SIEMENS

SIMATIC

Open TCP/IP Communication via Industrial Ethernet

Manual

Edition 12/2005
A5E00711636-01

Safety Guidelines

This manual contains notices you have to observe in order to ensure your personal safety, as well as to prevent damage to property. The notices referring to your personal safety are highlighted in the manual by a safety alert symbol, notices referring to property damage only have no safety alert symbol. The notices shown below are graded according to the degree of danger.

Danger
indicates that death or severe personal injury **will** result if proper precautions are not taken.

Warning
indicates that death or severe personal injury **may** result if proper precautions are not taken.

Caution
with a safety alert symbol indicates that minor personal injury can result if proper precautions are not taken.

Caution
without a safety alert symbol indicates that property damage can result if proper precautions are not taken.

Notice
indicates that an unintended result or situation can occur if the corresponding notice is not taken into account.

If more than one degree of danger is present, the warning notice representing the highest degree of danger will be used. A notice warning of injury to persons with a safety alert symbol may also include a warning relating to property damage.

Qualified Personnel

The device/system may only be set up and used in conjunction with this documentation. Commissioning and operation of a device/system may only be performed by **qualified personnel.** Within the context of the safety notices in this documentation qualified persons are defined as persons who are authorized to commission, ground and label devices, systems and circuits in accordance with established safety practices and standards.

Prescribed Usage

Note the following:

Warning
This device and its components may only be used for the applications described in the catalog or the technical description, and only in connection with devices or components from other manufacturers which have been approved or recommended by Siemens.
Correct, reliable operation of the product requires proper transport, storage, positioning and assembly as well as careful operation and maintenance.

Trademarks

All names identified by ® are registered trademarks of the Siemens AG.
The remaining trademarks in this publication may be trademarks whose use by third parties for their own purposes could violate the rights of the owner.

Disclaimer of Liability

We have reviewed the contents of this publication to ensure consistency with the hardware and software described. Since variance cannot be precluded entirely, we cannot guarantee full consistency. However, the information in this publication is reviewed regularly and any necessary corrections are included in subsequent editions.

Siemens AG
Automation and Drives
Postfach 4848
90437 NÜRNBERG
GERMANY

A5E00711636-01
12/2005

Copyright © Siemens AG 2005
Technical data subject to change

Contents

1 Open TCP/IP Communication via Industrial Ethernet

1.1 Overview

Open Communication via Industrial Ethernet

STEP 7 provides the following FBs and UDTs in the "Standard Library" located under "Communications Blocks" for exchanging data through the user program with other Ethernet-capable communications partners:

- Connection-oriented protocols: TCP native as per RFC 793, ISO on TCP as per RFC 1006:
  - UDT 65 "TCON_PAR" with the data structure for assigning connection parameters
  - FB 65 "TCON" for establishing a connection
  - FB 66 "TDISCON" for terminating a connection
  - FB 63 "TSEND" for sending data
  - FB 64 "TRCV" for receiving data
- Connectionless protocol: UDP as per RFC 768
  - UDT 65 "TCON_PAR" with the data structure for assigning parameters for the local communications access point
  - UDT 66 "TCON_ADR" with the data structure for assigning addressing parameters for the remote partner
  - FB 65 "TCON" for configuring the local communications access point
  - FB 66 "TDISCON" for closing the local communications access point
  - FB 67 "TUSEND" for sending data
  - FB 68 "TURCV" for receiving data

1.2 Function of FBs for Open Communication via Industrial Ethernet

Connection-oriented and Connectionless Protocols

The following types of protocols are distinguished in the data communication:

- Connection-oriented protocols:
  These establish a logical connection to the communication partner before data transmission is started. After the data transmission is complete, they then terminate the connection, if necessary. Connection-oriented protocols are used for data transmission when reliable, guaranteed delivery is of particular importance. In general, many logical connections can exist on one physical line.

  The following connection-oriented protocols are supported with FBs for open communication via Industrial Ethernet:

  - TCP/IP native as per RFC 793 (with connection types B#16#01 and B#16#11)
  - ISO on TCP as per RFC 1006 (with connection type B#16#12)

- Connectionless protocols:
  These work without a connection. There is thus no establishment and termination of a connection with a remote partner. Connectionless protocols transmit data unacknowledged, with no reliable, guaranteed delivery to the remote partner.

  The following connection-oriented protocols are supported with FBs for open communication via Industrial Ethernet: UDP as per RFC 768 (with connection type B#16#13)

How the function blocks actually function depends on the protocol variant being used. This is discussed in detail in the following section.

TCP native

During data transmission, no information about the length or about the start and end of a message is transmitted. This is not a problem during sending because the sender knows how many data bytes it will be sending. However, the receiver has no means of detecting where one message ends in the data stream and the next one begins. For this reason, it is recommended that the LEN parameter of FB 64 "TRCV" (number of bytes to be received) be assigned the same value as the LEN parameter of FB 63"TSEND" for the communication partner (number of bytes to be sent).

If you have specified the length of the data to be received (LEN parameter of FB 64 "TRCV") to be greater than the length of the data to be sent, FB 64 "TRCV" will only copy the received data into the receiver area (DATA parameter) after the length specified by the parameter value has been reached. This occurs only after the data from a following job have been received. Please note that in this case data from two different send jobs will be located in one and the same receiver area. If you do not know the exact length of the first message, you will have no way of detecting the end of the first message or the start of the second one.

If you have specified the length of the data to be received (DATA parameter of FB 64 "TRCV") to be less than the length of the sent data, FB 64 will copy as many bytes into the receiver range as you have specified in the LEN parameter. After this, it will set NDR to TRUE and write RCVD_LEN with the value of LEN. With each additional call, you will thus receive another block of sent data.

ISO on TCP

During data transmission, information on the length and the end of the message is also transmitted.

If you have specified the length of the data to be received (LEN parameter of FB 64 "TRCV") to be greater than the length of the data to be sent, FB 64 "TRCV" will copy the received data completely into the receiver range. After this, it will set NDR to TRUE and write RCVD_LEN with the length of the sent data.

If you have specified the length of the data to be received (DATA parameter of FB 64 "TRCV") to be less than the length of the sent data, FB 64 will not copy any data into the receiver range but instead will supply the following error information: ERROR=1, STATUS=W#16#8088.

UDP

Unlike with the TCP native and ISO on TCP protocols, with UDP you do not establish a connection. In this case, when calling the sending block FB 67 "TUSEND" you have to specify the address parameters of the receiver (IP address and port number). Similarly, after the conclusion of the receiving block FB 68 "TURCV", you will receive a reference to the address parameters of the sender (IP address and port number).

In order to be able to use the FBs 67 "TUSEND" and 68 "TURCV", you first have to call the FB 65 "TCON" on both the sending side and the receiving side. This step is necessary to configure the local communications access point.

With each new call of FB 67 "TUSEND", you re-reference the remote partner by specifying its IP address and its port number.

During data transmission, information on the length and the end of the message is also transmitted.

If you have specified the length of the data to be received (LEN parameter of FB 68 "TURCV") to be greater than the length of the data to be sent, FB 68 "TURCV" will copy the received data completely into the receiver range. After this, it will set NDR to TRUE and write RCVD_LEN with the length of the sent data.

If you have specified the length of the data to be received (DATA parameter of FB 68 "TURCV") to be less than the length of the sent data, FB 68 will not copy any data into the receiver range but instead will supply the following error information: ERROR = 1, STATUS = W#16#8088.

1.3 Assigning Parameters for Communications Connections with FCP native and ISO on TCP

Data Block for Assigning Parameters

To be able to assign parameters for TCP native and ISO on TSP for communications connections, create a DB that contains the data structure from the UDT 65 "TCON_PAR". This data structure contains the parameters necessary for configuring the connection. You will need such a data structure for every connection. You can assemble this structure in a global DB.

The CONNECT connection parameter address of FB 65 "TCON" contains a reference to the associated connection description (e.g. P#DB100.DBX0.0 byte 64).

Structure of the Connection Description (UDT 65)

Byte	Parameter	Data Type	Start Value	Description
0 to 1	block_length	WORD	W#16#40	Length of UDT 65: 64 Bytes (fixed)
2 to 3	id	WORD	W#16#0000	Reference to the connection (range of values: W#16#0001 to W#16#0FFF) You must specify the value of the parameter in the respective block with the ID.
4	connection_type	BYTE	B#16#01	Connection type: • B#16#11: TCP/IP native • B#16#12: ISO on TCP • B#16#01: TCP/IP native (compatibility mode)
5	active_est	BOOL	FALSE	ID for the way the connection is established: • FALSE: passive establishment • TRUE: active establishment
6	local_device_id	BYTE	B#16#02	• B#16#00: communication via CP • B#16#02: communication via the integrated IE interface for CPUs 315-2 PN/DP and 317-2 PN/DP • B#16#03: communication via the integrated IE interface for CPU 319-3 PN/DP • No. of the configured IE interface with WinAC RTX 2005 (possible values: B#16#01 to B#16#04)
7	local_tsap_id_len	BYTE	B#16#02	Length of parameter local_tsap_id used; possible values: • 0 or 2, if connection type = B#16#01 or B#16#11 For the active side, only the value B#16#00 is permitted. • 2 to 16, if connection type = B#16#12
8	rem_subnet_id_len	BYTE	B#16#00	This parameter is currently not used. You must assign B#16#00 to it.

Byte	Parameter	Data Type	Start Value	Description
9	rem_staddr_len	BYTE	B#16#00	Length of address for the remote connection transmission point: • 0: unspecified, i.e. parameter rem_staddr is irrelevant. • 4: valid IP address in the parameter rem_staddr
10	rem_tsap_id_len	BYTE	B#16#00	Length of parameter local_tsap_id used; possible values: • 0 or 2, if connection type = B#16#01 or B#16#11 For the passive side, only the value B#16#00 is permitted. • 2 to 16, if connection type = B#16#12
11	next_staddr_len	BYTE	B#16#00	Length of parameter next_staddr used
12 to 27	local_tsap_id	ARRAY [1..16] of BYTE	B#16#00 ...	With connection_type = • B#16#11: local port no. (possible values: 2000 to 5000), local_tsap_id[1] = high byte of port no. in hexadecimal representation, local_tsap_id[2] = low byte of port no. in hexadecimal representation, local_tsap_id[3-16] = irrelevant • B#16#12: local TSAP ID: local_tsap_id[1] = B#16#E0 (connection type T-connection), local_tsap_id[2] = Rack and slot in own CPU (bits 0 to 4 slot, bits 5 to 7: rack number), local_tsap_id[3-16] = TSAP extension • B#16#01: local port no. (possible values: 2000 to 5000), local_tsap_id[1] = low byte of port no. (in hexadecimal representation, local_tsap_id[2] = high byte of port no. (in hexadecimal representation, local_tsap_id[3-16] = irrelevant Note: Make sure that each value of local_tsap_id that you use in your CPU is unique.
28 to 33	rem_subnet_id	ARRAY [1..6] of BYTE	B#16#00 ...	This parameter is currently not used. You must assign 0 to it.

Byte	Parameter	Data Type	Start Value	Description
34 to 39	rem_staddr	ARRAY [1..6] of BYTE	B#16#00 ...	IP address for the remote connection transmission point:, e.g. 192.168.002.003: With connection_type = • B#16#1x: rem_staddr[1] = B#16#C0 (192), rem_staddr[2] = B#16#A8 (168), rem_staddr[3] = B#16#02 (002), rem_staddr[4] = B#16#03 (003), rem_staddr[5-6]= irrelevant • B#16#01: rem_staddr[1] = B#16#03 (003), rem_staddr[2] = B#16#02 (002), rem_staddr[3] = B#16#A8 (168), rem_staddr[4] = B#16#C0 (192), rem_staddr[5-6]= irrelevant
40 to 55	rem_tsap_id	ARRAY [1..16] of BYTE	B#16#00 ...	With connection_type = • B#16#11: remote port no. (possible values: 2000 to 5000), rem_tsap_id[1] = high byte of port no. (in hexadecimal representation, rem_tsap_id[2] = low byte of port no. (in hexadecimal representation, rem_tsap_id[3-16] = irrelevant • B#16#12: remote TSAP ID: rem_tsap_id[1] = B#16#E0 (connection type T-connection), rem_tsap_id[2] = Rack and slot for the remote connection transmission point (CPU) (bits 0 to 4: slot, bits 5 to 7: rack number), rem_tsap_id[3-16] = TSAP extension • B#16#01: remote port no. (possible values: 2000 to 5000), local_tsap_id[1] = low byte of port no. (in hexadecimal representation, local_tsap_id[2] = high byte of port no. (in hexadecimal representation, local_tsap_id[3-16] = irrelevant
56 to 61	next_staddr	ARRAY [1..6] of BYTE	B#16#00 ...	With local_device_id = • B#16#00: next_staddr[1]: Rack and slot of associated (local) CP (bits 0 to 4: slot, bits 5 to 7: rack number) next_staddr[2-6]: irrelevant • B#16#02: next_staddr[1-6]: irrelevant
62 to 63	spare	WORD	W#16#0000	irrelevant

CPU Dependencies for Connection Types

The following list specifies which connection type you can use for which CPU:

- connection_type=B#16#11 (TCP native): CPUs 31x-2 PN/DP as of firmware version V2.4, WinAC RTX as of V4.2 (WinAC RTX 2005)
- connection_type=B#16#12 (ISO on TCP): CPUs 31x-2 PN/DP as firmware version V2.4, S7-400 CPUs (without CPU 414-4H and CPU 417-4H) as of firmware version V4.1
- connection_type=B#16#01 (TCP native, compatibility mode): all CPUs 31x-2 PN/DP, WinAC RTX as of V4.2 (WinAC RTX 2005)

For information on the number of possible connections, please refer to the technical data for your CPU.

Establishing a Connection

The establishment of an active connection must be initiated by a communications partner A. The establishment of a passive connection must be initiated by a communications partner B. If both communications partners have initiated the establishment of a connection, the operating system can completely establish a connection.

In the parameters for the connection, you specify which communications partner activates the establishment of a connection and which establishes a passive connection upon request of the communications partner

See also:

Examples of Parameters for Communications Connections

1.4 Assigning Parameters for the Local Communications Access Point with UDP

Data Block for Assigning Parameters for the Local Communications Access Point

To assign parameters for the local communications access point, create a DB that contains the data structure from the UDT 65 "TCON_PAR". This data structure contains the parameters necessary for configuring the connection between the user program and the communications level of the operating system.

The CONNECT parameter of FB 65 "TCON" contains a reference to the address of the associated connection description (e.g. P#DB100.DBX0.0 byte 64).

Structure of the Connection Description (UDT 65)

Byte	Parameter	Data Type	Start Value	Description
0 to 1	block_length	WORD	W#16#40	Length of UDT 65: 64 Bytes (fixed)
2 to 3	id	WORD	W#16#0000	Reference to this connection between the user program and the communications level of the operating system (range of values: W#16#0001 to W#16#0FFF) You must specify the value of the parameter in the respective block with the ID.
4	connection_type	BYTE	B#16#01	Connection type: • B#16#13: UDP
5	active_est	BOOL	FALSE	ID for the way the connection is established: You must assign FALSE to this parameter since the communications access point can be used to both send and receive data.
6	local_device_id	BYTE	B#16#02	• B#16#02: communication via the integrated IE-interface for CPU 317-2 PN/DP • B#16#03: communication via the integrated IE-interface for CPU 319-3 PN/DP
7	local_tsap_id_len	BYTE	B#16#02	Length of parameter local_tsap_id used; possible value: 2
8	rem_subnet_id_len	BYTE	B#16#00	This parameter is currently not used. You must assign B#16#00 to it.
9	rem_staddr_len	BYTE	B#16#00	This parameter is currently not used. You must assign B#16#00 to it.
10	rem_tsap_id_len	BYTE	B#16#00	This parameter is currently not used. You must assign B#16#00 to it.
11	next_staddr_len	BYTE	B#16#00	This parameter is currently not used. You must assign B#16#00 to it.

Byte	Parameter	Data Type	Start Value	Description
12 to 27	local_tsap_id	ARRAY [1..16] of BYTE	B#16#00 ...	• Local port no. (possible values: 2000 to 5000), local_tsap_id[1] = high byte of port no. in hexadecimal representation, local_tsap_id[2] = low byte of port no. in hexadecimal representation, local_tsap_id[3-16] = irrelevant Note: Make sure that each value of local_tsap_id that you use in your CPU is unique.
28 to 33	rem_subnet_id	ARRAY [1..6] of BYTE	B#16#00 ...	This parameter is currently not used. You must assign 0 to it.
34 to 39	rem_staddr	ARRAY [1..6] of BYTE	B#16#00 ...	This parameter is currently not used. You must assign 0 to it.
40 to 55	rem_tsap_id	ARRAY [1..16] of BYTE	B#16#00 ...	This parameter is currently not used. You must assign 0 to it.
56 to 61	next_staddr	ARRAY [1..6] of BYTE	B#16#00 ...	This parameter is currently not used. You must assign 0 to it.
62 to 63	spare	WORD	W#16#0000	irrelevant

CPU Dependencies for UDP Connection Types

The UDP connection type (connection_type=B#16#13) exists for CPUs 31x-2 PN/DP as of firmware version V2.4.

For information on the number of possible connections between the user program and the communications level of the operating system, please refer to the technical data for your CPU.

Configuring the local communications access point

Each communications partner must configure its local communications point independently of the other partner. This pertains to establishing the connection between the user program and communications level of the operating system.

See also:

Examples of Parameters for Communications Connections

1.5 Structure of the Address Information for the Remote Partner with UDP

Overview

With FB 67 "TUSEND", at the parameter ADDR you transfer the address of the receiver. This address information must have structure specified below.

With FB 68 "TURCV", in the parameter ADDR you get the address of the sender of the data that were received. This address information must have structure specified below.

Data Block for the Address Information of the Remote Partner

You have to create an DB that contains one or more data structures as per UDT 66 "TADDR_PAR".

In parameter ADDR of FB 67 "TUSEND" you transfer and in parameter ADDR of FB 68 "TURCV" you receive a pointer to the address of the associated remote partner (e.g. P#DB100.DBX0.0 byte 8).

Structure of the Address Information for the Remote Partner (UDT 66)

Byte	Parameter	Data Type	Start Value	Description
0 to 3	rem_ip_addr	ARRAY [1..4] of BYTE	B#16#00 ...	IP address of the remote partner, e.g. 192.168.002.003: • rem_ip_addr[1] = B#16#C0 (192) • rem_ip_addr[2] = B#16#A8 (168) • rem_ip_addr[3] = B#16#02 (002) • rem_ip_addr[4] = B#16#03 (003)
4 to 5	rem_port_nr	ARRAY [1..2] of BYTE	B#16#00 ...	remote port no. (possible values: 2000 to 5000) • rem_port_nr[1] = high byte of port no. in hexadecimal representation • rem_port_nr[2] = low byte of port no. in hexadecimal representation
6 to 7	spare	ARRAY [1..2] of BYTE	B#16#00 ...	irrelevant

See also:

Examples of Parameters for Communications Connections

1.6 Examples of Parameters for Communications Connections

Example 1: Two S7-400-CPUs via CP 443-1 Adv.

Both communications partners are two CPUs 414-2 with firmware version V4.1.0. The communication occurs via two CPs 443-1 Adv. with firmware version V2.2.

The following table shows the most important data for both communications partners:

Property	**Communications Partner A: CPU 414-2 (FW V4.1.0) with CP 443-1 Adv. (FW V2.2)**	**Communications Partner B: CPU 414-2 (FW V4.1.0) with CP 443-1 Adv. (FW V2.2)**
Establish connection	Active	Passive
IP address	192.168.4.14	192.168.4.16
Physical address of CPU	Rack 0, Slot 3	Rack 0, Slot 4
Physical address of associated CP	Rack 0, Slot 6	Rack 1, Slot 8
Local TSAP-ID (Note: the coding of the actual TSAP to distinguish the connection occurs as of the third byte)	0xE0 03 54 43 50 2D 31	0xE0 04 54 43 50 2D 31

The following table shows the parameter entries in the DB relevant for active establishment of a connection by communications partner A:

Parameter	Data - Type	Value in Example	Description
id	WORD	W#16#0414	Reference to this connection
connection_type	BYTE	B#16#12	Connection type: ISO on TCP
active_est	BOOL	TRUE	Active connection establishment
local_device_id	BYTE	B#16#00	Communication AS-internal via CP
local_tsap_id_len	BYTE	B#16#07	Length of parameter local_tsap_id used
rem_staddr_len	BYTE	B#16#04	Length of address for the remote connection transmission point: • 4: valid IP address in parameter rem_staddr
rem_tsap_id_len	BYTE	B#16#07	Length of parameter rem_tsap_id used
next_staddr_len	BYTE	B#16#01	Length of parameter next_staddr used
local_tsap_id	ARRAY [1..16] of BYTE	• local_tsap_id[1] = B#16#E0 • local_tsap_id[2] = B#16#03 • local_tsap_id[3] = B#16#54 (ASCII equivalent of "T") • local_tsap_id[4] = B#16#43 (ASCII equivalent of "C") • local_tsap_id[5] = B#16#50 (ASCII equivalent of "P") • local_tsap_id[6] = B#16#2D (ASCII equivalent of "-") • local_tsap_id[7] = B#16#31 (ASCII equivalent of "1") • local_tsap_id[8-16] = irrelevant	Local TSAP-ID: 0xE0035443502D31
rem_staddr	ARRAY [1..6] of BYTE	"192.168.4.16" • rem_staddr[1] = B#16#C0 (192) • rem_staddr[2] = B#16#A8 (168) • rem_staddr[3] = B#16#04 (4) • rem_staddr[4] = B#16#10 (16) • rem_staddr[5-6] = irrelevant	IP address of the remote connection transmission point

Parameter	Data - Type	Value in Example	Description
rem_tsap_id	ARRAY [1..16] of BYTE	• rem_tsap_id[1] = B#16#E0 • rem_tsap_id[2] = B#16#04 • rem_tsap_id[3] = B#16#54 (ASCII equivalent of "T") • rem_tsap_id[4] = B#16#43 (ASCII equivalent of "C") • rem_tsap_id[5] = B#16#50 (ASCII equivalent of "P") • rem_tsap_id[6] = B#16#2D (ASCII equivalent of "-") • rem_tsap_id[7] = B#16#31 (ASCII equivalent of "1") • rem_tsap_id[8-16] = irrelevant	Remote TSAP-ID: 0xE0045443502D31
next_staddr	ARRAY [1..6] of BYTE	• next_staddr[1] = B#16#06 • next_staddr[2-6] = irrelevant	Rack = 0, slot = 6 (bits 7 to 5: rack no., bits 4 to 0: slot no.)

The following table shows the parameter entries in the DB relevant for passive establishment of a connection by communications partner B:

Parameter	Data - Type	Value in Example	Description
id	WORD	W#16#0416	Reference to this connection
connection_type	BYTE	B#16#12	Connection type: ISO on TCP
active_est	BOOL	FALSE	Passive connection establishment
local_device_id	BYTE	B#16#00	Communication AS-internal via CP
local_tsap_id_len	BYTE	B#16#07	Length of parameter used local_tsap_id
rem_staddr_len	BYTE	B#16#04	Length of address for the remote connection transmission point: • 4: valid IP address in parameter rem_staddr
rem_tsap_id_len	BYTE	B#16#07	Length of parameter used rem_tsap_id
next_staddr_len	BYTE	B#16#01	Length of parameter used next_staddr
local_tsap_id	ARRAY [1..16] of BYTE	• local_tsap_id[1] = B#16#E0 • local_tsap_id[2] = B#16#04 • local_tsap_id[3] = B#16#54 (ASCII equivalent of "T") • local_tsap_id[4] = B#16#43 (ASCII equivalent of "C") • local_tsap_id[5] = B#16#50 (ASCII equivalent of "P") • local_tsap_id[6] = B#16#2D (ASCII equivalent of "-") • local_tsap_id[7] = B#16#31 (ASCII equivalent of "1") • local_tsap_id[8-16] = irrelevant	Local TSAP-ID: 0xE0045443502D31
rem_staddr	ARRAY [1..6] of BYTE	"192.168.4.14" • rem_staddr[1] = B#16#C0 (192) • rem_staddr[2] = B#16#A8 (168) • rem_staddr[3] = B#16#04 (4) • rem_staddr[4] = B#16#0E (14) • rem_staddr[5-6] = irrelevant	IP address of the remote connection transmission point

Parameter	Data - Type	Value in Example	Description
rem_tsap_id	ARRAY [1..16] of BYTE	• rem_tsap_id[1] = B#16#E0 • rem_tsap_id[2] = B#16#03 • rem_tsap_id[3] = B#16#54 (ASCII equivalent of "T") • rem_tsap_id[4] = B#16#43 (ASCII equivalent of "C") • rem_tsap_id[5] = B#16#50 (ASCII equivalent of "P") • rem_tsap_id[6] = B#16#2D (ASCII equivalent of "-") • rem_tsap_id[7] = B#16#31 (ASCII equivalent of "1") • rem_tsap_id[8-16] = irrelevant	Remote TSAP-ID: 0xE0035443502D31
next_staddr	ARRAY [1..6] of BYTE	• next_staddr[1] = B#16#28 • next_staddr[2-6] = irrelevant	Rack = 1, lot = 8 (bits 7 to 5: rack no., bits 4 to 0: slot no.)

Example 2: Two S7-300 CPUs with integrated PROFINET interface

Both communications partners are two CPUs 319-3 PN/DP with firmware version V2.4.0. The following table shows the most important data for both communications partners:

Property	Communications Partner A: CPU 319-3 PN/DP (FW V2.4.0)	Communications Partner B: CPU 319-3 PN/DP (FW V2.4.0)
Establish connection	Active	Passive
IP address	192.168.3.142	192.168.3.125
Local port no.	irrelevant	2005

The following table shows the parameter entries in the DB relevant for active establishment of a connection by communications partner A:

Parameter	Data - Type	Value in Example	Description
id	WORD	W#16#0014	Reference to this connection
connection_type	BYTE	B#16#11	Connection type: TCP/IP native
active_est	BOOL	TRUE	Active connection establishment
local_device_id	BYTE	B#16#02	Communication via the integrated Ethernet interface
local_tsap_id_len	BYTE	B#16#00 (only this value is possible)	Parameter local_tsap_id is not used
rem_staddr_len	BYTE	B#16#04	Length of address for the remote connection transmission point: • 4: valid IP address in parameter rem_staddr
rem_tsap_id_len	BYTE	B#16#02 (only this value is possible)	Length of parameter rem_tsap_id used
rem_staddr	ARRAY [1..6] of BYTE	"192.168.3.125" • rem_staddr[1] = B#16#C0 (192) • rem_staddr[2] = B#16#A8 (168) • rem_staddr[3] = B#16#03 (3) • rem_staddr[4] = B#16#7D (125) • rem_staddr[5-6] = irrelevant	IP address of the remote connection transmission point
rem_tsap_id	ARRAY [1..16] of BYTE	"2005" • rem_tsap_id[1] = B#16#07 • rem_tsap_id[2] = B#16#D5 • rem_tsap_id[3-16] = irrelevant	Remote port no.: 2005 = W#16#07D5

The following table shows the parameter entries in the DB relevant for passive establishment of a connection by communications partner B:

Parameter	Data - Type	Value in Example	Description
id	WORD	W#16#000F	Reference to this connection
connection_type	BYTE	B#16#11	Connection type: TCP/IP native
active_est	BOOL	FALSE	Passive connection establishment
local_device_id	BYTE	B#16#02	Communication via the integrated Ethernet interface
local_tsap_id_len	BYTE	B#16#02 (only this value is possible)	Length of parameter local_tsap_id used
rem_staddr_len	BYTE	B#16#04	Length of address for the remote connection transmission point: • 4: valid IP address in parameter rem_staddr
rem_tsap_id_len	BYTE	B#16#00 (only this value is possible)	Length of parameter rem_tsap_id used
local_tsap_id	ARRAY [1..16] of BYTE	"2005" • local_tsap_id[1] = B#16#07 • local_tsap_id[2] = B#16#D5 • local_tsap_id[3-16] = irrelevant	Local port no.: 2005 = W#16#07D5
rem_staddr	ARRAY [1..6] of BYTE	"192.168.3.142" • rem_staddr[1] = B#16#C0 (192) • rem_staddr[2] = B#16#A8 (168) • rem_staddr[3] = B#16#03 (3) • rem_staddr[4] = B#16#8E (142) • rem_staddr[5-6] = irrelevant	IP address of the remote connection transmission point

Example 3: Two S7-300 CPUs with integrated PROFINET interface (example for communication via UDP)

Both communications partners are two CPUs 319-3 PN/DP with firmware version V2.4.0. The following table shows the most important data for both communications partners:

Property	Communications Partner A: CPU 319-3 PN/DP (FW V2.4.0)	Communications Partner B: CPU 319-3 PN/DP (FW V2.4.0)
Sender/receiver	Sender	Receiver
IP address	192.168.3.142	192.168.3.125
Local port no.	2004	2005

The following table shows the parameter entries in the DB relevant for the sender (communications partner A) for assigning parameters to the local communications access point:

Parameter	Data - Type	Value in Example	Description
id	WORD	W#16#0014	Reference to this connection between the application program and the communication level of the operating system.
connection_type	BYTE	B#16#13	Connection type: UDP
active_est	BOOL	FALSE	Only this value can be used with the connection type UDP.
local_device_id	BYTE	B#16#03	Communication via the integrated Ethernet interface
local_tsap_id_len	BYTE	B#16#02	Length of local_tsap_id parameter used
local_tsap_id	ARRAY [1..16] of BYTE	• local_tsap_id[1] = B#16#07 • local_tsap_id[2] = B#16#D4 • local_tsap_id[3-16] = irrelevant	Remote port no.: 2004 = W#16#07D4

The following table shows the parameter entries in the DB relevant for the receiver (communications partner B) for assigning parameters to the local communications access point:

Parameter	Data - Type	Value in Example	Description
id	WORD	W#16#000F	Reference to this connection between the application program and the communication level of the operating system.
connection_type	BYTE	B#16#13	Connection type: UDP
active_est	BOOL	FALSE	Only this value can be used with the connection type UDP.
local_device_id	BYTE	B#16#03	Communication via the integrated Ethernet interface
local_tsap_id_len	BYTE	B#16#02	Length of parameter local_tsap_id used
local_tsap_id	ARRAY [1..16] of BYTE	• local_tsap_id[1] = B#16#07 • local_tsap_id[2] = B#16#D5 • local_tsap_id[3-16] = irrelevant	Remote port no.: 2005 = W#16#07D5

When FB 67 "TUSEND" is called at the sender, you transfer the following address parameters for the receiver to a DB:

Parameter	Data - Type	Value in Example	Description
rem_ip_addr	ARRAY [1..4] of BYTE	• rem_ip_addr[1] = B#16#C0 (192) • rem_ip_addr[2] = B#16#A8 (168) • rem_ip_addr[3] = B#16#3 (3) • rem_ip_addr[4] = B#16#7D (125)	IP address of the receiver: 192.168.3.125
rem_port_nr	ARRAY [1..2] of BYTE	• rem_port_nr[1] = B#16#07 • rem_port_nr[2] = B#16#D5	Port no. of the receiver: 2005 = W#16#07D5

When FB 68 "TURCV" is called at the receiver, you receive the following address parameters for the sender in the DB:

Parameter	Data - Type	Value in Example	Description
rem_ip_addr	ARRAY [1..4] of BYTE	• rem_ip_addr[1] = B#16#C0 (192) • rem_ip_addr[2] = B#16#A8 (168) • rem_ip_addr[3] = B#16#3 (3) • rem_ip_addr[4] = B#16#8E (142)	IP address of the sender: 192.168.3.142
rem_port_nr	ARRAY [1..2] of BYTE	• rem_port_nr[1] = B#16#07 • rem_port_nr[2] = B#16#D4	Port no. of the sender: 2004 = W#16#07D4

1.7 Establishing a Connection with FB 65 "TCON"

Use with TCP native and ISO on TCP

Both communications partners call FB 65 "TCON" to establish the communications connection. In the parameters you specify which partner is the active communications transmission point and which is the passive one. For information on the number of possible connections, please refer to the technical data for your CPU.

After the connection is established, it is automatically monitored and maintained by the CPU.

If the connection is interrupted, such as due a line break or due to the remote communications partner, the active partner attempts to reestablish the connection. In this case, you do not have to call FB 65 "TCON" again.

An existing connection is terminated when FB 66 "TDISCON" is called or when the CPU has gone into STOP mode. To reestablish the connection, you will have to call FB 65 "TCON" again.

Use with UDP

Both communications partner call FB 65 "TCON" in order to configure their local communications access point. A connection is configured between the user program and the communications level of the operating system. No connection is established to the remote partner.

The local access point is used to send and receive UDP message frames.

Function

FB 65 "TCON" is an asynchronously functioning FB, which means that its processing extends over several FB calls. To start establishing a connection, call FB 65 with REQ = 1.

The job status is indicated at the output parameters RET_VAL and BUSY. STATUS corresponds to the RET_VAL output parameter of asynchronously functioning SFCs.

The following table shows the relationships between BUSY, DONE and ERROR. Using this table, you can determine the current status of FB 65 or when the establishment of the connection is complete.

BUSY	DONE	ERROR	Description
TRUE	irrelevant	irrelevant	The job is being processed.
FALSE	TRUE	FALSE	The job was completed successfully.
FALSE	FALSE	TRUE	The job was ended with an error. The cause of the error can be found in the STATUS parameter.
FALSE	FALSE	FALSE	The FB was not assigned a (new) job.

Parameters

Parameter	Declaration	Data Type	Memory Area	Description
REQ	INPUT	BOOL	I, Q, M, D, L	Control parameter REQUEST, initiates establishing the connection at rising edge.
ID	INPUT	WORD	M, D, constant	Reference to the connection to be established to the remote partner or between the user program and the communications level of the operating system. ID must be identical to the associated parameter ID in the local connection description. Range of values: W#16#0001 to W#16#0FFF
DONE	OUTPUT	BOOL	I, Q, M, D, L	DONE status parameter: • 0: Job not yet started or still running. • 1: Job executed without error.
BUSY	OUTPUT	BOOL	I, Q, M, D, L	• BUSY = 1: Job is not yet completed. • BUSY = 0: Job is completed.
ERROR	OUTPUT	BOOL	I, Q, M, D, L	ERROR status parameter: • ERROR=1: Error occurred during processing. STATUS provides detailed information on the type of error
STATUS	OUTPUT	WORD	M, D	STATUS status parameter: Error information
CONNECT	IN_OUT	ANY	D	Pointer to the associated connection description (UDT 65), see Assigning Parameters for Open Communications Connections with TCP native and ISO on TCP and Assigning Parameters for the Local Communications Access Point with UDP

Error Information

ERROR	STATUS (W#16#...)	Explanation
0	0000	Connection was able to be established
0	7000	Call with REQ=0, establishment of connection not initiated
0	7001	First call with REQ=1, connection being established
0	7002	Follow-on call (REQ irrelevant), connection being established
1	8086	The ID parameter must not have value of zero.
0	8087	Maximal number of connections reached; no additional connection possible
1	809B	The local_device_id in the connection description does not match the target CPU.
1	80A3	Attempt being made to re-establish an existing connection
1	80A7	Communications error: you have called TDISCON before TCON was complete. TDISCON must first complexly terminate the connection referenced by the ID.
1	80B3	Inconsistent parameters: • Error in the connection description • Local port (parameter local_tsap_id)is already present in another connection description • ID in the connection description different from the ID specified as parameter
1	80B4	When using the protocol variant ISO on TCP (connection_type = B#16#12) for passive establishment of a connection (active_est = FALSE), you violated one or both of the following conditions: "local_tsap_id_len >= B#16#02" and/or "local_tsap_id[1] = B#16#E0".
1	80C3	Temporary lack of resources in the CPU.
1	80C4	Temporary communications error: • The connection cannot be established at this time. • The interface is receiving new parameters.
1	8722	CONNECT parameter: Source area invalid: area does not exist in DB
1	8732	CONNECT parameter: The DB number lies outside the CPU-specific number range.
1	873A	CONNECT parameter: Access to connection description not possible (e.g. DB not available)
1	877F	CONNECT parameter: Internal error such as an invalid ANY reference

See also:

Assigning Parameters for Open Communications Connections with TCP native and ISO on TCP

Assigning Parameters for the Local Communications Access Point with UDP

Terminating a Connection with FB 66 "TDISCON"

Sending Data via TCP native and ISO on TCP with FB 63 "TSEND"

Receiving Data via TCP native and ISO on TCP with FB 64 "TRCV"

Sending Data via UDP with FB 67 "TUSEND"

Receiving Data via UDP with FB 68 "TURCV"

1.8 Terminating a Connection with FB 66 "TDISCON"

Use with TCP native and ISO on TCP

FB 66 "TDISCON" terminates a communications connection from the CPU to a communications partner.

Use with UDP

The FB 66 "TDISCON" closes the local communications access point. The connection between the user program and the communications level of the operating system is terminated.

Function

FB 66 "TDISCON" is an asynchronously functioning FB, which means that its processing extends over several FB calls. To start terminating a connection, call FB 66 with REQ = 1.

After FB 66 "TDISCON" has been successfully called, the ID specified for FB 65 "TCON" is no longer valid and thus cannot be used for sending or receiving.

The job status is indicated at the output parameters RET_VAL and BUSY. STATUS corresponds to the RET_VAL output parameter of asynchronously functioning SFCs.

The following table shows the relationships between BUSY, DONE and ERROR. Using this table, you can determine the current status of FB 66 or when the establishment of the connection is complete.

BUSY	DONE	ERROR	Description
TRUE	irrelevant	irrelevant	The job is being processed.
FALSE	TRUE	FALSE	The job was completed successfully.
FALSE	FALSE	TRUE	The job was ended with an error. The cause of the error can be found in the STATUS parameter.
FALSE	FALSE	FALSE	The FB was not assigned a (new) job.

Parameters

Parameter	Declaration	Data Type	Memory Area	Description
REQ	INPUT	BOOL	I, Q, M, D, L	Control parameter REQUEST, initiates terminating the connection specified by the ID. Initiation occurs at rising edge.
ID	INPUT	WORD	M, D, constant	Reference to the connection to be terminated to the remote partner or between the user program and the communications level of the operating system. ID must be identical to the associated parameter ID in the local connection description. Range of values: W#16#0001 to W#16#0FFF
DONE	OUTPUT	BOOL	I, Q, M, D, L	DONE status parameter: • 0: Job not yet started or still running. • 1: Job executed without error.
BUSY	OUTPUT	BOOL	I, Q, M, D, L	• BUSY = 1: Job is not yet completed. • BUSY = 0: Job is completed.
ERROR	OUTPUT	BOOL	I, Q, M, D, L	ERROR status parameter: • ERROR=1: Error occurred during processing. STATUS provides detailed information on the type of error
STATUS	OUTPUT	WORD	M, D	STATUS status parameter: Error information

Error Information

ERROR	STATUS (W#16#...)	Explanation
0	0000	Connection was able to be terminated
0	7000	First call with REQ=0, termination of connection not initiated
0	7001	First call with REQ=1, connection being terminated
0	7002	Follow-on call (REQ irrelevant), connection being terminated
1	8086	The ID parameter is not in the permitted address range
1	80A3	Attempt being made to terminate a non-existent connection
1	80C4	Temporary communications error: The interface is receiving new parameters.

See also:

Assigning Parameters for Open Communications Connections with TCP native and ISO on TCP

Assigning Parameters for the Local Communications Access Point with UDP

Establishing a Connection with FB 65 "TCON"

Sending Data via TCP native and ISO on TCP with FB 63 "TSEND"

Receiving Data via TCP native and ISO on TCP with FB 64 "TRCV"

Sending Data via UDP with FB 67 "TUSEND"

Receiving Data via UDP with FB 68 "TURCV"

1.9 Sending Data via TCP native and ISO on TCP with FB 63 "TSEND"

Description

FB 63 "TSEND" sends data over an existing communications connection.

Function

FB 63 "TSEND" is an asynchronously functioning FB, which means that its processing extends over several FB calls. To start sending data, call FB 63 with REQ = 1.

The job status is indicated at the output parameters BUSY and STATUS. STATUS corresponds to the RET_VAL output parameter of asynchronously functioning SFCs .

The following table shows the relationships between BUSY, DONE and ERROR. Using this table, you can determine the current status of FB 63 or when the establishment of the connection is complete.

BUSY	DONE	ERROR	Description
TRUE	irrelevant	irrelevant	The job is being processed.
FALSE	TRUE	FALSE	The job was completed successfully.
FALSE	FALSE	TRUE	The job was ended with an error. The cause of the error can be found in the STATUS parameter.
FALSE	FALSE	FALSE	The FB was not assigned a (new) job.

Note

Due to the asynchronous function of FB 63 "TSEND", you must keep the data in the sender area consistent until the DONE parameter or the ERROR parameter assumes the value TRUE.

Parameters

Parameter	Declaration	Data Type	Memory Area	Description
REQ	INPUT	BOOL	I, Q, M, D, L	Control parameter REQUEST, initiates the transmission at rising edge. At the first call with REQ=1, data are transmitted from the area specified by the DATA parameter.
ID	INPUT	WORD	M, D, constant	Reference to the connection to be terminated. ID must be identical to the associated parameter ID in the local connection description. Range of values: W#16#0001 to W#16#0FFF
LEN	INPUT	INT	I, Q, M, D, L	Number of bytes to be sent with the job Range of values: • 1 to 1460, if connection type is = B#16#01 • 1 to 8192, if connection type is = B#16#11 • 1 to 1452, if connection type is = B#16#12 and a CP is being used • 1 to 8192, if connection type is = B#16#12 and no CP is being used
DONE	OUTPUT	BOOL	I, Q, M, D, L	DONE status parameter: • 0: Job not yet started or still running. • 1: Job executed without error.
BUSY	OUTPUT	BOOL	I, Q, M, D, L	• BUSY = 1: Job is not yet completed. A new job cannot be triggered. • BUSY = 0: Job is completed.
ERROR	OUTPUT	BOOL	I, Q, M, D, L	ERROR status parameter: • ERROR=1: Error occurred during processing. STATUS provides detailed information on the type of error
STATUS	OUTPUT	WORD	M, D	STATUS status parameter: Error information
DATA	IN_OUT	ANY	I, M, D	Send area, contains address and length The address refers to: • The process image input table • The process image output table • A bit memory • A data block

Error Information

ERROR	STATUS (W#16#...)	Explanation
0	0000	Send job completed without error
0	7000	First call with REQ=0, sending not initiated
0	7001	First call with REQ=1, sending initiated
0	7002	Follow-on call (REQ irrelevant), job being processed Note: during this processing the operating system accesses the data in the DATA send buffer.
1	8085	LEN parameter has the value 0 or is greater than the largest permitted value
1	8086	The ID parameter is not in the permitted address range
0	8088	LEN parameter is larger than the memory area specified in DATA
1	80A1	Communications error: • FB 65 "TCON" was not yet called for the specified ID • The specified connection is currently being terminated. Transmission over this connection is not possible. • The interface is being reinitialized.
1	80B3	The parameter for the connection type (connection_type parameter in the connection description) is set to UDP. Please use the FB 67 "TUSEND".
1	80C3	The operating resources (memory) in the CPU are temporarily occupied.
1	80C4	Temporary communications error: • The connection to the communications partner cannot be established at this time. • The interface is receiving new parameters.
1	8822	DATA parameter: Source area invalid: area does not exist in DB.
	8824	DATA parameter: Range error in ANY pointer
1	8832	DATA parameter: DB number too large.
1	883A	DATA parameter: Access to send buffer not possible (e.g. due to deleted DB)
1	887F	DATA parameter: Internal error, such as an invalid ANY reference

See also:

Assigning Parameters for Open Communications Connections with TCP native and ISO on TCP

Assigning Parameters for the Local Communications Access Point with UDP

Establishing a Connection with FB 65 "TCON"

Terminating a Connection with FB 66 "TDISCON"

Receiving Data via TCP native and ISO on TCP with FB 64 "TRCV"

Sending Data via UDP with FB 67 "TUSEND"

Receiving Data via UDP with FB 68 "TURCV"

1.10 Receiving Data via TCP native and ISO on TCP with FB 64 "TRCV"

Description

FB 64 "TRCV" receives data over an existing communication connection.

The are two variants available for receiving and processing the data:

- Variant 1: Received data block is processed immediately.
- Variant 2: Received data block is stored in a receive buffer and is only processed when the buffer is full.

The following table shows the relationships between the connection type is shown in the following table:

Connection Type	Variant
B#16#01 and B#16#11	The user can specify the variant.
B#16#12	Variant 2 (fixed)

The following table describes both variants in detail.

Received data ...	Range (of Values) for LEN	Range (of Values) for RCVD_LEN	Description
are available immediately	0	1 to x	The data go into a buffer whose length x is specified in the ANY pointer of the receive buffer (DATA parameter). After being received, a data block is immediately available in the receive buffer. The amount of data received (RCVD_LEN parameter) can be no greater than the size specified in the DATA parameter. Receiving is indicated by NDR = 1.
are stored in the receive buffer. The data are available as soon as the configured length is reached.	• 11 to 1460, if the connection type = B#16#01 • 1 to 8192, if the connection type = B#16#11 • 1 to 1452, if the connection type = B#16#12 and a CP is being used • 1 to 8192, if the connection type = B#16#12 and no CP is being used	Same value as in the LEN parameter	The data go into a buffer whose length is specified by the LEN parameter. If this specified length is reached, the received data are made available in the DATA parameter (NDR = 1).

Function

FB 64 "TRCV" is an asynchronously functioning FB, which means that its processing extends over several FB calls. To start receiving data, call FB 64 with REQ = 1.

The job status is indicated at the output parameters BUSY and STATUS. STATUS corresponds to the RET_VAL output parameter of asynchronously functioning SFCs.

The following table shows the relationships between BUSY, DONE and ERROR. Using this table, you can determine the current status of FB 64 or when the receiving process is complete.

BUSY	DONE	ERROR	Description
TRUE	irrelevant	irrelevant	The job is being processed.
FALSE	TRUE	FALSE	The job was completed successfully.
FALSE	FALSE	TRUE	The job was ended with an error. The cause of the error can be found in the STATUS parameter.
FALSE	FALSE	FALSE	The FB was not assigned a (new) job.

Note

Due to the asynchronous function of FB 64 "TRCV", the data in the receiver area are only consistent when the NDR parameter assumes the value TRUE.

Parameters

Parameter	Declaration	Data Type	Memory Area	Description
EN_R	INPUT	BOOL	I, Q, M, D, L	Control parameter enabled to receive: when EN_R = 1, FB 64 "TRCV" is ready to receive.
ID	INPUT	WORD	M, D, constant	Reference to the connection to be terminated. ID must be identical to the associated parameter ID in the local connection description. Range of values: W#16#0001 to W#16#0FFF
LEN	INPUT	INT	I, Q, M, D, L	• LEN = 0 (ad hoc mode): use implied length specified in the ANY pointer for DATA. The received data are made available immediately when the block is called. The amount of data received is available in RCVD_LEN. • 1 <= LEN <= max: number of bytes to be received. The amount of data actually received is available in RCVD_LEN. The data are available after they have been completely received. "max" depends on the connection type: max = 1460 with connection type B#16#01, max = 8192 with connection type B#16#11, max = 1452 with connection type B#16#12 with a CP, max = 8192 with connection type B#16#12 without a CP
NDR	OUTPUT	BOOL	I, Q, M, D, L	NDR status parameter: • NDR = 0: Job not yet started or still running. • NDR = 1: Job successfully completed
ERROR	OUTPUT	BOOL	I, Q, M, D, L	ERROR status parameter: • ERROR=1: Error occurred during processing. STATUS provides detailed information on the type of error
BUSY	OUTPUT	BOOL	I, Q, M, D, L	• BUSY = 1: Job is not yet completed. A new job cannot be triggered. • BUSY = 0: Job is completed.
STATUS	OUTPUT	WORD	M, D	STATUS status parameter: Error information
RCVD_LEN	OUTPUT	INT	I, Q, M, D, L	Amount of data actually received, in bytes
DATA	IN_OUT	ANY	E, M, D	Receiving area, contains address and length The address refers to: • The process image input table • The process image output table • A bit memory • A data block

Error Information

ERROR	STATUS (W#16#...)	Explanation
0	0000	New data were accepted. The current length of the received data is shown in RCVD_LEN.
0	7000	First call with REQ=0, receiving not initiated
0	7001	Block is ready to receive.
0	7002	Follow-on call, job being processed Note: during this processing the operating system writes the operating system data to the DATA receive buffer. For this reason, an error could result in inconsistent data being in the receive buffer.
1	8085	LEN parameter is greater than the largest permitted value, or you changed the value of LEN from the one that existed during the first call
1	8086	The ID parameter is not in the permitted address range
0	8088	• Target buffer (DATA) is too small. • The value in LEN is greater than the receiver area specified by DATA. To correct the error if the connection type = B#16#12: increase the size of the DATA target buffer.
1	80A1	Communications error: • FB 65 "TCON" was not yet called for the specified ID • The specified connection is currently being terminated. Receiving over this connection is not possible. • The interface is receiving new parameters.
1	80B3	The parameter for the connection type (connection_type parameter in the connection description) is set to UDP. Please use the FB 68 "TRCV".
1	80C3	The operating resources (memory) in the CPU are temporarily occupied.
1	80C4	Temporary communications error: The connection is currently being terminated.
1	8922	DATA parameter: Target area invalid: area does not exist in DB.
	8824	DATA parameter: Range error in ANY pointer
1	8932	DATA parameter: DB number too large.
1	893A	DATA parameter: Access to receive buffer not possible (e.g. due to deleted DB
1	897F	DATA parameter: Internal error, such as an invalid ANY reference

See also:

Assigning Parameters for Open Communications Connections with TCP native and ISO on TCP

Assigning Parameters for the Local Communications Access Point with UDP

Establishing a Connection with FB 65 "TCON"

Terminating a Connection with FB 66 "TDISCON"

Sending Data via TCP native and ISO on TCP with FB 63 "TSEND"

Sending Data via UDP with FB 67 "TUSEND"

Receiving Data via UDP with FB 68 "TURCV"

1.11 Sending Data via UDP with FB 67 "TUSEND"

Description

FB 67 "TUSEND" sends data via UDP to the remote partner specified by the parameter ADDR.

Note

When sending separate data in sequence to different partners, you only need to adjust the parameter ADDR when calling FB 67 "TUSEND". It is not necessary to call FBs 65 "TCON" and 66 "TDISCON" again.

Function

FB 67 "TUSEND" is an asynchronously functioning FB, which means that its processing extends over several FB calls. To start sending data, call FB 67 with REQ = 1.

The job status is indicated at the output parameters BUSY and STATUS. STATUS corresponds to the RET_VAL output parameter of asynchronously functioning SFCs.

The following table shows the relationships between BUSY, DONE and ERROR. Using this table, you can determine the current status of FB 67 or when the sending process (transmission) is complete.

BUSY	DONE	ERROR	Description
TRUE	irrelevant	irrelevant	The job is being processed.
FALSE	TRUE	FALSE	The job was completed successfully.
FALSE	FALSE	TRUE	The job was ended with an error. The cause of the error can be found in the STATUS parameter.
FALSE	FALSE	FALSE	The FB was not assigned a (new) job.

Note

Due to the asynchronous function of FB 67 "TUSEND", you must keep the data in the sender area consistent until the DONE parameter or the ERROR parameter assumes the value TRUE.

Parameters

Parameter	Declaration	Data Type	Memory Area	Description
REQ	INPUT	BOOL	I, Q, M, D, L	Control parameter REQUEST, initiates the transmission at rising edge. At the first call with REQ=1, bytes are transmitted from the area specified by the DATA parameter.
ID	INPUT	WORD	M, D, constant	Reference to the associated connection between the user program and the communication level of the operating system. ID must be identical to the associated parameter ID in the local connection description. Range of values: W#16#0001 to W#16#0FFF
LEN	INPUT	INT	I, Q, M, D, L	Number of bytes to be sent with the job Range of values: 1 to 1460
DONE	OUTPUT	BOOL	I, Q, M, D, L	DONE status parameter: • 0: Job not yet started or still running. • 1: Job executed without error.
BUSY	OUTPUT	BOOL	I, Q, M, D, L	• BUSY = 1: Job is not yet completed. A new job cannot be triggered. • BUSY = 0: Job is completed.
ERROR	OUTPUT	BOOL	I, Q, M, D, L	ERROR status parameter: • ERROR=1: Error occurred during processing. STATUS provides detailed information on the type of error
STATUS	OUTPUT	WORD	M, D	STATUS status parameter: Error information
DATA	IN_OUT	ANY	I, Q, M, D	Sender area, contains address and length The address refers to: • The process image input table • The process image output table • A bit memory • A data block
ADDR	IN_OUT	ANY	D	Pointer to the address of the receiver (e.g. P#DB100.DBX0.0 byte 8), see Structure of the Address Information of the Remote Partner with UDP

Error Information

ERROR	STATUS (W#16#...)	Explanation
0	0000	Send job completed without error
0	7000	First call with REQ=1, sending not initiated
0	7001	First call with REQ=1, sending initiated
0	7002	Follow-on call (REQ irrelevant), job being processed Note: during this processing the operating system accesses the data in the DATA send buffer.
1	8085	LEN parameter has the value 0 or is greater than the largest permitted value
1	8086	The ID parameter is not in the permitted address range
0	8088	LEN parameter is larger than the memory area specified in DATA
1	80A1	Communications error: • FB 65 "TCON" was not yet called for the specified ID • The specified connection between the user program and the communication level of the operating system is currently being terminated. Transmission over this connection is not possible. • The interface is being reinitialized (receiving new parameters).
1	80B3	The parameter for the connection type (connection_type parameter in the connection description) is not set to UDP. Please use the FB 63 "TSEND".
1	80C3	The operating resources (memory) in the CPU are temporarily occupied.
1	80C4	Temporary communications error: • The connection between the user program and the communication level of the operating system cannot be established at this time. • The interface is receiving new parameters.
1	8822	DATA parameter: Source area invalid: area does not exist in DB.
1	8824	DATA parameter: Range error in ANY pointer
1	8832	DATA parameter: DB number too large.
1	883A	DATA parameter: Access to send buffer not possible (e.g. due to deleted DB)
1	887F	DATA parameter: Internal error, such as an invalid ANY reference

See also:

Assigning Parameters for Open Communications Connections with TCP native and ISO on TCP

Assigning Parameters for the Local Communications Access Point with UDP

Establishing a Connection with FB 65 "TCON"

Terminating a Connection with FB 66 "TDISCON"

Sending Data via TCP native and ISO on TCP with FB 63 "TSEND"

Receiving Data via TCP native and ISO on TCP with FB 64 "TRCV"

Receiving Data via UDP with FB 68 "TURCV"

1.12 Receiving Data via UDP with FB 68 "TURCV"

Description

FB 68 "TURCV" receives data via UDP. After successful completion of FB 68 "TURCV" the parameter ADDR will show you the address of the remote partner (the sender).

Function

FB 68 "TURCV" is an asynchronously functioning FB, which means that its processing extends over several FB calls. To start sending data, call FB 68 with REQ = 1.

The job status is indicated at the output parameters RET_VAL and BUSY. STATUS corresponds to the RET_VAL output parameter of asynchronously functioning SFCs.

The following table shows the relationships between BUSY, DONE and ERROR. Using this table, you can determine the current status of FB 68 or when the receiving process is complete.

BUSY	DONE	ERROR	Description
TRUE	irrelevant	irrelevant	The job is being processed.
FALSE	TRUE	FALSE	The job was completed successfully.
FALSE	FALSE	TRUE	The job was ended with an error. The cause of the error can be found in the STATUS parameter.
FALSE	FALSE	FALSE	The FB was not assigned a (new) job.

Note

Due to the asynchronous function of FB 68 "TURCV", the data in the receiver area are only consistent when the NDR parameter assumes the value TRUE.

Parameters

Parameter	Declaration	Data Type	Memory Area	Description
EN_R	INPUT	BOOL	I, Q, M, D, L	Control parameter enabled to receive: when EN_R = 1, FB 68 "TURCV" is ready to receive.
ID	INPUT	WORD	M, D, constant	Reference to the associated connection between the user program and the communication level of the operating system. ID must be identical to the associated parameter ID in the local connection description. Range of values: W#16#0001 to W#16#0FFF
LEN	INPUT	INT	I, Q, M, D, L	1 <= LEN <= 1460: number of bytes to be received. The received data are immediately available when the block is called. The amount of data received is available in RCVD_LEN.
NDR	OUTPUT	BOOL	I, Q, M, D, L	NDR status parameter: • NDR = 0: Job not yet started or still running. • NDR = 1: Job successfully completed
ERROR	OUTPUT	BOOL	I, Q, M, D, L	ERROR status parameter: • ERROR=1: Error occurred during processing. STATUS provides detailed information on the type of error
BUSY	OUTPUT	BOOL	I, Q, M, D, L	• BUSY = 1: Job is not yet completed. A new job cannot be triggered. • BUSY = 0: Job is completed.
STATUS	OUTPUT	WORD	M, D	STATUS status parameter: Error information
RCVD_LEN	OUTPUT	INT	I, Q, M, D, L	Amount of data actually received, in bytes
DATA	IN_OUT	ANY	I, Q, M, D	Receiver area, contains address and length The address refers to: • The process image input table • The process image output table • A bit memory • A data block
ADDR	IN_OUT	ANY	D	Pointer to the address of the sender (e.g. P#DB100.DBX0.0 byte 8), see Structure of the Address Information of the Remote Partner with UDP

Error Information

ERROR	STATUS (W#16#...)	Explanation
0	0000	New data were accepted. The current length of the received data is shown in RCVD_LEN.
0	7000	First call with REQ=0, receiving not initiated
0	7001	Block is ready to receive.
0	7002	Follow-on call, job being processed Note: during this processing the operating system writes the operating system data to the DATA receive buffer. For this reason, an error could result in inconsistent data being in the receive buffer.
1	8085	LEN parameter is greater than the largest permitted value, or you changed the value of LEN from the one that existed during the first call
1	8086	The ID parameter is not in the permitted address range
1	8088	• Target buffer (DATA) is too small. • The value in LEN is greater than the receiver area specified by DATA.
1	80A1	Communications error: • FB 65 "TCON" was not yet called for the specified ID • The specified connection between the user program and the communication level of the operating system is currently being terminated. Receiving over this connection is not possible. • The interface is being reinitialized (receiving new parameters).
1	80B3	The parameter for the connection type (connection_type parameter in the connection description) is not set to UDP. Please use the FB 68 "TRCV".
1	80C3	The operating resources (memory) in the CPU are temporarily occupied.
1	80C4	Temporary communications error: The connection is currently being established.
1	8922	DATA parameter: Target area invalid: area does not exist in DB.
1	8924	DATA parameter: Range error in ANY pointer
1	8932	DATA parameter: DB number too large.
1	893A	DATA parameter: Access to receive buffer not possible (e.g. due to deleted DB
1	897F	DATA parameter: Internal error, such as an invalid ANY reference

See also:

Assigning Parameters for Open Communications Connections for TCP native and ISO on TCP

Assigning Parameters for the Local Communications Access Point for UDP

Establishing a Connection with FB 65 "TCON"

Terminating a Connection with FB 66 "TDISCON"

Sending Data via TCP native and ISO on TCP with FB 63 "TSEND"

Receiving Data via TCP native and ISO on TCP with FB 64 "TRCV"

Sending Data via UDP with FB 67 "TUSEND"

Index