Application for Communication

applications & TOOLS

Data Transmission via FDL Connection with SDA Service Using AG_SEND / AG_RECV

SIEMENS

Warranty, Liability and Support

We do not accept any liability for the information contained in this document.

Any claims against us - based on whatever legal reason - resulting from the use of the examples, information, programs, engineering and performance data etc., described in this document shall be excluded. Such an exclusion shall not apply in the case of mandatory liability, e.g. under the German Product Liability Act ("Produkthaftungsgesetz"), in case of intent, gross negligence, or injury of life, body or health, guarantee for the quality of a product, fraudulent concealment of a deficiency or breach of a condition which goes to the root of the contract ("wesentliche Vertragspflichten"). However, claims arising from a breach of a condition which goes to the root of the contract shall be limited to the foreseeable damage which is intrinsic to the contract, unless caused by intent or gross negligence or based on mandatory liability for injury of life, body or health. The above provisions does not imply a change in the burden of proof to your detriment.

The Application Examples are not binding and do not claim to be complete regarding the circuits shown, equipping and any eventuality. They do not represent customer-specific solutions. They are only intended to provide support for typical applications. You are responsible in ensuring that the described products are correctly used.

These Application Examples do not relieve you of the responsibility in safely and professionally using, installing, operating and servicing equipment. When using these Application Examples, you recognize that Siemens cannot be made liable for any damage/claims beyond the liability clause described above. We reserve the right to make changes to these Application Examples at any time without prior notice. If there are any deviations between the recommendations provided in these Application Examples and other Siemens publications - e.g. Catalogs - then the contents of the other documents have priority.

For questions about this document please use the following e-mail-address:

csweb@ad.siemens.de

Table of Contents

1 Task Specification 4

2 Principle of the Automation Solution 5

2.1 Required Components 6

3 Function Mechanisms and Program Structures 8

3.1 Principles and characteristics of FDL communication 8

3.2 User interfaces for FDL communication 11

3.3 Principle structure of this example 14

3.4 Program structures 15

3.4.1 Block structure of the sending station (S7-300, station 1) 17

3.4.2 Block structure of the receiving station (S7-400, station 2) 19

3.5 Program execution in station 1 (sending station) 20

3.6 Program execution in the receiving station (station 2) 26

4 Installation of Hardware and Software 29

4.1 Hardware configuration 29

4.2 Software installation 29

4.3 Configuration for connection 30

5 Operator Control and Monitoring 32

5.1 Variable table for the sending station (S7-300, station 1) 33

5.2 Variable table in the receiving station (S7-400, station 2) 35

5.3 Starting data transmission 36

6 Summary 37

7 Further Literature 38

1 Task Specification

Technological task description / overview

The following illustration shows a simple example of two SIMATIC S7 control units linked by a configured FDL connection on the PROFIBUS.

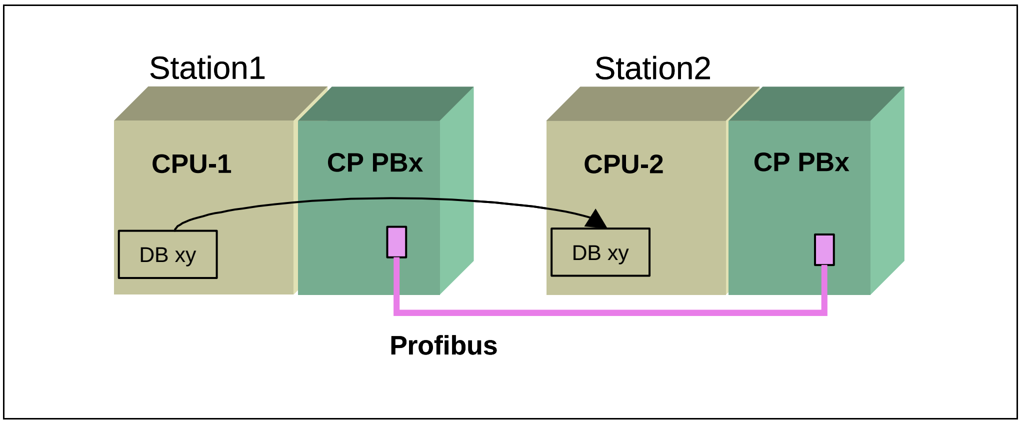


Figure 1-1 Illustration of the technological task

The task to be fulfilled is the provision of a most simple solution for the transmission of small or medium-sized amounts of data from station 1 to station 2, whereby one request to send may include a maximum of 240 bytes of user data to be transmitted.

Solution requirements

- Data transfer shall make use of the properties and possibilities offered by a specified FDL connection with SDA service (Send Data with Acknowledgement).
- 1024 bytes of user data shall be transmitted from the source area of the sending station to the destination area of the receiving station.
- The user program of the sending station shall be provided with a consistency function that splits the 1024 data bytes to be transmitted into 5 partial data sets of max. 240 bytes.
- The user program of the target station shall be provided with a consistency function that re-combines the partial data sets to one consistent data set with 1024 bytes of receive data in the target area.
- Only one single FDL connection shall be used for the transmission of send data.

2 Principle of the Automation Solution

Illustration of components

The illustration below shows the hardware structure of the example application with all components involved.

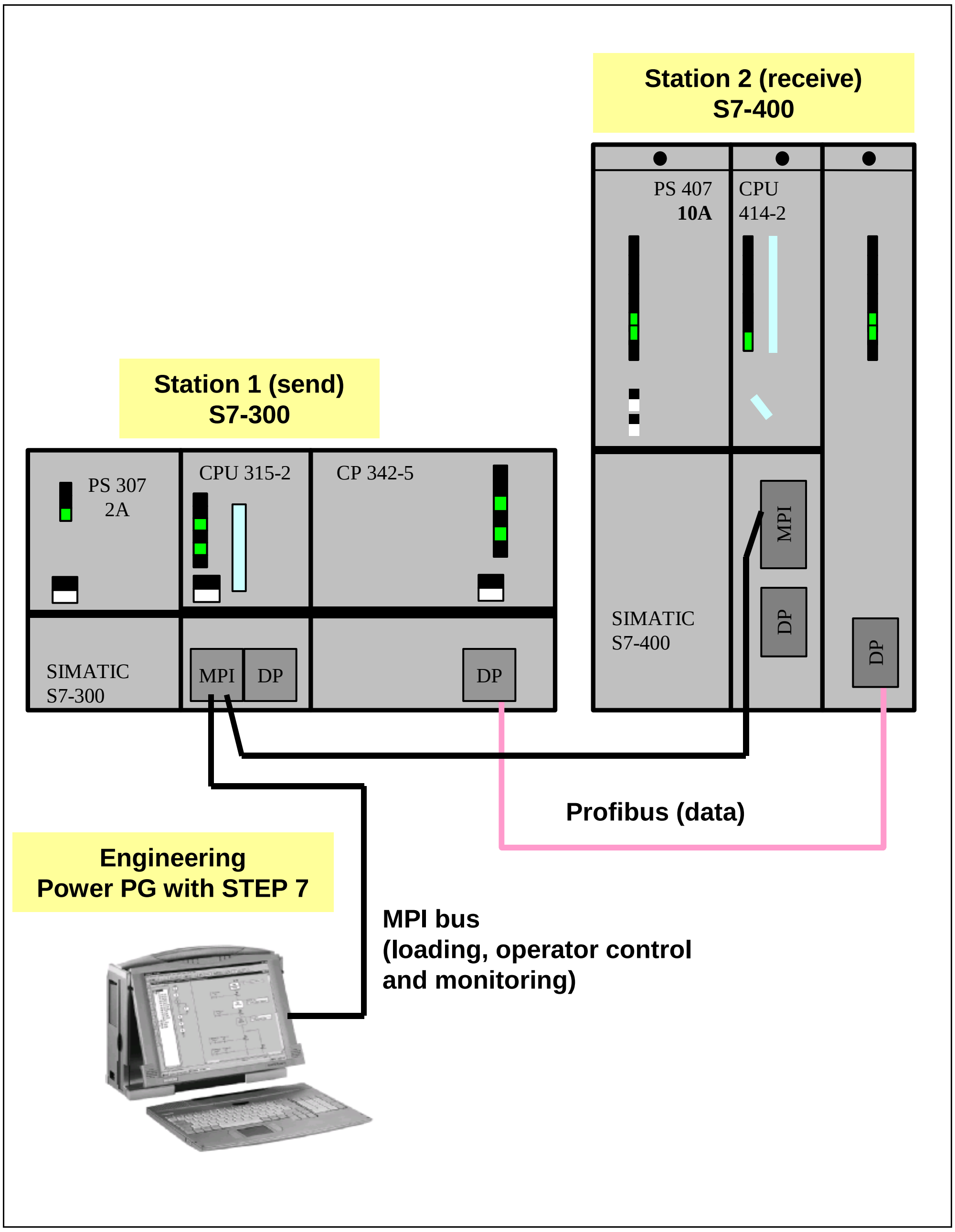


Figure 2-1 Overview of hardware components

2.1 Required Components

Hardware components

The following hardware components are required for realization.

Table 2-1 Hardware components of the S7-300 station

Component	MLFB / Order number	Note
Load current supply unit PS307 120/230V AC:24VDC/2A	6ES7307-1BA00-0AA0	Any PS with sufficient power specifications can be used here.
CPU 315-2 DP	6ES7315-2AG10-0AB0	FW V 2.0
CP 342-5 1)	6GK7342-5DA01-0XE0	A CP 343-5 may be used as an alternative.

1) **Warning**: The type of CP 342-5 (MLFB 6GK7342-5DA01-0XE0) mentioned in the list is no longer available.
If you use the later design version, please change the HW configuration as described in Chapter 4.3 "Configuration for connection".

Table 2-2 Hardware components of the S7-400 station

Component	MLFB / Order number	Note
Rack S7 400 UR2	6ES7400-1JA01-0AA0	Any UR or CR may be used here.
PS 407 10A	6ES7407-0KA00-0AA0	Any PS with sufficient power specifications can be used here.
CPU 414-2 DP	6ES7414-2XG03-0AB0	FW V 3.1
CP 443-5 Basic	6GK7443-5FX01-0XE0	A CP 443-5 Extended can also be used as an alternative.

Software components

The following software components are required for realization.

Component	MLFB / Order number	Note
Step 7 V 5.2	6ES7810-4CC06-0YX0	
NCM S7 Profibus	--	Part of the Step 7 SW.

Further components

Additionally, the following components are necessary for set-up, testing and commissioning:

- PROFIBUS cable
- 2x PROFIBUS connector
- MPI cable
- PG or PC with MPI interface

Example project

The Getting Started example application presented here includes this documentation and a STEP-7 example project which covers the hardware configuration and connections for both stations, as well as an executable STL user program.

Table 2-3 Components used in the example project

Component	Note
SH_0416_Komm_FDL_SDA_DOKU_v01_d.pdf	Documentation, German
SH_0416_Komm_FDL_SDA_DOKU_v01_e.pdf	Documentation, English
SH_0416_Komm_FDL_SDA_CODE_v01_e.zip	STEP 7 example project

Note

This example is based on a specific hardware configuration. When changing the hardware, the hardware configuration of the corresponding station must also be adjusted in the S7 project and compiled anew. If necessary, the STEP 7 program and configuration for connection must be modified accordingly. For further information on hardware and software setup please refer to Chapter 4 "Installation of Hardware and Software"

3 Function Mechanisms and Program Structures

This chapter explains the characteristics of FDL communication and the associated services.

The STEP 7 example project is based on a practical model explaining the techniques deployed to realize a user program that offers data transmission via a specified FDL connection with SDA service.

3.1 Principles and characteristics of FDL communication

FDL communication within the ISO/ OSI 7 reference model

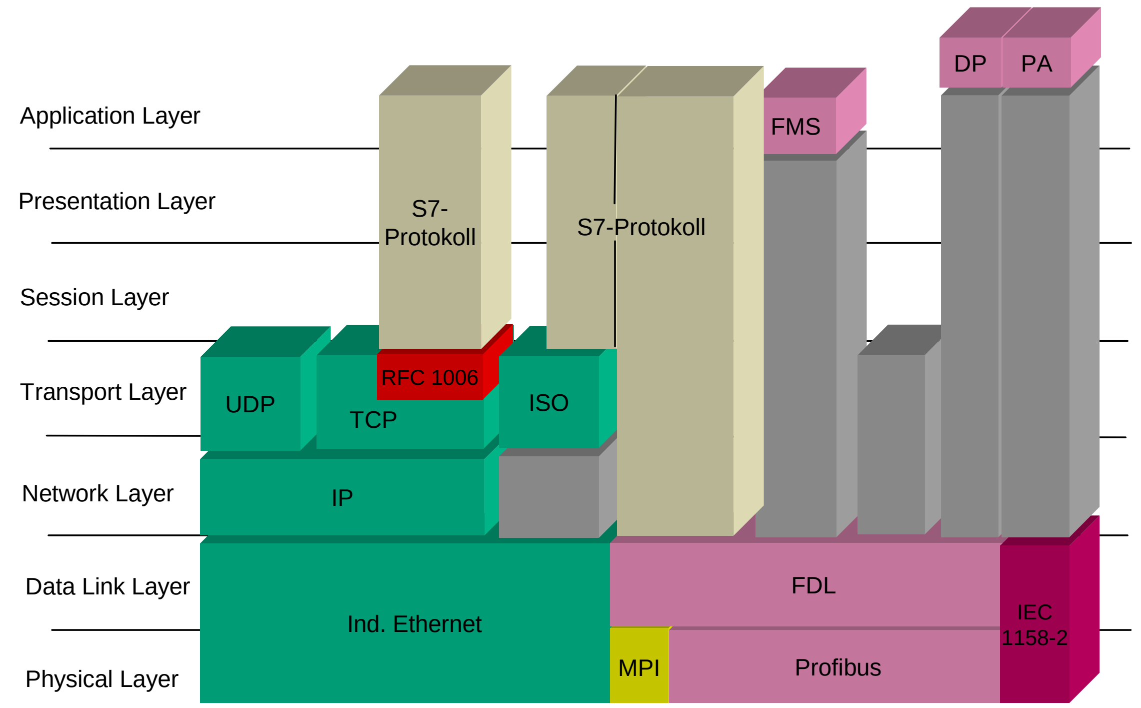


Figure 3-1 Comparison of protocols in the ISO / OSI 7 layer model

In the OSI reference model layers 1 and 2 are used for FDL communication. In the PROFIBUS layer 2 is specified as the FDL layer (Fieldbus Data Link). The layer above layer 2 is a further specific interface which can be assigned to layer 4. All further layers of the OSI reference model are not explicitly assigned.

The telegram formats of the PROFIBUS layer 2 offer high transmission security for FDL communication (Hamming Distance HD = 4). If an error occurs during telegram transmission, the relevant telegram will be repeated automatically.

- **Advantage**: the slim architecture (based on layer 4) offers efficient and quick data transmission for FDL communication.
- **Disadvantage**: since the higher levels of the OSI reference model are not explicitly defined, the user program requires some further programming with regard to the monitoring functions for data transmission and error messages from the communication modules as well as for data management.

In practice this means:

The FDL protocol acknowledges data only up to the CP level; i.e. the end point of connection is the communication processor (CP), not the user program on the CPU. The user program saves the data received by analyzing the status information of the corresponding communication modules.

Types of data transmission via FDL communication

FDL communication uses the following transmission services. These services refer to the relevant PROFIBUS specifications and are based on layer 2 of the OSI reference model:

- SDA - Send Data with Acknowledge
- SDN - Send Data with no Acknowledge.

The following types of connection are possible for FDL communication:

Table 3-1 Types of data transmission with FDL connection

Type of connection	Service	Description
Specified FDL connection (point-to-point) 1)	SDA	Local station and connected station are defined by configuration. Receipt of data with acknowledgement.
Unspecified FDL connection (point-to-point with free access to layer 2)	SDA	Local station defined by configuration. The connected station is addressed in the user program. Receipt of data with acknowledgement.
Broadcast connection	SDN	An active station transmits a message to all other stations. Receipt of data without acknowledgement.
Multicast connection	SDN	An active station transmits a message to a specific group of other stations. Receipt of data without acknowledgement.

1) The user program discussed here uses a specified **FDL communication with SDA**.

Key characteristics of FDL communication

FDL communication is defined by the following main properties:

- Data blocks can be exchanged bidirectionally via one communication link.

- Simultaneous transmission and receipt of telegrams via one line is possible (full duplex).

Systems suitable for FDL communication

FDL communication enables both an S7-homogenous and an S7-inhomogenous structure:

- SIMATIC S7 with PROFIBUS-CP
- SIMATIC S5 with PROFIBUS-CP
- SIMATIC S595U with PROFIBUS interface
- PC stations with PROFIBUS-CP
- Devices of different make which meet the SDA and SDN requirements according to PROFIBUS Standard EN 50170, Vol. 2

Basic performance data for FDL communication

FDL communication can be described by the following performance data:

Characteristic	S7 300	S7 400
Max. data length for a specified FDL connection (SDA)	240 bytes	240 bytes
Max. data length for an unspecified FDL connection (SDA)	236 bytes [1)]	236 bytes [1)]
Max. data length for broadcast or multicast connection (SDN)	236 bytes [1)]	236 bytes [1)]
Possible address ranges	Inputs, outputs in the process image, memory areas (flags) and data blocks	Inputs, outputs in the process image, memory areas (flags) and data blocks
Data consistency	240 bytes	240 bytes
Max. number of connections	See CP device manual [2)]	See CP device manual [2)]
Block types for services	FCs	FCs

1) After a request to send by AG_SEND, the user program registers the remote station in the command header of the command buffer. The command header has 4 bytes, consequently, the maximum size of user data is 236 bytes.

2) The maximum number of possible connections depends on the CP used, the type of CPU and the number of connections in multi-protocol mode.

3.2 User interfaces for FDL communication

Data transmission via a configured FDL connection is performed with the help of FC blocks which are available in the SEND/RECEIVE interface. Data transmission is effected on an event-controlled basis initiated by the user program.

- AG_SEND / AG_LSEND
  These blocks provide user data from a specified data area for transmission to the PROFIBUS -CP.
- AG_RECV / AG_LRECV
  These blocks are used to transfer the user data received from the PROFIBUS-CP to a defined data area.

These communication blocks are included in the SIMATIC NET_CP library and are suitable for use with both, PROFIBUS and Industrial Ethernet. For a detailed description of all blocks please refer to the NCM documentation for PROFIBUS or to the online help on blocks in STEP 7.

Note

- Later versions of the S7-CPs for S7-300 use only the FCs AG_SEND and AG_RECV which are also suitable for use with Industrial Ethernet for the transmission of larger data sets. Further details and downloads are available on the Customer Support site, Entry ID **8797900**.

The user program presented here uses the new block versions.

<u>The following description of communication blocks refers to a specified FDS connection with SDA.</u>

Block FC5 "AG_SEND" for data transmission

- The FC block AG_SEND is used to transfer data from one data area of the CPU to the PROFIBUS-CP of the sending station for data transmission to the receiving station via a configured connection.
- When all user data could be transmitted via PROFIBUS, faultless execution is signaled by DONE ==1.
- The addresses and TSAP of the partner stations are defined during configuration and provided with a connection ID which is then used for data transmission through AG_SEND.

FC5 „AG_SEND“

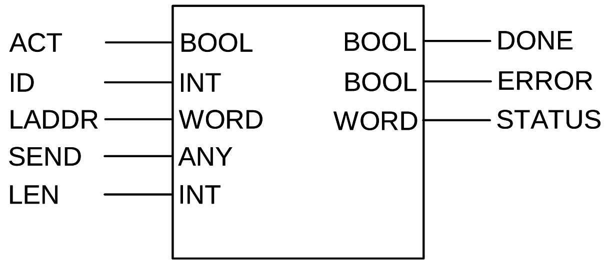


Table 3-2 Formal parameters of FC5 "AG_SEND"

Parameter	Comment
ACT	Control parameter to activate data exchange at a positive edge.
ID	Address parameter as defined in the configuration for connection of the corresponding station.
LADDR	Module start address of the CP as defined in the configuration for connection of the corresponding station.
SEND	Address for the FDL data area (send data), refers either to: • the process image area (inputs, outputs), • memory area (flag), • data block area.
LEN	Number of bytes to be transmitted by a send command.
DONE	Status parameter indicating whether the command has been executed without error [1)]
ERROR	Signals, if an error occurred during command execution. [(1]
STATUS	This parameter provides detailed information about command execution. [(1]

1) The parameters DONE, ERROR and STATUS form the block status information and are always to be considered as a whole for error analysis. Tables for status indication analysis are included in the NCM documentation for PROFIBUS or in the online help on blocks in STEP 7.

Block FC6 "AG_RECV" for data reception

- The FC block "AG_RECV" receives data transmitted from the sending station via a configured connection from the PROFIBUS-CP of the receiving station in the data area of the CPU.
- If all user data transmitted via PROFIBUS could be successfully received, faultless execution is signaled by NDR ==1.
- The addresses and TSAP of the partner stations are defined during configuration and provided with a connection ID which is then used for data reception with AG_RECV.

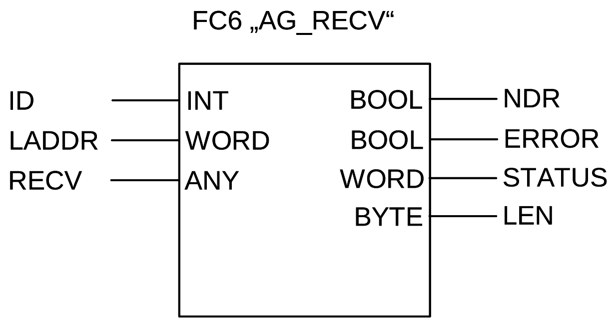


Figure 3-2 Receive block FC 5 "AG_RECV"

Table 3-3 Formal parameters of FC6 "AG_RECV"

Parameter	Comment
ID	Address parameter as defined in the configuration for connection of the corresponding station.
LADDR	Module start address of the CP as defined in the configuration for connection of the corresponding station.
RECV	Address for the FDL data area (receive data), refers either to: • the process image area (inputs, outputs), • memory area (flag), • data block area.
LEN	Number of bytes received in the FDL data area.
NDR	Status parameter indicating that new data has been received 1)
ERROR	Signals if an error has occurred during command execution. 1)
STATUS	This parameter provides detailed information on command execution. 1)

(1 The parameters NDR, ERROR and STATUS form the block status information and are always to be considered as a whole for error analysis. Tables for status indication analysis are included in the NCM documentation for PROFIBUS or in the online help on blocks in STEP 7.

3.3 Principle structure of this example

This simplified example shows how a data area of 1024 bytes can be transmitted from the CPU of the sending station to a data area of the same size in the CPU of the receiving station. Data transmission from the sending station to the receiving station is realized via a configured FDL connection.

Data flow model

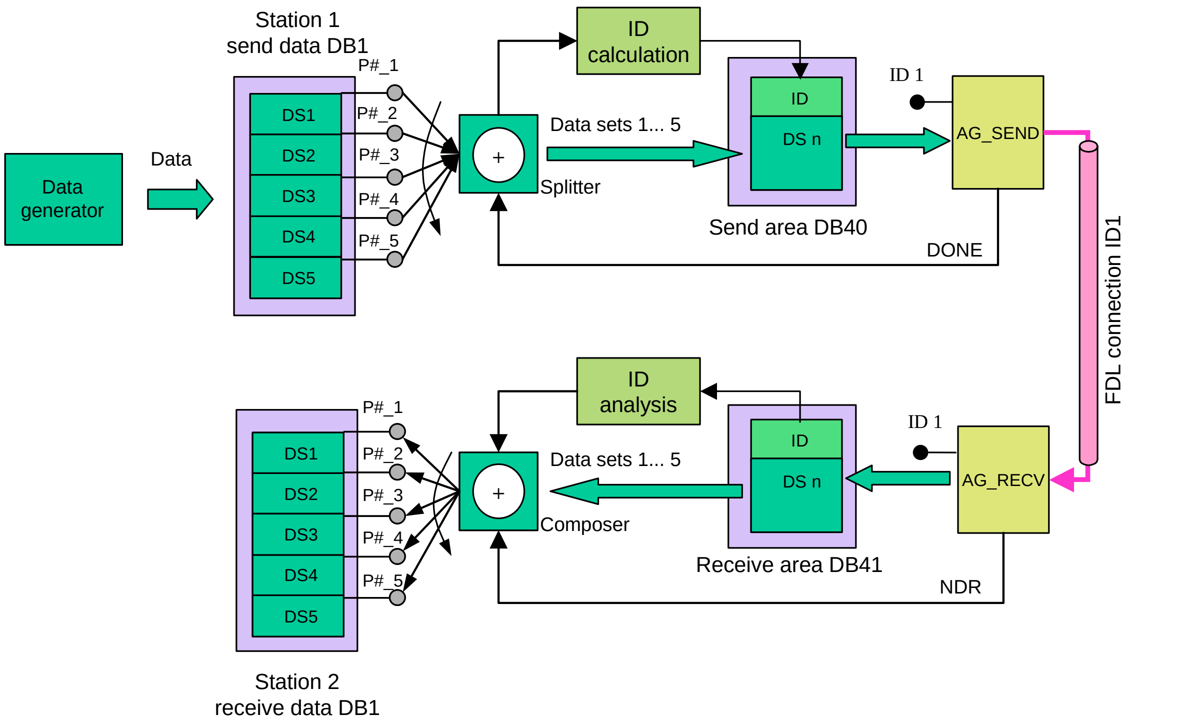


Figure 3-3 Data flow model for station 1 (sending station) and station 2 (receiving station)

Process description

The whole data area is broken down successively into 5 partial data sets of a defined size (from DS 1 to DS 5) which are then transmitted to the receiving station.

After transmission of all partial data sets, the data received shall match with the data sent.

Station 1, active (sending station "SIMATIC 300_act"):

Step	Description
1	Initiated via the table of variables, a data generator fills the send data with defined raw data.
2	The request to send for the first partial data set is triggered by the user. Each request to send initiates a consistency mechanism (splitter) which splits the data to be transmitted into partial data sets, each of which is provided with a clear ID; this ID and the partial data set form the send command.
3	Each further send command (out of 5) is started by the status parameter DONE of the previous AG_SEND block.
4	Transmission is finished when all 5 send commands of AG_SEND have been completed.

Station 2, passive (receiving station "SIMATIC 400_p")

Step	Description
1	The parameter NDR of the receive block AG_RECV is queried at cyclic intervals. After receipt of the first partial data set a flag is set that marks the start for the receipt of further partial data sets.
2	The composer evaluates the ID of each of the partial data sets received and writes the raw data read into the corresponding data area of the receiving station.
3	After receipt of the last partial data set the flag will be reset. The procedure is completed.

3.4 Program structures

The following chapter describes the setup and structure of the example program for the sending and the receiving station.

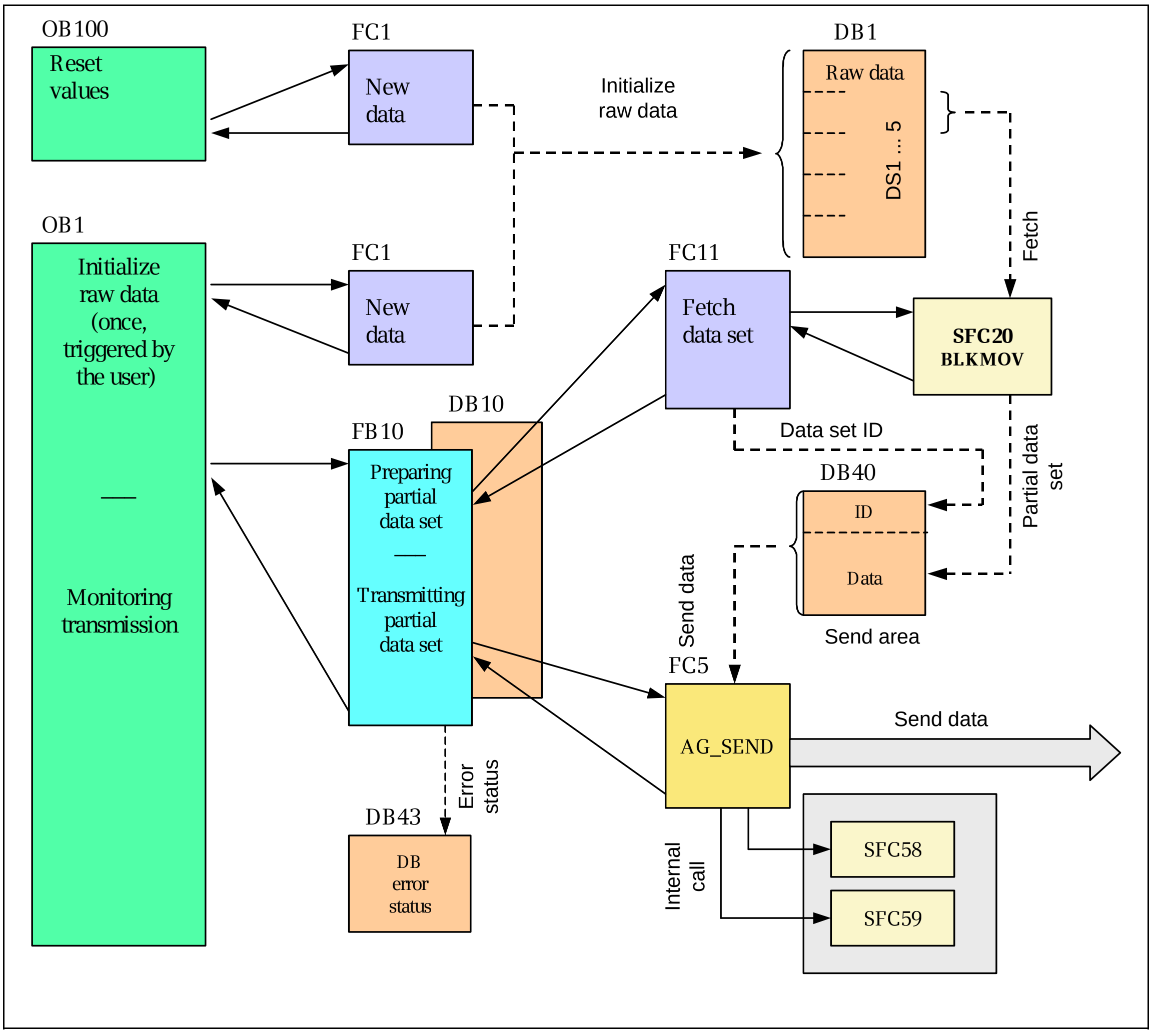
General preconditions for the user program

The user program must meet the following requirements:

- SEND / RECEIVE of data must be initiated by the active SIMATIC S7 unit.
- The communication blocks AG_SEND and AG_RECV must be called in each OB1 cycle.
- The partial data sets must be sent in their sequential order, i.e. each partial data set will be transmitted only after receipt of the previous partial data set has been acknowledged (DONE).

- The send command of AG_SEND must be active for only one cycle (ACT==1), so as to enable analysis of the AG_SEND status indications in the following cycles (ACT==0).
- Possible status values indicating errors in transmission via AG_SEND or AG_RECV shall be saved in a data block. This offers the user to read and to check the relevant error status.
- Operator control and monitoring: the transmission process shall be started via a table of variables and the procedures in the sending or receiving station shall be displayed. This requires the use of a simple user interface for the sending and the receiving station.
- The user program shall make only little use of OB1. Only block calls and user interfaces shall be included here.

3.4.1 Block structure of the sending station (S7-300, station 1)



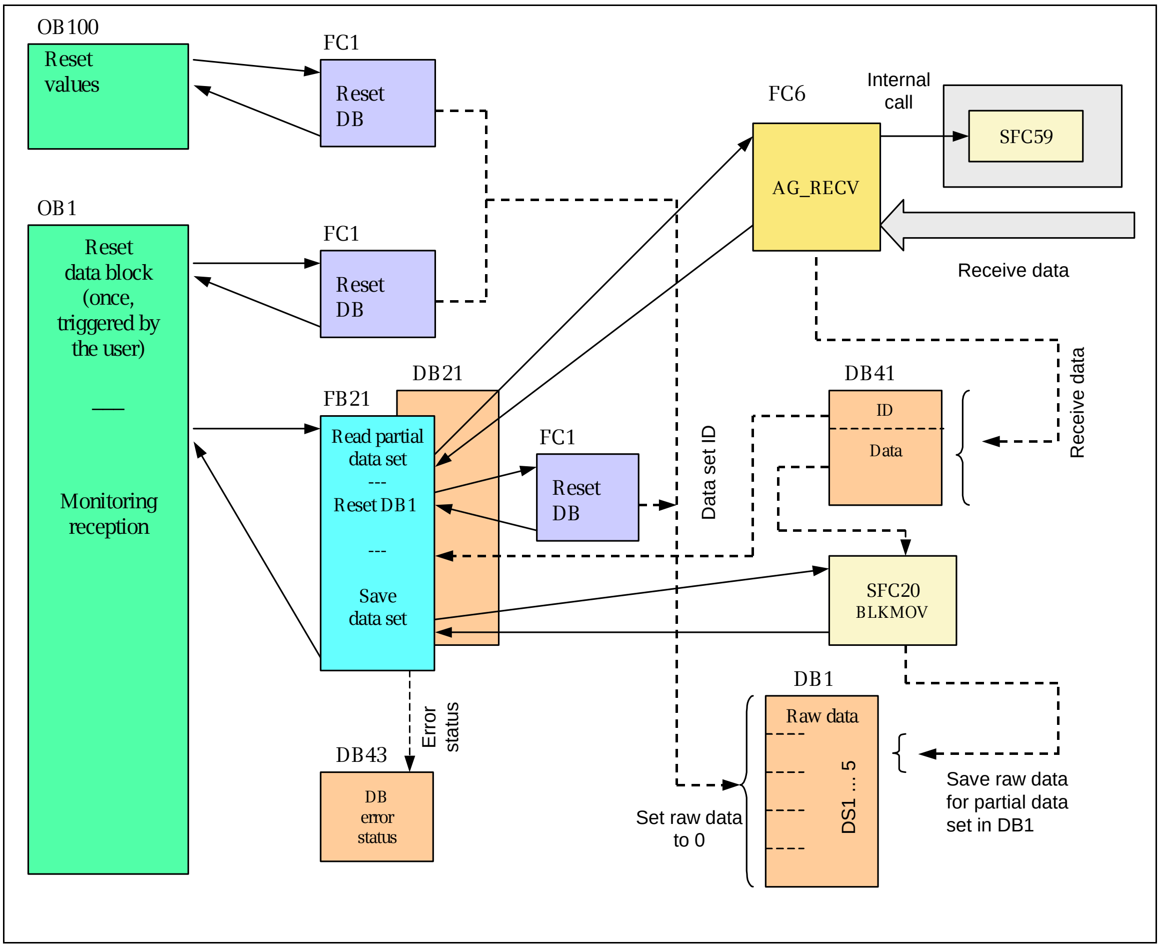
Description

The blocks in the sending station are defined as follows:

Block	Description
OB100	After start-up of the station, OB100 resets various values and control parameters used for program execution. When FC1 is called, the raw data is written to DB1 once.
OB1	• When selecting FC1 in OB1 it can be chosen whether new raw data shall be written to DB1. → Operated via VAT1. • Selection of FB10 in OB1 initiates a new transmission procedure for 1024 bytes of raw data. → Operated via VAT1.

Block	Description
FB10, IDB10	FB10 controls and monitors the whole process of transmission for all partial data sets from 1 to 5: • FC11 is used to generate the individual partial data sets. • FC5 (AG_SEND) is used to write the individual partial data sets to PROFIBUS-CP. • Possible error information of AG_SEND is saved in DB43.
FC1	FC1 writes the send data of DB1 in an ascending order of integer numbers beginning with the start value. This block is used to initialize DB1 with new data before transmission. *Note: FC1 of the sending station is not identical with FC1 of the receiving station (see receiving station).*
FC11	• After request from FB10, FC11 generates the individual partial data sets 1 to 5; this is effected by calling SFC20 (BLKMOV) to write a data area from DB1 to send buffer DB40. An error in SFC20 is saved in DB43. • In addition to the data sets, FC11 writes the data set number of the individual partial data sets (1 to 5) as ID code into the send buffer.
FC5	Initiated by FB10, FC5 (AG_SEND) transfers the send data from the send buffer DB40 to the PROFIBUS_CP. The CP is then used for further data transfer to the receiving station. This block is write-protected and can be found in the SIMATIC_NET_CP library. *Note:* Internally, FC5 calls SFC 58 (WR_REC) and SFC59 (RD_REC).
DB40	DB40 is the send buffer for the partial data set to be transmitted. This data block is suitable for a data set ID of 2 bytes and user data up to 238 bytes. The data is transmitted to station 1 (receiving station) by calling FC5 (AG_SEND).
DB43	Possible error codes are saved in DB43. • error code of FC5 (AG_SEND) (called in FB10) • error code of SFC20 (BLKMOV) (called in FB10)
DB1	Includes 1024 bytes of raw data to be transmitted.
SFC20	SFC20 "BLKMOV" (block move) is used to copy the contents of one memory area (= source area) to another memory area (= target area).
SFC58	SFC 58 "WR_REC" (write record) is used to transmit a RECORD data set to the addressed module (called indirectly via FC5).
SFC59	SFC 59 "RD_REC" (read record) is used to transmit the data set with the number RECNUM to the addressed module (called indirectly via FC5).

3.4.2 Block structure of the receiving station (S7-400, station 2)



Description

The blocks in the receiving station are defined as follows:

Block	Description
OB100	After start-up of the station, OB100 resets various values and control parameters used for program execution. In addition, calling FC1 causes a reset of DB1.
OB1	• When selecting FC1 in OB1 it can be chosen whether DB1 shall be reset before transmission. → Operated via VAT1. • Call of FB21 in OB1 with parameter "RECV_ACTIVE" which indicates that data is being received.

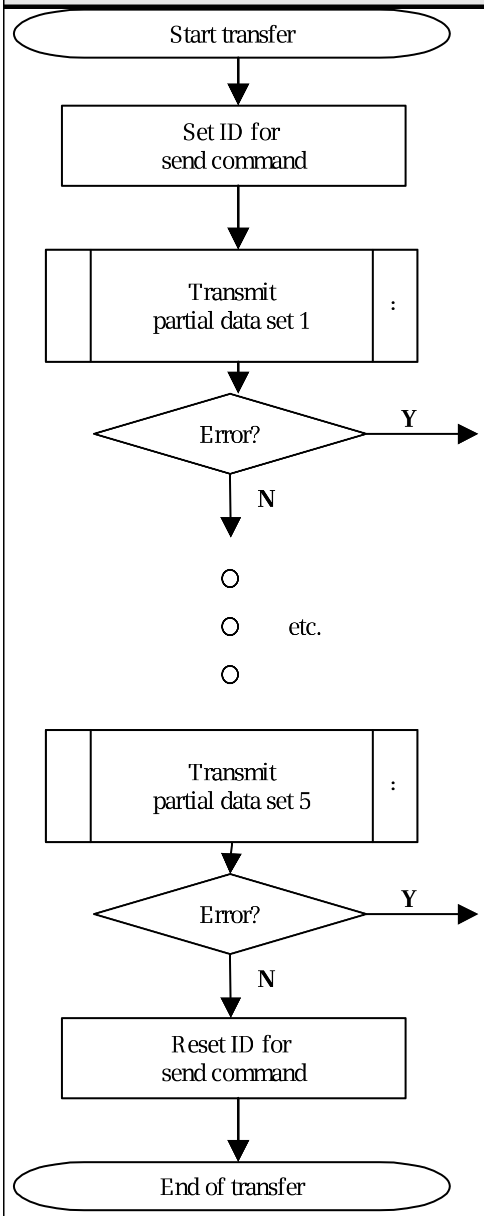
Block	Description
FB21, IDB21	FB21 controls and monitors the whole receiving process for all partial data sets 1 to 5: • When FC6 (AG_RECV) is called, FB21 reads the data set received from the PROFIBUS-CP. • FB21 is used to evaluate the data set ID and to call SFC 20 (BLKMOV) which writes the partial data set in compliance with the data set ID from the read buffer DB41 to the area reserved for the data area of DB1. An error in SFC20 will be saved in DB43. • Possible error information of FC5 is saved in DB43.
FC1	FC1 is used to reset all bytes of DB1 (1024 bytes of raw data) to 0. The following events initiate a call of FC1: Restart: call in OB100 User action (call in OB1): M40.1 "INIT_DB1_DATA" (operated via the table of variables) New data set: call in FB21 after partial data set no. 1 of a new data set has been received. *Note: FC1 of the receiving station is not identical with FC1 of the sending station (see sending station).*
FC6	Initiated by FB10, FC6 (AG_RECV) reads the receive data from the PROFIBUS_CP into the read buffer DB41.This block is write-protected and can be found in the SIMATIC_NET_CP library. *Note*: Internally, FC6 calls SFC59 (RD_REC).
DB41	DB41 is the receive buffer for the partial data set to be read. This data block is suitable for a data set ID of 2 bytes and user data up to 238 bytes. The data received are read out from PROFIBUS-CP by calling FC5 (AG_RECV).
DB43	Possible errors codes are saved in DB43. • error code of FC6 (AG_RECV) (called in FB21) • error code of SFC20 (BLKMOV) (called in FB21)
DB1	Includes 1024 bytes of raw data read.
SFC20	SFC20 "BLKMOV" (block move) is used to copy the contents of one memory area (= source area) to another memory area (= target area).
SFC59	SFC 59 "RD_REC" (read record) is used to read the data set with the number RECNUM from the addressed module (called indirectly via FC5).

3.5 Program execution in station 1 (sending station)

This chapter explains and illustrates program execution in the active station (sending station), as well as some major code segments.

Higher-level program execution

Table 3-4 Higher-level flowchart for the sending station

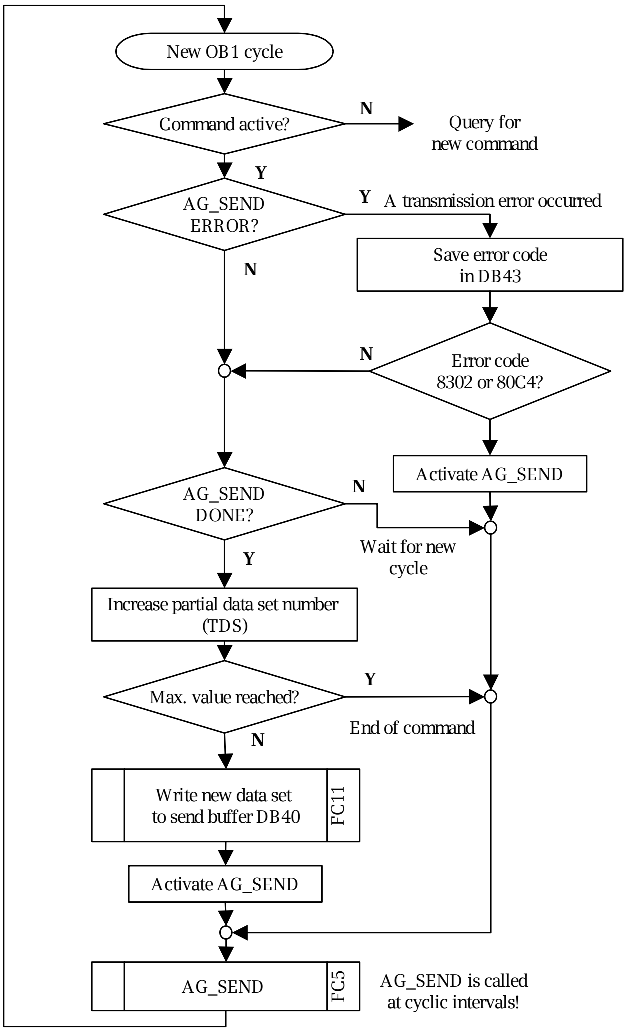


Flowchart	Description
	"TRANSM_START" in the variables table → VAT1 of the sending station is set to start transmission.
	The "TRANSM_ACTIVE" ID remains active until the transmission process is completed.
	A start pulse is used to set the trigger for transmission of the first partial data set. The data to be sent has been transmitted from the raw data area to the send buffer beforehand.
	An error identification routine checks whether an error occurred during transmission. The error code for the relevant send command is saved in a data block.
	If no transmission error occurs ...
	... all partial data sets 1 to 5 are transmitted to the receiving station.
	After successful transmission of all 5 partial data sets, the "TRANSM_ACTIVE" ID will be reset.
	Transmission is completed.

The flowchart above shows a simplified model of the procedures in the sending station for the transmission of 5 partial data sets. A detailed model for the transmission of any type of partial data set is illustrated in the next flowchart.

Detailed description of the transmission procedure

Table 3-5 Detailed flowchart of the transmission procedure

<table>
<tr><th>Flowchart</th><th>Description</th></tr>
<tr><td>


</td><td>
The transmission process is managed in FB10.
NW1: the variable #transm_active shows whether a send command is active.
NW2: the status parameters for the last call of AG_SEND are polled:
#send_error ==1 → error
If an error occurred, the error status will be saved in DB43; check for *temporary errors* (1 . If the result is positive, the previous send command will be repeated.
#send_done ==1 → as soon as the last send command has been successfully completed, the number of the next higher partial data set will be increased by 1. The highest partial data set number is 5.
NW3: FC11 is called to write the raw data for the new partial data set to the send buffer. In addition, FC11 writes the data set number as ID code to the send buffer.
NW4: Call of AG_SEND. For a send command, its ACT parameter must be set for one cycle: #activate_send ==1
Then the OB1 cycle starts anew.
</td></tr>
</table>

1) Temporary errors (e.g. h8302 or h80C4) occur sporadically and require repeat of the previous send command.

Code excerpt: calling AG_SEND (sending station, FB10, network 4)

```
      SET   ;
      S     #activate_send; //AG_SEND is hot
NETWORK
TITLE =send data sets
//"AG_SEND" is the SEND/RECEIVE interface for data transfer on FDL.
//Transfer = 240 BYTE (maximum load). All status will be saved and will be
//checked
//out in network 2 this block.
//
//NOTE:
//# For to send one data set, take the first call of AG_SEND with ACT=1.
//# For to check out the STATUS of this send order, take the following calls of
//AG_SEND with ACT=0.
SEND: NOP   0;

      CALL "AG_SEND" (
           ACT                      := #activate_send,// send is enabled/disabled
           ID                       := 1,// ID of connection
           LADDR                    := W#16#100,// logic moduladress of cp
           SEND                     := "SEND_BUFFER".send_data,// any-pointer to send buffer
           LEN                      := 240,// length of send-data (max=240 Byte)
           DONE                     := #send_done,
           ERROR                    := #send_error,
           STATUS                   := #send_status);// actual function status

      SET   ; //to check out the status of AG_SEND,
      R     #activate_send; //the next circle, ACT must be FALSE.
```

Description

The program excerpt above shows a call of AG_SEND in network 4 of FB10 in the sending station 1.

The parameter **LEN** shows the length of data to be sent which is always 240 bytes. The send data of DB40 are assigned to the parameter SEND. The values of the parameters **ID** and **LADDR** refer to the connection as configured for the relevant station. The parameters **ACT**, **DONE**, **ERROR** and **STATUS** are provided with stationary variables of FB10.

The variable #activate_send reflects the start event for the parameter ACT and will be reset immediately after a send command of AG_SEND in the same OB1 cycle. This is necessary to enable analysis of the status parameters DONE, ERROR and STATUS of AG_SEND in the subsequent program cycles by means of ACT==0.

The status parameters DONE, ERROR and STATUS are checked in network 2 of FB10 (*no program excerpt here)*. The error status (#send_status) of each error (#send_error ==1) is saved in DB43. If no error occurred ((#send_error ==0) and if the last send command was successfully processed (#send_done ==1), the send commands for the remaining partial data sets will be completed.

Masking of temporary transmission errors (no code excerpt):

Temporary transmission errors may occur even if there is no problem in connection. This type of error happens **spontaneously** during data transmission and requires a repeat of the last send command.

In this application a simple routine in network 2 of FB10 is used to identify such temporary errors which may occur quite frequently. The types of error states are masked as follows:

8302: no resources for reception in the target station

80C4: temporary communication error

These types of error will not lead to the generation of a new partial data set, but a repeat of the last send command.

Code excerpt: splitter for the formation of partial data sets (sending station, FC11, network 1)

```
BEGIN
NETWORK
TITLE =
//check out the number of data set and move raw data from DB1 to SEND_BUFFER or
//the next send order with AG_SEND

      NOP   0;
      L     #jmp_label;
      SPL   m00; //Wert >5
      SPA   m00; //Wert 0
      SPA   m01; //senden: Datensatz 1
      SPA   m02; //senden: Datensatz 2
      SPA   m03; //senden: Datensatz 3
      SPA   m04; //senden: Datensatz 4
      SPA   m05; //senden: Datensatz 5

//jump label =0 or >5: no function
m00:  NOP   0;
      BEA   ; //no function, end


//jump label = 1: fetch data set 1 and save in SEND BUFFER
m01:  NOP   0;

      L     #jmp_label;
      T     "SEND_BUFFER".send_data.number_data_set; //indicates actual number of data set

//move raw data from source into the send buffer
      CALL "BLKMOV" (
           SRCBLK                   := P#DB1.DBX0.0 BYTE 238,
           RET_VAL                  := #blkmov_error,
           DSTBLK                   := "SEND_BUFFER".send_data.raw_data);

      BEA   ;
```

Description

The formal parameter #jmp_label of FC11 is the transmission parameter for the requested partial data set number.

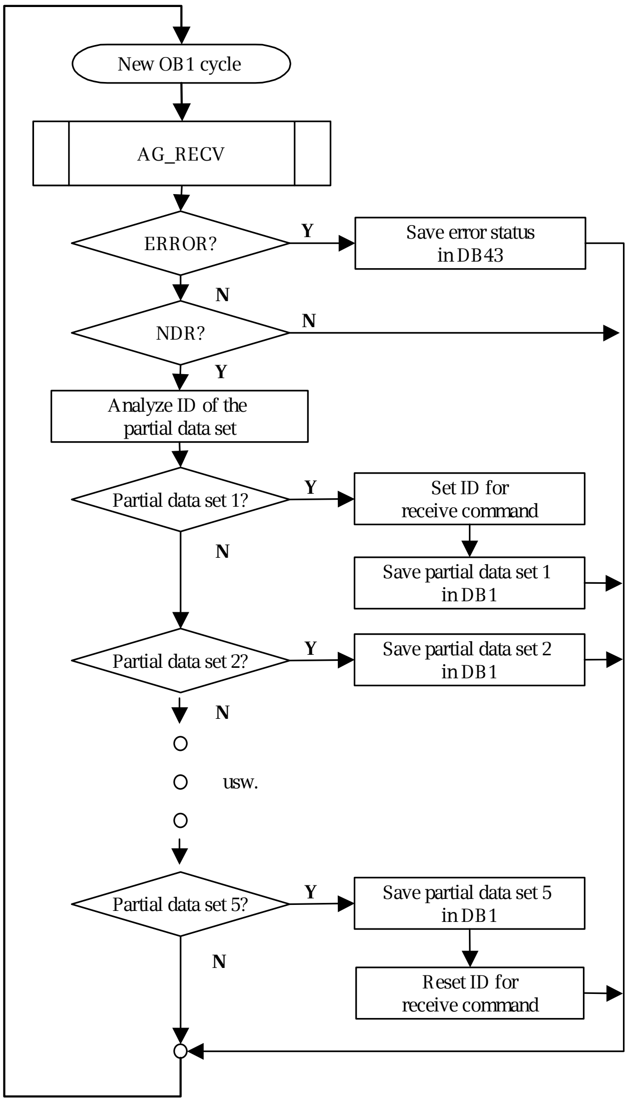
Step	
1.	A jump distributor checks the number of the partial data set #jmp_label.
2.	Depending on the variable #jmp_label the jump target is always another call of SFC20 (BLKMOV) used to transfer data from a jump target not identical with the previous one to the data area in DB1 of send buffer DB40.

The program excerpt above shows the jump target for the first partial data set (#jmp_label =1). At the beginning, the partial data set number (#jmp_label) is saved as ID for the current partial data set in a reserved area of DB40. Then SFC20 is called which moves the raw data from DB1 starting with byte 0 to a 238 bytes data area in the raw data area of DB40.

3.6 Program execution in the receiving station (station 2)

Program execution

Table 3-6 Principle of program execution in the receiving station

Flowchart	Description
	Data reception is managed in FB21. This block is called in OB1 at cyclic intervals. NW1: the procedure starts by calling AG_RECV, followed by a routine used to analyze the status information ERROR and NDR of AG_RECV: If an error occurred (ERROR ==1), the corresponding error status will be saved in DB43. If a partial data set has been received without errors (NDR ==1 and STATUS =0) the data received is stored in DB41 starting at DBW2. NW2: Here, the data set ID of each partial data set received is queried. ID "1" marks the start of a new receive command. In this case, an internal ID is set in FB21. Now, partial data set no. 1 and then all successive partial data sets "1" ... "5" are saved in an area of DB1 assigned to the specific data set depending on the individual IDs. After receipt of the last partial data set no. "5" the internal ID for a receive command in FB21 will be reset. The complete data set of 1024 bytes is read and saved in DB1. The routine waits for the next partial data set no. 1 from a new data block.

Code excerpt: calling AG_RECV (receiving station, FB21, network 1)

```
BEGIN
NETWORK
TITLE =
//in every circle AG_RECV is called for received data.
//if a data set is received, the actual (error) send status of received data will
//be saved.

      CALL "AG_RECV" (
           ID                       := 1,// id of connection
           LADDR                    := W#16#1FFD,// logic moduladress of cp
           RECV                     := "RECV_BUFFER".receive_data,// any-pointer to receive buffer
           NDR                      := #recv_ndr,
           ERROR                    := #recv_error,
           STATUS                   := #recv_status,// actual function status
           LEN                      := #recv_bufferlength);

      UN    #recv_error; // if no function error, jump
      SPB   M001;

      L     #recv_status; // save actual (error) send status
      T     "PARAMETER/ERROR_DB".receive_error_status;
      BEA   ;


M001: NOP   0;
      UN    #recv_ndr;
      BEB   ;
```

Description

The program excerpt above shows a call of AG_RECV in network 1 of FB21.

The parameter **LEN** corresponds to the data length of the data received and is saved in DB41.DBW0. The parameter **RECV** is assigned to the receive buffer of DB41 which starts from DB41.DBW2. The parameters **ID** and **LADDR** are defined in the configuration for connection of the relevant station.

The **status parameters NDR, ERROR and STATUS** are provided with stationary variables from FB21 and are analyzed in each program cycle of FB21 when AG_RECV is called.

The parameter NDR is queried in each program cycle in order to identify whether new data are to be received. If so (#recv_ndr ==1), this data will be saved in data block DB1 (not shown in the program excerpt above) with reference to the corresponding ID (partial data set number).

If an error occurs (#recv_error ==1) the error status (#recv_status) will be saved in DB43.DBW43.

Code excerpt: composer to re-combine the partial data sets received (receiving station, FB21, network 2)

```
//check out the number of received data set, part of incoming data set
//the number is the input for the next following jump target
      L     "RECV_BUFFER".receive_data.number_data_set;

      SPL   m00; //value >5
      SPA   m00; //value =0
      SPA   m01; //value =1: Datensatz 1
      SPA   m02; //value =1: Datensatz 2
      SPA   m03; //value =1: Datensatz 3
      SPA   m04; //value =1: Datensatz 4
      SPA   m05; //value =1: Datensatz 5

//jump label =0 or >5: has no function in this application
m00:  NOP   0;
      BEA   ;


//jump label =1: save data set 1
m01:  NOP   0;

//after incoming of the first data set, a global variable "RECV_ACTIVE" is set.
//this variable indicates that receive of data set 1 to 5 is active.
//this variable will be reset after receiving data set 5
      SET   ;
      S     #recv_active;

//clear DB1
      CALL "INIT_DATA" (//initialize DB1 with zeros
           init_db                   := TRUE);

      CALL "BLKMOV" (
           SRCBLK                    := "RECV_BUFFER".receive_data.raw_data,
           RET_VAL                   := #blkmov_error,
           DSTBLK                    := P#DB1.DBX0.0 BYTE 238);
      BEA   ;
```

Description

The variable "RECV_BUFFER".receive_data.number_data_set in FC11 corresponds to the ID of the partial data set received in the receive buffer. The value of the ID is identical with the number of the specific partial data set. Depending on the ID, the target of the jump distributor is always a further call of SFC20 (BLKMOV). The raw data received are copied to a data area of DB1 (receive data) different to that of the previous jump target.

The program excerpt above shows the jump target for the first partial data set ("RECV_BUFFER".receive_data.number_data_set =1). It starts with the call "INIT_DATA" which is used only with the first partial data set. All data of DB1 are set to 0. Then SFC20 is called. This block copies the raw data area of the receive buffer for partial data set no. 1 (238 bytes in size) to DB1 starting at byte 0.

4 Installation of Hardware and Software

4.1 Hardware configuration

For hardware configuration please refer to Chapter 2 "Principle of the Automation Solution".

The following MPI and PROFIBUS addresses are assigned to the project:

Station, module	MPI	PROFIBUS
Station 1, CPU 315-2DP	4	--
Station 1, CP 342-5	5	3
Station 2, CPU 414-2 DP	2	--
Station 2, CP 443-5 BASIC	--	2

For hardware installation, proceed as follows:

No.	Focus	Action
1	S7 Hardware	Mount the S7 400 unit to the rack as described in the installation guidelines.
2	S7 Hardware	Mount the S7 300 unit to the rack rail as described in the installation guidelines.
3	Bus cable connection	Install a fully prepared Profibus cable between the CP 342-5 and the CP 443-5 Profibus interfaces and screw-fix the connectors on the sockets.
4	Bus cable connection	Connect the MPI interfaces of both automation systems and the programming interface of the PG with a fully prepared Profibus cable.
5	Bus cable connection	Make sure that the cabling end points of both bus systems are secluded at both ends of the bus system using the matching resistors in the connectors.

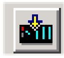
4.2 Software installation

STEP7

At this point, we will not go further into the installation of STEP7. the installation is self-explanatory and performed in the usual Windows environment.

Installing and loading the STEP7 project

Proceed as follows:

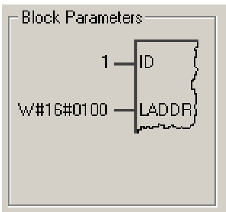
No.	Action	Note / Explanation
1	Open the STEP7 Manager.	
2	Select the menu command File > Retrieve... to extract the project zip file **"20987711_FDL_SDA_CODE_v10.zip"**	Use the browser to search the relevant project and click OK to confirm.
3	Select a directory where the project files shall be stored after extraction.	At the end of the retrieving process you will be asked whether you want to open the project with Step 7. Click **Yes** to confirm.
4	After you have opened the project, the SIMATIC Manager shows a project named "FDL_Getstart_03". Open the SIMATIC Manager project tree for this project.	The project tree is located on the left side of the SIMATIC Manager and in the top line the name of the project is shown.
5	Select the station S7 400 first and click the icon  to download.	This function is also available via Ctrl + L or by using the menu commands "PLC" -> "Download".
6	Now select the station S7 300 and click the icon  again to download.	This function can also be used by pressing Ctrl + L or by using the menu commands "PLC" -> "Download".

4.3 Configuration for connection

The application includes the complete configuration for connection. If you have performed the steps described in the previous chapter, the sending and receiving stations are already configured for connection by loading the hardware configuration and the program modules.

If you use a **different type of hardware structure** than described in Chapter 2 "Principle of the Automation Solution", you may perform the following steps **as an option**.

No.	Action	Note / Result
1	Switch all CPUs to STOP. Open the STEP7 project "FDL_Getstart_03". Select "File" -> "Save As..." to save the project.	
2	Change the hardware configuration for the S7 400 station. The changed hardware configuration must then be saved and compiled anew.	The CP parameters are defined in compliance with the example application.
3	Change the hardware configuration for the S7 300 station. The changed hardware configuration must be saved and compiled anew.	The CP parameters are defined in compliance with the example application.
4	Close the hardware configuration for both stations and open the connection configuration by clicking the project name in the SIMATIC Manager and then by double clicking the PROFIBUS symbol in the right window.	The connection configuration NetPro is opened.

No.	Action	Note / Result
5	Create a new connection between S7-300 and S7-400 in NetPro. Click the CPU in station S7-300. The table of connections is shown at the bottom of the NetPro window.	This connections table can be used to create a new connection or to delete or change previously configured connections.
6	Click an empty line in the table of connections and select the menu commands "Insert" ->"New Connection".	The dialog box "Insert New Connection" appears.
7	In the "Insert New Connection" dialog box select "FDL connection" under "Connection" > "Type" and click OK to confirm your selection. The selection box "Display properties before inserting" should remain checkmarked.	A property sheet appears that shows the properties of the new FDL connection.
8	Accept the settings for FDL communication shown in this dialog box. Click OK to close the dialog box.	Now you have defined a new connection.
9	Delete all previous connections used for the example application.	Only one connection for FDL communication should be defined.
10	Save and compile the network configuration and load the configuration into the stations.	Connection is properly configured.
11	Open the property sheet for FDL connection again for each station and select the block parameters ID and LADDR for both stations as shown in the illustration below.  Enter these parameters into the call command for AG_SEND in FB10 of the S7-300 and into the call command for AG_RECV in FB21 of the S7-400.	These parameters are used to identify the relevant connection. These parameters can be accessed only if the connection has been properly configured.
12	Load the changed program blocks into the stations and perform a configuration test.	Use the variable table as described in Chapter 4.4.

5 Operator Control and Monitoring

Introduction

Control of the example application is realized with the help of variable tables (VAT) for

- the active station 1 (sending station "SIMATIC 300_act"), VAT1
- the passive station 2 (receiving station "SIMATIC 400_p"), VAT1.

Activating the variable tables

The variable tables can be opened and activated as follows:

Table 5-1 Activating the variable tables

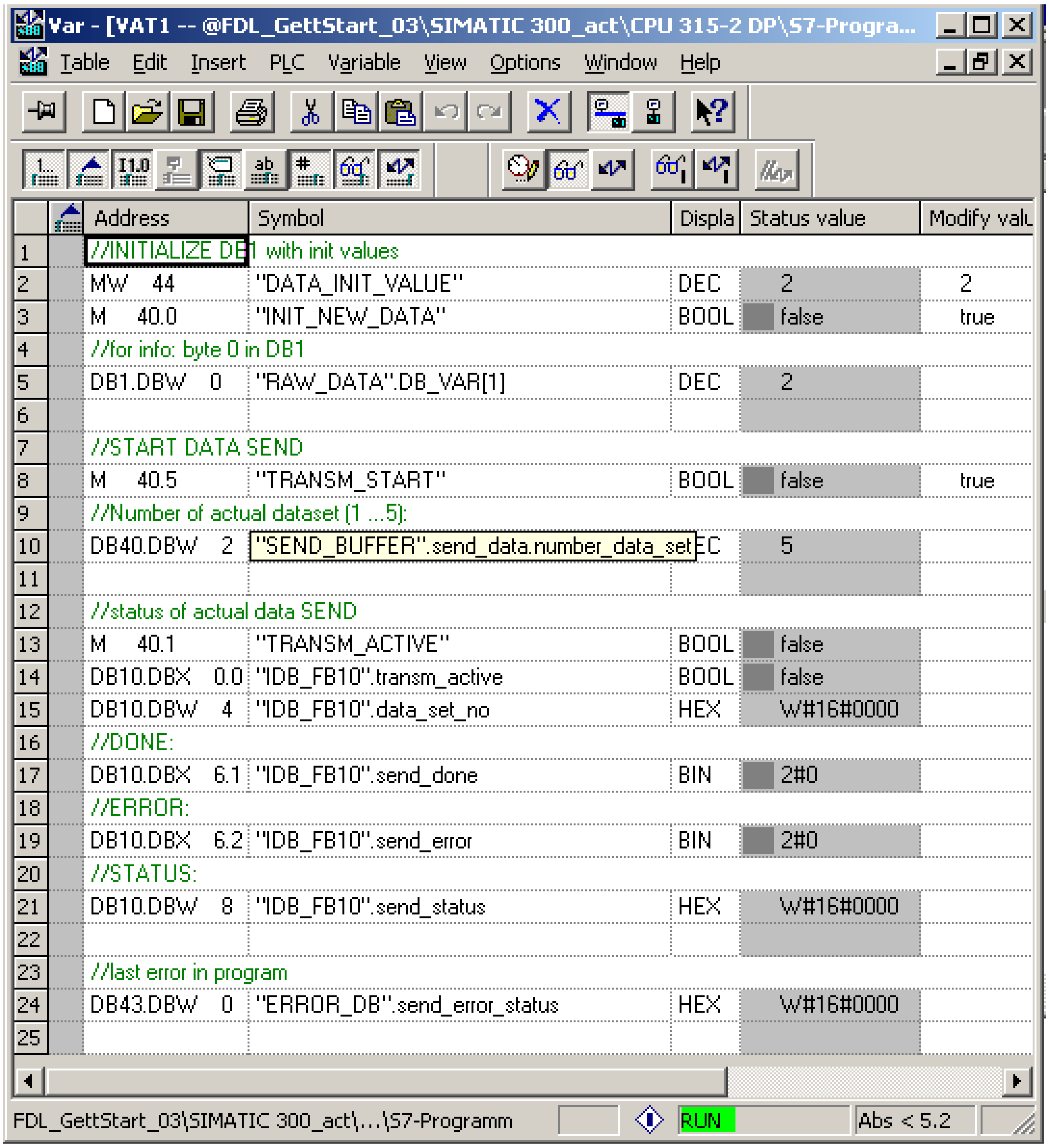
No.	Description	Remark
1	Connect the MPI cable of your PG with the MPI interface of any CPU.	The MPI cable is connected with a CPU.
2	Mark the folder "Blocks" of the station „SIMATIC 300_act“ in the SIMATIC Manager and open the variable table "VAT1".	
3	Mark the folder "Blocks" of the station "SIMATIC 400_p" in the SIMATIC Manager and open the variable table "VAT1".	Both variable tables are open.
4	Mark the opened variable table for the station "SIMATIC 300_act" and select "Window > Arrange > Horizontally" from the main menu of the variable tables.	The window on the left shows a variable table for the active station, the window section on the right shows a variable table for the passive station.
6	Select the menu command "Variable > Trigger" to open the "Trigger" dialog box. Choose the trigger point and the trigger conditions for operator control and monitoring: Monitoring: end of scan cycle, every cycle Modifying: beginning of scan cycle, once	
5	To enable monitoring of the defined variables, select the menu command "Variable" -> "Monitor" for each variable table.	 As an alternative, the button "Monitor variable" may be used.

Note

The monitoring function is active if the progress indicator in the status bar of the "Monitor variable" window is active (CPU in RUN).

5.1 Variable table for the sending station (S7-300, station 1)

The screenshot below shows the variable table for the sending station (status after transmission of 5 data sets while waiting for a new request to send).



	Address	Symbol	Displa	Status value	Modify valu
1	//INITIALIZE DB1 with init values				
2	MW 44	"DATA_INIT_VALUE"	DEC	2	2
3	M 40.0	"INIT_NEW_DATA"	BOOL	false	true
4	//for info: byte 0 in DB1				
5	DB1.DBW 0	"RAW_DATA".DB_VAR[1]	DEC	2	
6					
7	//START DATA SEND				
8	M 40.5	"TRANSM_START"	BOOL	false	true
9	//Number of actual dataset (1 ...5):				
10	DB40.DBW 2	"SEND_BUFFER".send_data.number_data_set	EC	5	
11					
12	//status of actual data SEND				
13	M 40.1	"TRANSM_ACTIVE"	BOOL	false	
14	DB10.DBX 0.0	"IDB_FB10".transm_active	BOOL	false	
15	DB10.DBW 4	"IDB_FB10".data_set_no	HEX	W#16#0000	
16	//DONE:				
17	DB10.DBX 6.1	"IDB_FB10".send_done	BIN	2#0	
18	//ERROR:				
19	DB10.DBX 6.2	"IDB_FB10".send_error	BIN	2#0	
20	//STATUS:				
21	DB10.DBW 8	"IDB_FB10".send_status	HEX	W#16#0000	
22					
23	//last error in program				
24	DB43.DBW 0	"ERROR_DB".send_error_status	HEX	W#16#0000	
25					

Operator control and monitoring with the variable table for the sending station:

Area	Comment
//INITIALIZE DB1:	Operator control: initializing DB1 (raw data)

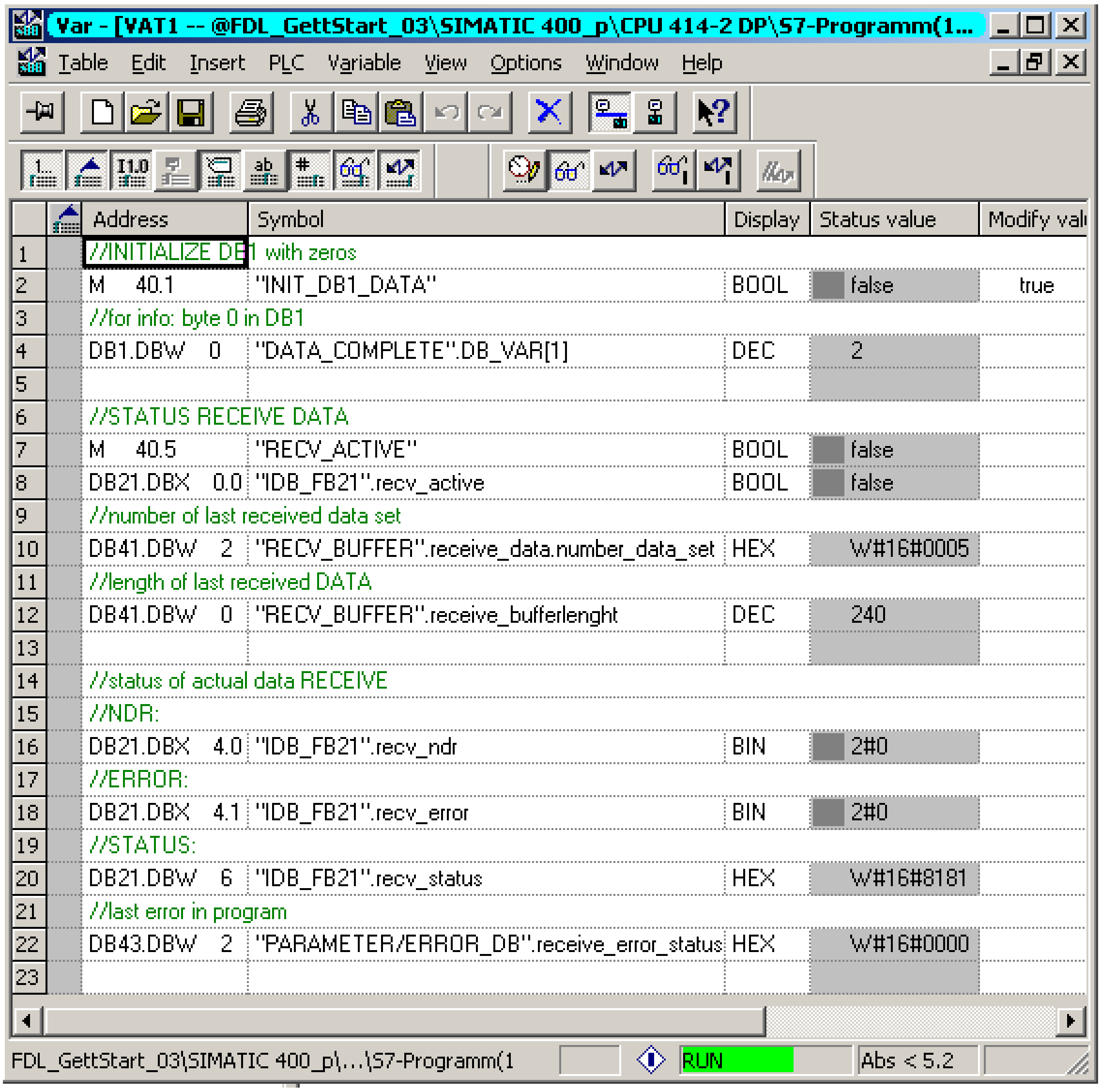
Area	Comment
	Available as an option, e.g. before a new send command
//for info: byte 0 in DB1	Shows a new section with current raw data from DB1.DBW0
//START DATA SEND	Operator control: start event for the transmission of 1024 bytes of raw data to station 2
//number of actual dataset (1 ...5):	The partial data set currently being sent to station 2.
//status of actual data SEND	Parameters and variables during transmission: • general status variables of FB10 for the active send process: • status parameters of AG_SEND: (done, error, status): changed during the send process.
//last error in program	The last error status of AG_SEND.

Information provided in the variable table of the sending station:

Address	Symbol	Function
M 40.1	"TRANSM_ACTIVE"	Remains set during the whole transmission process. After transmission of the last partial data set, this variable will be reset automatically.
DB40.DBW 2	"SEND_BUFFER".send_data.number_data_set	Changed from 1 to 5 in an ascending order depending on the partial data set presently transmitted. After completion of the whole transmission process the status value must be "5".
DB10.DBX 6.1	"IDB_FB10".send_done	Status parameter of AG_SEND
DB10.DBX 6.2	"IDB_FB10".send_error	Status parameter of AG_SEND
DB10.DBW 8	"IDB_FB10".send_status	Status parameter of AG_SEND
DB43.DBW 0	"ERROR_DB".send_error_status	Shows the last error status of AG_SEND. → analysis in the online help on blocks in STEP 7

5.2 Variable table in the receiving station (S7-400, station 2)

The screenshot below shows the variable table for the receiving station (status after reception of 5 partial data sets, data package completed, waiting for the receipt of new data).



Var - [VAT1 -- @FDL_GettStart_03\SIMATIC 400_p\CPU 414-2 DP\S7-Programm(1...

	Address	Symbol	Display	Status value	Modify value
1	//INITIALIZE DB1 with zeros				
2	M 40.1	"INIT_DB1_DATA"	BOOL	false	true
3	//for info: byte 0 in DB1				
4	DB1.DBW 0	"DATA_COMPLETE".DB_VAR[1]	DEC	2	
5					
6	//STATUS RECEIVE DATA				
7	M 40.5	"RECV_ACTIVE"	BOOL	false	
8	DB21.DBX 0.0	"IDB_FB21".recv_active	BOOL	false	
9	//number of last received data set				
10	DB41.DBW 2	"RECV_BUFFER".receive_data.number_data_set	HEX	W#16#0005	
11	//length of last received DATA				
12	DB41.DBW 0	"RECV_BUFFER".receive_bufferlenght	DEC	240	
13					
14	//status of actual data RECEIVE				
15	//NDR:				
16	DB21.DBX 4.0	"IDB_FB21".recv_ndr	BIN	2#0	
17	//ERROR:				
18	DB21.DBX 4.1	"IDB_FB21".recv_error	BIN	2#0	
19	//STATUS:				
20	DB21.DBW 6	"IDB_FB21".recv_status	HEX	W#16#8181	
21	//last error in program				
22	DB43.DBW 2	"PARAMETER/ERROR_DB".receive_error_status	HEX	W#16#0000	
23					

FDL_GettStart_03\SIMATIC 400_p\...\S7-Programm(1 RUN Abs < 5.2

Operator control and monitoring with the variable table for the receiving station:

Area	Comment
//INITIALIZE DB1:	Initializing DB1 (raw data reset to 0)
//for info: byte 0 in DB1	Shows a section with current raw data from DB1.DBW0
//STATUS	General status variables for the data reception

Area	Comment
RECEIVE DATA	
//number of last received data set	Number of the last partial data set received.
//length of last received DATA	Length of the last partial data set received (corresponds to parameter LEN of AG_RECV)
//status of actual data RECEIVE	Status information of AG_RECV (done, error , status)
//last error in program	Last error status of AG_RECV received.

Information provided in the variable table of the receiving station:

Address	Symbol	Function
M 40.5	"RECV_ACTIVE"	Remains set during the whole transmission process. After receipt of the last partial data set, this variable will be reset automatically.
DB41.DBW 2	"RECV_BUFFER".receive_data.number_data_set	Changed from 1 to 5 in an ascending order depending on the partial data set received. After completion of the whole transmission process the status value must be "5".
DB21.DBX 4.0	"IDB_FB21".recv_ndr	Status parameter of AG_RECV
DB21.DBX 4.1	"IDB_FB21".recv_error	Status parameter of AG_RECV
DB10.DBW 8	"IDB_FB21".recv_status	Status parameter of AG_RECV
DB43.DBW 2	"PARAMETER/ERROR_DB".receive_error_status	Shows the last error status of AG_RECV. Normally this value is "0". → analysis in the online help on blocks in STEP 7

5.3 Starting data transmission

The list of steps below describes how you can start transmission of 1024 bytes from the sending to the receiving station.

Note

- ! Please don't forget! The variable table must be "online" and "**Monitoring" must be activated** to ensure that all values will be updated.

No.	Action	Note
Initialize DB1 of the sending station with new data		
1	Enter a value between 0 and 100 in the last column ("Modify value") in line "MW 44" "DATA_INIT_VALUE" of the variable table for the **sending station**.	This is the start value to initialize DB1 with new data.
2	Mark this line and select "Variable" -> "Activate Modify Values".	The modify value is taken over as status value.
3	Mark the line M 40.0 "INIT_NEW_DATA" and select "Variable" -> "Modify Address to 0".	If set: the status value will be reset (to prepare a positive edge for step 4).
4	Keep this line marked and select "Variable" -> "Activate Modify Values". The positive edge is used to initialize DB1 anew with the start value defined under step 1.	A new status value is set.
5	Check the result in line "DB1.DBW0" "RAW_DATA".DB_VAR[1].	The start value as defined in step 1 is shown.
Set the values of DB1 of the receiving station to 0 (as an option)		
6	Mark the line "M 40.1" "INIT_DB1_DATA" in the variable table for the **receiving station** and select "Variable" -> "Activate Modify Values".	The status value is set.
7	Keep this line marked and select "Variable" -> "Modify Address to 0".	The status value is reset.
8	Check the result in line "DB1.DBW0" "DATA_COMPLETE".DB_VAR[1]".	The status value "0" is shown.
Send 1024 data bytes from DB1 of the sending station to DB1 of the receiving station		
9	Mark the line M 40.5 "TRANSM_START" and select "Variable" -> "Activate Modify Values".	Starts transmission of 1024 bytes of raw data, broken down into 5 partial data sets.
10	The transmission process will be finished automatically, when all 5 partial data sets have been received in the receiving station.	

For exact details on the status values for AG_SEND and AG_RECV please refer to the online help in STEP7.

Open the SIMATIC Manager, select the relevant block folder for this station and mark the corresponding communication block. Press F1 to open the online help for this block. When the window for "Help on FBs, CS for SIMATIC NET CPs" is open, click the link > FDL connections to open a further subwindow with a description of the status values "Meaning of this block" under > (see also) under > Analysis of display.

6 Summary

The Getting Started project for FDL communication with SDA described here is characterized by the following properties:

- The communication blocks AG_SEND and AG_RECV of the SEND / RECEIVE interface are called for data transmission.
- The procedures initiated via the communication blocks AG_SEND and AG_RECV ensure consistent data transmission.
- A maximum data length of 240 bytes is used for the communication command for communication blocks AG_SEND and AG_RECV via PROFIBUS.
- The individual partial data sets are consistently transmitted via the communications protocol for FDS communication.
- Since 1000 bytes shall be transmitted, the whole area is divided up into 5 individual data sets to provide a simplified consistency mechanism in the sending station; each of these partial data sets is provided with an ID and the partial data sets are then transmitted successively to the receiving station.
- A **simplified** consistency mechanism in the receiving station saves the read partial data sets into a separate area within a data block with reference to their specific IDs.

This Getting Started example application reflects the requirements for reliable transmission of the whole data area only **to a limited extent**. The following example shall explain how errors in transmission may happen:

Based on the program structure, the receiving station "knows" that transmission starts with partial data set 1 and ends with partial data set 5. A full consistency mechanism which checks the sequential order and the number of partial data sets as well as the data length is not realized in this example. If, for example, data set no. 5 is missing, transmission to the receiving station cannot be completed. The sending station will continue to send new data sets, since there is no possibility to acknowledge receipt of data. The new data sets received will overwrite the old data sets in the receiving station. This means that data transmission cannot be performed properly, since the consistency mechanism is insufficient.

7 Further Literature

Further details on FDL communication, function, structure and technical problems can be found in the following manuals:

- "Communication with SIMATIC" Manual (see Product Support, ID entry: 1254686).
- "SIMATIC NET NCM S7 for PROFIBUS" Manual, Volume 1 (includes important general information and notes on configuration for FDL communication, see Product Support, ID entry: 1158693)
- "System Software for S7-300/400 System and Standard Functions" Manual (see Product Support, ID entry: 1214574)