Industrial Automation

System *HIMatrix*

PROFIBUS DP Master/Slave

Manual

HIMA Paul Hildebrandt GmbH
Industrial Automation

HI 800 009 EED

For further information, please refer to the documentation contained in the CD-ROM „ELOP II Factory“ or consult our website www.hima.de.

Information requests should be sent to:

HIMA Paul Hildebrandt GmbH
Postfach 1261
68777 Brühl
Tel: +49(6202)709 0
Fax: +49(6202)709 107
e-mail: info@hima.com

Table of Contents Page

About this manual

This manual describes the configuration and operation of the HIMatrix PROFIBUS DP master and the HIMatrix PROFIBUS DP slave. It is addressed to qualified users who are familiar with the *ELOP II Factory programming* tool and the HIMA HIMatrix controllers.

This manual is organized in four parts:

1. *HIMatrix PROFIBUS DP Master* explains the menu functions and the dialog boxes within *ELOP II Factory* used to configure the HIMatrix PROFIBUS DP master.

2. *HIMatrix PROFIBUS DP Function Blocks* describes the function and configuration of the PROFIBUS DP master function blocks.

3. *HIMatrix PROFIBUS DP Slave* explains the menu functions and the dialog boxes within *ELOP II Factory* used to configure the HIMatrix PROFIBUS DP slave.

4. *Basics of PROFIBUS DP* presents to interested users the background information about the HIMatrix PROFIBUS DP protocol.

For further information about PROFIBUS DP, please refer to the following specifications provided by PNO Germany (see www.profibus.com):

- PROFIBUS Technology and Application, October 2002
- PROFIBUS Guideline No. 2.182 Version 1.2, July 2001

For questions, please contact HIMA directly.

Terminology

Term	Definition
ASIC	Application-specific integrated circuit
Cfg	Configuration control
CRC	Cyclic Redundancy Check
COM	Communication Module
CPU	Central module
DDLM	DP user interface (Direct Data Link Mapper)
DP	Decentralized Peripherals
DP-V0	DP Version 0: It makes the basic DP functionality available
DP-V1	DP Version 1: Additional elements, e.g. acyclic data traffic
DP-V2	DP Version 2: Further additional elements, e.g. isochronous slave operation
EMC	Electromagnetic Compatibility
EN	European Norm
FB	Field bus
FBD	Function Block Diagram
FDL	Telegram for recognizing a new master
GSD file	The GSD file contains the device master data of a product and is provided by the device manufacturer.
HAS	Highest Station Address
HWM	ELOP II Factory Hardware Management
Identifier	16 bit unique number assigned to a device PNO Germany (www.profibus.com)
LAS	List of Active Stations
Class 1 master	Master for user data traffic
Class 2 master	Master for project planning and commissioning (PC)
NIL	Nothing or null
ISO	Internationally Standard Organization
PADT (PC)	Programming and Debugging Tool (in accordance with IEC 61131-3)
PES	Programmable Electronic System
PI	PROFIBUS International
PM	ELOP II Factory Project Management
PNO	PROFIBUS Nutzerorganistion e.V.
SAP	Service Access Point
Slave	A slave is a passive station that exchanges reference data with the master.
Tbit	Unit of time for transferring one bit, reciprocal of the transfer rate (Example: 1Tbit at 12 Mbps = of 83 ns).
TMO	Time out
UART	Universal Asynchronous Receiver/ Transmitter
Validate	To confirm the validity of something

Introduction

PROFIBUS DP is an international, open field bus standard that is used when a fast reaction time is required for small amounts of data.

The HIMatrix PROFIBUS DP master fulfills the criteria specified in the European norm EN 50170 [5] and the globally binding IEC standard 61158 for PROFIBUS DP.

The HIMatrix PROFIBUS DP master can exchange cyclic and/or acyclic data with the connected PROFIBUS DP slave.

For the acyclic data traffic, different function blocks are available in ELOP II Factory. With these function blocks, the user can optimally adapt the HIMatrix PROFIBUS DP master to the requirements of the PROFIBUS DP slaves and of his project.

1 HIMatrix PROFIBUS DP Master

This chapter describes the characteristics of the HIMatrix PROFIBUS DP master and the ***ELOP II Factory*** menu functions and dialog boxes required for configuring the HIMatrix PROFIBUS DP master.

Note	For each HIMatrix controller type, the user can refer to the corresponding system documentation specifying the electrical and mechanical data. (See Engineering Manual HI 800 101 and data sheets of the HIMatrix controllers).

1.1 Equipment and System Requirements

HIMA ELOP II Factory	Version 5.2.0 and beyond
HIMatrix controllers	F20 F30, F35 and F60, Hardware Revision 02 and beyond
Operating system versions of the HIMatrix controllers	COM OS version 6.22 and beyond CPU OS version 4.50 and beyond
HIMatrix PROFIBUS DP Master Module	The serial field bus interface (FB1 or FB2) used on the HIMatrix controller must be equipped with an optional HIMatrix PROFIBUS DP module.
License number	Not used, activation via the module

1.2 PROFIBUS DP Master Features

Type of HIMatrix PROFIBUS DP master	DPV1 Master Class 1 with additional DPV2 Functions
Transfer rate	9.6 kbit/s up to 12 Mbit/s
Bus address	0 up to 125
Max. number of PROFIBUS DP masters	Only one PROFIBUS DP master can be configured for one resource. F30, F35, F60: optionally also two masters
Max. number of PROFIBUS DP slaves	Up to 125[1] slaves can be configured for one resource (in all master protocol instances); however, a maximum of 32 stations can be connected to a bus segment without repeaters.
Max. process data length per slave	DP-Output: max. 244 bytes DP-Input: max. 244 bytes
Connection monitoring	If the PROFIBUS-DP master is in the OPERATE state and the connection to the PROFIBUS DP slave is lost, the PROFIBUS-DP master detects this within a few of PROFIBUS-DP cycles. In this case, the connection state is set to OFF-LINE. The PROFIBUS-DP slaves input signals are ignored and instead initial values are used.

[1] According to the standard, a total of three repeaters may be used, such that a maximum of 122 slaves can be connected to each serial interface on a master.

Note	In addition to the PROFIBUS DP protocol, other protocols (e.g. Modbus, TCP S/R, …) can be simultaneously operated on a *HIMatrix* controller. Each *HIMatrix* controller can transmit a total of 16284 bytes of data and receive a total of 16284 bytes of data. The user can allocate the 16284 bytes freely among the different protocols, but no more than 8192 bytes may be used per protocol and direction.

1.3 Creating a HIMatrix PROFIBUS DP Master

Start *ELOP II Factory* and create a new project or load an existing project.

Switch to Hardware Management and select **New, PROFIBUS Master** from the protocol context menu to create a new PROFIBUS DP master in the resource.

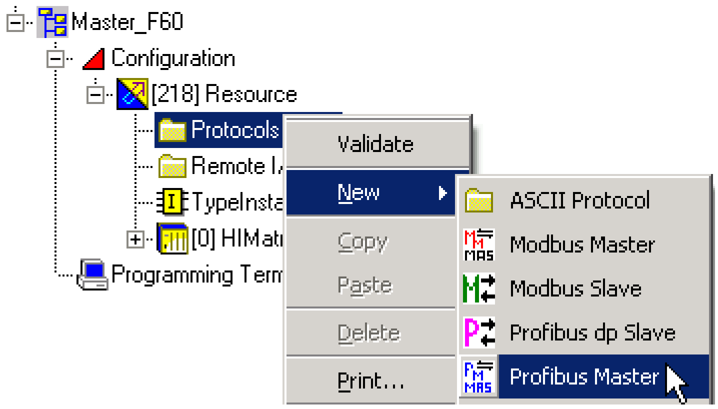


Figure 1: New *PROFIBUS DP Master*

1.4 Context Menu for HIMatrix PROFIBUS DP Master

The context menu for the HIMatrix PROFIBUS DP master contains the following functions.

PROFIBUS DP Master
Connect Signals
Validate
New
Import
Export
Copy
Paste
Delete
Print
Properties

1.4.1 Menu Function *Connect Signals*

The *Connect Signals* context menu function opens the *Signal Connections* dialog box.
The *Signal Connections* dialog box provides the two status signals

- Bus Error and
- Master State

for assessing the PROFIBUS DP master's status within the user program.

Signal	Description	Type
Bus Error	If a bus error occurs, an error code is set in the system signal *Bus Error*. An error code remains set until the bus error has been eliminated. Error codes: 0: OK, no bus error 1: Address error The master address is already available on the bus. 2: Bus malfunction A malfunction was detected on the bus, (e.g. bus was not properly terminated, and several stations are sending data simultaneously). 3: Protocol error An incorrect coded packet was received. 4: Hardware error The hardware reported an error, e.g. too short time periods. 5: Unknown error The master changed the status of an unknown reason. 6: Controller Reset The controller chip is reset if a serious bus error occurs. The status signal *M1BusErr* can be assessed within the application program	BYTE
Master State	The master state indicates the current protocol state. 0: OFFLINE 1: STOP 2: CLEAR 3: OPERATE (see chapter 1.6.4) The status signal *Master Status* can be assessed within the application program	BYTE

Table 1: PROFIBUS DP Master's Status Signals

1.4.2 Standard Menu Functions

1.4.2.1 Menu Function *Validate*

The master's and slave's parameter settings can be checked prior to generating the code. For this purpose, the corresponding object must be chosen in the structure tree and *Validate* must be selected from the context menu. Errors and warnings appear in the error-state display field.

Validating is also automatically performed prior to every code generation. If an error is detected during the validation process, the code generation is aborted.

1.4.2.2 Menu Function *New*

Selecting **New** to add new objects,

1.4.2.3 Menu Functions *Import* and *Export*

The **Import** and **Export** context menu functions display the elements in which can imported and exported via csv files.

1.4.2.4 Menu Functions Copy, Paste, Delete, Print

Copy:	Copy the object into the clipboard.
Paste:	Add the object to the clipboard.
Delete:	Delete the object from the project.
Print:	Send the object's configuration to the selected Printer.

1.4.3 Menu Function *Properties*

The *Properties* context menu function opens the *Properties* dialog box.
The dialog box contains the tabs *General*, *Timings, CPU/COM* and *Other*.

1.4.3.1 Tab *General*

Element	Description	Value
Type	PROFIBUS DP master	Display only
Name	Any unique name for a PROFIBUS DP master	
Address	Master station address Only one master station address may be present on the bus once.	Min: 0 Max: 125 Standard: 0
Interface	COM interface that should be used for the master.	FB1, FB2
Baud rate	Baud rate at which the bus is operated. Possible values: 9600 (9.6 kBaud) 19200 (19.2 kBaud) 45450 (45.45 kBaud) 93750 (93.75 kBaud) 187500 (187.5 kBaud) 500000 (500 kBaud) 1500000 (1.5 MBds) 3000000 (3 MBds) 6000000 (6 MBds) 12000000 (12 MBds)	

Table 2: General Settings for the PROFIBUS DP Master in the *Properties* Dialog Box

1.4.3.2 Tab *Timings*

Element	Description	Remark
MinTsdr [bit time]	Min. Station Delay Time Minimum time period that a PROFIBUS DP slave must wait before it may respond.	Min: 11 Max: 1023 Standard: 11
MaxTsdr [bit time]	Max. Station Delay Time: Maximum time period that a PROFIBUS DP slave may need to respond. MaxTsdr ≥ max (MaxTsdr of all slaves) The MaxTsdr values of the slaves are read from the GSD files and appear in the *Baud rates* tab located in the slave *Properties* dialog boxes	Min: 37 Max: 65525 Standard: 37
Tsl [bit time]	Slot Time Maximum time span that the master waits for a slave's acknowledgment. Tsl > MaxTsdr + 2*Tset +Tqui + 13	Min: 37 Max: 16383 Standard: 37
Tqui [bit time]	Quiet Time for Modulator Time that a station may need to switch from sending to receiving.	Min: 0 Max: 493 Standard: 0

Tset [bit time]	Setup Time Time for reacting to an event.	Min: 1 Max: 494 Standard: 1
Ttr [bit time]	Target token rotation time Maximum time available for a token rotation. A lower estimate of the Ttr is obtained using the following formula: Ttr=n*(198+T1+T2)+b*11+242+T1 +T2+Tsl See chapter 1.4.5	Min: 256 Max: 16777215 Standard: 999
Ttr [ms]	Target rotation time in ms	Display only
Min. Slave Interval [ms]	Minimum time interval for a PROFIBUS DP cycle The master must observe the *Min. Slave Interval* and not fall below it. However, the PROFIBUS DP cycle can be extended if the isochron mode is inactive and the portion of acyclic telegrams increases within a cycle. The value for the salve's *Min. Slave Interval* is read from the GSD file and appears in the *Features* tab located in the *Properties* dialog box. In the isochron mode, the value for *Min. Slave Interval* defines the time period for an isochron cycle. The isochron mode is activated if the options *Isochron Mode Sync* or *Isochron Mode Freeze* are activated. See also Refresh Rate [ms] between CPU/COM, chapter 1.4.3.3.	Min: 0 Max: 6553.5 Standard: 1
Data Control Time [ms]	Time span within which the master must report its current state on the bus Standard value: Data Control Time = slave's WDT	Range of values 0-65535 [10 ms] Standard: 1000

Table 3: Timings Settings for the PROFIBUS DP Master in the *Properties* Dialog Box

1.4.3.3 Tab *CPU/COM*

The parameters' default values provide the fastest possible data exchange of the PROFIBUS DP data between the COM processor (COM) and the CPU processor (CPU) within the *HIMatrix* controller.

These parameters should only be changed if the COM and/or CPU load must be lowered for an application and provided the process allows it.

Only experienced programmers should modify the parameters.

Increasing the COM and CPU refresh rate means that the effective refresh rate of the PROFIBUS DP data is also increased. The time requirements of the plant must be verified.

Please also consider the parameter *Min. Slave Interval [ms]* (see 1.4.3.2) which defines the refresh rate of the PROFIBUS DP data from/to the PROFIBUS DP slave.

The refresh rate of the PROFIBUS DP data can be increased according to the CPU/COM refresh rate.

Element	**Description**
Refresh Rate [ms]	Refresh rate in milliseconds at which the COM and CPU exchange protocol data. If the *Refresh Rate* is zero or lower than the controller's cycle time, data are exchanged as fast as possible. Range of values: 0 to (2^{31}-1) Default value: 0
Within one cycle	Activated: Transfer of all protocol data from the CPU to the COM within a CPU cycle. Deactivated: Transfer of all protocol data from the CPU to the COM, distributed over multiple CPU cycles, each with 900 bytes per data direction. This way, the controller's cycle time can also be lowered. Default value: activated

Table 4: PROFIBUS DP Master's Properties

1.4.3.4 Tab *Other*

Element	Description	Remark
Max. Retry Limit	Maximum number of resends attempted by a master if a slave does not respond.	Min: 0 Max: 7 Standard: 1
Highest Active Address	Highest Station Address (HSA) Highest station address to be expected for one master. Masters having a station address beyond the HSA are not included into the token ring.	Min: 0 Max: 125 Standard: 125
Isochron Mode Sync	The *Isochron Mode Sync* allows both a clock-controlled synchronization of the master and the slaves, and a simultaneous activation of the physical outputs of several slaves. If the *Isochron Mode Sync* is active, the master sends the *Sync* control command as a broadcast telegram to all slaves. As soon as the slaves supporting the *Isochron Mode Sync* receive the *Sync* control command, they synchronously switch the data from the application program to the physical outputs. The physical outputs' values remain frozen up to the next *Sync* control command. The cycle time is defined by the *Min. Slave Interval*. Condition: Ttr < Min. Slave Interval	Standard: FALSE
Isochron Mode Freeze	The *Isochron Mode Freeze* allows the master to simultaneously accept the input data from several slaves. If the *Isochron Mode Freeze* is active, the master sends the *Freeze* control command as a broadcast telegram to all slaves. As soon as the slaves supporting the *Isochron Freeze Mode* receive the *Freeze* control command , the physical inputs' signals are frozen to the current value. The master can thus read the values. The input data are only updated when the next *Freeze* control command is sent. The cycle time is defined by the *Min. Slave Interval*. Condition: Ttr < Min. Slave Interval	Standard: FALSE
Auto-Clear on error	If *Auto-Clear on error* is set in a slave that fails, the master assumes the CLEAR state.	Standard: FALSE
Time Master	The master is also time master and periodically sends the system time via the bus.	Standard: FALSE
Clock Sync Interval [ms]	Clock Synchronization Interval. Time interval within which the time master sends the system time over the bus.	Min: 0 Max: 65535 Standard: 0

Table 5: Other Settings for the PROFIBUS DP Master in the *Properties* Dialog Box

1.4.4 Isochron Mode (DPV2 and higher)

This function allows a clock-controlled synchronization in the master and the slaves, irrespective of congestion on the bus. The bus cycle is synchronized with a clock difference of <10 µs. Highly precise positioning processes can be thus implemented (see also page 110).

Note	To a certain degree, slaves (DPV0 slaves) that do not support isochron mode can also benefit from its advantages. To do so, the slaves must be assigned to group 8 and the parameters *Sync* and/or *Freeze* must be activated. The Sync and Freeze Modes are normally used simultaneously.

1.4.4.1 Isochron Mode Sync (DPV2 and higher)

The Isochron Sync Mode allows both a clock-controlled synchronization in the master and the slaves and the simultaneous activation of the outputs of several slaves.

1.4.4.2 Isochron Mode Freeze (DPV2 and higher)

The Isochron Mode Freeze allows the simultaneous transfer of the input data of multiple slaves.

1.4.5 Standard Values for different Transfer Rates

While configuring the PROFIBUS DP master, it must be considered that some parameters set in the *Timings* tab depend on the baud rate set in the *General* tab. For the first (initial) configuration, please use the standard values specified in Table 6. In a later step, the values are optimized.

	9.6k	**19.2k**	**45.45k**	**93.75k**	**187.5k**	**500k**	**1.5M**	**3M**	**6M**	**12M**
MinTsdr	11	11	11	11	11	11	11	11	11	11
MaxTsdr	60	60	400	60	60	100	150	250	450	800
Tsl bit time	100	100	640	100	100	200	300	400	600	1000
Tqui bit time	0	0	0	0	0	0	0	3	6	9
Tset bit time	1	1	95	1	1	1	1	4	8	16

Table 6: Default Values for Different Transfer Rates

All time values specified in Table 6 are expressed in T_{bit}
($1T_{bit}$ = 1/[bit/s]).

MinTsdr has at least 11 T_{bit},
as one character consists of 11 bits (1 start bit, 1 stop bit, 1 parity bit, 8 data bits).

Transmission Time for a Character		
Baud rate	T_{bit} = 1/Baud rate	Time
9600 bit/s	1 / 9600 = 104.166 µs	11*104.166 µs = of 114.583 ms
6 Mbit/s	1/ $6*10^6$ = of 166.667 ns	11*166.667 ns = 1.833 µs

Table 7: Transmission Time for a Character

1.4.5.1 Calculating the Target Token Rotation Time *Ttr*

The minimum target token rotation time *Ttr* can be estimated as follows:

$$Ttr_{min} = n * (198 + T1 + T2) + b * 11 + 242 + T1 + T2 + Tsl$$

n	:	Number of active slaves
b	:	Number of I/O data byte of the active slaves (input plus output)
T0	:	35 + 2 * Tset + Tqui
T1	:	If T0 < MinTsdr: T1 = MinTsdr
	:	If T0 > MinTsdr: T1 = T0
T2	:	If T0 < MaxTsdr: T2 = MaxTsdr
	:	If T0 > MaxTsdr: T2 = T0
Tsl	:	Maximum Slot Time: maximum time period that the master waits for a slave's response
198	:	Twice a telegram's header with variable length (for request and response)
242	:	Global_Control, FDL_Status_Req and token passing

Note The estimate of the minimum target token rotation time Ttr_{min} is only valid if the following conditions are met:

- Only one master is operating on the bus
- No transmissions are repeated
- There is no acyclic data traffic

To ensure smooth communications, please do not set Ttr_{min} to a value less than that calculated with the above formula. We recommend using a value two or three times greater than the result.

1.4.5.2 Example of Calculating the Target Token Rotation Time *Ttr*

The following configuration is available:

5 active slaves
(n = 5)

20 I/O data bytes per slave
(b = 100)

The following time constants for a transmission rate of 6 Mbit/s are taken from Table 6

MinTsdr	=	**11**	$\mathbf{T_{bit}}$
MaxTsdr	=	**450**	$\mathbf{T_{bit}}$
Tsl bit time	=	**600**	$\mathbf{T_{bit}}$
Tqui bit time	=	**6**	$\mathbf{T_{bit}}$
Tset bit time	=	**8**	$\mathbf{T_{bit}}$

T0 = 35 + 2 * Tset + Tqui
T0 = 35 + 2 * 8 + 6
T0 = 57 $\mathbf{T_{bit}}$

as T0 > MinTsdr: **T1 = 57** $\mathbf{T_{bit}}$
as T0 < MaxTsdr: **T2 = 450** $\mathbf{T_{bit}}$

The computed values are used in the formula for the minimum target token rotation time:

$$\mathbf{Ttr_{min} = n * (198 + T1 + T2) + b * 11 + 242 + T1 + T2 + Tsl}$$

Ttr_{min} = 5 (198+57+450)+100*11+242+57+450+600

Ttr_{min} [T_{bit}] = of 5974 T_{bit}

Ttr_{min} [µs] = 5974 T_{bit} * 166.67 ns = 995.68 µs

Note	*Ttr* is verified when it is entered into the dialog box. If the value *Ttr* set by the user is smaller than the value calculated by the program, an error message appears in the error-status display field. A minimum value for *Ttr* is also suggested. If the *Isochron Mode Sync* or the *Isochron Mode Sync Freeze* is set, the cycle time is defined by the parameter *Min. Slave Interval*. The *Ttr* must be smaller than the *Min. Slave Interval*. If this condition is not met in the isochron mode, an error message appears.

1.5 PROFIBUS DP Slave Context Menu

The context menu for the PROFIBUS DP slave contains the following menu functions:

PROFIBUS DP Slave
Connect Signals
Import GSD file
Insert Modules
User Parameters
Validate
New
Import
Export
Copy
Paste
Delete
Print
Properties

Please refere to chapter 1.4.2 for further information about the standard menu functions Validate, New, Import, Export, Copy, Paste, Delete and Print.

1.5.1 Menu Function *Connect Signals*

Connect Signals opens the *Signal Connections* dialog box in which the system signals *PNO Ident Number* and *Standard Diagnostic* can be connected to the user logic.

Name	Description	Model
PNO Ident Number	16 bit unique number assigned by the PNO Germany (www.profibus.com) to a device and identifying it.	WORD
Standard Diagnostic	With Standard Diagnostic, the slave informs the master about its current state. This signal always contains the last received standard diagnostics. The parameters comply with the diagnostic telegram in accordance with IEC 61158.	DWORD

Table 8: *Signal Connections* Dialog Box

1.5.2 Menu Function *Import GSD File*

Import GSD file opens a standard dialog box for loading a file. This file contains important data for setting the PROFIBUS DP slave's parameters.

Note	Not all GSD parameters are necessary for *ELOP II Factory.* That is why not all GSD parameters appear in *ELOP II Factory.*

The GSD file of the *HIMatrix* PROFIBUS DP slave provides the following modules:

PROFIBUS DP Master Input Modules	**Number**	**Type**
DP Input/ELOP Export	1	Byte
DP Input/ELOP Export	2	Bytes
DP Input/ELOP Export	4	Bytes
DP Input/ELOP Export	8	Bytes
DP Input/ELOP Export	16	Bytes
DP Input/ELOP Export	1	Word
DP Input/ELOP Export	2	Words
DP Input/ELOP Export	4	Words
DP Input/ELOP Export	8	Words
DP Input/ELOP Export	16	Words
PROFIBUS DP Master Output Modules	**Number**	**Type**
DP Output/ELOP Import	1	Byte
DP Output/ELOP Import	2	Bytes
DP Output/ELOP Import	4	Bytes
DP Output/ELOP Import	8	Bytes
DP Output/ELOP Import	16	Bytes
DP Output/ELOP Import	1	Word
DP Output/ELOP Import	2	Words
DP Output/ELOP Import	4	Words
DP Output/ELOP Import	8	Words
DP Output/ELOP Import	16	Words

Table 9: Modules provided by the HIMatrix GSD File `hix100ea.gsd`

See also www.hima.com and www.PROFIBUS.com.

Note	The manufacturer of the field device is responsible for the correctness of the GSD file. GSD files are ASCII files and can be viewed using an ASCII editor.

1.5.3 Menu Function *Insert Modules*

Insert Modules opens the *Insert Modules* dialog box for selecting the PROFIBUS DP Modules used.

The number of bytes that must actually be transferred, must be configured in the PROFIBUS DP master. For this purpose, modules defined in the PROFIBUS DP slave's GDS file, must be selected.

To configure the number of bytes for the PROFIBUS DP master's input and output signals, select enough modules to correspond to the slave's physical configuration.

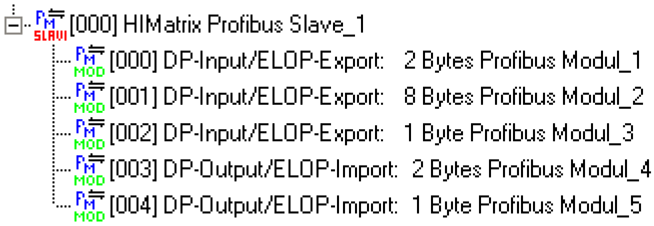


Figure 2: Modules from the *HIMatrix* GSD File suitable for this Example

The index for the PROFIBUS DP modules must be numbered without gaps in ascending order.

The order in which the PROFIBUS DP modules are arranged is not important for operation; however, we recommend organizing the DP input and output modules in an orderly manner to ensure an overview can be maintained.

Note	The number of modules used to achieve the necessary number of bytes is not important as long as the maximum of 32 modules is not exceeded. To avoid unnecessarily complicating the PROFIBUS DP master configuration, we recommend keeping the number of selected modules to a minimum. (This applies to HIMA slaves. For slaves from other manufacturers, please follow the instructions provided in the slave's manual).

1.5.3.1 Connecting Signals in the PROFIBUS DP Modules

Select *Connect Signals* from the context menu to open the *Signals Connection* dialog box for the desired PROFIBUS DP module. The sum of the signals, expressed in bytes, must be equal to the module's size (expressed in bytes).

Connecting Signals in the Input Modules

The signals received by the slave are entered in the *Inputs* tab of the input modules.

Connecting Signals in the Output Modules

The signals that the master sends to the slave are entered in the *Outputs* tab of the output modules.

1.5.4 Menu Function *User Parameters*

User Parameters opens the *User Parameters* dialog box.

The group's **start address** and the **number of signals** (not bytes) are defined in the user data field.

The number of bytes that must actually be transferred, must also be configured in the PROFIBUS DP master. For this purpose, PROFIBUS DP modules defined in the PROFIBUS DP slave's GDS file must be selected (see chapter 1.5.3).

Structure of the 32 byte User Data Field

The 32 byte user data field is structured as follows:

The 32 bytes are allocated in **eight groups**, with **four bytes per group**.

The groups 1 ... 4 define which and how many signals the PROFIBUS DP master receives from the PROFIBUS DP slave.

The groups 5 ... 8 define which and how many signals the PROFIBUS DP master sends to the PROFIBUS DP slave.

The first two bytes of each group specify the start address for the first signal to be read or to be written.

The last two bytes in each group specify the number of signals that should be received or sent.

32 Bytes User Data, allocated in Eight Groups		
Master Import/Slave Export	**Start address**	**Number of signals**
1st group (byte 0 to 3)	0,0	0,0
2nd group (byte 4 to 7)	0,0	0,0
3rd group (byte 8 to 11)	0,0	0,0
4th group (byte 12 to 15)	0,0	0,0
Master Export/Slave Import	**Start address**	**Number of signals**
5th group (byte 16 to 19)	0,0	0,0
6th group (byte 20 to 23)	0,0	0,0
7th group (byte 24 to 27)	0,0	0,0
8th group (byte 28 to 31)	0,0	0,0

Table 10: 32 User Data Bytes of the PROFIBUS DP Master

1.5.5 Configuring the User Data in Different Groups

It is not generally necessary to allocate signals (user data) in to various groups. It is entirely sufficient to define only the first signal group of the input and output signals, and to read or write the data 'en bloc'.

In applications that require that only selected signals are read and written, up to four signal groups for both the input and the output signals can be defined.

Example

The PROFIBUS DP master sends and receives the following signals from the PROFIBUS DP slave:

1st group: **4** input signals from start address **0**.

2nd group: **6** input signals from start address **50**.

4th group: **9** input signals from start address **100**.

5th group: **2** output signals from start address **10**.

User Data Configuration in the PROFIBUS DP Master

32 Bytes User Data, allocated in Eight Groups		
Master Import/Slave Export	**Start address**	**Number of signals**
1st group (byte 0 to 3)	**0,0**	**0,4**
2nd group (byte 4 to 7)	**0,50**	**0,6**
3rd group (byte 8 to 11)	0,0	0,0
4th group (byte 12 to 15)	**0,100**	**0,9**
Master Export/Slave Import	**Start address**	**Number of signals**
5th group (byte 16 to 19)	**0,10**	**0,2**
6th group (byte 20 to 23)	0,0	0,0
7th group (byte 24 to 27)	0,0	0,0
8th group (byte 28 to 31)	0,0	0,0

Table 11: 32 User Data Bytes of the PROFIBUS DP Slave

The structure of the *Edit User Parameters* dialog box depends on the loaded GSD file. The HIMatrix PROFIBUS DP Slave is e.g. more convenient for entering the user parameters (see chapter 3.5.2.3).

Further, the following user parameters must be specified in the *Edit User Parameters* dialog box for the HIMatrix PROFIBUS DP Master.

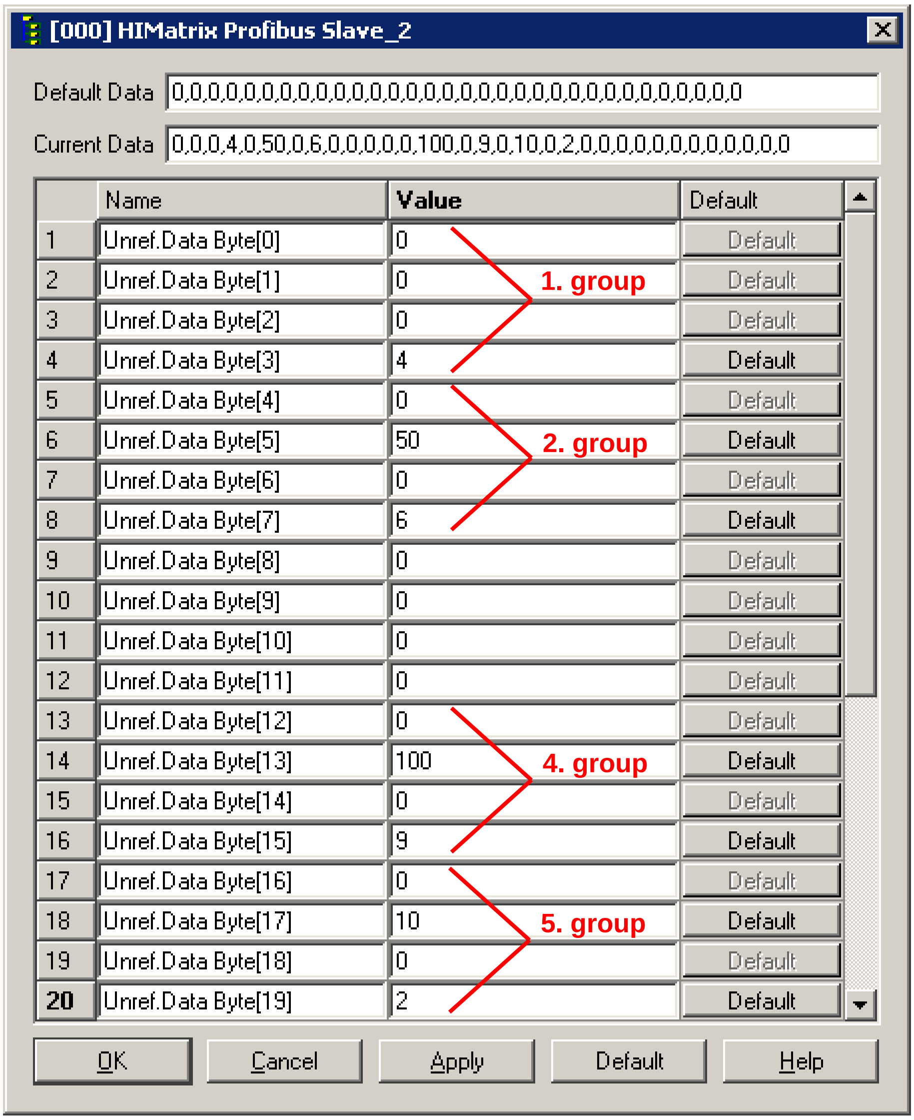


Figure 3: 32 User Data Bytes of the PROFIBUS DP Slave

1.5.6 Menu Function *Validate*

The master's and slave's parameter settings can be checked prior to generating the code. For this purpose, the PROFIBUS DP master must be chosen in the structure tree and *Validate* must be selected from the context menu. Errors and warnings appear in the error-state display field.

Validating is also automatically performed prior to every code generation. If an error is detected during the validation process, the code generation is aborted.

1.5.7 Menu Function *Properties*

Open the *Properties* dialog box.

The dialog box contains the following tabs:

- Parameter
- Groups
- DPV1
- Alarms
- Data
- Model
- Features
- Baud rates
- Acyclic

1.5.7.1 Tab *Parameter*

Element	**Description**	**Value**
Name	Name of the slave	Unique per master
Address	Slave's address of the	Min: 0 Max: 125 Standard: 125
Active	Slave 's state Only an active slave can communicate with a PROFIBUS DP master.	Default: TRUE
DPV0 Sync active	The Sync Mode allows to simultaneously activate the outputs of various DPV0 slaves. Caution! This field must be deactivated in DPV1 slaves operating in the *Isochron Mode Sync*	Default: FALSE
DPV0 Freeze active	The Freeze Mode allows to simultaneously accept the inputs of several DPV0 slaves. Caution! This field must be deactivated in DPV1-slaves operating in the *Isochron Mode Freeze*.	Default: FALSE
Watchdog active	If *Watchdog active* is selected, the master and the HIMatrix slave can recognize and act to a PROFIBUS-DP communication failure (see chapter 1.2 and 3.2). To also enter a watchdog time, first click the *Apply* button.	Default: FALSE
Watchdog Time [ms]	Default value: Slave's watchdog time > 6 * Ttr	Min: 0 Max: 65535 Default: 0
On failure send last data	FALSE: If a failure occurs, the connection is terminated and re-established. TRUE: If a failure occurs, the data continue to be sent, even without the slave's acknowledgement.	Default: FALSE

Auto Clear on failure	If Auto-Clear on failure is set to TRUE in the master and in the current slave, and if the current slave fails, the master switches the entire PROFIBUS DP into the safe state.	Default: FALSE

Table 12: Parameters of the Slave

1.5.7.2 Tab *Groups*

In this tab, the slaves can be organized into various groups. The Global Control commands, Sync and Freeze, can systematically address one or several groups. This is only of practical value in isochron mode, which always addresses group 8.

Element	Description	Value
Member of Group 1	Member of Group 1	Default: FALSE
Member of Group 2	Member of Group 2	
Member of Group 3	Member of Group 3	
Member of Group 4	Member of Group 4	
Member of Group 5	Member of Group 5	
Member of Group 6	Member of Group 6	
Member of Group 7	Member of Group 7	
Member of Group 8	Member of Group 8	

Table 13: Slave's Groups

1.5.7.3 Tab *DPV1*

This tab contains parameters which are only defined in DPV1 and higher. In DPV0 slaves, no parameters can be selected in this tab. The checkmarks in the *Supp* column designate the supported parameters. If parameters are mandatory, the *Required* column contains a checkmark.

Element	Description	Remark
DPV1	If the DPV1 mode is not activated, no DPV1 features can be used. In this case, the slave acts like a DPV0 slave and the parameterization data may have to be changed (refer to the slave's manual)	Default: FALSE
Failsafe	If this mode is activated, a master in CLEAR state does not send zeros as output data; rather, it sends an empty data packet (failsafe data packet) to the slave. In this way, the slave recognizes that it must place the safe output data on the outputs (the value of the safe output data is not necessarily zero).	Default: FALSE
Isochron Mode	This function allows a clock-controlled synchronization in the master and the slaves, irrespective of congestion on the bus. The bus cycle is synchronized with a clock difference of <10 μs. Highly precise positioning processes can be thus implemented.	Default: FALSE
Publisher Active	This function is required for the slave intercommunication. This allows the slaves to communicate with one another in a direct and time saving manner via broadcast without detouring through the master. This field must be activated if the slave should send data to the subscriber slaves as a publisher (see also tab *Features Subscriber*).	Default: FALSE
Prm Block Struct. Supp.	The slave supports structured parameterization data (read only).	Default: FALSE
Check Cfg Mode	Reduced configuration control: if Check Cfg Mode is activated, the slave can operate without the complete configuration. This field should be deactivated while commissioning.	Default: FALSE

Table 14: Slave's *DPV1* Tab in the PROFIBUS DP Master

1.5.7.4 Tab *Alarms*

Alarms can be activated in this tab. This is only possible with DPV1 slaves if DPV1 is activated and the slave supports alarms. The checkmarks in the *Supp* column designate the supported alarms. Mandatory alarms are noted in the *Required* column.

Element	Description	Value
Update Alarm	Alarm, if the module parameters changed.	Default: FALSE
Status Alarm	Alarm, if the module state changed.	
Vendor Alarm	Vendor specific alarm.	
Diagnosis Alarm	Alarm, if specific events such as short circuits, over temperature, etc. occur within a module.	
Process Alarm	Alarm, if important events occur in the process.	
Pull & Plug Alarm	Alarm, if a device is pulled or plugged in.	

Table 15: Slave's *Alarms* Tab

1.5.7.5 Tab *Data*

This tab specifies details about the supported data lengths and about the user data (extended parameterization data).

Element	Description	Value
Max. Input Len [Byte]	Maximum length of the input dates.	Display only
Max. Output Len [Byte]	Maximum length of the output data.	Display only
Max. Data Len [Byte]	Maximum total length of the input and output data.	Display only
User Prmdata Len [Byte]	Length of the user data.	Display only
User Prmdata	Extended parameterization data sent to the slave. We do not recommend editing here as it can be performed more comfortably with the *Change User parameter* command.	User size [Byte]
Max. Diag. Data Len	Maximum length of the diagnostic data sent by the slave.	Display only

Table 16: Slave's *Data* Tab

1.5.7.6 Tab *Model*

This tab displays self-explanatory details.

Element	Description	Value
Model	Manufacturer identification of the PROFIBUS DP slave	Display only
Manufacturer	Manufacturer of the field device	
Ident Number	Slave identification provided by PNO Germany (www.profibus.com)	
Revision	Release of the PROFIBUS DP slave	
Hardware release	Hardware release of the PROFIBUS DP slave	
Software release	Software release of the PROFIBUS DP slave	
GSD file name	File name of the GSD file	
Info Text	Additional details about the PROFIBUS DP slave	

Table 17: Slave's *Model* Tab

1.5.7.7 Tab *Features*

Element	Description	Remark
Modular Station	TRUE: Modular station. FALSE: Compact station	Display only
First slot number	The modules (slots) must be numbered without gaps, starting with this value.	
Max Modules	Maximum number of modules that can be installed in a modular station.	
Support for ‚Set Slave Add'	The slave supports dynamic address allocation.	
Min. Slave Interval [ms]	The minimum time period that must elapse between two cyclic calls of the slave.	
Diag. Update	Number of polling cycles that may pass before the slave's diagnostics reflect the current state.	
WDBase1ms Support	The slave supports 1 ms as a time base for the watchdogs	
DPV0 Sync Support	The slave supports DPV0 Sync.	
DPV0 Freeze	The slave supports DPV0 Freeze	
DPV1 Data Types	The slave supports the DPV1 data types.	

Extra Alarm SAP	The slave supports SAP 50 for acknowledging the alarm.	
Alarm Seq. Mode Count	Indicate how many active alarms the slave can simultaneously process. Zero means *one alarm of each model.*	

Table 18: Slave's *Features* Tab in the HIMatrix Master

1.5.7.8 Tab *Baud rates*

This tab specifies the *baud rates* that the slave supports and the corresponding *MaxTsdr*.

MaxTsdr is the time within which the slave must acknowledge a request from the master. The range of values depends on the slave and the transfer rate, and lies between 15 and 800 T_{bit}.

Element	Description	Value
9.6k	MaxTsdr = 15	Display only
19.2k	MaxTsdr = 15	
31.25k	Is not supported	
45.45k	Is not supported	
93.75k	MaxTsdr = 15	
187.5k	MaxTsdr = 15	
500k	MaxTsdr = 15	
1.5M	MaxTsdr = 25	
3M	MaxTsdr = 50	
6M	MaxTsdr = 100	
12M	MaxTsdr = 200	

Table 19: Slave's *Baud rate* Tab in the HIMatrix Master

1.5.7.9 Tab *Acyclic*

This tab contains some parameters for the acyclic data transfer.

Element	Description	Value
C1 Read/Write Support	The slave supports the acyclic data transfer.	Display only
C1 Read/Write required	The slave requires a cyclic data transfer.	
C1 Max Data Len[Byte]	Maximum length of an acyclic data packet.	
C1 Response Timeout [ms]	Time out for the acyclic data transfer.	

Table 20: Slave's *Acyclic* Tab in the HIMatrix Master

1.6 Diagnostics and Protocol States of the HIMatrix PROFIBUS DP Master

The user can verify the master's and the slaves' settings in the *PROFIBUSMs.* tab. The tab also contains details about the master's and the slaves' current status (e.g. cycle time).

Open the Hardware Management's control panel and select the *PROFIBUSMs.* tab.
The *PROFIBUSMs.* tab is divided into the *PB Master* and *PB Slave* areas.

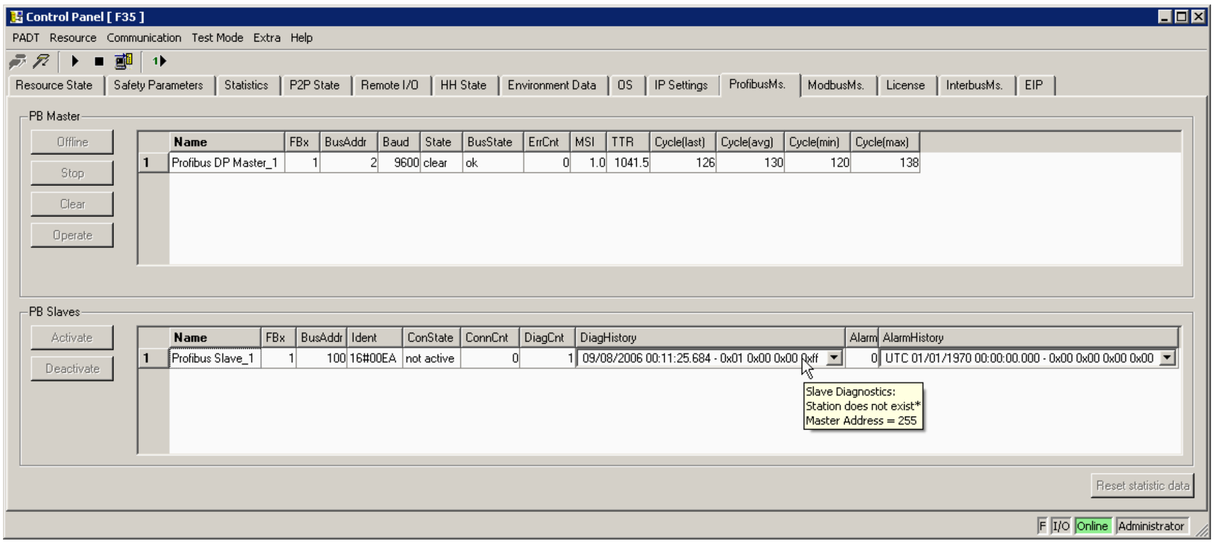


Figure 4: PROFIBUS DP in the Control-Panel

1.6.1 Reset statistic data

The **Reset statistic data** button resets the statistical data (cycle [min], cycle [max] …) to zero.

1.6.2 PB Master

In the *PB Master* area, the user can verify the settings and control the master. Further, details about the master's current status (e.g. cycle time) appear.

Four buttons and a display field are available.

1.6.2.1 Buttons

Using the buttons, the following commands can be executed for one or several selected PROFIBUS DP masters:

Offline:
Switch off the selected PROFIBUS DP master. If the master is switched off, it cannot perform any actions.

Stop:
Stop the selected PROFIBUS DP master. The PROFIBUS DP master still participates in the token protocol, but it does not send any data to the slaves.

Clear:

Activating the *CLEAR* button, the selected PROFIBUS DP master assumes a safe state and exchanges safe data with the slaves. The output data transmitted to the slaves, only contain zeros.
Failsafe slaves receive failsafe telegrams that do not contain any data.

The PROFIBUS DP master ignores the input data from the slaves and instead uses initial values within the application program.

Operate:

Start the selected PROFIBUS DP master. The PROFIBUS DP master exchanges cyclical I/O data with the slaves.

1.6.2.2 Display Field

The display field contains the states of the PROFIBUS DP masters connected to the bus.

Element	Description
Name	Name of the PROFIBUS DP Master
FBx	Assigned field bus interface (1, 2)
BusAddr	Master's bus address (0 to 125)
Baud	Master's baud rate The master can communicate using all baud rates specified in the standard. Cycle times can be set up to a lower limit of 2 ms.
State	Indicates the current protocol state. 0: OFFLINE 1: STOP 2: CLEAR 3: OPERATE 7: UNDEFINED
BusState	Bus error code: 0: OK 1: Address Error The address of the master is already available on the bus. 2: Bus malfunction A malfunction was detected on the bus, (e.g. bus was not properly terminated, and several stations are sending data simultaneously). 3: Protocol error An incorrect coded packet has been received. 4: Hardware error The hardware reported an error, e.g. too short time periods. 5: Unknown error The master changed the status for an unknown reason. 6: Controller Reset The controller chip is reset if a serious bus error occurs. The error code remains set until the bus error has been eliminated.

Table 21: Display Field

Element	Description
ErrCnt	Number of the bus error, so far
MSI	Min. Slave Interval in ms, resolution 0.1ms
Ttr	Target Token Rotation Time in ms, resolution 0.1ms
Cycle[last]	Last PROFIBUS DP cycle time [ms]
Cycle [avg]	Average PROFIBUS DP cycle time [ms]
Cycle [min]	Minimum PROFIBUS DP cycle time [ms]
Cycle [max]	Maximum PROFIBUS DP cycle time [ms]

Table 22: Display Field of *PB Master*

1.6.3 *PB Slave*

In the *PB Slave* area, the user can verify the settings and control the slaves. Further, current slave's diagnostics and alarms appear.

Two buttons and a display field are available.

1.6.3.1 Buttons

Using the following buttons, the selected slaves can be activated or deactivated.

Activate:

Activate the selected slave which can now exchange data with a PROFIBUS DP master.

Deactivate:

Deactivate the selected slave.

The communication is closed and initial data are sent to the application program.

1.6.3.2 Display Field

The display field contains the states of the PROFIBUS DP slaves connected to the bus.

Element	Description
Name	Name of the PROFIBUS DP Slave
FBx	Assigned slave's field bus interface
BusAddr	Slave's bus address
Ident	PNO identification number
ConState	Connection status 0: Deactivated, 1: Inactive (connection attempt) 2: Connected
ConnCnt	Number of previous connections.
DiagCnt	Number of previous diagnostic messages.
DiagHistory	First four bytes of the last diagnostic message with time stamp. A popup window opens if the mouse is positioned on the field.
AlarmCnt	Number of previous alarms.
AlarmHistory	First four bytes of the last alarm message with time stamp. A popup window opens if the mouse is positioned on the field.

Table 23: : Slave's Display Field

1.6.4 Protocol States of the HIMatrix PROFIBUS DP Master

The protocol state is displayed in the control panel (see Table 22) and can be assessed with the status signal *Master State* (see Table 1) within the application program.

Master State	Description
OFFLINE	The master is switched off; no bus activity.
STOP	The master participates in the token protocol but does not send any data to the slaves.
CLEAR	The master is in the safe state and exchanges safe data with the slaves. • The output data sent to the slaves only contain zeros. • The Failsafe slaves receive Failsafe telegrams that contain no data. • The slaves' input data are ignored and instead initial values are used.
OPERATE	The master is in the operating mode and cyclically exchanges I/O data with the slaves.

Table 24: Protocol States of the HIMatrix PROFIBUS DP Master

1.6.5 Behavior of the HIMatrix PROFIBUS DP Master

Behavior of the HIMatrix PROFIBUS DP master, depending on the controller operating state.

Controller State	Behavior of the HIMatrix PROFIBUS DP Master
STOP *)	If the controller is in the STOP state, the master is in the OFFLINE state.
RUN	If the controller is in the RUN state, the master tries to assume the OPERATE state.
STOP	If the controller assumes the STOP state, the master assumes the CLEAR state. If the master is already in the STOP or OFFLINE state, it remains in this state.

Table 25: Behavior of the HIMatrix PROFIBUS DP Master

*) After powering up the controller or loading the configuration

1.6.6 FBx LED Function in the PROFIBUS Master

The state of the serial PROFIBUS DP communication is displayed with the FBx LED on the corresponding configured serial interfaces (fb1, fb2).

FBx LED	Description
OFF	The PROFIBUS DP master is not configured or its configuration is invalid.
Blinking every 2 seconds	The PROFIBUS DP master is in the OFFLINE or STOP state.
ON	The PROFIBUS DP master is in the OPERATE or CLEAR state and exchanges data with the activated slaves.
Blinking, every second	At least one slave failed.

Table 26: FBx LED Function in the PROFIBUS Master

1.7 Example: Configuring the HIMatrix PROFIBUS DP Master

In the following example, the HIMatrix controller operates as PROFIBUS DP master and communicates with a Siemens ET 200M operating as PROFIBUS DP slave.

The HIMatrix controller reads 2 bytes and sends 1 byte.

In this example (using the Siemens GSD file), the PROFIBUS DP signal connections are completely configured within the HIMatrix controller.
The following parameters must be set in the ET 200M (e.g. via DIP switch block): Bus address = 3.

Note	The configuration of the Siemens controller described above is not exhaustive. This information is provided without guarantee (errors and omissions excepted); please refer to the Siemens documentation when developing projects with the Siemens ET 200M.

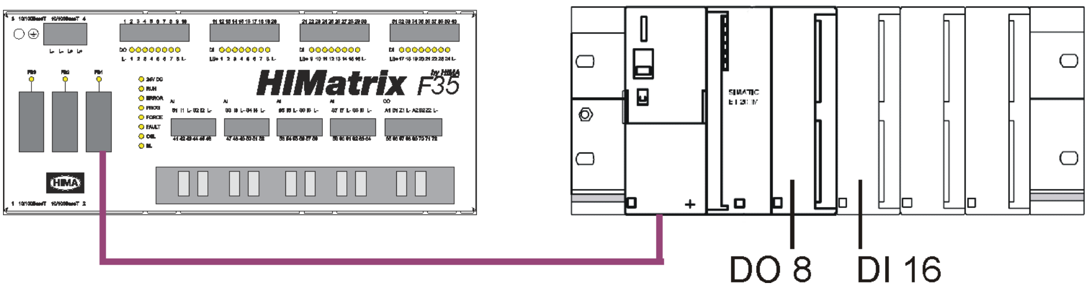


Figure 5: A HIMatrix (Master) with a Siemens ET 200M (Slave)

Start **ELOP II Factory** and create a new project or load an existing project.

Step 1: Creating a HIMatrix PROFIBUS DP master:

- Select **Protocols** in the resource's structure tree.
- Select **New, PROFIBUS Master** from the context menu.

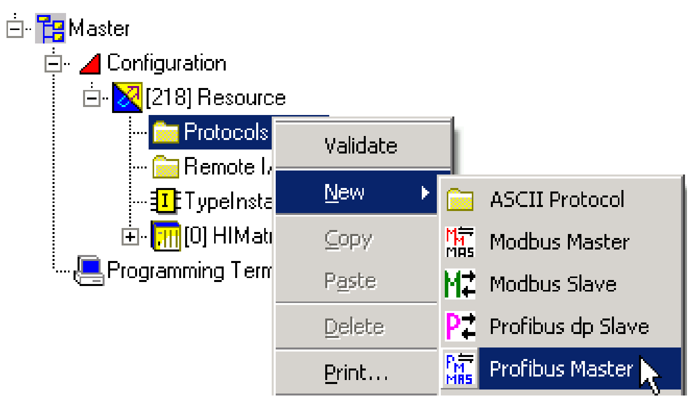


Figure 6: Creating a PROFIBUS DP Master

Step 2: Adding a PROFIBUS DP slave to the PROFIBUS DP master:

- Open the **Protocols** directory in the structure tree.
- Right-click on **PROFIBUS DP Master** and select **New, PROFIBUS Slave** from the context menu.

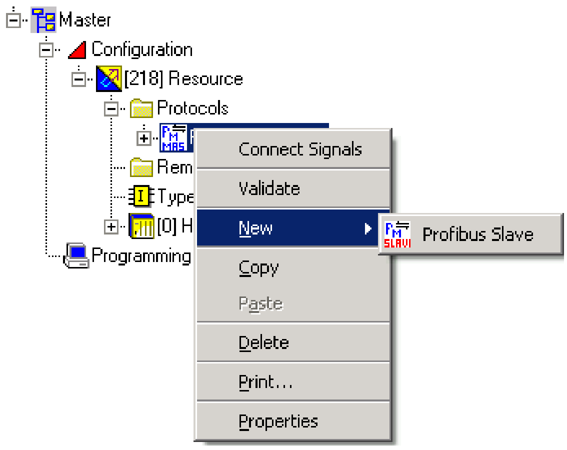


Figure 7: Creating a PROFIBUS DP Slave

Step 3: Configuring the PROFIBUS DP slave:

- Open the context menu for the *PROFIBUS DP slave*.
- Select **Import GSD File** from the slave's context menu.
- In the Windows standard dialog box for opening a file, select the GSD file for the compact device (e.g. for the ET 200M `si02801e.gsg`)

Step 4: Entering the address of the PROFIBUS DP slave:

- Open the slave's context menu and select **Properties**.
- Open the **Parameter** tab.
- Enter the PROFIBUS address of the Slave **1** in the *Address* input field.

Step 5: Creating the following modules:

- Select **Insert Module** from the slave's context menu.
- Select the module **Config for Slot1** (standard for the ET200M) in the *Insert Module* dialog box.
- Select the module **Config for Slot2** (standard for the ET200M) in the *Insert Module* dialog box.
- Select the module **Config for Slot3** (standard for the ET200M) in the *Insert Module* dialog box.
- Select the module **6ES7 322-8BF00-0AB0 8DO** in the *Insert Module* dialog box.
- Select the module **6ES7 321-7BH01-0AB0 16DI** in the *Insert Module* dialog box.

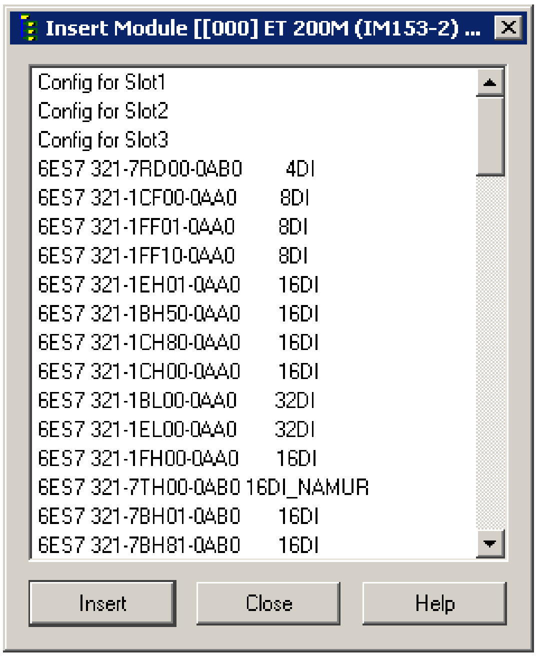


Figure 8: *Insert Module* Dialog Box"

Step 6: Numbering the modules:

- Open the **Properties** dialog box for the first module.
- Enter **1** into the field *Slot*.
- Repeat this action with the next modules and assign continuous slot numbers.

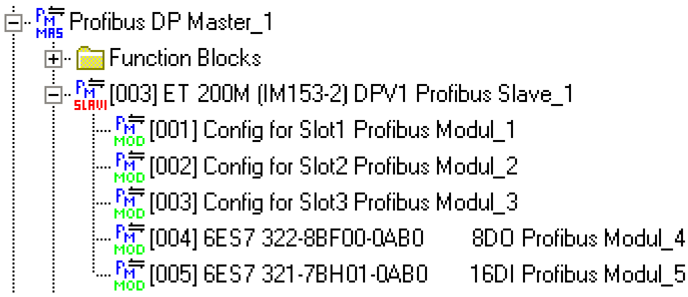


Figure 9: The PROFIBUS DP Modules for the Compact Device *Siemens ET 200M*

Note	The slot numbers of the PROFIBUS DP modules must be numbered continuously.

Step 7: Changing the slot number of *8DO PROFIBUS Modul_4*:

- Open the context menu for *8DO TO PROFIBUS Modul_4*.
- Select Edit User Parameters.
- Enter **4** into the input field *SlotNumber*.

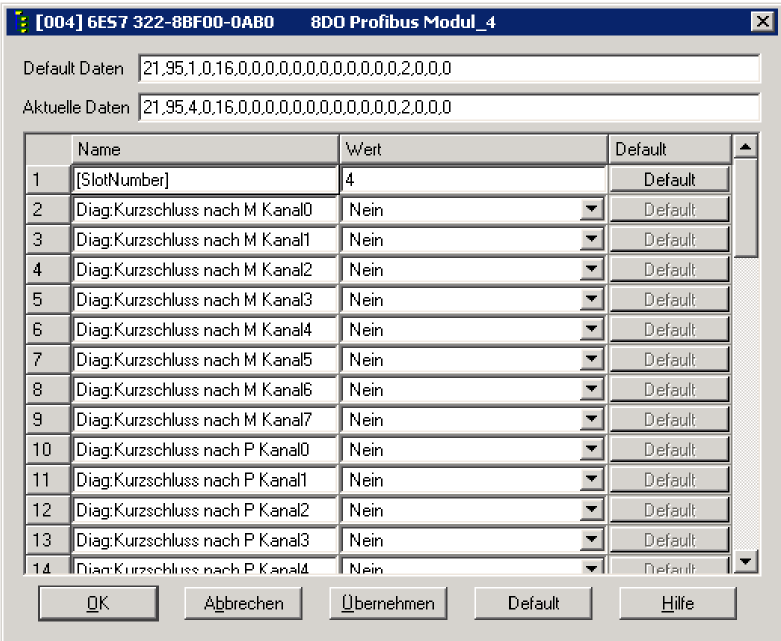


Figure 10: User Parameters of 8DO PROFIBUS Modul_4

Step 8: Changing the slot number of *16DI PROFIBUS Modul_5*:

- Open the context menu for *16DI PROFIBUS Modul_5*.
- Select Edit User Parameters.
- Enter **5** into the input field *SlotNumber*.

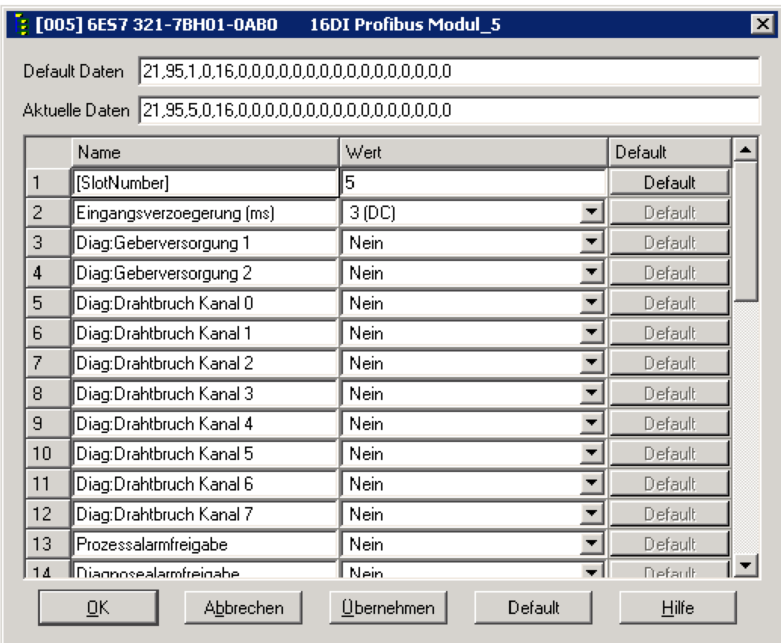


Figure 11: User Parameters of 16DI PROFIBUS Modul_5

Step 9: Creating the following signals in the signal editor:

- Open the *Signal editor* using the main menu **Signals, Editor**.
- Create the signals **ET200M_F35_1** and **ET200M_F35_2** of type Byte.
- Create the signal **F35_ET200M_1** of type Byte.

Step 10: Connecting the HIMatrix input signals to the signals of the slave *ET 200M*:

- Open the context menu for the PROFIBUS DP module *16DI PROFIBUS Modul_5*.
- Select **Connect Signals** from the context menu.
- Open the **Inputs** tab located in the *Signal Connections* dialog box.
- Open the *Signal editor* using the main menu **Signals, Editor**.
- Use drag&drop to copy the signals **ET200M_F35_1** and **ET200M_F35_2** from the *Signal editor* into the corresponding **Input signal** located in the *Inputs* tab of the *Signal Connections* dialog box.
- Click the **New offsets** button in the *Signal Connections* dialog box.
- Click the **Renumber** button in the *Renumber Offsets* dialog box.
- Click **OK** to close the dialog box.

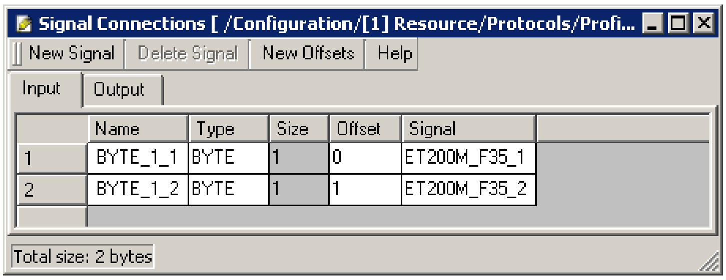


Figure 12: Input Signal of the HIMatrix Controller

Step 11: Connecting the HIMatrix output signals to the signals of the slave *ET 200M*:

- Open the context menu for the PROFIBUS module *8DO PROFIBUS Modul_4*.
- Select **Connect Signals** from the context menu.
- Open the **Ouputs** tab located in the *Signal Connections* dialog box.
- Open the *Signal editor* using the main menu **Signals, Editor**.
- Use drag&drop to copy the signal **F35_ET200M_1** from the *Signal editor* into the corresponding **Output signal** located in the *Outputs* tab of the *Signal Connections* dialog box.
- Click the **New offsets** button in the *Signal Connections* dialog box.
- Click the **Renumber** button in the *Renumber Offsets* dialog box.
- Click **OK** to close the dialog box.

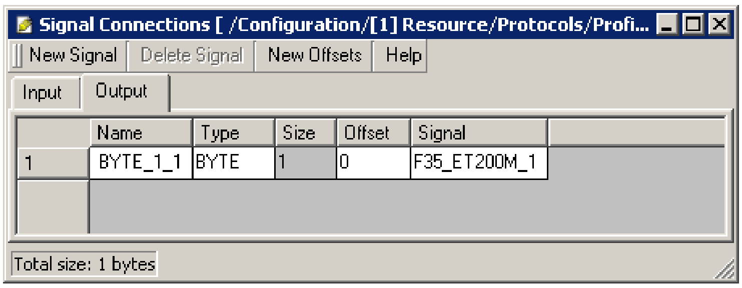


Figure 13: Output Signal of the HIMatrix Controller

Step 12: Determining the minimum target token rotation time

- Open the *Transfer rates* tab located in the **Properties** dialog box.
- Compare MaxTsdr from Table 6 with MaxTsdr from the Trasfer rates tab. To continue the calculation, use the highest of the two values for MaxTsdr.

The following time constants for a transfer rate of 500 kbit/s are taken from Table 6.

MinTsdr	=	**11**	$\mathbf{T_{bit}}$
MaxTsdr	=	**100**	$\mathbf{T_{bit}}$
Tsl	=	**200**	$\mathbf{T_{bit}}$
Tqui	=	**0**	$\mathbf{T_{bit}}$
Tset	=	**1**	$\mathbf{T_{bit}}$

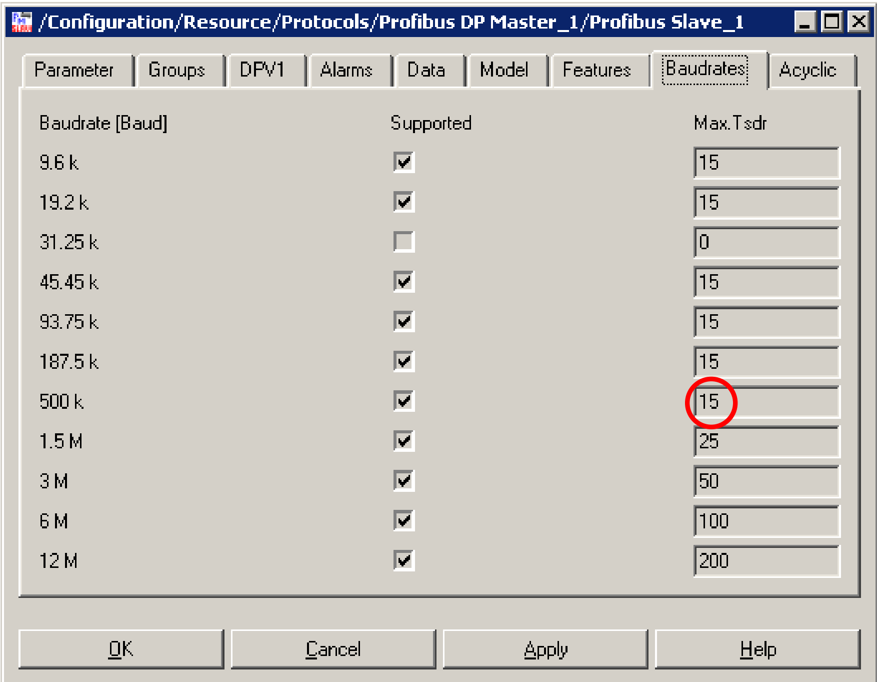


Figure 14: *MaxTsdr* for the Slave *Siemens ET 200M*

Calculation of the target token rotation time Ttr (see also chapter 1.4.5.1)

T0 = 35 + 2 * Tset + Tqui

T0 = 35 + 2 * 1 + 0

T0 = 37 $\mathbf{T_{bit}}$

As T0>MinTsdr: **T1 = 37 $\mathbf{T_{bit}}$**

As T0<MaxTsdr: **T2 = 37 $\mathbf{T_{bit}}$**

The computed values are used in the formula for the minimum target token rotation time:

Ttr_{min} = n * (198 + T1 + T2) + b * 11 + 242 + T1 + T2 + Tsl

In this example, one slave is active **(n = 1)**

3 I/O-data bytes are transferred by the slave (input and output) **(b = 3)**

Ttr_{min} = 1 (198 + 37 + 100)+ 3 * 11 + 242 + 37 + 37 + 200

Ttr_{min} [T_{bit}] = 884 T_{bit}

Ttr_{min} [µs] = 884 T_{bit} * (1/500 kBit) = 1.768 ms

Note	The estimate of the minimum target token rotation time Ttr_{min} is only valid if the following conditions are met: • Only one master is operating on the bus • No transmissions are repeated • There is no acyclic data traffic To ensure smooth communications, please do not set Ttr_{min} to a value less than that calculated with the above formula.

We recommend using a value two or three times greater than the computed target token rotation timeTtr_{min}.

Ttr = Ttr_{min} * 3

Ttr = 1768 ms * 3 = 5.304 ms

Ttr -> 6 ms

Step 13: Determining the data control time and the watchdog time:

Note	The data control time runs on the PROFIBUS DP master and the watchdog time runs on the PROFIBUS DP slave. Steps are only possible in the raster of 10 ms. We recommend setting both times to the same value.

- ❑ Determine the data control time using **Ttr**
  Data control time 6 * Ttr -> **40 ms**

- ❑ Determine the watchdog time of the slave using **Ttr**
  Watchdog time of the slave >= 6 * Ttr -> **40 ms**

Step 14: Setting the parameters for the PROFIBUS DP slave as specified in the following figure:

- Select **Properties** from the slave's context menu.
- Select the **Parameter** tab located in the *Properties* dialog box.
- Check the checkbox **Active** to activate the slave.
- Enter the value for the *Watchdog Time [ms]* from Step 13.

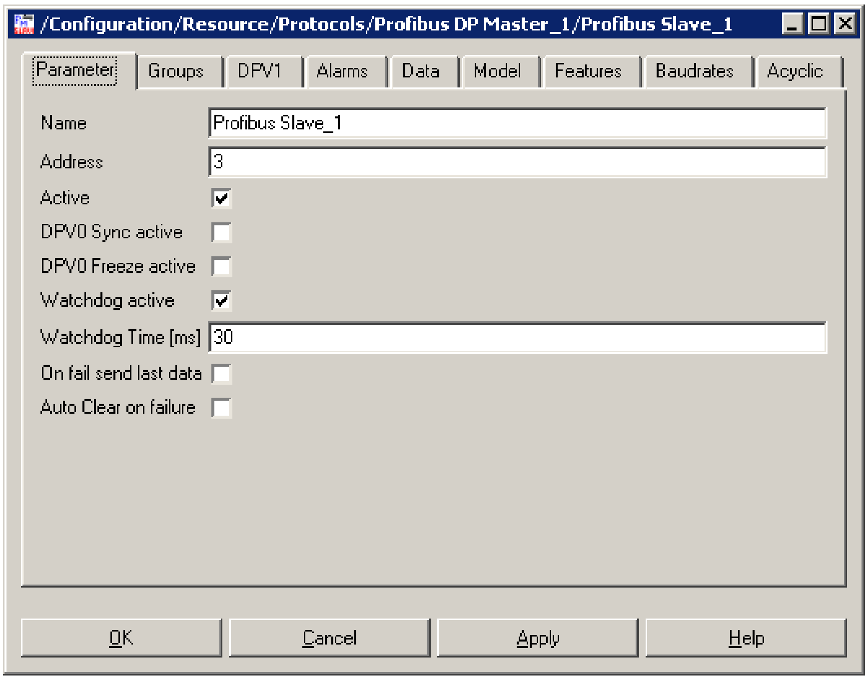


Figure 15: *Parameter* Tab of the Slave *Siemens ET 200M*

Note	The slave recognizes a master's failure if no data have been exchanged with the master within the watchdog time frame. If a master fails, the slave assumes the STOP state. The PROFIBUS DP slave watchdog time frame is independent of the watch-dog time frame detailing the greatest amount of time a resource may require to complete a program cycle. The following basic condition applies to the watchdog time Slave's watchdog time > 6 * Ttr.

Step 15: Determining the parameter *Min. Slave Interval [ms]*

- Select **Properties** from the slave's context menu.
- Select the **Features** tab.
- Note the value of *Min. Slave Interval [ms]* for Step 17.

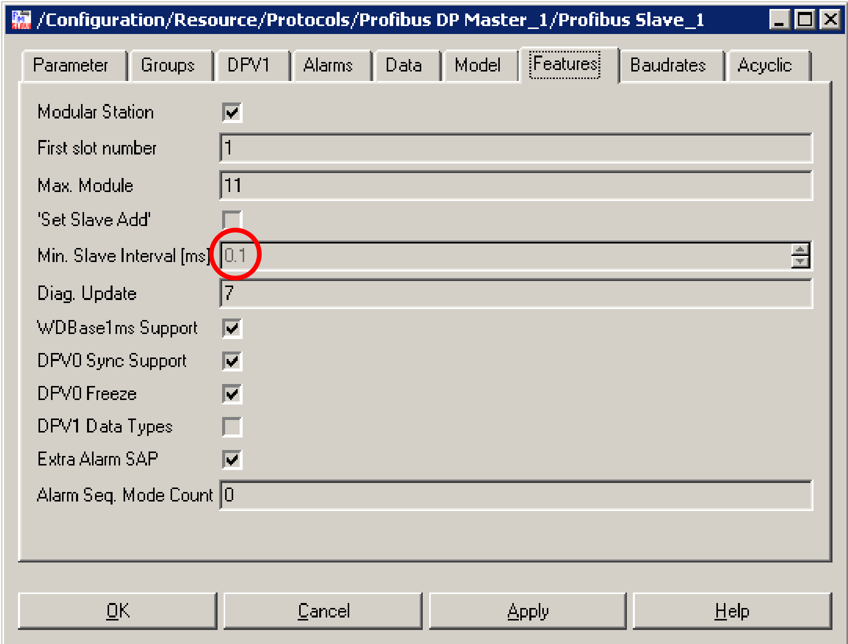


Figure 16: *Features* Tab of the Slave *Siemens ET 200M*

Step 16: Configuring the HIMatrix PROFIBUS DP Master:

- Open the context menu for the PROFIBUS DP master and select *Properties*.
- Select the **General** tab located in the *Properties* dialog box and configure the PROFIBUS DP master as specified in the following figure:

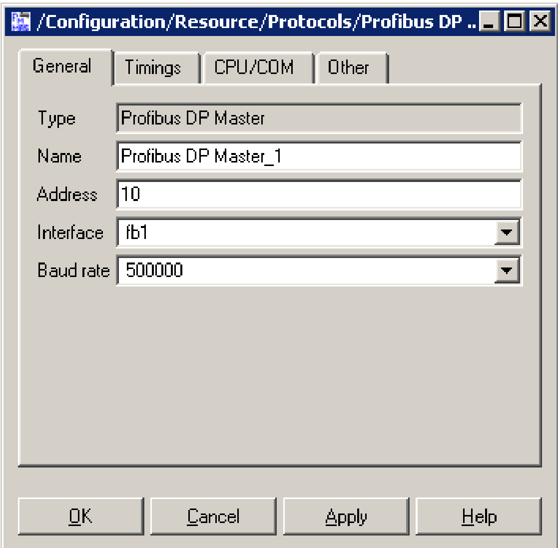


Figure 17: *General* Tab of the PROFIBUS DP Master

Step 17: Configuring the HIMatrix PROFIBUS DP master as specified in the following figure:

- Select the **Timings** tab located in the *Properties* dialog box.
- Enter the value for *Min. Slave Interval [ms]* from Step 15.
- Enter the value for the *Data Control time [ms]* from Step 13.

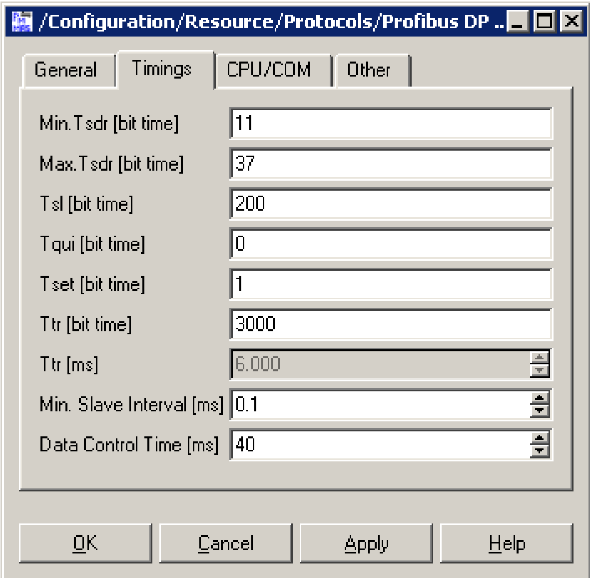


Figure 18: *Timings* Tab of the PROFIBUS DP Master

Note	If various slaves are configured, the highest values of the parameters *MaxTsdr [bit time]* and *Min. Slave Interval [ms]* are used.

Step 18: Configuring the other parameters of the HIMatrix PROFIBUS -DP-master as specified in the following figure:

- Select the **Other** tab located in the *Properties* dialog box.

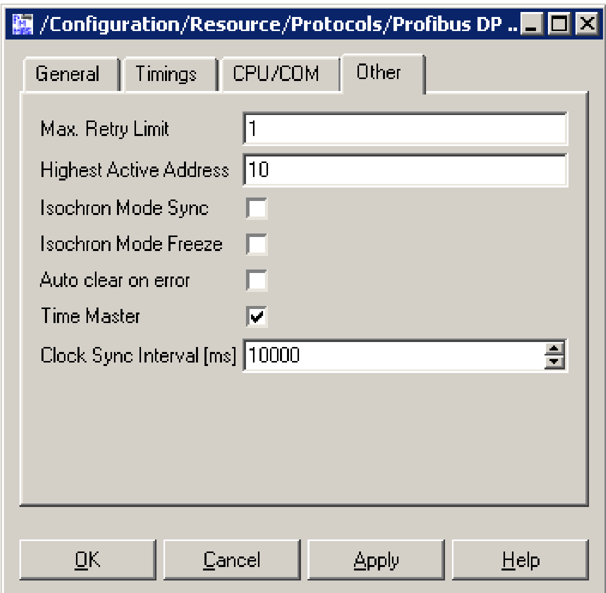


Figure 19: *Other* Tab of the PROFIBUS DP Master

Step 19: Verifying the PROFIBUS DP master's configuration:

- Open the context menu for the PROFIBUS DP masters and select **Validate**.

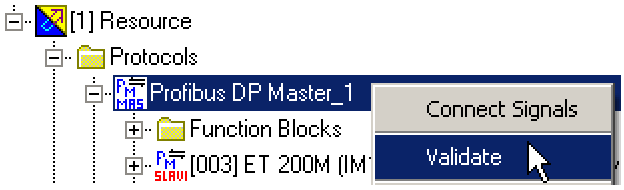


Figure 20: Verifying the PROFIBUS DP Master's Configuration

After validating the PRORFIBUS-DP master's configuration, potential errors and warnings appear in the error-state display field.

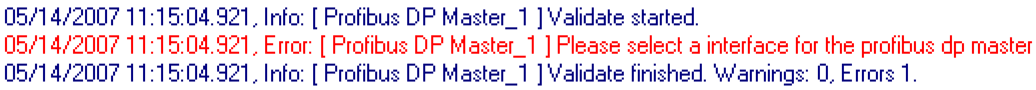


Figure 21: Error-State Display Field

2 HIMatrix PROFIBUS DP Function Blocks

Using the HIMatrix PROFIBUS DP function blocks, the user can adapt the HIMatrix PROFIBUS DP master and the PROFIBUS DP slaves assigned to it by tailoring them to best meet the project requirements.

The function blocks' parameters are set in the application program, such that the master and slave functions (alarms, diagnostic data, and states) can be set and read in the application program.

Note	Function blocks are required for special applications. They are not needed for the normal cyclic data traffic between master and slave!

The following function blocks are available:

Function Block	Function Description	Suitable beginning with Stage of Extension
MSTAT	Controlling the master's state via the application program	DP-V0
RALRM	Reading the slaves' alarm messages	DP-V1
RDIAG	Reading the slaves' diagnostic messages	DP-V0
RDREC	Reading the slaves' acyclic data records	DP-V1
SLACT	Controlling the slaves' state via the application program	DP-V0
WRREC	Writing the slaves' acyclic data records	DP-V1

Table 27: Stages of Extension Required for the Function Blocks within the PROFIBUS DP Slaves

Note	HIMatrix PROFIBUS DP masters operate with the stage of extension DP-V2. HIMatrix PROFIBUS DP slaves operate with the stage of extension DP-V0. Please note that for this reason, not all function blocks can be used with HIMA slaves.

2.1 Configuration of the Function Blocks

The PROFIBUS DP protocol and thus the HIMatrix PROFIBUS DP function blocks run on the controller's communication processor.

As a result, the PROFIBUS DP function blocks must be created within the ELOP II Factory's Hardware Management.

To control these function blocks with the application program, function blocks are created within the ELOP II Factory Project Management (see chapter 2.1.1). These can be used as standard function blocks in the FBD editor.

The function blocks in Project Management are connected to the corresponding function blocks in Hardware Management using common signals. These must be created in advance using the signal editor.

Convention:

Function block = Function block in the application program (Project Management)

Function block (HWM) = Function block in Hardware Management

2.1.1 PROFIBUS DP Function Block Library

The function block library *PBM_V...* must be added to the project using the function *Restore Project...* (Project's context menu).

The function block libraries are available on request from HIMA support.

Tel.: +49-(0)6202-709 185 or -259 / -261

E-mail: support@hima.com

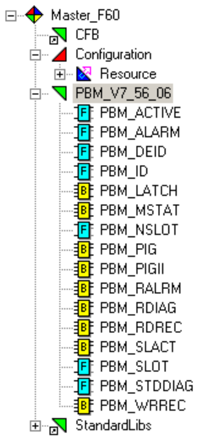


Figure 22: Function Blocks and Auxiliary Function Blocks

2.1.2 Configuring the Function Blocks in the Application Program

The required function blocks can be copied in the application program using drag&drop. Please configure the inputs and outputs as described for the individual function blocks (from chapter 2.2 on).

Upper Part of the Function Block

The upper part of the function block corresponds to the user interface that the application program uses for controlling it.

The signals used in the application program are connected here. The *prefix A* means "Application".

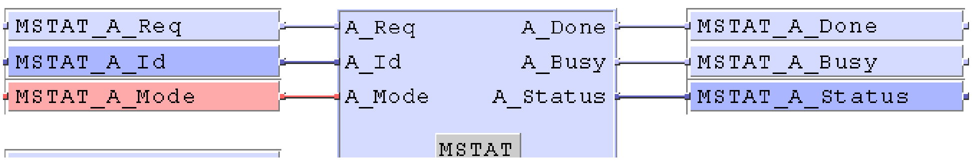


Figure 23: Upper Part of the function Block

Lower Part of the Function Block

The function block's lower part represents the connection to the function block in Hardware Management (HWM).

The signals that must be connected to the function block in Hardware Management are connected here. The *prefix F* means Field.

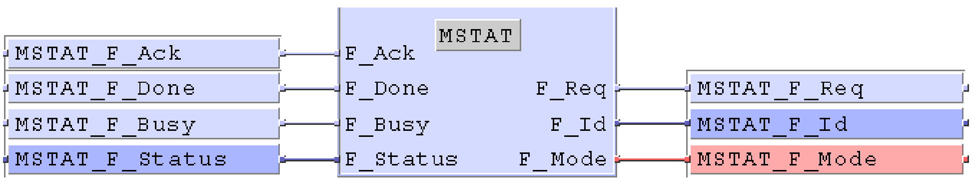


Figure 24: Lower Part of the Function Block

2.1.3 Configuring the Function Blocks within the Hardware Management

Selecting **Function Blocks, New** in the Hardware Management's structure tree, all the function blocks (HWM) of the PROFIBUS DP communication appear.

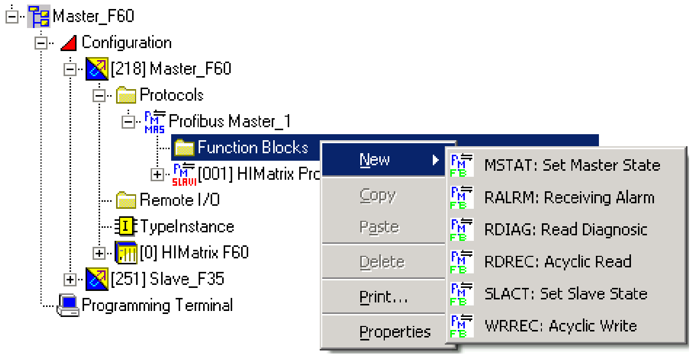


Figure 25: Function blocks within the Hardware Management (HWM)

Configuring the function block in Hardware Management:

- ❑ Open the resource's directory in the structure tree.
- ❑ Select **Protocols, PROFIBUS Master, Function Blocks, New** to choose the correct function block (HWM).
- ❑ Open the context menu for the function block (HWM) and select **Edit** to open the *Signal Connections* dialog box.

The inputs of the function block (HWM) must be connected to the same signals that are connected in the application program to the function block's *F_Outputs*.

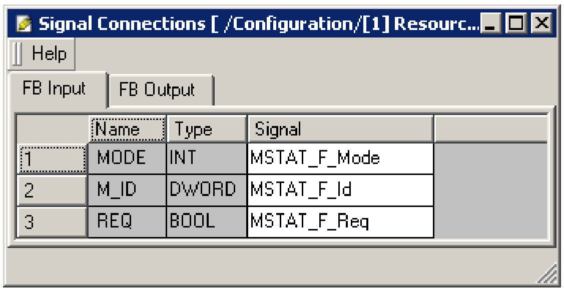


Figure 26: Inputs of the Function Block (HWM)

The outputs of the function block (HWM) must be connected to the same signals that are connected in the application program to the function block's *F_Inputs*.

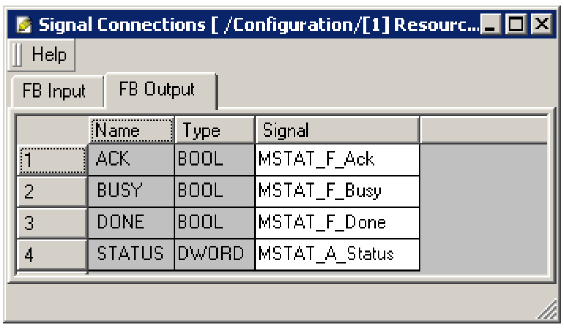


Figure 27: Outputs of the Function Block (HWM)

2.2 MSTAT Function Block

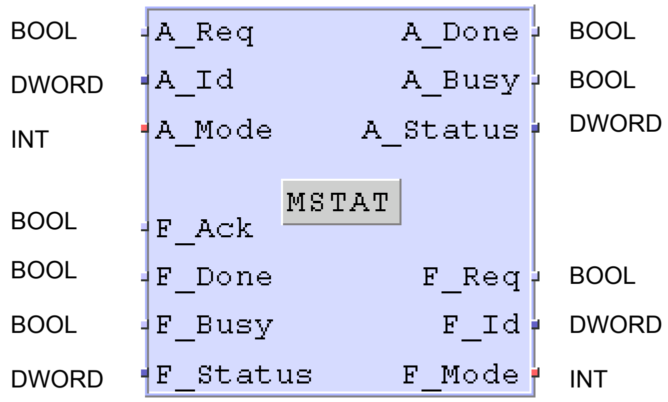


Figure 28: MSTAT Function Block

The application program can control the PROFIBUS DP master using the function block *MSTAT* (DPV0 and higher). The PROFIBUS DP master can thus be set to one of the following states using a timer or a switch connected to a physical input.

0	OFF-LINE
1	STOP
2	CLEAR
3	OPERATE

To configure the function block, use drag&drop to copy it from the function block library to the application program.

2.2.1 Inputs and Outputs of the Function Block with the *Prefix A*

The inputs and outputs with the *prefix A* correspond to the user interface that the application program uses for controlling the function block.

A_Inputs	**Description**	**Type**
A_Req	Rising edge starts the function block	BOOL
A_Id	Master Id (not used)	DWORD
A_Mode	The PROFIBUS DP master can be set to the following states 0: OFF-LINE 1: STOP 2: CLEAR 3: OPERATE	INT

Table 28: *A_Inputs* of the *MSTAT* Function Block

A_Outputs	Description	Type
A_Done	TRUE: The PROFIBUS DP master has been set to the state defined on the *A_Mode* input.	BOOL
A_Busy	TRUE: The setting process for the PROFIBUS DP master is still running.	BOOL
A_Status	State or error code (See 2.8 Error Codes of the Function Blocks).	DWORD

Table 29: *A_Outputs* of the *MSTAT* Function Block

2.2.2 Inputs and Outputs of the Function Block with the *Prefix F*

The inputs and outputs of the function block that have the *prefix F* are used for connecting to the *MSTAT* function block in Hardware Management.

Note The function blocks in Project Management are connected to the corresponding function blocks in Hardware Management using common signals. These must be created in advance using the signal editor

The *MSTAT* function block's *F_Inputs* in the application program must be connected to the same signals that are connected in Hardware Management to the *MSTAT* function block's *F_Outputs*.

F_Inputs	Type
F_ACK	BOOL
F_DONE	BOOL
F_BUSY	BOOL
F_STATUS	DWORD

Table 30: *F_Inputs* of the *MSTAT* Function Block

The *MSTAT* function block's *F_Outputs* in the application program must be connected to the same signals that are connected in Hardware Management to the *MSTAT* function block's inputs.

F_Outputs	Type
F_REQ	BOOL
F_ID	DWORD
F_MODE	INT

Table 31: *F_Outputs* of the *MSTAT* Function Block

2.2.3 Creating the *MSTAT* Function Block in Hardware-Management

Configuring the *MSTAT* function block in Hardware Management:

- Open the resource's directory in the structure tree.
- Select **Protocols, PROFIBUS DP Master, Function Blocks, New, MSTAT**
- Open the context menu for the *MSTAT* function block in Hardware Management and select **Connect Signals** to open the *Signal Connections* dialog box.

The following inputs of the *MSTAT* function block in Hardware Management must be connected to the same signals that are connected in the application program to the *MSTAT* function block's *F_Outputs*.

Inputs	Type
M_ID	DWORD
REQ	BOOL
MODE	INT

Table 32: Inputs in the *MSTAT* Dialog Box

The following *MSTAT* function block's Outputs in Hardware Management must be connected to the same signals that are connected to the *MSTAT* function block's *F_Inputs* within the application program.

Outputs	Type
ACK	BOOL
BUSY	BOOL
DONE	BOOL
STATUS	DWORD

Table 33: Outputs in the *MSTAT* Dialog Box

2.2.4 Operation Sequence

To operate the *MSTAT* function block, the following steps are essential:

1. Within the application program, set the **A_Mode** input to the desired state.
   If *A_Mode* is not set, an error code is output after step 2 on the *A_Status* output and the PROFIBUS DP master's state is not set.
2. Within the application program, set the **A_Req** input to TRUE.

Note	The function block reacts to the rising edge on *A_Req*.

3. The *A_Busy* output is set to TRUE until the *MSTAT* command has been processed. Afterwards, *A_Busy* is set to FALSE and *A_Done* is set to TRUE.
4. If the specified mode could not be set, an error code is output to the *A_Status* output.
5. The master's current mode can be derived from the "Master Status" signal (see PROFIBUS DP Master)

2.3 RALRM Function Block

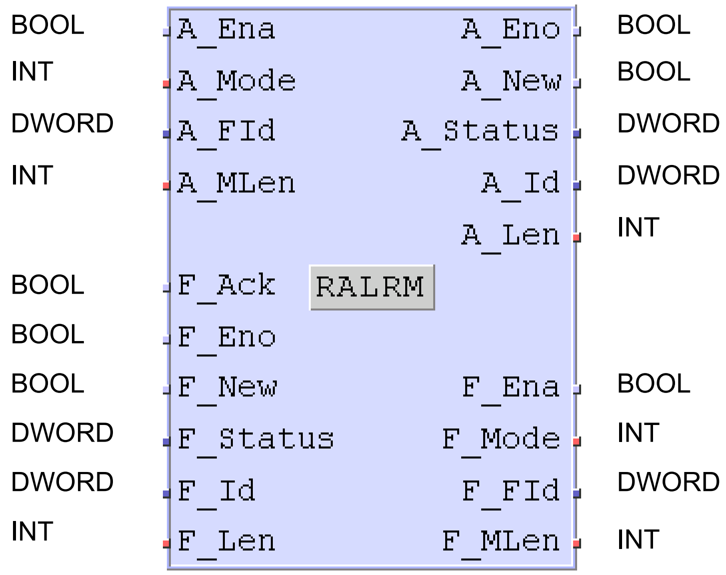


Figure 29: RALRM Function Block

The *RALRM* function block (DPV1 and higher) is used to evaluate the alarms.

Alarms are a special type of diagnostic messages that are handled with a high priority. Alarms report important events requiring a reaction by the application (e.g. a WRREC). Because the specific reaction depends on the manufacturer, please refer to the PROFIBUS DP slave manual for information about the reaction

As long as the *RALRM* function block is active, it waits for alarm messages from the slaves. If an alarm is received, the *A_NEW* output is set to TRUE for at least one cycle and the alarm data can be read from an alarm telegram. Prior to the next alarm, *A_NEW* is set to FALSE for at least one cycle. All alarms are acknowledged implicitly. No alarms are lost.

If several *RALRM* function blocks are used, the application program must be configured such that only one *RALRM* function block is active at any given time.

To configure the function block, use drag&drop to copy it from the function block library to the application program

2.3.1 Inputs and Outputs of the Function Block with the *Prefix A*

The inputs and outputs with the *prefix A* correspond to the user interface that the application program uses for controlling the function block.

A_Inputs	Description	Type
A_Ena	The function block is enabled with TRUE.	BOOL
A_Mode	Not used	INT
A_FID	Not used	DWORD
A_MLen	Maximum expected length of the received alarm data expressed in bytes	INT

Table 34: *A_Inputs* of the *RALRM* Function Block

A_Outputs	Description	Type
A_Eno	TRUE: The function block is active FALSE: The function block is not active	BOOL
A_New	TRUE: New alarm was received FALSE: No new alarm	BOOL
A_Status	State or error code (See 2.8 Error Codes of the Function Blocks).	DWORD
A_ID	Identification number of the slave triggering the alarm	DWORD
A_Len	Length of the received alarm data in bytes	INT

Table 35: *A_Outputs* of the *RALRM* Function Block

2.3.2 Inputs and Outputs of the Function Block with the *Prefix F*

The inputs and outputs of the function block that have the *prefix F* are used for connecting to the *RALRM* function block in Hardware Management.

Note	The function blocks in Project Management are connected to the corre-sponding function blocks in Hardware Management using common signals. These must be created in advance using the signal editor

The *RALRM* function block's *F_Inputs* in the application program must be connected to the same signals that are connected in the Hardware Management to the outputs of the *RALRM* function block.

F_Inputs	Type
F_ACK	BOOL
F_ENO	BOOL
F_NEW	BOOL
F_STATUS	DWORD
F_ID	DWORD
F_LEN	INT

Table 36: *F_Inputs* of the *RALRM* Function Block

The *RALRM* function block's *F_Outputs* in the application program must be connected to the same signals that are connected in Hardware Management to inputs of the *RALRM* function block.

F_Outputs	Type
F_Ena	BOOL
F_MODE	INT
F_FID	DWORD
F_MLEN	INT

Table 37: *F_Outputs* of the *RALRM* Function Block

2.3.3 *Creating the RALRM* Function Block in Hardware Management

Configuring the *RALRM* function block in Hardware Management:

- Open the resource's directory in the structure tree.
- Select **Protocols, PROFIBUS DP Master, Function Blocks, New, RALRM**
- Open the context menu for the *RALRM* function block in Hardware Management and select **Connect Signals** to open the *Signal Connections* dialog *box*

The following inputs of the *RALRM* function block in Hardware Management must be connected to the same signals that are connected in the application program to the *RALRM* function block's *F_Outputs*.

Inputs	Type
EN	BOOL
F_ID	DWORD
MLEN	INT
MODE	INT

Table 38: Inputs on the *RALRM* Dialog Box

The following outputs of the *RALRM* function block in Hardware Management must be connected to the same signals that are connected in the application program to the *RALRM* function block's *F_Inputs*.

Outputs	Type
ACK	BOOL
ENO	BOOL
ID	DWORD
LEN	INT
NEW	BOOL
STATUS	DWORD

Table 39: Outputs on the *RALRM* Dialog Box

Alarm data

The *Record* tab located in the *RALARM* dialog box contains signals that must be defined and whose structure must fit the alarm data. If no signals are defined, alarm data can be requested but not read.

An alarm message contains at least four bytes. The first four bytes of the alarm message contain the standard alarm data.

To simplify the evaluation of standard alarms, HIMA provides the auxiliary function block *ALARM* (see 2.9.2.3). To use it, combine the first four bytes into a signal of type DWORD and set this signal on the *IN* input of the *ALARM* auxiliary function block.

Note	If an alarm telegram contains more bytes than defined in the *Data* tab, only the preset number of bytes is accepted. The rest is cut off.

Alarm Data	Description
Byte 0	Length of the alarm message expressed in bytes (4 to 126)
Byte 1	Identification for the alarm type 1: Diagnostic alarm 2: Process alarm 3: Pull alarm 4: Plug alarm 5: Status alarm 6: Update alarm Others: Manufacturer specific. Please refer to the device manual provided by the manufacturer for more information about the specific meaning.
Byte 2	Slot number of the component sending the alarm
Byte 3	Bit 0..1: 0: no further information 1: inbound alarm, slot malfunction 2: outbound alarm, slot no longer malfunctioning 3: outbound alarm, continued slot malfunction Bit 2: Add Ack see Table 83 Bit 3 to 7: Alarm sequence number
Byte 4 to 126	Please refer to the device manual provided by the manufacturer for more information about the specific meaning.

Table 40: Alarm Telegram

Note The structure of the standard alarms (bytes 0…3) is standardized and identical for all manufacturers. Please consult the manual of the PROFIBUS DP slave for more information about bytes 4…126, the use of which is manufacturer specific.

Please note that devices built in accordance with the DPV0 standard do not support alarm telegrams.

2.3.4 Operation Sequence

To operate the *RALRM* function block, the following steps are essential:

1. On the application program's **A_Mlen** input, define the maximum amount of alarm data expressed in bytes that must be expected. *A_Mlen* cannot be modified during operation.
2. Set the **A_Ena** input to TRUE within the application program.

Note In contrast to other function blocks, the *RALRM* function block is only active as long as the *A_Ena* input is set to TRUE.

3. If the function block was started successfully, the *A_Eno* output is set to TRUE. If the function block could not be started, an error code is output to the *A_Status* output.
4. If a new alarm is received, the *A_New* output is set to TRUE for at least one cycle. During this time period, the alarm data of the slave triggering the alarm are contained in the outputs and can be evaluated.
5. Afterwards, the *A_New* output returns to FALSE for at least one cycle.
   The *A_Id* and *A_Len* outputs are reset to zero before the next alarm message can be received and evaluated.

2.4 *RDIAG* Function Block

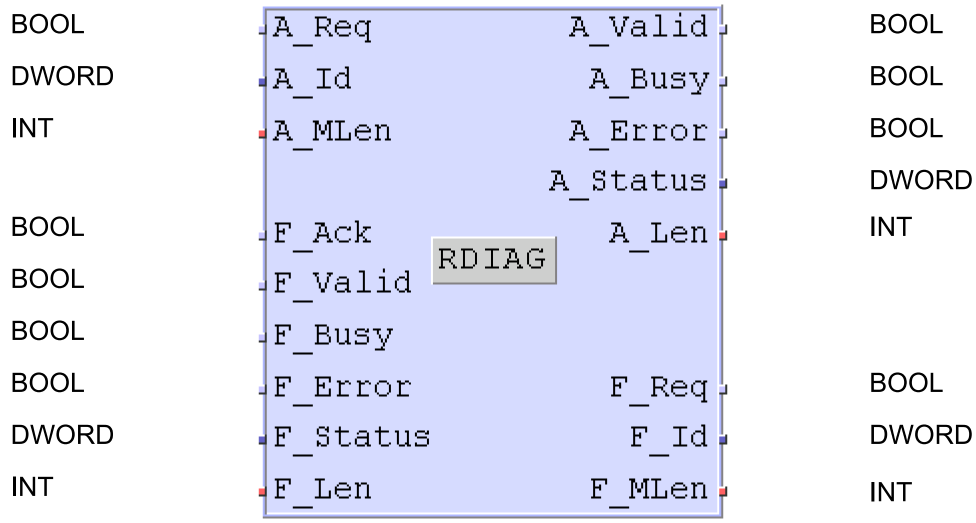


Figure 30: *RDIAG* Function Block

The *RDIAG* function block (DPV0 and higher) is used for reading a slave's current diagnostic message (6 bytes to 240 bytes).

As many RDIAG function blocks as desired may be simultaneously active within the HIMatrix PROFIBUS DP master.

To configure the function block, use drag&dropto copy it from the function block library to the application program.

2.4.1 Inputs and Outputs of the Function Block with *Prefix A*

The inputs and outputs with the *prefix A* correspond to the user interface that the application program uses for controlling the function block.

A_Inputs	**Description**	**Type**
A_Req	Rising edge starts a diagnostic message	BOOL
A_ID	Slave's identification number (see ID Auxiliary Function Block)	DWORD
A_MLen	Maximum length (in bytes) of the diagnostic mes- sage expected to be read	INT

Table 41: *A_Inputs* of *RDIAG* Function Block

"A_xxx"-Outp.	**Description**	**Type**
A_Valid	A new diagnostic message has been received and is valid	BOOL
A_Busy	TRUE: The reading process is still running	BOOL
A_Error	TRUE: An error occurred during the reading process	BOOL
A_Status	State or error code (See 2.8 Error Codes of the Function Blocks).	DWORD
A_Len	Length of the read diagnostic data in bytes	INT

Table 42: *A_Outputs* of *RDIAG* Function Block

2.4.2 Inputs and Outputs of the Function Block with the *Prefix F*

The inputs and outputs of the function block that have the *prefix F* are used for connecting to the *RDIAG* function block in Hardware Management.

Note	The function blocks in Project Management are connected to the corresponding function blocks in Hardware Management using common signals. These must be created in advance using the signal editor

The *RDIAG* function block's *F_Inputs* in the application program must be connected to the same signals that are connected in Hardware Management to the *RDIAG* function block's outputs.

F_Inputs	Type
F_ACK	BOOL
F_VALID	BOOL
F_BUSY	BOOL
F_ERROR	BOOL
F_Status	DWORD
F_LEN	INT

Table 43: *F_Inputs* of the *RDIAG* Function Block

The *RDIAG* function block's *F_Outputs* in the application program must be connected to the same signals that are connected in Hardware Management to the *RDIAG* function block's inputs.

F_Outputs	Type
F_Req	BOOL
F_Id	DWORD
F_Mlen	INT

Table 44: *F_Outputs* of the *RDIAG* Function Block

2.4.3 Creating the *RDIAG* Function Block in Hardware Management

Configuring the *RDIAG* function block in Hardware Management:

- Open the resource's directory in the structure tree.
- Select **Protocols, PROFIBUS DP Master, Function Blocks, New**, **RDIAG**
- Open the context menu for the *RDIAG* function block in Hardware Management and select **Connect Signals** to open the *Signal Connections* dialog box.

The following inputs of the *RDIAG* function block in Hardware Management must be connected to the same signals that are connected in the application program to the *RDIAG* function block's *F_Outputs*.

Inputs	Type
ID	DWORD
MLEN	INT
REQ	BOOL

Table 45: Inputs of the *RDIAG* Dialog Box

The following outputs of the *RDIAG* function block in Hardware Management must be connected to the same signals that are connected in the application program to the *RDIAG* function block's *F_Inputs*.

Outputs	Type
ACK	BOOL
BUSY	BOOL
ERROR	BOOL
LEN	INT
Status	DWORD
VALID	BOOL

Table 46: Outputs of the *RDIAG* Dialog Box

Diagnostic Data

The *Record* tab contains signals that must be defined and whose structure must fit the alarm data. If no signals are defined, alarm data can be requested, but not read. A diagnostic message contains at least six bytes and a maximum of 240 bytes. The first four bytes of the diagnostic message contain the standard diagnostics.

To simplify the evaluation of the standard alarms, HIMA provides the auxiliary function block *STDDIAG* (see 2.9.2.2) To use it, combine the first four bytes of the alarm message into a signal of type DWORD and set this signal on the *IN* input of the *STDDIAG* auxiliary function block.

Note	If a diagnostic telegram contains more bytes than defined in the *Record* tab, only the preset number of bytes is accepted. The rest is cut off.

Diagnostic Data	Description
Byte 0	Bytes 0 to 3 contain the standard diagnostics. The standard diagnostics can be decoded as a signal of the type DWORD using the *STDDIAG* auxiliary function block.
Byte 1	
Byte 2	
Byte 3	Bus address of the master to which a slave is assigned.
Byte 4	High byte (manufacturer ID)
Byte 5	Low byte (manufacturer ID)
Byte 6 to 240	Specific slave diagnostic data Please refer to the device manual provided by the manufacturer for more information about the specific meaning.

Table 47: The Layout of the Diagnostics Telegram

Note The HIMA slaves send a diagnostic telegram of six bytes in length. The meaning of these bytes is standardized.

The first six bytes of slaves from other manufacturers are only functionally identical. For further information about the diagnostic telegram, please refer to the description of the slave provided by the manufacturer.

2.4.4 Operation Sequence

To operate the *RDIAG* function block, the following steps are essential:

1. In the application program, connect the slave address with the **A_ID** input.
2. On the application program's **A_Mlen** input, define the maximum amount of diagnostic data expressed in bytes that must be expected.
3. In the application program, set the **A_Req** input to TRUE.

Note The function block reacts to a rising edge on *A_Req*.

4. The *A_Busy* output is set to TRUE until the diagnostic request has been processed. Afterwards, the *A_Busy* output is set to FALSE and *A_Valid* or *A_Error* is set to TRUE.
5. If the diagnostic telegram is valid, the *A_Valid* output is set to TRUE.
   The diagnostic data can be evaluated using the signals defined in the *Record* tab.
   The *A_Len* output contains the number of bytes that actually read.
6. If the diagnostic telegram could not be read successfully, the *A_Error* output is set to TRUE and an error code is output to the *A_Status* output.

2.5 *RDREC* Function Block

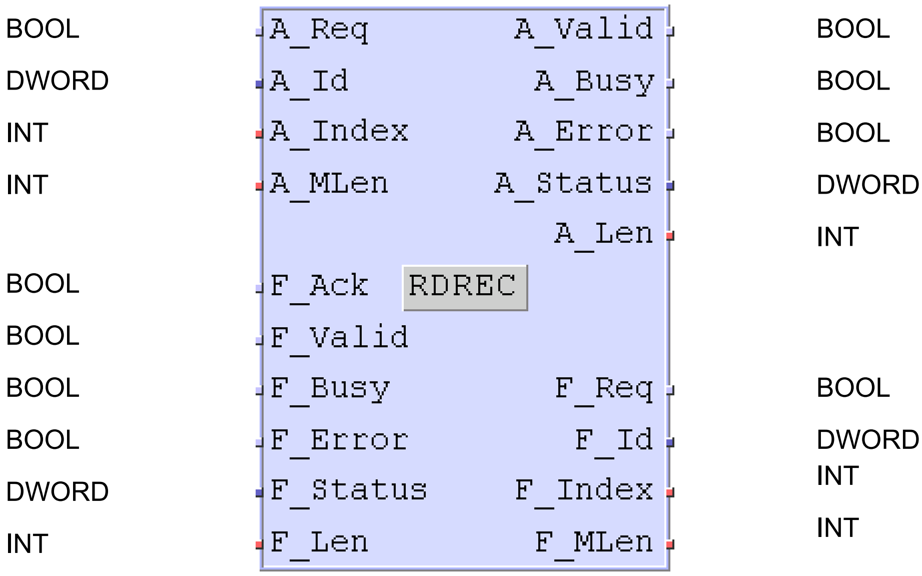


Figure 31: *RDREC* Function Block

The *RDREC* function block is used for acyclically reading a data record from a slave addressed on the *A_Index* input. Please consult the slave's manual to find out which data can be read.

This functionality is optional and is only defined with DPV1 and higher!

Up to 32 *RDREC* and/or *WRREC* function blocks can be simultaneously active in the HIMatrix PROFIBUS DP master.

To configure the function block, use drag&dropto copy it from the function block library to the application program.

2.5.1 Inputs and Outputs of the Function Block with the *Prefix A*

The inputs and outputs with the *prefix A* correspond to the user interface that the application program uses for controlling function block.

A_Inputs	Description	Type
A_Req	The rising edge starts the reading request	BOOL
A_Id	Slave's identification number (see ID Auxiliary Function Block)	DWORD
A_Index	Number of the data record to be read Consult the device manual provided by the manufacturer for more details about its meaning.	INT
A_MLen	Maximum length of the data to be read in bytes	INT

Table 48: A_Inputs of the *RDREC* Function Block

A_Outputs	Description	Type
A_Valid	A new data record was received and is valid.	BOOL
A_Busy	TRUE: The reading process is still running.	BOOL
A_Error	TRUE: An error occurred FALSE: No error	BOOL
A_Status	State or error code (See 2.8 Error Codes of the Function Blocks).	DWORD
A_Len	Length of the read data record information in bytes.	INT

Table 49: A_Outputs of the *RDREC* Function Block

2.5.2 Inputs and Outputs of the Function Block with the *Prefix F*

The inputs and outputs of the function block that have the *prefix F* are used for connecting to the *RDREC* function block in Hardware Management.

Note The function blocks in Project Management are connected to the corresponding function blocks in Hardware Management using common signals. These must be created in advance using the signal editor

The *RDREC* function block's *F_Inputs* in the application program must be connected to the same signals that are connected in Hardware Management to the *RDREC* function block's outputs.

F_Inputs	Type
F_Ack	BOOL
F_Valid	BOOL
F_Busy	BOOL
F_Error	BOOL
F_Status	DWORD
F_Len	INT

Table 50: *F_Inputs* of the *RDREC* Dialog Box

The *RDREC* function block's *F_Outputs* in the application program must be connected to the same signals that are connected in Hardware Management to the *RDREC* function block's inputs.

F_Outputs	Type
F_Req	BOOL
F_Id	DWORD
F_Index	INT
F_Mlen	INT

Table 51: *F_Outputs* of the *RDREC* Dialog Box

2.5.3 Creating the *RDREC* Function Block in Hardware Management

Configuring the MSTAT function block in Hardware Management:

- Open the resource's directory in the structure tree.
- Select **Protocols, PROFIBUS DP Master, Function Blocks, New, RDREC**
- Open the context menu for the *RDREC* function block in Hardware Management and select **Connect Signals** to open the *Signal Connections* dialog box.

The following inputs of the *RDREC* function block in Hardware Management must be connected to the same signals that are connected in the application program to the *RDREC* function block's *F_Outputs*.

Inputs	Type
ID	DWORD
INDEX	INT
MLEN	INT
REQ	BOOL

Table 52: Inputs of the *RDREC* Dialog Box

The following outputs of the *RDREC* function block in Hardware Management must be connected to the same signals that are connected in the application program to the *RDREC* function block's *F_Inputs*.

Outputs	Type
ACK	BOOL
BUSY	BOOL
ERROR	BOOL
LEN	INT
STATUS	DWORD
VALID	BOOL

Table 53: Outputs of the *RDREC* Dialog Box

Data	Description
No predefined signals	A user-defined data structure can be defined in the *Record* tab; however, the structure must adhere to the record's structure For details about the record structure, please refer to the manufacturer's operating instructions for the slave.

Table 54: Data of the *RDREC* Dialog Box

2.5.4 Operation Sequence

To operate the *RDREC* function block, the following steps are essential:

1. In the application program, set the slave address on the **A_ID** input.
2. On the application program's **A_Index** input, set the slave-specific index for the data record (manual provided by the manufacturer).
3. On the application program's **A_Len** input, set the length of the data record to be read.
4. In the application program, set briefly the **A_Req** input to TRUE.

Note	The function block reacts to a rising edge on *A_Req*.

5. The *A_Busy* output is TRUE until the data record request has been processed. Afterwards, the *A_Busy* output is set to FALSE and *A_Valid* or *A_Error* is set TRUE.
6. If the data record is valid, the *A_Valid* output is set to TRUE. The data record can be evaluated using the signals defined in the *Record* tab. The *A_Len* output contains the actual length of the data record that has been read.
7. If the data record could not be read successfully, the *A_Error* output is set to TRUE and an error code is output to the *A_Status* output.

Because the slave's data record is transmitted via the bus, this function call can take very long to complete.

2.6 *SLACT* Function Block

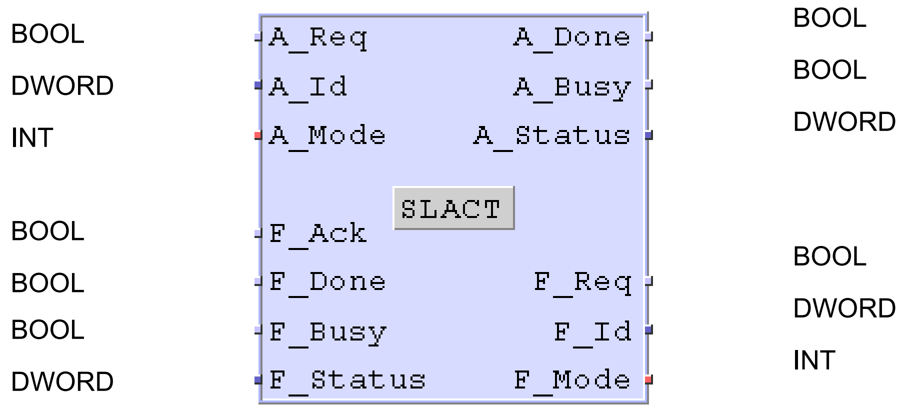


Figure 32: *SLACT* Function Block

The *SLACT* function block (DPV0 and higher) is used for activating and deactivating a slave from the PROFIBUS DP master's application program. The slave can thus be set to one of the following states using a timer or a switch connected to a physical input.

≠ 0: Active

= 0: Inactive

If various *SLACT* function blocks are used, the application program must be configured such that only one *SLACT* function block is active at a time.

To configure the function block, use drag&drop to copy it from the function block library to the application program.

2.6.1 Inputs and Outputs of the Function Block with the *Prefix A*

The inputs and outputs with the *prefix A* correspond to the user interface that the application program uses for controlling the function block.

A_Inputs	Description	Type
A_Req	Rising edge starts the function block	BOOL
A_ID	Slave's identification number (see ID Auxiliary Function Block)	DWORD
A_Mode	Target state for the slave PROFIBUS DP: ≠ 0: Active (connected) = 0: Not active (deactivated)	INT

Table 55: *A_Inputs* of the *SLACT* Function Block

A_Outputs	Description	Type
A_Done	TRUE: The PROFIBUS DP slave assumed the state defined on the *A_Mode* input.	BOOL
A_Busy	TRUE: The process for setting the PROFIBUS DP slave is still running.	BOOL
A_Status	State or error code (See 2.8 Error Codes of the Function Blocks).	DWORD

Table 56: *A_Outputs* of the *SLACT* Function Block

2.6.2 Inputs and Outputs of the Function Block with the *Prefix F*

The inputs and outputs of the function block that have the *prefix F* are used for connecting to the *SLACT* function block in Hardware Management.

Note	The function blocks in Project Management are connected to the corresponding function blocks in Hardware Management using common signals. These must be created in advance using the signal editor.

The *SLACT* function block's *F_Inputs* in the application program must be connected to the same signals that are connected in Hardware Management to the *SLACT* function block's outputs.

F_Inputs	Type
F_Ack	BOOL
F_Done	BOOL
F_Busy	BOOL
F_Status	DWORD

Table 57: *F_Inputs* of the *SLACT* Function Block

The *SLACT* function block's *F_Outputs* in the application program must be connected to the same signals that are connected in Hardware Management to the *SLACT* function block's inputs.

F_Outputs	Type
F_Req	BOOL
F_Id	DWORD
F_Mode	INT

Table 58: *F_Outputs* of the *SLACT* Function Block

2.6.3 Creating the *SLACT* Function Block in Hardware Management

Configuring the *SLACT* function block in Hardware Management:

- Open the resource's directory in the structure tree.
- Select **Protocols, PROFIBUS DP Master, Function Blocks, New, SLACT**
- Open the context menu for the *SLACT* function block in Hardware Management and select **Connect Signals** to open the *Signal Connections* dialog box.

The following inputs of the *SLACT* function block in Hardware Management must be connected to the same signals that are connected in the application program to the *SLACT* function block's *F_Outputs*.

Inputs	Type
ID	DWORD
REQ	BOOL
MODE	INT

Table 59: Inputs of the *SLACT* Dialog Box

The following outputs of the *SLACT* function block in Hardware Management must be connected to the same signals that are connected in the application program to the *SLACT* function block's *F_Inputs*.

Outputs	Type
ACK	BOOL
BUSY	BOOL
DONE	BOOL
STATUS	DWORD

Table 60: Outputs of the *SLACT* Dialog Box

2.6.4 Operation Sequence

To operate the *SLACT* function block, the following steps are essential:

1. In the application program, set the state desired for the slave on the **A_Mode** input.
2. On the application program's **A_ID** input, set the identifier for the slave address.
3. In the application program, set the **A_ Req** input to TRUE.

Note	The function block reacts to a rising edge on *A_Req.*

4. The *A_Busy* output is TRUE until the *SLACT* command has been processed. Afterwards, *A_Busy* is set to FALSE and *A_Done* is set to TRUE.
5. If the slave mode was able to be set successfully, it is output to the *A_Status* output.
6. If the slave mode was not able to be set successfully, an error code is output to the *A_Status* output.

The outputs in the dialog box must be connected to the function block's *F_Inputs* using signals.

2.7 *WRREC* Function Block

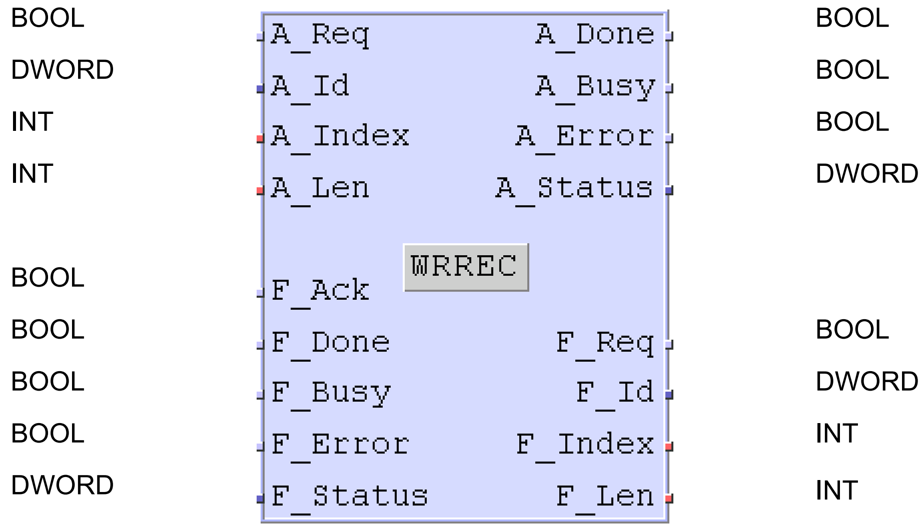


Figure 33: *WRREC* Function Block

The *WRREC* function block (DPV1 and higher) is used for acyclically writing a data record to a slave addressed with *A_Inde*x. Please consult the slave's manual to find out which data can be written.

Up to 32 *RDREC* and/or *WRREC* function blocks can simultaneously be active in the HIMatrix PROFIBUS DP master.

To configure the function block, use drag&drop to copy it from the function block library to the application program.

2.7.1 Inputs and Outputs of the function Block with the *Prefix A*

The inputs and outputs with the *Prefix A* correspond to the user interface that the application program uses for controlling the function block.

A_Inputs	Description	Type
A_Req	The rising edge starts the request for writing a data record.	BOOL
A_ID	Slave's identification number (see ID Auxiliary Function Block)	DWORD
A_Index	Number of the data record to be written. Consult the device manual provided by the manufacturer for more details about its meaning.	INT
A_Len	Length of the data record to be written in bytes	INT

Table 61: *A_Inputs* of the *WRREC* Function Block

A_Outputs	Description	Type
DONE	TRUE: The function block completed the writing process.	BOOL
BUSY	TRUE: The function block has not yet completed the writing process	BOOL
ERROR	TRUE: An error occurred FALSE: No error	BOOL
STATUS	State or error code (See 2.8 Error Codes of the Function Blocks).	DWORD

Table 62: *A_Outputs* of the *WRREC* Function Block

2.7.2 Inputs and Outputs of the Function Block with the *Prefix F*

The inputs and outputs of the function block that have the *Prefix F* are used for connecting to the *WRREC* function block in Hardware Management.

Note	The function blocks in Project Management are connected to the corresponding function blocks in Hardware Management using common signals. These must be created in advance using the signal editor.

The *WRREC* function block's *F_Inputs* in the application program must be connected to the same signals that are connected in Hardware Management to the *WRREC* function block's outputs.

F_Inputs	Type
F_Ack	BOOL
F_Done	BOOL
F_Busy	BOOL
F_Error	BOOL
F_Status	DWORD

Table 63: *F_Inputs* of the *WRREC* Dialog Box

The *WRREC* function block's *F_Outputs* in the application program must be connected to the same signals that are connected in Hardware Management to the *WRREC* function block's inputs.

F_Outputs	Type
F_Req	BOOL
F_Id	DWORD
F_Index	INT
F_Len	INT

Table 64: *F_Outputs* of the *WRREC* Dialog Box

2.7.3 Creating a *WRREC* Function Block in Hardware Management

Configuring the *WRREC* function block in Hardware Management:

- Open the resource's directory in the structure tree.
- Select **Protocols, PROFIBUS DP Master, Function Blocks, New, WRREC**
- Open the context menu for the *WRREC* function block in Hardware Management and select **Connect Signals** to open the *Signal Connections* dialog box.

The following inputs of the *WRREC* function block in Hardware Management must be connected to the same signals that are connected in the application program to the *WRREC* function block's *F_Outputs*.

Inputs	Type
ID	DWORD
INDEX	INT
LEN	INT
REQ	BOOL

Table 65: Inputs of the *WRREC* Dialog Box

The following outputs of the *WRREC* function block in Hardware Management must be connected to the same signals that are connected in the application program to the *WRREC* function block's *F_Inputs*.

Outputs	Type
ACK	BOOL
BUSY	BOOL
ERROR	BOOL
STATUS	DWORD
DONE	BOOL

Table 66: Outputs of the *WRREC* Dialog Box

Data	Description
No predefined signals	A user-defined data structure can be defined in the *Record* tab; however, the structure must adhere to the record's structure For details about the record structure, please refer to the manufacturer's operating instructions for the slave.

Table 67: Data in the *WRREC* Dialog Box

2.7.4 Operation Sequence

To operate the RALRM function block, the following steps are essential:

1. In the application program, set the slave address on the **A_ID** input.
2. On the application program's **A_Index** input, set the slave-specific index for the data record (manual provided by the manufacturer).
3. On the application program's **A_Len** input, set the length of the data record to be written.
4. In the application program, set the data record defined in the *Record* tab.
5. In the application program, set the **A_Req** input to TRUE.

Note	The function block reacts to a rising edge on *A_Req*.

6. The *A_Busy* output is set TRUE until to the data record has been written. Afterwards, the *A_Busy* output is set to FALSE and *A_Done* is set to TRUE.
7. If the data record could not be written successfully, the *A_Error* output is set to TRUE and an error code is output to the *A_Status* outputs.

2.8 Error Codes of the Function Blocks

If a function block is unable to correctly execute a command, an error code is output to the *A_Status* output (on the function block) and to the *STATUS* output (in the function block dialog box in HWM). The meaning of the error codes are described in the following table.

Error code	Symbol	Explanation
16#40800800	TEMP_NOT_AVAIL	Service temporary not available
16#40801000	INVALID_PARA	Invalid parameter
16#40801100	WRONG_STATE	The slave does not support the DPV1
16#40808000	FATAL_ERR	Fatal program error
16#40808100	BAD_CONFIG	Configuration error in the data area
16#40808200	PLC_STOPPED	The controller was stopped
16#4080A000	READ_ERR	Error while reading a record
16#4080A100	WRITE_ERR	Error while writing a record
16#4080A200	MODULE_FAILURE	The error cannot be specified in greater detail
16#4080B000	INVALID_INDEX	Index is invalid
16#4080B100	WRITE_LENGTH	Wrong length while writing
16#4080B200	INVALID_SLOT	Slot number invalid
16#4080B300	TYPE_CONFLICT	Wrong type
16#4080B400	INVALID_AREA	Wrong read/write area
16#4080B500	STATE_CONFLICT	Master in the wrong state
16#4080B600	ACCESS_DENIED	Slave not active (or similar)
16#4080B700	INVALID_RANGE	Wrong read/write range
16#4080B800	INVALID_PARAMETER	Wrong parameter value
16#4080B900	INVALID_TYPE	Wrong parameter type
16#4080C300	NO_RESOURCE	Slave not available
16#4080BA00	BAD_VALUE	Invalid Value
16#4080BB00	BUS_ERROR	Bus error
16#4080BC00	INVALID_SLAVE	Invalid slave ID
16#4080BD00	TIMEOUT	Time-out occurred
16#4080C000	READ_CONSTRAIN	Read constraint
16#4080C100	WRITE_CONSTRAIN	Write constraint
16#4080C200	BUSY	A function block of this type is already active
16#4080C300	NO_RESOURCE	Slave inactive

Table 68: Error Code Table for the Function Blocks

2.9 Auxiliary Function Blocks

The following auxiliary function blocks are run completely within the application program on the CPU.
The auxiliary function blocks are located in the same function block library **PBM_V…** as the function blocks (see chapter 2.1.1). The user can select the required auxiliary function blocks and use drag&drop to copy each of them directly to the application program.

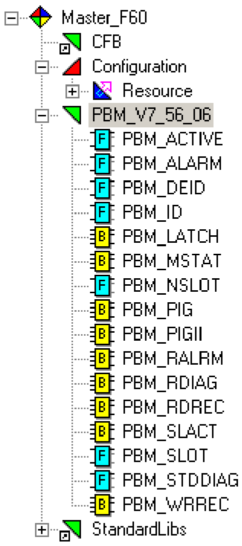


Figure 34: Auxiliary Function Blocks in Project Management

The following auxiliary function blocks are available:

Auxiliary Function blocks	Description
ID	Generate a four bytes identifier
SLOT	Create a SLOT identification number using a slot number
NSLOT	Create a continuous identification number for the slots
DEID	Decode the identification number
ACTIVE	Is the slave active or inactive?
STDDIAG	Decode a slave’s standard diagnostics
ALARM	Decode the alarm data
LATCH	Only used within other function blocks
PIG	Only used within other function blocks
PIGII	Only used within other function blocks

Table 69: Auxiliary Function Blocks and Corresponding Function

Note The signals for the auxiliary function blocks must be created in the Hardware Management editor. The signals are copied to the application program using drag&drop.

2.9.1 Auxiliary Function Blocks Using the Identifier

The following four auxiliary function blocks use an identifier (Identification number) of type DWORD.

These auxiliary function blocks set up the identifier such that it can be used by the HIMatrix PROFIBUS DP function blocks to identify the master, segment, slaves and modules/slots.

2.9.1.1 *ID* Auxiliary Function Block

(Create the identification number)

The *ID* function creates a four byte identifier. The input master is intended for future functions and is currently not used since each PROFIBUS DP function block is assigned unambiguously to a single master. The segment number is only important for class 2 masters and is thus not used.

These two inputs must either be opened or set to zero.

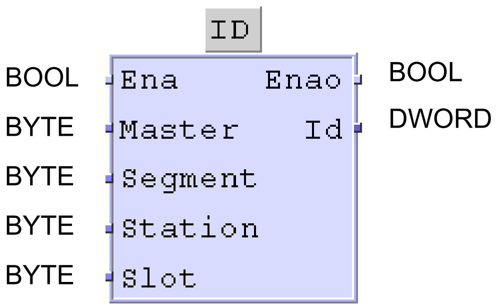


Figure 35: *ID* Auxiliary Function Block

Inputs	**Description**	**Type**
Ena	Not used	BOOL
Master	Not used	BYTE
Segment	Not used	BYTE
Station	Slave's bus address	BYTE
Slot	Slot or module number	BYTE

Table 70: Inputs of the *ID* Auxiliary Function Block

Outputs	**Description**	**Type**
Enao	Not used	BOOL
ID	Slave's identification number (Slave ID and slot number)	DWORD

Table 71: Outputs of the *ID* Auxiliary Function Block

2.9.1.2 *SLOT* Auxiliary Function Block

(Create an identification number using a slot number)

The *SLOT* function uses an identifier and a slot number to generate a new identifier that addresses the same slave as the first identifier but with the new slot number.

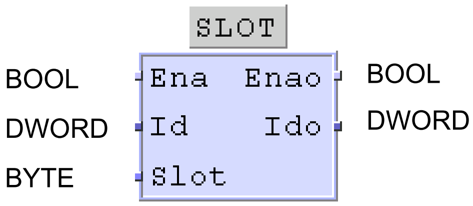


Figure 36: *SLOT* Auxiliary Function Block

Input	Description	Type
Ena	Not used	BOOL
ID	Slave component's logical address (slave ID and slot number)	DWORD
Slot	New slot or module number	BYTE

Table 72: Inputs of the *SLOT* Auxiliary Function Block

Outputs	Description	Type
Enao	Not used	BOOL
Id	Slave's identification number (Slave ID and slot number)	DWORD

Table 73: Outputs of the *SLOT* Auxiliary Function Block

2.9.1.3 *NSLOT* Auxiliary Function Block

(Create a continuous identification number for the slots)

The *NSLOT* function uses an identifier to generate a new identifier that addresses the next slot within the same slave. Ena must be set to TRUE to allow the auxiliary function block to run.

Enao is set to TRUE if the result on the Ido output is valid.

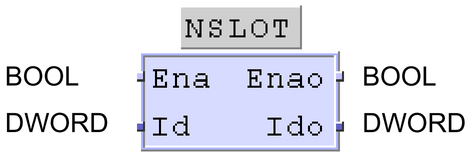


Figure 37: *NSLOT* Auxiliary Function Block

Input	Description	Type
Ena	The auxiliary function block runs as long as Ena is set to TRUE.	BOOL
Id	Slave's identification number (Slave ID and slot number)	DWORD

Table 74: Inputs of the *NSLOT* Auxiliary Function Block

Outputs	Description	Type
Enao	TRUE = the result is valid	BOOL
Ido	Slave identification number (Slave ID and slot number)	DWORD

Table 75: Outputs of the *NSLOT* Auxiliary Function Block

2.9.1.4 *DEID* Auxiliary Function Block

(Decode the identification number)

The *DEID* function decodes an identifier and analyzes its four components.

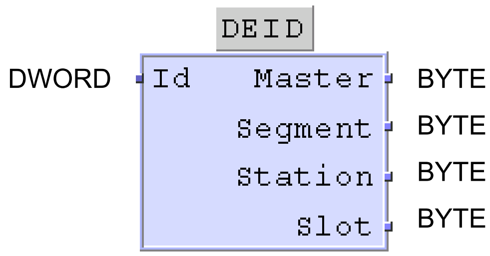


Figure 38: *DEID* Auxiliary Function Block

Input	Description	Type
Id	Slave's identification number (Slave ID and slot number)	DWORD

Table 76: Inputs of the *DEID* Auxiliary Function Block

Outputs	Description	Type
Master	Master's bus address	BYTE
Segment	Segment	BYTE
Stop	Slave's bus address	BYTE
Slot	Slot or module number	BYTE

Table 77: Outputs of the *DEID* Auxiliary Function Block

2.9.2 Auxiliary Function Blocks using the Standard Diagnostics

The following two auxiliary function blocks operate a slave's standard diagnostics.

The user can place the first four bytes of the slave's standard diagnostics in a signal of type DWORD.

- Create the signal **Stddiag** of type DWORD in the signal editor.
- Select a slave in Hardware Management.
- Open the slave 's context menu
- Select ***Connect Signals***.
- Use drag&drop to copy the signal **Stddiag** from the signal editor to the *Stati* tab located in the *Signal Connections*.

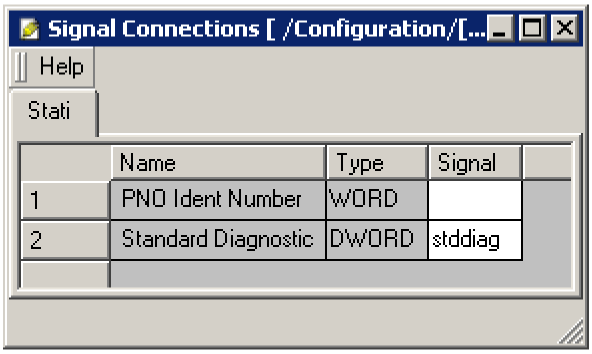


Figure 39: Reading the Standard Diagnostics of a PROFIBUS DP Slave

2.9.2.1 *ACTIVE* Auxiliary Function Block

(Slave active or inactive)

The *ACTIVE* function determines from a slave's standard diagnostics if the slave is active or inactive.

Figure 40: *ACTIVE* Auxiliary Function Block

Input	Description	Type
IN	Slave's standard diagnostics	DWORD

Table 78: Input of the *ACTIVE* Auxiliary Function Block

Outputs	Description	Type
OUT	TRUE: The slave is active FALSE: The slave is inactive	BOOL

Table 79: Outputs of the *ACTIVE* Auxiliary Function Block

2.9.2.2 *STDDIAG* Auxiliary Function Block

(Decode a slave's standard diagnostics)

The *STDDIAG* function decodes the slave's standard diagnostics of type DWORD.

The outputs of type BOOL in the *STDDIAG* auxiliary function block are set to TRUE if the corresponding bit has been set in the standard diagnostics.

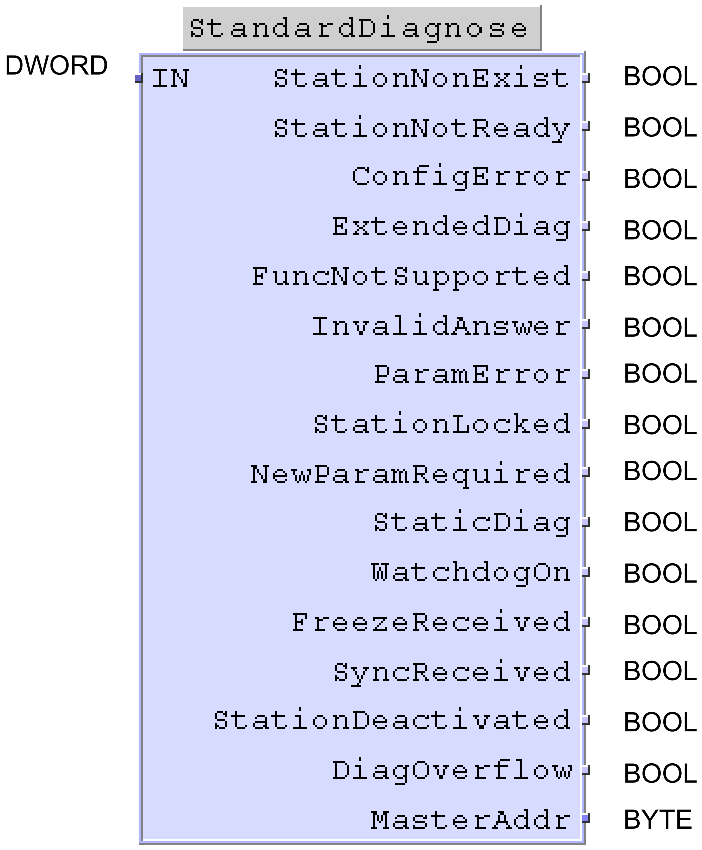


Figure 41: *STDDIAG* Auxiliary Function Block

Input	Description	Type
IN	Slave's standard diagnostics	DWORD

Table 80: Input of the *STDDIAG* Auxiliary Function Block

Outputs	Description	Type
StationNonExist	The slave does not exist	BOOL
StationNotReady	Slave not ready	BOOL
ConfigError	Configuration error	BOOL
ExtendedDiag	Extended diagnostics follows	BOOL
FuncNotSupported	The function is not supported	BOOL
InvalidAnswer	Invalid reply from slave	BOOL
ParamError	Parameter error	BOOL
StationLocked	Slave locked by another master	BOOL
NewParamRequired	New parameter data required	BOOL
StaticDiag	Static diagnostics	BOOL
WatchdogOn	Watchdog active	BOOL
FreezeReceived	Freeze command received	BOOL
SyncReceived	Sync command received	BOOL
StationDeactivated	The slave has been deactivated	BOOL
DiagOverflow	Diagnostics overflow	BOOL
MasterAddr	Master’s bus address	BYTE

Table 81: Outputs of the *STDDIAG* Auxiliary Function Block

2.9.2.3 *Alarm* Auxiliary Function Block

(Decode the alarm data)

The *Alarm* function operates on the standard alarm data. These are the first four bytes of an alarm message that are combined into a DWORD.

The auxiliary function block decodes the alarm data.

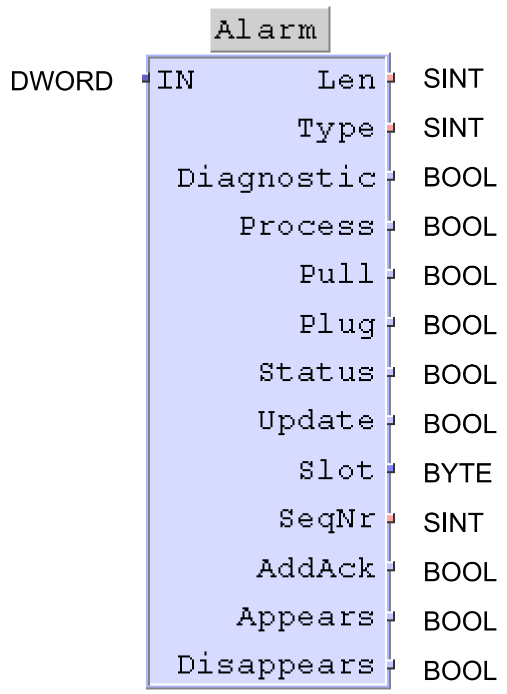


Figure 42: *Alarm* Auxiliary Function Block

Inputs	Description	Type
IN	Standard diagnostics	DWORD

Table 82: Inputs of the *Alarm* Auxiliary Function Block

<table>
<tr><th>Outputs</th><th colspan="2">Description</th><th>Type</th></tr>
<tr><td>Len</td><td colspan="2">Length of the entire alarm message.</td><td>SINT</td></tr>
<tr><td>Type</td><td colspan="2">1=Diagnostics alarm
2=Process
3=Pull alarm
4=Plug alarm
5=Status alarm
6=Update alarm
The other numbers are either reserved or manufacturer specific. Please look up the meaning of the corresponding alarms in the manual.</td><td>SINT</td></tr>
<tr><td>Diagnostic</td><td colspan="2">True = Diagnostics alarm</td><td>BOOL</td></tr>
<tr><td>Process</td><td colspan="2">True = Process alarm</td><td>BOOL</td></tr>
<tr><td>Pull</td><td colspan="2">True = Module has been pulled</td><td>BOOL</td></tr>
<tr><td>Plug</td><td colspan="2">True = Module has been plugged</td><td>BOOL</td></tr>
<tr><td>Status</td><td colspan="2">True = Status alarm</td><td>BOOL</td></tr>
<tr><td>Update</td><td colspan="2">True = Update alarm</td><td>BOOL</td></tr>
<tr><td>Slot</td><td colspan="2">Alarm Releasing Module</td><td>BYTE</td></tr>
<tr><td>SeqNr</td><td colspan="2">Alarm Sequence Number</td><td>SINT</td></tr>
<tr><td>AddAck</td><td colspan="2">TRUE means that the slave that triggered this alarm requires an additional acknowledgement from the application. The specific acknowledgement can be looked up in the manual.</td><td>BOOL</td></tr>
<tr><td>Appears
Disappears</td><td>FALSE
FALSE</td><td>If both are FALSE, no error has occurred up to this point.</td><td rowspan="4">BOOL</td></tr>
<tr><td>Appears Disappears</td><td>TRUE
FALSE</td><td>An error occurred and is still present.</td></tr>
<tr><td>Appears
Disappears</td><td>FALSE
TRUE</td><td>An error occurred and is disappearing.</td></tr>
<tr><td>Appears Disappears</td><td>TRUE
TRUE</td><td>If both are TRUE, the error disappears but the slave remains in a malfunction state.</td></tr>
</table>

Table 83: Functions of the *Alarm* Auxiliary Function Block

2.10 Example: Configuring the *RDIAG* Function Block

This example shows how to configure and use the *RDIAG* function block with the standard diagnostics of type DWORD. The standard diagnostics are read into the application program using the *RDIAG* function block and are decoded there using the *STDDIAG* auxiliary function block.

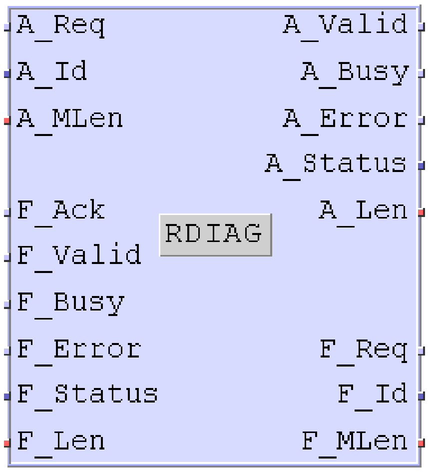


Figure 43: *RDIAG* Function Block

Start **ELOP II Factory** and create a new project or load en existing project.

Step 1: Creating the *RDIAG* function block in Hardware Management

- Select **Protocols, PROFIBUS Master, Function Blocks i**n the resource's structure tree.
- Open the context menu for the *Function Blocks* and select **New, RDIAG***.*

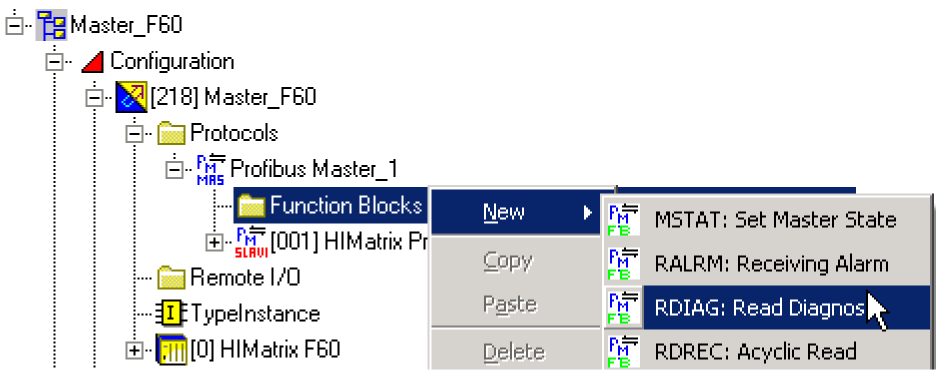


Figure 44: Creating the *RDIAG* Function Block

Step 2 Creating the signals required for the function block:

- Open the signal editor using the main menu **Signals, Editor**.
- Create the following signals in the signal editor.

Signals for *FB Outputs*		**Type**
RDIAG_Ack		BOOL
RDIAG_Busy		BOOL
RDIAG_Error		BOOL
RDIAG_Len		INT
RDIAG_Status		DWORD
RDIAG_Valid		BOOL
Signals for *FB Inputs*		**Type**
RDIAG_Id		DWORD
RDIAG_Mlen		INT
RDIAG_Req		BOOL
Signals for *FB Data*		**Type**
stddiag		DWORD
Signals	**Value**	**Type**
SlaveId	1	Byte
Request	-	BOOL
A_MLen	4	INT

Table 84: Signals for the *RDIAG* Function Block

Step 3 Connecting the signals with the *FB outputs*:

- Select **Connect Signals** from the *RDIAG*'s context menu to open the *Signal Connections* dialog box
- Use drag&drop to copy the signals from the signal editor to the **FB Outputs** tab located in the *Signal Connections* dialog box.

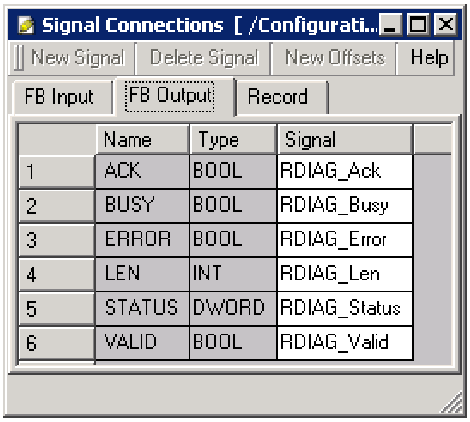


Figure 45: FB Outputs in the *Signal Connections* Dialog Box

Step 4 Connecting the signals with the *FB inputs*:

- Select **Connect Signals** from the *RDIAG*'s context menu to open the *Signal Connections* dialog box
- Use drag&drop to copy the signals from the signal editor to the **FB Inputs** tab located in the *Signal Connections* dialog box.

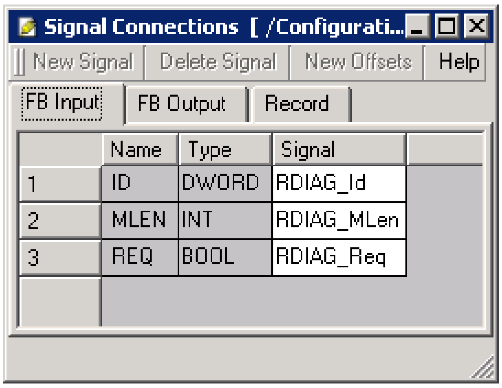


Figure 46: FB Inputs in the *Signal Connections* Dialog Box

Step 5 Connecting the signals ***stddiag*** with the *data*:

- Select **Connect Signals** from the *RDIAG*'s context menu to open the *Signal Connections* dialog box
- Use drag&drop to copy the signal **stddiag** from the signal editor to the **Record** tab located in the *Signal Connections* dialog box.

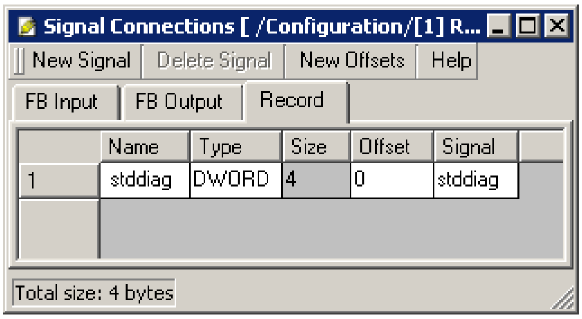


Figure 47: Data in the *Signal Connections* Dialog Box

Step 6 Opening the application program:

- Move to Project Management.
- Select the resource in which the *RDIAG* function block dialog box was created.
- Open the application program.

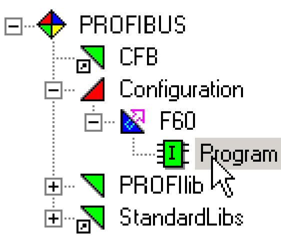


Figure 48: Application Program

Step 7: Adding function blocks to the FBD Editor:

- Open the function block library **PROFIlib** in the structure tree.
- Use drag&drop to copy the **RDIAG** function block to the FBD Editor.
- Use drag&drop to copy the **STDDIAG** auxiliary function block to the FBD Editor.
- Use drag&drop to copy the **ID** auxiliary function block to the FBD Editor.

Step 8 Creating the following logic in the FBD Editor.:

- Use drag&drop to copy all signals created in Step 3 to the FBD Editor.
- Connect the signals with the *function blocks*, see figure below.
- For monitoring purposes, create *Online Test Fields* on the function blocks' outputs.
- Close the FBD Editor.

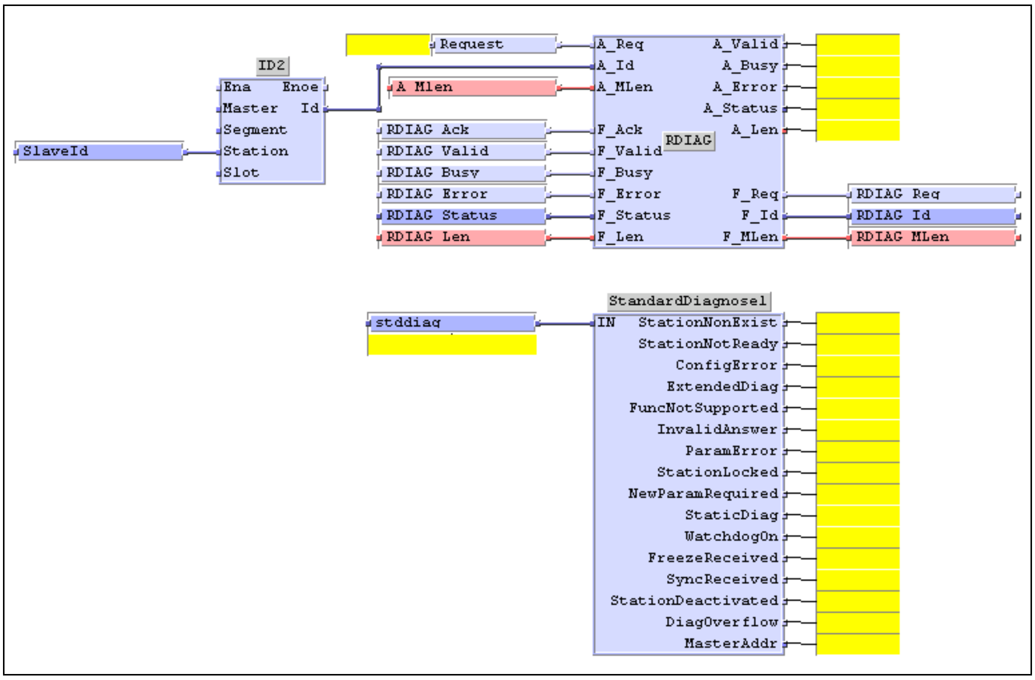


Figure 49: The completed Logic with the *RDIAG* Function Block in the FBD Editor

Step 9: Loading the code into the controller:

- Start the code generator for the resource.
- Please, make sure that the code has been generated without errors (see error-status display field).
- Load the code into the controller and leave the PADT connected to the controller for the next step.

Step 10: Verifying the logic with the online test:

- Open the resource's context menu in the structure tree and select **Online Test**.
- Set the signal **Request** to TRUE to read the slave's standard diagnostics (SlaveId=1).
- Refresh the standard diagnosis by briefly setting *Request* to FALSE and then back to TRUE.

Please, refer to chapter 2.4 for further information about the *RDIAG* function block's function.

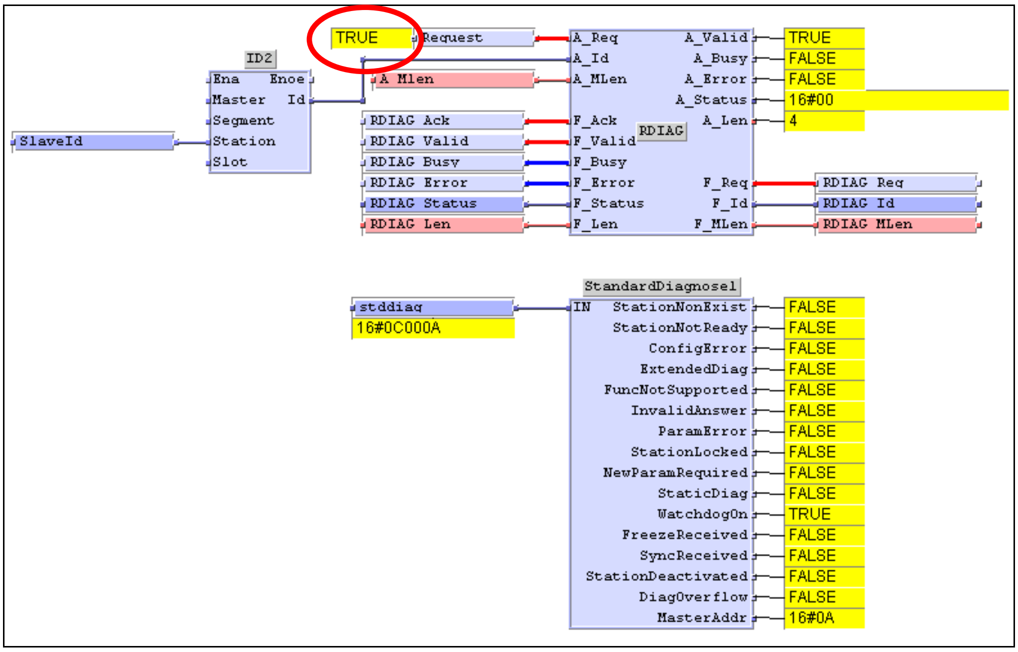


Figure 50: Online Test

3 HIMatrix PROFIBUS DP Slave

This chapter describes the characteristics of the HIMatrix PROFIBUS DP slave and the ***ELOP II Factory*** menu functions and dialog boxes required for configuring the HIMatrix PROFIBUS DP slave.

Note	For each HIMatrix controller type, the user can refer to the corresponding system documentation specifying the electrical and mechanical data. (See Engineering Manual HI 800 101 and data sheets of the HIMatrix controllers).

3.1 Equipment and System Requirements

HIMA ELOP II Factory	Version 3.2.0 and beyond
HIMatrix controller	F20 F30, F35 and F60 Hardware Revision: 00 and beyond
Operating system versions of the HIMatrix controller	COM OS Version 3.14 and beyond CPU OS Version 3.14 and beyond
HIMatrix PROFIBUS DP slave module	The serial field bus interface (FB1 or FB2) used on the HIMatrix controller must be equipped with an optional HIMatrix PROFIBUS DP module
License number	Not used, activation via the module

3.2 HIMatrix PROFIBUS DP Slave Characteristics

Type of PB slave	DPV0	
Transfer rate	9.6 kbit/s up to 12 Mbit/s	
Bus address	0 up to 125	
Max. number of slaves	Only one PROFIBUS DP slave per resource can be configured.	
Process data volume of the HIMA PB slave	DP output/ELOP import: DP input/ELOP export: Total:	max. 192 bytes max. 240 bytes max. 256 bytes
Protocol watchdog	If the PROFIBUS-DP slave is in the DATA EXCHANGE state and the connection to the PROFIBUS DP master is lost, the DP slave detects this when the *watchdog time [ms]* has expired (parameter within the master, see Chapter 1.5.7.1). In this case, the status signal *Data Valid* is set to FALSE and the connection state is set to OFFLINE (see chapter 3.4.1.1). The PROFIBUS-DP master input signals are ignored and instead initial values are used.	

3.3 FBx LED Function in the PROFIBUS DP Slave

The COM indicates the state of the local PROFIBUS-DP slave protocol using one of the LEDs assigned to the field bus interfaces. The states of these LEDs are specified in the following table.

FBx LED	Description
OFF	The PROFIBUS-DP slave protocol is not active! I.e. the resource is in the STOP state or no PROFIBUS DP slave is configured.
Blinking (0.5 Hz)	No data traffic! The PROFIBUS DP slave is configured and active, but there is no connection to the PROFIBUS-DP master.
COM OS Versions prior to V10 Blinking (0.5 Hz) **COM OS Version V11 and beyond** Blinking (2.5 Hz)	The PROFIBUS-DP Slave rejects the parameterization / configuration data of the PROFIBUS-DP master! I.e. the PROFIBUS-DP master's and/or the PROFIBUS-DP slave's configurations are faulty or do not correspond to one another.
ON	The PROFIBUS-DP slave protocol is active and is exchanging data with the PROFIBUS-DP master.

3.4 Context Menu for the HIMatix PROFIBUS DP Slave

The context menu for the PROFIBUS DP slave contains the following functions:

PROFIBUS DP Slave
Connect Signals
Validate
New
Import
Export
Copy
Paste
Delete
Print
Properties

Please refere to chapter 1.4.2 for further information about the standard menu functions Validate, New, Import, Export, Copy, Paste, Delete and Print.

3.4.1 Menu Function *Connect Signals*

The *Connect Signals* menu function opens the *Signal Connections* dialog box.

3.4.1.1 *Input* Tab

The signals that should be read in the controller are designated in the *Inputs* tab.

Further, the *Inputs* tab contains the following PROFIBUS DP slave status signals that can be assessed from within the application program.

Signal	Description	Type
Current baud rate	Baud rate currently used by the PROFIBUS DP slave protocol. Possible (decimal) values: 0 (no baud rate determined) 9600 (9.6 kBaud) 19200 (19.2 kBaud) 45450 (45.45 kBaud) 93750 (93.75 kBaud) 187500 (187.5 kBaud) 500000 (500 kBaud) 1500000 (1.5 MBaud) 3000000 (3 MBaud) 6000000 (6 MBaud) 12000000 (12 MBaud) Default value: 0	UDINT

Error code	If an error occurred within the PROFIBUS-DP slave protocol, it is transferred in this signal. The last occurred error is displayed. Possible (hexadecimal)values: 0x00: No error 0xE1: faulty parameterization by the PB-Master 0xD2: faulty configuration by the PB-Master Default value: 0x00	DWORD
Protocol State	Describe the PROFIBUS-DP slave protocol's status (see the status diagram depicted above) Possible (hexadecimal) value: 0xE1: The controller is disconnected from the bus or not active. 0xD2: The controller waits for a configuration from the master. 0xC3: The controller exchanges cyclic data with the master. Default value: 0xE1	BYTE
Slave ID	This signal contains the controller's PROFIBUS-DP slave-ID used on the bus. The slave ID must be configured beforehand by the user using the PADT. Possible (decimal) values: 0-125: Controller's PB-DP slave ID Default value: 0xFF	USINT
Master ID	This is the ID of the PROFIBUS master that parameterized and configured the PROFIBUS-DP slave. Possible (decimal) values: 0-125: Master's ID 255: The slave is not assigned to any master Default value: 0xFF	USINT
Data valid	If the status signal *Data valid* is set to TRUE, the slave received valid import data from the master. The status signal *Data valid* is set to FALSE if the watchdog time within the slave has expired. PROFIBUS-DP signals are not valid, Initial values are used! Default value: FALSE **Note:** If the master did not activate the slave's watchdog and the connection is lost, the status signal *Data valid* retains the value TRUE as the PROFIBUS DP slave has no means to recognize the loss of the connection. This fact must be taken into consideration when using this signal!	BOOL

Table 85: Status Signals of the HIMatrix PROFIBUS DP Slave

3.4.1.2 *Output* Tab

The *Output* tab specifies the signals that should be exported from the controller.

3.4.2 Menu Function *Properties*

Properties from the context menu for the HIMatrix PROFIBUS DP slave opens the *Properties* dialog box.
The dialog box contains the tabs *General* and *CPU/COM*.

3.4.2.1 *General* Tab

Element	Description	Value
Type	PROFIBUS DP slave	Display only
Station address	Slave's station address Only one slave's station address may be available on the bus.	Min: 0 Max: 125 Standard: 0
Interface	Field bus interface that should be used for the PROFIBUS DP slave.	FB1, FB2, FB3
Baud rate [bps]	Baud rate used for the bus. Possible values: 9600 (9.6 kBaud) 19200 (19.2 kBaud) 45450 (45.45 kBaud) 93750 (93.75 kBaud) 187500 (187.5 kBaud) 500000 (500 kBaud) 1500000 (1.5 MBaud) 3000000 (3 MBaud) 6000000 (6 MBaud) 12000000 (12 MBaud)	

Table 86: General Settings for the PROFIBUS DP Slave in the *Properties* Dialog Box

3.4.2.2 *CPU/COM* Tab

The default parameter values provide a fast means of exchanging PROFIBUS DP data between the COM processor (COM) and the PROFIBUS DP slave hardware within the *HIMatrix* controller.

These parameters should only be changed if the COM and/or CPU load must be lowered for an application and provided the process allows it.

Only experienced programmers should modify the parameters.

Increasing the COM and CPU refresh rate means that the effective refresh rate of the PROFIBUS DP data is also increased. The time requirements of the plant must be verified.

Please also consider the parameter *Min. Slave Interval [ms]* (see 1.4.3.2) which defines the minimum refresh rate of the PROFIBUS DP data between PROFIBUS DP master and PROFIBUS DP slave.

Element	Description
Refresh Rate [ms]	Refresh rate in milliseconds at which the COM and the PROFIBUS DP slave hardware exchange protocol data. Range of value: 4 to 1000 Default value: 10
Within one cycle	Activated: Transfer of all protocol data from the CPU to the COM within a CPU cycle. Deactivated: Transfer of all protocol data from the CPU to the COM, distributed over multiple CPU cycles, each with 900 bytes per data direction. This way, the controller's cycle time can also be lowered. Default value: activated

Table 87: Properties of the PROFIBUS DP Slave

3.5 Example: Configuring a HIMatrix PROFIBUS DP Slave

In this example, the HIMatrix PROFIBUS DP slave exchanges signals with a HIMatrix PROFIBUS DP master.

The example shows how the corresponding modules within the HIMatrix PROFIBUS DP master must be created and parameterized for the signals within the PROFIBUS DP slave.

3.5.1 Assigning Signals within the HIMatrix PROFIBUS DP Slave

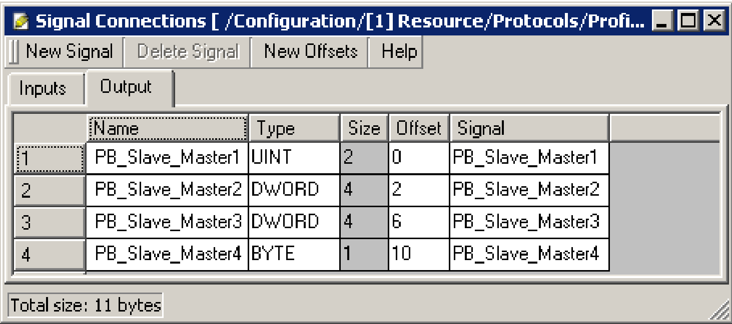
Note The start address for the input and output signals of the HIMatrix PROFIBUS DP slaves always start with 0.

If a PROFIBUS DP master (provided by another manufacturer) expects a higher start address, dummy signals must be inserted in front of the signals used.

In this example, the following signals are created within the PROFIBUS DP slave:

The output signals consist of **four signals** with a total of 11 bytes.

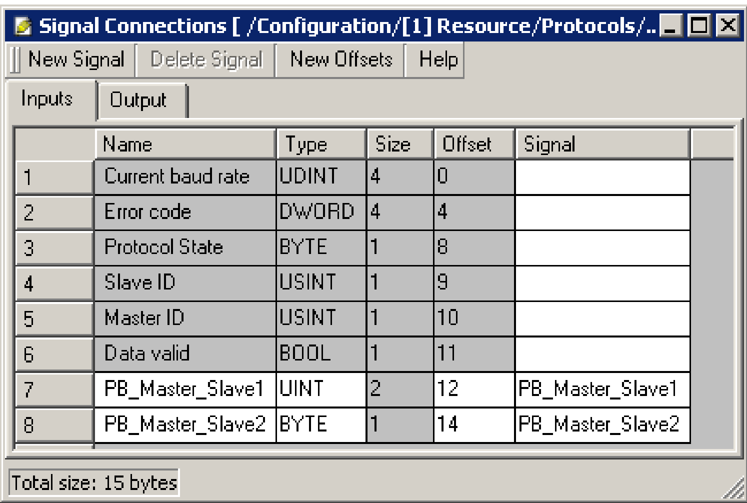
The output signal with the lowest offset has the start address 0.



	Name	Type	Size	Offset	Signal
1	PB_Slave_Master1	UINT	2	0	PB_Slave_Master1
2	PB_Slave_Master2	DWORD	4	2	PB_Slave_Master2
3	PB_Slave_Master3	DWORD	4	6	PB_Slave_Master3
4	PB_Slave_Master4	BYTE	1	10	PB_Slave_Master4

Figure 51: *Outputs* Tab in the HIMatrix PROFIBUS DP Slave

The input signals consist of **two signals** with a total of 3 bytes.
The input signal with the lowest offset has the start address 0. The offsets of the system signals (greyed out) are not taken into account.



	Name	Type	Size	Offset	Signal
1	Current baud rate	UDINT	4	0	
2	Error code	DWORD	4	4	
3	Protocol State	BYTE	1	8	
4	Slave ID	USINT	1	9	
5	Master ID	USINT	1	10	
6	Data valid	BOOL	1	11	
7	PB_Master_Slave1	UINT	2	12	PB_Master_Slave1
8	PB_Master_Slave2	BYTE	1	14	PB_Master_Slave2

Figure 52: *Inputs* Tab in the HIMatrix PROFIBUS DP Slave

After creating the signals click **New Offsets** to renumber the offsets.

3.5.2 Configuring the PROFIBUS DP Slave within the PROFIBUS DP Master

3.5.2.1 Creating the *HIMatrix* PROFIBUS DP Modules

The number of bytes that must actually be transferred must be configured within the PROFIBUS DP master. This is done by selecting the *Modules* defined in the PROFIBUS DP slave GSD file.

To configure the number of bytes for the PROFIBUS DP master's input and output signals, select enough modules to correspond to the slave's physical configuration.

The *HIMatrix* PROFIBUS DP slave GSD file is called *hix100ea.gsd* and offers the following modules:

PROFIBUS DP Master Input Modules	**Number**	**Type**
DP Input/ELOP Export	1	Byte
DP Input/ELOP Export	2	Bytes
DP Input/ELOP Export	4	Bytes
DP Input/ELOP Export	8	Bytes
DP Input/ELOP Export	16	Bytes
DP Input/ELOP Export	1	Word
DP Input/ELOP Export	2	Words
DP Input/ELOP Export	4	Words
DP Input/ELOP Export	8	Words
DP Input/ELOP Export	16	Words
PROFIBUS DP Master Output Modules	**Number**	**Type**
DP Output/ELOP Import	1	Byte
DP Output/ELOP Import	2	Bytes
DP Output/ELOP Import	4	Bytes
DP Output/ELOP Import	8	Bytes
DP Output/ELOP Import	16	Bytes
DP Output/ELOP Import	1	Word
DP Output/ELOP Import	2	Words
DP Output/ELOP Import	4	Words
DP Output/ELOP Import	8	Words
DP Output/ELOP Import	16	Words

Figure 53: Module in the *HIMatrix* GSD File `hix100ea.gsd`

Note The number of modules used to achieve the necessary number of bytes is not important as long as the maximum of 32 modules is not exceeded.

To avoid unnecessarily complicating the PROFIBUS DP master configuration, we recommend keeping the number of selected modules to a minimum.

In this example, the following modules are created in the PROFIBUS DP master to allow the PROFIBUS DP slave to receive **11 bytes** and to send **3 bytes**.

Number the *HIMatrix* PROFIBUS DP modules without gaps and in ascending order, starting with **0**.

The order in which the PROFIBUS DP modules are arranged is not important for operation; however, we recommend organizing the DP input and output modules in an orderly manner to ensure an overview can be maintained.

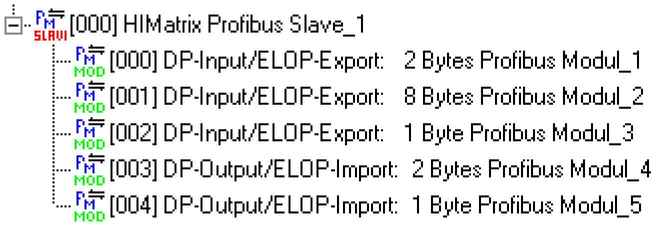


Figure 54: Modules from the *HIMatrix* GSD File suitable for this Example

3.5.2.2 Connecting Signals within the PROFIBUS DP Modules

Select *Connect Signals* from the context menu of the corresponding PROFIBUS DP module to open the *Signal Connections* dialog box.

Note	The sum of the signals in bytes must correspond to the module's size in bytes.

3.5.2.2.1 Connecting Signals within the Input Modules

The signals that the master receives from the slave are entered into the *Input* tab within the input modules *DP Input/ELOP Export:*.

After creating the signals, click **New Offsets** to renumber the offsets.

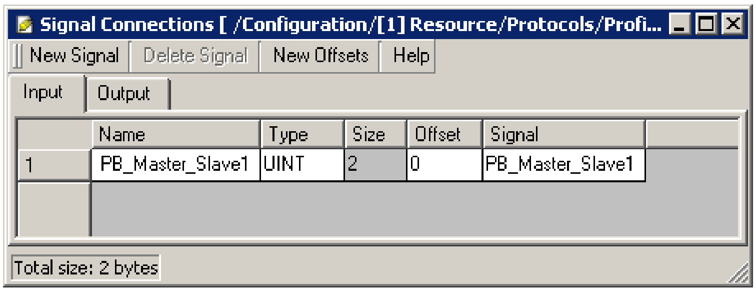


Figure 55: *Input* Tab within the *[000] DP Input/ELOP Export: 2 Bytes* Module

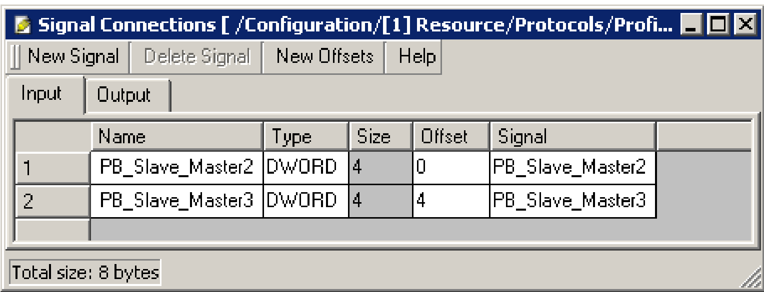


Figure 56: *Input* Tab within the *[001] DP Input/ELOP Export: 8 Bytes* Module

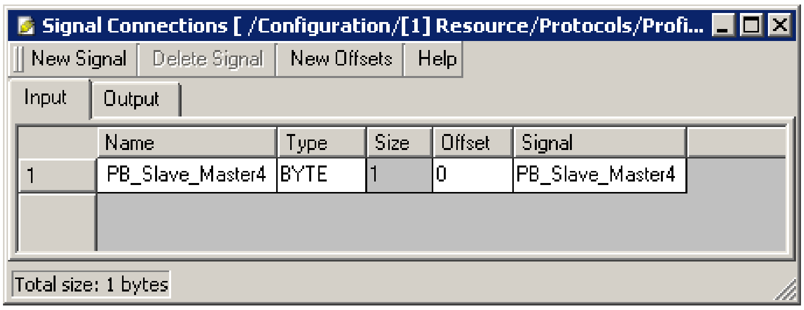


Figure 57: *Input* Tab within the *[002] DP Input/ELOP Export: 1 Byte* Module

3.5.2.2.2 Connecting Signals within the Output Modules

The *Output* tab within the output modules *DP Output/ELOP Import:* specifies the signals that the master sends to the slave.

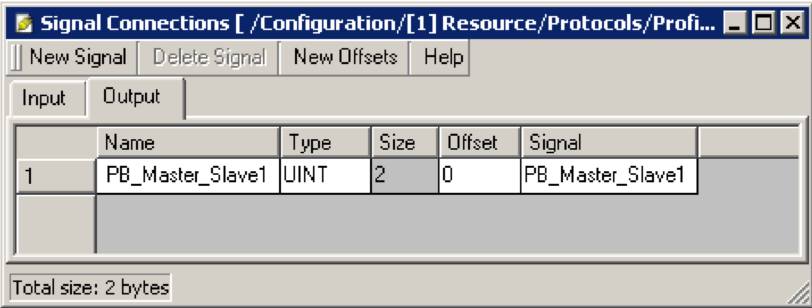


Figure 58: *Output* Tab within the *[003] DP-Output/ELOP-Import 2 Bytes* Module

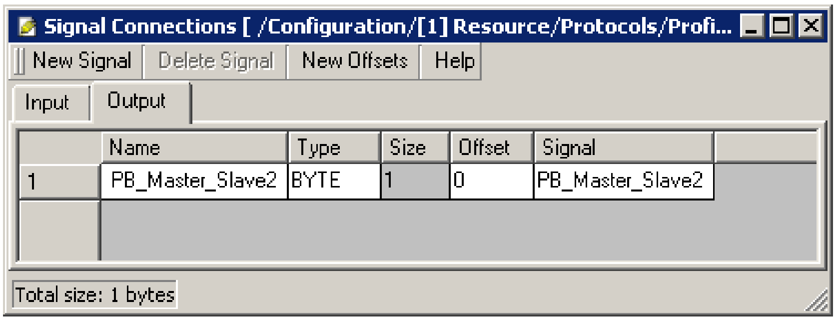


Figure 59: *Output* Tab within the *[004] DP-Output/ELOP-Import 1 Byte* Module

3.5.2.3 Creating the User Data within the PROFIBUS DP Master

Only the **start address** and the **number of signals** are defined in the parameter data. Further, the PROFIBUS DP master must be configured for the number of bytes that must actually be transferred. This is done by selecting **Modules** from the GSD file (see 3.5.2.1).

Within the HIMatrix PROFIBUS DP master, the 32 byte long user data field of the HIMatrix PROFIBUS DP slaves is represented as shown in Figure 60.

In this dialog box, the user data are also presented in a table to simplify the process of entering the **start address** and the **number of signals**.

In this example, the following user data are created in the PROFIBUS DP master to allow the PROFIBUS DP slave to receive **four signals** and to send **two signals**.

The start address of both the input and the output signals starts with 0 respectively.

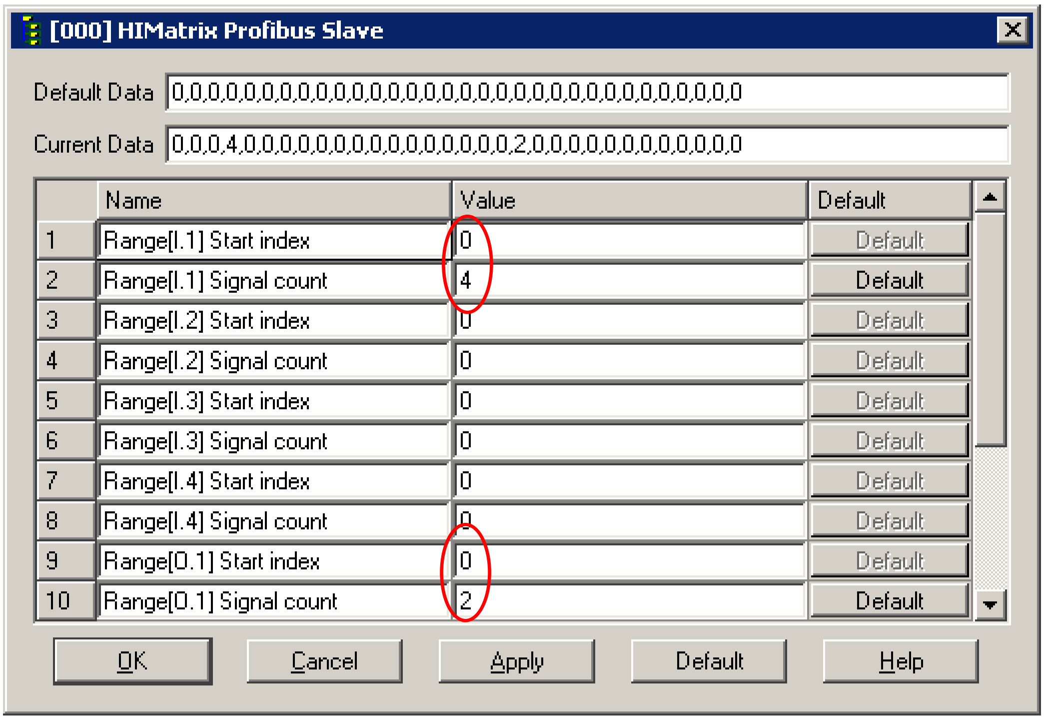


Figure 60: 32 Byte User Data for the HIMatrix PROFIBUS DP Slave within the Master

3.5.2.4 Verifying the PROFIBUS DP Master's Configuration

After configuring the PROFIBUS DP master, open the context menu for the PROFIBUS DP Master and select **Validate**.

After validating the PRORFIBUS-DP master's configuration, potential errors and warnings appear in the error-state display field.

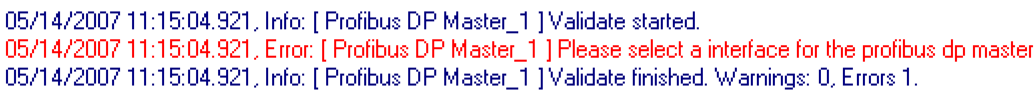


Figure 61: Error-State Display Field

4 HIMatrix PROFIBUS DP Basics

This chapter describes the basic principles and implementation of PROFIBUS DP communications for the HIMatrix controllers.

For those interested in background information about PROFIBUS DP communications, this section offers information about

- Basics hardware principles for serial data transmission via RS-485
- PROFIBUS DP telegrams and their associated protection mechanisms
- PROFIBUS DP cycles (acyclic, cyclic and isochronous)

For further information, please refer to the following specifications provided by PNO Germany (See www.profibus.com):

-PROFIBUS Technology and Application, October 2002

-PROFIBUS Guideline No. 2.182 Version 1.2, July 2001

Note:	For each HIMatrix controller type, the user can refer to the corresponding system documentation specifying the electrical and mechanical data. (See Engineering Manual HI 800 101 and data sheets of the HIMatrix controllers www.hima.com).

4.1 DP Stage of Extension

The DP (Decentralized Peripherals) stages of extension are specified in the IEC 61158.

DP-V0 Provide the basic DP functionalities. These include cyclic data exchange as well as station, module and channel specific diagnostics.

DP-V1 Contain extensions focusing on process automation. In addition to cyclical user data traffic, this emphasizes acyclic data traffic for parameterizing, operating, monitoring and alarm handling in "intelligent" field devices.

DP-V2 Contains further extensions such as the isochronous data exchange.

4.2 PROFIBUS DP Device Types

Definition of PROFIBUS DP device types

- The class 1 master is an active station and exchanges user data with the slave.
- The class 2 master is an active station and mostly a personal computer (PADT). PROFIBUS DP is configured and parameterized using class 2 masters.
- The slave is a passive station and exchanges user data with the master. A slave can be a simple field device, but also a complex controller.

4.3 Basic Hardware Principles for Serial Data Transfer

In the PROFIBUS DP's physical layer, data are transferred symmetrically in accordance with the RS-485 standard.

4.3.1 Basic Technical Characteristics of the RS-485 Transmission

The following table presents the basic technical characteristics of the RS-485 transmission used for the PROFIBUS DP.

Area	Sizes	Remark
NetworkTopology	Linear bus, active bus termination on both ends	Branch lines are only allowed for baud rates of up to 1.5 Mbps
Medium	Shielded, twisted cable	The shielding can be omitted depending on the environmental conditions
Number of stations	32 stations in every segment, without repeaters	With three repeaters, extendable to up to 122 stations on a single master
Connector	9 pin SUB-D connector	Available from HIMA

Table 88: Basic Technical Features of the RS-485 Transmission

The baud rate (transfer rate) can be set between 9.6 kBit/s and 12 MBit/s and applies to all stations connected to the bus.

The maximum line length depends on the selected baud rate. The details about the line length specified in Table 89 refer to cables of type A.

4.3.2 Range Depending on the Baud rate

Baud Rate	Range per Segment
9.6 kbps	1200 m
19.2 kbps	1200 m
93.75 kbps	1200 m
187.5 kbps	1000 m
500 kbps	400 m
1.5 Mbps	200 m
3 Mbps	100 m
6 Mbps	100 m
12 Mbps	100 m

Table 89: Range Depending on the Baud Rate

Note	The line length can be increased using bidirectional repeaters. A maximum of three repeaters may be switched between two stations. A line length of 4.8 km can be thus achieved. With time-critical applications, no more than 32 stations should be connected. For not time-critical applications, up to 126 subscribers are allowed (with three repeaters).

4.3.3 Bus Connection and Bus Termination

The inbound and the outbound data cable can be directly connected to the bus connector plug. In this way, branch lines are avoided and the bus connector plug can be plugged into and unplugged from the field bus device at any time without interrupting data traffic.

IEC 61158 recommends using a 9-pin sub D connector for PROFIBUS DP. Depending on the protection class of the field device, other connectors may also be used.

Figure 62 shows the pin assignment of the 9-pin sub D connector. The bus connection is designed as a socket on the field device.

The PROFIBUS DP's bus termination consists of a resistance combination ensuring a defined zero potential on the bus. The resistance combination is integrated in the PROFIBUS DP bus connector and can be activated using jumpers or switches.

The station at the end of the bus should also provide a 5 volt supply to pin 6.

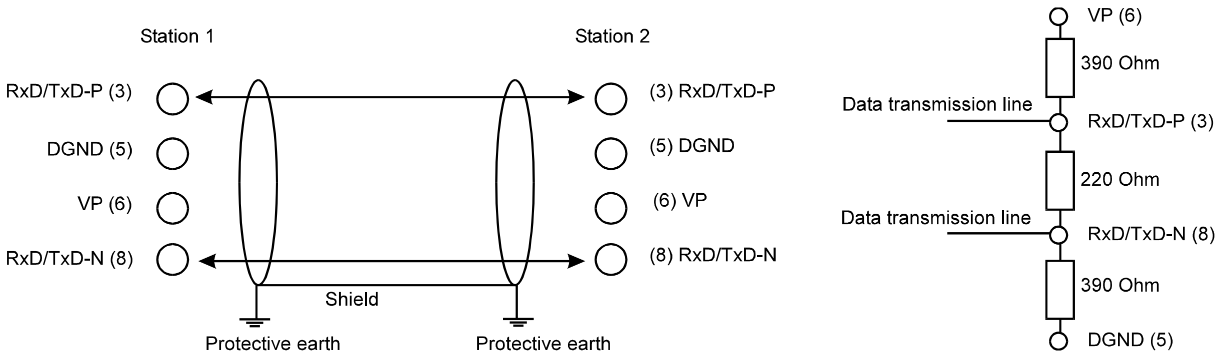


Figure 62: Bus Connection and Bus Termination, Pin Assignment of the Field Bus Interfaces

The signals on pins 3, 5, 6 and 8 are so-called mandatory signals and must always be available.

Pin	Signal	Description
1	-	Not used
2	-	Not used
3	RxD/TxD-P	Receiving/transmitting data plus (B wire)
4	RTS	Direction control for fiber-optic modems (TTL signal)
5	DGND	Data transfer potential (ground to 5V)
6	VP	Supply voltage of the termination resistors P (+5V)
7	-	Not used
8	RxD/TxD-N	Receiving-/transmitting data minus (A wire)
9	-	Not used

Table 90: Pin Assignment of the FB1 and FB2 Interfaces of the HIMatrix Controllers

4.3.4 PROFIBUS DP Bus Cable

The IEC 61158 specifies two bus cable types. Cables of type A may be used for all transfer rates up to 12 Mbps. Cables of type B are outdated and should not be used any longer.

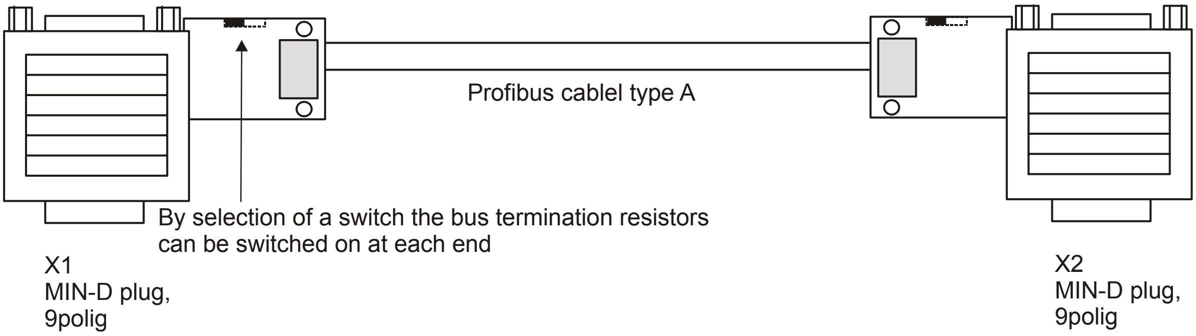


Figure 63: PROFIBUS DP Bus Cable, Type A with Bus Connector Plug

For the transmission medium, a shielded, symmetrical, two-wire line with the following characteristics should be used:

Characteristics	**Cable type A**	
Impendance	135 to 165	Ω
Capacitance	≤ 30	pf / m
Loop resistance	≤ 110	Ω / km
Wire gauge	> 0.64	mm
Wire cross-section	> 0.34	mm²

Table 91: Characteristics of the PROFIBUS DP RS-485 Cables of Type A

4.3.5 Bus Topology

All stations are connected to a common bus. Up to 32 stations can be connected per RS 485 segment. If more stations are required, further segments must be connected using repeaters. Each segment's beginning and end are provided with an active bus termination. The bus termination can usually be hooked up in the bus stations or in the bus connection plugs.

To ensure a smooth operation, both bus terminations must be power supplied.

The stations have a unique bus address in the range of 0..125. If more than 32 stations are utilized or if the network is extended, power amplifiers (repeaters) connecting the single bus segments must be used.

The maximum number of stations is reduced by 1 per each repeater used in a segment. That means that a maximum of 31 stations can be operated in one segment. In accordance with the standard, a total of three repeaters may be used, meaning that a maximum of 122 slaves can be connected to each serial interface on a class 1 master.

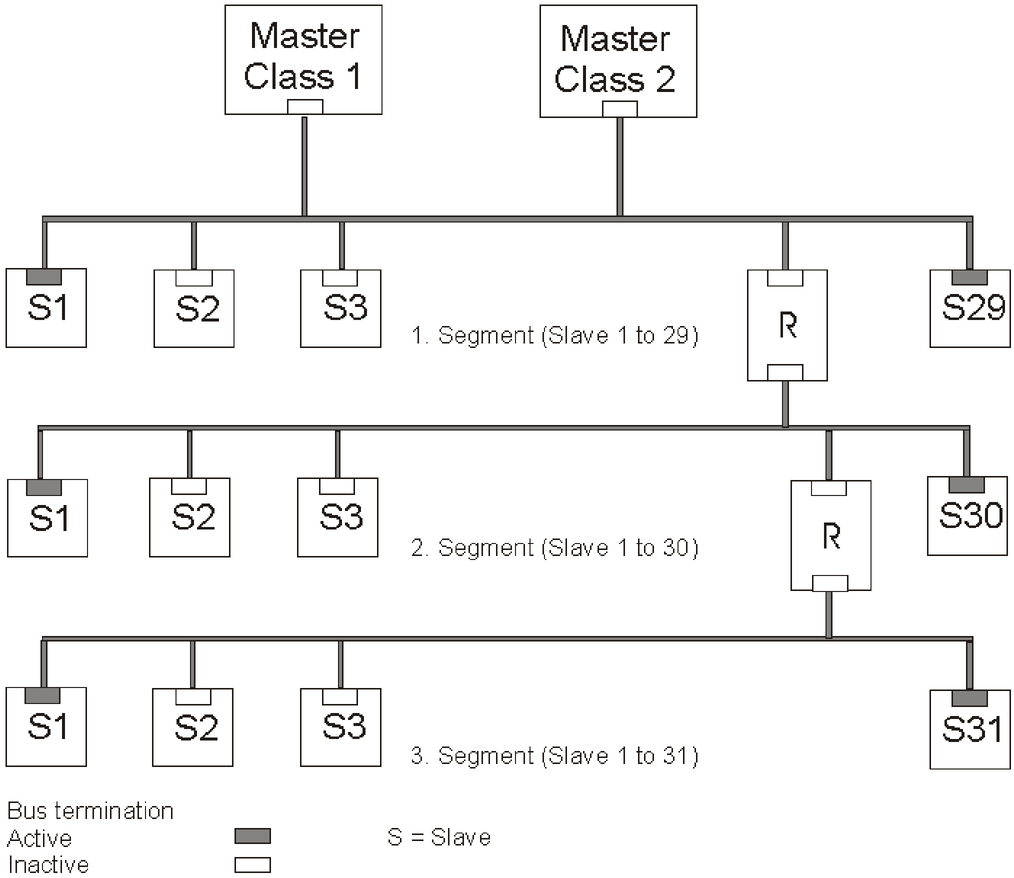


Figure 64: Bus Topology: Line Structure with Repeaters

The higher the number of bus slaves, the longer the time period before the slave information is available on the master. The more slaves are connected to the bus, the worse the system reaction time becomes.

Note	With transfer rates ≥ 1.5 Mbps, branch lines must absolutely be avoided. We recommend therefore only using appropriate bus connection plugs.

4.4 PROFIBUS DP Telegram Formats

Both of the following FDL bus access protocols are used for the PROFIBUS DP telegrams:

FDL Bus Access Protocol	Description
SDN (Send Data with No Acknowledge)	Send data to an individual, a group or to all stations without acknowledgement.
SRD (Send and Request Data with Reply)	Send data to an individual station and receive data from it with reply.

Table 92: FDL Bus Access Protocols used in PROFIBUS DP

The data transfer in PROFIBUS DP is character-based. Every transmitted character consists of eleven bits (a so-called UART character, i.e. eight data bits and three control bits).

1 Start bit	8 Data Bits (1 Byte)	1 Parity Bit	1 Stop Bit

Figure 65: UART -(Universal Asynchronous Receiver/ Transmitter) Character

The UART character is transferred block by block in so-called telegrams, e.g. a chain of UART characters. The following table describes the functional bytes for the used telegrams.

4.4.1 Functional Bytes for the PROFIBUS DP Telegrams

Function	Description	
SD1	Start Delimiter 1 (16#10)	Each of the four telegram formats (SD1 to SD4) begins with a start byte.
SD2	Start Delimiter 2 (16#68)	
SD3	Start Delimiter 3 (16# A2)	
SD4	Start Delimiter 4 (16# DC)	
FCS	Frame Check Sequence	Contain the telegram checksum
DA	Destination Address	Receiver station address
SA	Source Address	Sender station address
FC	Function Control	Control byte
L	Length Field	Number of data bytes
Lr	Length Field, redundant	Number of data bytes
ED	End Delimiter (16#16)	End of the telegram
SC	Single Char (16#E5)	Single character for acknowledgement purposes

Table 93: Description of the Functional Bytes of the PROFIBUS DP Telegrams

4.4.2 PROFIBUS DP Telegrams Used in the HIMatrix Controllers

Telegram with Fixed Length (L = 3), without Data

SD1	DA	SA	FC	FCS	ED
10H	XX	XX	XXH	X	16H

Telegram with Variable Length (L = 4 to 249), with up to 246 Data Bytes

SD2	L	Lr	SD2	DA	SA	FC	Data[1...246]	FCS	ED
68H	X	X	68H	XX	XX	X	X....	X	16H

Telegram with Fixed Length (L = 11), with Eight Data Bytes

SD3	DA	SA	FC	Data[8]	FCS	ED
A2H	XX	XX	X	X….	X	16H

Token Telegram

SD4	DA	SA

Confirmation without Data (Acknowledgement)

SSC SC

4.4.3 Possible Station Addresses in the Telegram Fields DA and SA

Addresses	Description
0 .. 125	Station addresses
126	Standard address for bus stations whose final address is assigned dynamically
127	Broad or multicast address

Table 94: Possible Station Addresses in the Telegram Fields DA and SA

4.4.4 PROFIBUS DP Telegram Mechanisms for Data Protection

The PROFIBUS DP telegrams ensure high transfer reliability, specified in the IEC 870-5-1 international standard. Faulty telegrams are automatically repeated up to seven times (see Table 5 *Max. Num. Resend*).

The following errors are detected:

- Start and end delimiter errors
- Incorrect telegram length
- Incorrect character format
- Frame control byte errors
- Protocol errors

An error leads to the following station reactions:

- Erroneously received function calls are generally not executed. The function call must be repeated.
- The function call must also be repeated if a faulty response is received.

4.4.5 PROFIBUS DP Bus Access Method

The bus access method provides a defined time window to every station. The station must perform its communication task within this time period.

4.4.5.1 Master/Slave Protocol

The bus assignment between a PROFIBUS DP master and a PROFIBUS DP slave is ensured by the master/slave method.

An active PROFIBUS DP master communicates with passive PROFIBUS DP slaves.
The PROFIBUS DP master with the token is authorized to send and may communicate with the PROFIBUS DP slaves assigned to it. The master assigns the bus to a slave for a certain time and within this time period the slave must respond.

4.4.5.2 Token Protocol

The bus assignment between automation devices (Class 1 masters) and/or programming devices (Class 2 masters) is ensured using the *token passing*.

All PROFIBUS DP masters connected to a common bus form a token ring. The active PROFIBUS DP master assumes the master function on the bus for the duration of its possession of the token.

In a token ring, the PROFIBUS DP masters are organized in ascending order as dictated by their respective station addresses. The token is passed on in this order until it is received by the PROFIBUS DP master with the highest station address.

This master passes the token on to the master with the lowest station address to close the token ring.

The token rotation time corresponds to one token cycle through all PROFIBUS DP masters. The *target rotation time* (*Ttr*) is the maximum time allowed for a token cycle.

The list of all active bus stations (LAS) contains all masters currently active on the bus. It is used for adding new masters and removing faulty ones without disturbing communications on the bus.

The Gap

The gap is the address range from an active station (master) to the next active station, ordered by ascending bus address.

The gap of the last active station (HAS) also contains the range from zero to the first active station (m).

HSA (Highest Active Station) is the highest address to be considered.

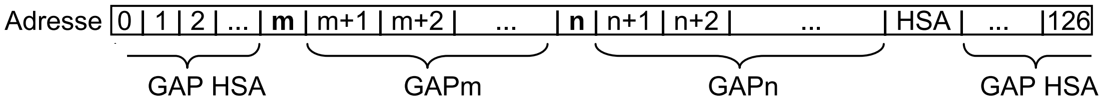


Figure 66: Master's Gaps within the Token Ring

Admitting a Master into the Token Ring

Each active PROFIBUS DP master maintains its gap list by periodically sending an *FDL_Status_Req* to every single address within its gap in ascending order. If an active or a passive master is reached at one of these addresses, the master answers.

If a new PROFIBUS DP master is connected to the bus, it remains passive. It builds up its list of all active stations (LAS) until it is contacted via the *FDL_Status_Req* by a PROFIBUS DP master located within its gap. If it acknowledges that it is *Ready* for the token ring, it is accepted in the token ring. PROFIBUS DP masters with an address in excess of the HSA are not addressed and thus not admitted in to the token ring.

If no data traffic is present on the bus after power is switched on or after a token loss, the PROFIBUS DP masters wait for a predefined period of time. The PROFIBUS DP master with the lowest address is the first to become active. It sends the token to itself two times, thus signaling the other stations that it is the only master in possession of the token. The other PROFIBUS DP masters then receive the token via token passing.

4.5 Isochronous PROFIBUS DP Cycle (DPV2 and higher)

The PROFIBUS DP cycle consists of two telegram phases: a fixed and cyclical phase and an event-driven and acyclic phase.

The acyclic phase can prolongate the corresponding PROFIBUS DP cycle. This effect is not wanted in specific applications and areas, such as drive technology.

To achieve a constant cycle time (t_{const}), the isochron mode is activated in the master such that the parameter *Min. Slave Interval [ms]* defines the constant cycle time (t_{const}). Parameterized in this way, the isochronous PROFIBUS DP cycle offers clock accuracy with a difference of < 10 µs.

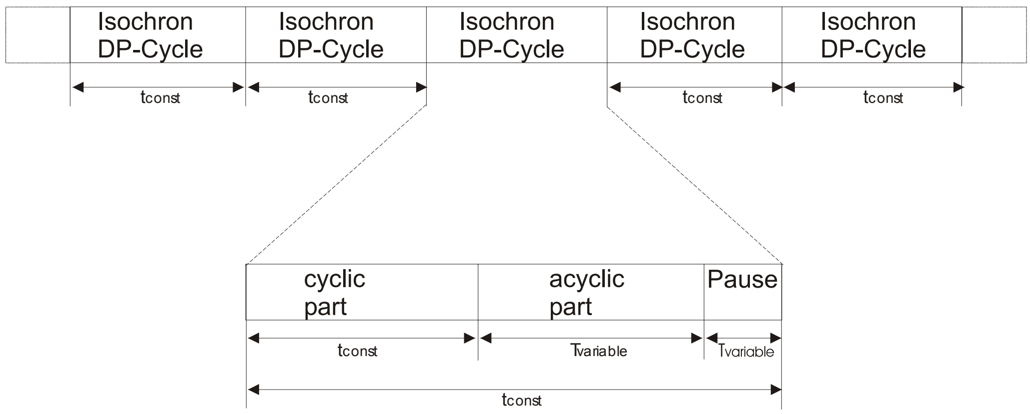


Figure 67: PROFIBUS DP Cycle

To determine the cyclical phase, the minimum target token rotation time must be calculated.

Further, a sufficiently large time interval (typically two to three times the minimum target token rotation time *Ttr*) must be reserved for the acyclic phase. If the reserved time is not needed, a break is taken prior to starting the next cycle to ensure the cycle time remains constant (see also chapter 1.4.5.1 Calculating the Target Token Rotation Time *Ttr*).

Note	The master is parameterized entering the DP cycle time determined by the user into *Min. Slave Interval [ms]*. To operate in the *Isochron mode*, one of the two parameters *Isochron Mode Sync* or *Isochron mode Freeze* must be activated in the master. Only one master may simultaneously operate in the isochron mode on the bus. Additional masters are not allowed.

4.6 Cyclical PROFIBUS DP Cycle (DPV0 and higher)

Cyclical master-slave communication between a class 1 master and a slave. This is the normal PROFIBUS DP connection for the cyclical exchange of I/O data.

The master-slave communication consists of the following phases:

- connection establishment
- data exchange and
- connection termination.

While establishing the connection, the master requests a diagnostic telegram from the slave to determine if the slave is ready. If this is the case, the master sends a parameter telegram to the slave with *lock bit* set. The slave acknowledges it with a SC and is blocked from now on for other masters.
The master sends the configuration data to the slave. The slave acknowledges it with a SC. Once the connection is established, the master requests diagnostics from the slave to ensure that the parameterization and configuration are correct. If no error is detected, the connection is maintained.

Parameterization data with *unlock bit* set are sent to terminate the communication. The slave acknowledges it again with a SC.

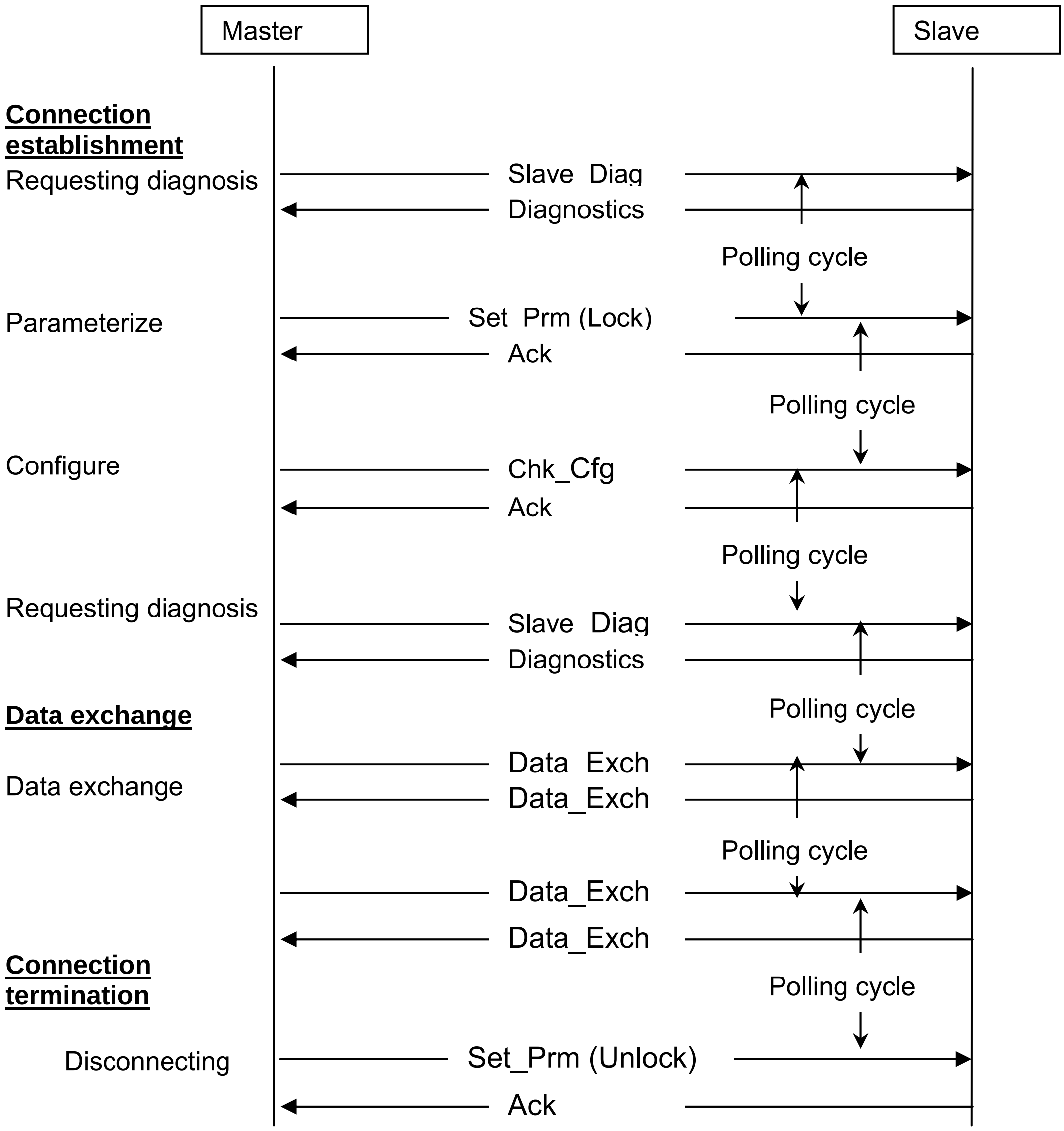


Figure 68: Master-Slave Communication

4.6.1 Polling Cycle

The PROFIBUS DP master communicates with its slaves within polling cycles. Each polling cycle consists of a cyclic phase, an optional acyclic phase, the maintenance of the gap list and token passing.

- A polling cycle starts with the master communicating its state to all slaves.
- During the cyclic phase, exactly one telegram (either data, parameterization, configuration or diagnostics) is requested from each slave.
- Aftewards, a single address *Station_x* within *Master_1*'s gap is polled per *FDL_Status_Req* (a telegram for recognizing a new master).
- Then, the acyclic communication phase takes place.
- At the end of a polling cycle, the token is passed to the next master. If no other master is active on the bus, the master passes the token to itself.

Time parameters such as *Tid, Tsl* etc. are explained in the next chapter.

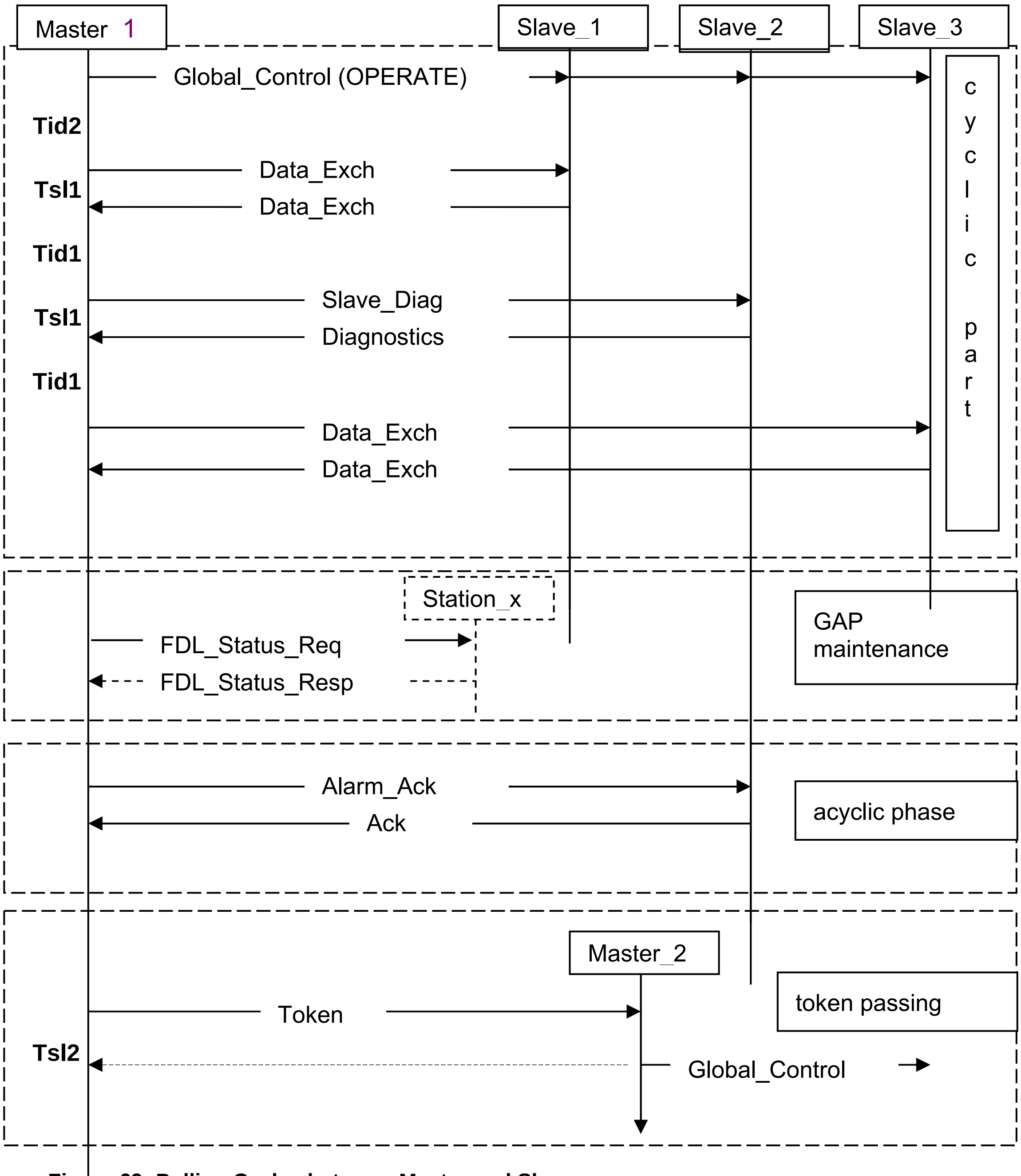


Figure 69: Polling Cycles between Master and Slaves

4.6.2 Time Parameters for the Polling Cycle

4.6.2.1 Idle Time (Tid)

The Idle Time is the period that elapses between two frames or two unacknowledged packets within a station.

Tid ≥ Tsyn + Tsm

Two idle times are distinguished:

Tid1

Time period between the reception of the acknowledgement or token frame and the transmission of the next frame.

Tid1 = max(Tsyn+Tsm, MinTsdr, Tsdi)

Tid2

Time period between the transmission of an unacknowledged packet (SDN, see Table 92) and the transmission of the next packet.

Tid2 = max(Tsyn+Tsm, MaxTsdr)

4.6.2.2 Slot Time (Tsl)

Maximum time period after sending a request that a station waits to receive the first character (11 bits) of a response.

Tsl = max(Tsl1, Tsl2)

Two (theoretical) times are distinguished:

Tsl1

Time period after a request that the sending station waits for the response packet to start.

Tsl1 = MaxTsdr + 2 * Ttd + Tsm + 11

Tsl2

Time period after the token passing which the sending station waits for the next packet to start. This way, the token sender recognizes that the token has arrived.

Tsl2 = Tid1 + 2 * Ttd + Tsm + 11

4.6.2.3 Synchronization Time (Tsyn)

Minimum time period that the transmission medium must idle (binary “1”) before a new request or token frame may begin.

Tsyn = 33 tBit

4.6.2.4 Station Delay Time (Tsdx)

Time that may elapse between the reception of the last bit of a packet and the transmission of the first bit of the next packet (from the transmission medium’s perspective).

Several variants exist:

Station Delay of Initiators (Tsdi)

Time that may elapse in the initiator (master) between receiving the acknowledgment packet and transmitting the next request or token packet.

Station Delay of Responders (Tsdr)

MinTsdr ≤ Tsdr ≤ MaxTsdr

Time that may elapse in the responder (slave) between the reception of a request packet and the transmission of the response packet.

Tsdr is associated with two boundary values:

MinTsdr	Minimum Station Delay of Responders Minimum time that must elapse
MaxTsdr	Maximum Station Delay of Responders Maximum time that may elapse

4.6.2.5 Quiet Time (Tqui)

Modulator dying time during which the sending and receiving options should be deactivated. If necessary, i.e. if the following condition is not met, the MinTsdr must be increased;

Tqui < MinTsdr

4.6.2.6 Safety Margin (Tsm)

Safety charge

Tsm = 2 + 2 * Tset + Tqui

4.6.2.7 Time-Out Time (Tto)

Time period in which the bus activity is controlled. If this time elapses without bus activity, the bus is considered as “dead” (token loss).

Tto = 6 * Tsl + 2 * *n* * Tsl

For active stations, the bus address is *n*, for passive stations *n = 130*.

4.6.2.8 Further Time Parameters for the Polling Cycle

Setup Time (Tset):

Time required for reacting to an event (interrupt handler).

Transmission Delay Time (Ttd):

Time required for transfer on to the medium, including repeaters

Target Rotation Time (Ttr):

Time configured for a token cycle.

Real Rotation Time (Trr):

Actual token rotation time.

Min Slave Interval (Msi):

Minimum time between two cyclical requests of a slave.

5 Literature

[1] First Steps ELOP II Factory
HIMA GmbH Brühl, 2006: HI 800 006

[2] Online Help in ELOP II Factory Hardware Management
HIMA GmbH Brühl, 2006

[3] HIMA HIMatrix Engineering Manual
HIMA GmbH Brühl, 2006: HI 800 101

[4] HIMA Fieldbus Communication Modules
HIMA GmbH Brühl, 2004: HI 800 129

[5] Manfred Popp: PROFIBUS DP/DP-V1-Grundlagen, Tipps und Tricks für Anwender,
Hüthig Verlag Heidelberg, 2000 ISBN 3-7785-2781-9

[6] PROFIBUS Intenational: PROFIBUS Technology and Application,
October 2002

[7] PROFIBUS Intenational: PROFIBUS Guideline No. 2.182
Version 1.2, Juli 2001
http://www.PROFIBUS.com/

[8] EN 50170 Band 2/3 PROFIBUS 50179 A1 + A3

HIMA
...the safe decision.

HIMA Paul Hildebrandt GmbH
Industrial Automation
Postfach 1261 • 68777 Brühl
Phone: (06202) 709-0 • Fax: (06202) 709-107
E-mail: info@hima.com • Internet: www.hima.de

(0908)