DRIVECOM **PROFIDrive**

DriveServer Guide Profibus-DP

Developing a DriveServer-compatible OPC BusServer for Profibus-DP controller boards

Version 1.0, 26. February 2002

Issued by:

DRIVECOM Nutzergruppe e.V.
Postfach 1102
D-32817 Blomberg, Germany
Tel. : +49 (0) 52 35 - 34 18 64
Fax : +49 (0) 52 35 - 34 18 62
Internet : http://www.DRIVECOM.org

PROFIBUS Nutzerorganisation e.V.
Haid-und-Neu-Str. 7
D-76131 Karlsruhe, Germany
Tel. : +49 (0) 72 19 - 65 85 90
Fax : +49 (0) 72 19 - 65 85 89
Internet: http://www.profibus.com

Authors:

Manufacturers of drive technology:

Dunker	SEW-Eurodrive	Bruchsal
Günzel	Moeller	Bonn
Jessen	Danfoss	Gråsten
Köslich	Baumüller	Nuremberg
Leurs	Rexroth Indramat	Lohr am Main
Mirbach	Lenze	Hamelin
Moeller-Nehring	Siemens	Erlangen
Pollmeier	ESR Pollmeier	Ober-Ramstadt
Senneka	Stöber	Pforzheim
Schlabach	Parker Hannifin	Offenburg
Schmitt	Siemens	Erlangen

Manufacturers of fieldbus controller boards:

Hadlich	ifak system GmbH	Magdeburg
Lange	Softing	Haar
Wagner	Hilscher	Hattersheim

Please direct all notes and comments to Mr. Stefan Pollmeier gl@esr-pollmeier.de.

Table of revisions:

Version	Name	Company	Comment
0.1	Mirbach	Lenze	Creation
1.0	Mirbach	Lenze	Integration of comments

Table of contents

1 Introduction

The DriveServer specification /3/ was developed as part of an initiative undertaken by the DRIVECOM User Group. It is based on the OPC Data Access 2.04 specification /1/ and makes a distinction between BusServers, which are fieldbus-specific, and drive servers, which are device-specific but not fieldbus-specific. This document details the specific requirements which must be met by a Profibus OPC server in order for it to be compatible with the DRIVECOM DriveServer specification.

The channel defined in the PROFIDrive profile V3 /2/ is used for parameter access. To ensure the correct implementation of this channel, a consortium was formed within the PROFIBUS PROFIDrive working group which created what is known as the profile server, an implementation of the PROFIDrive parameter channel. This implementation can and should be referred to when creating DRIVECOM-compatible PROFIBUS-DP BusServers.

1.1 Aims of this document

The DriveServer specification covers only those parts which are fieldbus-independent. To ensure that BusServer implementations are fully compatible and therefore interchangeable, in addition to the DriveServer specification, this document also contains specific details about the Profibus-DP fieldbus system. The target group for this document is therefore manufacturers of fieldbus cards and OPC servers which support Profibus-DP.

1.2 Sources of reference

It is in the interest of drive manufacturers that as many manufacturers of fieldbus products as possible support the DriveServer specification, so that test materials and documentation can be acquired from such companies. Specifically, the following materials and sources of reference were used:

Material	**Source of reference**
DriveServer in object code	Can be acquired from any drive manufacturer
Generic DRIVECOM DriveServer in source code	Can be acquired by joining the DRIVECOM User Group
BusServer	Sold by manufacturers of fieldbus cards
Profile server	Can be acquired by joining the PROFIDrive Profile Server consortium
Test devices	Can be acquired from drive manufacturers on request

2 Architecture

There are two ways to integrate (migrate) an existing PROFIBUS-DP OPC server into a DRIVECOM-compatible architecture. These are described briefly below.

2.1 Type 1: Using the profile server

The profile server is superimposed onto a PROFIBUS-DP OPC server, to which only the smallest of modifications have to be made. The PROFIDrive channel is integrated fully into the profile server, but communication itself takes place in the lower-level server. The advantage of this type is that it speeds up the integration of a standard OPC server. However, it does have the disadvantage of an additional OPC interface between the application itself, the OPC client, and the communication module. It is also more difficult to configure this type of system, as every OPC server usually has an obligatory minimum configuration.

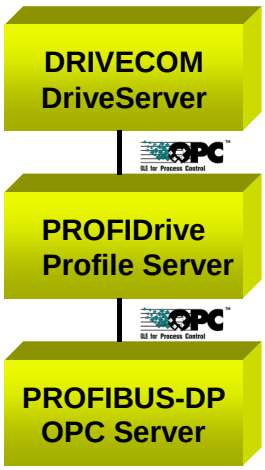

Figure 1: Architecture type 1

2.2 Type 2: Integrating the PROFIDrive channel into the PROFIBUS-DP server

The main advantage of integrating the PROFIDrive channel fully into the PROFIBUS-DP server is that it simplifies operation for the user, as there is only one OPC server. It can therefore be assumed that the performance of this OPC server will be better because one OPC interface has been removed. This type of migration is recommended for industrial applications.

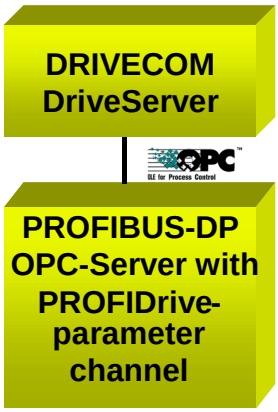

Figure 2: Architecture type 2

3 General requirements of PROFIBUS-DP BusServers

The requirements to be met by any PROFIBUS-DP BusServer, i.e. an OPC server, which can be operated directly beneath a DriveServer, are described below. Additional requirements are listed in Sections 4 and 5 depending on which of the architectures described in Section 2 is used.

3.1 Support of DPV1-MS-AC2 services

The PROFIBUS-DP server must support Class 2 master acyclic services (Initiate, Read, Write, Abort, DataTransport).

3.2 Automatic detection of the devices connected to the bus

The so-called Life-List detects the devices connected to the bus when the PROFIBUS-DP server starts up. The bus addresses of the connected devices can then be derived from the information in this list. This information is forwarded to the OPC client (which may be the profile server) using special DRIVECOM-OPC items (see 3.3). The device types are identified exactly by the OPC client (e.g. DriveServer).

3.3 DRIVECOM OPC items

The DriveServer specification requires that a number of OPC items are created in the name space for every device connected on the bus (see /3/ Section 3.3.2.1). The PROFIBUS-DP BusServer creates these items automatically during the start-up phase once it has created the Life-List. The items, which the DriveServer specification designates as "mandatory" for BusServers, are listed below. A PROFIBUS-DP server used in architecture type 1, however, must only implement `DS_Vendorname` (see 4.1). In this type of architecture, the profile server makes all three items available to its OPC client.

Item name	Description
`DS_Vendorname`	The manufacturer ID according to /2/ Appendix A6. This information can be found in PNU964 subindex 0.
`DS_Devicename`	The device type. This information can be found in PNU964 subindex 1.
`DS_DeviceID`	The device address derived from the Life-List. This information can also be found in PNU 918.

If, in addition to PROFIBUS-DP, a BusServer supports another fieldbus, it must also implement the DS_BusSystem item. This enables a client to distinguish whether the device in question is connected to PROFIBUS or to another bus.

If an OPC server is able to address a number of independent bus lines (e.g. 2 PC cards or cards with several channels) simultaneously, the `DS_BusPort` item must also be implemented.

The PROFIBUS-DP server can also tell the client which profile the device supports. The `DS_ProfileID` item can be used for this purpose if required.

3.4 Registration

During installation, a Profibus-DP BusServer must add the following component categories to the registry:

- `CATID_BusServer10`
- `CATID_PROFIBUS_DP`

The values of the above component categories can be found in /3/ Section 4.

4 Requirements of the profile server on PROFIBUS-DP OPC servers (architecture type 1)

The profile server can be used to create a speedy solution. In addition to the general requirements in Section 3, the requirements described below must also be met.

4.1 DRIVECOM OPC items

An OPC item called `DS_Vendorname` must be created in the name space for every PROFIBUS device identified using the Life-List. The content of this item is freely assignable.

4.2 Communication

Communication between the profile server and the PROFIBUS OPC server takes place via index 47 slot 0. In accordance with Section 3.3.2.1 of the DriveServer specification /3/, a syntax was defined which permits the dynamic creation during runtime of items which access PROFIBUS objects via slot/index addressing.

```
<dp_object>          := "SLOT" | "slot"
                        <slot_info><index_info><datatype_info>
<slot_info>          := (I|i)[0-9]+
<index_info>         := (S|s)[0-9]+
<datatype_info>      := (D|d)(VT_UI1 | VT_ARRAY | ... )
                        (The numerical values to be applied are listed in
                         /3/ Section 5)

index_info:          Object index
slot_info:           Object slot
```

The following item is required specifically:

`sloti0s47d8209:` Access to slot 0 index 47. The data type is VT_ARRAY| VT_UI1. Byte fields of **variable lengths** can therefore be transmitted. Index 47 is used for parameter access on PROFIDrive-compatible devices.

The following options are available for implementation in the PROFIBUS-DP OPC server:

1. The `sloti0s47d8209` item is added to the OPC name space automatically for each device, together with the OPC items from Section 3.3.

2. The PROFIBUS-DP OPC server enables items to be created dynamically in accordance with the above syntax. This procedure, which makes it possible to access any object, is highly recommended.

5 Requirements of PROFIBUS-DP BusServers with integrated PROFIDrive channel (architecture type 2)

To facilitate the operation of a PROFIBUS-DP BusServer, the profile server should be integrated into the BusServer and not installed as a stand-alone OPC server. This architecture is also recommended as a way to improve overall performance. In addition to the requirements in Section 3, the following requirements, which can essentially be taken from the source code of the profile server, must also be met.

5.1 Parameter access via the DRIVECOM syntax

A syntax which permits the dynamic creation of OPC items during runtime and does not require the items to be preconfigured is defined in /3/ Section 3.3.2.1. The parser for evaluating the syntax can be taken from the existing implementation of the profile server.

If PROFIBUS-DP servers permit the use of preconfigured OPC items, the two "dynamic and static creation of OPC items" operating modes should always be possible in parallel.

5.2 Single/multiple parameter access

The PROFIDrive profile defines single and multiple parameter access, i.e. the reading/writing of individual parameters or lists of parameters. By definition, the BusServer will always attempt multiple parameter access if:

- an OPC read or write is to be processed with a list of items or a group refresh and
- all the OPC items concerned are administered in a single group and
- all the OPC items concerned belong to the same device. If the list contains OPC items from several devices and several items are present for each device, the access attempts must be sorted accordingly and grouped into multiple parameter access attempts for each device. The Profibus message length must be observed.

If the maximum message length is exceeded due to the number of OPC items in a group, the requests should be grouped into suitable blocks and processed in several multiple parameter access attempts.

If multiple parameter access fails completely, it may not be supported by the device. In this case, single parameter access attempts should be processed.

The profile server implementation provides functions for both single and multiple parameter access.

5.3 Data types

Type conversion between OPC data types and PROFIDrive data types is governed by the following convention:

Variant type	PROFIDrive data type
VT_I1	integer8, BYTE
VT_I2	integer16, WORD
VT_I4	integer32, DWORD
VT_UI1	unsigned8, BYTE
VT_UI2	unsigned16, WORD
VT_UI4	unsigned32, DWORD
VT_R4 VT_R8	float, DWORD
VT_BSTR	VisibleString
VT_ARRAY | VT_UI1	all types (raw format)

If a type cannot be mapped directly, for example because the requested variant data type is shorter than the PROFIDrive data type, the first bytes of the PROFIDrive data type should be used. If the variant type is longer than the PROFIDrive type, the bytes which are not used should be assigned the value zero.

The length of the VT_ARRAY | VT_UI1 data type can be adapted dynamically. If a length is not explicitly specified when creating an OPC item with this data type, the length is derived from the fieldbus data stream and a field of an appropriate size is created. The length can vary for each read/write procedure call. If the length is explicitly specified, arrays of the specified length are transmitted. If the specified length differs from the actual length of the data, data may be truncated or unused field elements assigned the value zero.

Example 1:

Specified length: 32 bits (4 bytes)

Bytes received: AA BB CC DD EE FF GG HH

Reading the items supplies: AA BB CC DD

Example 2:

Specified length: 64 bits (8 bytes)

Bytes received: AA BB CC DD

Reading the items supplies: AA BB CC DD 00 00 00 00

5.4 Multi-axis devices

If a PROFIBUS device supports more than one axis, the number of axes will be noted in PNU 964 at subindex 5. The DRIVECOM items must then be created in the name space for each axis. A PROFIBUS device with lower-level axes may be placed on a lower level in the hierarchy of the name space. The value of the `DS_DeviceID` item must be different for each axis, otherwise the integrity of the DriveServer name space will be compromised. It should comprise the device address, a separator and the axis number. The convention used in the profile server and illustrated below, in which an underscore is used to separate the device address and the axis number, is recommended.

The following convention has been defined:

(content of PNU 918) + "_" + "(axis number)

Assuming a master device with Profibus address 7 and two slave axes, the name space of the BusServer with integrated PROFIDrive channel might look like this:

```
Server_XYZ
      Station7
            DS_Vendorname
            DS_DeviceID                 Value: "7"
            Axis_1
                  DS_Vendorname
                  DS_DeviceID           Value: "7_1"
            Axis_2
                  DS_Vendorname
                  DS_DeviceID           Value: "7_2"
```

The DriveServer name space would then look like this:

```
BusServer Name
      7
            DS_Vendorname
            DS_Devicename
            DS_DeviceID
      7_1
            DS_Vendorname
            DS_Devicename
            DS_DeviceID
      7_2
            DS_Vendorname
            DS_Devicename
            DS_DeviceID
            ...
```

5.5 Error codes

The PROFIDrive profile defines a series of numerical codes which are fed back in the event of an error. However, these values cannot be fed back using OPC methods and must therefore be converted to OPC standard codes using the table below. The OPC specification /1/ specifies which feedback value is permitted for which OPC function.

PROFIDrive error code	OPC feedback value
0	OPC_E_INVALIDITEMID
1	OPC_E_BADRIGHTS
2	OPC_E_RANGE
3	DISP_E_BADINDEX
4	E_INVALIDARG
5	OPC_E_BADTYPE
6	OPC_E_BADRIGHTS
7	OPC_E_BADRIGHTS
9	E_INVALIDARG
11	E_ACCESSDENIED
15	E_INVALIDARG
17	E_INVALIDARG
20	E_INVALIDARG
21	E_OUTOFMEMORY
22	OPC_E_INVALIDITEMID
23	E_INVALIDARG
24	DISP_E_BADPARAMCOUNT
All others	E_FAIL

6 Glossary

BusServer	An OPC server used for communicating with devices. The server's name space and the items available are communication-specific.
DriveServer	An OPC server used for communicating with devices (drives). The server's name space and the items available are device-specific
Name space	Overview of the data points known to the OPC server. A name space can be structured as a hierarchy or in `flat` format, i.e. all available data items appear in a list.
PNU	Parameter number of a PROFIDrive parameter
Profile server	An OPC server which implements the parameter channel of the PROFIDrive profile.
Slot/Index	Addressing mode in DPV1. The slot is used on the slave to address a real or virtual module; it is not evaluated by PROFIDrive. A data block in the slave is addressed with the index.
Index/Subindex	Addressing of the parameters in the drive controller. This index is not the same as the DPV1 index!

7 References

/1/ OLE for Process Control, Data Access Custom Interface Standard, Version 2.0

http://www.opcfoundation.org/

/2/ PROFIDrive – Profile Drive Technology, Version 3

http://www.profibus.com/

/3/ DriveServer specification, Version 1.1

http://www.drivecom.org/