PHOENIX CONTACT

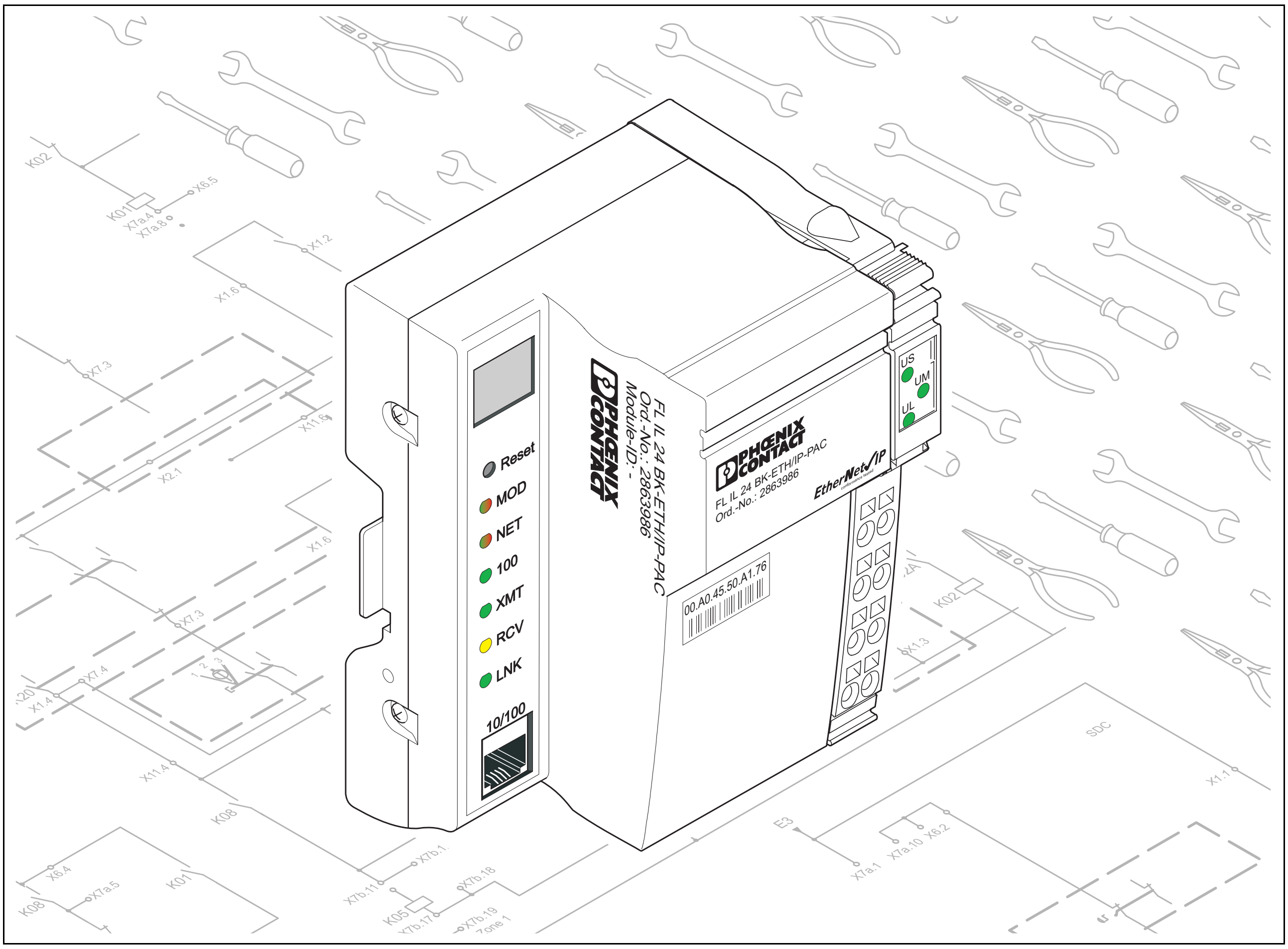
INSPIRING INNOVATIONS

User Manual

UM EN FL IL 24 BK ETH/IP-PAC

Order No.: 28 88 02 6

Inline Ethernet/IP Bus Coupler



User Manual
Inline Ethernet/IP Bus Coupler

Designation: UM EN FL IL 24 BK ETH/IP-PAC

Revision: 00

Order No.: 28 88 02 6

This manual is valid for:

Designation	Revision	Order No.
FL IL 24 BK ETH/IP-PAC	HW/FW 01/101	28 63 98 6

7183_en_00

Please Observe the Following Notes

In order to guarantee the safe use of the product described, please read this manual carefully. The following notes give you information on how to use this manual.

Qualifications of the User Group

The product usage described in this manual is exclusively aimed at electricians, or personnel trained by electricians, who are familiar with the valid national standards and other regulations on electrotechnology, in particular the pertinent safety concepts. Phoenix Contact assumes no liability for damage to any products resulting from disregard of information contained in this manual.

Explanation of Symbols Used

The *attention* symbol refers to an operating procedure which, if not carefully followed, could result in damage to equipment or personal injury.

The *note* symbol informs you of conditions that must strictly be observed to achieve error-free operation. It also gives you tips and advice on hardware and software optimization to save you extra work.

The *text* symbol refers you to detailed sources of information (manuals, data sheets, literature, etc.) on the subject matter, product, etc. This text also provides helpful information for the orientation in the manual.

We Are Interested in Your Opinion

We are constantly striving to improve the quality of our documents.

Should you have any suggestions or recommendations for improving the contents and layout of our documents, please send us your comments.

PHOENIX CONTACT GmbH & Co. KG
Documentation Services
32823 Blomberg
GERMANY

Phone +49 - (0) 52 35 - 3-00
Telefax +49 - (0) 52 35 - 3-4 20 21
E-Mail tecdoc@phoenixcontact.com

General Terms and Conditions of Use for Technical Documentation

Phoenix Contact GmbH & Co. KG reserves the right to alter, correct, and/or improve the technical documentation and the products described in the technical documentation at its own discretion and without giving any notice.

The receipt of technical documentation (in particular data sheets, installation instructions, manuals, etc.) does not constitute any further duty on the part of Phoenix Contact GmbH & Co. KG to furnish information on alterations to products and/or technical documentation. Any other agreement shall only apply if expressly confirmed in writing by Phoenix Contact GmbH & Co. KG. Please note that the supplied documentation is a product-specific documentation only, with no validity as general documentation.

Although Phoenix Contact GmbH & Co. KG makes every effort to ensure that the information content is accurate, up-to-date and state-of-the-art, technical inaccuracies and/or printing errors in the information cannot be ruled out. Phoenix Contact GmbH & Co. KG does not offer any guarantees as to the reliability, accuracy or completeness of the information appearing on the Website. Phoenix Contact GmbH & Co. KG accepts no liability or responsibility for errors or omissions in the content of the technical documentation (in particular data sheets, installation instructions, manuals, etc.).

As far as is permissible by applicable jurisdiction, no guarantee or claim for liability for defects whatsoever shall be granted in conjunction with the information available in the technical documentation, whether expressly mentioned or implied. This information does not include any guarantees on quality, does not describe any fair marketable quality and does not make any claims as to quality guarantees or guarantees on the suitability for a special purpose. Phoenix Contact GmbH & Co. KG reserves the right to alter, correct, and/or improve the information and the products described in the information at its own discretion and without giving any notice.

Internet

You will find current information on products from Phoenix Contact on the Internet at:

www.phoenixcontact.com.

All the latest Phoenix Contact documentation can be found on the Internet at:

www.download.phoenixcontact.com.

Table of Contents

Section 1

This section informs you about
– the basic structure of low-level signal modules
– the assignment and meaning of the diagnostic and status indicators
– potential and data routing
– housing dimensions and labeling options for the modules
– general information about the circuit diagrams for the module.

1 FL IL 24 BK ETH/IP-PAC

1.1 General Functions

1.1.1 Product Description

Ethernet/IP Inline bus coupler

Features

– Ethernet coupler for the Inline I/O system
– Ethernet TCP/IP
  - 10/100 Base-T(X)
  - Management via SNMP
  - Integrated web server
– Ethernet/IP protocol
– Up to 63 additional Inline modules can be connected
– Flexible installation system for Ethernet
– IP parameter setting via BootP, web-based management (WBM) or SNMP

Applications

– Connection of sensors/actuators via Ethernet/IP

Front view of the FL IL 24 BK ETH/IP-PAC

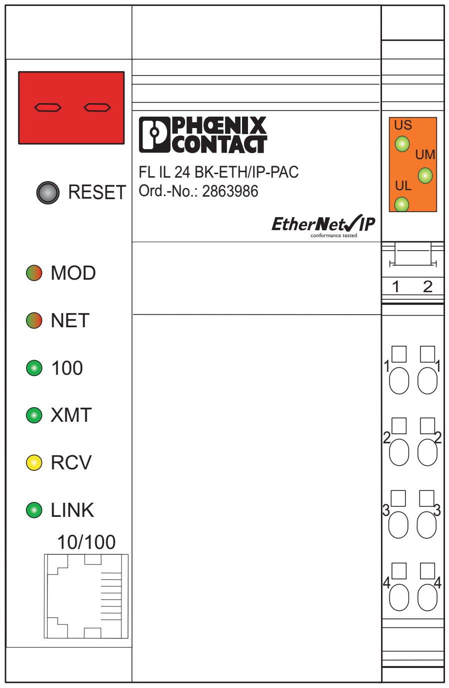


Figure 1-1 Front view of the FL IL 24 BK ETH/IP-PAC

1.2 Structure of the FL IL 24 BK ETH/IP-PAC Bus Coupler

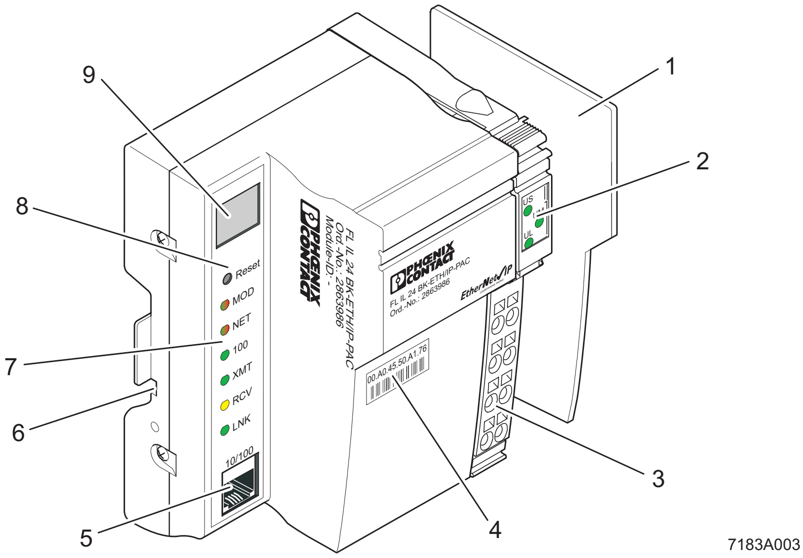


Figure 1-2 Structure of the FL IL 24 BK ETH/IP-PAC bus coupler

The bus coupler has the following components:

1 End plate to protect the last Inline module
2 Inline diagnostic indicators
3 24 V DC supply and functional earth ground connector
4 MAC address in clear text and as a barcode
5 Ethernet interface (twisted pair cables in RJ45 format)
6 Two FE contacts for grounding the bus coupler using a DIN rail (on the back of the module)
7 Ethernet status and diagnostic indicators
8 Reset button
9 7-segment display for the device status (Ethernet communication unit)

1.2.1 Local Status and Diagnostic Indicators

Table 1-1 Local status and diagnostic indicators

Des.	Color	Status	Meaning
Module Electronics			
UL	Green	ON	24 V supply, 7 V communications power/interface supply present
		OFF	24 V supply, 7 V communications power/interface supply not present
UM	Green	ON	24 V main circuit supply present
		OFF	24 V main circuit supply not present
US	Green	ON	24 V segment supply present
		OFF	24 V segment supply not present
Ethernet/IP Port			
MOD	Green/ Red	Green, ON	Module operating without errors
		Flashing, Green	Standby – module waiting for configuration
		Red, ON	A major error has occurred
		Flashing, Red	A recoverable error has occurred
		Flashing, Red/Green	Self test
		OFF	Module supply voltage not present
NET	Green/ Red	Green, ON	At least one connection is established
		Flashing, Green	IP parameter received, no connection established
		Red, ON	The own IP address is already in use by another device in the network
		Flashing, Red	At least one connection in the "Connection Timeout" status
		Flashing, Red/Green	Self test
		OFF	Module waiting for IP parameter assignment
100	Green	ON	Operation at 100 Mbps
		OFF	Operation at 10 Mbps (if LNK LED active)
		OFF	Operation in half duplex mode (if LNK LED active)
XMT	Green	ON	Data telegrams are being transmitted
		OFF	Data telegrams are not being transmitted
RCV	Yellow	ON	Data telegrams are being received (shows every kind of network traffic)
		OFF	Data telegrams are not being received
LNK	Green	ON	Physical network connection ready to operate
		OFF	Physical network connection interrupted or not present

Reset Button

The reset button is on the front plate. When the reset button is pressed the bus coupler is completely initialized and booted. Inline system outputs are reset and inputs are not read.

1.3 Connecting the Supply Voltage

The module is operated using a +24 V DC SELV.

Typical Connection of the Supply Voltage

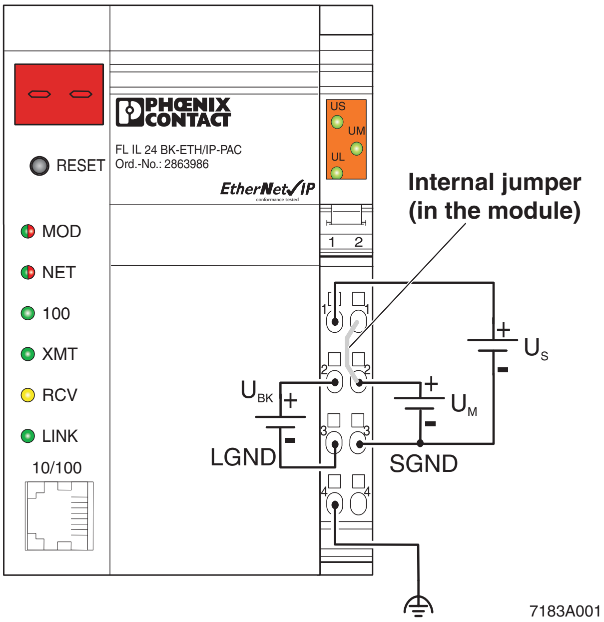


Figure 1-3 Typical connection of the supply voltage

1.4 Connector Assignment

Table 1-2 Connector assignment

Terminal Point/ Connector	Assignment/Power Connector		Wire Color/Remark
1.1	24 V DC (U_S)	24 V segment supply	The supplied voltage is directly led to the potential jumper.
1.2	24 V DC (U_{BK})	24 V supply	The communications power for the bus coupler and the connected local bus devices is generated from this power. The 24 V analog power (U_{ANA}) for the local bus devices is also generated.
2.1, 2.2	24 V DC (U_M)	Main voltage	The main voltage is routed to the local bus devices via the potential jumpers.

Table 1-2 Connector assignment

Terminal Point/ Connector	Assignment/Power Connector		Wire Color/Remark
1.3	LGND	Reference potential logic ground for U_{BK}	The potential is the reference ground for the communications power U_{BK}.
2.3	SGND	Reference potential for U_S and U_M	The reference potential is directly led to the potential jumper and is, at the same time, ground reference for the main and segment supply.
1.4, 2.4	FE	Functional earth ground (FE)	The functional earth ground must be connected to the 24 V DC supply/functional earth ground connection. The contacts are directly connected to the potential jumper and FE springs on the bottom of the housing. The terminal is grounded when it is snapped onto a grounded DIN rail. Functional earth ground is only used to discharge interference.

The GND potential jumper carries the total current from the main and segment circuits. The total current must not exceed the maximum current carrying capacity of the potential jumper (8 A). If the 8 A limit is reached at one of the potential jumpers U_S, U_M, and GND during configuration, a new power terminal must be used.

Functional earth ground must be connected through the 24 V DC supply/functional earth ground connection.

1.5 Supported Inline Modules

For a list of all I/O terminals permitted for use with the Ethernet/IP bus coupler, please refer to the application note AH IL BK IO LIST.

1.6 Basic Structure of Low-Level Signal Modules

Regardless of the function and the design width, an Inline low-level signal module consists of the electronics base (or base for short) and the plug-in connector (or connector for short).

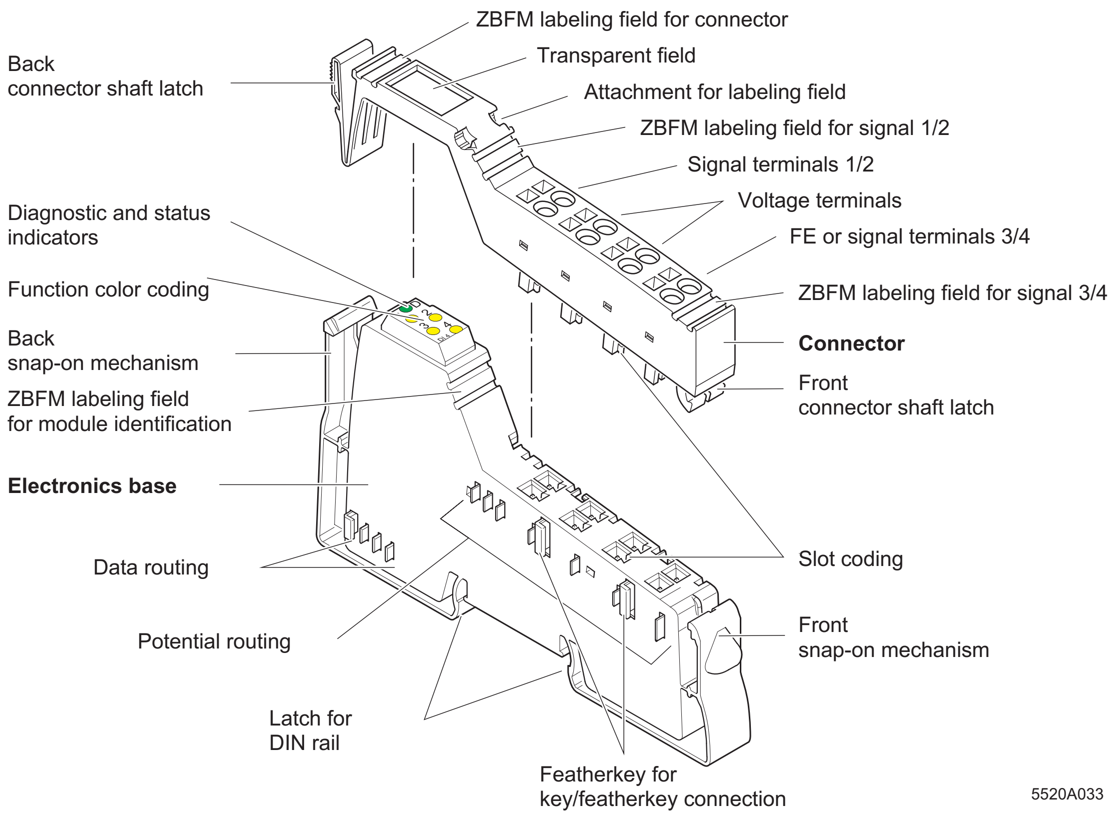


Figure 1-4 Basic structure of an Inline module

The most important components shown in Figure 1-4 are described in sections "Electronics Base" on page 1-10 and "Connectors" on page 1-10.

ZBFM: Zack marker strips, flat
(see also section "Function Identification and Labeling" on page 1-13)

The components required for labeling are listed in the Phoenix Contact "CLIPLINE" catalog.

1.6.1 Electronics Base

The electronics base holds the entire electronics for the Inline module and the potential and data routing.

Design widths

The electronics bases for low-level signal modules are available in a width of 8 terminal points (8-slot terminal) or 2 terminal points (2-slot terminal). Exceptions are combinations of these two basic terminal widths (see also section "Dimensions of Low-Level Signal Modules" on page 1-16).

1.6.2 Connectors

The I/O or supply voltages are connected using a pluggable connector.

Advantages

This pluggable connection offers the following advantages:

– Simple exchange of module electronics for servicing. There is no need to remove the wiring.
– Different connectors can be used on one electronics base, depending on your requirements.

Connector width

Regardless of the width of the electronics base, the connectors have a width of two terminal points. This means that you must plug 1 connector on a 2-slot base, 2 connectors on a 4-slot base, and 4 connectors on an 8-slot base.

Connector types

The following connector types are available:

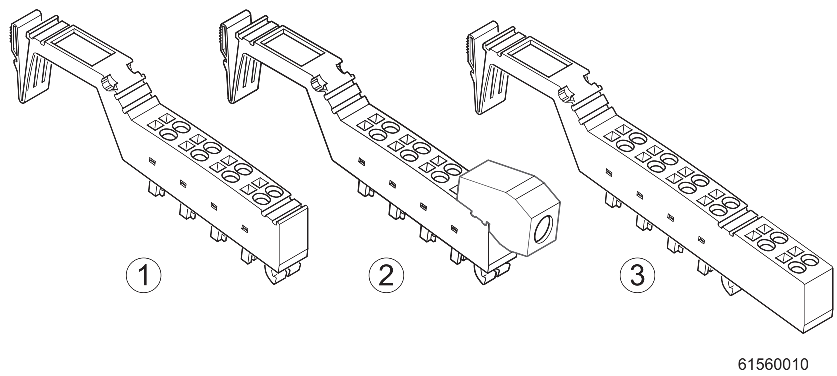


Figure 1-5 Inline connector types

1 Standard connector

The green standard connector is used for the connection of two signals in 4-wire technology (e.g., digital I/O signals).
The black standard connector is used for supply terminals. The adjacent contacts are jumpered internally (see Figure 1-6 on page 1-11).

2 Shield connector

This green connector is used for signals connected using shielded cables (e.g., analog I/O signals).
FE or shielding is connected via a shield connection clamp rather than via a terminal point.

3 Extended double signal connector

This green connector is used for the connection of four signals in 3-wire technology (e.g., digital I/O signals).

Connector identification

All connectors are offered with and without color print. The connectors with color print (marked with CP in the Order Designation) have terminal points that are color-coded according to their functions.

The following colors indicate the signals of the terminal points:

Table 1-3 Terminal point color-coding

Color	Terminal Point Signal
Red	+
Blue	-
Green/yellow	Functional earth ground

Internal structure of the connectors

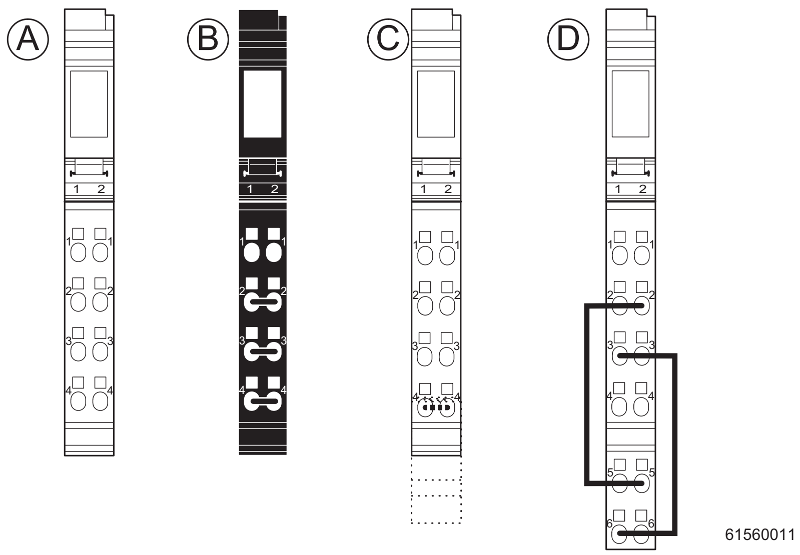


Figure 1-6 Internal structure of the connectors

A Green connector for I/O connection

B Black connector for supply terminals

C Shield connector for analog terminals

D Double signal connector for I/O connection

Jumpered terminal points already integrated in the connectors are shown in Figure 1-6.

The shield connector is jumpered through the shield connection. All other connectors are jumpered through terminal point connection.

To avoid a malfunction, only snap a connector on a terminal that is appropriate for this connection. Refer to the module-specific data sheet to select the correct connectors.

The black connector must not be placed on a module for which a double signal connector is to be used. Mixing this up leads to a short circuit between two signal terminal points (1.4 - 2.4).

Only snap black connectors onto a supply terminal.
When the terminal points are jumpered, power is carried through the jumpering in the connector and not through the printed circuit board of the module. The black connector must not be placed on a module for which a double signal connector is to be used. Incorrect connection may lead to a short circuit between two signal terminal points (1.4 - 2.4).

Connector keying

You can prevent mismatching of connectors by keying the base and the connector.

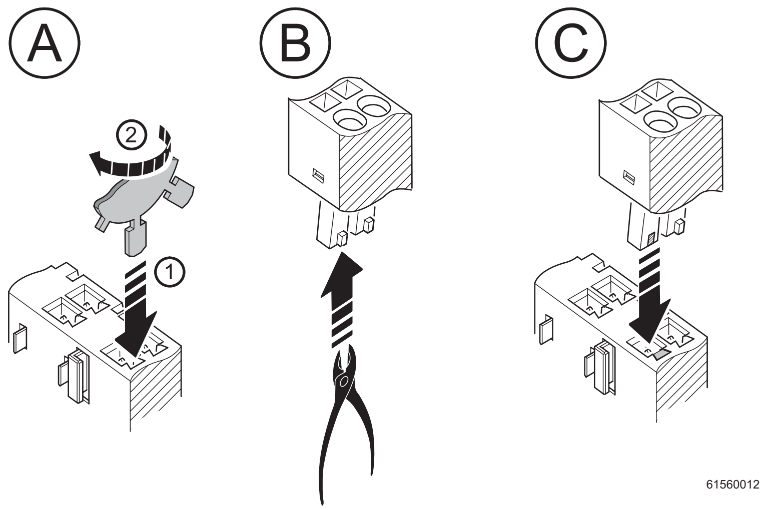


Figure 1-7 Connector keying

- Plug a keying profile (disc) into the keyway in the base (1) and turn it away from the small plate (2) (Figure 1-7, detail A).
- Use a diagonal cutter to cut off the keying tab from the connector (Figure 1-7, detail B).

Now, only the base and connector with the same keying will fit together (Figure 1-7, detail C).

1.7 Function Identification and Labeling

Function identification

The modules are color-coded to enable visual identification of the functions (1 in Figure 1-8).

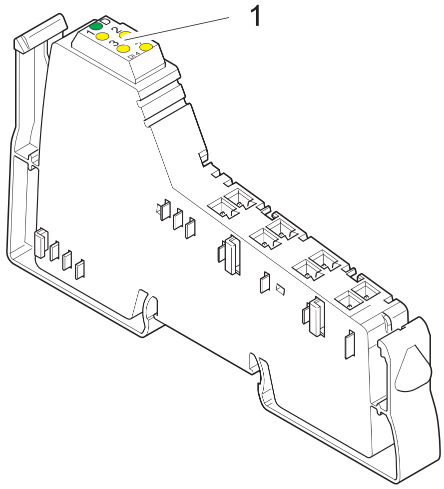


Figure 1-8 Function identification

The following colors indicate the functions:

Table 1-4 Module color-coding

Color	Function of the Module
Light blue	Digital input 24 V DC area
Pink	Digital output 24 V DC area
Blue	Digital input 120/230 V AC area
Red	Digital output 120/230 V AC area
Green	Analog input
Yellow	Analog output
Orange	Fieldbus coupler, special function modules
Black	Power terminal/segment terminal

Connector identification

The color-coding of the terminal points is described on page 1-11.

Labeling/terminal point numbering

Terminal point numbering is illustrated using the example of an 8-slot module.

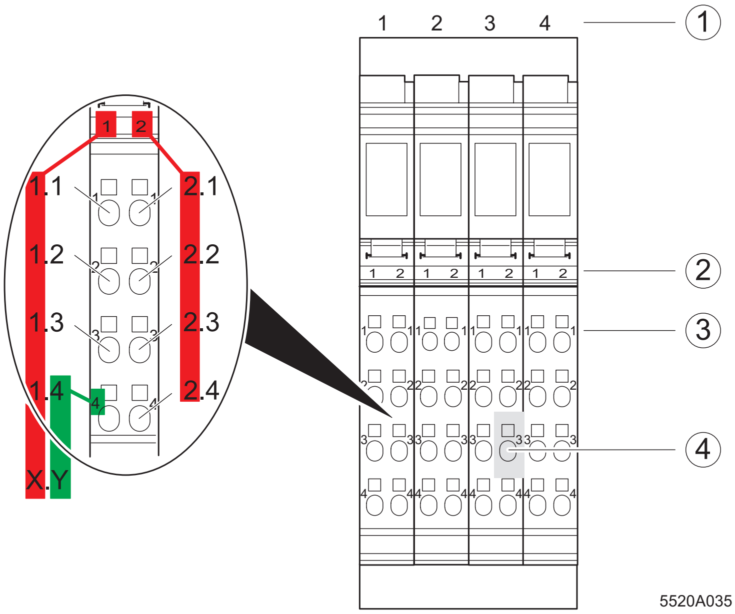


Figure 1-9 Terminal point numbering

Slot/connector

The slots (connectors) on a base are numbered consecutively (1 in Figure 1-9). This numbering is **not** shown on the actual module.

Terminal point

The terminal points on each connector are marked X.Y.

X is the number of the terminal point row on the connector. It is indicated above the terminal point row (2 in Figure 1-9).

Y is the terminal point number in a row. It is directly indicated on the terminal point (3 in Figure 1-9).

The precise designation for a connection point is thus specified by the slot and terminal point. The highlighted terminal point (4 in Figure 1-9) would be numbered as follows: slot 3, terminal point 2.3.

Additional labeling

In addition to this module marking, you can identify the slots, terminal points, and connections using marker strips and labeling fields.

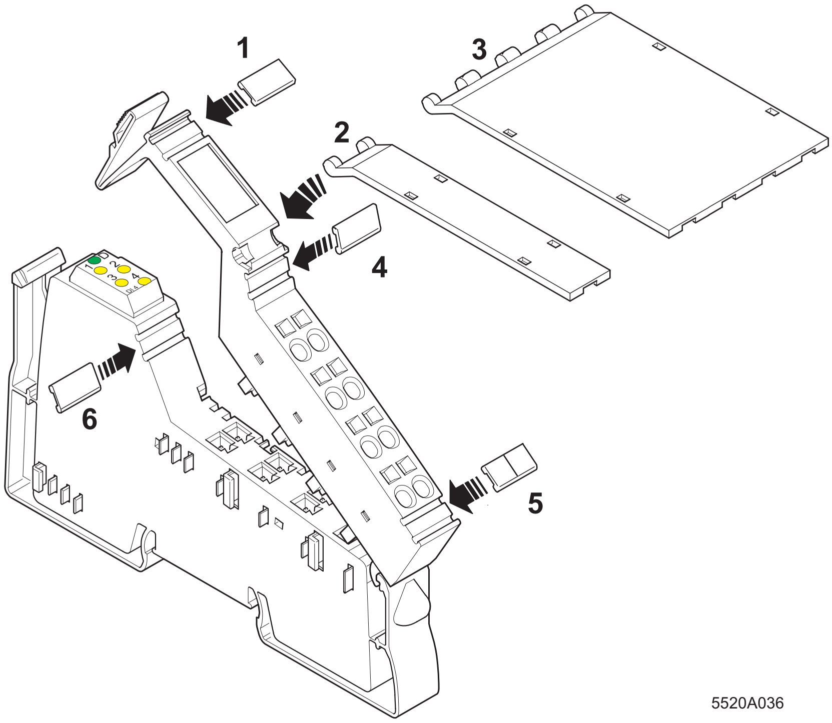


Figure 1-10 Labeling of modules

Various options are available for labeling the slots and terminal points:

1 Each connector can be labeled individually with Zack marker strips.

2 / 3 Another option is to use a large labeling field. This labeling field is available in two widths, either as a labeling field covering one connector (2) or as a labeling field covering four connectors (3). You can label each channel individually with free text. On the upper connector head there is a keyway for attaching this labeling field. The labeling field can be tilted up and down. In each end position there is a small latch which ensures that the labeling field remains in place.

4 / 5 Each signal can be labeled individually using Zack markers. On a double signal connector, the upper keyway (4) is designed for labeling signals 1/2 and the lower keyway (5) is for signals 3/4.

6 On the electronics base each slot can be labeled individually using Zack markers. These markers are covered when a connector is plugged in.

Using the markers on the connector and on the electronics base, you can clearly assign both connector and slot.

> The components required for labeling are listed in the Phoenix Contact "CLIPLINE" catalog.

1.8 Dimensions of Low-Level Signal Modules

Today, small I/O stations are frequently installed in 80 mm (3.150 in.) standard control boxes. Inline modules are designed so that they can be used in this type of control box. The housing dimensions of a module are determined by the dimensions of the electronics base and the dimensions of the connector.
The electronics bases of the low-level signal modules are available in three design widths (12.2 mm [0.480 in], 24.4 mm [0.961 in.] and 48.8 mm [1.921 in.]).
It accepts either one, two, or four 12.2 mm (0.480 in.) wide connectors.
When a connector is plugged in, each terminal depth is 71.5 mm (2.815 in.).
The height of the module depends on the connector used. The connectors are available in three different versions (see Figure 1-14).

2-slot housing

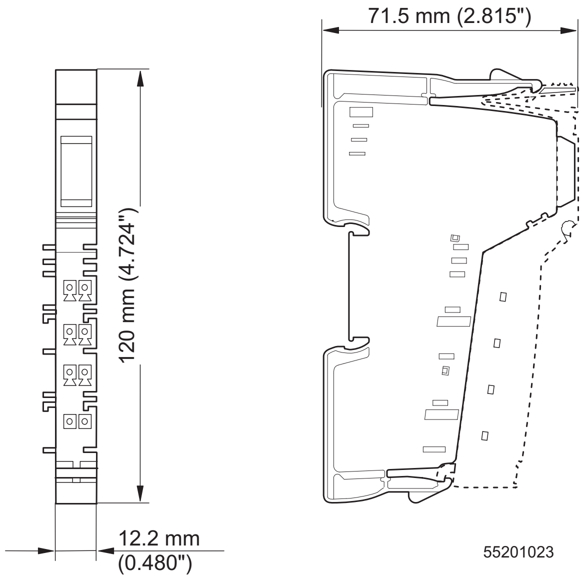


Figure 1-11 Dimensions of the electronics bases (2-slot housing)

4-slot housing

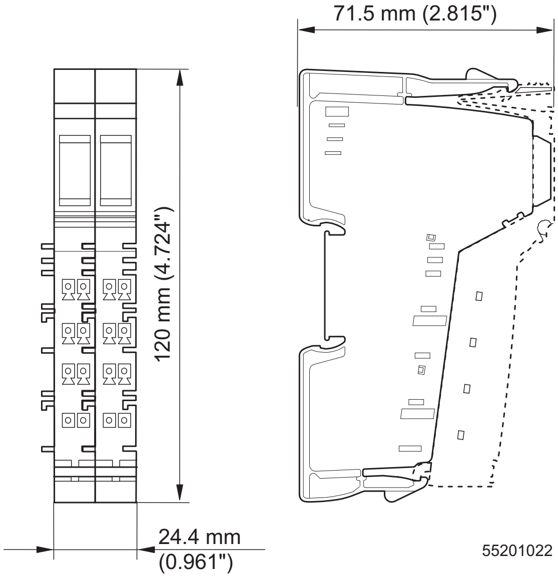


Figure 1-12 Dimensions of the electronics bases (4-slot housing)

8-slot housing

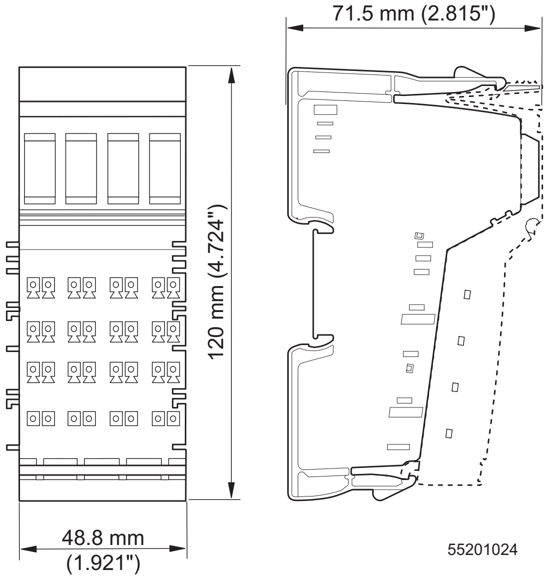


Figure 1-13 Dimensions of the electronics bases (8-slot housing)

Connector

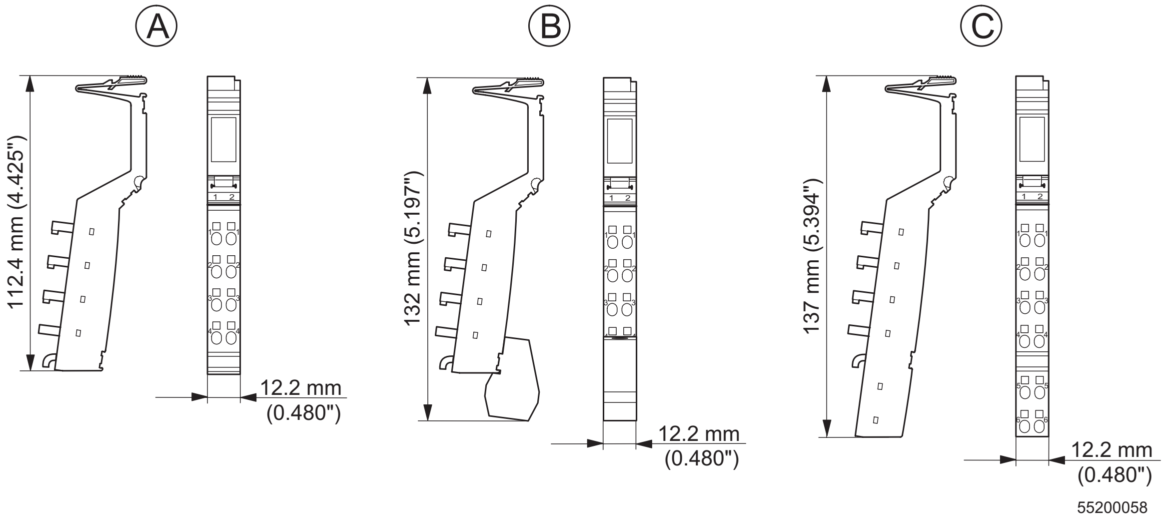


Figure 1-14 Connector dimensions

Key:

A	Standard connector
B	Shield connector
C	Extended double signal connector

The depth of the connector does not influence the overall depth of the module.

1.9 Electrical Potential and Data Routing

An important feature of the INTERBUS Inline and Ethernet/Inline bus coupler product ranges is their internal potential routing system. The electrical connection between the individual station devices is created automatically when the station is installed. When the individual station devices are connected, a power rail is created for the relevant circuit. It is created mechanically through the interlocking of knife and featherkey contacts on the adjacent modules.

A special segment circuit eliminates the need for additional external potential jumpering to neighboring modules.

Two independent circuits are created in one station: the logic circuit and the I/O circuit.

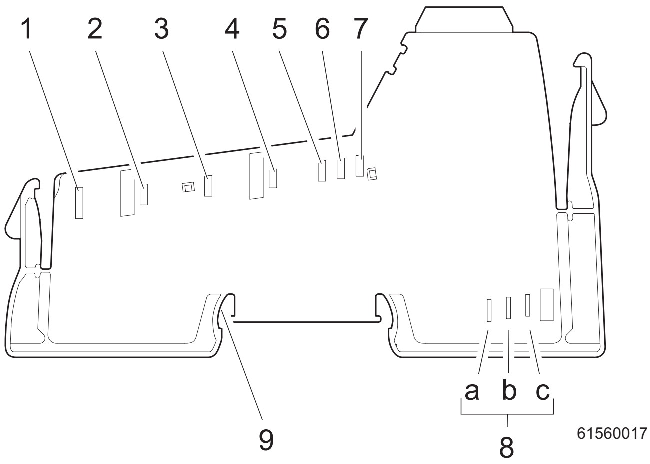


Figure 1-15 Potential and data routing

Table 1-5 Potential jumper (see Figure 1-15)

No.	Function		Meaning
1	FE	FE	Functional earth ground
2	SGND	SGND	Ground of segment supply and main supply
3	24 V	U_M	Supply for main circuit (with overload protection, if necessary)
4	24 V	U_S	Supply for segment circuit (with overload protection, if necessary) This jumper does not exist in the120/230 V AC power levels.
5	LGND	U_{L-}	Ground of communications power and I/O supply for analog modules
6	24 V	U_{ANA}	I/O supply for analog modules
7	7.5 V	U_{L+}	Supply for module electronics
(9)	FE spring		FE contact to DIN rail

The GND potential jumper carries the total current from the main and segment circuits. The total current must not exceed the maximum current carrying capacity of the potential jumper (8 A). If the 8 A limit is reached at one of the potential jumpers U_S, U_M, and GND during configuration, a new power terminal must be used.

The FE potential jumper must be connected via terminal point 1.4 or 2.4 at the Ethernet/IP bus coupler to a grounding terminal (see Figure 1-9). The FE potential jumper is led through all of the modules and connected via the FE spring to the grounded DIN rail of every supply terminal.

Table 1-6 Data jumper (see Figure 1-15)

No.	Function	Meaning
8a	DI1	Local bus signal (Data IN)
8b	DO1	Local bus signal (Data OUT)
8c	DCLK	Clock signal, local bus

1.10 Circuits Within an Inline Station and Provision of the Supply Voltages

There are several circuits within an Inline station. These are automatically set up when the modules have been properly installed. The voltages of the different circuits are supplied to the connected modules via the potential jumpers.

Please refer to the module-specific data sheet for the circuit to which the I/O circuit of a special module is to be connected.

Load capacity of the jumper contacts

Observe the maximum current carrying capacity of the jumper contacts on the side for each circuit. The load capacities for all potential jumpers are given in the following sections.

The arrangement of the potential jumpers can be found in section "Electrical Potential and Data Routing" on page 1-19.
For voltage connection, please refer to the notes given in the module-specific data sheets.

1.10.1 Supply of the Ethernet/IP Bus Coupler

The supply voltage U_{BK} and the segment voltage U_S **must** be connected to the Ethernet/IP bus coupler. From the supply voltage U_{BK}, the voltages for the logic circuit U_L (7.5 V) and the supply of the modules for analog signals U_{ANA} (24 V) are internally generated. The segment voltage is used to supply the sensors and actuators.

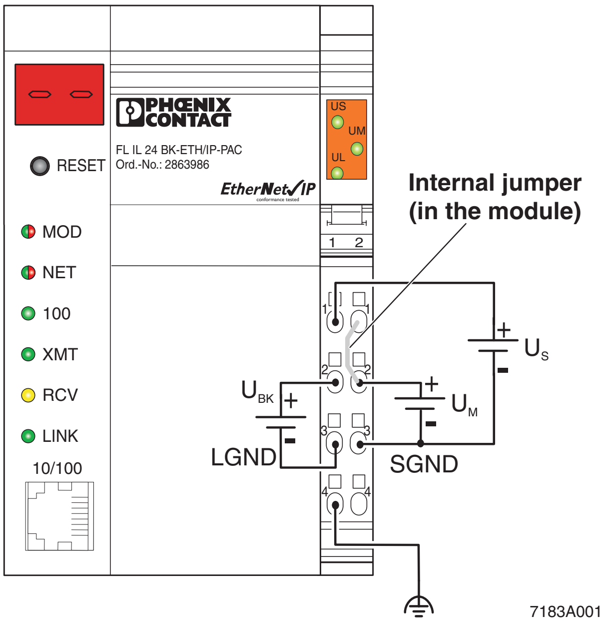


Figure 1-16 Typical connection of the supply voltage

1.10.2 Logic Circuit U_L

The logic circuit with communications power U_L starts at the bus coupler, is led through all modules of a station and cannot be supplied via another supply terminal.

Function

The logic circuit provides the communications power for all modules in the station.

Voltage

The voltage in this circuit is 7.5 V DC.

Generation of U_L

The communications power U_L is generated from the supply voltage U_{BK} of the bus coupler.
The communications power is not electrically isolated from the 24 V input voltage for the bus coupler.

Current carrying capacity

The maximum current carrying capacity of U_L is 2 A.

1.10.3 Analog Circuit U_{ANA}

The analog circuit with the supply for the analog modules (also referred to as analog voltage) U_{ANA} is supplied at the bus coupler and is led through all the modules in an Inline station. Power cannot be supplied by the supply terminals. U_{ANA} is not electrically isolated from U_{BK}.

Function The module I/O devices for analog signals are supplied from the analog circuit.

Voltage The voltage in this circuit is 24 V.

Generation of U_{ANA} The analog voltage U_{ANA} is generated from the main voltage U_{BK} of the bus coupler.

Current carrying capacity The maximum current carrying capacity of U_{ANA} is 0.5 A.

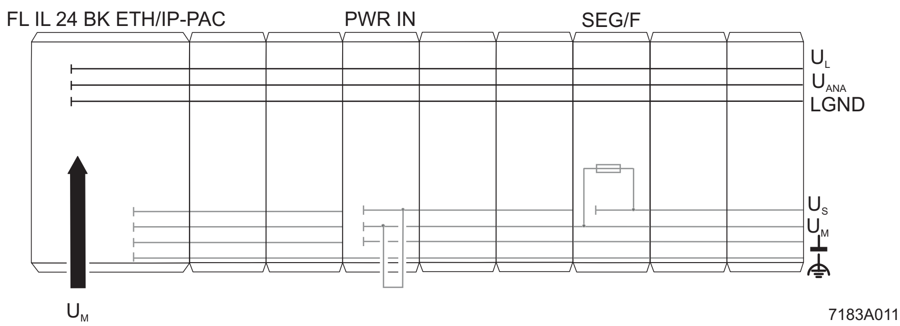


Figure 1-17 Logic and analog circuit

FL IL 24 BK ETH/IP-PAC	Ethernet/IP bus coupler
PWR IN	Power terminal
SEG/F	Segment terminal with fuse as an example of a segment terminal

1.10.4 Main Circuit U_M

The main circuit with the main voltage U_M starts at the bus coupler or a power terminal and is led through all subsequent modules until it reaches the next power terminal. A new circuit that is electrically isolated from the previous one begins at the next power terminal.

Several power terminals can be used within one station.

Function

Several independent segments can be created within the main circuit. The main circuit provides the main voltage for these segments. For example, a separate supply for the actuators can be provided in this way.

Voltage

> The maximum voltage in this circuit is 24 V DC. U_M can only be a maximum of 250 V AC when using special PWR-IN modules.

Current carrying capacity

The maximum current carrying capacity is 8 A (total current with the segment circuit). If the limit value of the common GND potential jumper for U_M and U_S is reached (total current of U_S and U_M), a new power terminal must be used.

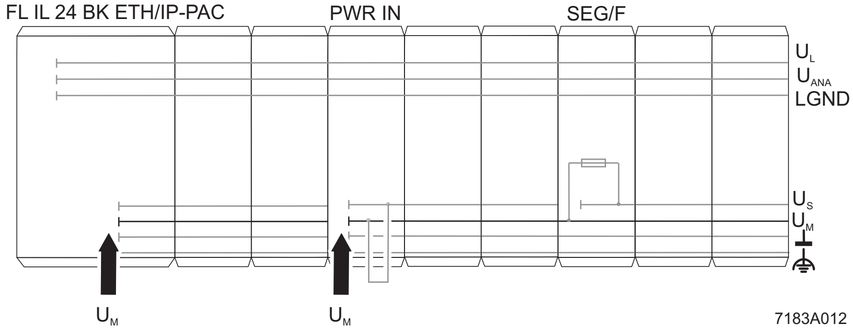


Figure 1-18 Main circuit

FL IL 24 BK ETH/IP-PAC	Ethernet/IP bus coupler
PWR IN	Power terminal
SEG/F	Segment terminal with fuse as an example of a segment terminal

Generation of U_M

In the simplest case, the main voltage U_M can be supplied at the bus coupler. In this case it is 24 V DC.

The main voltage U_M can also be supplied via a power terminal. A power terminal **must** be used if:

1. Different voltage areas (e.g., 120 V AC) are to be created.
2. Electrical isolation is to be created.
3. The maximum current carrying capacity of a potential jumper (U_M, U_S or GND, total current of U_S and U_M) is reached.

1.10.5 Segment Circuit

The segment circuit or auxiliary circuit with the segment voltage U_S starts at the Ethernet/IP bus coupler or a supply terminal (power terminal or segment terminal) and is led through all subsequent modules until it reaches the next supply terminal.

Function

You can use several segment terminals within a main circuit, and in this way segment the main circuit. It has the same reference ground as the main circuit. This means that circuits with different fuses can be created within the station without external cross wiring.

Voltage

The voltage in this circuit should not exceed 24 V DC.

Current carrying capacity

The current carrying capacity is 8 A, maximum (total current with the main circuit). If the limit value of the common potential jumper for U_M and/or U_S is reached (total current of U_S and U_M), a new power terminal must be used.

Generation of U_S

There are various ways of providing the segment voltage U_S:

1 You can supply the segment voltage at the Ethernet/Inline bus coupler or a power terminal.

2 You can tap the segment voltage from the main voltage at the Ethernet/Inline bus coupler or a power terminal using a jumper or a switch.

3 You can use a segment terminal with a fuse. Within this terminal the segment voltage is automatically tapped from the main power.

4 You can use a segment terminal without a fuse and tap the segment voltage from the main voltage using a jumper or a switch.

With 120 V/230 V AC voltage levels, segments cannot be created. In this case, only the main circuit is used.

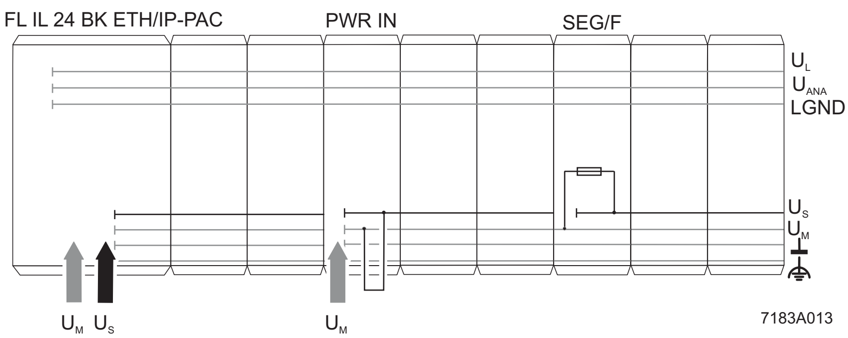


Figure 1-19 Segment circuit

FL IL 24 BK ETH/IP-PAC	Ethernet//IP bus coupler
PWR IN	Power terminal
SEG/F	Segment terminal with fuse as an example of a segment terminal

1.11 Voltage Concept

The Ethernet/IP bus coupler and the Inline local bus system have a defined voltage and grounding concept.

This avoids an undesirable effect on I/O devices in the logic area, suppresses undesirable compensating currents, and increases noise immunity.

Electrical isolation: Ethernet

The Ethernet interface is electrically isolated from the bus coupler logic. The Ethernet cable shield is directly connected to functional earth ground. The device has two functional earth ground springs, which have contact to the DIN rail when they are snapped on. The springs are used to discharge interference, rather than serve as protective earth ground. To ensure effective interference discharge, even for dirty DIN rails, functional earth ground is also led to terminals 1.4 and 2.4. Always ground either terminal 1.4 or 2.4 (see Figure 1-29 on page 1-38). This also grounds the Inline station of the bus coupler sufficiently up to the first power terminal.

A 120 V AC or 230 V AC power terminal interrupts the FE potential jumper. Therefore a 24 V DC power terminal, which is located directly behind such an area, must also be grounded using the FE terminal point.

To avoid the flow of compensating currents, connect a suitably sized equipotential bonding cable parallel to the Ethernet cable.

No electrical isolation of the Inline communications power

The bus coupler does not have electrical isolation for the Inline module communications power. U_{BK} (24 V), U_L (7.5 V), and U_{ANA} (24 V) are not electrically isolated.

Isolated supply for logic and I/O devices

The logic and I/O devices can be supplied by separate power supply units. If you wish to use different potentials for the communications power (U_{BK}) and the segment/main voltage (U_S/U_M), do not connect the GND and GND_{UBK} grounds of the supply voltages.

Option 1

The Fieldbus coupler main voltage U_M and the I/O supply U_S are provided separately with the same ground potential from **two** voltage supplies:

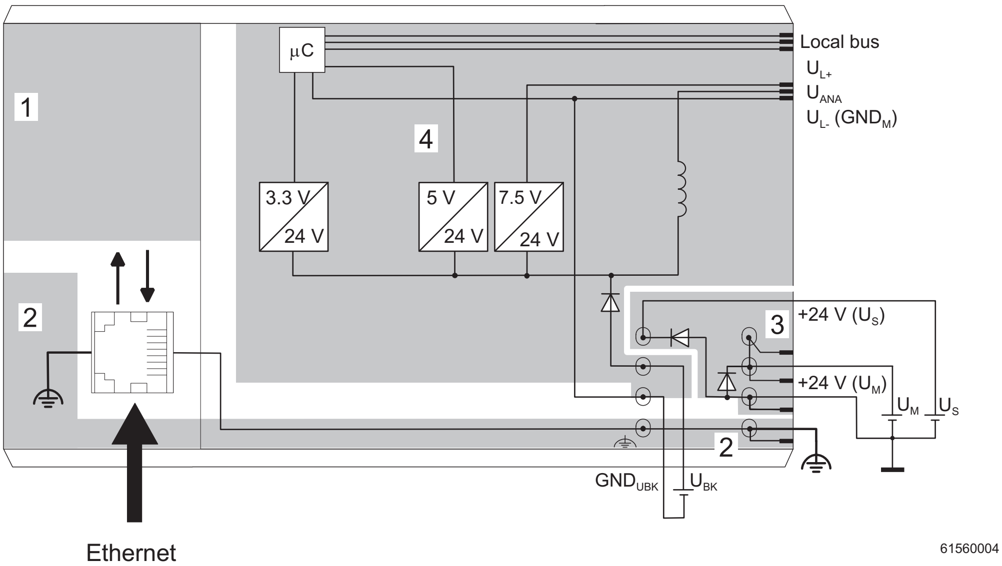


Figure 1-20 Potential areas in the bus coupler (two voltage supplies)

Potential areas:

1 Ethernet interface area

2 Functional earth ground (PE) and (shield) Ethernet interface area

3 Main voltage U_M and I/O voltage U_S area

4 Inline communications power

Option 2

Common supply of voltages U_{BK}, U_M, and U_S from **one** voltage supply:

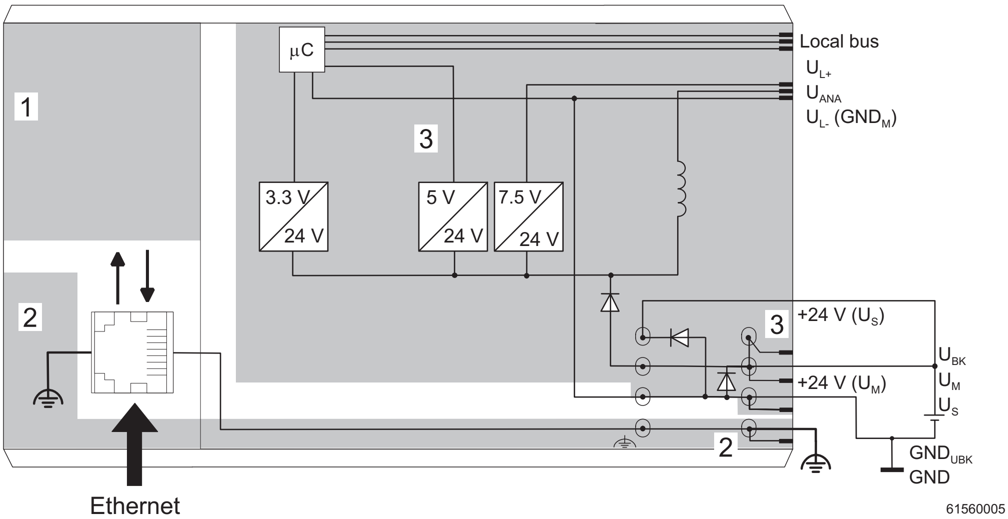


Figure 1-21   Bus coupler potentials (one voltage supply)

Potential areas:

1   Ethernet interface area
2   Functional earth ground / (shield) Ethernet interface area, bus coupler
3   Main voltage U_M and I/O voltage U_S area

The connector on the right can only be used when all the voltages supplied to the bus coupler have the same reference potential. Simply insert the external jumper to correctly connect all the supply points (see "Typical connection of the supply voltage" on page 1-21).

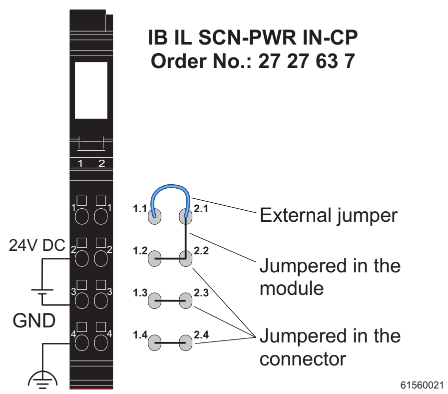


Figure 1-22   Power connector for supply from a single power supply unit

Potentials: Digital module

The isolation of the I/O circuit of a digital module to the communications power is only ensured if U_{BK} and U_M/U_S are provided from separate voltage supplies.

An example of this principle is shown in Figure 1-23 on a section of an Inline station.

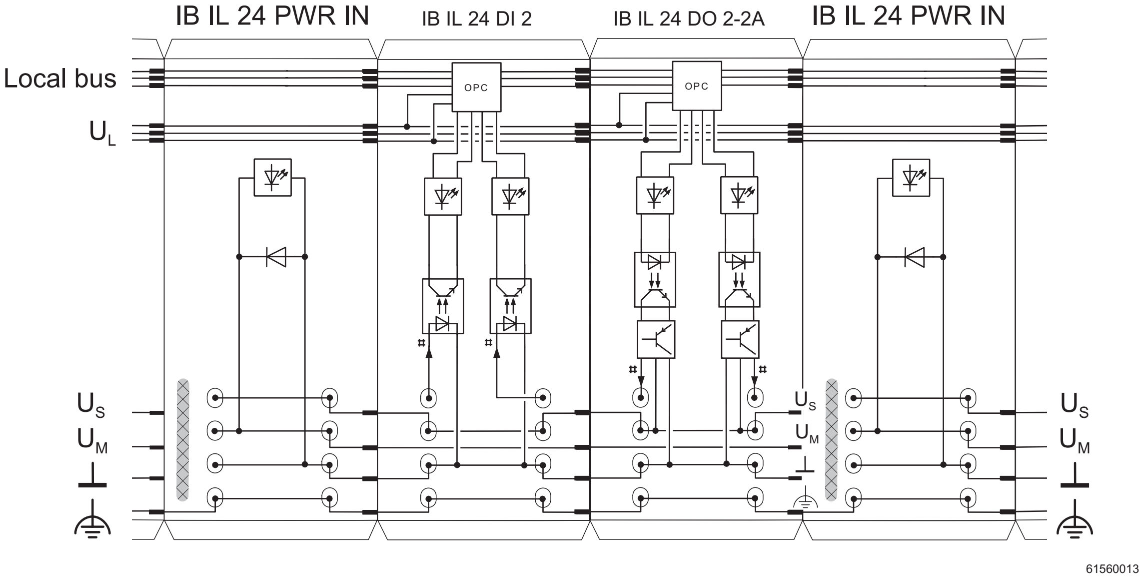


Figure 1-23 Example: Interruption/creation of the potential jumpers using the power terminal

The areas hatched in the figure XXXXX show the points at which the potential jumpers are interrupted.

Potentials: Analog module

The I/O circuit (measurement amplifier) of an analog module receives floating power from the 24 V supply voltage U_{ANA}. The power supply unit with electrical isolation is a component of an analog module. The voltage U_{ANA} is looped through in each module and, in this way, is also available to the next module.

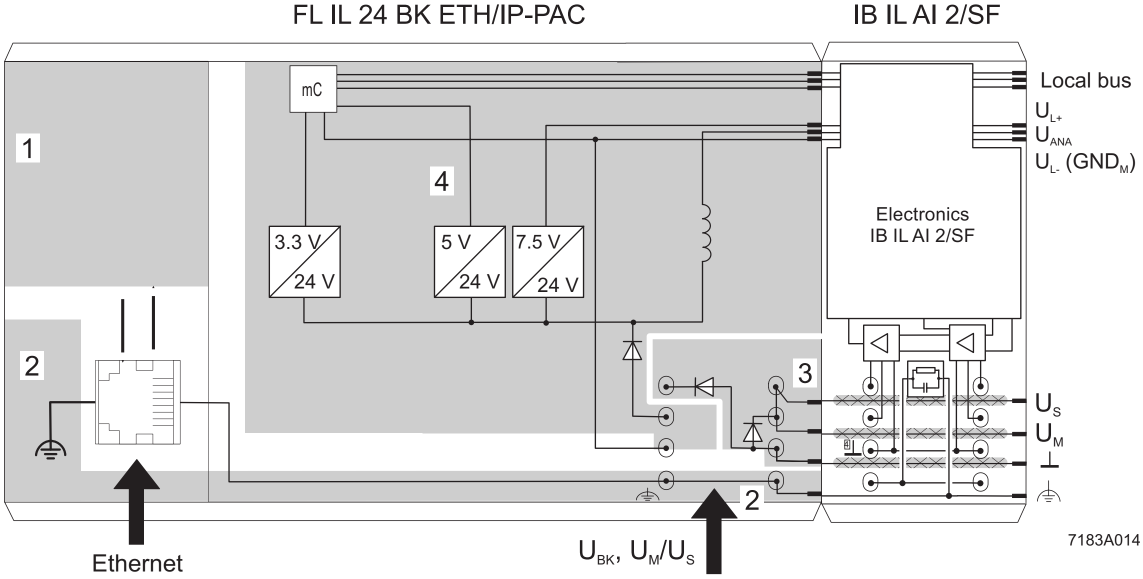


Figure 1-24 Electrical isolation between Ethernet bus coupler and analog module

The potential jumpers hatched XXXXX in the figure are not used in the analog module. This means that the 24 V supply of the bus coupler (U_{BK}) or the power terminal is always electrically isolated from the I/O circuit (measurement amplifier) of the analog module. The I/O circuit of the analog module is supplied from the analog circuit U_{ANA}.

I/O supply electrically isolated from one another

Several electrically isolated segment or main circuits can be created by using power terminals. A power terminal interrupts the U_S/U_M and GND potential jumpers and has terminal points for another power supply unit. In this way, the I/O circuits of the Inline modules are electrically isolated from one another before and after the power terminal.

During this process the 24 V power supply units on the low voltage side must not be connected to one another.
One method of electrical isolation using a power terminal is illustrated in Figure 1-25. If a number of grounds are connected, e.g., to functional earth ground, electrical isolation is lost.

Because U_S and U_M can be supplied separately, it is possible to create separate segment circuits using a segment terminal. Using a switch, it is possible, for example, to create a switched segment circuit (see Figure 1-25 on page 1-30). U_S and U_M can be protected separately, yet still have a common ground potential. Please observe the maximum total current of 8 A.

I/O supplies electrically isolated from one another.

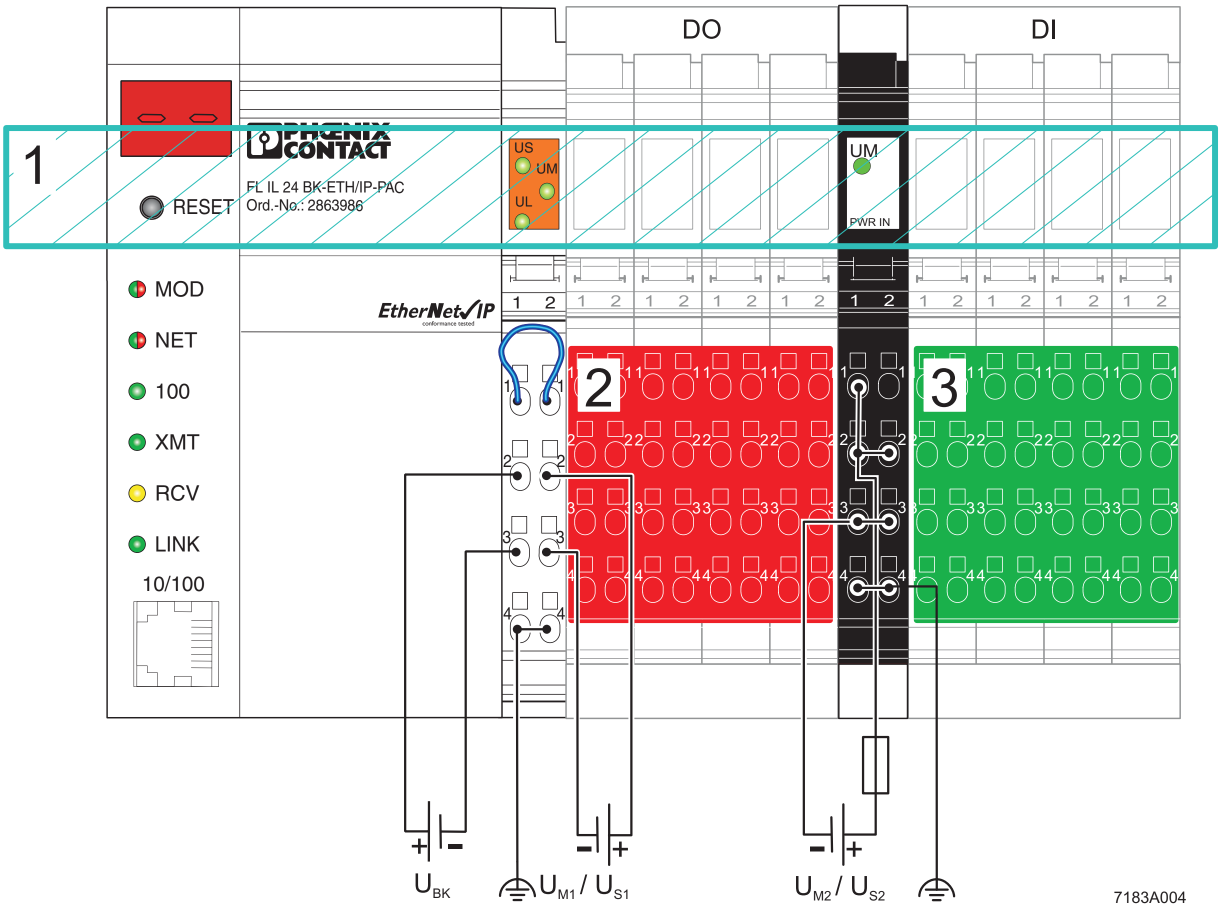


Figure 1-25 Structure of I/O supplies that are electrically isolated from one another

Potentials within the station:

1 Bus logic of the station

2 I/O (outputs)

3 I/O (inputs)

1.12 Mounting/Removing Modules and Connecting Cables

1.12.1 Installation Instructions

To ensure installation is carried out correctly, please read the "Installation Instructions for the Electrical Engineer" supplied with the bus coupler.

Do not replace modules while the power is connected

Before removing or mounting a module, disconnect the power to the entire station. Make sure the entire station is reassembled before switching the power back on.
Failure to observe this rule may damage the module.

1.12.2 Mounting and Removing Inline Modules

An Inline station can be set up by mounting the individual components side by side. No tools are required. Mounting side by side automatically creates voltage and bus signal connections (potential and data routing) between the individual station components.

The modules are mounted perpendicular to the DIN rail. This ensures that they can be easily mounted and removed even within limited space.

After a station has been set up, individual modules can be exchanged by pulling them out or plugging them in. Tools are not required.

DIN rail

All Inline modules are mounted on 35 mm (1.378 in.) standard DIN rails.

End clamp/CLIPFIX

Mount end clamps on both sides of the Inline station. The end clamps ensure that the Inline station is correctly assembled. End clamps fix the Inline station on both sides and keep it from moving side to side on the DIN rail. Phoenix Contact recommends using the CLIPFIX 35 (Order No. 30 22 21 8) or E/UK end clamps (Order No. 12 01 44 2).

To remove the bus coupler, the left end clamp must be removed first.

End plate

An Ethernet Inline station **must** be terminated with an end plate. It has no electrical function. It protects the station against ESD pulses and the user against dangerous contact voltage. The end plate is supplied with the bus coupler and must not be ordered separately.

1.12.3 Mounting

When mounting a module, proceed as follows (Figure 1-26):

- First snap on the electronics base, which is required for mounting the station, perpendicular to the DIN rail (detail A).

> Ensure that **all** featherkeys and keyways of adjacent modules are interlocked (detail B).

The keyway/featherkey connection links adjacent modules and ensures safe potential routing.

- Next, attach the connectors to the corresponding base.

First, place the front connector shaft latching in the front snap-on mechanism (detail C).

Then press the top of the connector towards the base until it snaps into the back snap-on mechanism (detail D).

> The keyways of an electronics base do not continue when a connector has been installed on the base. When snapping on an electronics base, there must be no connector on the left-hand side of the base. If a connector is present, it will have to be removed.

> Use end clamps to fix the Inline station to the DIN rail (see Ordering Data).

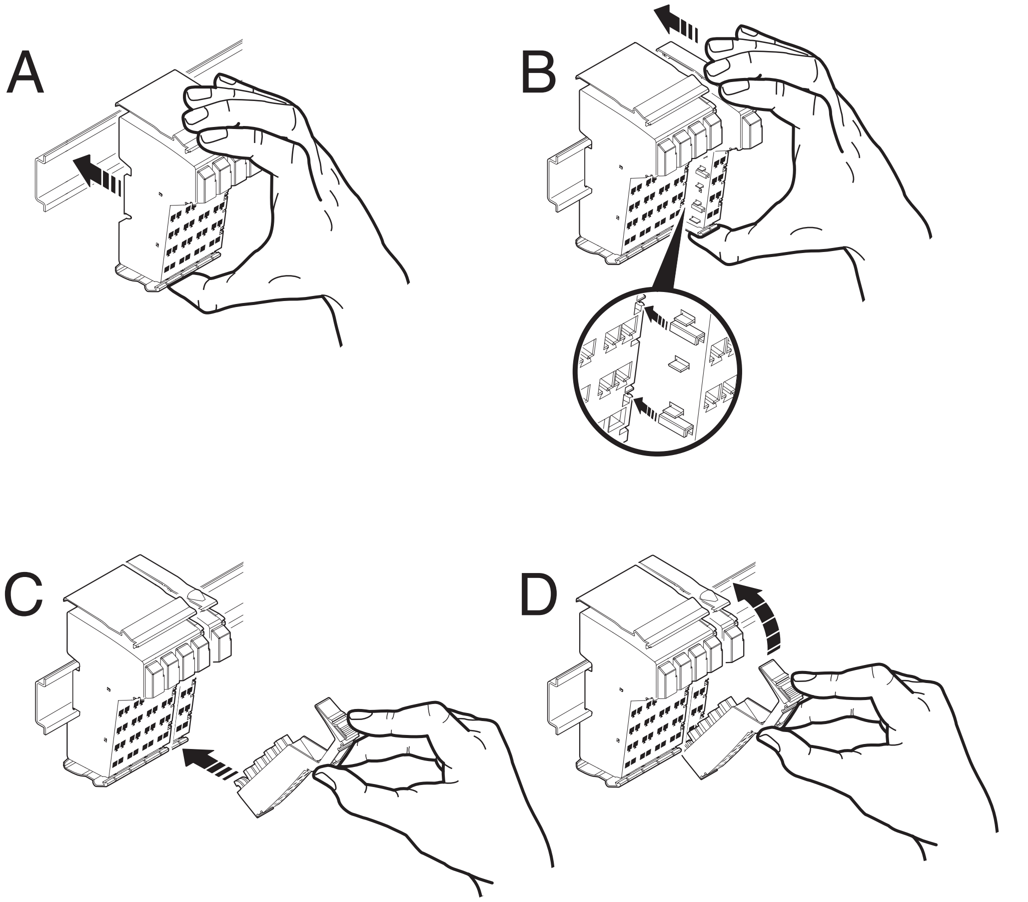


Figure 1-26 Snapping on a module

1.12.4 Removing

When removing a module, proceed as follows (Figure 1-27):

- If there is a labeling field, remove it (A1 in detail A).

> If a module has more than one connector, all of these must be removed. Below is a description of how to remove a 2-slot module.

- Lift the connector of the module to be removed by pressing on the back connector shaft latching (A2 in detail A).
- Remove the connector (detail B).
- Remove the left-adjacent and right-adjacent connectors of the neighboring modules (detail C). This prevents the potential routing featherkeys and the keyway/featherkey connection from being damaged. You also have more space available for accessing the module.
- Press the release mechanism, (D1 in detail D) and remove the electronics base from the DIN rail by pulling the base straight back (D2 in detail D). If you have not removed the connector of the next module on the left, remove it now in order to protect the potential routing featherkeys and the keyway/featherkey connection.

> To remove the bus coupler, the left end clamp must be removed first.

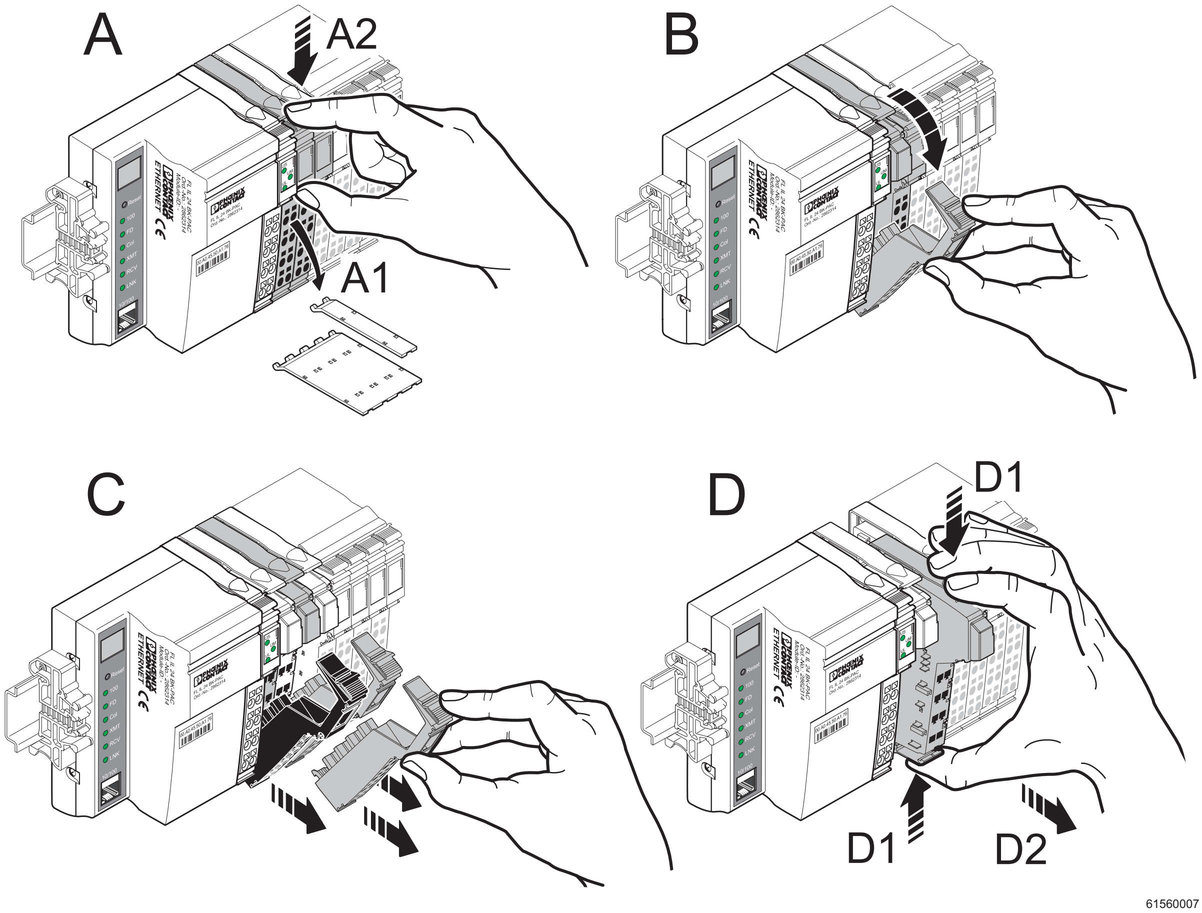


Figure 1-27 Removing a module

Replacing a module

If you want to replace a module within the Inline station, follow the removal procedure described above. Do not snap the connector of the module directly to the left back on yet. First, insert the base of the new module. Then reconnect all the connectors.

Use end clamps to fix the Inline station to the DIN rail (see Ordering Data).

1.12.5 Replacing a Fuse

The power and segment terminals are available with or without fuses.

For modules with fuses, the voltage presence and the fuse state are monitored and indicated by diagnostic indicators.

If a fuse is not present or defective, you must insert or replace it.

Observe the following when replacing a fuse in order to protect your health and the system.

1. Use the screwdriver carefully to avoid injury.
2. Lift the fuse out at the metal contact. Do not lift the fuse out at the glass part as you may break it.
3. Carefully lift the fuse out at one side and remove it by hand. Make sure the fuse does not fall into your system.

Follow these steps when replacing a fuse (see Figure 1-28):

- Lift the fuse lever (A).
- Insert the screwdriver behind a **metal contact** of the fuse (B).
- Carefully lift the metal contact of the fuse (C).
- Remove the fuse by hand (D).
- Insert a new fuse (E).
- Push the fuse lever down again until it clicks into place (F).

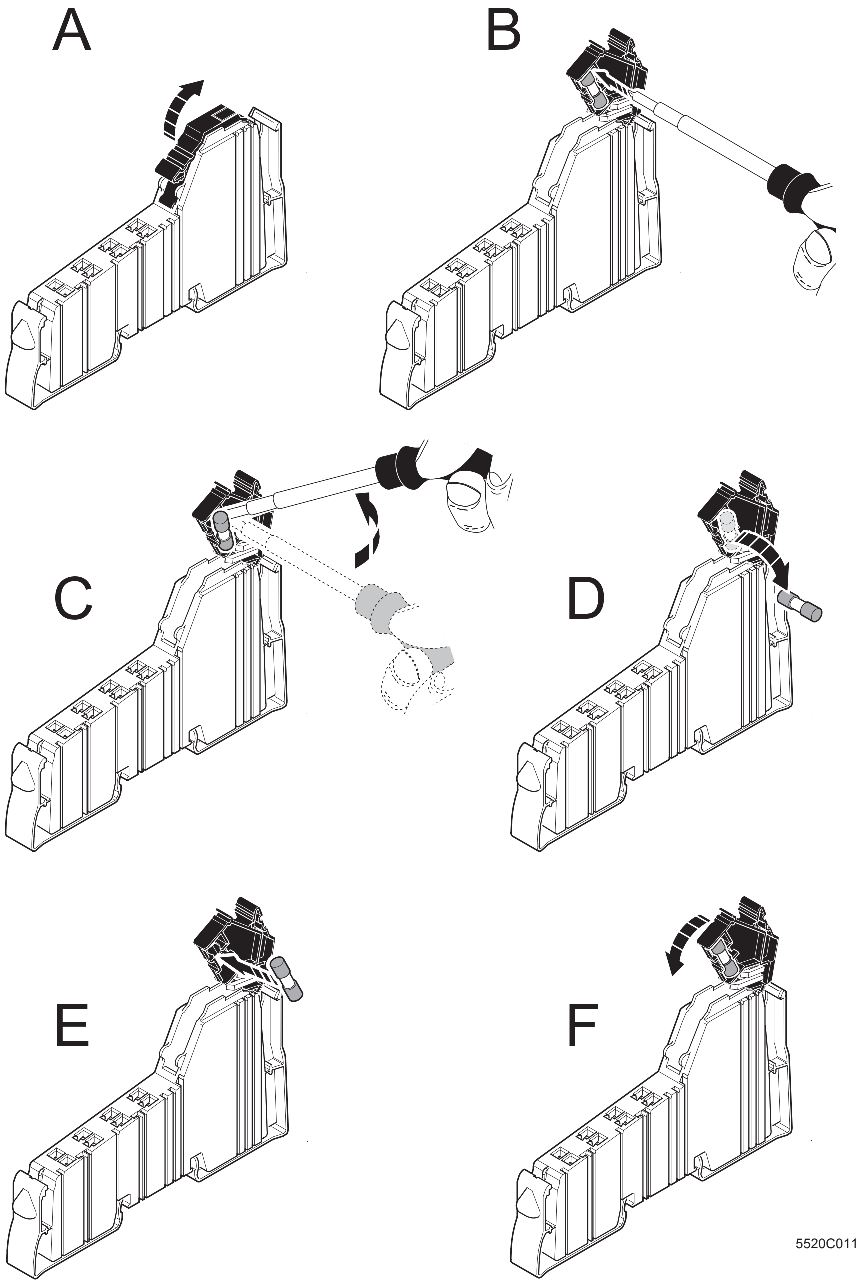


Figure 1-28 Replacing a fuse

1.13 Grounding an Inline Station

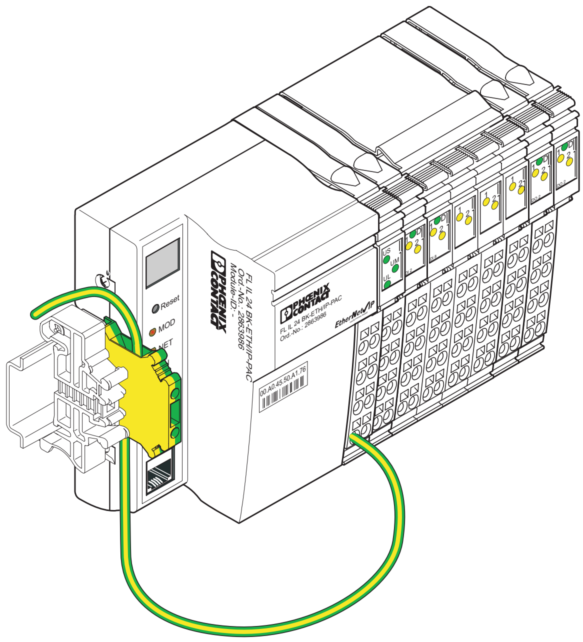
All devices in an Inline station must be grounded so that any possible interference is shielded and discharged to ground potential. A wire of at least 1.5 mm^2 (16 AWG) must be used for grounding.

Ethernet/IP bus coupler and supply terminals

The bus coupler, power terminals, and segment terminals have FE springs (metal clips) on the bottom of the electronics base. These springs create an electric connection to the DIN rail. Use grounding terminal blocks to connect the DIN rail to protective earth ground. The modules are grounded when they are snapped onto the DIN rail.

Required additional grounding

In order to ensure reliable grounding even if the DIN rail is dirty or the metal clip has been damaged, Phoenix Contact specifies that the bus coupler must also be grounded via the FE terminal point (e.g., with the USLKG 5 universal ground terminal block, Order No. 04 41 50 4, see Figure 1-29).



7183A005

Figure 1-29 Additional grounding of the FL IL 24 BK ETH/IP-PAC

FE potential jumper

The FE potential jumper (functional earth ground) runs from the bus coupler through the entire Inline station. Ground the DIN rail. FE is grounded when a module is snapped onto the DIN rail correctly. If supply terminals are part of the station, the FE potential jumper is also connected with the grounded DIN rail.

> The function of FE is to discharge interference. It does not provide shock protection for people.

Low-level signal

The other Inline low-level signal modules are automatically grounded via the FE potential jumper when they are mounted adjacent to other modules.

Power level

The FE potential jumper is also connected to the power modules.

1.13.1 Shielding an Inline Station

Shielding is used to reduce the effects of interference on the system.

In the Inline station, the Ethernet cable and the module connecting cables for analog signals are shielded.

Observe the following when using shielded cables:

- Fasten the shielding so that as much of the braided shield as possible is held underneath the clamp of the shield connection.
- Make sure there is good contact between the connector and module.
- Do not damage or squeeze the wires. Do not strip off the wires too far.
- Make a clean wire connection.

1.13.2 Shielding Analog Sensors and Actuators

Always connect analog sensors and actuators using shielded, twisted-pair cables.

Connect the shielding to the shield connector. The method for connecting the shielding is described in section 1.14.2, "Connecting Shielded Cables Using the Shield Connector".

Analog input and output modules require different shielding connections. The cable lengths must also be considered.

Table 1-7 Overview: shield connection of analog sensors/actuators

Module Type	Connection to the Module	Cable Length	Connection to the Sensor/Actuator
Analog input module IB IL AI 2/SF	Within the module, ground is connected to FE via an RC element.	< 10 m (32.81 ft.)	–
		> 10 m (32.81 ft.)	Connect the sensor directly to PE
Analog output module IB IL AO ...	Via shield connection clamp directly to FE	< 10 m (32.81 ft.)	–
		> 10 m (32.81 ft.)	Isolate the actuator with an RC element and connect it to PE

Connecting an IB IL 24 AI 2/SF Analog Input Module

- Connect the shielding to the shield connector (see section 1.14.2, "Connecting Shielded Cables Using the Shield Connector").
- When connecting the sensor shielding with FE potential, ensure a large surface connection.

Within the module, ground is connected to FE via an RC element.

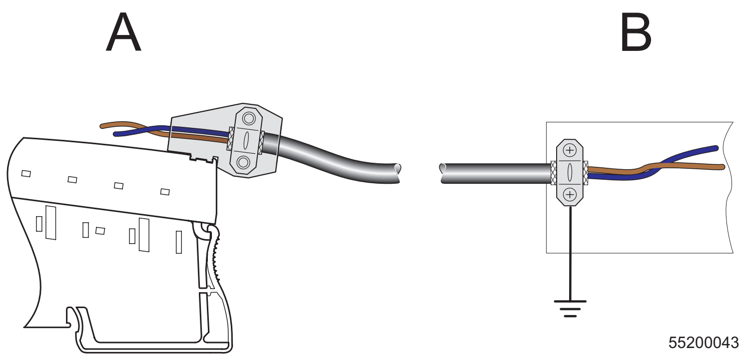


Figure 1-30 Connection of analog sensors, signal cables > 10 m

A Module side
B Sensor side

If you want to use both channels of the IB IL AI 2/SF module, there are different ways of connecting the shielding, depending on the cross-section.

1. Use a multi-wire cable for the connection of both sensors and connect the shielding to the shield connector as described above.
2. Use a thin cable for the connection of each sensor and connect the shielding of both cables together to the shield connector.
3. Use the standard connector (IB IL SCN-8; without shield connector). Twist the braided shield of each cable and place it on one of the terminal points to be used for FE connection.
   You should only use this option if the cross-section is too large and the first two methods are not possible.

Connecting an Analog Output Module IB IL AO ...

- Connect the shielding via the shield connector (see section 1.14.2, "Connecting Shielded Cables Using the Shield Connector").
- When connecting the shielding with the FE potential, ensure a large surface connection.

Danger of creating ground loops

The shielding must only be directly connected to ground potential at one point.

– For **cable lengths exceeding 10 meters (32.81 ft.)** the actuator side should always be isolated by means of an RC element.
The capacitor C should typically have values of 1 nF to 15 nF. The resistor R should be at least 10 MΩ.

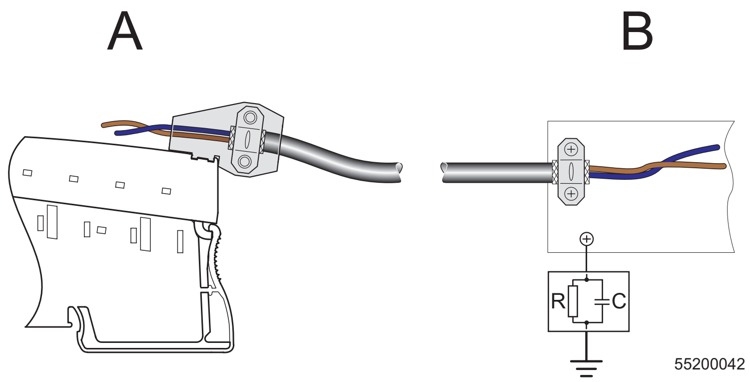


Figure 1-31 Connection of actuators, signal cables > 10 m

A Module side
B Actuator side

1.14 Connecting Cables

Both shielded and unshielded cables are used in a station.

The cables for the I/O devices and supply voltages are connected using the spring-cage connection method. This means that signals up to 250 V AC/DC and 5 A with a conductor cross section of 0.2 mm^2 through 1.5 mm^2 (AWG 25 - 16) can be connected.

The Ethernet cable is connected via an 8-pos. RJ45 connector.

1.14.1 Connecting Unshielded Cables

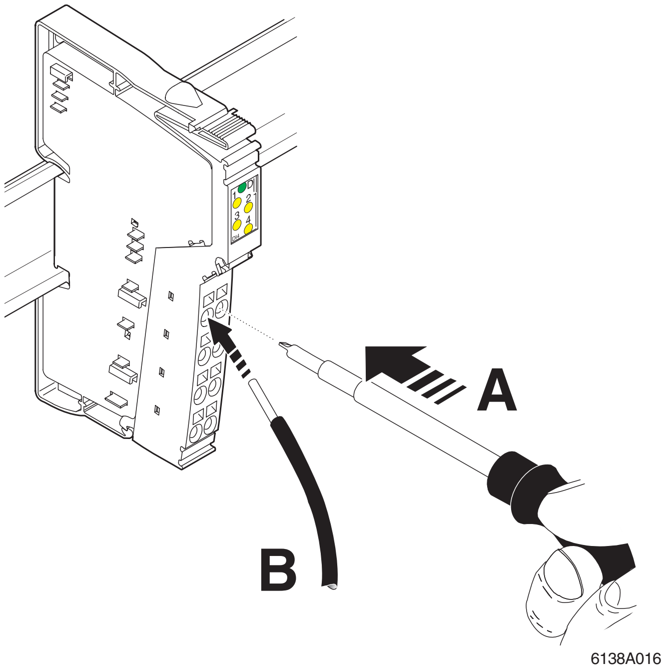


Figure 1-32 Connecting unshielded cables

Wire the connectors as required for your application.

For connector assignment, please refer to the appropriate module-specific data sheet.

When wiring, proceed as follows:

- Strip 8 mm (0.315 in.) off the cable. Fieldbus coupler and Inline wiring is normally done without ferrules. However, it is possible to use ferrules. If using ferrules, make sure they are properly crimped.
- Push a screwdriver into the slot of the appropriate terminal point (Figure 1-32, detail 1), so that you can insert the wire into the spring opening.
  Phoenix Contact recommends using a SFZ 1-0,6X3,5 screwdriver (Order No. 12 04 51 7; see Phoenix Contact "CLIPLINE" catalog).
- Insert the wire (Figure 1-32, detail B). Remove the screwdriver from the opening. This clamps the wire.
- After installation, the wires and the terminal points should be labeled.

1.14.2 Connecting Shielded Cables Using the Shield Connector

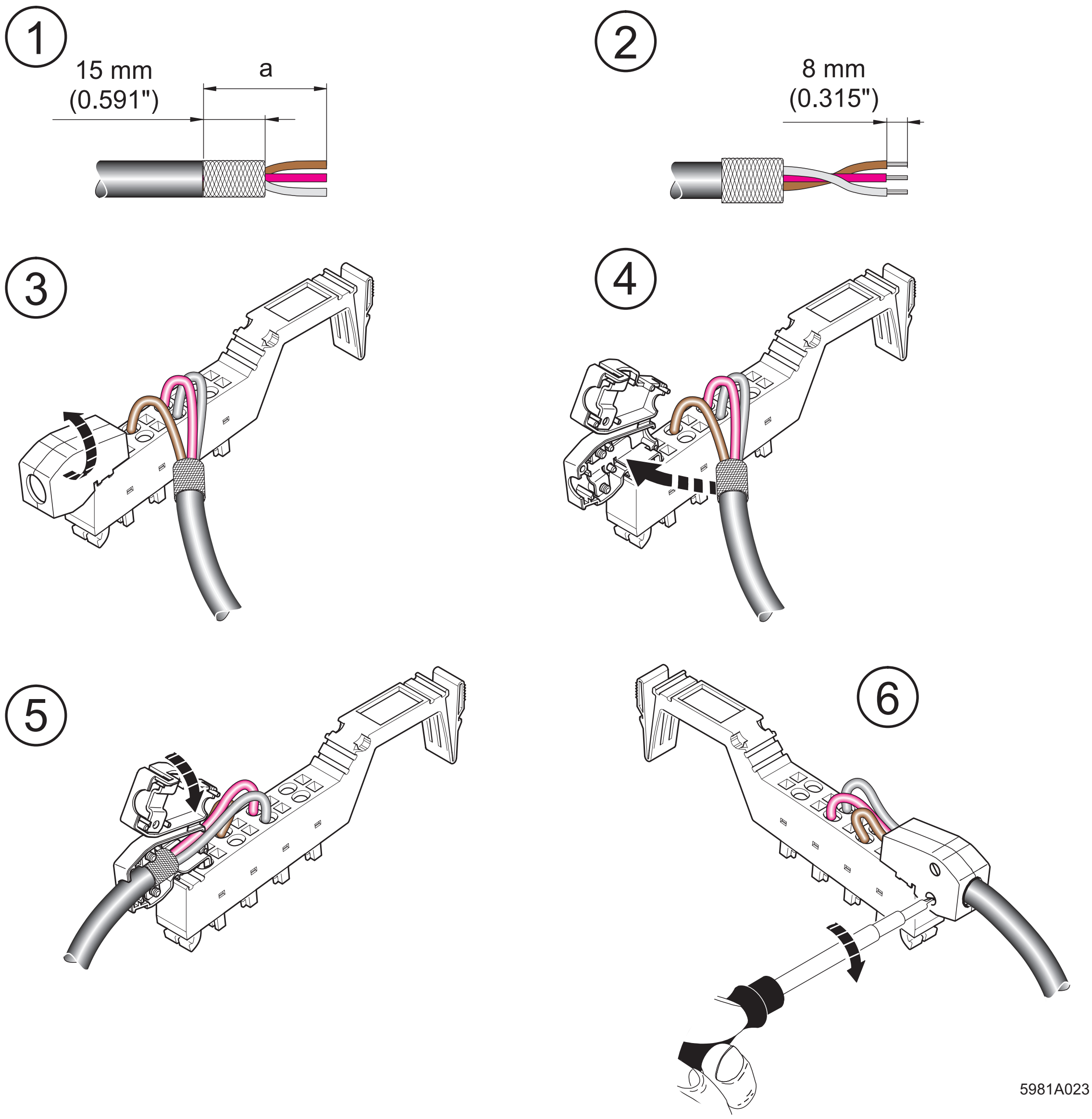


Figure 1-33 Connecting the shield to the shield connector

DOWNLOADED FROM WWW.SCATTS.CO.UK

This section describes the connection of a shielded cable, using an "analog cable" as an example.

Connection should be carried out as follows:

Stripping cables

- Strip the outer cable sheath to the desired length (a). (1)
  The desired length (a) depends on the connection position of the wires and whether there should be a large or a small space between the connection point and the shield connection.
- Shorten the braided shield to 15 mm (0.591 in.). (1)
- Fold the braided shield back over the outer sheath. (2)
- Remove the protective foil.
- Strip 8 mm (0.315 in.) off the wires. (2)

Inline wiring is normally done without ferrules. However, it is possible to use ferrules. If using ferrules, make sure they are properly crimped.

Wiring the connectors

- Push a screwdriver into the slot of the appropriate terminal point (Figure 1-32 on page 1-42, detail 1), so that you can insert the wire into the spring opening.
  Phoenix Contact recommends using a SFZ 1-0,6X3,5 screwdriver
  (Order No. 12 04 51 7; see Phoenix Contact "CLIPLINE" catalog).
- Insert the wire (Figure 1-32 on page 1-42, detail 2). Remove the screwdriver from the opening. This clamps the wire.

For connector assignment, please refer to the appropriate module-specific data sheet.

Connecting the shield

- Open the shield connector. (3)
- Check the direction of the shield connection clamp in the shield connector (see Figure 1-34).
- Place the cable with the folded braided shield in the shield connector. (4)
- Close the shield connector. (5)
- Fasten the screws on the shield connector using a screwdriver. (6)

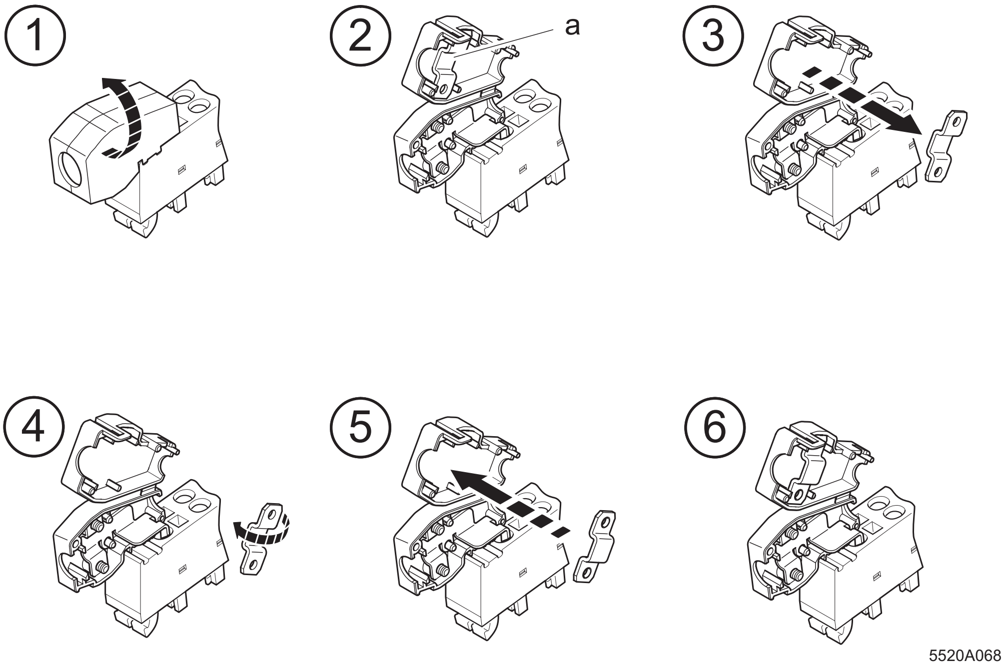


Figure 1-34 Shield connection clamp alignment

Shield connection clamp

The shield connection clamp (a in Figure 1-34, detail 2) in the shield connector can be used in various ways depending on the cross section of the cable. For thicker cables, the dip in the clamp must be turned away from the cable (Figure 1-34, detail 2). For thinner cables, the dip in the clamp is turned towards the cable (Figure 1-34, detail 6).

If you need to change the direction of the shield connection clamp, proceed as shown in Figure 1-34:

- Open the shield connector housing (1).
- The shield connection is delivered with the clamp positioned for connecting thicker cables (2).
- Remove the clamp (3), turn it to suit the cross-section of the cable (4), then reinsert the clamp (5).
- Figure 6 shows the position of the clamp for a thin cable.

1.15 Connecting the Voltage Supply

To operate a station you must provide the supply voltage for the bus coupler, logic of the modules, and the sensors and actuators.

The voltage supplies are connected using unshielded cables (see section 1.14.1).

For the connector assignment of the supply voltage connections, please refer to the module-specific data sheets for power and segment terminals.

Do not replace terminals while the power is connected

Before removing or mounting a module, disconnect the power to the entire station. Make sure the entire station is reassembled before switching the power back on.

1.15.1 Power Terminal Supply

Apart from supplying the I/O voltage at the Fieldbus coupler, it is also possible to provide the voltage using a power terminal.

U_M

24 V Main Circuit Supply

The main power is reintroduced at the power terminal.

U_S

24 V Segment Circuit Supply

The segment voltage can be supplied at the power terminal or generated from the main power. Install a jumper or create a segment circuit using a switch to tap the voltage U_S from the main circuit U_M.

Electrical isolation

You can create a new potential area through the power terminal.

Voltage areas

Power terminals can be used to create substations with different voltage areas. Depending on the power terminal, it is possible to work with 24 V DC, 120 V AC or 230 V AC.

Use appropriate power terminals for different voltage areas

To use different voltage areas within a station, a new power terminal must be used for each area.

Dangerous voltage

When the power terminal is removed, the metal contacts are freely accessible. With 120 V or 230 V power terminals, it should be assumed that dangerous voltage is present. You **must** disconnect power to the station **before removing** a terminal.

If these instructions are not followed, there is a danger of damage to health, or even of a life-threatening injury.

1.15.2 Provision of the Segment Voltage Supply at Power Terminals

You cannot provide voltage at the segment terminal.

A segment terminal can be used to create a new partial circuit (segment circuit) within the main circuit. This segment circuit permits the separate supply of power outputs and digital sensors and actuators.

You can use a jumper to tap the segment voltage from the main circuit. If you use a switch, you can control the segment circuit externally.

You can create a protected segment circuit without additional wiring by means of a segment terminal with a fuse.

1.15.3 Demands on the Power Supply Units

Use power supply units with safe isolation

Only use power supply units that ensure safe isolation between the primary and secondary circuits according to EN 50178.

For additional voltage supply requirements, please refer to the data sheets for the segment and power terminals.

1.16 Connecting Sensors and Actuators

Sensors and actuators are connected using connectors. Each module-specific data sheet indicates the connector(s) to be used for that specific module.

Connect the unshielded cable as described in Section 1.14.1 on page 1-42 and the shielded cable as described in Section 1.14.2 on page 1-44.

1.16.1 Connection Methods for Sensors and Actuators

Most of the digital I/O modules in the Inline product range permit the connection of sensors and actuators in 2, 3 and 4-wire technology.

Because of the different types of connectors, a single connector can support the following connection methods:

– 2 sensors or actuators in 2, 3 or 4-wire technology
– 4 sensors or actuators in 2 or 3-wire technology
– 2 sensors or actuators in 2 or 3-wire technology with shielding (for analog sensors or actuators)

When connecting analog devices, please refer to the module-specific data sheets, as the connection method for analog devices differs from that for digital devices.

1.16.2 Connection Examples for Digital I/O Modules

Various connection options are described below using 24 V DC modules as an example. For the 120 V/230 V AC area, the data changes accordingly. A connection example is given in each module-specific data sheet.

Table 1-8 Overview of the connections used for digital input modules

Connection	Representation in the Figure	2-wire	3-wire	4-wire
Sensor signal IN	IN	X	X	X
Sensor supply U_S / U_M	U_S (+24 V)	X	X	X
Ground GND	GND (⊥)	–	X	X
Ground/FE shielding	FE (⏚)	–	–	X

X Used
– Not used

Table 1-9 Overview of the connections used for digital output modules

Connection	Representation in the Figure	2-wire	3-wire	4-wire
Actuator signal OUT	OUT	X	X	X
Actuator supply U_S	U_S (+24 V)	–	–	X
Ground GND	GND (⊥)	X	X	X
Ground/FE shielding	FE (⏚)	–	X	X

X Used
– Not used

In the following figures U_S designates the supply voltage. Depending on which potential jumper is accessed, the supply voltage is either the main voltage U_M or the segment voltage U_S.

Section 2

This section informs you about
– Web-Based Management
– I/O Memory Mapping
– I/O Data Transfer

2 Startup/Operation

2.1 Default Upon Delivery/Default Settings

Per default upon delivery, the following functions and features are available:

- The password is "private"
- The bus coupler has no valid IP parameters:
  IP address: 0.0.0.0
  Subnet mask: 0.0.0.0
  Gateway: 0.0.0.0
- Expert mode deactivated
- System description: Ethernet/IP bus coupler
  System contact: Unknown
  System name: FL IL 24 BK ETH/IP-PAC
  System location: Unknown
- HW watchdog activated (default parameter: 0x00000001)
- No INTERBUS configuration stored. All entries set to 0x0000
- Fault response mode: 1
- Protocol switch: 0

2.2 Sending BootP Requests

Initial Startup:

During initial startup, the device sends a BootP request without interruption until it receives a valid IP address. The requests are transmitted at varying intervals (2 s, 4 s, 8 s, 2 s, 4 s, etc.) so that the network is not loaded unnecessarily. If valid IP parameters are received, they are saved as configuration data by the device.

Further restarts:

If the device already has valid configuration data, it only sends three more BootP requests on a restart. If it receives a BootP reply, the new parameters are saved. If the device does not receive a reply, it starts with the previous configuration.

2.3 Web-Based Management

2.3.1 Event Table

This internal event table operates as a First-In-First-Out (FIFO) stack. It shows at least 20 events. When event 21 occurs, event 1 is dropped from the table. All events from the event table are pure information and have no influence on the operation of the FL IL 24 BK ETH/IP-PAC.

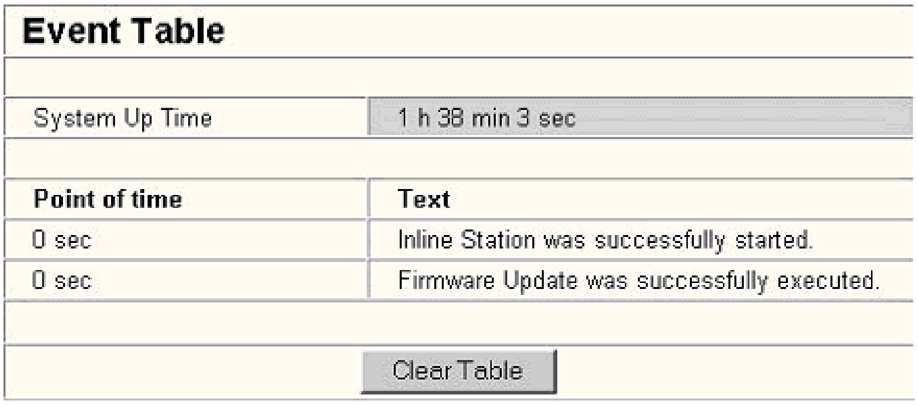


Figure 2-1 Event table in the WBM

2.3.2 Diagnostics

On this website, you can have a look at the INTERBUS diagnostics. The diagnostic status and diagnostic parameter register are shown here. The display values are also displayed here (in the diagnostic register). To monitor the diagnostic registers and the display, a separate website is implemented which will start in a separate browser window. The polling time of this side is user selectable. The user can set a time between 1 and 30 seconds. The standard value for the polling time is 3 seconds, approximately.

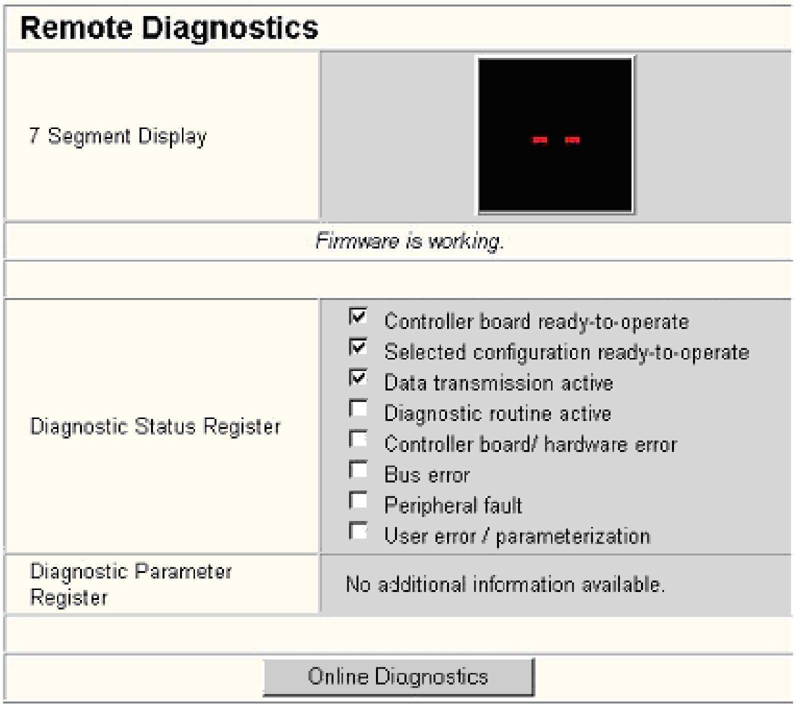


Figure 2-2 Remote Diagnostics in WBM

2.3.3 Process Data

This side allows the user to access process data. After the password input the writing of process data will be allowed. The password is valid for the entire time the side is loaded. After reloading the side the password has to be reentered. The bus structure is simulated with text fields in the current INTERBUS color code. Only the text fields for output modules can be edited. There is no mechanism implemented to refresh this website. The user can refresh the current website via the "write" or "read" buttons.

2.3.4 Bus Configuration

This website shows the current configuration of the Inline station. If an error occurs, the error location is indicated: The number of the affected element is highlighted in red.

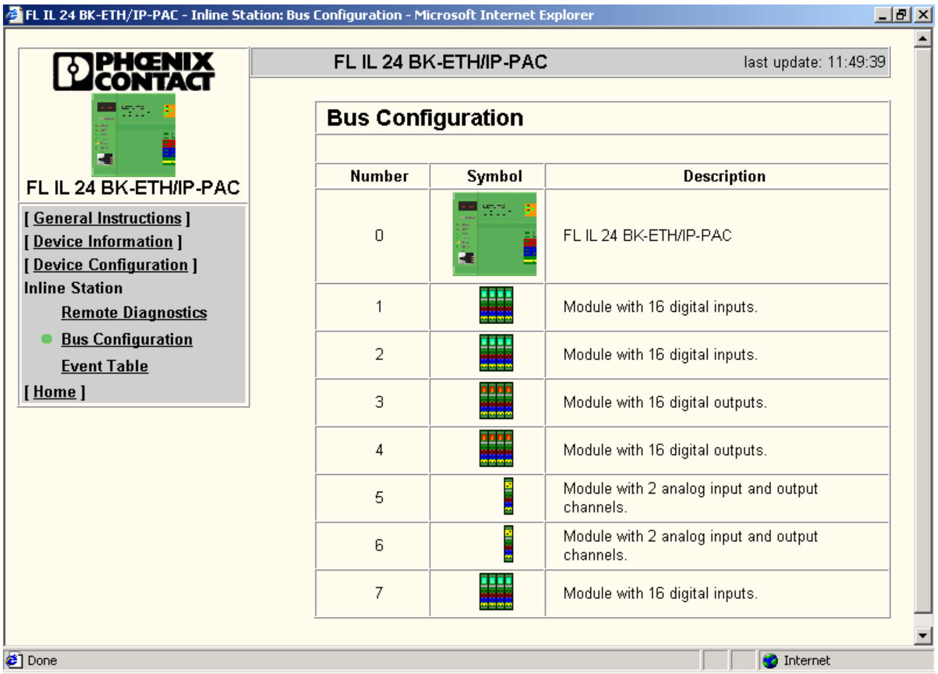


Figure 2-3 Bus Configuration in WBM

2.3.5 Monitoring Functions

Monitoring the Ethernet/IP connections is executed by the Ethernet/IP protocol stack with the help of connection timers. If a connection is aborted, due to a broken cable or a control system without connection reset, the outputs are set back to the pre-defined state. The state is set either in the DOP or in the AOP objects. In order to regain control over the outputs, the error must be removed.

2.3.6 Password Protection

The bus terminal is protected by two passwords (case-sensitive). The password for read access is "public", while the password for read and write access via SNMP and WBM is "private". All status changes to the bus terminal are only possible after the password for read and write access has been entered. The password can be changed at any time. Your unique password must be between 4 and 12 characters long.

> If you forget the password, the device can be re-enabled by Phoenix Contact. Ensure you have the exact device designation and serial number ready when you contact the telephone number indicated on the last page.

DOWNLOADED FROM WWW.SCATTS.CO.UK

2.3.7 Traps

When important events occur, e.g., a configuration change, the bus coupler sends a trap to a trap manager defined by the user. This enables the network administrator to react quickly to these events and to ensure network availability. Traps are usually only transmitted once.

Supported Traps

– ColdStart - sent twice each time the device is restarted.
– PasswordChange - sent after the password is changed successfully.
– FWHealth - sent after any changes to the firmware operating status.
– Configuration - sent after any changes to the hardware configuration.

2.3.8 Autoconfiguration

That means you can now change your local bus configuration and make it active via the web page. To use this function you have to stop your I/O connection. That means you can not use ADD ALL function when you have running I/O connection.

2.4 Configuration

Configuration services can be accomplished by using defined CIP objects. Some configuration features are also made available to the user through the web server.

2.4.1 Watchdog Integration

A watchdog realizes differences in the normal operation mode of the firmware.

If it is not set back within a special interval it will be presumed that an software or a hardware error has appeared on the device. In this case a Fast Interrupt Service (FIRS) will be started. Within the FIRS, a reset is generated and in a special register a hint is inserted that the watchdog timeout has been exceeded. A reset is taking place.
After the reset, the Boot Loader reads the special register. On the display, "4A" is indicated and no other action is executed. The user must reset the device. After the reset, the special register will be cleared, so that the device returns to normal operation mode.

To avoid conflicts at low data transmission quality due to frequently exceeded watchdog timeouts, the watchdog mechanism can be switched off. In order to switch the watchdog mechanism ON/OFF, an SNMP object can either be described or the watchdog is de-/activated via WBM.

Per default, the watchdog is activated, the timeout value is set in the firmware.

> Please do not mistake this watchdog for a process data watchdog.

2.4.2 I/O Control System With Ethernet/IP

With the help of the Configuration Object, the user can control which data are transmitted via Ethernet/IP.

Data	Location			Default Setting	Control Object		
	Class	Instance	Attribute		Class	Instance	Attribute
Run/Idle	-	-	-	Enabled	0x64	1	43
Inline Control Byte	0x65	1	20	Disabled	0x64	1	32
Actual DOPs	0x09	X	3	Enabled	0x64	1	4
Reserved DOPs	0x09	X	3	Disabled	0x64	1	23
Padding	NULL			Enabled if number of DOP bytes uneven	0x64	1	14
Actual AOPs	0x0B	X	3	Enabled	0x64	1	6
Reserved AOPs	0x0B	X	3	Disabled	0x64	1	36
AIP Control Data	0x0A	X	101	Disabled	0x0A	X	102
Special Function Data	0x67	X	4	Enabled	0x67	X	7
PCP Data	–	–	–	Disabled	0x64	1	38
PCP Module X Process Data	0x69	X	20	Disabled	0x69	X	21
PCP Module X Request Fragment	0x69	X	15	Disabled	0x69	X	17
SCO Data	–	–	–	Disabled	0x64	1	39
SCO Module X Control Word	0x6A	X	6	Disabled	0x6A	X	31
SCO Module X Transmit Fragment	0x6A	X	10	Disabled	0x6A	X	32

Table 2-1 Produced data

Data	Location			Default Setting	Control Object		
	Class	Instance	Attribute		Class	Instance	Attribute
Run/Idle	–	–	–	Disabled	0x64	1	42
Inline Status Byte	0x65	1	4	Enabled	0x64	1	11
Inline First Faulted Module	0x65	1	5	Enabled	0x64	1	11
DIP Faults	0x08	X	4	Disabled	0x64	1	15
DOP Faults	0x09	X	4	Disabled	0x64	1	16
AIP Faults	0x0A	X	4	Disabled	0x64	1	17
AOP Faults	0x0B	X	4	Disabled	0x64	1	18
Special Function Faults	0x67	X	6	Disabled	0x64	1	19
Actual DIPs	0x08	X	3	Enabled	0x64	1	3
Reserved DIPs	0x08	X	3	Disabled	0x64	1	22
Padding	NULL			Enabled if number of DIP bytes uneven	0x64	1	14
Actual AIPs	0x0A	X	3	Enabled	0x64	1	5
Reserved AIPs	0x0A	X	3	Disabled	0x64	1	35
AOP Response Data	0x0B	X	100	Disabled	0x0B	X	102
Special Function Data	0x67	X	4	Enabled	0x67	X	7
PCP Data	–	–	–	Disabled	0x64	1	38
PCP Module X Process Data	0x69	X	19	Disabled	0x69	X	21
PCP Module X Response Fragment	0x69	X	16	Disabled	0x69	X	17
SCO Data	–	–	–	Disabled	0x64	1	39
SCO Module X Status Word	0x6A	X	5	Disabled	0x6A	X	31
SCO Module X Receive Fragment	0x6A	X	9	Disabled	0x6A	X	32

In order to obtain cycle consistency between the I/O data and the station bus, an exchange buffer is used for the bus coupler. The exchange mechanism ensures that the required I/O data are available at the right time - being read and write protected.

2.5 The CIP I/O Module Capacity

The bus coupler is capable of processing the maximum number of instances (points) for 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[1]
– Serial Communication Object	8 instances[1]

2.6 Configuring the Inline Station

> If using a serial or another type of PCP module, read this section then refer to section 4, "Serial and Other PCP Inline Modules".

2.6.1 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)

> Any module that is not recognized will be placed into the special function category.
>
> When configuring the Inline Ethernet/IP™ bus coupler, it is recommended that you remove the connection to the Ethernet/IP ™ scanner or make sure the scan list for the Ethernet/IP ™ scanner is empty.

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

Configuration Methods

When creating, adding to, or changing an Inline Ethernet/IP 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

1. Electronic Data Sheet (EDS) file
2. Auto-configuration (no software required)
3. Sending an explicit message

Electronic Data Sheet (EDS) File Method

The EDS file is the software interface between the bus coupler and a configuration software package. The EDS file contains information about the number of produced and consumed bytes and user-settable parameters.

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 Ethernet/IP 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 (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 the Pad I/O-Parameter to a 0 or 1 (default). If not used, proceed to step 5.

> 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 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 section "Reserving I/O Memory for Future System Expansion" on page 2-18. 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 Reserve Digital Inputs, Parameter Reserve Digital Outputs, Parameter Reserve Analog Inputs and Parameter 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 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.

Auto-configuration Method

Auto-configuration allows for in-the-field configuration of the bus coupler without any software.

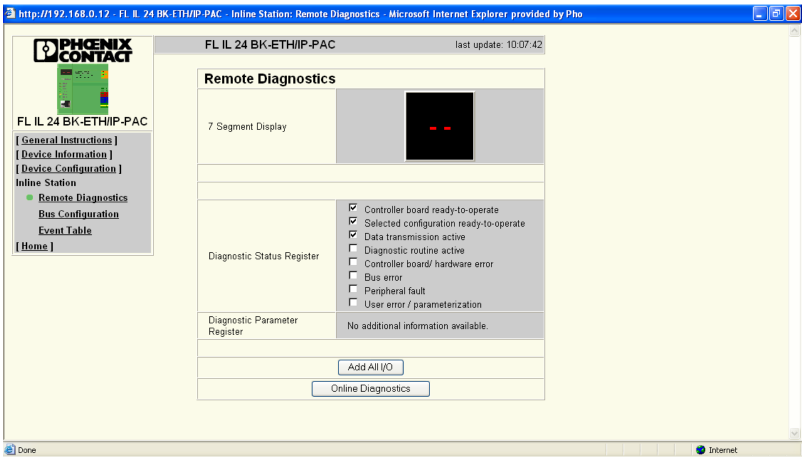


Figure 2-4 Remote Diagnostics web page

Sending an Explicit Message Method

When configuring the bus coupler of an Inline station by sending an explicit message, observe the decisions stated in "Electronic Data Sheet (EDS) File Method". The parameters listed there 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 Ethernet/IP 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 Ethernet/IP™ 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.

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 construction software can be used to send explicit messages.

2.6.2 Analog Input (AI) Modules

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 "Class Instance Editor" in the Construction Software. 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.

> 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.

2.6.3 Thermocouple and RTD Modules

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.

> 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}, $0x0A_{he}x$).

> 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 2-5 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.

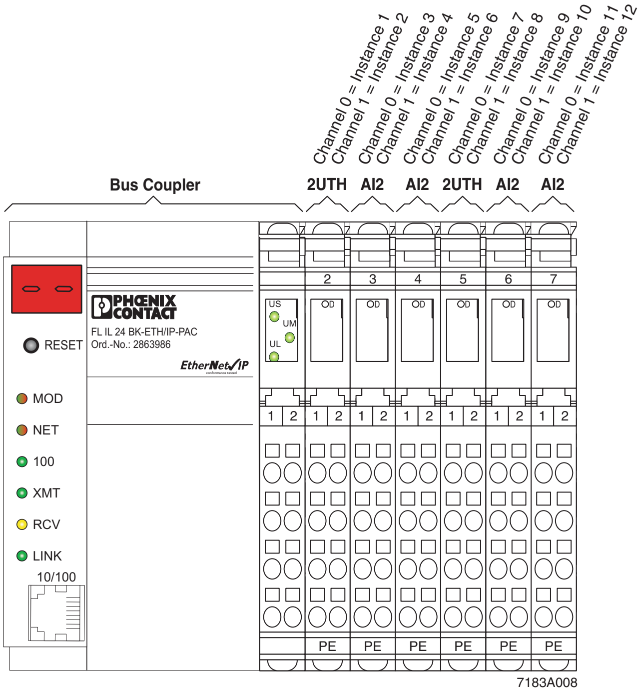


Figure 2-5 I/O station with analog input terminals and thermocouple terminals

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

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.

> AIP Object (Class Code 10_{dec}, $0x0A_{hex}$) settings are described in Appendix A of this manual.

2.6.4 Special Function Modules

General Configuration

Special function modules such as the incremental encoder, absolute encoder and the high-speed counter are configurable through the produced/consumed 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}, $0x67_{hex}$).

> 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.

2.7 Understanding I/O Memory Mapping

2.7.1 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

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

Consumed

1. Inline Control Byte
2. DOP States (values)
3. DOP Reserve Points
4. PAD Byte (optional)
5. AOP States (values)
6. AOP Reserved Words (optional)
7. AIP Configuration Words (optional)
8. Special Function
9. PCP Special Function (process & fragment)
10. 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 bus coupler 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.

> 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 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 Pad I/O 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 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 2-6 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 consists of the following I/O modules:

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

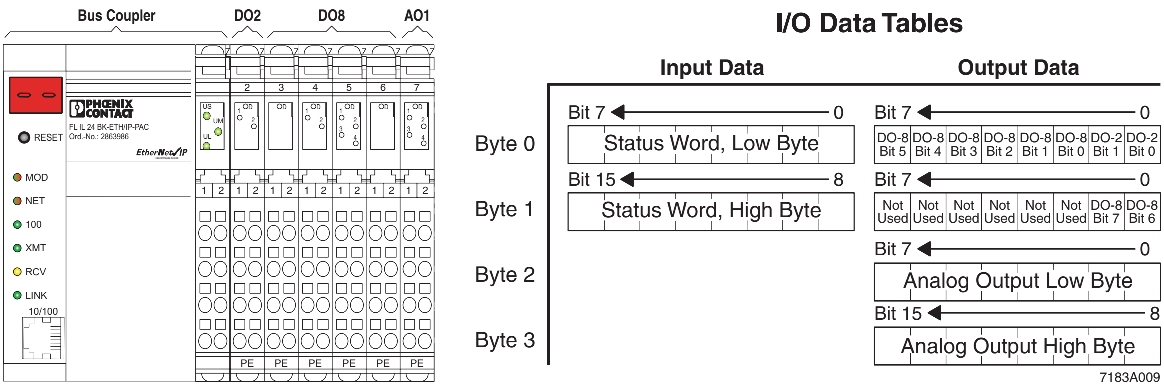
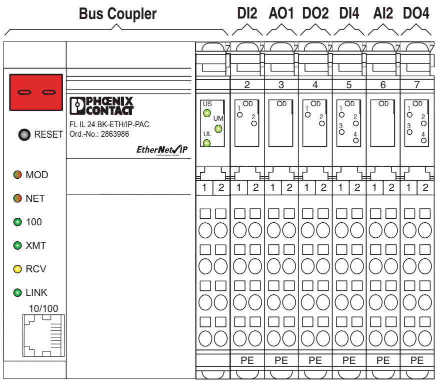


Figure 2-6 Example of an I/O table (Memory Map) consisting of analog and digital output modules

Figure 2-7 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).



I/O Data Tables

	Input Data	Output Data
Byte 0	Bit 7 ← 0: Input Status Word, Low Byte	Bit 7 ← 0: Not Used / Not Used / DO-4 Bit 3 / DO-4 Bit 2 / DO-4 Bit 1 / DO-4 Bit 0 / DO-2 Bit 1 / DO-2 Bit 0
Byte 1	Bit 15 ← 8: Input Status Word, High Byte	Bit 7 ← 0: Analog Output Low Byte
Byte 2	Bit 7 ← 0: Not Used / Not Used / DI-4 Bit 3 / DI-4 Bit 2 / DI-4 Bit 1 / DI-4 Bit 0 / DI-2 Bit 1 / DI-2 Bit 0	Bit 15 ← 8: Analog Output High Byte
Byte 3	Bit 7 ← 0: Analog Input Low Byte 1	
Byte 4	Bit 15 ← 8: Analog Input High Byte 1	
Byte 5	Bit 7 ← 0: Analog Input Low Byte 2	
Byte 6	Bit 15 ← 8: Analog Input High Byte 2	

7183A010

Figure 2-7 Example of an I/O table (Memory Map) consisting of digital & analog input and output modules

2.7.2 Reserving I/O Memory for Future System Expansion

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

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.

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.

2.8 I/O Data Transfer

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). For operational considerations see section "Maximum Connection Consideration" on page 2-22.

Implicit

An implicit connection provides a dedicated path from a producing application to one or more consuming applications and is typically used for Real-time Data Transfer.

Explicit

An explicit connection is a generic connection between two devices where a request is sent and an acknowledge is expected. It is used for configuration and data transfer.

2.8.1 I/O Scan Methods

The Ethernet/IP 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:

- Cyclic
- Change of State (COS)
- Application triggered

2.8.2 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.

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 2-8 shows how "actual input" or "latched" data is selected.

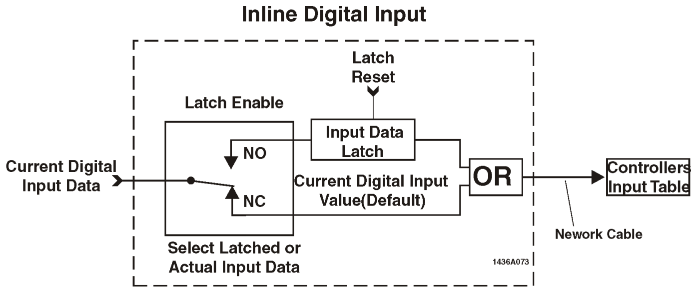


Figure 2-8 Latched or current data selection

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 produced data 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.

> 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.

Discrete Output Point (DOP) Object (Class $0x09_{hex}$)

The DOP object models discrete outputs in the Ethernet/IP 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.

Analog Input Point (AIP) Object (Class $0x0A_{hex}$)

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

Analog Output Point (AOP) Object (Class $0x0B_{hex}$)

The AOP object models analog outputs in the Ethernet/IP 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.

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
– Analog Outputs 510 instances

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 Ethernet/IP object.

– Incremental encoder
– Absolute encoder
– High Speed counter

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

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

2.9 System Operational Guidelines

2.9.1 Repeat Packet Interval (RPI) Settings

When setting up a Ethernet/IP system, care must be exhibited when setting the RPI value in the control system scanner. Depending on the vendor's implementation, this value may range from 5ms to 100's of ms in 5ms increments. The RPI value establishes the rate at which the scanner will send Ethernet/IP messages (Packets). It also establishes the max rate that the Inline station(in this case) will send messages. Though the value is set in the PC/PLC scanner is also transferred by the scanner to the BK so that the system is working on the same time base. In addition to setting the speed of the network updates, the RPI value is used to set the rate at which the scanner expects to receive back in time, the scanner will assume there is a problem, stop I/O communications and the I/O station will go into it' fault response mode.

As is true with most Ethernet/IP devices (Inline included) the CPU in the BK, splits it's time servicing the Ethernet NW, performing internal functions such as updating internal web pages, and of course scanning I/O. In larger I/O systems setting the RPI too low many overload the BK with a level of NW traffic it can not accomplish. In these cases the module will stop communicating and go into a fault state. In addition to physically larger I/O stations use of PCP communications (RS232) modules, etc requires extra processing therefore potentially higher (larger ms) RPI settings. While actual settings will vary based in the station configuration and application requirements, as a general rule of thumb.

– Configurations requiring RPI rates below 10ms should be tested in advance to confirm operation.
– Configurations requiring PCP modules should use RPI settings at a minimum of 20 ms. Settings below 20 ms should be tested in advance.

2.9.2 Maximum Connection Consideration

The module firmware supports up to 128 connections total (any mix implicit or explicit). Application considerations such as CPU loading, frequency of data updates (RPI parameter), and I/O quantity scanned will impact the actual maximum connections. Fewer connections allow faster data update rates (RPI value).For maximum I/O performance the quantity of connections should be limited to 8 or less.

2.10 Ethernet/IP and Managed vs Unmanaged Switches

Ethernet/IP is based on a producer consumer communications model where multicast Ethernet communications allow fast "report by exception" response. Switches normally take Multicast messages and transmit them out all switch ports (like a hub). This means that when using unmanaged switches, the more multicast devices (Ethernet/IP device are added to the system, all devices will see higher and higher multicast traffic, which can have 2 outcomes.

More bandwidth on the NW is used up, slowing response.

Each device has to spend more and more CPU power to review and reject messages that are not addressed to it (this is analogous to having to spend time reviewing and tossing junk ail would you Receive from the post office). With too many messages to process the device can become overloaded, fail to respond at the correct RPI rate and stop communicating. This overload situation can equally affect PC/PLC scanners as well as I/O devices.

With Ethernet/IP, unmanaged switches should only be used in small isolated systems (systems not connected to Enterprise or plan wide networks). For medium sized or any size of high speed control systems, managed switches are recommended. To manage the Multicast traffic, the Internet Group Management Protocol (IGMP) snooping function is required to be supported in the switch. When connecting a control system to a large plant or Enterprise network, Virtual Lan Switch functionality or the use of routers will typically be necessary.

The ODVA Website contains additional information.

Section 3

This section informs you about
– the diagnostic and status indicators for Inline modules
– Inline module diagnostics available over the ntework

3 Diagnostics

3.1 Diagnostic and Status Indicators

All modules are provided with LED diagnostic and status indicators for local error diagnostics.

Diagnostics

The diagnostic indicators (red/green) indicate the type and location of the error. The module is functioning correctly if all of the green LEDs are on.

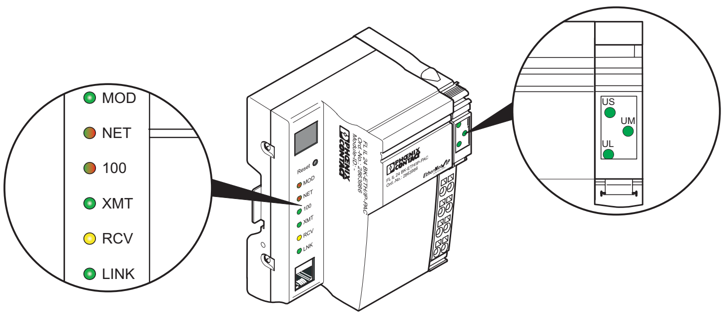
Once an error has been removed, the indicators immediately display the current status.

Status

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

> For information about the diagnostic and status indicators on each module, please refer to the module-specific data sheet.

3.1.1 LEDs on the Ethernet/IP Bus Coupler



7183A006

Figure 3-1 LEDs on the Ethernet/IP bus coupler

Diagnostics

The following states can be read on the bus coupler:

Table 3-1 Local status and diagnostic indicators

Des.	Color	Status	Meaning
Module Electronics			
UL	Green	ON	24 V supply, 7 V communications power/interface supply present
		OFF	24 V supply, 7 V communications power/interface supply not present
UM	Green	ON	24 V main circuit supply present
		OFF	24 V main circuit supply not present
US	Green	ON	24 V segment supply present
		OFF	24 V segment supply not present
Ethernet/IP Port			
MOD	Green/ Red	Green, ON	Module operating without errors
		Flashing, Green	Standby – module waiting for configuration
		Red, ON	A major error has occurred
		Flashing, Red	A recoverable error has occurred
		Flashing, Red/Green	Self test
		OFF	Module supply voltage not present
NET	Green/ Red	Green, ON	At least one connection is established
		Flashing, Green	IP parameter received, no connection established
		Red, ON	The own IP address is already in use by another device in the network
		Flashing, Red	At least one connection in the "Connection Timeout" status
		Flashing, Red/Green	Self test
		OFF	Module waiting for IP parameter assignment
100	Green	ON	Operation at 100 Mbps
		OFF	Operation at 10 Mbps (if LNK LED active)
		OFF	Operation in half duplex mode (if LNK LED active)
XMT	Green	ON	Data telegrams are being transmitted
		OFF	Data telegrams are not being transmitted
RCV	Yellow	ON	Data telegrams are being received (shows every kind of network traffic)
		OFF	Data telegrams are not being received
LNK	Green	ON	Physical network connection ready to operate
		OFF	Physical network connection interrupted or not present

3.1.2 Meaning of the 7-Segment Display

The firmware is started after the device has been connected to the power supply or the reset key has been pressed. The following sequence is displayed:

Table 3-2 Display during module startup

Display	Meaning
01	Boot Loader is started, BootP requests are being sent
bo	Firmware is extracted
02	Firmware is started
--	Operation

Table 3-3 Boot loader error messages

Display	Meaning	Remedy
17	The transfer of the firmware failed during tftp download (display changes from "03" to "17")	• Check the physical connection • Establish a point-to-point connection. • Make sure that the file (with the specified file name) exists and is in the correct directory. • Check the IP address of the tftp server • Activate the tftp server • Repeat the download
19	The tftp download was completed successfully, but the file is not a valid firmware version for the bus coupler	• Provide a valid firmware version with the previously specified file name • Repeat the download

The points under "Remedy" are recommendations; they do not all have to be carried out for every error.

Table 3-4 Firmware error messages

Display	Meaning	Remedy
80	An error occurred in the firmware	• Restart the device (power up or reset)
81	An error occurred when accessing the EEPROM.	• Restart the device (power up or reset)
82	The current configuration could not be activated	• Check the Inline status (Class 0x65, Instance 1, Attribute 4)
83	The current configuration could not be activated because the current configuration and the reference configuration are not the same	• Activate the current configuration (Class 0x64, Instance 1, Attribute 7)
bF	Bus error, the bus was stopped due to an error	• Check the bus devices • Check the Inline status (Class 0x65, Instance 1, Attribute 4)
nF	Network error, an Ethernet/IO connection in the "Timeout" state	• Check the Ethernet connection • Check the control system • Check the connection status (Class 5, Instance x, Attribute 1)
PF	I/O error	• Check the connected modules • Read out the faulty module (Class 0x65, Instance 1, Attribute 5)
nC	Not connected, no Ethernet/IP connection	• Check the Ethernet connection • Check the control system

The points under "Remedy" are recommendations; they do not all have to be carried out for every error.

If several errors occur at the same time, the error with the highest priority is shown. The priority of the individual errors is shown in the table below.

Table 3-5 Priority of error messages

Priority	Display	Meaning
1	8x	Firmware error
2	bF	Bus error, the bus was stopped due to an error
3	nF	Network error, Ethernet/IP timeout
4	PF	I/O error regarding the Inline modules
5	nC	All Ethernet/IP connections were aborted after at least one connection had been established

3.1.3 Indicators on the Supply Terminal

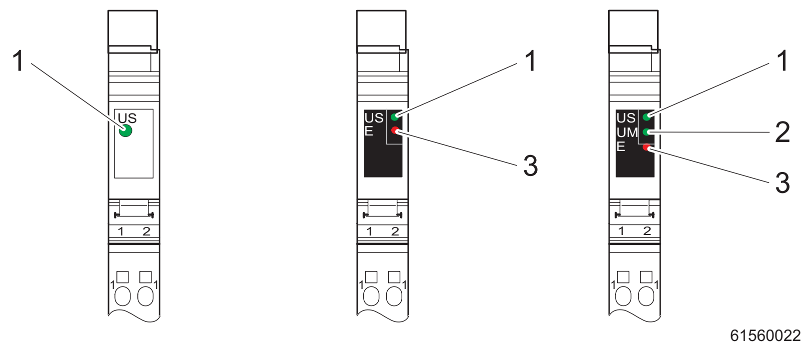


Figure 3-2 Possible indicators on supply terminals (segment terminal with and without fuse and power terminal)

Diagnostics

The following states can be read from the supply terminals:

Table 3-6 Diagnostic LED on the power terminal

LED	Color	State	Description of the LED States
UM (2)	Green	ON	24 V main circuit supply present
		OFF	Main circuit supply not present

Table 3-7 Diagnostic LED on the segment terminal

LED	Color	State	Description of the LED States
US (1)	Green	ON	24 V segment circuit supply present
		OFF	Segment circuit supply not present

Table 3-8 Additional LED on supply terminals with fuse

LED	Color	State	Description of the LED States
E (3)	Red	ON	Fuse not present or blown
		OFF	Fuse OK

On modules with fuses, the green LED indicates that the main or segment voltage is present **at the line side** of the fuse, meaning that if the green LED is on, there is voltage on the line side of the fuse. If the red LED is also on, the voltage is not present on the output side. Either no fuse is available or it is faulty.

3.1.4 Indicators on the Input/Output Modules

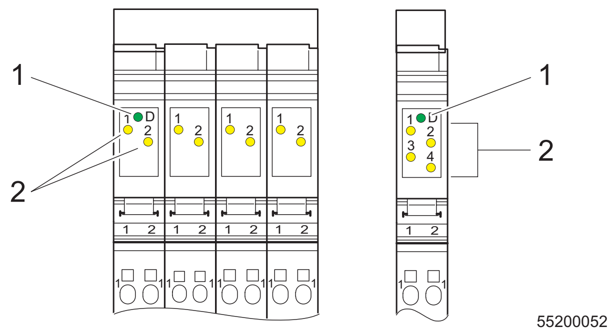


Figure 3-3 I/O module indicators

Diagnostics

The following states can be read from the I/O modules:

Table 3-9 Diagnostic LED of the I/O modules

LED	Color	State	Description of the LED States
D (1)	Green	ON	Local bus active
		Flashing:	
		0.5 Hz (slow)	Communications power present, local bus not active
		2 Hz (medium)	Communications power present, I/O error
		4 Hz (fast)	Communications power present, module in front of the flashing module has failed or the module itself is faulty; Modules following the flashing module are not part of the configuration frame.
		OFF	Communications power not present, local bus not active

Status

The status of the input or output can be read on the relevant yellow LED:

Table 3-10 Status LEDs of the I/O terminals

LED	Color	State	Description of the LED States
1, 2, 3, 4 (2)	Yellow	ON	Relevant input/output set
		OFF	Relevant input/output not set

Assignment Between Status LED and Input/Output

For the assignment of a status LED and the corresponding input/output, please refer to the module-specific data sheet.

3.1.5 Indicators on Other Inline Modules

For diagnostic and status indicators on other Inline modules (e.g., special function modules or power modules), please refer to the module-specific data sheet.

3.1.6 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 3-5 on page 3-11 shows an Inline station with a localized error at module 4.

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.

Determining Errors

Figure 3-6 on page 3-12 shows three possible scenarios for determining an error at module 4. Each of these possibilities is described below. Table 3-13 on page 3-11 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.

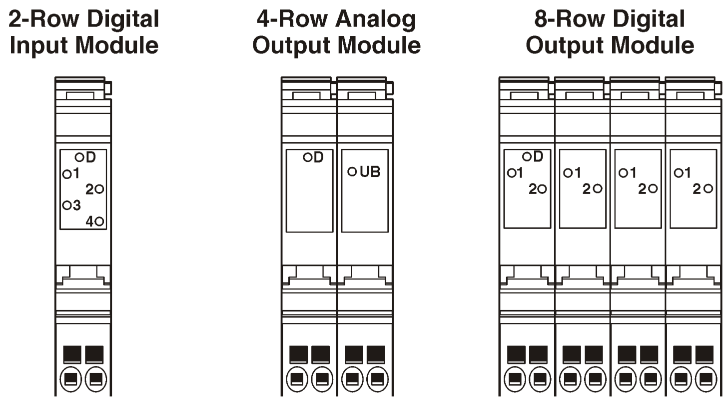


Figure 3-4 LEDs on 2-row digital, 4-row analog and 8-row digital module

Table 3-11 I/O "D" diagnostic LEDs (green)

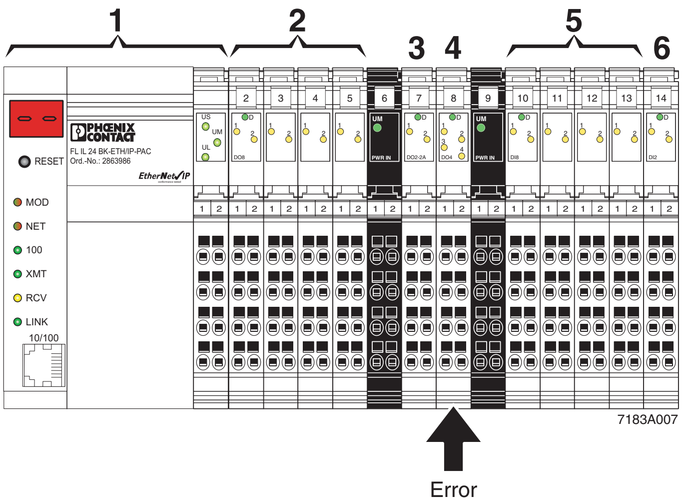
LED	Color	State	Description of the LED States
D	Green	ON	Device ready
		Flashing:	
		0.5 Hz (slow)	Communications power present, local bus not active
		2 Hz (medium)	Communications power present, I/O error
		4 Hz (fast)	Communications power present, module in front of the flashing module has failed or the module itself is faulty; Modules following the flashing module are not part of the configuration frame.
		OFF	Communications power not present, local bus not active
UB	Green	ON	I/O voltage/current is present

Table 3-12 I/O status LEDs (yellow)

LED	Color	State	Description of the LED States
1, 2, 3, 4	Yellow	ON	Relevant input/output set
		OFF	Relevant input/output not set

Table 3-13 Examples of error effects on I/O and bus

I/O Error		**Bus Error**	
Error: Short circuit in module No. 4 (IB IL 24 DO 4)		Error: Incoming bus after module No. 3 and before module No. 4 has been interrupted	
Effect:		Effect:	
Control system:	Error message to the control system	Control system:	Error can be located by the control system
Bus coupler	Indicators remain unchanged	Module No. 4:	Green "D" LED flashes at 4 Hz (fast)
Module No. 4:	Green "D" LED flashes at 4 Hz (fast)	Other modules:	Green "D" LED on all other modules flashes at 0.5 Hz (slow)
Other modules:	Remain unchanged		



Modules Used in this Example

1 Bus Coupler
2 IB IL 24 DO 8
3 IB IL 24 DO 2-2A
4 IB IL 24 DO 4
5 IB IL 24 DI 8
6 IB IL 24 DI 2

Figure 3-5 LEDs on 2-row digital, 4-row analog and 8-row digital module

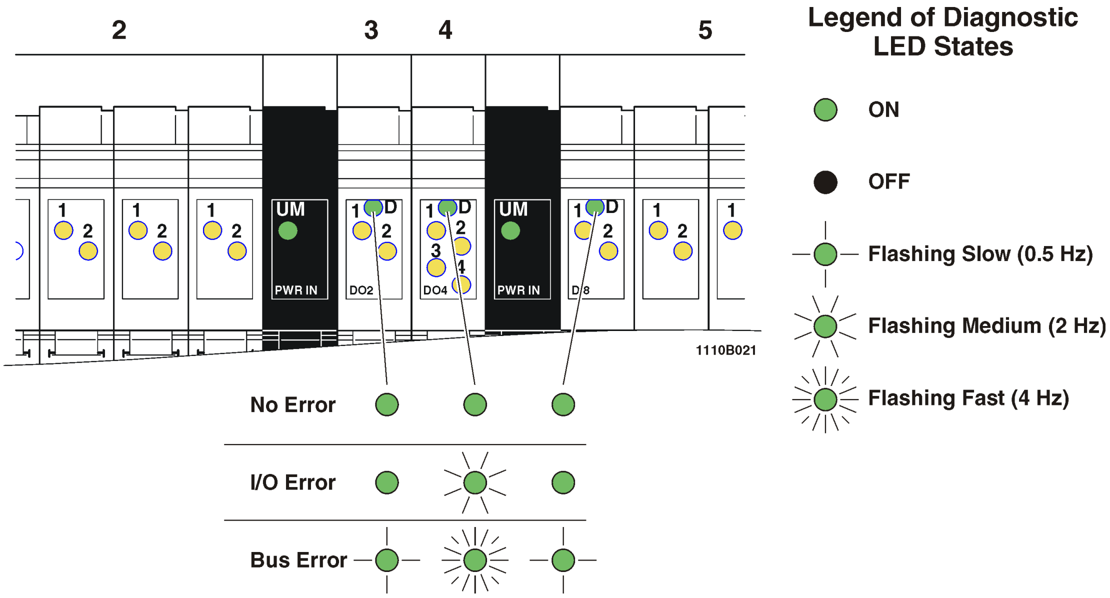


Figure 3-6 Determining errors using various diagnostic LED patterns

3.2 Available Network Diagnostics

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 produced data. The final method of retrieval is to explicitly query the attributes from the Configuration Object (Class Code $0x64_{hex}$) and Inline Object (Class Code $0x65_{hex}$).

3.2.1 Inline Status Word

By default, Inline Status Word data is made available to the user as two bytes of diagnostic data in the produced data. 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 produced data, the status word can be disabled by setting Configuration Object (Class 100_{dec}, $0x64_{hex}$) Attribute 11 "Use Inline Status" to 0, then setting Configuration Object (Class 100_{dec}, $0x64_{hex}$) Attribute 7 "Add All I/O" to 1.

3.2.2 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 MOD 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 MOD LED when in a faulted condition.

3.2.3 Bit Meanings for Inline Status Word (Byte 0)

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.

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.

Bit 2 Power Fault

The Power Fault bit will be set when any of the power supplies (U_L, U_S, U_M, Ethernet/IP) are in an under voltage condition (less than 11 V DC).

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.

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.

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).

Bit 6 Outputs Set to Preprogram Ethernet/IP 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 Ethernet/IP fault state and will no longer respond to the controller.

Bit 7

Reserved for future use.

3.2.4 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).

> 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 MOD 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.

3.2.5 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 Ethernet/IP™ implicit or 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.

3.2.6 Inline Control Byte

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

For an explanation of latching input states, refer to I/O Data Transfer described in section "I/O Data Transfer" on page 2-19.

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.

3.2.7 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.

Parameter 16

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

Parameter 17

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

Parameter 18

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

Parameter 19

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

Parameter 20

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

3.2.8 Fault/Idle State and Value

The bus coupler supports the standard Ethernet/IP 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.

Digital Output Support:

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

Analog Output Support:

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

> Appendix A will detail the DOP (Class Code $0x09_{hex}$) and AOP (Class Code $0x0B_{hex}$) fault/idle values and states.

3.2.9 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 on page 4-11.

> 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 inTable 4-2 on page 4-11. If the format has been changed, the user must refer to the module specific data sheet to determine what error code has been received.

Table 3-14 Status LEDs for the I/O terminals

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

3.2.10 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.

☞ 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 4

This section informs you about

- Communication methods for the Inline serial modules (RS-232 or RS-485/422)
- Communication methods for any other module that supports the PCP protocol
- Serial and generic PCP modules produced and consumed sizes
- I/O memory mapping
- RS-232 and RS-485/422 configuration brief

4 Serial and Other PCP Inline Modules

4.1 General

4.1.1 Exchanging Device Parameters

Communication relationships

Before data can be exchanged between two PCP devices of the INTERBUS, a logical connection must exist between the two devices. These logic connections are called communication relationships. Communication relationships are produced between application processes.

Communication relationship list (CRL)

The controller board creates a list for every PCP device where all permitted communication relationships are specified - regardless of when they are used. In this communication relationship list (CRL) the connection type and the contextual conditions, i.e., the connection parameter, under which the communication relationship can be established are stored. The CRL can be manually changed as required.

The logic connections configured in the communication relationship list ensure a smooth data exchange between two communication devices. The connection parameter (contextual conditions) of both communication partners are checked for compatibility before an information exchange takes place. This happens while the connection is being established.

Communication reference number (CR)

The communication relationship list is established line by line. Every permissible communication relationship contains a number, the communication reference number (CR). In this way the communication relationship is uniquely identified. The unique identification is necessary in order to enable a distinction between the individual devices. The INTERBUS controller board automatically numbers the devices during the initialization of the INTERBUS system. It assigns the numbers beginning with 2 according to the physical bus configuration.

4.1.2 Basic Service Operations

(Service) Primitives

A service is divided into individual basic service operations (primitives).

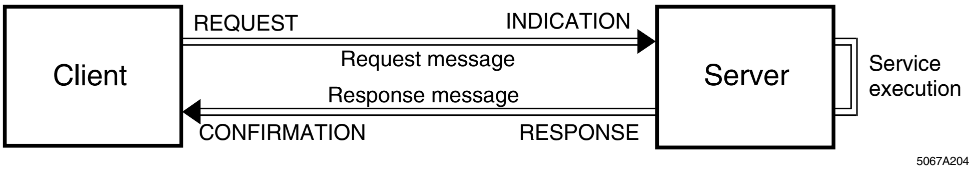


Figure 4-1 CP basic service operations (confirmed services)

Request

First of all, the client sends a service request to the server. In the application example, the automation device sends, e.g. the request to transmit a parameter value to the frequency inverter.

Indication

This request is reported as service input to the server. In the application example, the input of a parameter value is reported to the frequency inverter.

Response

The server executes the service and then sends a service response to the client. In the application example, the frequency inverter takes over the new parameter setting and sends an answer that it received the parameter value and the setting.

Confirmation

The execution of the service is reported to the client as service confirmation. In the application example, the parameter transmission to the frequency inverter is confirmed to the automation device. The bus transmits the content of the information in the form of a PDU message (Protocol Data Unit).

4.1.3 Communication Phases

The communication-oriented communication is divided into three phases:

– Connection establishment
– Data transfer
– Connection release

Connection establishment phase

In the connection establishment phase a PCP device, acting as a client, tries to establish communication with another PCP device, acting as a server. During this process, the contextual conditions (connection parameters) specified in the communication relationship lists for both devices are checked. If the context conditions correspond, the data transfer phase is started. Otherwise the connection establishment attempt is aborted with an error message.

Data transfer phase

In the data transfer phase the PCP devices exchange data subject to the context conditions. The connection remains open until it is specifically aborted or aborted after a communication error.

Connection abort phase

Once the data exchange is complete, the connection can be aborted using the connection abort function. In the event of a communication error, the connection is aborted automatically. Data can now only be exchanged after a new connection has been established.

4.1.4 Description of the Communication Services

At the beginning, every PCP service is described here with its basic functions and the corresponding parameters. Later on the services are only mentioned without explicit description.

Command Code / Message Code

Every service and every message is identified via a specified code.

Parameter Counter

Indicates the number of the subsequent data words, each of 2 bytes. If there is only a 1 byte parameter in a line then this is also counted.

Invoke ID

Invoke ID does not apply to this PCP version 00. The value is therefore always 00.

Communication Reference (CR)

CR indicates the communication reference that is set for the communication relationship between host controller board and the remote device. The standard setting for the first PCP module is CR=2, for the second PCP module CR=3, etc.

Basic Service Request

Service
Modules
Index (LSB)
Index (MSB)
Subindex
Data [0]
Data [x]

Service Request w/Invoke ID

Service
Invoke ID
Modules
Index (LSB)
Index (MSB)
Subindex
Length
Data [0]
Data [x]

Basic Service Response Success

Service
Status
Length
Data [0]
Data [x]

Basic Service Response Success w/Invoke ID

Service
Invoke ID
Modules
Index (LSB)
Index (MSB)
Subindex
Length
Data [0]
Data [x]

Basic Service Error Response

Service
Status
Error Class
Error Code
Additional Error (LSB)
Additional Error (MSB)

Service w/Invoke ID Error Response

Service
Invoke ID
Status
Error Class
Error Code
Additional Error (LSB)
Additional Error (MSB)

Basic Service Abort Response

Service
Status
Abort ID
Abort Code
Abort Reason [0]
Abort Reason [1]

Service w/Invoke ID Abort Response

Service
Invoke ID
Status
Abort ID
Abort Code
Abort Reason [0]
Abort Reason [1]

Basic Service Reject Response

Service
Status
PDU Type
Reject Code
Additional Error (LSB)
Additional Error (MSB)

Service w/Invoke ID Reject Response

Service
Invoke ID
Status
PDU Type
Reject Code
Additional Error (LSB)
Additional Error (MSB)

Supported Services

Code (hex)	Service
00	NOP
01	PCP Read
02	PCP Write
03	Read PDU Size
09	PCP Read w/Invoke ID
10	PCP Write w/Invoke ID

4.2 Communications Methods

Communications to an Inline serial module and to a generic PCP Inline module can be accomplished in two main ways each. The type of module being used, either serial or generic PCP, determines what types of communication methods are available to the user.

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 Ethernet/IPTM scanner and/or personal preference.

When using the Inline RS-232 or RS-485/422 modules, simplified methods 1 or 2 can be used. If using any (including serial) that supports PCP, methods 3 or 4 must be used.

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

An example of a "Generic" PCP Inline module is the:
- ASI MA IB IL Inline AS-I Gateway

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 bus coupler can be powered 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 auto configuration switch is ON, at next power-up the BKwill erase any custom settings described in the note 3 above. The bus coupler can hold up to 64 bytes of incoming and 64bytes of outgoing data before data will overflow internal buffers.

4.2.1 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).

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.

Format of Fragmented Serial Output Data

Byte 0: Control Byte

Table 4-1 Control byte

7	6	5	4	3	2	1	0
Reserved	Reserved	ReceiveAck.	Transmit Request	Reset Request	Number of Data BytesReceived (Per Fragment)		

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.

Format of Fragmented Serial Input Data

Produced Response

Byte 0: (See Table 4-2)

Bytes 1 thru 7: Data-block, if necessary

Table 4-2 Status byte

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

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 resync 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.

Serial Fragmentation Examples

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

1. Fragmented Read
2. Fragmented Write
3. Error Handling, Communications Backplane Break
4. Error Handling, Host Communications Loss

1. Fragmented Read Example

Table 4-3 on page 4-12 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 4-3 Serial fragmented read example

	Input Data Table (Event Sequence for Byte 0 thru 7)								Output Data Table (Event Sequence for Byte 0 thru 7)								
	Byte							Byte	Byte							Byte	
	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	
								Event 1 ->	Con-trol	TX Data 1	TX Data 2	TX Data 3	TX Data 4	TX Data 5	TX Data 6	TX Data 7	O:1.1-O:1.4
									0x00	XX	XX	XX	XX	XX	XX	XX	
I:1.1 - I:1.4	Status	RX Data 1	RX Data 2	RX Data 3	RX Data 4	RX Data 5	RX Data 6	RX Data 7	**<- Event 2**								
	0x00	XX	XX	XX	XX	XX	XX	XX									
								Event 3 ->	Con-trol	TX Data 1	TX Data 2	TX Data 3	TX Data 4	TX Data 5	TX Data 6	TX Data 7	O:1.1-O:1.4
									0x00	XX	XX	XX	XX	XX	XX	XX	
I:1.1 - I:1.4	Status	RX Data 1	RX Data 2	RX Data 3	RX Data 4	RX Data 5	RX Data 6	RX Data 7	**<- Event 4**								
	0x27	0x48 "H"	0x65 "e"	0x6C "l"	0x6C "l"	0x6F "o"	0x20 " "	0x57 "W"									
								Event 5 ->	Con-trol	TX Data 1	TX Data 2	TX Data 3	TX Data 4	TX Data 5	TX Data 6	TX Data 7	O:1.1-O:1.4
									0x20	XX	XX	XX	XX	XX	XX	XX	
I:1.1 - I:1.4	Status	RX Data 1	RX Data 2	RX Data 3	RX Data 4	RX Data 5	RX Data 6	RX Data 7	**<- Event 6**								
	0x05	0x6F "o"	0x72 "r"	0x6C "l"	0x64 "d"	0x21 "!"	XX	XX									
								Event 7 ->	Con-trol	TX Data 1	TX Data 2	TX Data 3	TX Data 4	TX Data 5	TX Data 6	TX Data 7	O:1.1-O:1.4
									0x00	XX	XX	XX	XX	XX	XX	XX	
I:1.1 - I:1.4	Status	RX Data 1	RX Data 2	RX Data 3	RX Data 4	RX Data 5	RX Data 6	RX Data 7	**<- Event 8**								
	0x00	XX	XX	XX	XX	XX	XX	XX									

DOWNLOADED FROM WWW.SCATTS.CO.UK

The following paragraphs explain those events listed in Table 4-3.

Table 4-4 through Table 4-10 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 4-3.

Event 1

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

Table 4-4 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

Event 2

Table 4-5 shows the slaves response to the idle state by replying with a 0x00.

Table 4-5 Slave to master idle response

Status	RXData 1	RXData 2	RXData 3	RXData 4	RXData 5	RXData 6	RXData 7
0x00	XX	XX	XX	XX	XX	XX	XX

Event 3

Master still sending the idle message.

Event 4

Table 4-6 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 4-6 Slave to master indication that new data has been received

Status	RXData 1	RXData 2	RXData 3	RXData 4	RXData 5	RXData 6	RXData 7
0x27	0x48 "H"	0x65 "e“	0x6C "l"	0x6C "l"	0x6F "o"	0x20 " "	0x57 "W"

Event 5

Table 4-7 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 4-7 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

Event 6

Table 4-8 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 4-8 Slave to master indication that more data is present

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

Event 7

Table 4-9 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 4-9 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

Event 8

Table 4-10 shows the slave to master indication that no more data is present.

Table 4-10 Slave to master, no more data indication

Status	RXData 1	RXData 2	RXData 3	RXData 4	RXData 5	RXData 6	RXData 7
0x00	XX	XX	XX	XX	XX	XX	XX

2. Fragmented Write Example

Table 4-11 on page 4-15 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 4-11 Serial fragmented write example

	Input Data Table (Event Sequence for Byte 0 thru 7)								Output Data Table (Event Sequence for Byte 0 thru 7)								
	Byte 0	1	2	3	4	5	6	Byte 7	Byte 0	1	2	3	4	5	6	Byte 7	
								Event 1 ->	Con-trol	TX Data 1	TX Data 2	TX Data 3	TX Data 4	TX Data 5	TX Data 6	TX Data 7	O:1.1-O:1.4
									0x00	XX	XX	XX	XX	XX	XX	XX	
I:1.1 - I:1.4	Status	RX Data 1	RX Data 2	RX Data 3	RX Data 4	RX Data 5	RX Data 6	RX Data 7	**<- Event 2**								
	0x00	XX	XX	XX	XX	XX	XX	XX									
								Event 3 ->	Con-trol	TX Data 1	TX Data 2	TX Data 3	TX Data 4	TX Data 5	TX Data 6	TX Data 7	O:1.1-O:1.4
									0x17	0x48 "H"	0x65 "e"	0x6C "l"	0x6C "l"	0x6F "o"	0x20 " "	0x57 "W"	
I:1.1 - I:1.4	Status	RX Data 1	RX Data 2	RX Data 3	RX Data 4	RX Data 5	RX Data 6	RX Data 7	**<- Event 4**								
	0x10	XX	XX	XX	XX	XX	XX	XX									
								Event 5 ->	Con-trol	TX Data 1	TX Data 2	TX Data 3	TX Data 4	TX Data 5	TX Data 6	TX Data 7	O:1.1-O:1.4
									0x05	0x6F "o"	0x72 "r"	0x6C "l"	0x64 "d"	0x21 "!"	XX	XX	
I:1.1 - I:1.4	Status	RX Data 1	RX Data 2	RX Data 3	RX Data 4	RX Data 5	RX Data 6	RX Data 7	**<- Event 6**								
	0x00	XX	XX	XX	XX	XX	XX	XX									
								Event 7 ->	Con-trol	TX Data 1	TX Data 2	TX Data 3	TX Data 4	TX Data 5	TX Data 6	TX Data 7	O:1.1-O:1.4
									0x00	XX	XX	XX	XX	XX	XX	XX	
I:1.1 - I:1.4	Status	RX Data 1	RX Data 2	RX Data 3	RX Data 4	RX Data 5	RX Data 6	RX Data 7	**<- Event 8**								
	0x00	XX	XX	XX	XX	XX	XX	XX									

The following paragraphs explain in detail those events listed in Table 4-11.

Table 4-12 through Table 4-17 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 4-11.

Event 1

Table 4-12 shows the transmission from a master to slave when the serial fragmentation is in "Idle" mode.

Table 4-12 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

Event 2

Table 4-13 shows the slave's response to the idle state by replying with a 0x00.

Table 4-13 Slave to master idle response

Status	RXData 1	RXData 2	RXData 3	RXData 4	RXData 5	RXData 6	RXData 7
0x00	XX	XX	XX	XX	XX	XX	XX

Event 3

Table 4-14 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 4-14 Master to slave data transmission

Status	RXData 1	RXData 2	RXData 3	RXData 4	RXData 5	RXData 6	RXData 7
0x27	0x48 "H"	0x65 "e"	0x6C "l"	0x6C "l"	0x6F "o"	0x20 " "	0x57 "W"

Event 4

Table 4-15 shows that the bus coupler 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 4-15 Slave to master acknowledge of data transmission

Status	RXData 1	RXData 2	RXData 3	RXData 4	RXData 5	RXData 6	RXData 7
0x00	XX	XX	XX	XX	XX	XX	XX

Event 5

Table 4-16 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 4-16 Master to slave indication for data transmission

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

Event 6

Table 4-17 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 4-17 Slave to master output data is "Qued" response

Status	RXData 1	RXData 2	RXData 3	RXData 4	RXData 5	RXData 6	RXData 7
0x00	XX	XX	XX	XX	XX	XX	XX

Event 7 and Event 8

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

3. Fragmented Error Handling, Loss of Inline Backplane

Communications Example

This serial fragmentation error handling example, shown in Table 4-18, 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 4-18 Serial fragmentation error example 1

	Input Data Table (Event Sequence for Byte 0 thru 7)								Output Data Table (Event Sequence for Byte 0 thru 7)								
	Byte							Byte	Byte							Byte	
	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	
								Event 1 ->	Con-trol	TX Data 1	TX Data 2	TX Data 3	TX Data 4	TX Data 5	TX Data 6	TX Data 7	O:1.1-O:1.4
									0x00	XX	XX	XX	XX	XX	XX	XX	
I:1.1 - I:1.4	Status	RX Data 1	RX Data 2	RX Data 3	RX Data 4	RX Data 5	RX Data 6	RX Data 7	**<- Event 2**								
	0x00	XX	XX	XX	XX	XX	XX	XX									
								Event 3 ->	Con-trol	TX Data 1	TX Data 2	TX Data 3	TX Data 4	TX Data 5	TX Data 6	TX Data 7	O:1.1-O:1.4
									0x00	XX	XX	XX	XX	XX	XX	XX	
I:1.1 - I:1.4	Status	RX Data 1	RX Data 2	RX Data 3	RX Data 4	RX Data 5	RX Data 6	RX Data 7	**<- Event 4**								
	0x27	0x48 "H"	0x65 "e"	0x6C "l"	0x6C "l"	0x6F "o"	0x20 " "	0x57 "W"									
								Event 5 ->	A data path is broken on the Inline Station's communications back-plane during a read								O:1.1-O:1.4
								Event 6 ->	Con-trol	TX Data 1	TX Data 2	TX Data 3	TX Data 4	TX Data 5	TX Data 6	TX Data 7	O:1.1-O:1.4
									0x20	XX	XX	XX	XX	XX	XX	XX	
I:1.1 - I:1.4	Status	RX Data 1	RX Data 2	RX Data 3	RX Data 4	RX Data 5	RX Data 6	RX Data 7	**<- Event 7**								
	0x05	0x6F "o"	0x72 "r"	0x6C "l"	0x64 "d"	0x21 "!"	XX	XX									
								Event 8 ->	Con-trol	TX Data 1	TX Data 2	TX Data 3	TX Data 4	TX Data 5	TX Data 6	TX Data 7	O:1.1-O:1.4
									0x00	XX	XX	XX	XX	XX	XX	XX	
I:1.1 - I:1.4	Status	RX Data 1	RX Data 2	RX Data 3	RX Data 4	RX Data 5	RX Data 6	RX Data 7	**<- Event 9**								
	0x00	XX	XX	XX	XX	XX	XX	XX									

DOWNLOADED FROM WWW.SCATTS.CO.UK

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 re-established thus data continues to be received.

4. 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.

4.2.2 Method 2: Transfer of Serial Data Using Explicit Messages

> The ability to send explicit messages is a function of the Ethernet/IP 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.

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 Ethernet/IP 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_{dec} (Get Attribute Single)

Class = 106_{dec} (Serial Communications Object)

Instance = 1 (In this case the 1^{st} occurrence of a serial module)

Attribute = 7 (Receive data)

> Instance is determined by the physical location on the Inline station. The 1^{st} instance will be assigned to the serial module (RS-232 or RS-485/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.

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 modules present. This header may include such information as: transaction ID, command, nodes address and confirmation (positive or negative).

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_{dec} (Set Attribute Single)

Class = 106_{dec} (Serial Communications Object)

Instance = 1 (In this case the 1^{st} occurrence of a serial module)

Attribute = 8 (Transmit data)

Instance is determined by the physical location on the Inline station. The 1^{st} instance will be assigned to the serial module (RS-232 or RS-485/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.

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.

4.2.3 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).

> 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 "sub network".

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.

> Information pertaining to the supported indexes and subindexes can be found in the module-specific data sheet and/or manual.

For each PCP Inline module, by default, 8 bytes (1 fragment) are added to the Ethernet/IP 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 module-specific 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:

1. Request (Output Data)

Byte 0	Service
Byte 1	Module-number
Byte 2	Index low
Byte 3	Index high
Byte 4	Subindex
Byte 5	Length
Bytes 6 thru N	Data-block, if necessary

2. Successful Response: (Input Data)

Byte 0	Service
Byte 1	Status
Byte 2	Length
Bytes 3 thru N	Data-block, if necessary

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

3. Errored Produced Response: (Input Data)

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

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" relationshipbetween 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.

Fragment Types

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

1. Start fragment
2. Middle fragment
3. Last fragment
4. Abort/Error fragment

1. 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

Byte 0, Service Definition of the Start Fragment (Shown in Table 4-19)

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 4-19 Service byte 0 definition of the start fragment

7	6	5	4	3	2	1	0
Reserved	0	0	No Fragment/ Fragment	PCP Service			

Bits 6 thru 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

Bits 3 thru 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

2. 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

Byte 0, Service Definition of a Middle Fragment (Shown in Table 4-20)

Bit 7	Request/Response. See start fragment for definition.
Bits 6 thru 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 4-20 Service byte definition of the middle fragment

7	6	5	4	3	2	1	0
Request/ Response	0	1	No Frag- ment/ Fragment	Fragment Number (0x01 - 01F)			

3. 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

Byte 0, Service Definition of the Last Fragment (Shown in Table 4-21)

Bit 7	Request/Response. See start fragment for definition.
Bits 6 thru 5	For a last fragment, bits will always be 10.
	10 = last fragment
Bits 4 thru 0	Reserved

Table 4-21 Service byte definition of the last fragment

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

4. Abort/Error fragment:

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

Byte 0, Service Definition of a PCP Abort/Error Fragment (Shown in Table 4-22)

Bit 7 Request/Response. See start fragment for definition.

0 = Request

1 = Response

Table 4-22 Service byte definition of the PCP abort/error fragment

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

Bits 6 thru 5 Fragment Type. For an abort/error fragment, these bits will always be an 11.

11= Abort / Error fragment

Bits 4 thru 0 Reserved

Byte 1, Status Definition of an Abort/Error Fragment (Shown in Table 4-23)

– 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 4-23 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 Chan-nel Busy	PCPError	Module Comm Error

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.

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:

1. Fragmented Read
2. Fragmented Write
3. Error Handling, Communications Backplane Break
4. Error Handling, Host Communications Loss

1. Fragmentation Read Example

Table 4-24 on page 4-27 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 4-24 I/O events for a read sequence using PCP fragmentation

	Input Data Table (Event Sequence for Byte 0 thru 7)								Output Data Table (Event Sequence for Byte 0 thru 7)								
	Byte							Byte	Byte							Byte	
	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	
								Event 1 ->	Ser-vice	Not used	Not used	Not used	Not used	Not used	Not used	Not used	O:1.1-O:1.4
									0x00	XX	XX	XX	XX	XX	XX	XX	
I:1.1 - I:1.4	Ser-vice	Not used	Not used	Not used	Not used	Not used	Not used	Not used	**<- Event 2**								
	0x00	XX	XX	XX	XX	XX	XX	XX									
								Event 3 ->	Ser-vice	Mod-ule No.	Index Low	Index High	Sub-Index	Length	Not used	Not used	O:1.1-O:1.4
									0x01	0x01	0xE0	0x5F	0x00	0x00	XX	XX	
I:1.1 - I:1.4	Ser-vice	Status	Length	D0	D1	D2	D3	D4	**<- Event 4**								
	0x91	0x00	0x10	0x01	0x02	0x03	0x04	0x05									
								Event 5 ->	Ser-vice	Not used	Not used	Not used	Not used	Not used	Not used	Not used	O:1.1-O:1.4
									0x91	XX	XX	XX	XX	XX	XX	XX	
I:1.1 - I:1.4	Ser-vice	D5	D6	D7	D8	D9	D10	D11	**<- Event 6**								
	0xA1	0x06	0x07	0x08	0x09	0x0A	0x0B	0x0C									
								Event 7 ->	Ser-vice	Not used	Not used	Not used	Not used	Not used	Not used	Not used	O:1.1-O:1.4
									0xA1	XX	XX	XX	XX	XX	XX	XX	
I:1.1 - I:1.4	Ser-vice	D12	D13	D14	D15	Not used	Not used	Not used	**<- Event 8**								
	0xC0	0x0D	0x0E	0x0F	0x10	XX	XX	XX									
								Event 9 ->	Ser-vice	Not used	Not used	Not used	Not used	Not used	Not used	Not used	O:1.1-O:1.4
									0xC0	XX	XX	XX	XX	XX	XX	XX	

DOWNLOADED FROM WWW.SCATTS.CO.UK

Table 4-24 I/O events for a read sequence using PCP fragmentation (Continued)

Input Data Table (Event Sequence for Byte 0 thru 7)									Output Data Table (Event Sequence for Byte 0 thru 7)								
								Event 10 ->	Ser-vice	Not used	Not used	Not used	Not used	Not used	Not used	Not used	O:1.1-O:1.4
									0x00	XX	XX	XX	XX	XX	XX	XX	
I:1.1 - I:1.4	Status	Not used	Not used	Not used	Not used	Not used	Not used	Not used	**<- Event 11**								
	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00									

In the following paragraphs the events in Table 4-24 will be explained in detail.

Table 4-25 through Table 4-35 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 4-24.

Event 1

Table 4-25 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 4-25 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

Event 2

Table 4-26 shows the slaves response to the idle state by replying with a 0x00 no action acknowledge.

Table 4-26 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

Event 3

Table 4-27 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 4-27 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

Event 4

Table 4-28 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

> It is 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 4-28 Slave to master, response to the read service

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

Event 5

Table 4-29 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 4-29 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

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 4-30 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
Bits 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 4-30 Slave to master, replay with first middle fragment

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

Event 7

Table 4-31 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 4-31 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

Event 8

Table 4-32 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 4-11. 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 = 1	Reserved

Table 4-32 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

Event 9

Table 4-33 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 4-33 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

Events 10 and 11

Knowing that the last fragment was received the master issues an idle service to the slave as shown in Table 4-34 and the slave respond with it's reply as shown in Table 4-35.

Table 4-34 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

Table 4-35 Slave to saster 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

2. Fragmented Write Example

Table 4-36 on page 4-33 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 4-36 I/O events for a write sequence using PCP fragmentation

	Input Data Table (Event Sequence for Byte 0 thru 7)								Output Data Table (Event Sequence for Byte 0 thru 7)								
	Byte 0	1	2	3	4	5	6	Byte 7	Byte 0	1	2	3	4	5	6	Byte 7	
								Event 1 ->	Ser-vice	Not used	Not used	Not used	Not used	Not used	Not used	Not used	O:1.1-O:1.4
									0x00	XX	XX	XX	XX	XX	XX	XX	
I:1.1 - I:1.4	Ser-vice	Not used	Not used	Not used	Not used	Not used	Not used	Not used	**<- Event 2**								
	0x00	XX	XX	XX	XX	XX	XX	XX									
								Event 3 ->	Ser-vice	Mod-ule No.	Index Low	Index High	Sub-Index	Length	D0	D1	O:1.1-O:1.4
									0x12	0x01	0xE0	0x5F	0x00	0x10	0x01	0x02	
I:1.1 - I:1.4	Ser-vice	Not used	Not used	Not used	Not used	Not used	Not used	Not used	**<- Event 4**								
	0x12	XX	XX	XX	XX	XX	XX	XX									
								Event 5 ->	Ser-vice	D2	D3	D4	D5	D6	D7	D8	O:1.1-O:1.4
									0x21	0x03	0x04	0x05	0x06	0x07	0x08	0x09	
I:1.1 - I:1.4	Ser-vice	Not used	Not used	Not used	Not used	Not used	Not used	Not used	**<- Event 6**								
	0x21	XX	XX	XX	XX	XX	XX	XX									
								Event 7 ->	Ser-vice	D9	D10	D11	D12	D13	D14	D15	O:1.1-O:1.4
									0x40	0x09	0x0A	0x0B	0x0C	0x0D	0x0E	0x0F	
I:1.1 - I:1.4	Ser-vice	Status	Not used	Not used	Not used	Not used	Not used	Not used	**<- Event 8**								
	0x82	0x00	XX	XX	XX	XX	XX	XX									
								Event 9 ->	Ser-vice	Not used	Not used	Not used	Not used	Not used	Not used	Not used	O:1.1-O:1.4
									0x82	XX	XX	XX	XX	XX	XX	XX	

Table 4-36 I/O events for a write sequence using PCP fragmentation (Continued)

	Input Data Table (Event Sequence for Byte 0 thru 7)								Output Data Table (Event Sequence for Byte 0 thru 7)								
								Event 10 ->	Ser-vice	Not used	Not used	Not used	Not used	Not used	Not used	Not used	O:1.1-O:1.4
									0x00	XX	XX	XX	XX	XX	XX	XX	
I:1.1 - I:1.4	Status	Not used	Not used	Not used	Not used	Not used	Not used	Not used	**<- Event 11**								
	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00									

DOWNLOADED FROM WWW.SCATTS.CO.UK

In the following paragraphs the events in Table 4-36 will be explained in detail.

Table 4-37 through Table 4-47 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 4-36.

Event 1

Table 4-37 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 4-37 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

Event 2

Table 4-38 shows the slaves response to the idle state by replying with a 0x00 no action acknowledge.

Table 4-38 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

Event 3

Table 4-39 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 3 thru 0	0x02	Write service
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	0x10	16 bytes
Bytes 6 and 7	XX	First 2 data bytes

Table 4-39 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

Event 4

Table 4-40 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 4-40 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

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 4-41 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 4-41 Master to slave, sending the 1st middle fragment

Service	D2	D3	D4	D5	D6	D7	D8
0x21	0x03	0x04	0x05	0x06	0x07	0x08	0x9

Event 6

Table 4-42 shows the slave to master acknowledgment. The service byte reflection indicates that the 1st middle fragment data was received and processed.

Table 4-42 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

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 4-43 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 4-43 Master to slave last fragment

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

Event 8

Table 4-44 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 4-44 Slave to master last fragment acknowledge

Service	Not used	Not used	Not used	Not used	Not used	Not used	Not used
0x82	XX	XX	XX	XX	XX	XX	XX

Event 9

Table 4-45 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 4-45 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

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 4-46 and the slave respond with its reply as shown in Table 4-47.

Table 4-46 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

Table 4-47 Slave to master response to idle

Service	Not used	Not used	Not used	Not used	Not used	Not used	Not used
0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00

3. Fragmented Error Handling, Loss of Inline Backplane

Communications

This PCP fragmentation error handling example, shown in Table 4-48, will show how the fragmentation will react during a break in the communications path on the Inline Station's backplane.

Table 4-48 Local communications error sequence using PCP fragmentation

	Input Data Table (Event Sequence for Byte 0 thru 7)								Output Data Table (Event Sequence for Byte 0 thru 7)								
	Byte 0	1	2	3	4	5	6	Byte 7	Byte 0	1	2	3	4	5	6	Byte 7	
								Event 1 ->	Ser-vice	Not used	Not used	Not used	Not used	Not used	Not used	Not used	O:1.1-O:1.4
									0x00	XX	XX	XX	XX	XX	XX	XX	
I:1.1 - I:1.4	Ser-vice	Not used	Not used	Not used	Not used	Not used	Not used	Not used	**<- Event 2**								
	0x00	XX	XX	XX	XX	XX	XX	XX									
								Event 3 ->	Ser-vice	Mod-ule No.	Index Low	Index High	Sub-Index	Lengt h	Not used	Not used	O:1.1-O:1.4
									0x01	0x01	0xE0	0x5F	0x00	0x00	XX	XX	
I:1.1 - I:1.4	Ser-vice	Status	Length	D0	D1	D2	D3	D4	**<- Event 4**								
	0x91	0x00	0x10	0x01	0x02	0x03	0x04	0x05									
								Event 5 ->	Ser-vice	Not used	Not used	Not used	Not used	Not used	Not used	Not used	O:1.1-O:1.4
									0x91	XX	XX	XX	XX	XX	XX	XX	
I:1.1 - I:1.4	Ser-vice	D5	D6	D7	D8	D9	D10	D11	**<- Event 6**								
	0xA1	0x06	0x07	0x08	0x09	0x0A	0x0B	0x0C									
								Event 7 ->	Ser-vice	Not used	Not used	Not used	Not used	Not used	Not used	Not used	O:1.1-O:1.4
									0xA1	XX	XX	XX	XX	XX	XX	XX	
								Event 8 ->	**Error** – Inline Station encounters a communications path break between local I/O modules								
I:1.1 - I:1.4	Ser-vice	Status	Not used	Not used	Not used	Not used	Not used	Not used	**<- Event 9**								
	0xE0	0x01	XX	XX	XX	XX	XX	XX									
I:1.1 - I:1.4	Ser-vice	Not used	Not used	Not used	Not used	Not used	Not used	Not used	**<- Event 10**								
	0xE0	XX	XX	XX	XX	XX	XX	XX									

Table 4-48 Local communications error sequence using PCP fragmentation (Continued)

Input Data Table (Event Sequence for Byte 0 thru 7)									**Output Data Table (Event Sequence for Byte 0 thru 7)**								
								Event 11 ->	**Error** – Repaired								O:1.1- O:1.4
I:1.1 - I:1.4	Ser-vice	D12	D13	D14	D15	Not used	Not used	Not used	**<- Event 12**								
	0xC0	0x0D	0x0E	0x0F	0x10	XX	XX	XX									
								Event 13 ->	Ser-vice	Not used	Not used	Not used	Not used	Not used	Not used	Not used	O:1.1- O:1.4
									0xC0	XX	XX	XX	XX	XX	XX	XX	
I:1.1 - I:1.4	Ser-vice	Not used	Not used	Not used	Not used	Not used	Not used	Not used	**<- Event 14**								
	0x00	XX	XX	XX	XX	XX	XX	XX									
								Event 15 ->	Ser-vice	Not used	Not used	Not used	Not used	Not used	Not used	Not used	O:1.1- O:1.4
									0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	

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 its 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 4-49.

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 = 1and bit 5 = 1
Bits 4 thru 0 = 0	Reserved
Byte 1 = 1	Identifies a module communication Error
Bytes 2 thru 7	Not used

Table 4-49 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

Event 10

Data is no longer being received in the input table.

> Once the connection is re-established 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.

4. 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.

4.2.4 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 section "Communications Methods" on page 4-8.

> The ability to send explicit messages is a function of the Ethernet/IP™ 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.

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_{dec} ($0x69_{hex}$), PCP Special Function Object
Instance	Select the occurrence of the PCP device within the oobject
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_{dec} ($0x69_{hex}$), 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_{dec} ($0x69_{hex}$), PCP Special Function Object
Instance	Select the occurrence of the PCP device within the object
Attribute	14_{dec}, PCP Read Data

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.

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_{dec} ($0x69_{hex}$), PCP Special Function Object
Instance	Select the occurrence of the PCP device within the object
Attribute	9_{dec}, 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_{dec} ($0x69_{hex}$), 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_{dec} ($0x69_{hex}$), 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

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.

4.3 Serial and Generic PCP Modules Produced and Consumed Sizes

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

4.3.1 Determining Produced and Consumed Size

The bus coupler can auto-configure itself to the Inline I/O connected to it (refer to 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 EthernetI/P 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. Table 4-50 gives and example of calculating the total number of bytes produced and consumed by a single Inline RS-232 module.

Table 4-50 Calculation of a serial module's produced and consumed bytes

	Produced	Consumed
Bytes required by the RS-232 module for status and control (found in the data sheet)	2	2
	+	+
Bytes required to Rx/Tx serial data (fragmentation)	8	8
Total used by each RS232 module	10 bytes	10 bytes

4.3.2 Removing or Adding Fragmentation Data

Fragmentation data must not be removed if using methods 1 or 3 described in this section under "Communications Methods" on page 4-8.

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 Ethernet/IP I/O".

To add or remove the Serial Data (8 bytes of fragment data) from the Ethernet/IP 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 Ethernet/IP I/O".

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

Class 105

Instance X

Attribute 17 (PCP Fragment Data)

  0 = removes data

  1 = adds data

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

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

4.4.1 I/O Mapping Rules

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.

4.5 Configuration Brief for the RS-232 and RS-485/422 Modules

4.5.1 General Configuration

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 RS-232 and RS-485/422 modules. Information provided in this section must be used in conjunction with the module-specific data sheet.

In order to change any setting, Refer to the module-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 Ethernet/IP 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 RS-232 or RS-485/422 modules is as follows:

Service	Set Attribute Single, 16_{dec} ($0x10_{hex}$)
Class	Code 106_{dec} ($0x6A_{hex}$) Serial Communications object
Instance	1 (1^{st} serial module)
Attribute	12, This attribute is used to modify the baud rate
Data	08 (Refer to the RS-232 module's data sheet) Code "08" represents a baud rate of 19.2 K

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 RS-232 and RS-485/422 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.

Section 5

This section informs you about

– technical data and ordering data of the bus coupler

5 Technical Data and Ordering Data

5.1 Technical Data

General Data	
Function	Ethernet/IP bus coupler
Housing dimensions (width x height x depth)	90 mm x 116 mm x 72 mm
Permissible operating temperature (EN 60204-1)	0°C to 55°C
Permissible storage temperature (EN 60204-1)	-25°C to 85°C
Degree of protection	IP20, DIN 40050, IEC 60529
Class of protection	Class 3 VDE 0106; IEC 60536
Humidity (operation) (EN 60204-1)	5% to 90%, no condensation
Humidity (storage) (EN 60204-1)	5% to 95%, no condensation
Air pressure (operation)	80 kPa to 108 kPa, 2000 m above sea level
Air pressure (storage)	70 kPa to 108 kPa, 3000 m above sea level
Preferred mounting position	Perpendicular to a standard DIN rail
Connection to protective earth ground	The functional earth ground must be connected to the 24 V DC supply/functional earth ground connection. The contacts are directly connected to the potential jumper and FE springs on the bottom of the housing. The terminal is grounded when it is snapped onto a grounded DIN rail. Functional earth ground is only used to discharge interference.
Environmental compatibility	Free from substances which would hinder coating with paint or varnish (according to VW specification)
Resistance to solvents	Standard solvents
Weight	270 g, typical

24 V Main Supply/24 V Segment Supply	
Connection method	Spring-cage terminals
Recommended cable lengths	30 m, maximum; do not route cable through outdoor areas
Voltage continuation	Through potential routing
Special demands on the voltage supply	The supplies U_M/U_S and the bus coupler supply U_{BK} do not have the same ground potential because they are supplied by two separate power supply units.
Behavior in the event of voltage fluctuations	Voltages (main and segment supply) that are transferred from the bus coupler to the potential jumpers follow the supply voltages without delay.
Nominal value	24 V DC
Tolerance	-15%/+20% (according to EN 61131-2)
Ripple	±5%
Permissible range	19.2 V to 30 V
Current carrying capacity	8 A, maximum (total current of U_S and U_M)
Safety equipment	
Surge voltage	Input protective diodes (can be destroyed by permanent overload) Pulse loads up to 1500 V are short circuited by the input protective diode.
Polarity reversal	Parallel diodes against polarity reversal; in the event of an error the high current through the diodes causes the preconnected fuse to blow.

This 24 V area must be fused externally. The power supply unit must be able to supply 4 times (400%) the nominal current of the external fuse, to ensure that the fuse blows safely in the event of an error.

24 V Bus Coupler Supply	
Connection method	Spring-cage terminals
Recommended cable lengths	30 m, maximum; do not route cable through outdoor areas
Voltage continuation	Via potential routing U_L,U_{ANA}
Safety equipment	
Surge voltage	Input protective diodes (can be destroyed by permanent overload) Pulse loads up to 1500 V are short circuited by the input protective diode.
Polarity reversal	Serial diode in the lead path of the power supply unit; in the event of an error only a low current flows. In the event of an error the fuse in the external power supply unit does not trip. Ensure protection of 2 A by fuses through the external power supply unit.

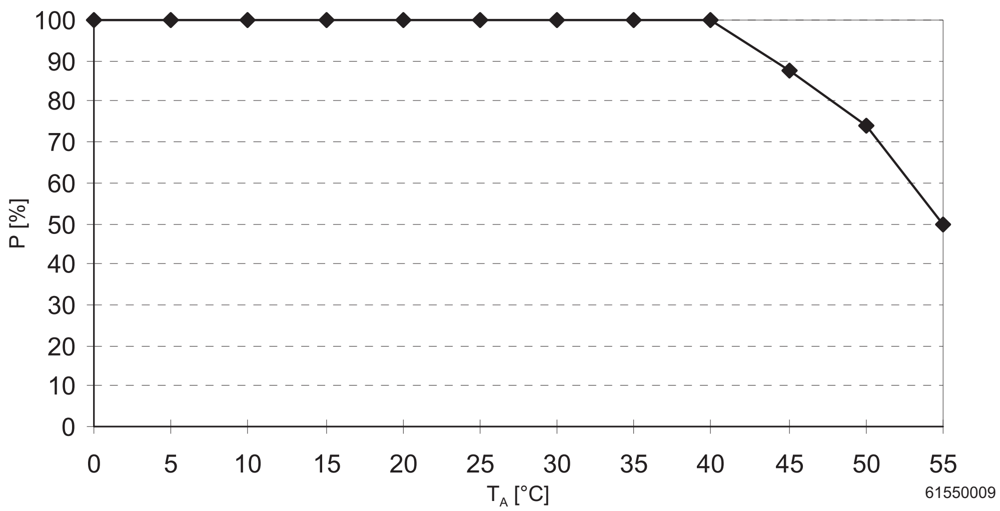
Observe the current consumption of the modules

Observe the logic current consumption of each device when configuring an Inline station. This information is given in every module-specific data sheet. The current consumption can differ depending on the individual module. The permissible number of devices that can be connected therefore depends on the specific station structure.

Nominal value	24 V DC
Tolerance	-15%/+20% (according to EN 61131-2)
Ripple	±5%
Permissible range	19.2 V to 30 V
Minimum current consumption at nominal voltage	92 mA (At no-load operation, i.e., Ethernet connected, no local bus devices connected, bus inactive)
Maximum current consumption at nominal voltage	1.5 A (Loading the 7.5 V communications power with 2 A, the 24 V analog voltage with 0.5 A)

24 V Module Supply	
- Communications Power (Potential Jumper)	
Nominal value	7.5 V DC
Tolerance	±5%
Ripple	±1.5%
Maximum output current	2 A DC (observe derating)
Safety equipment	Electronic short-circuit protection
- Analog Supply (Potential Jumper)	
Nominal value	24 V DC
Tolerance	-15%/+20%
Ripple	±5%
Maximum output current	0.5 A DC (observe derating)
Safety equipment	Electronic short-circuit protection

Derating of the Communications Power and the Analog Terminal Supply



P [%]	Loading capacity of the power supply unit for communications power and analog supply in %
T_A [°C]	Ambient temperature in °C

Power Dissipation

Formula to Calculate the Power Dissipation of the Electronics

$P_{EL} = P_{BUS} + P_{PERI}$

$$P_{EL} = 2.6\ W + (1.1 \frac{W}{A}\ x \sum_{n=0}^{a} I_{Ln}) + (0.7 \frac{W}{A}\ x \sum_{m=0}^{b} I_{Lm})$$

Where	
P_{EL}	Total power dissipation in the terminal
P_{BUS}	Power dissipation for bus operation without I/O load (permanent)
P_{PERI}	Power dissipation with I/O connected
I_{Ln}	Current consumption of the device *n* from the communications power
n	Index of the number of connected devices (n = 1 to a)
a	Number of connected devices (with communications power supply)
$\sum_{n=0}^{a} I_{Ln}$	Total current consumption of the devices from the 7.5 V communications power (2 A, maximum)
I_{Lm}	Current consumption of the device *m* from the analog supply
m	Index of the number of connected analog devices (m = 1 to b)
b	Number of connected analog devices (supplied with analog voltage)
$\sum_{m=0}^{b} I_{Ln}$	Total current consumption of the devices from the 24 V analog supply (0.5 A, maximum)

Power Dissipation/Derating

Using the maximum currents 2 A (logic current) and 0.5 A (current for analog terminals) in the formula to calculate the power dissipation when the I/O is connected gives the following result:

P_{PERI} = 2.2 W + 0.35 W = 2.55 W

2.55 W corresponds to 100% current carrying capacity of the power supply unit in the derating curves on page 5-5.

Make sure that the indicated nominal current carrying capacity in the derating curves is not exceeded when the ambient temperature is above 40°C. Corresponding with the formula, the total current carrying capacity of the connected I/O is relevant (P_{PERI}). If, for example, no current is drawn from the analog supply, the percentage of current coming from the communications power can be increased.

Example:

Ambient temperature: 55°C

Nominal current carrying capacity of the communications power and analog supply: 50% according to the diagram

I_{LLogic} = 1 A, $I_{LAnalog}$ = 0.25 A

P_{PERI} = 1.1 W + 0.175 W

P_{PERI} = 1.275 W (corresponds to 50% of 2.55 W)

Possible logic current if the analog supply is not loaded:

P_{PERI} = 1.1 W/A x I_{LLogic} + 0 W

P_{PERI} / 1.1 W/A = I_{LLogic}

I_{LLogic} = 1.275 W / 1.1 W/A

I_{LLogic} = 1.159 A

Safety Equipment	
Surge voltage (segment supply/main supply/bus coupler supply)	Input protective diodes (can be destroyed by permanent overload) Pulse loads up to 1500 V are short circuited by the input protective diode.
Polarity reversal (segment supply/main supply)	Parallel diodes against polarity reversal; in the event of an error the high current through the diodes causes the preconnected fuse to blow.
Polarity reversal (bus coupler supply)	Serial diode in the lead path of the power supply unit; in the event of an error only a low current flows. In the event of an error the fuse in the external power supply unit does not trip. Ensure protection of 2 A by fuses through the external power supply unit.

Bus Interface of the Lower-Level System Bus	
Interface	Inline local bus
Electrical isolation	No
Number of Inline terminals that can be connected	
Limited by software	63, maximum
Limited by power supply unit	Maximum logic current consumption of the connected local bus modules: $I_{max} \leq 2$ A DC

Observe the current consumption of the modules

Observe the logic current consumption of each device when configuring an Inline station. This information is given in every module-specific data sheet. The current consumption can differ depending on the individual module. The permissible number of devices that can be connected therefore depends on the specific station structure.

Interfaces	
Ethernet Interface	
Number	One
Connection format	8-pos. RJ45 socket on the bus coupler
Connection medium	Twisted pair cable with a conductor cross section of 0.14 mm^2 to 0.22 mm^2
Cable impedance	100 Ω
Transmission speed	10/100 Mbps
Maximum network segment expansion	100 m

Mechanical Tests	
Shock test according to IEC 60068-2-27	Operation: 25g, 11 ms period, half-sine shock pulse Storage/transport: 50g, 11 ms period, half-sine shock pulse
Vibration resistance according to IEC 60068-2-6	Operation/storage/transport: 5g, 150 Hz, Criterion A
Free fall according to IEC 60068-2-32	1 m

Conformance With EMC Directives

Developed according to IEC 61000-6.2

IEC 61000-4-2 (ESD)	Criterion B 6 kV contact discharge 6 kV air discharge (without labeling field) 8 kV air discharge (with labeling field in place)
IEC 61000-4-3 (radiated noise immunity)	Criterion A
IEC 61000-4-4 (burst)	Criterion B
IEC 61000-4-5 (surge)	Criterion B
IEC 61000-4-6 (conducted noise immunity)	Criterion A
IEC 61000-4-8 (noise immunity against magnetic fields)	Criterion A
EN 55011 (noise emission)	Class A

Warning

Portable radiotelephone equipment (P ≥ 2 W) must not be operated any closer than 2 m. There should be no strong radio transmitters or ISM (industrial scientific and medical) devices in the vicinity.

Approvals

Approvals	cUL 508, cUL 2279, cUL 1604 Class 1 Div 2

5.2 Ordering Data

Bus Coupler

Description	Type	Order No.
Ethernet/IP bus coupler with connector and labeling field	FL IL 24 BK ETH/IP-PAC	28 63 98 6
Connector, with color print	IB IL SCN-8-CP	27 27 60 8
Labeling field	IB IL FIELD 8	27 27 50 1

Accessories and Tools

Description	Type	Order No.
RJ45 **gray** connector set for linear cable (2 pieces)	FL PLUG RJ45 GR/2	27 44 85 6
RJ45 **green** connector set for crossed cable (2 pieces)	FL PLUG RJ45 GN/2	27 44 57 1
Double sheathed Ethernet cable	FL CAT5 HEAVY	27 44 81 4
Flexible Ethernet cable	FL CAT5 FLEX	27 44 83 0
Assembly tool for RJ45 connector	FL CRIMPTOOL	27 44 86 9
Media converter 660 nm	FL MC 10BASE-T/FO POF	27 44 51 3
Voltage supplies	QUINT-PS ... see "INTERFACE" catalog	
Keying profile (100 pcs./package)	CP-MSTB see "COMBICON" catalog	17 34 63 4
Zack markers for labeling terminals	ZB 6 ... see "CLIPLINE" catalog	
Labeling field covering one connector	IB IL FIELD 2	27 27 50 1
Labeling field covering four connectors	IB IL FIELD 8	27 27 51 5
Insert strips for IB IL FIELD 2, perforated, can be labeled using a laser printer, marker pen or CMS system (72 strips, 1 pcs./package)	ESL 62X10	08 09 49 2
Insert strips for IB IL FIELD 8, perforated, can be labeled using a laser printer, marker pen or CMS system (15 strips, 5 pcs./package)	ESL 62X46	08 09 50 2
DIN EN 50022 DIN rail, 2 meters perforated unperforated	 NS 35/ 7,5 PERF 2000MM NS 35/ 7,5 UNPERF 2000MM	 08 01 73 3 08 01 68 1
End clamp snapped on without tools (50 pcs./package)	CLIPFIX 35	30 22 21 8
End clamp fixed using screws (50 pcs./package)	E/UK	12 01 44 2
Screwdriver according to DIN 5264, blade width 3.5 mm	SZF 1-0,6X3,5	12 04 51 7

Software

Description	Type	Order No.
Factory Manager, network management software	FL SWT	28 31 04 4

Documentation

Description	Type	Order No.
"Configuring and Installing the INTERBUS Inline Product Range" user manual	IB IL SYS PRO UM E	27 43 04 8

PHOENIX CONTACT GmbH & Co. KG
Flachsmarktstr. 8
32825 Blomberg
Germany

+ 49 - (0) 52 35 - 3-00

+ 49 - (0) 52 35 - 3-4 12 00

www.phoenixcontact.com

Worldwide Locations:
www.phoenixcontact.com/salesnetwork

HOTLINE

If problems occur which cannot be solved with the help of this documentation, please contact our hotline:

+ 49 - (0) 52 35 - 3-4 18 88

factoryline-service@phoenixcontact.com

Section A

This section informs you about
- ODVA standard Discrete Input Points (DIP)
- Discrete Output Points (DOP)
- Analog Input Points (AIP)
- Analog Output Points (AOP)
- user defined Configuration, Inline Interface, Inline Module, Inline Special Function, PCP Special Function and Serial Communication objects

A Ethernet/IP Object Classes, Message Types, and Services

A 1 General

The Inline bus coupler supports CIP 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 CIP Class Services

The FL IL 24 BK ETH/IP-PAC supports the following class services and instance services:

Table A-1 CIP Class Services

Service Code	Service Name
01 (0x01)	Get_Attribute_All
02 (0x02)	Set_Attribute_All
05 (0x05)	Reset
14 (0x0E)	Get_Attribute_Single
16 (0x10)	Set_Attribute_Single

A 3 CIP Object Classes

The FL IL 24 BK ETH/IP-PAC supports the following CIP object classes:

Table A-2 CIP Object Classes

Class Code	Object Type
01 (0x01)	Identity
02 (0x02)	Router
04 (0x04)	Assembly
05 (0x05)	Connection Object
06 (0x06)	Connection Manager
08 (0x08)	Discrete Input Point
09 (0x09)	Discrete 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 Object
106 (0x6A)	Serial Object
244 (0xF4)	Port Object
245 (0xF5)	TCP/IP Interface Object
246 (0xF6)	Ethernet Link Object

A 4 Identity Object (Class Code: 01 (0x01))

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

A 4.1 Identity Object Class Attributes

Table A-3 Identity Object Class Attributes

Attribute	Access	Name	Type	Value
1	Get	Reset	UINT	1
2	Get	Max Object Instance	UINT	1
6	Get	Max. Class Identifier	UINT	7
7	Get/Set	Max. Instance Attribute	UINT	8

A 4.2 Identity Object Instance Attributes

Table A-4 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	8165
4	Get	Revision	STRUCT OF	
		Major Revision	BYTE	1
		Minor Revision	BYTE	1
5	Get	Device Status	UINT	
6	Get	Serial Number	DOUBLE	
7	Get	Product Name	STRUCT OF	
		Length	LENGTH	19
		Name	NAME	Ethernet/IP Bus Coupler
8	Get	State	USINT	
10	Get/Set	Heartbeat Interval	USINT	Heartbeat interval in seconds

Product Code - Attribute 3

The Product Code is fixed at 8165 for the FL IL 24 BK ETH/IP-PAC. 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.

A 4.3 Identity Object Common Services

Table A-5 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

Device Status – Attribute 5

Table A-6 Device Status – Attribute 5

Bit Number	Name	Meaning
0	Owned	= 0, not owned = 1, allocated
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 fault (DOP short circuit)
9	Minor Unrecoverable Fault	= 0, no fault = 1, minor unrecoverable faults (PF Fault, Fault Cycles)
10	Major Recoverable Fault	= 0, no fault = 1, major recoverable faults (Loss of +24 V)
11	Major Unrecoverable Fault	= 0, no fault = 1, major unrecoverable faults (Checksum A/D)
12 - 15	Reserved	

Serial Number – Attribute 6

The serial number is encoded in the product during the manufacturing cycle and is guaranteed to be unique across all product lines produced by Phoenix Contact.

Device Name – Attribute 7

Device Name provides a character array containing the short text string
FL IL 24 BK ETH/IP-PAC.

Device State – Attribute 8

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

Table A-7 Device State – Attribute 8

Status	Meaning	Cause	Interpretation
0	Non-existent		
1	Self test		
2	Standby		
3	Operating	Normal operating mode	
4	Major recoverable fault	– CFC Fault – Power Fault – Module Change Fault – Configuration Fault – Connection Fault	– Non-existent – Standby – Operating – Major recoverable fault
5	Major unrecoverable fault	Memory checksum error	Major unrecoverable fault

A 5 Router Object (Class Code: 02 (0x02))

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 Class Attributes

Table A-8 Router Object Class Attributes

Attribute	Access	Name	Type	Value
1	Get	Reset	UINT	2
6	Get	Max. class identifier	UINT	7
7	Get	Max. instance attribute	UINT	2

A 5.2 Router Object, Instance 1 Attributes

Table A-9 Router Object, Instance 1 Attributes

Attribute	Access	Name	Type	Value
1	Get	Router Class List	Array	12 00 00 01 00 02 00 04 00 06 00 08 00 09 00 08 00 0B 00 64 00 65 00 66 00 67 00 68 00 69 00 6A 00 F4 00 F5 00 F6
6	Get	Max. Connections	UINT	128

A 5.3 Router Object Common Services

Table A-10 Router Object Common Services

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

A 6 Assembly Object (Class Code: 04 (0x04))

The Assembly objects bind attributes of multiple objects to allow data to or from each object to be transmitted or received via a single connection.

A 6.1 Assembly Object Class Attributes

Table A-11 Assembly Object Class Attributes

Attribute	Access	Name	Type	Value
1	Get	Revision	UINT	2
2	Get	Max. Class ID	UINT	101

A 6.2 Assembly Object, Instance 100 Attributes

Table A-12 Assembly Object, Instance 100 Attributes

Attribute	Access	Name	Type	Value
3	Get	Data	See "Configuration Object Class Attributes" on page A-24	

Table A-13 Assembly Object, Instance 101 Attributes

Attribute	Access	Name	Type	Value
3	Get/Set	Config Out Data	See "Configuration Object Class Attributes" on page A-24	

A 6.3 Assembly Object Common Services

Table A-14 Assembly Object Common Services

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

Assembly Instance 100

Assembly Instance 100 is used to generate I/O product data. Assembly Instance 100 consists of a variable number of bytes that is specified via the configuration object.

Assembly Instance 101

Assembly Instance 101 is used to generate I/O data and it consists of a variable number of digital output states and a variable number of analog output values as specified by the configuration object.

A 7 Connection Object (Class Code: 05 (0x05))

The Connection objects manage the characteristics of each communication connection. The modules simultaneously support 128 CIP Class 1 and/or Class 3 connections

A 7.1 Connection Object Class Attributes

Table A-15 Connection Object Class Attributes

Attribute	Access	Name	Type	Value
1	Get	Revision	UINT	2
2	Get	Max. Instance	UINT	Number of active connections
6	Get	Max. Class Attribute	UINT	7
7	Get	Max. Instance Attribute	UINT	17

A 7.2 Connection Object, Instance 1 Attributes (Explicit Connections)

Table A-16 Connection Object, Instance 1 Attributes (Explicit Connections)

Attribute	Access	Name	Type	Value
1	Get	State	USINT	1
2	Get	Instance Type	USINT	0 = Explicit Message
3	Get	Transport Class Trigger	USINT	0x83
4	Get	Max. Instance Attribute	USINT	2
5	Get	Consumed Connection	UINT	2
6	Get	Initial Comm. Char.	USINT	0x21
7	Get	Production Size	UINT	30
8	Get	Consumed Size	UINT	35
9	Get/Set	Expected Packet Rate	UINT	Default 2500 ms
10	Get	Production Connection ID	UINT	2
11	Get	Consumed Connection ID	UINT	2
12	Get/Set	Timeout Action	USINT	3
13	Get	Production Path Length	USINT	0
14	Get	Production Path		null
15	Get	Consumed Path Length	USINT	0
16	Get	Consumed Path		null
17	Get/Set	Inhibit Time	USINT	0

A 7.3 Connection Object, Instance 2 Attributes (I/O connection)

Table A-17 Connection Object, Instance 2 Attributes (I/O connection)

Attribute	Access	Name	Type	Value
1	Get	State	USINT	1
2	Get	Instance Type	USINT	1 = I/O message
3	Get	Transport Class Trigger	USINT	0x83
6	Get	Transport Class Trigger	USINT	0x1
7	Get	Production Size	UINT	See "Configuration Object Class Attributes" on page A-24
8	Get	Consumed Size	UINT	
9	Get/Set	Expected Packet Rate		Default 2500 ms
10	Get	Production Connection ID	UINT	2
11	Get	Consumed Connection ID	UINT	2
12	Get/Set	Timeout Action	USINT	3
13	Get	Production Path Length	USINT	6
14	Get	Production Path	STRUCT OF	
		Log. Seg., Class	USINT	0x20
		Class Number	USINT	0x04
		Log. Seg., Instance	USINT	0x24
		Instance Number	USINT	0x64
		Log. Seg., Attribute	USINT	0x30
		Attribute Number	USINT	0x03
15	Get	Consumed Path Length	USINT	6
16	Get	Consumed Path	STRUCT OF	
		Log. Seg., Class	USINT	0x20
		Class Number	USINT	0x04
		Log. Seg., Instance	USINT	0x24
		Instance Number	USINT	0x65
		Log. Seg., Attribute	USINT	0x30
		Attribute Number	USINT	0x03
17	Get/Set	Inhibit Time	USINT	0

A 7.4 Connection Object Common Services

Table A-18 Connection Object Common Services

Service Code	Class	Instance	Service Name
05 (0x05)	Yes	Yes	Reset
09 (0x09)	Yes	Yes	Delete
14 (0x0E)	Yes	Yes	Get_Attribute_Single
16 (0x10)	No	Yes	Set_Attribute_Single

Connection Status – Attribute 1

Table A-19 Connection Status – Attribute 1

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

Timeout Action – Attribute 12

Table A-20 Timeout Action – Attribute 12

Connection State	Interpretation
0	Timeout (I/O messaging default)
1	Auto Delete (explicit messaging, fixed value)
2	Auto Reset
3	Deferred Delete

Timeout

In the event of a loss of an I/O connection the station will go to a fault state and will attempt to set the outputs to pre-programmed fault states. Individual digital output fault states are configured in the DOP Object (Class 09_{dec}, $0x09_{hex}$) Attribute 5 and 6. Individual analog output fault states are configured in the AOP Object (Class 11_{dec}, $0x0B_{hex}$) Attribute 9 and 11. The connection(s) must be deleted before they can be re-established. The I/O connection(s) can be deleted two ways:

1. Per connection: Setting Connection Object (Class 05_{dec}, $0x05_{hex}$) Code 09 to 1 for each connection instance.
2. All connections: Setting Identity Object (Class 01_{dec}, $0x01_{hex}$) Attribute 1 to 1.

Auto Delete

In the event of a loss of an explicit messaging connection the station will not go to a fault state and the explicit messaging connection will be automatically deleted.

Auto Reset

In the event of a loss of an I/O connection the station will not go into a fault state. The outputs will hold last state and the I/O connection will be auto deleted. When the connection is re-established the station will begin I/O update automatically.

Deferred Delete

In the event of a loss of an explicit messaging connection, the connection is not automatically deleted. The I/O connection(s) can be deleted two ways:

1. Per connection: Setting Connection Object (Class 05_{dec}, $0x05_{hex}$) Code 09 to 1 for each connection instance.
2. All connections: Setting Identity Object (Class 01_{dec}, $0x01_{hex}$) Attribute 1 to 1

A 8 Discrete Input Point (DIP) Object (Class Code: 08 (0x08))

The Discrete Input Point (DIP) Object models discrete inputs in a poduct. 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 Mask Object Class Attributes

Table A-21 DIP Mask Object Class Attributes

Attribute	Access	Name	Type	Value
1	Get	Revision	UINT	2
2	Get	Max. Object Instance	UINT	Number of DIPs
6	Get	Max. Class Identifier	UINT	7
7	Get	Max. Instance Attribute	UINT	8

A 8.2 DIP Object, Instance 1 (Number of DIPs) Attributes

Table A-22 DIP Object, Instance 1 (Number of DIPs) Attributes

Attribute	Access	Name	Type	Value
3	Get	Value	BOOL	0 == OFF, 1 == ON
4	Get	Status	BOOL	0 == OK, 1 == Fault
100	Get/Set	Enable Latched Input	BOOL	0 == OFF, 1 == Latch
101	Get/Set	Latch Level	BOOL	0 = Latch Low, 1 = Latch High

A 8.3 DIP Object Common Services

Table A-23 DIP Object Common Services

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

A 8.4 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 V DC power is not present the circuitry cannot accurately determine the state of the inputs and will set the input status bits of inputs 1 to 24. The status bits are cleared when the 24 V DC power is restored.

Enable Latch Input – Attribute 100

When set to 1, the corresponding input instance is latched at the state defined in attribute 101.

Latch Enable – 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.

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 (0x09))

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

A 9.1 DOP Mask Object Class Attributes

Table A-24 DOP Mask Object Class Attributes

Attribute	Access	Name	Type	Value
1	Get	Revision	UINT	1
2	Get	Max. Object Instance	UINT	Number of DOPs
6	Get	Max. Class Identifier	UINT	7
7	Get/Set	Max. Instance Attribute	UINT	8

A 9.2 DOP Object, Instance 1 Number of DOPs Attributes

Table A-25 DOP Object, Instance 1 Number of DOPs Attributes

Attribute	Access	Name	Type	Value
3	Get/Set	Value	BOOL	State of Output
4	Get	Status	BOOL	Status of Output
5	Get/Set	Fault State	BOOL	0 = Fault Value (default) 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

A 9.3 DOP Object Common Services

Table A-26 DOP Object Common Services

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

A 9.4 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.

Table A-27 Fault State – Attribute 5

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

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 the action to be taken if an idle condition is detected. Idle conditions occur if an I/O request is received with less than the calculated number of bytes of the Run/Idle header is used. Refer to the configuration object to determine the size of the Consume Data. An I/O request of 0 bytes is typically used to force an idle condition.

Table A-28 Idle State – Attribute 5

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

Idle Value – Attribute 8

The Fault 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 (0x0A))

The FL IL 24 BK ETH/IP-PAC supports variable analog inputs. There is a separate instance for each analog input of the device.

A 10.1 AIP Object Class Attributes

Table A-29 AIP Object Class Attributes

Attribute	Access	Name	Type	Value
1	Get	Revision	UINT	2
2	Get	Max. Object Instance	UINT	Number of AIPs
6	Get	Max. Class Identifier	UINT	7
7	Get	Max. Instance Attribute	UINT	102

A 10.2 AIP Object, Instance 1 (Number of AIPs) Attributes

Table A-30 AIP Object, Instance 1 (Number of AIPs) Attributes

Attribute	Access	Name	Type	Value
3	Get	Value	UINT	0 - 0xFFFF
4	Get	Status	BOOL	0 = OK
7	Get/Set	Range	USINT	2
100	Get/Set	Override Range	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

A 10.3 AIP Object Common Services

Table A-31 AIP Object Common Services

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

A 10.4 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.

Table A-32 Range – Attribute 7

Range Value	Description
0	-10 V to +10 V
2	0 V to +10 V (default)
3	+4 mA to +20 mA
6	0 mA to +20 mA
7	-20 mA to +20 mA

Override Range – Attribute 100

A "Set" will give the user access right to a configuration output word (programming word) found in attribute 101.

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.

This is an NV (NV=non volatile) attribute and constant modification requires NV life cycles. Use the control data in I/O for dynamic modification of the control word.

Analog Input Configuration Word in Ethernet/IP I/O – Attribute 102

This attribute places the configuration word (configurable modules only) for the specified input channel (instance) into the Ethernet/IP 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 AI 8 multiplexed module, where the configuration data may change frequently. Note that this attribute is settable for each input instance (word). Modules, such as the AI 8, may have more than one input channel per input word.

A 11 Analog Output Point (AOP) Object (Class Code: 11 (0x0B))

The FL IL 24 BK ETH/IP-PAC supports analog output points (AOP). There is a separate instance for each analog output of the device.

A 11.1 AOP Object Class Attributes

Table A-33 AOP Object Class Attributes

Attribute	Access	Name	Type	Value
1	Get	Revision	UINT	2
2	Get	Max. Object Instance	UINT	Number of AOPs
6	Get	Max. Class Identifier	UINT	7
7	Get	Max. Instance Attribute	UINT	100

A 11.2 AOP Object, Instance 1 (Number of AOPs) Attributes

Table A-34 AOP Object, Instance 1 (Number of AOPs) Attributes

Attribute	Access	Name	Type	Value
3	Get/Set	Value	UINT	0 - 0xFFFF
7	Get/Set	Output Range	BYTE	3 = (-10 to +10)
8	Get	Value Data Type	USINT	6 = UINT
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	AOP Response Data	UINT	

A 11.3 AOP Object Common Services

Table A-35 AOP Object Common Services

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

A 11.4 AOP Object Attributes

Value – Attribute 3

The analog output value is given in offset binary format. The value provided must be within the following range 0..65535 (0 .. 0xFFFFH).

Table A-36 Value – Attribute 3

Value	Output Voltage
0	-10 V
8000_{hex} (2048)	0 V
$FFFF_{hex}$ (4095)	+10 V

Range – Attribute 7

The analog output range defaults to a fixed value of 1 (0 to +10 V DC).

If the analog output module reports that it the range is different, it will change to 3 (-10 to +10 V DC).

Type – Attribute 8

Specified for the analog output data type: 6 (UINT).

Fault State – Attribute 9

The Fault State determines the action to be taken if a fault state is detected. Fault conditions include software conditions (connection timeout).

Table A-37 Fault State – Attribute 9

Fault State	Action Taken
0	Hold last value
1	Set to low limit (0 V DC)
2	Set to high limit (+10 V)
3	Set to value determined by fault value

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.

Table A-38 Idle State – Attribute 10

Idle State	Action Taken
0	Hold last value
1	Set to low limit (0 V DC)
2	Set to high limit (+10 V)
3	Set to value determined by fault value

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 provided must be within the following range: 0..65535 (0..0xFFFFH).

Idle Value – Attribute 12

The Fault Value is used to set the output if the Idle State bit is set to 3 and an Idle condition occurs. The value provided must be within the following range: 0..65535 (0..0xFFFFH).

AOP Response Data – Attribute 100

The Response Value attribute contains the input data associated with the analog output module. Refer to the module-specific data sheet for information on the meaning of this data value.

A 12 Configuration Object (Class Code: 100 (0x64))

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 12.1 Configuration Object Class Attributes

Table A-39 Configuration Object Class Attributes

Attribute	Access	Name	Type	Value
1	Get	Revision	UINT	2
2	Get	Max. Object Instance	UINT	1
6	Get	Max. Class Identifier	UINT	7
7	Get	Max. Instance Attribute	UINT	255

A 12.2 Configuration Object, Instance 1 Attributes

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.

Table A-40 Configuration Object, Instance 1 Attributes

Attribute	Access	Name	Type	Description
3	Get/Set	Number Digital Inputs	USINT	The Number Digital Inputs attribute determines the number of digital input points to be returned in the produced response packet. Number of bytes = (number of channels + 7) / 8
4	Get/Set	Number Digital Outputs	USINT	The Number Digital Outputs attribute determines the number of digital input points to be processed in the consumed request packet. Number of bytes = (number of channels + 7) / 8
5	Get/Set	Number Analog Inputs	USINT	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. Number of bytes = number of channels * 2
6	Get/Set	Number Analog Outputs	USINT	The Number Analog Oututs attribute determines the number of analog output channels. Each analog output consumes 2 bytes of data in the consumed request packet. Number of bytes = number of channels * 2
7	Get/Set	Add All I/O	BOOL	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	The Accept New Configuration attribute will keep the current produced/consumed 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.

Table A-40 Configuration Object, Instance 1 Attributes (Continued)

Attribute	Access	Name	Type	Description
10	Get/Set	Add All Mode	BYTE	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, SOPs, AIPs, AOPs, Special Functions, PCP Special Function and Serial Communication modules are added to the poll. No faults will be added by default. **Fault Value:** Bit 15: – Bit 14:– Bit 13: – Bit 12: SPC Bit 11: AOPs Bit 10: AIPs Bit 9: DOPs Bit 8: DIPs **I/Os:** Bit 7: – Bit 6: SCO Bit 5: PCP Bit 4: SPC Bit 3: AOPs Bit 2: AIPs Bit 1: DOPs Bit 0: DIPs
11	Get/Set	Use Inline Status	BOOL	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 2 byte to the produced size. (Default = set)
12	Get/Set	Include DSUP	BOOL	When set the first byte of I/O contains the Device Supervision Exeption Status byte (adds 1 byte to Produced Size)
13	Get/Set	Special Functions	BOOL	When set, the bus coupler will put the process data for the special function modules with the Data in I/O attribute set in the I/O Command and I/O Response. Default = 0)
14	Get/Set	Pad I/O	BOOL	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. (Default = set)
15	Get/Set	Number of DIP Faults	UINT	Selects the number of DIP faults to be added to the poll response. Number of bytes = (number of channels + 7) / 8
16	Get/Set	Number of DOP Faults	UINT	Selects the number of DOP faults to be added to the poll response. Number of bytes = (number of channels + 7) / 8
17	Get/Set	Number of AIP Faults	UINT	Selects the number of AIP faults to be added to the poll response. Number of bytes = number of channels * 2

Table A-40 Configuration Object, Instance 1 Attributes (Continued)

Attribute	Access	Name	Type	Description
18	Get/Set	Number of AOP Faults	UINT	Selects the number of AOP faults to be added to the poll response. Number of bytes = number of channels * 2
19	Get/Set	Number of Special Faults	UINT	Selects the number of Special Function faults to be added to the poll response. Number of bytes = (number of channels + 7) / 8
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	This attribute allows the user to reserve bits in the produced/consumed /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	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	Fault Mode	USINT	This attribute defines which action is taken during a major failure such as those described in attributes 25 through 32.
32	Get/Set	Inline Control Byte in Poll	BOOL	Default = 0
38	Get/Set	Number of PCPs in I/O	UINT	
39	Get/Set	Number of SCOs in I/O	UINT	
40	Get/Set	Number of PCP Faults in I/O	UINT	
41	Get/Set	Number of SCO Faults in I/O	UINT	
42	Get/Set	Run/Idle in Produce I/O	BOOL	Default = 0
43	Get/Set	Run/Idle in Consume I/O	BOOL	Default = 1
44	Get/Set	Default WTA for I/O Consumption	USINT	Default = 0

A 12.3 Configuration Object Common Services

Table A-41 Configuration Object Common Services

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

A 13 Inline Interface Object (Class Code: 101 (0x65))

The Inline Interface Object allows the user to control and monitor the Inline interface on the FL IL 24 BK ETH/IP-PAC.

A 13.1 Inline Interface Object Class Attributes

Table A-42 Inline Interface Object Class Attributes

Attribute	Access	Name	Type	Value
1	Get	Revision	UINT	2
2	Get	Max. Object Instance	UINT	1
6	Get	Max. Class Identifier	UINT	7
7	Get	Max. Instance Attribute	UINT	20

A 13.2 Inline Interface Object, Instance 1 Attributes

Table A-43 Inline Interface Object, Instance 1 Attributes

Attribute	Access	Name	Type	Description
4	Get	Inline Status	BYTE	Bit 7: – Bit 6: Fault Cycles Bit 5: Connections Bit 4: Configuring Bit 3: Module Change Bit 2: Power Fault Bit 1: Peripheral Fault Bit 0: CRC
5	Get	First Faulted Module	USINT	Contains the number of the first module that is faulted (1 = bus coupler)
6	Get/Set	Max Retry	USINT	Sets the number of local data transmissions that the coupler will accept before flagging an error. (Default = 32)
7	Get	Number of Modules	UINT	Displays the number of Inline I/O modules that the bus coupler detected.
8	Get	Number of Bits	UINT	Displays the Process Data size of the Inline modules in Bits (input & output bits).
9	Get	Number of Bytes	UINT	Displays the Process Data size of the Inline modules in Bytes (input & output bytes).
11	Get	Scans Per Second	UINT	Displays the number of local I/O scans per second.
12	Get	Loop Diagnostic Count	UINT	Displays the loop diagnostic count during a connection failure.
13	Get	Connection Failure Endpoint #1	USINT	Displays the number of the module at the first end of a connection failure.
14	Get	Connection Failure Endpoint #2	USINT	Displays the number of the module at the second end of a connection failure.

Table A-43 Inline Interface Object, Instance 1 Attributes

Attribute	Access	Name	Type	Description
15	Get/Set	Latched Inline Status	BYTE	Displays the latched value of the Inline Status during the last failure. (See Inline Status Attribute)
16	Get	Latched Faulted Module	USINT	Contains the number of the first module that was faulted during the last fault. 1 = bus coupler
17	Get	Latched Connection Failure Endpoint #1	USINT	Displays the number of the module at the first end of a connection failure latched during the last connection failure.
18	Get	Latched Connection Failure Endpoint #2	USINT	Displays the number of the module at the second end of a connection failure latched during the last connection failure.
19	Get	Power Supply Fault	BYTE	Displays the status of the power supplies connected to the buscoupler. A value of 1 indicates a power supply out of range. A value of 0 indicates a power supply out is OK. Bit 7 - 4: – Bit 3: U_M Bit 2: U_S Bit 1: U_L Bit 0: RES
20	Get/Set	Inline Control Byte	BYTE	Bit 7 - 2: – Bit 1: Clear Input Latches Bit 0: Acknowledge, Latched PF
21	Get	Error History (most recent)	UINT	Contains the most recent logged error information
...	...	...	...	...
30	Get	Error History (last saved)	UINT	Contains the last saved error information

A 13.3 Inline Interface Object Common Services

Table A-44 Inline Interface 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

A 14 Inline Module Object (Class Code: 102 (0x66))

The Inline Module Object allows the user to monitor the Inline modules attached to the FL IL 24 BK ETH/IP-PAC.

A 14.1 Inline Module Object Class Attributes

Table A-45 Inline Module Object Class Attributes

Attribute	Access	Name	Type	Value
1	Get	Revision	UINT	2
2	Get	Max. Object Instance	UINT	Number of Inline modules
6	Get	Max. Class Identifier	UINT	7
7	Get	Max. Instance Attribute	UINT	7

A 14.2 Inline Module Object, Instance 1 to 63 Attributes

Table A-46 Inline Module Object, Instance 1 to 63 Attributes

Attribute	Access	Name	Type	Description
3	Get	Inline Module ID (Config)	UINT	Displays the 16-bit ID for the module as the unit was configured.
4	Get	Inline Module ID (Current)	UINT	Displays the 16-bit ID for the module currently.
5	Get	CIP Class	USINT	Reflects the Number of the CIP Class that the module is mapped to (i.e. 8 = DIP, 9 = DOP, etc).
6	Get	First CIP Instance	UINT	Reflects the first instance of the CIP object that this module is mapped to.
7	Get	Last CIP Instance	UINT	Reflects the last instance of the CIP object that this module is mapped to.

A 14.3 Inline Module Object Common Services

Table A-47 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

A 15 Inline Special Function Object (Class Code: 103 (0x67))

The Inline Special Function object allows the user to control and monitor the Inline modules attached to the FL IL 24 BK ETH/IP-PAC that cannot be mapped to any standard Ethernet/IP object.

A 15.1 Inline Special Function Object Class Attributes

Table A-48 Inline Special Function Object Class Attributes

Attribute	Access	Name	Type	Value
1	Get	Revision	UINT	2
2	Get	Max. Object Instance	UINT	Number of Inline SPC modules
6	Get	Max. Class Identifier	UINT	7
7	Get	Max. Instance Attribute	UINT	7

A 15.2 Inline Special Function Object, Number of Modules Attributes

Table A-49 Inline Special Function Object, Number of Modules 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	Get/Set	Data In I/O	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 buscoupler and is therefore, only selectable when the I/O connection is not in the established state.

A 15.3 Inline Special Function Object Common Services

Table A-50 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

A 16 COS Mask Object (Class Code: 104 (0x68))

The COS Mask Object allows the user to control which bits in the Config In Data Attribute cause a COS message to be generated.

A network heartbeat will update all COS inputs

A 16.1 COS Mask Object Class Attributes

Table A-51 COS Mask Object Class Attributes

Attribute	Access	Name	Type	Value
1	Get	Revision	UINT	1
2	Get	Max. Object Instance	UINT	1
6	Get	Max. Class Identifier	UINT	7
7	Get	Max. Instance Attribute	UINT	20

A 16.2 COS Mask Object, Instance 1 Attributes

Table A-52 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.
5	Get/Set	COS Byte Value	BYTE	Gets or sets the byte value in the COS Mask Array that is indexed by attribute 4.
6	Get/Set	COS Add All Mode	WORD	Automatically generates a COS Mask basedon the following bits, when add all attribute is set to a 1. **Faults:** Bit 15: – Bit 7: Control Bit 14: SCO Bit 6: SCO Bit 13: PCP Bit 5: PCP Bit 12: SPC Bit 4: SPC Bit 11: AOPs Bit 3: AOPs Bit 10: AIPs Bit 2: AIPs Bit 9: DOPs Bit 1: DOPs Bit 8: DIPs 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.
8	Get/Set	Enable Device Supervisor Exception Status	BOOL	When set to a 1, generates a bit pattern I the COS mask array. Then a COS message is generated as soon as any of the bits it the Device Supervisor Exception Status byte changes state.
9	Get/Set	Enable Inline Status (include by default)	BOOL	When set to a 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 a 1, a COS message will be generated when any DIP Fault changes state.
11	Get/Set	Enable DOP Faults	BOOL	When set to a 1, a COS message will be generated when any DOP Fault changes state.
12	Get/Set	Enable AIP Faults	BOOL	When set to a 1, a COS message will be generated when any AIP Fault changes state.
13	Get/Set	Enable AOP Faults	BOOL	When set to a 1, a COS message will be generated when any AOP Fault changes state.
14	Get/Set	Enable Special Function Faults	BOOL	When set to a 1, a COS message will be generated when any Special Function Module Fault changes state.
15	Get/Set	Enable DIPs	BOOL	When set to a 1, a COS message will be generated when any DIP value changes state.

Table A-52 COS Mask Object, Instance 1 Attributes (Continued)

Attribute	Access	Name	Type	Description
16	Get/Set	Enable AIPs	BOOL	When set to a 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 a 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 = =xFF== any time an AIP's upper 8 bits change, aCOS message will be generated.

A 16.3 COS Mask Object Common Services

Table A-53 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

A 17 PCP Special Function Object (Class Code: 105 (0x69))

Table A-54 PCP Object Class Attributes

Attribute	Access	Name	Type	Value
1	Get	Revision	UINT	1
2	Get	Max. Object Instance	UINT	0 - 8 (number of connected PCP devices)
6	Get	Max. Class Identifier	UINT	7
7	Get	Max. Instance Attribute	UINT	23

A 17.1 PCP Object, Instance 1 to Number of PCP Modules (Maximum of Eight) Attributes

Table A-55 PCP Object, 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 tranferred 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:** Byte 1: Service Byte 2: Module number Byte 3: Index LSB Byte 4: Index MSB 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 an can be up to the PDU size for the module (typically 64 bytes). See PDU Length Attribute.

Table A-55 PCP Object, 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 th PCP module. The format is as follow(s): **Succesful Response:** Byte 1: Service Byte 2: Status Byte 3: Length Byte 4: Data-block, if necessary Byte N: Data-block, if necessary **Error Response:** Byte 1: Service 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.
9	Get/Set	PCP Write Index	UINT	Sets the Index of the PCP Object that will be writeen when the user writes to the PCP Write Data Atribute (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 refernced by th 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 read 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 read 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.

Table A-55 PCP Object, Instance 1 to Number of PCP Modules (Maximum of Eight) Attributes (Continued)

Attribute	Access	Name	Type	Description
15	Get/Set	PCP Request Fragment		Allows the user to send requests to the PCP modules. See fragmented service information in section "Communications Methods" on page 4-8.
		Service	STRUCT OF BYTE	
		Data	ARRAY OF BYTE	
16	Get	PCP Response Fragment		Allows the user to get response from the PCP modules. See fragmented service information in section "Communications Methods" on page 4-8.
		Service	STRUCT OF BYTE	
		Data	ARRAY OF BYTE	
17	Get/Set	PCP Fragment Data	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 FL IL 24 BK ETH/IP-PAC. Data size is determined the I/O module.
20	Get/Set	Process Data OUT	ARRAY	Data sent from the FL IL 24 BK ETH/IP-PAC 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 Invoke ID	USINT	
23	Get/Set	PCP Read Invoke ID	USINT	

A 17.2 PCP Object Common Services

Table A-56 PCP Object Common Services

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

A 18 Serial Communications Object (Class Code: 106 (0x6A))

The Serial Communications Object allows the user to control and transfer serial data on PCP based RS-232 and RS-485/422 modules.

A 18.1 Serial Communications Object Class Attributes

Table A-57 Serial Communications Object Class Attributes

Attribute	Access	Name	Type	Value
1	Get	Revision	UINT	1
2	Get	Max. Object Instance	UINT	0 - 8 (number of connected serial devices)
6	Get	Max. Class Identifier	UINT	7
7	Get	Max. Instance Attribute	UINT	24

A 18.2 Serial Communications Object, Instance 1 to (Number of Serial Modules (Maximum of Eight) Attributes

Table A-58 Serial Communications Object, Instance 1 to (Number of Serial Modules (Maximum of Eight) Attributes

Attribute	Access	NV	Name	Type	Description
3	Get	V	Module Type	USINT	Value indicates type of serial module. 0 = RS-232 1 = RS-485/RS-422
4	Get	V	Module Status	BOOL	Value indicates module status. 0 = OK 1 = Faulted
5	Get	V	Serial Status Word	WORD	Serial Status Word Setting **Serial Status Word-MSB:** Bit 15 - 8: Number of Received Characters (Mode Dependant) **Serial Status Word-LSB:** Bit 7: Reserved Bit 6: Transmit Buffer Not Empty Bit 5: Transmit Buffer Full Bit 4: Receive Buffer Full Bit 3: Re-Init Executed Bit 2: Send Error Bit 1: Receive Error Bit 0: Receive Buffer Not Empty

Table A-58 Serial Communications Object, Instance 1 to (Number of Serial Modules (Maximum of Eight) Attributes (Continued)

Attribute	Access	NV	Name	Type	Description
6	Get/Set	V	Serial Control Word	WORD	Serial Control Word Setting **Serial Control Word-MSB:** Bit 15 - 8: Reserved **Serial Control Word-LSB:** Bit 7: DTR Bit 6-4: Reserved Bit 3: Execute Re-Init Bit 2: Reset Send Error Bit 1: Reset Receive Error Bit 0: Reserved
7	Get	V	Receive Data	SHORT_ STRING	Allows user to read Serial Data in one Get_Attribute_Single command.
8	Get/Set	V	Transmit Data	SHORT_ STRING	Allows user to transmit Serial Data in one Set_Attribute_Single command.
9	Get	V	Receive Data Fragment		Allows user to read serial data in fragments. See section "Communications Methods" on page 4-8.
			Service	STRUCT OF BYTE	
			Data	ARRAY OF BYTE	
10	Get/Set	V	Transmit Data Fragment		Allows user to write serial data in fragments. See section "Communications Methods" on page 4-8.
			Service	STRUCT OF BYTE	
			Data	ARRAY OF BYTE	
11	Get/Set	NV	Protocol	USINT	Protocol Settings **00_{hex}: Transparent** 01_{hex}: End-to-end 02_{hex}: Dual buffer 03_{hex}: 3964R 04_{hex}: XON/XOFF

Table A-58 Serial Communications Object, Instance 1 to (Number of Serial Modules (Maximum of Eight) Attributes (Continued)

Attribute	Access	NV	Name	Type	Description
12	Get/Set	NV	Baud Rate	USINT	Baud Rate Settings 00_{hex}: 110 01_{hex}: 300 02_{hex}: 600 03_{hex}: 1200 04_{hex}: 1800 05_{hex}: 2400 06_{hex}: 4800 **07_{hex}: 9600** 08_{hex}: 19200
13	Get/Set	NV	Data Width	USINT	Data Width Setting **Code Data / Bits / Parity / Stop bits** 00_{hex} 7 Even 1 01_{hex} 7 Odd 1 **02_{hex} 8 Even 1** 03_{hex} 8 Odd 1 04_{hex} 8 Without 1 1 05_{hex} 7 Without 1 1 06_{hex} 7 Even 2 07_{hex} 7 Odd 2 08_{hex} 8 Even 2 09_{hex} 8 Odd 2 $0A_{hex}$ 8 Without 2 2 $0B_{hex}$ 7 Without 2 2
16	Get/Set	NV	Error Pattern	USINT	Error Pattern Settings **24_{hex}: $** XX_{hex}: Any character
17	Get/Set	NV	First Delimiter	USINT	First Delimiter Settings **$0D_{hex}$: Carriage return (CR)** XX_{hex}: Any character
18	Get/Set	NV	Second Delimiter	USINT	Second Delimiter Settings **$0A_{hex}$: Line feed (LF)** XX_{hex}: Any character

Table A-58 Serial Communications Object, Instance 1 to (Number of Serial Modules (Maximum of Eight) Attributes (Continued)

Attribute	Access	NV	Name	Type	Description
19	Get/Set	NV	3964R Priority	USINT	3964R Priority Settings **00_{hex}: Low priority** 01_{hex}: High priority
20	Get/Set	NV	Output Type	USINT	Output Type Settings **00_{hex}: RS-232 (Default for RS-232 Module Type)** **01_{hex}: RS-485 (Default for RS-485 Module Type)** 02_{hex}: RS-422
21	Get/Set	NV	DTR Control	USINT	DTR Control Settings (only valid for RS-232 type) **00_{hex}: Automatic** 01_{hex}: Via process data
22	Get/Set	NV	Rotation Switch	USINT	Rotation Switch Settings **00_{hex}: No rotation** 01_{hex}: Rotation
23	Get/Set	NV	XON Pattern	USINT	XON Pattern Settings **11_{hex}: Default** XX_{hex}: Any character (not the same as XOFF pattern)
24	Get/Set	NV	XOFF Pattern	USINT	XOFF Pattern Settings **13_{hex}: Default** XX_{hex}: Any character (not the same as XON pattern)

Table A-58 Serial Communications Object, Instance 1 to (Number of Serial Modules (Maximum of Eight) Attributes (Continued)

Attribute	Access	NV	Name	Type	Description
31	Get/Set	NV	Status Control Word In Ethernet/IP I/O	BOOL	If this attribute is set, the serial status word is added to the Produce Data of all I/O connections. The serial control word is added to the Consume Data. This attribute affects the produce and the consume sizes of all I/O connections, and is therefore only settable when there are no I/O connections in the established state.
32	Get/Set	NV	Fragment Data In Ethernet/IP I/O	BOOL	If this attribute is set, the Receive Data Fragment is added to the Produce Data of all I/O connections. The Transmit Data Fragment is added to the Consume Data. This attribute affects the produce and the consume sizes of the I/O connections, and can therefore only be set if no I/O connections have been established.
33	Get/Set	NV	Enable Serial Object	BOOL	If this attribute is set, the serial object controls the buffering of serial data. If this attribute is set to 0, the serial modules can be accessed directly via the respective PCP class, instead of the SCO class.

A 18.3 Serial Communications Object Common Services

Table A-59 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

A 19 Port Object Class Definition (Class Code 244 (0xF4))

A 19.1 Port Object Class Attributes

Table A-60 Port Object Class Attributes

Attribute	Access	Name	Type	Value
1	Get	Revision	UINT	1
2	Get	Max. Object Instance	UINT	1 (an Ethernet port on the bus coupler)
6	Get	Max. Class Identifier	UINT	9
7	Get	Max. Instance Attribute	UINT	6
8	Get	Entry Port	UINT	1 (only one port is supported)
9	Get	All Ports	ARRAY OF STRUCT	

A 19.2 Port Object Instance Attributes

Table A-61 Port Object Instance Attributes

Attribute	Access	NV	Name	Type	Value
1	Get	V	Port Type	UINT	1
2	Get	V	Port Number	UINT	2
3	Get	V	Port Object	UINT	3
4	Get	V	Port Name	SHORT_ STRING	4

A 19.3 Port Object Common Services

Table A-62 Port Object Common Services

Service Code	Class	Instance	Service Name
1 (0x01)	Yes	Yes	Get_Attribute_All
2 (0x2)	No	No	Set_Attribute_All

A 20 TCP/IP Interface Object (Class Code 245 (0xF5))

A 20.1 TCP/IP Interface Object Class Attributes

Table A-63 TCP/IP Interface Object Class Attributes

Attribute	Access	Name	Type	Value
1	Get	Revision	UINT	1
2	Get	Max. Object Instance	UINT	1
6	Get	Max. Class Identifier	UINT	7
7	Get	Max. Instance Attribute	UINT	6

A 20.2 TCP/IP Interface Object Instance Attributes

Table A-64 TCP/IP Interface Object Instance Attributes

Attribute	Access	NV	Name	Type	Value
1	Get	V	Status	DWORD	1
2	Get	V	Configuration Compatibility	DWORD	2
3	Set	NV	Configuration Control	DWORD	3
4	Get	V	Physical Link Object Path Size Path	STRUCT OF UINT Padded EPATH	4
5	Set	NV	Interface Configuration	STRUCT OF	5
			IP Address	UDINT	
			Network Mask	UDINT	
			Gateway Address	UDINT	
			Name Server	UDINT	
			Name Server 2	UDINT	
			Domain Server	STRING	
6	Set	NV	Host Name	STRING	6

TCP/IP Interface Object Common Services

Table A-65 TCP/IP Interface Object Common Services

Service Code	Class	Instance	Service Name
1 (0x01)	Yes	Yes	Get_Attribute_All
2 (0x2)	No	Yes	Set_Attribute_All
14 (0x0E)	Yes	Yes	Get_Attribute_Single
16 (0x10)	No	Yes	Set_Attribute_Single

Status – Attribute 1

Indicates the status of the Configuration attribute:

Bit(s)	Called	Definition
0 - 3	Interface Configuration Status	0 = Not configured 1 = Valid configuration 2 - 15 = Reserved
4 - 31	Reserved	Reserved (set to 0)

Configuration Capability – Attribute 2

Indicates whether the device has optional network capabilities.

Bit(s)	Called	Definition
0	BootP Client	1 = BootP-capable, can receive the required configuration via BootP
4	Configuration Settable	1 = can be configured. Indicates that the interface configuration attribute can be set.
5 - 31	Reserved	Reserved (set to 0)

Configuration Control – Attribute 3

The configuration control attribute tells the device how to determine its own network configuration.

Bit(s)	Called	Definition
0 - 3	Startup Configuration	0 = use configuration from the NV 1 = use BootP 2 -15 = Reserved
4 - 31	Reserved	Reserved (set to 0)

A 21 Ethernet Link Object (Class Code 246 (0xF6))

A 21.1 Ethernet Link Object Class Attributes

Table A-66 Ethernet Link Object Class Attributes

Attribute	Access	Name	Type	Value
1	Get	Revision	UINT	1
2	Get	Max. Object Instance	UINT	1
6	Get	Max. Class Identifier	UINT	7
7	Get	Max. Instance Attribute	UINT	3

A 21.2 Ethernet Link Object Instance Attributes

Table A-67 Ethernet Link Object Instance Attributes

Attribute	Access	NV	Name	Type	Value
1	Get	V	Interface Speed	UDINT	1
2	Get	V	Interface Flags	DWORD	2
3	Get	V	Physical Address	ARRAY of 6 USINTS	3

Interface Flags

Bit(s)	Called	Definition
0	Link Status	Indicates whether or not the Ethernet 802.3 communications interface is connected to an active network. 0 indicates an inactive link; 1 indicates an active link.
1	Half/Full Duplex	0 indicates the interface is running in half duplex mode; 1 indicates full duplex mode. Note that if the link status flag is 0, then the value of the half/full duplex flag is indeterminate.
2 - 31	Reserved	Reserved (set to 0)

Interface Speed

The Interface Speed attribute shall indicate whether the interface is running at 10 Mbps, 100 Mbps, 1 Gbps, etc.
The Interface Speed is intended to indicate the media bandwidth. In full duplex mode the attribute is not doubled.

Physical Address

The Physical Address attribute contains the interface's MAC layer address. The Physical Address is an array of octets. The recommended display format is "XX-XX-XX-XX-XX-XX", starting with the first octet. The Physical Address attribute cannot be set.

Ethernet address shall be assigned by the manufacturer, and shall be unique per IEEE 802.3 requirements.

A 21.3 Ethernet Link Object Common Services

Table A-68 Ethernet Link Object Common Services

Service Code	Class	Instance	Service Name
1 (0x01)	Yes	Yes	Get_Attribute_All
14 (0x0E)	Yes	Yes	Get_Attribute_Single

B Appendices

B 1 List of Figures

Section 1

Section 2

Section 3

Section 4

B 2 List of Tables

Section 1

Section 2

Section 3

Section 4

Appendix A

SCATTERGOOD & JOHNSON LTD

ELECTRICAL ENGINEERING & FLUID CONTROL DISTRIBUTORS

Est.1899

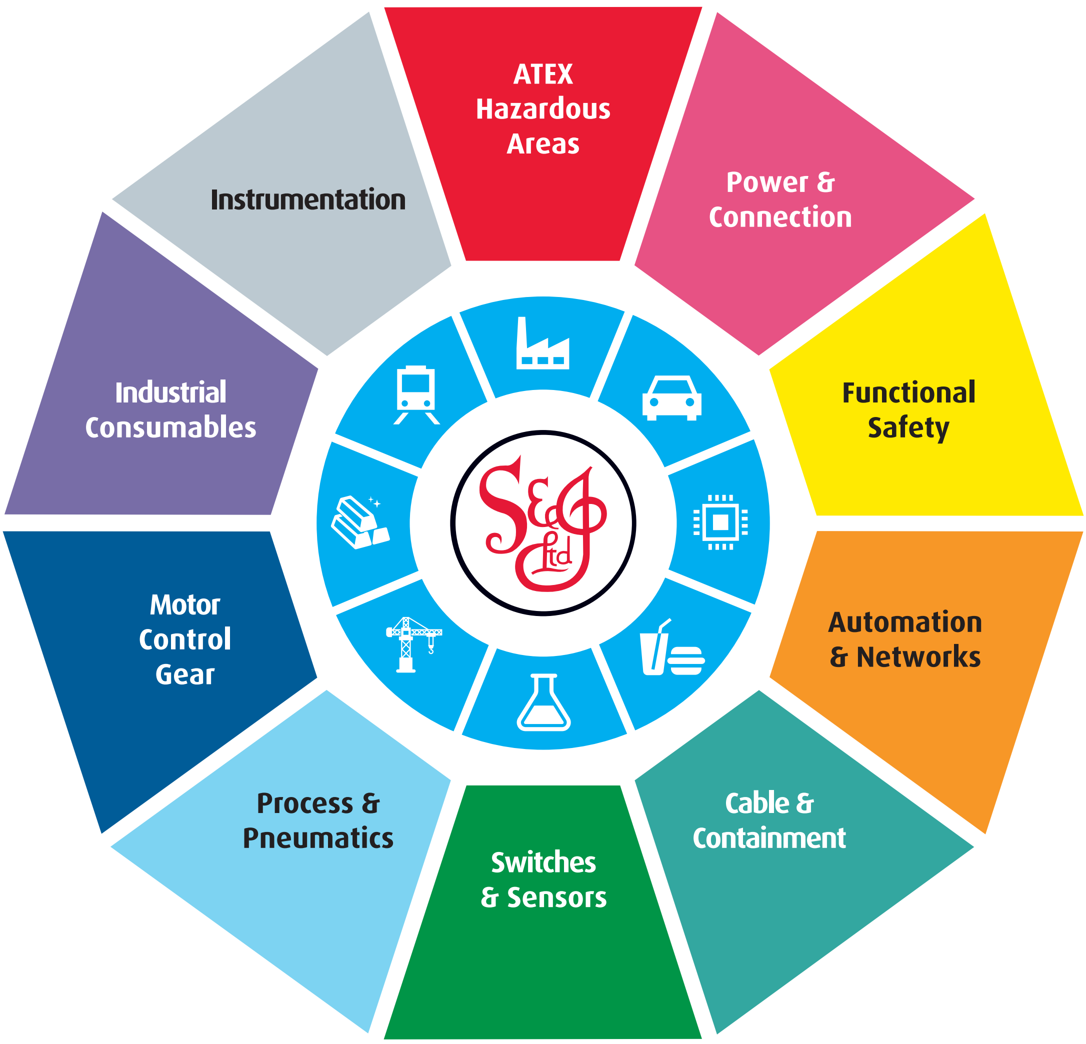
At Scattergood & Johnson Ltd, we pride ourselves on being a technical distributor to specialist industries.

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