Quick-Start Guide

AUTOMATION

WAGO *SPEEDWAY 767*

ETHERNET, 8 DI, 24 V DC

767-1301

Fieldbus Coupler

Version 1.0.0

WAGO®
INNOVATIVE CONNECTIONS

Introduction

Every conceivable measure has been taken to ensure the accuracy and completeness of this documentation. However, as errors can never be fully excluded, we always appreciate any information or suggestions for improving the documentation. E-mail: documentation@wago.com

Service and Technical Support

Additional information regarding this and other products (e.g., data sheets) is available on our website www.wago.com.

If you can not eliminate errors or faults using the measures described in this manual we will be glad to assist you further. Contact us at:

AUTOMATION Support
Phone: +49 571 887 555
Fax: +49 571 887 8555
E-mail: support@wago.com

Additional Support

The Seminar and Training department offers useful seminars to support you in the use of WAGO products. For more information, visit our website or call +49 571 887-327, or send an e-mail to training@wago.com.

Trademarks

We wish to point out that the software and hardware terms as well as the trademarks of companies used and/or mentioned in the present manual are generally protected by trademark or patent.

Table of Contents

1 Safety

1.1 Notes on This Quick-Start Guide

The fieldbus coupler shall only be installed and operated in conjunction with these quick-start instructions and the system description. Detailed information on the coupler is provided in the 767-1301 manual.

⚠ WARNING

Observe release notes!
Please note that, within the SPEEDWAY system, a function is provided **without restriction** only if all system's components have the same system-wide firmware release.Therefore, always observe the appropriate release notes on products used.

NOTICE

Supply layout!
In addition to these operating instructions, you will need the "WAGO SPEEDWAY 767, System Description and Information" manual, which can be downloaded at www.wago.com. There you will find information regarding supply layout, etc.

1.2 Explanation of Symbols

⚠ DANGER

Warning of physical injury
Indicates a direct hazard with a high level of risk which leads to death or serious physical injury if not avoided.

⚠ DANGER

Warning of physical injury from electric current
Indicates a direct hazard with a high level of risk which leads to death or serious physical injury if not avoided.

⚠ WARNING

Warning of physical injury
Indicates a possible hazard with a moderate level of risk, which may lead to death or (serious) physical injury if not avoided.

⚠ CAUTION

Warning of physical injury
Indicates possible hazards with a low level of risk, which could lead to minor or moderate physical injuries if not avoided.

NOTICE

Warning of damage to equipment
Indicates a possible hazard that could lead to equipment damage if not avoided.

NOTICE

Warning of damage to equipment from electrostatic discharge
Indicates a possible hazard that could lead to equipment damage if not avoided.

Note

Please note
Indicates possible malfunction, which does not lead to equipment damage if it is not avoided.

Information

Reference to additional information
Indicates other sources of information which are not an integral part of this documentation, such as the Internet.

1.3 Validity of This Quick-Start Guide

These operating instructions are only applicable to the positive-switching WAGO SPEEDWAY 767 Series fieldbus coupler, item number 767-1301.

1.4 Use in Compliance with Underlying Provisions

The EtherNet/IP fieldbus coupler is used to record and process digital and analog field signals. The data is gathered by the fieldbus coupler and made available to a higher-level controller for further processing.

The fieldbus coupler shall not be used to control safety-related functions; i.e., emergency-off devices shall not be operated with this fieldbus coupler.

The fieldbus coupler shall only be used as a unit or in combination with I/O modules from the WAGO SPEEDWAY 767 Series.

The fieldbus coupler was developed for applications requiring IP 67 (NEMA type 6, 6P) protection.

The fieldbus coupler is expandable by a maximum of 64 I/O modules from the WAGO SPEEDWAY 767 Series.

Applications other than those described in this manual are not permitted.

1.5 Personnel Qualification

All sequences implemented on the fieldbus coupler may only be carried out by electrical specialists with sufficient knowledge in automation. The specialists must be familiar with the current standards and guidelines for the fieldbus coupler and automation environment.

1.6 Basic Safety Information

This section contains a summary of the most important warnings, which are also repeated in the individual sections. They serve as a protection to your health and a protection from equipment damage on the 767 Series components (fieldbus coupler and the I/O modules connected to it). Read and adhere to the following safety precautions before using the fieldbus coupler.

⚠ DANGER

Electric voltage!
Operate the 767 Series components exclusively with 24VDC PELV (Protective Extra Low Voltage) or SELV (Safety Extra Low Voltage) voltage sources. Failure to comply may result in electric shock.

⚠ CAUTION

Hot connection sockets!
Even when taking into account derating, high surface temperatures on the metallic connection sockets and on the enclosure can arise during operation. If the 767 Series component has been in operation, allow it to cool off before moving it.

NOTICE

The highest current carrying capacity of the supply contacts is 4A!
Always observe the maximum current carrying capacity per supply line (U_{LS}, U_A) for each 767 Series component and the overall power consumption for all 767 Series component. Neither of these values shall exceed 4A since an increase in current causes the contacts to overheat and damages the 767 Series components. Information regarding the power demand of each 767 Series component can be found in the corresponding data sheet, which is available from www.wago.com.

NOTICE

Exposed connections!
If connections have not been closed with protective covers, liquid or dirt can penetrate the fieldbus coupler and ruin it. Close all unneeded connections with protective caps to ensure that IP 67 degree of protection is provided.

- Disconnect the power supply from the system on which you wish to mount the 767 Series component.
- Always keep the cover cap of the DIP switch closed.
- Observe the appropriate accident prevention regulations for your system during assembly, startup, maintenance and repairs.
- The operating instructions of the 767 Series components and the system description must be readily available at the workplace.

- Observe the exact positioning (coding) between plug and socket.
- The 767 Series component shall not come in contact with substances having seeping and insulating properties. Otherwise, additional measures shall be taken for the devices, such as installation of an enclosure that is resistant to the above-mentioned substance properties.
- Electronic components fulfilling the ESD requirements according to the IEC 61000-6-2 are integrated in the 767 Series component. Since, under unfavorable circumstances, higher voltages may also occur due to electrical charge in the field, discharge must be ensured before performing work on the 767 system.
- Observe the correct layout of the potential equalization.
- Keep all cables a sufficient distance away from electromagnetic sources of interference in order to maintain a high level of interference resistance of the 767 system against electromagnetic emissions. Use only shielded cables at the necessary locations, and always observe the appropriate standards for EMC-suitable installations.
- For the power supply and for the S-BUS, use only pre-assembled WAGO system cables, so the specified characteristics of the technical data can be achieved.
- Replace defective or damaged 767 Series component (e.g., deformed connections), else function disruptions can occur in the respective fieldbus stations or nodes.
- When laying any cables, make sure that you do not lay them within the shear range of movable machine parts.
- For each activity, observe the corresponding personnel qualification in Section 1.5.
- Observe the marking on the front and rear side of the 767 Series component.

1.7 Safety Equipment

All 767 Series products are designed according to the IP 67 safety class. This consists of, among other things, complete touch protection of electric voltages and currents – even when wet.

1.8 Technical Condition of the 767 Components

If any change is made to the 767 Series components or software and firmware without the written approval of WAGO Kontakttechnik GmbH & Co. KG, all liability claims are nullified.

1.9 Notes on Operation

When integrating the 767 Series components in your machine or system, all the currently applicable norms, regulations and guidelines shall be observed during all activities. The emergency stop equipment shall remain effective in all operating modes of the system and machine.

For protection from electromagnetic interferences

- Connect your system to protective earth (PE), and
- Ensure that the cable routing and the installation of the fieldbus cable, S-BUS cable, supply cable, and sensor cable are correct.

The following elements for 24V supply shall be present:

- Outer lightning protection on buildings
- Inner lightning protection of supply lines and signal lines
- Safe electrical separation of low voltage 24VDC through PELV (Protective Extra Low Voltage) or SELV (Safety Extra Low Voltage) voltage sources

2 Description

This quick-start guide describes the most important steps for starting the fieldbus coupler via digital output module.

2.1 Accessories

This manual was created using the following software and hardware:

Name	Beschreibung	Best.-Nr./Version
WAGOframe CD-ROM Ver. 3.0.0	FDT/DTM frame application	759-370 Ver. 1.0.8
WAGO USB driver	USB driver for 767 Series	759-922 Ver. 1.3.2
WAGO service interface DTM	DTM for communication	759-371 Ver. 2.1.0
WAGO DTM for fieldbus coupler and I/O modules	Device DTM for 767 series (programmable) fieldbus coupler and I/O modules	759-361 Ver. 2.1.0
System update DTM	DTM for firmware update	759-362 Ver. 1.0.0
Fieldbus coupler	FC, 8DI, 24 V DC (8xM8)	767-1301
Digital output module	8DO, 24V DC 0.5A (8xM8)	767-4801
USB communication cable (WAGOframe, CoDeSys 3)	M8 plug, straight	756-4101/0042-0030
Ethernet cable (D coded), with one end of the cable fitted	M12 right angle plug, 2 m	756-1202/0060-0200
Ethernet plug	RJ-45, IP 20	750-975
S-BUS cable (B coded) with both ends of cable fitted	M12 socket, right angle, M12 plug, right angle, 0.2 m	756-1306/0060-0002
S-BUS terminating plug (B coded)	M12 plug, straight	756-9409/0060-0000
Power supply cable (A coded), both ends of cable fitted	M12 socket, right angle, M12 plug, right angle, 0.2 m	756-3106/0040-0002

When placing an order, also take into account an additional power supply cable between the power supply unit and the fieldbus coupler, in addition to the number of I/O modules needed.

3 Connection

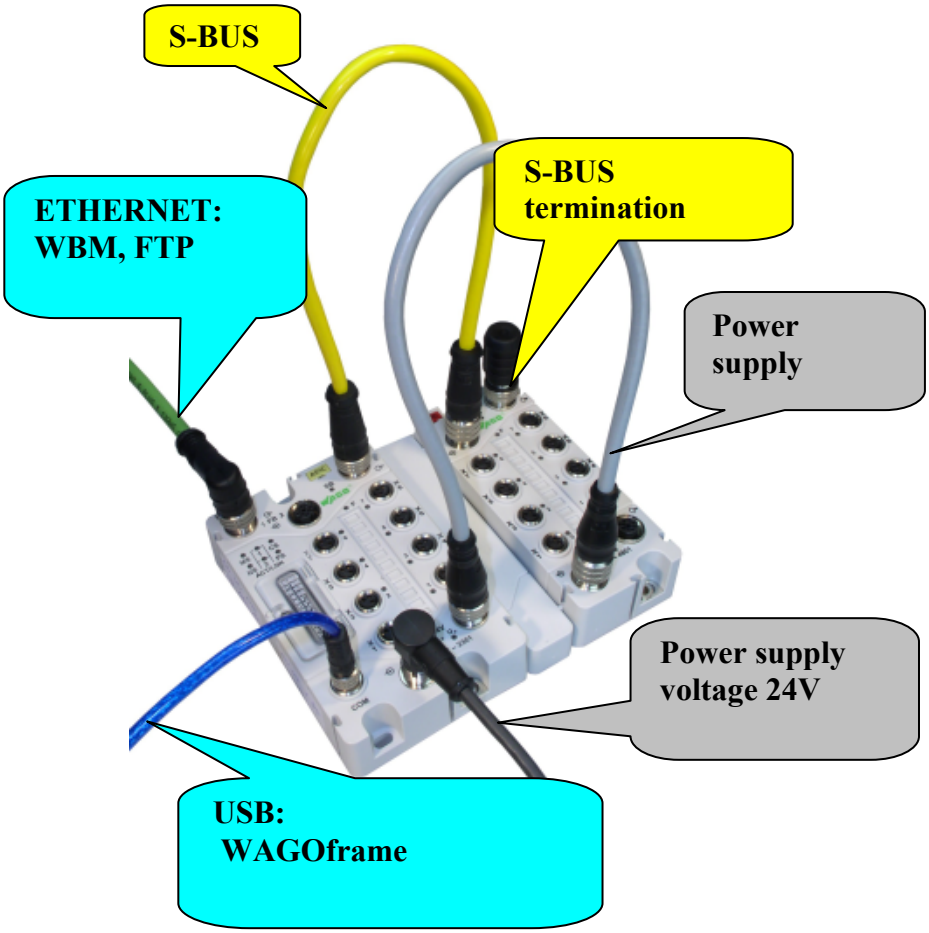
NOTICE

Exposed connections!
If connections have not been closed with protective covers, liquid or dirt can penetrate the fieldbus coupler and ruin it. Close all unneeded connections with protective caps to ensure that IP 67 degree of protection is provided.

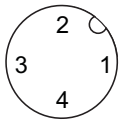
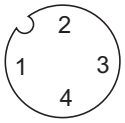
Then tighten the connector by hand when power is switched off. Using mechanical means to tighten it may strip the threads. In such a case, the fieldbus coupler is to be replaced.

Tightening torques for the connectors: 0.6 Nm

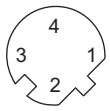
Wire the 767 components as shown in the following figure:



Supply voltage (M12 connectors, A coded, 4 poles):
Power supply feed-in for logic circuits and outputs is provided via separate contacts.

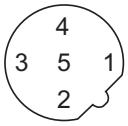
Connection		Contact	Description
IN 	OUT 	1	24VDC U_{LS}
		2	24VDC U_A
		3	0V U_{LS}
		4	0V U_A

Ethernet:
Connect the Ethernet cable fitted at one end as follows to implement a patch cable for use at a switch or a hub:

Connection	Contact	Color	Color	Contact
IN 	**1 (TX+)**	Yellow	Green-white	**1 (TX+)**
	2 (RX+)	Blue	Green	**2 (TX-)**
	3 (TX-)	Brown	Yellow-white	**3 (RX+)**
	4 (RX-)	White	Yellow	**6 (RX-)**
	Thread	Shielded		**Shielded**

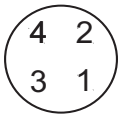
Connect TX to RX to implement a crossover cable for use with a PC.

S-BUS:
The system bus is used for internal communication between the fieldbus coupler and the 767 Series I/O modules connected to it.

Connection	Contact	Description
OUT 	1	TD +
	2	TD -
	3	RD -
	4	RD +
	5	0VDC
	Connecting thread	Shielded

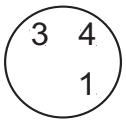
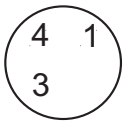
Service port (USB):
The fieldbus coupler provides the following service at the service port:

- I/O-Service
  The function "I/O-Service" is used by the FDT/DTM frame application "WAGOframe" to assign parameters to and configure the fieldbus coupler.

Connection	Contact	Description
IN 	1	+ 5V
	2	- Data
	3	+ Data
	4	0VDC
	Connecting thread	Shielded

The USB Speedway device driver is part of the installation programs of the "WAGOframe" application. For more information, see Chapter 7.1.

Digital inputs:
The sensor cables provide power to the connected sensors and transfer the sensor signals. The table below outlines the connection assignments for the sensor connections:

Connection		Contact	Description
IN	IN	1	24VDC (supplied from U_{LS})
 X1, X3, X5, X7	 X2, X4, X6, X8	3	0VDC
		4	Input

NOTICE

The highest current carrying capacity of the supply contacts is 4A!
Ensure that the sensors from the U_{LS} supply line are supplied with power. The sensor's power consumption is to be taken into consideration when determining the present power demand for the U_{LS} supply line.

NOTICE

The total maximum power consumption of the sensors must not exceed 400mA (50mA/channel).
Please note that the combined power consumption of all connected sensors is not to exceed 400mA. The distribution of power among the existing connections is depending on the individual power requirements of the sensors.

4 EtherNet/IP

4.1 EDS File

The EDS file contains the characteristics of the fieldbus coupler and information regarding its communication capabilities. The EDS file is imported and installed by the corresponding configuration software.

The EDS file can be obtained at www.wago.com. When installing the EDS file, please refer to the information provided in the documentation of the configuration software you are using.

4.2 Process Image

After starting up the fieldbus coupler, it automatically identifies all connected I/O modules. The fieldbus coupler uses this to create a local process image, divided into an input and output data range.

If you have selected the fieldbus EtherNet/IP for the fieldbus coupler, the process image is structured with word alignment. Data larger than one byte is displayed internally in Intel format ("Little Endian"). If both analog and digital I/O modules are connected to the node, the digital input and output data are summarized byte by byte and arranged after the analog data in the process image.

The size of the process image of an Ethernet node can be identified from the connected 767 Series component. The process image has a total size of 4096 bytes: 2048 bytes of input data and 2048 bytes of output data.

Note

The input and output process images can accommodate more data than your higher-level controller can transmit. If the data is larger than the telegram of the higher-level controller, use the assembly instances to select which data should be transmitted.

You can obtain access to the process image via the assembly instances or directly via the EtherNet/IP objects. Please see Sections 4.6 and 4.7 for more information.

4.3 Process Data Exchange

The assembly instances are required in order to exchange process data between the higher-level controller and the fieldbus coupler. Using these assembly instances, you can write and read the inputs and outputs of the 767 Series components.

4.3.1 Assembly Instances

A process data exchange is initiated using the assembly instances. From the existing static assembly instances (see Section 4.6.3), select one instance for each sending direction.

For transmission of process data from the higher-level controller (originator) to the fieldbus coupler (target), the following assembly instances are available:

- Instance 101: for analog and digital output data
- Instance 102: for digital output data
- Instance 103: for analog output data
- Instance 113: for analog and digital output data with diagnostic acknowledgement

For transmission of process data from the fieldbus coupler to the higher-level controller, the following assembly instances are available:

- Instance 104: for analog and digital input data, including status byte
- Instance 105: for digital input data, including status byte
- Instance 106: for analog input data, including status byte
- Instance 107: for analog and digital input data
- Instance 108: for digital input data
- Instance 109: for analog input data
- Instance 112: for analog and digital input data with status byte, current error and diagnostics

Note

If you select an assembly instance without status information (107 – 109), information such as fieldbus errors, pending diagnostics and process data status (valid/invalid) are not available and cannot be integrated into the application of the higher-level controller. The status byte is described in class 64_{hex}, attribute 5.

4.3.2 Identifying the Data Length

After you have selected the necessary assembly instance, you need to identify the length of the data in each sending direction of the respective instance for the configuration software of the higher-level controller. You have the following options:

Reading out the lengths via the assembly instances

The simplest option involves reading out the instance attribute 4 from each assembly instance between 101 and 113, which should be used. Attribute 4 of the assembly classes (ID = 4) indicates the total number of data bytes transmitted in this direction. This number includes the overall size of the respective input or output data, including status byte, error and diagnostics. The individual data sizes are not identified by reading out the assembly instances. This is only possible using the instance attribute of class 64_{hex}.

Reading out the lengths via class 64_{hex}, instance 1

The bit lengths of the analog and digital I/O modules connected to the Ethernet node are read out via the instance attributes 7 – 10.

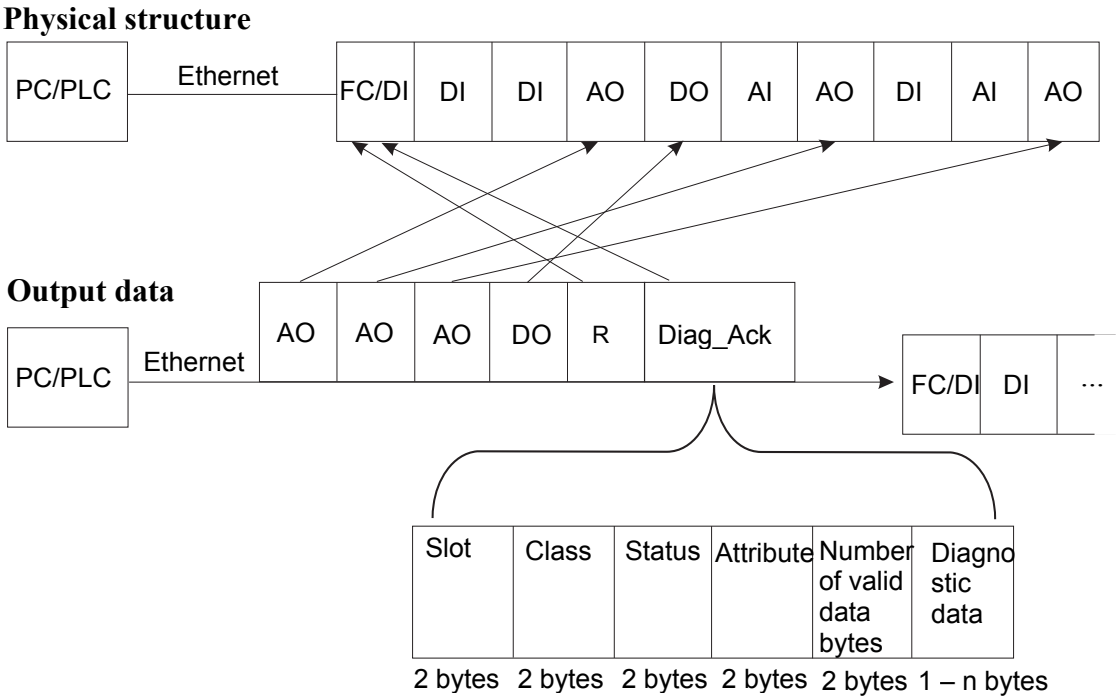
Depending on which assembly instance you select for exchanging the input and output data, the status byte is also to be added, as well as three bytes for the current error and/or diagnosis. The lengths of the diagnostic data is made up of a fixed quantity of 10 bytes and a variable quantity. The variable quantity is dependent on the I/O modules connected to the 767 node. This quantity of bytes can be read out via attribute 53.

Detailed information regarding the instance attributes can be found in the corresponding table of class 64_{hex}.

4.3.3 Example with Assembly Instances 101 and 104

In this example, the assembly instance 101 is used for exchanging data between the higher-level controller and the fieldbus coupler and includes all analog and digital output data. The assembly instance 104, which includes the status byte in addition to the analog and digital input data, is used for exchanging data between the fieldbus coupler and the higher-level controller.

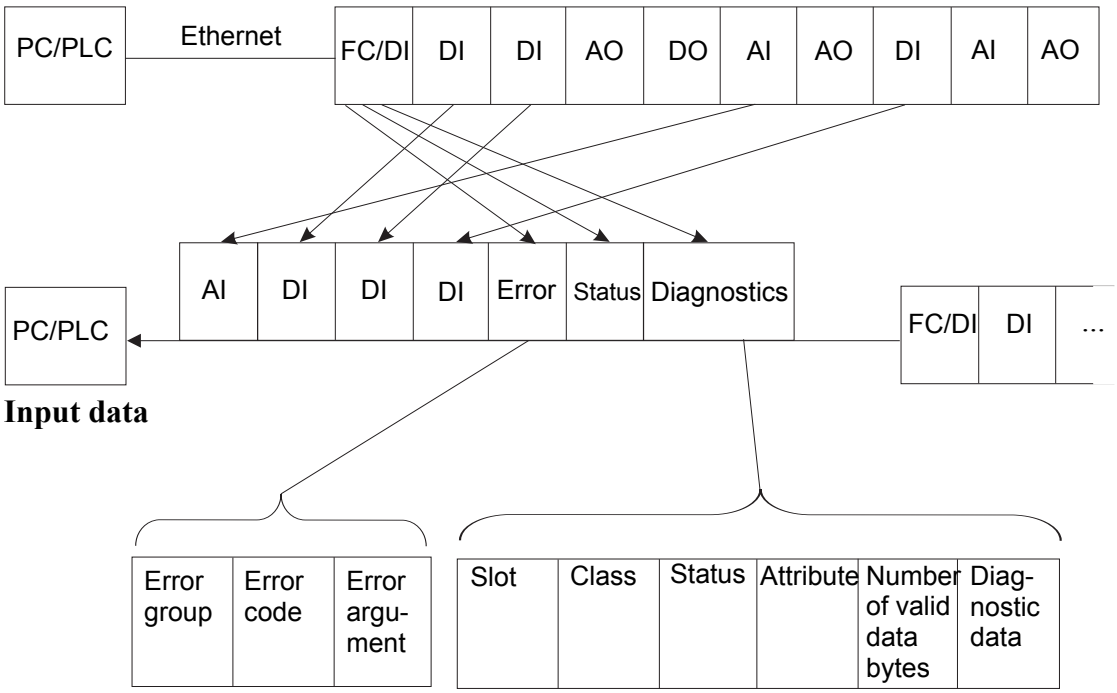
A sample configuration results in the following structure of process data, which is sent from the higher-level controller to the fieldbus coupler:



FC/DI:	Digital inputs on the fieldbus coupler
DI:	Digital input, 8 bits
DO:	Digital output, 8 bits
AI:	Analog input, 8 bytes
AO:	Analog output, 8 bytes
Diag_Ack:	Diagnostic acknowledgment, 10 bytes + x bytes
R:	Reserved, 4 bytes

Process data sent from the fieldbus coupler to the higher-level controller:

Physical structure



DI:	Digital input, 8 bits
DO:	Digital output, 8 bits
AI:	Analog input, 8 bytes
AO:	Analog output, 8 bytes
Error:	3 bytes
Status:	1 byte
Diagnostics:	10 bytes + x bytes
Error group:	1 byte
Error code:	1 byte
Error argument:	1 byte

4.4 Diagnostics

Certain diagnostic information is made available to you in the network. This can include, for example, error messages and warnings from the fieldbus coupler or the connected I/O modules. You have several options for receiving diagnostic information from the fieldbus coupler:

- Diagnostics integrated in the input and output data,
- Diagnostics via explicit messages,
- Diagnostics mixed via input and output data and explicit messages

The diagnostic information of all connected I/O modules and of the fieldbus coupler is stored in the latter for collection. When storing diagnostic information, bit 3 is also set in the status byte. If more diagnostic information is reported to the fieldbus coupler than is collected, an overflow of the internal memory can occur. If this condition arises, bit 2 is set in the status byte, through which a loss of diagnostic information is reported. From this point on, the respective current diagnostic information is stored and the oldest is discarded. More information on status bytes can be found in the next chapter and under attribute 5 of the configuration class of the fieldbus coupler.

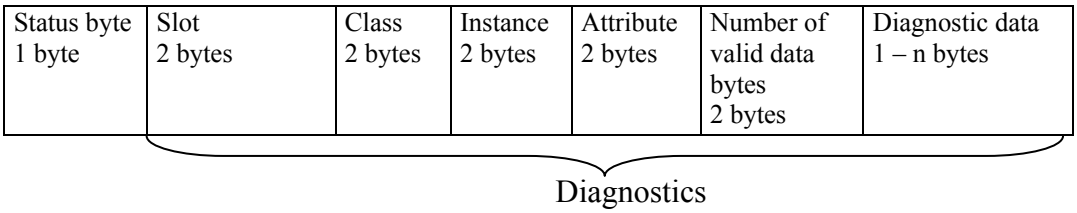
The following sections describe how to access diagnostic information via the Ethernet/IP protocol. As with the process data, the data here that is larger than one byte is displayed in Intel format ("Little Endian").

4.4.1 Diagnostics Integrated in the Input and Output Data

Using this option, diagnostic information is attached behind the process image. This is performed using assembly instances 104 to 106, 112 or 113. Assembly instances 104 to 106 add the status byte into the process image. A diagnostic information is provided if bit 3 is set in this status byte. For these assembly instances, the information content is not provided via input and output data, but can be read out via explicit messages, for example. For more information, see Section 4.4.2.

In addition to the status byte, the assembly instance 112 also contains additional diagnostic information, which describes the diagnostic type and origin. The fieldbus coupler communicates to the higher-level controller its status and the diagnostic information via the assembly instance 112. Instead of the status byte, the assembly instance 113 has a reserved byte whose content is irrelevant.

The diagnostic information is divided up as follows:



Example:
In the entire 767 node, the analog output modules deliver a maximum of 4 bytes of diagnostic data. This number can also be read out with an explicit message on class 100, instance 1, attribute 35. The digital output module on slot 3 reports a recent error, which contains the following data:

Diagnostic Message	**Status byte**	**Slot** 2 bytes	**Class** 2 bytes	**Instance** 2 bytes	**Attribute** 2 bytes	**Number of valid data bytes** 2 bytes	**Diagnostic data** 1 – n bytes
Data	0x08	0x0003	0x0009	0x0005	0x0081	0x0001	0x01 0x00 0x00 0x00

Status byte:
By means of the placed bit 3 in the status byte (0x08), a pending diagnostic message is displayed.

Slot:
The slot 0x0003 indicates that the third I/O module in the 767 node has displayed a diagnostic message.

Class:
0x0009 designates the "Digital Output Point" class as an error class assigned to the digital output modules.

Instance:
The instance 0x0005 identifies channel 5 of the digital output module as the source of the error.

Attribute:
The attribute 0x0081 (129) indicates the "Open Load" error. The description of the diagnostic source on the basis of class, instance and attribute (CIA) can be found in the manuals of the respective I/O modules (appendix).

Number of valid data bytes:
The length of the diagnostic data is identified by the I/O module that can deliver the greatest number of diagnostic bytes. If only a portion of the diagnostic data is used in a current diagnostic message, this portion is determined by the "Number of valid data bytes". Beginning with the next byte, only the number of bytes specified here is valid.

In the example, the value 0x01 indicates that only the next byte in the diagnostic data contains a valid value. The subsequent 3 bytes are reserved and their content is irrelevant.

Diagnostic data:
The first byte of diagnostic data contains a 1 and thus indicates an incoming diagnosis on channel 3 of the digital output module, such as "Open Load". The value 0 indicates an outgoing diagnosis.

Some diagnostic messages require a confirmation. By means of the assembly instance 113, the diagnostic confirmations are transmitted from the higher-level controller to the fieldbus coupler. The data contents are shown in the following table:

Data from fieldbus coupler to higher-level controller			
Process image	Error (3 bytes)	Status byte	Diagnostic message (10 + x bytes)
Data from higher-level controller to fieldbus coupler			
Process image	Reserved (4 bytes)		Diagnostic confirmation (10 + x bytes)

By reserving 4 bytes, the beginning of the diagnostic message and diagnostic confirmation is identical as it relates to the process image (offset of 4 bytes).

Note

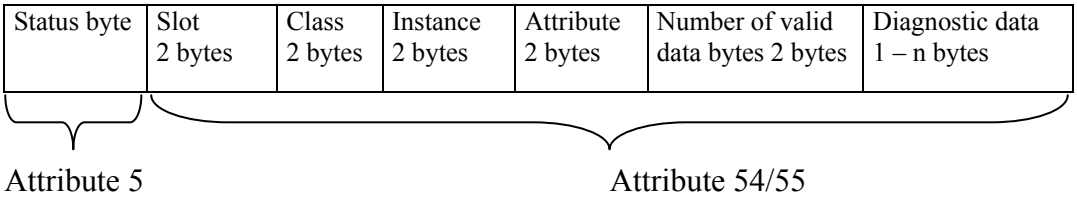
If several diagnostic messages are pending simultaneously, they are transmitted back-to-back. Therefore, the individual diagnostic messages are only available in a cycle; i.e., in an Ethernet telegram. If available, the next diagnostic message would then by transmitted in the next Ethernet telegram.

Information

To confirm the diagnosis, copy the data from the diagnostic message into the diagnostic confirmation.

4.4.2 Diagnostics via Explicit Messages

Using explicit messages, the complete diagnosis can also be read out via attributes. The individual data contents of the diagnostic message or confirmation can be read out from the attributes of class 64_{hex}, instance 1.



Attribute 5: This attribute contains the status byte.

Attribute 54: This contains the diagnostic message, which is composed of slot, class, instance, attribute, number of valid data bytes and diagnostic data.

Attribute 55: This contains the diagnostic confirmation, which is composed of slot, class, instance, attribute, number of valid data bytes and diagnostic data.

The descriptions of individual contents, such as slot, etc., can be found in Section 4.4.1.

The status byte can be read out via attribute 5 by bit 3 indicating whether or not a diagnosis is pending. This bit remains set as long as diagnostic messages are pending. If the diagnosis is only processed via explicit messages, attribute 5 should be read out in certain time intervals in order to detect a diagnostic message. If a diagnostic message is pending, the diagnostic source can be read out via attribute 54 of this class. If a confirmation of the diagnosis is necessary, the user can employ attribute 55 for this purpose.

The data structure of attribute 54 and 55 corresponds to the structure as sent in the input and output data (see figure shown above). Please note that the data is sent in "little endian" format and that you also confirm it in this format. If no other diagnostic is available, it is then indicated by resetting bit 3 of attribute 5. Furthermore, the whole data content of attribute 54 is set to zero.

Note

A diagnostic message can only be read out via attribute 54! If another read action is implemented, either the next pending diagnostic message is delivered or none at all (data content of attribute 54 is zero).

Information

To confirm the diagnosis, copy the data from the diagnostic message into the diagnostic confirmation.

4.4.3 Diagnostics Mixed via Input and Output Data and Explicit Messages

There are several options for receiving and confirming the diagnosis mixed via input and output data and explicit messages. The assembly instances 104, 105 and 106 offer these alternatives:

- **Instance 104**
  For analog and digital input data, including status byte.

- **Instance 105**
  For digital input data, including status byte.

- **Instance 106**
  For analog input data, including status byte.

In these assembly instances, the status byte is integrated and gives information about a pending diagnosis (bit 3 set). The actual diagnosis can then be read out by means of the explicit messages from class 100, instance 1, attribute 54. With attribute 55 of the same class, the diagnosis is confirmed. For more information, see Section 4.4.2.

Other possibilities for combinations of input and output data and explicit messages are also available to link the diagnostic information to the overall concept.

4.5 Object Model

For network communication, EtherNet/IP utilizes an object model in which all functions and data of the fieldbus coupler are described. Each node in the network is depicted as a collection of objects.

The object model contains terms that are defined as follows:

- **Object:**
  An object is an abstract representation of individual, related components within a device. It is determined by its data or attributes, its outwardly applied functions or services, and by its defined behavior.

- **Class:**
  A series of objects which all represent the same type of system components. A class is the generalization of an object. All objects in a class are identical in form and behavior, but can comprise differing attribute values.

- **Instance:**
  A specific and physical occurrence of an object. For example, New Zealand is an instance of the "land" object class. The terms "object," "instance" and "object instance" all refer to a specific instance.

- **Attribute:**
  The description of an externally visible characteristic or of an object's function. Attributes typically deliver status information or regulate the function of an object. For example, the ASCII name of an object and the repetition frequency of a periodic object.

- **Service:**
  A function supported by an object and/or an object class. CIP defines a group of common services that are applied to the attributes.

- **Behavior:**
  Specification of how an object functions. The functions result from various occurrences, which are determined by the object, e.g. receiving service requests, recording internal errors or the sequence of timers.

4.6 CIP Classes

CIP classes are included in the CIP specification of ODVA. They describe the properties (Volume 1) of Ethernet and CAN independent of their physical interface. The physical interface is described in a separate specification. For EtherNet/IP, this is Volume 2, which describes the adaption of EtherNet/IP to CIP.

For this purpose, WAGO uses classes 1, 2, 4 – 6 and also $F4_{hex}$, which are described in Volume 1 ("Common Industrial Protocol"). Classes $F5_{hex}$ and $F6_{hex}$ are supported from Volume 2 (EtherNet/IP Adaption of CIP). The classes in the following table are described in more detail in subsequent sections. The description of WAGO-specific classes begins in Section 4.7.

Table 1: CIP classes

Class	Name	Information
01_{hex}	Identity Object	See Section 4.6.1
02_{hex}	Message Router Object	See Section 4.6.2
04_{hex}	Assembly Object	See Section 4.6.3
05_{hex}	Connection Object	See Section 4.6.4
06_{hex}	Connection Manager Object	See Section 4.6.5
$F4_{hex}$	Port Class Object	See Section 4.6.6
$F5_{hex}$	TCP/IP Interface Object	See Section 4.6.7
$F6_{hex}$	Ethernet Link Object	See Section 4.6.8

Table 2: Explanation of table headers

Column Header	Description
Attribute ID	Integer value which is assigned to the respective attribute
Access	**Set:** The attribute can be accessed using the Set_Attribute service (writing/modifying the attribute value). **Note:** If an attribute supports the Set_Attribute service, it can also be accessed with the Get_Attribute service. **Get:** The attribute can be accessed using the Get_Attribute services (reading the attribute value). **Get_Attribute_All:** Delivers the content of all attributes. **Set_Attribute_Single:** Modifies an attribute value. **Reset:** Performs a restart. 0: Emulates a restart. 1: Emulates a restart and restores the default settings.

Table 2: Explanation of table headers

Column Header	Description
NV	**NV (non-volatile):** The attribute is permanently saved in the fieldbus coupler. **V (volatile):** The attribute is not permanently saved in the fieldbus coupler.
Name	Identification of the attribute.
Data type	Identification of the CIP data type of the attribute.
Description	Brief description of the attribute.
Default Value	Factory setting

4.6.1 Identity Object (01 hex)

The identity class provides general information about the fieldbus coupler that clearly identifies it.

Class Attributes

Table 3: "Identity Object" class: class attributes

Attribute ID	Access	Name	Data Type	Description	Default Value
1	Get	Revision	UINT	Version of object	1 (0x0001)
2	Get	Max Instance	UINT	Maximum instance	1 (0x0001)
3	Get	Max ID Number of Class Attributes	UINT	Maximum number of class attributes (only applies to Get_Attribute_All service)	0 (0x0000)
4	Get	Max ID Number of Instance Attribute	UINT	Maximum number of instance attributes (only applies to Get_Attribute_All service)	0 (0x0000)

Instance Attributes 1

Table 4: "Identity Object" class: instance attributes

Attribute ID	Access	Name	Data Type	Description	Default Value
1	Get	Vendor ID	UINT	Manufacturer identification	40 (0x0028)
2	Get	Device Type	UINT	General type designation of fieldbus coupler	12 (0x000C)
3	Get	Product Code	UINT	Identification of fieldbus coupler	1301 (0x0515)
4	Get	Revision	STRUCT of:	Version of identity object	Firmware dependent
		Major Revision	USINT		
		Minor Revision	USINT		

Table 4: "Identity Object" class: instance attributes

Attribute ID	Access	Name	Data Type	Description	Default Value
5	Get	Status	WORD	See *	Dependent on current status
6	Get	Serial Number	UDINT	Serial Number	-
7	Get	Product Name	SHORT_String	Product name	WAGO FC ETHERNET 8DI 24VDC

Overview of Services

Table 5: "Identity Object" class: services

Service Code	Service Available		Service Name	Description
	Class	Instance		
01 hex	Yes	Yes	Get_Attribute_All	Delivers content of all attributes.
05 hex	No	Yes	Reset	Carries out the reset service. **Service parameter:** **0**: Emulates a restart. **1**: Emulates a restart and restores the default settings.
0E hex	No	Yes	Get_Attribute_Single	Delivers content of respective attribute.

* **Bit 0:** Assignment to a master; **Bit 1=0** (reserved); **Bit 2:** Configured: (= 0: configuration is unchanged; =1: configuration differs from manufacturer's parameters); **Bit 3 = 0** (reserved); **Bit 4-7:** Extended device status: (= 0010: at least one faulty I/O connection, = 0011: no I/O connection established);
Bit 8-11: unused
Bit 12-15 = 0 (reserved)

4.6.2 Message Router Object (02 hex)

The "Message Router Object" provides connection points (in the form of classes or instances), which can use a client for addressing services (reading, writing). These messages can be transmitted both when connected and when unconnected from the client to the fieldbus coupler.

Class Attributes

Table 6: "Message Router Object" class: class attributes

Attribute ID	Access	Name	Data Type	Description	Default Value
1	Get	Revision	UINT	Version of object	1 (0x0001)
2	Get	Number of Attributes	UINT	Number of attributes	0 (0x0000)
3	Get	Number of Services	UINT	Number of services	0 (0x0000)
4	Get	Max ID Number of Class Attributes	UINT	Maximum number of class attributes	0 (0x0000)
5	Get	Max ID Number of Instance Attributes	UINT	Maximum number of instance attributes	0 (0x0000)

Note

The class attributes are only accessible with the Get_Attribute_All service.

Instance Attributes 1

Table 7: "Message Router Object" class: instance attributes

Attribute ID	Access	Name	Data Type	Description	Default Value
1	Get	Object List	STRUCT of:	-	
		Number	UINT	Number of implemented classes	25 (0x0019)
		Classes	UINT	Implemented classes	0x0001, 0x0002, 0x0004, 0x0006, 0x00F4, 0x00F5, 0x00F6, 0x0064 –0x0074, 0x0082, 0x00A0, 0x00A1, 0x00A2, 0x00A6, 0x00A7, 0x00AA, 0x00AB, 0x00A3, 0x00A4, 0x00A5, 0x00A8, 0x00A9, 0x00AC, 0x00AD
2	Get	Number Available	UINT	Maximum number of differing connections	128 (0x0080)

Overview of Services

Table 8: "Message router object" class: services

Service Code	Service Available		Service Name	Description
	Class	Instance		
01 hex	Yes	No	Get_Attribute_All	Delivers content of all attributes
0E hex	No	Yes	Get_Attribute_Single	Delivers content of respective attribute

4.6.3 Assembly Object (04 hex)

By means of the assembly classes, even several diverse objects can be combined. These could be, for example, input and output data, status and control information or diagnostic information. WAGO uses the manufacturer-specific instances in order to provide these objects for you in various arrangements. This gives you an efficient way to exchange process data. The following is a description of the individual instances with their contents and arrangements.

Class Attributes

Table 9 "Assembly Object" class: class attributes

Attribute ID	Access	Name	Data Type	Description	Default Value
1	Get	Revision	UINT	Version of object	2 (0x0002)
2	Get	Max Instance	UINT	Highest instance	113 (0x0071)

Instance Attributes 101 (65 hex)

This assembly instance contains analog and digital output data.

Table 10: "Assembly Object" class: instance attributes 101 (65 hex)

Attribute ID	Access	Name	Data Type	Description	Default Value
3	Get/Set	Data	ARRAY of BYTE	Only analog and digital output data are contained in the process image	-
4	Get	Data Size	ARRAY of BYTE	Number of bytes in process image	-

Instance Attributes 102 (66 hex)

This assembly instance contains digital output data only.

Table 11: "Assembly Object" class: instance attributes 102 (66 hex)

Attribute ID	Access	Name	Data Type	Description	Default Value
3	Get/Set	Data	ARRAY of BYTE	Only digital output data is contained in the process image	-
4	Get	Data Size	ARRAY of BYTE	Number of bytes in process image	-

Instance Attributes 103 (67 hex)

This assembly instance contains analog output data only.

Table 12: "Assembly Object" class: instance attributes 103 (67 hex)

Attribute ID	Access	Name	Data Type	Description	Default Value
3	Get/Set	Data	ARRAY of BYTE	Only analog output data is contained in the process image	-
4	Get	Data Size	ARRAY of BYTE	Number of bytes in process image	-

Instance Attributes 104 (68 hex)

This assembly instance contains analog and digital input data and the status only.

Table 13: "Assembly Object" class: instance attributes 104 (68 hex)

Attribute ID	Access	Name	Data Type	Description	Default Value
3	Get	Data	ARRAY of BYTE	Only analog and digital input data and the status are contained in the process image.	-
4	Get	Data Size	ARRAY of BYTE	Number of bytes in process image	-

Instance Attributes 105 (69 hex)

This assembly instance contains digital input data and the status only.

Table 14: "Assembly Object" class: instance attributes 105 (69 hex)

Attribute ID	Access	Name	Data Type	Description	Default Value
3	Get	Data	ARRAY of BYTE	Only digital input data and the status are contained in the process image	-
4	Get	Data Size	ARRAY of BYTE	Number of bytes in process image	-

Instance Attributes 106 (6A $_{hex}$)

This assembly instance contains analog input data and the status only.

Table 15: "Assembly Object" class: instance attributes 106 (6A $_{hex}$)

Attribute ID	Access	Name	Data Type	Description	Default Value
3	Get	Data	ARRAY of BYTE	Only analog input data and the status are contained in the process image	-
4	Get	Data Size	ARRAY of BYTE	Number of bytes in process image	-

Instance Attributes 107 (6B $_{hex}$)

This assembly instance contains analog and digital input data.

Table 16: "Assembly Object" class: instance attributes 107 (6B $_{hex}$)

Attribute ID	Access	Name	Data Type	Description	Default Value
3	Get	Data	ARRAY of BYTE	Only analog and digital input data are contained in the process image	-
4	Get	Data Size	ARRAY of BYTE	Number of bytes in process image	-

Instance Attributes 108 (6C $_{hex}$)

This assembly instance contains digital input data.

Table 17: "Assembly Object" class: instance attributes 108 (6C $_{hex}$)

Attribute ID	Access	Name	Data Type	Description	Default Value
3	Get	Data	ARRAY of BYTE	Only digital input data is contained in the process image	-
4	Get	Data Size	ARRAY of BYTE	Number of bytes in process image	-

Instance Attributes 109 (6D $_{hex}$)

This assembly instance contains analog input data.

Table 18: "Assembly Object" class: instance attributes 109 (6D $_{hex}$)

Attribute ID	Access	Name	Data Type	Description	Default Value
3	Get	Data	ARRAY of BYTE	Only analog input data is contained in the process image	-
4	Get	Data Size	ARRAY of BYTE	Number of bytes in process image	-

Instance Attributes 112 (70 $_{hex}$)

This assembly instance contains analog and digital input data, current errors, the status and the diagnosis.

Table 19: "Assembly Object" class: instance attributes 112 (70 $_{hex}$)

Attribute ID	Access	Name	Data Type	Description	Default Value
3	Get	Data	ARRAY of BYTE	Only analog and digital input data, current errors, the status and the diagnosis are contained in the process image.	-
4	Get	Data Size	ARRAY of BYTE	Number of bytes in process image	-

Instance Attributes 113 (71 $_{hex}$)

This assembly instance contains analog and digital output data, 4 reserved bytes*, the status and the diagnostic confirmation.

Table 20: "Assembly Object" class: instance attributes 113 (71 $_{hex}$)

Attribute ID	Access	Name	Data Type	Description	Default Value
3	Get/Set	Data	ARRAY of BYTE	Only analog and digital output data, 4 reserved bytes*, the status and the diagnostic confirmation are contained in the process image.	-
4	Get	Data Size	ARRAY of BYTE	Number of bytes in process image	

Overview of Services

Table 21: "Assembly Object" class: services

Service Code	Service Available		Service Name	Description
	Class	Instance		
0E hex	Yes	Yes	Get_Attribute_Single	Delivers content of respective attribute.
10 hex	No	Yes	Set_Attribute_Single	Modifies an attribute value.

The software inspects the writing of attribute 3 of assembly instances 101 through 103 and 113. If the limit value has been exceeded, it is identified and, if necessary, corrected. However, a write request is not rejected. This means that if less data is received than expected, only this data is written. If more data is received than expected, the received data at the upper limit is deleted. In the case of explicit messages, however, a defined CIP is generated even though the data has been written.

* These reserved bytes merely assist in providing the identical data width of the status and diagnostic information as the assembly instance 112: 1 byte status + 3 byte error.

Instance Attribute 198 (C6 $_{hex}$) "Input Only"

This instance assists in setting up the connection when no outputs are to be addressed or when inputs are requested that are already being used in an exclusive owner connection. The data length of this instance is always zero.

This instance can only be used in the "consumed path" (from the viewpoint of the slave).

Instance Attribute 199 (C7 $_{hex}$) "Listen Only"

With this instance, a connection can be set up that attaches to an existing exclusive owner connection. Here the new connection has the same transmission parameters as the exclusive owner connection. If the exclusive owner connection is terminated, this connection is likewise automatically terminated. The data length of this instance is always zero.

This instance can only be used in the "consumed path" (from the viewpoint of the slave).

4.6.4 Connection Object (05 $_{hex}$)

Because the connections are established and terminated via the connection manager, the class and instance attributes of this class are not visible.

4.6.5 Connection Manager Object (06 $_{hex}$)

The "Connection Manager Object" provides the internal resources that are required for the input and output data and explicit messages. In addition, the administration of this resource is an assignment of the "Connection Manager Object".

For each connection (input and output data or explicit), another instance of the connection class is created. The connection parameters are extracted from the "Forward Open" service, which is responsible for establishing a connection.

The following services are supported for the first instance:

- Forward_Open
- Unconnected_Send
- Forward_Close

No class and instance attributes are visible.

4.6.6 Port Class Object (F4 hex)

The "Port Class Object" specifies the existing CIP ports on the fieldbus coupler. There is one instance for each CIP port.

Class Attributes

Table 22: "Port Class Object" class: class attributes

Attribute ID	Access	Name	Data Type	Description	Default Value
1	Get	Revision	UINT	Version of object	1 (0x0001)
2	Get	Max Instance	UINT	Maximum number of instances	1 (0x0001)
3	Get	Num Instances	UINT	Number of current ports	1 (0x0001)
8	Get	Entry Port	UINT	Instance of port object from which the request arrived	1 (0x0001)
9	Get	All Ports	Array of Struct UINT	Array of instance attributes 1 and 2 of all instances	0 (0x0000) 0 (0x0000) 4 (0x0004) 2 (0x0002)

Instance Attributes 1

Table 23: "Port Class Object" class: instance attributes

Attribute ID	Access	Name	Data Type	Description	Default Value
1	Get	Port Type	UINT	-	4 (0x0004)
2	Get	Port Number	UINT	CIP port number	2 (0x0002) (EtherNet/IP)
3	Get	Port Object	UINT	Number of 16-bit words in the subsequent path	2 (0x0002)
			Padded EPATH	Object that manages this port	0x20 0xF5 0x24 0x01 (corresponds to TCP/IP interface object)
4	Get	Port Name	Short String	Port name	„“
7	Get	Node Address	Padded EPATH	Port segment (IP address)	Depends on IP address

Overview of Services

Table 24: "Port Class Object" class: services

Service Code	Service Available		Service Name	Description
	Class	Instance		
01 hex	Yes	Yes	Get_Attribute_All	Delivers content of all attributes
0E hex	Yes	Yes	Get_Attribute_Single	Delivers content of respective attribute

4.6.7 TCP/IP Interface Object (F5 hex)

The "TCP/IP Interface Object" provides for the configuration of the TCP/IP network interface of a fieldbus coupler. Examples of configurable objects include the IP address, the network mask and the gateway address of the fieldbus coupler.

The underlying physical communications interface that is connected with the TCP/IP interface object can be any interface supported by the TCP/IP protocol. Examples of components that can be connected to a TCP/IP interface object include the following: an Ethernet interface 802.3, an ATM interface or a serial interface for protocols such as PPP. The TCP/IP interface object provides an attribute, which is identified by the link-specific object for the connected physical communications interface. The link-specific object should typically provide link-specific counters as well as any link-specific configuration attributes.

Each device must support exactly one instance of the TCP/IP interface object for each TCP/IP-compatible communications interface. A request for access to the first instance of the TCP/IP interface object must always refer to the instance connected with the interface, which is used to submit the request.

Class Attributes

Table 25: "TCP/IP Interface Object" class: class attributes

Attribute ID	Access	Name	Data Type	Description	Default Value
1	Get	Revision	UINT	Version of object	1 (0x0001)
2	Get	Max Instance	UINT	Maximum number of instances	1 (0x0001)
3	Get	Num Instance	UINT	Number of currently instanced connections	1 (0x0001)

Instance Attributes 1

Table 26: "TCP/IP Interface Object" class: instance attributes

Attribute ID	Access	NV	Name	Data Type	Description	Default Value
1	Get	V	Status	DWORD	Interface status	-
2	Get	V	Configuration Capability	DWORD	Interface flags for possible configuration types	23 (0x00000017)
3	Get/Set	NV	Configuration Control	DWORD	Determines how the fieldbus coupler arrives at its TCP/IP configuration after the first restart.	17 (0x00000011)

Table 26: "TCP/IP Interface Object" class: instance attributes

Attribute ID	Access	NV	Name	Data Type	Description	Default Value
4	Get	V	Physical Link Object	STRUCT of	-	
			Path	UINT	Number of 16-bit words in the subsequent path	2 (0x0002)
				Padded EPATH	Logical path that points to the physical link object	0x20 0xF6 0x24 0x03 (corresponds to the Ethernet Link Object)
5	Set	NV	Interface Configuration	STRUCT of	-	
			IP Address	UDINT	IP address	0
			Network Mask	UDINT	Network mask	255 (0xFF) 255 (0xFF) 0 (0x00) 0 (0x00)
			Gateway Address	UDINT	IP address of the standard gateway	0
			Name Server	UDINT	IP address of the primary name server	0
			Name Server 2	UDINT	IP address of the secondary name server	0
			Domain Name	STRING	Standard domain name	„ “
6	Get/Set	NV	Host Name	STRING	Device name	MAC-ID

Overview of Services

Table 27: "TCP/IP Interface Object" class: services

Service Code	Service Available		Service Name	Description
	Class	Instance		
01 hex	Yes	Yes	Get_Attribute_All	Delivers content of all attributes.
0E hex	Yes	Yes	Get_Attribute_Single	Delivers content of respective attribute.
10 hex	No	Yes	Set_Attribute_Single	Modifies an attribute value

4.6.8 Ethernet Link Object (F6 hex)

The "Ethernet Link Object" contains link-specific counter and status information for an Ethernet 802.3 communications interface. Each device must support exactly one instance of the Ethernet Link Object for each Ethernet IEEE 802.3 communications interface on the module. An Ethernet link object instance for an internal interface can also be used for the devices, e.g. an internal port with an integrated switch.

Class Attributes

Table 28: "Ethernet Link Object" class: class attributes

Attribute ID	Access	Name	Data Type	Description	Default Value
1	Get	Revision	UINT	Version of object	3 (0x0003)
2	Get	Max Instance	UDINT	Maximum number of instances	3 (00000003)
3	Get	Num Instances	UDINT	Number of currently instanced connections	3 (00000003)

Instance Attribute 1

Table 29: "Ethernet Link Object" class: instance attributes 1

Attribute ID	Access	Name	Data Type	Description	Default Value
1	Get	Interface Speed	UDINT	Transfer rate	10 (0x0A) oder 100 (0x64)
2	Get	Interface Flags	DWORD	Interface configuration information/status information Bit 0: Link status Bit 1: Half/full duplex Bit 2 – 4: Detection status Bit 5: Manual settings require reset Bit 6: Local hardware error Bit 7 – 31: Reserved	Value is dependent upon Ethernet connection.
3	Get	Physical Address	ARRAY of 6 UINTs	MAC address	MAC ID of the fieldbus coupler
6	Get/Set	Interface Control	STRUCT of:	Configuration of physical interface	
		Control Bits	WORD	Interface control bits Bit 0: Automatic detection Bit 1: Default duplex mode Bit 2 – 15: Reserved	0x0001
		Forced Interface Speed	UINT	Preset interface speed	0 (0x0000)

Table 29: "Ethernet Link Object" class: instance attributes 1

Attribute ID	Access	Name	Data Type	Description	Default Value
7	Get	Interface Type	USINT	Interface type: Value 0: Unknown Value 1: Internal interface; e.g., in the case of an integrated switch. Value 2: Twisted pair (e.g., 100Base-TX). Value 3: Glass fiber (e.g., 100Base-FX). Values 4 – 256: Reserved	2 (0x02)
8	Get	Interface State	USINT	Interface status: Value 0: Unknown Value 1: Interface active and ready to send/receive. Value 2: Interface deactivated. Value 3: Interface is testing. Values 4 – 256: Reserved	-
9	Get/Set	Admin State	USINT	Administration status: Value 0: Reserved Value 1: Activate interface. Value 2: Deactivate interface. If this is the sole CIP interface, a request to deactivate is acknowledged with an error (error code 0x09) Values 3 – 256: Reserved	1 (0x01)
10	Get	Interface Label	SHORT_ STRING	Readable identification	"Port 1"

Instance Attributes 2

Table 30: "Ethernet Link Object" class: instance attributes 2

Attribute ID	Access	Name	Data Type	Description	Default Value
1	Get	Interface Speed	UDINT	Transfer rate	10 (0x0A) or 100 (0x64)
2	Get	Interface Flags	DWORD	Interface configuration information/status information Bit 0: Link status Bit 1: Half/full duplex Bit 2 – 4: Detection status Bit 5: Manual settings require reset Bit 6: Local hardware error Bit 7 – 31: Reserved	Value is dependent upon Ethernet connection.
3	Get	Physical Address	ARRAY of 6 UINTs	MAC address	MAC ID of the fieldbus coupler
6	Get/Set	Interface Control	STRUCT of:	Configuration of physical interface	-
		Control Bits	WORD	Interface control bits Bit 0: Automatic detection Bit 1: Default duplex mode Bit 2 – 15: Reserved	1 (0x0001)
		Forced Interface Speed	UINT	Preset interface speed	0 (0x0000)
7	Get	Interface Type	USINT	Interface type: Value 0: Unknown Value 1: Internal interface; e.g., in the case of an integrated switch. Value 2: Twisted pair (e.g., 100Base-TX). Value 3: Glass fiber (e.g., 100Base-FX). Values 4 – 256: Reserved	2 (0x02)
8	Get	Interface State	USINT	Interface status: Value 0: Unknown Value 1: Interface active and ready to send/receive. Value 2: Interface deactivated. Value 3: Interface is testing. Values 4 – 256: Reserved	-

Table 30: "Ethernet Link Object" class: instance attributes 2

Attribute ID	Access	Name	Data Type	Description	Default Value
9	Get/Set	Admin State	USINT	Administration status: Value 0: Reserved Value 1: Activate interface. Value 2: Deactivate interface. If this is the sole CIP interface, a request to deactivate is acknowledged with an error (error code 0x09) Values 3 – 256: Reserved	1 (0x01)
10	Get	Interface Label	SHORT_ STRING	Readable identification	"Port 2"

Instance Attributes 3

Table 31: "Ethernet Link Object" class: instance attributes 3

Attribute ID	Access	Name	Data Type	Description	Default Value
1	Get	Interface Speed	UDINT	Transfer rate	100 (0x64)
2	Get	Interface Flags	DWORD	Interface configuration information/status information Bit 0: Link status Bit 1: Half/full duplex Bit 2 – 4: Detection status Bit 5: Manual settings require reset Bit 6: Local hardware error Bit 7 – 31: Reserved	Value is dependent upon Ethernet connection.
3	Get	Physical Address	ARRAY of 6 UINTs	MAC address	MAC ID of the fieldbus coupler
6	Get	Interface Control	STRUCT of:	Configuration of physical interface	
		Control Bits	WORD	Interface control bits Bit 0: Automatic detection Bit 1: Default duplex mode Bit 2 – 15: Reserved	0x0002
		Forced Interface Speed	UINT	Preset interface speed	100 (0x0064)
7	Get	Interface Type	USINT	Interface type: Value 0: Unknown Value 1: Internal interface; e.g., in the case of an integrated switch. Value 2: Twisted pair (e.g., 100Base-TX). Value 3: Glass fiber (e.g., 100Base-FX). Values 4 – 256: Reserved	1 (0x01)

Table 31: "Ethernet Link Object" class: instance attributes 3

Attribute ID	Access	Name	Data Type	Description	Default Value
8	Get	Interface State	USINT	Interface status: Value 0: Unknown Value 1: Interface active and ready to send/receive. Value 2: Interface deactivated. Value 3: Interface is testing. Values 4 – 256: Reserved	-
9	Get	Admin State	USINT	Administration status: Value 0: Reserved Value 1: Activate interface. Value 2: Deactivate interface. If this is the sole CIP interface, a request to deactivate is acknowledged with an error (error code 0x09) Values 3 – 256: Reserved	1 (0x01)
10	Get	Interface Label	SHORT_ STRING	Readable identification	"Internal port 3"

Overview of Services

Table 32: "Ethernet Link Object" class: services

Service Code	Service Available		Service Name	Description
	Class	Instance		
01 hex	Yes	Yes	Get_Attribute_All	Delivers content of all attributes.
0E hex	Yes	Yes	Get_Attribute_Single	Delivers content of respective attribute.
10hex	No	Yes	Set_Attribute_Single	Modifies an attribute value.

4.7 WAGO-Specific Classes

This manufacturer-specific addition contains information on the 767 Series components that is not contained in the CIP classes. The WAGO-specific classes are listed in the appendix.

Table 33: WAGO-specific classes

Name	Class	Information
Coupler Configuration Object	64 hex	See 767-130 manual
Node Information Class	82 hex	
Discrete Input Point	65 hex	
Discrete Input Point Extended 1	69 hex	
Discrete Input Point Extended 2	6D hex	
Discrete Input Point Extended 3	71 hex	
Discrete Output Point	66 hex	
Discrete Output Point Extended 1	6A hex	
Discrete Output Point Extended 2	6E hex	
Discrete Output Point Extended 3	72 hex	
Analog Input Point	67 hex	
Analog Input Point Extended 1	6B hex	
Analog Input Point Extended 2	6F hex	
Analog Input Point Extended 3	73 hex	
Analog Output Point	68 hex	
Analog Output Point Extended 1	6C hex	
Analog Output Point Extended 2	70 hex	
Analog Output Point Extended 3	74 hex	

5 MODBUS

The 767 Series modular concept makes it possible to attach up to 64 external I/O modules to the fieldbus coupler. However, the variable node design as well as the large number of differing I/O modules make it impossible to statically assign input and output data to fixed MODBUS addresses. The only exceptions are the "digital" MODBUS services. For these, the MODBUS address is identical to the channel number, i.e., the 47th digital input can always be found at the MODBUS address "46."

By adding or removing I/O modules, the structure of the process images is changed which also changes the MODBUS addresses of the individual I/O module channels.

The MODBUS communication is performed via service calls. To do this, the MODBUS master (client) sends a request telegram to port 502 of the MODBUS slave (server). The MODBUS slave returns the result of the service call in a response telegram to the MODBUS master.

The most important elements of a MODBUS telegram are:

Term	Description
FunctionCode (FC)	Service identifier: Read or write operation to bits or words
Address	Operation start address
Count	Number of bits or words depending on the service
[Data]	Process data

The service identifier or "FunctionCode" (FC) first determine whether a read or write operation is involved. It also determines the basic data type to which the operation is to be applied. Therefore, the meaning of the parameters "Address" and "Count" is also dependent on the function code. Thus "address :=3" can stand for a bit or a word in the input or output process image.

The MODBUS protocol is largely based on the following basic data types:

Data Type	Length	Description
Discrete Inputs	1 bit	Digital inputs
Coils	1 bit	Digital outputs
Input Register	16 bits	Analog input data
Holding Register	16 bits	Analog output data

One or more "FunctionCodes" are defined for each basic data type.

Although digital and analog process data from the fieldbus coupler and I/O modules is combined in a process image, the first digital output or input is always reached with the "digital" MODBUS services at address 0. You get to the first analog output or input via the word services.

Note

The fieldbus coupler supports up to 15 socket connections simultaneously via MODBUS/TCP. For MODBUS/UDP, there are up to 5 telegrams that can be stored temporarily.

5.1 Process Image

After starting up the fieldbus coupler, it automatically identifies all connected I/O modules. The fieldbus coupler uses this to create a local process image, divided into an input and output data range. The data from the analog I/O modules and subsequently the data from the digital I/O modules are initially stored in these areas.
The size of the process image can be identified from the connected 767 components.

5.2 Process Data Exchange

Data exchange between the MODBUS/TCP master and the I/O modules is achieved using the MODBUS functions implemented in the fieldbus coupler via bit-by-bit or word-by-word reading and writing routines. There are four different types of process data in the fieldbus coupler:

- Input words
- Output words
- Input bits
- Output bits

5.3 Accessing the Process Image via MODBUS Functions

The following table describes the access types, which can be used to access logical address ranges.

Table 34: MODBUS function codes (FC)

FC	Name	Description
FC1	Read coils	Several digital output values are read out
FC2	Read inputs discrete	Read several digital input and output values
FC3	Read holding registers	Read several analog input and output values
FC4	Read input registers	Read several analog input and output values
FC5	Write coil	A single digital output value is written
FC6	Write single register	A single analog output value is written
FC15	Force multiple coils	Several digital output values are written
FC16	Write multiple registers	Several analog output values are written
FC22	Write register screen	Write individual bits to a register using an AND or OR screen
FC23	Read/write multiple registers	Write and read operation on analog input and output values

Note

Note the number system when addressing!
The examples listed use the hexadecimal system (e.g., 0x000) as the number format. Addressing starts with 0.
Depending on the software controller, the format and start of addressing can vary. In this case, all addresses are to be converted accordingly.

5.3.1 Register Services

With the register services, you can identify or modify the status of analog input and output modules using the function codes FC3, FC4 and FC6, FC16, FC22 and FC23.

Table 35: Read analog input terminals using FC3, FC4, FC23

MODBUS address		Description
HEX	DEC	
0x0000 – 0x00FF	0 – 255	Read analog or digital input values (part 1). Physical address space of the input data of 256 words.
0x0100 – 0x01FF	256 – 511	Invalid range ("Illegal data address")
0x0200 – 0x02FF	512 – 767	Read back of analog or digital output values (part 1). Physical size of the address space: 256 words.
0x0300 – 0x0FFF	768 – 4095	Invalid range ("Illegal data address")
0x1000 – 0x2FFF	4096 – 12287	Configuration register. See Section 5.3.3.
0x3000 – 0x5FFF	12288 – 24575	Invalid range ("Illegal data address")
0x6000 – 0x62FB	24576 – 25339	Read the input data (part 2). Additional address space of 764 words.
0x62FC – 0x6FFF	25340 – 28671	Invalid range ("Illegal data address")
0x7000 – 0x72FB	28672 – 29435	Read output data (part 2). Additional address space of 764 words.

Table 36: Write analog output terminals using FC6, FC16, FC22, FC23

MODBUS address		Description
HEX	DEC	
0x0000 – 0x00FF	0 – 255	Write analog or digital output values (part 1). Physical address space of the output data of 256 words.
0x0100 – 0x01FF	256 – 511	Invalid range ("Illegal data address")
0x0200 – 0x02FF	512 – 767	Write analog or digital output values (part 1). Physical size of the address space: 256 words. **Note:** Here you write the same output values of the address range 0x0000 – 0x00FF.
0x0300 – 0x0FFF	768 – 4095	Invalid range ("Illegal data address")
0x1000 – 0x2FFF	4096 – 12287	Configuration register. See Section 5.3.3.
0x3000 – 0x5FFF	12288 – 24575	Invalid range ("Illegal data address")
0x6000 – 0x62FB	24576 – 25339	Write the output data (part 2). Additional address space of 764 words.
0x62FC – 0x6FFF	25340 – 28671	Invalid range ("Illegal data address")
0x7000 – 0x72FB	(28672 – 29435)	Write the output data (part 2). Additional address space of 764 words. **Note:** Here you write the same output values of the address range 0x6000 – 0x62FC.
0x72FC – 0x7FFF	29436 – 32767	Invalid range ("Illegal data address")

5.3.2 Bit Services

With the digital bit services, you can identify or modify the status of digital input and output modules using the function codes FC1, FC2 and FC5, FC15.

Table 37: Read digital input terminals using FC1, FC2

MODBUS address		Description
HEX	DEC	
0x0000 – 0x01FF	0 – 511	Read the first 512 input values
0x0200 – 0x03FF	512 – 1023	Read the first 512 output values
0x0400 – 0x7FFF	1024 – 32767	Invalid range ("Illegal data address")
0x8000 – 0x85F7	32768 – 34295	Read the input values in the range of 513 to 2039
0x85F8 – 0x8FFF	34296 – 36863	Invalid range ("Illegal data address")
0x9000 – 0x95F7	36864 – 38391	Read the output values in the range of 513 to 2039
0xA000 – 0xFFFF	40960 – 65535	Invalid range ("Illegal data address")

Table 38: Write digital output terminal using FC5, FC15

MODBUS address		Description
HEX	DEC	
0x0000 – 0x01FF	0 – 511	Write the first 512 output values.
0x0200 – 0x03FF	512 – 1023	Write the first 512 output values.
0x0400 – 0x7FFF	1024 – 32767	Invalid range ("Illegal data address")
0x8000 – 0x85F7	32768 – 34295	Write the output values in the range of 513 to 2039.
0x85F8 – 0x8FFF	34296 – 36863	Invalid range ("Illegal data address")
0x9000 – 0x95F7	36864 – 38391	Write the output values in the range of 513 to 2039.
0x95F8 – 0xFFFF	38392 – 65535	Invalid range ("Illegal data address")

5.3.3 Configuration Register

By using the configuration register, you can configure the fieldbus coupler and read information using it. Access register using MODBUS functions FC4 "Read Input Registers" and FC6 "Write Single Register".

Table 39: Configuration register

Address	Access	Length (word)	Description
0x1000	R/W	1	Read/write watchdog time
0x1001	R/W	1	Watchdog coding screen 1-16
0x1002	R/W	1	Watchdog coding screen 17-32
0x1003	R/W	1	Watchgod trigger
0x1004	R	1	Minimum trigger time
0x1005	R/W	1	Stop watchdog (write sequence 0xAAAA, 0x5555)
0x1006	R	1	Watchdog status
0x1007	R/W	1	Watchdog restart (write sequence 0x1)
0x1008	RW	1	Stop watchdog (write sequence 0x55AA or 0xAA55)
0x1009	R/W	1	MODBUS and http connection stop at a watchdog timeout
0x100A	R/W	1	Watchdog configuration
0x100B	W	1	Save watchdog parameters
0x1020	R	1 – 2	Display error group via LED
0x1021	R	1	Display error argument and error code via LED
0x1022	R	1 – 4	Number of analog output data in the process image (in bits)
0x1023	R	1 – 3	Number of analog input data in the process image (in bits)
0x1024	R	1 – 2	Number of digital input data in the process image (in bits)
0x1025	R	1	Number of digital input data in the process image (in bits)
0x1028	R/W	1	Boot configuration
0x1029	R	9	MODBUS/TCP statistics
0x102A	R	1	Number of TCP connections
0x1030	R/W	1	Configuration MODBUS/TCP timeout
0x1031	R	1	Read the MAC ID of the fieldbus coupler
0x2000	R	1	Constant 0x0000
0x2001	R	1	Constant 0xFFFF
0x2002	R	1	Constant 0x1234
0x2003	R	1	Constant 0xAAAA
0x2004	R	1	Constant 0x5555
0x2005	R	1	Constant 0x7FFF
0x2006	R	1	Constant 0x8000
0x2007	R	1	Constant 0x3FFF
0x2008	R	1	Constant 0x4000
0x2010	R	1	Firmware index
0x2011	R	1	Series designation (767)
0x2012	R	1	Fieldbus type (1301)
0x2013	R	1	Firmware versions (major revision)
0x2014	R	1	Firmare versions (minor revision)
0x2020	R	16	Fieldbus coupler short description
0x2021	R	8	Firmware compile time
0x2022	R	8	Firmware compile date
0x2023	R	32	Firmware loader version

Table 39: Configuration register

Address	Access	Length (word)	Description
0x2030	R	65	Description of the connected I/O modules (0 – 64)
0x2040	W	1	Software reset (write sequence 0x55AA or 0xAA55)
0x2041	W	1	Formate the user drive
0x2042	W	1	Extract the HTML pages from the firmware
0x2043	W	1	Default settings
0x2099	R/W	1	MODBUS version change (compatibility mode for Release 1)

5.3.4 Watchdog Behavior

A timing function (timeout) is periodically initiated in the fieldbus coupler by the higher-level controller for monitoring the TCP connection. Because this time is always restarted before communication takes place, it cannot reach its upper value in the case of error-free communication. If this time has run out, a fieldbus failure exists. In this case, the fieldbus coupler answers all subsequent MODBUS "TCP/IP requests" with the exception code 0x0004 (slave device failure). Separate registers exist in the fieldbus coupler for controlling the watchdog and requesting its status through the higher-level controller.

The watchdog is not yet activated when the power supply is switched on. The timeout value must first be determined (register 0x1000). The watchdog can be activated by writing a value other than 0 in the register (0x1001).

5.3.4.1 Watchdog Register

Table 40: Register address 0x1000

Register address 0x1000 (4096_{dec})	
Value	Watchdog time
Access	Read/write
Default	0x0000
Description	The register saves the value for the timeout. The default value must be changed to a value not equal to zero, so that the watchdog can be started. The time is set in multiples of 100 ms, e.g., 0x0009 means a timeout time of 0.9 seconds. This value cannot be changed when the watchdog is running. There is no code by which the current data value can be written again while the watchdog is active.

Table 41: Register address 0x1001

Register address 0x1001 (4097_{dec})	
Value	Watchdog function coding screen, function code 1...16
Access	Read/write
Default	0x0000
Description	Use this screen to set the function codes to trigger the watchdog function. The function code can be selected with "1". $(= 2^{(\text{function code-1})} + 2^{(\text{function code-1})} + ...)$ Example: Function code 5 → $2^{(5-1)} = 2^4$ → bit 4 is set to 1 A value not equal to zero starts the watchdog function. If only codes from unsupported functions are entered in the screen, the watchdog does not start. An existing error is reset and the process image can be described again. Again here, no changes can be made when the watchdog is running. When the watchdog is active, there is no code by which the current data value can be written again.

Table 42: Register address 0x1002

Register address 0x1002 (4098_{dec})	
Value	Watchdog function coding screen, function code 17...32
Access	Read/write
Default	0x0000
Description	The same function as before, but with the function codes 17 to 32. These codes are not supported. Therefore, this register should be left at the default value. There is no exception code by which the current data value can be written again while the watchdog is active.

Table 43: Register address 0x1003

Register address 0x1003 (4099_{dec})	
Value	Watchgod trigger
Access	Read/write
Default	0x0000
Description	This register is used for an alternative trigger method. The watchdog is triggered by writing different values to this register. Successive values must differ in size. The watchdog starts when values not equal to zero are written. A watchdog error is reset and it is again possible to write process data.

Table 44: Register address 0x1004

Register address 0x1004 (4100_{dec})	
Value	Minimum current trigger time
Access	Read/write
Default	0xFFFF
Description	This register saves the current smallest watchdog trigger time. When the watchdog is triggered, the saved value is compared to the current value. If the current value is smaller than the saved value, it is replaced by the current value. The unit is 100 ms/digit. The saved value is modified by writing new values. This has no effect on the watchdog. The value 0x0000 is not permitted.

Table 45: Register address 0x1005

Register address 0x1005 (4101_{dez})	
Value	Stop watchdog
Access	Read/write
Default	0x0000
Description	If the value 0xAAAA followed by the value 0x5555 is written to this register, the watchdog stops. The watchdog error response is blocked. A watchdog error is reset and it is again possible to write to the process data.

Table 46: Register address 0x1006

Register address 0x1006 (4102_{dec})	
Value	While watchdog runs
Access	Read
Default	0x0000
Description	Current watchdog status at 0x0000: Watchdog inactive at 0x0001: Watchdog active at 0x0002: Watchdog timed out

Table 47: Register address 0x1007

Register address 0x1007 (4103_{dez})	
Value	Restart watchdog
Access	Read/write
Default	0x0001
Description	Writing 0x1 to the register starts the watchdog again. If the watchdog was stopped before the overflow, it is not started again.

Table 48: Register address 0x1008

Register address 0x1008 (4104_{dec})	
Value	Just pause watchdog
Access	Read/write
Default	0x0000
Description	By writing the values 0x0AA55 or 0X55AA, the watchdog is paused if active. The watchdog error response is temporarily disabled. An existing watchdog error is reset and it is again possible to write to the watchdog register.

Table 49: Register address 0x1009

Register address 0x1009 (4105_{dec})	
Value	Close MODBUS socket after watchdog timeout
Access	Read/write
Description	0: MODBUS socket is not closed 1: MODBUS socket is closed

Table 50: Register address 0x100A

Register address 0x100A (4106_{dec})	
Value	Alternative watchdog
Access	Read/write
Default	0x0000
Description	Write a time value to register 0x1000 Register 0x100A = 0x0001: Watchdog is switched to active The watchdog is started with the first MODBUS telegram. Each MODBUS/TCP command triggers the watchdog. All outputs are set to zero after the watchdog time has elapsed. The outputs can be set again by writing again. The register 0x100A is permanent and also register 0x1000. When the watchdog is on, the time value in register 0x1000 can no longer be modified.

Table 51: Register address 0x100B

Register address 0x100B (4107_{dec})	
Value	Save Watchdog Parameters
Access	Write
Default	0x0000
Description	By writing "1" to register Register 0x100B, registers 0x1000, 0x1001, 0x1002 are set to permanent.

5.3.4.2 Diagnostics Registers

The following registers can be read to determine an error in the fieldbus node.

Table 52: Register address 0x1020

Register address 0x1020 (4128_{dec})	
Value	LED Error Group
Access	Read
Description	Indication of the group number

Table 53: Register address 0x1021

Register address 0x1021 (4129_{dec})	
Value	LED Error Code/Agrument
Access	Read
Description	Indication of the error code and error argument

5.3.4.3 Configuration Registers

The following registers can be read to determine the configuration of the connected I/O modules:

Table 54: Register address 0x1022

Register address 0x1022 (4130_{dez})	
Value	CnfLen.AnalogOut
Access	Read
Description	Number of I/O bits for the process data words of the outputs

Table 55: Register address 0x1023

Register address 0x1023 (4131_{dec})	
Value	CnfLen.AnalogInp
Access	Read
Description	Number of I/O bits for the process data words of the inputs

Table 56: Register address 0x1024

Register address 0x1024 (4132_{dec})	
Value	CnfLen.DigitalOut
Access	Read
Description	Number of I/O bits for the process data bits of the outputs

Table 57: Register address 0x1025

Register address 0x1025 (4133_{dec})	
Value	CnfLen.DigitalInp
Access	Read
Description	Number of I/O bits for the process data bits of the inputs

Table 58: Register address 0x1028

Register address 0x1028 (4136_{dec})	
Value	Boot options
Access	Read/write
Description	Boot configuration: 1: BootP 2: DHCP 4: EEPROM

Table 59: Register address 0x1029

Register address 0x1029 (4137_{dec}) with upt to 9 words	
Value	MODBUS/TCP statistics
Access	Read/write
Description	1 word SlaveDeviceFailure → Terminal bus error, fieldbus error with watchdog switched on 1 word BadProtocol → Error in MODBUS/TCP header 1 word BadLength → Incorrect telegram length 1 word BadFunction → Invalid function code 1 word Bad Address → Invalid register address 1 word BadData → Invalid value 1 word TooManyRegisters → Number of registers to be processed too high, read/write 125/100 1 word TooManyBits → Number of coils to be processed too high, read/write 2000/800 1 word ModTcpMessageCounter → Number of MODBUS/TCP telegrams received The register is reset by writing 0xAA55 or 0x55AA.

Table 60: Register address 0x102A

Register address 0x102A (4138_{dec}) with up to 1 word	
Value	MODBUS/TCP connections
Access	Read
Description	Number of TCP connections

Table 61: Register address 0x1030

Register address 0x1030 (4144_{dec}) with up to 1 word	
Value	Configuration MODBUS/TCP timeout
Access	Read/write
Default	0x0000
Description	This register saves the value for TCP connection monitoring. The time base is 100 ms, the minimum value is 10 ms. Open TCP connections are automatically closed if the time entered was exceeded per connection. A query on the connection triggers the watchdog.

Table 62: Register address 0x1031

Register address 0x1031 (4145_{dec}) with up to 3 words	
Value	Read the MAC ID of the fieldbus coupler
Access	Read
Description	Output of the MAC ID, length 3 words

Table 63: Register address 0x2030

Register address 0x2030 (8240_{dec}) with up to 65 words	
Value	Description of the connected I/O module
Access	Read I/O modules 0...64
Description	Output: e.g., "3801" for the input module 767-3801

Table 64: Register address 0x2040

Register address 0x2040 (8256_{dec})	
Value	Perform software reset
Access	Write (write sequence 0xAA55 or 0x55AA)
Description	The fieldbus coupler performs a restart by writing the values 0xAA55 or 0x55AA.

Table 65: Register address 0x2041

Register address 0x2041 (8257_{dec})	
Value	Flash format
Access	Write (write sequence 0xAA55 or 0x55AA)
Description	The Flash file system is reformatted.

Table 66: Register address 0x2042

Register address 0x2042 (8258_{dez})	
Value	Extract files
Access	Write (write sequence 0xAA55 or 0x55AA)
Description	The default files (HTML pages) of the fieldbus coupler are extracted and written to the Flash file system.

Table 67: Register address 0x2043

Register address 0x2043 (8259_{dec})	
Value	0x55AA
Access	Write
Description	Default settings

5.3.4.4 Firmware Information Register

The following registers are used to read information about the fieldbus coupler firmware.

Table 68: Register address 0x2010

Register 0x2010 (8208_{dec}) with up to 1 word	
Value	Revision
Access	Read
Description	Firmware index, e.g., 0005 for version 5

Table 69: Register address 0x2011

Register address 0x2011 (8209_{dec}) with up to 1 word	
Value	Series code
Access	Read
Description	WAGO series number, e.g., 0767 for WAGO SPEEDWAY 767

Table 70: Register address 0x2012

Register address 0x2012 (8210_{dec}) with up to 1 word	
Value	Item number
Access	Read
Description	WAGO item number, e.g., 1301 for the fieldbus coupler 767-1301, etc.

Table 71: Register address 0x2013

Register address 0x2013 (8211_{dec}) with up to 1 word	
Value	Major sub item code
Access	Read
Description	Firmware version major revision

Table 72: Register address 0x2014

Register address 0x2014 (8212_{dec}) with up to 1 word	
Value	Minor sub item code
Access	Read
Description	Firmware version minor revision

Table 73: Register address 0x2020

Register address 0x2020 (8224_{dec}) with up to 16 words	
Value	Description
Access	Read
Description	Information about the fieldbus coupler, 16 words

Table 74: Register address 0x2021

Register address 0x2021 (8225_{dec}) with up to 8 words	
Value	Description
Access	Read
Description	Time of the firmware version, 8 words

Table 75: Register address 0x2022

Register address 0x2022 (8226_{dec}) with up to 8 words	
Value	Description
Access	Read
Description	Date of the firmware version, 8 words

Table 76: Register address 0x2023

Register address 0x2023 (8227_{dec}) with up to 32 words	
Value	Description
Access	Read
Description	Information about programming the firmware, 32 words

5.3.4.5 Constant Register

The following registers contain constants that can be used to test communication with the master:

Table 77: Register address 0x2000

Register address 0x2000 (8192_{dec})	
Value	Zero
Access	Read
Description	Constant with zero

Table 78: Register address 0x2001

Register address 0x2001 (8193_{dec})	
Value	Ones
Access	Read
Description	Constant with ones. • "-1" if constant is declared as "signed int" • "MAXVALUE" if constant is declared as "unsigned int"

Table 79: Register address 0x2002

Register address 0x2002 (8194_{dec})	
Value	1,2,3,4
Access	Read
Description	Constant value for testing whether high and low bytes are exchanged (Intel/Motorola format). Should appear in the master as 0x1234. If 0x3412 appears, high and low bytes must be exchanged.

Table 80: Register address 0x2003

Register address 0x2003 (8195_{dec})	
Value	Screen 1
Access	Read
Description	Constant that displays whether all bits are present. Used together with register 0x2004.

Table 81: Register address 0x2004

Register address 0x2004 (8196_{dec})	
Value	Screen 1
Access	Read
Description	Constant that displays whether all bits are present. Used together with register 0x2003.

Table 82: Register address 0x2005

Register address 0x2005 (8197_{dec})	
Value	Largest positive number
Access	Read
Description	Constant to check the arithmetic

Table 83: Register address 0x2006

Register address 0x2006 (8198_{dec})	
Value	Largest negative number
Access	Read
Description	Constant to check the arithmatic

Table 84: Register address 0x2007

Register address 0x2007 (8199_{dec})	
Value	Largest half positive number
Access	Read
Description	Constant to check the arithmatic

Table 85: Register address 0x2008

Register address 0x2008 (8200_{dec})	
Value	Largest half negative number
Access	Read
Description	Constant to check the arithmatic

6 Ethernet Interface

6.1 IP Address Allocation

The fieldbus coupler can obtain its IP address dynamically (from a server) or it can be configured with a static (fixed) IP address.

The IP address can be assigned dynamically via the BootP or DHCP protocol. IP addresses assigned dynamically are not saved. A BootP or DHCP server is therefore required each time the fieldbus coupler is switched on.

You can configure an IP address via:

- using the DIP switch on the fieldbus coupler,
- via web-based management or
- the "WAGOframe" configuration software.

In the default status, the assignment of dynamic IP addresses is activated via the BootP protocol.

To access the web-based management in a browser, the fieldbus coupler requires a known IP address. This initial IP address can be assigned via BootP, DIP switch or using the WAGOframe configuration software.

With the default status as the basis, allocation of an IP address via DIP switches and BootP is presented below. For a complete list of IP address allocations, refer to the manual for the 767-1301.

6.1.1 Setting a Permanent IP Address using DIP Switches

In the default status, the top three bytes (network ID) are assigned "192.168.1." and the subnet mask 255.255.255.0. Use switches 1 – 8 to set the last byte (host ID) of the IP address. To do this, switch 9 (SN) must be set to "On".

Switch	1	2	3	4	5	6	7	8	9	10	11	12
Binary value/ Functions	2^0 (1)	2^1 (2)	2^2 (4)	2^3 (8)	2^4 (16)	2^5 (32)	2^6 (64)	2^7 (128)	SN	-	Boot/ Execute	Reset
Switch setting	Off	Off	Off	Off	Off	Off	Off	Off	Off	Off	Off	Off

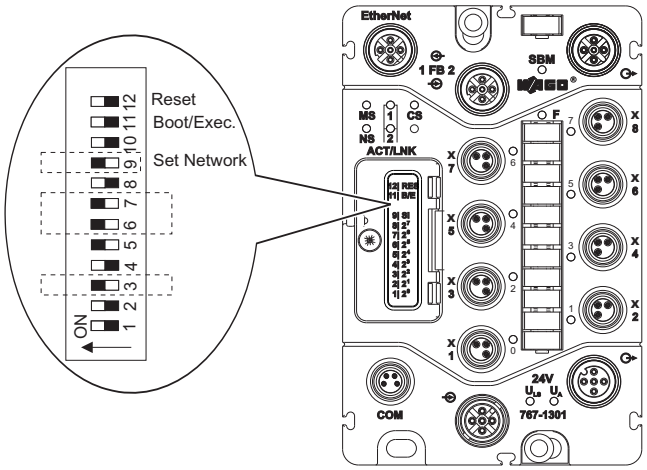
If you were to set the IP address 192.168.1.100, you would proceed as follows:

1. Open the cover by unscrewing the M3 screw with a screwdriver.

2. Set switch 9 (SN) to "On" to activate an IP address by means of the DIP switch.

3. For Host ID 100 set switches 3, 6, 7 to "On".
($2^2+2^5+2^6 = 4+32+64 = 100$).

4. Close the cover and screw it securely in place to maintain degree of protection IP 67.



With the 8 available switches you can set addresses from 1 – 254. Addresses 0 and 255 are reserved and may not be used.

6.1.2 Allocation of a Dynamic IP Address using BootP

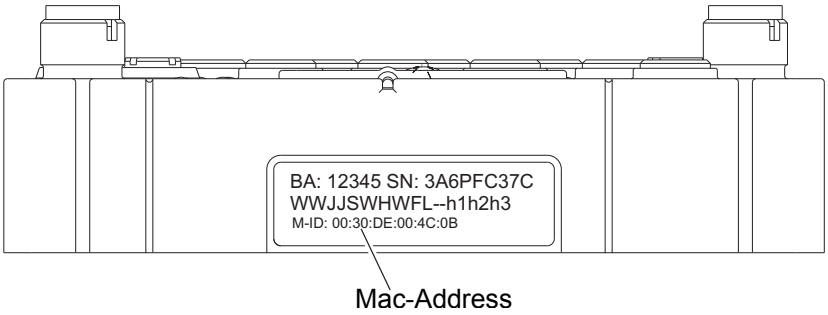
In the default status, IP address allocation is activated via the BootP protocol. Assigning the IP address using the BootP protocol is illustrated here with the example of the WAGO BootP server.

BootP-Server konfigurieren

Configuring the WAGO BootP server

In order to communicate with the fieldbus coupler in the local network, the MAC address of the fieldbus coupler must first be specified and an IP address must then be assigned.

1. Take note of the MAC address (Mac-Address) of the fieldbus coupler. This can be found on the label on the side of the fieldbus coupler.



Example of a MAC address on the adhesive label on the left side of the fieldbus coupler

2. Click the **[Start]** button and go to **Programs** > **WAGO Software > WAGO BootP Server** to launch the WAGO BootP server.

3. Open the configuration file in which you assign the MAC address to an IP address from the local network. Click the **[Edit BootPtab]** button in the BootP server. The configuration file opens.

4. Click in the line that starts with "node", e.g.,
   `"my1301:ht=1:ha=0030DE00001C:ip=192.168.1.100:sm=255.255.255.0:".`

5. Replace the MAC address consisting of twelve characters after ":ha=" with the address printed on the label.

6. Enter an IP address after "ip=". In the example shown here, this is 192.168.1.100. Separate the individual numbers with a decimal point.

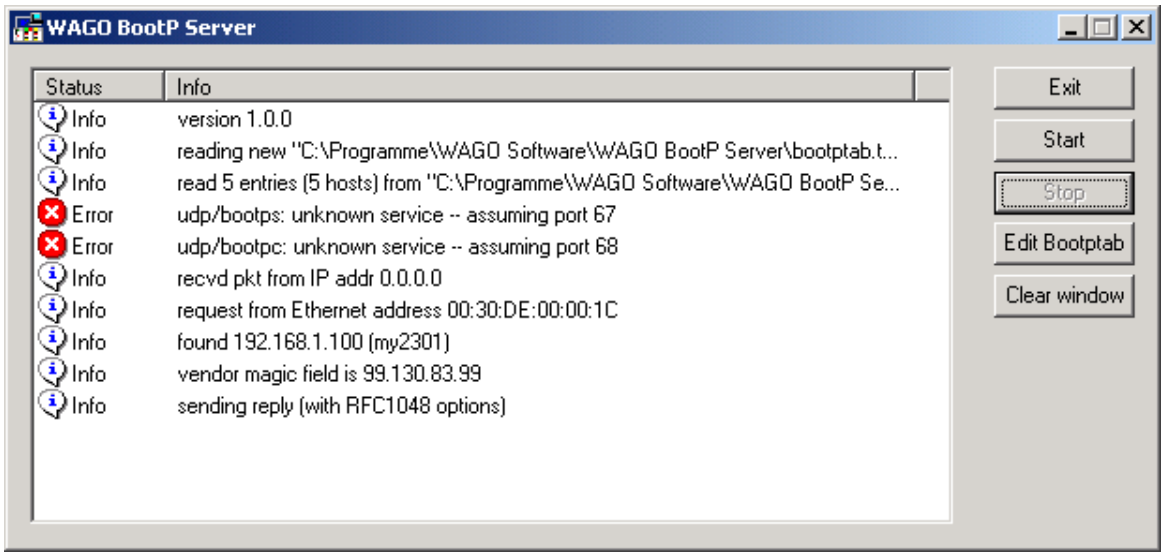
7. To address more than one fieldbus node, add a line for each node with the corresponding assignment for each additional fieldbus coupler to the bootptab.txt file. Repeat steps 5 to 7 for this.

8. Save the new settings in bootptab.txt file. To do this, click on "File" in the menu and select "Save".

9. Close the editor.

Assigning an IP address using the WAGO BootP server

1. To start the BootP server click the button **[Start]** in the BootP dialog window that is opened. Various messages will be displayed in the window. The error messages indicate that some services (e.g. port 67, port 68) in the operating system have not been defined. You can ignore these error messages.



Dialogfenster des WAGO-BootP-Servers mit Nachrichten

2. Restart the fieldbus coupler by disconnecting it from the power supply and then reconnecting it.

   A request appears from the fieldbus coupler. The BootP server replies that the IP address has been accepted (no errors). The IP address is now temporarily available in the fieldbus coupler, but is not yet permanently saved. When restarted, the fieldbus coupler attempts to obtain a new IP address from the BootP server.

3. Click **[Stop]** and then **[Exit]** to close the BootP Server.

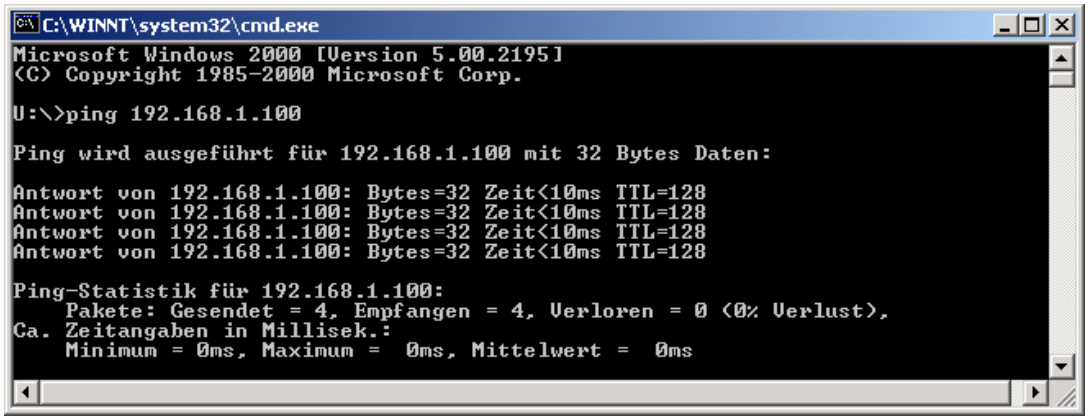
4. To permanently save the IP address in the fieldbus coupler, select the option "Static IP" in the "TCP/IP view of the web-based management. Or use WAGOframe by selecting "Device memory" under the "IP address obtained from" parameter.

6.1.3 Testing the Network Connection

Carry out a ping network function to check whether the fieldbus coupler can be reached at the IP address you have assigned in the network. To do this, open the prompt in MS Windows by clicking "Start" and selecting **Programs** > **Run**. Enter `cmd` in the dialog box and click **[OK]**.

1. In the DOS window, enter the command ping and the IP address of the fieldbus coupler; e.g.: `ping 192.168.1.100`.

2. Press the **[Enter]** key. Your PC sends out an inquiry that is answered by the fieldbus coupler. This reply appears in the DOS window. If the error message "Timeout" appears, the fieldbus coupler has not responded properly. You then need to check your network settings.



6.2 User Administration

Access to the web-based management and via FTP is password protected. In the default status, the following users and access authorizations are defined:

Table 86: User administration

User name	Password	WBM	FTP	WAGOframe
admin	wago	Full access	Full access	Full access
user	user	Limited access	Full access	Limited access
guest	guest	Limited access	Full access	Read access

Use WBM and WAGOframe to change the passwords.

6.3 File System

Two partitions of the file system are available to the user including a RAM disk and a partition in Flash memory.

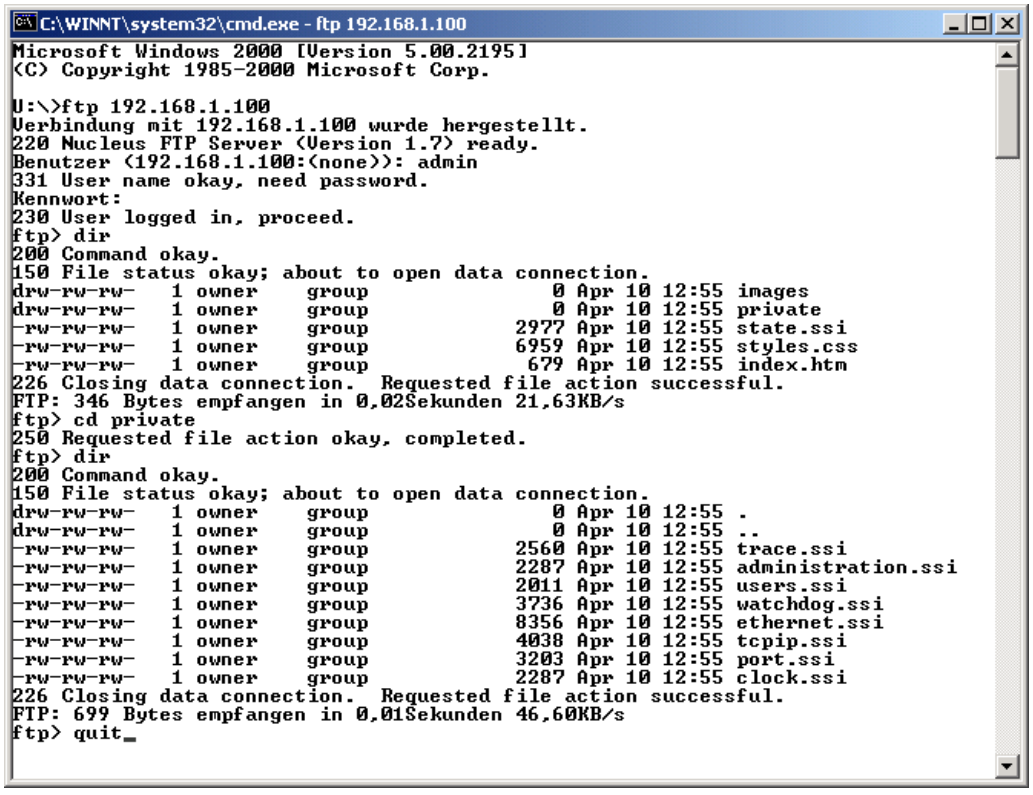
Table 87: Summary of file system partitions

Partition	Format	Type	Size	Use
R:\	FAT 12	RAM disk	1MB	-
U:\	FAT 12	Flash disk	1MB	WBM pages

The R:\ partition is designed as nonpermanent RAM disk and can be used to temporarily store data. These files are lost when the system is restarted. To save files permanently, use partitions "U:\.
You can access the partitions above via FTP.

6.3.1 Access via FTP

Partition U:\ is the start directory for incoming FTP connections.

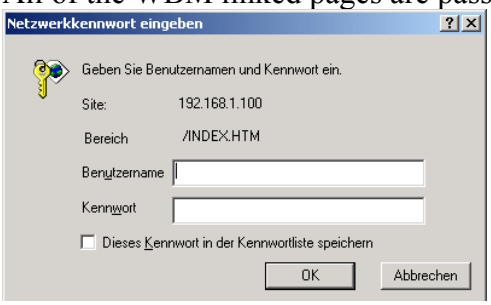


The example here illustrates the contents of the folder "U:\private\". These files represent the source code for the web-based management.

6.4 Web-based Management (WBM)

Nearly all settings can be viewed and edited via Web-Based Management. To use this, start your browser and enter "http://" into the address line, followed by the IP address of the 767 node (for example: http://192.168.1.100).

All of the WBM linked pages are password protected.



Benutzer	Passwort
admin	wago
user	user
guest	guest

Further information is given on the specific WBM page or in the 767-1301 manual.

7 WAGOframe

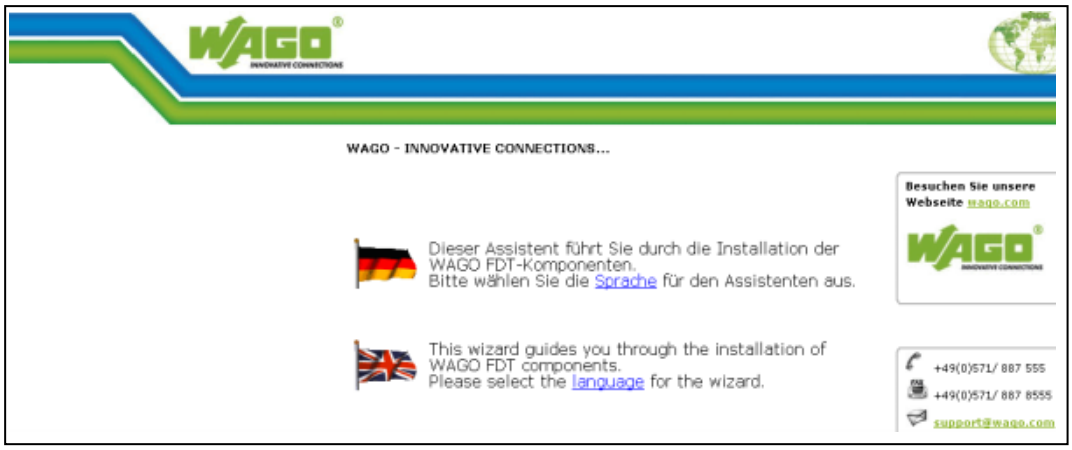
WAGOframe is an FDT/DTM frame application for configuration, diagnosis and updating of FDT-compliant field devices.

FDT/DTM is a manufacturer-independent concept for setting of parameters for field devices from different manufacturers using only a single program.The phrase "Field Device Tool"(FDT) not only represents a concrete program, but also defines the interfaces that a program must deal with in order to cooperate with DTMs from different manufacturers.

A "Device Type Manager"(DTM) groups all the setting options for a field device (including graphic interfaces) into a single program that is executed in an FDT/DTM frame application. The specification makes a distinction between device DTMs, communication DTMs and gateway DTMs.

7.1 Installation

A browser-based wizard leads you through the installation of the required drivers, DTMs and programs. When performing an installation from the CD, the Internet browser used will open automatically to the following page:



1. When performing an installation from the file system switch to the directory "~/WAGOframe CD-ROM_v3.0.0" and open the file "Deutsch_main.htm".

2. To install all the required components, select "Series 767 + WAGOframe".

3. The following page will be displayed. Execute all five installation programs one after the other to install the configuration software.

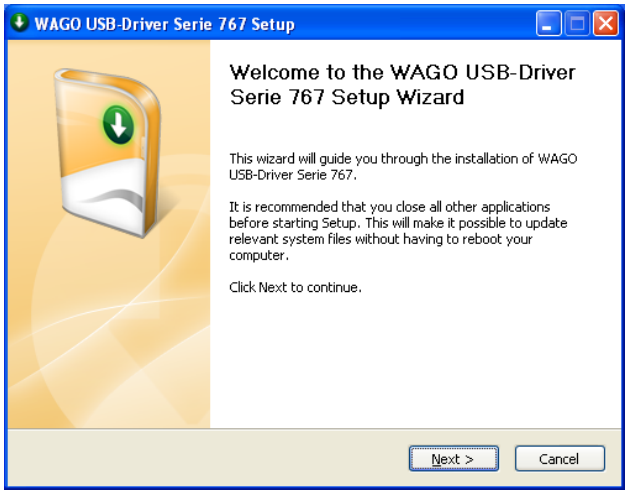
Note

Installation of the USB driver 759-922 is not required if this driver has already been installed via "CoDeSys 3" setup. Any USB driver that is already installed will then be updated.

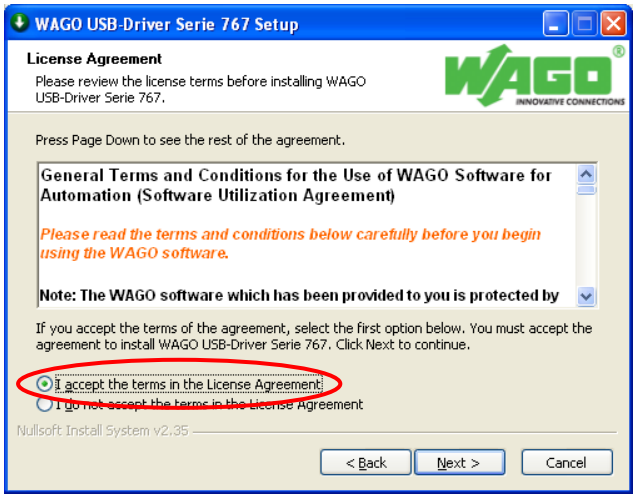
If an older version of "WAGOframe" is installed, this must first be de-installed with the installation routine. The same applies to DTMs.

The installation sequence for the individual components is always the same, beginning with language selection.

1. Continue by clicking **[OK]**.

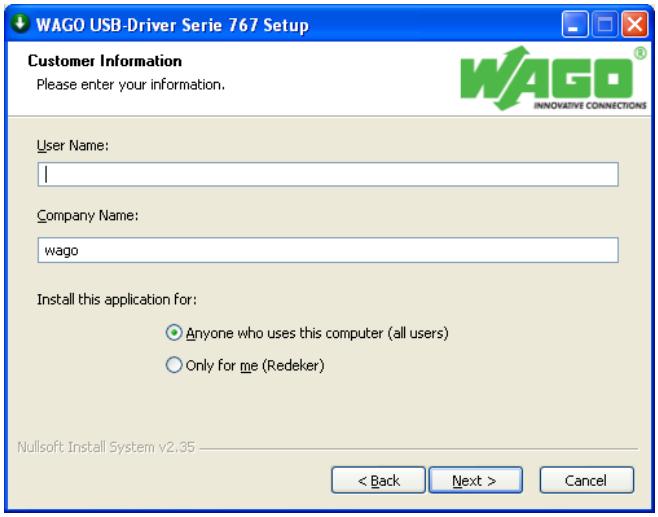


2. Continue by clicking **[Next >]**.

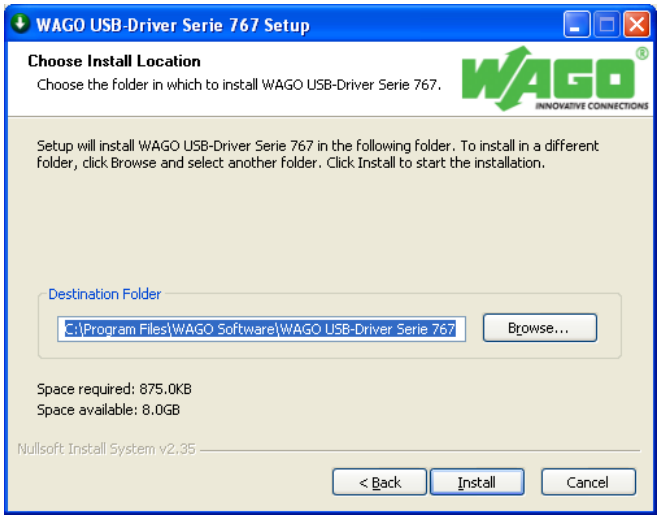


3. Read the Terms of Use for the WAGO USB driver before selecting "I accept...". Confirm this selection. Then click on **[Next]**.

4. Check your personal settings.

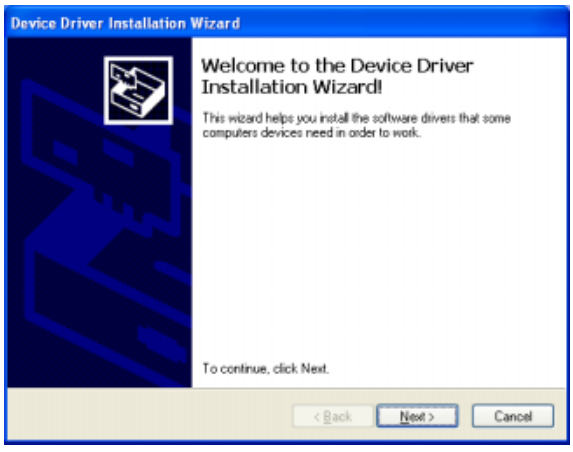


5. Continue by clicking **[Next >]**.

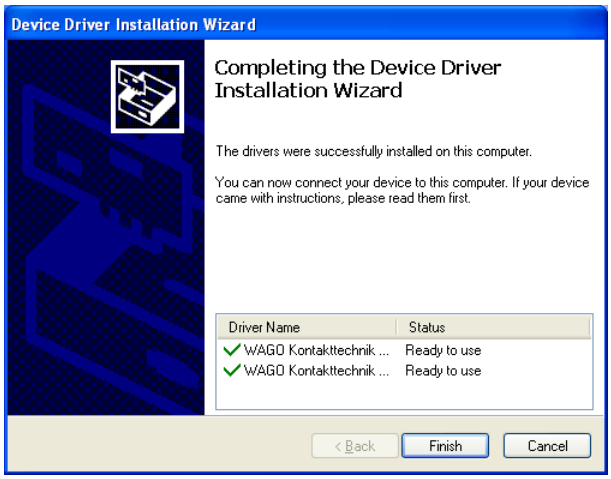


6. Select the installation path under which the application is to be saved.

7. Start the installation by clicking the **[Install]** button.



8. Continue by clicking **[Next >]**.



9. Complete the installation by clicking **[Finish]**.

10. Quit the wizard by clicking the **[Finish]** button.

Repeat these steps for the other four software components:

- WAGOframe: Item No. 759-370 (Version 3.0.0)
- WAGO service interface DTM: Item No. 759-371 (Version 2.1.0)
- DTM for the fieldbus coupler and I/O modules: Item No. 759-361 (Version 2.1.0)
- DTM for the system update: Item No. 759-362 (Version 1.0.0)

7.2 Starting Up the Fieldbus Coupler

Now switch on the power for the fieldbus coupler and link its service port to a free USB slot on your PC using the USB communication cable 756-4101/0042-0030.

Communication between WAGOframe and the fieldbus coupler is carried out via the communication DTM "WAGO Service Interface". The COM port that is used depends on the USB slot and must be parameterized accordingly in the WAGO Service Interface.

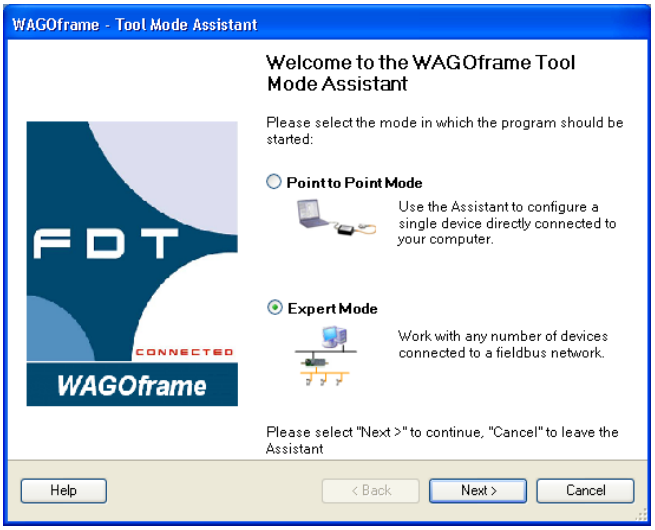
Note

If the fieldbus coupler is linked to a different USB slot at a later time, this will also result in a change to the COM ports used by "I/O-Service". In this case the setting for the serial interface must be adapted accordingly in the communication DTM.

7.3 Operation

Start the WAGOframe program using the icon on the desktop, or via **Start > Programs > WAGO Software > WAGOframe > WAGOframe**.

After starting you can select either the "Point to point mode" or the "Expert mode". The "Point to point mode" was developed specially for the configuration of simple devices (such as WAGO Jumpflex). Use the "Expert mode" to set parameters for the 767 components.

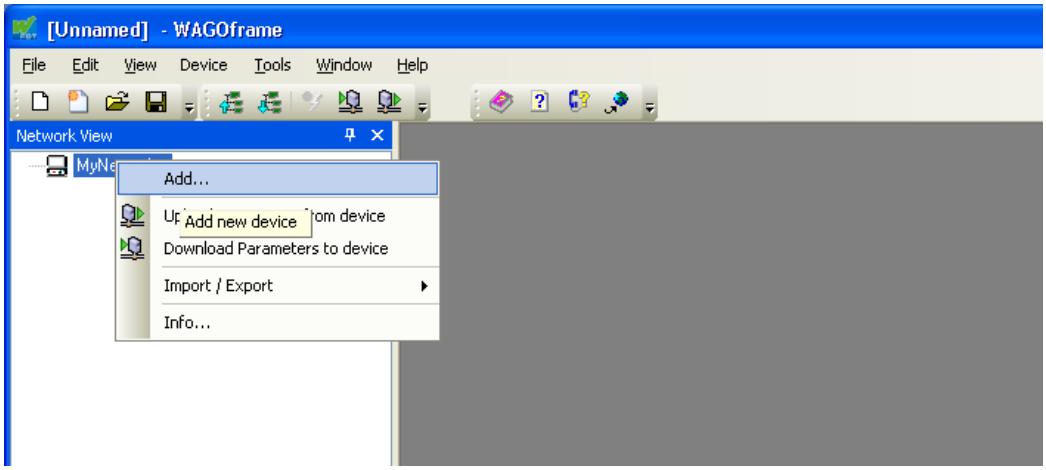


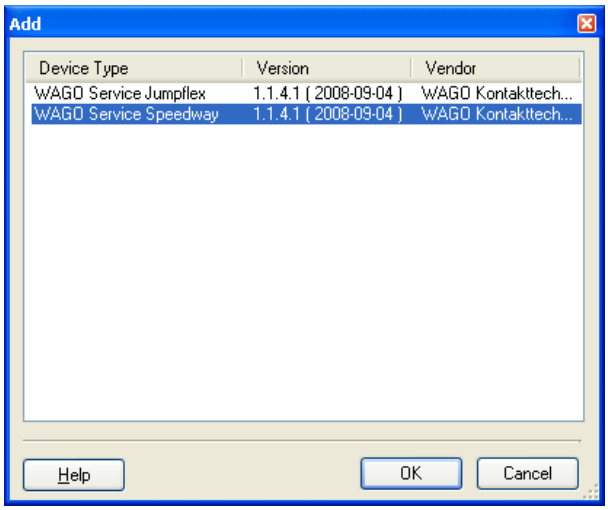
1. Switch to the "Expert mode" and click **[Next >]**.

No device catalog will have been created on the initial call-up.

2. Click **[Yes]** in the dialog window that then appears to configure a device catalog on your PC.

3. In the window "Network view" mark the element "Network" and select the function **Add...** using the context menu (right mouse key). A dialog window then opens showing all available communication drivers.

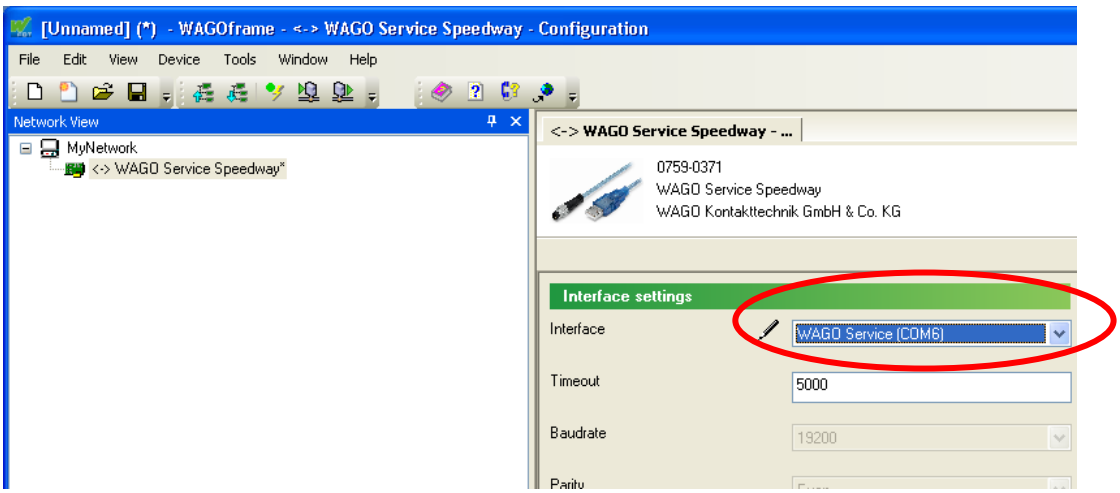




4. Select the "WAGO Service Speedway" communication driver and confirm your selection by clicking **[OK]**.

5. In the window "Network view" double-click on the newly added element "<->WAGO Service Speedway".

Setting the serial interface parameters for the "WAGO Service Interface" communication DTM.



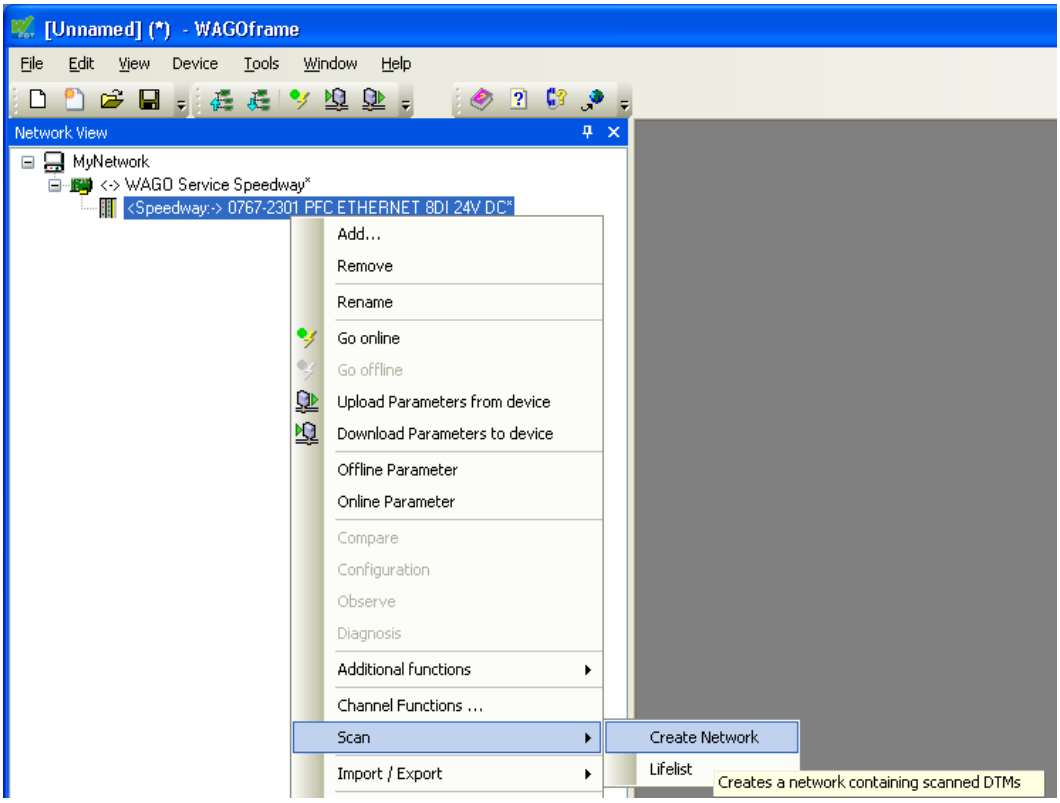
1. Select the interface to be used from the list and accept these settings.
   If there are no entries in the list of available interfaces, check to ensure that the fieldbus coupler is switched on and that it is connected to your PC by the USB communication cable.

2. To complete the network configuration you can now select individual DTMs from the catalog, or conduct a network scan.

When adding elements manually please note that each FDT frame application differentiates between online and offline modes. Each of these modes has a separate scope of functions.

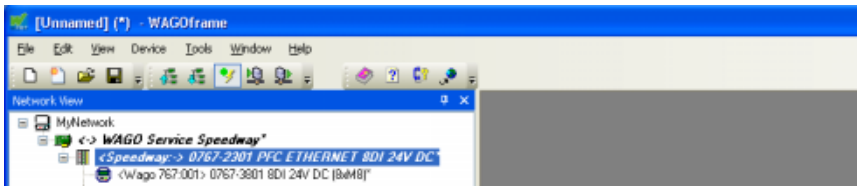
In the online mode, there is a direct link between the display and the connected 767 components.
The offline mode enables the parameterization of a device that is not yet present. It will continue to be used to reduce data transfer between WAGOframe and the device. If a device is in the online mode, its name is displayed in ***bold and cursive*** font in the network window.

3. To conduct a network scan, open the context menu (right mouse key) of the element "WAGO Service Speedway" in the "Network" window and then select the entry **Create network** under the menu item **Scan**.



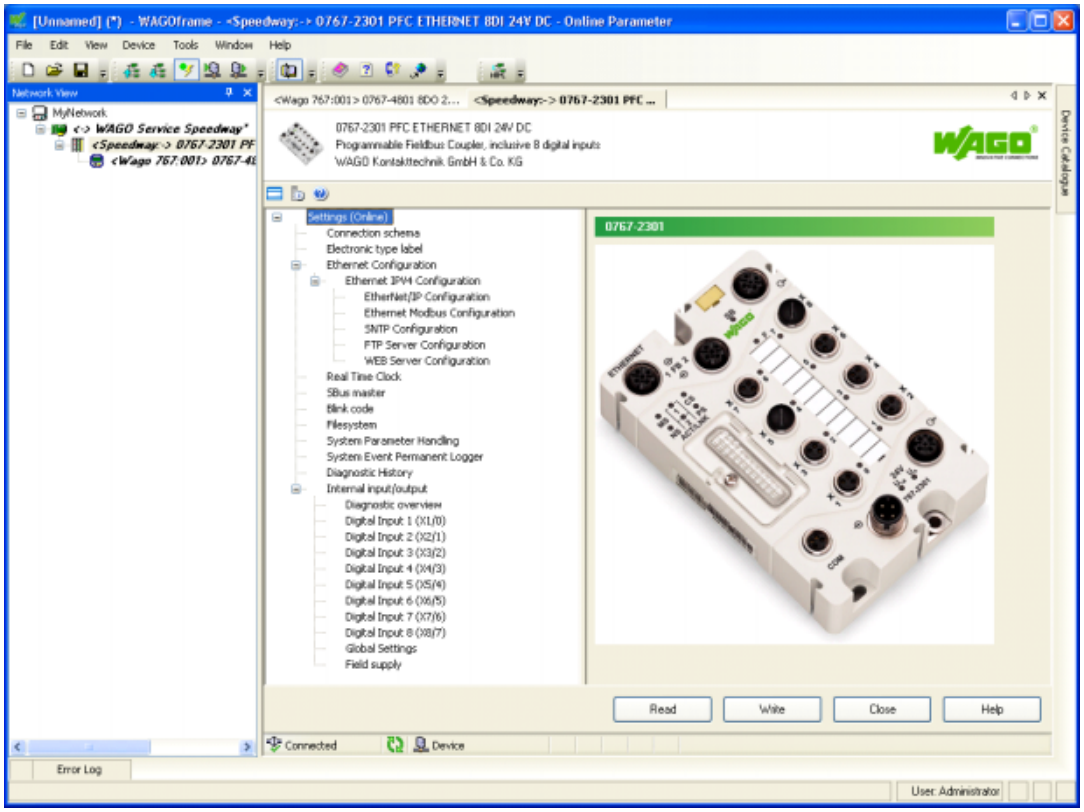
4. After performing a network scan, WAGOframe should have found the fieldbus coupler and the connected modules .

5. A "Live List" is created if no matching DTM is found for the fieldbus coupler when scanning the network. Click on **[Add all and continue]** in the list.



6. Note that the fieldbus coupler is in the online mode, while the module is in the offline mode. Mark module and select the entry **[Go online]** from the context menu to switch to the online mode.

7. Open the detail view by double-clicking the fieldbus coupler. WAGOframe should appear as follows after editing the window width and closing the device catalog:



The current operating mode is shown in brackets behind the first node of the tree structure (left window). Please note that changing between operating modes is only possible when the detail window concerned is closed.

Explanations of the buttons:

- **[Read]**
  Reads and displays the paremeters found in the fieldbus coupler.

- **[Write]**
  Writes the modified values to the fieldbus coupler.

- **[Close]**
  Closes the parameterization user interface (DTM).

- **[Help]**
  Opens the online help for an entry selected previously (e.g., digital input, blink code).

8 System Update

Firmware update for the 767 Series components is performed via system update. To ensure that the fieldbus node remains consistent and executable after updating the firmware, system update must be performed for both fieldbus coupler and connected I/O modules.

NOTICE

System update
Before updating, observe the following measures to prevent any possible damage to the 767 system:
- The power supply must not be disconnected while updating!
- To exclude any interferences by the fieldbus, the fieldbus cable must be disconnected before updating!

Requirement:
- You have installed the DTM (759-370) WAGOframe
- You have installed the DTM (759-371) WAGO service interface
- You have installed the DTM (759-922) USB driver
- You have installed the DTM (759-362) system update.
- Update packages are available for the connected 767 Series components.

System Update Procedure

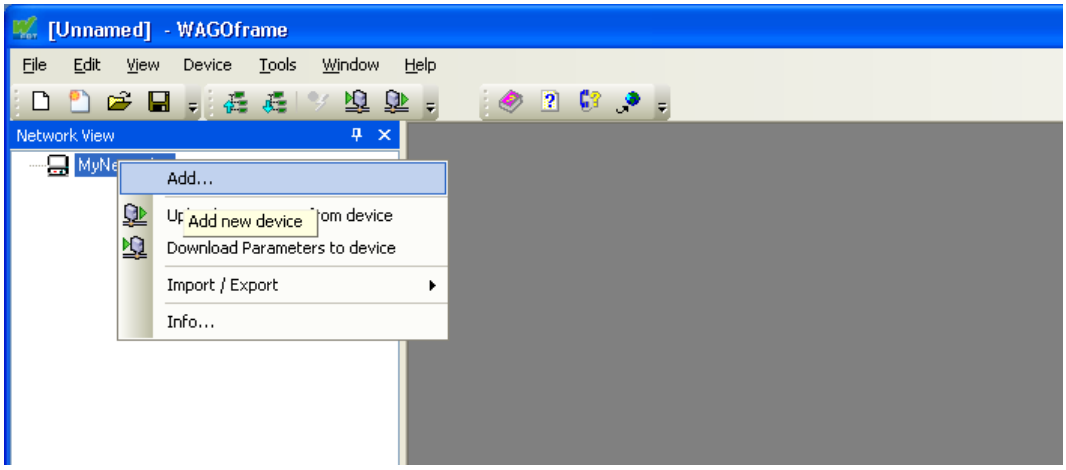
To perform a system update for each 767 Series component, please complete the following steps:

1. Read 767 components' parameters and save them on your PC.
2. Update 767 Series components' firmware.
3. Rewrite parameters from your PC to the 767 Series components.
4. Set parameters to valid and finish the procedure.

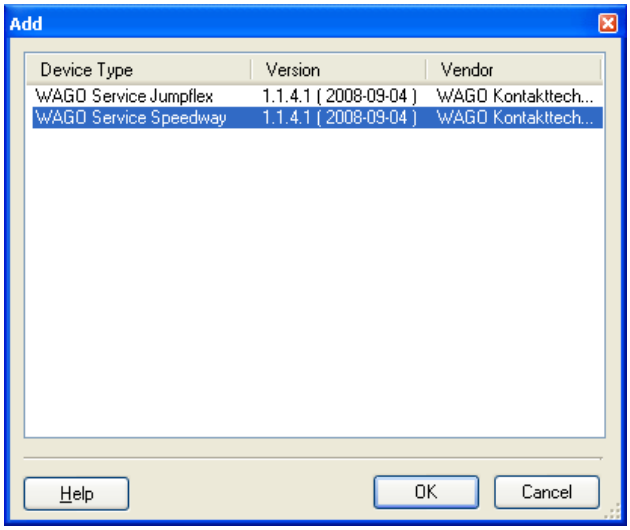
8.1 Adding the DTM System Update

To add the DTM system update to WAGOframe, please proceed as follows:

1. Right-click on "MyNetwork" in the "Network View" pane.



2. In the context menu, select **Add…**. The "Add..." dialog box opens.

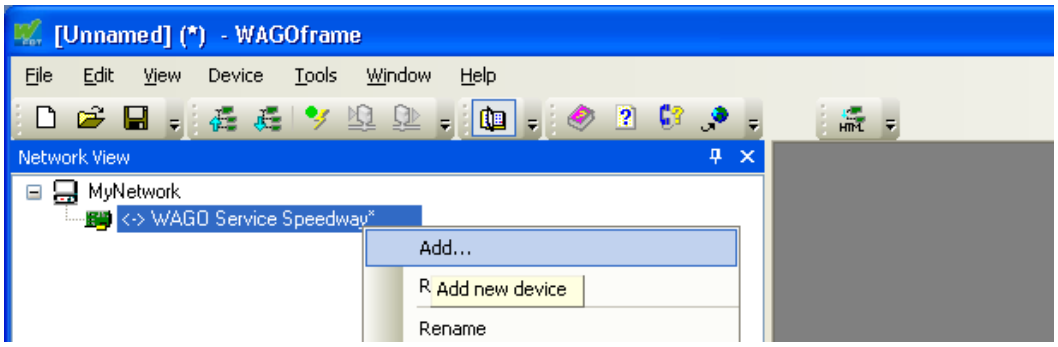


3. In the "Add" dialog box, select **WAGO Service Speedway**.

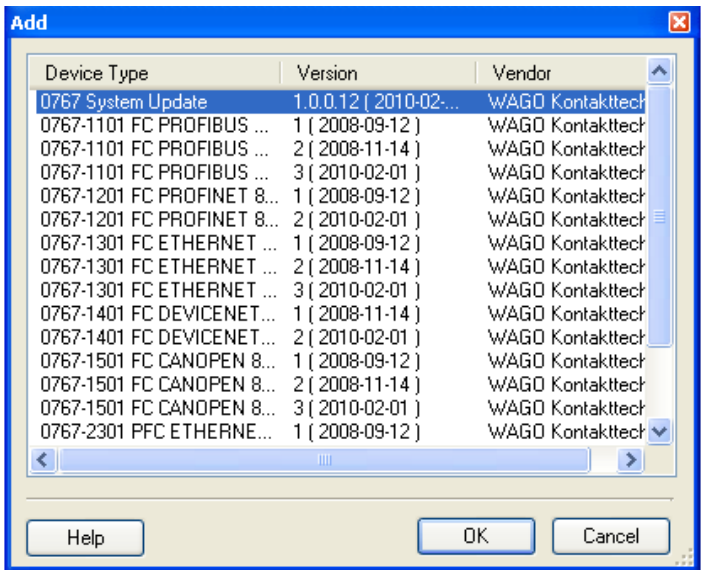
4. Click **[OK]** to confirm your selection.

5. In "Network View", right-click on the "WAGO Service Speedway" device driver.

6. In the **Add...** context menu, select "Add". The "Add" dialog box opens.



7. In the "Add" dialog box, select the DTM **0767 System Update**.



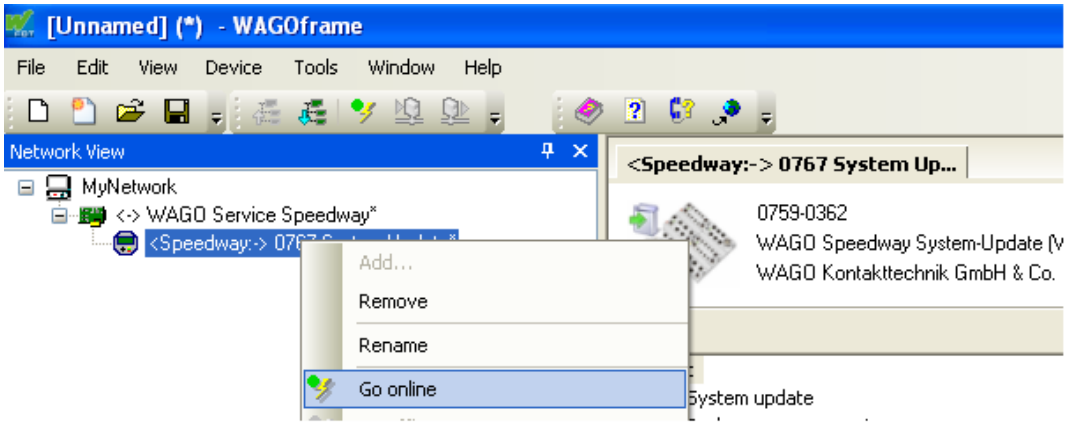
8. Click **[OK]** to confirm your selection.

8.2 Go online to 767 Nodes using Update DTM

The firmware can only be updated when a communication connection exists between Update DTM and 767 node. Please proceed as follows:

1. In the "Network View", right-click on the "<Speedway:> 0767 System Update" device driver.

2. In the context menu, select **Go online**. When the progress bar displays 100% and the entry is displayed in ***bold italics***, the communication connection is established.



8.3 Updating the 767 Components

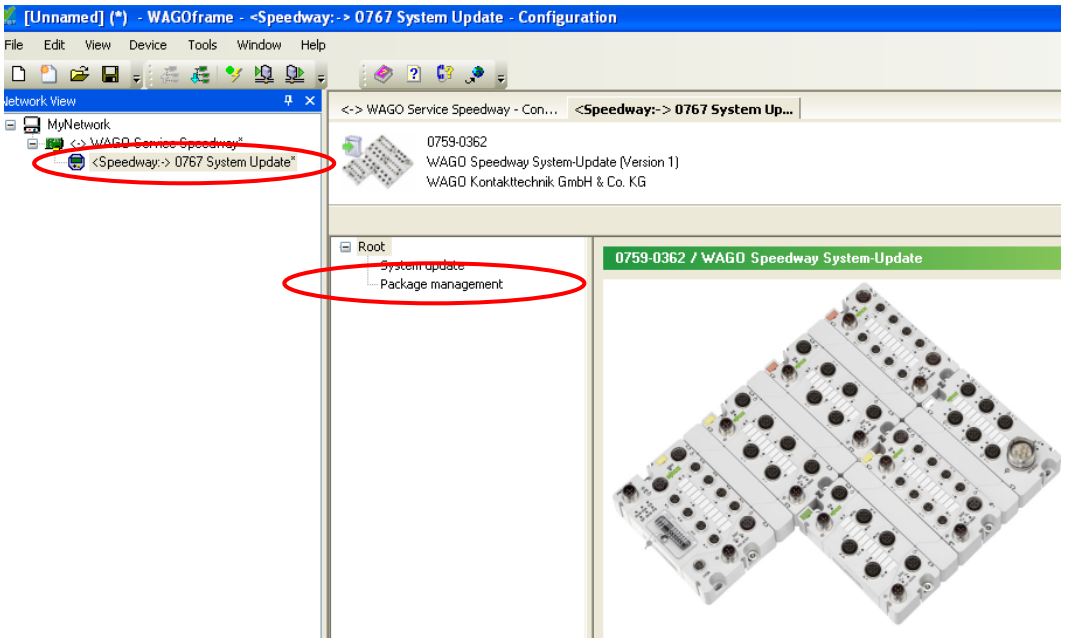
The current firmware is available from WAGO Support. Send an e-mail with the subject "Current Speedway Firmware" and the item number of the respective 767 components to: support@wago.com.

Import firmware packages

To use the received firmware packages, import them into the DTM system update. Please proceed as follows:

1. Save the received firmware packages with the "*.wup" extension to any directory on your PC.

2. Open the DTM user interface by double clicking **"0767 System Update"** in the network view.



3. Click "Package Management" in the left window of WAGOframe.

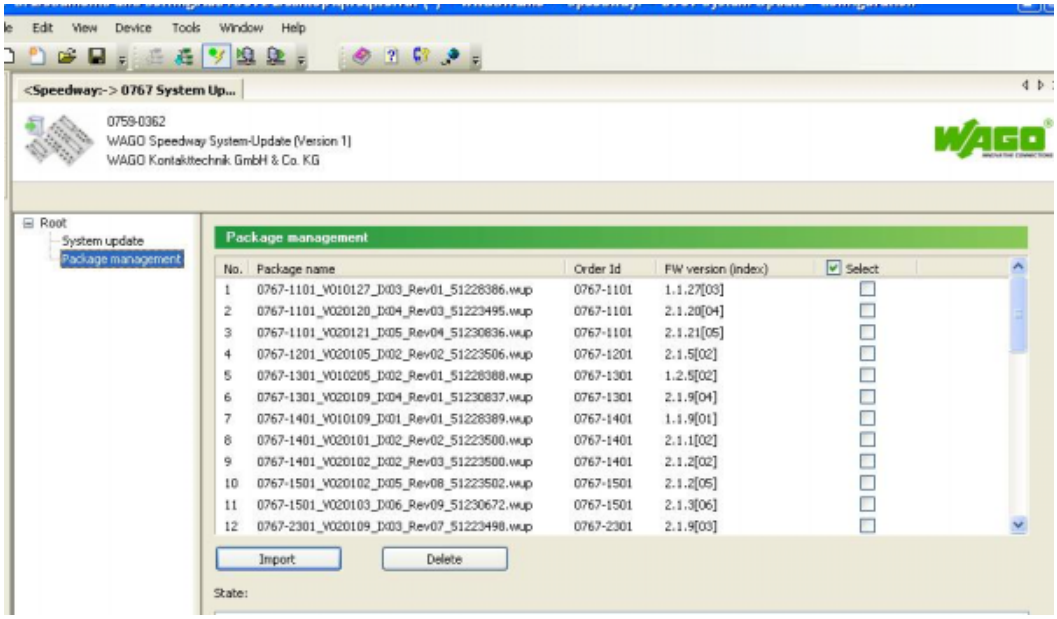
4. To import the received firmware files, click **[Import]**. In the window that opens, go to the directory where you have saved the firmware files and select the file to be used. Click **[Open]** to apply the files.

Delete firmware packages

To maintain a clear "Package Management" interface, you can remove unneeded update packages from the view. Please proceed as follows:

1. Select the required firmware files in the right window.

2. Click **[Delete]** to remove the selected firmware packages.



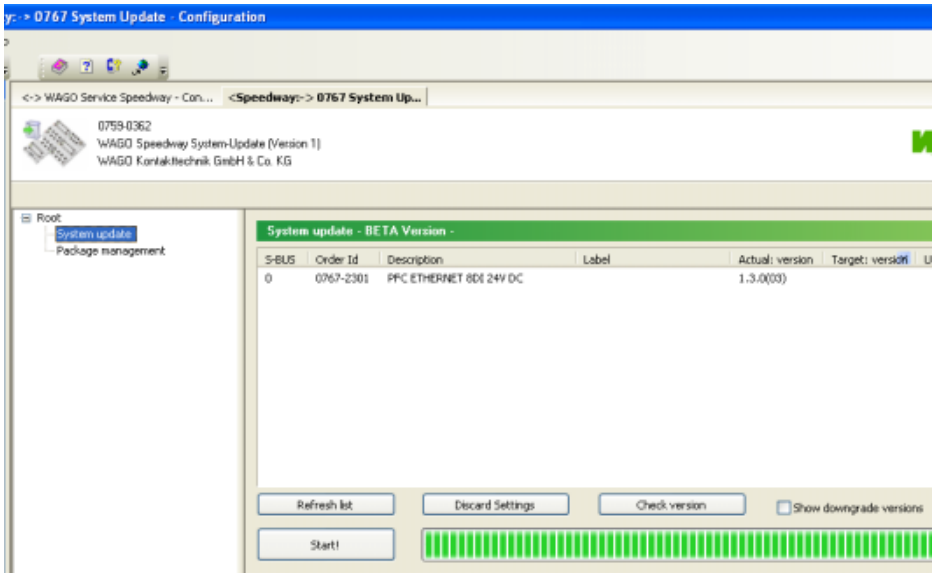
System update

Note

When updating the firmware of the fieldbus coupler, the saved module parameters may be overwritten. Therefore, check your existing configuration after updating the firmware.

Perform the system update here. The module settings you made normally remain unchanged. Otherwise, a corresponding warning message appears. If you still want to update the firmware, the 767 components are returned to their default state.

1. Click "System Update" in the left window.

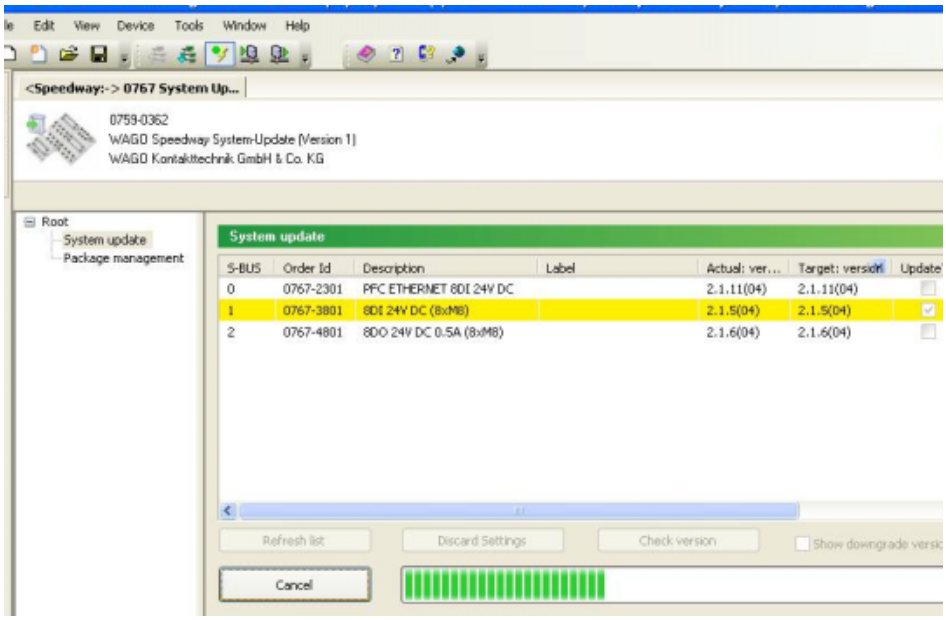


2. The fieldbus coupler with all connected I/O modules are listed in the right window. All 767 components that can be updated are pre-selected. If the pre-selection is incorrect or if specific 767 components should not be updated, deselect them.
   "Actual: Version": Firmware currently present on the device
   "Target: Version": Version of the firmware that should be loaded into the 767 components. If multiple "Target" versions can be selected, select the one relevant to you.

3. Click **[Start!]** to update the system. The 767 components are marked in yellow while being updated.

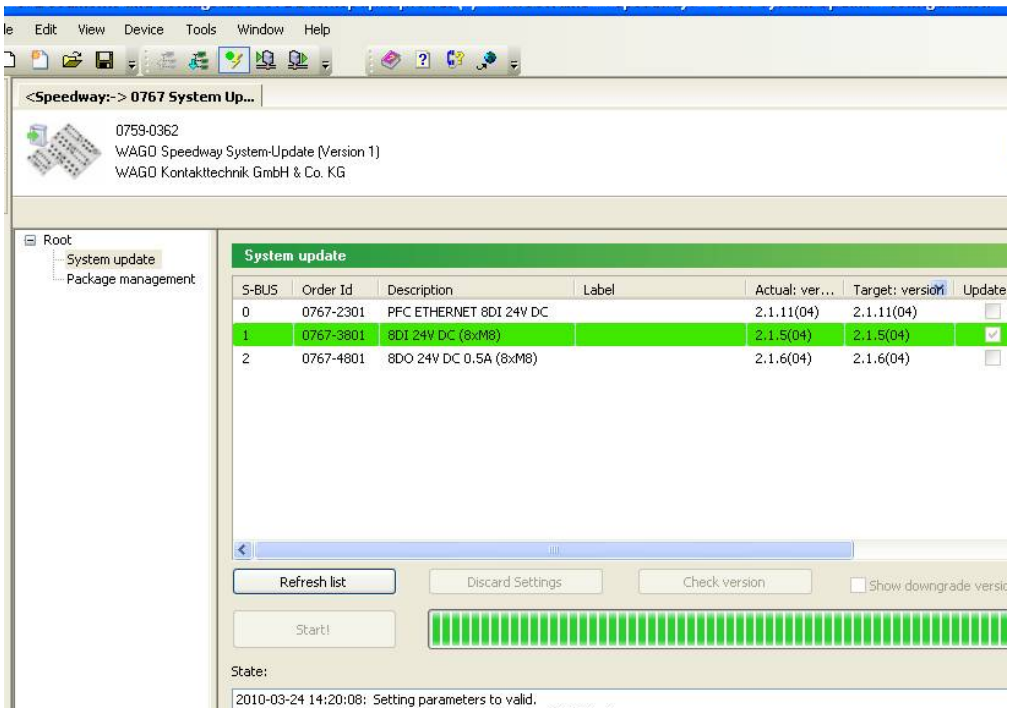
Note

During the firmware updating process, the fieldbus coupler disconnects each of its COM ports. A PC equipped with Windows 2000 will detect this, and a Windows message appears. This is not an error message. Confirm the message by pressing **[OK]**.



Button	Description
[Update List]	Use this function to read the node design and the view updates.
[Discard Settings]	Discard the selections and settings performed by you.
[Check Versions]	If you made your selection, click this button to perform a plausibility check. This checks whether the configuration you selected is possible (also automatically performed when starting the system update).
Display downgrade versions	If this checkbox is selected, the versions to downgrade a device are also displayed in the list of target versions.
[Start!]/[Abort]	Start/abort system udpate.

4. If the system update is finished, the updated 767 components are displayed in green (see figure).



During system update, all relevant information is stored on your PC. Update can be repeated subsequently if the system update fails (components are displayed in red). In this case, original parameters remain unchanged.

If the update for a device fails, please contact WAGO Support.

Information

Further information is given in the WAGOframe manual.

WAGO Kontakttechnik GmbH & Co. KG
Postfach 2880 D-32385 Minden
Hansastraße 27 D-32423 Minden
Telefon: 05 71/8 87 – 0
Telefax: 05 71/8 87 – 1 69
E-Mail: info@wago.com

Internet: http://www.wago.com