Overview about Services and Configuration of Profibus DP

The following information describes the operation of ProfiBus DP. For more detailed information on ProfiBus, you may refer to the ProfiBus Trade Organization at the PTO website www.profibus.com.

lProfiBus was created in 1989 by the German government in cooperation with several manufacturers of automation equipment. It is a messaging format specifically designed for high-speed serial I/O in factory and building automation applications. It is an open standard (IEC 61158) and is recognized as the fastest FieldBus in operation today. It is based on RS485 and the European EN50170 Electrical Specification. The DP suffix refers to “Decentralized Periphery”, which is used to describe distributed I/O devices connected via a fast serial data link with a central controller. To contrast, a programmable logic controller (PLC) normally has its input/output channels arranged centrally. By introducing a network bus between the main controller (master) and its I/O channels (slaves), we have *decentralized* the I/O.

ProfiBus is based on universal international standards and oriented to the OSI (Open System Interconnection) reference model per international standard ISO 7498. In this model, every layer handles precisely defined tasks. Layer 1 of this model is the physical layer and defines the physical transmission characteristics. Layer 2 is the data link layer and defines the bus access protocol. Layer 7 is the application layer and defines the application functions. ProfiBus DP uses only layers 1 & 2 of this model, plus the user interface. Layers 3 to 7 are not used.

A ProfiBus system uses a bus *master* to poll *slave* devices distributed in multi-drop fashion on an RS485 serial bus. A ProfiBus slave is any peripheral device (I/O transducer, valve, network drive, or other measuring device) which processes information and sends its output to the master. The slave forms a “passive station” on the network since it does not have bus access rights, and can only acknowledge received messages, or send response messages to the master upon request. It is important to note that all ProfiBus slaves have the same priority, and all network communication originates from the master.

A ProfiBus master forms an “active station” on the network. ProfiBus DP defines two classes of masters. A class 1 master handles the normal communication or exchange of data with the slaves assigned to it. A class 2 master is a special device primarily used for commissioning slaves and for diagnostic purposes. Some masters may support both class 1 and class 2 functionality. Master-to-master communication is normally not permitted in Profibus, except in order to grant bus access rights to another master via the exchange of a token. Note
that the exchange of bus access rights via this “token ring” only applies between masters on the bus.

A class 1 master device is normally a central programmable controller (PLC), or a PC running special software. The class 1 master sets the baud rate and the slave’s auto-detect this rate. The class 1 master handles the data exchange with the slaves assigned to it, and acts as the main controller for the exchange of I/O information with its distributed slaves, cyclically retrieving user I/O data according to a defined message cycle. A master can communicate actively with its assigned slaves, but only passively (upon request) with another class 2 master device.

The class 2 master is usually a configuration device, perhaps a laptop or programming console, and is provided for commissioning, maintenance, or diagnostic purposes. It acts like a “supervisory” master in that it can actively communicate with class 1 masters and their slaves, in addition to its own slaves, but usually only for the purpose of configuration, problem diagnosis, and data/parameter exchange. That is, class 2 masters may only briefly take over control of a slave. All exchanges between a class 2 master and class 1 master originate with the class 2 master.

ProfiBus DP normally operates using a cyclic transfer of data between master(s) and slave(s) on an RS485 network. That is, an assigned master periodically requests (polls) each node (slave) on the network. All data communication exchanges between a master and slave originate from the master device. Each slave device is assigned to one master and only that master may write output data to that slave. Other masters may read information from any slave, but can only write output data to their own assigned slaves.

Masters can address individual slaves, a defined group of slaves (multi-cast), or can broadcast a telegram to all connected slaves. Slaves will return a response to all telegrams addressed to them individually, but do not respond to broadcast or multi-cast telegrams from a master device. ProfiBus sends Broadcast and Multi-Cast messages as global control telegrams using address 127 and an optional group number for a targeted group of slaves.

Because ProfiBus uses a cyclic (periodic) polling mechanism between masters and slaves, it is also *deterministic*. That is, the behavior of a ProfiBus system can be reliably predicted over time. In fact, ProfiBus was designed to guarantee a deterministic response. To contrast, CAN and Ethernet are event-driven bus systems and consequently form non-deterministic systems.

The length (and timing) of the I/O data to be transferred from a single slave to a master is predefined in the slave's device data base or *GSD file*. The GSD files of each device connected via the network (slaves and class 1 masters only) are compiled into a master parameter record which contains parameterization and configuration data, an address allocation list, and the bus parameters for all connected stations. A master uses this information to set up communication with each slave during startup.

After a master receives its master parameter record, it is ready to begin exchanging data with its slaves. During startup, after a system reset, or upon return to power, a master will attempt to re-establish contact with all the slaves assigned to it before assuming the cyclic exchange of I/O data. Each slave must already have a unique valid address from 0-125 in order to communicate with the master.

ProfiBus DP most often uses a single class 1 master device (*mono-master*), cyclically polling many distributed slaves.
Although, mono-master operation is generally recommended, it is not mandatory. That is, a ProfiBus system may have more than one class 1 master, but master to master communication is not permitted, except for the granting of bus access rights via token exchange.

To illustrate the idea of communication between masters in a ProfiBus DP system, a class 1 master cyclically exchanges data with all of the slaves assigned to it, one at a time, according to its list of assigned slaves taken from the master record. After completition of the cyclical exchange with its Slaves, the class 1 master may use the bandiwith by realising asynchronous data exchanges with the Slaves using both confirmed (read inputs, read outputs) and/or not confirmed services (freeze, sync).

Key Concepts

ProfiBus DP –

- International Standard IEC 61158
- Open standard based on EN 50170.
- Fastest Fieldbus standard to date with data rates up to 12MB.
- Plug & play operation.
- Up to 244 bytes of input/output data per message.
- Up to 126 stations may connect to the bus.

Class 1 Master –

- Central controller that exchanges I/O data with connected slaves.
- Determines the baud rate (slaves auto-detect this rate).
- Manages the token transfer between masters. Detects another master during the gap time.

Class 2 Master –

- Diagnostic, configuration, or startup tool.
- Can only control one slave at a time.
- Does not have write-access to the slave.
- Does not have a GSD file.

Slave -

- A passive station which can only respond per a master request and acknowledge messages. A slave has no bus control rights.
- The GSD file defines the slave for the master.

.

Before we examine slave operation in detail, we need to get a little background information on a device's GSD file.

GSD FILES

The GSD file is an electronic device data sheet or device data base file that identifies the ProfiBus device. All ProfiBus devices (class 1 masters and slaves) have their own GSD files. The GSD file is the fundamental building block for the master parameter record. Use of the GSD file by a ProfiBus configuration tool permits plug & play interoperability among different devices from different manufacturers. This file does not reside within the device itself, but usually on a separate disk/drive. It is an ASCII text file that contains device-specific data, such as, vendor identification information, supported baud rates, supported message length, number of input/output data, meaning of diagnostic messages, timing information, plus options and features supported, data formats, and available I/O signals. For modular ProfiBus systems, a GSD file may contain several configurations (one for each I/O module), one of which will be found valid during startup.

A GSD file is named by combining a vendor name identifier with the device's ident_ number. For example, "ACRO06F3.GSD" is used for the Acromag 981PB-1012 device. The suffix ".GSD" denotes a language independent GSD file. A ".GSE" file would specify an English file, ".GSF" for French, ".GSS" for Spanish, ".GSI" for Italian, and ".GSG" for German.

The GSD file begins with the specifier "#Profibus_DP". In the body of the file, the parameters are specified as parameters of a keyword (as in "keyword = parameter", see below). Comment lines begin with a semicolon. Case is not significant and the sequence of parameters is not important. Lines are limited to 80 characters, but may be continued by placing a backslash character "\" at the end of the line to be continued. A GSD file is divided into sections as follows:

GSD General Specifications	Keyword	Keyword
	Vendor_name	187.5_supp
This section contains information on vendor and device names, hardware and software revisions, ident_number, supported baud rates, reaction time intervals at supported baud rates for monitoring times, and optional signal support at the bus connector.	Model_Name	500_supp
	Revision	1.5M_supp
	Ident_Number	MaxTsdr_9.6
	Protocol_Ident	MaxTsdr_19.2
	Station_type	MaxTsdr_93.75
	FMS_Support	MaxTsdr_187.5
	Hardware_Release	MaxTsdr_500
	Software_Release	MaxTsdr_1.5M
	9.6_supp	Redundancy
	19.2_supp	Repeater_Ctrl_Sig
	93.75_supp	24V_Pins

GSD Slave Specifications	Keyword	Keyword
This section contains all	Freeze_Mode_supp	Max_Input_Len
slave-related specifications, such as the number and type of I/O channels, specification of diagnostic text, auto-baud support, alternate mode support and information on available modules with modular devices.	Sync_Mode_supp	Max_Output_Len
	Auto_Baud_supp	Max_Data_Len
	Set_Slave_Add_supp	Unit_Diag_Bit
	User_Prm_Data_Len	Diag_Text
	User_Prm_Data	Unit_Diag_Area
	Min_Slave_Intervall	Module
	Modular_Station	Channel_Diag
	Max_Module	

To give you an idea of what a GSD file might look like, consider the following partial GSD file taken from Acromag Model 983PB-2012.

```
#Profibus_DP
GSD_Revision          = 1                ; Version of the GSD file
Vendor_Name           = “Acromag, Inc.”  ; Vendor name
Model_Name            = “983PB-2012”     ; Product name
Revision              = “A”
Ident_Number          = 0x06F1           ; Ident Number
Protocol_Ident        = 0                ; ProfiBus DP Only (1-DP/FMS)
Station_Type          = 0                ; Type of device (Slave)
Hardware_Release      = “A”              ; Hardware version of the device
Software_Release      = “A”              ; Software version of the device
;
9.6_supp              = 1                ; 9600bps Supported
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               ; Maximum response time
MaxTsdr_19.2          = 60               ; at different baud rates.
MaxTsdr_93.75         = 60
MaxTsdr_187.5         = 60
MaxTsdr_500           = 100
MaxTsdr_1.5M          = 150
MaxTsdr_3M            = 250
MaxTsdr_6M            = 450
MaxTsdr_12M           = 800
;
Redundancy            = 0                ; Redundancy Not Supported
Repeater_Ctrl_Sig     = 2                ; Includes RTS Support w/TTL
Implementation_Type   = “SPC3”           ; Uses Siemens SPC3 ASIC
24V_Pins              = 0                ; Does Not Include 24V
Fail_Safe             = 1                ; Supports Fail-Safe Mode
Freeze_Mode_supp      = 1                ; Supports FREEZE
Sync_Mode_supp        = 1                ; Supports SYNC
Auto_Baud_supp        = 1                ; Includes Auto Baud Detection
Set_Slave_Add_supp    = 1                ; Addr can be set via ProfiBus
User_Prm_Data_Len     = 3
User_Prm_Data         = 0x00,0x00,0x00   ; Module Specific Parameters
; 00H = Set outputs to 0
; 01H = Maintain Last Output Values
; 02H = Set output to user-defined values in bytes 1 and 2
; Byte 1 is the lower byte of user-defined output data which is outputs 0 to 7
; Byte 2 is the upper byte of user-defined output data which is outputs 8 to 11
;
```

```
Slave_Family            = 3
Min_Slave_Interval      = 1             ; Min_Slave_Interval is 100us
Modular_Station         = 0             ; 0-compact, 1-modular
Max_Diag_Data_Len       = 6             ; No User Diagnostics are Sent
; I/O Byte
Module                  = “12 CH DIG I/O:xxxx1198 76543210” 0x11,0x21
EndModule
;
```

ProfiBus DP uses two types of transmission services in sending message telegrams that are defined in Layer 2 (The Data Link Layer) of the ISO/OSI model and summarized below:

SRD (Send and Request Data with acknowledge)

With SRD, data is sent and received in one telegram cycle. That is, the master sends output data to the slave and receives input data from the slave in its response (if applicable) within a specified period of time. The important thing to remember about this service, is that a master may send output data to a slave and request input data from the slave in its response, all in a single telegram cycle. This is the transmission service most often used in ProfiBus DP that makes the data exchange very efficient for mixed I/O devices.

SDN (Send Data with No acknowledge)

This service is used when a message must be sent simultaneously to a group of slaves (multi-cast), or all slaves (broadcast). Slaves do not respond or acknowledge broadcast or multi-cast messages.

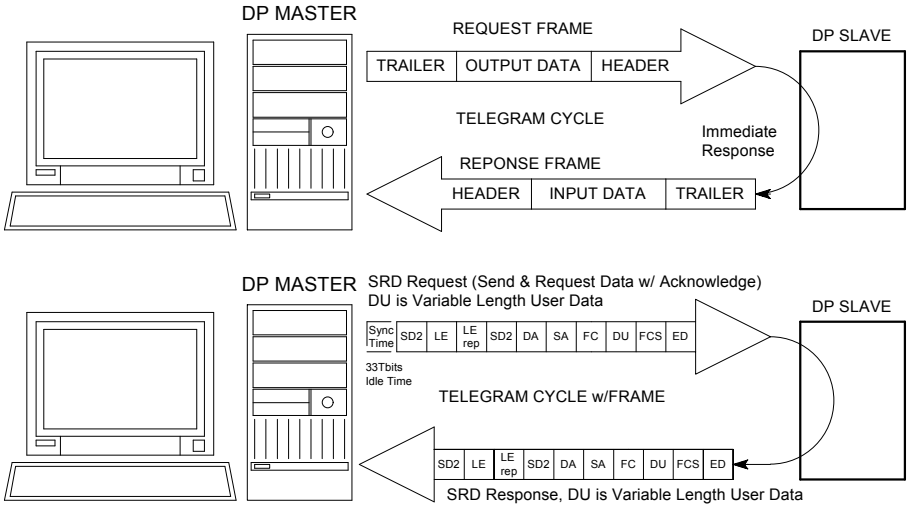
A ProfiBus telegram may contain up to 256 bytes--up to 244 bytes of data per node per message, plus 11 bytes of overhead. This overhead is referred to as the *Telegram Header*. All telegram headers are 11 bytes, except for Data_Exchange telegrams which have 9 bytes of header information (the DSAP and SSAP bytes are dropped). Note that 12 bytes is a lot of overhead for a single message and this makes ProfiBus less efficient for small amounts of data. However, since up to 244 bytes of data may occur per message, and since the output data is sent and the input data received in a single telegram cycle, this makes ProfiBus more efficient when large amounts of data must be transferred.

Note that an idle state of at least
33Tbits (sync-time in bit time) must be present before every request telegram to be sent, and all data is transferred without gaps between individual characters. All data exchanges between a Master and Slave are handled in the telegram header using Service Access Points (SAP's). ProfiBus DP uses SAP's 54 to 62, plus the default SAP (Data_Exchange).

ProfiBus DP Telegram Header Abbreviations and Frame Bytes

SD	1 byte	Start Delimiter (used to distinguish telegram format).
LE	1 byte	Net Data Length (DU) + DA + SA + FC + DSAP + SSAP.
LEr	1 byte	Length *repeated.*
DA	1 byte	Destination Address– Where this message goes to.
SA	1 byte	Source Address – Where this message came from. The address of the sending station.
FC	1 byte	Function Code (FC=Type/Priority of this message). Used to identify the type of telegram, such as request, acknowledgement, or response telegrams (FC=13 signals diagnostic data). See below.
DSAP	1 byte	Destination Service Access Point (COM port of receiver). The destination station uses this to determine which service is to be executed.
SSAP	1 byte	Source Service Access Point (COM port of sender).
DU	1 to 32b (or 1-244b)	Data Units/ Net Data from 1 to 244 bytes.
FCS	1 byte	Frame Checking Sequence (ASIC addition of the bytes within the specified length).
ED	1 byte	End Delimiter (always 16H).

The following picture depicts the telegram sequence between a class 1 master and DP slave. The paragraphs that follow describe each telegram frame byte in greater detail.



Data exchanges are handled in the telegram header using Service Access Points (SAP's). The SAP tells what data is to be transmitted or which function is to be performed. Only telegrams that include data fields use DSAP & SSAP bytes (i.e. SD2 & SD3 telegrams). Recall that SRD transmission combines an output message with an input response in a single telegram cycle. The telegram header contains an SSAP (Source Service Access Point) and/or DSAP (Destination Service Access Point) that indicates the service(s) to be executed. One exception is the cyclical Data_Exchange telegram which is performed with the default SAP (SSAP or DSAP is not provided in its header). Additionally, some telegrams may only provide a DSAP or SSAP, but not both.

The inclusion of a DSAP or SSAP entry in a request or response telegram is identified by setting the highest bit in the address byte of the DA (Destination Address) or SA (Source Address), respectively. Based on the detected SAP's, each station is able to recognize which data has been requested and which response data is to be supplied. ProfiBus DP uses SAP's 54 to 62 listed below, plus the default SAP.

SAP	SERVICE
Default SAP=0	Cyclical Data Exchange (Write_Read_Data)
SAP54	*Master-to-Master SAP (M-M Communication)*
SAP55	Change Station Address (Set_Slave_Add)
SAP56	Read Inputs (Rd_Inp)
SAP57	Read Outputs (Rd_Outp)
SAP58	Control Commands to a DP Slave (Global_Control)
SAP59	Read Configuration Data (Get_Cfg)
SAP60	Read Diagnostic Data (Slave_Diagnosis)
SAP61	Send Parameterization Data (Set_Prm)
SAP62	Check Configuration Data (Chk_Cfg)

SAP55 is optional and may be disabled if the slave does not provide non-volatile storage memory for the station address. Note that SAP's 56, 57, and 58 are not enabled until the DP slave assumes the Data_Exchange state. SAP's 59, 60, 61, and 62 are always enabled.

Note that the DSAP & SSAP entries in a request telegram are also included in the response telegram, where DA + SA + DSAP + SSAP in the response message corresponds to SA + DA + SSAP + DSAP in the request telegram (content position flips).

.

.

Global_Control Service – SAP 58

ProfiBus DP uses the Global Control function to send special commands addressed to a single slave, a specific group of slaves (multi-cast), or to all slaves at once (broadcast). ProfiBus sends broadcast and multi-cast messages as global control telegrams using address 127 (an optional group number is included for a select group with multi-cast).

Using SDN transmission (Send Data with No acknowledge), a class 1 master will use the Global_Control service to inform the slaves of his mode (Operating or Clear Mode), or to send commands such as sync, unsync, freeze, unfreeze, and clear data to a group of slaves, typically for synchronization purposes. Note that a slave will only accept this command from the same master that parameterized and configured it. There is no response returned to an SDN telegram.

Sync: The output states transferred in Data_Exchange are delivered and frozen. The output data which follows is held until the next Sync command or Unsync command.

Unsync: This control cancels the sync command.

Freeze: This causes the states of the inputs to be read and frozen until the next Freeze command or Unfreeze. Slaves must ensure that following a freeze command, the last frozen values of the inputs must be transferred in the next data exchange cycle.

Unfreeze: This control cancels the freeze command

Sync & Freeze are optional slave services and may not be supported by some slaves. A master uses a Freeze telegram to make a slave or group of slaves freeze their inputs in the current state. A Sync telegram causes the output data currently available to be transferred to the outputs, to be frozen. An Unsync & Unfreeze command will cancel this state.

Use of Freeze

In closed-loop control systems and for the synchronization of drives, etc., it is sometimes necessary to have a precise time image of the process inputs of a group. Freeze is used to accomplish this as follows:

The master sends a Freeze command to the selected group *(A global control telegram goes to the selected group at a precise time).* This causes all addressed slaves to freeze their inputs. During the next Data Exchange cycle, the slaves transfer the frozen inputs of the group to the master.

After the master receives this data, the master sends an Unfreeze command to the group and the bus system returns to the normal data exchange mode again, and all input changes are transferred during each data cycle.

Use of Sync/Unsync

For the time-controlled operation of output devices which belong to a group, sync and unsync are used as follows:

After the data is frozen with a Freeze command and has been processed by the master, the master reacts by sending a sync command to the slave group to obtain the outputs. During the next data cycle, the master supplies the slave group with the data to be output, then concludes this cycle with an unsync command in the following data exchange cycle. Unsync causes the slaves to transfer their outputs at a precise time.

Unlike CAN and Ethernet which are event-driven busses, ProfiBus was designed to guarantee a deterministic response.. The *determinism* of a system refers to the ability to precisely predict the behavior of the system over time.

ProfiBus uses a polling mechanism between master and slave. The time it takes a slave to respond to a message from the master is the *reaction time*. Even if a ProfiBus system receives many I/O signal changes at some point in time, there is no change in reaction time. Further, even if another master (class 2) is used to perform diagnostics on a slave device while it is communicating with its class 1 master, the reaction time for the system will remain the same. This is because the class 2 master used to perform diagnostics will not be allowed to use more time than the configured gap time within the bus cycle.

Because ProfiBus is deterministic, we can calculate a reliable system reaction time. But before we get into the details of calculating bus cycle times, we must define a few terms as follows:

Bit-Time: To help simplify timing calculations, it is convenient to normalize the time units with respect to the baud rate by using units of Bit-Time (Tbit). One bit-time is the time it takes to transmit one bit and is the reciprocal of the transmission rate (baud rate). For example:

1 Tbit (Bit Time) at 12MB = 1/12000000bps = 83ns/bit

Sync-Time (T_{SYN}): The synchronization time is the minimum time a station must remain in the idle state before it can accept another request. For ProfiBus DP, an idle state of 33Tbits (bit-time) must be present before every request telegram and this is called the sync-time.

Slave Reaction Time (T_{SDR}): The reaction time is the time it takes a slave to respond to a message. This time is often expressed as a minimum value (min T_{SDR}), or maximum value (max T_{SDR}). Min T_{SDR} is set within the parameterization telegram during startup. Max T_{SDR} varies with the transmission rate and is specified at the supported baud rates within the device GSD file. For ProfiBus DP, this value may range from a minimum of 11Tbits (min T_{SDR} default) to a maximum of 255Tbits.

Initiator Idle Time (T_{ID1}): After receiving the last character of a telegram, the initiator must wait this amount of time until it sends the next telegram. The idle time (Tid1) is the time between transmission of the last bit of a frame (no acknowledge) and the transmission of the first bit of the next frame. It is at least the sync time (T_{SYN}), plus some safety margin (Tsm), but is also calculated as the maximum of these three values: T_{SYN} + Tsm, min T_{SDR}, or T_{SDI} (station delay of telegram initiator). The addition of safety margin (Tsm) is very important at high baud rates.