Communication between PC and SIMATIC S7 over PROFIBUS Network Components

S7 Communication

FAQ · May 2011
Question

How do you exchange data between PC and SIMATIC S7 over PROFIBUS network components using the S7 communication?

Answer

The instructions and notes listed in this document provide a detailed answer to this question.
# Table of Contents

1. **Introduction** .................................................................................................................. 5
2. **Overview of the Sample Program** .................................................................................. 6
   - Plant configuration ........................................................................................................... 6
   - Function of the sample program ...................................................................................... 6
   - Function overview ............................................................................................................ 7
   - Data communication ......................................................................................................... 7
   - Active partner and passive partner .................................................................................. 7
   - Program overview ............................................................................................................ 8
   - User program of the active partner PC station ............................................................. 8
   - User program of the passive partner S7-400 station ..................................................... 8
3. **Plant Configuration** ....................................................................................................... 9
   - Overview .......................................................................................................................... 9
3.1 **Overview** .................................................................................................................... 9
3.2 **Hardware and Software Components Used** ................................................................ 9
   - Hardware components ..................................................................................................... 9
   - Required cables and other hardware .............................................................................. 10
   - Software requirements .................................................................................................... 10
   - Wiring ................................................................................................................................ 10
4. **Function Mechanisms of the Sample Program** .............................................................. 11
   - Working Method of the Sample Program ......................................................................... 11
   - Data Communication ....................................................................................................... 11
   - Sequence of the User Program in the Active Partner PC Station ................................ 11
     - Increase data ............................................................................................................... 11
   - Sequence of the User Program in the Active Partner S7-400 Station ......................... 11
   - Overview .......................................................................................................................... 11
4.1 **Functions and Function Blocks** .................................................................................. 12
   - Functions and Function Blocks in the User Program of the S7-400 Station ................... 12
4.2 **Details of the User Program of the PC Station** ............................................................. 12
   - Program Sequence .......................................................................................................... 12
   - WndProc () ..................................................................................................................... 13
     - Overview ....................................................................................................................... 13
   - My_write_request () ...................................................................................................... 14
     - Overview ....................................................................................................................... 14
   - Parameters of the s7_write_req () function ................................................................. 14
     - Return value .................................................................................................................. 15
   - My_read_request () ....................................................................................................... 15
     - Overview ....................................................................................................................... 15
   - Parameters of the s7_read_req () function .................................................................... 15
     - Return value .................................................................................................................. 16
   - My_get_read_confirmation () ........................................................................................ 16
     - Overview ....................................................................................................................... 16
   - Parameters of the s7_get_read_cnf () function ............................................................. 16
   - Return value .................................................................................................................... 16
4.3 **Program Details of the User Program of the S7-400 Station** ..................................... 16
   - OB1 ................................................................................................................................ 17
     - Overview ....................................................................................................................... 17
4.4 **Other Notes, Tips and Tricks, etc** ................................................................................ 18
   - Not Possible to Download Hardware Configuration, Connection or Blocks to the S7-400 CPU ......................................................................................................................... 19
## Table of Contents

6.2 No Change in the Values of the Written and Read Data ................... 19
6.3 No Change in the Network Status...................................................... 20
1 Introduction

This document provides information about the examples of PROFIBUS communication between a PC station and a SIMATIC S7 using the PUT and GET services based on the S7 communication and incorporating network components. Optical Link Modules (OLM) are used as network components.

Chapter 2 gives an overview of the sample program.
Chapter 4 gives information about the function mechanisms of the sample program.

This document contains

- An overview of the plant configuration.
- An introduction to the blocks required and their communication structure.
- Details of the user program of the PC station and SIMATIC S7.
- Information on how to use the sample programs.
- Other notes, tips and tricks, etc.
2 Overview of the Sample Program

Plant configuration

Figure 2-1

Alarm contacts of the OLM

The alarm contacts of the OLMs are made to the digital input module of the S7-400. The input areas used are

- E 0.0 = alarm contact OLM 1
- E 0.1 = alarm contact OLM 2

Function of the sample program

You can see whether the example is functioning correctly in the dialog window of the PC. This window is displayed when you start the PC application. The data sent and received changes its value constantly. The status of the OLMs is also displayed. The status changes automatically upon data transfer.
Function overview

The overview of functions shows the principal working method of the sample program.

Figure 2-2

Active partner
PC station

Prepare data for sending

Write data (2 bytes)

Provide memory area

Read data (2 bytes)

Save and display received data

Passive partner
S7-400 station

Data communication

2 bytes of data are exchanged cyclically between the two stations involved in the communication. The user data is in the first byte and the OLM status in the second byte. These are displayed in the relevant dialog window on the PC.

Active partner and passive partner

The active partner writes the data to the marker area of the passive partner. Then the active partner reads back the data from the same marker area. With the write procedure the network status is initialized with "0" so that a new status can be recognized.

In the passive partner the alarm contacts of the network components are updated in the basic cycle (OB1) and written to the second byte of the data field.
Program overview

Figure 2-3

User program of the active partner PC station

The network status is determined from the two bytes received.

After the data has been written and read, the data to be written is incremented by the value of "1" and the write/read cycle starts anew with the PUT and GET functions.

User program of the passive partner S7-400 station

The alarm contacts are updated cyclically and written to the send buffer.

A user program is not required for data transfer, because the necessary functions are called automatically by the internal operating system.
3 Plant Configuration

This chapter gives you an overview of the configuration and the hardware and software components used to create the sample program.

3.1 Overview

Table 3-1 gives an overview of the configuration used to create the sample program.

Table 3-1

<table>
<thead>
<tr>
<th>Bus system</th>
<th>PROFIBUS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Communication protocol</td>
<td>S7 communication</td>
</tr>
<tr>
<td>Active partner</td>
<td>PC station</td>
</tr>
<tr>
<td>Passive partner</td>
<td>SIMATIC S5</td>
</tr>
<tr>
<td>Communications processors</td>
<td>CP5613, CP443-5 Basic</td>
</tr>
<tr>
<td>Services</td>
<td>PUT and GET with evaluation of alarm contacts</td>
</tr>
</tbody>
</table>

The S7 communication is used in this example. Data is exchanged between a PC station and a SIMATIC S7 via this protocol and the PUT / GET services.

In the active partner PC station a CP5613 communications processor is used for data communication by means of S7 communication and the PUT / GET services.

A CP443-5 Basic communications processor is used in the passive partner S7-400 station.

3.2 Hardware and Software Components Used

Hardware components

The following modules in the active partner PC station were used to create the sample program.

Table 3-2

<table>
<thead>
<tr>
<th>Module</th>
<th>Order number</th>
</tr>
</thead>
<tbody>
<tr>
<td>PC</td>
<td></td>
</tr>
<tr>
<td>CP5613</td>
<td>6GK1561-3AA00</td>
</tr>
</tbody>
</table>

The following modules in the passive partner S7-400 station were used to create the sample program.

Table 3-3

<table>
<thead>
<tr>
<th>Slot</th>
<th>Module</th>
<th>Order number</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>PS 407 10A</td>
<td>6ES7407-0KA00-0AA0</td>
</tr>
<tr>
<td>2</td>
<td>CPU 414-1</td>
<td>6ES7414-1XG02-0AB0</td>
</tr>
<tr>
<td>4</td>
<td>CP443-5 Basic</td>
<td>6GK7 443-5FX00-0XE0</td>
</tr>
</tbody>
</table>
Notes

The sample project has been created with a specific hardware configuration. This must be maintained to ensure proper functioning.

If you use a different configuration, with a different CPU or CP, for example, then you must change the sample program accordingly.

Required cables and other hardware

- MPI cable
- PROFIBUS bus cable
- OLM for PROFIBUS
- SIMATIC Field PG or PC with MPI interface

Software requirements

- STEP 7 V5.03 or higher
- NCM S7 INDUSTRIAL ETHERNET V5.03 or higher
- SIMATIC NET PC software S7-5613/Windows NT for CP5613 or higher

Wiring

You must set the isolated contacts to the input module of the S7-400 station to evaluate the alarm contacts of the OLMs.

Figure 3-1
4 Function Mechanisms of the Sample Program

This chapter describes how the sample program functions and which blocks are required for communication and their communication structure.

4.1 Working Method of the Sample Program

4.1.1 Data Communication

2 bytes of data are exchanged cyclically between the two stations involved in the communication.

4.1.2 Sequence of the User Program in the Active Partner PC Station

You need a user program in the active partner PC station. This uses functions from a library for the communication. The exact functions of the user program are described in section 4.3.

Increase data

The value of the data to be written is increased by one and the write/read cycle starts over.

4.1.3 Sequence of the User Program in the Active Partner S7-400 Station

In the active partner S7-400 station the alarm contacts of the OLMs are read in and put as status byte in the send buffer for the passive partner. In this way the passive partner reads out the network status in addition to the user data.

4.1.4 Overview

Figure 4-1 gives a graphical view of how the user programs work in the active and passive partners.

Figure 4-1
4.2 Functions and Function Blocks

4.2.1 Functions and Function Blocks in the User Program of the S7-400 Station

No functions and function blocks are called in the user program of the S7-400 station to transfer data. The data is provided by the internal operating system functions.

4.3 Details of the User Program of the PC Station

4.3.1 Program Sequence

Figure 4-2

![Flowchart Diagram]
4.3.2 WndProc()

Overview

Figure 4-3

```c
/***********************
Function: WndProc

DESCRIPTION: Processes messages for the main window.
**************************/

LRESULT CALLBACK WndProc(HWND hWnd, UINT uiMessage,
                      WPARAM wParam, LPARAM lParam)
{
    /* Message handling */
    switch(uiMessage) {
    /* Point of the menu clicked */
    case WM_COMMAND:
        switch(wParam) {
            /* Manipulate "Write / Read - Start" */
            case IDM_START:
            /* Call the function which initializes the CP */
            /* Establishing connection to remote station*/
            my_init();
            break;
            default:
                break;
        }

    /* S7 messages from the local CP */
    case MY_MSG_ID:
        /* Call the function which processes the messages */
        my_receive();
        break;
    case WM_DESTROY:
        PostQuitMessage(0);
        break;
    default:
        return(DefWindowProc(hWnd, uiMessage,
                         wParam, lParam));
    }
    return (0);
} /* End of WndProc */

/* Functions for dialog boxes */
```

Description

The WndProc() function is called cyclically. It controls the sequence of the application and is therefore comparable with an OB1 in SIMATIC S7.

The WndProc() function processes EVENTS originating from the application window. When the user sends new data, the application generates a message. The messages are used as control values in the WndProc() function.
4 Function Mechanisms of the Sample Program

4.3.3 My_write_request()

Overview

Figure 4-4

/* ***************************************************************
   Function: my_write_request
   Description: writes to a variable of the server.
   Used S7 functions:
   - s7_write_req
****************************************************************/
void my_write_request(void)
{
    int32 iRet;
    struct S7_WRITE_PARA WriteParameter; /* Structure for write */
    /* Increment write data */
    if (uiWriteDataSize)
    {
        uiWriteData++;
    }
    else
    {
        uiWriteData = 1;
    }
    /* Setting the values of the elements of the structure */
    /* Writing to MW25 of remote station (2 Byte). */
    WriteParameter.access = S7_ACCESS_SIMO_ADDRESS; /* by symbol */
    /* Set name of variable to write: */
    /* 
    * wprint(WriteParameter.var_name, "MW25");
    * WriteParameter.var_length = 2; /* 2 Byte of data */
    * WriteParameter.value[0] = 0x00; /* reset one error */
    * WriteParameter.value[1] = 1BYTE{uiWriteData}; /* data to write */
    */
    /* Writing to a variable of a server. */
    iRet = s7_write_req(CPDescriptor, CRef, 0, &WriteParameter, NULL);
    /* ERROR? */
    if (iRet != S7_OK)
    {
        my_error_handler();
    }
} /* End of my_write_request */

Description

The my_write_request() function is called in the my_receive() function. The my_write_request() function writes the data to the passive station.

Parameters of the s7_write_req() function

The s7_write_req() function is called with the parameters below.

Table 4-1

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPDescriptor</td>
<td>This parameter contains the addresses of the CP and VFD.</td>
</tr>
<tr>
<td>CRef</td>
<td>This parameter references the connection.</td>
</tr>
<tr>
<td>WriteParameter</td>
<td>This parameter is a structure. It consists of different parameters. The following parameters are required for the function.</td>
</tr>
<tr>
<td></td>
<td>- access: This parameter identifies the type of access. In this example the type access is given by symbols.</td>
</tr>
<tr>
<td></td>
<td>- var_name: The name of the variable is stored in this parameter.</td>
</tr>
<tr>
<td></td>
<td>- varlength: The length of the data to be written is stored in this parameter. In this example 2 bytes of data are transferred.</td>
</tr>
</tbody>
</table>
4 Function Mechanisms of the Sample Program

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value[]:</td>
<td>This parameter is an array in which the data to be sent is stored. The HIBYTE is set to the value “0” because the network status of the passive station is entered in this byte.</td>
</tr>
</tbody>
</table>

Return value

The return value of the s7_write_req() function is stored in the iRet parameter. If the return value does not have the value "S7_OK", then the fault is output and the application is terminated.

4.3.4 My_read_request()

Overview

Figure 4-5

\******************************************************************************
 Function: my_read_request
 Description: Reads a variable from the server.
 Used S7 FUNCTIONS: s7_read_req
 \******************************************************************************

void my_read_request(void) {
    int iRet;
    struct S7_READ_VARA ReadParameter; /* Structure for read */
    /* Parameters */
    /* Setting the values of the elements of the structure. */
    /* Read MB25 of remote station (2 Bytes). */
    ReadParameter.access = S7_ACCESS_PUBLIC_ADDRESS; /* by symbol */
    /* Set name of variable to read: */
    wprintf("ReadParameter.var_name, "MB25");
    /* Reading the variable of the server */
    iRet = s7_read_req(CPDescriptor, CRef, 0, &ReadParameter);
    /* ERROR? */
    if (iRet != S7_OK)
        my_error_handler();
} /* end of my_read_request */

Description

The my_read_request() function is called in the my_receive() function. The my_read_request() function reads the data from the passive station.

Parameters of the s7_read_req() function

The s7_read_req() function is called with the parameters below.

Table 4-2

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPDescriptor</td>
<td>This parameter contains the addresses of the CP and VFD.</td>
</tr>
<tr>
<td>CRef</td>
<td>This parameter references the connection.</td>
</tr>
<tr>
<td>ReadParameter</td>
<td>This parameter is a structure. It consists of different parameters.</td>
</tr>
<tr>
<td></td>
<td>The following parameters are required for the function.</td>
</tr>
<tr>
<td></td>
<td>• access:</td>
</tr>
<tr>
<td></td>
<td>This parameter identifies the type of access. In this example the type access is given by symbols.</td>
</tr>
<tr>
<td></td>
<td>• var_name:</td>
</tr>
<tr>
<td></td>
<td>The name of the variable is stored in this parameter.</td>
</tr>
</tbody>
</table>
4 Function Mechanisms of the Sample Program

Return value

- The return value of the s7_read_req () function is stored in the iRet parameter. If the return value does not have the value "S7_OK", then the fault is output and the application is terminated.

4.3.5 My_get_read_confirmation ()

Overview

Figure 4-6

```
//******************************************************
funcion: my_get_read_confirmation
description: The message about the completion of a variable read job is evaluated.
used s7 functions:
- s7_get_read_cnf
*******************************************************/
void my_get_read_confirmation(void)
{
    int16 iRet;
    ord16 ulVariableLength = 2;  /* Saves after the call the length */
    /* of the variable that was read. */
    /* Evaluate the message. */
    /* Save the length of the read variable. */
    /* Save the value of the variable (in cValue). */
    /* Error */
    if(iRet != S7_OK)
        my_error_handler();
} /* End of my_get_read_confirmation */
```

Description

When the read job has been completed is evaluated in the my_get_read_confirmation () function.

Parameters of the s7_get_read_cnf () function

The s7_get_read_cnf () function is called with the parameters below.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>uiVariableLength</td>
<td>The length of the data read is stored in this parameter.</td>
</tr>
<tr>
<td>cValue</td>
<td>The value of the variable is stored in this parameter.</td>
</tr>
</tbody>
</table>

Return value

The return value of the s7_get_read_cnf () function is stored in the iRet parameter. If the return value does not have the value "S7_OK", then the fault is output and the application is terminated.

4.4 Program Details of the User Program of the S7-400 Station

No functions or function blocks are required for data transfer in the S7-400 station. The alarm contacts are evaluated in OB1.
4.4.1 OB1

Overview

Figure 4-7

```
// The PC always write Zero in Byte 1 of Datafield
// The State of OLM will be insert State for the Get-Function
// 0 = no data available | CPU State STOP ? |
// 1 = OLM OK
// 2 = OLM 1 error
// 4 = OLM 2 error
// 6 = OLM 1+2 error (in this case the PC can’t read data because the
// connection is broken)
// StatusByte | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
//            | 1 | OLM OK | OLM1 ERR | OLM2 ERR |   |   |   |
// Error OLM1
// UN E 0.0
// = M 25.1
// Error OLM2
// UN E 0.1
// = M 25.2
// No ERROR
// U E 0.0
// U E 0.1
// = M 25.0
// EE
```

Description

OB1 is the block responsible for cyclic processing of the user program. The alarm contacts are evaluated in OB1.
5 How to Use the Sample Program

You operate the user program of the PC station via a dialog in which the data is also displayed.

In the dialog you click the "YES" button to trigger a send job. The data is written to the S7-400 station and read out again.

The data read out is incremented by the value of "1" and display in the dialog on the PC station. The network status is updated automatically when the data read out is received.

If there is a fault, the component is displayed and the error reported. If no correct status can be received, this is reported with the message "No Information".

In the dialog you click the "NO" button to close the user program of the PC station.
6 Other Notes, Tips and Tricks, etc.

This chapter gives you more notes, tips and tricks for the commissioning and working of the sample program.

6.1 Not Possible to Download Hardware Configuration, Connection or Blocks to the S7-400 CPU

If it is not possible to download the hardware configuration, connection or blocks to the CPU, then make the following checks and take the following measures to remedy the situation.

Table 6-1

<table>
<thead>
<tr>
<th>Check</th>
<th>Remedy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Check that the MPI cable is connected on the SIMATIC Field PG and on the MPI or MPI/DP interface of the S7-400 CPU. If the MPI cable is not connected to the SIMATIC Field PG and to the MPI or MPI/DP interface of the S7-400 CPU, no connection can be established to the S7-400 station and the configuration cannot be downloaded to the CPU.</td>
<td>Connect the MPI cable to the RS485 interface of the SIMATIC Field PG and to the MPI or MPI/DP interface of the S7-400 CPU. Go to Start → SIMATIC → STEP 7 and select the menu item &quot;Set PG/PC interface&quot;. Under &quot;Interface parameters used&quot; you select the module to which the MPI cable on the SIMATIC Field PG is connected, CP5611 (MPI), for example. Apply the settings with &quot;OK&quot;.</td>
</tr>
<tr>
<td>Check the settings in &quot;Set PG/PC interface...&quot;. The interface parameters used must be assigned to the access point of the application &quot;S7ONLINE (STEP 7)&quot;. If the settings in &quot;Set PG/PC interface...&quot; are not correct, no connection can be established between the SIMATIC Field PG and S7-400 CPU.</td>
<td></td>
</tr>
<tr>
<td>Open the hardware configuration. The hardware configuration in the STEP 7 project must match the configuration of the S7-400 station.</td>
<td>Change the hardware configuration in the STEP 7 project to match the configuration of the S7-400 station.</td>
</tr>
<tr>
<td>Check that the CPU has been completely reset.</td>
<td>If the CPU has not been completely reset, then do a complete reset of the S7-400 CPU.</td>
</tr>
</tbody>
</table>

6.2 No Change in the Values of the Written and Read Data

If the values of the written and read data do not change, then make the following checks and take the following measures to remedy the situation.
### 6 Other Notes, Tips and Tricks, etc.

#### Table 6-2

<table>
<thead>
<tr>
<th>Check</th>
<th>Remedy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Check whether the PROFIBUS bus cable is connected to the CP443-5 in the S7-400 station and to CP5613 in the PC station. If the PROFIBUS cable is not connected to CP443-5 in the S7-400 station and to CP5613 in the PC station, no connection can be established between the two stations. Data exchange is not possible.</td>
<td>Connect the PROFIBUS bus cable to the RS485 interface of the CP443-5 and the OLM. Connect another PROFIBUS bus cable to the RS485 interface of the CP5613 and the OLM.</td>
</tr>
</tbody>
</table>

#### 6.3 No Change in the Network Status

If the network status does not change, then make the following checks and take the following measures to remedy the situation.

#### Table 6-3

<table>
<thead>
<tr>
<th>Check</th>
<th>Remedy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Check whether the PROFIBUS bus cable is connected to the CP443-5 in the S7-400 station and to CP5613 in the PC station. If the PROFIBUS cable is not connected to CP443-5 in the S7-400 station and to CP5613 in the PC station, no connection can be established between the two stations. Data exchange is not possible.</td>
<td>Connect the PROFIBUS bus cable to the RS485 interface of the CP443-5 and the OLM. Connect another PROFIBUS bus cable to the RS485 interface of the CP5613 and the OLM.</td>
</tr>
<tr>
<td>Check that both supply points on the OLM are connected to the power supply. If both supply points are not connected, the OLM reports failure of the power supply.</td>
<td>Connect the second supply point to a power supply as well.</td>
</tr>
<tr>
<td>Check whether monitoring of the port is enabled. You can enable and disable the monitoring function individually for each port. At the ports where monitoring is enabled a node must be connected and the network adapter enabled so that the OLM recognizes a link. You can determine this from the control LEDs (LS [port number]) on the front.</td>
<td>Set the switch on the top side of the OLM to &quot;1&quot; for the ports to be monitored.</td>
</tr>
<tr>
<td>Check that the CPU is in &quot;RUN&quot; mode. Since the user program evaluates the alarm contacts in the S7-400 station the CPU must be in &quot;RUN&quot; mode to deliver a correct status.</td>
<td>Switch the CPU to &quot;RUN&quot; mode via the key switch or the PG command.</td>
</tr>
</tbody>
</table>

---

**Notes**

If these notes, tips and tricks etc. for the commissioning and working of the sample program have not been of assistance, then repeat the commissioning of the sample program.