

## Purpose

The purpose of the document is to make a short description of how to set up a PROFIBUS FMS connection between an IGSS32 system and a Simatic S7 that uses both polled and event oriented communication.

## *Content*

The document contains a description of how to set up a connection that can be used for normal polling and extends this description with guidelines for adding event oriented communication i.e. the simatic writes unsolicited data to IGSS32.

The document assumes the reader to have a reasonable high level of knowledge to the involved systems (IGSS32, Simatic and Profibus FMS).

## *Prerequisites*

This description assumes the presence of:

- 1 IGSS32 version 2 or higher communication server
- 1 Softing ProfiBoard
- 1 Simatic S7/400 equipped with CP443-5 basic
- 1 PG with Simatic Step7 version 5.1 or higher

This document is accompanied by an Simatic S7 project file called [FMSEVENT.ZIP](#). This file contains a minimum Step7 project that shows how to use FMS communication with the IGSS32 system and Simatic S7. Throughout this document a number of examples are used. These examples refer more or less directly to the contents of this Step 7 project. The Step 7 project can therefore be used as a basis for testing and building up event oriented communication on top of the Profibus FMS protocol.

## *Connection set up*

Install the ProfiBoard in the IGSS32 server and install the appropriate board drivers available from Softing GmbH. Start the IGSS32 driver setup program and add the Profibus FMS protocol driver to a profile. Choose a set of device setting defining the primary Profibus parameters. These could e.g. be:

Local station address = 10

Baudrate = 1,5Mbps

Highest station address = 126

Tslot = 3000

Min Tsdr = 150

Max Tsdr = 980

Tquiet = 0

Tsetup = 240

TTR = 115000

Gap factor = 50

Max. retry limit = 1

Add a node to the board and setup the node specific Profibus parameters. These could e.g. be:

IGSS node number = 3

Remote Profibus Address = 3

Source SAP number (SSAP) = 11

Destination SAP number (DSAP) = 11

Send High Priority = 0

Send Low Priority = 241

Receive High Priority = 0

Receive Low Priority = 241

And set up the node specific FMS parameters. These could e.g. be:

Connection type = MMAC

Control interval = 0

SCC = 3

RCC = 3

SAC = 0

RAC = 0

Extension: Click "Not Used"

In the Step7 you have to set up a FMS connection. This is done by starting NetPro and creating a Profibus network and setting up the network parameters. These could e.g. be:

Highest Profibus address = 126

Transmission Rate = 1,5 Mbit/s

Profile = Universal (DP/FMS)

Selecting the "Universal (DP/FMS)" profile fixes the Profibus bus parameters at predefined values. These values should correspond to the values set up the IGSS32 DriverSetup program.

After creating the network you should connect the CP443-5 to it by clicking on the CP and adding it to the network by using the "Properties" button in the "General/Interface" box.

Now the IGSS32 system has to be defined as a communication partner. This is done by creating a new network object using the menu item "Insert/Network Item" and selecting a network object of type "Other station". Attach the IGSS32 network object to the Profibus net using the "Properties/Interfaces" dialog on the object.

The IGSS32 system and the Simatic S7 systems are now defined on the same network and you need to make a logical connection. This is done by clicking on the CPU of the S7 system and selecting "New connection". Select IGSS32 as the communication partner and the communication type as "FMS Connection". In the "General" dialog you should set the "Partner type" to "General FMS Master" and press the "Options" button. This opens a new dialog where you should set up the Local and Remote LSAPs. These should correspond to the Source- and destination SAPs that has been set up in the IGSS32 DriverSetup program (e.g. Local SAP = Remote SAP = 11). Press the "Details" button to set up the FMS communication parameters. These parameters should correspond to the node specific FMS parameters that have been set up using the IGSS32 DriverSetup program. (e.g. Control Interval = 0, Send High Prio = Recv High Prio = 0, Send Low Prio = Recv Low Prio = 241, SCC = RCC = 3 and SAC = RAC = 0). All other parameters in the "FMS connection options" dialog could be left at the default settings (please ensure however that "Read" and "Write" are enabled in the "Services" pane).

Now the logical FMS connection between the IGSS32 system and the Simatic S7 has been defined. To create an object in the Simatic that can be read from IGSS32 you need to create a new data block and go into the "Symbols"-editor and select the object. Click on the object and select the "Special object properties/Communication". Enable the "Use symbol as communication variable" switch and select the "Structure" dialog. The "Structure" dialog is used to define the FMS structure of the data object. In order for IGSS32 to be able to read/poll the object it has to be of FMS type "ARRAY[XXX] OF unsigned16". This could e.g. be done by selecting the "To first structure level of a DB" switch and setting an FMS base index of e.g. 100 and "number of reserved indexes" to 1. This set up makes the data block available on the FMS network as a FMS array object with index number 100.

Download the project to the Simatic S7 system!

To read/poll the object in IGSS32 you could e.g. define an analog object with data group 100 (data group should be equal to FMS object index) and offset 1 (the first data word in the object). Install and start the IGSS32 configuration to monitor the data on the diagram.

## ***Adding event oriented communication***

The IGSS32 system also acts as an FMS server. This allows other stations on the Profibus FMS network to send unsolicited data to IGSS32 using the IGSS32 event oriented protocol.

The IGSS32 system publishes one special FMS object for this purpose. The FMS object has index 21 and is of type integer array with a length of 100 words. In order to be able to write from the Simatic S7 to this object you need to install and call the "WRITE" function block (FB6) in the PLC. The function call could e.g. look like this:

```
CALL "WRITE" , DB401  
  
REQ :=M7.0
```

```

ID      :=DW#16#20001

VAR_1  :=P#DB201.DBX6.0 BYTE 7

SD_1   :=P#DB10.DBX 0.0 WORD 100

DONE   :=DB201.DBX13.0

ERROR  :=DB201.DBX13.1

STATUS:="SOME_DATA".DATA[0]

```

DB401 is an instance DB used as working space for the “WRITE” function.

REQ is a bit used for triggering the “WRITE” job. In this case M7.0.

ID is the ID number of the connection to the IGSS32 system.

VAR\_1 is a pointer to a DB describing variable to write to.

SD\_1 is a pointer to the actual data to be sent.

DONE is a bit flag indicating job completion. In this case DB201.DBX13.0

ERROR is an error flag. In this case DB201.DBX13.1

STATUS is a status word.

The DB’s pointed to could e.g. look like this:

```

DB201
+0.0  Write_REQ      BOOL    FALSE
+2.0  Write_ID       DWORD   DW#16#20001  The ID of the connection to IGSS
+6.0  Write_VAR_Index  STRING[4]  '<21>' Write to IGSS object index 21
+12.0 Write_Done     BOOL    FALSE
+12.1 Write_Error    BOOL    FALSE
+14.0 Write_Status   WORD    W#16#0
+16.0 Write_Statusspeicher  WORD    W#16#0

```

```
DB10
+0.0  IGSS_EVENT_TELEGRAM ARRAY[1..100]
      WORD
```

The actual content of DB10 containing the IGSS\_EVENT\_TELEGRAM structure should look like this:

```
DB10.DBW0  Length of following IGSS event telegram
DB10.DBW2  1. word of IGSS event telegram
DB10.DBW4  2. word of IGSS event telegram
...
...
```

Example (1):

[efesotomasyon.com](http://efesotomasyon.com)

```
DB10.DBW0  W#16#0004  // Total length
DB10.DBW2  W#16#0103  // Data telegram with length 3
DB10.DBW4  W#16#0330  // Node 3, data group 0x30 = 48
DB10.DBW6  W#16#0050  // Offset 0x50 = 80
DB10.DBW8  W#16#1234  // Value
```

Example (2):

```
DB10.DBW0  W#16#0006  // Total length
DB10.DBW2  W#16#0105  // Data telegram with length 5
DB10.DBW4  W#16#0320  // Node 3, data group 0x20 = 32
DB10.DBW6  W#16#0010  // Offset 0x10 = 16
DB10.DBW8  W#16#1234  // Value 1
DB10.DBW8  W#16#2345  // Value 2
DB10.DBW8  W#16#3456  // Value 3
```

With these function blocks in the Simatic it is possible to send an event oriented telegram to the IGSS32 system by triggering a WRITE command (making a pulse on M7.0). Please refer to the IGSS32 event oriented communication documentation for further description on the format IGSS32 of the telegram in the event oriented protocol. Please refer to the various Simatic documentation for further information on how to use the “WRITE” function block properly.