only
DIR	**Boolean**	Read Only
CD	**Boolean**	Read Only
CU	**Boolean**	Read Only
UIE	**Boolean**	Read/Write
UIL	**Boolean**	Read/Write
FE	**Boolean**	Read/Write
CE	**Boolean**	Read/Write
LPM	**Boolean**	Read/Write
HPM	**Boolean**	Read/Write
UFM	**Boolean**	Read/Write
OFM	**Boolean**	Read/Write
LPI	**Boolean**	Read/Write
HPI	**Boolean**	Read/Write
UFI	**Boolean**	Read/Write
OFI	**Boolean**	Read/Write
UF	**Boolean**	Read/Write

Element Field	Default Type	Access
OF	**Boolean**	Read/Write
MD	**Boolean**	Read/Write

Ranges

PLC Model	File Number	Max. Element
Micrologix	N/A	254
All SLC	N/A	N/A
PLC5	N/A	N/A

Examples

Example	Description
HSC:0.OMB	Output mask setting for high speed counter 0.
HSC:1.ED	Error detected indicator for high speed counter 1.

Real-Time Clock File (RTC)

The RTC files are a structured type whose data is accessed by specifying an element and a field. The default data types are shown in **bold**.

See Also: ***Function File Options***

Syntax	Data Type	Access
RTC:<element>.<field>	Depends on field	Depends on field

The following fields are allowed for each element. For the meaning of each field, refer to the PLC's documentation.

Element Field	Data Type	Access
YR	**Word**, Short	Read/Write
MON	**Word**, Short	Read/Write
DAY	**Word**, Short	Read/Write
HR	**Word**, Short	Read/Write
MIN	**Word**, Short	Read/Write
SEC	**Word**, Short	Read/Write
DOW	**Word**, Short	Read/Write
DS	**Boolean**	Read Only
BL	**Boolean**	Read Only
_SET (for block writes)	**Boolean**	Read/Write

Ranges

PLC Model	File Number	Max. Element
Micrologix	N/A	254

PLC Model	File Number	Max. Element
All SLC	N/A	N/A
PLC5	N/A	N/A

Examples

Example	Description
RTC:0.YR	Year setting for real-time clock 0.
RTC:0.BL	Battery low indicator for real-time clock 0.

Channel 0 Communication Status File (CS0)

To access the communication status file for channel 0, specify a word and optionally a bit in the word. The default data types are shown in **bold**.

See Also:* *Function File Options

Syntax	Data Type	Access
CS0:<word>	Short, **Word**, BCD, DWord, Long, LBCD	Depends on <word> and <bit>
CS0:<word>/<bit>	**Boolean**	Depends on <word> and <bit>
CS0/bit	**Boolean**	Depends on <word> and <bit>

Ranges

PLC Model	File Number	Max. Element
Micrologix	N/A	254
All SLC	N/A	N/A
PLC5	N/A	N/A

Examples

Example	Description
CS0:0	Word 0.
CS0:4/2	Bit 2 word 4 = MCP.

For more information on CS0 words/bit meanings, refer to the Rockwell documentation.

Channel 1 Communication Status File (CS1)

To access the communication status file for channel 1, specify a word and optionally a bit in the word. The default data types are shown in **bold**.

See Also:* *Function File Options

Syntax	Data Type	Access
CS1:<word>	Short, **Word**, BCD, DWord, Long, LBCD	Depends on <word> and <bit>
CS1:<word>/<bit>	**Boolean**	Depends on <word> and <bit>
CS1/bit	**Boolean**	Depends on <word> and <bit>

Ranges

PLC Model	File Number	Max. Element
Micrologix	N/A	254
All SLC	N/A	N/A
PLC5	N/A	N/A

Examples

Example	Description
CS1:0	Word 0
CS1:4/2	Bit 2 word 4 = MCP

For more information on CS1 words/bit meanings, refer to the Rockwell documentation.

I/O Module Status File (IOS)

To access the I/O module status file, specify a word and optionally a bit in the word. The default data type for each syntax is shown in **bold**.

See Also: ***Function File Options***

Syntax	Data Type	Access
IOS:<word>	Short, **Word**, BCD, DWord, Long, LBCD	Depends on <word> and <bit>
IOS:<word>/<bit>	**Boolean**	Depends on <word> and <bit>
IOS/bit	**Boolean**	Depends on <word> and <bit>

Ranges

PLC Model	File Number	Max. Element
Micrologix	N/A	254
All SLC	N/A	N/A
PLC5	N/A	N/A

Examples

Example	Description
IOS:0	Word 0.
IOS:4/2	Bit 2 word 4.

For a listing of 1769 expansion I/O status codes, refer to the instruction manual.

Event Log Messages

The following information concerns messages posted to the Event Log pane in the main user interface. Consult the server help on filtering and sorting the Event Log detail view. Server help contains many common messages, so should also be searched. Generally, the type of message (informational, warning) and troubleshooting information is provided whenever possible.

Unable to read block on device. | Block start = '<address>', Status = <code>, Extended Status = <code>.

Error Type:
Warning

Possible Cause:
Node cannot be found or a duplicate node is detected.

Possible Solution:
Check the status and extended status codes returned by the PLC. The codes are displayed in hexadecimal.

Note:
This error message applies to local node errors. Status code errors in the low nibble of the status code indicate errors found by the local node. The driver continues to retry reading these blocks of data periodically. Errors found by the local node occur when the KF module cannot see the destination PLC on the network for some reason.

Unable to read block on device. Block deactivated. | Block start = '<address>', Status = <code>, Extended Status = <code>.

Error Type:
Warning

Possible Cause:

1. The address requested in the block does not exist in the PLC.
2. Processor is in program mode.

Possible Solution:
Check the status and extended status codes returned by the PLC. The codes are displayed in hexadecimal.

Note:
This error message applies to remote node errors. Status code errors in the high nibble of the status code indicate errors found by the PLC. These errors are generated when the block of data the driver is asking for is not available in the PLC. The driver does not ask for these blocks again after receiving this kind of error. This kind of error can be generated if the address does not exist in the PLC.

Unable to read block on device. | Block start = '<address>', Status = <code>, Extended Status = <code>.

Error Type:
Warning

Possible Cause:
The address requested in the block does not exist in the PLC.

Possible Solution:
Check the status and extended status codes returned by the PLC. The codes are displayed in hexadecimal.

Unable to read block on device. Block deactivated. | Block start = '<address>', Status = <code>, Extended Status = <code>.

Error Type:
Warning

Possible Cause:

1. The address requested in the block does not exist in the PLC.
2. Node cannot be found or a duplicate node is detected.

Possible Solution:
Check the status and extended status codes returned by the PLC. The codes are displayed in hexadecimal.

Note:
This error message applies to local node errors. Status code errors in the low nibble of the status code indicate errors found by the local node. The driver continues to retry reading these blocks of data periodically. Errors found by the local node occur when the KF module cannot see the destination PLC on the network for some reason.

Unable to read function file on device. | Function file = '<function file element>', Status = <code>, Extended Status = <code>.

Error Type:
Warning

Possible Cause:
Node cannot be found or a duplicate node is detected.

Possible Solution:
Check the status and extended status codes returned by the PLC. The codes are displayed in hexadecimal.

Note:
This error message applies to local node errors. Status code errors in the low nibble of the status code indicate errors found by the local node. The driver continues to retry reading this function file periodically. Errors found by the local node occur when the KF module cannot see the destination PLC on the network for some reason.

Unable to read function file on device. Block deactivated. | Function file = '<function file element>', Status = <status>, Extended Status = <status>.

Error Type:
Warning

Possible Cause:

1. The function file address requested in the block does not exist in the PLC.
2. Processor is in program mode.

Possible Solution:
Check the status and extended status codes returned by the PLC. The codes are displayed in hexadecimal.

Note:
This error message applies to remote node errors. Status code errors in the high nibble of the status code indicate errors found by the PLC. These errors are generated when the function file the driver is asking for is not available in the PLC. The driver does not ask for this function file again after receiving this kind of error. This kind of error can be generated if the function file address does not exist in the PLC.

Unable to read block on device. Framing error. | Block start = '<address>'.

Error Type:
Warning

Possible Cause:

1. Unexpected frame received.
2. Frame size mismatch.

Possible Solution:
The device is returning an invalid read response or one of unexpected size.

Unable to read function file on device. Framing error. | Function file = '<function file element>'.

Error Type:
Warning

Possible Cause:

1. Unexpected frame received.
2. Frame size mismatch.

Possible Solution:
The device is returning an invalid function file read response or one of unexpected size.

Unable to read block on device. Checksum error. | Block start = '<address>'.

Error Type:
Warning

Possible Cause:
There is a problem with the cable connecting the devices, causing noise and checksum errors.

Possible Solution:
Inspect cabling between the host PC and the device.

Unable to read function file on device. Checksum error. | Function file = '<function file element>'.

Error Type:
Warning

Possible Cause:
There is a problem with the cable connecting the devices, causing noise and checksum errors.

Possible Solution:
Inspect cabling between the host PC and the device.

Unable to read block on device. Slave sink/source full. | Block start = '<start address>'.

Error Type:
Warning

Possible Cause:
The slave device cannot accept anymore requests from the master. The client may be requesting data too fast.

Possible Solution:
The driver automatically polls and re-polls the slave to empty its source and make room for responses from requests previously in the full sink. If this error occurs too often, decrease the update rate on suspected tags.

Unable to read function file on device. Slave sink/source full. | Function file = '<function file element>'.

Error Type:
Warning

Possible Cause:
The slave device cannot accept anymore requests from the master. The client may be requesting data too fast.

Possible Solution:

The driver automatically polls and re-polls the slave to empty its source and make room for responses from requests previously in the full sink. If this error occurs too often, decrease the update rate on suspected function file tags.

Unable to read block on device. Slave source empty. | Block start = '<address>'.

Error Type:
Warning

Possible Cause:
The slave device does not have a response prepared for the data request. The slave poll delay may be set too short.

Possible Solution:
The driver automatically polls and re-polls the slave for a poll response. If this error occurs often, increase the slave poll delay on the channel.

Unable to read function file on device. Slave source empty. | Function file = '<function file element>'.

Error Type:
Warning

Possible Cause:
The slave device does not have a response prepared for the request of function file element. The slave poll delay may be set too short.

Possible Solution:
The driver automatically polls and re-polls the slave for a poll response. If this error occurs often, increase the slave poll delay on the channel.

Error writing to address. | Tag address = '<address>', Status = <status>, Extended Status = <status>.

Error Type:
Warning

Possible Cause:

1. Node cannot be found or a duplicate node is detected.
2. The address requested in the block does not exist in the PLC.
3. Processor is in program mode.

Possible Solution:
Check the status and extended status codes returned by the PLC. The codes are displayed in hexadecimal.

Note:

1. Status code errors in the low nibble of the status code indicate errors found by the local node. Errors found by the local node occur when the KF module cannot see the destination PLC on the network for some reason.
2. Status code errors in the high nibble of the status code indicate errors found by the PLC. These errors are generated when the block of data the driver is asking for is not available in the PLC.

Error writing to address. Framing error. | Tag address = '<address>'.

Error Type:

Warning

Possible Cause:

1. Unexpected frame received.
2. Frame size mismatch.

Possible Solution:

The device is returning an invalid write response or one of unexpected size.

Checksum error occurred writing to address. | Tag address = '<address>'.

Error Type:

Warning

Possible Cause:

There is a problem with the cable connecting the devices causing noise and checksum errors.

Possible Solution:

Inspect / correct cabling between the host PC and the device.

Error writing to address. Slave sink/source full. | Tag address = '<address>'.

Error Type:

Warning

Possible Cause:

The slave device cannot accept any more requests from the master. The client may be requesting data too fast.

Possible Solution:

The driver automatically polls and re-polls the slave to empty its source and make room for responses from requests previously in the full sink. If this error occurs often, decrease the update rate on suspected tags, not necessarily the tag being written.

Error writing to address. Slave source empty. | Tag address = '<address>'.

Error Type:

Warning

Possible Cause:
The slave device does not have a response prepared for the write request. The slave poll delay may be too short.

Possible Solution:
The driver automatically polls and re-polls the slave for a poll response. If this error occurs often, increase the slave poll delay on the channel.

Device timed out writing to address. | Tag address = '<address>'.

Error Type:
Warning

Possible Cause:
The device is not responding.

Possible Solution:
Inspect cabling between the host PC and the device. Verify the device is on and operating properly.

Unable to read block on device. | Block start = '<address>', Status = <code>, Extended Status = <code>.

Error Type:
Warning

Possible Cause:
Node cannot be found or duplicate node detected.

Possible Solution:
Check the status and extended status codes returned by the PLC. The codes are displayed in hexadecimal.

Note:
This error message applies to local node errors. Status code errors in the low nibble of the status code indicate errors found by the local node. The driver continues to retry reading these blocks of data periodically. Errors found by the local node occur when the KF module cannot see the destination PLC on the network for some reason.

Unable to read block on device. Block deactivated. | Block start = '<address>', Status = <code>, Extended Status = <code>.

Error Type:
Warning

Possible Cause:
The address requested in the block does not exist in the PLC.

Possible Solution:
Check the status and extended status codes returned by the PLC. The codes are displayed in hexadecimal.

Note:
This error message applies to remote node errors. Status code errors in the high nibble of the status code indicate errors found by the PLC. These errors are generated when the block of data the driver is asking for is not available in the PLC. The driver does not ask for these blocks again after receiving this kind of error. This kind of error can be generated if the address does not exist in the PLC.

Unable to read block on device. Device replied with a NAK. | Block start = '<address>'.

Error Type:
Warning

Unable to read function file on device. Device replied with a NAK. | Function file = '<function file element>'.

Error Type:
Warning

Unable to read block on device. Memory map error. | Block start = '<address>'.

Error Type:
Warning

Possible Cause:
There was an error writing to memory on the server.

Possible Solution:
Verify the device is online, communicating, and has a valid / uncorrupt memory map before trying again.

Unable to read function file on device. Memory map error. | Function file = '<function file element>'.

Error Type:
Warning

Possible Cause:
There was an error attempting to access the memory map on the device.

Possible Solution:
Verify the device is online, communicating, and has a valid / uncorrupt memory map before trying again.

Unable to read function file on device. Device replied with an unexpected NAK. Check the device link protocol. | Function file = '<function file element>'.

Error Type:
Warning

Possible Cause:

The server sent an invalid response or one of unexpected size. The server's protocol link settings may not match the device configuration.

Possible Solution:
Correct the device's link protocol settings to match that of the channel and try again.

Unable to read block on device. Device replied with an unexpected NAK. Check the device link protocol. | Block start = '<address>'.

Error Type:
Warning

Possible Cause:
The server sent an invalid response or one of unexpected size. The server's protocol link settings may not match the device configuration.

Possible Solution:
Correct the device's link protocol settings to match that of the channel and try again.

Unable to write to address on device. Packet length is out of range. | Tag address = '<address>', Packet length range = <min> to <max> (bytes).

Error Type:
Informational

Possible Cause:
There is an issue with the packet length.

Possible Solution:
Correct the packet length to be within the supported range and try again.

Unable to write to address on device. TNS is out of range. | Tag address = '<address>', TNS range = <min> to <max>.

Error Type:
Informational

Possible Cause:
An invalid address was provided.

Possible Solution:
Verify the address is within the supported range and try again.

Error Mask Definitions

B = Hardware break detected
F = Framing error
E = I/O error
O = Character buffer overrun
R = RX buffer overrun

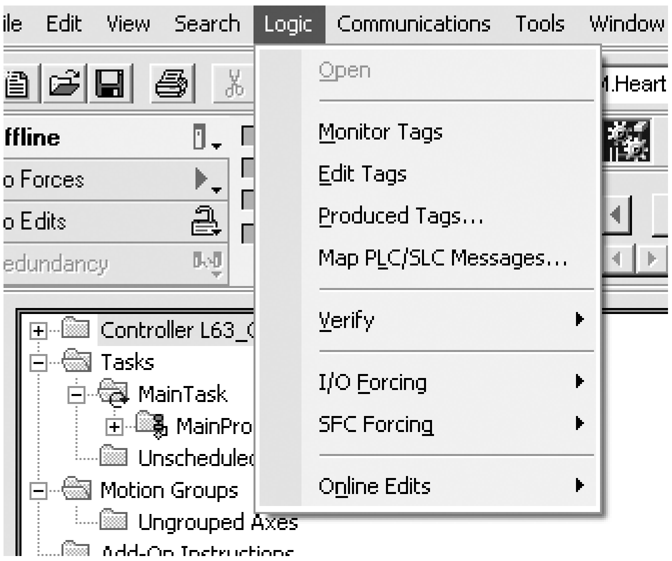
P = Received byte parity error
T = TX buffer full

Appendix: Communicating with RSLogix5000 Family Controllers

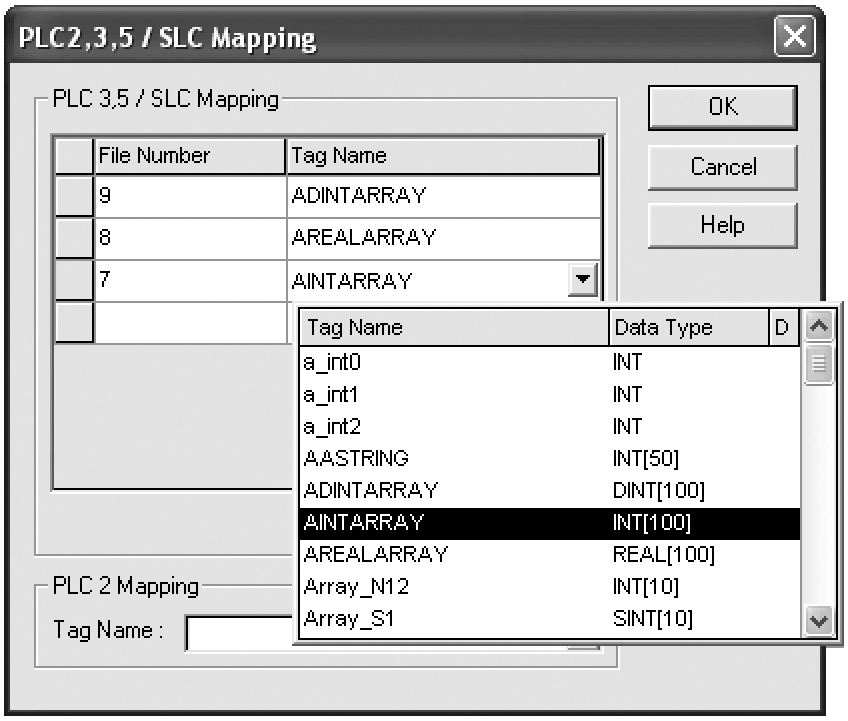
The Allen-Bradley DF1 Driver can be used to connect to an RSLogix5000 controller's serial port. Users must configure the RSLogix5000 Controller Tags to be mapped to the PLC/SLC data tables. For information on enabling communications, refer to the instructions below.

Note: A NULL Modem cable is required for this connection.

1. To start, open an RSLogix5000 development software application (such as RSLogix5000 or Studio 5000 Logix Designer). Then, select **Logic** | **Map PLC/SLC Messages**.



2. Next, map the tags to the different data files as shown in the image below.

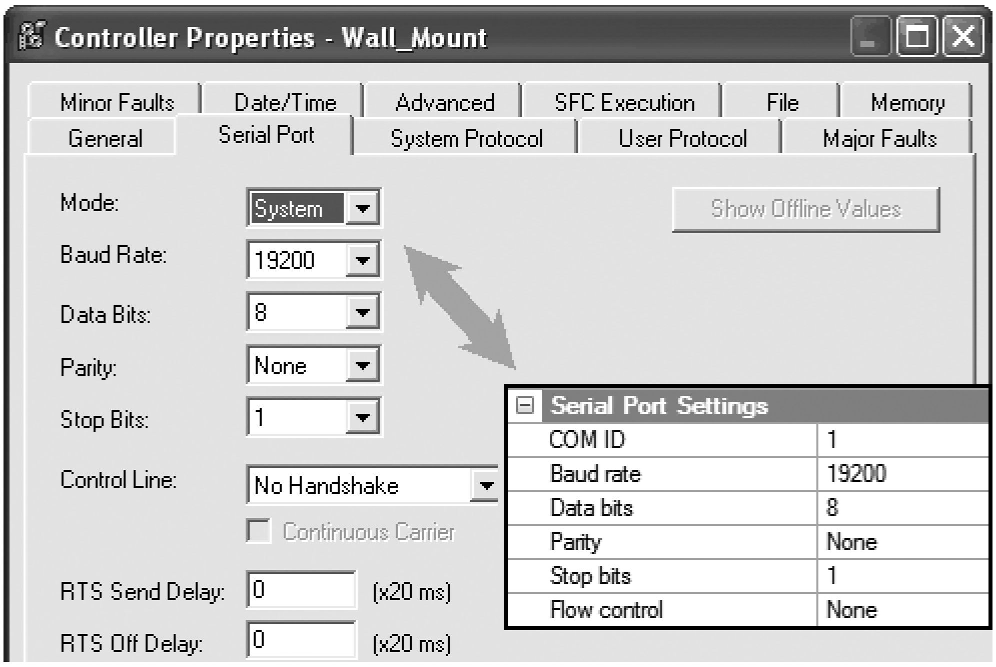


File Number	Tag Name
9	ADINTARRAY
8	AREALARRAY
7	AINTARRAY

Tag Name	Data Type	D
a_int0	INT	
a_int1	INT	
a_int2	INT	
AASTRING	INT[50]	
ADINTARRAY	DINT[100]	
AINTARRAY	INT[100]	
AREALARRAY	REAL[100]	
Array_N12	INT[10]	
Array_S1	SINT[10]	

3. Ensure that the Allen-Bradley DF1 Driver and controller have matching settings for the Serial Port and System Protocol (such as the baud rate, data bits, and parity properties shown in the images below).

Note: The Link Protocol should be set to **Full Duplex**.

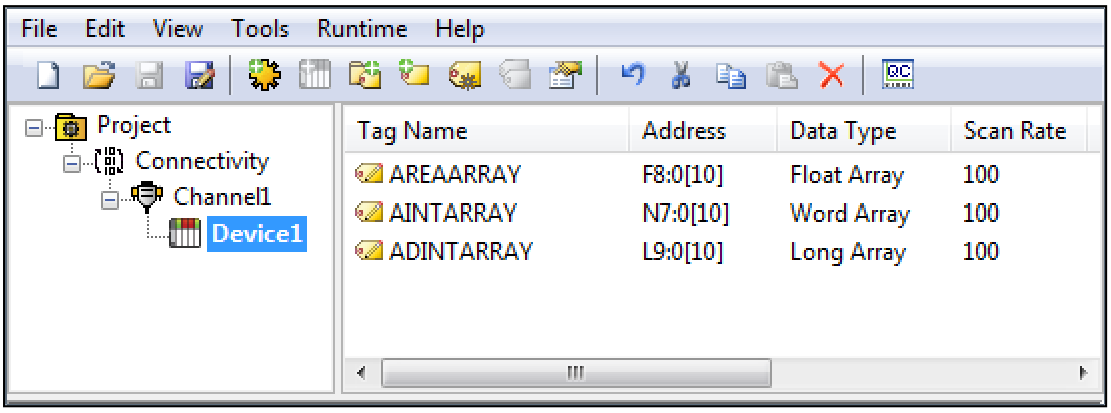


Mode: System
Baud Rate: 19200
Data Bits: 8
Parity: None
Stop Bits: 1
Control Line: No Handshake
Continuous Carrier
RTS Send Delay: 0 (x20 ms)
RTS Off Delay: 0 (x20 ms)

Serial Port Settings	
COM ID	1
Baud rate	19200
Data bits	8
Parity	None
Stop bits	1
Flow control	None

4. Next, create the desired tags within the Allen-Bradley DF1 Driver.

Note: The MicroLogix model is the only model that supports "L" data types, which are 32-bit data types equivalent to the controller's DINT data type.



Tag Name	Address	Data Type	Scan Rate
AREAARRAY	F8:0[10]	Float Array	100
AINTARRAY	N7:0[10]	Word Array	100
ADINTARRAY	L9:0[10]	Long Array	100

Resources

In addition to this user manual, there are a variety of resources available to assist customers, answer questions, provide more detail about specific implementations, or help with troubleshooting specific issues.

Knowledge Base
Whitepapers
Connectivity Guides
Technical Notes
Training Programs
Training Videos
Kepware Technical Support
PTC Technical Support

Index