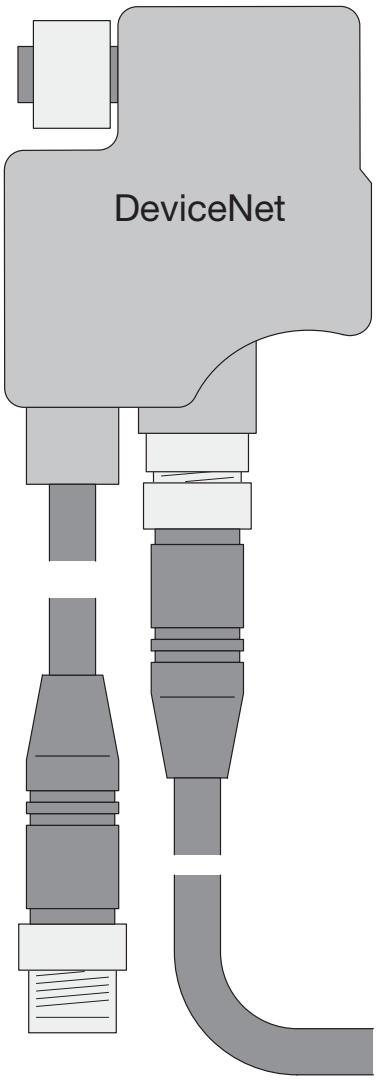
Technical Description

FBP FieldBusPlug

V6

DeviceNet FieldBusPlug
DNP21-FBP



Please note the following

Target group

This description is intended for the use of trained specialists in electrical installation and control and automation engineering, who are familiar with the applicable national standards.

Safety requirements

The responsible staff must ensure that the application or use of the products described satisfy all the requirements for safety, including all the relevant laws, regulations, guidelines and standards.

Content

List of Figures

Overview

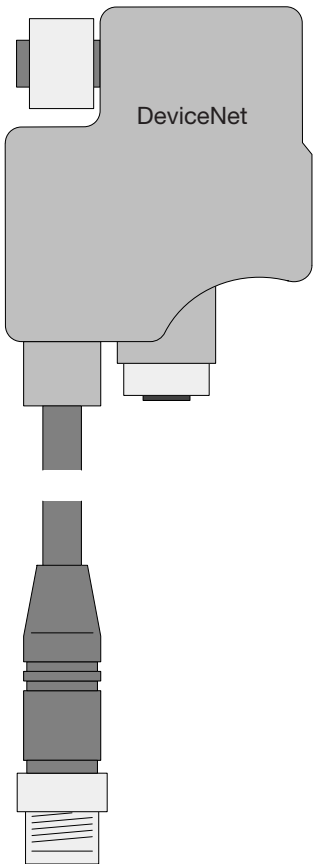


Figure 1: DNP21-FBP

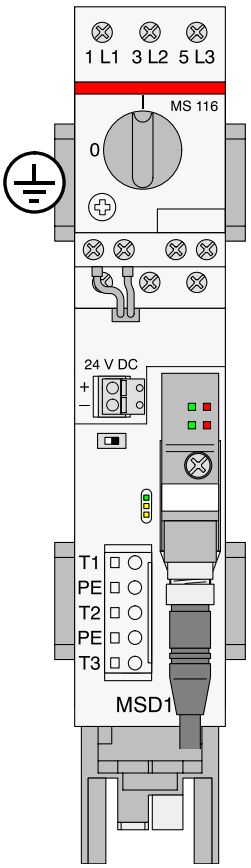


Figure 2: DNP21-FBP, plugged to the motor starter MSD11-FBP

Purpose and short description

The DeviceNet FieldBusPlug DNP21-FBP establishes the field bus connection between the DeviceNet bus and the terminal devices connected to this bus. The DeviceNet FieldBusPlugs are slaves (nodes) on the DeviceNet bus. The terminal devices must have a field bus-neutral interface (e.g. as provided by the ABB FBP modules MSD11-FBP, MSR22-FBP, MFI21-FBP UMC22-FBP, softstarters, breakers and others).

The data exchange between the DeviceNet FieldBusPlug and the terminal device can be performed in two ways:

- parallel

  The data are exchanged directly via the connections of the field bus-neutral interface. Scope of data: max. 1 digital output (control signal to terminal device) plus 2 digital inputs (2 feedback signals from terminal device).

- serial

  The signals are exchanged with the help of a serial data protocol via the field bus-neutral interface. Binary, analog, parameter and diagnosis data are sent and received.

The DNP21-FBP as well as the connected terminal device are powered by the DeviceNet power supply unit, if the terminal device is set to "internal supply". If the setting is "external supply", an additional 24 V DC power supply unit has to be used to supply the electronic circuits of the terminal device.

There is **no** electrical isolation between the DeviceNet bus signals and the field bus-neutral interface.

The DeviceNet FieldBusPlugs have to be addressed, i.e. they must have a unique address used to access the connected terminal device.

Once the address is set, it is stored in the FieldBusPlug, even in case of supply voltage breakdown.

The parameter data of the terminal device are stored in the DeviceNet coupler (master) and sent to the terminal device (slave) with power-on.

According to the DeviceNet standard, the addresses 0 to 63 can be used. The addresses 0, 62 and 63 are reserved and should not be used for slaves. Addressing is carried out by means of separate programming units or by a software tool of the controller manufacturer.

Some of the terminal devices, e.g. the UMC22-FBP, must be parameterized. How to set the parameters is described in the bus-specific software description. Normally, parameter setting is performed by means of a software tool of the controller manufacturer.

For diagnosis purposes, the DeviceNet FieldBusPlugs are equipped with four LEDs (see Fig. 4).

In order to build up a DeviceNet bus or a part of it using DeviceNet FieldBusPlugs, the FieldBusPlugs must be simply connected in series, i.e. the cable of the first FieldBusPlug is plugged to the DeviceNet bus distributor (in the direction of the coupler / gateway), the cable of the second FieldBusPlug is plugged to the socket of the first plug, etc. To make work easier, the DeviceNet FieldBusPlugs are available with different cable lengths.

For very long distances, several cable extensions are available as well as cable coils and male and female plug connectors for self-mounting.

Unplugging the FieldBusPlug from the terminal device does not lead to an interruption of the fieldbus. Within the fieldbus, the FieldBusPlug behaves like a T connector.

The DeviceNet bus must be terminated with a terminating resistor of **120** Ω at each end of the bus. Terminating resistors are part of the assortment.

When determining the total DeviceNet bus length, the permanently connected cables of the DeviceNet FieldBusPlugs must be taken into account. They are part of the DeviceNet bus.

Due to their compound construction, the DeviceNet FieldBusPlugs comply with the requirements of **IP 65** and consequently can also be mounted outside the control cabinet.
In order to meet this, the open connector of the last FieldBusPlug must be covered with the closing cap, which is part of delivery.

Connector pin assignment

Figure 3 shows the connector pin assignment of the FieldBusPlugs for

- the DeviceNet interface (plug at the cable end and bus interface to the next FieldBusPlug)
- the field bus-neutral interface to the terminal device.

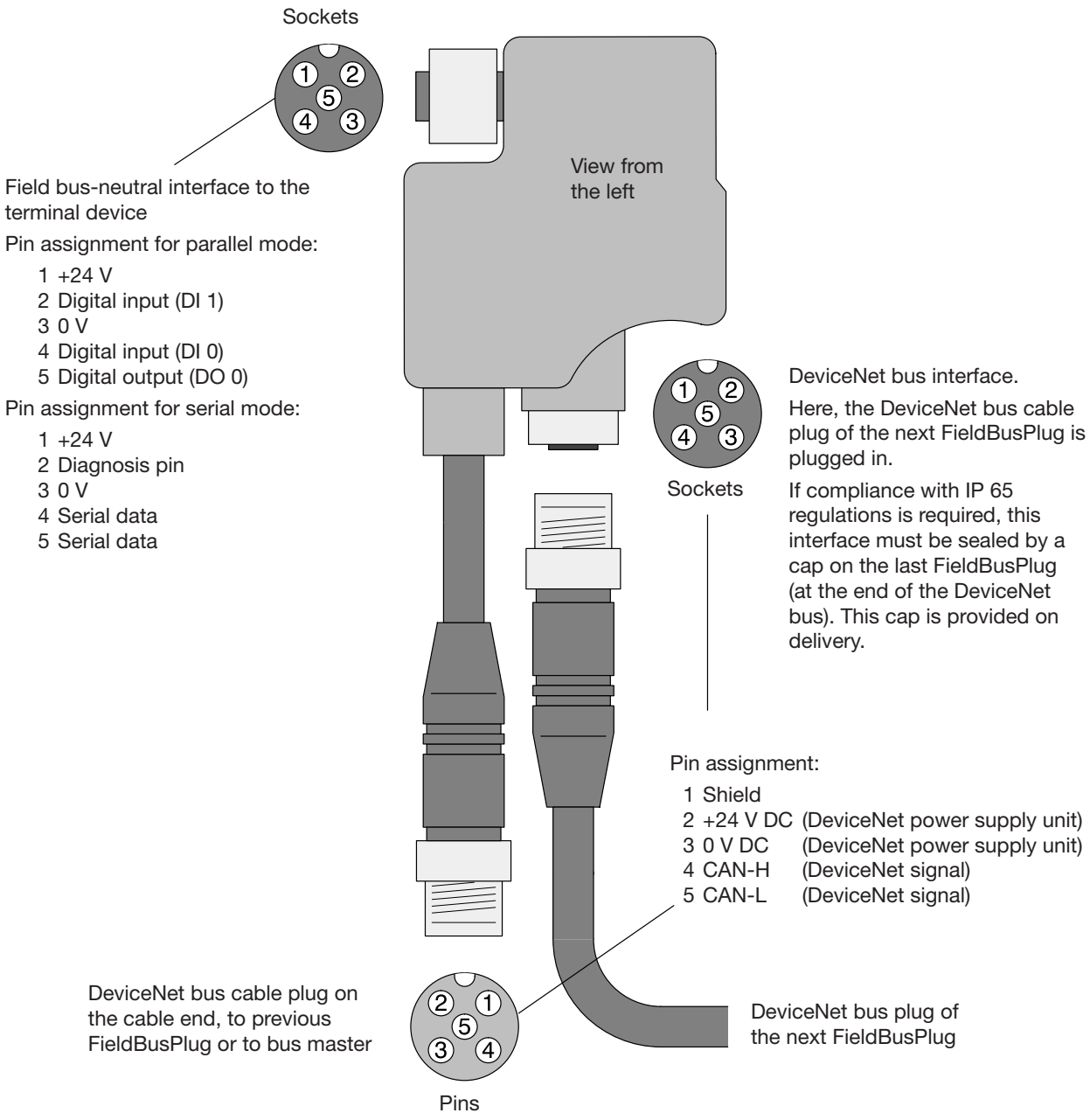
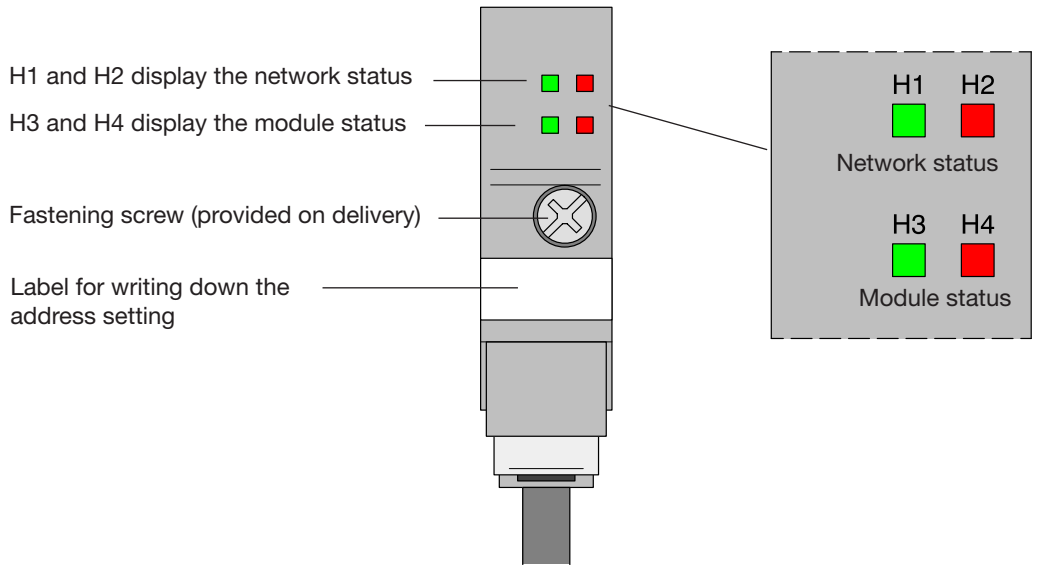


Figure 3: DNP21-FBP, connector pin assignment

Indicators and operating elements on the front plate

Figure 4 shows the indicators and operating elements on the front plate.



Meanings of the LEDs

Network status		Module status		Status / Cause
green LED H1	red LED H2	green LED H3	red LED H4	
off	off			No supply voltage, not online.
flashes	off			Online, but not yet listed in scan list.
on	off			Online and listed in scan list.
off	flashes			I/O connection in timeout state.
off	on			Critical connection error, no communication, e.g. address used twice, bus switched off.
flashes	flashes			Special communication error.
		off	off	No supply voltage connected to the module.
		on	off	Module is working correctly.
		flashes	off	Module in standby mode or missing parameterization.
		off	flashes	Error which can be acknowledged.
		off	on	Unrecoverable module error, module replacement possibly required.
		flashes	flashes	Module is running a self-test.

Figure 4: DNP21-FBP, indicators and operating elements on the front plate

Mapping of I/O data and diagnosis data

For non-modular devices, e.g. MFI21-FBP, UMC22-FBP

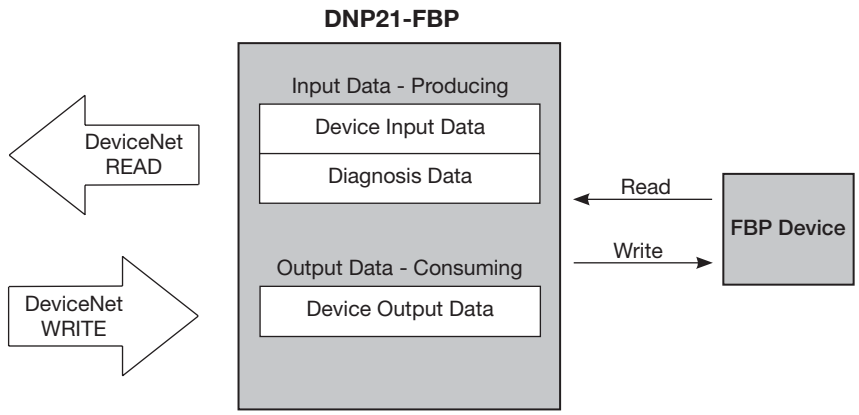


Figure 5: DNP21-FBP, mapping of I/O data and diagnosis data for non-modular devices

For modular devices, e.g. S500

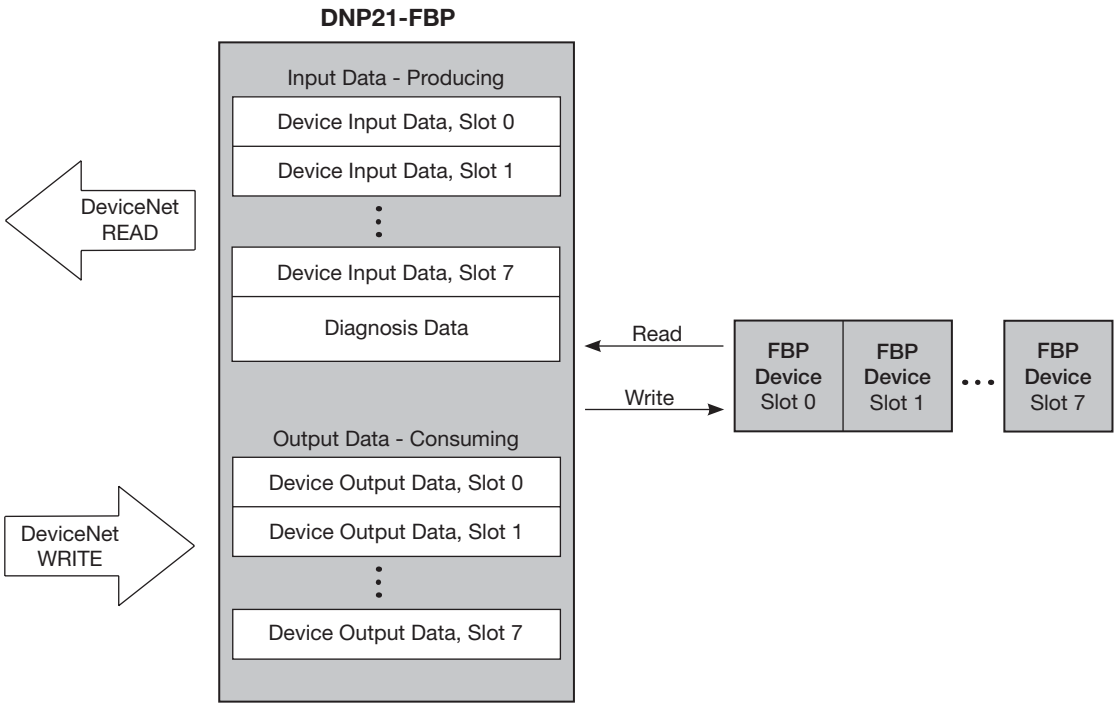


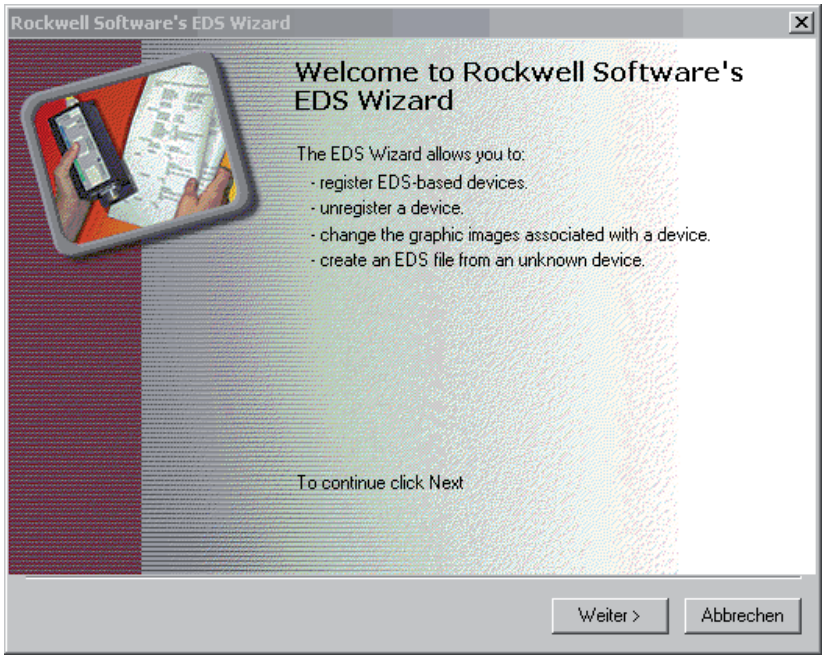
Figure 6: DNP21-FBP, mapping of I/O data and diagnosis data for modular devices

Please refer to the corresponding device manual for a detailed description of the device I/O data and diagnosis data.

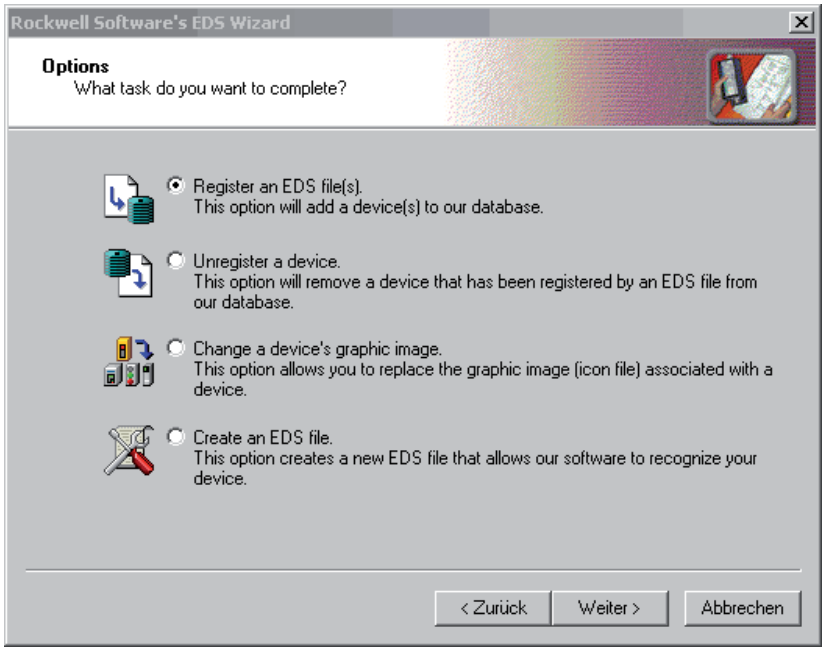
Integrating a DNP21-FBP with a connected FBP device in RSNetWorx (Rockwell Software)

1. Loading the EDS files of the FBP devices to be connected

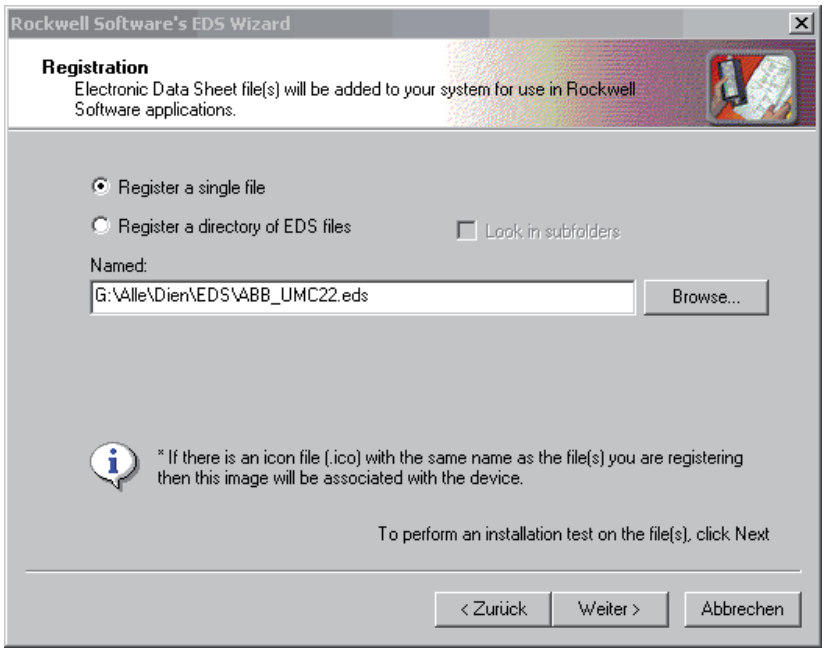
Click *Tools -> EDS Wizard*.



Select the option "Register an EDS file".



Select the EDS file or an EDS directory and follow the instructions given by the EDS Wizard.

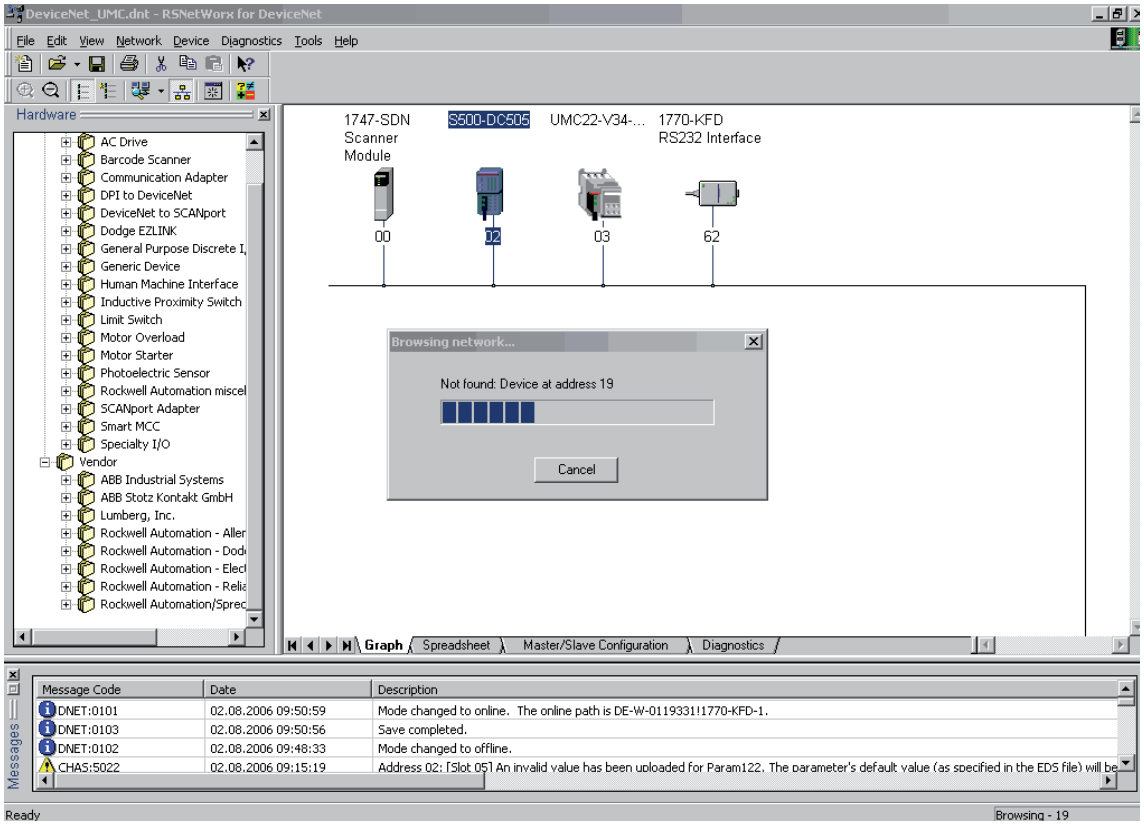


After the installation has been completed successfully, the registered new devices are available in RSNetworx.

2. Configuring the FBP devices on the DeviceNet

Connect the DNP21 to a registered device and to the DeviceNet. Make sure that no bus address (MAC ID) is assigned twice.

After you have completed the bus setup, switch RSNetworx to the online mode by clicking *Network -> Online*. The current bus configuration is imported automatically.



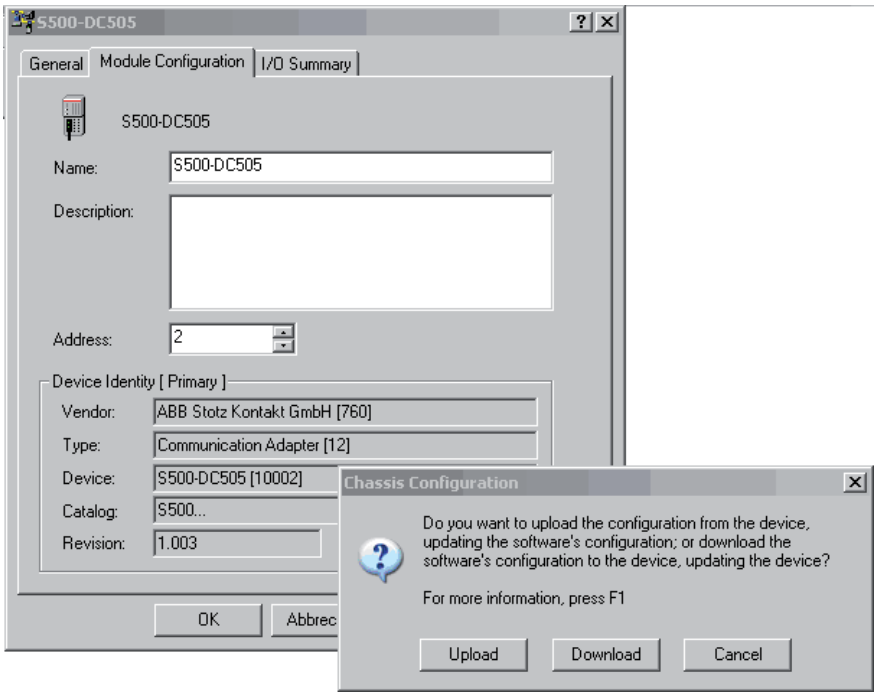
Then, you have to configure the devices and set the device parameters. For this purpose, open the device dialog by double clicking the corresponding device entry.

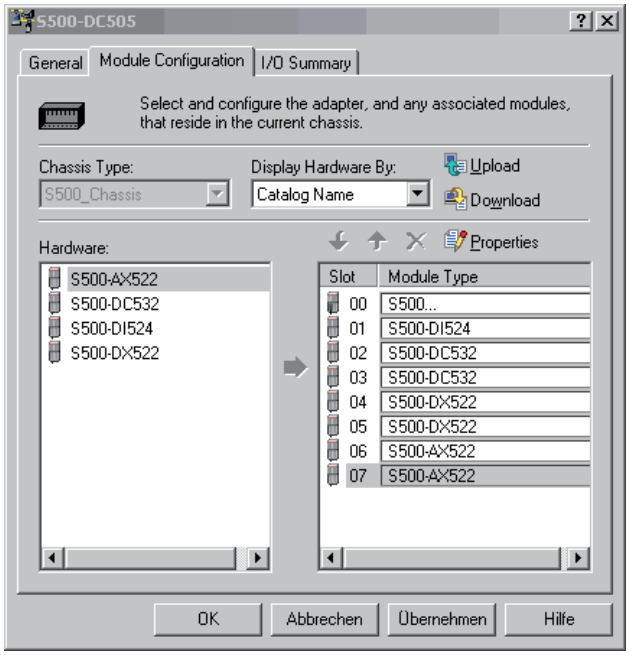
The configuration procedure is described below using two examples:

1. Configuring modular devices – S500-FBP
2. Configuring non-modular devices – UMC22-FBP

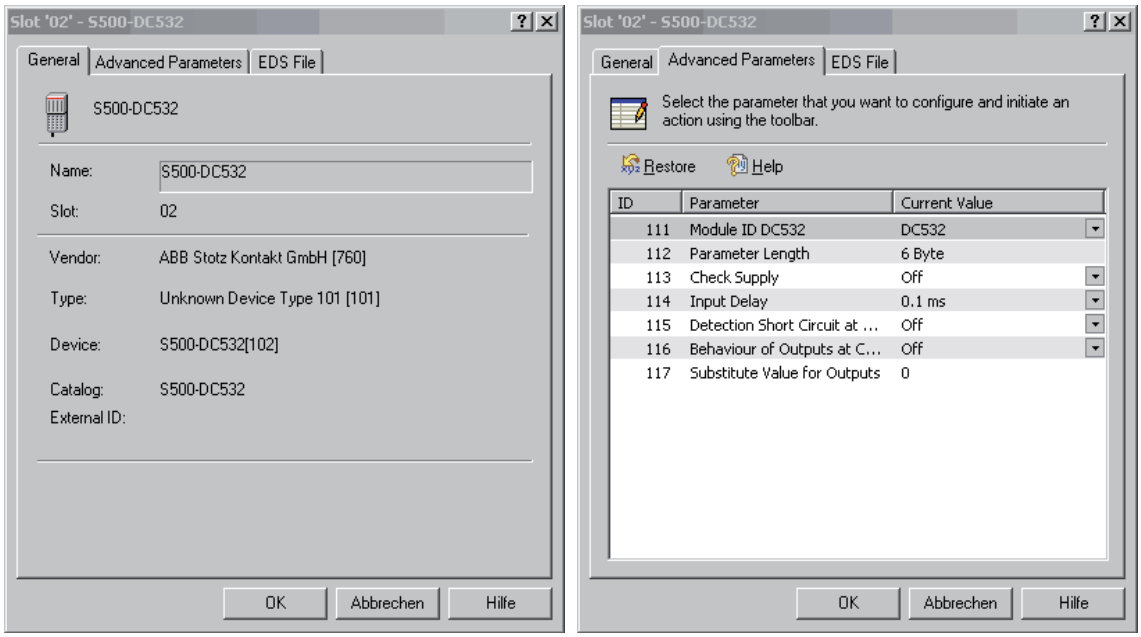
Example 1: Procedure for configuring modular devices (e.g. S500-DC505)

Open the "Module Configuration" tab to import the current module configuration using *Upload*.





Open the properties dialog of the corresponding device by selecting *Properties* and edit the module parameters in the "Advanced Parameters" tab (refer to the following figures).

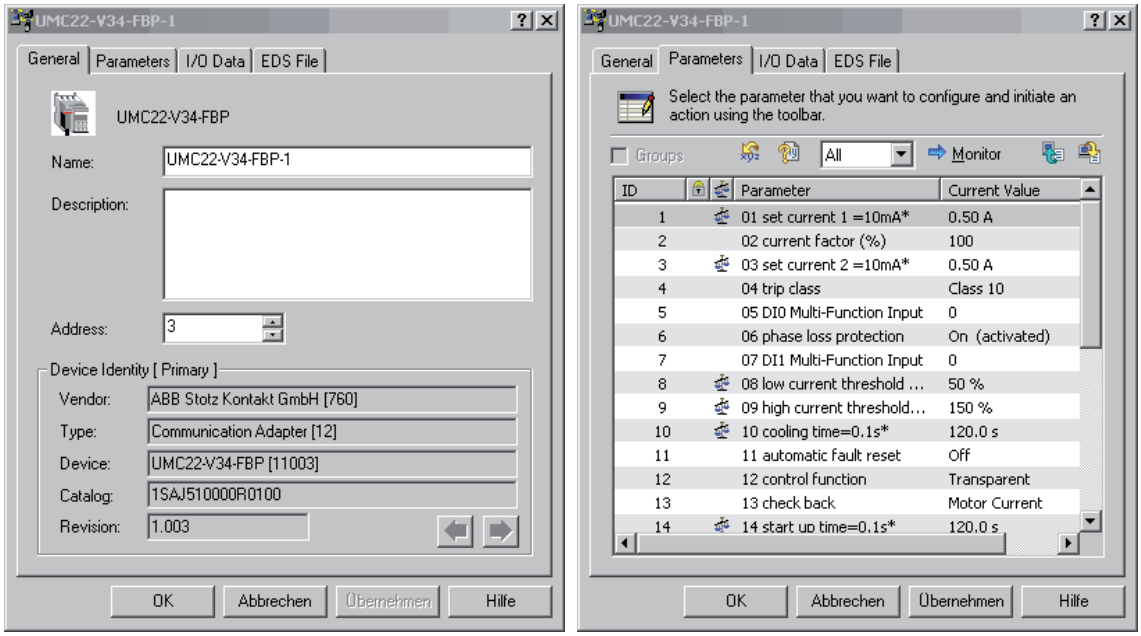


Confirm both dialogs with *OK* and click the *Download* button to download the changed parameters to the device.

Please refer to the corresponding device manual for a detailed description of the modules and the respective parameters.

Example 2: Procedure for configuring non-modular devices (e.g. UMC22-FBP)

Open the device dialog by double clicking the device entry of the device to be configured in the bus configuration.



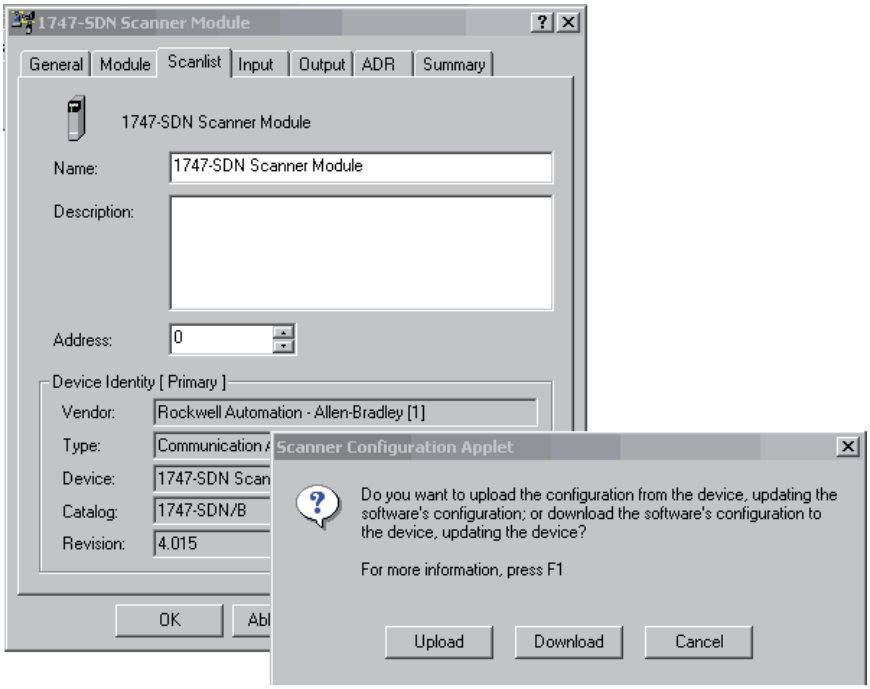
Open the "Parameters" tab and set the parameters as required for the application. After this, click the *Download* button to transfer all parameters or only individual parameters to the device.

Please refer to the device manual of the device to be configured for a detailed description of the corresponding parameters.

3. Adding the bus master to the scan list

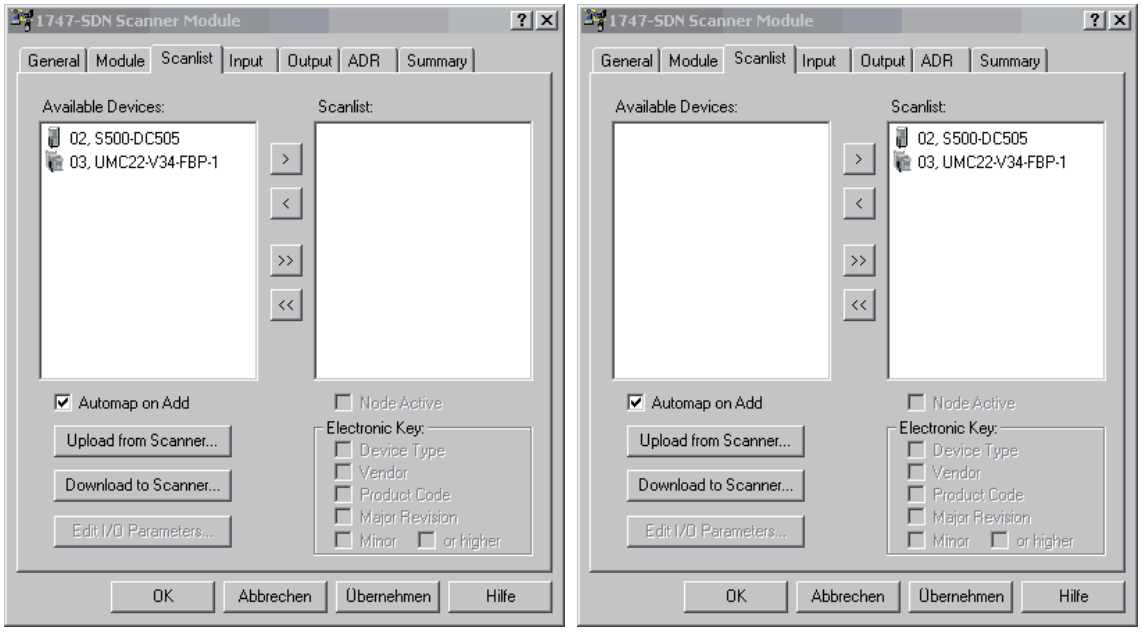
After you have finished the configuration and parameter setting for the devices, it is necessary to add the devices to the scan list of the DeviceNet master.

For this purpose, open the bus master dialog by double clicking its icon (here: DeviceNet Scanner 1747 SDN).



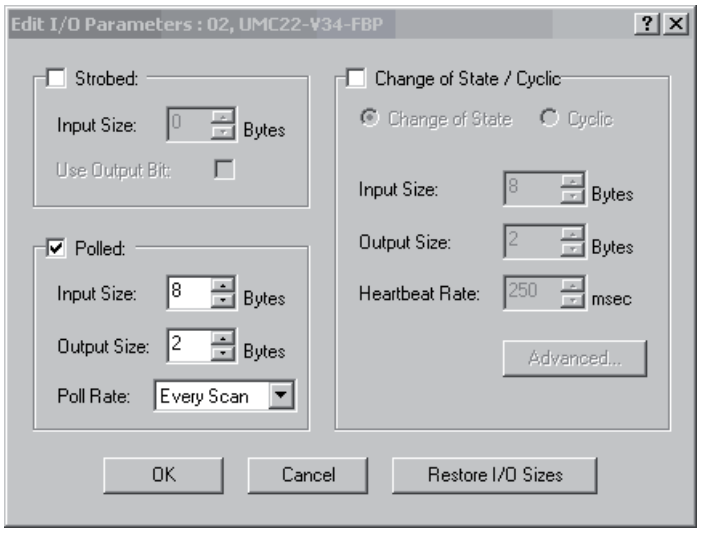
When opening the "Scanlist" tab for the first time, click the *Upload* button to load the current configuration stored in the master.

Use the arrow buttons to add the new devices to the scan list.

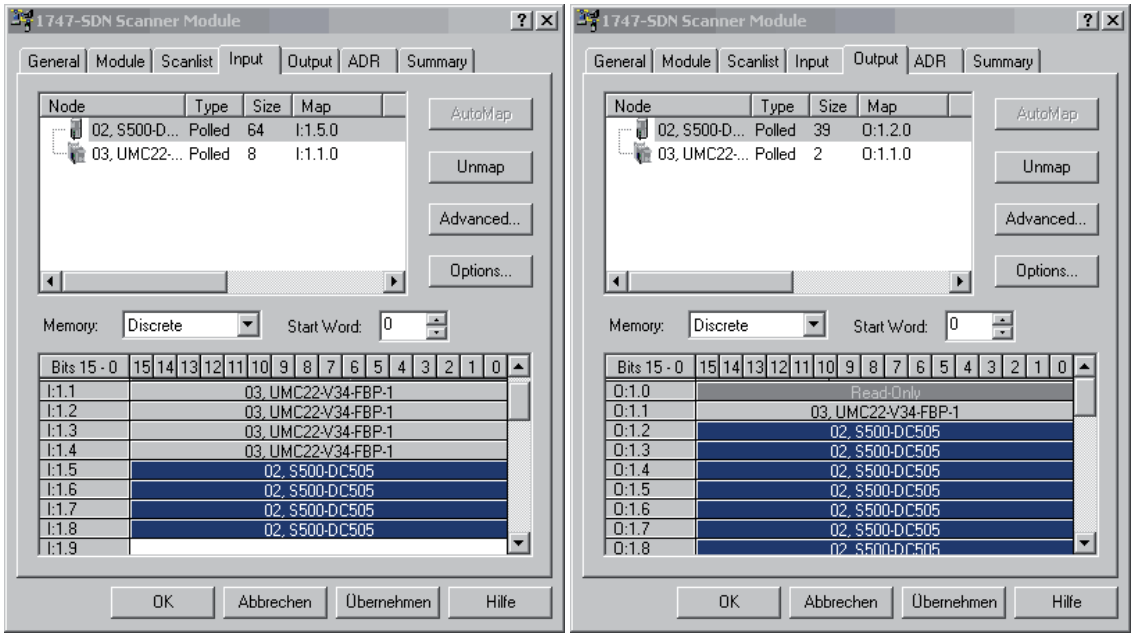


If desired, you can now set the I/O parameters of the devices. To do so, highlight a device in the scan list and click the *Edit I/O Parameters* button.

When opened, the I/O parameters dialog displays the default values as specified in the EDS file.



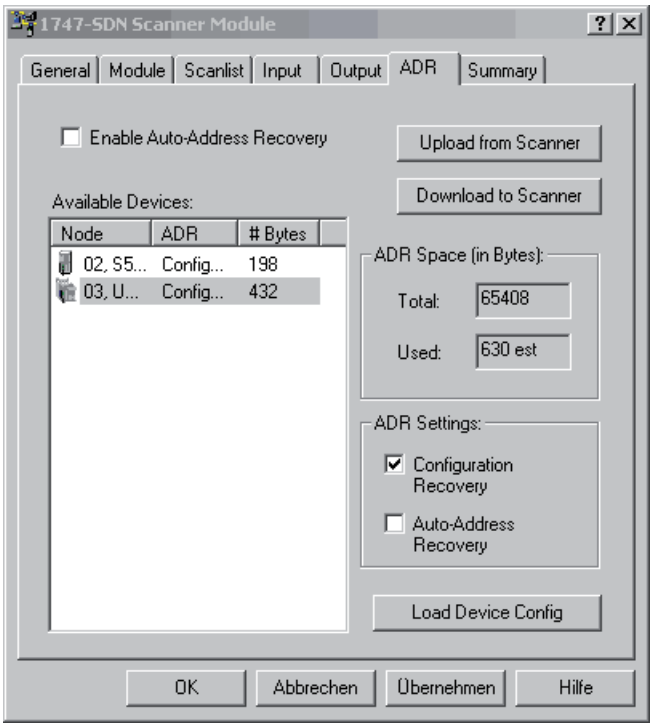
Now, you can use the "Input" and "Output" tabs of the DeviceNet master to specify the mapping of I/O data according to your PLC program.



This enables data access from the PLC program using the addresses displayed on the left.

In order to enable the automatic download of the parameters to the devices in case of a replacement of the device or in case of a restart of the master, it is necessary to specify the corresponding settings in the "ADR" tab. This is particularly important, if an FBP device does not store the parameters in a non-volatile memory.

Open the "ADR" tab.



Highlight the entry of the device for which you want to activate the automatic parameter download and click the *Load Device Config* button.

Then, enable the automatic download function by checking the checkbox "Configuration Recovery".

Finally, click the *Download to Scanner* button to transfer the functionality to the bus master.

Integrating a DNP21-FBP with a connected FBP device in SYCON.net -

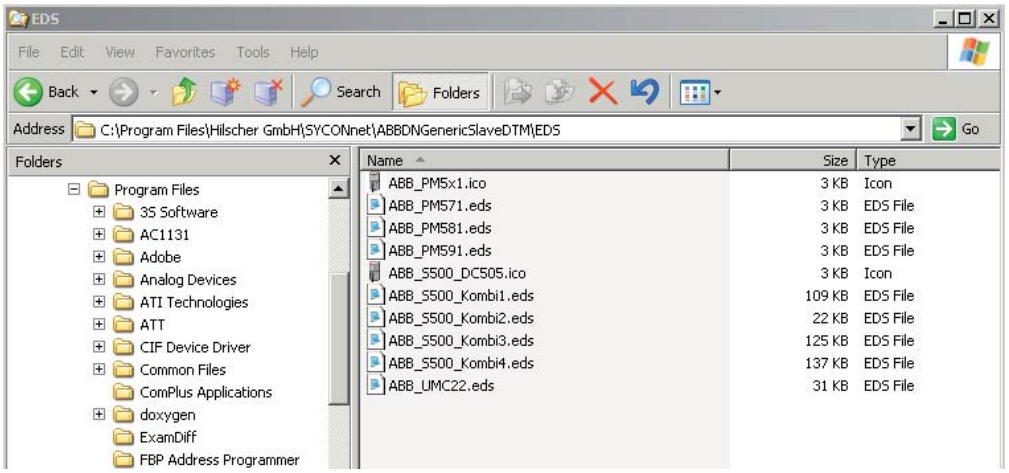
Same Procedure for modular and non-modular devices

Since Sycon.net cannot manage modular EDS, a firm EDS must be produced for modular devices. The configuration is then alike for modular and non-modular devices.

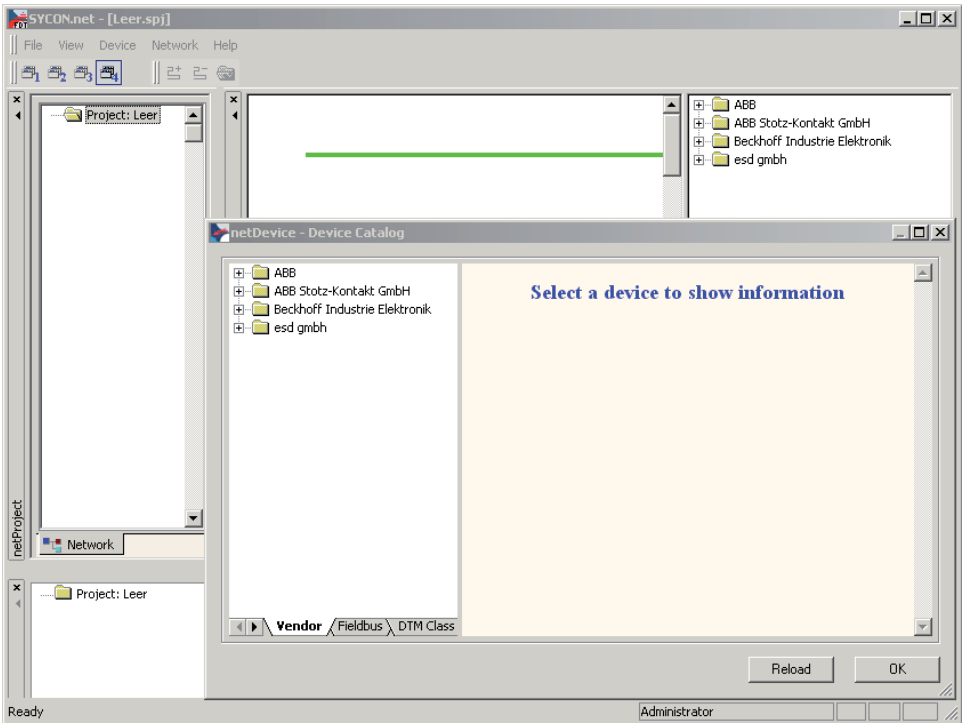
The procedure is described here by the example of a S500 module combination, consisting of DC505, AX522 and DC532. The associated EDS has the name „ABB_S500_Kombi1.eds“.

1. Loading the EDS

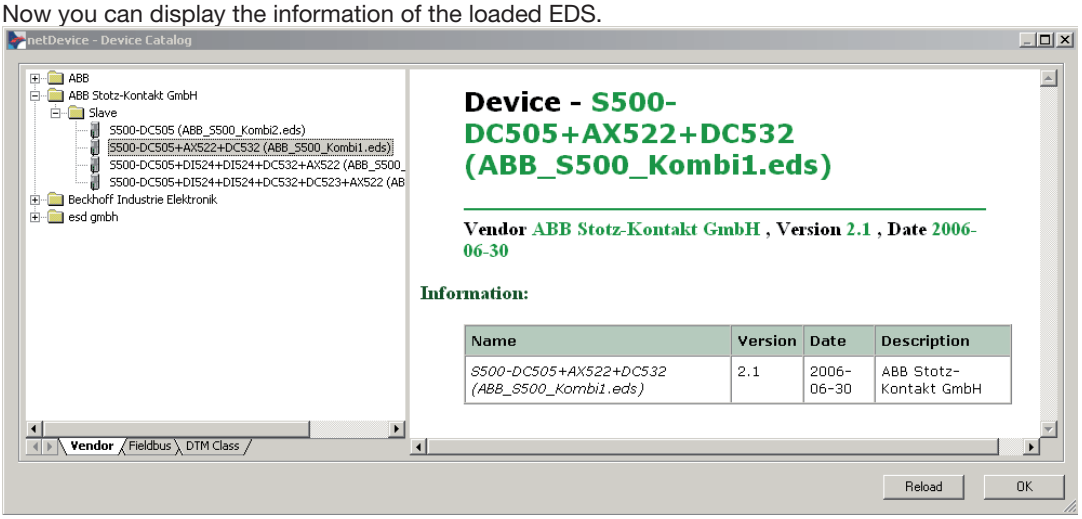
To make the device known in SYCON.net, the EDS as well as the associated icon (if available) must be copied into the following Windows directory:



Now open a new or empty project in SYCON.net and update the decive catalog. Therefore you must select “the device catalog“ in the menu „Network“ . After the window has opened click on „Reload“ to load the EDS.



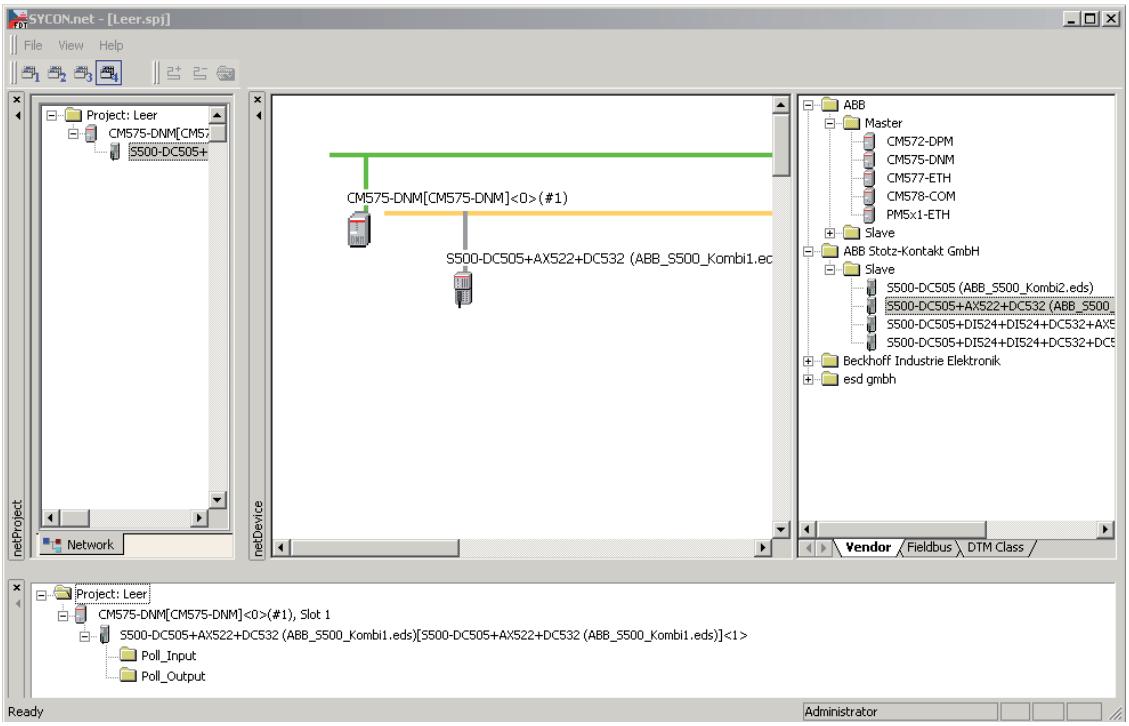
Now you can display the information of the loaded EDS.



After the device catalog was closed, you can start with the bus configuration.

First you select the AC500 DeviceNet bus coupler CM575-DNM from the master list and pull it on by drag and drop to the green internal bus.

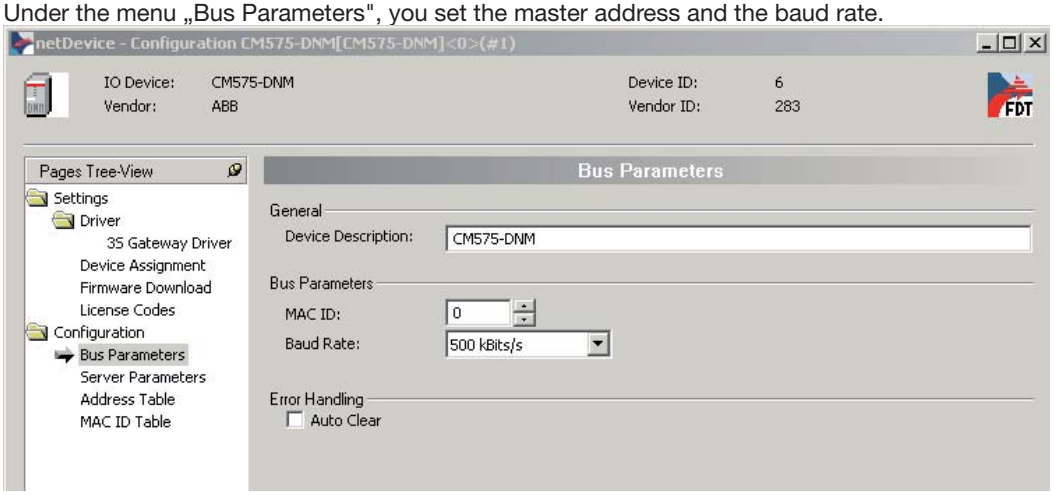
Then you select the desired DeviceNet slave device from the slave list and pull it again by drag and drop to the yellow DeviceNet line.



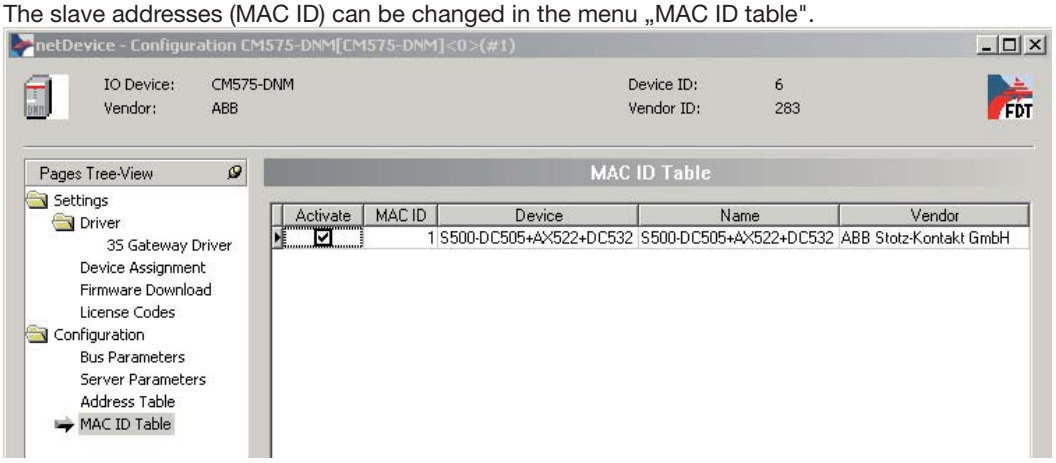
2. Master and bus settings

The dialog for setting the general bus parameters of the DeviceNet is activated by a doubleclick on the symbol of the CM575-DNM bus master.

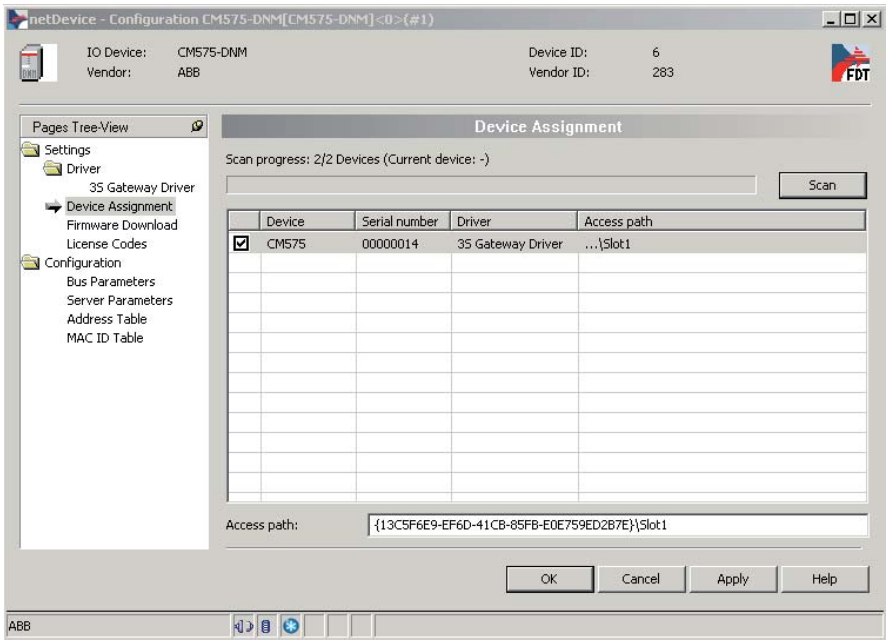
Under the menu „Bus Parameters", you set the master address and the baud rate.



The slave addresses (MAC ID) can be changed in the menu „MAC ID table".



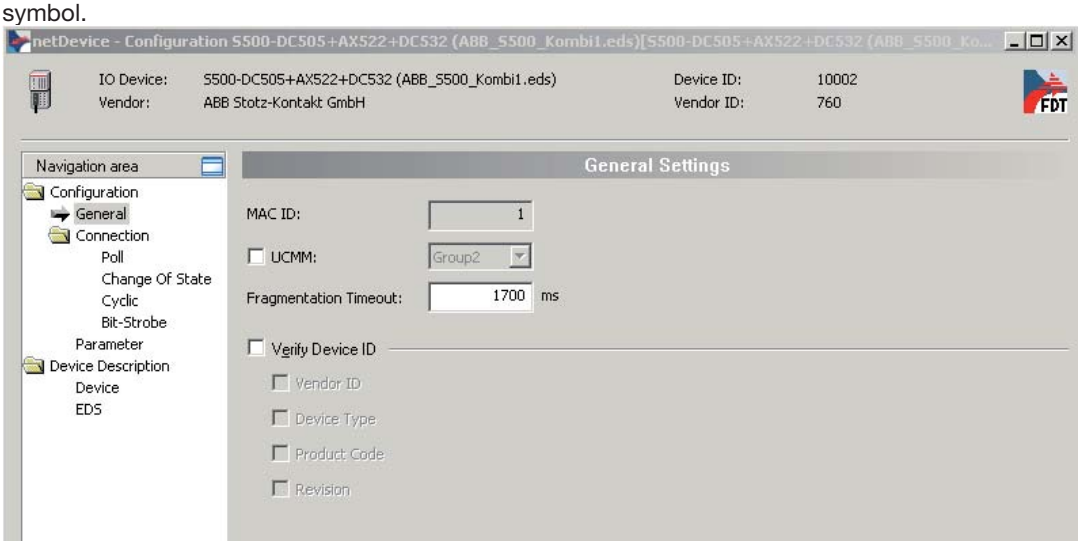
In order to download the parameterization into the bus master, you must activate the corresponding driver in the menu „Device Assignment" (device allocation).



For further settings please refer to the AC500 system manual.

3. Slave parameters

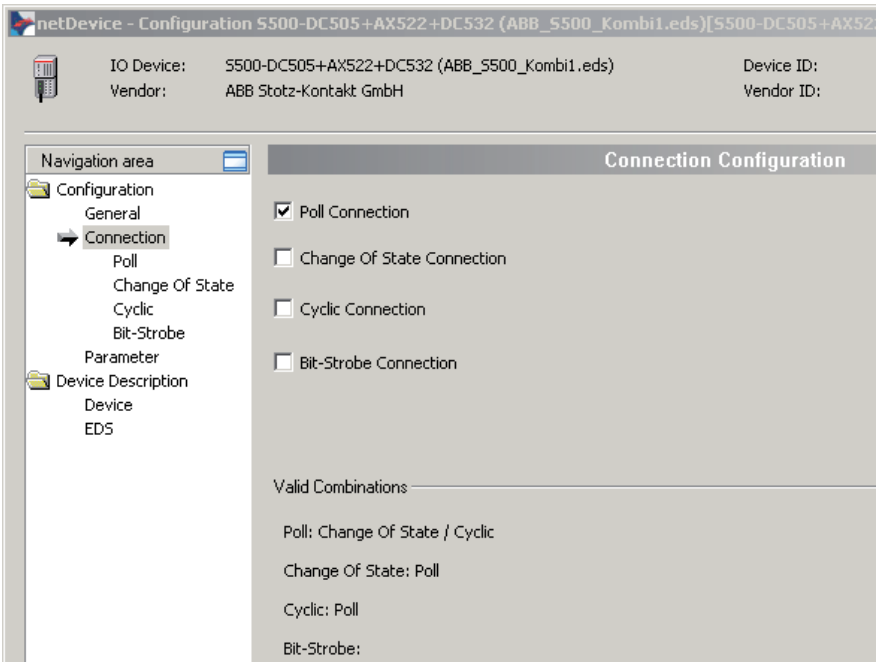
Now you have to set the slave parameters. Open the parameter dialog with doubleclick on the slave symbol.



As described above, you can change the MAC ID (bus address) of a slave only in the dialog of the master settings.

Since the DNP21-FBP is a Group2Only-Slave, you must select Group2 and remove the hook at UCMM in the „General settings".

The further settings, which concern the connecting parameters, you do under the menu option "Connection", according to the requirements of your DeviceNet network.

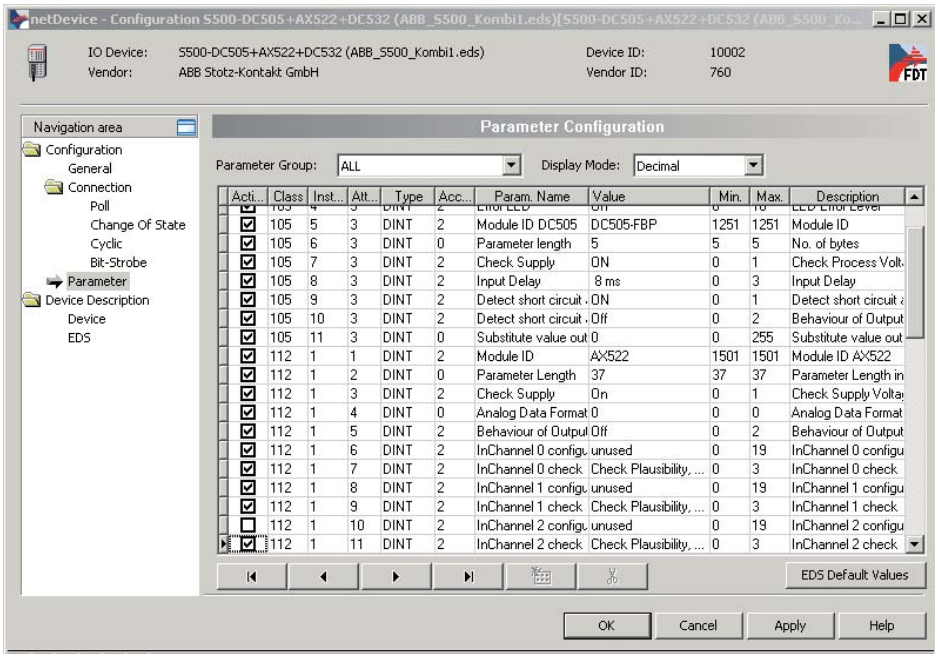


Detailed information for the operation of the SYCON.net you will find in the AC500 system documentation.

The real device parameters are set in the menu „Parameter".

All parameters, which have to be transferred to the device at each bus start, must be marked with a hook in the column „Activating".

For the S500 devices in our example all parameters must be activated!

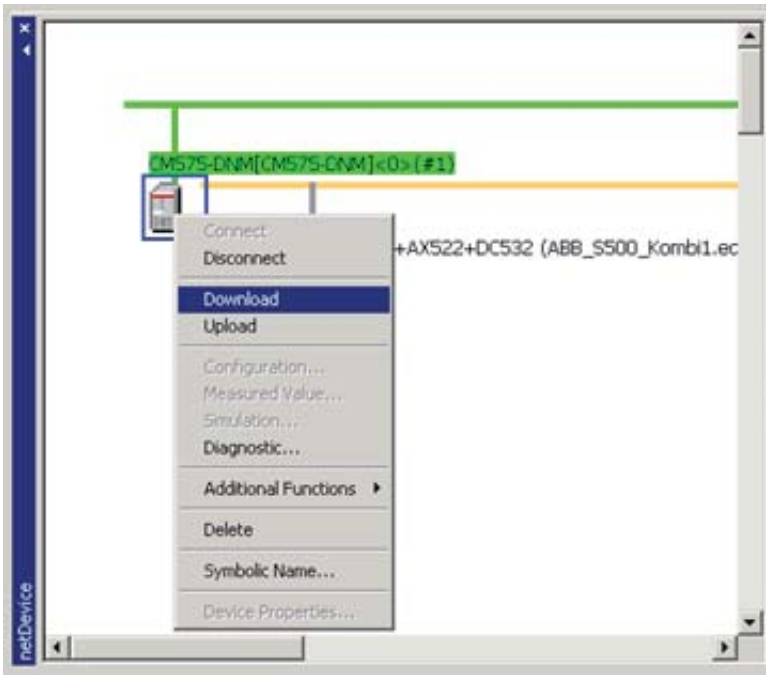


After all settings have been carried out, they are transferred to the current configuration by a click on „OK" or “Apply".

4. Download

If all parameters are set, click with the right mouse button on the symbol of the bus master and select „Connect".

After clicking again with the right mouse button, you can load the set bus and device parameters by „Download" into the master.



They are non-volatile stored there.

DeviceNet definitions

DeviceNet is defined fully in the DeviceNet Specification published by the Open DeviceNet Vendor Association, Inc. (ODVA). (http://www.odva.org).

DeviceNet is based on *CAN*. Refer to the CAN (Controller Area Network) Specification – Version 2.0, Part A, Robert Bosch GmbH, 1991 and ISO 11898: 1993.

CAN ID (or just **identifier**) is an eleven-bit field that identifies each message on any CAN network. CAN IDs must be unique. A major purpose of the DeviceNet specification is to ensure this uniqueness. It is also referred to as the "Connection ID" in DeviceNet.

MAC ID (Medium Access Control IDentification) is more commonly called node number. DeviceNet node numbers have six bits, so can be between 0 and 63.

DeviceNet is a **producer-consumer** network. Nodes **produce** data to the network using particular CAN IDs, and they recognize (other) CAN IDs and **consume** the associated data from the network. Nodes do not necessarily send messages to each other, although such a relationship usually exists. (For explicit messages, it always exists.)

A **connection** is a logical link between a node and one or two CAN IDs (for production and/or consumption). Most nodes have at least two connections (one explicit, one I/O) in their normal operational state. A connection is not a logical link between nodes, although such a relationship usually exists. (For explicit messages, it always exists.)

An **object** is a data table element which has a defined behavior as well as a defined data structure. In other words, different objects respond differently to the same event. From the explicit messaging point of view, the most significant implication is that most services (commands) are not applicable to all objects. See also *instance* below.

A **class** is a group of similar objects. The structure and behavior of a class of objects are defined in its **class definition**. The definitions of open classes are part of the DeviceNet specification. Vendor-specific classes are defined by the vendor. Publication of vendor-specific features is encouraged but not required.

An **instance** is an actual object which has a special meaning. A related verb “to instantiate” means “to create”. The terms “object instance”, “object” and “instance” are all used loosely to refer to an actual object, and are to that extent interchangeable.

An **attribute** is an actual datum that can be read or (in some cases) written. Different attributes of an object can have different data types.

A **service** is a code that defines a request or response (e.g. read, write, error).

An object, class, instance, attribute or service is said to be **open** if it is defined in the DeviceNet specification, or is reserved for future definition. The opposite of open is **vendor-specific**.

Explicit messages are general-purpose messages between two nodes called **client** and **server**. The client sends commands to the server, which sends replies. Each explicit message states (explicitly!) the service code and whichever of class, instance and attribute are relevant

Unlike explicit messages, **implicit** or **I/O (Input/Output)** messages contain only data. The meaning is implicit. Implicit messages are produced by one node and can be consumed by one or more nodes.

Most DeviceNet nodes use the **Predefined Master/Slave Connection Set**. This simplified communications paradigm permits straightforward migration from earlier networks. In this context, controllers such as PLCs and computers are usually masters and devices are usually slaves. It is possible for a node to simultaneously be both master and slave (to a different master, of course). It is possible for a node to simultaneously be a master/slave and also to participate in more sophisticated communications.

The Predefined Master/Slave Connection Set includes four types of I/O connection: Poll, Strobe and COS/Cyclic. In this context, the terms “input” and “output” are used from the master’s point of view. Masters consume input data and produce output data. Slaves consume output data and produce input data.

Poll messaging involves an output message from the master to the slave, which responds with an input message. These messages can be of any reasonable length, including zero.

Strobe (also known as **Bit-Strobe**) messaging involves an output message broadcast by the master to all slaves configured for this type of messaging. Each such slave responds with an input message. The output message has one bit of data for each slave. The input message is from zero to eight bytes long.

COS (Change Of State)/Cyclic messaging has similar characteristics as Poll messaging, except that messages are produced when the data changes or at a fixed rate. Because COS messaging always has a cyclic background (if no change of state occurs for a certain time, a message is produced anyway), the two are treated in many ways as one message type.

Scanner is the common name for a master node, usually associated with a PLC or computer.

Connection originator is the formal name for a node that creates connections, such as a master.

Many simple devices are **UCMM incapable** (or, more correctly, don't have a UCMM). **UCMM** stands for “UnConnected Message Manager”. The UCMM is the most flexible method of creating explicit connections. Omitting the UCMM permits lower cost microprocessors to be used, but means the device must be only a slave and can only communicate directly with its master. The master then provides the UCMM on behalf of the slave, but can only provide ONE such “proxy” connection per slave. This connection may be required at any time by a configuration or troubleshooting tool, so cannot be assumed to be available for other purposes. A device without a UCMM may also be called a **group 2 only server** (because it only receives messages with identifiers in the range known as “group 2”).

A **configuration tool** is a device or computer program that is used to configure DeviceNet nodes. It uses explicit messages to do this. Rockwell Software's RSNetWorx for DeviceNet is an example of such a tool.

EDS Files are text files that contain information about devices that is useful to configuration tools. The format of EDS files is defined in the specification.

A **Path** is a data structure used within one object to refer to another object. Typically, it encodes class, instance and attribute (but can do more). It is also used in EDS files. It is often called a **DeviceNet path** to avoid confusion with, for example, a DOS path.

DeviceNet information to the properties of the DNP12-FBP

General information

The DNP21-FBP (FieldbusPlug for DeviceNet) operates as a slave on the DeviceNet network. The unit supports Explicit Messages and Polled as well as COS/Cyclic I/O Messages of the predefined master/slave connection set. It does not support the Unconnected Message Manager (UCMM).

The number of inputs and outputs supported by the DNP21-FBP depends on the FBP-Device which is connected.

Message types

As a group 2 only slave device, the DNP21-FBP supports the following message types.

CAN Identifier	Group 2 Message Type
10xxxxxx111	Duplicate MAC ID Check Messages
10xxxxxx110	Unconnected Explicit Request Message
10xxxxxx101	Master I/O Poll Command Message
10xxxxxx100	Master Explicit Request Message

xxxxxx = Node address

Class services

As a group 2 only slave device, the DNP21-FBP supports the following class services and instance services.

Service Code	Service Name
14 (0x0E)	Get Attribute Single
16 (0x10)	Set Attribute Single
75 (0x4B)	Allocate Group 2 Identifier Set
76 (0x4C)	Release Group 2 Identifier Set

Object classes

The DNP21-FBP supports the following DeviceNet Classes.

Class	Object
001 (0x01)	Identity
002 (0x02)	Message Router
003 (0x03)	DeviceNet
004 (0x04)	Assembly
005 (0x05)	Connection
043 (0x2B)	Acknowledge Handler
100 (0x64)	ABB Discrete Input
101 (0x65)	ABB Discrete Output
102 (0x66)	ABB Analog Input
103 (0x67)	ABB Analog Output
105 (0x69)	ABB Parameter
112 (0x70)	ABB Parameter Modular Devices
128 (0x80)	ABB Query

Class Code 001 (0x01): Identity Object

The Identity Object is required on all devices and provides identification of and general information about the device. Only one instance (0x01) is supported

Class Attributes

None

Instance Attributes

Attribute	Access	Name	Type	Value
1	Get	Vendor	UINT	760 (0x2F8)
2	Get	Product Type	UINT	12 (0xC)
3	Get	Product Code	UINT	*1)
4	Get	Revision Major Revision Minor Revision	STRUCT OF USINT USINT	 1 3
5	Get	Device Status	UINT	*2)
6	Get	Serial Number	UINT	unique
7	Get	Product Name Length Name	STRUCT OF USINT STRING[Length]	 7 General

*1) The product code is obtained from the device connected to the DNP21-FBP. If a device using parallel communication is connected, the product code is 0x0001.

*2) Device status

byte 0	0x00	not owned
	0x01	owned (allocated)
byte 1	0x00	no fault
	0x04	recoverable fault
	0x08	unrecoverable fault

Common Services

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

Class Code 002 (0x02): Message Router Object

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.

Class Attributes

None

Instance Attributes

None

Common Services

None

Class Code 003 (0x03): DeviceNet Object

The DeviceNet Object is used to provide the configuration and status of a physical attachment to DeviceNet. A product must support only one DeviceNet Object per physical network attachment.

Class Attributes

Attribute	Access	Name	Type	Value
1	Get	Revision	UINT	2

Instance Attributes

Attribute	Access	Name	Type	Value
1	Get/Set *)	MAC ID	USINT	*1)
2	Get	Baud Rate	USINT	*2)
3	Get	Bus Off Interrupt	BOOL	0 *3)
4	Get/Set	Bus Off Counter	USINT	*4)
5	Get	Allocation Information Choice Byte Master Node Address	STRUCT OF BYTE USINT	*5)

*) The MAC ID is settable only if the address of the connected device is greater than 63 or if a parallel device is connected.

*1) The MAC ID is obtained from the device connected to the DNP21-FBP. If a device using parallel communication is connected, the MAC ID will be read from the non-volatile memory.

*2) The DNP21-FBP detects the Baud Rate automatically by means of an implemented autobaud function.

*3) Node will be held in reset if a Bus Off state is encountered.

*4) Bus Off Counter will be forced to 0 whenever set regardless of the data value provided.

*5) Allocation byte

bit 0	Explicit messaging
bit 1	Polled I/O
bit 4	COS I/O
bit 5	Cyclic I/O
bit 6	Acknowledge Suppression

Common Services

Service Code	Class	Instance	Service Name
14 (0x0E)	Yes	Yes	Get_Attribute_Single
16 (0x10)	No	Yes	Set_Attribute_Single
75 (0x4B)	No	Yes	Allocate_Master/Slave
76 (0x4C)	No	Yes	Release_Master/Slave

Class Code 004 (0x04): Assembly Object

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

Class Attributes

None

Instance Attributes

Assembly Object, **Instance 100**, Attributes

Attribute	Access	Name	Type	Value
3	Get/Set	Producing Data	ARRAY OF [n] BYTE	*1)

*1) The size of the data array depends on the Produced Data size of the device connected to the DNP21-FBP.

Assembly Object, **Instance 101**, Attributes

Attribute	Access	Name	Type	Value
3	Get/Set	Consuming Data	ARRAY OF [n] BYTE	*1)

*1) The size of the data array depends on the Consumed Data size of the device connected to the DNP21-FBP. For detailed information, please refer to the device manual.

Common Services

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

Class Code 005 (0x05): Connection Object

The Connection Class allocates and manages the internal resources associated with both I/O and Explicit Messaging connections. The specific instance generated by the Connection Class is referred to as a Connection Instance or a Connection Object.

Three instances of the Connection Class will be supported. Instance 1 will be the Explicit Message Connection, instance 2 will be the Polled I/O Connection, and instance 4 will be the COS/Cyclic IO Connection.

Class Attributes

None

Instance Attributes

Connection Object, **Instance 1 = Explicit Message Connection**, Attributes

Attribute	Access	Name	Type	Value
1	Get	State	USINT	*1)
2	Get	Instance Type	USINT	**0 = Explicit Message**
3	Get	Transport Class Trigger	BYTE	0x83 - Server, Transport Class 3
4	Get	Produced Connection ID	UINT	10xxxxxx011 *2)
5	Get	Consumed Connection ID	UINT	10xxxxxx100 *2)
6	Get	Initial Comm. Characteristics	BYTE	0x21
7	Get	Produced Connection Size	UINT	*3)
8	Get	Consumed Connection Size	UINT	*3)
9	Get/Set	Expected Packet Rate	UINT	in ms
12 (0x0C)	Get/Set	Watchdog Timeout Action	USINT	01 = auto delete
13 (0x0D)	Get	Produced Conn. Path Length	UINT	0
14 (0x0E)	Get	Produced Connection Path	EPATH	Empty
15 (0x0F)	Get	Consumed Conn. Path Length	UINT	0
16 (0x10)	Get	Consumed Connection Path	EPATH	Empty
17 (0x11)	Get	Production Inhibit Time	UINT	0

*1) Connection state

0	nonexistent
1	configuring
2	wait for connection ID
3	established
4	timed out
5	deferred

*2) xxxxxx = node address

*3) The size of Produced and Consumed Connection depends on the device connected to the DNP21-FBP. For detailed information, please refer to the device manual.

Connection Object, **Instance 2 = Polled I/O Message Connection**, Attributes

Attribute	Access	Name	Type	Value
1	Get	State	USINT	*1)
2	Get	Instance Type	USINT	**1 = I/O Connection**
3	Get	Transport Class Trigger	BYTE	0x82 - Server, Transport Class 2 *4)
4	Get	Produced Connection ID	UINT	01111xxxxxx *2)
5	Get	Consumed Connection ID	UINT	10xxxxxx101 *2)
6	Get	Initial Comm. Characteristics	BYTE	0x01
7	Get	Produced Connection Size	UINT	*3)
8	Get	Consumed Connection Size	UINT	*3)
9	Get/Set	Expected Packet Rate	UINT	in ms
12 (0x0C)	Get	Watchdog Timeout Action	USINT	0 = time out
13 (0x0D)	Get	Produced Conn. Path Length	UINT	6
14 (0x0E)	Get	Produced Connection Path	EPATH	20 04 24 64 30 03
15 (0x0F)	Get	Consumed Conn. Path Length	UINT	6
16 (0x10)	Get	Consumed Connection Path	EPATH	20 04 24 65 30 03
17 (0x11)	Get	Production Inhibit Time	UINT	0

*1) Connection state

0	nonexistent
1	configuring
2	wait for connection ID
3	established
4	timed out
5	deferred

*2) xxxxxx = node address

*3) The size of Produced and Consumed Connection depends on the device connected to the DNP21-FBP. For detailed information, please refer to the device manual.

*4) If alloc_choice = polled and ack suppression is enabled then value = 0x80.

Connection Object, **Instance 4 = COS/Cyclic I/O Message Connection**, Attributes

Attribute	Access	Name	Type	Value
1	Get	State	USINT	*1)
2	Get	Instance Type	USINT	**1 = I/O Connection**
3	Get	Transport Class Trigger	BYTE	*4)
4	Get	Produced Connection ID	UINT	01101xxxxxx *2)
5	Get	Consumed Connection ID	UINT	10xxxxxx101 *2)
6	Get	Initial Comm. Characteristics	BYTE	0x01 (acknowledged) 0x0F (unacknowledged)
7	Get	Produced Connection Size	UINT	*3)
8	Get	Consumed Connection Size	UINT	*3)
9	Get/Set	Expected Packet Rate	UINT	in ms
12 (0x0C)	Get	Watchdog Timeout Action	USINT	0 = time out
13 (0x0D)	Get	Produced Conn. Path Length	UINT	6
14 (0x0E)	Get	Produced Connection Path	EPATH	20 04 24 64 30 03
15 (0x0F)	Get	Consumed Conn. Path Length	UINT	6
16 (0x10)	Get	Consumed Connection Path	EPATH	20 04 24 65 30 03
17 (0x11)	Get/Set	Production Inhibit Time	UINT	in ms

*1) Connection state

0	nonexistent
1	configuring
2	wait for connection ID
3	established
4	timed out
5	deferred

*2) xxxxxx = node address

*3) The size of Produced and Consumed Connection depends on the device connected to the DNP21-FBP. For detailed information, please refer to the device manual.

*4) Transport Class Trigger

0x00	Cyclic, unacknowledged
0x02	Cyclic, acknowledged
0x10	COS, unacknowledged
0x12	COS, acknowledged

Common Services

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

Class Code 043 (0x2B): Acknowledge Handler Object

The Acknowledge Handler Object is used to manage the reception of message acknowledgements. This object communicates with a message producing Application Object within a device. The Acknowledge Handler Object notifies the producing application of acknowledge reception, acknowledge timeouts and production retry limit.

Class Attributes

None

Instance Attributes

Attribute	Access	Name	Type	Value
1	Get/Set	Acknowledge Timeout	UINT	in ms
2	Get	Retry Limit	USINT	1
3	Get	COS Producing Connection Instance	UINT	4

Common Services

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

Class Code 100 (0x64): ABB Discrete Input Object

The ABB Discrete Input Object models the discrete inputs of the FBP device connected to the DNP21-FBP. 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.

Class Attributes

Attribute	Access	Name	Type	Value
2	Get	Number of Discrete Inputs	USINT	*1)

*1) The number of discrete inputs depends on the device connected to the DNP21-FBP. For detailed information, please refer to the device manual.

Instance Attributes

Attribute	Access	Name	Type	Value
3	Get	Discrete Input Value	BOOL	Input State

Common Services

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

Class Code 101 (0x65): ABB Discrete Output Object

The ABB Discrete Output Object models the discrete outputs of the FBP device connected to the DNP21-FBP. You can use this object in applications as simple as an actuator or as complex as a discrete I/O control module.

There is a separate instance for each discrete output available on the device.

Class Attributes

Attribute	Access	Name	Type	Value
2	Get	Number of Discrete Outputs	USINT	*1)

*1) The number of discrete outputs depends on the device connected to the DNP21-FBP. For detailed information, please refer to the device manual.

Instance Attributes

Attribute	Access	Name	Type	Value
3	Get/Set	Discrete Output Value	BOOL	Output State

Common Services

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

Class Code 102 (0x66): ABB Analog Input Object

The ABB Analog Input Object models the analog inputs of the FBP device connected to the DNP21-FBP.

There is a separate instance for each analog input available on the device.

Class Attributes

Attribute	Access	Name	Type	Value
2	Get	Number of Analog Inputs	USINT	*1)

*1) The number of analog inputs depends on the device connected to the DNP21-FBP. For detailed information, please refer to the device manual.

Instance Attributes

Attribute	Access	Name	Type	Value
3	Get	Analog Input Value	BOOL	Input State

Common Services

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

Class Code 103 (0x67): ABB Analog Output Object

The ABB Analog Output Object models the analog outputs of the FBP device connected to the DNP21-FBP.

There is a separate instance for each analog output available on the device.

Class Attributes

Attribute	Access	Name	Type	Value
2	Get	Number of Analog Outputs	USINT	*1)

*1) The number of analog outputs depends on the device connected to the DNP21-FBP. For detailed information, please refer to the device manual.

Instance Attributes

Attribute	Access	Name	Type	Value
3	Get/Set	Analog Output Value	BOOL	Output State

Common Services

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

Class Code 105 (0x69): ABB Parameter Object

The ABB Parameter Object provides the parameter for **non-modular** devices and for the adapter module (slot 0) of modular devices. The index always corresponds to the parameter number. The attribute is fixed to 3.

The use of the ABB Parameter Object provides a known public interface to a device's configuration data. In addition, this object also provides all the information necessary to define and describe each of a device's individual configuration parameters.

This object allows a device to fully identify a configurable parameter by supplying a full description of the parameter, including minimum and maximum values and a human-readable text string describing the parameter.

The complete description of an FBP device's parameters can be found in the device manual.

Configuration tools obtain the parameter description automatically from the EDS file.

Class Attributes

Attribute	Access	Name	Type	Value
2	Get	Number of Parameters (Max. Instance)	USINT	*1)

*1) The number of parameters depends on the device connected to the DNP21-FBP. For detailed information, please refer to the device manual.

Instance Attributes

Attribute	Access	Name	Type	Value
3	Get/Set *2)	Parameter Value	LINT	Actual value

*2) The Set service is not available for read-only parameters.

Common Services

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

Class Code 112 (0x70): ABB Parameter Modular Object

The ABB Parameter Modular Object provides the parameter for **modular** expansion devices which are connected to an adapter module. The index always corresponds to the slot number (beginning with slot 1) and the attribute corresponds to the parameter number of the module plugged into the slot.

The use of the ABB Parameter Modular Object provides a known public interface to a device's configuration data. In addition, this object also provides all the information necessary to define and describe each of a device's individual configuration parameters.

This object allows a device to fully identify a configurable parameter by supplying a full description of the parameter, including minimum and maximum values and a human-readable text string describing the parameter.

The complete description of a modular FBP device's parameters can be found in the device manual.

Configuration tools obtain the parameter description automatically from the EDS file.

Class Attributes

Attribute	Access	Name	Type	Value
2	Get	Max. Number of Expansion Modules	USINT	7

Instance Attributes

Attribute	Access	Name	Type	Value
3	Get/Set *1)	Parameter Value	LINT	Actual value

*1) The Set service is not available for read-only parameters.

Common Services

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

Class Code 128 (0x80): ABB Query Object

The ABB Query Object provides an array filled with the External IDs of all modules connected to the adapter module. It is used for modular devices only

By means of this Object, configuration tools are able to read the module configuration from the DNP21-FBP.

Class Attributes

None

Instance Attributes

Attribute	Access	Name	Type	Value
1	Get	Query External ID List	ARRAY [0..7] OF USINT	External IDs of the expansion modules actually connected *1)

*1) Empty slots are represented by the external ID "00 00".

Common Services

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

Technical data

Supply voltages	
DeviceNet power supply unit	24 V DC ± 1% (required acc. to DeviceNet specification)
FieldBusPlug works correctly at	DeviceNet supply voltage = 11.0 ... 24.7 V DC
Required power line failure bridging time of the power supply unit	10 ms min.
Recommended power supply unit	ABB Type: CP-C 24/5.0 (22-28 V / 5 A) Order number: 1SVR 427 024 R0000
Current consumption	
from DeviceNet power supply unit	DNP21-FBP: 18.5 mA (24 V) typ. If also the terminal device is supplied by the DeviceNet power supply unit (switch position on the device = INTERNAL), the load must not exceed 200 mA.
Mounting	on the terminal device, fixed with a screw (provided on delivery) or by M12 box nut fixing
Setup of a DeviceNet bus (or a section)	by connecting the FieldBusPlugs in series (first bus plug to coupler/master, second bus plug to socket of the first FieldBusPlug, etc.)
Bus terminating resistors	120 Ω at each end of the bus
Modes of data communication between FieldBusPlug and terminal device	parallel and serial
Scope of data	according to DeviceNet specifications
Construction of FieldBusPlug cable	Round cable, black, 2 x 0.34 mm^2 for supply voltage 2 x 0.25 mm^2 for data lines 3 shields
Load capacity of plugs and cables	4 A max.
Pin assignment of the interfaces	see Figure 3
Degree of protection (see also Figure 3)	IP 65, if M12 box nut fixing is used at the terminal device (e.g. sensor) IP 20, if mounting is performed using the supplied fastening screw (e.g. for MSD11-FBP)
Ambient temperature	
storage	-20...+75 °C
operation	0...+55 °C
Dimensions	see Figure 7
Total power dissipation of the unit	
DNP21-FBP	0.525 W max.
Weight plug with cable 0.25 m	0.09 kg
plug with cable 0.5 m	0.10 kg
plug with cable 1 m	0.13 kg
plug with cable 5 m	0.35 kg
Bus address setting	by special software of the PLC manufacturer or using a manual addressing unit with PC connection
Possible addresses	1 to 61 (0, 62 and 63 are reserved)
Diagnosis (see Figure 4)	4 LEDs on the front plate
green LED, red LED	network status
green LED, red LED	module status

Ordering data

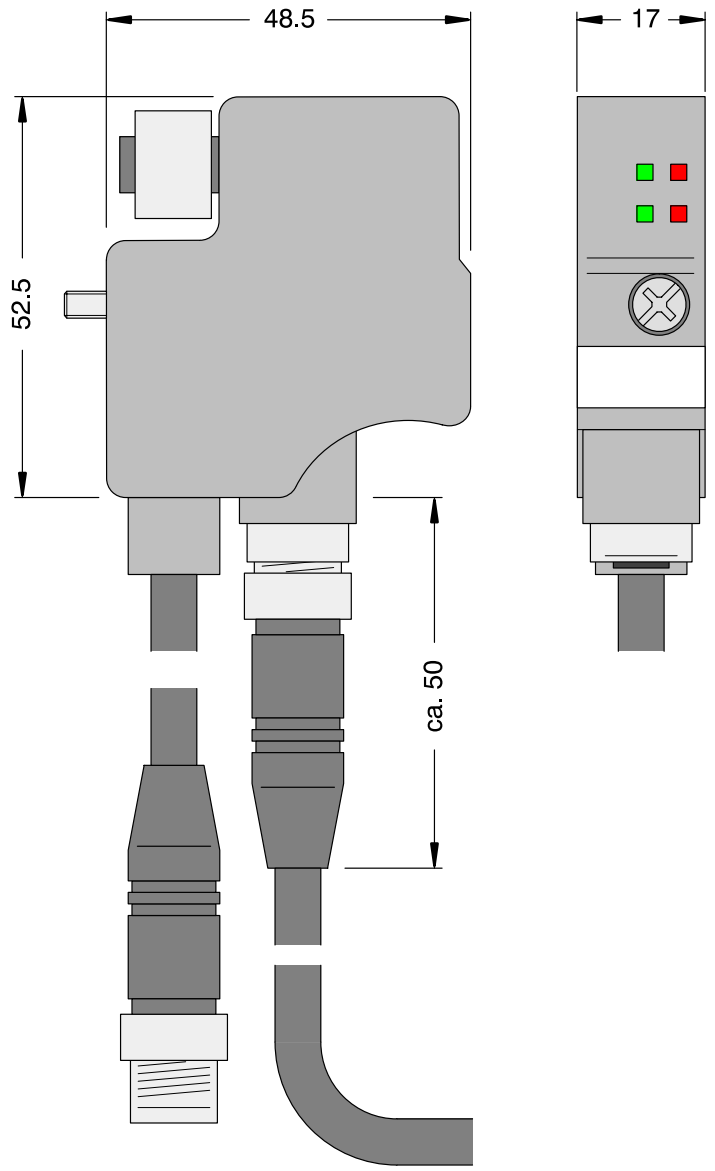
A fastening screw, an address label and a terminal cap for the bus are supplied along with the FieldBusPlug.

Type	Designation	Order number
DNP21-FBP.025	DeviceNet FieldBusPlug, cable length 0.25 m	1SAJ 230 000 R0003
DNP21-FBP.050	DeviceNet FieldBusPlug, cable length 0.5 m	1SAJ 230 000 R0005
DNP21-FBP.100	DeviceNet FieldBusPlug, cable length 1 m	1SAJ 230 000 R0010
DNP21-FBP.500	DeviceNet FieldBusPlug, cable length 5 m	1SAJ 230 000 R0050

Accessories

Type	Designation	Order number
DNX11-FBP.100	DeviceNet extension cable, length 1 m	1SAJ 923 001 R0010
DNX11-FBP.300	DeviceNet extension cable, length 3 m	1SAJ 923 001 R0030
DNX11-FBP.500	DeviceNet extension cable, length 5 m	1SAJ 923 001 R0050
DNF11-FBP.050	DeviceNet round cable, female plug attached at one end, 0.5 m, sheath partially removed, wire-end ferrules attached	1SAJ 923 002 R0005
DNM11-FBP.050	DeviceNet round cable, male plug attached at one end, 0.5 m, sheath partially removed, wire-end ferrules attached	1SAJ 923 003 R0005
DNC11-FBP.999	DeviceNet round cable on 100 m coil	1SAJ 923 004 R1000
DNM11-FBP.0	Male DeviceNet connector for round cable	1SAJ 923 005 R0001
DNF11-FBP.0	Female DeviceNet connector for round cable	1SAJ 923 006 R0001
DNR11-FBP.120	DeviceNet terminating resistor	1SAJ 923 007 R0001

Mechanical dimensions



All dimensions in mm.

Figure 7: DNP21-FBP, mechanical dimensions

Manual No. 2CDC 193 001 D0205

ABB STOTZ-KONTAKT GmbH

Eppelheimer Straße 82
69123 Heidelberg
Germany

Postfach 101680
69006 Heidelberg
Germany

Telephone +49 (0) 6221 701- 0
Telefax +49 (0) 6221 701- 240
E-mail desst.helpline@de.abb.com
Internet http://www.abb.de/stotz-kontakt