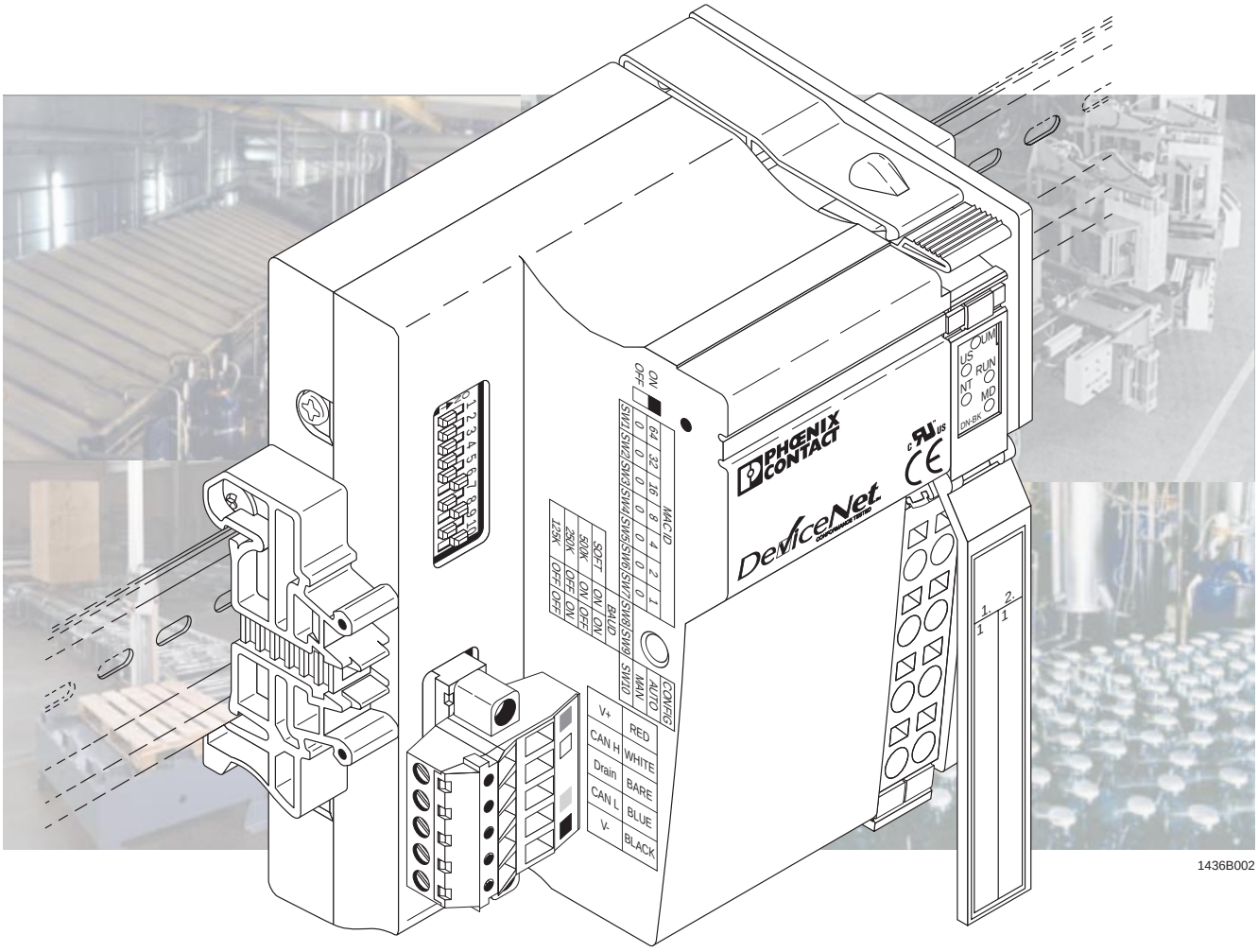
PHOENIX CONTACT
INSPIRING INNOVATIONS

Inline DeviceNet™ User Manual for IL DN BK3

Firmware 2.01 or Higher



1436B002

PHOENIX CONTACT
INSPIRING INNOVATIONS

Inline DeviceNet™ User Manual for IL DN BK3

Firmware 2.01 or Higher

Headquarters, U.S.
Phoenix Contact Inc.
P.O. Box 4100
Harrisburg, PA 17111-0100
Phone: 800-888-7388
717-944-1300
Fax: 717-944-1625
Email: info@phoenixcon.com
Website: www.phoenixcon.com

Technical Service
Phone: 800-322-3225

Headquarters, Canada
Phone: 905-890-2820
Fax: 905-890-0180

Type: IL DN BK3 UM E • PN: 5603326 • Rev E • Issued: August 2005 1436E

This Manual Contains Information on the

Inline DeviceNet™ Bus Coupler IL DN BK3

The information given herein is based on data believed to be reliable, but Phoenix Contact Inc. makes no warranties expressed or implied as to its accuracy and assumes no liability arising out of its use by others. This publication is not intended to be taken as a license to operate under, or recommendation to infringe upon, any patents.

Table of Contents

Table of Contents

Table of Contents

Appendixes

APPENDIX A
DeviceNet™ Object Classes, Message Types and Services

Appendixes

Appendixes

APPENDIX B
Electronic Data Sheet (EDS) File

APPENDIX C
Tips and Examples

APPENDIX D
Appliction Note 1597A for Editing the COS Mask

APPENDIX E
Terms, Definitions, Symbols and Conversion Tables

Preface
IL DN BK3 User Manual

Preface Contents

About this Manual

In order to guarantee the safe use of your device, we recommend that you read this manual carefully. The following notes give you information on how to use this manual.

a. User Requirements

The products described in this manual should be installed/operated/maintained only by qualified application programmers and software engineers, electricians or persons instructed by them, who are familiar with automation safety concepts and applicable national standards. Phoenix Contact assumes no liability for damage to any products resulting from disregard of information contained in this manual.

b. Purpose of this Manual

This manual contains the information necessary to understand and to configure an Inline IL DN BK3 DeviceNet™ bus coupler to it's connected I/O modules.

c. Who Should Use this Manual

This manual is written on the assumption that the reader possesses basic DeviceNet™ knowledge. Use this manual if you are responsible for configuring and installing an Inline system using DeviceNet™

d. Using this Manual

Specifications within the text of this manual are given in the International System of Units (SI), with English equivalents in parentheses. Fully capitalized words within the text indicate markings found on the equipment. Warnings, Cautions and Notes are used to emphasize critical instructions:

WARNING

An operating procedure, practice, etc., which, if not carefully followed, could result in personal injury.

CAUTION

An operating procedure, practice, etc., which, if not strictly observed, could result in damage to the equipment.

Note

Highlights important information about an operating procedure or the equipment.

For ease of looking for specific information in this manual, we have provided the following help:

- A main table of contents covering all subject matters is provided at the front of this manual.
- A table of contents covering information within a section or an appendix is provided at the front of the individual section or appendix.
- Appendixes covering specific topics are provided following the last section in the manual. Included as appendixes are a glossary of terms, abbreviations and symbols, and several conversion tables.

e. Getting Started

In the first section you are introduced to DeviceNet™ basics. The following sections contain general information that applies to all modules or module groups of the Inline family. Topics are for example:

- Example topology of a DeviceNet™ system
- Installation instructions
- Terminal wiring
- Common technical data

If you need specific information on a module refer to the module-specific data sheets.

f. Related Documentation

For additional information regarding DeviceNet™ in general, refer to the DeviceNet™ Specification. For specific information on the individual Inline modules, see the corresponding module-specific data sheets.

g. Current Documentation on the Internet

Make sure you are always working with the latest documentation published. The latest changes or additional information can be found on the Internet at:

http://www.phoenixcon.com (Info Service)

h. Statement of Legal Authority

This manual, including all illustrations contained herein, is copyright protected. Use of this manual by any third party in departure from the copyright provision is forbidden. Reproduction, translation, and electronic or photographic archiving or alteration requires the express written consent of Phoenix Contact. Violators are liable for damages.

Phoenix Contact reserves the right to make any technical changes that serve the purpose of technical progress.

Phoenix Contact reserves all rights in the case of patent award or listing of a registered design. External products are always named without reference to patent rights. The existence of such rights shall not be excluded.

i. Validity of Documentation

This manual contains descriptions of Inline modules that were available when this manual was published.

Phoenix Contact reserves the right to make any technical extensions and changes to the system that would serve the purpose of technical progress. Up to the time that a new manual revision is published, any updates or changes will be documented on the Internet at:

http://www.phoenixcon.com (Info Service)

DeviceNet™ is a trademark of Open DeviceNet Association
RSNetWorx™ is a trademark of Rockwell Software

SECTION 1
Introduction to the Inline DeviceNet™ Bus Coupler

Section 1 Contents

1.1 General

This section provides information about the DeviceNet™ bus coupler as part of DeviceNet™ and the Inline product family.

1.2 The DeviceNet™ System

DeviceNet™ is a communications link that transmits data between control systems (e.g., PLCs, PCs, VMEbus computers, robot controllers, etc.) and distributed industrial devices (such as switches, sensors, valve manifolds, motor starters, bar code readers, drives, displays, and operator interfaces) to network and eliminate expensive hard wiring.

DeviceNet™ has a linear structure. This structure consists of a main trunk line with drop lines routed to the networked devices. Power and signal is routed on the same network cable.

Communications is very flexible with DeviceNet™. Peer-to-Peer , Multi-Cast (one-to-many), Master/Slave, bit-strobe, and change-of-state (exception-based) communication options are available when using DeviceNet™.

The DeviceNet™ topology allows for the removal and replacement of powered devices from the network with no interruptions to the rest of the network.

1.3 Example Topology of a DeviceNet™ System

DeviceNet™ utilizes a linear bus topology. See Figure 1-1. Terminating resistors are required at each end of the trunk. Drops, made of trunk or drop cable, may be as long as 6 m (20 feet), and each drop can support one or more nodes.

The total length of trunk and drop cable that can be used on the network depends upon the type of cable and/or the data rate. See Table 1-1 when determining the length of the network based on the data rate. See the DeviceNet™ specification for more details on network lengths vs current draw.

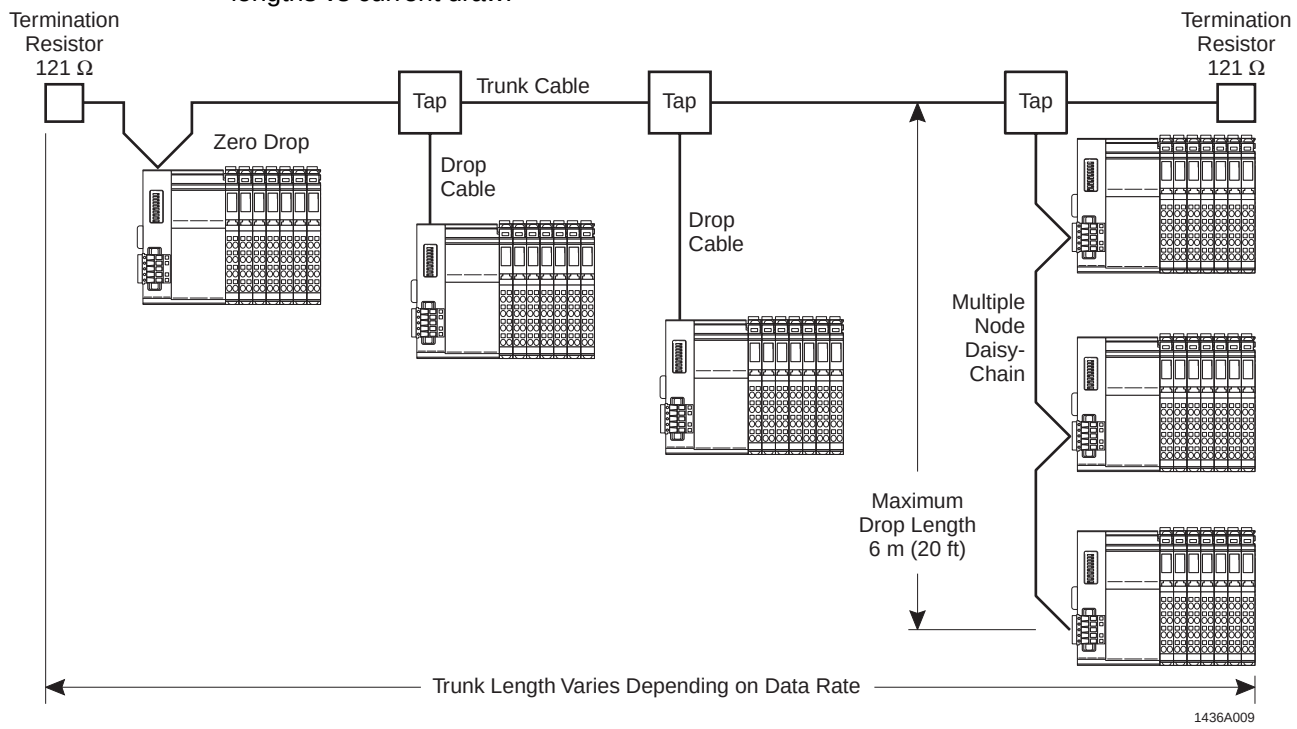


Figure 1-1. DeviceNet Topology Example

Table 1-1. Maximum Cable Length

Baud Rate	Baud Rate DIP-Switch Settings	Trunk Distance		Drop Length	
		Thick	Thin	Maximum	Cumulative
125k baud	8 OFF, 9 OFF	500 m (1640 ft)	100 m (328 ft)	6 m (20 ft)	156 m (512 ft)
250k baud	8 OFF, 9 ON	250 m (820 ft)	100 m (328 ft)	6 m (20 ft)	78 m (256 ft)
500k baud	8 ON, 9 OFF	100 m (328 ft)	100 m (328 ft)	6 m (20 ft)	39 m (128 ft)

1436A076

A. Bus Coupler Module

The first step in setting up a modular I/O station is to connect the bus coupler module to the DeviceNet™ cable. I/O modules may be installed branching off from these bus coupler modules, to create a local bus. The bus coupler module also supplies communications power to the connected I/O modules.

A breakdown of the supply voltage on the bus coupler module stops the communications to the modules connected to the bus coupler and causes an error message for the node.

Tasks of the bus coupler module:

- Coupling of DeviceNet™ and the Inline I/O modules
- Supplying the I/O modules with communications power
- Electrical isolation of the local I/O
- Providing diagnostic information from the connected I/O to DeviceNet™

B. Trunk Cable

The trunk cable is used to connect nodes to each other if no drops exist on the network. If drops are present, device taps will be linked via the trunk cable (See Figure 1-1). The maximum number of nodes on DeviceNet™ is limited to 64.

C. Drop Cable

The drop cable is used to connect drops to the trunk cable. This connection is made by using a network tap. Drop cable is typically thinner and more flexible than trunk cable.

D. Termination Resistors

DeviceNet™ requires a terminating resistor to be installed at each end of the trunk. These must be 121 ohms, 1% metal film, and have a power dissipation rating of 0.25 W.

E. Installation Local Bus (Loop 2)

The installation local bus cable is a 2-wire, unshielded cable that transmits I/O data, communications power and sensor power. See Figure 1-2. You can build a network of Loop 2 devices up to 200 meters (656 feet) with up to 63 Loop 2 devices. Note that each Loop 2 device will count towards the total number of devices (63) on the Inline station.

The installation local bus communicates to DeviceNet™ by using a bus coupler and a Loop 2 branch terminal. The branch terminal forms a subnetwork that connects Loop 2 machine mountable (enclosureless) I/O devices.

Note

General Loop 2 technical data can be found in Section 5 of this manual. For additional information on Loop 2, refer to the “Loop 2 System Manual” (IB L2 SYS PRO UM E).

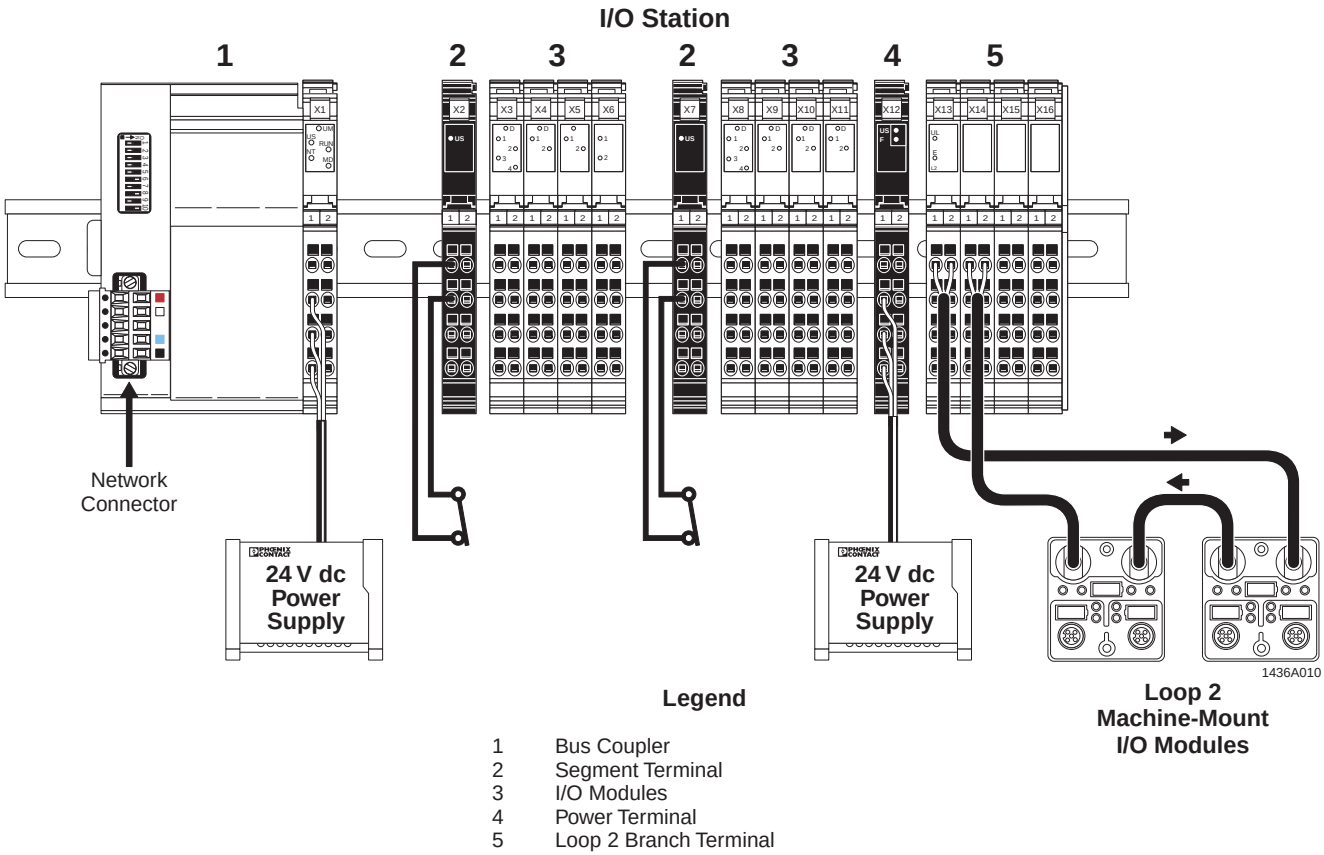


Figure 1-2. Example of a Basic Inline Station

F. Inline Product Family

The Inline product family is a modular automation system that can be integrated into an INTERBUS, DeviceNet™, CANopen, Profibus™, or MUX system. Inline modules are joined together to create functional units that meet the requirements of the application. See figure 1-2, shown with the DeviceNet™ bus coupler.

The control cabinet layout is designed according to function. Both communication and power routing is accomplished automatically by the physical interconnections between the I/O modules. Additional networking options permit the Inline station to branch out to various machine mounted I/O modules such as Loop 2, or AS-i devices.

1.4 DeviceNet™ Bus Coupler

A. Basic Functionality

The bus coupler, shown in Figure 1-3, provides an interface between DeviceNet™ and the Phoenix Contact family of Inline I/O modules. These modules include all of the basic digital, analog and special function. It also provides the required bus signal conditioning and the power supply for the connected station components. Bus couplers are currently available for the connection of copper cables.

The bus coupler provides the initial connection for the main supply, U_M, and the segment (I/O) supply, US, to the station. You can also provide the main supply U_M and the segment supply U_S using a power terminal and/or a segment terminal respectively.

All hardware and software implemented in this design conforms to either the DeviceNet™ specification issued by ODVA or the Inline family technical specifications.

1 Mounting Rail
2 Grounding Terminal (USLKG 5)
3* End Clamp (CLIPFIX 35)
4 Bus Coupler
5 End Plate
6* Power Connector
7* Labeling Field
8 Grounding Wire 0.2 to 1.5 mm² (24 to 16 AWG)
9 Twin COMBICON DeviceNet™ Network Connector
10 Electronic Data Sheet (EDS) Disk

* Included with IL DN BK3-PAC

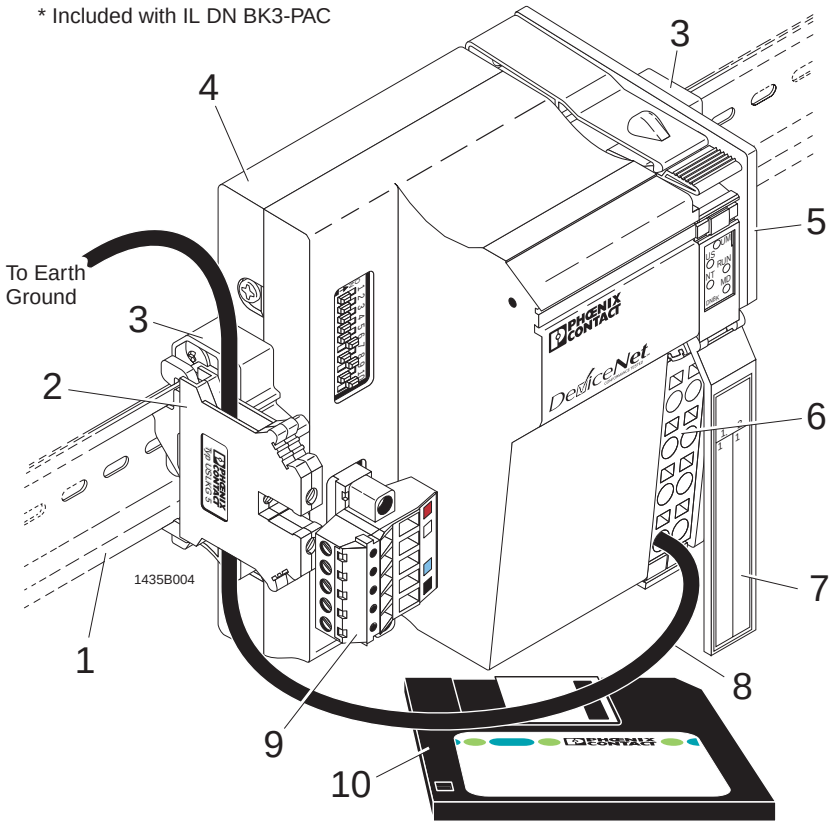


Figure 1-3. Features of the Inline DeviceNet™ Bus Coupler

B. Inline I/O Modules Supported

Module support includes the following:

- All standard Inline digital I/O
- All standard Inline analog I/O
- Inline power terminal
- Inline segment terminal
- Inline IP67 local bus (Loop 2) branch terminal
- Inline absolute encoder
- Inline incremental encoder module

Note

The "INC IN" module will appear as an analog module

- Inline motor starter
- Inline counter
- Inline safety relay module
- PCP and serial modules

C. Maximum Number of Devices

The maximum number of devices that you can connect to a bus coupler is determined by the following parameters:

- Up to 63 devices can be connected to a bus coupler.
- This number includes all the devices after the bus coupler with input or output data, i.e., the Inline modules and the modules for Loop 2.
- The bus coupler can supply a maximum current of 2 amps for logic power (communications).
- The current carrying capacity of the voltage jumpers is limited. For the limit values of the individual voltage jumpers, refer to Section 2 of this manual .

CAUTION

Observe the current consumption of each device for a given power supply. Power consumption specifications can be found in the product-specific data sheets. Also, refer to "Configuring an Inline Station" under "Tips on Working with Inline" in Appendix C of this manual.

SECTION 2
Setting Up an Inline DeviceNet™ Station

Section 2 Contents

Setting Up an Inline DeviceNet™ Station

Tables

2.1 General

This section provides information on the:

- Setting up an Inline station
- Installing the DeviceNet™ bus coupler and Inline I/O modules
- Wiring and connections
- Grounding considerations and requirements

2.2 Inline Electronics Base Units

The electronics base houses all the electronics for the Inline module. It also controls the routing of voltage and data signals.

A. Voltage and Data Routing

Voltage and data routing are located in the base. The advantage of this is that the elements of a station are not limited to a certain height. As a result, flat modules, e.g., carrier groups for receiving contactors, can be integrated into the station. As all the modules are snapped onto the mounting rail, there is a secure interface between all the station modules.

Voltage and data signals are transferred from module-to-module through dual-beam contacts having both a male end and a female end, see Figure 2-1. The male end of the contact protrudes out of the left-hand side of the module's housing as shown in Figure 2-2. The female ends extend into channels molded into the right-side of the module's housing. As modules are connected together, they form the various jumpers or busses

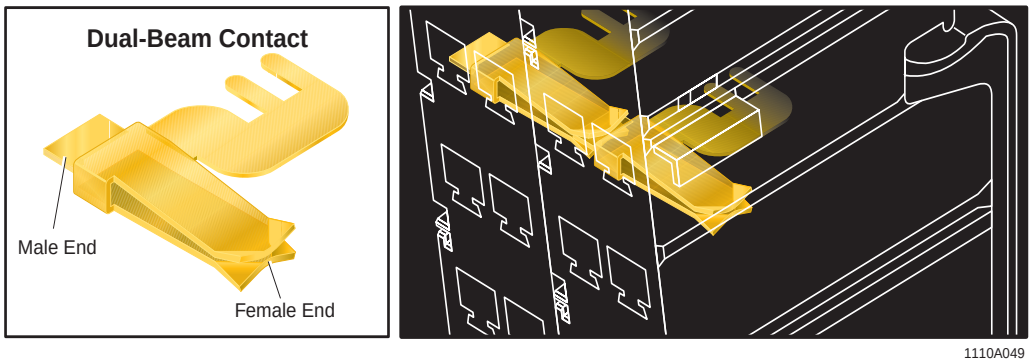


Figure 2-1. Voltage and Data Contacts

B. DIN-Rail Latches

With the connector removed, pressing the upper and lower DIN-Rail release levers on the electronic base unit at the same time releases the DIN-Rail latches, enabling the module to be removed from the DIN-Rail. See Figure 2-3.

C. Alignment Keys and Keyways

The electronics base of Inline products are designed with molded-in alignment keys and keyways. The alignment keys are located on the left-hand side of the module and the alignment keyways are located on the right-hand side of the module. See Figure 2-4. During installation, the keys on the module being installed slide into the keyways of the previously installed module to the left. The keys and keyways help to align the modules during installation and to hold them together after being installed. This ensures an excellent electrical connection of the voltage and data contacts between modules.

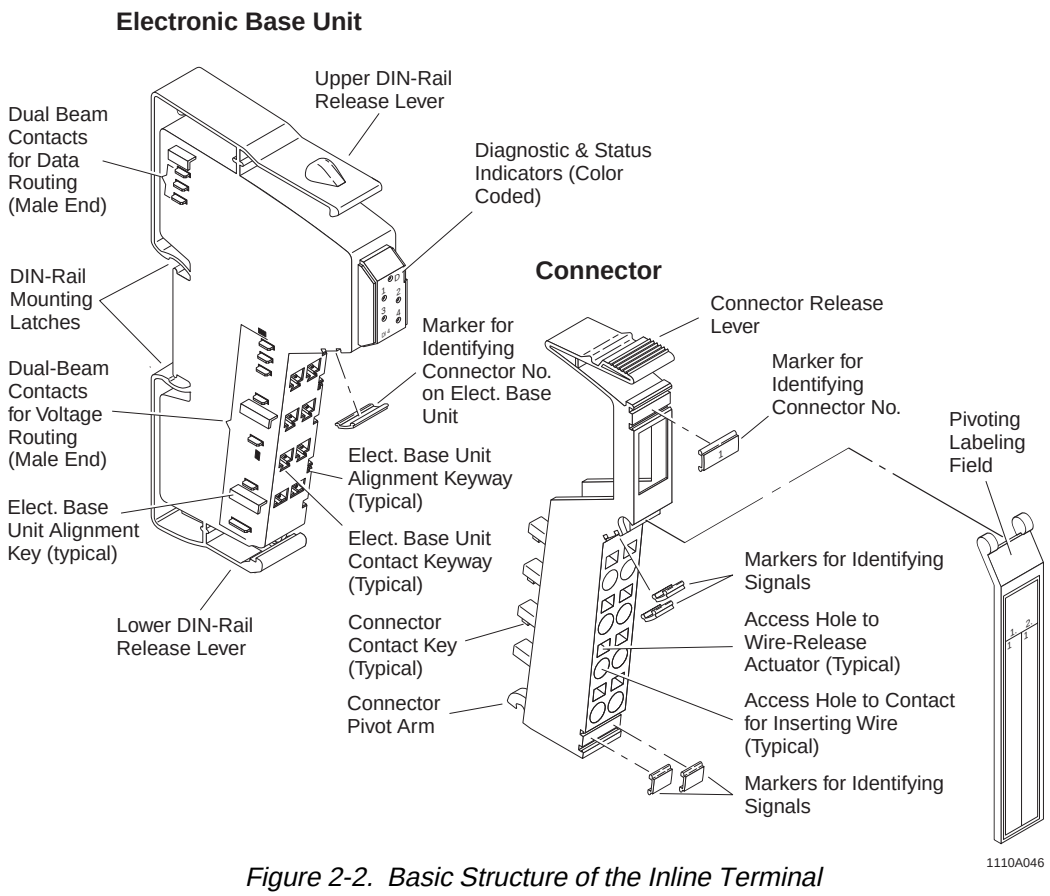


Figure 2-2. Basic Structure of the Inline Terminal

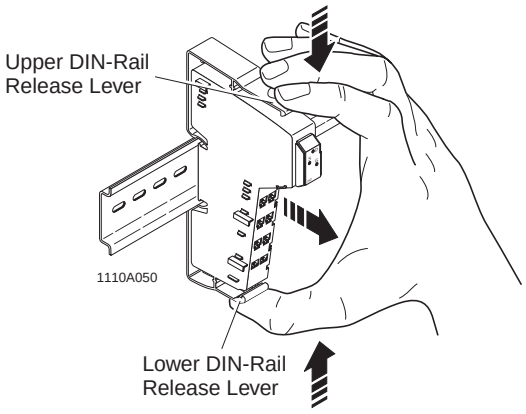


Figure 2-3. Using DIN-Rail Release Latches to Remove and Install Terminals

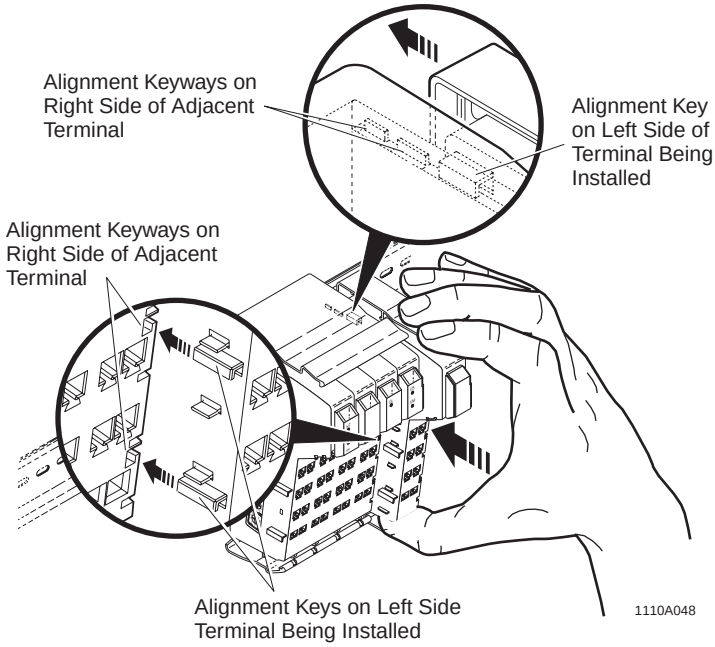


Figure 2-4. Alignment Keys and Keyways

2.3 Installation Instructions

2.3.1 Setting Up an Inline Station

An Inline station is set up by mounting the individual modules side by side on a mounting rail. No tools are required to install the modules. Figure 2-5 provides an example of an Inline station using the DeviceNet™ bus coupler. As you install the various modules onto the mounting rail, voltage and bus signal connections between the modules are created.

A. Mounting Rail

All Inline modules mount to a standard 35 mm DIN-rail. See Figure 2-5.

B. End Clamps

Install end clamps on both sides of the station as shown in Figure 2-5. The end clamps keep the modules from moving on the mounting rail and help to ensure the integrity of the connections between modules. Phoenix Contact recommends using CLIPFIX 35 (Order No. 30 22 21 8) end clamps.

C. End Plate

Install an end plate tight against the last module of the station. See Figure 2-5. The end plate covers the open contacts of the last module. It protects the user from dangerous voltages and the Inline station from ESD pulses. An end plate is supplied with the field bus coupler, it does not have to be ordered separately.

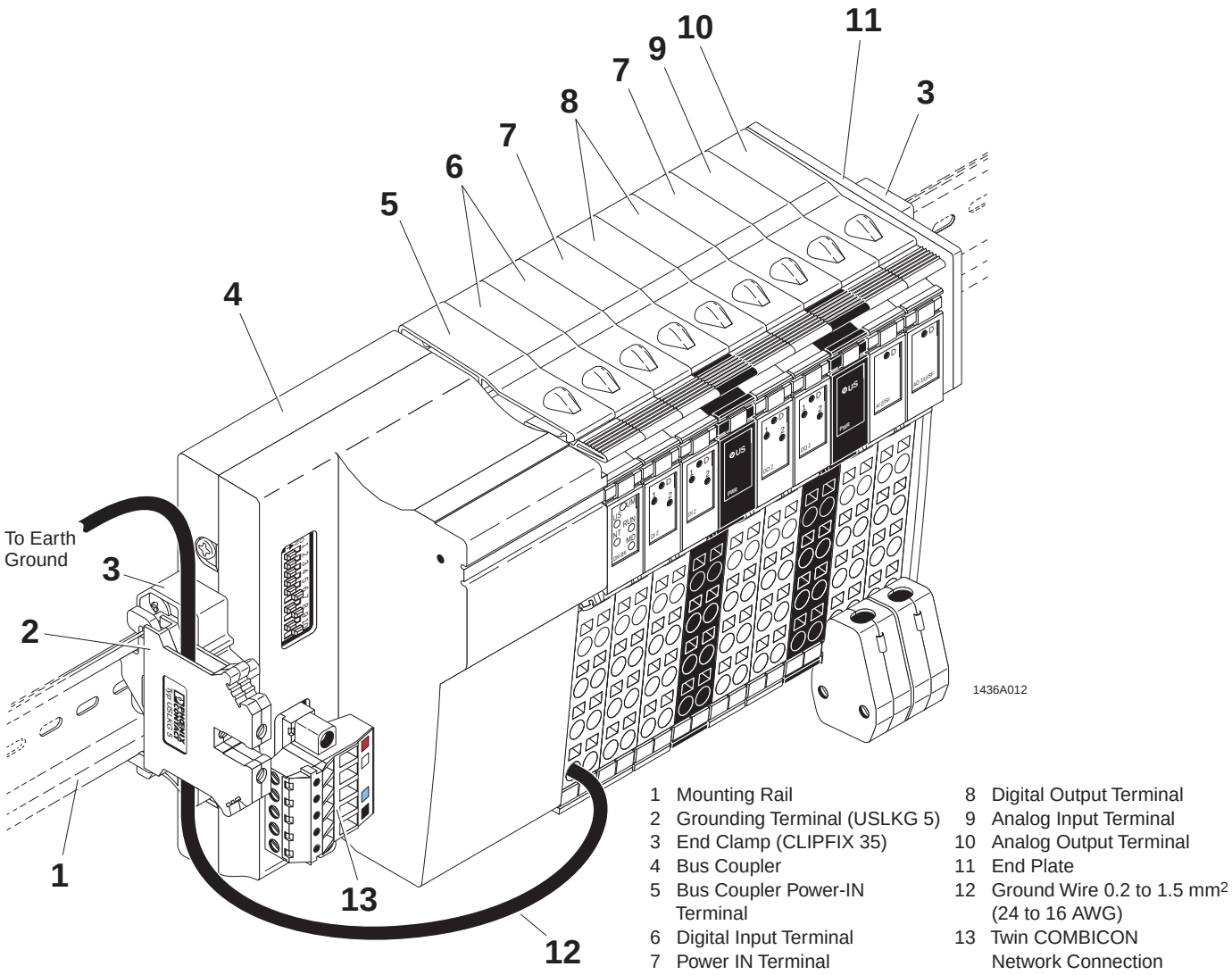


Figure 2-5. Example of an Inline Station with a DeviceNet™ Bus Coupler

2.3.2 Installing Inline Modules

Modules are installed from left to right on the mounting rail. Install modules to the mounting rail as described in the following steps.

⚠ WARNING

Never install or remove a module while power is being applied to any part of the Inline station. Before installing or removing a module, disconnect power to the entire station. Make sure work on the entire station is complete before switching power back on.

⚠ WARNING

Do not connect or disconnect any connector while power is ON. This can cause arcing that could damage bus coupler electronics or cause personal injury.

Note

If the electronics bases of the modules require labels/markers, check that labeling for each of the electronics bases is correct.

1. Attach the electronics base for the DeviceNet™ bus coupler to the mounting rail, by pushing the base straight in and onto the mounting rail as shown in Figure 2-6, View A.

⚠ CAUTION

Any twisting or pivoting of the electronics base during mounting may damage the electrical contacts. Always push straight in on the base .

2. Obtain the electronics base for the next Inline module to be installed in the station. Align the keys on the left side of the base being installed with the keyways on the right side of the bus coupler power terminal. See Figure 2-6, View B. Once aligned, push the new electronics base of the Inline module straight in and onto the mounting rail.
3. Continue attaching the electronics bases for the rest of the modules in the station as described in step 2.
4. When the electronic bases of all the modules are installed, obtain one of the end clamps provided with the bus coupler. Position the end clamp tight up against the left side of the bus coupler, then push it onto the mounting rail. See Figure 2-6, View C.
5. Obtain the end plate provided with the bus coupler. Position the end plate tight against the last module in the station, the push it onto the mounting rail. See Figure 2-6, View D.
6. Obtain the second end clamp provided with the bus coupler. Position the end clamp tight up against the end plate, then push it onto the mounting rail. See Figure 2-6, View D.

Note

Connectors can be installed onto the electronics bases with or without wires. Connectors can be installed in any order, however, we recommend starting from the left and working to the right.

Note

If connectors require markers, check that the labeling for each of the connectors is correct.

7. Obtain the two connectors supplied with the bus coupler. Using a blade-type screw-driver, attach the 5-pin, with 2x5 contacts, network connector to the bus coupler. See Figure 2-6, View E. Next, install the bus coupler power terminal connector by placing the pivot arm of the connector into the lower DIN-Rail release lever of the electronics base. See Figure 2-6, View F. Then pivot the connector upwards until the connector latch snaps into the locking tab of the upper DIN-Rail releases lever.
8. Continue installing the remaining connectors in the Inline station. Next, attach any markers or labeling fields to the connectors, as required. Then check all connectors again to ensure that they are properly identified.

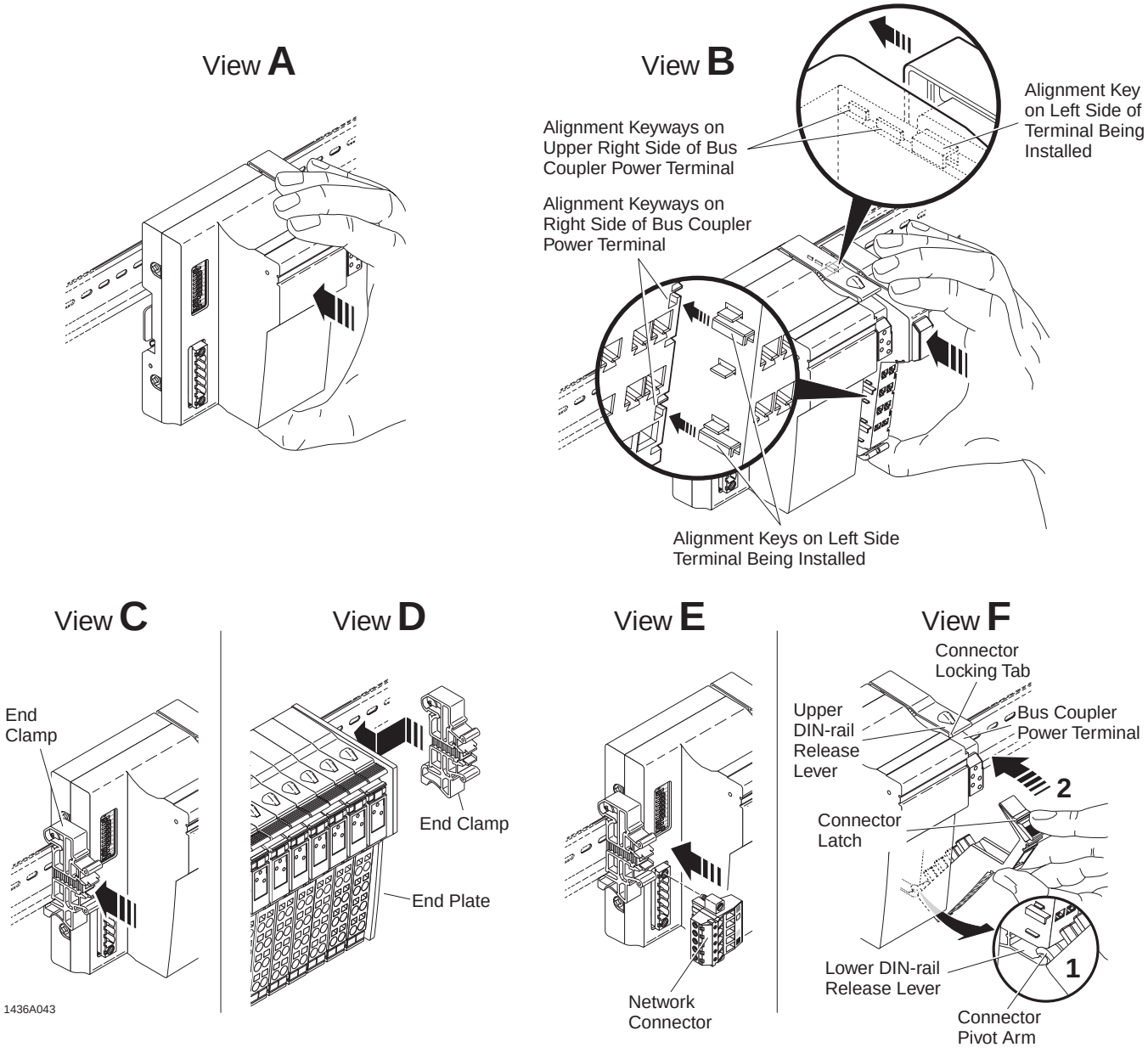


Figure 2-6. Installing Inline Terminals

2.3.3 Installing Inline Analog Modules

Analog modules are installed in the same manner as the other modules. However, to ensure optimal performance from the analog modules, i t is essential that you observe the following precautions.

CAUTION

Current flowing through the U_M and U_S voltage jumpers can cause a temperature rise inside of the analog module (including temperature modules). This can adversely affecting analog readings. To minimize heat build up, Phoenix Contact recommends that a power terminal be installed ahead of all analog modules. This will isolate the U_M and U_S voltage jumpers in the module from the main bus circuit (U_M, U_S and GND), therefore, reducing the amount of heat generated.

We also recommend that all analog modules be placed near the end of the Inline station and temperature modules (IB IL TEMP 2/UTH) be placed at the very end of the station.

2.3.4 Removing Inline Modules

When removing a module, proceed as described in the following steps. Figure 2-7 shows the removal of a 2-slot module. However, all modules can be removed in the same manner as that described in this procedure.

WARNING

Never install or remove any module while power is being applied to any part of the Inline station. Before installing or removing a module, disconnect power to the entire station. Make sure work on the entire station is complete before switching power back ON.

Do not connect or disconnect any connector while power is ON. This can cause arcing that could damage the modules electronics or cause personal injury.

CAUTION

You should never attempt to remove a module from the mounting rail with its connector(s) attached or if the connector is attached to the module immediately to the left of the module being removed. The protruding contacts on the left side of the module being removed will hit the connector of the adjacent module, preventing it from being removed. To ensure that the contacts do not become damaged during removal, we recommend that you remove the connectors from the module being removed and from the adjacent modules to the left and right of the module being removed.

1. Remove labeling fields, if installed, from the connector of the module(s) being removed and from the connectors of the neighboring modules. See Figure 2-7, View A.

2. Remove the connector(s) of the module being replaced by pressing down on the connector release lever, pivoting the connector downward, then lifting it out of the electronics base. See Figure 2-7, View B
3. Remove the connector(s) from both the left-side and right-side adjacent modules. See Figure 2-7, View C.

CAUTION

Any twisting or pivoting of the electronics base during removal may damage the electrical contacts. Always pull straight out on the base.

4. Remove the module's electronics base by pressing inward on both the upper and lower DIN-Rail release levers and pulling straight out. See Figure 2-7, View D.

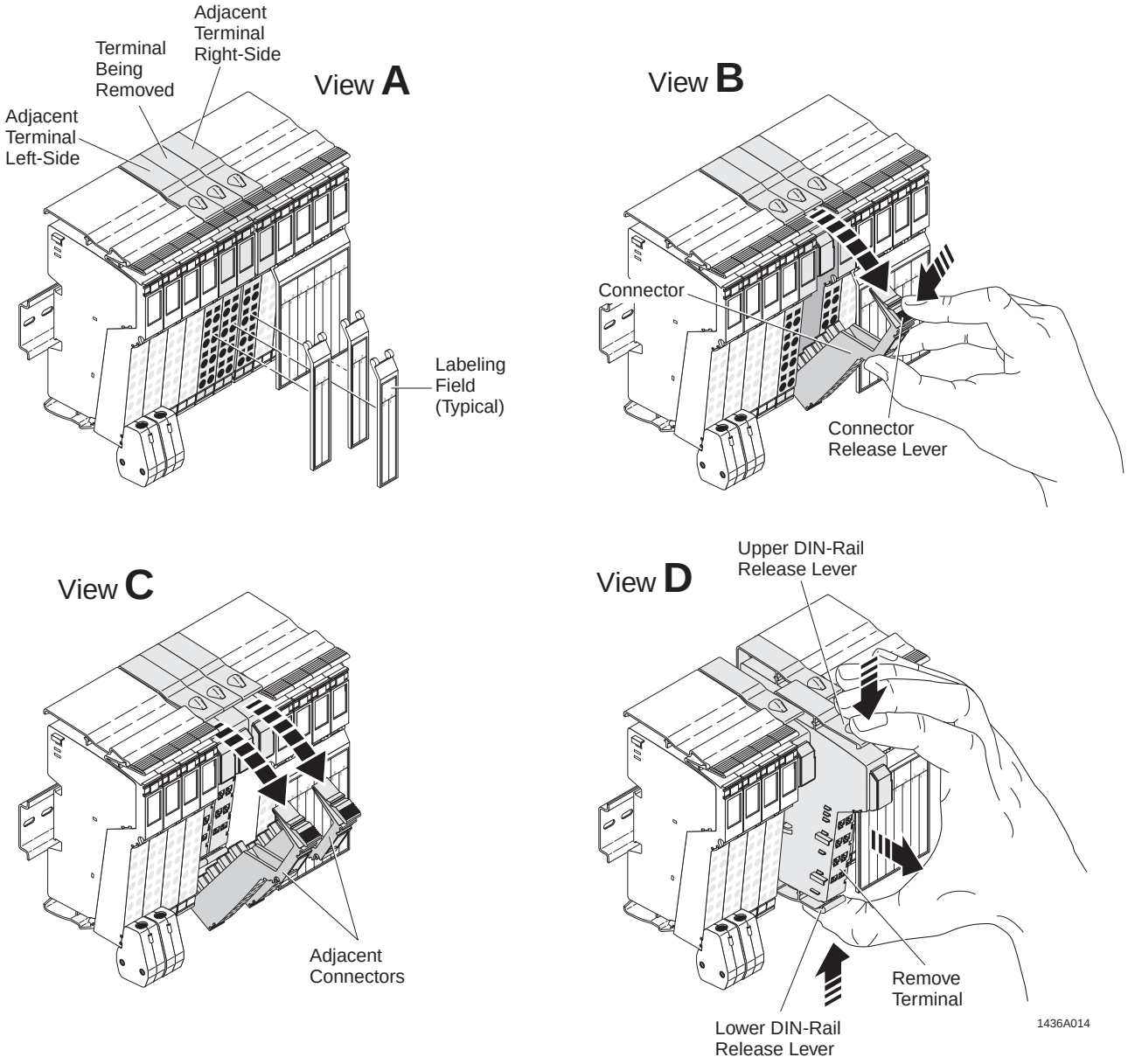


Figure 2-7. Replacing a Terminal

2.3.5 Replacing Inline Modules

If you need to replace a module(s) within the Inline station, first remove the module being replaced as described in Paragraph 2.3.4, “Removing Inline Modules”. Then install the replacement module as described in Paragraph 2.3.2, “Installing Inline Modules”.

2.3.6 Eliminating Inline Modules

When eliminating a module(s), it will be necessary to remove and reinstall all modules in the Inline station that are mounted to the right of the module being eliminated. To do this, follow the same basic procedures, including all cautions, as described in Paragraph 2.3.4 (“Removing Inline Modules”) and those in Paragraph 2.3.2 (“Installing Inline Modules”). The main difference from these procedures is that all connectors and modules to the right of the eliminated module(s) will have to be removed, repositioned, and then reinstalled onto the mounting rail.

⚠ WARNING

THIS EQUIPMENT IS SUITABLE FOR USE IN CLASS 1, DIVISION 2, GROUPS A, B, C, AND D OR NON-HAZARDOUS LOCATIONS ONLY.

WARNING – EXPLOSION HAZARD – SUBSTITUTION OF COMPONENTS MAY IMPAIR SUITABILITY FOR CLASS 1, DIVISION 2.

WARNING – EXPLOSION HAZARD – DO NOT DISCONNECT EQUIPMENT UNLESS POWER HAS BEEN SWITCHED OFF OR THE AREA IS KNOWN TO BE NON-HAZARDOUS.

2.4 Typical Inline Connector Numbering and Terminal Assignments

2.4.1 Inline Connector Numbering

Figure 2-8 shows the standard numbering scheme used for wiring all Inline connectors.

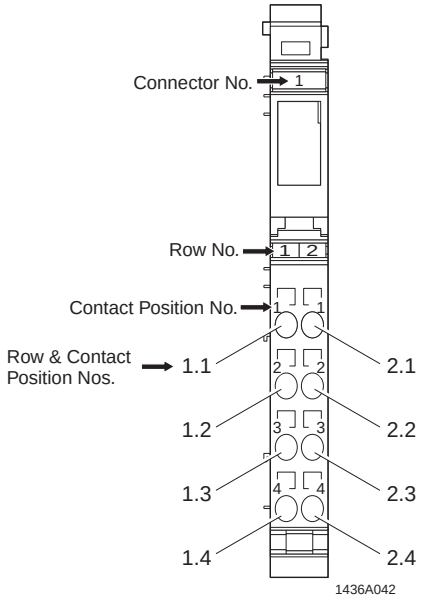


Figure 2-8. Typical Inline Connector Numbering

2.5 Bus Coupler Wiring

2.5.1 Bus Coupler Connector Numbering

Figure 2-9 shows the standard numbering scheme for wiring the bus coupler. View A shows connector numbering for the 5-pin with 2 rows of 5 contacts network connector. View B shows connector numbering for the 2-row power connector.

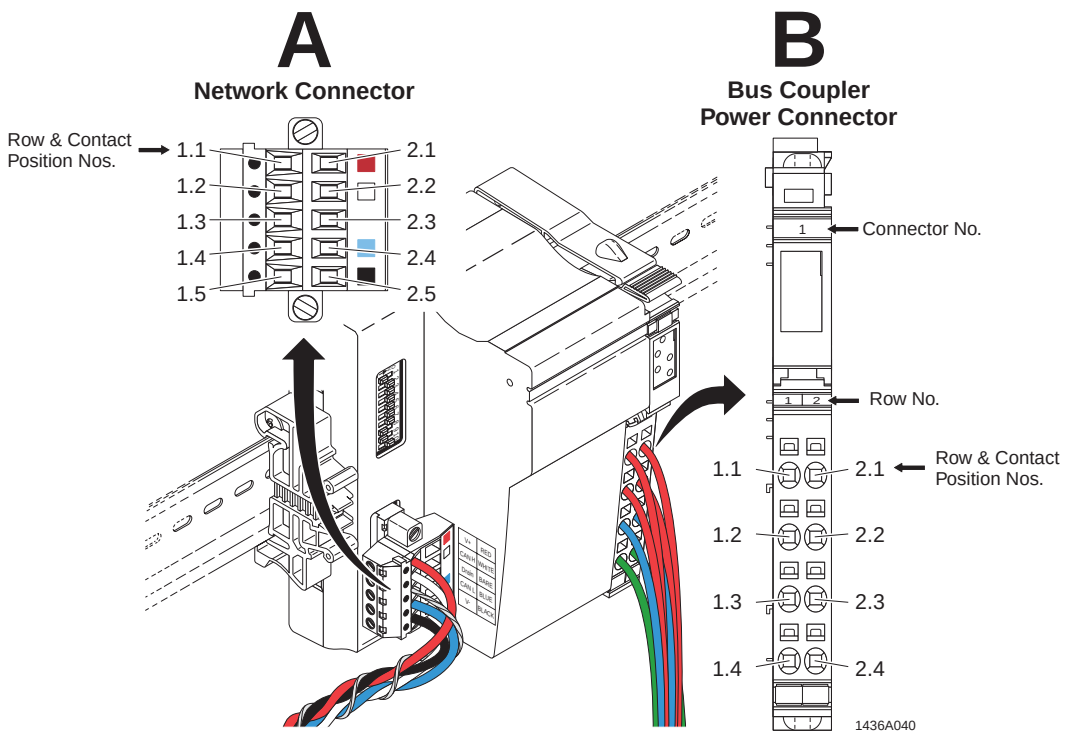


Figure 2-9. DeviceNet™ Bus Coupler Connector Numbering

2.5.2 Bus Coupler Terminal Assignments

Table 2-1 lists terminal assignments for the bus coupler connectors. Figure 2-10 shows a wiring schematic for the power connector that includes internal connections. Note that the bus coupler provides I/O power to the main (U_M) and segment (U_S) circuits for the Inline station. Logic power (communications) and analog power (U_{ANA} are also supplied by the bus coupler through the U_L connection.

Table 2-1. Bus Coupler Terminal Assignments

Bus Coupler Terminal Assignments			
Network Connector		**Power Connector**	
1.1 & 2.1	+V (+24 V), [red]	1.1	+24 V dc (Segment circuit U_S), [red]
1.2 & 2.2	CAN H (white)	1.2	+24 V dc (Logic and analog circuit U_L), [red]
1.3 & 2.3	Drain (bare)	1.3	U_L GND (Logic and analog ground), [blue]
1.4 & 2.4	CAN L (blue)	1.4	FE (Functional Earth ground), [green]
1.5 & 2.5	-V (+24 V RTN), [black]	2.1 & 2.2	+24 V dc (Main circuit U_M), [red]
		2.3	GND (U_S and U_M), [blue]
		2.4	FE (Functional Earth ground), [green]

1436A044

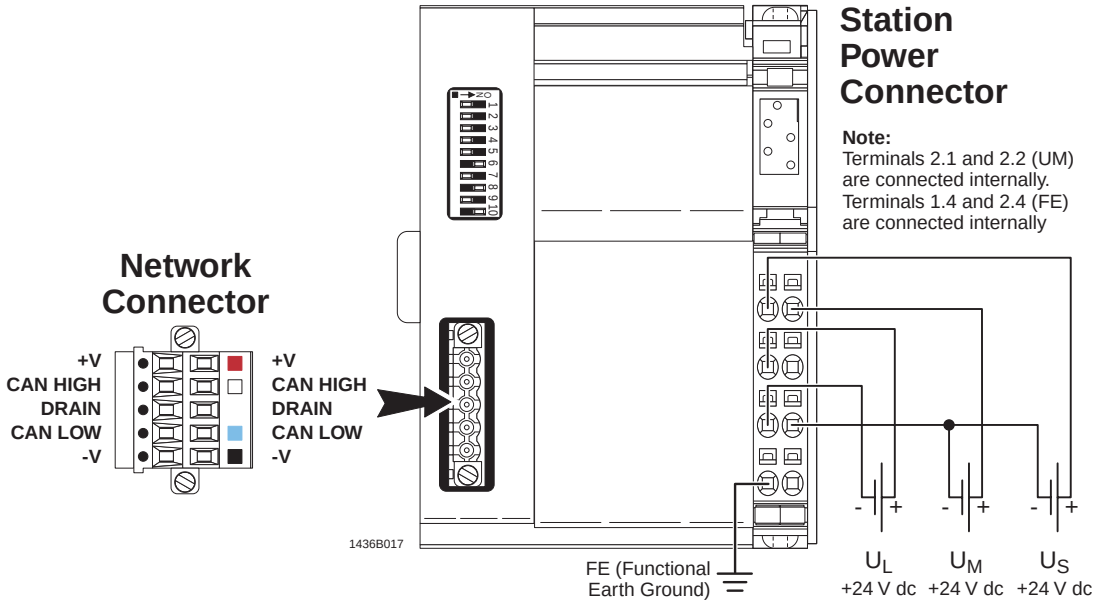


Figure 2-10. Network and Station Power Connections

2.5.3 Wire Type and Strip-length Requirements

For the bus coupler network connector, use standard 5-wire, DeviceNet™ cabling. The bus coupler power connector uses wires with diameters of 0.22 to 1.5 mm² (AWG 24 - 16) and utilizes spring-clamp technology for making gas-tight connections. See Figure 2-11 for strip-length requirements. Figure 2-12 shows how to install or remove wires from the terminal.

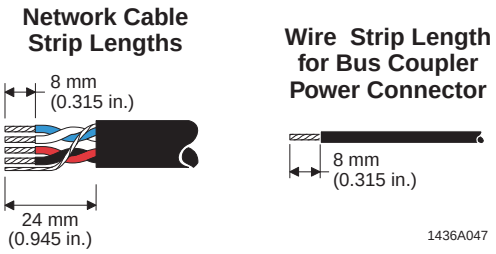


Figure 2-11. Strip-length Requirements

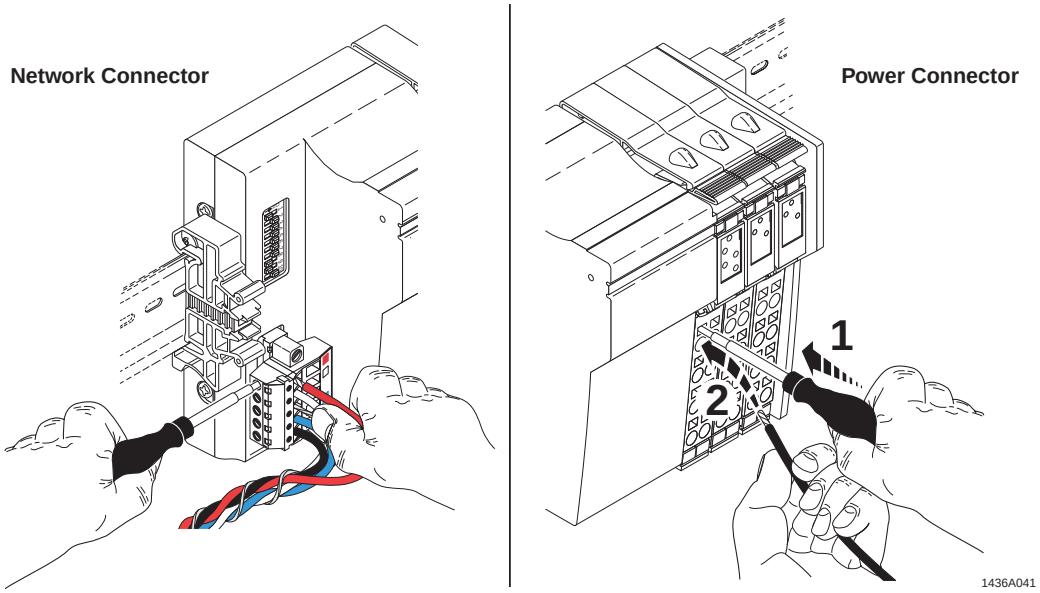


Figure 2-12. Installing and Removing Cables and Wires

2.6 Network Considerations

2.6.1 Wiring

Figure 2-13 shows an example of setting up a standard trunk and drop network.

A. Power

DeviceNet™ network power supplied to the bus coupler must be in the range of 11 volts DC to 28 volts DC. A supplied voltage of less than 11 volts DC will generate an Inline status error (see Section 4 of this manual). Each bus coupler requires 45-50 mA, at 24 volts DC, of the total available network current.

Note

To power Inline I/O module communications, DeviceNet™ power can be jumpered from the network connector, positions 1.1/2.1 (+V) and 1.5/2.5 (-V) to positions 1.2 (+24 V U_L) and 1.3 (U_L return) of the power connector. Inserting this jumper will draw additional current from the network. Refer to the paragraph titled “Bus Coupler Power Supplies” in this section and the I/O module specific data sheet to determine the exact additional U_L current draw.

B. Termination Resistors

A 121 Ω, 1% metal film, 0.25 W, termination resistor must be added at the extreme ends of the DeviceNet™ system. See Figure 2-13, Detail D.

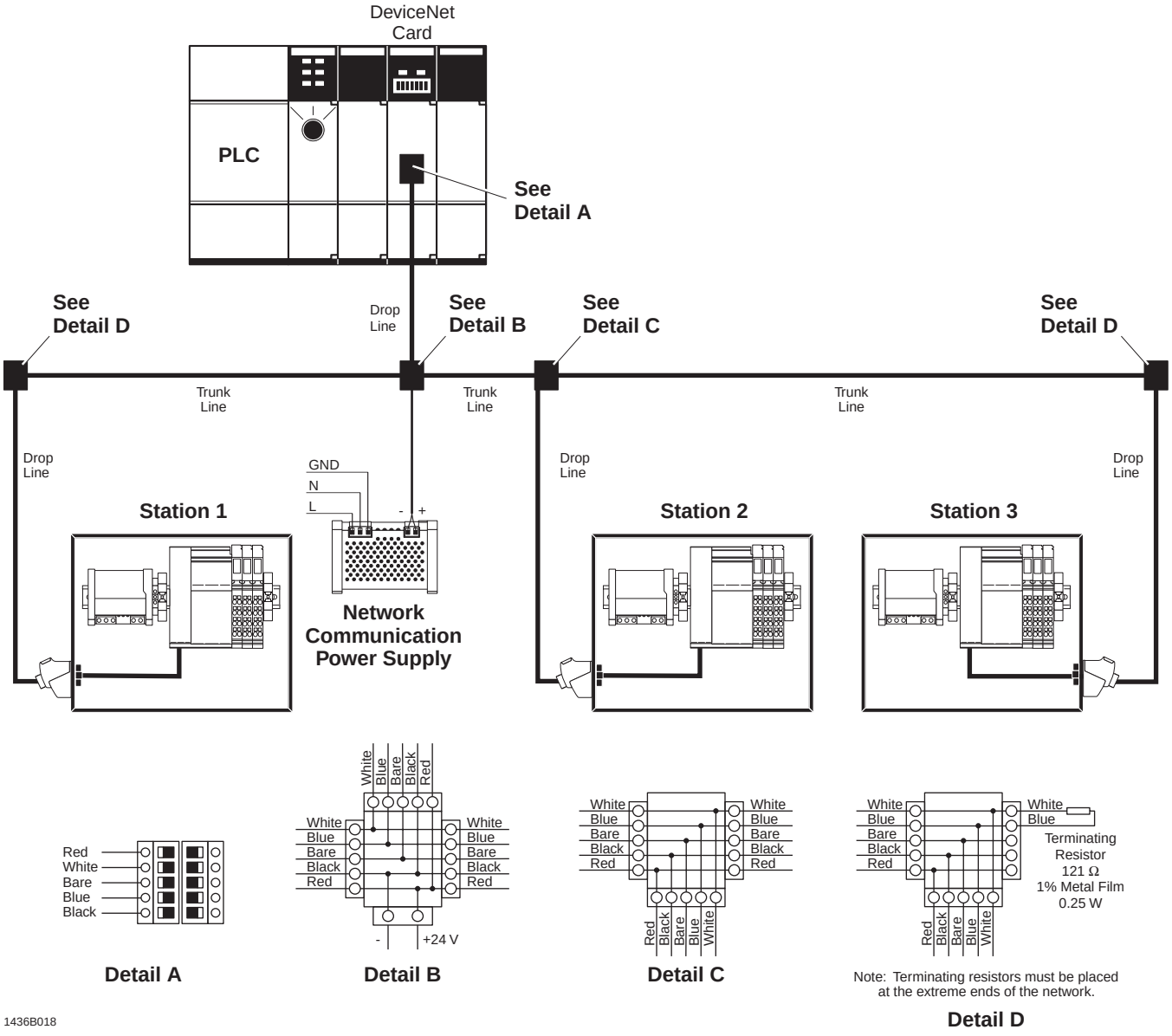


Figure 2-13. Example of Setting Up Network Communications and Network Power in an Inline DeviceNet™ System.

C. Daisy Chaining

Note

If daisy chaining is required, the network cable must be terminated to the network connector prior to installing the connector to the bus coupler.

The bus coupler 10-pin network connector contains two rows of contacts with 5 contacts in each row. Refer to Figure 2-10. These tw s of contacts provide a means for daisy chaining additional Inline stations. See Figure 2-14.

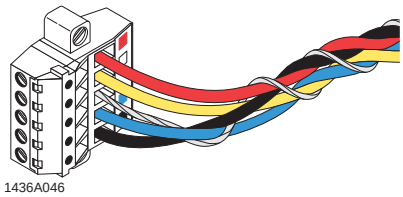


Figure 2-14. Termination of the Network Cable

2.6.2. Wire and Cable Central Grounding Specifications

WARNING

Grounding protects human beings and machines from dangerous voltages. To avoid dangers, it is essential that you always follow the grounding procedures outlined in the product-specific data sheet(s), this manual, and local codes.

DeviceNet™ communications power should only be grounded to earth at a single point. Typically this is done in the control cabinet where the DeviceNet™ power resides (+V, -V).

Grounding requirements for DeviceNet™ such as power return (-V), the drain (bare) wire, and the cable shields all need to be directly connected to earth ground. The ideal spot for making the earth ground connection for the entire network is in the physical center of the network. The ground connection should be made using either a 25 mm (1 in.) wide flat, braided, copper cable or a 10 mm^2 (8 AWG) wire. The wire run for this connection should be no longer than 3 meters (10 ft.).

Figure 2-15 shows an example of a system grounding scheme using a single power supply. Take note that the power supply's chassis is directly connected to earth ground.

CAUTION

All DeviceNet™ components must be grounded to avoid possible signal interferences.

The Inline bus coupler is both an isolated physical layer and I/O device. The only strictly DeviceNet™ grounding consideration is the drain (bare) wire. It can be terminated in position 1.3 or 2.3 of the network connector. Refer Figure 2-10.

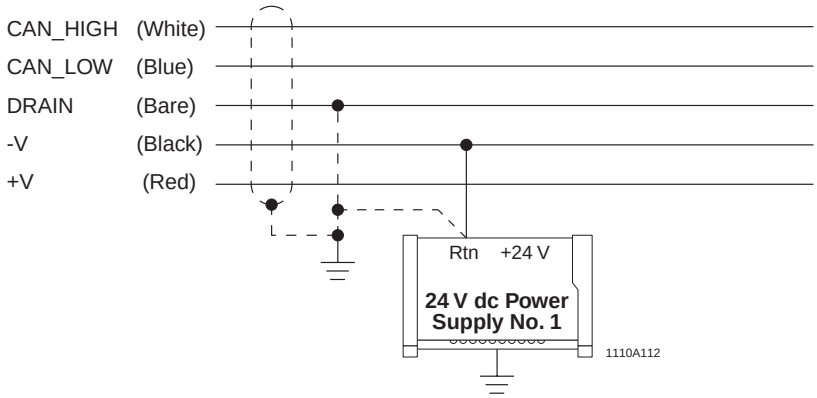
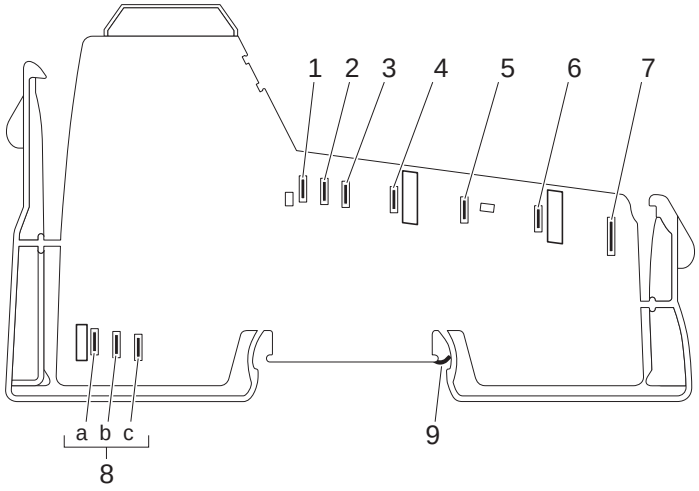


Figure 2-15. DeviceNet™ System Grounding Scheme

2.7 Inline Station Considerations

2.7.1 Bus Coupler Power Supplies

The bus coupler has 3 external power supply connections U_L (logic), U_S (segment) and U_M (main). The 7.5 volt internal U_L (communications) supply and the +24 volt U_{ANA} (analog) supply are derived from the external +24 volt U_L. The +24 volt (U_L) external power supply can be connected to DeviceNet™ or another external supply. Contact positions and current ratings are shown in Figure 2-16.



Item No.	Function & Designation		Voltage to Contact No. 6 (GND)		Current Maximum
			Minimum	Maximum	
Voltage Jumper					
1	7.5 V	U_{L+}	7.5 V dc*	7.87 V dc*	2 A
2	24 V	U_{ANA}	19.2 V dc*	30 V dc*	0.5 A
3	GNDL	U_L	0 V*	0 V dc*	2.5 A
4	24 V	U_S	19.2 V dc	30 V dc	8 A
5	24 V	U_M	19.2 V dc	30 V dc	8 A
6	GND	GND	0 V*	0 V*	8 A
7	FE	FE	Not Defined	Not Defined	Not Defined
9	FE Spring				
Data Jumper					
8a	Bus Signal				
8b	Bus Signal				
8c	Reserved				

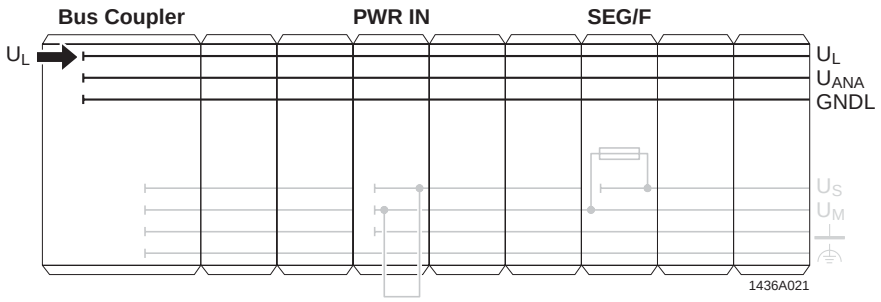
1110A062

* Contact no. 3 is reference potential for the logic. Contact no. 6 is reference potential for the I/O. If there is no electrical isolation betweeen logic and I/O, both have the same potential

Figure 2-16. Current and Voltage Distribution

A. +7.5 Volt Communications (U_L) Power Supply

Communications power starts at the bus coupler and flows through all module voltage contacts to form the U_L bus (voltage jumper) of the Inline station. See Figure 2-17.



Legend

PWR IN	Power Terminal
SEG/F	Segment Terminal With Fuse

Figure 2-17. Logic (U_L) and Analog Circuit (U_{ANA})

Notes

The maximum current carrying capacity of this U_L circuit is 2 A. Once this current limit is reached, a new station must be created using a new bus coupler.

Communications power is not electrically isolated from the +24 V input voltage for the bus coupler or power must be reapplied using an IB IL PWR IN/R terminal.

B. + 24 Volt Analog (U_{ANA}) Power Supply

Analog power starts at the bus coupler and flows through all module voltage contacts to form the U_{ANA} bus (voltage jumper) of the Inline station. See Figure 2-17. Analog I/O signals for Inline modules are supplied by this analog power supply.

Notes

U_{ANA} analog voltage is generated by the external U_L power supply.

The maximum current carrying capacity of U_{ANA} is 0.5 A. Once this current limit is reached, a new station must be created using a new bus coupler.

C. +24 Volt Main Circuit U_M Power Supply

Main power starts at the bus coupler and flows through all module voltage contacts to form the U_M bus (voltage jumper) of the Inline station. See Figure 2-18. Additional power terminals must be installed to create multiple main U_M circuits if any of the following conditions or requirements occur:

- The combined current carrying capacity of both the Main (U_M) and the segment (U_S) exceed 8 amps. Note that ground is common between U_M and U_S.
- Electrical isolation from the previous circuit is required
- A different voltage range needs to be created (120 volts AC for example).

Notes

Initiators having short-circuit protection or those that do not need protection can be supplied by the main (U_M) circuit.

The main (U_M) circuit can be used to supply any module that does not need to be isolated in the event of an emergency stop.

Independent segment(s) (functional group of modules) can be created within the main circuit by using a segment terminal(s) to tap the main (U_M) power. For example: a group of digital outputs connected to the same e-stop.

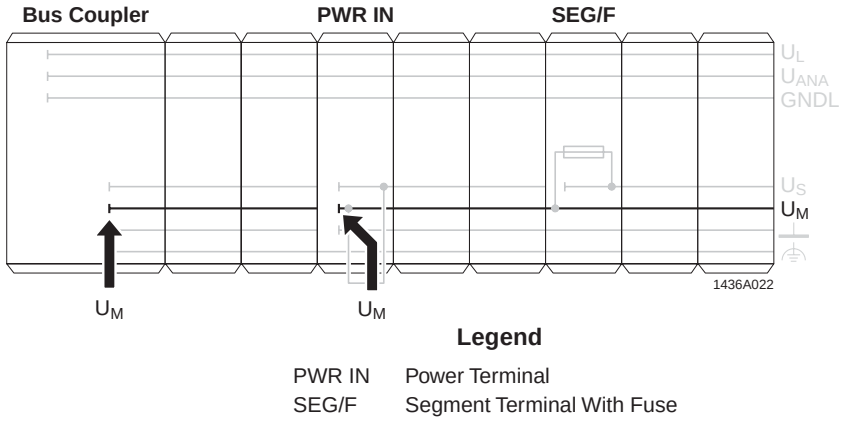
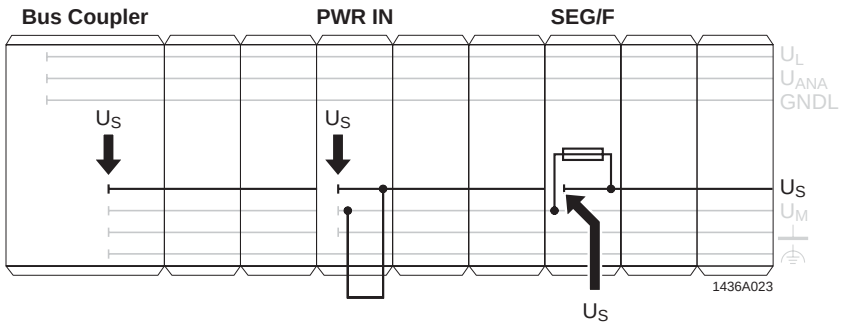


Figure 2-18. Main (U_M)Circuit

D. +24 Volt Segment Circuit U_S Power Supply

Segment power U_S (I/O power) starts at the bus coupler or a supply terminal and flows through all module voltage contacts to form the U_S bus (voltage jumper) of the Inline station. See figure 2-19. Additional power terminals must be installed to create multiple segment U_S circuits if any of the following conditions or requirements occur:

- The combined current carrying capacity of both the Main (U_M) and the segment (U_S) exceed 8 amps. Note that ground is common between U_M and U_S.
- Electrical isolation from the previous circuit is required.



Legend

PWR IN	Power Terminal
SEG/F	Segment Terminal With Fuse

Figure 2-19. Segment (U_S) Circuit

Note

The segment circuit U_S can be used to supply any module that needs to be isolated from the main U_M circuit.

Segment terminals do not exist for AC power. 120/230 V AC power terminals must be used to create a 120/230 V AC circuit.

Independent segment(s) (functional group of modules) can be created within the main circuit by using a segment terminal(s) to tap the main (U_M) power. For example: a group of digital outputs connected to the same e-stop.

E. Setting Up Inline Station Power Supplies

Figure 2-20 shows examples of an Inline station using either single or dual power supplies. The single power supply option can be used in those cases where local I/O communications power (U_L) can be connected to both U_S and U_M. The dual power supply option can be used when local I/O communications power (U_L) needs to be isolated from U_M/U_S (I/O power).

Power for U_L (shown in Figure 2-20) could be supplied by one of the following methods:

- Connect DeviceNet™ power to U_L
- Use a separate supply for U_L
- Power U_L with the U_S and/or U_M supply

Note

U_M/U_S should be installed and protected independently of the main DeviceNet™ supply. Power supply isolation will allow DeviceNet™ communications to continue during an I/O device failure.

⚠ CAUTION

Use power supplies with safe isolation. Use power supplies that ensure safe isolation between primary and secondary circuit (according to EN 50178).

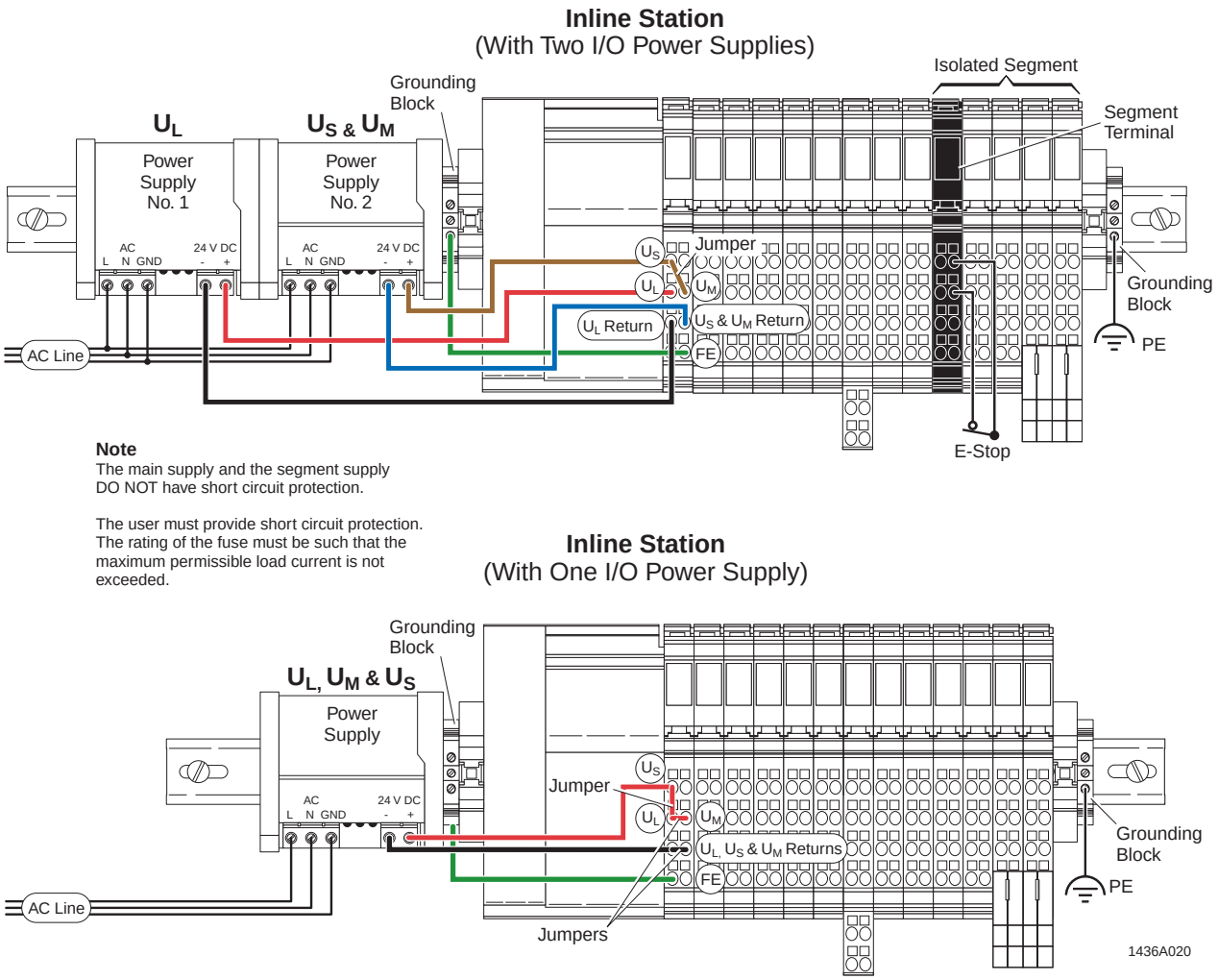


Figure 2-20. Setting Up Power in an Inline Station

2.7.2 Local Inline Station Earth Ground Considerations

A. Functional Earth Ground (FE) Voltage Jumper (Bus)

Note

The purpose of Function Earth (FE) ground is to discharge interferences. It does not provide shock protection for human beings.

The Inline station must be mounted onto a DIN-Rail that is connected to earth ground. The earth ground path, referred to as a Functional Earth (FE) ground, is made between the bus coupler and DIN-Rail by two grounding tab located on the back of the module. See Figure 2-21. From the bus coupler, the FE ground flows through all module voltage contacts to form the FE bus (voltage jumper) of the Inline station. See figure 2-22.

All other Inline modules are automatically grounded through the FE voltage jumper (bus) as they are installed.

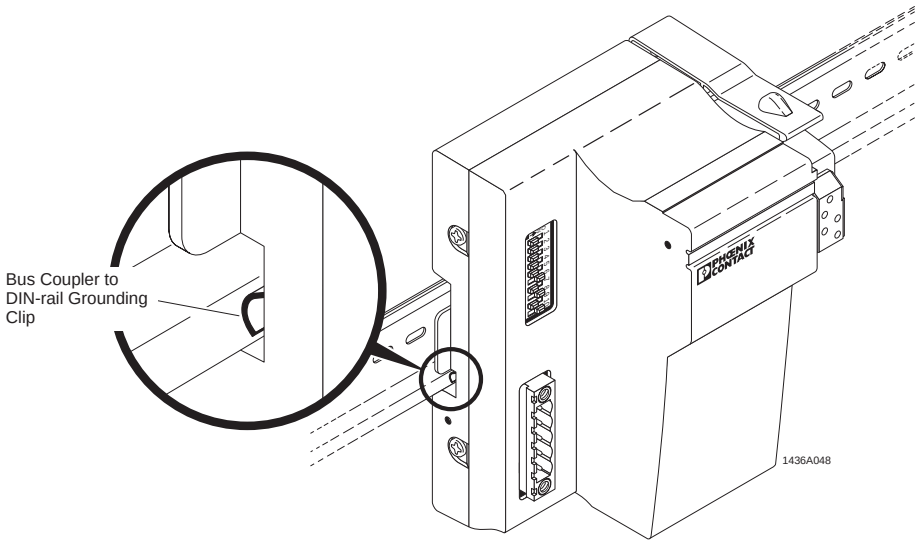


Figure 2-21. FE Grounding Tab on Bus Coupler

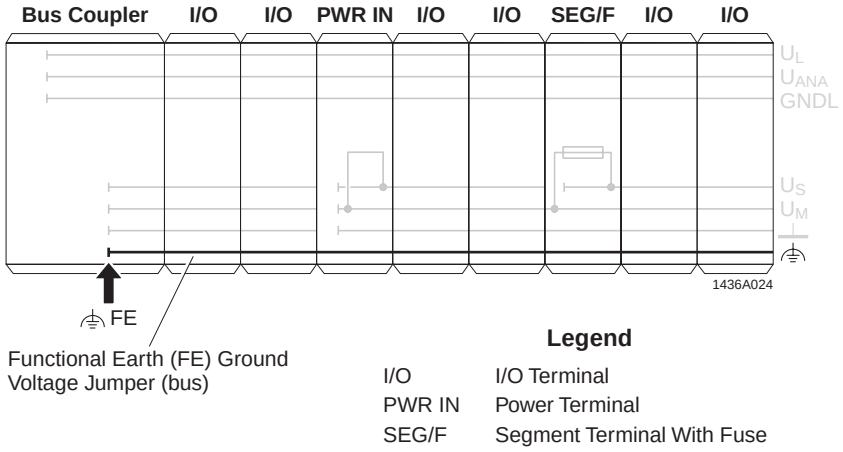


Figure 2-22. Functional Earth (FE) Ground Circuit

B. Power and Segment Terminals

Like the bus coupler, power terminals and segment terminals have an Functional Earth (FE) grounding tab on the back of the electronics base.

C. Additional Grounding Terminal Blocks (USLKG 5)

We recommended that grounding security for the Inline station be increased by installing an additional grounding terminal block to each end of the DIN-Rail. Next, attach a 0.2 - 1.5 mm^2 (24 to 16 AWG) wire from the bus coupler FE connection (positions 1.4 or 2.4) to the terminal block closest to the bus coupler. Finally, connect a 2.5 mm^2 (14 AWG) wire from the grounding block to a central Protective Earth ground. See Figure 2-23.

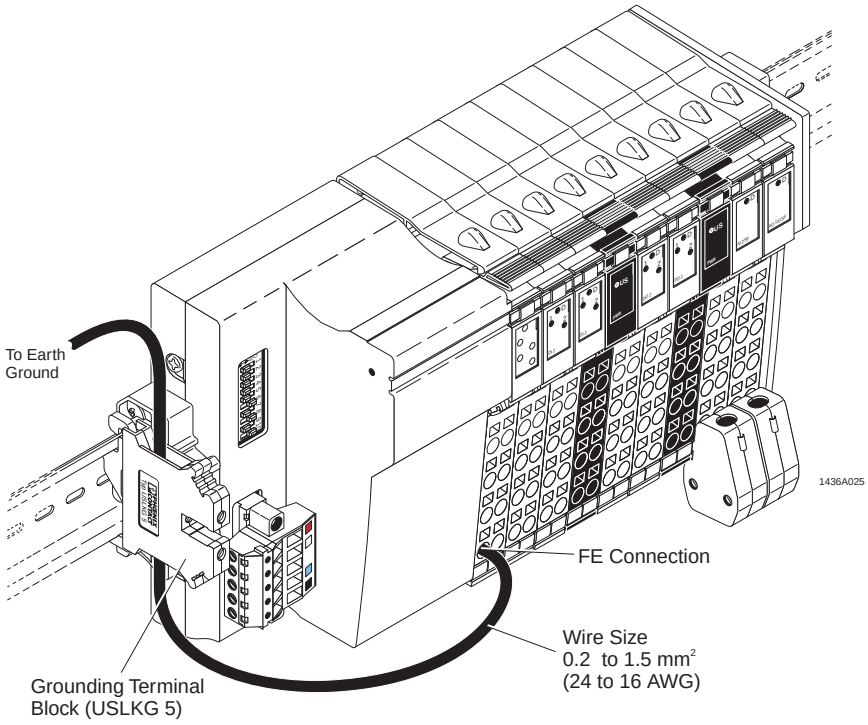


Figure 2-23. Additional Grounding of the Bus Coupler

SECTION 3
Configuration and Operation

Section 3 Contents

Configuration and Operation

3.1 Introduction

This section provides the quickest path to go online with the Inline DeviceNet™ system.

Notes

Each procedure in this section deals with information on how to install, operate and troubleshoot an Inline DeviceNet™ network.

Information in this section is presented with the assumption that the user has a working knowledge of DeviceNet™ and the RSNetWorx™ (or equivalent) configuration software. By default the bus coupler will be ready to exchange data in the polled I/O scan mode.

When this unit is first received it will be set up to use address 63 and a baud rate of 125K. This station is delivered for operation in fault response mode 0, Stop on Fail. In the Stop on Fail mode a local I/O failure (except for the peripheral fault) will cause all outputs to turn off. The PF bit is set in the Inline Status word when any output is shorted or during the loss of power to an intelligent segment module.

By default, Inline station diagnostics (Inline Status word) are included in the produced response. The PF bit is included in the diagnostic word and will need to be evaluated by the application software to determine the required action for the local station outputs.

3.2 DIP Switch Settings for Address (MAC ID) and Baud Rate

Each bus coupler has ten DIP switches. These switches are located on the left side of the DeviceNet™ bus coupler. See Figure 3-1. DIP switches 1 through 7 are used to set the MAC ID. DIP switches 8 and 9 are used to set the baud rate. DIP switch 10 sets the bus coupler configuration mode as described in Figure 3-1.

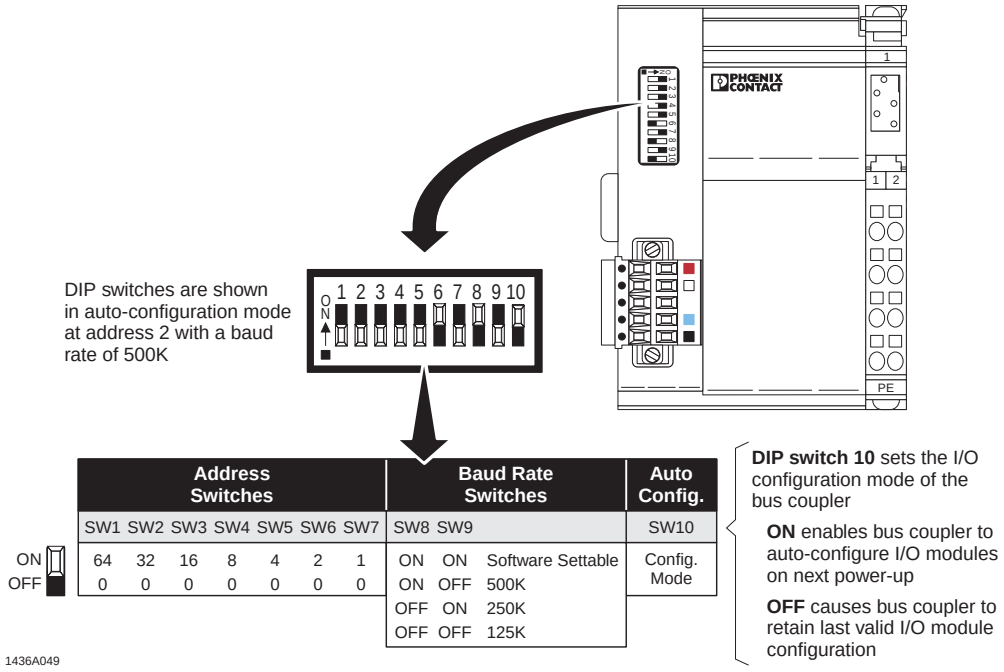


Figure 3-1. DeviceNet™ Bus Coupler DIP Switches

A. Setting the Node Address (MAC ID)

The node address is set using DIP switches 1 through 7. DIP switch 7 is the least significant digit of the MAC ID and DIP switch 1 is the most significant digit. Valid MAC ID settings range from 0 to 63 . Note that DeviceNet™ power needs to be cycled (OFF & ON) in order to implement any changes in a hardware selected node address (MAC ID).

B. Software Setting of the Node Address (MAC ID) Using Switches 1 through 7

The software setting of the node address is also set using DIP switches 1 through 7. Any combination of DIP switch settings greater than 63 will allow the MAC ID to be set using software. This software setting is made by sending an explicit message using Service Code 16 dec, 0x10 hex (Set_Attribute_Single) to the DeviceNet™ Object. Then set service parameters as follows.

Service Code 16 dec, 0x10 hex (Set_Attribute_Single)

- Parameter 1 Class Code 3
- Parameter 2 Instance 1
- Parameter 3 Attribute 1
- Parameter 4 Data (Desired address 0 - 63)

Note

The MAC ID software setting will default to the last valid physical setting of the DIP switches.

C. Setting the Baud Rate

The baud rate is set using DIP switches 8 and 9. DIP switch settings for various baud rates are shown in Figure 3-1.

D. Software Setting of the Baud Rate

The software setting for the baud rate is also set using DIP switches 8 and 9. Setting DIP switches 8 and 9 to the ON position allows the software to set the baud rate. The software setting is made by sending an explicit message using the Service Code 16 dec, 0x10 hex (Set_Attribute_Single) to the DeviceNet™ Object. Then set service parameters as described below.

Service Code 16 dec, 0x10 hex (Set_Attribute_Single)

- Parameter 1 Class Code 3
- Parameter 2 Instance 1
- Parameter 3 Attribute 2
- Parameter 4 Data: 0 = 125 k bits/s
  1 = 250 k bits/s
  2 = 500 k bits/s

Note

The software setting will default to the last valid physical DIP switch setting.

E. Restoring Factory Default Settings

Set DIP switches 1 through 7 to the ON position. Then apply power to the bus coupler to restore all settings to factory default.

3.2.1 Faulted Node Recovery (FNR)

Note

DIP switches can be used to physically set the bus coupler's MAC ID. Faulted Node Recovery gives the user an alternative method by assigning the MAC ID using software. Typically, Faulted Node Recovery is used for devices that do not have mechanical switches to set the MAC ID.

Fault Node Recovery (FNR) is the ability to recover two or more nodes connected to a DeviceNet™ network that have the same MAC ID. This includes sensors, actuators, drives and all other DeviceNet™ nodes. The advantage of FNR is that all devices on the DeviceNet™ network can be connected using the same MAC ID (same DIP switch settings) and then given a unique MAC ID through configuration software.

Example Procedure Using Inline Bus Couplers

If using FNR, use the following procedure:

1. Set all bus couplers to the same MAC ID. Value must be 64 decimal or greater.

Note

When installing bus couplers, the user may want to record the serial number of each bus coupler and its location on the network.

2. Install all bus couplers on the network.
3. Using the RSNetworx™ (or equivalent DeviceNet™ configuration package having Faulted Node Recovery capability) Faulted Address Recovery Wizard, go online to view all installed bus couplers.
4. From the displayed list of bus couplers (with serial numbers), use one of the following two methods for identifying and setting the unique MAC ID.

Method 1 - By Serial Number

1. Select the first bus coupler and serial number from the displayed list.
2. Match the serial number to its location on the network.
3. Assign an unused MAC ID to the bus coupler.
4. Continue assigning unused MAC IDs to remaining bus couplers.

Method 2 - By Visual observation

1. Select the first bus coupler and serial number from the displayed list.
2. Press LED button to blink NT LED for selected bus coupler.
3. Visually search each bus coupler on the network to determine which NT LED is blinking.
4. Assign an unused MAC ID to the bus coupler.
5. Continue assigning unused MAC IDs to remaining bus couplers.

3.3 Determining I/O Module Capacity

The bus coupler is capable of processing the maximum number of instances (points) for DeviceNet™/vendor-specific objects listed below.

- Discrete Input Points (DIP) 510 instances
- Discrete Output Points (DOP) 510 instances
- Analog Inputs Points (AIP) 510 instances (128 max. in poll)
- Analog Output Points (AOP) 510 instances (128 max. in poll)
- Special Function Object 63 instances
- PCP Special Function Object 8 instances*
- Serial Communication Object 8 instances*

* Total instances shared between the PCP Special Function and Serial Communication Objects can not exceed 8.

There are certain considerations that must be observed when determining the number of I/O devices that can be connected to the bus coupler. These considerations are described in the following paragraphs.

1. The bus coupler cannot provide more than 2 amps of communications/logic power (U_L).

Note

If U_L power requires more than 2 amp a IB IL 24 PWR IN/R terminal can be inserted to reinject U_L power.

2. The maximum number of devices connected to the bus coupler cannot exceed 63.
3. Analog modules cannot draw more than 0.5 amps from the analog supply (U_{ANA}). Note that analog modules also require current from the 2 amp logic supply (U_L).
4. Loop 2 devices
   a. Only the IB IL 24 L2 (Inline Loop 2 controller) uses the logic supply (U_L)
   b. Loop 2 I/O devices will use current from the segment power (U_S)
   c. Maximum load for the IB IL 24 L2 is 1.8 amps. After the 1.8 amps is allocated, additional power must be applied to the loop.
   d. The IB IL L2 does not count towards the 63 device maximum.
   e. Each Loop 2 I/O device does count towards the 63 device maximum.

Note

There are 2 amps of the current available on the Inline communications supply (U_L) and 0.5 amps of current available on the analog supply (U_{ANA}). Refer to the specific I/O module's data sheet or the Phoenix Contact Automation Catalog to determine the amount of current draw required for each I/O module or device to be connected to the bus coupler. The total amount of current draw for all modules/devices cannot exceed the 2 amp and 0.5 amp ratings stated above.

3.4 Configuring the Inline Station

Note

If using a serial or another type of PCP module, read this section then refer to section 6

3.4.1 Bus Coupler

Configuration of the bus coupler allows it to read and communicate specific information about the I/O modules connected on its local bus (backplane). Specific local bus information includes:

- How many I/O modules are on the local bus
- Position of the I/O modules on the local bus
- How many points or channels (instances) each module contains
- Bytes of produced and consumed data

  Types of I/O modules:
  - Digital input
  - Digital output
  - Analog input
  - Analog output
  - Special function (incremental encoder, absolute encoder, high speed counter, other)
  - PCP and serial (AS-i master, RS232, RS485, and others)

Notes

Any module that is not recognized will be placed into the special function category.

When configuring the Inline DeviceNet™ bus coupler, it is recommended that you remove the connection to the DeviceNet™ scanner or make sure the scan list for the DeviceNet™ scanner is empty.

3.4.1.1 Configuration Methods

When creating, adding to, or changing an Inline DeviceNet™ station, the I/O configuration stored in the bus coupler must be updated to match the new configuration of the station.

Configure the bus coupler using one of the following 3 methods

- The Electronic Data Sheet (EDS) file
- Auto-configuration (no software required)
- Sending an explicit message

A. The Electronic Data Sheet (EDS) File Method

The EDS file is the software interface between the bus coupler and a DeviceNet™ configuration software package such as RSNetWorx™. The EDS file contains information about the number of produced and consumed bytes. It also provides the user-settable parameters such as what should be included under poll options.

Notes

Appendix B in this manual describes the EDS file and each of the EDS file parameters.

EDS file Parameter 9, (Add All I/O) must be used for new or changed configurations when I/O modules are to be included in the scan.

The following procedure describes how to configure a bus coupler using the Electronic Data Sheet. Repeat this procedure for each bus coupler on the network.

1. Obtain a list of I/O modules that will be used in the DeviceNet™ station.
2. Determine which types of I/O modules will be included in the scan. By default, all modules will be included. If the default needs to be altered, a new value must be downloaded to the bus coupler through EDS Parameter 8 (Add All Mode).
3. Determine whether the "Inline Status" (diagnostic) word needs to be included in the produced size. By default, these 2 bytes will be added to the produced size. By using the produced data channel, the Inline Status word gives the user the ability to locate and define any faults that could occur on the Inline local bus. If the user decides to disable this feature, then the user must download a 0 to the bus coupler using Parameter 1 (Use Inline Status) in the EDS file.
4. Determine whether analog or special function modules are used. If used, the user has the option to set Parameter 2 (Pad I/O) to a 0 or 1 (default). If not used, proceed to step 5.

Note

Depending on the I/O configuration, the analog or special function data may start on a odd byte. If this is the case, it is possible that this word could span two words in the master scanner. By setting the Parameter 2 (Pad I/O) to 1 (default) one byte will be added to the produced/consumed size, thereby, forcing the first word based module to start on an even byte. If this word already starts on an even byte and the user sets Parameter 2 (Pad I/O) to 1 (default), no additional bytes will be added to the produced /consumed size.

5. If future system expansions are anticipated, determine if there is a need to reserve digital points (bits) or analog words in the scan. If so, determine the number of points (bits) or words to be reserved, then add to this the number of points or words currently being used. For more information, refer to "Reserving I/O Memory for Future System Expansion" in this section. This total number of points or words must be downloaded to the bus coupler. Reserving points or words in local I/O memory is accomplished by using Parameter 3 (Reserve Digital Inputs), Parameter 4 (Reserve Digital Outputs), Parameter 5 (Reserve Analog Inputs) and Parameter 6 (Reserve Analog Outputs).
6. Enter the new station configuration parameters into the flash memory of the bus coupler by changing the default value of Parameter 9 (Add All I/O) from a 0 (False) to a 1 (True). Depending on the size of the local bus, the user may have to wait until the downloading of the new configuration is completed. Completion of the download can be determined by observing the state of the "MD" LED on the bus coupler and the "D" LEDs on the I/O modules. While downloading, the LEDs will blink, Once the download is completed, the blinking stops and the new configuration is now stored in the flash memory of the bus coupler.

B. Auto-configuration Method

Auto-configuration allows for in-the-field configuration of the bus coupler without any software. Defaults settings entered under auto-configuration are:

- Two bytes will be added to the produced size for the Inline Status Word (diagnostic word).
- All modules will be added to the produced and consumed response as dictated by their class..
- Pad I/O will be used.
- No digital or analog I/O reservations for future system expansion will be made.

Note

An explanation of each of these EDS parameters can be found in Appendix B of this manual.

The following procedure describes how to use the auto-configuration method.

1. Set DIP switches 1 through 7 to the desired address. Set DIP switches 8 and 9 to the desired baud rate. Set DIP switch 10 to "ON" to allow auto-configuration.
2. Check that all required I/O devices are connected.
3. Power up the bus coupler using network and U_L (communication/logic) power. Note that at initial power up, the new address and baud rate are automatically stored in the bus coupler's flash memory.
4. Set DIP switch 10 to the "OFF" position and power cycle to store current configuration.

Note

At this point both the "MD" LED on the bus coupler and the "D" LEDs on the I/O modules should begin blinking. Once the LEDs stop blinking and remain ON, storage of the I/O configuration in the bus coupler's flash memory is completed.

C. Sending an Explicit Message Method

Note

When configuring the bus coupler of an Inline station by sending an explicit message, observe the decisions stated under paragraph 3.4.1.1, part A. The parameters listed in part "A" can also be configured by sending multiple explicit messages to the Configuration Object (Class Code 100 dec, 0x64 hex).

When using the explicit message method to change defaults, the following command structure must be used to configure the DeviceNet™ bus coupler.

- Service Code 16 dec, 0x10 hex
- Class Code 100 dec, 0x64 hex
- Instance 1
- Attribute X (X = Attribute to be change)
- Attribute Data 1

If no default settings need to be changed, you still must send one explicit message using the following command structure to configure the DeviceNet™ bus coupler.

- Service Code 16 dec, 0x10 hex
- Class Code 100 dec, 0x64 hex
- Instance 1
- Attribute 7 (Add All I/O)
- Attribute Data 1

Setting Attribute 7 (Add All I/O) to a 1 instructs the bus coupler to scan its local bus and store its current configuration into the bus coupler's flash memory. This configuration will remain in flash memory until the next "Add All I/O" is sent or until a different configuration method is used.

Note

The service that allows Attribute 7 to be set is Service Code 16 dec, 0x10 hex (Set_Attribute_Single). Object classes, and services are described in Appendix A. The RSNetWorx™ or an equivalent software package can be used to send explicit messages.

3.4.2 Analog Input (AI) Modules

3.4.2.1 General Configuration

Non-multiplexed (standard) analog input (AI) modules default to a unipolar range of 0 V DC to +10 V DC. To change this range, the range attribute can be set in the Analog Input Point (AIP) Object (Class Code 10 dec, 0x0A hex). Set the range by sending an explicit message using the "Class Instance Editor" in RSNetWorx™. Additionally, attributes 100 and 101 could be used to write a custom configuration value to the analog input module. Refer to the Thermocouple and RTD module paragraph below to determine how to use attributes 100 and 101. Enabling attribute 101 will override any range settings.

Once the range is set for the new value, the bus coupler will retain that setting in flash memory. If the bus coupler is replaced the configuration will need to be redone.

Note

Appendix A provides details of the AIP Object (Class Code 10 dec, 0x0A hex) range settings.

Multiplexed analog input (AI) modules such as the AI8 will appear to have only as many channels as the module has data words. The BK has no way of determining how many channels are contained within those words. By default, the control words for an analog input module are not placed in the poll. However, with multiplexed modules, it is often desirable to change the control word frequently. The user can instruct the BK to place these control words into the poll. This is done by enabling attribute 102 "AIP configuration word in poll" of the AIP object. Enabling attribute 102 will override both attribute 101 and any range settings.

3.4.3 Thermocouple and RTD Modules

3.4.3.1 General Configuration

This section describes how to change default settings for the Inline 2-channel thermocouple module. However, changing the default settings for the 2-channel RTD modules is accomplished in the same manner.

Note

There is no "Thermocouple" or "RTD Object". To change default settings, the Analog Input Points (AIPs) Object (Class Code 10 dec, 0x0A hex) must be used. Refer to Appendix A.

Default settings for the thermocouple modules are:

- Sensor type: K
- Resolution: 0.1°C (1 microvolt)
- Output format: 15 bits and 1 sign bit with extended diagnostics
- Cold junction: Internal

In order to change any setting, Refer to the thermocouple data sheet 5722 to determine the appropriate attribute settings for the AIP Object (Class Code 10 dec, 0x0A hex).

Note

Keep in mind that thermocouple or RTD instances will appear as analog input instances. There are 2 instances for every thermocouple or RTD module. Channel 0 will be the first instance and Channel 1 will be the second.

The user must keep track of which instances are analog inputs and which instances are thermocouple inputs. Figure 3-2 shows a station where instances 1 and 2 of the AIP are used by the 2-channel thermocouple. Instances 3, 4, 5 and 6 of the AIP are used by the next two, 2-channel analog input modules. Instances 7 and 8 of the AIP are used by the next 2-channel thermocouple module. The last two, 2-channel analog input modules occupy instances 9,10,11 and 12.

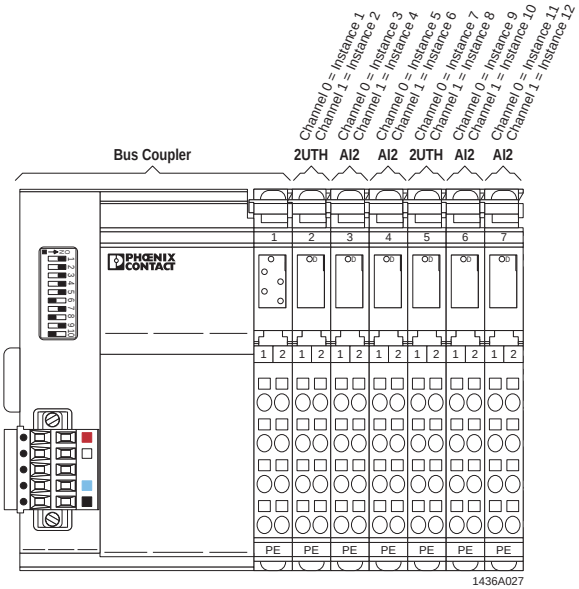


Figure 3-2. I/O Station with Analog Input Terminals and Thermocouple Terminals

Note

Default settings are modified by sending an explicit message to a specific instance. This message can be sent using the "Class, Instance Editor" window in RSNetWorx™.

Using attribute 100 along with the correct instance in the AIP object is one way to determine how a thermocouple, RTD, or analog input module is configured.

If Attribute 100 = 0 (default) in the AIP, the user has access to all standard AIP attributes. If Attribute 100 = 1 the user has access to Attribute 101 (Input Configuration word). By assigning a value to the Attribute 101, the user will be able to configure the thermocouple or RTD module(s). The correct configuration value for attribute 101 can be determined by using the module specific data sheets for the thermocouple, or RTD modules. It is also possible to use the same method as multiplexed modules, whereby the configuration word is placed permanently into the poll data.

Once the new thermocouple setting is made, the new configuration will be stored in flash memory the bus coupler. If the bus coupler is replaced the configuration will need to be redone.

Note

AIP Object (Class Code 10 dec, 0x0A hex) settings are described in Appendix A of this manual.

3.4.4 Special Function Modules

3.4.4.1 General Configuration

Special function modules such as the incremental encoder, absolute encoder and the high-speed counter are configurable through the polled data channel by default. Output word(s) that are assigned to the special function module are used to program the terminal. The user must refer to the specific special function module's data sheet or manual to determine what codes need to be written to the associated output word(s).

If the special function module is not included in the poll, it can be programmed through the use of explicit messages Special Function Object (Class Code 103 dec, 0x67 hex).

Note

The programming of a special function module cannot be stored in the bus coupler flash memory. The user's application will have to implement a programming subroutine.

3.4.5 Using RSNetWorx™

3.4.5.1 Adding the EDS File

The EDS file for the DeviceNet™ bus coupler can be added to the RSNetWorx™ as described in the following procedure.

1. From the RSNetWorx™ menu bar, select "Tools". Then from the "Tools" drop-down list select "Install EDS Wizard".

Note

EDS, BMP and ICO files for DeviceNet™ are available on the Phoenix Contact web site at www.phoenixcon.com, under the Infoservice link.

2. Follow the EDS Wizard options to register an EDS file.
3. In the EDS Wizard, browse for and select the bus coupler EDS file. Then associate (select) the proper icon. This will add the EDS and visual icon to RSNetWorx™.

3.4.5.2 Setting the Node Address

Note

This section may be skipped if using the DIP switches to set the actual node address.

Make sure DIP switches 1 through 7 are set to a node address that is greater than 63. The following procedure requires that the A-B 1770-KFD module (or equivalent) is connected to the DeviceNet™ bus coupler.

1. From the RSNetWorx™ menu bar, select "Network" then "Online". Form the "Online" window the appropriate interface tool driver can be selected. See Figure 3-3.
2. From the "Browse for network" window, select the previously installed in RSLinx™ driver that you want to use. For discussion purposes in this procedure, we will be using the A-B 1770-KFD RS-232 Interface V1.50. Once the adapter is selected, click OK.

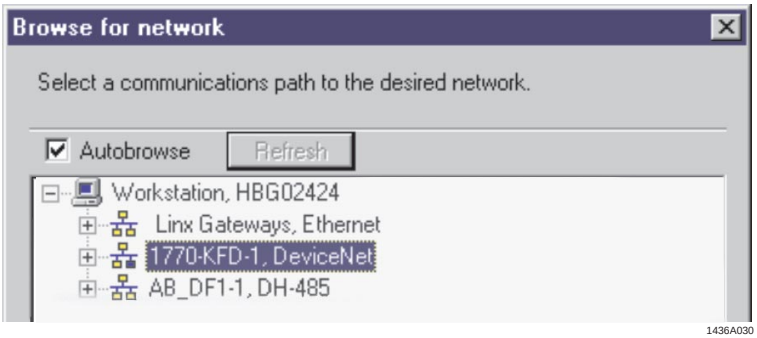


Figure 3-3. RSNetWorx™ Browse for network Window

3. Once the driver is selected, click OK. The network browse will start. Make sure the connection status icon shows a moving on-line connection. See Figure 3-4.
4. Wait for the browse operation to be completed. In our example, the network shows both the icon for the KFD module and the icon for the DeviceNet™ bus coupler. See Figure 3-4. Note the last valid node address setting for the bus coupler (in our example) is 2.

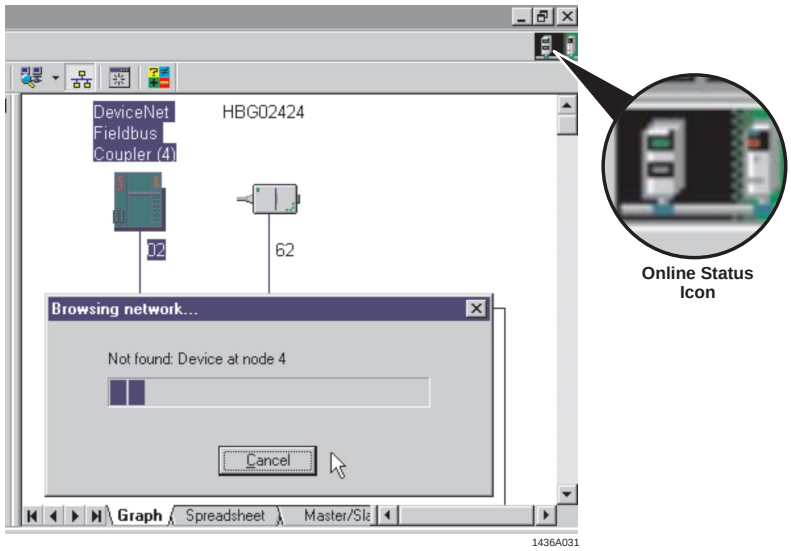


Figure 3-4. Network Browse Results

5. Click on the bus coupler icon so that it is highlighted.
6. From the Device menu, choose "Class Instance Editor". At this point the "Class Instance Editor" window will appear. See Figure 3-5.

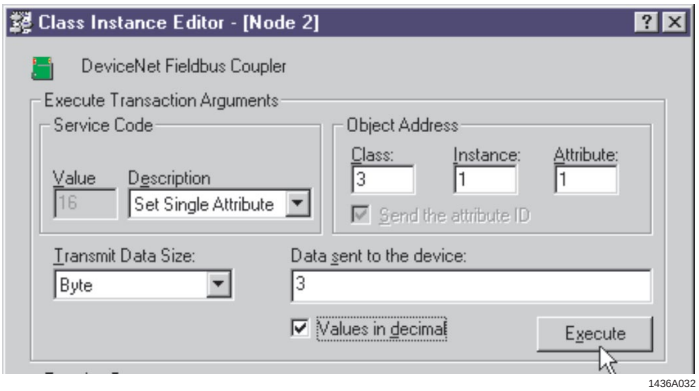


Figure 3-5. Class Instance Editor Window

7. From the "Class Instance Editor" window, Locate the proper text boxes and enter the following:
   - Class: 3
   - Instance: 1
   - Attribute: 1

8. Locate the "Data sent to the device" text box and enter your desired address. In our example, the address is being changed to 3. Make sure the data type is sent to a byte.
9. Locate the "Service Code" drop down menu, make sure that it is set to "Set Attribute Single".
10. Click on the "Execute" button to download the new address parameter. Thereafter, the next time you choose the "Browse" option, the new node address will appear.

3.4.5.3 Setting the Baud Rate

Note

This section may be skipped if the DIP switches are being set to the actual baud rate.

With the A-B 1770-KFD module (or equivalent) connected to the Inline DeviceNet™ bus coupler, set DIP switches 8 and 9 to the ON position. Then cycle power. Set the new baud rate as described in the following procedure.

1. From the RSNetWorx™ menu bar, select "Network" then choose "Online". Form the "Online" window (refer to figure 3-3), select the appropriate interface tool driver.
2. From the "Browse for network" window, select the previously installed in RSLinx™ driver that you want to use. For discussion purposes, this procedure will be using the A-B 1770-KFD RS-232 Interface V1.50. Once the adapter is selected, click OK.
3. Once the driver is selected, click OK. The network browse will start. Make sure the connection status icon shows that there is a moving online connection. Refer to Figure 3-4.

Note

To go online, the initial baud rate must match the last valid DIP switch setting.

4. Wait for the browse operation to be completed. In our example, the network shows both the icon for the KFD module and the icon for the DeviceNet™ bus coupler. Refer to Figure 3-4.
5. Click on the bus coupler icon so that it is highlighted.
6. From the Device menu, choose "Class Instance Editor". At this point the "Class Instance Editor" window will appear. Refer to Figure 3-5.
7. From the "Class Instance Editor" window, Locate the proper text boxes and enter the following:
   - Class: 3
   - Instance: 1
   - Attribute: 2
8. Locate the "Data sent to the device" text box. Then enter your desired baud rate. In our example, the baud rate is being changed to 500K.
   - 0 = 125 k bits/s
   - 1 = 250 k bits/s
   - 2 = 500 k bits/s
9. Locate the "Service Code" drop down menu, make sure that it is set to "Set Attribute Single".
10. Click on the "Execute" button to download new baud rate.
11. Reconnect to the bus coupler using the new baud rate as described in Steps 1 through 4.

3.4.5.4 Adding the I/O to the Bus Coupler's Memory

Note

If the bus coupler EDS file is not present in the Device List under Generic\Phoenix Contact\..., install the EDS file as previously described.

1. Make an online connection to the DeviceNet™ bus coupler using the A-B KFD module (or equivalent).

⚠ CAUTION

To prevent communication errors to the device, make sure that the device is not included in the master controllers scan list, or unplug the network cable from the scanner.

Note

At this point the user must know what EDS parameters are required for the application. When determining parameters, consider Parameters 1, 2 and 9. Parameter 9 is mandatory for the I/O configuration using the EDS file. See Appendix B for a description of the EDS file parameters.

2. Double click the Inline icon. Then select the "Parameters" tab. At this point the EDS file parameters window will be displayed. See Figure 3-6.

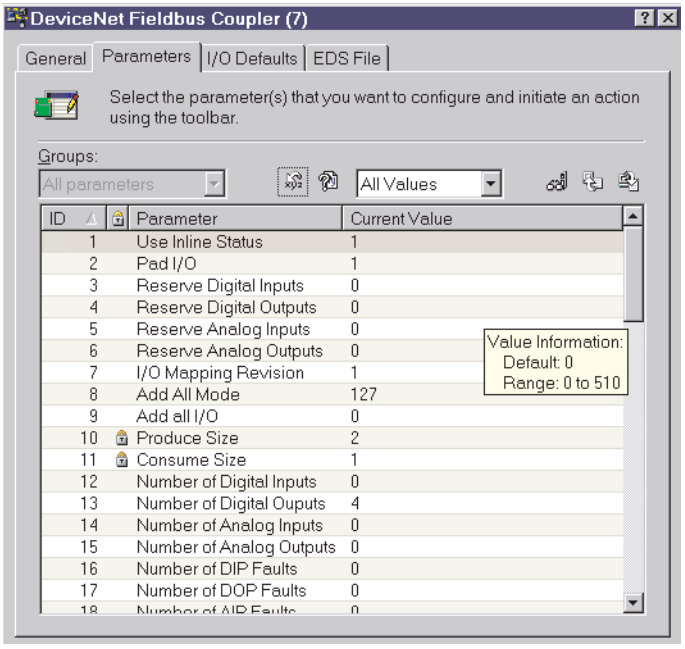


Figure 3-6. EDS File Parameters Window

3. Ask yourself the following two questions.

   Question 1: Should the Inline Status Word (EDS Parameter 1) be added to the produced data?

   If the answer is NO, double click on Parameter 1 (Use Inline Status). Set the value to a 0. Then download the parameter to the device. Proceed to Question 2.

   If the answer is YES, double click on Parameter 1. Set the value to a 1 (default) and download. By default the Inline status word will be added to the produced data size (adds two bytes to the produced size). Proceed to Question 2.

   Question 2: Should the word oriented modules start on a even byte in the I/O memory?

   If the answer is NO, double click on Parameter 2. Set value to a 0. Then download to the device. Proceed to Step 4.

   If the answer is YES, double click on Parameter 2 (Pad I/O). Set value to a 1 (default) and download. This may add 1 byte to the produced/consumed size. Proceed to Step 4.

4. Double click on Parameter 9 (Add All I/O). Set to 1, then download to the device.

Note

The bus coupler will retain its last known valid I/O configuration. The last configuration is stored in flash memory. If an I/O module has been added or deleted from the Inline station, the DeviceNet™ bus coupler must be reconfigured using Parameter 9 (Add All I/O). Parameter 9 will ensure that the new configuration is stored in flash memory. Optionally DIP switch 10 can be set to the ON position and the station can be power cycled to auto-configure the bus coupler.

3.5 Understanding I/O Memory Mapping

3.5.1 Bus Coupler Mapping

The I/O image in the bus coupler flash memory contains all produced-data (input data) and consumed-data (output data) derived from the I/O modules connected to it. I/O image data is added to the poll through the use of Parameter 9 (Add All I/O). Configuration through EDS and RSNetWorx™ is explained in the previous section.

An I/O image could contain the following produced and consumed elements in the priority order listed below.

Produced	Consumed
1. Inline Status	1. Inline Control Byte
2. DIP Faults	2. DOP States (values)
3. DOP Faults	3. DOP Reserve Points
4. AIP Faults	4. PAD Byte (optional)
5. AOP Faults	5. AOP States (values)
6. Special Function Faults	6. AOP Reserved Words (optional)
7. DIP States (values)	7. AIP Configuration Words (optional)
8. DIP Reserve	8. Special Function
9. PAD Byte (optional)	9. PCP Special Function (process & fragment)
10. AIP States (values)	10. Serial Communication (process & fragment)
11. AIP Reserve	
12. Special Function	
13. PCP Special Function (process & fragment)	
14. Serial Communication (process & fragment)	

All produced elements of the same priority will be mapped together regardless of their location. However, their relative location to the BK will be used to determine their instance values (sequential ordering). This same approach applies to consumed elements.

Analog channels will start at the first completely unused byte after the last digital module. If the total number of digital points of the same image is not modulus 8, there will be unused bits between the digital data area and the analog data area.

Note

Depending on what I/O modules are connected to the bus coupler determines whether or not analog data starts on an even or odd byte. In those cases when analog data starts on an odd byte, analog data will span two words in the master scanner. If you prefer to have analog data to start on an even byte, set the EDS Parameter 2 (Pad I/O) to a 1. Then download to the bus coupler.

This will prevent analog data from starting on an odd byte without regard to the I/O modules connected to the bus coupler. Once Parameter 2 is set, one byte of unused I/O data may be added to the produced/consumed size. This byte of unused I/O data will force the analog word to always start on an even byte in the master scanner. If the physical configuration dictates that the analog word starts on an even byte and Parameter 2 (Pad I/O) will not add a byte of data to the I/O data size.

The physical order of data in the I/O table is determined by the position of the modules on the local bus. The first module connected to the Inline DeviceNet™ bus coupler will reside in the first I/O byte (keeping in mind the "which data type comes first" rule). Furthermore, the LSB of the first module will be assigned to the first instance. The next module of the same type and image will line up next to the first module without leaving any “gaps” in the I/O table.

Figure 3-7 shows an example I/O table (memory map) consisting of digital and analog output modules. The total amount of input bytes (Inline Status Word) would be 2 and the total amount of output bytes would be 4. This example consist of the following I/O modules:

- 2-bit digital output
- 8-bit digital output
- 1-channel analog output.

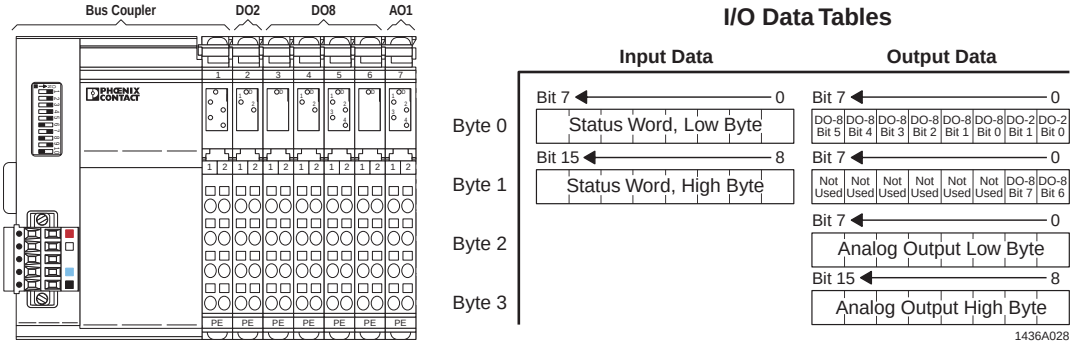


Figure 3-7. Example of an I/O Table (Memory Map) Consisting of Analog and Digital Output Modules .

Figure 3-8 shows an example I/O table consisting of a digital/analog, input/output modules:

- 2-bit digital input
- 1-channel analog output
- 2-bit digital output
- 4-bit digital input
- 2-channel analog input
- 4-bit digital output

Figure 3-8 shows that the total number of input bytes is 7 (byte 0 through byte 6). This includes the Inline Status Word. Figure 3-8 also shows that the total number of output bytes is 3 (byte 0 through byte 2).

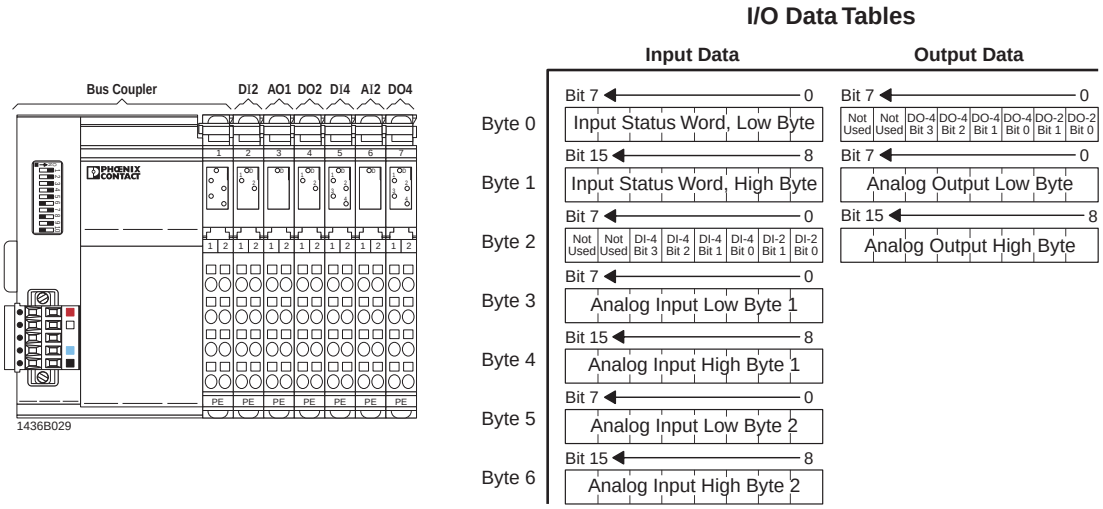


Figure 3-8. Example of an I/O Table (Memory Map) Consisting of Digital & Analog, Input & Output Modules

In our example, 7 and 3 were entered in the "Polled" box of the "Edit I/O Parameters" window. See Figure 3-9. The 7 (bytes) would be entered in "Rx" text box and the 3 (bytes) would be entered in the "Tx" text box.

The "Edit I/O Parameters" window can be opened as described in the following procedure:

1. Right click the scanner icon.
2. Select "Properties".
3. Select "Scanlist" tab then click "Upload".
4. Add an available device or select a device from the scan list to edit.
5. Select "Edit I/O Parameters".

Note

When determining the total number of bytes to be configured for the DeviceNet™ bus coupler, the Inline status word must be added to the number of input bytes. Configured produced and consumed size can be read online by double clicking on the device icon then uploading the parameters.

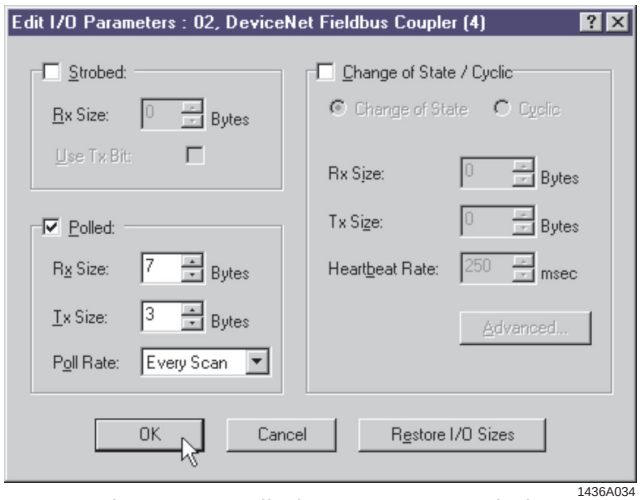


Figure 3-9. Edit I/O Parameters Window

3.5.2 Reserving I/O Memory for Future System Expansion

Note

Memory reservation is only available for digital and analog modules. It is not required special function modules.

A. Rules for Reserving I/O Memory

1. Reserved I/O points will take up physical space in the produced and/or consumed data.
2. After reserving digital and/or analog I/O, any new modules added must be connected after (anywhere to the right of) the last digital/analog module of the same type and image (input or output) on the local bus.
3. If special function modules are added, they must be added after the right most special function module on the station. After adding the special function module, the station must be reconfigured to add the special function data to the scan. This additional data will not effect existing mapping of the master scanner.
4. If you want to reserve space for analog input configuration words, you must add the number of analog input configuration words to be reserved to the number of analog output words to be reserved. This will be the total number of analog output words to be reserved.

   When adding modules to this shared reserve space, analog output words will be added to the lower end of this space and analog input configuration words will be added to the upper end of this space until the entire space is used. For information about analog input configuration words, refer to the paragraph titled "Analog Input Modules" in this section.

B. Ways to Reserve I/O Memory

The bus coupler can reserve digital or analog I/O in either the input or output image. This will allow for future system expansion(s) without having to change the master scanner's I/O tables. The actual reservation can be done in the following two ways:

1. By using EDS file Parameter 3 (Reserve Digital Inputs), Parameter 4 (Reserve Digital Outputs), Parameter 5 (Reserve Analog Inputs), Parameter 6 (Reserve Analog Outputs). The entry downloaded to the bus coupler will be equal to the current physical number of I/O points on the local bus, plus the number of I/O points to be reserved.
2. By sending an explicit message to the Configuration Object (Class Code 100 dec, 0x64 hex). The user can reserve a digital bit by writing to Parameters 22 (Reserve Digital Inputs), 23 (Reserve Digital Outputs), 35 (Reserve Analog Inputs) and 36 (Reserve Analog Outputs). The entry downloaded to the bus coupler will be equal to the current physical number of I/O points on the local bus, plus the number of I/O points to be reserved.

3.5.3 I/O Mapping Revision

This parameter controls which DeviceNet™ I/O class Inline modules will be placed into. The default (1) will use the latest information stored in the firmware. A Mapping Revision of "0" will use mapping information that is compatible with the IL DN BK 1 DeviceNet™ bus coupler. For example, on the IL DN BK1, the IB IL 24 AI8, and IB IL AO2/BP modules would be classified as Special Function modules, whereas, on the IL DN BK 3 they would be considered analog input and output modules respectively. Any modules not identified will be placed into the Special Function category. See Table 1.

Table 1. Inline Module ID's Mapped to DeviceNet™ Objects

Module ID Code (Decimal)	Module ID Code (Hex)	Type of Inline Module	Mapping Revision 0 Il DN BK1 (FW 2.xx & 5.xx)	Mapping Revision 1 Il DN BK3 (FW 2.01)
83	53	AI2O2, Analog Input/Output with parameter-data	SF	AIP, AOP
91	5B	AIO2, Analog Output with parameter-data	SF	AOP
95	5F	AI2O, Analog Input with parameter-data	SF	AIP
99	63	Loop AI2O2, Analog Input/Output with parameter-data	SF	AIP, AOP
102	66	ENCOM appliance	SF	AIP
103	67	ENCOM appliance	SF	SF
107	6B	Loop AIO2, Analog Output with parameter-data	SF	AOP
111	6F	Loop AI2O, Analog Input with parameter-data	SF	AIP
113	71	Analog Output INTERBUS Loop	AOP	AOP
114	72	Analog Input INTERBUS Loop	AIP	AIP
115	73	Loop Analog Input/Output	AIP	AIP
117	75	Analog Output for Safety	AOP	AOP
118	76	Analog Input for Safety	SF	AIP
119	77	Analog Input/Output for Safety	SF	SF
121	79	Analog Output with profile	SF	AOP
122	7A	Analog Input with profile	SF	AIP
123	7B	Analog Input/Output with profile	SF	SF
125	7D	Analog Output	AOP	AOP
126	7E	Analog Input	AIP	AIP
127	7F	Analog Input/Output	AIP	AIP
173	AD	Digital Output for Safety	SF	DOP
174	AE	Digital Input for Safety	SF	DIP
175	AF	Digital Input/Output for Safety	SF	DIP, DOP
177	B1	Digital Output INTERBUS Loop	DOP	DOP
178	B2	Digital Input INTERBUS Loop	DIP	DIP
179	B3	Digital Input/Output INTERBUS LOOP	DIP, DOP	DIP, DOP
181	B5	Digital Output with profile	SF	DOP
182	B6	Digital Input with profile	SF	DIP
183	B7	Digital Input/Output with profile	SF	SF
185	B9	Digital Output with Busmaster special-treatment	SF	DOP
186	BA	Digital Input with Busmaster special-treatment	SF	DIP
189	BD	Digital Output	DOP	DOP
190	BE	Digital Input	DIP	DIP
191	BF	Digital Input/Output	SF	SF

1436B125

3.6 I/O Data Transfer

Note

A detailed explanation of the following objects and their attributes can be found in Appendix A.

I/O data transfer can be accomplished by establishing a I/O connection (implicit) or by establishing a message connection (explicit).

A. Implicit

An implicit connection provides a dedicated path from a producing application to one or more consuming applications and is typically handled in a master with slave(s) relationship.

B. Explicit

An explicit connection is a generic connection between two devices where a request is sent and a acknowledge is expected.

Note

The bus coupler supports the Unconnected Message Manager (UCMM), which provides for the dynamic establishment of messaging connections.

3.6.1 I/O Scan Methods

The DeviceNet™ master will scan the bus coupler through the use of several implicit/explicit I/O scan types. The following scan methods are available to the user:

- Polled
- Explicit
- Change of State (COS)
- Cyclic
- Bit Strobe
- I/O Peer-to-Peer
- Explicit Peer-to-Peer

The bus coupler can support 10 DeviceNet™ connections:

- 3 dynamic I/O (I/O Peer-to-Peer)
- 3 dynamic explicit (explicit Peer-to-Peer)
- 1 master /slave explicit
- 1 polled
- 1 COS/cyclic
- 1-bit strobe

A. Polled

Note

Typically polled I/O is used with DeviceNet™ networks

The poll command is an I/O message that is transmitted by the master. A poll command is directed towards a specific node. The master must transmit a separate poll command for each node that is included in it's scan list.

The poll response is an I/O message generated by the slave and sent to the master after the poll command is received.

B. Change of State (COS) or Cyclic

A change of state or cyclic message is used to receive produced data from a slave device. This message is triggered by either a change of state or by the expiration of the cyclic time period. The response size is equal to the number of digital, analog and special function modules input bytes, plus two bytes for the Inline Status word by default.

By default, the change of state response is triggered by either digital inputs or the Inline Status word. If the trigger determination for the COS message needs to be modified the Build COS Mask Mode Attribute must be given the proper value to set the desired response trigger event(s). After the Build COS Mask Mode is set, a Build COS Mask must occur for the changes to go into effect. These attributes can be set by either using the EDS file or by sending explicit messages to the COS Mask Object (Class Code 104 dec, 0x68 hex).

If analog inputs are being used to trigger the COS response, the user has the ability to mask the lower bit weights to eliminate heavy network traffic due to fluctuating LSBs that may trigger the COS response. The Analog Mask Attribute can be found in the EDS file or can be set by sending an explicit message to the COS Mask object (Class Code 104 dec, 0x68 hex). The default mask for analog inputs, RTD and thermocouples is one byte.

Specific bits can be set to be the trigger by using the COS Mask Index and COS Mask Byte Value attributes. An example is shown in Appendix D.

C. Bit Strobe

The bit strobe is for inputs only. The bit-strobe command, transmitted by the master, has multicast capabilities. Multiple nodes can receive and respond to the same bit-strobe command.

When using the bit-strobe command, the value of this 1-bit command sent to the bus coupler is ignored. After which, the response is generated by the node and is sent to the master.

DeviceNet™ will use eight bytes of consumed data for all devices that use the bit-strobe connection.

The produced size of the response is limited to eight bytes. By default, this is equal to the number of digital, analog and special function module input bytes, plus two bytes for the Inline Status word.

D. I/O Peer-to-Peer

I/O peer-to-peer allows nodes to communicate in a implicit manner, without the need for a master or as an additional connection besides the master to slave connection. The bus coupler can offer this type of connection to another I/O peer-to-peer device that is able to establish the connection. An example of this relationship would be a motion controller using the I/O peer-to-peer connection to read inputs connected on an Inline station for control decisions.

E. Explicit Peer-to-Peer

Explicit peer-to-peer allows nodes to communicate in a explicit manner, without the need for a master or as an additional connection besides the master to slave connection. The bus coupler can offer this type of connection to another explicit peer to peer device that is able to establish the connection. An example of this relationship would be a configuration tool sending a message to the bus coupler.

3.6.2 I/O Communications Objects

Bus coupler I/O communication objects can be accessed through the use of the explicit messages. Listed below are the objects used to transfer I/O data for the bus coupler. If data needs to be transferred to a device that is not in a scan list, a “Get” or “Set” explicit message service can be sent to the proper class, instance and attribute in question.

A. Discrete Input Point (DIP) Object (Class Code 08 dec, 0x08 hex)

The DIP object models discrete inputs in the bus coupler. There is a separate instance for each digital input point available on the device. Attributes include Value and Status.

Setting DIP Inputs to Latch

Each DIP point can be independently configured to latch on a desired state. This is accomplished by using Attributes 100 and 101. Figure 3-10 shows how "actual input" or "latched" data is selected.

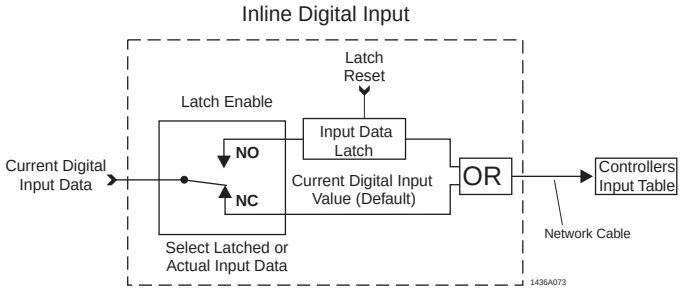


Figure 3-10. Latched or Current Data Selection

Attribute 100 is used it to enable the latching feature for any specific DIP. When set to a logic 0 (default), the latching feature is OFF. When set to logic 1, the latching feature is ON (enabled).

Attribute 101 determines the latch level of a specific DIP. Setting Attribute 101 to a logic 0 enables the DIP to select a low-level latch. Setting Attribute 101 to a logic 1 enables the DIP to select a high-level latch.

Enabling the latch and setting the desired latch state must be done by sending an explicit message.

Resetting the latched condition must be done by setting bit 1 in the Inline Control byte. This clear will effect all latched inputs. After the latches are reset, bit 1 in the Inline Control byte must be set back to 0 to allow for the next latched condition to occur when the control byte is in the consumed data.

By default the Inline Control byte is not included in the consumed data command. The user can add this to the consumed data by issuing an explicit message to the Configuration Object (Class Code 100 dec, 0x64 hex, Attribute 32).

If the user doesn't want to clear the latches through the polled I/O then an explicit message to the Inline Interface Object (Class Code 101 dec, 0x65 hex, Attribute 20) can be sent. Setting Attribute 20 to a 2 will reset all latches enabling the next input latch. It will also automatically reset the Attribute 20 value to 0.

Note

Latch values are retained during operation and will not be cleared until the latches are reset. Once a reset is received the latches will re-initialize to the value that allows the input level to be captured. This initialization depends on the value determined by Attribute 101, Latch Level.

B. Discrete Output Point (DOP) Object (Class 0x09 hex)

The DOP object models discrete outputs in the DeviceNet™ bus coupler. There is a separate instance for each digital output point available on the device. However, the value of the status is the same for all the given points on a particular I/O module. Other attributes include: Value, Status, Fault State, Fault Value, Idle State and Idle Value.

C. Analog Input Point (AIP) Object (Class 0x0A hex)

The AIP object models analog inputs in the DeviceNet™ bus coupler. There is a separate instance for each analog input point available on the device. Attributes include: Value, Status and Range.

D. Analog Output Point (AOP) Object (Class 0x0B hex)

The AOP object models analog outputs in the DeviceNet™ bus coupler. There is a separate instance for each analog output point available on the device. Attributes include: Value, Output Range, Value Data Type, Fault State, Idle State, Fault Value and Idle Value.

E. Accessing Analog and Digital Instances 1 through 510

The bus coupler automatically supports the following number of instances for the specific object types when accessed using produced/consumed data that is mapped to a scanner.

- Digital Inputs 510 instances
- Digital Outputs 510 instances
- Analog Inputs 510 instances (128 max. in poll)
- Analog Outputs 510 instances (128 max. in poll)

However, if explicit messages are being used retrieve I/O data the user will need to access the Configuration Object (Class Code 100 dec, 0x64 hex), attribute 33 to determine if instances 1 through 255 or 256 through 510 can be accessed. (There is no bank select for special function data.)

By default, Attribute 33, decimal (I/O Bank Select) is set to a 0, which allows the user to explicitly access instances 1 through 255. If the user needs to access instances 256 through 510 explicitly, the I/O Bank Select (Attribute 33) must be set explicitly in the Configuration Object (Class Code 100 dec, 0x64 hex).

Example: To access the 256th digital input point, using an explicit message, proceed as follows:

1. The user must set the I/O Bank Select to a 1.
2. A Get (Get_Attribute_Single) to Instance 1 of the DIP object accesses the value for Instance 256.

Note

Setting the bank select to a 1 adds 255 instances to the instance called out in the explicit message.

F. Inline Special Function Object (Class 0x43 hex)

The Inline special function object gives the user the ability to control and monitor the below listed modules and any other module that doesn't map to a standard DeviceNet™ object.

- Incremental encoder
- Absolute encoder
- High Speed counter

G. PCP Special Function Object

By default, the PCP Special Function Object contains one instance for each PCP module. If the bus coupler detects that the modules is designed for serial communications, it will also create an instance in the Serial communications Object. An example of a PCP module is:

- AS-i master

H. Serial Communications Object

The Serial Communications Object contains one instance for each PCP module that is designed for serial communications. It is possible to access an instance of the Serial Communication Object into an instance in the PCP Special Function Object. Examples of Serial Communications PCP modules are:

- RS 232
- RS 485

3.7 Fault Response Modes

Fault response is a configurable setting that will control the behavior of the Inline station in the event of one of the 3 below listed errors occur:

A. Module Change Error

Bit 3 in the Inline status word

B. Configuring Error

Bit 4 in the Inline status word

C. Module Connection Error

Bit 5 in the Inline status word

The bus coupler has the ability to run in any of the fault response modes described below.

Note

The use of Loop 2 I/O will only allow the station to operate in modes 0 or 1. If you are running your station in modes 2 or 3 and Loop 2 is added, the mode will automatically change to 0.

D. Fault Response Mode 0: Stop on Fault (Default)

Outputs are turned OFF automatically when a local failure occurs. Once the failure(s) is fixed, either a reconfiguration (Class 64 hex, Instance 1, Attribute 6) or a power cycle (U_L and DeviceNet™ power) needs to occur before the station becomes operational again.

E. Fault Response Mode 1: Auto Restart

Outputs are turned OFF automatically when a local failure occurs. Once failure is fixed, the station will begin to update I/O automatically.

F. Fault Response Mode 2: Goto Fault State

Outputs are turned OFF automatically for up to two seconds when a local failure occurs. After two seconds the station will attempt to set the outputs to the pre-programed DeviceNet™ fault states. The station will not recover its lost I/O, even after the fault condition has been cleared, without a reconfiguration or power cycle (U_L, DeviceNet™). The default for a preprogrammed fault state is OFF.

G. Fault Response Mode 3: Continue on Fault

Outputs are turned OFF automatically for up to two seconds when a local failure occurs. After two seconds the station will continue to update all I/O that is still on line. The station will not recover it's lost I/O, even after the fault condition is cleared, without a reconfiguration or power cycle (U_L, DeviceNet™).

H. Changing the Fault Response Mode

Note

Fault response mode is a flash memory setting in the bus coupler. A fault response mode cannot be changed by reconfigurating the I/O station or by cycling power (U_L, DeviceNet™).

Changing the fault response from the default (mode 0) to a different mode can be accomplished in two ways: (1) an explicit message is sent to the Configuration Object (described in Appendix A); (2) EDS Parameter 26 (Fault Mode) is set to desired fault response mode number (described above) then downloaded to the bus coupler.

SECTION 4
Diagnostics

Section 4 Contents

4.1 General

This section provides information about the diagnostic and status indicators for Inline modules. It also provides information about Inline module diagnostics that are available over the network.

4.2 Diagnostic and Status Indicators

All modules are provided with diagnostic and status indicators for rapid local error detection.

A. Diagnostics Indicators

The diagnostic indicators (red or green) show the state of the Inline modules. When a module is operating normally, all its diagnostic LEDs are green.

After an error is detected, the indicators immediately display the current status.

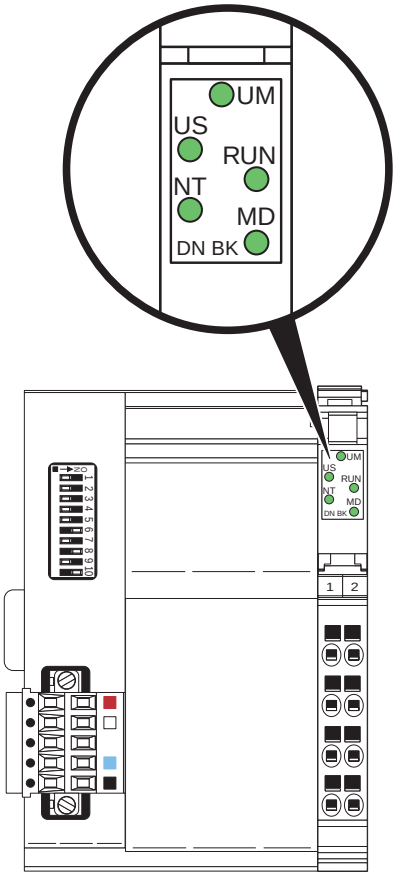
B. Status Indicators

The status LED indicators (yellow) display the status of the relevant inputs/outputs or of the connected device.

Each different type of module has different diagnostic and status indicators. Refer to the module-specific data sheet to see which diagnostic and status LED indicators apply to that module.

4.2.1 Inline DeviceNet™ Bus Coupler LEDs

Figure 4-1 provides the different LED states that can be read from the bus coupler.



LED	Status	Meaning
NT - Network Status (Green/Red LED)		
	OFF:	Not powered/not online
	Flashing Green:	Online, not connected
	Green:	Link OK, online, connected
	Flashing Red	Connection timeout
	Red:	Critical link failure
MD - Module status (Green/Red LED)		
	OFF:	No power present
	Green:	Device operational
	Flashing Green/Red	Initializing
	Flashing Green:	Device needs commissioning
	Flashing Red:	Minor fault
	Red:	Critical fault
RUN - Backplane communications status (multi-color LED)		
	Solid Green:	Communications active
	Flashing Green (1 Hz):	Peripheral fault
	Solid Red:	Communications have stopped
	Red/Amber (1 Hz):	Current configuration does not match the flashed
	Green/Amber (1 Hz)	Outputs being driven with fault state data
	OFF:	U_L supply not present
US - Segment supply (Green LED)		
	ON:	Segment supply present
	OFF:	Segment supply not present
UM - Main Suply (Green LED)		
	ON:	Main supply present
	OFF:	Main supply not present

1436B035

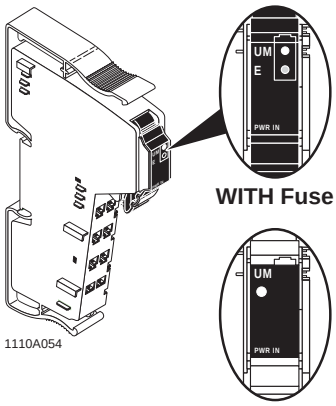
Figure 4-1. Diagnostic LEDs on the Bus Coupler

4.2.2 Power Terminal LEDs

Figure 4-2 provides the different LED states that can be read from the power terminal (with or without fusing).

Note

On terminals with fusing, the green LED indicates that the main or segment voltage is present at the line side of the fuse. If the red LED is also on, there is no voltage on the load side of the fuse.



LED	Status	Meaning
UM	Green LED	Supply voltage
	ON:	Supply voltage present
	OFF:	Supply voltage not present
E	Red LED	Fuse Status
	ON:	Fuse not present or blown
	OFF:	Fuse OK

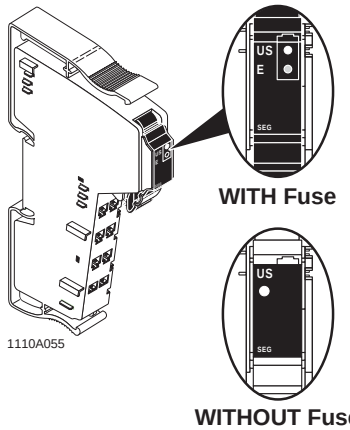
Figure 4-2. Power Terminal Diagnostic LEDs

4.2.3 Segment Terminal LEDs

Figure 4-3 provides the different LED states that can be read from the segment terminal (with or without fuse).

Note

On terminals with fusing, the green LED indicates that the main or segment voltage is present at the line side of the fuse, meaning that when the green LED is on, voltage is present at the line side of the fuse. If the red LED is also on, there is no voltage on the load side of the fuse.



LED	Status	Meaning
US	Green LED	Supply voltage in segment circuit
	ON:	Supply voltage present
	OFF:	Supply voltage not present
E	Red LED	Fuse Status
	ON:	Fuse not present or blown
	OFF:	Fuse OK

Figure 4-3. Diagnostic LEDs on the Segment Terminal

4.2.4 I/O Module LEDs

All LEDs of input/output modules are electrically located in the logic area, not the I/O circuit. Figure 4-4 and Figure 4-5 shows most of the I/O modules and cover the position of the different LED status indicators that can be viewed from an I/O module.

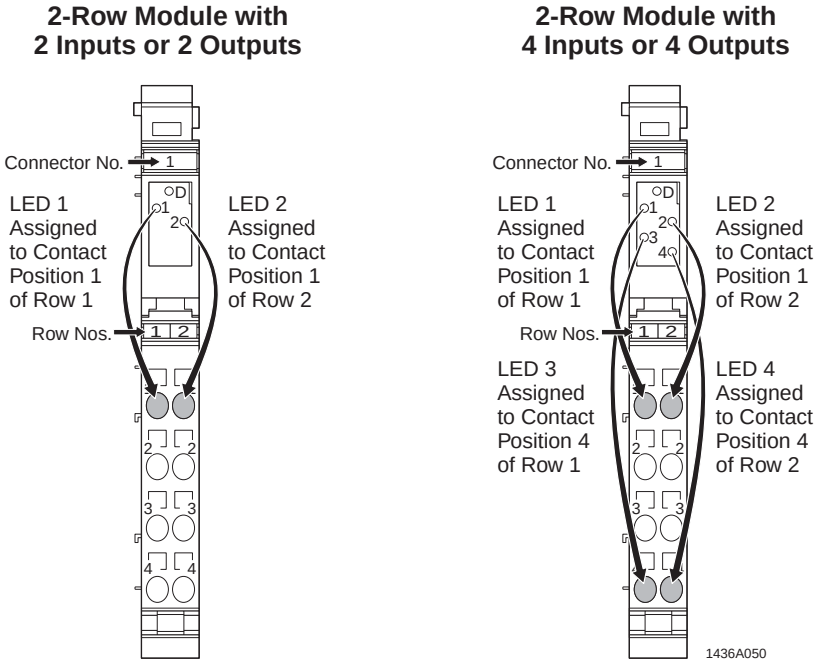


Figure 4-4. LED Assignments for 2-Row Modules

4.2.5 Error Localization Using LEDs

Inline diagnostic indicators clearly denote the location of errors. If an error is detected, the error is displayed at the station and the device on which the error has occurred is reported to the control system. Figure 4-6 shows an Inline station with a localized error at module 4.

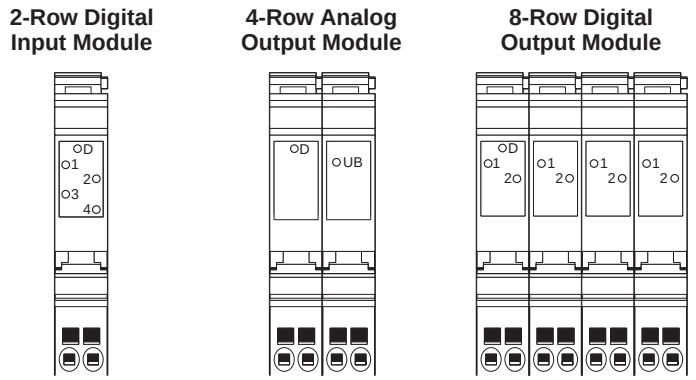
Note

IB IL 24 PWR IN power terminals are not numbered because they are not bus devices (they do not contain protocol chips) and therefore do not have indicators for error diagnostics.

A. Determining Errors

Figure 4-7 shows three possible scenarios for determining an error at module 4. Each of these possibilities is described below. Table 4-1 describes the effects of I/O and bus errors.

- If there are no errors, the green LEDs on the bus coupler and the other modules will remain lit.
- If the LED on module 4 is flashing at medium speed (2 Hz), a DO module will indicate an I/O error (maybe a short circuit caused by a defective actuator).
- If the LED on module 4 is flashing fast and the LEDs on modules 3 and 5 are flashing slow, it indicates a bus error on module 4 or between modules 3 and 4.



I/O "D" Diagnostic LEDs (Green)

LED	Status	Meaning
D	Green LED	Diagnostic
	ON:	Device ready
	Flashing (slow, 0.5 Hz)	Communications power present, no local communications
	Flashing (medium, 2 Hz)	Communications power present, peripheral fault
	Flashing (fast, 4 Hz)	Communications power present; module to left of flashing module has failed; modules to right of flashing module are not part of configuration frame
	OFF:	Communications power not present
UB	Green LED	I/O voltage/current is present

I/O Status LEDs (Yellow)

LED	Status	Meaning
1, 2, 3 & 4	Yellow LED	Status of input/output
	ON:	Relevant input/output set
	OFF:	Relevant input/output not set

1436A051

Figure 4-5. LEDS on 2-Row Digital, 4-Row Analog and 8-Row Digital Module

Table 4-1. Example of Error Effects on I/O and Bus

I/O Error
Error — Short circuit on module no. 4 (IB IL 24 DO 4) Effect: Control System: Error message to the control system Bus Coupler: Indicators remain unchanged Module No. 4: Green "D" LED flashes at 2 Hz (medium) Other modules: Remain unchanged

Bus Error
Error — Incoming bus after module no. 3 and before module no.4 has been interrupted Effect: Control System: Error can be located by the control system Module No. 4: Green "D" LED flashes at 4 Hz (fast) Other Modules: Green "D" LED on all other modules flash at 0.5 Hz (slow)

1436A052

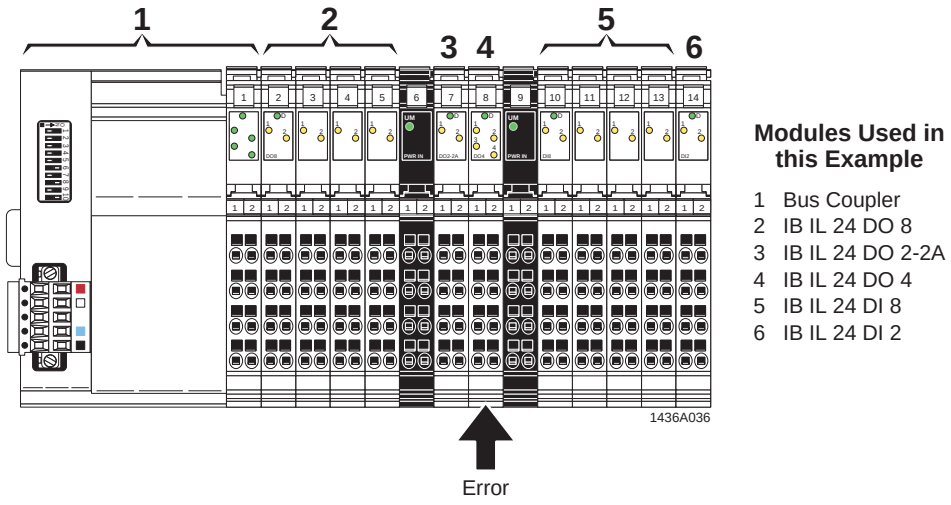


Figure 4-6. Example Station for Error Localization

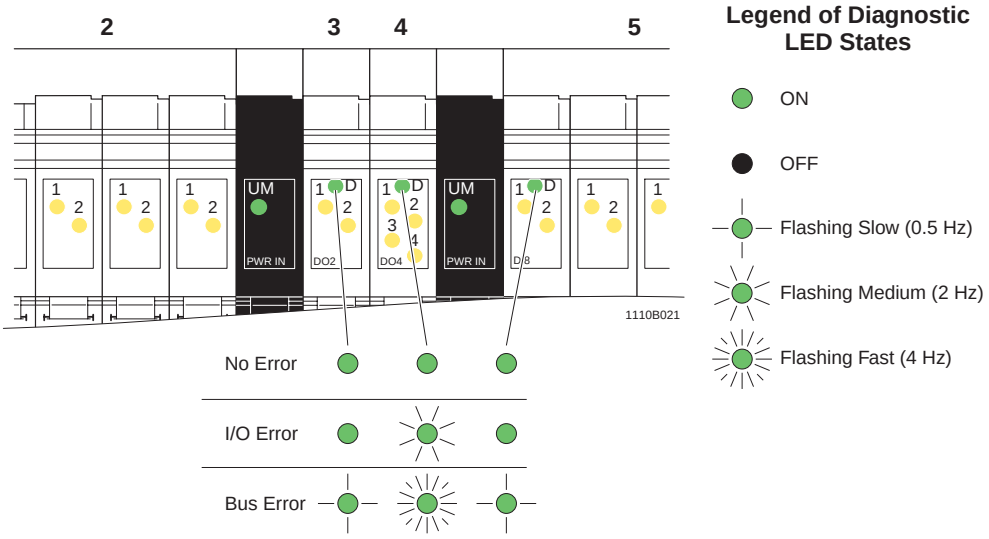


Figure 4-7. Determining Errors Using Various Diagnostic LED Patterns

4.3 Available Network Diagnostics

Note

A detailed explanation of object classes can be found in Appendix A.

Diagnostic information is made available through several mechanisms. The EDS file allows the user to read the Inline Status Word and condition of the standard DIP's, DOP's, AIP's, AOP's and special function modules. This Inline Status Word and I/O point/channel status can also be mapped directly to the polled I/O. The final method of retrieval is to explicitly query the attributes from the Configuration Object (Class Code 0x64 hex) and Inline Object (Class Code 0x65 hex).

4.3.1 Inline Status Word

By default, Inline Status Word data is made available to the user as two bytes of diagnostic data in the polled scan. These two bytes contain the Inline fault code (byte 0) and the number of the first module in the local bus that is faulted (byte 1). The status word adds 2 bytes to the produced data size by default. The status word updates status automatically so when an error is cleared the status will be set back to a 0.

If the user needs to remove this data from the poll, the status word can be disabled by sending a 0 in the "Use Inline Status" parameter during an "Add All I/O" using either the EDS file or an explicit message to the Configuration Object (Class Code 0x64 hex.).

4.3.2 Major/Minor Faults

By default all Inline Status Word fault bits (byte 0, bits 0, 2-6) except for bit 1 are considered major recoverable faults, as defined in the Identity Object state and status attributes, and will flash the red MD LED on the bus coupler when that specific type of failure occurs. Bit 1, peripheral fault, is assigned as a minor recoverable (default value) fault and will not flash the MD LED when in a faulted condition. These default values can be changed by using the EDS file or by sending an explicit message to the Configuration Object (Class Code 0x64 hex). Possible fault action setting are:

- 0 None
- 1 Minor Recoverable Fault
- 2 Major Recoverable Fault

4.3.3 Bit Meanings for Inline Status Word (Byte 0)

A. Bit 0 CRC Error

The CRC error bit will be set when a data transmission error occurs due to unwanted interference on the Inline local bus. The EDS parameter number 23, "Max Retry", will allow the module to retransmit the data cycle up to the number of times that the "Max Retry" parameter is set to. If the transmission does not pass the CRC after the "Max Retry" has expired then the CRC error bit is set.

B. Bit 1 Peripheral Fault

The Peripheral Fault bit will be set when any output is shorted or a loss of power to an intelligent segment module.

C. Bit 2 Power Fault

The Power Fault bit will be set when any of the power supplies (U_L, U_S, UM, DeviceNet™) are in an under voltage condition (less than 11 V dc).

D. Bit 3 Module Change

The Module Change bit will be set when the configuration present on the Inline local bus does not match the configuration that was stored in flash during the last configuration cycle.

E. Bit 4 Configuring Error

The Configuring Error bit will be set when the bus coupler is not able to talk to the first I/O module connected to it. Possible failures include the bus coupler itself or the first I/O module connected to it. Power down and reconnect the I/O to the bus coupler.

F. Bit 5 Module Connection Error

The Module Connection Error bit will be set when the bus coupler is no longer able to talk to the modules connected to it and can determine the failure position. This failure occurs due to a broken data path. The exact path "between what two modules" can be read from the Inline Interface Object (Class Code 0x65 hex).

G. Bit 6 Outputs Set to Preprogram DeviceNet™ Fault State

This bit can only be set in the Fault Response mode 2 (described in Section 3). It is made available to let the application know that the local outputs have gone to their preprogrammed DeviceNet™ fault state and will no longer respond to the controller.

H. Bit 7

Reserved for future use

4.3.4 Bit Meanings for Inline Status Word (Byte 1)

Contains the first failed device number. The device number determines the position on the Inline station where a failure or warning has occurred. These positions are numbered starting at the bus coupler being assigned with a 1. The numbering will continue to the right up to 64, which is the maximum number of devices that can be connected to an Inline station (63 I/O devices + 1 bus coupler).

Notes

Inline local errors will not be sent over the network unless the Inline Status Word is in the poll or an explicit message to the Inline Object is sent periodically.

These errors by default are considered a major (except for a peripheral fault) error and the MD LED on the bus coupler will blink red. A determination must be made regarding the Inline Status Word and it's desired effect on the network and/or failing node through the users application.

4.3.5. Latched Diagnostics

The bus coupler will latch the last occurring Inline Status Word fault, module number and connection point 1 and 2 failures. This benefits the user by capturing any fault that may be occurring intermittently and that is occurring too quickly to be updated by the DeviceNet™ poll or by an explicit message. These latched values can not be cleared until the station is reconfigured. The following latched diagnostics are available through the EDS file or by sending an explicit message to the Inline Interface Object (Class Code 0x65 hex).

- **Latched Inline Status Word:** This parameter will contain the last reported Inline station failure. The bit weights signify the same failures as described in the Inline Status Word (byte 0).
- **Latched Faulted Module:** This parameter will contain the first failed module location that was reported during the last Inline station fault. The bit weights signify the same failures as described in the Inline Status Word (byte 1).
- **Latched Connection Failure Endpoint 1:** This parameter will contain the number of the module that was reported on the first end of a connection failure.
- **Latched Connection Failure Endpoint 2:** This parameter will contain the number of the module that was reported on the other end of a connection failure.

4.3.6 Inline Control Byte

The Inline Control Byte is used to acknowledge latched peripheral faults (bit 0) or to clear latched inputs states (bit 1).

Note

For an explanation of latching input states, refer to I/O Data Transfer described in Section 3.

By default, the Inline Control Byte is not added to the poll. It can be added by setting Instance 1, Attribute 32 of the Configuration Object (Class Code 0x64 hex) to a 1. If the user would rather access this byte through an explicit message, a Get or Set can be sent to the Inline Interface Object (Class Code 0x65 hex), Instance 1, Attribute 20.

- **Bit 0:** When set to a 1, will attempt to clear all latched peripheral faults.
- **Bit 1:** When set to a 1, will clear all latched input states.

The latched peripheral fault can only be generated by certain Inline modules. Examples of this type of module are the IB IL SEG-ELF and the IB IL 24 EDI 2-DESINA.

4.3.7 I/O Point/Channel Status

The fault status of a digital, analog, or special function point is either 0 (functioning) or 1 (failed). The fault status can be added to the poll through the EDS file or solicited by issuing an explicit message to the Configuration Object (Class Code 0x64 hex), see Appendix A. When adding this status to the poll, a bit for each point or channel will be assigned to the input image. This will occur before the mapping of the actual point or channel. The mapping assignments of these status bits will occur in order of their instance on the local bus.

I/O status bits can be added to the poll through the use of the EDS file. Parameters 16 through 20 (described in the following paragraphs) allow for respective status bits to be added to the poll.

A. Parameter 16

Selects the number of DIP faults added to the poll response on a point basis.

B. Parameter 17

Selects the number of DOP faults added to the poll response on a point basis.

C. Parameter 18

Selects the number of AIP faults added to the poll response on a channel basis.

D. Parameter 19

Selects the number of AOP faults added to the poll response on a channel basis.

E. Parameter 20

Selects the number of special function faults added to the poll response on a channel basis.

4.3.8 Fault/Idle State and Value

The bus coupler supports the standard DeviceNet™ DOP (Discrete Output Points), AOP (Analog Output Points), fault or idle states, and values. These values can be set and read by the use of an explicit message. Fault states will only occur during a network error. They will not occur after an Inline local error. The default value for the AOPs and DOPs is zero. Idle states and values will occur when a PLC is taken out of the run state. The default value for the idle state is also zero.

A. Digital Output Support:

- Holds last state
- Turn off during a faulted condition (default)
- Turn on during a faulted condition

B. Analog Output Support:

- Hold last value
- Set to low limit
- Set to high limit
- Set to value determined by the fault value attribute

Note

Appendix A will detail the DOP (Class Code 0x09 hex) and AOP (Class Code 0x0B hex) fault/idle values and states.

4.3.9 Analog Input, Thermocouple and RTD Fault Codes

Inline analog inputs, thermocouples and RTDs can report diagnostic codes. These codes must be read from the produced response or the AIP detailed in Appendix A. A list of these codes, in Inline format "IL", is shown in Table 4-2.

Note:

Error codes are dependent on the type of module and how that module's format is configured. By default error codes are received in the Inline "IL" format and can be viewed as shown in Table 4-2. If the format has been changed, the user must refer to the module specific data sheet to determined what error code has been received.

Table 4-2. Error Messages of Analog Input Modules

Code (hex)	Error Message
8001	Under-range
8002	Open Circuit
8004	Measured value invalid
8008	Cold junction defective
8020	I/O Supply Voltage Faulty
8010	Configuration invalid
8040	Module defective
8080	Over-range

1436A053

4.3.10 Error History

Error history provides access to the last ten errors that have been stored in the bus coupler. These errors can be accessed using either the EDS file (Parameter 56 (most recent) thru Parameter 65 (oldest) or by using the Inline Interface Object Class 101, Instance 1, Attribute 21 (most recent) thru attribute 30 (oldest). As error values are added, existing values will be shifted to older parameters. The new value is then placed in the "most recent" parameter and the value in the "Last Saved" parameter is discarded.

The error history entry will contain the faulted module number in the high byte and the Inline status code in the low byte.

Note

A "0" for an error history entry represents a point where an error was removed.

An error history value may be recorded at a point where an error is detected but is not yet localized. When the error is localized, a new error history value will be added.

SECTION 5
Technical Data

Section 5 Contents

5-1 General

This section provides technical information for:

- Inline DeviceNet™ System
- Loop 2 Products
- Inline I/O System

Data presented in the following tables is based on the product being mounted in the preferred vertical mounting position. Please refer to the DeviceNet™ specification and to the module-specific data sheets for additional information or to confirm the accuracy of technical data presented.

Phoenix Contact reserves the right to make any necessary changes that advances the technology and performance of the product.

Table 5-1. General Bus Coupler Data

Order designation	IL DN BK3-PAC
Order number	27 18 78 5
Housing dimensions (width x heigh x depth)	91 mm x 120 mm x 71.5 mm
Degree of protection	IP20 according to IEC 60536
Class of protection	Class 3, according to VDE 0106, IEC 60536
Weight	210 g (without connector)
Certifications	CE. cURus (File: E140324), cULus (Class 1, Div 2, File: 199827)

1436B065

5.2 Inline DeviceNet™ System

Table 5-2. Inline DeviceNet™ System Data

Transmission speed	500 kbits/s, 250 kbits/s, 125 kbits/s
Transmission reliability	CRC check
Maximum distance	500 m (1640 ft)
Protocol	control and information (CIP) with CAN as the transport layer
Nodes	64 maximum
Bus coupler voltage range (+V, -V)	11-28 V
Bus coupler nominal current (+V, -V)	45-50 mA
Bus coupler certification	ODVA

1436B062

5.3 Loop 2 System

Table 5-3. Loop 2 System Data

Number of devices in Loop 2	63 max.
Total length of Loop 2	200 m (656 ft)
Distance between 2 modules or between branch terminal and I/O	20 m (65.6 ft) max.
Maximum current of the branch terminal module in the logic area	2 A
Maximum current consumption of the I/O modules	1.8 A
UL1604, Class 1, Div 2, Groups A,B,C & D	

Notes:
1. Not all of the limit values can be met at the same time when configuring Loop 2. The maximum extension of Loop 2 is reached when one of the limit values is reached.
2. For further technical data on Loop 2, refer to the Loop 2 manual and product-specific data sheets.

1436A063

5.4 Inline I/O System

Table 5-4. Inline I/O System Data

Number of I/O devices in an Inline system	63 max.
Maximum current of the bus coupler module in the logic area (UL)	2 A
Maximum current consumption of the I/O modules	see module-specific data sheet
Maximum current carrying capacity of the voltage jumper U_{ANA}	0.5 A
Maximum current carrying capacity of the voltage jumpers U_M and U_S (total current)	8 A

Note:
Observe the current consumption of every device on the individual voltage jumpers when configuring an Inline station. The logic current consumption is values are listed in each terminal-specific data sheet. Current consumption can differ, depending on the type of module. If the maximum current carrying capacity of a voltage jumper (8 A) is reached, an additional power terminal must be used.

1436A064

Table 5-5. Bus Coupler Emissions Tests

Housing emissions	
Radiated	EN55011, Group 1, Class A
Conducted	EN55011, Group 1, Class A
Harmonic disturbances	EN61000-3-2

1436A059

Table 5-6. Bus Coupler Immunity Tests

Immunity Test Regulations: EN61000-6-2:1999, In accordance with:	
Voltage fluctuations and flickers	EN61000-3-3
Electrostatic discharge (ESD)	EN 61000-4-2, Criteria B
Air discharge	±8 kV
Contact discharge	±6 kV
Continuous radiated fields	EN61000-4-3
Electrical fast transients (EFT) / bursts	EN 61000-4-4, Criteria B
AC/DC power ports	4 kV
Signal port	2 kV
Conducted immunity	EN61000-4-6, Criteria B, 10 V
Voltage surge	EN 61000-4-5, Criteria B
DC power port (differential and common)	±2 kV, ±1 kV, ±0.5 kV
Shield for Signal lines	±2 kV, ±1 kV, ±0.5 kV
Power frequency magnetic fields	EN 61000-4-8, Criteria A, 50 Hz, 30 A/m
Voltage dips and interruptions	EN 61000-4-11, Criteria B

1436A060

Table 5-7. Ambient Conditions

Regulations	developed according to VDE 0160, UL 508
Ambient temperature, operating	-25°C to +55°C
Ambient temperature, storage/transport	-25°C to +85°C
Temperature cycles (speed of changing from positive-to-negative and vice versa)	0.5 K/min. (no condensation)
Humidity, operating	75% on average; 85% occasionally (no condensation)
Note: Ranging from -25°C to +55°C, appropriate measures must be taken against increased humidity (>85%).	
Humidity, storage/transport	75% average; 85% occasionally
Note: For a short period, slight condensation may appear on the housing, if for example, the terminal is brought into a closed room with a vehicle.	
Degree of protection according to IEC 60529	IP20
Degree of protection according to IEC 60536	Class 3
Housing material, basic	plastic Arnite PVC-Free PA6.6, self-extinguishing (V0)
Degree of pollution according to EN 50178	2; condensation not permissible in operation
Surge voltage class	II (low-level signal); III (power level)
Gases that endanger the functions (according to DIN 40046-36, DIN 40046-37)	
Sulphur dioxide (SO_2) concentration 10 ± 0.3 ppm. Ambient conditions:	
Temperature	25°C (± 2°C)
Humidity	75% (± 5%)
Test duration	10 days
Hydrogen sulfide (H_2S) concentration 1 ± 0.3 ppm. Ambient conditions:	
Temperature	25°C (± 2°C)
Humidity	75% (± 5%)
Test duration	4 days
Resistance of the housing material to termites	resistant
Resistance of the housing material to fungi	resistant

1436A072

Table 5-8. Mechanical Data

Vibration test	5 g load, 2 hours for each space direction (low-level signal)
Sinusoidal vibrations according to IEC 60068-2-6	2 g load, 2 hours for each space direction (power level)
Shock test according to IEC 60068-2-27	30 g load for 11 ms, half sinusoidal wave, three shocks in each space direction and orientation

1436A061

Table 5-9. I/O Connection Specifications

Connection type used on all modules	spring clamp
Cable diameter, low-level signals (typical)	0.2 mm^2 up to 1.5 mm^2 (16 to 24 AWG)
Cable diameter, low-level signals (connection of equalizing conductors for thermocouples to the IB IL TEMP 2 UTH module)	0.13 mm^2 up to 1.5 mm^2 (16 to 24 AWG)
Cable diameter power-level (power terminal, motor connection, brake connection)	0.2 mm^2 up to 2.5 mm^2 (14 to 24 AWG) (flexible and inflexible cable types)
Cable diameter power-level (manual mode)	0.14 mm^2 up to 1.5 mm^2 (16 to 24 AWG) (flexible and inflexible cable types)

1436A071

Table 5-10. I/O Module Parameters for Low-Level Signals
(Does not Include Special Function Modules)

Parameter	Minimum Value	Maximum Value
Digital module input/output voltage Digital module input/output current	18.2 V dc 0.1 mA	253 V ac 5 A
Analog module input/output voltage Analog module input/output current	0 V 0 A	50 V 40 mA

1436A068

Table 5-11. U_L (7.5 Volt Supply of the Bus Logic)

Nominal voltage .....7.5 V (converted from external 24 V dc)
Ripple .....± 5%
Load current .....2 A maximum
Connection .....voltage jumpers on the sides of the module housing

Notes:

1. Voltage is produced in the bus coupler by a DC/DC converter from the 24 V supply voltage.
2. U_L is not electrically isolated from the 24 V bus coupler voltage.
3. U_L is not electrically isolated from the I/O voltage U_M and U_S.
4. Communications power U_L is electronically short-circuit protected.

1436A067

Table 5-12. U_M & U_S (Supply of Modules for Digital Signals in the 24 Volt Range)

Nominal voltage.....24 V dc
Tolerance .....-15%/+20%
Ripple .....± 5%
Permissible voltage range .....19 V dc to 30 V dc, ripple included
Load current .....8 A maximum
Connection .....voltage jumpers on the sides of the module housing

Notes:

1. Segment circuit U_S — All digital outputs and initiator supplies without individual short- circuit protection are connected to the segment circuit U_S.
2. Main circuit U_M — Initiator supplies with individual short-circuit protection are connected to the main U_M.

1436A066

Table 5-13. U_{ANA} (Supply of Modules for Analog Signals)

Nominal voltage.....24 V dc
Tolerance .....-15%/+20%
Ripple .....± 5%
Permissible voltage range .....19 V dc to 30 V dc, ripple included
Load current .....500 mA maximum
Connection .....voltage jumpers on the sides of the module housing

Notes:

1. Isolation of the 24 V input voltage by means of a diode.
2. Smoothing through π-filter; corner frequency 9.8 kHz and attenuation of 40 dB/decade.
3. U_{ANA} is not electrically isolated from the 24 V bus coupler supplies and the 7.5 V communications power.

1436A069

5.5 Inline Accessories Ordering Information

Table 5-14 provides most of the tools, accessories, and spare parts you will need to develop an Inline DeviceNet™ station.

To order various Inline modules, refer to the latest Phoenix Contact catalog. Manuals and data sheets for Inline I/O products can be obtained from the Phoenix Contact internet web site at: www.phoenixcon.com (select Info Service)

Table 5-14. Inline Connectors and Accessories Parts List

Description	Type	Order No.	Pcs/Pkt
Coding Profile (see COMBICON catalog)	CP-MSTB	17 34 63 4	100
Zack "Quick" Marker Strips for labeling terminals (see CLIPLINE catalog)	Marking Strips	ZBFM6..	—
Labeling Field, 2-row width	IB IL Field 2	27 27 50 1	10
Labeling Field, 8-row width	IB IL Field 8	27 27 51 5	10
DIN-Rail, perforated	NS 35/7.5	08 01 73 3	2 meters
DIN-Rail, unperforated	NS 35/7.5	08 01 68 1	2 meters
End Clamp	CLIPFIX 35	30 22 21 8	50
Grounding Terminal Block, 10 to 24 AWG	USLKG 5	04 41 50 4	50
Screwdriver, blade type, 3.5 mm wide	SZF 1	12 04 51 7	1
Wire stripper, 10 to 30 AWG	QUICK-WIREFOX 6	12 04 38 4	1
Wire stripper, up to 25 mm insulation dia.	KAMES 2	12 06 00 7	1
Power supply, 120 V ac to 24 V dc, 2.5 amp	QUINT-PS 120AC/24DC/2.5	29 39 05 6	1
Power supply, 120 V ac to 24 V dc, 5 amp	QUINT-PS 120AC/24DC/5	29 39 06 9	1
Power supply, 120 V ac to 24 V dc, 10 amp	QUINT-PS 120AC/24DC/10	29 39 07 2	1
Power supply, 120 V ac to 24 V dc, 1 amp	QUINT-PS 120AC/24DC/1	29 39 24 7	1
Power Connector, with color code	IB IL SCN-PWR IN-CP	27 27 63 7	10
Power Connector, without color code	IB IL SCN-PWR IN	27 27 46 2	10
I/O Connector, 2 signals, 4-wire connection, without color code	IB IL SCN-8	27 26 33 7	10
I/O Connector, 2 signals, 4-wire connection, color coded	IB IL SCN-8-CP	27 27 60 8	10
I/O Connector, 4 signals, 3-wire connection, without color code	IB IL SCN-12	27 26 34 0	10
I/O Connector, digital input modules, 4 signals, 3-wire connection, color coded	IB IL SCN-12-ICP	27 27 61 1	10
I/O Connector, digital output modules, 4 signals, 3-wire connection, color coded	IB IL SCN-12-OCP	27 27 62 4	10
I/O Connector with shield clamp	IB IL SCN-6 SHIELD	27 26 35 3	5
Network Connector	MSTBP 2,5/5-STF-5.08 AU PRTD	28 62 57 6	50

1436A070

SECTION 6
Serial and Other PCP Inline Modules

Section 6 Contents

6.1 General

This section provides the following information on the use of the Inline serial modules and any other module that uses the Peripheral Communications Protocol (PCP):

- Communication methods for the Inline serial modules (e.g. RS232 or RS485/RS422)
- Communication methods for any other module that supports the PCP protocol.
- Serial and Generic PCP Modules Produced and Consumed Sizes
- I/O memory mapping
- RS232 and RS485/RS422 configuration brief

6.2 Communications Methods

Communications to an Inline serial module can be accomplished in two ways. Likewise, communications to a generic PCP module can be accomplished in two ways. The type of module being used, either serial or generic PCP, determines what types of communication methods are available to the user (by default).

The first method available to the user is the sending or receiving of data using process/cyclic data channel (fragmentation) and the second is by sending explicit messages. Choosing between the two methods is a matter of the capabilities of the DeviceNet™ scanner and/or personal preference.

When using the Inline RS232 or RS485/RS422 modules, simplified methods 1 or 2 can be used. If using any module (including serial) that supports PCP, methods 3 or 4 can be used.

Note

Serial modules are not supported under mapping revision 0.

Method 1. Transfer of Serial Data Using Serial Fragmentation
Method 2. Transfer of Serial Data Using Explicit Messages
Method 3. Transfer of Generic PCP Data Using PCP Fragmentation
Method 4. Transfer of Generic PCP Data Using Explicit Messages

Supported serial modules are:

- IB IL RS 485/422 (Inline RS485/422 module)
- IB IL RS 232 (Inline RS232 module)

An example of a “Generic” PCP Inline module is the:

- ASI MA IB IL (Inline AS-i Gateway)

Notes

1. PCP is not required when using the ASI MA IB IL, if the AS-i branch has less than 32 slaves.

2. If the PCP module looses power, a reset service, with a data value of "1", can be issued to the Inline Interface object (101 dec, 0x65 hex) or the BK can be power cycled.

3. You can access a serial module as a PCP Special Function module. To do this, first disable the module's instance in the Serial Communication Object. Then, remove the module's serial process and fragment data from the poll using the Serial Communication Object. Finally, add the PCP special function process and fragment data into the poll using the PCP Special Function Object.

CAUTION

If the autoconfiguration switch is ON, at next power-up the BK will erase any custom settings described in the note 3 above.

The bus coupler can hold up to 64 bytes of incoming and 64 bytes of outgoing data before data will overflow internal buffers.

6.2.1 Method 1. Transfer of Data Using Serial Fragmentation

This method defines how serial data is exchanged using process/cyclic data. The messages that are required to read and write data are encoded into the high-speed data stream. The protocol is handled using a series of message fragments that are initiated by a client request and then followed up with a server response.

Each fragment contains 8 bytes. Every fragment includes a control byte for a request (output data) and a status byte for a response (input data).

Note

These fragments were specifically designed for the serial Inline modules and will not work for other PCP based Inline modules.

Depending on the amount of data to be sent, the number of fragments required to read/write data can vary. Fragments are eight bytes in length. Each fragment contains a request format and a response format. These formats are detailed in the following paragraphs.

6.2.1.1 Format of Fragmented Serial Output Data

Consumed Request Format

Byte 0

Control Byte (See Table 6-1)

Table 6-1. Control Byte

7	6	5	4	3	2	1	0
Reserved	Reserved	Receive Ack.	Transmit Request	Reset Request	Number of Data Bytes to Transmit (Per Fragment)		

1436A082

Bytes 1 thru 7

Data-block, If necessary

Bit Definitions

Bit 7

Reserved

Bit 6

Reserved

Bit 5

Receive Ack (RxAck). This bit must be set to the value of the Receive Request bit (RxReq) in order for the module to receive more data. This tells the module that the “master” has received the data and has processed it.

Bit 4

Transmit Request. This bit must be toggled to signal the module that there is new data to transmit. On devices that cannot ensure data consistency, the user should first set the number of bytes and place the proper data into the TxData mapping before toggling this bit.

Bit 3

Reset Request. When this bit is set, all errors are cleared, the buffers are flushed, and the RxReq and TxAck bits are cleared. This allows re-sync of the protocol.

Bits 2 thru 0

Number of Bytes to Transmit. This tells the module how many bytes of the data are valid and should be transmitted.

6.2.1.2 Format of Fragmented Serial Input Data:

Produced Response

Byte 0

Status Byte (See Table 6-2)

Bytes 1 thru 7

Data-block, if necessary

Table 6-2. Status Byte

7	6	5	4	3	2	1	0
Error	Reserved	Receive Request	Transmit Ack.	Reset Ack.	Number of Data Bytes Received (Per Fragment)		

1436A083

Bit Definitions

Bit 7

Error. When this value is set, an error has occurred such as parity or overrun. The user should query the status parameter for more information.

Bit 6

Reserved

Bit 5

Receive Request. This value is toggled to indicate new data has been received. The user must acknowledge the reception of data by echoing back this value in the Receive Ack bit.

Bit 4

Transmit Ack. When this value is equal to the Transmit Request, it indicates that the output data has been queued into the output buffer. Once they are equal the user can then send more data.

Bit 3

Reset Ack. If a Reset has been requested, this value will be set to 1 to indicate that the serial port has been reset and the buffers have been flushed. This causes the TxAck and RxReq to be reset to 0 allowing re-sync of protocol.

Bits 2 thru 0

Number of Bytes Received. Indicates the number of valid data bytes that are in the data section of the input data.

6.2.1.3 Serial Fragmentation Examples

The following examples will show how to read, write and handle errors using serial process data fragmentation.

A. Fragmented Read

B. Fragmented Write

C. Error Handling, Communications Backplane Break

D. Error Handling, Host Communications Loss

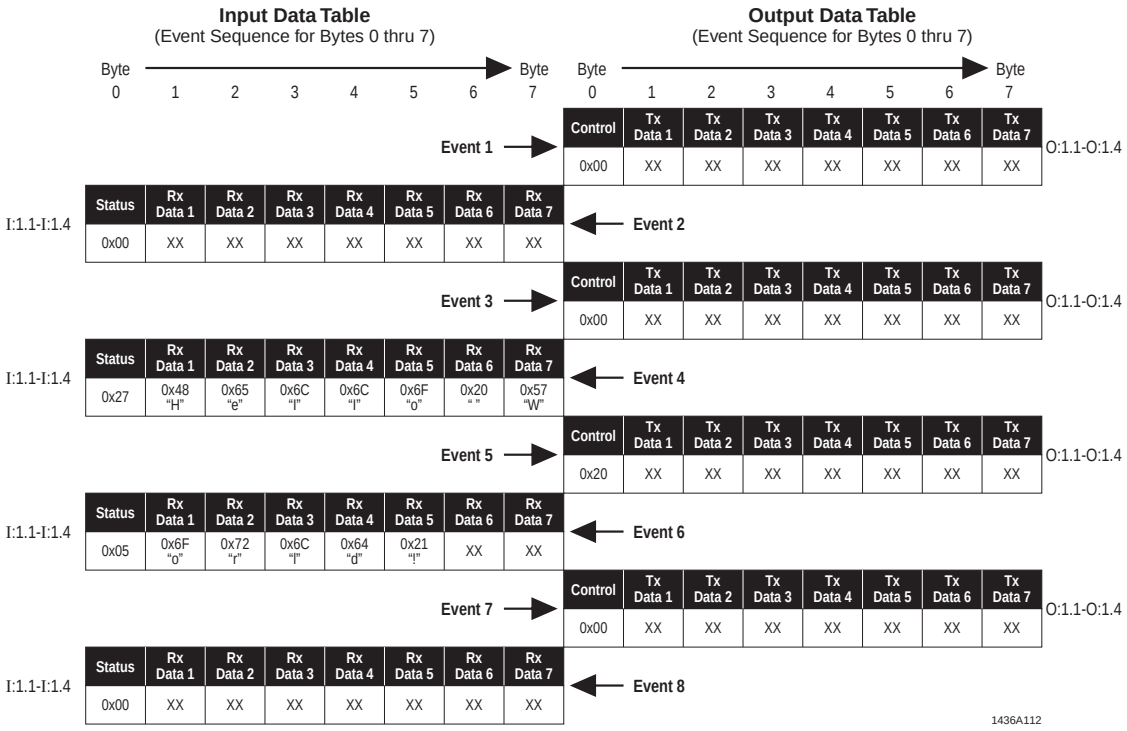
A. Fragmented Read Example

Table 6-3 shows an example I/O data table that contains only eight bytes of input data and eight bytes of output data. This I/O is shown in a event by event sequence that demonstrates how these eight bytes of I/O are updated when reading data using Serial PCP fragmentation. The sequence works on this following basic principle:

Event x. Client issues server 8 bytes of output data

Event x+1. Server responds by updating 8 bytes of input data

Table 6-3. Serial Fragmented Read Example



The following paragraphs explain those events listed in Table 6.3.

Tables 6-4 through 6-10 demonstrate the order of events when issuing a fragmented read service to a PCP device. Each “Event“ should be referenced to the I/O data table shown in Table 6-3.

Event 1

Table 6-4 shows the transmission from a master to slave when the serial fragmentation is in “Idle” mode.

Table 6-4. Master to Slave Idle Transmission

Control	Tx Data 1	Tx Data 2	Tx Data 3	Tx Data 4	Tx Data 5	Tx Data 6	Tx Data 7
0x00	XX	XX	XX	XX	XX	XX	XX

1436A113

Event 2

Table 6-5 shows the slaves response to the idle state by replying with a 0x00.

Table 6-5. Slave to Master Idle Response

Status	Rx Data 1	Rx Data 2	Rx Data 3	Rx Data 4	Rx Data 5	Rx Data 6	Rx Data 7
0x00	XX	XX	XX	XX	XX	XX	XX

1436A114

Event 3

Master still sending the idle message.

Event 4

Table 6-6 shows that the slave has received 7 bytes of new data. This indication is explained as follows:

Status, Byte 0 = 0x27

Bit 5 = 1

Receive Request being toggled Indicates that new data is present.

Bits 2 thru 0 = 7

Shows the number of bytes received.

Table 6-6. Slave to Master Indication that New Data has been Received

Rx Data 1	Rx Data 1	Rx Data 2	Rx Data 3	Rx Data 4	Rx Data 5	Rx Data 6	Rx Data 7
0x27	0x48 “H”	0x65 “e”	0x6C “l”	0x6C “l”	0x6F “o”	0x20 “ ”	0x57 “W”

1436A116

Event 5

Table 6-7 shows the master to slave acknowledgment of a Receive Request indication.

Bit 5 = 1

After the 7 bytes have been received and processed the Receive Ack. Bit is set to the same value as the Receive Request bit to signal the module that the master is ready to receive more data.

Table 6-7. Master to slave, Receive Data Acknowledge

Control	Tx Data 1	Tx Data 2	Tx Data 3	Tx Data 4	Tx Data 5	Tx Data 6	Tx Data 7
0x20	XX	XX	XX	XX	XX	XX	XX

1436A117

Event 6

Table 6-8 shows that the slave has received 5 additional bytes of data:

Status, byte 0 = 0x05

Bit 5 = 0

Receive Request being toggled (In event 6 bit 5 = 0) Indicates that new data is present.

Bits 2 thru 0 = 5

Shows the number of bytes received.

Table 6-8. Slave to Master Indication that More Data is Present

Status	Rx Data 1	Rx Data 2	Rx Data 3	Rx Data 4	Rx Data 5	Rx Data 6	Rx Data 7
0x05	0x6F “o”	0x72 “r”	0x6C “l”	0x64 “d”	0x21 “!”	XX	XX

1436A118

Event 7

Table 6-9 shows the master to slave acknowledgment of a Receive Request indication.

Bit 5 = 0

After the 5 bytes have been received and processed the Receive Ack. Bit is set to the same value as the Receive Request bit to signal the module that the master is ready to receive more data.

Table 6-9. Master to Slave, Receive Data Acknowledge

Control	Tx Data 1	Tx Data 2	Tx Data 3	Tx Data 4	Tx Data 5	Tx Data 6	Tx Data 7
0x00	XX	XX	XX	XX	XX	XX	XX

1436A113

Event 8

Table 6-10 shows the slave to master indication that no more data is present.

Table 6-10. Slave to Master, No More Data Indication

Status	Rx Data 1	Rx Data 2	Rx Data 3	Rx Data 4	Rx Data 5	Rx Data 6	Rx Data 7
0x00	XX	XX	XX	XX	XX	XX	XX

1436A114

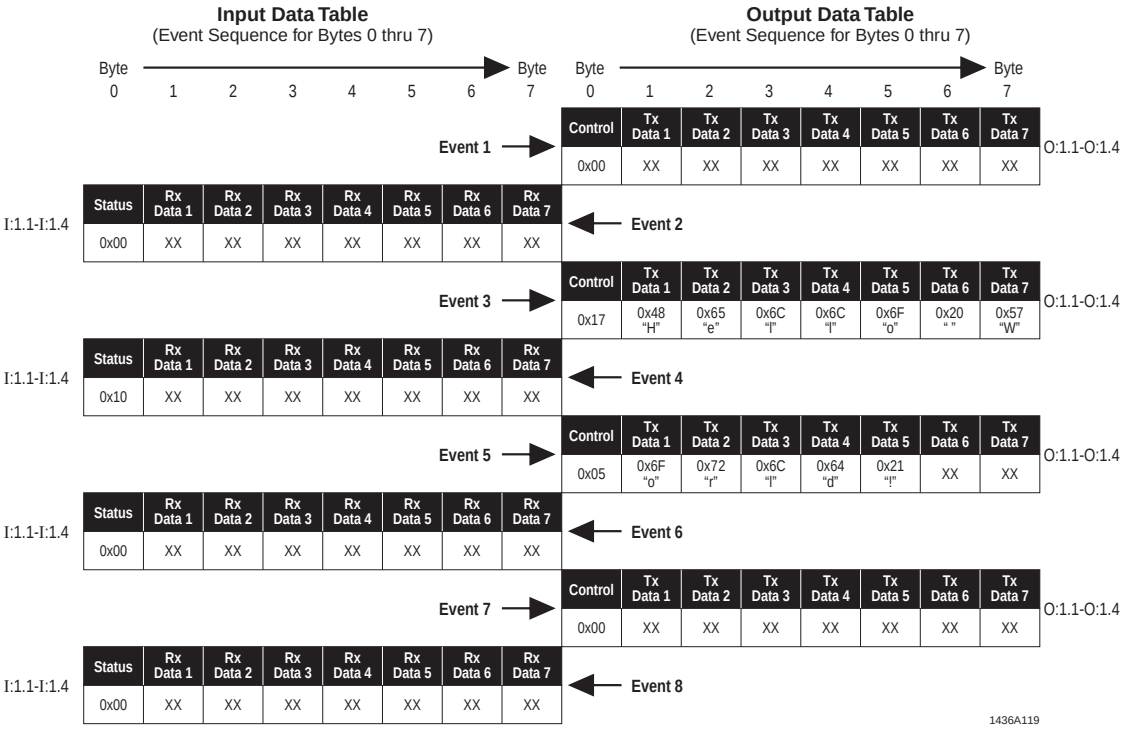
B. Fragmented Write Example

Table 6-11 shows an example I/O data table that contains only eight bytes of input data and eight bytes of output data. This I/O is shown in a event by event sequence that demonstrates how these eight bytes of I/O are updated when reading data using Serial PCP fragmentation. The sequence works on this following basic principle:

Event x. Client issues server 8 bytes of output data

Event x+1. Server responds by updating 8 bytes of input data

Table 6-11. Serial Fragmentation Write Example



The following paragraphs explain in detail those events listed in Table 6.11.

Tables 6-12 through 6-19 demonstrate the order of events when issuing a fragmented read service to a PCP device. Each “Event “ should be referenced to the I/O data table shown in Table 6-19.

Event 1

Table 6-12 shows the transmission from a master to slave when the serial fragmentation is in “Idle” mode.

Table 6-12. Master to Slave Idle Transmission

Control	Tx Data 1	Tx Data 2	Tx Data 3	Tx Data 4	Tx Data 5	Tx Data 6	Tx Data 7
0x00	XX	XX	XX	XX	XX	XX	XX

1436A113

Event 2

Table 6-13 shows the slave's response to the idle state by replying with a 0x00.

Table 6-13. Slave to Master Idle Response

Status	Rx Data 1	Rx Data 2	Rx Data 3	Rx Data 4	Rx Data 5	Rx Data 6	Rx Data 7
0x00	XX	XX	XX	XX	XX	XX	XX

1436A114

Event 3

Table 6-14 shows the master to slave transmission of 7 bytes of data.

Control, byte 0 = 0x17

Bit 4 = 1

Transmit request is toggled indicating that new data is being transmitted.

Bits 2 thru 0 = 7

Shows the number of bytes to transmit.

Table 6-14. Master to Slave Data Transmission

Control	Tx Data 1	Tx Data 2	Tx Data 3	Tx Data 4	Tx Data 5	Tx Data 6	Tx Data 7
0x17	0x48 “H”	0x65 “e”	0x6C “l”	0x6C “l”	0x6F “o”	0x20 “ ”	0x57 “W”

1436A120

Event 4

Table 6-15 shows that the BK has received the data.

Status, byte 0 = 0x10

Bit 4 = 1

Transmit Acknowledge is being set to the same value as Transmit Request to indicate that the module is ready to receive more data.

Bits 2 thru 0 = 0

0 bytes are being received at this time

Table 6-15. Slave to Master Acknowledge of Data Transmission

Status	Rx Data 1	Rx Data 2	Rx Data 3	Rx Data 4	Rx Data 5	Rx Data 6	Rx Data 7
0x10	XX	XX	XX	XX	XX	XX	XX

1436A121

Event 5

Table 6-16 shows the master to slave acknowledgment of Transmit Request indication.

Control, byte 0 = 0x05

Bit 4 = 0

Transmit Request has been toggled indicating that there is more data to be transmitted.

Bits 2 thru 0 = 5

Shows 5 bytes to transmit.

Table 6-16. Master to Slave indication for Data Transmission

Control	Tx Data 1	Tx Data 2	Tx Data 3	Tx Data 4	Tx Data 5	Tx Data 6	Tx Data 7
0x05	0x6F "o"	0x72 "r"	0x6C "l"	0x64 "d"	0x21 "!"	XX	XX

1436A122

Event 6

Table 6-17 shows that the slave has received 5 additional bytes of data: This indication is explained as follows:

Status, byte 0 = 0x00

Bit 4 = 0

Transmit Acknowledge is being set to the same value as Transmit Request to indicate that the module is ready to received more data.

Bits 2 thru 0 = 5

No data is being received at this time

Table 6-17. Slave to Master Output Data is "Qued" Response

Status	Rx Data 1	Rx Data 2	Rx Data 3	Rx Data 4	Rx Data 5	Rx Data 6	Rx Data 7
0x00	XX	XX	XX	XX	XX	XX	XX

1436A114

Event 7 and Event 8

Master to slave and Slave to Master Idle mode. (No more data to send.)

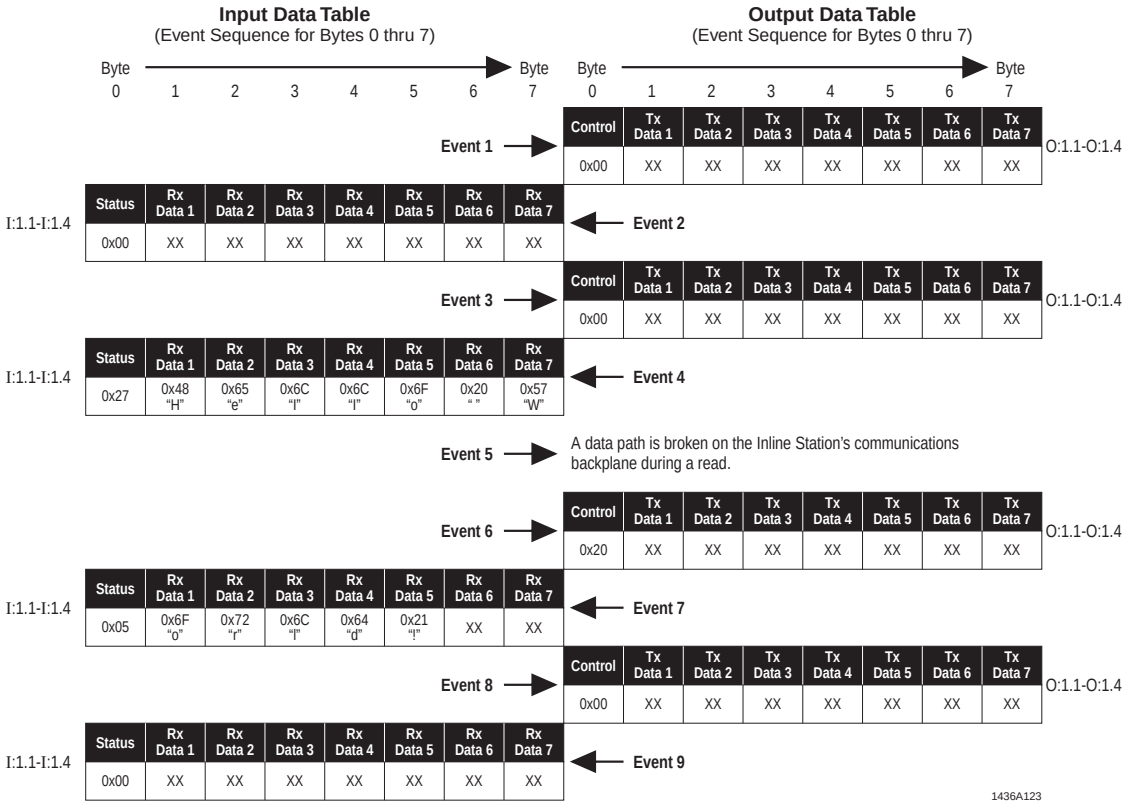
C Fragmented Error Handling, Loss of Inline Backplane Communications Example

This serial fragmentation error handling example, shown in Table 6-18, will show how the fragmentation will react during a break in the communications path on the Inline Station's backplane. The error-handling sequence is also applicable for a write sequence.

Events 1 thru 4

Refer to the Fragmented Read Example for an explanation of events 1 thru 4.

Table 6-18. Serial Framentation Error Example 1



Event 5

An error has occurred. The communications path between the bus coupler and its I/O has been broken. If incoming serial data overflows the buffer on the serial module, overflow data will be lost.

Events 6 thru 9

Error is repaired and the connection is reestablished thus data continues to be received.

D. Fragmented Error Handling, Loss of Communications from the Host to the Bus Coupler

If communications is lost between the host controller an the Inline bus coupler during a read or write service the bus coupler will wait for the error to be corrected (network cable is repaired) and then continue to finish the serial fragmentation transaction.

6.2.2 Method 2. Transfer of Serial Data Using Explicit Messages

Note

The ability to send explicit messages is a function of the DeviceNet™ I/O scanner. Not all scanners have an explicit messaging channel available to the user. This manual will not document the mechanics of the actual sending of an explicit message. Information of this type must be provided in the documentation for the I/O scanner.

Configuration software can be an option to understand the structure of an explicit message before the message is actually integrated into the control program.

Explicit messages can be sent as an alternative to using fragmentation and the high-speed data channel. For this method the Serial Communications Object, Class Code 106 dec, (0x6A hex) will need to be directly accessed using an explicit message.

The Serial Communications Object contains attributes used for the sending and receiving of data to or from a serial module. When using Method 2, the 8 bytes of fragmentation produced and consumed data, as well as the Status and Control words can be remove from the scan. For easy access, it may be advantageous to allow the Status and Control words to remain in the scan.

6.2.2.1 Receiving Serial Data

Attribute 7 of the Serial Communications Object (SCO) is the Receive Data parameter. Using this attribute along with the status word, bit 0 “Receive Buffer is not Empty” , is required to receive data from a serial module. The status word (and a control word) from the serial module(s) is automatically added to the DeviceNet™ scan by default. (See the respective serial modules data sheet for more information on specific control functions and status capabilities.) The user will need to monitor bit 0. When bit 0 is set, there is data present from the serial module. At this point the user will send an explicit message with the following parameters:

Node Address

Service Code = 14 decimal (Get Attribute Single)
Class = 106 decimal (Serial Communications Object)
Instance = 1 (In this case the 1st occurrence of a serial module)
Attribute = 7 (Receive data)

Note

Instance is determined by the physical location on the Inline station. The 1st instance will be assigned to the serial module (RS232 or RS485/RS422) located closest to the bus coupler and the last instance (Maximum of 8) will be assigned to the right most module.

When this message is sent, the bus coupler will send it’s response back to the sender. Typically a I/O scanner control bit is set when an explicit response is present. At that point the response can be read. The response will include a positive or negative confirmation.

Important

The first data byte returned by bus coupler will be the number of bytes to follow. These “following” bytes are the actually data that was received by the serial module. Keep in mind that the user will need to understand the explicit message response format dictated by the I/O scanner. It is probable that several bytes of data pertaining to the response “header” will be returned before the actual data returned from the serial module is present. This header may include such information as: transaction ID, command, nodes address and confirmation (positive or negative).

6.2.2.2 Transmitting Serial Data

Attribute 8 of the Serial Communications Object (SCO) is the Transmit Data parameter. This attribute is required to transfer data to a serial module. To transmit data to the serial module the user will send an explicit message with the following parameters:

Node Address

Service Code = 16 decimal (Set Attribute Single)
Class = 106 dec. (Serial Communications Object)
Instance = 1 (In this case the 1st occurrence of a serial module)
Attribute = 8 (Transmit data)

Note

Instance is determined by the physical location on the Inline station. The 1st instance will be assigned to the serial module (RS232 or RS485/RS422) located closest to the bus coupler and the last instance (Maximum of 8) will be assigned to the right most module.

When this message is sent, the bus coupler will send it’s response back to the sender. Typically a I/O scanner control bit is set when an explicit response is present. At that point the response can be read. When sending a transmit command the user can expect a positive or negative confirmation in return.

Important

The first byte transmitted to the bus coupler will be the number of bytes to follow. These “following” bytes are the actual data that is being sent to the serial module.

6.2.3 Method 3. Transfer of Data Using PCP Fragmentation

This method defines how PCP data is exchanged, using process/cyclic data, with an Inline module that supports peripheral communications protocol (PCP) and is not a serial module. An example of this type of module would be the Inline AS-i-Gateway (ASI MA IB IL).

Note

Typically the ASI MA IB IL will not require the use of PCP data exchange. However PCP data exchange will be required if there are more than 31 slaves on the AS-i “subnetwork”.

The messages that are required to read and write data are encoded into the high-speed data stream. The protocol is handled using a series of message fragments that are initiated by a client start request and then followed up with a server response. These fragments were specifically designed to be used with any Inline PCP modules.

These process data messages are used to read or write to a specific slave device's memory location that is access by an Index and subindex designation. Beside the exchange of normal I/O data PCP process data communications can be used to parameterize an Inline module or retrieve informative data.

Note

Information pertaining to the supported indexes, subindexes and Invoke ID can be found in the specific module's data sheet and/or manual.

For each PCP Inline module, by default, 8 bytes (1 fragment) are added to the DeviceNet™ produced size and eight bytes are added to the consumed size. These eight bytes can only be used to send PCP data messages and are in addition to any other I/O data that might also be added into the scan. This type of information can be found in the specific module's data sheet.

An example of this type of information would be a status and control word. These two bytes would also be added to the produced size and to the consumed size in addition to the eight bytes allocated for the "I/O messaging" connection. For this example there would be a total of 10 produced bytes and 10 consumed bytes allocated for this one Inline PCP module. When using PCP fragmentation a request will be sent and a response will be returned the format of a request and a response is as follows:

a. Request (Output Data) – NO Invoke ID

Byte 0
Service

Byte 1
Module-number

Byte 2
Index low

Byte 3
Index high

Byte 4
Subindex

Byte 5
Length

Byte 6 thru N
Data-block, if necessary

b. Request (Output Data) – With Invoke ID

Byte 0
Service

Byte 1
Invoke ID

Byte 2
Module-number

Byte 3
Index low

Byte 4
Index high

Byte 5
Subindex

Byte 6
Length

Byte 7 thru N
Data-block, if necessary

c. Successful Response: (Input Data) – NO Invoke ID

Byte 0
Service

Byte 1
Status

Byte 2
Length

Byte 3 thru N
Data-block, if necessary

If the response was not successful the response will be returned in the following format:

d. Errored Produced Response: (Input Data) – NO Invoke ID

Byte 0
Service

Byte 1
Status

Byte 2
Error class

Byte 3
Error code

Byte 4
Additional error code 1 LSB, if necessary

Byte 5
Additional error code 1 MSB, if necessary

Byte 6
Additional error code 2 LSB, if necessary

Byte 7
Additional Error Code 2 MSB, if necessary

e. Successful Response: (Input Data) – With Invoke ID

Byte 0
Service

Byte 1
Invoke ID

Byte 2
Status

Byte 3
Length

Byte 4 thru N
Data-block, if necessary

If the response was not successful the response will be returned in the following format:

d. Errored Produced Response: (Input Data) – With Invoke ID

Byte 0
Service

Byte 1
Invoke ID

Byte 2
Status

Byte 3
Error class

Byte 4
Error code

Byte 5
Additional error code 1 LSB, if necessary

Byte 6
Additional error code 1 MSB, if necessary

Byte 7
Additional error code 2 LSB, if necessary

Byte 8
Additional Error Code 2 MSB, if necessary

The Response Service byte is a reflection of the request service byte, with the exception of the request/response bit (see the definition of the Start Fragment for more information on this bit).

CAUTION

It is important to keep track of the "client-service" relationship between the master and the slave. For example, a read request involves a single non-fragmented service. The slave responds with a completely new service to transfer the response data. This new service begins with a start fragment and ends (if enough data was requested) with a last fragment. The write request on the other hand, starts with a start fragment and ends (if enough data was sent) with a last fragment. The slave responds with a non-fragmented start fragment.

6.2.3.1 Fragment Types

Four transfer-fragment types are distinguished by the service-byte (byte 0 of each fragment). The types are as follows:

A. Start fragment
B. Middle fragment
C. Last fragment
D. Abort/Error fragment

A. Start Fragment

To begin any message a start fragment must be sent. The start fragment's eight bytes have the following format:

Byte 0
Service

Byte 1
Module number

Byte 2
Index low

Byte 3
Index high

Byte 4
Subindex

Byte 5
Length

Byte 6
Data-block, if necessary

Byte 7
Data-block, if necessary

1. Byte 0, Service Definition of the Start Fragment (Shown in Table 6-19)

Bit 7

Request/Response. The request response bit is set when the actual Inline PCP module responds to the PCP service that the user requested. The amount of time that it takes to process this service depends on the PCP channel size, the number of Inline modules, and the actual service requested. The user can use this bit to know when the service is actually processed by the Inline PCP module. For a read request, this bit will be set by the bus coupler as soon as the read request is called. For a write request, this bit is set as soon as all data reaches the PCP module.

0 = Request (Service in process)
1 = Response (Service is processed)

Table 6-19. Service Byte 0, Definition of the Start Fragment

7	6	5	4	3	2	1	0
Request/ Response	0	0	No Frag- ment/ Fragment	PCP Service			

1436A085

Bit 6:5

Fragment Type. For a start fragment these bits will always be a zero.

00 = Start - fragment

Bit 4

Fragmented Bit 4 informs the slave as to whether the message contains more than eight bytes (7 data bytes) or not.

0 = Doesn't fragment
1 = Fragments

Bit 3:0

PCP Service. Bits 3 thru 0 informs the slave of the type of message (service) being sent. The means are described as follows:

0x00 = no action (Clears input bytes in response)
0x01 = Read
0x02 = Write
0x03 = Read PDU length
0x04-0x0F = Reserved

B. Middle Fragment

Any write request or read response requires more than the available number of data bytes in both a start fragment and a last fragment, then a middle fragment must be used. Middle fragments have the following format.

Byte 0

Service

Byte 1

Data-block, if necessary

Bytes 2 thru 7

Data-blocks, if necessary

1. Byte 0, Service Definition of a Middle Fragment (Shown in Table 6-20)

Bit 7

Request/Response. See start fragment for definition.

Bit 6:5

Fragment type - For a middle fragment these bits will always be a 01.
01= Middle fragment

Bit 4

0 Count - Bits 4 thru 0 keep track of how many middle fragments have been sent. 31 is the maximum number of middle fragments that can be counted (1–0x1F). If more fragments are needed, the fragment number will roll over to 0 and fragments can continue to be sent.

Table 6-20. Service Byte Definition of the Middle Fragment

7	6	5	4	3	2	1	0
Request/ Response	0	1	Fragment Number (0x01 - 0X1F)				

1436A086

C. Last Fragment

To recognize the end of a fragmented message, a last fragment must be issued. A last fragment has the following format:

Byte 0

Service

Bytes 1 thru 7

Data-block, if necessary

1. Byte 0, Service Definition of the Last Fragment (see Table 6-21)

Bit 7

Request/Response. See start fragment for definition.

Bit 6:5

For a last fragment, bits will always be 10.

10 = last fragment

Bit 4:0

Reserved

Table 6-21. Service Byte Definition of the Last Fragment

7	6	5	4	3	2	1	0
Request/ Response	1	0	Reserved				

1436A087

D. Abort/Error fragment:

If a transmission error is detected an abort/error fragment will be generated.

1. Byte 0, Service Definition of a PCP Abort/Error Fragment (see Table 6-22)

Bit 7

Request/Response. See start fragment for definition.

0 = Request
1 = Response

Table 6-22. Service Byte Definition of the PCP Abort / Error Fragment

7	6	5	4	3	2	1	0
Request/ Response	1	1	Reserved				

1436A088

Bit 6:5

Fragment Type. For an abort/error fragment, these bits will always be an 11.

11= Abort / Error fragment

Bit 4:0

Reserved

2. Byte 1, Status Definition of a Abort/Error Fragment (See Table 6-23)

Module COMM Error: When set, communication with the module is no longer possible.

PCP Error: A PCP Service-Specific error has occurred. See the Error Class and Error Code bytes.

PCP Channel Busy: a PCP transaction is already in progress, such as from an explicit request.

Fragmentation Error: An error has occurred with either the type or sequence of the fragments (i.e. middle received after last, middle fragment 2 received before 1, etc).

Table 6-23. Status Byte Definition of the PCP Abort/Error Fragment

7	6	5	4	3	2	1	0
Reserved	Reserved	Reserved	Reserved	Fragmen- tation Error	PCP Channel Busy	PCP Error	Module Comm Error

1436A089

3. Byte 2-7, Error Class and Error Code

Bytes 2 and 3 will display the error class and error code. If there is any addition error information it will be displayed in bytes 4-7. To interpret the error information, a PCP reference manual must be consulted.

6.2.3.2 PCP Fragmentation Examples

The following examples will show how to read, write and handle errors using PCP process data fragmentation. The examples to follow are:

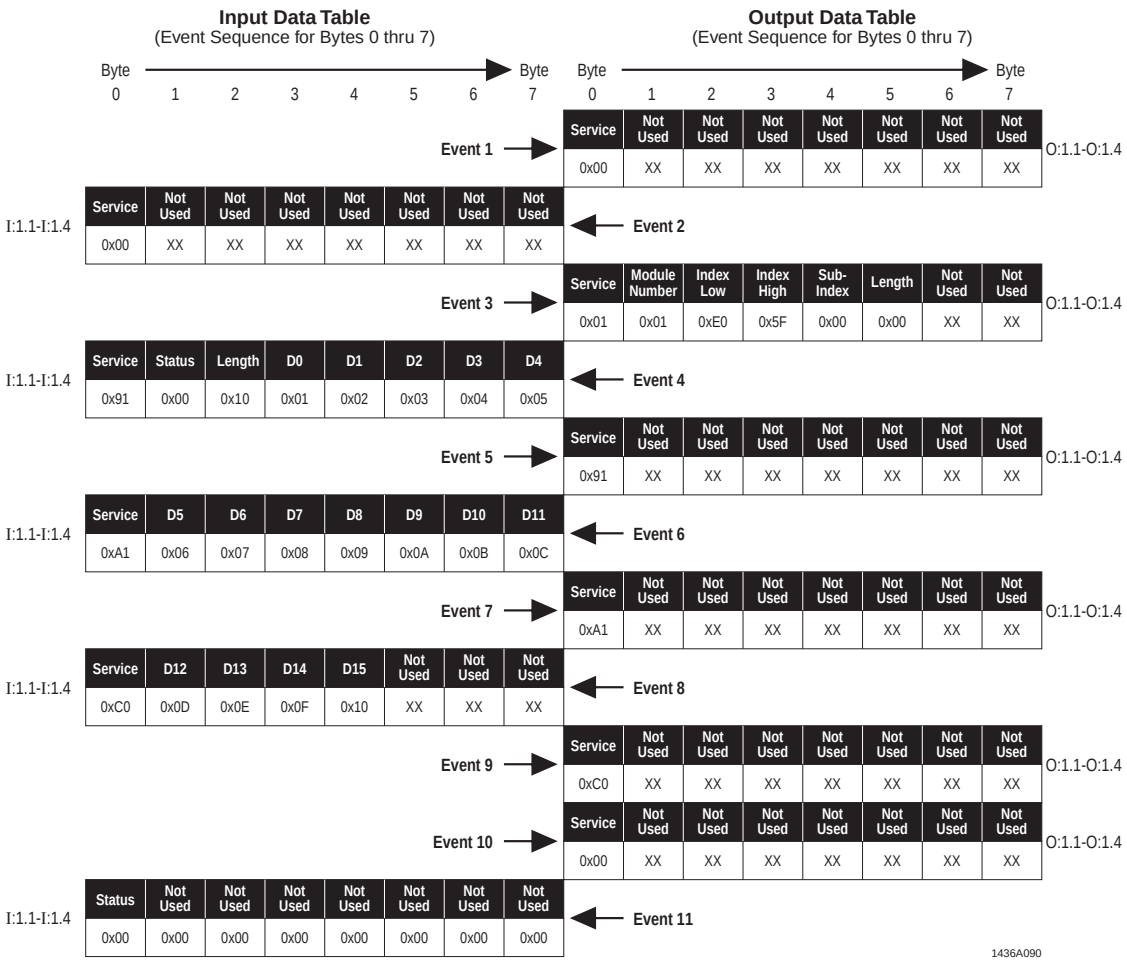
A. Fragmented Read
B. Fragmented Write
C. Error Handling, Communications Backplane Break
D. Error Handling, Host Communications Loss

A. Fragmented Read Example

Table 6-24 shows an example I/O data table that contains only eight bytes of input data and eight bytes of output data. This I/O is shown in a event by event sequence that demonstrates how these eight bytes of I/O are updated when reading data using PCP fragmentation. The sequence works on this following basic principle:

Event x. Client issues server 8 bytes of output data
Event x+1. Server responds by updating 8 bytes of input data

Table 6-24. I/O Events for a Read Sequence Using PCP Fragmentation



In the following paragraphs the events in Table 6.24 will be explained in detail.

Tables 6-25 through 6-35 demonstrate the order of events when issuing a fragmented read service to a PCP device. Each “Event“ should be referenced to the I/O data table shown in Table 6-24.

Event 1

Table 6-25 shows the transmission from a master to slave when the PCP fragmentation is in “Idle” mode. Note that in byte 0 the service 0x00 is being sent.

Table 6-25. Master to Slave Idle Request, Sending a 0x00 No Action Service

Service	Not Used	Not Used	Not Used	Not Used	Not Used	Not Used	Not Used
0x00	XX	XX	XX	XX	XX	XX	XX

1436A091

Event 2

Table 6-26 shows the slaves response to the idle state by replying with a 0x00 no action acknowledge.

Table 6-26. Slave to Master Idle Response, 0x00 No Action Acknowledge

Service	Not Used	Not Used	Not Used	Not Used	Not Used	Not Used	Not Used
0x00	XX	XX	XX	XX	XX	XX	XX

1436A091

Event 3

Table 6-27 shows how the start fragment request is sent from the master to the slave to initiate a read. This message is built with the format shown below. Note that the target index is 0x5FE0.

Service byte 0	0x01	Read
Module Number byte 1	0x01	First PCP module on the Inline station
Index Low, byte 2	0xE0	Low byte of the PCP index to be read
Index high, byte 3	0x5F	High Byte of the PCP index to be read
Subindex, byte 4	0x00	subindex is zero
Length, byte 5	0x00	No length requirement
Bytes 6 and 7	XX	Not used

Table 6-27. Master to Slave Read Request, Sending a 0x01 Service

Service	Module Number	Index Low	Index High	Sub-Index	Length	Not Used	Not Used
0x01	0x01	0xE0	0x5F	0x00	0x00	XX	XX

1436A092

Event 4

Table 6-28 shows the response from the slave that includes the first 5 bytes of actual data that was requested from index 0x5FE0. This reply is explained as follows:

Service byte 0 = 0x91

Bit 7 = 1

This signifies that the fragment is a response

Bit 6 = 0

This signifies a start fragment

Bit 5 = 0

This signifies a start fragment

Bit 4 = 1

This denotes that the response will be sent in fragments

Bits 3 thru 0 = 1

This signifies a read service

Status 0x00

No Errors

Length 0x10

Informs the master that there will be 16 data bytes in the message

Note

It's important to realize that event 4 is really the start fragment of the slave's message containing the requested data.

D0 - D4

First 5 bytes of the message (bytes D5-D15 to be sent in the following Fragments)

Table 6-28. Slave to Master, Response to the Read Service

Service	Status	Length	D0	D1	D2	D3	D4
0x91	0x00	0x10	0x01	0x02	0x03	0x04	0x05

1436A093

Event 5

Table 6-29 shows the master to slave acknowledgment of the response being received. In this acknowledge the service byte reflection is the indication that the 1st response was received.

Table 6-29. Master to Slave Acknowledge

Service	Not Used	Not Used	Not Used	Not Used	Not Used	Not Used	Not Used
0x91	XX	XX	XX	XX	XX	XX	XX

1436A094

Event 6

Since the reception of the first 5 bytes of data has been acknowledged the slave is ready to send the next fragment. This second fragment is a middle fragment with a slightly different format. Table 6-30 shows the response from the slave that includes the service byte and 7 more bytes of data that was requested from index 0x5FE0. This reply is explained as follows:

Service Byte 0 = 0xA1

Bit 7 = 1

This signifies that the fragment is a response

Bit 6 = 0

Designates a middle fragment when bit 6 = 0 and bit 5 = 1

Bit 5 = 1

Designates a middle fragment when bit 6 = 0 and bit 5 = 1

Bit 4 thru 0 = 1

These bit count the fragments. 1 = the 1st middle fragment

Bytes 1 thru 7 = 7

Specifies 7 additional bytes of data (5 received in the first response)

Table 6-30. Slave to Master, Reply with First Middle Fragment

Service	D5	D6	D7	D8	D9	D10	D11
0xA1	0x06	0x07	0x08	0x09	0x0A	0x0B	0x0C

1436A095

Event 7

Table 6-31 shows the master to slave acknowledgment of the response being received. In this acknowledge the service byte reflection is the indication that the 1st middle fragment response was received.

Table 6-31. Master to Slave Acknowledge of the First Middle Fragment

Service	Not Used	Not Used	Not Used	Not Used	Not Used	Not Used	Not Used
0xA1	XX	XX	XX	XX	XX	XX	XX

1436A096

Event 8

Table 6-32 shows the slave to master reception of the last fragment that was requested from index 0x5FE0. This fragment returns the last 4 data bytes as expected by the length byte shown in Table 6-11. This response is explained as follows:

Service byte 0 = 0xC0

Bit 7 = 1

This signifies that the fragment is a response

Bit 6 = 1

Designates a last fragment when bit 6 = 1 and bit 5 = 0

Bit 5 = 0

Designates a last fragment when bit 6 = 1 and bit 5 = 0

Bits 4 thru 0

Reserved

Table 6-32. Slave to Master Last Fragment Response

Service	D12	D13	D14	D15	Not Used	Not Used	Not Used
0xC0	0x0D	0x0E	0x0F	0x10	XX	XX	XX

1436A097

Event 9

Table 6-33 shows the master to slave acknowledgment of the last fragment being received. In this acknowledge the service byte reflection is the indication that the last fragment response was received.

Table 6-33. Master to Slave Last Fragment Acknowledge

Service	Not Used	Not Used	Not Used	Not Used	Not Used	Not Used	Not Used
0xC0	XX	XX	XX	XX	XX	XX	XX

1436A098

Events 10 and 11

Knowing that the last fragment was received the master issues an idle service to the slave as shown in Table 6-34 and the slave respond with it's reply as shown in Table 6-35.

Table 6-34. Master to Slave Idle Service

Service	Not Used	Not Used	Not Used	Not Used	Not Used	Not Used	Not Used
0x00	XX	XX	XX	XX	XX	XX	XX

1436A099

Table 6-35. Slave to Master Idle Service Response

Status	Not Used	Not Used	Not Used	Not Used	Not Used	Not Used	Not Used
0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00

1436A100

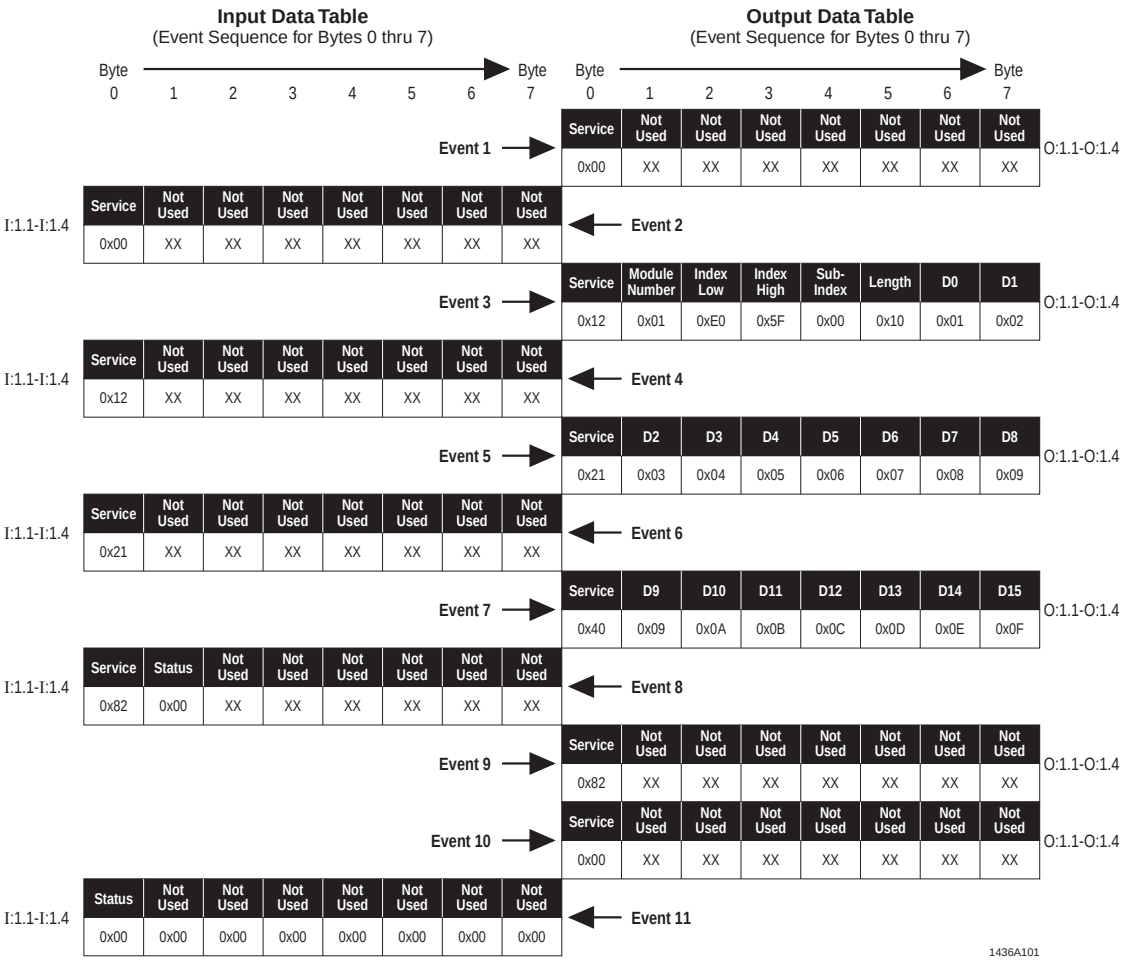
B. Fragmented Write Example

Table 6-36 shows an example I/O data table that contains only eight bytes of input data and eight bytes of output data. This I/O is shown in a event by event sequence that demonstrates how these eight bytes of I/O are updated when reading data using PCP fragmentation. The sequence works on this following basic principle:

Event x. Client issues server 8 bytes of output data

Event x+1. Server responds by updating 8 bytes of input data

Table 6-36. I/O Events for a Write Sequence Using PCP Fragmentation



In the following paragraphs the events in Table 6-36 will be explained in detail.

Tables 6-37 through 6-47 demonstrate the order of events when issuing a fragmented write service to a PCP device. Each “Event“ should be referenced to the I/O data table shown in Table 6-36.

Event 1

Table 6-37 shows the transmission from a master to slave when the PCP fragmentation is in “Idle” mode. Note that in byte 0 the service 0x00 is being sent.

Table 6-37. Master to Slave Idle Request, Sending a 0x00 No Action Service

Service	Not Used	Not Used	Not Used	Not Used	Not Used	Not Used	Not Used
0x00	XX	XX	XX	XX	XX	XX	XX

1436A091

Event 2

Table 6-38 shows the slaves response to the idle state by replying with a 0x00 no action acknowledge.

Table 6-38. Slave to Master Idle Response, 0x00 No Action Acknowledge

Service	Not Used	Not Used	Not Used	Not Used	Not Used	Not Used	Not Used
0x00	XX	XX	XX	XX	XX	XX	XX

1436A091

Event 3

Table 6-39 shows how the start fragment request is sent from the master to the slave to initiate a write. This message is built with the format shown below. Note that the target index is 0x5FE0.

Service byte 0 (0x12) Fragmented Write

Bit 4
Indicates that the write is fragmented

Bits 0 thru 3
Write service (0x02)

Module Number byte 1 (0x01)
First PCP module on the Inline station

Index Low, byte 2 (0xE0)
Low byte of the PCP index to be read

Index High, byte 3 (0x5F)
High byte of the PCP index to be read

Subindex, byte 4 (0x00)
subindex is zero

Length, byte 5 (0x00)

16 bytes

Bytes 6 and 7 XX

First 2 data bytes

Table 6-39. Master to Slave Write Request, Sending a 0x02 Service

Service	Module Number	Index Low	Index High	Sub-Index	Length	D0	D1
0x12	0x01	0xE0	0x5F	0x00	0x10	0x01	0x02

1436A102

Event 4

Table 6-40 shows the acknowledge from the slave that indicates the write request fragment was processed. In this acknowledge the service byte reflection is the indication that the 1st response was received.

Table 6-40. Slave to Master, Acknowledge of the Write Service

Service	Not Used	Not Used	Not Used	Not Used	Not Used	Not Used	Not Used
0x12	XX	XX	XX	XX	XX	XX	XX

1436A103

Event 5

Since the reception of the first 2 bytes of data has been processed and acknowledged the master is ready to send the next fragment. This second fragment is a middle fragment with a slightly different format. Table 6-41 shows the service byte and the next 7 bytes of data to be written to index 0x5FE0 . This request is explained as follows:

Service Byte 0 (0x21)

Bit 7 = 0

This signifies that the fragment is a request

Bit 6 = 0

Designates a middle fragment when bit 6 = 0 and bit 5 = 1

Bit 5 = 1

Designates a middle fragment when bit 6 = 0 and bit 5 = 1

Bits 4 thru 0 = 1

These bit count the fragments. 1 = the 1st middle fragment

Bytes 1 thru 7 = 7

More bytes of data (2 sent with first request)

Table 6-41. Master to Slave, Sending the 1st Middle Fragment

Service	D2	D3	D4	D5	D6	D7	D8
0x21	0x03	0x04	0x05	0x06	0x07	0x08	0x09

1436A104

Event 6

Table 6-42 shows the slave to master acknowledgment. The service byte reflection indicates that the 1st middle fragment data was received and processed.

Table 6-42. Slave to Master, Acknowledgement of 1st Middle Fragment

Service	Not Used	Not Used	Not Used	Not Used	Not Used	Not Used	Not Used
0x21	XX	XX	XX	XX	XX	XX	XX

1436A105

Event 7

Since the reception of the first middle fragment of data has been acknowledged the master is ready to send the next fragment. This next fragment is the last fragment. Table 6-43 shows the last fragment that includes the service byte and 7 more bytes of data that is being sent to index 0x5FE0 (16 bytes total). This last fragment request is explained as follows:

Service Byte 0 = 0x40

Bit 7 = 0

This signifies that the fragment is a request

Bit 6 = 1

Designates a last fragment when bit 6 = 1 and bit 5 = 0

Bit 5 = 0

Designates a last fragment when bit 6 = 1 and bit 5 = 0

Bits 4 thru 0 = 0

Reserved

Bytes 1 thru 7 = Last 7 bytes of data

Table 6-43. Master to Slave Last Fragment

Service	D9	D10	D11	D12	D13	D14	D15
0x40	0x09	0x0A	0x0B	0x0C	0x0D	0x0E	0x0F

1436A106

Event 8

Table 6-44 shows the slave to master acknowledgment of the last fragment being received and processed. In this acknowledge the service byte reflection is the indication that the last fragment data was received. It is important to realize that event 8 is really a start fragment from the slave signalling the response & status information for the write request.

Table 6-44. Slave to Master Last Fragment Acknowledge

Service	Not Used	Not Used	Not Used	Not Used	Not Used	Not Used	Not Used
0x82	0x00	XX	XX	XX	XX	XX	XX

1436A107

Event 9

Table 6-45 shows the master to slave acknowledgment of the last fragment being received. In this acknowledge the service byte reflection is the indication that the last fragment response was received.

Table 6-45. Master to Slave Acknowledge

Service	Not Used	Not Used	Not Used	Not Used	Not Used	Not Used	Not Used
0x82	XX	XX	XX	XX	XX	XX	XX

1436A108

Events 10 and 11

Knowing that the last fragment was received the master can issue an idle service to the slave as shown in Table 6-46 and the slave respond with it's reply as shown in Table 6-47.

Table 6-46. Master to Slave Idle

Service	Not Used	Not Used	Not Used	Not Used	Not Used	Not Used	Not Used
0x00	XX	XX	XX	XX	XX	XX	XX

1436A099

Table 6-47. Slave to Master Response to Idle

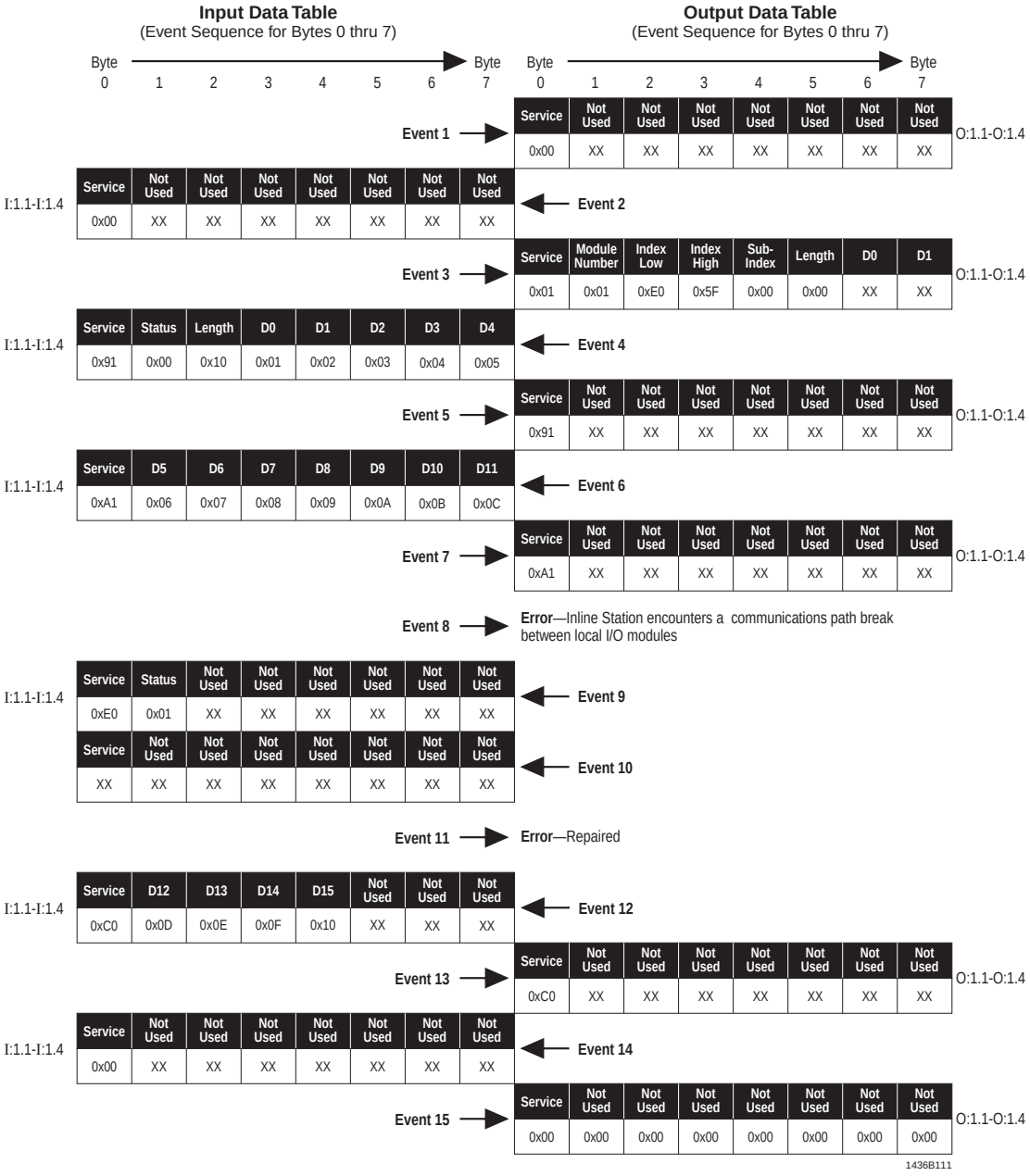
Status	Not Used	Not Used	Not Used	Not Used	Not Used	Not Used	Not Used
0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00

1436A100

C. Fragmented Error Handling, Loss of Inline Backplane Communications

This PCP fragmentation error handling example, shown in Table 6-48, will show how the fragmentation will react during a break in the communications path on the Inline Station's backplane.

Table 6-48. Local Communications Error Sequence Using PCP Fragmentation



Events 1 thru 7

For an explanation of events 1 thru 7, look at the examples in this section for a read sequence.

Event 8

An error has occurred. The communications path between the bus coupler and it's I/O has been broken.

Event 9

A PCP Abort/Error fragment has been issued from the slave to the master as shown in Table 6-49.

Service Byte 0 = 0xE0

Bit 7 = 1
This signifies that the fragment is a response

Bit 6 = 1
Designates an abort/error fragment when bit 6 = 1and bit 5 = 1

Bit 5 = 1
Designates an abort/error fragment when bit 6 = 1 and bit 5 = 1

Bits 4 thru 0 = 0
Reserved

Byte 1 = 1
Identifies a module communication Error

Bytes 2 thru 7
Not used

Table 6-49. Abort/Error Fragment

Service	Status	Not Used	Not Used	Not Used	Not Used	Not Used	Not Used
0xE0	0x01	XX	XX	XX	XX	XX	XX

1436A109

Event 10

Data is no longer being received in the input table.

Note

Once the connection is reestablished the data will continue to be sent.

Event 11

In this example the communications error is corrected at this point.

Events 12-15

Read service is completed as described previously in the section.

D. Fragmented Error Handling, Loss of Communications from the Host to the Bus Coupler

If communications is lost between the host controller an the Inline bus coupler during a read or write service the bus coupler will wait for the service to be completed (network cable is repaired) and acknowledge the completion of the service once the transaction has ended.

6.2.4 Method 4. Transfer PCP Data Using Explicit Messages

Explicit messages can be sent as an alternative to using the high-speed data channel and PCP fragmentation to send PCP messages. For method 4 the PCP Special Function object, Class Code 105 dec(0x69 hex) will need to be directly accessed using an explicit message. By default this method does not apply to serial modules. Refer to Communications Methods in this section.

Notes

The ability to send explicit messages is a function of the DeviceNet™ I/O scanner. Not all scanners have an explicit messaging channel available to the user. This manual will not document the mechanics of the actual sending of an explicit message. Information of this type must be provided in the documentation for the I/O scanner.

Configuration software can be an option to understand the structure of an explicit message before the message is actually integrated into the control program.

The PCP Special Function Object, Class Code 105 dec. (0x69 hex) is detailed further in Appendix A

When sending PCP messages using explicit messages I/O scan size can be reduced by removing the eight bytes of fragmentation data that is added to the produced and consumed sizes by default. This can be accomplished by using the PCP Special Function Object.

6.2.4.1 Reading PCP Data

When reading PCP data from a PCP module that is not a serial based module, the PCP Special Function Object, Class Code 105 dec. (0x69 hex) must be used. Within this object, there are two sub-methods that can be used to read data explicitly. Sub-method A involves directly requesting/reading the complete PCP services using attributes 6 and 7. Sub-method B provides greater efficiency. It fixes the module number, index, and subindex so that only the data is received for each request (attributes 8, 12, 13, and 14). The following procedure uses sub-method B.

Three explicit messages will be required to read a specific PCP memory area. They are as follows:

1. Select Index
2. Select subindex
3. Read data

These messages will have the format shown below and will need to be sent with the required service.

1. Build Select Index Message: Message is sent using the Set Attribute Single service (16 dec.)

Class Code	105 decimal (0x69 hex), PCP Special Function Object
Instance	Select the occurrence of the PCP device within the object
Attribute	12 dec., PCP Read Index (Example: Index number 5FE0 hex)

2. Build Select Subindex Message: Message is sent using the Set Attribute Single service (16 dec.)

Class Code	105 decimal (0x69 hex), PCP Special Function Object
Instance	Select the occurrence of the PCP device within the object
Attribute	13 dec., PCP Read Subindex (Example: Subindex number 0)

3. Build Read Data Message: Message is sent using the Get Attribute Single service (14 dec.)

Class Code	105 decimal (0x69 hex), PCP Special Function Object
Instance	Select the occurrence of the PCP device within the object
Attribute	14 dec., PCPReadData

Note

Instance is determined by the physical location on the Inline station. The 1st instance will be assigned to the PCP module located closest to the bus coupler and the last instance (Maximum of 8) will be assigned to the right most PCP module. Serial modules will occupy an instance in this object.

The PCP module attribute (attribute 8) defaults to the instance value.

When message 3 is sent, the bus coupler will send the data back to the sender.

CAUTION

The first data byte returned by bus coupler will be the number of bytes to follow. These “following” bytes are the actual data that was sent by the PCP module.

6.2.4.2 Sending PCP Data

When sending PCP data from a PCP module that is not a serial based module, the PCP Special Function Object, Class Code 105 dec. (0x69 hex) must be used. Within this object, there are two sub-methods that can be used to write data explicitly. Sub-method A involves directly requesting/writing the complete PCP services using attributes 6 and 7. Sub-method B provides greater efficiency. It fixes the module number, index, and subindex so that only the data is sent for each request (attributes 8, 9, 10, and 11). The following procedure uses sub-method B.

Three explicit messages will be required to write a specific PCP memory area. They are:

1. Select Index
2. Select subindex
3. Write data

These messages will have the format shown below and will need to be sent with the required service.

1. Build Select Index Message: Message is sent using the Set Attribute Single service (16 dec.)

Class Code	105 decimal (0x69 hex), PCP Special Function Object
Instance	Select the occurrence of the PCP device within the object
Attribute	9, PCP Write Index (Example: Index number 5FE0 hex)

2. Build Select Subindex Message: Message is sent using the Set Attribute Single service (16 dec.)

Class Code	105 decimal (0x69 hex), PCP Special Function Object
Instance	Select the occurrence of the PCP device within the object
Attribute	10 dec., PCP Write Subindex (Example: Subindex number 0)

3. Build Read Data Message: Message is sent using the Set Attribute Single service (16 dec.)

Class Code	105 decimal (0x69 hex), PCP Special Function Object
Instance	Select the occurrence of the PCP device within the object
Attribute	11 dec., PCP Write Data
Data	First byte contains the number of bytes to be sent, then the actual data

Note

Instance is determined by the physical location on the Inline station. The 1st instance will be assigned to the PCP module located closest to the bus coupler and the last instance (Maximum of 8) will be assigned to the right most PCP module. Serial modules will occupy an instance in this object.

The PCPmodule attribute (attribute 8) defaults to the instance value.

When message 3 is sent, the bus coupler will receive the data from the sender.

CAUTION

The first data byte returned by bus coupler will be the number of bytes to follow. These “following” bytes are the actual data that was sent by the PCP module.

6.3 Serial and Generic PCP Modules Produced and Consumed Sizes

Note

Section 3 provides additional information in regards to how data is mapped into the scanner and other considerations

6.3.1 Determining Produced and Consumed Size

Note

The bus coupler can auto-configure itself to the Inline I/O connected to it (Refer Section 3). Once this is done the total number of produced and consumed data, for the entire Inline station (will include fragmentation data), can be read from the EDS file.

By default any module that uses the PCP protocol (includes the serial modules) will have 8 bytes of produced data and 8 bytes of consumed data added to the network scan. These bytes are used to transfer data back and fourth between a DeviceNet™ scanner and, for example, an Inline serial module. This data transfer using 8 bytes is required when using process/cyclic data to Tx/Rx serial data (fragmentation).

In addition to the bytes used for transferring serial or other PCP data there may be additional data produced or consumed by the I/O module. This additional data must be added to the 8 bytes described in the previous paragraph. The number of additional process data bytes can be found in the specific Inline module's data sheet or manual. These process data bytes are will be added to the scan ahead of the 8 bytes of fragmentation data in the scanner’s I/O memory. Figure 6-1 gives and example of calculating the total number of bytes produced and consumed by a single Inline RS232 module.

	Produced	Comsumed
Bytes required by the RS232 module for status and control (found in data sheet)	2	2
+	+	+
Bytes required to Tx/Rx serial data (fragmentation)	8	8
Total used by each RS232 module	10 bytes	10 bytes

1436A084

Figure 6-1. Calculation of a Serial Module’s Produced and Consumed Bytes

6.3.2 Removing or Adding Fragmentation Data

Note

Fragmentation data must not be removed if using methods 1 or 3 described in this section under "Communications Methods".

If serial or other PCP data is going to be transmitted using explicit messages, then the 8 bytes used for the process/cyclic data messaging (fragmentation) that is added to the scan by default, will not be needed. The unused 8 bytes of produced and 8 bytes of consumed should be removed to ensure the best possible network performance.

If the user has a serial module and wants to remove or add the fragmentation data (8 bytes) they must send the following explicit messages. This must be sent to the Serial Communications Object, Class Code 106 dec. (0x6A hex), using the proper instance and attribute 32, “Fragment Data in DNET I/O”.

To add or remove the Serial Data (8 bytes of fragment data) from the Dnet I/O, set the following:

Class 106

Instance X

Attribute 32 (Serial Communications Object Fragment Data)
0 = removes data
1 = adds data

If the user has any other PCP module and wants to add/remove the fragmentation data (8 bytes) they must send an explicit message. This must be sent to the PCP Special Function Object Class Code 105 dec. (0x69 hex), using the proper instance and attribute 17, “PCP Fragment Data in Dnet I/O”.

To add or remove the Other PCP Data from the Dnet I/O, set the following:

Class 105

Instance X

Attribute 17 (PCP Fragment Data)
0 = removes data
1 = adds data

6.4 I/O Memory Mapping, Serial and Special Function PCP Modules

Note

Section 3 provides additional information in regards to how data is mapped into the scanner and other considerations.

6.4.1 I/O Mapping Rules

Note

Section 3 describes configuration methods (mapping) in greater detail.

The I/O image in the bus coupler flash memory contains all produced-data (input data) and consumed-data (output data) derived from the I/O modules connected to it. I/O image data is added to the poll through the use of Parameter 9 (Add All I/O), or by using auto configuration.

An I/O image could contain the Inline Status word (included by default in the produced data), command byte (not included in the consumed data by default), module fault data, reserved I/O space, digital, analog, special function (no PCP), special function PCP modules or serial modules (process data first then fragmentation data).

Mapping priority is determined by the type of module without regard to its location to the BK or other modules of different types. However, it does take into account the order of modules of the same type that exist on the station.

6.5 Configuration Brief for the RS232 and RS485/RS422 Modules

6.5.1 General Configuration

Note

Appendix A provides details of the Serial Communications Object (Class Code 106 dec, 0x6A hex) for configuration attributes.

This section describes how to change default settings for the Inline RS232 and RS485/ RS422 modules. Information provided in this section must be used in conjunction with the specific module’s data sheet.

In order to change any setting, Refer to the module's specific data sheet to determine the appropriate attribute settings for the Serial Communications Object (Class Code 106 dec, 0x6A hex).

The default settings can only be changed by sending an explicit message. An explicit message can either be sent from a control program or a DeviceNet™ configuration software package. Once the desired parameters have been updated the settings are stored in flash memory of the bus coupler. If the bus coupler is replaced, the serial configuration will need to be sent again, unless the configuration explicit messages are embedded into the control program.

The explicit message format required to configure the RS232 or RS485/RS422 modules is as follows:

Service	Set Attribute Single, 16 dec. (0x10 hex)
Class Code	106 dec. (0x6A hex) Serial Communications object
Instance	1 (1st serial module)
Attribute	12, This attribute is used to modify the baud rate
Data	08, (Refer to the RS232 module's data sheet) Code "08" represents a baud rate of 19.2K

Note

Instance is the occurrence of the module within the Serial Communications object. Instances are assigned by the physical order of the serial Inline modules on the station starting with the module closest to the bus coupler being assigned to instance 1. The next module to follow will be assigned to instance 2 and so on up to a maximum of 8 instances. (There is a maximum of 8 PCP modules of any kind allowed to reside on the Inline station.)

Both the RS232 and RS485/RS422 modules occupy instances in the Serial Communications object. If one of each reside on the station the closest to the bus coupler will be assigned to instance 1and the other will be assigned to instance 2.

APPENDIX A
DeviceNet™ Object Classes, Message Types and Services

Appendix A Contents

Appendix A Contents (continued)

A.1 General

The Inline bus coupler supports DeviceNet™ using ODVA standard Discrete Input Points (DIP), Discrete Output Points (DOP), Analog Input Points (AIP) and Analog Output Points (AOP). Additional objects include user defined Configuration, Inline Interface, Inline Module, Inline Special Function, PCP Special Function and Serial Communication objects.

A.2 DeviceNet™ Class Services

The bus coupler supports the following class services and instance services

Service Code	Service Name
05 (0x05)	Reset
14 (0x0E)	Get_Attribute_Single
16 (0x10)	Set_Attribute_Single

1110B160

A.3 DeviceNet™ Object Classes

The bus coupler supports the following object classes.

Class Code	Object Type
01 (0x01)	Identity
02 (0x02)	Router
03 (0x03)	DeviceNet
05 (0x05)	Connection
08 (0x08)	Digital Input Point
09 (0x09)	Digital Output Point
10 (0x0A)	Analog Input Point
11 (0x0B)	Analog Output Point
43 (0x2D)	Acknowledge Handler
100 (0x64)	Configuration Object
101 (0x65)	Inline Interface Object
102 (0x66)	Inline Module Object
103 (0x67)	Inline Special Function Object
104 (0x68)	COS Mask Object
105 (0x69)	PCP Special Function Object
106 (0x70)	Serial Communications Object

1436B054

A.4 Identity Object (Class Code 01 dec, 0x01 hex)

The Identity Object is required on all devices and provides identification of and general information about the device.

A.4.1 Identity Object, Instance Attributes

Attribute	Access	Name	Type	Value
1	Get	Vendor	UINT	562
2	Get	Product Type	UINT	0 = Generic Device
3	Get	Product Code	UINT	8162
4	Get	Revision Major Revision Minor Revision	STRUCT OF Byte Byte	 Major revision value 1 Minor revision value 1
5	Get	Device Status	UINT	
6	Get	Serial Number	Double	
7	Get	Product Name Length Name	STRUCT OF USINT STRING [6]	 19 DeviceNet™ Bus Coupler
8	Get	State	USINT	
10	Get/Set	Heartbeat Interval	USINT	Heartbeat interval in seconds

1436A003

A.4.2 Identity Object Common Services

Service Code	Class	Instance	Service Name
05 (0x05)	No	Yes	Reset
14 (0x0E)	Yes	Yes	Get_Attribute_Single
16 (0x10)	No	Yes	Set_Attribute_Single

1110A164

A.4.3 Identity Object Attributes

Product Code – Attribute 3

The Product Code is fixed at 8162 for the IL DN BK3. The product code is used within the Electronic Data Sheet format to uniquely identify the product type.

Revision Information – Attribute 4

The major revision number will increment as functional enhancements are implemented. The minor firmware revision control number is incremented if minor changes are incorporated.

Device Status - Attribute 5

Bit Number	Name	Meaning
0	Owned	Not defined
1	Reserved	
2	Configured	= 0, not configured — this bit is not supported
3	Reserved	
4 - 7	User defined	
8	Minor Recoverable fault	= 0, no fault = 1, minor recoverable faults (DOP short circuit)
9	Minor Unrecoverable fault	= 0, no fault = 1, minor unrecoverable faults
10	Major Recoverable fault	= 0, no fault = 1, major recoverable faults (Loss of +24 V dc)
11	Major Unrecoverable fault	= 0, no fault = 1, major unrecoverable faults (Checksum A/D)
12 - 15	Reserved	

1110A165

Serial Number - Attribute 6

The Serial Number is encoded in the product during the manufacturing cycle and is guaranteed to be unique.

Device Name - Attribute 7

The Device Name attribute provides a character array containing the short string "DeviceNet Inline BK.

Device State - Attribute 8

The Device State reflects whether any errors have occurred and the severity of the errors. The following table lists those states that are supported. The only exit from a Major Unrecoverable fault condition is power cycling the device.

State	Interpretation	Causes
0	Non-existent	
1	Self Test	
2	Standby	
3	Operating	Normal Operating Mode
4	Major Recoverable Fault	Loss of 24 V dc power
5	Major Unrecoverable Fault	Memory Checksum failure

1110A166

A.5 Router Object (Class Code 02 dec, 0x02 hex)

The Message Router Object provides a messaging connection point through which a Client may address a service to any object class or instance residing in the physical device.

A.5.1 Router Object Common Services

Service Code	Class	Instance	Service Name
14 (0x0E)	Yes	No	Get_Attribute_Single

1110A169

A.6 DeviceNet™ Object (Class Code 03 dec, 0x03 hex)

The DeviceNet™ Object defines how the node interfaces to the DeviceNet™ system.

A.6.1 DeviceNet™ Object, Instance 1 Attributes

Attribute	Access	Name	Type	Description
1	Get/Set	MACID	USINT	The MAC ID is set using DIP switches No. 1 (MSB) and No. 7 (LSB). DIP switches are located on the side of the bus coupler. Valid MAC ID addresses are 0 to 63 (0 to 3F Hex). Setting the switches address to value greater than 63 will disable the switches, which allows software setting of the MAC ID. The software setting defaults to the last hardware setting. The switch is only read during power up. **DIP Switch Settings** MAC ID: SW1 SW2 SW3 SW4 SW5 SW6 SW7 — Data Rate: SW8 SW9 — Auto Config: SW10 ON: 64 32 16 8 4 2 1 OFF: 0 0 0 0 0 0 0
2	Get/Set	Baud Rate	USINT	The Data Rate is set using DIP switches 8 and 9 as shown below. The software setting defaults to the last hardware setting. **SW8 SW9** ON ON = software selectable baud rate ON OFF = 500 kbits/s OFF ON = 250 kbits/s OFF OFF = 125 kbits/s
3	Get/Set	Bus Off Interrupt	BOOL	The Bus Off Interrupt (BOI) determines the action if a Bus Off state is encountered. BOI = Action 0 = Hold chip in Off state (default) 1 = If possible, reset CAN chip
4	Get/Set	Bus Off Counter	USINT	The Bus Off Counter will be forced to 0 (zero) whenever set, regardless of the data value provided.
5	Get/Set	Allocation Information Choice Byte Master Node Addr.	STRUCT of BYTE USINT	Allocation_byte bit 0 = explicit set to 1 to allocation bit 1 = polled set to 1 to allocation bit 2 - 7 = reserved (always 0)
6*	Get	MAC ID Switch Change	BOOL	0 = No change 1 = MAC ID has changed
7*	Get	Baud Rate Switch Change	BOOL	0 = No change 1 = Baud Rate switch has changed
8*	Get	Current Address Switch Setting	USINT	Displays current value of Address switches
9*	Get	Current Baud Rate Switch Setting	USINT	Displays current value of Baud Rate switch

1436A004

* Attribute is not supported if switches are set in the software-settable mode (Address switches set to >63 and Baud Rate switch set to >2)

A.6.2 DeviceNet™ Object Common Services

Service Code	Class	Instance	Service Name
14 (0x0E)	Yes	Yes	Get_Attribute_Single
16 (0x10)	No	Yes	Get_Attribute_Single
75 (0x4B)	No	Yes	Allocate Master/Slave
76 (0x4C)	No	Yes	Release Master/Slave

1110A174

A.7 Connection Object (Class Code 05 dec, 0x05 hex)

The Connection Objects manage the characteristics of each communication connection. As a Group II Only Slave device, the unit supports one explicit message connection and a POLL message connection.

A.7.1 Connection Object, Instance 1 Attributes (Explicit Message)

Attribute	Access	Name	Type	Value
1	Get	State	USINT	
2	Get	Instance Type	USINT	0 = Explicit Message
3	Get	Transport Class Trigger	USINT	0x83
4	Get	Production Connection	UINT	
5	Get	Consumed Connection	UINT	
6	Get	Initial Comm. Char.	USINT	0x21
7	Get	Production Size	UINT	244
8	Get	Consumed Size	UINT	244
9	Get/Set	Expected Packet Rate	UINT	0
12	Get/Set	Timeout Action	USINT	
13	Get	Prod. Path Length	USINT	0
14	Get	Production Path		(Null)
15	Get	Cons. Path Length	USINT	0
16	Get	Consumed Path		(Null)

1436A055

A.7.2 Connection Object, Instance 2 Attributes (POLL Connection)

Attribute	Access	Name	Type	Value
1	Get	State	USINT	
2	Get	Instance Type	USINT	1 = I/O Message
3	Get	Transport Class Trigger	USINT	0x83
4	Get	Production Connection	UINT	
5	Get	Consumed Connection	UINT	
6	Get	Initial Comm. Char.	USINT	0x1
7	Get	Production Size	UINT	See Class Code: 100 (0x64)
8	Get	Consumed Size	UINT	See Class Code: 100 (0x64)
9	Get/Set	Expected Packet Rate	UINT	default 2500 msec
12	Get/Set	Timeout Action	USINT	
13	Get	Prod. Path Length	USINT	6
14	Get	Production Path Log. Seg., Class Class Number Log.Seg., Instance Instance Number Log.Seg., Attribute Attribute Number	STRUCT of USINT USINT USINT USINT USINT USINT	 0x20 0x04 0x24 0x64 0x30 0x03
15	Get	Cons. Path Length	USINT	6
16	Get	Consumed Path Log. Seg., Class Class Number Log.Seg., Instance Instance Number Log.Seg., Attribute Attribute Number	STRUCT of USINT USINT USINT USINT USINT USINT	 0x20 0x04 0x24 0x65 0x30 0x03

1110A181

A.7.3 Connection Object, Instance 3 Attributes (STROBE connection)

Refer to the ODVA DeviceNet™ specification.

A.7.4 Connection Object, Instance 3 Attributes (COS/CYCLIC connection)

Refer to the ODVA DeviceNet™ specification.

A.7.5 Connection Object, Instances 5-10 Attributes (UCMM or Dynamic I/O connections)

Refer to the ODVA DeviceNet™ specification.

A.7.6 Connection Object Common Services

Service Code	Class	Instance	Service Name
05 (0x05)	Yes	Yes	Reset
14 (0x0E)	Yes	Yes	Get_Attribute_Single
16 (0x10)	No	Yes	Get_Attribute_Single

1110A182

A.7.7 Connection Object Attributes

Connection State	Interpretation
0	Non-existent
1	Configuring
3	Established
4	Timed Out

1110A183

Connection Status – Attribute 1

Connection ID – Attribute 4 and 5

Where: xxxxxx = Node Address.

Connection 1 Produced Connection ID: 10xxxxxx011

Connection 1 Consumed Connection ID: 10xxxxxx100

Connection 2 Produced Connection ID: 01111xxxxxx

Connection 2 Consumed Connection ID: 10xxxxxx101

Watch Dog Activity – Attribute 12

Watch Dog Timeout Activity:

0 = Timeout (I/O Messaging default)

1 = Auto Delete (Explicit Messaging, fixed value)

2 = Auto Reset

A.8 Discrete Input Point (DIP) Object (Class Code 08 dec, 0x08 hex)

The Discrete Input Point (DIP) Object models discrete inputs in a product. You can use this object in applications as simple as a toggle switch or as complex as a discrete I/O control module. There is a separate instance for each discrete input available on the device. There is a maximum of 510 DIPs.

A.8.1 DIP Object, Instance 1 to NUMBER OF DIPs Attributes

Attribute	Access	Name	Type	Value
3	Get	Value	BOOL	0 = OFF 1 = ON
4	Get	Status	BOOL	0 = okay 1 = fault
100	Get/Set	Enable Latched Input	BOOL	0 = OFF 1 = ON
101	Get/Set	Latch Level	BOOL	0 = Latch on “0” 1 = Latch on “1” (default)

1110A185

A.8.2 DIP Object Common Services

Service Code	Class	Instance	Service Name
14 (0x0E)	Yes	Yes	Get_Attribute_Single

1110A186

A.8.3 DIP Object Attributes

Input State – Attribute 3

Attribute 3 provides the state of the specific digital input. A value of 0 indicates an OFF state and a value of 1 indicates an ON state. The Digital inputs provide feedback of the digital output states. If the corresponding output state is set to 0 these points may be used as inputs.

Input Status – Attribute 4

The Input status bit indicates if an error has occurred associated with a physical input. If the +24 Vdc power is not present the circuitry cannot accurately determine the state of the inputs and will set the Input Status bits . The status bits are cleared when the +24 Vdc power is restored.

Enable Latched Inputs - Attribute 100

This attribute allows the users to enable the latching feature of any specific DIP. “0” (default) will disable this feature, a “1” will enable this feature.

Latch Level - Attribute 101

Attribute 101 determines the latch level of a specific DIP. “0” will enable the DIP to select a low-level latch, a “1” (default) will enable the DIP to select a high-level latch.

Note

Bit 1 in the control byte will clear all latched DIPs. Bit 1 must be manually cleared when the control byte is in the poll. This will allow the next latched DIP to occur. If sending an explicit message, this bit will be cleared automatically.

A.9 Discrete Output Point (DOP) Object (Class Code 09 dec, 0x09 hex)

The Discrete Output Point (DOP) Object models discrete outputs in a product. You can use this object in applications as simple as a actuator or as complex as a discrete I/O control module. There is a separate instance for each discrete output available on the device. There is a maximum of 510 DOPs.

A.9.1 DOP Object, Instance 1 to NUMBER OF DOPs Attributes

Attribute	Access	Name	Type	Value
3	Get/Set	Value	BOOL	State of Output
4	Get/Set	Status	BOOL	Status of Output
5	Get/Set	Fault State	BOOL	0 = fault value 1 = no chg
6	Get/Set	Fault Value	BOOL	0 = Off 1 = On
7	Get/Set	Idle State	BOOL	0 = Idle value 1 = no chg
8	Get/Set	Idle Value	BOOL	0 = Off 1 = On

1110A188

A.9.2 DOP Object Common Services

Service Code	Class	Instance	Service Name
14 (0x0E)	Yes	Yes	Get_Attribute_Single
16 (0x10)	No	yes	Set_Attribute_Single

1110A178

A.9.3 DOP Object Attributes

Output State – Attribute 3

The Output State bit, attribute 3, allows for a Get or Set of any output instance. By setting a value of 0 (zero) the output will turn OFF. By writing a value of 1 the output will turn ON. A Get to attribute 3 will provide feedback as to the current state of the output.

Output Status – Attribute 4

The Output Status bit, attribute 4, indicates a fault condition. The output status will be set to 1 if either the I/O power drops below 18 V dc or if a short-circuit condition is detected on any of the outputs. The low-voltage status bit may be read through Class (0x64hex), Instance 1, Attribute 7.

Fault State – Attribute 5

The Fault State determines what action is taken if a software fault condition is detected due to a connection timeout.

Fault State	Action Taken
0	Set the output to the state determined by the Fault Value
1	Leave the output in the current state

1110A190

Fault Value – Attribute 6

The Fault Value determines the state of the DOP output if the Fault State bit is clear and a fault condition occurs.

Idle State – Attribute 7

The Idle State determines what action is taken if an idle condition is detected. Idle conditions occur if a Poll request packet is received with less than the calculated number of bytes. Refer to the Configuration object to determine the size of the Poll Request packets. A poll request of 0 bytes is typically used to force an idle condition.

Idle State	Action Taken
0	Set the output to the state determined by the Idle Value
1	Leave the output in the current state

1110A191

Idle Value – Attribute 8

The Idle Value is used to set the output if the Idle State bit is clear and an idle condition occurs.

A.10 Analog Input Point (AIP) Object (Class Code 10 dec, 0x0A hex)

There is a separate instance for each discrete input available on the device. There is a maximum of 510 AIPs.

A.10.1 AIP Object, Instance 1 to 510 Attributes

Attribute	Access	Name	Type	Value
3	Get	Value	UINT	0-0xFFFF
4	Get	Status	BOOL	0 = ok
7	Get/Set	Range	Byte	2
100	Get/Set	Use Attribute 101	BOOL	0 = No 1 = Yes
101	Get/Set	AIP Configuration Word	UINT	Use value specified in module-specific data sheet
102	Get/Set	AIP Configuration Word in Poll	BOOL	0 = Data not in Poll 1 = Data in Poll

1436A126

A.10.2 AIP Object Common Services

Service Code	Class	Instance	Service Name
14 (0x0E)	Yes	Yes	Get_Attribute_Single
16 (0x10)	No	yes	Set_Attribute_Single

1110A178

A.10.3 AIP Object Attributes

Value - Attribute 3

Analog input values are reported using Offset Binary encoding when operating in the bipolar range. Unipolar inputs are reported as unsigned integers. The Range attribute determines the type of data returned. See specific analog input data sheet for details.

Status – Attribute 4

If the analog input status bit is set it indicates that a hardware fault has occurred during the previous analog read. The value is left at the last valid value read. A fault during the analog input function results in a Major Unrecoverable Fault condition (see Identity object).

Range - Attribute 7

The AIP Range value is used when performing Explicit Message reads to the AIP or during polling. The AIP Range values are retained in Flash memory.

Range Value	Description
0	-10 to +10 Volts
2	0 to + 10 Volts (Default)
3	+4 to +20 mA
6	0 to + 20 mA
7	-20 to + 20 mA

1110A194

Use Attribute 101 – Attribute 100

A “Set” will give the user access right to a configuration output word (programming word) found in attribute 101

Note

Attribute 100 must be set to gain access to attribute 101.

Input Configuration Word - Attribute 101

This word can be used when the default settings for the analog input module need to be changed. Programming information can be obtained from the module-specific data sheet.

Analog Input Configuration Word in DNET I/O - Attribute 102

This attribute places the configuration word (configurable modules only) for the specified input channel (instance) into the DNET I/O. This data supercedes any data in Attribute 101. It allows easier and faster access to the configuration word for modules such as the AI8 multiplexed module, where the configuration data may change frequently. Note that this attribute is settable for each input instance (word). Modules, such as the AI8, may have more than one input channel per input word.

A.11 Analog Output Point (AOP) Object (Class Code 11 dec, 0x0B hex)

The bus coupler supports Analog Output Point (AOP). There is a separate instance for each discrete output available on the device. There are a maximum of 510 AOPs.

A.11.1 AOP Object, Instance 1 to 510 Attributes

Attribute	Access	Name	Type	Value
3	Get/Set	Value	UINT	0...0xFFFF
4	Get	Status	BOOL	0 = ok
9	Get/Set	Fault State	BYTE	0...3
10	Get/Set	Idle State	BYTE	0...3
11	Get/Set	Fault Value	INT	0...0xFFFF
12	Get/Set	Idle Value	INT	0...0xFFFF
100	Get	Response Value	UINT	0...0xFFFF

1436B127

A.11.2 AOP Object Common Services

Service Code	Class	Instance	Service Name
14 (0x0E)	Yes	Yes	Get_Attribute_Single
16 (0x10)	No	yes	Set_Attribute_Single

1110A178

A.11.3 AOP Object Attributes

Value – Attribute 3

The analog output value is given in offset binary format. The value provided must be in the range 0...65535 (0...0xFFFFH), [0x7FFF for Loop 2 devices].

Value – Attribute 4

If the analog output status bit is set it indicates that a hardware fault has occurred during the previous analog write. The value is left at the last valid value write. A fault during the analog output function results in a Major Unrecoverable Fault condition (see Identity object).

Fault State – Attribute 9

The Fault State determines what action is taken if a fault condition is detected. Fault conditions include software conditions (connection timeout).

Fault State	Action Taken
0	Hold the last value
1	Set to low limit (0 V dc)
2	Set to high limit (+10 V dc)
3	Set to value determined by Fault Value.

1110A198

Idle State – Attribute 10

The Idle State determines what action is taken if an idle condition is detected. Idle conditions occur if a consumed request packet is received with less than the calculated number of bytes. Refer to the Configuration object to determine the size of the consumed request packets. A consumed request of 0 bytes is typically used to force an idle condition.

Idle State	Action Taken
0	Hold the last value
1	Set to low limit (0 V dc)
2	Set to high limit (+10 V dc)
3	Set to value determined by Fault Value.

1110A199

Fault Value – Attribute 11

The Fault Value determines the output if the Fault State bit is set to 3 and a fault condition occurs. The value must be in the range 0...65535 (0...0FFFFH), [0x7FFF for Loop 2 devices].

Idle Value – Attribute 12

The Idle Value is used to set the output if the Idle State bit is set to 3 and an idle condition occurs. The value must be in the range 0...65535 (0...0FFFFH), [0x7FFF for Loop 2 devices].

Note:

Fault State and Idle State settings 1 "Set to low limit" and 2 "Set to high limit" can not be used with the Loop 2 analog output modules and the IB IL AO2/U/BP (2 channel Inline bipolar analog output module). Only settings 0 "Hold the last value" and 3 "Set to value determined by Fault Value" can be used.

Response Value – Attribute 100

The Response Value attribute contains the input data associated with the analog output module. Refer to the module's specific data sheet for information on the meaning of this data value.

A.12 Acknowledge Handler Object (Class Code 43 dec, 0x2D hex)

The Acknowledge Handler Object is used to manage the reception of messages acknowledgments.

A.12.1 Acknowledge Handler Object Instance Attributes

Attribute	Access	Name	Type	Value
1	Get/Set	Acknowledge Timer	UINT	See ODVA Specs.
2	Get/Set	Retry Limit	USINT	See ODVA Specs.
3	Get/Set	COS Producing Connection Instance	UINT	See ODVA Specs.
4	Get	Ack List Size	BYTE	See ODVA Specs.
5	Get	Ack List	BYTE, Array of UINT	See ODVA Specs.
6	Get	Data with ACK Path List Size	BYTE	See ODVA Specs.
7	Get	Data with ACK Path List Size	BYTE, Array of USINT, UINT	See ODVA Specs.

1110A242

A.12.2 Acknowledge Handler Object Common Services

Service Code	Class	Instance	Service Name
14 (0x0E)	Yes	Yes	Get_Attribute_Single
16 (0x10)	No	yes	Set_Attribute_Single

1110A178

A.13 Configuration Object (Class Code 100 dec, 0x64 hex)

The bus coupler poll request/response packets can be large. In some applications it may be desired to reduce the packet size if not all the I/O channels are in use. The configuration object will adjust the poll request/response packet sizes. In addition, the configuration object gives access to several operational parameters such as power supply and temperature conditions.

A.13.1 Configuration Object, Instance 1 Attributes

Attribute	Access	Name	Type	Description
3	Get/Set	Number Digital Inputs	USINT (see note)	The Number Digital Inputs attribute determines the number of digital input points to be returned in the produced response packet.
4	Get/Set	Number Digital Outputs	USINT (see note)	The Number Digital Outputs attribute determines the number of output points to be processed in the consumed request packet.
5	Get/Set	Number Analog Inputs	USINT (see note)	The Number Analog Inputs attribute determines the number of analog input channels to be returned in the produced response packet. Each analog input produces 2 bytes of data in the produced response packet.
6	Get/Set	Number Analog Outputs	USINT (see note)	The Number Analog Outputs attribute determines the number of analog output channels. Each analog output consumes 2 bytes of data in the consumed request packet.
7	Get/Set	Add All I/O	BOOL (see note)	The Add All I/O attribute will add all Inline I/O modules to the produced and consumed data sizes.
8	Get/Set	Accept New Configuration	BOOL (see note)	The Accept New Configuration attribute will keep the current polled I/O setup even though modules may have been added or deleted. This will clear the I/O module change flag in the status attribute.
9	Get	Module Change Flag	BYTE	Inline Modules have been changed since last configuration.
10	Get/Set	Add All Mode	BYTE (see note)	The Add All Mode attribute allows the user to select the type of I/Os and faults that will be added when the Add All I/O attribute (7) is set. Default is 0x007F meaning that all DIPs, DOPs, AIPs, AOPs, Special Function, PCP Special Function, and Serial Communication modules are added to the poll. No faults will be added by default.

Note
Changing the configuration object may cause the consumed and produced sizes to be changed. These values are retained in flash memory and may only be set when the I/O connection is not in the running state.

Fault Value							
Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
			SPC	AOPs	AIPs	DOPs	DIPs
I/Os							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	SCO	PCP	SPC	AOPs	AIPs	DOPs	DIPs

1436B077

A.13.2 Configuration Object, Instance 1 Attributes (continued)

Attribute	Access	Name	Type	Description
11	Get/Set	Use Inline Status	BOOL (see note)	When set, the first byte of the poll response contains the Inline status. The second byte contains the number of the first module in the local bus that is faulted. Adds 1 byte to the produced size.
13	Get/Set	Special Function	BOOL (see note)	When set, the bus coupler will put the Process data for the special function modules with the Data in Poll attribute set in the Poll Command and Poll Response.
14	Get/Set	Pad I/O	BOOL (see note)	When set, this attribute will add an extra byte, if necessary, to align the analog or special function inputs and outputs to word boundaries. Will add 0 to 1 byte to the Consumed and/or Produced size.
15	Get/Set	Number of DIP Faults	UINT (see note)	Selects the number of DIP faults to be added to the poll response.
16	Get/Set	Number of DOP Faults	UINT (see note)	Selects the number of DOP faults to be added to the poll response.
17	Get/Set	Number of AIP Faults	UINT (see note)	Selects the number of AIP faults to be added to the poll response.
18	Get/Set	Number of AOP Faults	UINT (see note)	Selects the number of AOP faults to be added to the poll response.
19	Get/Set	Number of Special Faults	UINT (see note)	Selects the number of Special Function faults to be added to the poll response.
20	Get	Produced Size	UINT	This attribute allows the user to determine the Produced Size of the device.
21	Get	Consumed Size	UINT	This attribute allows the user to determine the Consumed Size of the device.
22	Get/Set	Number of Reserved DIPs	UINT (see note)	This attribute allows the user to reserve bits in the polled I/O for future expansion of actual Digital Input Points (DIP). The polled I/O contains the number of actual bits or reserved bits (whichever is greater).
23	Get/Set	Number of Reserved DOPs	UINT (see note)	This attribute allows the user to reserve bits in the polled I/O for future expansion of actual Digital Output Points (DOP). The polled I/O contains the number of actual bits or reserved bits (whichever is greater).
24	Get/Set	Fault Mode	USINT	This attribute allows the user to select which action is taken during a major failure such as those described in attributes 25 through 32.
25	Get/Set	CRC Fault Mode	USINT	Determines the affect on the Identity Object State and Status Attributes during a CRC Fault.

Value	Effect on Identity Status
0	None
1	Minor Recoverable Fault
2	Major Recoverable Fault

1436B007

A.13.3 Configuration Object, Instance 1 Attributes (continued)

<table>
<tr><th>Attribute</th><th>Access</th><th>Name</th><th>Type</th><th>Description</th></tr>
<tr><td>26</td><td>Get/Set</td><td>PF Fault Mode</td><td>USINT</td><td>Determines the affect on the Identity Object State and Status Attributes during a PF
<table>
<tr><th>Value</th><th>Effect on Identity Status</th></tr>
<tr><td>0</td><td>None</td></tr>
<tr><td>1</td><td>Minor Recoverable Fault</td></tr>
<tr><td>2</td><td>Major Recoverable Fault</td></tr>
</table></td></tr>
<tr><td>27</td><td>Get/Set</td><td>Power Fault Mode</td><td>USINT</td><td>Determines the affect on the Identity Object State and Status Attributes during a Power Fault.
<table>
<tr><th>Value</th><th>Effect on Identity Status</th></tr>
<tr><td>0</td><td>None</td></tr>
<tr><td>1</td><td>Minor Recoverable Fault</td></tr>
<tr><td>2</td><td>Major Recoverable Fault</td></tr>
</table></td></tr>
<tr><td>28</td><td>Get/Set</td><td>Module Change Fault</td><td>USINT</td><td>Determines the affect on the Identity Object State and Status Attributes during a Module Change Fault.
<table>
<tr><th>Value</th><th>Effect on Identity Status</th></tr>
<tr><td>0</td><td>None</td></tr>
<tr><td>1</td><td>Minor Recoverable Fault</td></tr>
<tr><td>2</td><td>Major Recoverable Fault</td></tr>
</table></td></tr>
<tr><td>29</td><td>Get/Set</td><td>Configuration Fault Mode</td><td>USINT</td><td>Determines the affect on the Identity Object State and Status Attributes during a Configuration Fault.
<table>
<tr><th>Value</th><th>Effect on Identity Status</th></tr>
<tr><td>0</td><td>None</td></tr>
<tr><td>1</td><td>Minor Recoverable Fault</td></tr>
<tr><td>2</td><td>Major Recoverable Fault</td></tr>
</table></td></tr>
<tr><td>30</td><td>Get/Set</td><td>Connection Fault Mode</td><td>USINT</td><td>Determines the affect on the Identity Object State and Status Attributes during a Connection Fault.
<table>
<tr><th>Value</th><th>Effect on Identity Status</th></tr>
<tr><td>0</td><td>None</td></tr>
<tr><td>1</td><td>Minor Recoverable Fault</td></tr>
<tr><td>2</td><td>Major Recoverable Fault</td></tr>
</table></td></tr>
<tr><td>31</td><td>Get/Set</td><td>Fault Mode Cycles</td><td>USINT</td><td>Determines the affect on the Identity Object State and Status Attributes during a Fault Cycles Mode.
<table>
<tr><th>Value</th><th>Effect on Identity Status</th></tr>
<tr><td>0</td><td>None</td></tr>
<tr><td>1</td><td>Minor Recoverable Fault</td></tr>
<tr><td>2</td><td>Major Recoverable Fault</td></tr>
</table></td></tr>
<tr><td>32</td><td>Get/Set</td><td>Inline Control Byte in Poll</td><td>BOOL (see note)</td><td>When set to 1, the first byte of the Poll Command is the Inline Byte (see Inline Object).</td></tr>
</table>

1436A078

A.13.4 Configuration Object, Instance 1 Attributes (continued)

Attribute	Access	Name	Type	Description
33	Get/Set	I/O Bank Select	Byte	0 = Allows access to I/O instances 1 - 255 1 = Allows access to I/O instances 256 - 510
34	Get/Set	DeviceNet Power Fault Detect	Byte	When UL and DeviceNet power supplies are wired separately: 0 = Unit will not detect DeviceNet power fault. Outputs will hold last state until station is reset. Outputs will then turn OFF. 1 = Unit detects DeviceNet power fault. Outputs will turn OFF.
35	Get/Set	Number of reserved AIPs	UINT	This attribute allows the user to reserve words in the polled I/O for future expansion of actual Analog Input Points (AIP). The polled I/O contains the number of actual words or reserved words (whichever is greater).
36	Get/Set	Number of reserved AOPs	UINT	This attribute allows the user to reserve words in the polled I/O for future expansion of actual Analog Output Points (AOP). The polled I/O contains the number of actual bits or reserved bits (whichever is greater).
37	Get/Set	I/O mapping revision	USINT	Determines how types of Inline modules will be classified in the DeviceNet™ Objects.

1436B056

A.13.5 Configuration Object Common Services

Service Code	Class	Instance	Service Name
14 (0x0E)	Yes	Yes	Get_Attribute_Single
16 (0x10)	No	yes	Set_Attribute_Single

1110A178

A.14 Inline Interface Object (Class Code 101 dec, 0x65 hex)

The Inline Interface Object allows the user to control and monitor the Inline interface on the coupler.

A.14.1 Inline Interface Object Common Services

Service Code	Class	Instance	Service Name
14 (0x0E)	Yes	Yes	Get_Attribute_Single
16 (0x10)	No	yes	Set_Attribute_Single

1110A178

A.14.2 Inline Interface Object, Instance 1 Attributes

<table>
<tr><th>Attribute</th><th>Access</th><th>Name</th><th>Type</th><th>Description</th></tr>
<tr><td>3</td><td>Get</td><td>Inline Baud Rate</td><td>USINT</td><td>0 = 500 kbits/s</td></tr>
<tr><td>4</td><td>Get</td><td>Inline Status</td><td>BYTE</td><td>Inline Status<table><tr><th>Bit 7</th><th>Bit 6</th><th>Bit 5</th><th>Bit 4</th><th>Bit 3</th><th>Bit 2</th><th>Bit 1</th><th>Bit 0</th></tr><tr><td></td><td>Fault Cycles</td><td>Connec-tion</td><td>Configur-ing</td><td>Module Change</td><td>Power Fault</td><td>Peripheral Fault</td><td>CRC</td></tr></table></td></tr>
<tr><td>5</td><td>Get</td><td>First Faulted Module</td><td>USINT</td><td>Contains the number of the first module that is faulted
1 = bus coupler</td></tr>
<tr><td>6</td><td>Get/Set</td><td>Max Retry</td><td>USINT</td><td>Sets the number of local data transmissions that the coupler will accept before flagging an error. Default = 32.</td></tr>
<tr><td>7</td><td>Get</td><td>Number of Modules</td><td>UINT</td><td>Displays the number of Inline I/O modules that the bus coupler detected.</td></tr>
<tr><td>8</td><td>Get</td><td>Number of Bits</td><td>UINT</td><td>Displays the Process Data size of the Inline modules in Bits (input & output bits).</td></tr>
<tr><td>9</td><td>Get</td><td>Number of Bytes</td><td>UINT</td><td>Displays the Process Data size of the Inline modules in Bytes (input & output bytes).</td></tr>
<tr><td>11</td><td>Get</td><td>Scans Per Second</td><td>UINT</td><td>Displays the number of local I/O scans per second.</td></tr>
<tr><td>12</td><td>Get</td><td>Loop Diagnostic Count</td><td>UINT</td><td>Displays the loop diagnostic count during a connection failure.</td></tr>
<tr><td>13</td><td>Get</td><td>Connection Failure Endpoint #1</td><td>USINT</td><td>Displays the number of the module at the first end of a connection failure.</td></tr>
<tr><td>14</td><td>Get</td><td>Connection Failure Endpoint #2</td><td>USINT</td><td>Displays the number of the module at the second end of a connection failure.</td></tr>
<tr><td>15</td><td>Get/Set</td><td>Latched Inline Status</td><td>BYTE</td><td>Displays the latched value of the Inline Status during the last failure. See Inline Status Attribute.</td></tr>
<tr><td>16</td><td>Get</td><td>Latched Faulted Module</td><td>USINT</td><td>Contains the number of the first module that was faulted during the last fault; 1 = bus coupler.</td></tr>
<tr><td>17</td><td>Get</td><td>Latched Connection Failure Endpoint # 1</td><td>USINT</td><td>Displays the number of the module at the first end of a connection failure latched during the last connection failure.</td></tr>
<tr><td>18</td><td>Get</td><td>Latched Connection Failure Endpoint # 2</td><td>USINT</td><td>Displays the number of the module at the second end of a connection failure latched during the last connection failure.</td></tr>
<tr><td>19</td><td>Get</td><td>Power Supply Fault</td><td>BYTE</td><td>Displays the status of the power supplies connected to the bus coupler
• A value of 1 indicates a power supply out of range.
• A value of 0 indicates power supply is OK.
Power Supply Fault<table><tr><th>Bit 7</th><th>Bit 6</th><th>Bit 5</th><th>Bit 4</th><th>Bit 3</th><th>Bit 2</th><th>Bit 1</th><th>Bit 0</th></tr><tr><td></td><td></td><td></td><td></td><td>UM</td><td>US</td><td>UL</td><td>DNET</td></tr></table></td></tr>
<tr><td>20</td><td>Get/Set</td><td>Inline Control Byte</td><td>BYTE</td><td>Inline Control Byte<table><tr><th>Bit 7</th><th>Bit 6</th><th>Bit 5</th><th>Bit 4</th><th>Bit 3</th><th>Bit 2</th><th>Bit 1</th><th>Bit 0</th></tr><tr><td></td><td></td><td></td><td></td><td></td><td></td><td>Clear input latches</td><td>Acknowl-edge, latched PF</td></tr></table></td></tr>
<tr><td>21
↓
30</td><td>Get
↓
Get</td><td>Error History (most recent)
↓
Error History (last saved)</td><td>UINT
↓
UINT</td><td>Contains the most recent logged error information.
↓
Contains the last saved error information.</td></tr>
</table>

1436B005

A.15 Inline Module Object (Class Code 102 dec, 0x66 hex)

The Inline Module Object allows the user to monitor the Inline modules attached to the coupler.

A.15.1 Inline Module Object, Instance 1 to 63 Attributes

Attribute	Access	Name	Type	Description
3	Get	Module Id (Config)	UINT	Displays the 16-bit ID for the module as the unit was configured.
4	Get	Module Id (Current)	UINT	Displays the 16-bit ID for the module currently.
5	Get	DeviceNet™ Class	USINT	Reflects the Number of the DeviceNet™ Class that the module is mapped to (i.e. 8 = DIP, 9 = DOP, etc.).
6	Get	First DeviceNet™ Instance	UINT	Reflects the first instance of the DeviceNet™ object that this module is mapped to.
7	Get	Last DeviceNet™ Instance	UINT	Reflects the last instance of the DeviceNet™ object that this module is mapped to.

1436A057

A.15.2 Inline Module Object Common Services

Service Code	Class	Instance	Service Name
14 (0x0E)	Yes	Yes	Get_Attribute_Single
16 (0x10)	No	yes	Set_Attribute_Single

1110A178

A.16 Inline Special Function Object (Class Code 103 dec, 0x67 hex)

The Inline Special Function Object allows the user to control and monitor the Inline modules attached to the coupler that do not map to any standard DNet Object.

A.16.1 Inline Special Function Object, Instance 1 to 63 Attributes

Attribute	Access	Name	Type	Description
3	Get	In Data	ARRAY	Input data returned from Inline module. Data size is determined by the module.
4	Get/Set	Out Data	ARRAY	Output data sent from the bus coupler to the Inline module. Data size is determined by the module.
5	Get	Data Size	USINT	Number of bytes of Process Data used by the module.
6	Get	Status	BOOL	Reflects the status of the Special Function module. 0 = OK 1 = Faulted
7	Ge/Set	Data In Poll	BOOL	When set, the IN data is in the produced response and the OUT data is in the consumed request . This attribute affects the produce and consume size of the bus coupler and is therefore, only selectable when the Poll connection is not in the established state.

1436A006

A.16.2 Inline Special Function Object Common Services

Service Code	Class	Instance	Service Name
14 (0x0E)	Yes	Yes	Get_Attribute_Single
16 (0x10)	No	yes	Set_Attribute_Single

1110A178

A.17 Mask Object (Class Code 104 dec, 0x68 hex)

The COS Mask Object allows the user control which bits in the Config In Data Attribute cause a COS message to be generated.

Note

A network heartbeat will update all COS inputs.

A.17.1 COS Mask Object Common Services

Service Code	Class	Instance	Service Name
14 (0x0E)	Yes	Yes	Get_Attribute_Single
16 (0x10)	No	yes	Set_Attribute_Single

1110A178

A.17.2 COS Mask Object, Instance 1 Attributes

Attribute	Access	Name	Type	Description
3	Get/Set	COS Mask Value	USINT	Contains the actual mask byte value that is ANDed with the Config IN Data to determine whether to generate a COS message or not.
4	Get/Set	COS Mask Index	USINT	Points to specific byte in the COS Mask Array allowing the byte value to be get or set. See Appendix D
5	Get/Set	COS Byte Value	BYTE	Gets or sets the byte value in the COS Mask Array that is indexed by Attribute 4. See Appendix D
6	Get/Set	COS Add All Mode	WORD	Automatically generates a COS Mask based on the following bits, when the add all attribute is set to a 1. **Faults**: Bit 15 — Bit 14: Inline Status — Bit 13 — Bit 12: SPC — Bit 11: AOPs — Bit 10: AIPs — Bit 9: DOPs — Bit 8: DIPs. **I/Os**: Bit 7 — Bit 6 — Bit 5 — Bit 4: SPC — Bit 3 — Bit 2: AIPs — Bit 1 — Bit 0: DIPs
7	Get/Set	COS Add All	BOOL	When set to a 1, generates a COS Mask based on the COS Add All Mode attribute.
9	Get/Set	Enable Inline Status (include by default)	BOOL	When set to 1, a COS message will be generated when any bits in the Inline Status Word change state.
10	Get/Set	Enable DIP Faults (include by default)	BOOL	When set to 1, a COS message will be generated when any DIP Fault changes state.
11	Get/Set	Enable DOP Faults	BOOL	When set to 1, a COS message will be generated when any DOP Fault changes state.
12	Get/Set	Enable AIP Faults	BOOL	When set to 1, a COS message will be generated when any AIP Fault changes state.
13	Get/Set	Enable AOP Faults	BOOL	When set to 1, a COS message will be generated when any AOP Fault changes state.
14	Get/Set	Enable Special Function Faults	BOOL	When set to 1, a COS message will be generated when any Special Function Module Fault changes state.
15	Get/Set	Enable DIPs	BOOL	When set to 1, a COS message will be generated when any DIP value changes state.
16	Get/Set	Enable AIPs	BOOL	When set to 1, a COS message will be generated when any AIP value changes state.
17	Get/Set	Enable Special Function In Data	BOOL	When set to 1, a COS message will be generated when any Special Function In Data changes state.
18	Get/Set	AIP Mask	UINT	This value generates a bit pattern used to mask each AIP value. Example: AIP Mask = 0xFF00 any time an AIP's upper 8 bits change, a COS message will be generated.

1436B058

A.18 PCP Special Function Object (Class Code 105 dec, 0x69 hex)

The PCP Special Function Object allows the user to access I/O modules that support the PCP protocol.

A.18.1 Instance 1 to Number of PCP Modules (maximum of eight) Attributes

Attribute	Access	Name	Type	Description
3	Get	PDU Size	USINT	Contains the value of the PDU size for the PCP channel of a module (typical 64 bytes). It is the maximum number of bytes that the PCP channel can transfer per request/ response.
4	Get	PCP Size	USINT	Contains the value of the PCP channel size. It indicates the number of bytes that can be transferred during each Inline station scan.
5	Get	Status	BOOL	When the PCP Module Status attribute is 1, it indicates a module error. If 0, it indicates the module is ok.
6	Get/Set	PCP Request	ARRAY OF BYTE	Sends a request to the PCP module. The response can be read in Attribute 7. Only read/write services are available. **Request – No Invoke ID** • Byte 0: Service • Byte 1: Module Number • Byte 2: Index LSB • Byte 3: Index MSB • Byte 4: Subindex • Byte 5: Length • Byte 6: Data-block, if necessary • Byte N: Data-block, if necessary **Request – With Invoke ID** • Byte 0: Service • Byte 1: Invoke ID • Byte 2: Module Number • Byte 3: Index Low • Byte 4: Index High • Byte 5: Subindex • Byte 6: Length • Byte 7: Data-block, if necessary • Byte N: Data-block, if necessary Data length is determined by the PCP Index and PCP Subindex of the object to be read/written. The maximum data length is variable and can be up to the PDU size for the module (typically 64 bytes). See PDU Length Attribute.

1436C129-1

A.18.1 Instance 1 to Number of PCP Modules (maximum of eight) Attributes (continued)

Attribute	Access	Name	Type	Description
7	Get	PCP Response	ARRAY OF BYTE	Gets a response from the PCP module. The format is as follow(s) **Successful Response – No Invoke ID** • Byte 0: Service • Byte 1: Status • Byte 2: Length • Byte 3: Data-block, if necessary • Byte N: Data-block, if necessary **Successful Response – With Invoke ID** • Byte 0: Service • Byte 1: Invoke ID • Byte 2: Status • Byte 3: Length • Byte 4: Data-block, if necessary • Byte N: Data-block, if necessary Data length is determined by the PCP Index and PCP Subindex of the object to be read/written. The maximum data length is variable and can be up to the PDU size for the module (typically 64 bytes). See PDU Length Attribute. **Error Response – No Invoke ID** • Byte 0: Service • Byte 1: Status • Byte 2: Error Class • Byte 3: Error Code • Byte 4: Additional Error Code 1 LSB, if necessary • Byte 5: Additional Error Code 1 MSB, if necessary • Byte 6: Additional Error Code 2 LSB, if necessary • Byte 7: Additional Error Code 2 MSB, if necessary **Error Response – With Invoke ID** • Byte 0: Service • Byte 1: Invoke ID • Byte 2: Status • Byte 3: Error Class • Byte 4: Error Code • Byte 5: Additional Error Code 1 LSB, if necessary • Byte 6: Additional Error Code 1 MSB, if necessary • Byte 7: Additional Error Code 2 LSB, if necessary • Byte 8: Additional Error Code 2 MSB, if necessary Data length is determined by the PCP Index and PCP Subindex of the object to be read/written. The maximum data length is variable and can be up to the PDU size for the module (typically 64 bytes). See PDU Length Attribute.
8	Get/Set	PCP Module	USINT	Set the PCP module number that the reads and writes (Attributes 11 and 14) will access.

1436C129-2

A.18.1 Instance 1 to Number of PCP Modules (maximum of eight) Attributes (continued)

Attribute	Access	Name	Type	Description
9	Get/Set	PCP Write Index	UINT	Sets the Index of the PCP Object that will be written when the user writes to the PCP Write Data Attribute (Attribute 11).
10	Get/Set	PCP Write Subindex	USINT	Sets the Subindex of the PCP Object that will be written when the user writes to the PCP Write Data Attribute (Attribute 11).
11	Get/Set	PCP Write Data	SHORT_STRING	Allows the user to write data to the PCP Object that is referenced by the PCP Module Attribute, PCP Write Index, and PCP Write Subindex. The first byte in the string indicates the number of bytes to be written.
12	Get/Set	PCP Read Index	UINT	Sets the Index of the PCP Object that will be read when the user reads from the PCP Read Data Attribute (Attribute 14).
13	Get/Set	PCP Read Subindex	USINT	Sets the Subindex of the PCP Object that will be read when the user reads from the PCP Read Data Attribute (Attribute 14).
14	Get	PCP Read Data	SHORT_STRING	Allows the user to read data from the PCP Object that is referenced by the PCP Module Attribute, PCP Read Index, and PCP Read Subindex. The first byte in the string indicates the number of bytes to be read.
15	Get/Set	PCP Request Fragment	Service: STRUCT of BYTE Data: ARRAY of BYTE[7]	Allows the user to send requests to the PCP modules. See fragmented service information in Section 6.
16	Get	PCP Response Fragment	Service: STRUCT of BYTE Data: ARRAY of BYTE[7]	Allows the user to get responses from the PCP modules. See fragmented service information in Section 6.
17	Get/Set	PCP Fragment Data in Poll	BOOL	If set, the PCP Fragmented Message Data for this instance is added to the Poll.
18	Get	Process Data Size	USINT	Number of bytes of Process Data used by the module.
19	Get	Process Data In	ARRAY	Data returned from the Inline module to IL DN BK. Data size is determined the I/O module.
20	Get/Set	Process Data Out	ARRAY	Data sent from the IL DN BK to the Inline module. Data size is determined the I/O module.
21	Get/Set	Process Data in Poll	BOOL	If set, the PCP module's Process Data for this instance is added to the Poll.
22	Get/Set	PCP Write Data Invoke ID	USINT	Invoke ID Number can be found in the specific module's data sheet or manual.
23	Get/Set	PCP Read Data Invoke ID	USINT	Invoke ID Number can be found in the specific module's data sheet or manual.

1436C129-3

A.18.2 PCP Special Function Object Common Services

Service Code	Class	Instance	Service Name
14 (0x0E)	Yes	Yes	Get_Attribute_Single
16 (0x10)	No	yes	Set_Attribute_Single

1110A178

A.19 Serial Communications Object (Class Code 106 dec, 0x6A hex)

The Serial Communications Object allows the user to control and transfer serial data on RS232 and RS485/RS422 modules.

A.19.1 Instance 1 to Number of Serial Modules (Maximum of eight) Attributes

<table>
<tr><th>Attribute</th><th>Access</th><th>Name</th><th>Type</th><th>Description</th></tr>
<tr><td>3</td><td>Get</td><td>Module Type</td><td>USINT</td><td>Value indicates type of serial module.
0 = RS232
1 = RS485/RS422</td></tr>
<tr><td>4</td><td>Get</td><td>Module Status</td><td>BOOL</td><td>Value indicates module status.
0 = OK
1 = Faulted</td></tr>
<tr><td>5</td><td>Get</td><td>Serial Status Word</td><td>WORD</td><td>Serial Status Word Settings
<table>
<tr><th colspan="8">Serial Status Word—MSB</th></tr>
<tr><td>Bit 15</td><td>Bit 14</td><td>Bit 13</td><td>Bit 12</td><td>Bit 11</td><td>Bit 10</td><td>Bit 9</td><td>Bit 8</td></tr>
<tr><td colspan="8">Number of Received Characters—Mode Dependant</td></tr>
</table>
<table>
<tr><th colspan="8">Serial Status Word—LSB</th></tr>
<tr><td>Bit 7</td><td>Bit 6</td><td>Bit 5</td><td>Bit 4</td><td>Bit 3</td><td>Bit 2</td><td>Bit 1</td><td>Bit 0</td></tr>
<tr><td>Reserved</td><td>Transmit Buffer Not Empty</td><td>Transmit Buffer Full</td><td>Receive Buffer Full</td><td>Re-Init Executed</td><td>Send Error</td><td>Receive Error</td><td>Receive Buffer Not Empty</td></tr>
</table></td></tr>
<tr><td>6</td><td>Get/Set</td><td>Serial Control Word</td><td>WORD</td><td>Serial Control Word Settings
<table>
<tr><th colspan="8">Serial Control Word—MSB</th></tr>
<tr><td>Bit 15</td><td>Bit 14</td><td>Bit 13</td><td>Bit 12</td><td>Bit 11</td><td>Bit 10</td><td>Bit 9</td><td>Bit 8</td></tr>
<tr><td colspan="8">Reserved</td></tr>
</table>
<table>
<tr><th colspan="8">Serial Control Word—LSB</th></tr>
<tr><td>Bit 7</td><td>Bit 6</td><td>Bit 5</td><td>Bit 4</td><td>Bit 3</td><td>Bit 2</td><td>Bit 1</td><td>Bit 0</td></tr>
<tr><td>DTR</td><td>Reserved</td><td>Reserved</td><td>Reserved</td><td>Execute Re-Init</td><td>Reset Send Error</td><td>Reset Receive Error</td><td>Reserved</td></tr>
</table></td></tr>
<tr><td>7</td><td>Get</td><td>Receive Data</td><td>SHORT_STRING</td><td>Allows user to read Serial Data in one Get_Attribute_Single command.</td></tr>
<tr><td>8</td><td>Get/Set</td><td>Transmit Data</td><td>SHORT_STRING</td><td>Allows user to transmit Serial Data in one Set_Attribute_Single command.</td></tr>
<tr><td>9</td><td>Get</td><td>Receive Data Fragment</td><td>STRUCT of
Service: BYTE
Data: Array of BYTE</td><td>Allows user to read Serial Data in fragments. See Section 6.</td></tr>
<tr><td>10</td><td>Get/Set</td><td>Transmit Data Fragment</td><td>STRUCT of
Service: BYTE
Data: Array of BYTE</td><td>Allows user to write Serial Data in fragments. See Section 6.</td></tr>
<tr><td>11</td><td>Get/Set</td><td>Protocol</td><td>USINT</td><td>Protocol Settings
<table>
<tr><th>Code</th><th>Meaning</th></tr>
<tr><td>00 hex</td><td>Transparent</td></tr>
<tr><td>01 hex</td><td>End-to-end</td></tr>
<tr><td>02 hex</td><td>Dual buffer</td></tr>
<tr><td>03 hex</td><td>3964R</td></tr>
<tr><td>04 hex</td><td>XON/XOFF</td></tr>
</table></td></tr>
<tr><td>12</td><td>Get/Set</td><td>Baud Rate</td><td>USINT</td><td>Baud Rate Settings
<table>
<tr><th>Code</th><th>Meaning</th></tr>
<tr><td>00 hex</td><td>110</td></tr>
<tr><td>01 hex</td><td>300</td></tr>
<tr><td>02 hex</td><td>600</td></tr>
<tr><td>03 hex</td><td>1200</td></tr>
<tr><td>04 hex</td><td>1800</td></tr>
<tr><td>05 hex</td><td>2400</td></tr>
<tr><td>06 hex</td><td>4800</td></tr>
<tr><td>07 hex</td><td>9600</td></tr>
<tr><td>08 hex</td><td>19200</td></tr>
</table></td></tr>
</table>

1436A128-1

A.19.2 Instance 1 to Number of Serial Modules (Maximum of eight) Attributes (continued)

<table>
<tr><th>Attribute</th><th>Access</th><th>Name</th><th>Type</th><th>Description</th></tr>
<tr><td>13</td><td>Get/Set</td><td>Data Width</td><td>USINT</td><td>Data Width Settings
<table>
<tr><th rowspan="2">Code
Data</th><th colspan="3">Meaning</th></tr>
<tr><th>Bits</th><th>Parity</th><th>Stop Bits</th></tr>
<tr><td>00 hex</td><td>7</td><td>Even</td><td>1</td></tr>
<tr><td>01 hex</td><td>7</td><td>Odd</td><td>1</td></tr>
<tr><td>02 hex</td><td>8</td><td>Even</td><td>1</td></tr>
<tr><td>03 hex</td><td>8</td><td>Odd</td><td>1</td></tr>
<tr><td>04 hex</td><td>8</td><td>Without 1</td><td>1</td></tr>
<tr><td>05 hex</td><td>7</td><td>Without 1</td><td>1</td></tr>
<tr><td>06 hex</td><td>7</td><td>Even</td><td>2</td></tr>
<tr><td>07 hex</td><td>7</td><td>Odd</td><td>2</td></tr>
<tr><td>08 hex</td><td>8</td><td>Even</td><td>2</td></tr>
<tr><td>09 hex</td><td>8</td><td>Odd</td><td>2</td></tr>
<tr><td>0A hex</td><td>8</td><td>Without 2</td><td>2</td></tr>
<tr><td>0B hex</td><td>7</td><td>Without 2</td><td>2</td></tr>
</table></td></tr>
<tr><td>16</td><td>Get/Set</td><td>Errror Pattern</td><td>USINT</td><td>Error Pattern Settings
<table>
<tr><th>Code</th><th>Meaning</th></tr>
<tr><td>24 hex</td><td>$</td></tr>
<tr><td>XX hex</td><td>Any character</td></tr>
</table></td></tr>
<tr><td>17</td><td>Get/Set</td><td>First Delimiter</td><td>USINT</td><td>First Delimiter Settings
<table>
<tr><th>Code</th><th>Meaning</th></tr>
<tr><td>0D hex</td><td>Carriage return (CR)</td></tr>
<tr><td>XX hex</td><td>Any character</td></tr>
</table></td></tr>
<tr><td>18</td><td>Get/Set</td><td>Second Delimiter</td><td>USINT</td><td>Second Delimiter Settings
<table>
<tr><th>Code</th><th>Meaning</th></tr>
<tr><td>0A hex</td><td>Line Feed (LF)</td></tr>
<tr><td>XX hex</td><td>Any character</td></tr>
</table></td></tr>
<tr><td>19</td><td>Get/Set</td><td>3964R Priority</td><td>USINT</td><td>3694R Priority Settings
<table>
<tr><th>Code</th><th>Meaning</th></tr>
<tr><td>00 hex</td><td>Low priority</td></tr>
<tr><td>01 hex</td><td>High priority</td></tr>
</table></td></tr>
<tr><td>20</td><td>Get/Set</td><td>Output Type</td><td>USINT</td><td>Output Type Settings
<table>
<tr><th>Code</th><th>Meaning</th></tr>
<tr><td>00 hex</td><td>RS-232 (Default for RS232 Module Type)</td></tr>
<tr><td>01 hex</td><td>RS-485 (Default for RS485 Module Type)</td></tr>
<tr><td>02 hex</td><td>RS-422</td></tr>
</table></td></tr>
<tr><td>21</td><td>Get/Set</td><td>DTR Control</td><td>USINT</td><td>DTR Control Settings (only valid for RS232 type)
<table>
<tr><th>Code</th><th>Meaning</th></tr>
<tr><td>00 hex</td><td>Automatic</td></tr>
<tr><td>01 hex</td><td>via process data</td></tr>
</table></td></tr>
<tr><td>22</td><td>Get/Set</td><td>Rotation Switch</td><td>USINT</td><td>Rotation Switch Settings
<table>
<tr><th>Code</th><th>Meaning</th></tr>
<tr><td>00 hex</td><td>No rotation</td></tr>
<tr><td>01 hex</td><td>Rotation</td></tr>
</table></td></tr>
<tr><td>23</td><td>Get/Set</td><td>XON Pattern</td><td>USINT</td><td>XON Pattern Settings
<table>
<tr><th>Code</th><th>Meaning</th></tr>
<tr><td>11 hex</td><td>Default</td></tr>
<tr><td>XX hex</td><td>Any character (not the same as XOFF pattern)</td></tr>
</table></td></tr>
</table>

1436A128-2

A.19.3 Instance 1 to Number of Serial Modules (Maximum of eight) Attributes (continued)

<table>
<tr><th>Attribute</th><th>Access</th><th>Name</th><th>Type</th><th>Description</th></tr>
<tr><td>24</td><td>Get/Set</td><td>XOFF Pattern</td><td>USINT</td><td>XOFF Pattern Settings
<table><tr><th>Code</th><th>Meaning</th></tr><tr><td>13 hex</td><td>Default</td></tr><tr><td>XX hex</td><td>Any character (not the same as XON pattern)</td></tr></table></td></tr>
<tr><td>31</td><td>Get/Set</td><td>Status Control Word in DNET I/O</td><td>BOOL</td><td>If set, the Serial Status Word is added to the Produced data and the Serial Control Word is added to the Consumed data.</td></tr>
<tr><td>32</td><td>Get/Set</td><td>Fragment Data in DNET I/O</td><td>BOOL</td><td>If set, the Received Data Fragment is added to the Produced data and the Transmit Data Fragment is added to the Consumed data.</td></tr>
<tr><td>33</td><td>Get/Set</td><td>Enable Serial Object</td><td>BOOL</td><td>If set to 0, the Serial Object for this instance is disabled. This allows the PCP Special Function Object to communicate with the serial module. See Section 6.</td></tr>
</table>

1436A128-3

A.19.4 Serial Communications Object Common Services

Service Code	Class	Instance	Service Name
14 (0x0E)	Yes	Yes	Get_Attribute_Single
16 (0x10)	No	yes	Set_Attribute_Single

1110A178

APPENDIX B
Electronic Data Sheet (EDS) File

Appendix B Contents

B.1 General

This section assumes that the user has a working knowledge of DeviceNet™ and RSNetWorx™ configuration software.

The bus coupler supports DeviceNet™ using ODVA standard Discrete Input Points (DIP), Discrete Output Points (DOP), Analog Input Points (AIP) and Analog Output Points (AOP). Additional objects include user defined Configuration, Inline Interface, Inline Module, Inline Special Function, PCP Special Function and Serial Communications objects.

The EDS (Electronic Data Sheet) file is the software interface between the bus coupler and the RSNetWorx™ configuration software package.

Within the EDS file, parameters are made available to (1) configure poll size, (2) evaluate the Inline station's status, (3) update system configurations, and (4) display other informative system characteristics. Table B-1 lists applicable EDS file parameters, their default values, and a description about their use.

Note

Any configuration changes after the initial power up configuration will require the use of the EDS file, an auto-configuration sequence or an Explicit Message to update the configuration. Explicit Messages and auto-configuration are covered in Section 3.

Table B-1. EDS File Parameters

Param. No.	Parameter Name	Default Value	Parameter Description
1	Use Inline Status	1	0 = (False) saved to the coupler disables the 2-byte addition to the poll. 1 = (True) adds 2 bytes of diagnostic data to the polled produced size. Bits are defined in Section 6 under Diagnostics.
2	Pad I/O	0	0 = (False) disables this up to 1-byte pad. 1 = (True) saved to the device will cause analog values or special function data to start on an even I/O memory byte. If the pad is not needed it will not be added.
3	Reserve Digital Inputs	0	0 = No digital input reservation will be made in the poll. x = Number of digital inputs to be reserved + number of currently used digital inputs.
4	Reserve Digital Outputs	0	0 = No digital output reservation will be made in the poll. x = Number of digital outputs to be reserved + number of currently used digital outputs.
5	Reserve Analog Inputs	0	0 = No analog input reservation will be made in the poll. x = Number of analog inputs to be reserved + number of currently used analog inputs.
6	Reserve Analog Outputs	0	0 = No analog output reservation will be made in the poll. x = Number of analog outputs to be reserved + number of currently used analog outputs.
7	I/O Mapping Revision	1	This parameter controls which DeviceNet™ I/O class Inline modules will be placed into. See Section 3.
8	Add All Mode	127	127 dec. allows all digital, analog, special function, PCP, and Serial modules to be added to the poll after an “Add All I/O”. Refer to Appendix A, Class Code 100 dec, 0x64 hex for additional information.
9	Add All I/O	0	1 = (True) saved to the coupler will add all Inline I/O to the polled connection
10	Produced Size	0	An upload from the device will display the current Produced Size.
11	Consumed Size	0	An upload from the device will display the current Consumed Size.
12	Number of Digital Inputs	0	An upload from the device will display the current number of digital inputs.
13	Number of Digital Outputs	0	An upload from the device will display the current number of digital outputs.
14	Number of Analog Inputs	0	An upload from the device will display the current number of analog inputs.
15	Number of Analog Outputs	0	An upload from the device will display the current number of analog outputs.
16	Number of DIP Faults	0	X = Number of DIP faults to be added to the poll.
17	Number of DOP Faults	0	X = Number of DOP faults to be added to the poll.
18	Number of AIP Faults	0	X = Number of AIP faults to be added to the poll.
19	Number of AOP Faults	0	X = Number of AOP faults to be added to the poll.
20	Number of Special Faults	0	X = Number of special function faults to be added to the poll.
21	Byte 0, Inline Status Word	0	An upload from the device will display byte 0 of the Inline status word.
22	First Faulted Module	0	An upload from the device will display the first module with a fault. Byte 1 of the Inline status word.

1436B008-1

Table B-1. EDS File Parameters (continued)

Param. No.	Parameter Name	Default Value	Parameter Description
23	Number of Modules	0	An upload displays the number of modules on the Inline backplane.
24	Number of Bits	0	An upload displays the number of bits on the Inline backplane (input bits + output bits).
25	Local Scans Per Second	0	An upload will display the number of local scans per second.
26	Fault Mode	0	Controls how the DeviceNet™ interface reacts to a fault. Outputs turn off by default.
27	Loop Diagnostic Count	0	Displays Loop diagnostic count during connection failure.
28	Connection Failure Endpoint #1	0	Displays the number of the module at the first end of the failed connection.
29	Connection Failure Endpoint #2	0	Displays the number of the module at the other end of the failed connection.
30	Latched Inline Status	0	Displays the latched error code of the last fault.
31	Latched Faulted Module	0	Displays the module number that had a problem during the last fault.
32	Latched Connection Failure Endpoint #1	0	Displays the number of the module at the first connection failure latched during the last connection error.
33	Latched Connection Failure Endpoint #2	0	Displays the number of the module at the second connection failure latched during the last connection error.
34	Power Supply Status	0	Displays the status of the power supplies connected to the module.
35	CRC Fault Mode	Major	Selects error severity for a CRC fault.
36	PF Fault Mode	Minor	Selects error severity for a Peripheral Fault.
37	Power Fault Mode	Minor	Selects error severity for a Power Fault.
38	Module Change Fault Mode	Major	Selects error severity for a Module Change Fault.
39	Configuring Fault Mode	Major	Selects error severity for a Configuring Fault.
40	Connection Fault Mode	Major	Selects error severity for a Connection Fault.
41	Fault Cycles Fault Mode	Minor	Selects error severity for a Fault Cycles Fault.
42	COS Mask Index	0	Index of COS Mask Byte Value. Must be set to the desired byte number to access the COS Mask byte value.
43	COS Byte Value	0	This value is a mask to determine whether to send a COS message or not. Refer to Appendix A, Class Code 104 dec, 0x64 hex.
44	COS Add All Mode	16385	Determines what type of COS Mask will be generated during a COS Add All I/O. Refer to Appendix A, Class Code 104 dec, 0x68 hex.
45	COS Add All	0	Add all inputs defined in the COS “Add All Mode“.
46	COS Enable Inline Status Word	1	Causes a COS message to be generated on an Inline Status Word transition.
47	COS DIP Fault	0	Causes a COS message to be generated when a DIP fault changes.
48	COS DOP Fault	0	Causes a COS message to be generated when a DOP fault changes.
49	COS AIP Fault	0	Causes a COS message to be generated when a AIP fault changes.
50	COS AOP Fault	0	Causes a COS message to be generated when a AOP fault changes.

1436B008-2

Table B-1. EDS File Parameters (continued)

Param. No.	Parameter Name	Default Value	Parameter Description
51	COS Special Function Fault	0	Causes a COS message to be generated when a special function fault changes.
52	COS Enable DIP	1	Causes a COS message to be generated when any DIP changes state.
53	COS Enable AIP	0	Causes a COS message to be generated when any AIP changes state.
54	COS Enable Special Function	0	Causes a COS message to be generated when any special function changes state.
55	COS AIP Mask Value	65280	Mask value for analog inputs.
56	Inline Error History 1	0	Most Recent Error Entry
57	Inline Error History 2	0	Second Most Recent Error Entry
58	Inline Error History 3	0	Third Most Recent Error Entry
59	Inline Error History 4	0	Fourth Most Recent Error Entry
60	Inline Error History 5	0	Fifth Most Recent Error Entry
61	Inline Error History 6	0	Sixth Most Recent Error Entry
62	Inline Error History 7	0	Seventh Most Recent Error Entry
63	Inline Error History 8	0	Eighth Most Recent Error Entry
64	Inline Error History 9	0	Ninth Most Recent Error Entry
65	Inline Error History 10	0	Last Saved Error Entry

1436B008-3

APPENDIX C
Tips and Examples

Appendix C Contents

Tips and Examples

C.1 General

This section provides examples, tips and considerations that may be helpful when designing and implementing an Inline station.

C.2 Configuration Examples

C.2.1 Inline Segment Terminal with Four (4) Fused Digital Outputs

Figure C-1 shows a circuit diagram that uses a fused segment terminal (IB IL 24 SEG/F) and a digital output terminal with 4 outputs.

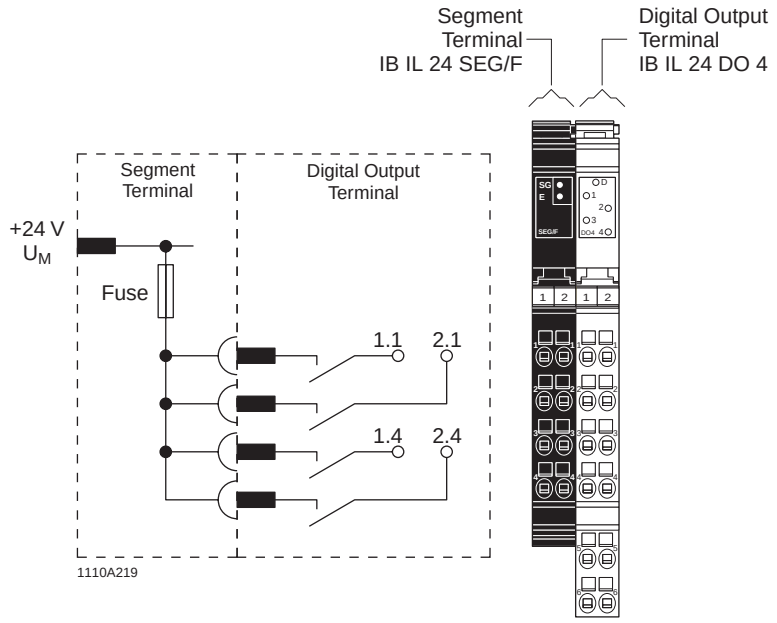


Figure C-1. Configuration Example for Four Fused Digital Outputs

C.2.2 Emergency Stop Circuitry with Two (2) Fused Outputs

Figure C-2 shows a diagram for establishing an emergency stop circuit for two fused outputs (U_S power). The E-Stop circuit consists of a Segment Terminal with fuse (IB IL 24 SEG/F), Safety Relay Terminal (IB IL 24 SAFE 1), and a Digital Output Terminal (IB IL 24 DO 2-2A) with two outputs.

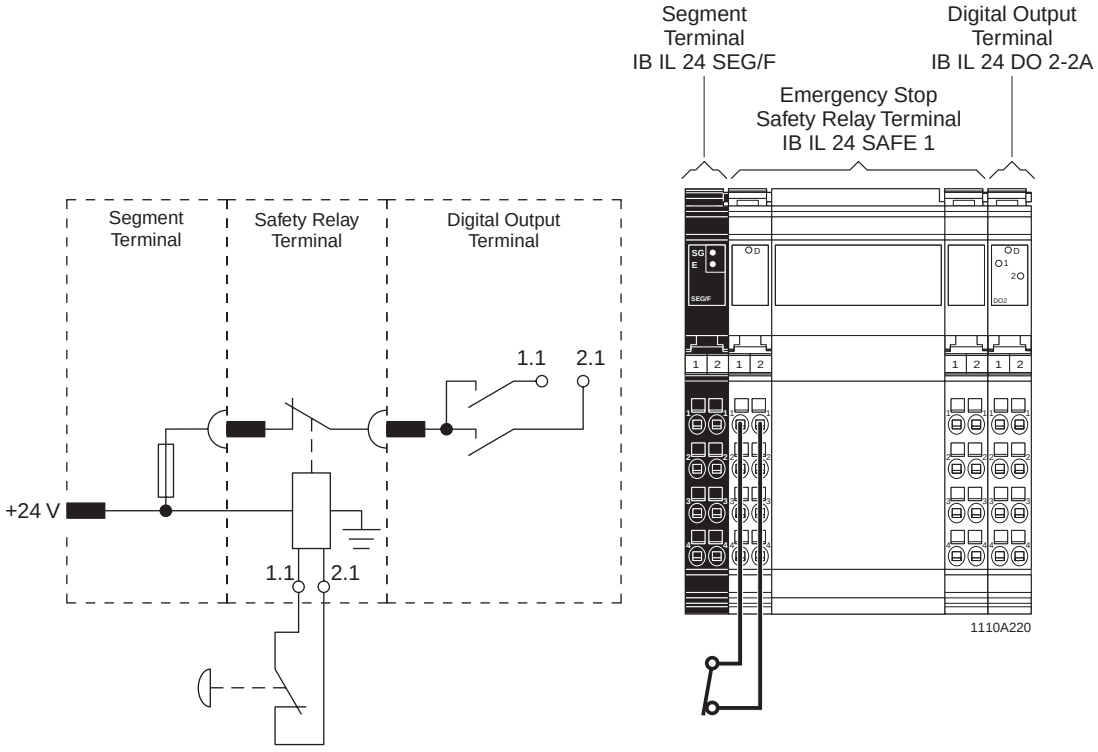


Figure C-2. Configuration Example for Emergency Stop Circuitry

C.2.3 Example Plant Layout

Figure C-3 shows an example plant layout controlled by a PC and using the Inline Family of products. A brief description of Inline products and their function is provided in the following paragraphs.

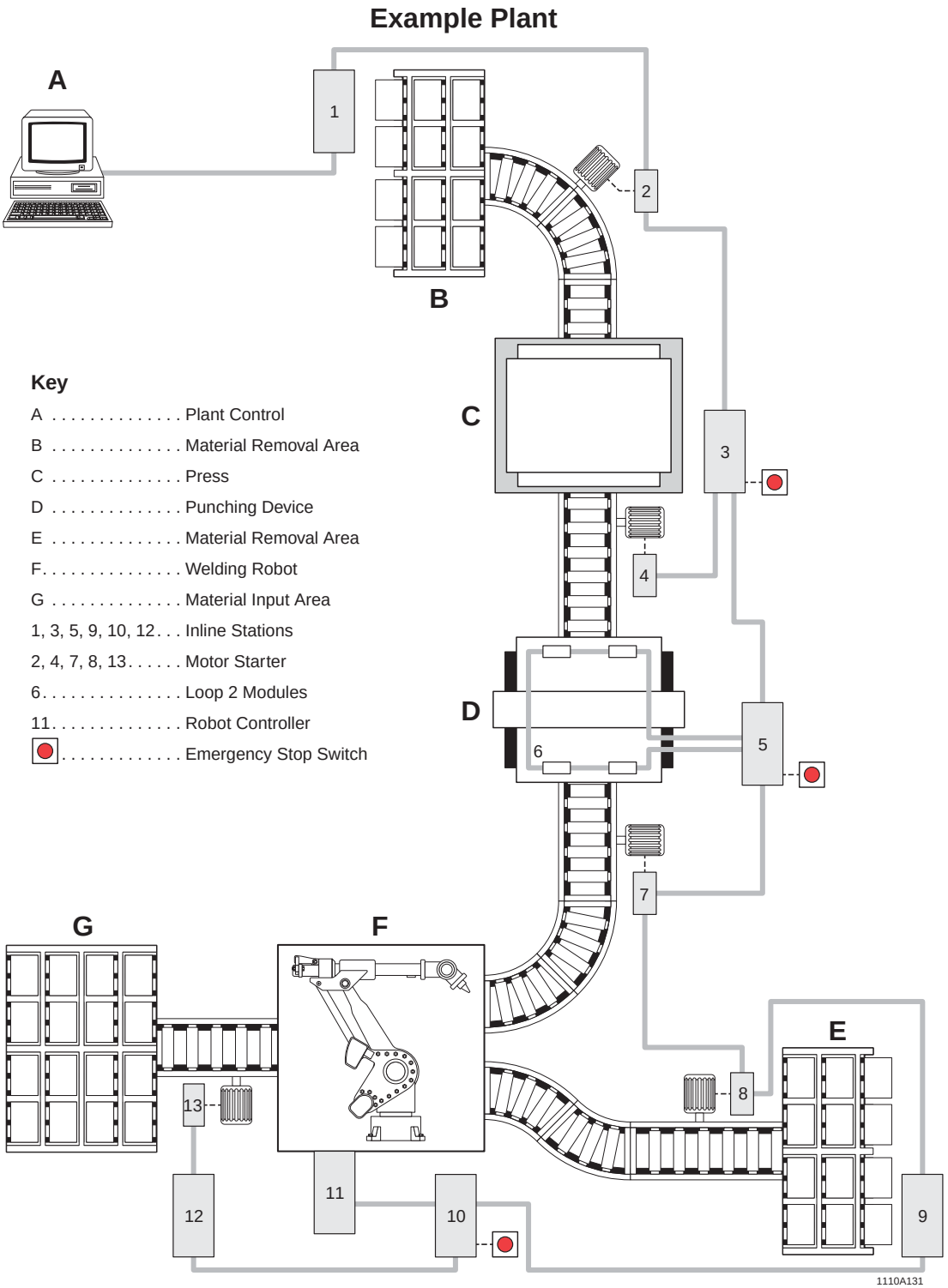


Figure C-3. Example Plant Layout Controlled by a PC and Using Inline

Tour of the Example Plant (see Figure C-3)

Inline station (1) modules control the removal of material from area "B".

Motor starter (2) is directly connected to the remote bus and controls the conveyor belt motor between area "B" and area "C".

Inline station (3) modules control the press at area "C". A remote bus branch from station (3) controls motor starter (4) which in turn controls the motor that drives the conveyor belt between area "C" and area "D". Because this conveyor must be protected, an emergency stop switch is installed at Inline station (3).

Inline station (5) modules control the punching operation located in area "D". A series of Loop 2 modules (6) connected to station (5) are used to monitor punch status. An emergency stop switch is required at station (5) to protect the punching device.

The conveyor belt between area "D" and area "F" is controlled by motor starter (7). The conveyor belt between area "F" and area "E" is controlled by motor starter (8).

Inline station (9) modules control the removal of material from area "E".

A robotic control system (11) is connected to and controlled by Inline station (10). An emergency stop switch has been installed at this location to protect the robot.

Inline station (12) modules control the storage of materials in the material input area "G".

Motor starter 13 is directly connected to the remote bus and controls the conveyor belt motor.

C.3 Tips on Working with Inline

C.3.1 Determining a Voltage Failure with Passive Power and Segment Terminals

Passive terminals cannot indicate a voltage failure over the bus. However, by using power and segment terminals that are fused, the red LED on these terminals, when lit, indicates a local voltage failure. We recommend monitoring segment voltage over the bus by using a digital input terminal.

C.3.2 Sequencing Terminals within an Inline Station as Related to Current Consumption

The sequence of the terminals within an Inline station should depend on the current consumption of the I/O through voltage jumpers U_M and U_S. Within a station, place the terminals with the highest current consumption first. This limits high supply current from having to flow through the entire U_M and U_S circuits. We recommend placing terminals within the station as follows:

- Digital output terminals with 8-slot housing
- Digital output terminals with 2-slot housing
- Digital input terminals with 8-slot housing
- Digital input terminals with 2-slot housing
- Special function modules in any order
- Analog terminals in any order

C.3.3 Special Consideration for Temperature Modules

High current flowing through the voltage jumpers U_M and U_S leads to a temperature rise of the voltage jumpers. This temperature rise of the jumpers leads to a temperature rise inside the terminal. Therefore, IB IL TEMP 2/UTH terminals should be placed at the end of the main circuit.

APPENDIX D
Appliction Note 1597A for Editing the COS Mask

Appendix D Contents

D.1 General

Application Note 1597A describes how to select an input trigger for generating a Change-of-State (COS) network scan. This application note is applicable to both DeviceNet™ IL DN BK2 and IL DN BK3 bus couplers.

Editing the Change-of-State (COS) Mask for Inline DeviceNet™ Bus Couplers IL DN BK 2 and IL DN BK 3

Application Note 1597A

April 2002

Background

The bus coupler receives data in a Change-of-State (COS) type scan. By default any transition of any bit for the Inline Status Word or any digital input point transition will generate a COS scan.

This application note will explain how to mask your produced data to set up the COS trigger on a per bit (event) basis. This information will allow the user to customize their COS trigger event(s) as opposed to allowing any Inline Status or digital input bit to generate a COS state.

Note

If an analog or fault event needs to generate a trigger for a COS, refer to the "COS Mask Object" in Appendix A of the Inline DeviceNet™ IL DN BK2 or IL DN BK3 manuals.

Introduction

The COS mask can be viewed or modified by sending explicit messages to the COS Mask Object (Class Code 104 dec, 0x68 hex) or by using the EDS file. If the EDS file is used, information in this application note can be adapted to accomplish the same results.

Figure 1 shows how the mask bit value coupled with an input transition may or may not generate a COS scan on the DeviceNet™ network. The actual mask bit value is the user selectable enable that determines whether or not the transition will generate the COS trigger.

To set the mask bit value Attributes 4 (COS Mask Index) and 5 (COS Mask Byte Value) will be used in this document.

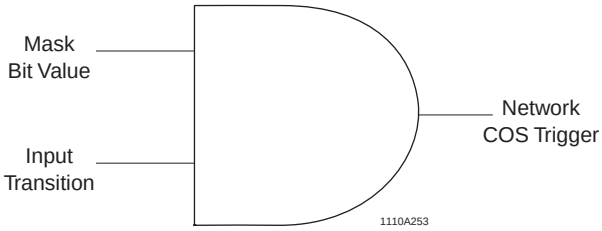


Figure 1. Logic Behind Generating a COS Event

Figure 2 shows the example system that will be discussed. This system has 3 bytes of produced data. 2 bytes for the Inline Status word and 1 byte for the (2) DI-4's. By default in this example all bits associated to the produced data will generated a COS scan.

This example will show how to set the 3rd bit of the 2nd DI-4 to be the only bit capable of generating a COS scan.

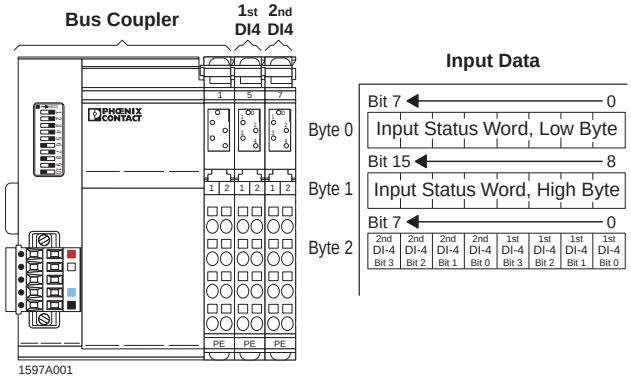


Figure 2. Example System

DeviceNet™ is a trademark of Open DeviceNet Vendor Association.
RSNetWorx™ is a trademark of Rockwell Software.y.

Phoenix Contact Inc. • P.O. Box 4100 • Harrisburg, PA 17111 • Phone: 1-800-586-5525 • FAX: (717) 944-5157

Getting Started

Figure 3 shows the Mask Index's and Mask Byte Value's that would be default for our example system shown in Figure 2. The Mask index works like a pointer that can be moved up or down to access the Mask Byte Value data that is located at the specified Mask Index. The following steps need to be taken to program the desired mask values.

Mask Index	Mask Byte Value	
	Binary	Hex
0	1111 1111	FF
1	1111 1111	FF
2	1111 1111	FF

1110A250

Figure 3. COS Mask Index 0, Default COS Mask Byte Value

Step 1. By default the Mask Index is pointing towards zero but to ensure this position a Set Single Attribute should be done by send the following explicit message:

Message 1. Set pointer to COS Mask Index 0

Service Code:	16 Set Single Attribute
Class Code:	0x68 (COS Mask Object)
Instance:	1
Attribute:	4 (COS Mask Index)
Data:	0 (word)

Message 2. Write 0x00 to the COS Mask Byte Value

This message will write 0x00 to the first byte of data that is the mask for the low byte of the Inline Status word. 0x00 will disable all of these bits from generating a COS scan as shown in figure 4. This can be done with the following explicit message:

Service Code:	16 Set Single Attribute
Class Code:	0x68 (COS Mask Object)
Instance:	1
Attribute:	5 (COS Mask Byte Value)
Data:	0 (word) (This value is 0xFF by default)

Mask Index	Mask Byte Value	
	Binary	Hex
0	0000 0000	00
1	1111 1111	FF
2	1111 1111	FF

1110A254

Figure 4. COS Mask Index 0, New COS Mask Byte Value

Step 2. Move the pointer to the second index and write 0x00 as the COS Mask Value by using the following 2 explicit messages. Results are shown in Figure 5.

Message 1. Set pointer to COS Mask Index 1 (being moved from 0).

Service Code:	16 Set Single Attribute
Class Code:	0x68 (COS Mask Object)
Instance:	1
Attribute:	4 (COS Mask Index)
Data:	1 (word).

Message 2. Write 0x00 to the COS Mask Byte Value (This byte is the mask for the upper byte of the Inline Status word.)

Service Code:	16 Set Single Attribute
Class Code:	0x68 (COS Mask Object)
Instance:	1
Attribute:	5 (COS Mask Byte Value)
Data:	0 (word) (This value is 0xFF by default)

Mask Index	Mask Byte Value	
	Binary	Hex
0	0000 0000	00
1	0000 0000	00
2	1111 1111	FF

1110A251

Figure 5. COS Mask Index 1, New Mask Byte Value

Step 3. Move the pointer to the second index and write 0x40 as the COS Mask Value by using the following 2 explicit messages. Results are shown in Figure 6. The 0x40 will enable the 3rd bit of the second DI-4 to be the only bit enabled to trigger a COS scan on the DeviceNet™ network.

Message 1. Set pointer to COS Mask Index 2 (being moved from 1).

Service Code:	16 Set Single Attribute
Class Code:	0x68 (COS Mask Object)
Instance:	1
Attribute:	4 (COS Mask Index)
Data:	2 (word).

Message 2. Write 0x40 to the COS Mask Byte Value (This byte is the mask for the 3rd byte of produced data occupied by the 1st and 2nd DI-4 modules.)

Service Code:	16 Set Single Attribute
Class Code:	0x68 (COS Mask Object)
Instance:	1
Attribute:	5 (COS Mask Byte Value)
Data:	0x40h (word) (This value is 0xFF by default)

Mask Index	Mask Byte Value	
	Binary	Hex
0	0000 0000	00
1	0000 0000	00
→ 2	0100 0000	40

1110A252

Figure 6. COS Mask Index 2, New COS Mask Byte Value

Headquarters, U.S.

P.O. Box 4100
Harrisburg, PA 17111-0100
Phone: 1-800-888-7388
In Canada: (905) 890-2820
Web Site: www.phoenixcon.com

Automation Systems Group

Phone: 1-800-586-5525
FAX: (717) 944-5157
Documentation FAX: 1-800-944-9901

APPENDIX E
Terms, Definitions, Symbols and Conversion Tables

Appendix E Contents

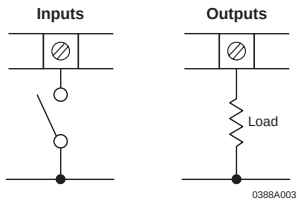
Terms, Definitions, Symbols and Conversion Tables

E.1 General

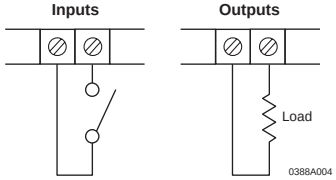
This section provides the meanings for many of the terms, abbreviations, acronyms and symbols found in both Inline and DeviceNet™ documentation. Some of the words phrases, acronyms and symbols are of international origin. All entries are listed in alphabetical order.

E.2 Wire Termination Methods for I/O Devices

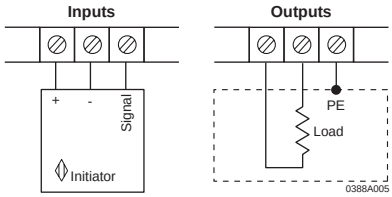
1-Wire Termination: Wire termination method, for I/O devices, has only one termination connection per point. Additional termination connections are required.



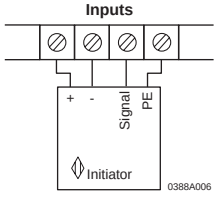
2-Wire Termination: Wire termination method, for I/O devices, has two termination connections per point.



3-Wire Termination: Wire termination method, for I/O devices, has three termination connections per point.



4-Wire Termination: Wire termination method, for input devices, has four termination connections per point.



E.3 Terms and Definitions

Actuator: A device that influences the behavior of a process and thereby causes a change in the process variables. Example actuators include switches, lamps, relays, etc.

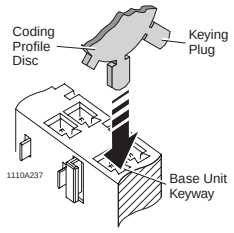
Analog Input (AI): An I/O device that will read an analog signal, convert it to a digital representation then transmit it back to the scanner.

Analog Output (AO): An I/O device that will generate an analog signal from a digital representation that was transmitted from the scanner.

Bus Terminal (BK) - DeviceNet™: The first terminal of an I/O station, the bus terminal is an interface between the I/O station and DeviceNet™. It provides bus signal conditioning and distributes supplied power to connected I/O modules.

Canadian Standard Association (CSA): Canadian certifying agency, in charge of conducting tests on device safety.

Coding & Keying of Connectors: By coding and keying the pluggable connectors and their bases ensures proper mating and electrical performance.



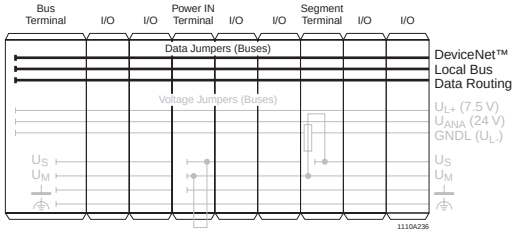
Controller Board - Host: The host controller board connects Programmable Logic Controllers (PLCs) or computer systems (PCs, VME, etc.) to the DeviceNet™ system.

Counter (CNT): Type of I/O module for counting or generating high-speed pulses.

Cyclic Redundancy Check (CRC): A test method for ensuring faithful communications.

Cycle Time: The time that the DeviceNet™ system needs to read all data from the connected devices and to write data to connected devices.

Data Routing: (1) Data signals from the host controller are transmitted along the station through data jumpers (buses). The jumpers for each module are automatically created when the module is snapped into place on the DIN-rail.



(2) General term for devices with different functions and application purposes that participate in data exchange. These include controller boards, interface boards, BK terminals, I/O modules, high-tech controllers drive controllers, valve manifolds, encoders, ID systems, operating and display units, etc.

Diagnostic Indicators: Red and green LEDs on a terminal used for detection and isolation of a malfunction as well as determining the state of terminal and network operations. Refer to the terminal-specific data sheet for diagnostic indicators applicable to that terminal. *Also see; Status Indicators.*

Digital Input (DI): An I/O device that will read digital signals, then transmits them back to the scanner.

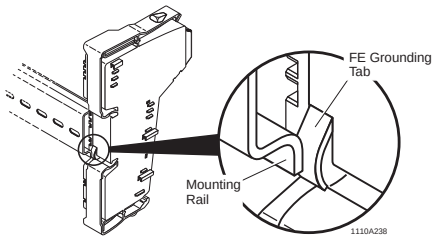
Digital Input/Output (DIO): An I/O device that will read and generate digital signals to and from the scanner.

Digital Output (DO): An I/O device that will generate digital signals transmitted from the scanner.

Electrical Isolation: Circuits of an electrical device are galvanically separated from each other.

FE: See Grounding

FE Spring: A spring-clip on the back of a module that forms a ground connection between the module and DIN-Rail.



View Looking at Back of Module

GND: *See Grounding*

GNDL: *See Grounding*

Grounding: DeviceNet™ and Inline use a variety of different grounding schemes to obtain maximum efficiency, reduce noise and most importantly, protect the user. The following paragraphs describe each of the various grounding schemes used in this documentation are defined in the following paragraphs. The IEC graphic symbol representing each grounding method is also provided.

a. Earth Ground (General Symbol)

IEC 617
02-15-01
1388A003

b. FE (Functional Earth Ground): This ground is noiseless and is used for ground cable shields and to suppress noise and interference voltages.

IEC 617
02-15-02
1388A004

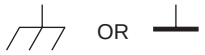
The Functional Earth ground is a low impedance path between electrical circuits and earth such as noise immunity improvement (EN 61131).

This ground connection must be separated from parts with hazardous voltages by double or reinforced isolation (EN 60950)

c. PE (Protective Earth Ground): This ground is used to protect human beings from shock hazards. It is also used for grounding devices.

IEC 617
02-15-03
1388A005

d. GND (Ground); 0 Volts or Chassis: This ground refers to common voltage return lines and is usually isolated from FE and PE.



IEC 617
02-15-04
1388A006

This ground is used primarily for the main and segment circuit voltage jumpers (buses). The total current of the main circuit must not exceed 8 A. Likewise, The total current of the segment circuit must not exceed 8 A.

e. GNDL (Ground): This ground circuit is used with both the logic supply and the I/O supply for analog modules. *See U_L (Voltage Logic) and U_{ANA} (Voltage Analog).*

ID Cycle: A process used to identify all I/O devices. Information returned includes: module count, module ID numbers and quantity, and the location of system words.

ID Number: A numeric value between 1 and 255 that all I/O devices contain. This number is used to inform the host controller of its device classification.

Input: The applied force or signal which drives a circuit or device

Input/Output (I/O) Device: A generic term for a device that gathers (inputs) or generates (outputs) control signals.

IP20: An IEC environmental specification defined as “protection against solid objects greater than 12 mm.” Closely matches the NEMA 1 (ventilated) general purpose specification.

IP65: An IEC environmental specification defined as "dust-tight and protection against water jets." Closely matches the NEMA 12 in-dust use, dust-tight and drip-tight specification.

IP67: An IEC environmental specification defined as “dust-tight and protected against water entry at one meter immersion.” Closely matches the NEMA 4 dust-tight and watertight specification.

Light Wave Link (LWL): A designation for any I/O device that uses fiber optic communications at the electrical level.

Network: A communications link that connects devices together. The link operates under a protocol understood by all devices.

Node: Refers to a station on the network.

NPN: Defines the direction of current flow in an I/O device. NPN is also referred to as Sink *(see Sink)*.

PE: See Grounding

PNP: Defines the direction of current flow for an I/O device. PNP is also referred to as Source *(see Source)*.

Power Terminal: Power inputted to this terminal is distributed through the station's internal main voltage jumper.
See U_M (Voltage Main).

Resistance Temperature Device (RTD): A type of input device used to measure temperature.

RS-232: The standard serial communications for computer peripherals.

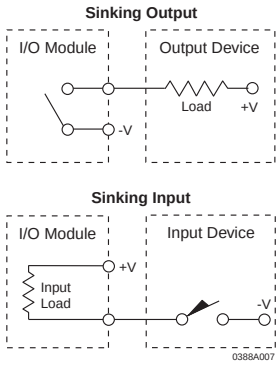
RS-422: The standard differential serial communications for computer peripherals.

Scan Cycle: A process used to read and write to I/O devices.

Scan Cycle Time: Term used to define the time required to read all input devices and write to all output devices.

Segment Terminal: Used to create a segment circuit. *See U_S (Voltage Segment).*

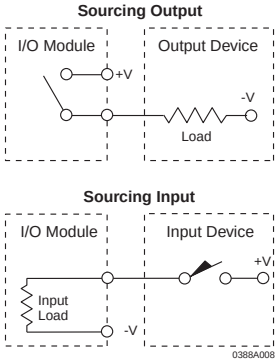
Sink: Defines the direction of current flow for an I/O device. A sink point will have its common side connected to positive voltage (+V). The signal line will go active when driven negative with respect to the +V. Sink is also referred to as NPN.



Software Driver (SWD): A floppy disk containing programs or drivers applicable to a specific product.

Software Tools (SWT): A floppy disk containing utility programs for a specific product or product group.

Source: Defines the direction of current flow for an I/O device. A source point will have its common connected to negative voltage (return). The signal line will go active when driven positive with respect to the return. Source is also referred to as PNP.



Special Function Module: A module that is dedicated to a specific function such as counter module, incremental encoder module, etc.

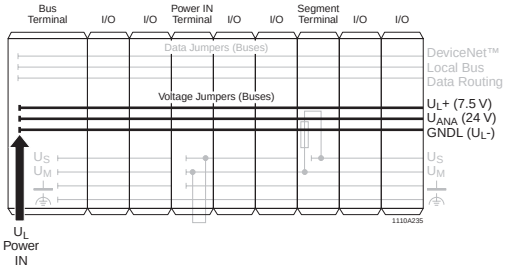
Status Indicators: A panel of yellow LEDs on an I/O module that indicate the state of various inputs or outputs of the module and to clearly denote the location of errors. Refer to the module-specific data sheet for status indicators applicable that module. *Also see; Diagnostic Indicators.*

Transmission Medium: In addition to the standard transmission cables made of copper, DeviceNet™ can also transmit signals using other media such as fiber optics, slip rings, and infrared.

Transmission Time: The interval between the start of data being transmitted by one function unit and the end of this data being received by another.

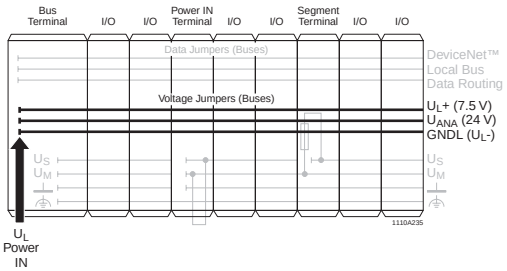
Trunk Line: Main network cable providing connection to nodes through the use of drop links.

U_{ANA} (Voltage Analog): The analog circuit provides power for analog modules. The analog circuit starts at the bus terminal and travels through all modules in the station.



The analog circuit operates at 24 volts and has a maximum current rating of 0.5 amps. Once this limit is reached, a new station must be created.

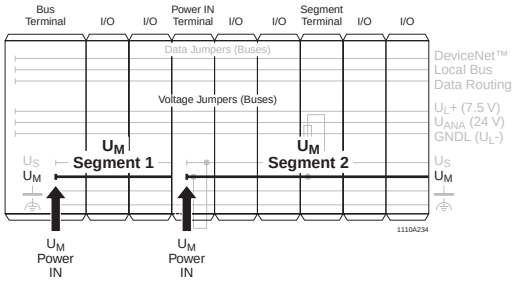
U_L (Voltage Logic): The logic circuit provides communications power to all modules. The logic circuit starts at the bus terminal and travels through all modules in the station.



The logic circuit operates at 7.5 volts and has a maximum current rating of 2 amps. Once this limit is reached, a new station must be created. This circuit is not electrically isolated from the 24 volt input voltage of the bus terminal.

GNDL is the ground circuit for the logic supply and the I/O supply for analog modules

U_M (Voltage Main): The main circuit supplies power to initiators that need individual short-circuit protection, or to initiators that do not need short circuit protection. The main circuit starts at the bus terminal or at a power terminal and travels through all subsequent modules until it reaches the next power terminal, at which point the circuit is interrupted. The next power terminal starts a new circuit that is electrically isolated from the previous circuit.



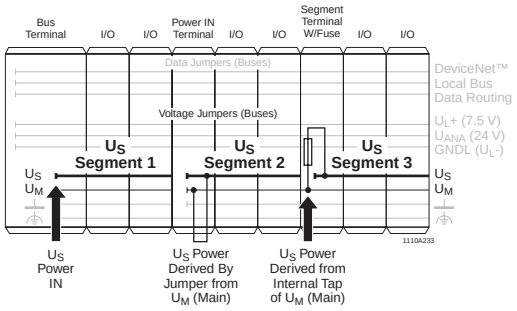
Power to the main circuit (U_M) is supplied through the bus terminal or through a power terminal. In turn, the main circuit supplies power to those terminals that need to be isolated from the I/O voltage such as an emergency stop. Several independent circuits or segments can be created within the station.

The maximum voltage in the main circuit is 250 Volts and the maximum current capacity of the main circuit is 8 amps.

Underwriters Laboratories (UL): U.S. certifying agency, in charge of conducting tests on device safety.

Universal Thermocouple (UTH): An I/O module for reading any type of thermocouple device.

U_S (Voltage Segment): The segment circuit supplies I/O power to all terminals or to groups of terminals in a station. U_S power starts at the bus or supply terminal (power or segment) and travels through the U_S voltage jumper (bus) until it reaches the last terminal in the station or another supply terminal, where voltage is interrupted. This allows multiple supply terminals to be used in order to create different voltages in different segments of the station. For example; 24 V dc and 230 V ac.

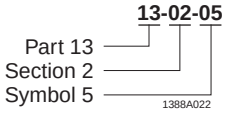


Segment circuits are also used to create separate current circuits such as an emergency stop circuit.

E.4 IEC (International Electrotechnical Commission) Graphic Symbols for Diagrams

This section shows those symbols used in this document. Most of the symbols shown have an IEC-617 serial number. The serial number is composed of three groups as described below. Those symbols that do not have an IEC serial number have been created from combining various aspect of more than one symbol to pictorially explain a component or a function.

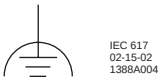
1. The first two digits indicate the "Part" number of the IEC-617 publication.
2. The second two digits identify the "Section" number within a given part.
3. The third two digits (01 to 99) indicate the specific symbol.



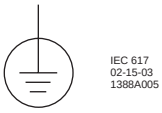
A. IEC Part 2 — Symbol elements, qualifying symbols and other symbols having general applications.

Earth Ground:

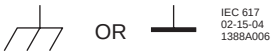
Protective Earth Ground:



Functional Earth Ground (FE):



Frame Ground:



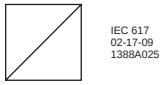
Ideal Current Source:

Ideal Voltage Source:

Analog Signals:

Digital Signals:

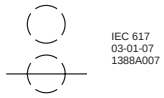
Converter, General



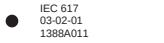
B. IEC Part 3 — Conductors and connecting devices

Multiple Conductors:

Shielded Conductor(s):

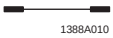


Junction Connection:



Contact/Terminal:

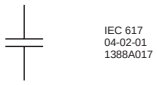
Voltage Jumper (Bus):



Resistor:

C. IEC Part 4 — Basic passive components

Capacitor:

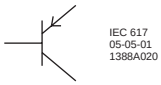


Diode:

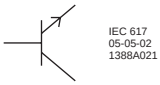
D. IEC Part 5 — Semiconductors and electron tubes

Light Emitting Diode:

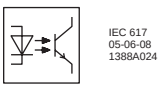
PNP Transistor:



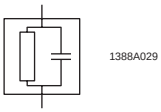
NPN Transistor:



Optocoupler:



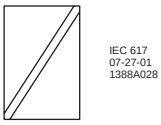
Coupling Circuit, Internal:



E. IEC Part 7 — Switchgear, controlgear, and protective devices

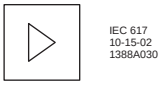
Fuse:

Coupling Device with Optical Isolation:



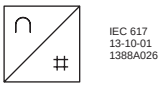
F. IEC Part 10 — Telecommunications: Transmission

Amplifier, General Symbol:

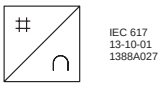


G. IEC Part 13 — Analog elements

Converter, Analog to Digital:



Converter, Digital to Analog:



E.5 Metric to US Standards Conversion Tables

A. Metric to English Conversion Formulas

Measurement	English Standard to Metric Units		Metric to English Standard Units	
Length	1 in.	25.40 mm	1 mm	0.03937 in.
	1 ft.	30.48 cm	1 cm	0.39370 in.
	1 yd	0.91 m	1 m	3.28084 ft
	1 miy	1.61 km	1 km	0.62137 mi
Temperature	$°C = \frac{5}{9}(°F - 32)$		$°F = \frac{9 \times °C}{5} + 32$	
Power	1 hp = 0.746 kW		1 kW = 1.34 hp	

1110A231

B. Temperature Conversion Table

°C	°F	°C	°F	°C	°F	°C	°F	°C	°F
95	203.0	65	149.0	35	95.0	5	41.0	-25	-13.0
90	194.0	60	140.0	30	86.0	0	32.0	-30	-22.0
85	185.0	55	131.0	25	77.0	-5	23.0	-35	-31.0
80	176.0	50	122.0	20	68.0	-10	14.0	-40	-40.0
75	167.0	45	113.0	15	59.0	-15	5.0	-45	-49.0
70	158.0	40	104.0	10	50.0	-20	-4.0	-50	-58.0

1110A232

C. Conductor Cross Section Conversion Table from mm² to AWG (American Wire Gauge)

mm²	0.13	0.22	0.25	0.5	0.75	1	1.5	2.5	4	6	10	16	25	35	50
AWG	26	24	24	20	18	18	16	14	12	10	8	6	4	2	1/0

1110A239

D. Millimeter to Inch Conversion Table

mm	in.	mm	in.	mm	in.	mm	in.	mm	in.	mm	in.
1	0.039	26	1.024	51	2.008	76	2.992	101	3.976	126	4.961
2	0.079	27	1.063	52	2.047	77	3.031	102	4.016	127	5.000
3	0.118	28	1.102	53	2.087	78	3.070	103	4.056	128	5.039
4	0.157	29	1.142	54	2.126	79	3.110	104	4.095	129	5.078
5	0.197	30	1.181	55	2.165	80	3.150	105	4.134	130	5.118
6	0.236	31	1.220	56	2.205	81	3.189	106	4.173	131	5.157
7	0.276	32	1.260	57	2.244	82	3.228	107	4.213	132	5.197
8	0.315	33	1.299	58	2.283	83	3.268	108	4.252	133	5.236
9	0.354	34	1.339	59	2.323	84	3.307	109	4.291	134	5.276
10	0.394	35	1.378	60	2.362	85	3.347	110	4.330	135	5.315
11	0.433	36	1.417	61	2.402	86	3.386	111	4.370	136	5.354
12	0.472	37	1.457	62	2.441	87	3.425	112	4.409	137	5.394
13	0.512	38	1.496	63	2.480	88	3.465	113	4.449	138	5.433
14	0.551	39	1.535	64	2.520	89	3.504	114	4.489	139	5.472
15	0.591	40	1.575	65	2.559	90	3.543	115	4.528	140	5.512
16	0.630	41	1.614	66	2.598	91	3.583	116	4.567	141	5.551
17	0.669	42	1.654	67	2.638	92	3.622	117	4.606	142	5.591
18	0.709	43	1.692	68	2.677	93	3.661	118	4.646	143	5.630
19	0.748	44	1.732	69	2.716	94	3.701	119	4.685	144	5.669
20	0.787	45	1.772	70	2.756	95	3.740	120	4.724	145	5.709
21	0.827	46	1.811	71	2.795	96	3.780	121	4.764	146	5.748
22	0.866	47	1.850	72	2.835	97	3.819	122	4.803	147	5.787
23	0.906	48	1.890	73	2.874	98	3.858	123	4.843	148	5.827
24	0.945	49	1.929	74	2.913	99	3.898	124	4.882	149	5.866
25	0.984	50	1.969	75	2.953	100	3.937	125	4.921	150	5.906

1110A240

Headquarters, U.S.

Phoenix Contact Inc.
P.O. Box 4100
Harrisburg, PA 17111-0100
Phone: 800-888-7388
717-944-1300
Fax: 717-944-1625
Email: info@phoenixcon.com
Web site: www.phoenixcon.com

Technical Service

Phone: 800-322-3225

Headquarters, Canada

Phoenix Contact Ltd.
235 Watline Avenue
Mississauga, Ontario L4Z 1P3
Phone: 905-890-2820
Fax: 905-890-0180

Technical Service

Phone: 800-890-2828

SCATTERGOOD & JOHNSON LTD
ELECTRICAL ENGINEERING & FLUID CONTROL DISTRIBUTORS
Est.1899

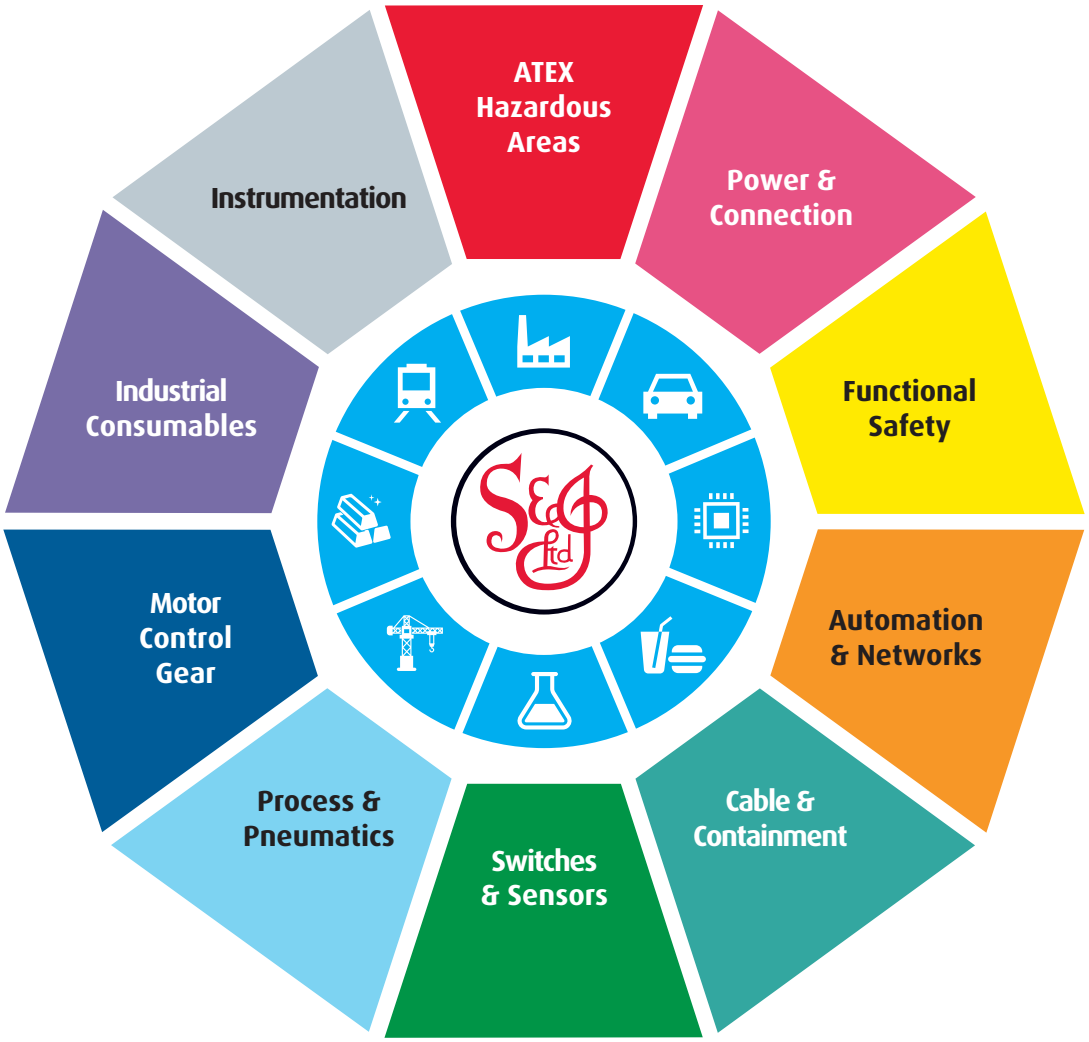
At Scattergood & Johnson Ltd, we pride ourselves on being a technical distributor to specialist industries.

Working with a range of quality product suppliers across a number of specialist markets, we are not your average 'box shifter' - we are your technical and supply chain partner.

We fully support every product we sell - for free! Our internal team and external sales engineers can answer any product or application question, no matter the complexity.

Backing up this technical ability is a range of 50,000+ products available from stock for nationwide next day delivery (same day if required!), or you can collect what you need from any of our trade counters around the UK.

Select your specialist interest below to learn more about how we can help.



Online, In Branch and On the Road - Scattergood & Johnson Ltd, there when you need us.

www.scatts.co.uk