SIEMENS

EN 50 170 Vol 2

Working with

PROFIBUS-DP

Device Description Data Files GSD

Version: 1.1

Date: August 23,99

Liability Exclusion
We have tested the contents of this document regarding agreement with the hardware and software described. Nevertheless, deviations can't be excluded, so that we don't guarantee complete agreement. However, the data in this document is checked periodically. Necessary corrections are included in subsequent editions. We gratefully accept suggestions for improvement.

Subject to technical changes.

Table of Contents

1 The PROFIBUS DP Device Description Data(E) File 4
1.1 Introduction 4
1.2 Who Needs a GSD(E) File? 5
1.3 Who does what with the GDS(E) File? 5
1.4 How does the Configuring Tool Process the GSD(E) Files? 5
1.5 Where Does the User Obtain the GSD(E) Files? 5
1.6 How Can a GSD(E) File be Created? 6
1.7 How Can a GSD(E) file be Checked for Correctness? 6
2 The Structure of a GSD(E) File 7
2.1 General PROFIBUS DP Key Words in the GSD(E) File 7
2.1.1 Slave-Related Key Words for PROFIBUS DP 13
3 Relationship of GSD(E) File, Configuring Tool, and DPS2/DPSE Software 22
3.1 Parameter Assignment 22
3.2 Configuring 27
4 Sample Files for GSD(E) File Entries 31

1 The PROFIBUS DP Device Description Data(E) File

1.1 Introduction

The provided information is based on PROFIBUS EN 50170 part 2 and the additional implementation guideline. The document was defined to our best knowledge, however, in case of any doubt EN 50170 and the implementation guideline takes precedence.

PROFIBUS DP according to EN 50170 and PROFIBUS DP/V1 support many possibilities to implement data exchange between bus master and the connected slaves. From the simplest slave that services only a few input/output channels up to the intelligent slave that handles preprocessing tasks, a PROFIBUS DP master can carry out data exchange. For that reason, field devices with PROFIBUS DP connection can be optimally adapted to the respective automation task. In order to cover this large variety safely and conveniently, the bus master (Class 1 master) needs the technical data of the connected field device in the form of a Device Description Data(E) file (GSD(E) file). The GSD(E) file is to be generated by the field device manufacturer as an ASCII file in the form of an electronic data sheet (for example, MSDOS Editor). In order to describe the technical details of a field device uniformly, a large number of key words were defined that uniquely define a certain attribute of the field device. This ensures, among other things, that different field devices by different manufacturers can exchange data with any master that conforms to standard. An accredited test lab tests the complete standard-conforming performance. Simple field devices can be described with a few key words.

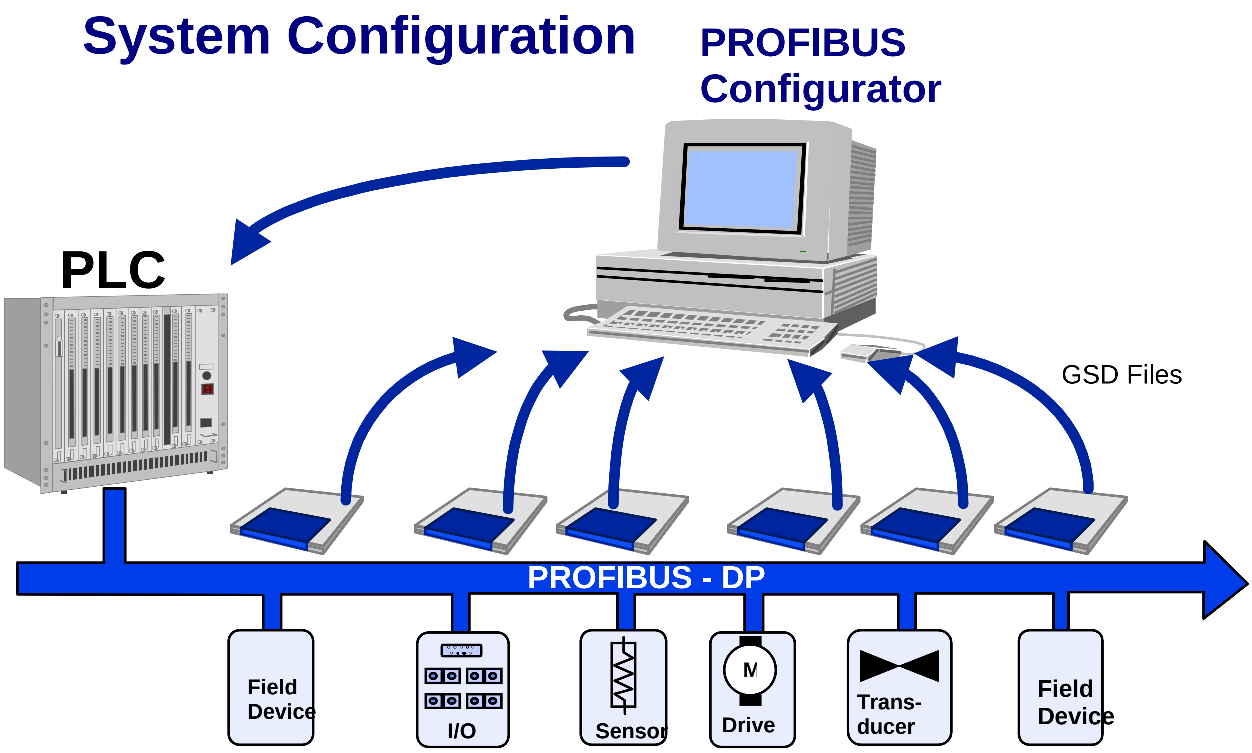


Figure 1-1: The Meaning of the Device Description Data(E) File

Essentially, the following data is included in a GSD(E) file:

- The supported transmission rates
- The length of the input/output data to be exchanged
- The meaning of the diagnostic data, and possibly of the user parameters
- Type of field device (compact station, modular station)
- Text assignments for symbolic configuring
- The supported services (sync/freeze mode, …)

1.2 Who Needs a GSD(E) File?

Every Class 1 master and all field devices with slave functionality have to be described by the manufacturer with a GDS(E) file.

1.3 Who does what with the GDS(E) File?

Configuring tools for the PROFIBUS DP master that is to be configured interpret the content of the GSD(E) files of the slaves, and from it, generate a master parameter set for the Class 1 master that handles the user data traffic. In part, Class 2 master functionalities are integrated, in order to load configuring data to the Class 1 master. A Class 2 master needs the GSD(E) files of a Class 1 master in order to recognize, for example, in which form the configuring data can be loaded to the Class 1 master. If a Class 1 master supports the services Upload and Download, the configuring data can be loaded to the Class 1 master online, and existing configuring data can be changed online (refer to Figure 1-1).
Based on the information from the GSD(E) files, the Class 1 master recognizes the following: the degree of expansion of the bus, which services the respective slave supports, and in which form the data is to be exchanged.

1.4 How does the Configuring Tool Process the GSD(E) Files?

GSD(E) files are needed during configuring and commissioning. Every manufacturer of a PROFIBUS DP Class 1 master makes a configuring tool available for configuring the Class 1 master that knows the internal data structure of the Class 1 master, and of the host system. When configuring a system, the GSD(E) files that are needed respectively are to be made known to the configuring tool. Usually, this is done by copying the GSD(E) files to the hard disk of the PC (the exact path indication is provided in the description of the configuring tool). When a system is configured, the configuring tool interprets the data of the GSD(E) file for the field device that was selected. In addition, validity checks are performed so that the configuring data is structured logically correct.
At the end of configuring, the user can select in what form the compiled configuring data is to be transferred to the Class 1 master (usually on a diskette, Flash-EPROM, or online). When commissioning the system, the interpretation of the GSD(E) file can provide information regarding errors that might occur.

1.5 Where Does the User Obtain the GSD(E) Files?

The manufacturer supplies the GSD(E) files for the respective field device, together with the respective product. Some manufacturers include GSD(E) files with the configuring tool. GSD(E) files that are not included in the configuring tool can be obtained as follows:

- through the Internet (address: http://www.ad.siemens.de contains all GSD(E) files of the Siemens corporation)
- through the Internet address of the PROFIBUS Trade Organization (PNO) (address: http://www.Profibus.com).

- on diskette, depending on the company

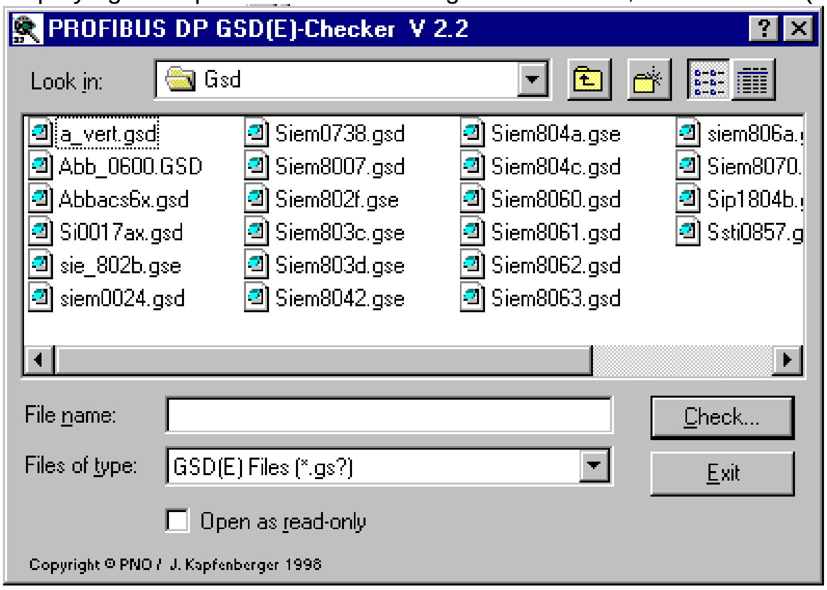
1.6 *How Can a GSD(E) File be Created?*

GSD(E) files are created as ASCII files with an ASCII Editor by describing each feature of the field device with a standardized key word.

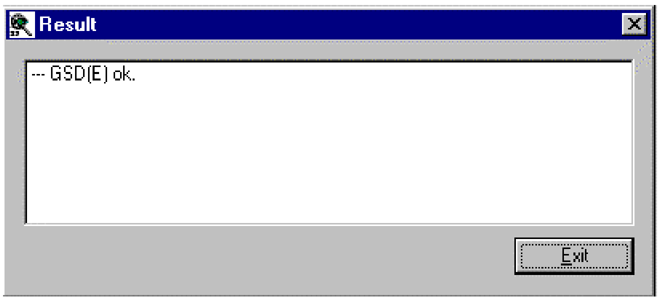
1.7 *How Can a GSD(E) file be Checked for Correctness?*

After the GSD(E) file has been created, it has to be checked with a GSD(E) Checker for correctness. The GSD(E) Checker is located on the Internet under the address http://www. Profibus.com, and can be loaded by the user. It is running under Windows 3.11, Windows 95, and Windows NT.

Example: A GSD(E) file is to be checked by the GSD(E) Checker:
Call the GSD(E) Checker by double-clicking on gsdchek1.exe. The GSD(E) Checker appears, displaying the input mask shown in Figure 1-2. Then, click the GSD(E) file that is to be checked.



If there are still errors in the file, the GSD(E) Checker indicates the number of the line. Using the key words, check what type of error the line contains. If the GSD(E) file is OK, the GSD(E) Checker indicates: GSD(E) OK.



2 The Structure of a GSD(E) File

A GSD(E) file exists once if it is configured independent of language (*.gsd). If it is generated in a certain language, it may exist more often. One GSD(E) file is then to be used per language, where only the parameters of the type Visible String may differ. The language-related GSD(E) files differ in the last letter of the extension (*.gs?).

Default (independent of language):	?=d
German	?=g
English	?=e
French	?=f
Italian	?=i
Portuguese	?=p
Spanish	?=s

Tabel 1-1: Language-Dependent GSD(E) Files

Example of a GSD(E) Name:
Abc_0008.gsd (this means the following:
Abc_ = 4 characters free to choose
0008 = Ident number 0008 assigned by the PNO, always 4 characters
.gs**d** = **d**efault. Language-independent GSD(E) file)

2.1 General PROFIBUS DP Key Words in the GSD(E) File

Each line starts with one of the key words below. The key words described below are standardized, and are to be used only in the described designation. Key words that are company-specific can be defined, and have to be interpreted that way. These self-defined key words are not to be read by configuring tools of other companies. A PROFIBUS DP GSD(E) file always starts with the key word **#Profibus_DP**.
The type ID specified for the keyword refers to the parameter with the same name. Regarding the parameters, the following differentiation is made:

- Mandatory (M) (absolutely required)
- Optional (O) (possible in addition)
- Optional with default = 0 if not present (D); (the key words marked with *) should always be specified because of the better readability of the GSD(E) file, even if the default setting is 0.
- At least one of the group (G) matches the corresponding baudrate.

GSD_Revision: (M starting with GSD_Revision 1)
Version ID of the GSD(E) file format
Type: Unsigned8
Example: GSD_Revision= 1

Vendor_Name: (M)
Vendor Name.
Type: Visible String (32)
Example: Vendor_Name= "Corp_ABC & Co"
Model_Name: (M)
Manufacturer Name (Controller Type) of the DP device. This name is indicated in the configuring tool.
Type: Visible String (32)
Example: Model_Name= "Modular I/O Station"

Revision: (M)
Version of the DP device.
Type: Visible String (32)
Example: Revision= "Version 01"

Revision_Number: (O starting with GSD_Revision 1)
Version ID . The value of the Revision_Number has to agree with the value of the Revision_Number in the slave-specific diagnosis if provided.
Type: Unsigned8 (1 bis 63)
Example: Revision_Number= 05

Ident_Number: (M)
Identifies the device type of the DP device. Each field device is characterized by an Ident number allocated by the PROFIBUS Trade Organization (PNO) which establishes a unique reference to the GSD(E) file, and thus to the technical data of the field device. Field device variants that can be described with **one** GSD(E) file, may use the same Ident number (for example, modular devices). Data exchange with a field device is possible only if the PROFIBUS DP device identifies itself clearly with the Ident number of the field device during system power-up (parameter assignment message).
Type: Unsigned16
Example: Ident_Number=0x00A2

Protocol_Ident: (M)
Protocol used for the DP devices.
Type: Unsigned8
    0: PROFIBUS-DP,
    16 to 255: manufacturer-specific
Example: Protocol_Ident= 0 ; here, a PROFIBUS DP device is described

Station_Type: (M)
DP device type.
Type: Unsigned8
    0: DP Slave,
    1: DP Master (Class 1)
Example: Stations_Type= 0 ; here, a PROFIBUS DP slave is described

FMS_supp: (D) [1)]
This device is a FMS/DP mixed device.
Type: Boolean (1: True)
Example: FMS_supp= 0 ; it is a pure DP device

Hardware_Release: (M)
Hardware release of the DP device.
Type: Visible String (32)
Example: Hardware_Release= "Hardware Release HW= 0.1"

Software_Release (M)
Software release of the DP device.
Type: Visible String (32)
Example: Software_Release= "Software Release SW= 1.01"

[1)] Although this key word is not mandatory, this detail should always be defined because of the easier readability of a GSD(E).

9.6_supp: (G)
The DP device supports the baudrate 9.6 kBaud.
Type: Boolean (1: True)
Example: 9.6_supp= 1 ; the device supports the specified baudrate

19.2_supp: (G)
The DP device supports the baudrate 19.2 kBaud.
Type: Boolean (1: True)
Example: 19.2_supp= 1 ; the device supports the specified baudrate

31.25_supp: (G)
The DP device supports the baudrate 31.25 kBaud.
Type: Boolean (1: True)
Example: 31.25_supp= 1 ; the device supports the specified baudrate

45.45_supp: (G)
The DP device supports the baudrate 45.45 kBaud.
Type: Boolean (1: True)
Example: 45.45_supp= 1 ; the device supports the specified baudrate

93.75_supp: (G)
The DP device supports the baudrate 93.75 kBaud.
Type: Boolean (1: True)
Example: 93.75_supp= 1 ; the device supports the specified baudrate

187.5_supp: (G)
The DP device supports the baudrate 187.5 kBaud.
Type: Boolean (1: True)
Example: 187.5_supp= 1 ; the device supports the specified baudrate

500_supp: (G)
The DP device supports the baudrate 500 kBaud.
Type: Boolean (1: True)
Example: 500_supp= 1 ; the device supports the specified baudrate

1.5M_supp: (G)
The DP device supports the baudrate 1.5 MBaud.
Type: Boolean (1: True)
Example: 1.5M_supp= 1 ; the device supports the specified baudrate

3M_supp: (G)
The DP device supports the baudrate 3 MBaud.
Type: Boolean (1: True)
Example: 3M_supp= 1 ; the device supports the specified baudrate

6M_supp: (G)
The DP device supports the baudrate 6 MBaud.
Type: Boolean (1: True)
Example: 6M_supp= 1 ; the device supports the specified baudrate

12M_supp: (G)
The DP device supports the baudrate 12 MBaud.
Type: Boolean (1: True)
Example: 12M_supp= 1 ; the device supports the specified baudrate

MaxTsdr_9.6: (G) (Value= 60)
This is the time that a responder needs as a maximum at a baudrate of 9.6 kBaud to respond to a request message.
Type: Unsigned16
Time base: bit time
Input: MaxTsdr_9.6= 60

MaxTsdr_19.2: (G) (Value= 60)
This is the time that a responder needs as a maximum at a baudrate of 19.2 kBaud to respond to a request message.
Type: Unsigned16
Time base: bit time

MaxTsdr_31.25: (G) (Value= 60)
This is the time that a responder needs as a maximum at a baudrate of 31.25 kBaud to respond to a request message.
Type: Unsigned16
Time base: bit time

MaxTsdr_45.45: (G) (Value= 60)
This is the time that a responder needs as a maximum at a baudrate of 45.45 kBaud to respond to a request message.
Type: Unsigned16
Time base: bit time

MaxTsdr_93.75: (G) (Value= 60)
This is the time that a responder needs as a maximum at a baudrate of 93.75 kBaud to respond to a request message.
Type: Unsigned16
Time base: bit time

MaxTsdr_187.5: (G) (Value= 60)
This is the time that a responder needs as a maximum at a baudrate of 187.5 kBaud to respond to a request message.
Type: Unsigned16
Time base: bit time

MaxTsdr_500: (G) (Value= 100)
This is the time that a responder needs as a maximum at a baudrate of 500 kBaud to respond to a request message. (refer to DIN 19245 Part 1\4.91 Section 4.1.7).
Type: Unsigned16
Time base: bit time

MaxTsdr_1.5M: (G) (Value= 150)
This is the time that a responder needs as a maximum at a baudrate of 1.5 MBaud to respond to a request message (refer to DIN 19245 Part 1\4.91 Section 4.1.7).
Type: Unsigned16
Time base: bit time

MaxTsdr_3M: (G) (Value= 250)
This is the time that a responder needs as a maximum at a baudrate of 3 MBaud to respond to a request message.
Type: Unsigned16
Time base: bit time

MaxTsdr_6M: (G) (Value= 450)
This is the time that a responder needs as a maximum at a baudrate of 6 MBaud to respond to a request message..
Type: Unsigned16
Time base: bit time

MaxTsdr_12M: (G) (Value= 800)
This is the time that a responder needs as a maximum at a baudrate of 12 MBaud to respond to a request message.
Type: Unsigned16
Time base: bit time

Redundancy: (D)
This value indicates whether a device supports redundant transmission or not.
Type: Boolean
0: No, 1: Redundancy supported.
Example: Redundancy= 0

Repeater_Ctrl_Sig: (D) [2)]
Here, the level of the connector signal CNTR-P is specified.
Type: Unsigned8
0: Not connected, 1: RS485, 2:TTL
Example: Repeater_Ctrl_Sig= 2

[2)] Although this key word is not mandatory, this detail should always be defined because of easier readibility.

24V_Pins: (D) [2)]
Here, the meaning of the connector signals M24V and P24V is specified.
Type: Unsigned8
0: Not connected, 1:Input, 2:Output
Example: 24V_Pins= 0

Implementation_Type: (O starting with GSD_Revision 1) [2)]
Here, a description is provided of what standard implementation is used in the DP slave; for example, standard software solution, controller solution, or ASIC solution. The name is specifed by the manufacturer of the standard solution. From this detail, configuring tools can already perform checks.
Type: Visible String (32)
Example: Implementation_Type= "SPC3 solution" or "Software solution"; when using the key word SPC3, the configuring tool COM PROFIBUS locks the first User_Prm_Byte for the user.

Bitmap_Device: (O starting with GSD_Revision 1)
Here, the file name (*.DIB) of the bitmap file is specified in DIB format (70*40 pixel (width*height) 16 colors), which normally contains the symbolic representation of the device. Depending on the configuring tool used, the bit map that is used is to be copied either to a certain directory, or the exact path is to be indicated -including the bitmap- prior to being used. Regarding this, read the description of the configuring tool used.
Type: Visible String (8)
Example: Bitmap_Device= "OK_state"

Bitmap_Diag: (O starting with GSD_Revision 1)
Here, the file name (*.DIB) of the bitmap file is specified in DIB format (70*40 pixel (width*height) 16 colors), which contains the symbolic representation of the device if there is a diagnosis. Depending on the configuring tool used, the bit map that is used is to be copied either to a certain directory, or the exact path is to be indicated -including the bitmap- prior to being used. Regarding this, read the description of the configuring tool used.
Type: Visible String (8)
Example: Bitmap_Diag= "Diag_sta"

Bitmap_SF: (O starting with GSD_Revision 1)
Here, the file name (*.DIB) of the bitmap file is specified in DIB format (70*40 pixel (width*height) 16 colors), which contains the symbolic representation of the device in special operating modes. The meaning is manufacturer-specific. Depending on the configuring tool used, the bit map that is used is to be copied either to a certain directory, or the exact path is to be indicated -including the bitmap- prior to being used. Regarding this, read the description of the configuring tool used.

Type: Visible String (8)
Example: Bitmap_SF= "SF_state"

2.1.1 Slave-Related Key Words for PROFIBUS DP

Freeze_Mode_supp: (D)[1])

The DP device supports the freeze mode. During power-up, the parameter assignment message specifies whether the slave is to support the freeze mode. The freeze mode is activated with a global control message and causes the inputs of the slave to be "frozen" in the momentary state. DP slaves that support the freeze mode have to ensure that in the next data cycle after the freeze control command, the values of the inputs that were frozen last are transmitted to the bus.
Type: Boolean (1: True)
Example: Freeze_Mode= 1 ;Freeze Mode is supported in the slave

Sync_Mode_supp: (D) [1)]
The DP device supports the sync mode. During power-up, the parameter assignment message specifies whether the slave is to support the sync mode. The sync mode is activated with a global control message and causes the slave to keep the outputs in the momentary state.
Type: Boolean (1: True)
Example: Sync_Mode= 1 ;Sync-Mode is supported in the slave

[1] Should always be specified

Field devices that support the sync/freeze mode can be combined into groups.

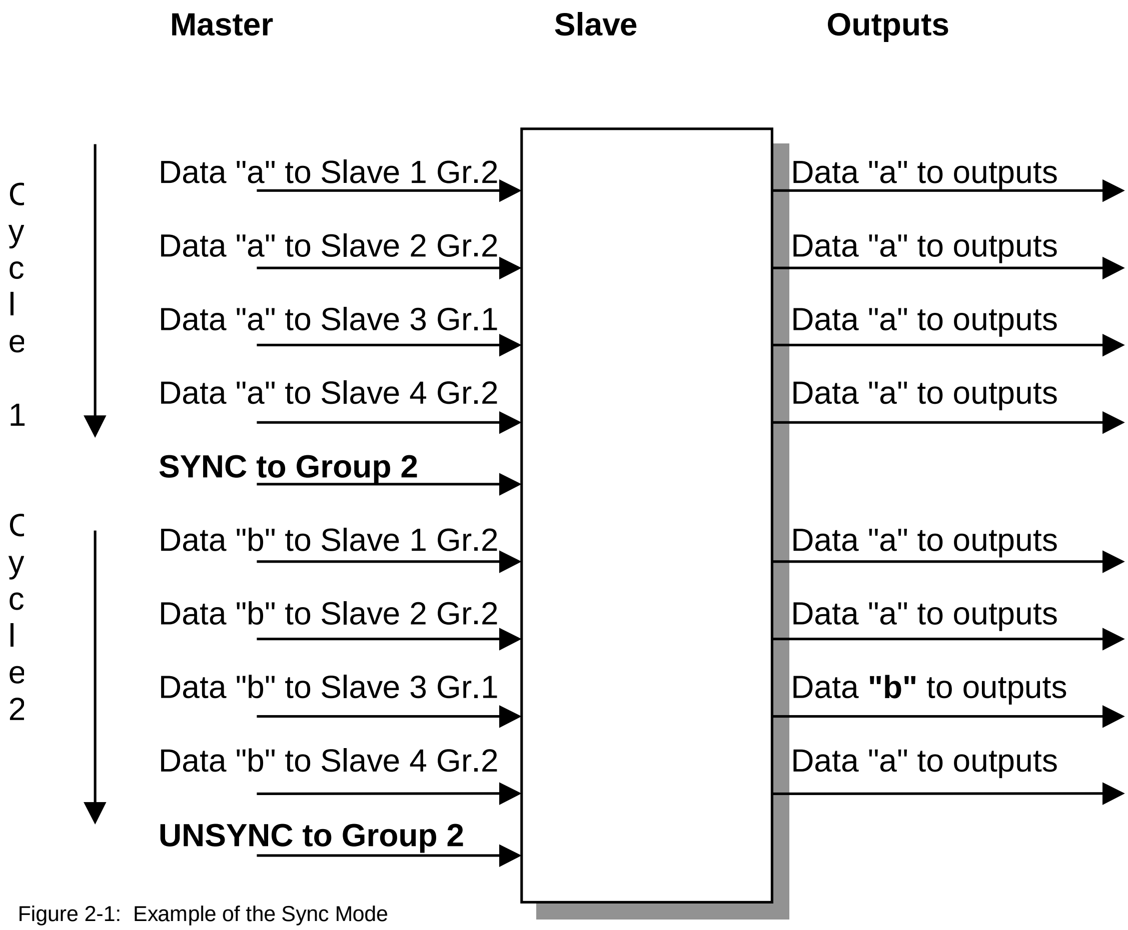


Figure 2-1: Example of the Sync Mode

In the next bus cycle after the UNSYNC command, the outputs are updated again.

Auto_Baud_supp: (D)[2]
The DP device supports the automatic transmission rate recognition. It automatically sets itself to the transmission rate specified by the master.
Type: Boolean (1: True)
Example: Auto_Baud_supp= 1 ; the function is supported

Set_Slave_Add_supp: (D) [2]
The DP device supports the function Set_Slave_Add for setting the slave address via the PROFIBUS.
Type: Boolean (1: True)
Example: Set_Slave_Add_supp= 1 ; the function is supported

[2] should always be specified

Fail_Safe: (D starting with GSD_Revision 1)
Here it is specified whether the DP slave accepts a data message without data instead of a data message with data = 0 in the CLEAR mode of the DP master (Class 1). As a matter of standard, the PROFIBUS DP master sets the outputs to zero if it is in the CLEAR mode. Here, the user can specify the preassignments of the outputs.
Type: Boolean (1: True)
Example: Fail_Safe=1 ; means the slave accepts a data message without data in the
; Clear mode

Max_Diag_Data_Len: (M starting with GSD_Revision 1) [2]
Here, the maximum length of the diagnostic information (Diag_Data) is specified. At least, the 6 octets of the system diagnosis have to be always specified. This key word should always be indicated so that the bus master can optimize its memory location.
Type: Unsigned8 (6 - 244)
Example: Max_Diag_Data_Len= 10 ; the field device supplies 4 user diagnoses

Example: Max_Diag_Data_Len= 78 ; The field device suppliest 70 device related
; user diagnostics + 6 Bytes
; standard diagnostics + 2 headerbytes

Content in diagnosis telegram:

Bytes 1 - 6	Standard Diagnosis		
Byte 7	**0011 1111**	(Headerbyte 1)	; (63 bytes User diags part I, including Byte 7, ; Headerbyte
Bytes 8 - 69	User diags (part I)		; --> 62 bytes user diag (user specific)
Byte 70	**0000 1001**	(Headerbyte 2)	; 9 bytes User diags part II, incuding byte 70 ; Headerbyte
Byte 71 - 78	User diags (part II)		; - -> 8 bytes user diag (user specific)

Max_User_Prm_Data_Len: (O starting with GSD_Revision 1)
Here, the maximum length of the User_Prm_Data is specified. The length of the transferred user parameters can have the specified maximum **or less**. They can also exist of User_Prm_Data and Ext_Module_Prm_Data.
The definition of this key word **excludes** the evaluation of User_Prm_Data_Len.
Type: Unsigned8 (0 - 237)
Example: Max_User_Prm_Data_Len= 120 ; as a maximum, 120 user parameters are
; possible from the field device

Modul_Offset: (D starting with GSD_Revision 1)
Here, the slot number is specified that is to appear as the first slot number in the configuring tool at configuring (is used to improve representation). In the case of modular devices, manufacturers sometimes designate as modules such units that the PROFIBUS DP can't address directly (such as PROFIBUS interface, power supply, CPU).
Type: Unsigned8
Example: Module_Offset=3 ; representation of the I/O modules starts withOffset 3

Slave_Family: (M starting with GSD_Revision 1)
In order to be able to find the individual slaves more easily when configuring a plant, the slaves are combined into families. The slave families are visualized for the user with the configuring tool. With the key word Slave_family, the DP slave is assigned to a function class. The family name is structured hierarchically. In addition to the main family, subfamilies can be formed that are attached with "@". A maximum of 3 subfamilies can be defined. Assignment to a slave family

facilitates finding a GSD(E) file when configuring, since configuring tools file the stored GSD(E) files according to the Slave_Family.

Example: Slave_Family=3@Digital@24V
The following main families are specified:
0: General (no assignment to the following categories possible)
1: Drives
2: Switching devices
3: I/Os
4: Valves
5: Controllers
6: MMIs
7: Encoders
8: NCc/RCs
9: Gateways
10: PLCs
11: Ident systems
12-255: reserved
Type: Unsigned8
Example: Slave_Family=7 ; the GSD(E) file is stored under the category Encoders

User_Prm_Data_Len: (D)
Here, the length of the user-specific parameters (User_Prm_Data) is specified. When this keyword is defind and no Max_User_Prm_Data_Len is defined the user parameters have to have exactly that specified length. Please note that some ASICs need user-specific data.
Type: Unsigned8
Example: User_Prm_Data_Len= 5

User_Prm_Data: (O)
Type: Octet String
Meaning: Manufacturer-sepcific field. Provides the default value for User_Prm_Data. If this parameter is used, its length has to agree with the User_Prm_Data_Len.
Example: User_Prm_Data= 0x00,0x10,0xdf,0x00,0x23

Min_Slave_Intervall: (M)
This time specifies the minimum interval between two poll cycles for the DP device.
Type: Unsigned16
Time base: 100 µs
Example: Min_Slave_Intervall= 10 ; corresponds to a poll cycle of 1ms

The maximum time for the Min_Slave_Interval at the baudrates is:
up to 1500 kbit/s max. 20 (2 ms)
at 12 000 kbit/s max. 6 (0.6 ms)

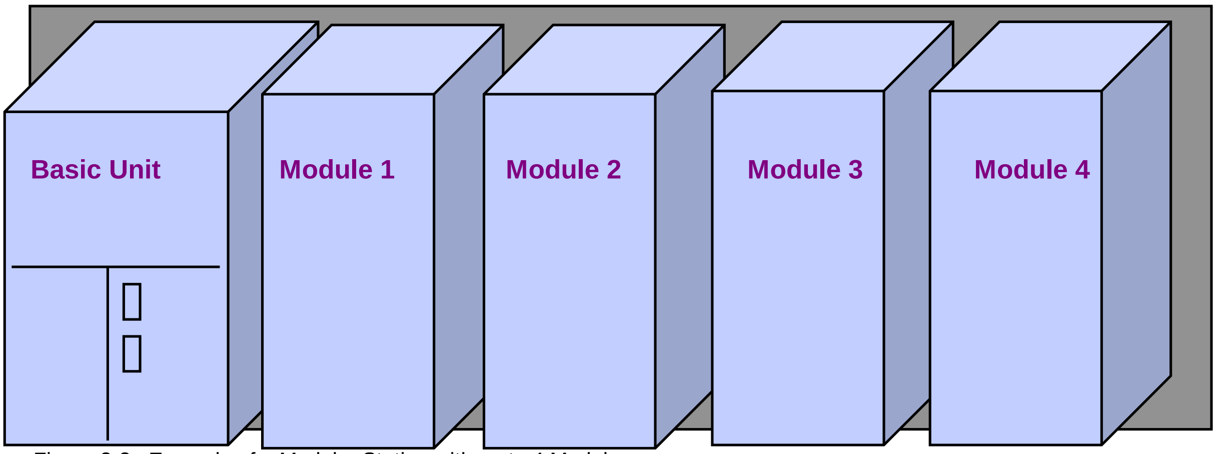


Figure 2-2: Example of a Modular Station with up to 4 Modules

Modular_Station: (D) [3]
Here it is specified whether the DP device is a modular station. Modular stations can be created from several modules. A list of the different modules that can be used in the field device is to be specified in the GSD(E) file. A module is either a physical unit (refer to Figure 2-1) or a logical unit. When configuring, the configuring engineer can symbolically select the modules defined in the GSD(E) file, and thus set up the modular station.
Type: Boolean (0: compact device, 1: modular device)

Max_Module: (M if Modular_Station)
Here, the maximum number of the modules is specified that can be inserted in the described device. The list of modules provided in the GSD(E) file may be longer.
Type: Unsigned8
Example: Max_Module= 4 ; 4 modules can be inserted

Max_Input_Len: (M if Modular_Station)
Here, the maximum length of the input data of a modular station is specified in bytes.
Type: Unsigned8
Example: Max_Input_Len= 100

Max_Output_Len: (M if Modular_Station)
Here, the maximum length of the output data of a modular station is specified in bytes.
Type: Unsigned8
Example: Max_Output_Len= 100

Max_Data_Len: (M if Modular_Station
Here, the largest sum of the lengths of the input/output data of a modular station is specified in bytes.
Type: Unsigned16
Example: Max_Data_Len= 200

Unit_Diag_Bit: (O)
To display manufacturer-specific status- and error messages of a DP slave centrally, it is possible to assign a text (Diag_Text) to a bit in the device-related diagnostic field.
Parameters used:
Bit:
Type: Unsigned16
Meaning: Bit position in the device-related diagnostic field

[3] should always be specified

(LSB in the first byte is Bit 0).
Diag_Text:
Type: Visible String (32)
Example: Unit_Diag_Bit(0x12)="Short circuit on Channel 0...7" ; Bit No. 18 decimal means that a
; short circuit is present in the
; area of Channel 0 ... 7

Unit_Diag_Area: (O)
Between the key words Unit_Diag_Area and Unit_Diag_Area_End, the assignment of values in a bit field in the device-related diagnostic field to texts (Diag_Text) is specified.
Parameters used:
First_Bit:
Type: Unsigned16
Meaning: first bit position of the bit field
(LSB in the first byte is Bit 0)
Last_Bit:
Type: Unsigned16
Meaning: Last bit position of the bit field. The bit field may consist of
16 bits maximum.
Value:
Type: Unsigned16
Meaning: Value in the bit field

Diag_Text:
Type: Visible String (32)
Example:
Unit_Diag_Area = 0 to 5 ;
Value(0) = "Faultless"
Value(1) = "Error on Input 0 to 23"
Value(2) = "Error on Output 0 to 15"
Value(3) = "24V failed"
Unit_Diag_Area_End

Module: (M) (refer also to Chapter 3.2)
Between the key words Module and EndModule, the following is provided: IDs of a DP compact device and the IDs of a module of a modular DP slave are specified; manufacturer-specific error types in the channel-related diagnostic field are specified; the Ext_User_Prm_Data is described. If, in the case of modular slaves, empty slots are to be defined as blank module (ID(s) 0x00), the empty module has to be defined. Otherwise, empty slots will not show up in the configuration data.
If the key word Channel_Diag is used outside the key words Module and EndModule, the same manufacturer-specific error type in the channel-related diagnostic field for all other modules.
If the key words Ext_User_Prm_Data_Ref or Ext_User_Prm_Data_Const are used outside the key words Module and EndModule, the associated User_Prm_Data area refers to the entire device , and the data in the parameter Offset to the entire User_Prm_Data. This User_Prm_Data area is placed at the start of the User_Prm_Data.
The module-specific User_Prm_Data is directly appended to the device-specific User_Prm_Data in the sequence in which the associated modules were configured. If the key words Ext_User_Prm_Data_Ref or Ext_User_Prm_Data_Const are used within the key words Module and EndModule, the data in the parameter Offset refers only to the start of the User_Prm_Data area that is assigned to this module.
Parameters used:

Mod_Name:
Type: Visible String (32)
Meaning: Name of a module used in a modular DP station, or device designation of a compact

DP slave.

Config:
Type: Octet String (17)
Type: Octet String (244) (O starting with GSD_Revision 1)
Meaning: Here, the ID or IDs of the module of a modular DP slave or of a compact DP device is/are specified.

Module_Reference: (O starting with GSD_Revision 1)
Type: Unsigned16
Meaning: Here, the reference of the module description is specified. This reference has to be unique for a device (same Ident_Number). This type of referencing is useful in order to make language-independent configuring possible in a language-dependent system, or in order to recognize modules.

Examples:
```
Modular_Station=1               ;modular station
Max_Module=4
Module="Leerslot" 0x00          ; 0 is the ID for an empty slot (for example, PS module, etc.)
EndModule
                                ; The selection possibilities between Module ... EndModule
                                ; are displayed in the configuring tool
Module="2 Bytes Output" 0x21    ; in plain text
EndModule

Module="2 Bytes Input" 0x11     ;
EndModule
```

Ext_Module_Prm_Data_Len: (O starting with GSD_Revision 1)
Type: Unsigned8
Meaning: Here, the length of the associated User_Prm_Data is defined (the user parameters of a special module)
Channel_Diag: *(O)*
With the key word Channel_Diag, the assignment of manufacturer-specific error types (Error_Type) in the channel-related diagnostic field to the texts (Diag_Text) is specified.

Parameters used:
Error_Type:
Type: Unsigned8 (16 <= Error_Type <= 31)

Diag_Text:
Type: Visible String(32)

Ext_User_Prm_Data_Ref: (O starting with GSD_Revision 1)
Here, the reference to a User_Prm_Data description is specified. The definition of this key word excludes the evaluation of User_Prm_Data. If areas overlap when describing User_Prm_Data, the area defined last in the GSD(E) file has priority.
Parameters used:
Reference_Offset:
Type: Unsigned8
Meaning: Here, the offset is defined within the associated part of the User_Prm_Data.

Reference_Number:
Type: Unsigned8
Meaning: This reference number has to be the same as the reference number

that is defined in the User_Prm_Data description.

Ext_User_Prm_Data_Const: (O starting with GSD_Revision 1)
Here, a constant part of the User_Prm_Data is specified. The definition of this key word excludes the evaluation of User_Prm_Data. If the areas overlap when describing the User_Prm_Data , the area defined last in the GSD(E) file has priority.
Parameters used:
Const_Offset:
Type: Unsigned8
Meaning: Here, the offset is defined within the associated part of the User_Prm_Data.

Const_Prm_Data:
Type: Octet String
Meaning: Here, constants or pre-assignments are defined within the User_Prm_Data.

ExtUserPrmData: (O starting with GSD_Revision 1)
Between the key words ExtUserPrmData and EndExtUserPrmData, a parameter of the User_Prm_Data is described. The definition of this key word excludes the evaluation of User_Prm_Data.
Parameters used:
Reference_Number:
Type: Unsigned8
Meaning: Here, the reference of the User_Prm_Data description is specified. This refeence has to be unique.

Ext_User_Prm_Data_Name:
Type: Visible String (32)
Meaning: Plain text description of the parameter

Data_Type_Name:
Type: Visible String (32)
Meaning: Name of the data type of the parameter described

Default_Value:
Type: DataType (has to correspond to the Data_Type_Name)
Meaning: Default value of the parameter described

Min_Value:
Type: Data_Type (has to correspond to the Data_Type_Name)
Meaning: Minimum value of the parameter described

Max_Value:
Type: Data_Type (has to correspond to the Data_Type_Name)
Meaning: Maximum value of the parameter described

Allowed_Values:
Type: Data_Type_Array (16) (has to correspond to the Data_Type_Name)
Meaning: Permissible values of the parameter described

Prm_Text_Ref:
Type: Unsigned8
Meaning: This reference number has to be the same as the reference number defined in the PrmText description.

PrmText:
Between the key words PrmText and EndPrmText, possible values of a parameter are described. Texts are assigned to these values for symbolic configuring.
Parameters used:
Reference_Number:
Type: Unsigned8
Meaning: Here, the reference of the PrmText description is specified. This reference has to be unique.
Text_Item:
Parameters used:
Prm_Data_Type:
Type: Data_Type (has to correspond to the Data_Type_Name in the parameter description)
Meaning: Here, the value of the parameter is specified that is to be described.
Text:
Type: Visible String (32)
Meaning: Description of the parameter value

Example of Reference Texts:

```
ExtUserPrmData=9 "Threshold reached" ; Text Reference 9
        Bit (4-5)  2  0000-0003          ;  Bits 4 to 5 in the User Octet No. x mean, that a
                                         ;threshold that has been reached is to be displayed.
Prm_Text_Ref=1                ; The value ranges from 0..3,  and the default setting  =2
        .                                ;The reference text that is located under PrmText = 1 is
                                         ; displayed in the configuring tool.
        .
PrmText= 1
        Text (0)= "Threshold Limit 100"
        Text (1)= "Threshold Limit 200"
        Text (2)= "Threshold Limit 300"
        Text (3)= "Threshold Limit 400"
EndPrmText
Ext_User_Prm_Data_Ref(x)=9               ;Text Reference 9
                                         ;The xth user byte is influenced (is explained in Chapter 3.1)
```


3 Relationship of GSD(E) File, Configuring Tool, and DPS2/DPSE Software

3.1 Parameter Assignment

The performance of a field device can be determined through settings with a dip switch. Assigning parameters with a handheld provides a more convenient solution. However, with PROFIBUS DP, the device attribute and the performance of the modules defined within a device can also be described once by using the entries in a GSD file. During configuring, the definitive selection of the defined parameters is made, and thus the definitive performance of the field device. When the system is powered up, the bus master sends a parameter assignment message to the configured slaves. The first 7 octets in the parameter assignment message (from the master to the slave) are defined by the system. Starting with Octet 8 up to Octet 244, user-specific information can be defined which is also to be evaluated user-specific. In the user parameters, for example, setting parameters and value ranges can be defined. The slave's response to a parameter assignment message is always "E5H" (positive acknowledgement). Before a field device branches into data exchange, a check is made with a diagnostic query whether the parameters were assigned successfully.
To make configuring easier for the configuring engineer -that is, the configuring engineer doesn't have to know the meaning of the bits and bytes for the field device- plain texts can be assigned to the defined bit combinations. Below, a few examples are provided that describe the relationship of GSD(E) file, configuring tool, and DPS2/DPSE software (the DPS2/DPSE software is a software available from Siemens that appreciably simplifies controlling the PROFIBUS ASIC SPC3).

Example 1 General Parameter Assignment (not module-related))

Entry in the GSD(E) file.

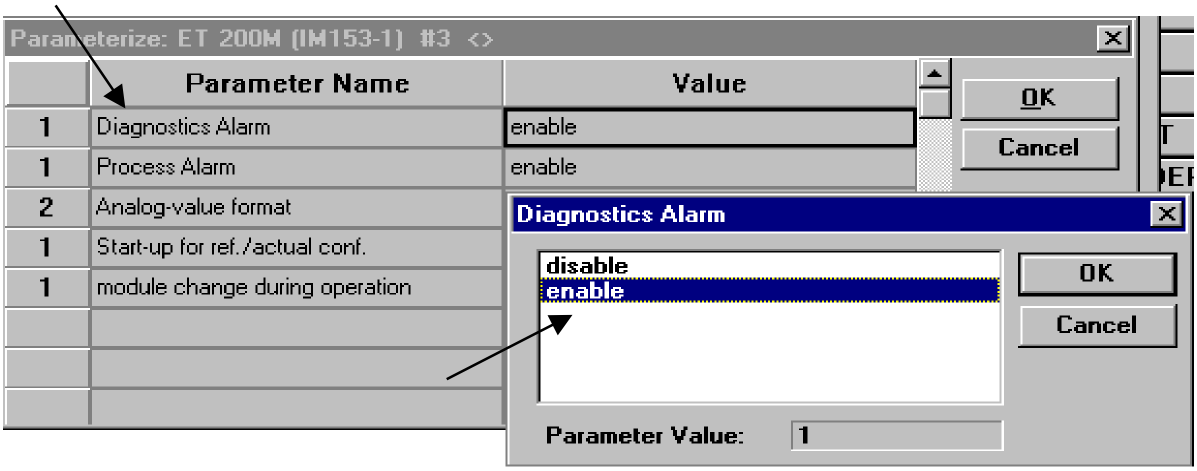
```
PrmText= 1                          ; In the configuring tool, the reference text is displayed,
                                    ; under PrmText = 1
        Text (0)= "disable"         ; Value 0 means disable
        Text (1)= "enable"          ; Value 1 means enable
EndPrmText

ExtUserPrmData=1 "Diagnostic Alarm"
        Bit (5)  1  0-1             ; Bit 5 in this reference refers to PrmText=1

                                    ; The bit has the value range 0..1
                                    ; Bit 5 is preassigned 1
Prm_Text_Ref=1
EndExtUserprmdata                   .

Max User Prm Data Len= 50           ; 50 user-specific parameters maximum can be
                                    ; transmitted in the parameter assignment message
Ext_User_Prm_Data_Const (0)= \      ; The first 4 octets are assigned the values 0, 1, 2, 3
0x00,0x01,0x02,0x03                 ; constantly. The value 3 is to be replaced according to
                                    ; the text reference. Bit 5 in Byte 2 = 0
Ext_User_Prm_Text_Ref(2)=1          : Text Reference 1 belongs to Byte 2
```

Explanation of the example above:

In general: The texts that are referenced have to be located in front of the reference.

Through the instruction Ext_User_Prm_Data_Const(0), the user parameters are preassigned with a constant number sequence.
During configuring, the user wants to specify whether a diagnostic alarm (depending on Octet 2 of the user parameters (Ext_User_Prm_Const(0)) is to be generated. The default selection is that no diagnostic alarm is to be generated. If the user wants to change this, he can symbolically select the response according to previous referencing (**Ext_User_Prm_Text_Ref(2)=1 on ExtUserPrmData=1 "Diagnostic Alarm" on PrmText= 1**).

In the configuring tool (here: COM PROFIBUS by Siemens), the following is displayed when configuring the user parameters:



PrmText=9
Text(0)="Deactivated"
Text(34)="Current 0..20 mA"
Text(35)="Current 4..20 mA"
EndPrmText

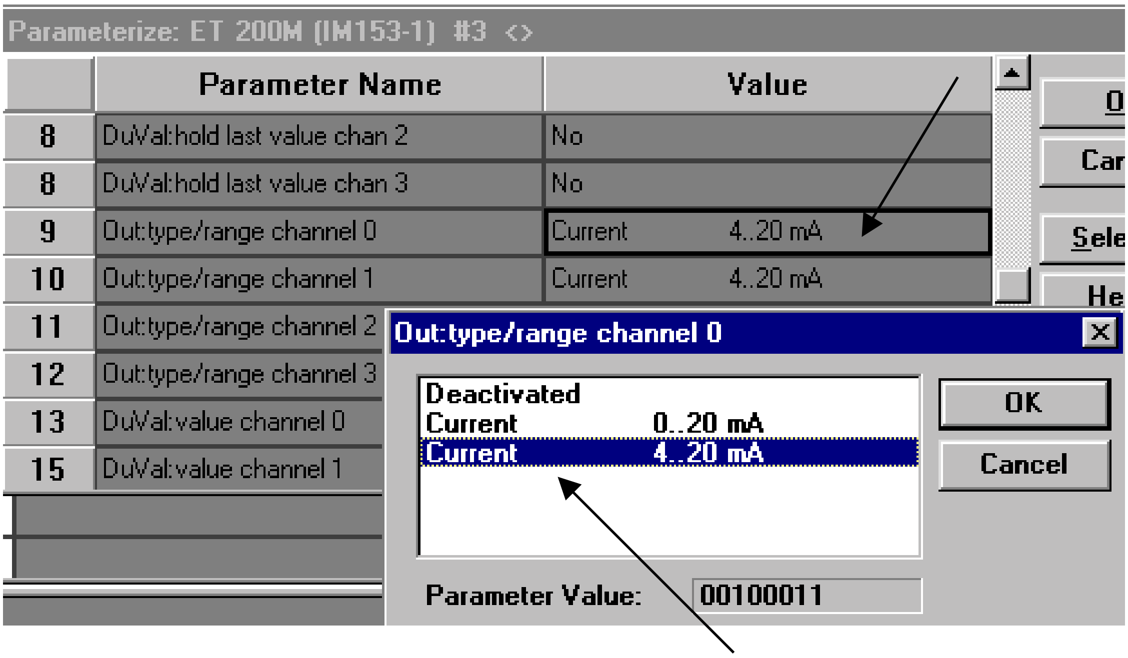
ExtUserPrmData=207 "Out:type/range channel 0"
BitArea(0-7) 35 000-035
Prm_Text_Ref=9
EndExtUserPrmData

Module="6ES7 332-5RD00-0AB0 2AO" 0x83,0x41,0x00,0x25,0xD8
Ext_Module_Prm_Data_Len=21 : 21 octets follow as pre-assignment
Ext_User_Prm_Data_Const(0)= \
0x15,0x5F,0x04,0x00,0x10,0x00,0x00,0x00,0x00,**0x19**,0x19,0x00,0x00,0x00,0x00,\
0x00,0x00,0x00,0x00,0x00,0x00
Ext_User_Prm_Data_Ref(9)=207
EndModule

Explanation of the previous example:

In general: The texts that are referenced have to be in front of the reference.
Through the instruction Ext_User_Prm_Data_Const(0), the user parameters are preassigned a constant number sequence.
To the pre-assignment in user parameter Octet 9 according to referencing (**Ext_User_Prm_Data_Ref(9)=207**, count-wise starting with 0 to **ExtUserPrmData=207 "Out:type/range channel 0"** - all values between 0 .. 35 refer to **PrmText=9-**), the final value is to be assigned. Texts are stored for the values 0,34,35. After this step is completed, the configuring tool enters the hex value for **Current 4…20mA** in the user parameters.

The associated part in the configuring tool looks like this:



For the PROFIBUS ASIC SPC3, Siemens offers a software that provides a simple interface to the user, and relieves him <<it?>> of the register descriptions of the ASIC. The SPC3 evaluates the standard parameters autonomously. Only if user parameters are defined do they have to be evaluated by the user, and the ASIC has to be informed of the result of the check (.._OK or ..._NOK). The designations in capitals are predefined macros.

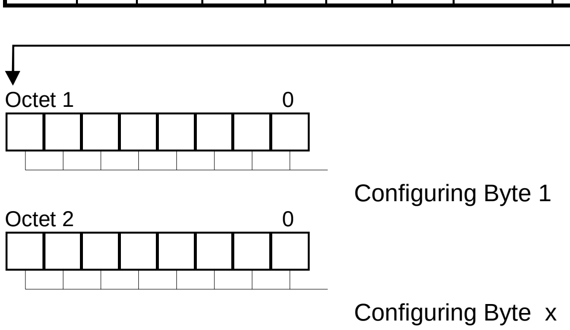
When using the DPS2/DPSE software for the PROFIBUS ASIC SPC3, the relevant code location looks like this, for example:

```
.
.
if(DPS2_GET_IND_NEW_PRM_DATA())
{   /*=== New parameter  data ===*/
        UBYTE   SPC3_PTR_ATTR * prm_ptr;
        UBYTE   param_data_len, prm_result;
        UBYTE   ii;

        prm_result = DPS2_PRM_FINISHED;
        do
        { /* Check parameter until no conflict behavior */
                prm_ptr = DPS2_GET_PRM_BUF_PTR();
                param_data_len = DPS2_GET_PRM_LEN();

                /* data_length_netto must be 28 (7 bytes norm + 21 byte user-part) */
                if (param_data_len == 28)
                {
                        if(!user_set_user_prm_values(prm_ptr)) /* call user-specific function */
                        {
                                /* an error was detected in the user-prm-data */
                                prm_result = DPS2_SET_PRM_DATA_NOT_OK();
                        }
                        else
                        {
                                /* user_prm_data is correct, look for range channel 0 */
                                switch(prm_ptr[17])
                                {
                                case 0:
                                        /* deactivated */
                                        break;
                                case 34:
                                        /* current 0..20 mA */
                                        break;
                                case 35:
                                        /* current 4..20 mA */
                                        break;
                                }

                                prm_result = DPS2_SET_PRM_DATA_OK();
                        }
                }
                else
                {
                        prm_result = DPS2_SET_PRM_DATA_NOT_OK();
                }
        } while(prm_result == DPS2_PRM_CONFLICT);
```

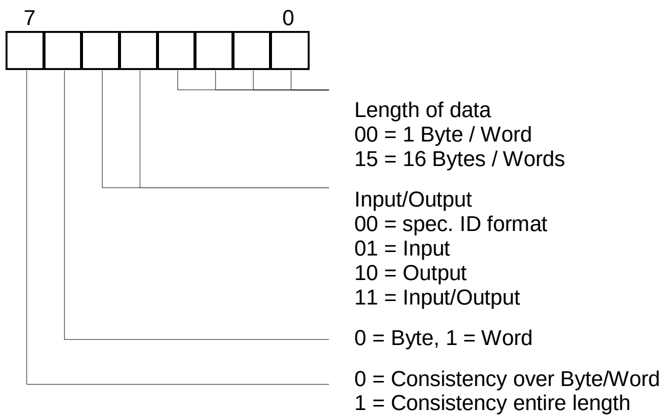

3.2 Configuring

After the parameter assignment, the field device expects a configuring message. With the configuring data during system power-up, the slave is informed of the number of the input/output data and/or any device-specific configuration. If the transmitted configuration is OK, the slave responds with “E5H”. In the GSD(E) file, a station is described either as a compact station (fixed I/O length can’t be changed), or as a modular station (one or several modules are combined into a station). The data length in both directions, specified during configuring, is monitored by the master as well as the slave at every data exchange. If there is a deviation, the data exchange is cancelled and a diagnostic message is issued.
The configuration of a field device can be described with the general and the special ID format. Below, only an example for the general ID format is provided.

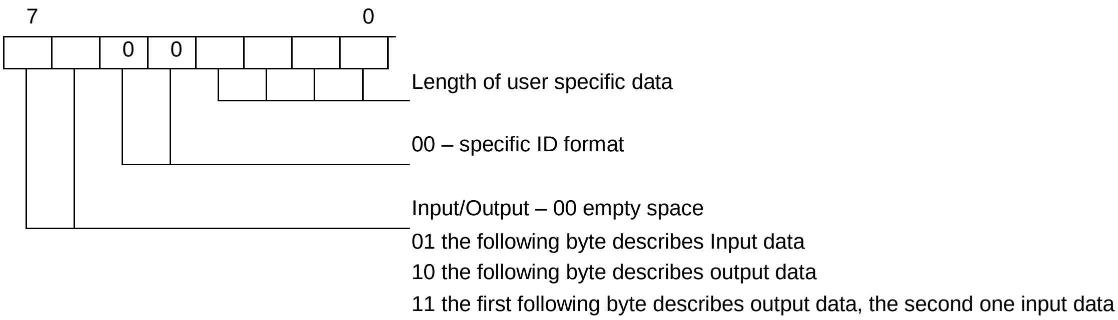
SD	LE	LEr	SD	DA	SA	FC	DSAP	SSAP	DU	FCS	ED
68H	x	x	x	8x	8x	x	62/3EH	62/3EH	x ..	x	16H

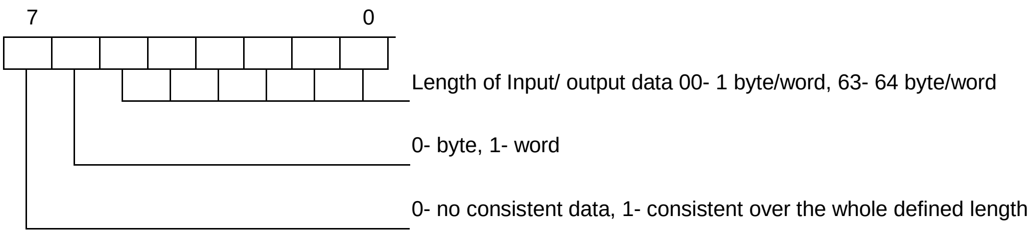


Structure of an Octet in the Configuring Message :



Structure of the special ID format





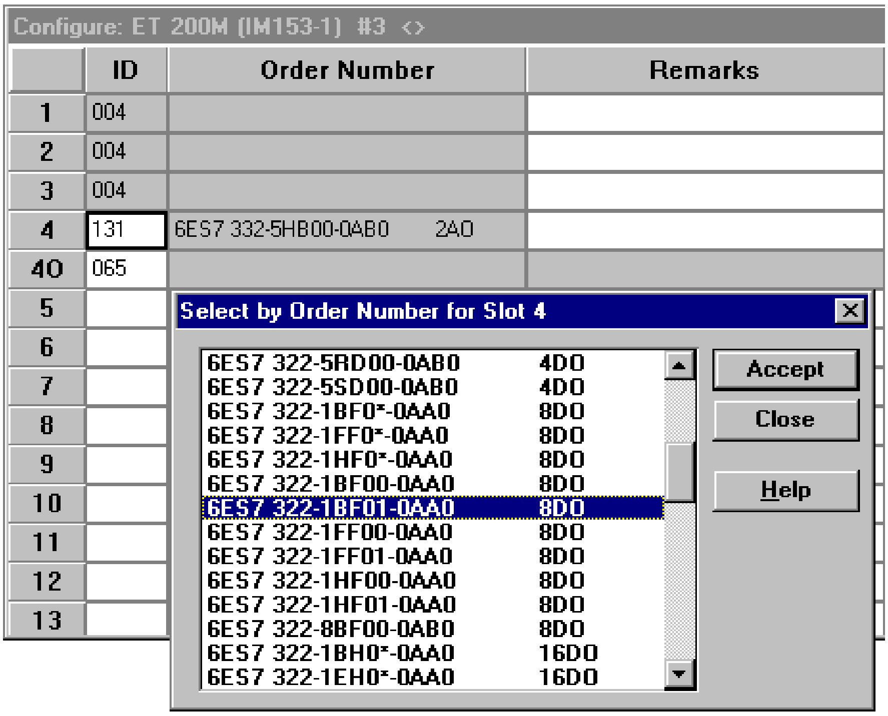
The special configuration format exists always out of at least 2 bytes.

In the GSD(E) file, the corresponding definitions look like this:

Example:

```
Module="6ES7 322-1BF01-0AA0        8DO" 0x83,0x00,0x00,0x2F,0xC8
Ext_Module_Prm_Data_Len=21         ; the module needs 21 data
Ext_User_Prm_Data_Const(0)= \      ; the 21 data is specified as constant values
0x15,0x5F,0x04,0x00,0x10,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,\
0x00,0x00,0x00,0x00,0x00,0x00
Ext_User_Prm_Data_Ref(2)=28           ; here, the text references for the individual
Ext_User_Prm_Data_Ref(6)=29           ; user parameters are specified
Ext_User_Prm_Data_Ref(7)=30
Ext_User_Prm_Data_Ref(8)=31
Ext_User_Prm_Data_Ref(9)=32
Ext_User_Prm_Data_Ref(10)=33
Ext_User_Prm_Data_Ref(11)=34
```


In the configuring tool COM PROFIBUS, the associated configuration looks like this:



When using the DPS2/DPSE software for the PROFIBUS ASIC SPC3, the relevant code location looks like this, for example:

```
if(DPS2_GET_IND_NEW_CFG_DATA())
   {                                        /*=== received new configuration ===*/
   UBYTE DPS2_PTR_ATTR * cfg_ptr;
   UBYTE i, config_data_len, cfg_result, result;

   cfg_result = DPS2_CFG_FINISHED;
   result = DPS_CFG_OK;
   do
      {  /* check configuration data until no conflict behavior m*/
      cfg_ptr = DPS2_GET_CFG_BUF_PTR();   /* set pointer to config_data_block  */
config_data_len = DPS2_GET_CFG_LEN();
```

/* User evaluation*/
/* Checking the received configuration */
/* Possibilities of the result of the check */

```
user_io_data_len_ptr = dps2_calculate_inp_outp_len      /* enter buffer organization with */
(cfg_ptr,(UWORD)config_data_len);                       /* the current lengths in SPC3  */
```



```
if (( user_io_data_len_ptr -> inp_data_len <= MAX_INP_DATA_LEN ) && (user_io_data_len_ptr ->
        outp_data_len <= MAX_OUTP_DATA_LEN ))
result = DPS_CFG_UPDATE;
            result = DPS_CFG_FAULT ;    */
{
        result = DPS.CFG_UPDATE;
        } else
{
result = DPS_CFG_FAULT;
}
if (result == DPS_CFG_UPDATE)
                {
if (user_io_data_len_ptr != (DPS2_IO_DATA_LEN *)0)
                    {
DPS2_SET_IO_DATA_LEN(user_io_data_len_ptr);
                    }
else
result = DPS_CFG_FAULT;
                }
             }
switch (result)
             {
case DPS_CFG_OK: cfg_result = DPS2_SET_CFG_DATA_OK();
break;
case DPS_CFG_FAULT: cfg_result = DPS2_SET_CFG_DATA_NOT_OK();
break;
case DPS_CFG_UPDATE: cfg_result = DPS2_SET_CFG_DATA_UPDATE();
break;
             }
          }

} while(cfg_result == DPS2_CFG_CONFLICT);
   }
```

After the configuration message, the master once more polls the diagnosis in the slave. If no errors were detected during configuring and parameter assignment, the field device is in data exchange.

4 Sample Files for GSD(E) File Entries

General Example

```
;==============================================================
; GSD(E) file for product (device name), company (manufacturer)
; Version : (version of the GSD file) - (contact person, phone)
; (General product information; for example,  Sync_mode_supp )
;==============================================================
;General Parameters
;1st line has to start with #Profibus_DP if it is            (M)
;a DP device

#Profibus_DP
;Manufacturer's name, 32 characters max.                     (M)
Vendor_Name = "Manufacturer"
;Product name; 32 characters max.                            (M)
Model_Name = "Product name"
;Version 32 characters max.                                  (M)
Revision = "Version 1"
;Ident number of product unsigned 16                         (M)
Ident_Number = 0x8023
;Protocol ID   0=DP device                                   (M)
Protocol_Ident = 0
;Device type   0=Slave, 1=Master(Class1)                     (M)
Station_Type = 0
;DP device type 0=only DP, 1=DP and FMS                      (D)
FMS_supp = 0
;Hardware release  32 characters max.                        (M)
Hardware_Release = "A01"
;Software release  32 characters max.                        (M)
Software_Release = "Z01"
;Here, all supported baudrates of a
;DP device have to be listed
;Product supports 9.6kBaud                                   (G)
9.6_supp = 1
19.2_supp = 1
93.75_supp = 1
187.5_supp = 1
500_supp = 1
1.5M_supp = 1
3M_supp = 1
6M_supp = 1
12M_supp = 1
MaxTsdr_9.6 = 60
MaxTsdr_19.2 = 60
MaxTsdr_93.75 = 60
MaxTsdr_187.5 = 60
MaxTsdr_500 = 100
MaxTsdr_1.5M = 150
MaxTsdr_3M = 250
MaxTsdr_6M = 450
MaxTsdr_12M = 800
;Redundant transmission engineering  0=No, 1=yes             (D)
```

Redundancy = 0
;Signal level (CNTR-P) Pin 4 of the 9-pole SUB-D (D)
;0-not available, 1-RS485, 2-TTL
Repeater_Ctrl_Sig = 2
;Meaning of the 24V pins of the 9-pole SUB-D (D)
;0-not available, 1-Input, 2-Output
24V_Pins = 0
;
;--Slave-specific values-----
;
;Freeze Mode is supported 0=No, 1=Yes (D)
Freeze_Mode_supp = 0
;Sync Mode is supported 0=No, 1=Yes (D)
Sync_Mode_supp = 1
;Autom. baudrate search is supported 0=No, 1=Yes (D)
Auto_Baud_supp = 1
;The product can be addressed via the bus
;0=No, 1=Yes (D)
Set_Slave_Add_supp = 0
;Expanded parameterization values (user data length) (D)
;unsigned 8
User_Prm_Data_Len = 0x05
;Values to be preassigned (O)
User_Prm_Data = 0x01,0x02,0x03,0x04,0x05
;Minimum refresh time of a call message (M)
;to the slave unsigned 16 (Basis 100us)
Min_Slave_Intervall = 0x0016
;alternatively, the value can be written in decimals
;**Min_Slave_Intervall** = 22

Example 1: Modular Station

;Product description 0=compact device, 1=modular (D)
Modular_Station = 1
;Max. number of modules that are sent to the slave (M)
;as configuration unsigned 8; in Example 1,
;12 modules maximum can be selected from the available modules
Max_Module = 0x0C
;alternatively, the value can be written in decimals = 12
;Max. number of inputs in bytes unsigned 8 (M)
Max_Input_Len = 0x10
;alternatively, the value can be written in decimals = 16
;Max. number of outputs in bytes unsigned 8 (M)
Max_Output_Len = 0x08
;alternatively, the value can be written in decimals = 08
;max. sum of input and output bytes unsigned 16 (M)
Max_Data_Len = 0x0018
;alternatively, the value can be written in decimals = 24
;Device-related diagnosis in plain text (O)
;Bit location in the device-related diagnosis unsigned 16
;Plain text display 32 characters maximum
Unit_Diag_Bit(0000) = "Slow_Mode active"
Unit_Diag_Bit(0001) = "Wrong_Config_Length"
Unit_Diag_Bit(0002) = "Modul_fault"
Unit_Diag_Bit(0006) = "Power failure"


```
Unit_Diag_Bit(0009) = "Short circuit to Plus"
;Module description; each module is inserted between Module - EndModule
;32 characters are available for plain text representation
;The ID is an  octet string
;Module for empty slot
Module= "Leerplatz " 0x00                <<empty slot>>
EndModule
;Input modules byte-organized
Module = "1 Byte DE " 0x10               <<DI>>
EndModule
Module = "2 Byte DE " 0x11
Channel_Diag(16) = "Uebertemperatur oder Ueberlast" <<overtemp. or overload>>
Channel_Diag(17) = "Kabelbruch oder Kurzschluss"     <<broken cable or short circuit>>
EndModule
;Output modules byte-organized
Module = "1 Byte DA " 0x20
EndModule
;Input/output modules byte-organized
Module = "1 Byte DE/DA " 0x30            <<DI/DO>>
EndModule
Module = "2 Byte DE/DA " 0x31
EndModule
Module = "2 Byte DE/DA " 0x11,0x21
EndModule
;End GSD file Example 1
```

Example 2: Compact station described in the modular mode (3 possible configurations)

```
.
.
;Product description  0=compact device, 1=modular          (D)
Modular_Station = 1
;Max. number of modules that are sent to the slave         (M)
;as configuration  unsigned 8; in Example 2,
;1 module maximum can be selected
;from the modules that are available
Max_Module = 01
;Maximum number of inputs  unsigned 8                      (M)
Max_Input_Len = 20
;Max. number of outputs  unsigned 8                        (M)
Max_Output_Len = 20
;Max.  sum of the input and output data  unsigned 16       (M)
Max_Data_Len = 40
;Module description; each module is inserted between Module - EndModule
;32 characters are available for plain text display
;The ID is an octet string
;Module Selection 1
Module= "Auswahl 1 20Byte E/A PPO Typ1" 0xF3,0xF3,0xF1  <<selection;  I/O>>
EndModule
;Module Selection 2
Module= "Auswahl 2 16Byte E/A PPO Typ2" 0xF3,0xF3
EndModule
;Module Selection 3
Module= "Auswahl 3 2Byte E, 7Byte A" 0x11,0x26              <<selection;  I, O>>
EndModule
```


Example 3: Compact Station

```
;Product Description                (D)
Modular_Station = 0                 ;0=compact device
Unit_Diag_Area = 0-5
Value(0) = "Fehlerfrei"                          <<faultless>>
Value(1) = "Fehler auf Eingang 0 - 23"           <<error on input …>>
Value(2) = "Fehler auf Ausgang 0 - 15"           <<error on output …>>
Value(3) = "24V ausgefallen"                     <<24V failed>>
Unit_Diag_Area_End
;Module description; each module is inserted between Module - EndModule ;
;32 characters are available for plain text display
;The ID is an octet string
;Modules for compact station
Module= "Kompaktgeraet 16E/16A " 0x11,0x21  <<compact device  16I/16O>>
EndModule
```

Example 4: Compact station with several modules, to be able to assign a text to each module

```
;Product description  0=compact device, 1=modular        (D)
Modular_Station = 0
;Module description; each module is inserted between Module - EndModule
;32 characters are available for clear text display
;The ID is an octet string
;Modules for compact station
;Output module byte-organized
Module = "1 Byte DA " 0x20
EndModule
;Input/output module byte-organized
Module = "1 Byte DE/DA " 0x30                    <<DI/DO>>
EndModule
Module = "2 Byte DE/DA " 0x31
EndModule
```

Example 5: GSD(E) file of the modular station ET 200 X by Siemens

Based on the GSD(E)file below, an explanation is provided as to how the parameter assignment message can be structured symbolically. The relevant text passages are in bold print, indented, and marked with a reference consisting of a number of a letter. The references are used only as an explanation in this example, and are not included in the original GSD(E).
A modular module system is to be configured (refer to Figure 2-2). First, the first 3 user parameter bytes are set (refer to Reference A, with the pre-assignment 0x40,0x20,0x00).

The message structure of the user parameters is as follows:

Octet 1 ... Octet 7	**Octet 8**	**Octet 9**	**Octet 10**
x....x	**0x40**	**0x20**	**0x00**

The 2nd byte for setting the diagnosis can be changed. For this, please refer to Reference B (GSD file following). Reference B permits setting the diagnostic alarm. For this, Bit 5 is evaluated with the Default Value 1 and the Value Range 0..1 (Bit(5) 1 0-1), and reference is made to the text according to Reference C. Here, the user can select whether it wants to lock or unlock the diagnostic texts. If diagnostic processing is locked, the message structure looks like this:

Octet 1 ... Octet 7	**Octet 8**	**Octet 9**	**Octet 10**
x....x	**0x40**	**0x00**	**0x00**

As the next step, parameters are assigned to the first module. The analog input module with the designation "6ES7 144-1FB30-0XB0 2AE 10V" is selected.

Starting with Text Reference 1, the user part of the parameter assignment message now looks like this:

Standard	Presetting			Module 6ES7 144-1FB30-0XB0 2AE 10 V								
Octet 1..7	8	9	10	11	12	13	14	15	16	**17**	18	19
x...x	0x40	0x00	0x00	0x09	0x5F	0x05	0x01	0x00	0x00	**0x0A**	0x19	0x19

The presetting is to be modified (Octet 6 = 0x0A). For that reason, refer to Reference 2). In Reference 2), Bits 0..3 are relevant. The default value is 10, and the value ranges from 0..10. The associated symbolic text reference is 3). In Text Reference 3), the user can select the symbol 50 Hz or 60 Hz. Here, the value 60 Hz is selected.

The user part of the parameter assignment message now looks like this:

Standard	Presetting			Module 6ES7 144-1FB30-0XB0 2AE 10 V								
Octet 1..7	8	9	10	11	12	13	14	15	16	**17**	18	19
x...x	0x40	0x00	0x00	0x09	0x5F	0x05	0x01	0x00	0x00	**0x05**	0x19	0x19

Additional modules can be configured in in the same manner.

GSD(E) file for the above example.

```
;==========================================================
; GSD-File  for ET 200X 8DI-2  DP              SIEMENS AG
; MLFB    : 6ES7 141-1BF01-0XB0                <<order number>>
;
; Version : 18.05.98 SX
; File    : SI__803D.GSG
;===========================================================
#Profibus_DP
; <Prm-Text-Def-List>
Referenz C)     PrmText=1              ; here, the user selects lock or unlock
                Text(0)="sperren"      <<lock>>
                Text(1)="freigeben"    <<unlock>>
                EndPrmText
PrmText=2
Text(0)="SIMATIC S7"
Text(1)="SIMATIC S5"
EndPrmText
```


PrmText=3
Text(0)="sperren" <<lock>>
Text(1)="freigeben" <<unlock>>
EndPrmText
Referenz 3) PrmText=4
Text(5)="60 Hz" ;here the user selects 60 Hz
Text(10)="50 Hz"
EndPrmText
PrmText=5
Text(0)="deaktiviert"
Text(6425)="Spannung +/- 10 V"
EndPrmText
PrmText=6
Text(8995)="Strom (4-DMU) 4 .. 20 mA" <<current (4wire transducer)>>
Text(9252)="Strom (4-DMU) +/- 20 mA"
EndPrmText
PrmText=7
Text(0)="deaktiviert" <<deactivated>>
Text(13107)="Strom (2-DMU) 4 .. 20 mA"
EndPrmText
PrmText=8
Text(0)="deaktiviert"
Text(33410)="RTD-4L Pt 100 Standard"
EndPrmText
PrmText=9
Text(0)="deaktiviert"
Text(6425)="Spannung +/- 10 V" <<voltage>>
EndPrmText
PrmText=10
Text(8995)="Strom 4 .. 20 mA" <<current>>
Text(9252)="Strom +/- 20 mA"
EndPrmText
; <Ext-User-Prm-Data-Def-List>
Referenz B) ExtUserPrmData=1 "Diagnosealarm" <<diagnostic alarm>>
Bit(5) 1 0-1
Prm_Text_Ref=1 → additional reference according to <<after?>> **C)**
EndExtUserPrmData
ExtUserPrmData=2 "[SlotNumber]"
Unsigned8 1 1-11
EndExtUserPrmData
ExtUserPrmData=3 "Formatdarstellung" <<format representation>>
Bit(0) 0 0-1
Prm_Text_Ref=2
EndExtUserPrmData
Referenz 2) ExtUserPrmData=4 "Stoerfrequenzunterdrueckung E0/1" <<interf. freq. suppr.>>
BitArea(0-3) 10 005-010
Prm_Text_Ref=4 → additional reference according to 3)
EndExtUserPrmData
ExtUserPrmData=5 "[SlotNumber]"
Unsigned8 1 1-11
EndExtUserPrmData
ExtUserPrmData=6 "Formatdarstellung"
Bit(0) 0 0-1
Prm_Text_Ref=2
EndExtUserPrmData
ExtUserPrmData=7 "Stoerfrequenzunterdrueckung E0/1" <<interference frequency suppr.>>


```
BitArea(0-3) 10 005-010
Prm_Text_Ref=4
EndExtUserPrmData
ExtUserPrmData=8 "Messart/-bereich E 0/1"          <<type of measurement/meas. range>>
Unsigned16 9252 8995-9252
Prm_Text_Ref=6
EndExtUserPrmData
ExtUserPrmData=9 "[SlotNumber]"
Unsigned8 1 1-11
EndExtUserPrmData
ExtUserPrmData=10 "Formatdarstellung"          <<format representation>>
Bit(0) 0 0-1
Prm_Text_Ref=2
EndExtUserPrmData
ExtUserPrmData=11 "Stoerfrequenzunterdrueckung E0/1"  <<interference freq. suppr.>>
BitArea(0-3) 10 005-010
Prm_Text_Ref=4
EndExtUserPrmData
ExtUserPrmData=12 "[SlotNumber]"
Unsigned8 1 1-11
EndExtUserPrmData
ExtUserPrmData=13 "Formatdarstellung"
Bit(0) 0 0-1
Prm_Text_Ref=2
EndExtUserPrmData
ExtUserPrmData=14 "Stoerfrequenzunterdrueckung E0/1"
BitArea(0-3) 10 005-010
Prm_Text_Ref=4
EndExtUserPrmData
ExtUserPrmData=15 "[SlotNumber]"
Unsigned8 1 1-11
EndExtUserPrmData
ExtUserPrmData=16 "Formatdarstellung"
Bit(0) 0 0-1
Prm_Text_Ref=2
EndExtUserPrmData
ExtUserPrmData=17 "[SlotNumber]"
Unsigned8 1 1-11
EndExtUserPrmData
ExtUserPrmData=18 "Formatdarstellung"
Bit(0) 0 0-1
Prm_Text_Ref=2
EndExtUserPrmData
ExtUserPrmData=19 "Ausgabeart/-bereich A 0/1"          <<output type/output range>>
Unsigned16 9252 8995-9252
Prm_Text_Ref=10
EndExtUserPrmData
; <Unit Definition List>
GSD_Revision=1
Vendor_Name="SIEMENS"
Model_Name="ET 200X 8DI-2  DP"
Revision="V2.0a"
Ident_Number=0x803D
Protocol_Ident=0
Station_Type=0
Hardware_Release="A1.0"
```



```
Software_Release="Z1.0"
9.6_supp=1
19.2_supp=1
93.75_supp=1
187.5_supp=1
500_supp=1
1.5M_supp=1
3M_supp=1
6M_supp=1
12M_supp=1
MaxTsdr_9.6=60
MaxTsdr_19.2=60
MaxTsdr_93.75=60
MaxTsdr_187.5=60
MaxTsdr_500=100
MaxTsdr_1.5M=150
MaxTsdr_3M=250
MaxTsdr_6M=450
MaxTsdr_12M=800
Implementation_Type="SPC3"
Bitmap_Device="ET200X1"
; Slave Specification:
OrderNumber="6ES7 141-1BF01-0XB0"
Periphery="ET 200"
MaxResponseDelay=0
Freeze_Mode_supp=1
Sync_Mode_supp=1
Auto_Baud_supp=1
Fail_Safe=1
Min_Slave_Intervall=3
Max_Diag_Data_Len=32
Modul_Offset=1
Slave_Family=3@TdF@ET200X
Modular_Station=1
Max_Module=11
Max_Input_Len=104
Max_Output_Len=104
Max_Data_Len=208
; UserPrmData: Length and Preset:
User_Prm_Data_Len=3
User_Prm_Data=0x40,0x20,0x00
Max_User_Prm_Data_Len=121
```

Referenz A) Ext_User_Prm_Data_Const(0)=0x40,0x20,0x00
Ext_User_Prm_Data_Ref(1)=1 → additional reference according to B)

```
; Unit Diagnostics:
Unit_Diag_Bit(0024)="Baugruppenstoerung"                 <<module fault>>
Unit_Diag_Bit(0026)="Externer Fehler (Drahtbruch)"       <<external error (wire break)>>
Unit_Diag_Bit(0028)="Externe Hilfsspannung fehlt"        <<no external auxiliary voltage>>
Unit_Diag_Bit(0031)="Parametrierfehler Baugruppe"        <<parameterization error module>>
; <Module Definition List>
FixPresetModules=1
Module="Config for Slot1" 0x04,0x00,0x00,0xAD,0xC4
Preset=1
EndModule
Module="Config for Slot2" 0x04,0x00,0x00,0x8B,0x40
```


Preset=1
EndModule
Module="Config for Slot3" 0x04,0x00,0x00,0x8F,0xC0
Preset=1
EndModule
Module="Config for Slot4" 0x43,0x00,0x00,0x9F,0xC9
Preset=1
EndModule
Module="6ES7 141-1BD30-0XA0 4DE" 0x43,0x00,0x00,0x8F,0xC9
EndModule
Module="6ES7 141-1BF30-0XA0 8DE" 0x43,0x00,0x00,0x9F,0xC9
EndModule
Module="6ES7 142-1BD30-0XA0 4DA 0,5A" 0x83,0x00,0x00,0x8F,0xC8
EndModule
Module="6ES7 142-1BD40-0XA0 4DA 2A" 0x83,0x00,0x00,0x8F,0xC8
EndModule
Referenz 1) Module="6ES7 144-1FB30-0XB0 2AE 10V" 0x43,0x41,0x00,0x15,0xC3
Ext_Module_Prm_Data_Len=9
Ext_User_Prm_Data_Const(0)=0x09,0x5F,0x05,0x01,0x00,0x00,0x0A,0x19,0x19
Ext_User_Prm_Data_Ref(2)=2
Ext_User_Prm_Data_Ref(5)=3
Ext_User_Prm_Data_Ref(6)=4
EndModule → additional reference according to 2)
Module="6ES7 144-1GB30-0XB0 2AE 20mA" 0x43,0x41,0x00,0x15,0xC3
Ext_Module_Prm_Data_Len=9
Ext_User_Prm_Data_Const(0)=0x09,0x5F,0x05,0x01,0x00,0x00,0x0A,0x24,0x24
Ext_User_Prm_Data_Ref(2)=5
Ext_User_Prm_Data_Ref(5)=6
Ext_User_Prm_Data_Ref(6)=7
Ext_User_Prm_Data_Ref(7)=8
EndModule
Module="6ES7 144-1GB40-0XB0 2AE 4-20mA" 0x43,0x41,0x00,0x15,0xC3
Ext_Module_Prm_Data_Len=9
Ext_User_Prm_Data_Const(0)=0x09,0x5F,0x05,0x01,0x00,0x00,0x0A,0x33,0x33
Ext_User_Prm_Data_Ref(2)=9
Ext_User_Prm_Data_Ref(5)=10
Ext_User_Prm_Data_Ref(6)=11
EndModule
Module="6ES7 144-1JB30-0XB0 2AE Pt100" 0x43,0x41,0x00,0x15,0xC3
Ext_Module_Prm_Data_Len=9
Ext_User_Prm_Data_Const(0)=0x09,0x5F,0x05,0x01,0x00,0x00,0x0A,0x82,0x82
Ext_User_Prm_Data_Ref(2)=12
Ext_User_Prm_Data_Ref(5)=13
Ext_User_Prm_Data_Ref(6)=14
EndModule
Module="6ES7 145-1FB30-0XB0 2AA 10V" 0x83,0x41,0x00,0x25,0xD8
Ext_Module_Prm_Data_Len=9
Ext_User_Prm_Data_Const(0)=0x09,0x5F,0x05,0x01,0x00,0x00,0x00,0x19,0x19
Ext_User_Prm_Data_Ref(2)=15
Ext_User_Prm_Data_Ref(5)=16
EndModule
Module="6ES7 145-1GB30-0XB0 2AA 20mA" 0x83,0x41,0x00,0x25,0xD8
Ext_Module_Prm_Data_Len=9
Ext_User_Prm_Data_Const(0)=0x09,0x5F,0x05,0x01,0x00,0x00,0x00,0x24,0x24
Ext_User_Prm_Data_Ref(2)=17


```
Ext_User_Prm_Data_Ref(5)=18
Ext_User_Prm_Data_Ref(7)=19
EndModule
Module="6GK7 142-2AH00-0XA0 CP 142-2" 0xC2,0x0F,0x0F,0xBC,0xC3
EndModule
Module="3RK1 300-**S00-0AA* 4DX" 0xC2,0x00,0x00,0xCF,0xC9
EndModule
Module="3RK1 300-**S00-1AA* 4DX" 0xC2,0x00,0x00,0xDF,0xC9
EndModule
Module="3RK1 300-0*S10-0AA* 4DX" 0xC2,0x00,0x00,0xEF,0xC9
EndModule
Module="3RK1 300-0*S10-1AA* 4DX" 0xC2,0x00,0x00,0xFF,0xC9
EndModule
```