



# **Drive System SD2**

## **Profibus Connection via Anybus Communicator**

Description of Profibus to support the serial DNC protocol



## Copyright

Copyright © 2012 SIEB & MEYER AG.

All rights reserved.

This manual or extracts thereof may only be copied with the explicit authorization of SIEB & MEYER AG.

## Trademarks

All product, font and company names mentioned in this manual may be trademarks or registered trademarks of their respective companies.

## SIEB & MEYER worldwide

For questions regarding our products and technical problems please contact us.

SIEB & MEYER AG  
Auf dem Schmaarkamp 21  
D 21339 Lüneburg  
Germany

Phone: +49 4131 203 0  
Fax: +49 4131 203 2000  
[support@sieb-meyer.de](mailto:support@sieb-meyer.de)  
<http://www.sieb-meyer.com>

SIEB & MEYER Asia Co. Ltd.  
4 Fl, No. 532, Sec. 1  
Min-Sheng N. Road  
Kwei-Shan Hsiang  
333 Tao-Yuan Hsien  
Taiwan

Phone: +886 3 311 5560  
Fax: +886 3 322 1224  
[smasia@ms42.hinet.net](mailto:smasia@ms42.hinet.net)  
<http://www.sieb-meyer.com>

SIEB & MEYER Shenzhen Trading Co. Ltd.  
1st floor, B room of D1 block, DongNan GongMao  
Building  
Dongjiaotou Shekou, Houhai Ave, Nanshan District  
Shenzhen City, 518067  
P.R. China

Phone: +86 755 2681 1417 / +86 755 2681 2487  
Fax: +86 755 2681 2967  
[sma-china@umail.hinet.net](mailto:sma-china@umail.hinet.net)  
<http://www.sieb-meyer.com>

SIEB & MEYER USA, LLC  
3975 Port Union Road  
Fairfield, OH 45014  
USA

Phone: +1 513 563 0860  
Fax: +1 513 563 7576  
[info@sieb-meyerusa.com](mailto:info@sieb-meyerusa.com)  
<http://www.sieb-meyer.com>

Description of the Program S7_DNC	1
Program Content	2
Program Control	3
Symbols	4
Hardware Configuration in the PLC	5
Anybus Communicator by HMS	6
Hardware Description	7
Error Detection	8
Related Documents	9



<b>1</b>	<b>Description of the Program S7_DNC .....</b>	<b><a href="#">7</a></b>
1.1	Installation .....	<a href="#">7</a>
<b>2</b>	<b>Program Content .....</b>	<b><a href="#">9</a></b>
2.1	Organization block .....	<a href="#">9</a>
2.2	Function Blocks .....	<a href="#">9</a>
2.3	Global Data Blocks .....	<a href="#">9</a>
2.4	Instance Data Blocks .....	<a href="#">9</a>
2.5	User-defined Data Types .....	<a href="#">9</a>
2.6	S7 System Functions and S7 System Function Blocks .....	<a href="#">10</a>
<b>3</b>	<b>Program Control .....</b>	<b><a href="#">11</a></b>
3.1	Organization Block .....	<a href="#">11</a>
3.1.1	Organization block MAIN_DNC .....	<a href="#">11</a>
3.1.2	Principle PLC Sequence .....	<a href="#">12</a>
3.2	Function Blocks .....	<a href="#">13</a>
3.2.1	Function blockRD_DNC .....	<a href="#">13</a>
3.2.1.1	General Description .....	<a href="#">13</a>
3.2.1.2	Calling the Function Block .....	<a href="#">13</a>
3.2.1.3	Inputs and Outputs .....	<a href="#">13</a>
3.2.2	Function blockWR_DNC .....	<a href="#">13</a>
3.2.2.1	General Description .....	<a href="#">13</a>
3.2.2.2	Calling the Function Block .....	<a href="#">14</a>
3.2.2.3	Inputs and Outputs .....	<a href="#">14</a>
3.2.3	Function blockINIT_DNC .....	<a href="#">14</a>
3.2.3.1	General Description .....	<a href="#">14</a>
3.2.3.2	Calling the Function Block .....	<a href="#">14</a>
3.2.3.3	Inputs and Outputs .....	<a href="#">15</a>
3.3	Global Data Blocks .....	<a href="#">15</a>
3.3.1	Data BlockDB_ERRORCODE .....	<a href="#">15</a>
3.3.2	Data BlockDB_GLOBALDATA .....	<a href="#">15</a>
3.4	Instance Data Blocks .....	<a href="#">15</a>
3.5	User-defined Data Types .....	<a href="#">16</a>
3.5.1	User-defined Data TypeAXIS_DATA_IN .....	<a href="#">16</a>
3.5.2	User-defined Data TypeAXIS_DATA_OUT .....	<a href="#">17</a>
3.5.3	User-defined Data TypeAXIS_REF .....	<a href="#">18</a>
3.6	System Functions and Blocks .....	<a href="#">18</a>
<b>4</b>	<b>Symbols .....</b>	<b><a href="#">19</a></b>
4.1	Symbol Table .....	<a href="#">19</a>
<b>5</b>	<b>Hardware Configuration in the PLC .....</b>	<b><a href="#">21</a></b>
5.1	GSD File .....	<a href="#">21</a>
5.2	Settings in the Hardware Configuration .....	<a href="#">21</a>
5.3	Defining the Addresses in the Modules .....	<a href="#">22</a>
<b>6</b>	<b>Anybus Communicator by HMS .....</b>	<b><a href="#">23</a></b>
6.1	Telegram Structure .....	<a href="#">23</a>
<b>7</b>	<b>Hardware Description .....</b>	<b><a href="#">25</a></b>
7.1	Connecting the Devices .....	<a href="#">25</a>

7.2	Anybus Communicator by HMS .....	<a href="#">25</a>
7.2.1	Connectors, Switches and Indicators .....	<a href="#">26</a>
8	Error Detection .....	<a href="#">29</a>
8.1	Errors in S7 PLC .....	<a href="#">29</a>
8.2	Error in Anybus Communicator .....	<a href="#">29</a>
8.3	Error in Drive Control SD2S .....	<a href="#">30</a>
9	Related Documents .....	<a href="#">31</a>

# 1 Description of the Program S7\_DNC

1

The program S7\_DNC allows connecting an SD2S module by SIEB & MEYER to a SIMATIC S7 CPU with Profibus interface via Anybus Communicator provided by HMS (Profibus-DP Serial Gateway).

The program has been created using the engineering tool S7-SCL V5.1. All data blocks necessary for the communication are involved in the program. In addition the program provides three system blocks of the S7, which are used during the process.

In order to use the program for the communication between S7 and the SD2S devices by SIEB & MEYER, you need the following hardware and software components:

- ▶ SIMATIC STEP 7 starting from version 4.02
- ▶ SIMATIC S7 300(/400) starting from CPU 315-2DP
- ▶ HMS Anybus Communicator ABC-PDP-B (AB7209-B)
- ▶ SIEB & MEYER SD2S

The used communication protocol is based on the custom-designed drive profile DS402 by CANopen.

- ▶ 10 byte Profibus user data, DNC header and checksum are generated in the adapter

## 1.1 Installation

- ◆ At first create a directory for the application program.  
Example: D:\Data\Siemens\SM\_AG\S7Prog
- ◆ Copy the ZIP file of the program (e.g. "S7\_DNC\_V100.zip") into the created directory.
- ◆ Unpack the ZIP file.
- ◆ Start the SIMATIC Manager.
- ◆ Select the menu "File → Open".
- ◆ Click the button "Browse" to select the created directory and open the project "S7\_DNC".





## 2 Program Content

The program S7\_DNC contains the following objects as SCL sources. As SCL sources they can be compiled to function and data blocks with the desired block number via the symbol table.

2

### 2.1 Organization block

Symbolic name	Description
MAIN_DNC	Example main program

### 2.2 Function Blocks

Symbolic name	Description
INIT_DNC	Initializing the bus data
RD_DNC	Analyzing the bus data towards the axis structure
WR_DNC	Compiling the bus data from axis structure

### 2.3 Global Data Blocks

Symbolic name	Description
DB_ERRORCODE	Error codes
DB_GLOBALDATA	Global used variables

### 2.4 Instance Data Blocks

Symbolic name	Description
DB_INIT_DNC	Data block for data initializing of axis 0 (INIT_DNC)
DB_HEARTBEAT	Data block for timer (TON)
DB_RD_DNC	Data block for read function (RD_DNC)
DB_WR_DNC	Data block for write function (WR_DNC)

### 2.5 User-defined Data Types

Symbolic name	Description
AXIS_DATA_IN	Data structure of the received data from Profibus
AXIS_DATA_OUT	Data structure of the transmitted data to Profibus
AXIS_REF	Data structure of the axis data

## 2.6 S7 System Functions and S7 System Function Blocks

Function block	Symbolic name	Description
SFB4	TON	Timer
SFC14	DPRD_DAT	Read DP slaves
SFC15	DPWR_DAT	Write DP slave
SFC20	BLKMOV	Copy variables block by block

## 3 Program Control

This chapter provides information on the individual blocks and their functions.

### 3.1 Organization Block

3

#### 3.1.1 Organization block MAIN\_DNC

The program is controlled via the organization block MAIN\_DNC. This block gives an example in which way the information is transmitted. The data from the bus are read, allocated to the axes, processed and written to the bus again. The individual program steps are processed by the function blocks.

Please note that the data from the Profibus module first must be copied into the axis structure via SFC14. Using the function block RD\_DNC the data is then analyzed in the axis structure. After analyzing is complete the data is processed using the function block WR\_DNC and the adjacent inputs. Then the data is transferred to the axis structure and copied into the Profibus module via SFC15.

The the cycle starts again: load → process → output.

Before the proper sequential control comes into operation the data must be initialized via the function block INIT\_DNC once when the program is started or after a fault.

### 3.1.2 Principle PLC Sequence

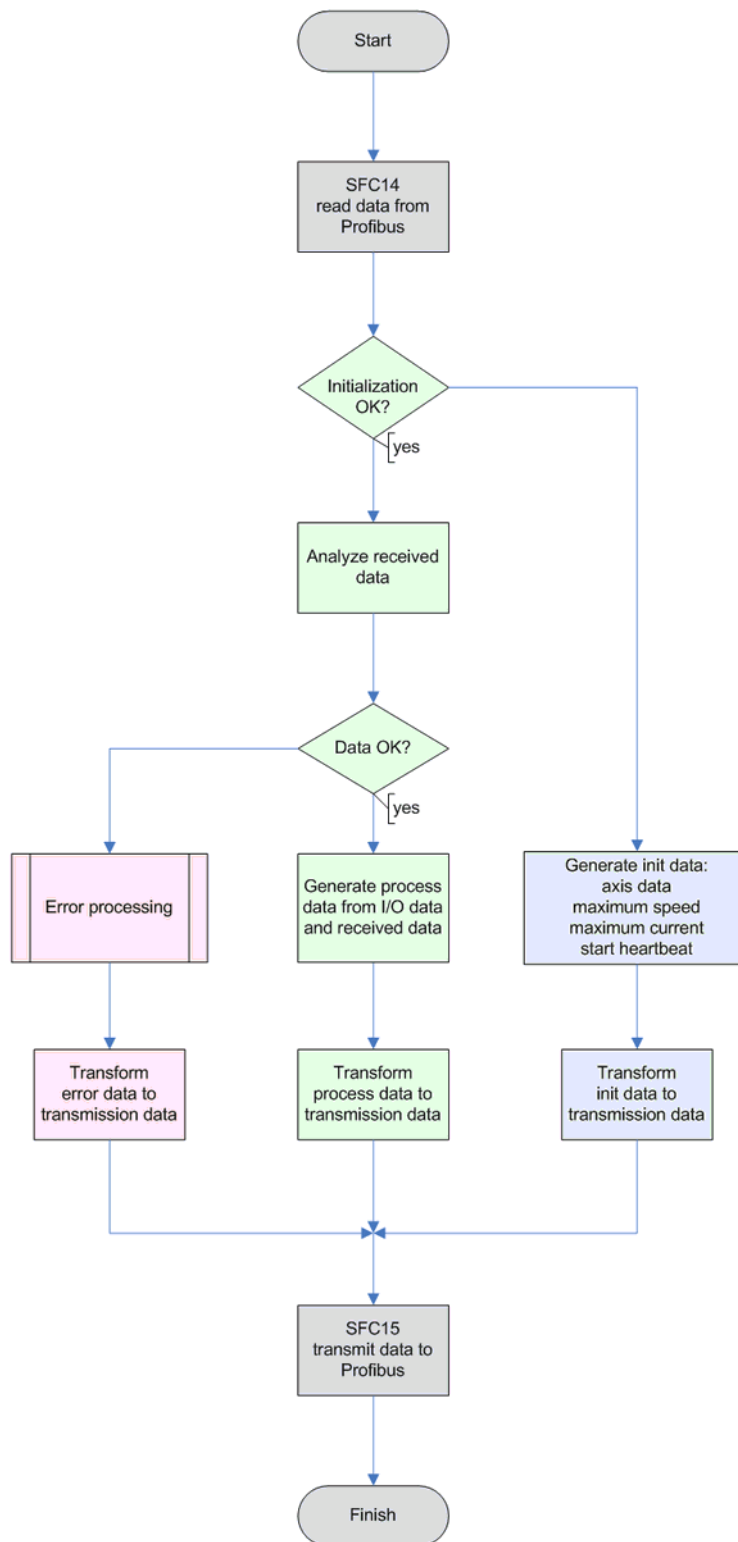


Fig. 1: Program sequence

## 3.2 Function Blocks

### 3.2.1 Function blockRD\_DNC

#### 3.2.1.1 General Description

The function block RD\_DNC processes the data of the SD2S DP slave. The data of the SD2S device are read out of the bus structure by the function block. Afterwards the individual information is filtered from the telegram.

The information is deposited in the instance data block and in the axis structure Axis. Thus, it is globally available in the whole application program. In order to process the data, parameters must be passed to the function block (FB). Then the information of the device is available at the outputs of the block.

The outputs Act\_Velocity, Act\_AxisError and Act\_Current return the current process data.

The values of the actual velocity and the actual current are transmitted in a standardized form in the DNC protocol. The DNC protocol is represented in the data structure AXIS\_DATA\_IN. The range of values is between -16383 and +16383, corresponding to -maximum value to +maximum value.

#### 3.2.1.2 Calling the Function Block

RD\_DNC.DB\_RD\_DNC 0

(Axis := DB\_GLOBALDATA.SD\_Axis[0]);

#### 3.2.1.3 Inputs and Outputs

Parameter	In / Out	Type	Description
Axis	in_out	AXIS_REF	Data structure of the slave
ErrorID	out	WORD	Error number of the block
Act_Velocity	out	INT	Actual velocity
Act_AxisError	out	WORD	Axis error
Act_Current	out	INT	Actual current

When the function block returns an error value >0x8000, this is an error code of the internal used S7 block BLKMOV by SIEMENS.

### 3.2.2 Function blockWR\_DNC

#### 3.2.2.1 General Description

The function block WR\_DNC compiles a reference value telegram from the data in the axis structure Axis and writes the new parameters and control bits into the bus structure. In order to write into a DP slave, parameters must be passed to the function block (FB). At the outputs of the block the information of the respective module is available.

The reference values of the process data are transmitted in Fault\_Reset, Regulator\_On, Start and Velocity.

The value of the speed setting is transmitted in a standardized form in the DNC protocol. The DNC protocol is represented in the data structure AXIS\_DATA\_OUT. The range of values is between -16383 and +16383, corresponding to -maximum value to +maximum value.

3

### 3.2.2.2 Calling the Function Block

```
// calling the reference value telegram for process data
```

```
WR_DNC.DB_WR_DNC
```

```
(NMT_Command := 1, Fault_Reset := IX0.7, Regulator_On := IX0.0,
```

```
Start := IX0.1, QuickStop := IX0.2,
```

```
Velocity := INT_TO_REAL (WORD_TO_INT (BYTE_TO_WORD (IB4))) * 400.0 * Direction,
```

```
Axis := DB_GLOBALDATA.SD_Axis);
```

### 3.2.2.3 Inputs and Outputs

Parameter	In / Out	Type	Description
NMT_Command	in	INT	<ul style="list-style-type: none"> <li>0 = no reaction</li> <li>1 = operational</li> <li>2 = pre-operational</li> </ul>
PDO_Select	in	INT	Presently always = 0
Fault_Reset	in	BOOL	Reset of the axis error
Regulator_On	in	BOOL	Switching on the output stage
Start	in	BOOL	Enable movements
Velocity	in	INT	Speed setting
Axis	in_out	INT	Data structure of the axis
ErrorID	out	WORD	Error code

When the function block returns an error value >0x8000, this is an error code of the internal used S7 block BLKMOV by SIEMENS.

## 3.2.3 Function block INIT\_DNC

### 3.2.3.1 General Description

Using the function block INIT\_DNC the command bits are initialized and deposited in the axis structure. The reference values are initialized with the value zero.

Afterwards the initializing data must be sent to the axis once.

### 3.2.3.2 Calling the Function Block

```
INIT_DNC.DB_ INIT_DNC (Axis := DB_GLOBALDATA.SD_Axis[0]);
```

### 3.2.3.3 Inputs and Outputs

Parameter	In / Out	Type	Description
Axis	in_out	AXIS_REF	Data structure of the slave

## 3.3 Global Data Blocks

3

### 3.3.1 Data BlockDB\_ERRORCODE

The global data block DB\_ERRORCODE contains the structure of the error codes that are returned by the function block.

The error codes have the following meanings:

Error code	Value	Meaning
ERR_None	0	No error
ERR_WrongState	1	The axis is in the wrong device status for the upcoming command.
ERR_Parameter	2	Parameter error
ERR_NoRemote	3	The axis is not in the remote mode.
ERR_Heartbeat	4	Error in data communication
ERR_Checksum	5	Error in checksum calculation

### 3.3.2 Data BlockDB\_GLOBALDATA

The global data block DB\_GLOBALDATA contains the structure of the program-relevant data that must be adapted to the particular project.

The following variables are used:

Variable	Meaning
ErrorID	Global error variable
prog_init	<ul style="list-style-type: none"> <li>0 = program data not initialized yet</li> <li>1 = program data initialized</li> </ul>
Heartbeat_Time	Communication monitoring with timeout after 500 ms
DP_Base_Address	Base address Profibus
DP_Axis_present	Determines the existence of the modules compiled in the project
SD_Axis	Array of the data structure of the projected module

## 3.4 Instance Data Blocks

A separate set of instance data blocks is required for each axis. Each instance data block contains the input parameters, the output parameters, the in-out parameters and the static variables of the respective function block.

## 3.5 User-defined Data Types

The user-defined data types describe the data structure AXIS\_REF of each axis and the input and output data of the Profibus module.

The data can be accessed via the global variables.

The structure of AXIS\_REF is based on the standards by the PLCopen group. The structures of the bus parameters are based on the CANopen protocol. A description of the telegram structure is part of the documents "SD2\_CAN-Connection.pdf" and "SD2\_DeviceControl.pdf".

### 3.5.1 User-defined Data Type AXIS\_DATA\_IN

The user-defined data type AXIS\_DATA\_IN contains the data structure of the received data of a module from Profibus. The actual value setting is indicated by an array made up of 6 bytes that are defined dependent on the operating mode.

Name	Type	Meaning
<b>Header byte 0..1</b>		
NMT_Cmd0	BOOL	NMT command bits 00: no reaction; 01: operational 10: pre-operational; 11: no reaction
NMT_Cmd1	BOOL	
Toggle bit	BOOL	Heartbeat
PDO_Select0	BOOL	PDO selection bits
PDO_Select1	BOOL	
PDO_Select2	BOOL	
PDO_res06	BOOL	Reserved
PDO_res07	BOOL	Reserved
PDO_res10	BYTE	Reserved
<b>Status word Byte 2..3</b>		
ReadyToSwitchOn	BOOL	Ready to be switched on
SwitchedOn	BOOL	Switched on
OperationEnabled	BOOL	Operation is enabled
Fault	BOOL	Fault occurred
VoltageEnabled	BOOL	Voltage is enabled
QuickStop	BOOL	Quick stop
SwitchOnDisabled	BOOL	Switch on is disabled
Warning	BOOL	Warning
smres010	BOOL	Reserved
Remote	BOOL	Remote mode
TargetReached	BOOL	Reference value reached
InternalLimitActive	BOOL	Internal limit reached
OperationModeSpecific0	BOOL	Dependent on the operating mode
OperationModeSpecific1	BOOL	Dependent on the operating mode
IstwertTelegrammKennung1	BOOL	Code of actual value telegram
IstwertTelegrammKennung0	BOOL	Code of actual value telegram
<b>Actual values byte 4..9</b>		
Act_Position	DINT	Actual position
Act_Velocity	INT	Actual velocity



Name	Type	Meaning
Act_Current	INT	actual current
<b>Service channel byte 10..15</b>		
Act_Value[0] to Act_Value[5]	6 BYTE	Current actual values in the velocity mode: <ul style="list-style-type: none"> <li>▸ actual velocity</li> <li>▸ actual error status</li> <li>▸ actual current</li> </ul>

### 3.5.2 User-defined Data Type AXIS\_DATA\_OUT

3

The user-defined data type AXIS\_DATA\_OUT contains the data structure of the transmission data of a module to the Profibus. The reference value setting is indicated by an array made up of 6 bytes that are defined dependent on the operating mode.

Name	Type	Meaning
<b>Header byte 0..1</b>		
NMT_Cmd0	BOOL	NMT command bits 00: no reaction; 01: operational 10: pre-operational; 11: no reaction
NMT_Cmd1	BOOL	
Toggle bit	BOOL	Heartbeat
PDO_Select0	BOOL	PDO selection bits
PDO_Select1	BOOL	
PDO_Select2	BOOL	
PDO_res06	BOOL	Reserved
PDO_res07	BOOL	Reserved
PDO_res10	BYTE	Reserved
<b>Control word Byte 2..3</b>		
SwitchOn	BOOL	Switch on power unit
EnableVoltage	BOOL	Enable voltage at the power unit
QuickStop	BOOL	Quick stop
EnableOperation	BOOL	Operation enable
Mode0	BOOL	Operation mode bit 0
Mode1	BOOL	Operation mode bit 1
Mode2	BOOL	Operation mode bit 2
FaultReset	BOOL	Error reset
Hold	BOOL	Hold
res011	BOOL	Reserved
res012	BOOL	Reserved
smres013	BOOL	Reserved
smres014	BOOL	Reserved
smres015	BOOL	Reserved
SollwertTelegrammID0	BOOL	Code of reference value telegram
SollwertTelegrammID1	BOOL	Code of reference value telegram
<b>Byte 4..9</b>		
Act_Value[0] to Act_Value[5]	6 BYTE	Reference value setting in the velocity mode: <ul style="list-style-type: none"> <li>▸ reference speed</li> <li>▸ reserve</li> </ul>

### 3.5.3 User-defined Data Type AXIS\_REF

The user-defined data type AXIS\_REF contains the data structure of the axis data based on the PLCopen standard.

Name	Type	Meaning
AxisNo	WORD	Axis number
AxisName	STRING	Name of axis
init_ok	BOOL	TRUE: all parameters are initialized
State_NotReadyToSwitchOn	BOOL	Module status that is made up of the status bits 0 ... 3 and 5 ... 6 of the bus protocol.  There is always only one module state set, the others are not set.
State_SwitchOnDisabled	BOOL	
State_ReadyToSwitchOn	BOOL	
State_SwitchedOn	BOOL	
State_OperationEnabled	BOOL	
State_QuickStopActive	BOOL	
State_FaultReactActive	BOOL	
State_Fault	BOOL	
VelocityMode_OutData	STRUCT spt_velocity: INT; res0: INT; res1: INT; END_STRUCT	Reference values in the velocity mode
VelocityMode_InData	STRUCT act_velocity: INT; act_axiserror: WORD; act_current: INT; END_STRUCT	Actual values in the velocity mode
InData	AXIS_DATA_IN	Data block (PB->PLC)
OutData	AXIS_DATA_OUT	Data block (PLC->PB)

## 3.6 System Functions and Blocks

The blocks of the S7 system functions are described in the according manuals by SIEMENS.

## 4 Symbols

All function and data blocks receive a block number during compiling. These block numbers can be assigned in the symbol table. The blocks then can be called using their numbers (DB100, FB100 or FB101) or their symbol names (RD\_DNC, WR\_DNC).

### 4.1 Symbol Table

As an example in the following you see an extract of the symbol table including the assignment of the block numbers.

Symbol	Address	Data type	Comment
DB_INIT_DNC	DB2	FB2	Instance data block
DB_RD_DNC	DB10	FB3	Instance data block
DB_WR_DNC	DB11	FB4	Instance data block
DB_HEARTBEAT	DB12	SFB4	Instance data block
DB_ERRORCODE	DB100	DB100	Global data block
DB_GLOBALDATA	DB101	DB101	Global data block
INIT_DNC	FB2	FB2	Function block
RD_DNC	FB3	FB3	Function block
WR_DNC	FB4	FB4	Function block
MAIN_DNC	OB1	OB1	Organization block
DPRD_DAT	SFC14	SFC14	System function
DPWR_DAT	SFC15	SFC15	System function
BLKMOV	SFC20	SFC20	System function
AXIS_DATA_IN	UDT1	UDT1	User-defined data type
AXIS_DATA_OUT	UDT2	UDT2	User-defined data type
AXIS_REF	UDT3	UDT3	User-defined data type



## 5 Hardware Configuration in the PLC

The SIEB & MEYER module is called via the serial interface of the gateway (Anybus Communicator). The Anybus Communicator is integrated into the system via the Profibus DP. The Anybus Communicator is projected as DP-slave in the software SIEMENS Step 7 with inputs and outputs of 10 bytes each. Therefore the appropriate GSD file by HMS is required. Please refer to the SIMATIC documentation on how to integrate new GSD files into the programming environment.

### 5.1 GSD File

In order to project the Anybus Communicator on the PLC the GSD file is required.

HMS released the following notes about these files:

- ▶ HMSB1803.gsd to be used with the -C versions of AB7000 and AB7029. This GSD file together with a -C version of the Communicator supports Failsafe functionality.
- ▶ HMS\_1803.gsd to be used with the -B versions of AB7000 and AB7029 and does not support Failsafe functionality.
- ▶ HMS recommends to upgrade any previous installed ABC Config Tool to version 3.00 or later (the latest version is available on [www.anybus.com](http://www.anybus.com))

### 5.2 Settings in the Hardware Configuration

At first the object "Anybus Communicator – Slave" is dragged to the Profibus DP master system in the station window of the software "HW Config".

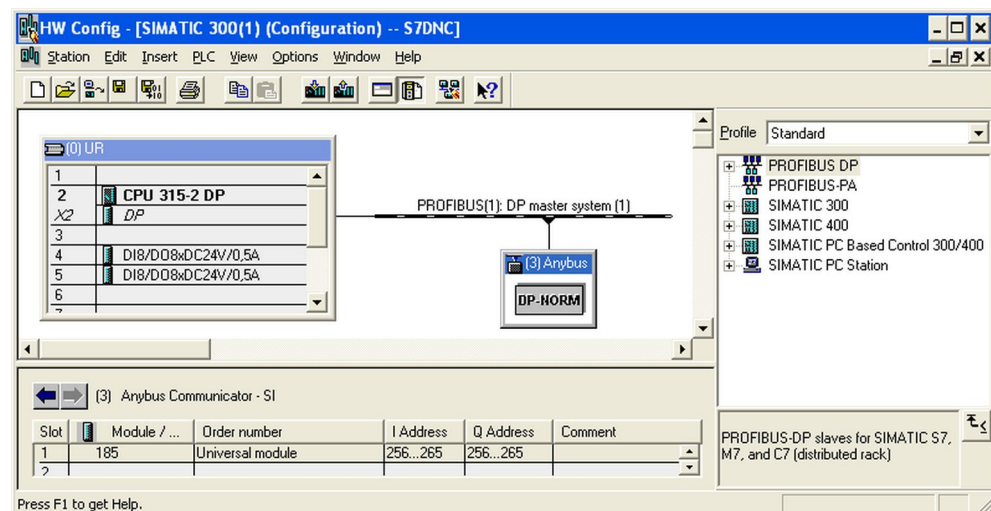


Fig. 2: SIMATIC software "HW Config"

The inserted module is highlighted. By double-click the properties window is opened. There you can enter a module name and assign the Profibus address.

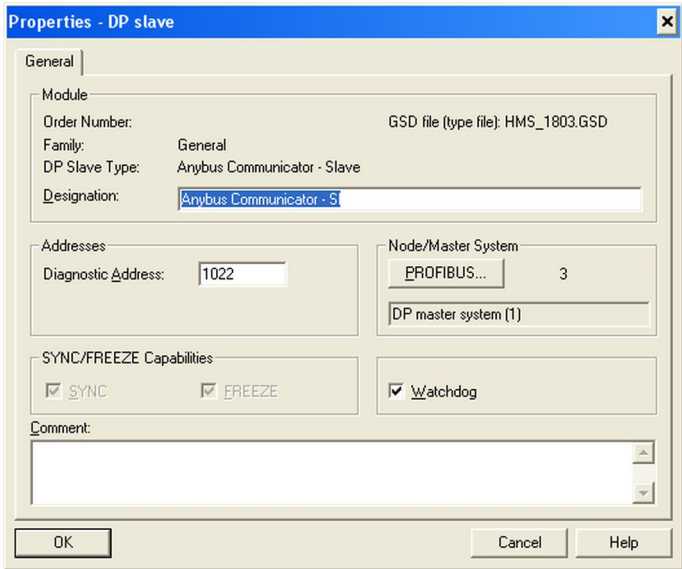


Fig. 3: Dialog window "Properties - DP slave"

In the lower part of the station window the detailed view of the Anybus Communicator is indicated in table form.

The object "Universal module" is dragged into the detailed view of the software "HW Config". By double-click you can open a properties window again and define the addresses of the input and the output area. Therewith data consistence is defined over the entire length.

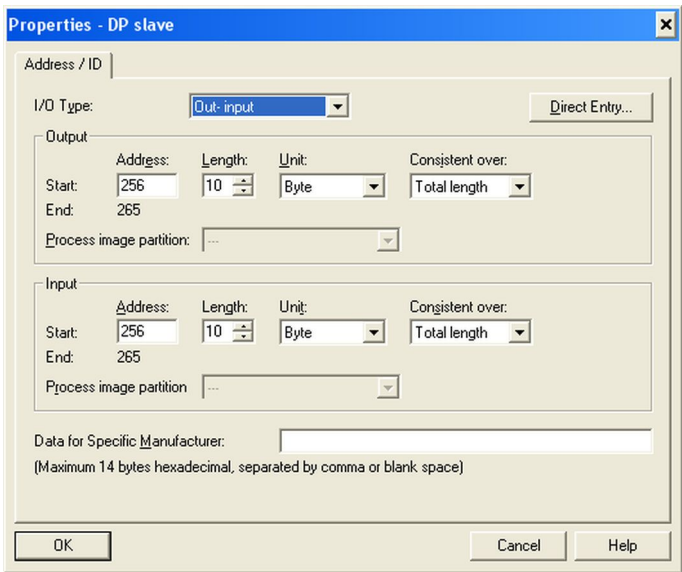


Fig. 4: Dialog window "Properties - DP slave"

## 5.3 Defining the Addresses in the Modules

Beside defining the addresses in the PLC also the SD2S modules and the gateway must be set to the right address. Refer to the device documentation for additional information.

## 6 Anybus Communicator by HMS

Anybus Communicator is configured by SIEB & MEYER before delivery.



The user only needs to set the Profibus address by means of the address selection switches to establish the connection between the devices.

### 6.1 Telegram Structure

Anybus Communicator receives the Profibus telegrams of the PLC and sends them as transmit telegrams to the drive via the subnetwork. After receiving and processing the drive sends a response telegram to Anybus Communicator. The latter forwards the response to the PLC via Profibus.

The telegrams are made up of the CAN data of the communication with the PLC via Profibus. The content of the CAN data is described in the document "8\_Byte\_DNC.doc".





# 7 Hardware Description

For initial operation and use of the mentioned devices the according regulations, directives and standards must be taken into account. For further information refer to the according product manuals.

An Anybus Communicator is used to connect devices providing a serial interface to the Profibus. This gateway is the master for the serial connection and the slave for Profibus DP.

## 7.1 Connecting the Devices

The following figures indicates the connection establishment between PLC, Anybus Communicator and drive control in principle.

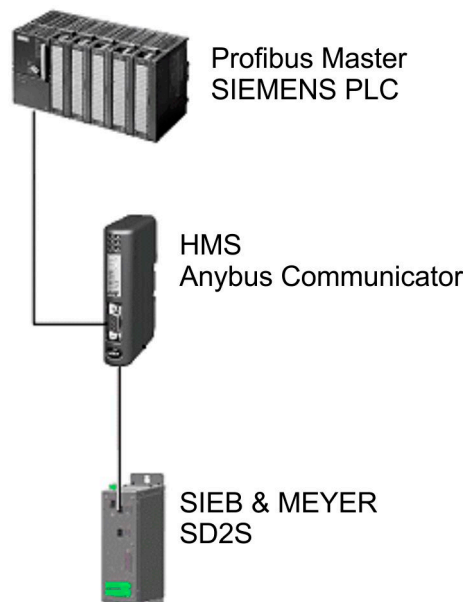


Fig. 5: Device connection

## 7.2 Anybus Communicator by HMS

Detailed information on Anybus Communicator is to find in the HMS user manual ([www.anybus.com](http://www.anybus.com)).

## 7.2.1 Connectors, Switches and Indicators

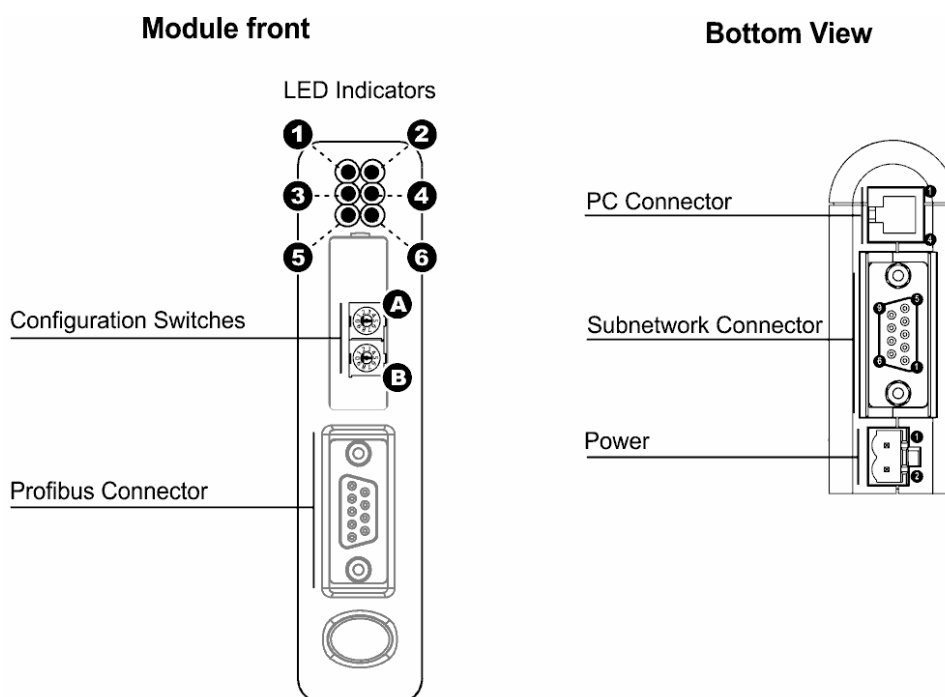


Fig. 6: Device views of Anybus Communicator by HMS

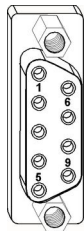
### LED indicators

LED no	Status	Meaning
1 - Online	Green	Online
	Off	Not online
2 - Offline	Red	Offline
	Off	Not offline
3 - Not used	–	–
4 - Fieldbus Diagnostics	Off	No diagnostics present
	Red, flashing 1 Hz	Error in configuration
	Red, flashing 2 Hz	Error in user parameter data
	Red, flashing 4 Hz	Error in initialization
5 - Subnet Status	Flashing green	Running, but one or more transaction errors
	Green	Running
	Red	Transaction error/timeout or subnet stopped
6 - Device Status	Off	Power off
	Alternating red/green	Invalid or missing configuration
	Green	Initialization
	Flashing green	Running
	Red	Boot loader mode
	Flashing red	Note the flash sequence pattern and contact the HMS support department.

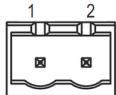
### Configuration switches A, B

Set the PROFIBUS node address by using the switches as follows: Node address = (switch B × 10) + (switch A × 1)

## Profibus connector

Female submin D connector	Pin	Description
	Housing	Cable shield
	1	–
	2	–
	3	B-line
	4	RTS
	5	GND bus
	6	+5 V bus out
	7	–
	8	A-line
	9	–

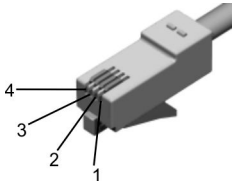
## Power connector

Connector	Pin	Description
	1	+ 24 V DC
	2	GND

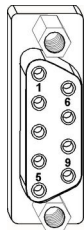
### Notes

- ▶ Use 60/75 or 75 °C copper (Cu) wire only.
- ▶ The terminal tightening torque must be between 5...7 lbs-in (0.5...0.8 Nm).

## PC connector (RJ11)

RJ11 male connector	Pin	Description
	1	GND
	2	GND
	3	RS232 Rx
	4	RS232 Tx

## Subnetwork connector

Female submin D connector	Pin	Description
	Housing	Cable shield
	1	+5 V output (max. 100 mA)
	2	RS232 Rx
	3	RS232 Tx
	4	–
	5	GND Signal
	6	RS422 Rx+
	7	RS422 Rx-
	8	RS485+ / RS422 Tx+
	9	RS485- / RS422 Tx-



## 8 Error Detection

In the following several errors regarding the conversion from Profibus to serial and their detection are described.

### 8.1 Errors in S7 PLC

#### **PLC failure**

Anybus Communicator detects a bus error and sets the output data to the subnetwork to zero. The drive SD2S switches off because the control word has the value zero. The drive SD2S signals a bus error when the heartbeat does not toggle, provided that this function is selected.

#### **PLC stops**

S7 PLC sets the output data to zero. Anybus Communicator forwards this data to the subnetwork. The drive SD2S switches off because the control word has the value zero. The drive SD2S signals a bus error when the heartbeat does not toggle, provided that this function is selected.

#### **System Function Block SFC 15**

When the system function block SFC15 is not called, the present data pattern is kept and sent again and again. Anybus Communicator forwards this data to the subnetwork. The drive is not able to distinguish whether the same data shall be sent repeatedly or the SFC can not be called. The drive SD2S signals a bus error when the heartbeat does not toggle, provided that this function is selected.

### 8.2 Error in Anybus Communicator

#### **ABC failure**

S7 detects that the participant is not existent. It signal a bus error and stops the program. The drive SD2S does not receive any telegrams. The drive can not detect whether telegrams are sent or not. The drive SD2S signals a bus error when the heartbeat does not toggle, provided that this function is selected.

#### **Error Detection Field Bus**

If Anybus Communicator detects a field bus error, it sets the output data to the subnetwork to zero. The drive SD2S switches off because the control word has the value zero. The drive SD2S signals a bus error when the heartbeat does not toggle, provided that this function is selected.

#### **Error Detection Subnetwork**

If Anybus Communicator detects an error in the subnetwork, it sets the output data to the field bus to zero. The status in the PLC program can be set correspondingly and the PLC program can be set to switch to an error routine.

## 8.3 Error in Drive Control SD2S

Due to a failure of the drive SD2S the DNC telegram is replied faulty or not at all.

Thus, Anybus Communicator does not receive a response to data it had sent. After a timeout sending data is repeated as often as programmed. If Anybus Communicator does not receive a response continuously, it identifies an error in the subnetwork and sets the output data of the field bus to zero. The status in the PLC program can be set correspondingly to switch the PLC program to an error routine.

## 9 Related Documents

The following documents provide more information on this topic:

Supplier	Document name
SIEMENS	Manuals regarding Step 7 programming
HMS	ABC_PDP_User_Manual
SIEB & MEYERAG	8_Byte_DNC.doc
SIEB & MEYERAG	SD2_DeviceControl.pdf
SIEB & MEYERAG	SD2_CAN-Connection.pdf
SIEB & MEYERAG	drivemaster2_UserManual.pdf

