



Application Report No.: 200.4.6

Topic: Acyclic parameter channel (DP-V1) of Profibus-DP and Siemens S7 PLC

Type: All

Branch: All

Subgroup: Industrial communication

Drive controller: 82vector/motec with Profibus I/O E82ZAFPC201

Classification: ☐ Confidential, for Lenze employees only
☒ Can be sent to customers

Preface / aim of the Application Report

- This Application Report provides a general overview of the structure and the different functions of a Profibus-DP network / slave.
- This overview is followed by a description of the general structure and the function principle of the acyclic parameter channel DP-V1. The implementation on the Lenze code level is explained using the example of a parameter write and read requests.
- In addition, we give step-by-step instructions of what is to be configured in the Siemens STEP 7 programming software in connection with DP-V1. These instructions are completed by a short description of the STEP 7 example program which can be downloaded from the Lenze homepage at www.lenze.com.

Status	Date	Version	Author
First DE edition	16/05/2006	V 1.0	Harms IK
First EN edition	31/05/2006	V 1.0	Harms IK
Expanded version			
Update	10/08/2007	V1.1	Harms IK



Table of contents

1.	Introduction	3
2.	Bus cycle.....	5
3.	DP-V1 basics	6
3.1.	DP-V1 parameter request	7
3.2.	DP-V1 parameter response	10
3.3.	Examples for reading out / requesting a code	12
3.3.1.	General	12
3.3.2.	Reading out code L-C0061.....	12
3.3.3.	Changing / writing to code L-C0012	14
3.4.	Coding of the field in parameter request / parameter response	16
3.5.	Error numbers in DP-V1 parameter response	17
4.	General characteristics of the Lenze DP-V1 slave module	18
4.1.	GSE file	18
5.	Configuration notes for Siemens PLC and DP-V1	19
5.1.	Step 1: Configuration of the CPU with HW Konfig	19
5.2.	Step 2: Configuration of the Lenze Profibus slave E82ZAFPC201 Profibus I/O	22
5.3.	Step 3: Setting of equidistant DP bus cycle.....	24
5.4.	Step 4: Saving and compiling of the configured hardware.....	26
6.	Program example for S7 PLC with DP-V1	27
6.1.	General.....	27
6.2.	Program description.....	27
6.2.1.	Variable table DP-V1.....	28
6.2.2.	Functions FC100 – FC108	29
6.2.3.	Data blocks DB100 – DB108.....	31
6.3.	Siemens system functions SFB 53 'WRREC' and SFB 52 'RDREC'.....	32
6.3.1.	'SFB 53 WRREC'	32
6.3.2.	'SFB 52 RDREC'	33



1. Introduction

The Profibus-DP fieldbus is specified according to a series of standards, which have been developed by the Profibus User Organization PUO and its members. The functions of a Profibus slave can be subdivided into different groups. Figure 1 gives an overview of the slave device architecture, which is subdivided into 7 different layers and a profile layer.

Layer

Profiles	Profidrive V2.0		ProfiSAFE	Profibus-PA	Redundancy	Profidrive V3.x
7	FMS	DP-V0		DP-V1	DP-V2	
3-6	Not used					
2	Profibus-FDL					
1	RS 485			IEC 61158-2 low speed		Optical fibre

Figure 1: Profibus architecture

Layer 1:

The physical layer 1 describes the transmission technology (RS485 or optical fibre). IEC 61158-2 low speed is a special transmission technology for Profibus-PA devices (31.25 kbps).

Layer 2:

Here the bus access method and the error-protected transmission of the data blocks from the sender to the receiver are provided (FDL=Fieldbus Data Link).

Layers 3-6:

Not used on Profibus

Layer 7:

Includes different communication services which can be provided by a slave device.

FMS: No longer up to date

DP-V0: **Cyclic** process data and, if required, cyclic parameter data

DP-V1: Cyclic process data and **acyclic** parameter data

DP-V2: DP-V0 + V1, **motion control** functions such as broadcast telegrams (lateral communication) and synchronisation telegrams



Profiles:

Over the years a number of profiles have been specified for different Profibus applications. The ones mainly used are Profidrive (profile for variable-speed controllers) and ProfiSAVE for safety-relevant system parts. The communication services of layer 7 are used by the different profiles with DP-V1 and DP-V2 being strongly connected to Profidrive V3.x.

Up to now all Lenze Profibus communication modules have met the DP-V0 standard. In this connection the cyclic parameter channel is of disadvantage because the parameter data is always transmitted even if the Profibus master does not initiate new parameter requests. In this case the last data would be transmitted repeatedly. Such a method is negative for signal transmission because information already known is transmitted and the transmission effectiveness (bus cycle) decreases.

To increase the effectiveness of data transmission it thus makes sense to transmit parameter data only if it is new and up to date. This can be done with an acyclic parameter channel. The channel is only used for **new** requests. In addition, the DP-V1 standard offers another advantage. Profibus-DP networks have so far been operated with one Profibus master, preferably in conjunction with a PLC. This master is responsible for the cyclic process data exchange with the DP slave devices and is also called 'class 1 DP master' or MAC1 for short. However, on Profibus-DP networks several masters can be operated with bus access being controlled by the token passing method. A second master can be useful for engineering, configuration and operator devices (HMI) as well as for commissioning, maintaining and diagnosing the system. This 'class 2 DP master', or MAC2 for short, only uses the acyclic DP-V1 services. Further masters on the Profibus do not make sense since the cyclic process data is generated in the PLC.

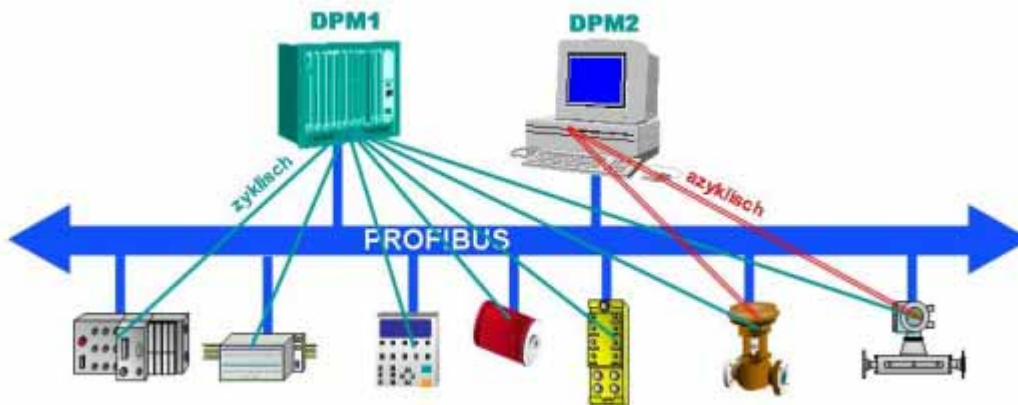


Figure 2: Basic structure of Profibus-DP with class 1 and class 2 masters

Moreover, the cyclic parameter channel of DP-V0 occupies 8 bytes of I/O memory space for each slave. On large Profibus networks with many stations this can create bottlenecks in the I/O memory area of the PLC and a PLC with larger I/O memory has to be used. The acyclic parameter channel to DP-V1 does **not** occupy I/O memory space. It is accessed via the slave **diagnosis address**.



2. Bus cycle

The Profibus bus cycle consists of fixed time intervals. Within these fixed time intervals the cyclic process data (I/O data exchange) is exchanged with each slave and event-controlled acyclic services can occur. Acyclic telegram components are:

- Data exchange during the initialisation phase of the Profibus network
- DP slave diagnostic functions
- Class 2 master communication
- Master / master communication
- Telegram repetition in the event of faults
- Acyclic data exchange according to DP-V1

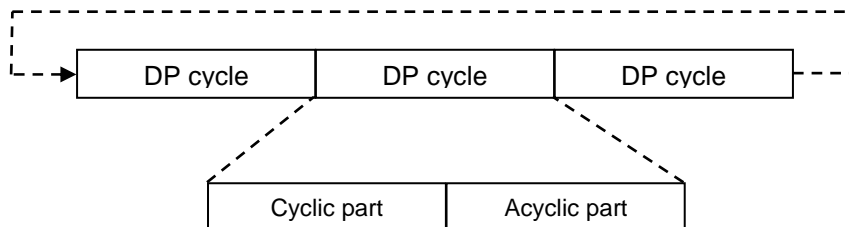


Figure 3: Basic structure of a Profibus cycle

The cyclic part is fixedly allocated through the I/O data width (process data) of the PLC whereas the acyclic part can be adjusted. Especially for drive technology applications a DP cycle which remains exactly the same is of importance to ensure the dynamic performance of the application. For this purpose the DP master reserves a certain time interval for the acyclic communication part (DP-V1) in an equidistant DP cycle. The master ensures that this reserved time interval is not exceeded by permitting only a certain number of acyclic telegram events. If the reserved time interval is not used for sending acyclic telegrams, the master bridges the missing difference to the set equidistance time with telegrams sent to itself (pause). So the reserved equidistance time is met exactly.

The time base for the equidistant DP bus cycle is selected in the course of the master configuration.

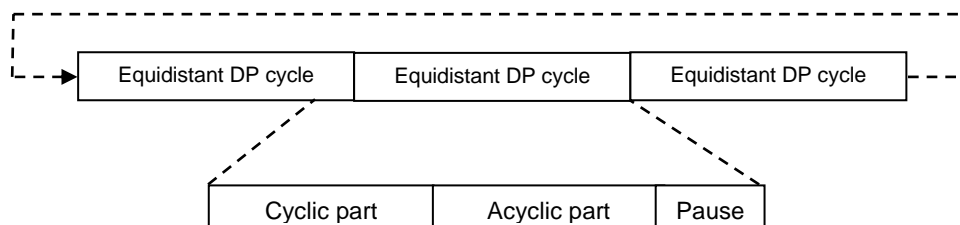


Figure 4: Basic structure of an equidistant Profibus cycle



3. DP-V1 basics

The acyclic parameter request always starts with a 'write request'. The 'write request' includes the parameters to be accessed in the slave for reading or writing.

Since the parameter access in the slave normally takes much longer than the DP cycle, the master contacts the slave via a 'read request' during each acyclic part of the bus cycle to determine whether the parameter access is completed and the 'read response' can be sent back. Figure 5 shows that the slave sends the parameter response to the master in the form of a read response when parameter processing in the slave is completed. DB47 stands for data set number 47 and is specified according to the Profibus standard.

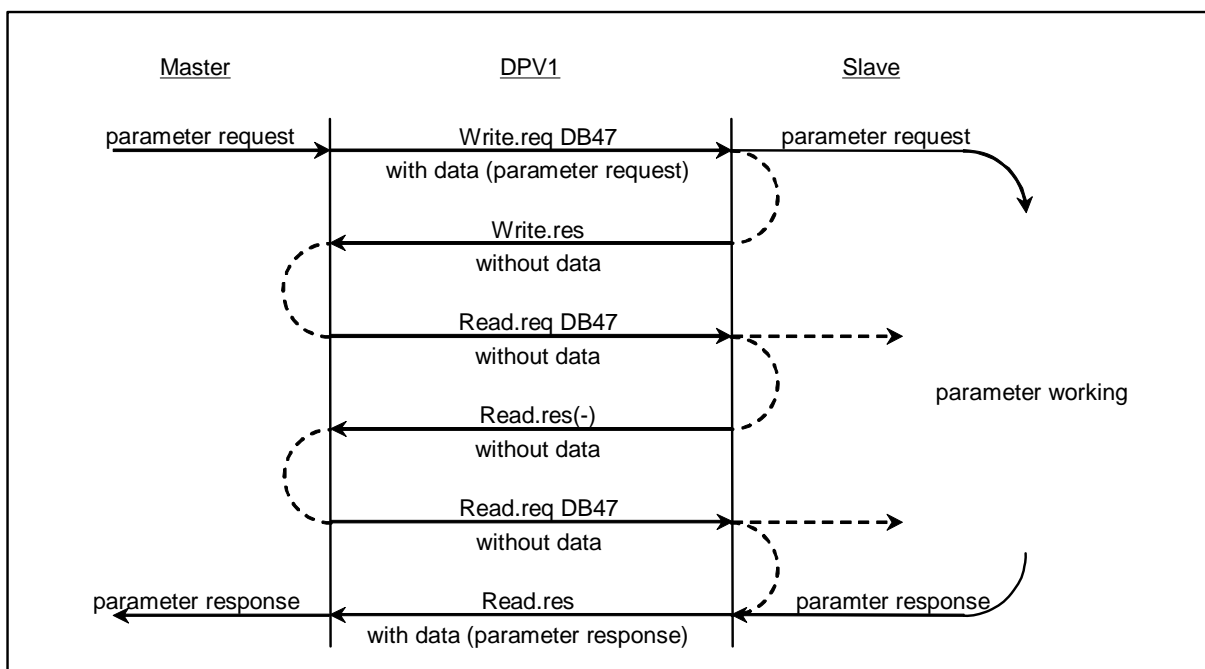


Figure 5: Acyclic telegram exchange

A 'write request' parameter request consists of three parts:

- Request header: Request ID and number of parameters to be accessed.
Addressing of an axis or of multi-axis drives.
- Parameter address: Addressing of one or several parameters (codes).
- Parameter value: For each parameter addressed there is an area for the parameter values.



A 'read response' to a parameter request consists of a maximum of three parts depending on whether codes are read or written to (request ID byte 2) and on whether the request has been executed fault-free or faulty.

Response header: Response ID and number of parameters that have been accessed.
Addressing of an axis or of multi-axis drives.

Parameter value / error code: Read parameter value(s) of the code or error code

The following table shows the number of parts and their dependencies.

Request ID	Error occurred	Number of parts in the 'read response'
0x01 request code(s)	No	2
	Yes	2
0x02 change code(s)	No	1
	Yes	2

3.1. DP-V1 parameter request

The following table shows the structure of a DP-V1 parameter 'write request' from the Profibus master to the slave. The request reference is always 0x00. The number of axes is either 0x00 or 0x01 for Lenze drives because there are no multi-axis controllers with only one Profibus connection. Due to the option of transferring one or several code access(es) in one parameter request the length of the request is variable.

	Designation	Meaning
Request header		
Byte 1	Request reference	0x00
Byte 2	Request ID	0x01 request code(s) 0x02 change code(s)
Byte 3	Axis	0x00 - 0x01
Byte 4	Number of parameters	0x01 - 0x25 quantity 1 - 37
Parameter address		
Byte 5	Attribute	0x10 data type 'value'
Byte 6	Number of elements	0x01 – 0x75 quantity 1 - 117
Bytes 7, 8	Index / parameter	0x0001 – 0xFFFF
Bytes 9, 10	Subindex	
Parameter value		
Only for request ID 0x02 in byte 2 (write to code(s))		
Byte 11	Format	0x43 double word (preferably)
Byte 12	Number of values	0x00 – 0xEA, for several values
Bytes 13, 14, 15, 16	Value	Single value



If several codes are to be accessed in one request, they also must be listed in the parameter address part and they must match the value specified in byte 4 (number of parameters).

Example for a parameter request with read access to two codes:

	Designation	Meaning
Request header		
Byte 1	Request reference	0x00
Byte 2	Request ID	0x01 request code(s)
Byte 3	Axis	0x00
Byte 4	Number of parameters	0x02
Parameter address		
Byte 5	Attribute	0x10 data type 'value'
Byte 6	Number of elements	0x01 – 0x75
Bytes 7, 8	Index / parameter	0x0001 – 0xFFFF
Bytes 9, 10	Subindex	
Byte 11	Attribute	0x10 data type 'value'
Byte 12	Number of elements	0x01 – 0x75
Bytes 13, 14	Index / parameter	0x0001 – 0xFFFF
Bytes 15, 16	Subindex	

If the codes / subindexes are not accessed for reading, but for writing, the parameter value part is added and the request ID in byte 2 changes from 0x01 to **0x02**.

Example for a parameter request with write access to two codes:

Parameter value		
Only for request ID 0x02 in byte 2 (write to code(s))		
Byte 17	Format	0x43 double word (preferably)
Byte 18	Number of values	0x01
Bytes 19, 20, 21, 22	Value	Single value in code 1
Byte 23	Format	0x43 double word (preferably)
Byte 24	Number of values	0x01
Bytes 25, 26, 27, 28	Value	Single value in code 2



If several subcodes of a code are to be accessed for reading in a request, the number of subcodes must be entered in byte 6 of the parameter address part (number of elements). You do not have to start with subcode 1, it is also possible to start with a subcode >1. The subcode you want to start with must be entered in bytes 9 & 10 of the parameter address part (subindex).

Example for a parameter request with read access on a code with two subindexes (subcodes 3 and 4 of the code):

	Designation	Meaning
Request header		
Byte 1	Request reference	0x00
Byte 2	Request ID	0x01 request code(s)
Byte 3	Axis	0x00
Byte 4	Number of parameters	0x01
Parameter address		
Byte 5	Attribute	0x10 data type 'value'
Byte 6	Number of elements	0x02
Bytes 7, 8	Index / parameter	0x0001 – 0xFFFF
Bytes 9, 10	Subindex	0x03

If the codes / subindexes are not accessed for reading, but for writing, the parameter value part is added. In this part the number of the values entered (byte 12) must be adapted. In addition, the request ID in byte 2 changes from 0x01 to **0x02**.

Example for a parameter request with write access to a code with two subindexes:

Parameter value		
Only for request ID 0x02 in byte 2 (write to code(s))		
Byte 11	Format	0x43 double word (preferably)
Byte 12	Number of values	0x02
Bytes 13, 14, 15, 16	Value	Single value in subcode 1
Bytes 17, 18, 19, 20	Value	Single value in subcode 2



3.2. DP-V1 parameter response

The following table shows the structure of a DP-V1 parameter 'read response' from the Profibus slave to the master. The request reference is taken from the parameter request and has the value 0x00. The number of axes is either 0x00 or 0x01 for Lenze drives because there are no multi-axis controllers with only one Profibus connection. Due to the option of transferring one or several code access(es) in one parameter request the length of the response is variable.

	Designation	Meaning
Response header		
Byte 1	Request reference	0x00 mirrored from parameter request
Byte 2	Response ID	0x01 request code(s) (+) 0x02 change code(s) (+) 0x81 request code(s) (-) 0x82 change code(s) (-)
Byte 3	Axis	0x00 - 0x01 mirrored from parameter request
Byte 4	Number of parameters	0x01 - 0x25 quantity 1 - 37
Parameter value / error code		
Only for response ID 0x01 in byte 2 (request code(s) (+)) or response IDs 0x81 and 0x82 in byte 2 (error occurred)		
Byte 5	Format	0x43 double word (preferably)
Byte 6	Number of values	0x00 – 0xEA, for several values
Bytes 7, 8, 9, 10	Value	Single value / error code
Byte n	Value n	Single value / error code



If an error occurs while reading out (requesting) codes, only the parameter values appearing before the error are transmitted in the parameter response telegram.

Example: While reading out 2 codes, an error occurs when the second code is read out.

	Designation	Meaning
Response header		
Byte 1	Request reference	0x00 mirrored from parameter request
Byte 2	Response ID	0x81 request code(s) (-)
Byte 3	Axis	0x00 - 0x01 mirrored from parameter request
Byte 4	Number of parameters	0x02
Parameter value / error code		
Byte 5	Format	0x43 double word (preferably)
Byte 6	Number of values	0x01
Bytes 7, 8, 9, 10	Value	Single value
Byte 11	Format	0x44 error occurred
Byte 12	Number of values	0x01
Bytes 13, 14	Value / error code	Error code

If codes have been changed successfully, the parameter response only consists of the response header part. If an error occurred while processing the parameter request, the parameter response contains an error code.

Example for a parameter response to a write access to two codes with an error occurring while processing the second code:

	Designation	Meaning
Response header		
Byte 1	Request reference	0x00 mirrored from parameter request
Byte 2	Response ID	0x82 change code(s) (-)
Byte 3	Axis	0x00 mirrored from parameter request
Byte 4	Number of parameters	0x02
Parameter value / error code		
Byte 11	Format	0x44 error
Byte 12	Number of values	0x02 error while processing the second code
Bytes 13, 14	Value	Error code



3.3. Examples for reading out / requesting a code

3.3.1. General

For DP-V1 too, the index of a Lenze code is calculated with the following formula:

$$\text{Index} = 24575 - \text{Lenze code}$$

Example: Lenze code L-C0061 => $24575 - 61 = 24514 = 5FC2$

The formatting of a Lenze code parameter value depends on the data format which is listed in the attribute table of the respective controller. Preferably the data format is of type 'FIX32' including a multiplying factor of 10,000 due to the max. 4 decimal positions.

3.3.2. Reading out code L-C0061

The heatsink temperature L-C0061 (current value: 43°C) of the controller is to be read.

Parameter request

	Designation	Value
Request header		
Byte 1	Request reference	0x00
Byte 2	Request ID	0x01 request code(s)
Byte 3	Axis	0x00
Byte 4	Number of parameters	0x01 one code
Parameter address		
Byte 5	Attribute	0x10 data type 'value'
Byte 6	Number of elements	0x01
Bytes 7, 8	Index / parameter	0x5FC2 L-C0061
Bytes 9, 10	Subindex	0x0000



Parameter response for fault-free execution

	Designation	Meaning
Response header		
Byte 1	Request reference	0x00
Byte 2	Response ID	0x01 request code(s) (+)
Byte 3	Axis	0x00
Byte 4	Number of parameters	0x01
Parameter value / error code Only for response ID 0x01 in byte 2 (request code(s) (+)) or response IDs 0x81 and 0x82 in byte 2 (error occurred)		
Byte 5	Format	0x43 double word
Byte 6	Number of values	0x01
Bytes 7, 8, 9, 10	Value	0x00068FB0

The parameter value 0x00068FB0 must be converted into decimal and divided by 10,000 to get the correct parameter value of 43°C for the heatsink temperature.

Parameter response for a read error

	Designation	Meaning
Response header		
Byte 1	Request reference	0x00
Byte 2	Response ID	0x81 request code(s) (-)
Byte 3	Axis	0x00
Byte 4	Number of parameters	0x01
Parameter value / error code Only for response ID 0x01 in byte 2 (request code(s) (+)) or response IDs 0x81 and 0x82 in byte 2 (error occurred)		
Byte 5	Format	0x44 error
Byte 6	Number of values	0x01
Bytes 7, 8, 9, 10	Value	0x00xx0000

The two **xx** in byte 8 (value) indicate the error number (listed in chapter 3.5).



3.3.3. Changing / writing to code L-C0012

The acceleration time L-C0012 of the controller is to be set to 20s.

Parameter request

	Designation	Meaning
Request header		
Byte 1	Request reference	0x00
Byte 2	Request ID	0x02 change code(s)
Byte 3	Axis	0x00
Byte 4	Number of parameters	0x01
Parameter address		
Byte 5	Attribute	0x10 data type 'value'
Byte 6	Number of elements	0x01
Bytes 7, 8	Index / parameter	0x5FF3
Bytes 9, 10	Subindex	0x0000
Parameter value		
Only for request ID 0x02 in byte 2 (write to code(s))		
Byte 11	Format	0x43 double word
Byte 12	Number of values	0x01
Bytes 13, 14, 15, 16	Value	0x00030D40

Parameter response for fault-free transmission

	Designation	Meaning
Response header		
Byte 1	Request reference	0x00
Byte 2	Response ID	0x02 change code(s) (+)
Byte 3	Axis	0x00
Byte 4	Number of parameters	0x01



Parameter response after a write error

	Designation	Meaning
Response header		
Byte 1	Request reference	0x00
Byte 2	Response ID	0x82 change code(s) (-)
Byte 3	Axis	0x00
Byte 4	Number of parameters	0x01
Parameter value / error code		
Only for response ID 0x01 in byte 2 (request code(s) (+)) or response IDs 0x81 and 0x82 in byte 2 (error occurred)		
Byte 5	Format	0x44 error
Byte 6	Number of values	0x01
Bytes 7, 8, 9, 10	Value	0x00xx0000

The two **xx** in byte 8 (value) indicate the error number (listed in chapter 3.5).



3.4. Coding of the field in parameter request / parameter response

Field	Data type	Value	Comment
Request reference	Unsigned8	0x00 reserved 0x01 - 0xFF	
Request ID	Unsigned8	0x00 reserved 0x01 request parameter 0x02 change parameter 0x03 - 0x3F reserved 0x40 - 0x7F manufacturer-specific 0x80 - 0xFF reserved	Parameter request 'write request'
Response ID	Unsigned8	0x00 reserved 0x01 request parameter(+) 0x02 change parameter(+) 0x03 - 0x3F reserved 0x40 - 0x7F manufacturer-specific 0x80 reserved 0x81 request parameter(-) 0x82 change parameter(-) 0x83 - 0xBF reserved 0xC0 - 0xFF manufacturer-specific	Parameter response 'read response' (+) positive answer (-) negative answer
Axis	Unsigned8	0x00 - 0xFF number 0 - 255	
Number of parameters	Unsigned8	0x00 reserved 0x01 - 0x25 quantity 1 - 37 0x26 - 0xFF reserved	Limitation through DP-V1 telegram length
Attribute	Unsigned8	0x00 reserved 0x10 value 0x20 description 0x30 text 0x40 - 0x70 reserved 0x80 - 0xF0 manufacturer-specific	The four less significant bits are reserved for (future) expansion of number of elements to 12 bits
Number of elements / subindexes	Unsigned8	0x00 special function 0x01 - 0x75 quantity 1 - 117 0x76 - 0xFF reserved	Limitation through DP-V1 telegram length
Parameter number	Unsigned16	0x0000 reserved 0x0001 - 0xFFFF number 1 - 65535	
Subindex	Unsigned16	0x0000 - 0xFFFF number 0 - 65535	
Format	Unsigned8	0x00 reserved 0x01 - 0x36 data type 0x37 - 0x3F reserved 0x40 null 0x41 byte 0x42 word 0x43 double word 0x44 error 0x45 - 0xFF reserved	
Number of values	Unsigned8	0x00 - 0xEA number 0 - 234 0xEB - 0xFF reserved	Limitation through DP-V1 telegram length
Error number	Unsigned16	0x0000 - 0x00FF error number (see table 3.5)	Parameter response 'read response', the more significant byte is reserved.



3.5. Error numbers in DP-V1 parameter response

Error number	Meaning	Used at
0x00	Impermissible parameter number	Access to unavailable parameter
0x01	Parameter value cannot be changed	Change access to a parameter value that cannot be changed
0x02	Low or high limit exceeded	Change access with value outside the value limits
0x03	Faulty subindex	Access to unavailable subindex
0x04	No array	Access with subindex to non-indexed parameter
0x05	Incorrect data type	Change access with value that does not match the data type of the parameter
0x06	Setting not permitted	Change access with value unequal to 0 where this is not permitted
0x07	Description element cannot be change	Change access to a description element that cannot be changed
0x08	Reserved	(PROFIDrive Profile V2: PPO-Write requested in IR not available)
0x09	No description data available	Access to unavailable description without rights to change parameters
0x0A	Reserved	(PROFIDrive Profile V2: Access group wrong)
0x0B	No operation priority	Change access without rights to change parameters
0x0C-0x0E	Reserved	(PROFIDrive Profile V2)
0x0F	No text array available	Access to text array that is not available (parameter value is available)
0x10	Reserved	(PROFIDrive Profile V2: No PPO-Write)
0x11	Request cannot be executed because of operating state	Access is temporarily not possible for reasons that are not specified in detail
0x12	Reserved	(PROFIDrive Profile V2: other errors)
0x13	Reserved	(PROFIDrive Profile V2: Data cannot be read in cyclic interchange)
0x14	Value impermissible	Change access with a value that is within the value limits but is not permissible for other long-term reasons (parameter with defined single values)
0x15	Response too long	The length of the current response exceeds the maximum transmittable length
0x16	Parameter address impermissible	Illegal value or value which is not supported for the attribute, number of elements, parameter number or subindex or a combination
0x17	Illegal format	Write request. Illegal format or format of the parameter data which is not supported
0x18	Number of values are not consistent	Write request. Number of the values of the parameter data do not match the number of elements in the parameter address
0x19 - 0x64	Reserved	-
0x65 - 0xFF	Manufacturer-specific	-

Error numbers 0x00 - 0x13 are taken from PROFIDrive Profile Version 2. Values that cannot be assigned are reserved for future use.



4. General characteristics of the Lenze DP-V1 slave module

- Compatibility to PKW (parameter characteristic) requests according to PROFIDrive profile, version 2
- 16-bit wide address for both parameter number and subindex.
- Transmission of complete arrays or part of arrays or of the complete parameter description.
- Transmission of different parameters with one access (multiparameter requests)
Support of multiparameter requests starting with **firmware version 0.9** of the E82ZAFPC201 module
- Only **one** parameter request processed at a time (no pipelining).
- A parameter request/response must fit into one data block (max. 240 bytes).
Requests/responses cannot be split over several data blocks. Due to the slave characteristics or bus configuration the maximum length of the data blocks can be smaller than 240 bytes.
- Spontaneous messages are not transmitted.
- There are no cyclic parameter requests.
- There are no DP-V1 alarms.
- Profile-specific parameters can be read independently of the slave status.

4.1. GSE file

The DP-V1 functions of the fieldbus function module E82ZAFPC201 of version FV 0.8 or higher require a new version of the GSE file 'Lenz081B.gse'. This file is available in the download area of the Lenze homepage at www.lenze.com. Since the name of the GSE file has not changed, you have to replace the old version by the new one in the STEP 7 hardware configuration.

The two GSE files can only be distinguished by opening them with a text editor. The GSE files start with some general information. The figure below shows that this GSE file has been adapted for DP-V1. Moreover, the revision number has been changed from 2 to 3.

```
; (c) Lenze
;
; Profibus-DP          Geraetestammdatei
;
; Autor:               ELE/Industrielle Kommunikation (TH)
; Erstellungsdatum:    14.06.2000
; Aenderungen:        10.02.2005 Ergänzung um DPV1 Parametrierdaten
;
; Allgemeine Angaben:
;
; #Profibus_DP
;
; GSD_Revision=3
;
```

Excerpt from GSE file 'Lenz081B.gse' for E82ZAFPC201 >= FV 0.8



5. Configuration notes for Siemens PLC and DP-V1

In the following the single configuration steps and notes concerning the Siemens PLC and DP-V1 are listed.

Siemens software used: SIMATIC Manager V5.3 + SP2
Siemens hardware used: CPU 315-2 PN/DP

The use of DP-V1 is possible with STEP 7 V5.1 SP 2 or higher.

5.1. Step 1: Configuration of the CPU with HW Konfig

Step	Comment	Note
1	Create a new project in the SIMATIC Manager under your SIMATIC Station used (e.g. SIMATIC 300 Station).	
2	Open the hardware configuration ' HW Konfig '.	
3	Import the Lenze GSE file ' LENZ081B.GSE ' needed for the fieldbus function module Profibus I/O E82ZAFPC201 by selecting the point ' Install GSE file ' under ' Options ' (menu bar). Then select the GSE file LENZ081B.GSE from your hard disk and confirm with ' OK '.	The Lenze GSE file LENZ081B.GSE is available in the download area of the Lenze homepage at www.lenze.com
4	Select the ' Standard ' hardware profile in the ' Profile ' pull-down menu of the ' Hardware Catalog ' window.	This is the default setting of the hardware catalog.
5	Drag and drop a Rail from the hardware catalog into the HW Konfig window and add the required CPU with Profibus interface (e.g. CPU 315-2 PN/DP) with drag and drop. Required path: 1.) Rail: \SIMATIC 300\RACK-300 2.) CPU: \SIMATIC 300\CPU\CPU 315-2 PN/DP	After adding the CPU 315-2 PN/DP a dialog box for setting the Ethernet interface opens.



6

Allocate an IP address and a subnet mask.
Click on 'New'.
The dialog box for setting the Ethernet subnet appears.

Close the dialog box with 'OK'.

7

Open the properties of the '**MPI/DP Interface**' with a double-click.
Here '**DP**' must be set for the interface type.

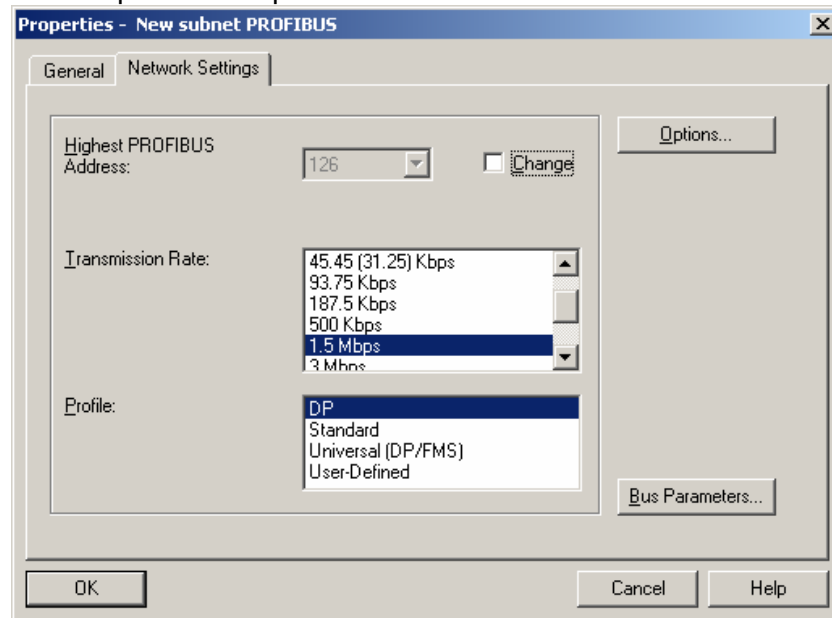
Close the dialog box with 'OK'.

On delivery the interface is set to MPI to enable addressing via the MPI adapters.



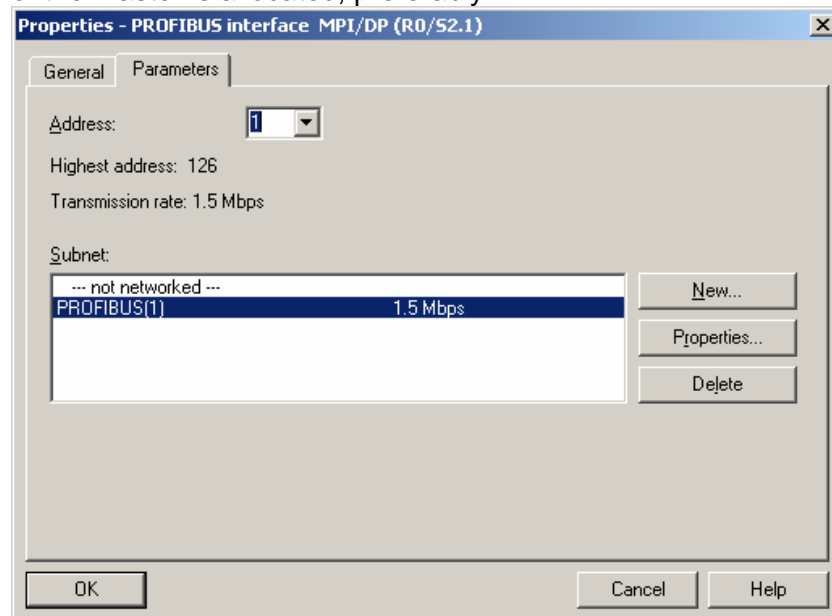
8

When the interface type is set to '**DP**', the dialog box for setting the PROFIBUS interface opens. Click on '**New**'. In the '**New subnet Profibus**' dialog box you can set the transmission rate and the profile. The profile must be set to '**DP**'.



Close the dialog box with '**OK**'.

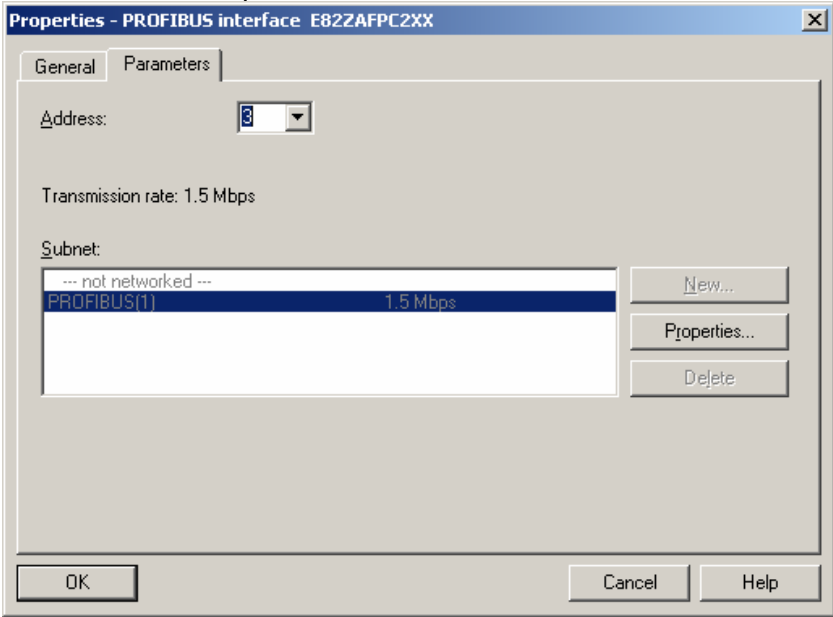
The dialog box for setting the PROFIBUS properties indicates the new Profibus subnet. Via the '**Address**' entry the Profibus address of the master is allocated, preferably 1.



Close all dialog boxes with '**OK**'.



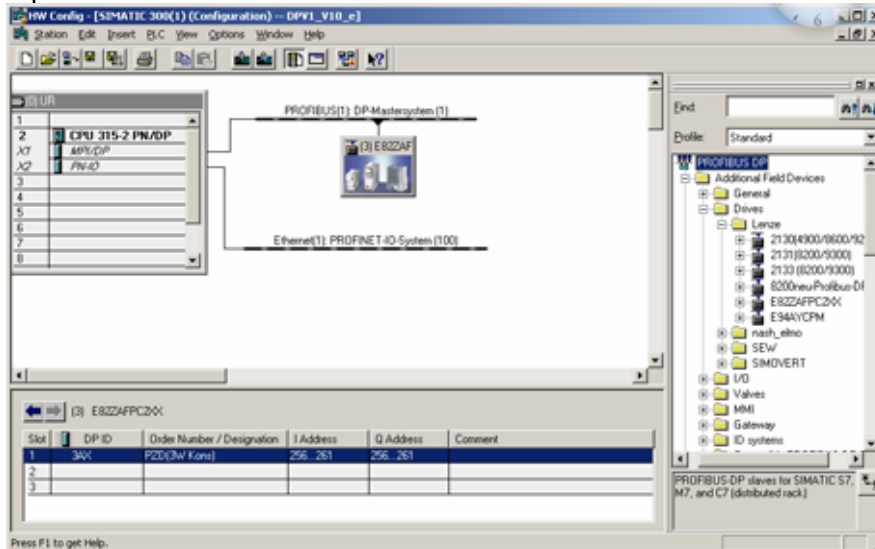
5.2. Step 2: Configuration of the Lenze Profibus slave E82ZAFPC201 Profibus I/O

Step	Comment	Note
1	Select the ' Standard ' hardware profile from the ' Profile ' pull-down menu in the ' Hardware Catalog ' window.	
2	Drag and drop the Profibus module E82ZAFPC2XX from the Hardware Catalog (PROFIBUS-DP\Additional Field Devices\Drives\Lenze) into the HW Konfig window onto the PROFIBUS subnet.	A dialog box for setting the properties of the PROFIBUS interface E82ZAFPC2XX opens.
3	<p>Enter the correct Profibus slave address, which is set with the DIP switch at the front-panel of the module.</p>  <p>Close the dialog box with 'OK'.</p>	Now you have established a DP station with the address 3 on the PROFIBUS subnet "(1)".
4	Open the Lenze slave E82ZAFPC2XX in the hardware catalog. A number of ' Parameter and process data combinations ' (e.g. PAR + PZD (1W)) are available.	The parameter combinations concern the cyclic DP-V0 parameter channel!



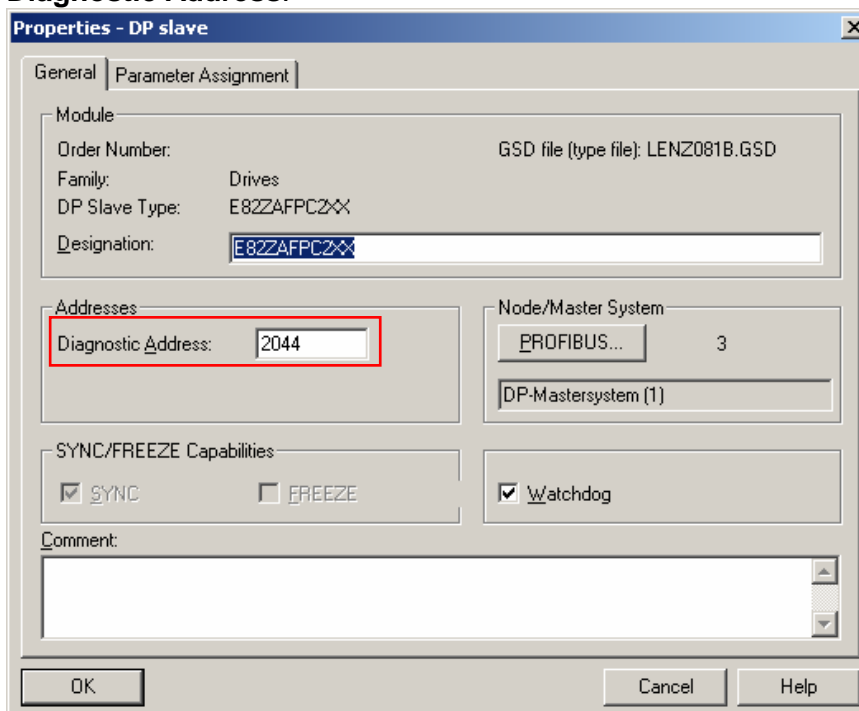
5

Drag and drop the required selection of **'process data combinations'** into slot 1 of the **E82ZAFPC2XX** Profibus slave station window. In this example PZD (3W Kons) is selected. In slot 1 you can find the PIW and POW addresses of the 3 process data words.



6

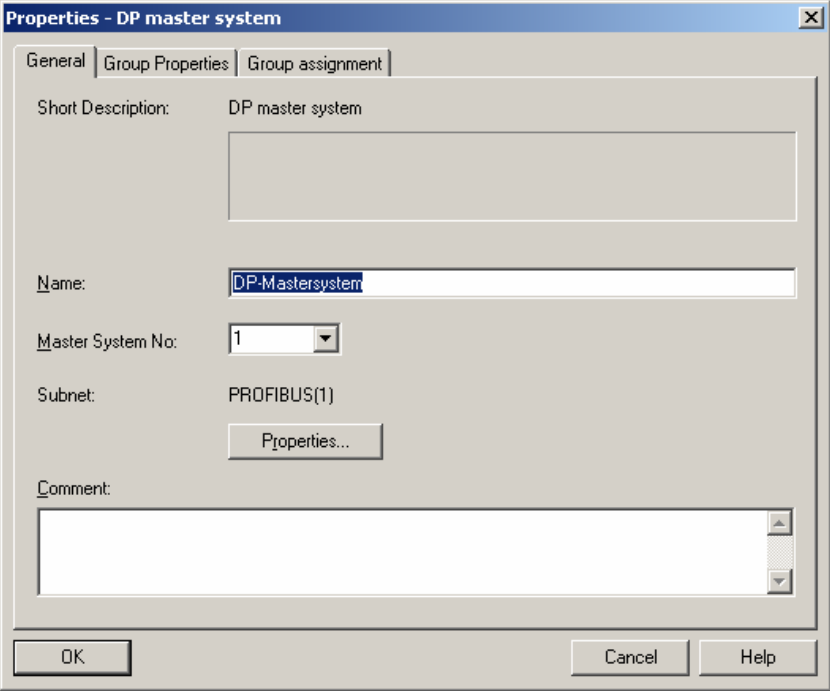
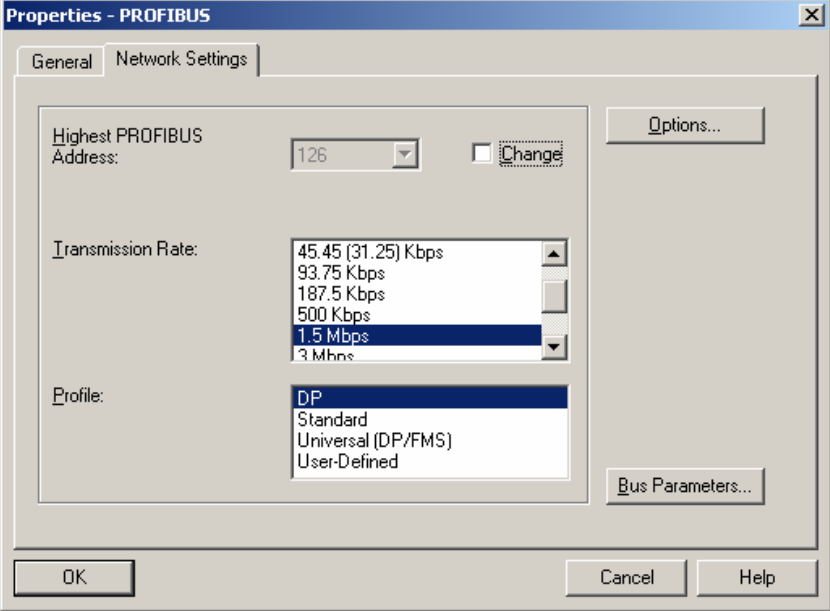
Double-click on the Profibus slave (3) E82ZAFPC2XX to get to **'Properties – DP Slave'**. In this dialog box you find the **Diagnostic Address**.



This address is required for DP-V1 communication



5.3. Step 3: Setting of equidistant DP bus cycle

1	<p>A double-click on the 'Profibus(1)' subnet opens the 'Properties – DP master system' dialog box.</p> 	
2	<p>Via the 'Properties' button in this dialog box you can open the 'Properties – PROFIBUS' dialog box.</p> 	



3

On the '**Network Settings**' tab you can open the '**Options**' dialog box via the '**Options**' button. By activating the '**Activate constant bus cycle time**' the dialog box changes so that settings are possible.

Options

Constant Bus Cycle Time | Cables

☒ Activate constant bus cycle time

Optimize DP cycle (and Ti, To if necessary): Recalculate

Number of PGs/DPs/TDs etc. on PROFIBUS

Configured: 0 Total: 0

Constant DP Cycle: 32.000 ms Time base: 0.001 ms Details ...

(min = 0.902 ms; max = 11000.000 ms)

Slave Synchronization

☐ Times Ti and To same for all slaves
(otherwise: make setting in slave properties)

Time Ti (read in process values): 10.666 ms

Time To (output process values): 10.666 ms

OK Cancel Help

When all Profibus slaves have been configured, you can optimise the DP cycle via the '**Recalculate**' button.

Options

Constant Bus Cycle Time | Cables

☒ Activate constant bus cycle time

Optimize DP cycle (and Ti, To if necessary): Recalculate

Number of PGs/DPs/TDs etc. on PROFIBUS

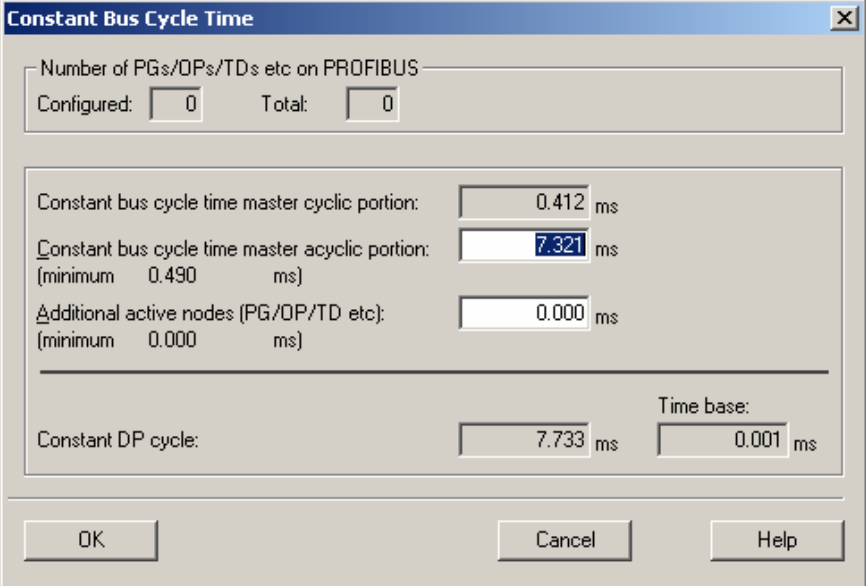
Configured: 0 Total: 0

Constant DP Cycle: 7.733 ms Time base: 0.001 ms Details ...

(min = 0.902 ms; max = 11000.000 ms)

In this example a DP cycle of 7.733 ms has been calculated.



4	<p>Via the 'Details' button you can open the following dialog box.</p>  <p>Here you can see that the cyclic part of the DP cycle is 0.412 ms and the acyclic part is 7.321 ms. You can decrease the acyclic part if this is necessary.</p>	<p><u>Note:</u> If the value set for the acyclic part is near the indicated minimum possible time, bus faults may cause the disconnection of the complete PROFIBUS subnet in some cases because the acyclic telegrams are not processed any longer!</p>
5	<p>The Profibus configuration is now complete and you can close all dialog boxes with 'OK'.</p>	

5.4. Step 4: Saving and compiling of the configured hardware

Step	Comment	Note
1	End the hardware configuration by calling the menu command Station > Save and compile .	The hardware data is accepted by the STEP7 project.
2	Transfer the hardware configuration with PLC > Download when the CPU state is STOP.	
3	Important: When a PLC with MPI/DP combination interface is used, the interface is changed to DP after the execution of ' Download to Module ' command. This means that you have to adapt the communication path to the PLC!	
4	Now the hardware configuration is completed.	

The PLC should change to the RUN state when everything has been configured correctly and the configured hardware has been transferred and there should not be any system error messages. If system error messages appear, the hardware has not been configured correctly.



6. Program example for S7 PLC with DP-V1

6.1. General

The program example **DPV1_V11.zip** can be unpacked with the retrieve function of STEP 7. It consists of several parts which can be activated via different flags. The individual parts exemplarily represent the different options of the DP-V1 services. They can be operated via the variable table **DP-V1**. All flags required are listed in the variable table **DP-V1**.

The PLC program example is available in the download area of the Lenze homepage at www.lenze.com. The example program consists of a total of 9 different program parts, which are started via the activation flags M2.0 – M3.0:

- | | |
|---------------------------------------|------|
| • Read a code | M2.0 |
| • Write to a code | M2.1 |
| • Read several codes | M2.2 |
| • Write to several codes | M2.3 |
| • Read several subcodes of a code | M2.4 |
| • Write to several subcodes of a code | M2.5 |
| • Read multiparameters | M2.6 |
| • Write to multiparameters | M2.7 |
| • Read a string code | M3.0 |

Multiparameter requests are requests which access several codes and/or several subcodes. String codes are codes which do not contain values, but characters (e.g. L-C0200 software ID number)

6.2. Program description

In OB1 all functions (FC100 – FC108) are invoked via the jump instruction 'Call'.

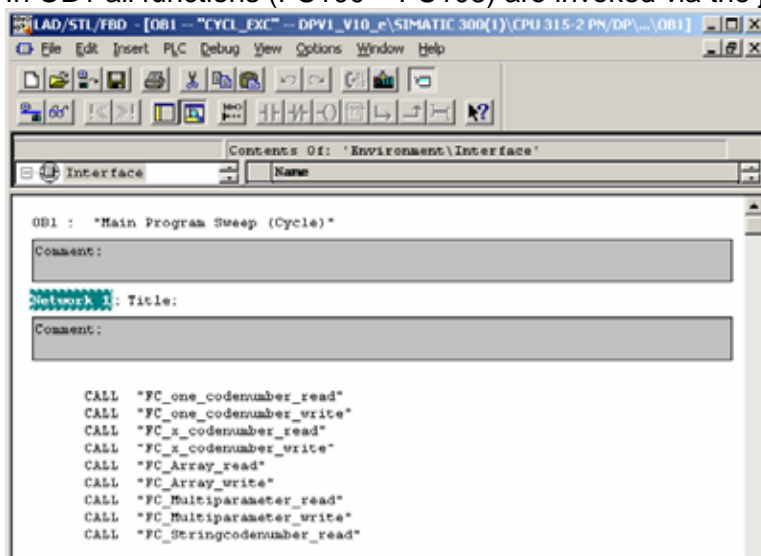


Figure 6: Programming in OB1

In the individual functions (FC100 – FC108) there is always an activation flag (M2.0 – M3.0) in network 1. If this flag is not activated, the whole block is skipped and execution continues at the end of the block.



6.2.1. Variable table DP-V1

The example program can be operated via the variable table 'DP-V1'.

	Address	Symbol	Displa	Status value	Modify valu
1	M 2.0	"One_codenumbr_read"	BOOL		
2	M 2.1	"One_codenumbr_wwrite"	BOOL		
3	M 2.2	"x_codenumbr_read"	BOOL		
4	M 2.3	"x_codenumbr_wwrite"	BOOL		
5	M 2.4	"Array_codenumbr_read"	BOOL		
6	M 2.5	"Array_codenumbr_wwrite"	BOOL		
7	M 2.6	"multiparameter_read"	BOOL		
8	M 2.7	"multiparameter_wwrite"	BOOL		
9	M 3.0	"stringcodenumbr_read"	BOOL		
10					
11	M 4.0	"Start_DP_V1"	BOOL		
12	M 4.6	"error_RDREC"	BOOL		
13	M 4.7	"read_ok"	BOOL		
14	M 5.0	"write_ok"	BOOL		
15	MD 30	"parameter1_error_value"	DEC		
16	MD 34	"parameter2_error_value"	DEC		
17	MD 38	"parameter3_error_value"	DEC		
18	MD 42	"parameter4_error_value"	DEC		
19	MD 46	"parameter5_error_value"	DEC		
20	MD 50	"parameter6_error_value"	DEC		
21	MD 54	"parameter7_error_value"	DEC		
22	MD 58	"parameter8_error_value"	DEC		
23	MD 62	"parameter9_error_value"	DEC		
24	MD 66	"parameter10_error_value"	DEC		
25	MD 70	"parameter11_error_value"	DEC		
26					

Figure 7: Variable table 'DP-V1'

After activating **one** activation flag, the flag M4.0 '**Start_DP_V1**' can be used to start the respective DP-V1 request.

Only one activation flag may be set at a time!!!!

When the DP-V1 request has been processed by the Profibus slave, the flag M4.0 '**Start_DP_V1**' is automatically reset.

The occurrence of an error during the DP-V1 request is indicated by the flag '**error_RDREC**' (M4.6). A fault-free read request is indicated by the flag '**read_ok**' (M4.7) and a fault-free write request by the flag '**write_ok**' (M5.0).



The following table lists the different flags for program operation.

M2.0 – M3.0	Activation flags for FC100 – FC108
M4.0	Start command DP-V1
M4.6	Error during parameter request
M4.7	Read ok
M5.0	Write ok
MD30 – MD70	Parameter value / error code

The following table lists the connections between code accesses, functions, data blocks and activation flags.

Function	Blocks	Data	Code access	Activation flag
Read one code	FC100, DB60, DB61	DB100	L-C0061	M2.0
Write to one code	FC101, DB62, DB63	DB101	L-C0012	M2.1
Read x codes	FC102, DB64, DB65	DB102	L-C0061, L-C0012	M2.2
Write to x codes	FC103, DB66, DB67	DB103	L-C0012, L-C0013	M2.3
Read array code	FC104, DB68, DB69	DB104	L-C1510 subcodes 1-10	M2.4
Write to array code	FC105, DB70, DB71	DB105	L-C1511 subcodes 1-10	M2.5
Read multiparameter	FC106, DB72, DB73	DB106	L-C1509, L-C1510 subcodes 1-10	M2.6
Write to multiparameter	FC107, DB74, DB75	DB107	L-C0012, L-C1511 subcodes 2-5	M2.7
Read string code	FC108, DB76, DB77	DB108	L-C0200	M3.0

6.2.2. Functions FC100 – FC108

The functions (FC100 – FC108) have an identical structure.

The flag '**Start_DP_V1**' (M4.0 positive edge) can be used to start the processing of the respective DP-V1 request. Then the data is sent from the respective data block to the selected Profibus slave via SFB 53. In the data blocks the code accesses are stored according to the DP-V1 protocol.

When the transmission of the SFB 53 is complete (output **DONE** = TRUE), data fetching with the SFB 52 is started.

When the output of the SFB52 indicates '**VALID**' = TRUE, the read data is evaluated and the flags '**Start_DP_V1**' and '**done_WRREC**' (M4.4) are reset.



The occurrence of an error during the request is indicated by the flag '**error_RDREC**' (M4.6). A fault-free read request is indicated by the flag '**read_ok**' (M4.7) and a fault-free write request by the flag '**write_ok**' (M5.0).

```
U  "One_codenummer_read"
SPBN End1

U  "Start_DP_V1"
FP  "pos_Flanke_WRREC"
=   "send_trigger"

U  "send_trigger"          // start SFB WDREC
S  "start_SFB_WDREC"
R  "read_ok"
R  "write_ok"

CALL "WRREC", DB60
REQ := "start_SFB_WDREC"    // start DP Write
ID  := DW#16#7FC
INDEX := 47
LEN  := 10
DONE := "done_WRREC"
BUSY := M11.5
ERROR := M11.6
STATUS := MD12
RECORD := P#DB100.DBX0.0 BYTE 10

U  "done_WRREC"
S  "start_SFB_RDREC"      //start SFB RDREC
R  "start_SFB_WDREC"      //reset SFB WDREC

CALL "RDREC", DB61
REQ := "start_SFB_RDREC"   //start DP Read
ID  := DW#16#7FC
INDEX := 47
MLEN := 10
VALID := "reset_done_RDREC"
BUSY := M10.5
ERROR := M10.6
STATUS := MD16
LEN  := MW20
RECORD := P#DB100.DBX10.0 BYTE 10

U  "reset_done_RDREC"
R  "start_SFB_RDREC"      //reset SFB RDRECC
R  "Start_DP_V1"          //reset start DP_V1

L  "DB_One_codenummer_read".Response_ID_011 //checking if an error is occurred
L  129
==|
=  "error_RDREC"

L  "DB_One_codenummer_read".Response_ID_011 //Request without error
L  1
==|
=  "read_ok"

L  "DB_One_codenummer_read".Parametervalue011
T  "parameter1_error_value" //Parameter value or Error code
End1: NOP 0
```

Figure 7: Programming of function FC100



6.2.3. Data blocks DB100 – DB108

One of the data blocks DB100 – DB108 belongs to each of the functions FC100 – FC108. In the data blocks the individual DP-V1 parameter requests are stored, starting with the data sent with the SFB53 and followed by the data read with the SFB52. This is exemplarily shown for the DB100.

Address	Name	Type	Initial value	Comment
0.0		STRUCT		
+0.0	Request_Reference001	BYTE	B#16#0	Request Reference 0x00
+1.0	Request_ID_001	BYTE	B#16#1	Request ID 0x01 read
+2.0	Axis001	BYTE	B#16#0	Axis 0x00
+3.0	Number_Parameter001	BYTE	B#16#1	Number of Paramters 0x01
+4.0	Attribute001	BYTE	B#16#10	Attribute 0x10 Data typ
+5.0	Number_Subcodes001	BYTE	B#16#0	Number of Subcodes 0x00
+6.0	Index001	INT	24514	Index = 24575 - codenumber C0061
+8.0	Subindex001	INT	0	Subindex 0
+10.0	Request_Refernce011	BYTE	B#16#0	Request Reference 0x00
+11.0	Response_ID_011	BYTE	B#16#0	Response ID 0x01
+12.0	Axis011	BYTE	B#16#0	Axis 0x00
+13.0	Number_Paramater011	BYTE	B#16#0	Number of Parameters 0x01
+14.0	Format011	BYTE	B#16#0	Format 0x43
+15.0	Number_Values011	BYTE	B#16#0	Number of values 0x01
+16.0	Paramtervalue011	DINT	L#0	reading Paramter value
=20.0		END_STRUCT		

Figure 8: Data block DB100

The parameter request is stored in bytes 0.0 to 8.0 of the data block address.

- Request reference
- Request ID
- Axis
- Number of parameters
- Attribute
- Number of subcodes

The parameter response is stored in bytes 10.0 to 20.0.

- Request reference
- Response ID
- Axis
- Number of parameters
- Format
- Number of values
- Parameter value



6.3. Siemens system functions SFB 53 'WRREC' and SFB 52 'RDREC'

The interface of SFB 53 'WRREC' and SFB 52 'RDREC' is identical to the one of the function blocks 'WRREC' and 'RDREC' defined in the 'PROFIBUS Guideline PROFIBUS Communication and Proxy Function Blocks according to IEC 61131-3'.

6.3.1. 'SFB 53 WRREC'

SFB 53 'WRREC' (write record) is used to transmit a data record to a Profibus slave with the corresponding diagnostic address. The system block is provided with several inputs and outputs, which are described below:

Parameter	Declaration	Data type	Memory area	Description
REQ	INPUT	BOOL	E, A, M, D, L, Const.	REQ = 1: Transfer data record
ID	INPUT	DWORD	E, A, M, D, L, Const.	Logical address of the DP slave. For an output module, bit 15 must be set (e.g. for address 5: ID=DW#16#8005).
INDEX	INPUT	INT	E, A, M, D, L, Const.	Data record number 47
LEN	INPUT	INT	E, A, M, D, L, Const.	Maximum byte length of the data record to be transferred
DONE	OUTPUT	BOOL	E, A, M, D, L	Data record was transferred
BUSY	OUTPUT	BOOL	E, A, M, D, L	BUSY = 1: The write process is not yet terminated
ERROR	OUTPUT	BOOL	E, A, M, D, L	ERROR = 1: A write error has occurred
STATUS	OUTPUT	DWORD	E, A, M, D, L	Call ID (bytes 2 and 3) or error code
RECORD	IN_OUT	ANY	E, A, M, D, L	Data record



6.3.2. 'SFB 52 RDREC'

With the SFB52 'RDREC' (read record) you read a data record with the number INDEX from a DP slave that has been addressed via the diagnostic address. The system block is provided with several inputs and outputs which are described below:

Parameter	Declaration	Data type	Memory area	Description
REQ	INPUT	BOOL	E, A, M, D, L, Const.	REQ = 1: Transfer data record
ID	INPUT	DWORD	E, A, M, D, L, Const.	Logical address of the DP slave, For an output module, bit 15 must be set (e.g. for address 5: ID:=DW#16#8005).
INDEX	INPUT	INT	E, A, M, D, L, Const.	Data record number 47
MLEN	INPUT	INT	E, A, M, D, L, Const.	Maximum length in bytes of the data record information to be fetched
VALID	OUTPUT	BOOL	E, A, M, D, L	New data record was received and valid
BUSY	OUTPUT	BOOL	E, A, M, D, L	BUSY = 1: The read process is not yet terminated.
ERROR	OUTPUT	BOOL	E, A, M, D, L	ERROR = 1: A read error has occurred.
STATUS	OUTPUT	DWORD	E, A, M, D, L	Call ID (bytes 2 and 3) or error code
LEN	OUTPUT	INT	E, A, M, D, L	Length of the fetched data record information
RECORD	IN_OUT	ANY	E, A, M, D, L	Target area for the fetched data record.