SIEMENS

Ingenuity for life

S7 Communication with PUT/GET

S7-1200 CPUs / S7-300 CPUs

https://support.industry.siemens.com/cs/ww/en/view/82212115

Siemens Industry Online Support

This entry originates from Siemens Industry Online Support. The conditions of use specified there apply (www.siemens.com/nutzungsbedingungen).

Security Information

Siemens provides products and solutions with industrial security functions that support the secure operation of plants, systems, machines and networks.

In order to protect plants, systems, machines and networks against cyber threats, it is necessary to implement – and continuously maintain – a holistic, state-of-the-art industrial security concept. Siemens' products and solutions only form one element of such a concept.

Customers are responsible to prevent unauthorized access to their plants, systems, machines and networks. Systems, machines and components should only be connected to the enterprise network or the internet if and to the extent necessary and with appropriate security measures (e.g. use of firewalls and network segmentation) in place.

Additionally, Siemens' guidance on appropriate security measures should be taken into account. For more information about industrial security, please visit http://www.siemens.com/industrialsecurity.

Siemens' products and solutions undergo continuous development to make them more secure. Siemens strongly recommends to apply product updates as soon as available and to always use the latest product versions. Use of product versions that are no longer supported, and failure to apply latest updates may increase the customer's exposure to cyber threats.

To stay informed about product updates, subscribe to the Siemens Industrial Security RSS Feed under http://www.siemens.com/industrialsecurity.

Contents

1 Introduction

You can use the S7 Communication, for example, for data transfer via the integrated PROFINET interface and Industrial Ethernet interface of the S7-1200/S7-300 CPUs.

The following instructions are available for S7 Communication:

- PUT for sending data
- GET for receiving data

In STEP 7 V16 (TIA Portal) you will find the above-mentioned instructions in the "Instructions" task card under "Communication > S7 Communication".

This sample program shows how to call the "PUT" and "GET" instructions in the user program of the S7-1200/S7-300 CPUs in order to transfer more than 160 bytes of data via an S7 connection.

Only one job at a time can be triggered by the "PUT" and "GET" instructions via a configured S7 connection.

It is not possible to trigger multiple jobs simultaneously by the "PUT" and "GET" instructions via a configured S7 connection.

The "PUT" and "GET" instructions can only be called sequentially, that is one after the other. A job is triggered only when the previous job is completed.

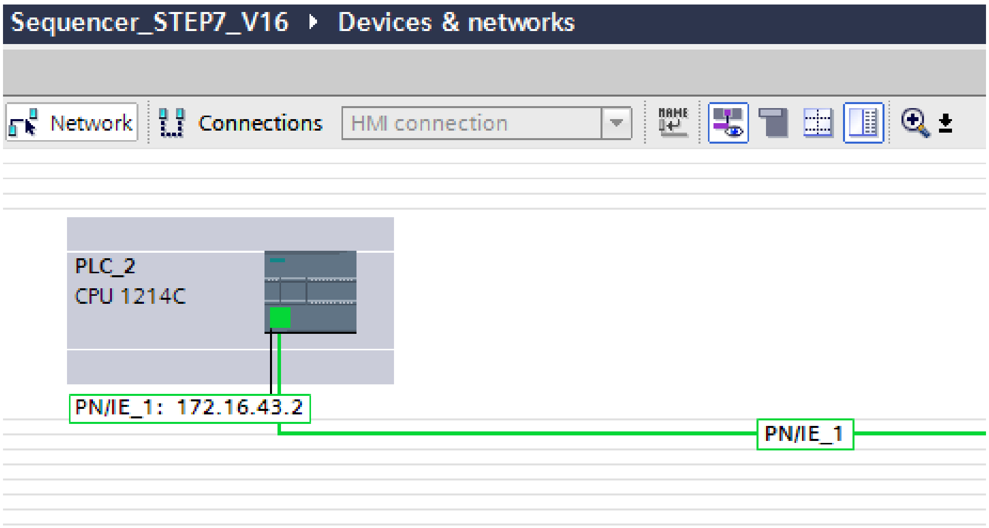
In this sample program the "PUT" and "GET" instructions are called sequentially so that the S7-1200 CPU or S7-300 CPU can send and receive up to 400 bytes of data via a configured S7 connection.

2 Configuration

2.1 Configuration of the Hardware

1. In the network view of the "Devices & networks" editor you create an S7-1200 CPU or an S7-300 CPU.
2. Network the S7 CPU and, if necessary, the CP and set the IP addresses, for example: 172.16.43.2.

Figure 2-1



2.2 Configuration of the S7 Connection

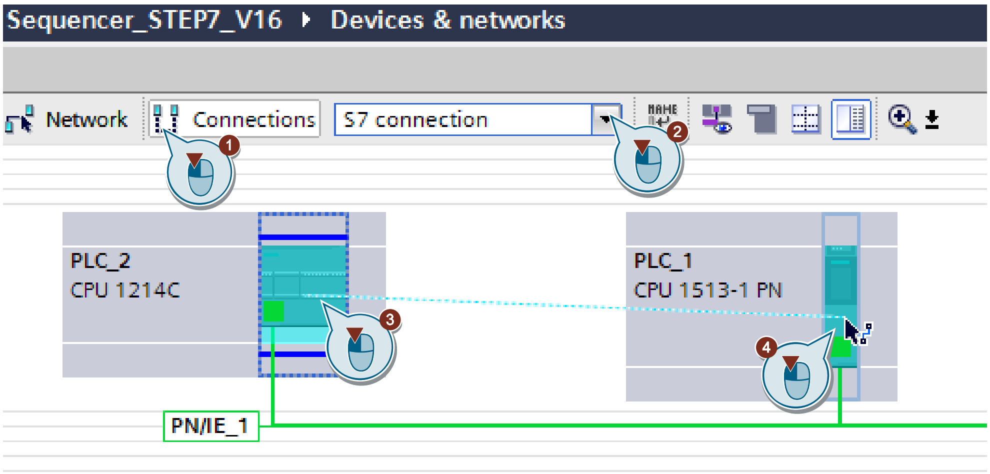
2.2.1 Configuring the S7 Connection Graphically

Requirements

To create the S7 connection graphically the S7-1200 CPU or S7-300 CPU has to be configured in the same project as the communication partner. If you create the S7 connection graphically, the S7 connection is configured on both sides.

Instructions

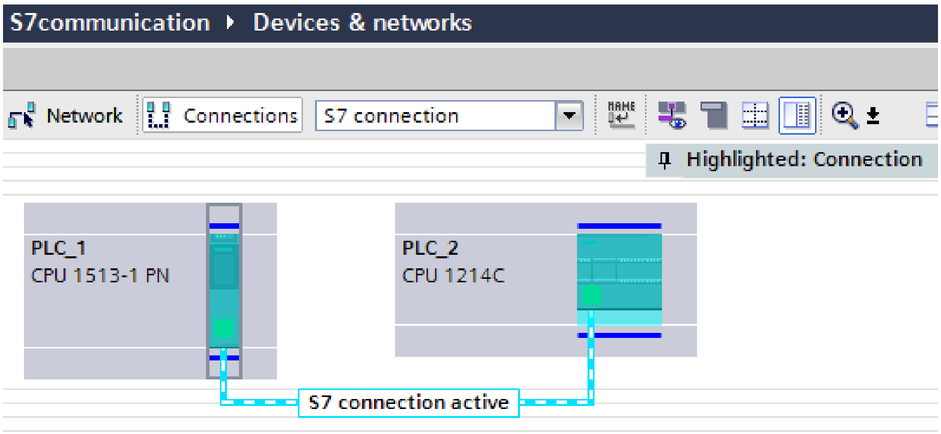
Figure 2-2



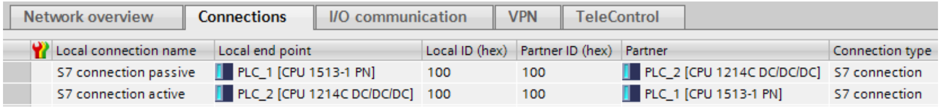
1. In the Network view you click the "Connections" icon to enable the Connection mode.
2. Select "S7 connection" as the connection type in the adjacent drop-down list box.
   In the Network view, all devices that are available for an S7 connection are highlighted in color.
3. With the button held down drag the mouse cursor from the module (CPU, CP or CM) where the S7 connection is to start (active connection establishment) to the module (CPU, CP or CM) where the S7 connection is to finish (passive connection establishment).
4. Release the mouse button when the cursor is on the target device to create the S7 connection between the communication partners.

Result

- An S7 connection configured on both sides is created.
- The connection path is displayed highlighted in the graphical area of the Network view.



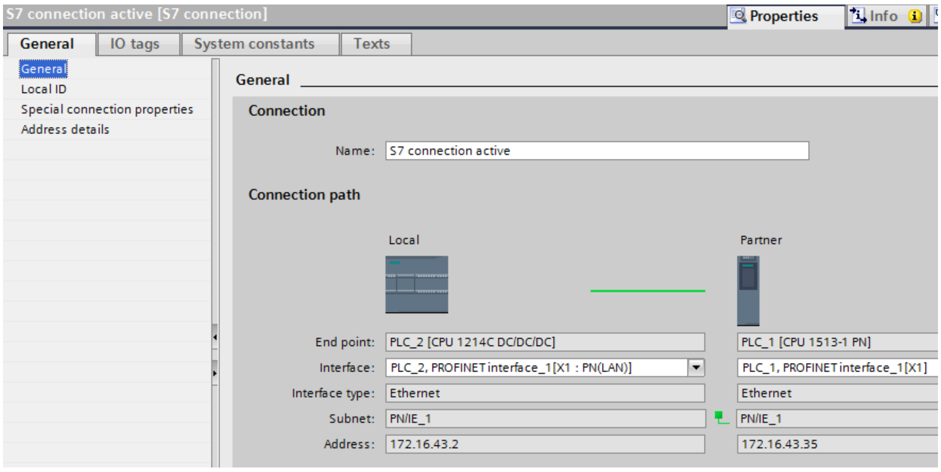
- The S7 connection is entered in the "Connections" table in the table area of the Network view. Since the S7 connection is configured on both sides two connections are created:
  - One S7 connection uses the S7-1200 CPU as local endpoint. This sets up the S7 connection actively.
  - The second S7 connection uses an S7-1500 CPU as local endpoint. This participates passively in establishing the connection.



Local connection name	Local end point	Local ID (hex)	Partner ID (hex)	Partner	Connection type
S7 connection passive	PLC_1 [CPU 1513-1 PN]	100	100	PLC_2 [CPU 1214C DC/DC/DC]	S7 connection
S7 connection active	PLC_2 [CPU 1214C DC/DC/DC]	100	100	PLC_1 [CPU 1513-1 PN]	S7 connection

Setting the connection parameters

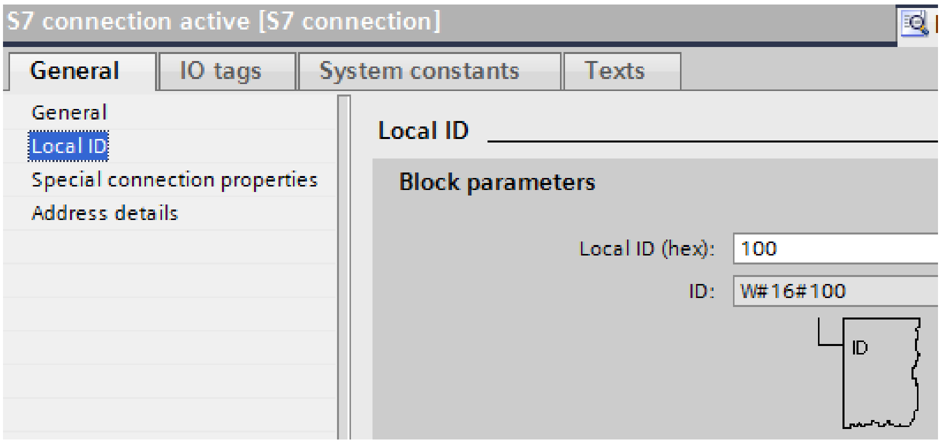
1. In the connection table, mark the S7 connections that have the S7-1200 CPU as their local endpoint.
   The Properties of the S7 connection are displayed in the inspector window.
2. Under "General" the "General" tab displays the general connection parameters that identify the local endpoint.
   You can assign the connection route and specify the connection route in full.



Name: S7 connection active

	Local	Partner
End point:	PLC_2 [CPU 1214C DC/DC/DC]	PLC_1 [CPU 1513-1 PN]
Interface:	PLC_2, PROFINET interface_1[X1 : PN(LAN)]	PLC_1, PROFINET interface_1[X1]
Interface type:	Ethernet	Ethernet
Subnet:	PN/IE_1	PN/IE_1
Address:	172.16.43.2	172.16.43.35

3. In the “General” tab under “Local ID” you see the local ID of the module from which the connection is viewed (local partner). You can change the local ID. You specify the local ID later at the "ID" input parameter of the "PUT" or "GET" instruction.
   In this example we use the local ID with the value 100 (hex).



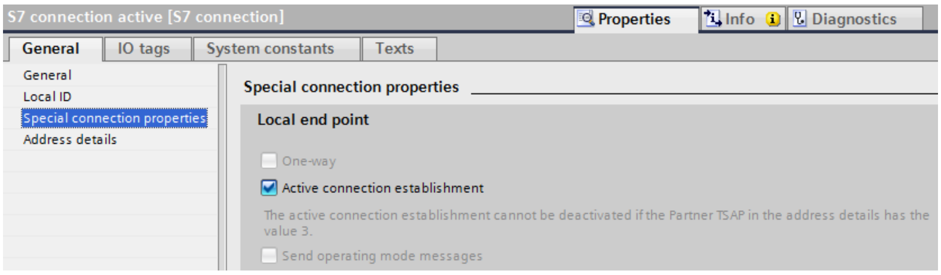
Local ID

Block parameters

Local ID (hex): 100

ID: W#16#100

4. In the "General” tab under "Special connection properties" you can set which connection partner can actively establish the connection as well as actively send and receive.



Special connection properties

Local end point

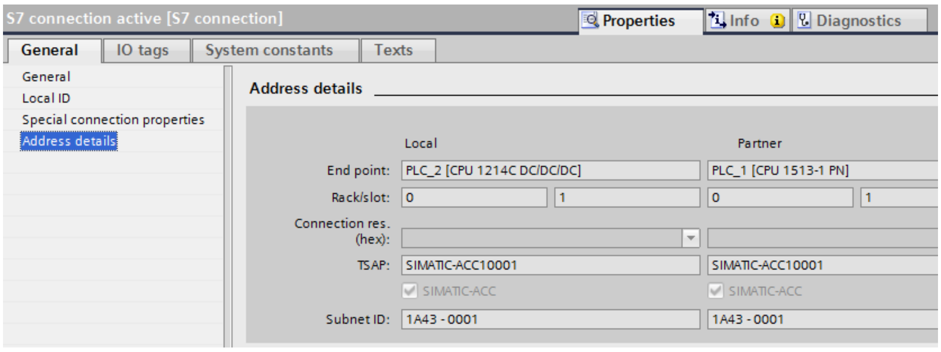
☐ One-way

☑ Active connection establishment

The active connection establishment cannot be deactivated if the Partner TSAP in the address details has the value 3.

☐ Send operating mode messages

5. The address details of the S7 connection are displayed here. The values are taken from the current configuration and cannot be changed.



Address details

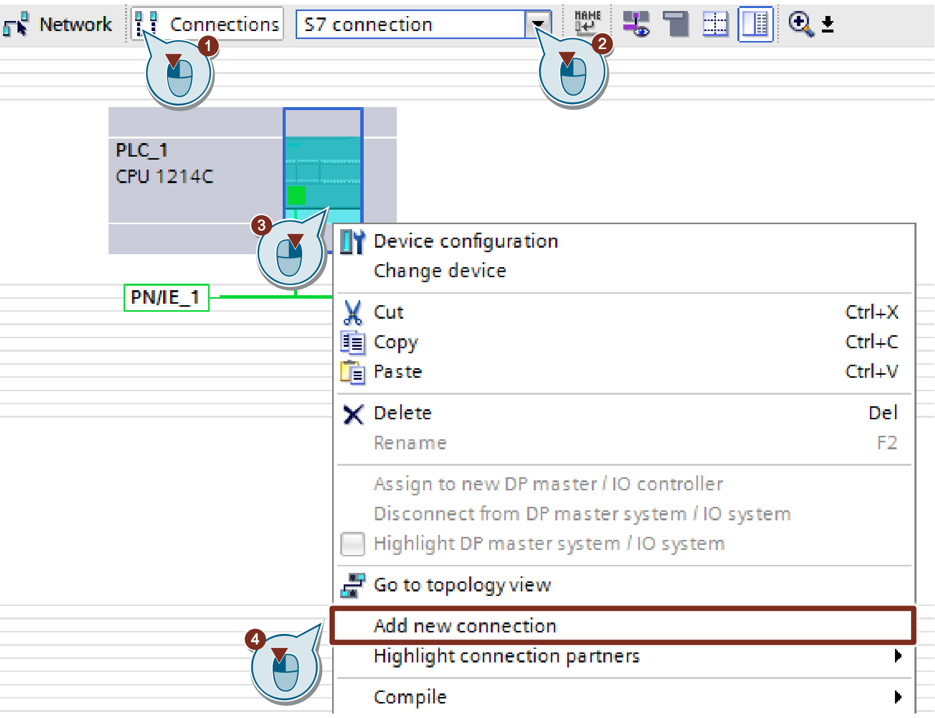
	Local		Partner	
End point:	PLC_2 [CPU 1214C DC/DC/DC]		PLC_1 [CPU 1513-1 PN]	
Rack/slot:	0	1	0	1
Connection res. (hex):				
TSAP:	SIMATIC-ACC10001		SIMATIC-ACC10001	
	☑ SIMATIC-ACC		☑ SIMATIC-ACC	
Subnet ID:	1A43 - 0001		1A43 - 0001	

2.2.2 Configure S7 Connection Unilaterally with Unspecified Partner

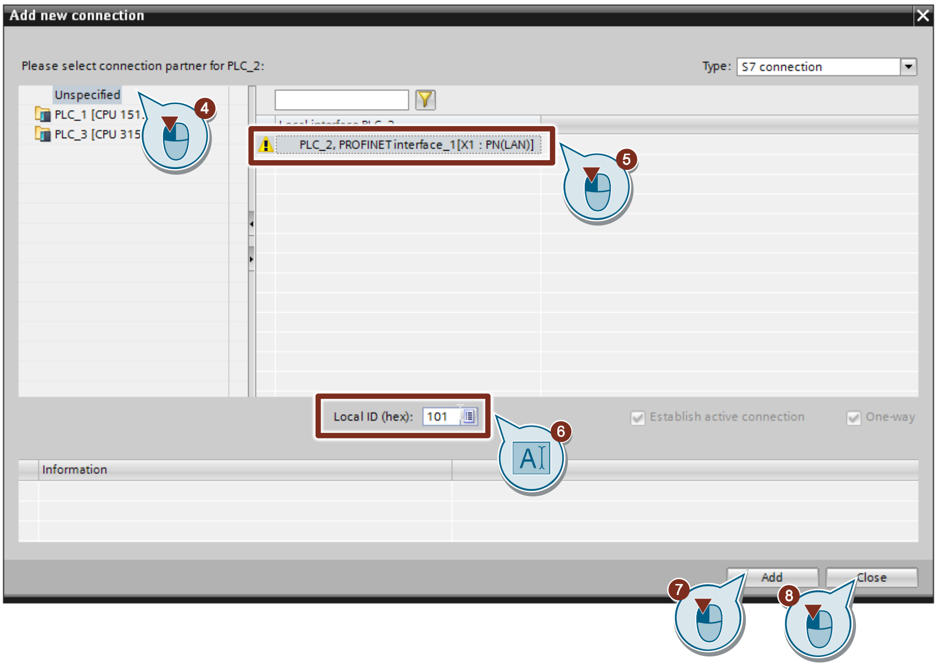
If you are using the communication services "PUT" and "GET" for data exchange, you can create the S7 connection unilaterally with unspecified partner, for instance when communication partners are configured in different projects. Create the S7 connection in the active S7 CPU.

Instructions

1. In the Network view you click the "Connections" icon to enable the Connection mode.
2. Select "S7 connection" as the connection type in the adjacent drop-down list box.
   In the Network view, all devices that are available for an S7 connection are highlighted in color.
3. Right-click the S7 CPU that actively establishes the connection, S7-1200 CPU, for example.
   The pop-up menu opens.
4. Select the "Add new connection" item.
   The "Add new connection" dialog opens.



5. Specify the following connection partner: "Unspecified".
6. Select the local interface. In this example, the integrated interface of the S7-1200 CPU is used. If the S7 connection is to be established via a CP, select the CP.
7. Enter the local ID for the S7 connection, 101 (hex), for example.
8. Click the "Add" button to add the unspecified S7 connection.
9. Click the "Close" button to end the dialog.



Add new connection

Please select connection partner for PLC_2:

Type: S7 connection

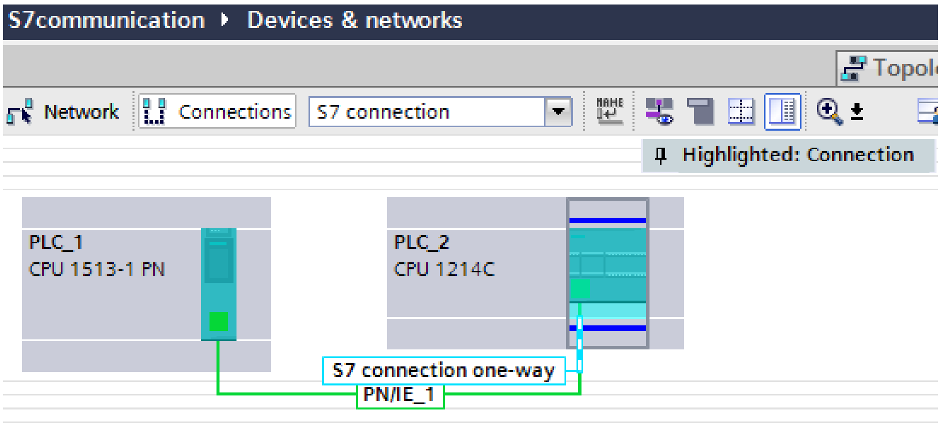
Local ID (hex): 101

☑ Establish active connection

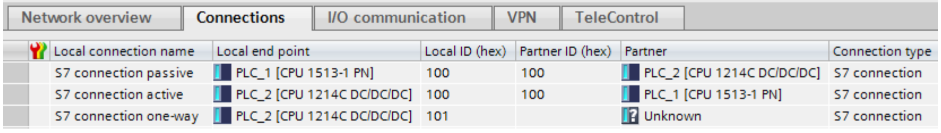
☑ One-way

Result

- An unspecified S7 connection is created.
- The connection path is displayed highlighted in the graphical area of the Network view.



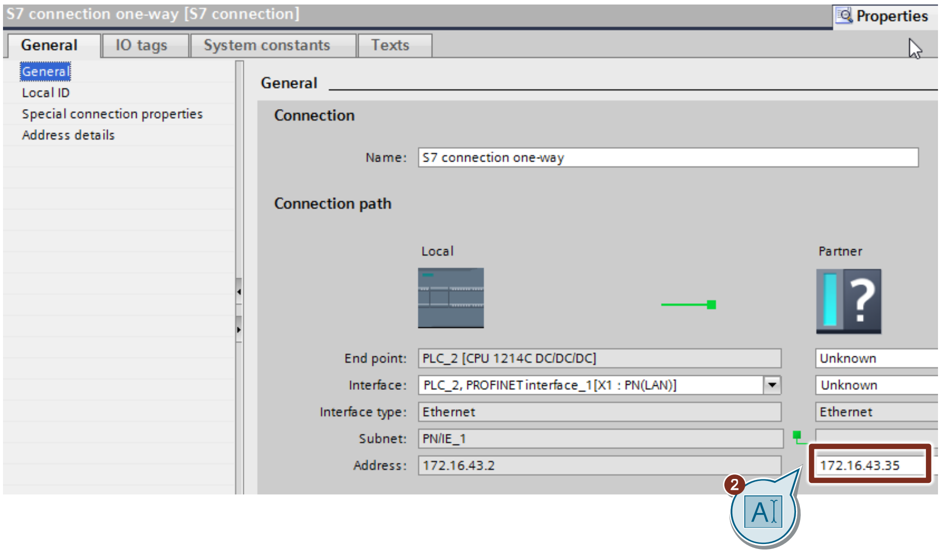
- The S7 connection is entered in the "Connections" table in the table area of the Network view.



Local connection name	Local end point	Local ID (hex)	Partner ID (hex)	Partner	Connection type
S7 connection passive	PLC_1 [CPU 1513-1 PN]	100	100	PLC_2 [CPU 1214C DC/DC/DC]	S7 connection
S7 connection active	PLC_2 [CPU 1214C DC/DC/DC]	100	100	PLC_1 [CPU 1513-1 PN]	S7 connection
S7 connection one-way	PLC_2 [CPU 1214C DC/DC/DC]	101		Unknown	S7 connection

Setting the connection parameters

1. In the “Connections” table, mark the unspecified S7 connections.
   The Properties of the S7 connection are displayed in the inspector window.
2. Under "General" the "General" tab displays the general connection parameters that identify the local endpoint. Enter the IP address of the communication partner, for example 172.16.43.35 (IP address of the S7-1500 CPU).



S7 connection one-way [S7 connection]

General

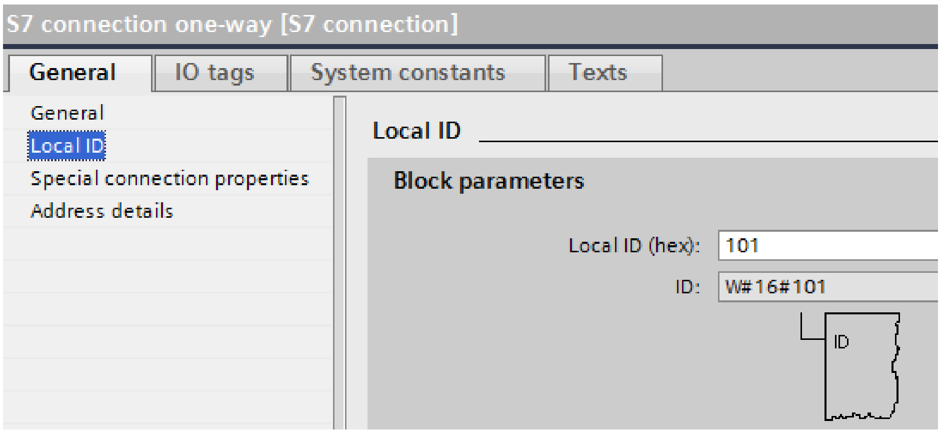
Connection

Name: S7 connection one-way

Connection path

	Local	Partner
End point:	PLC_2 [CPU 1214C DC/DC/DC]	Unknown
Interface:	PLC_2, PROFINET interface_1[X1 : PN(LAN)]	Unknown
Interface type:	Ethernet	Ethernet
Subnet:	PN/IE_1	
Address:	172.16.43.2	172.16.43.35

3. In the “General” tab under “Local ID” you see the local ID of the module from which the connection is viewed (local partner). You can change the local ID. You specify the local ID later at the "ID" input parameter of the "PUT" or "GET" instruction.
   In this example we use the local ID with the value 101 (hex).



S7 connection one-way [S7 connection]

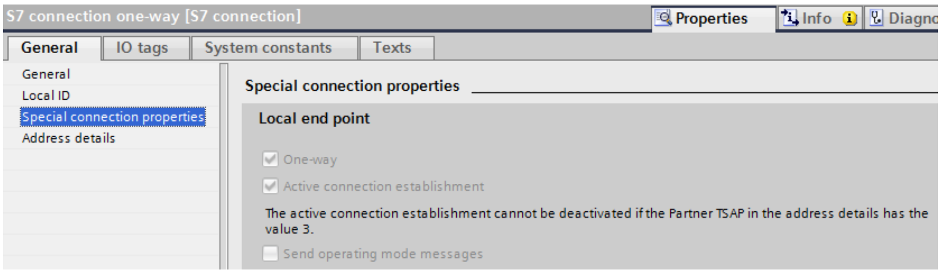
Local ID

Block parameters

Local ID (hex): 101

ID: W#16#101

4. In the "General" under "Special connection properties" you see that the S7 connection is configured unilaterally. Unilaterally means that the connection partner is server for this connection and cannot actively send or receive. The S7 CPU that is used as the local endpoint for the S7 connection, actively establishes the S7 connection as a client and actively sends and receive.



Special connection properties

Local end point

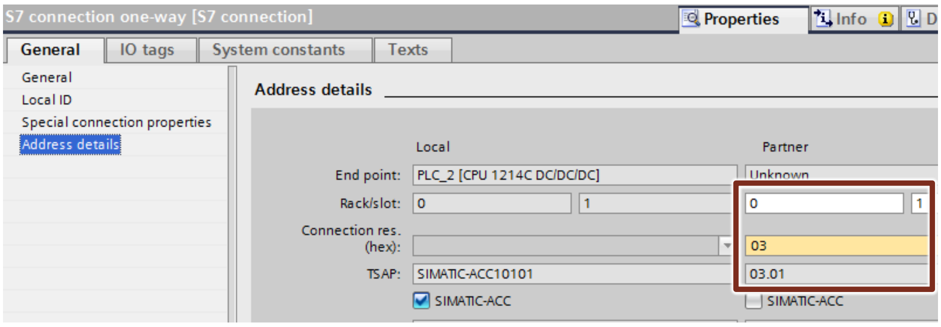
☑ One-way

☑ Active connection establishment

The active connection establishment cannot be deactivated if the Partner TSAP in the address details has the value 3.

□ Send operating mode messages

5. In the "General" tab under "Address details" you enter the partner TSAP. Since the S7 connection is configured unilaterally to an unspecified partner, the partner TSAP is composed as follows:
03 <slot of CPU or CP/CM>, for example:
   - S7-1500 CPU in slot 1: 03.01



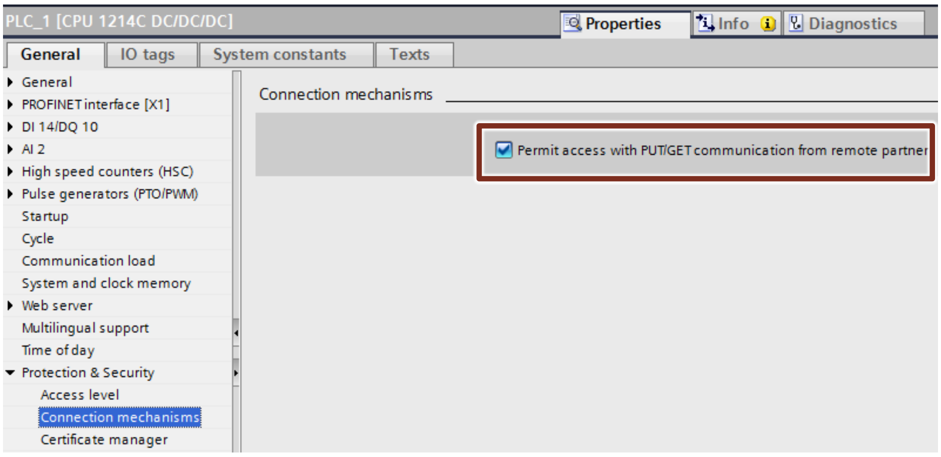
Address details

	Local		Partner	
End point:	PLC_2 [CPU 1214C DC/DC/DC]		Unknown	
Rack/slot:	0	1	0	1
Connection res. (hex):			03	
TSAP:	SIMATIC-ACC10101		03.01	
	☑ SIMATIC-ACC		□ SIMATIC-ACC	

2.3 Permit Access with PUT/GET Communication from Remote Partner

In the S7-1200 CPU and in the S7-1500 CPU you enable the function "Permit access with PUT/GET communication from remote partner". This enables the partner CPU to access the data in the S7-1200 CPU or S7-1500 CPU using the "PUT" and "GET" instructions.

1. In the Device view or Network view of the "Devices & networks" editor you mark the S7 CPU.
2. In the "General" tab, under "Protection & Security > Connection mechanisms" you enable the "Permit access with PUT/GET communication from remote partner" function.



Connection mechanisms

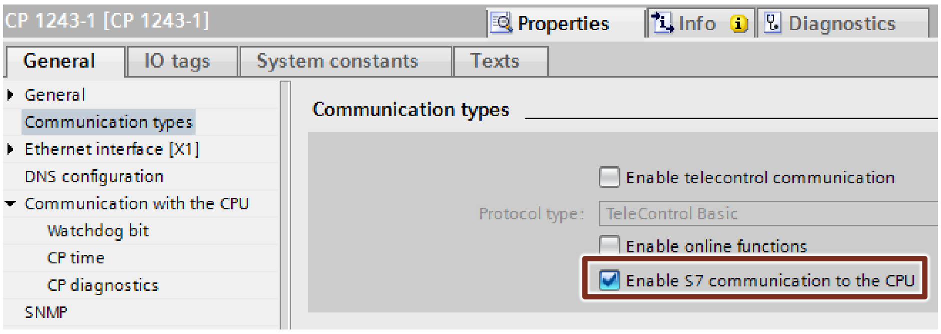
☑ Permit access with PUT/GET communication from remote partner

2.4 Enable S7 communication to the CPU

With the S7-1200, enable S7 communication to the CPU in the CP. In order to release the function of S7 communication with the assigned CPU and the S7 routing.

If you configure an S7 connection that runs via the CP of the S7-1200, then you have to activate this option.

1. In the Device view or Network view of the "Devices & networks" editor you mark the CP in the S7-1200.
2. In the "General" tab, under "Communication types" you enable the "Enable S7 communication to the CPU" option.



Communication types

☐ Enable telecontrol communication

Protocol type: TeleControl Basic

☐ Enable online functions

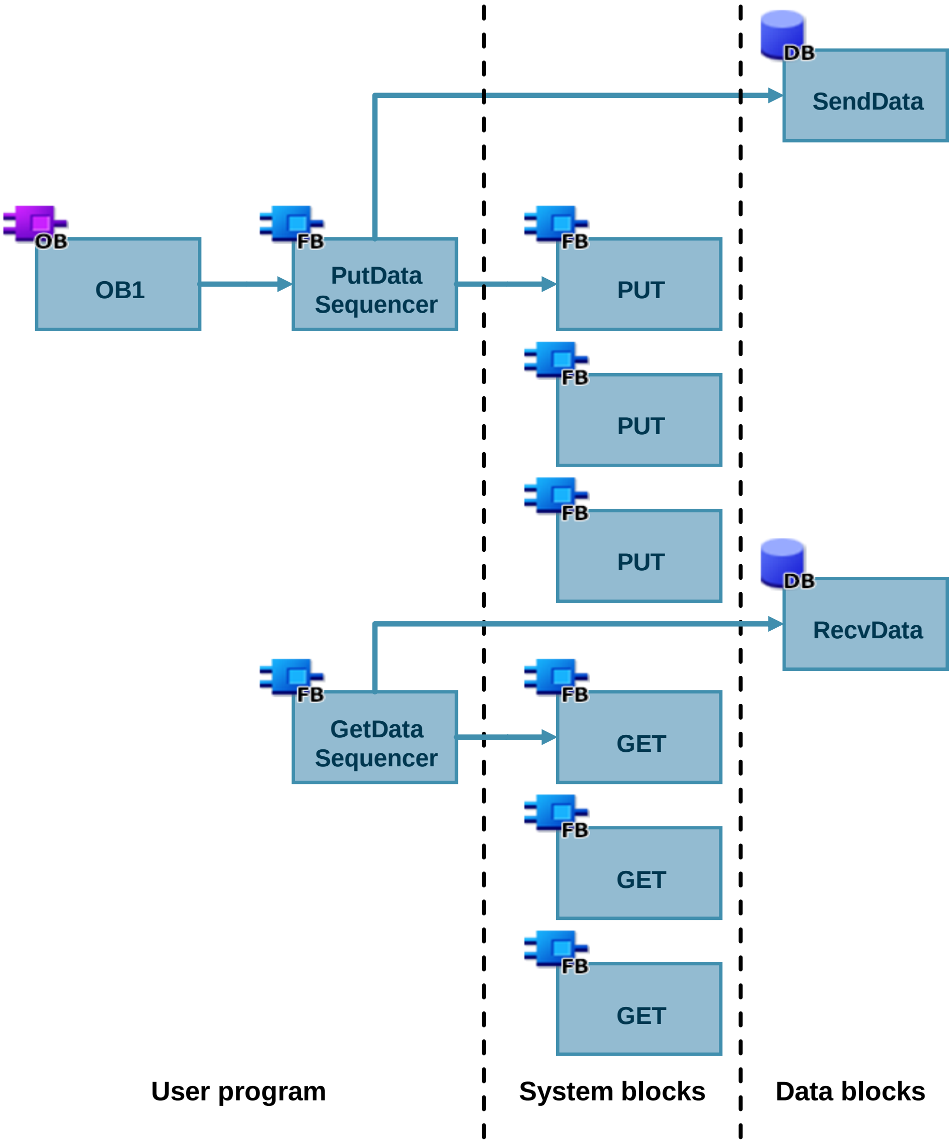
☑ Enable S7 communication to the CPU

3 User Program of the S7-1200 CPU or S7-300 CPU (Active S7 CPU)

3.1 Overview

The following figure shows an overview of the user program of the active S7 CPU.

Figure 3-1



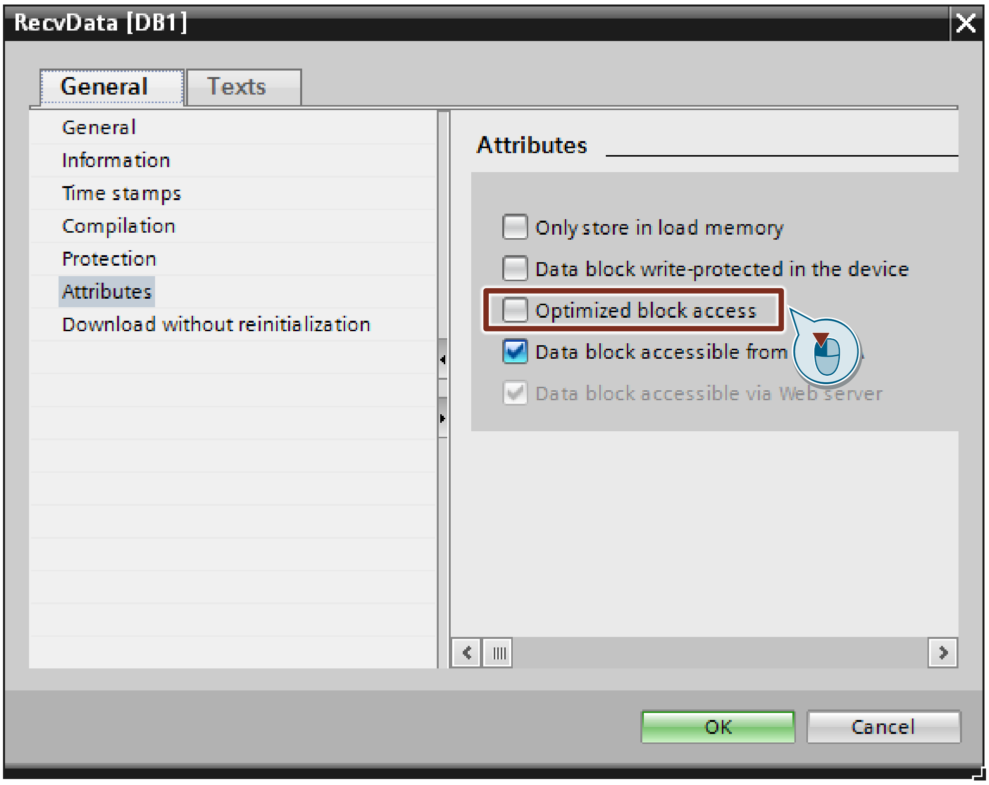
3.2 Create Send and Receive Data Areas

3.2.1 DB1 "RecvData"

The data read from the partner CPU is stored in the data block DB1 "RecvData" of the active S7 CPU.

In the Properties of the DB1 "RecvData", under "Attributes" you disable the "Optimized block access" function.

Figure 3-2



Attributes

☐ Only store in load memory

☐ Data block write-protected in the device

☐ Optimized block access

☑ Data block accessible from

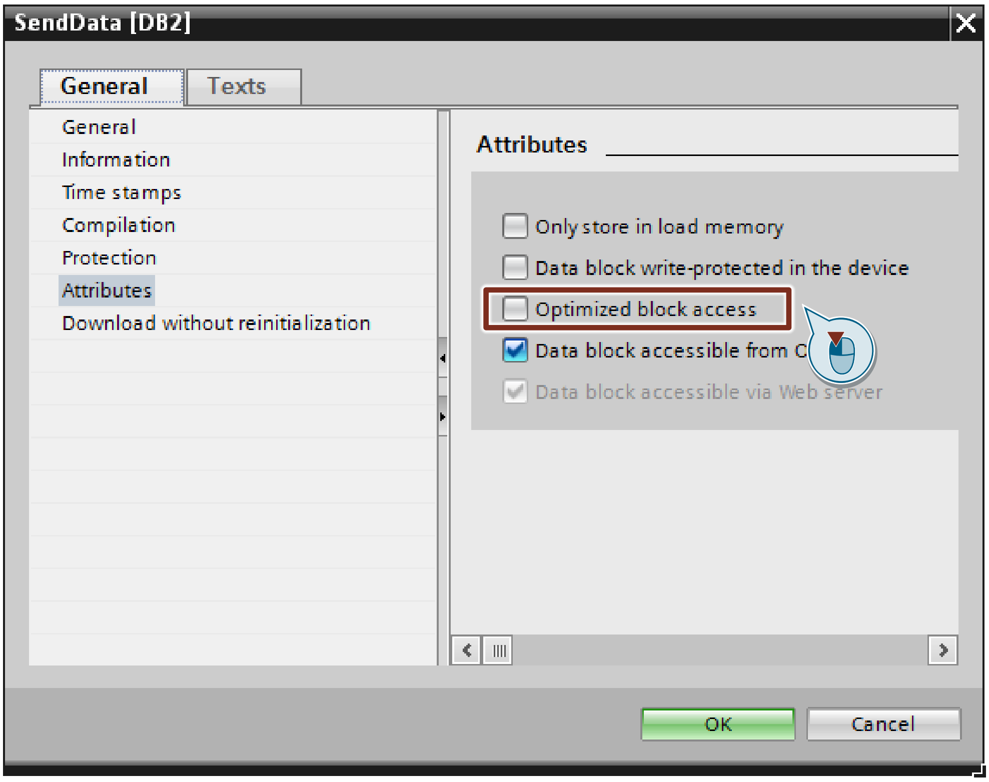
☑ Data block accessible via Web server

3.2.2 DB2 "SendData"

The data that is transferred to the partner CPU is stored in the data block DB2 "SendData" of the active S7 CPU.

In the Properties of the DB2 "SendData", under "Attributes" you disable the "Optimized block access" function.

Figure 3-3



□ Only store in load memory

□ Data block write-protected in the device

□ Optimized block access

☑ Data block accessible from C

☑ Data block accessible via Web server

3.3 Create Variables for Parameters of Function Blocks "PutDataSequence" and "GetDataSequence"

Create the following variables in DB10 "GeneralData" to assign the input and output parameters of the function blocks "PutDataSequence" and "GetDataSequence". PLC data types are used as data type.

Table 3-1

Variable	PLC data type	Description
put	typePut	Variables for parameterizing FB12 "PutDataSequence”.
get	typeGet	Variables for parameterizing FB13 "GetDataSequence”.
diagnostic	typeDiagnostic	Variable to store status of FBs "PutDataSequence" and "GetDataSequence" in case of error.

3.3.1 PLC Data Type "typePut"

The following table shows the structure of "typePut” data type.

Table 3-2

Parameter	Data type	Start value	Description
execute	Bool	false	Control parameter for FB12 "PutDataSequence".
connectionId	Word	16#0	Addressing parameter for specifying the connection to the communication partner.
done	Bool	false	Status parameters
busy	Bool	false	Status parameters
error	Bool	false	Status parameters
status	Word	16#0	Status parameters

3.3.2 PLC Data Type "typeGet"

The following table shows the structure of the "typeGet" data type.

Table 3-3

Parameter	Data type	Start value	Description
execute	Bool	false	Control parameter for FB13 "GetDataSequence".
connectionId	Word	16#0	Addressing parameter for specifying the connection to the communication partner.
done	Bool	false	Status parameters
busy	Bool	false	Status parameters
error	Bool	false	Status parameters
status	Word	16#0	Status parameters

3.3.3 PLC Data Type "typeDiagnostic"

The following table shows the structure of the "typeDiagnostic" data type.

Table 3-4

Parameter	Data type	Start value	Description
statusPut	Word	16#0	Parameter to store the status of FB12 "PutDataSequence".
statusGet	Word	16#0	Parameter to store the status of FB13 "GetDataSequence”.

3.4 FB12 "PutDataSequence"

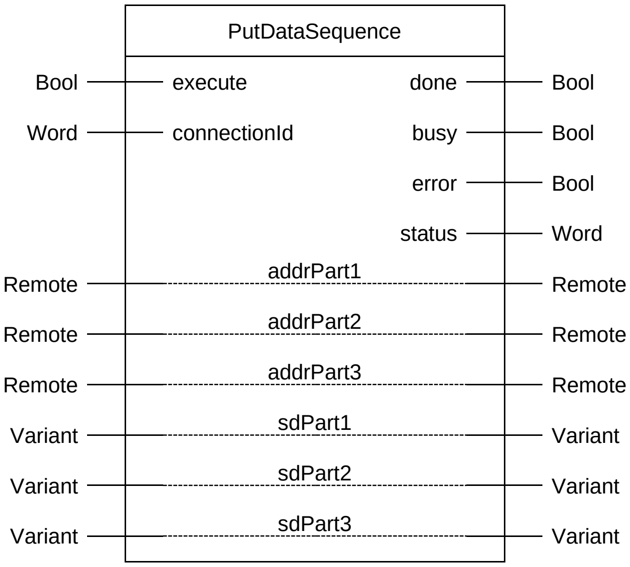
The FB12 "PutDataSequence" calls the "PUT" instruction sequentially to execute the following functions:

- Send up to 400 bytes of data to the partner via the configured S7 connection as soon as the "execute" input recognizes a positive edge. When the Write job is running, it is not possible to trigger a new Write job.
- Output the status of the FB and data transmission at the "status" output.

FB12 "PutDataSequence" is called cyclically in OB1.

The following figure shows the call of the FB12 "PutDataSequence” in OB1.

Figure 3-4



3.4.1 Parameters of FB12 "PutDataSequence"

The following table shows the parameters of FB1 "PutDataSequence.

Table 3-5

Name	P type	Data type	Comment
execute	IN	Bool	Control parameter: Enables the Write job on a rising edge.
connectionId	IN	Word	Local ID: Addressing parameter for specifying the connection to the partner CPU. **Note** The local ID is available in the Properties of the configured S7 connection. The following local IDs are used in this example: • S7-1200 CPU: 100 (hex) • S7-300 CPU: 1 (hex)
done	OUT	Bool	TRUE: The Write job was executed error-free.
busy	OUT	Bool	TRUE: The Write job is running.
error	OUT	Bool	Status parameters "error" and "status": • "error" = 0: - "status" = 0000 (hex): neither warning nor error - "status" <> 0000 (hex): Warning, The "status" parameter provides detailed information. • "error" = 1: An error has occurred. The "status" parameter provides detailed information about the type of error.
status	OUT	Word	
addrPart1	IN_OUT	Remote	Pointer to the area to be written to in the partner CPU (max. 160 bytes).
addrPart2	IN_OUT	Remote	This parameter is preset with the value "NULL”. Here you can define another area to be written to in the partner CPU (max. 160 bytes).
addrPart3	IN_OUT	Remote	This parameter is preset with the value "NULL”. Here you can define another area to be written to in the partner CPU (max. 160 bytes).
sdPart1	IN_OUT	Variant	Pointer to the area in your own S7 CPU that contains the data to be sent (max. 160 bytes).
sdPart2	IN_OUT	Variant	This parameter is preset with the value "NULL”. Here you can define another area in your own S7 CPU that contains the data to be sent (max. 160 bytes).
sdPart3	IN_OUT	Variant	This parameter is preset with the value "NULL”. Here you can define another area in your own S7 CPU that contains the data to be sent (max. 160 bytes).

Assign the variables created in section 3.3 to the parameters of FB12 "PutDataSequence".

The following table shows the assignment of the parameters of the "put" variable to the parameters of FB12 "PutDataSequence".

Table 3-6

Parameters of FB12 "PutDataSequence"	Parameters of the "put" variable	Start value	Note
execute	execute	false	Set the "execute" parameter to the value "TRUE" to start the Write job. The Write job is started on a rising edge. Reset the "execute" parameter to the value "FALSE" if the Write job is completed with "done" = true or "error" = true.
connectionId	connectionId	16#100	Enter the local ID of the connection that you defined during the connection configuration.
done	done	false	–
busy	busy	false	–
error	error	false	–
status	status	16#0	–
addrPart1	–	P#DB1.DBX0.0 BYTE 160	Only absolute addressing is permitted.
addrPart2	–	P#DB1.DBX160.0 BYTE 160	
addrPart3	–	P#DB1.DBX320.0 BYTE 80	
sdPart1	–	P#DB2.DBX0.0 BYTE 160	–
sdPart2	–	P#DB2.DBX160.0 BYTE 160	–
sdPart3	–	P#DB2.DBX320.0 BYTE 80	–

3.5 FB13 "GetDataSequence"

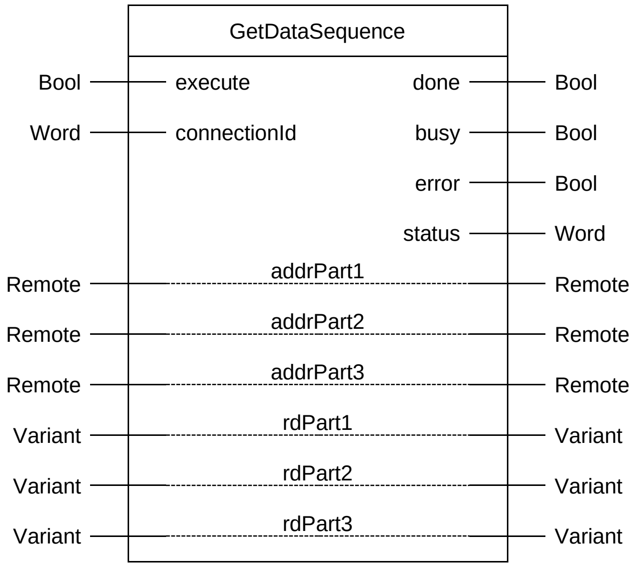
The FB13 "GetDataSequence" calls the "GET" instruction sequentially to execute the following functions:

- Receive up to 400 bytes of data from the partner via the configured S7 connection as soon as the "execute" input recognizes a positive edge. When the Read job is running, it is not possible to trigger a new Read job.
- Output the status of the FB and data transmission at the "status" output.

FB13 "GetDataSequence" is called cyclically in OB1.

The following figure shows the call of the FB13 "GetDataSequence" in OB1.

Figure 3-5



3.5.1 Parameters of FB13 "GetDataSequence"

The following table shows the parameters of FB13 "GetDataSequence”.

Table 3-7

Name	P type	Data type	Comment
execute	IN	Bool	Control parameter: Enables the Read job on a rising edge.
connectionId	IN	Word	Local ID: Addressing parameter for specifying the connection to the partner CPU. **Note** The local ID is available in the Properties of the configured S7 connection. The following local IDs are used in this example: • S7-1200 CPU: 100 (hex) • S7-300 CPU: 1 (hex)
done	OUT	Bool	TRUE: The Read job was executed error-free.
busy	OUT	Bool	TRUE: The read job is running.
error	OUT	Bool	Status parameters "error" and "status": • "error" = 0: - "status" = 0000 (hex): neither warning nor error - "status" <> 0000 (hex): Warning, The "status" parameter provides detailed information. • "error" = 1: An error has occurred. The "status" parameter provides detailed information about the type of error.
status	OUT	Word	
addrPart1	IN_OUT	Remote	Pointer to the area to be read in the partner CPU (max. 160 bytes).
addrPart2	IN_OUT	Remote	This parameter is preset with the value "NULL”. Here you can define another area to be read in the partner CPU (max. 160 bytes).
addrPart3	IN_OUT	Remote	This parameter is preset with the value "NULL”. Here you can define another area to be read in the partner CPU (max. 160 bytes).
rdPart1	IN_OUT	Variant	Pointer to the area in your own S7-CPU in which the read data is stored (max. 160 bytes).
rdPart2	IN_OUT	Variant	This parameter is preset with the value "NULL”. Here you can define another area in your own S7 CPU in which the read data is stored (max. 160 bytes).
rdPart3	IN_OUT	Variant	This parameter is preset with the value "NULL”. Here you can define another area in your own S7 CPU in which the read data is stored (max. 160 bytes).

Assign the variables created in section 3.3 to the parameters of FB13 "GetDataSequence".

The following table shows the assignment of the parameters of the "get" variable to the parameters of FB13 "GetDataSequence".

Table 3-8

Parameters of FB13 "GetDataSequence"	Parameters of the "get" variable	Start value	Note
execute	execute	false	Set the "execute" parameter to the value "TRUE" to start the Read job. The Read job is started on a rising edge. Reset the "execute" parameter to the value "FALSE" if the Read job is completed with "done" = true or "error" = true.
connectionId	connectionId	16#100	Enter the local ID of the connection that you defined during the connection configuration.
done	done	false	–
busy	busy	false	–
error	error	false	–
status	status	16#0	–
addrPart1	–	P#DB2.DBX0.0 BYTE 160	Only absolute addressing is permitted.
addrPart2	–	P#DB2.DBX160.0 BYTE 160	
addrPart3	–	P#DB2.DBX320.0 BYTE 80	
rdPart1	–	P#DB1.DBX0.0 BYTE 160	–
rdPart2	–	P#DB1.DBX160.0 BYTE 160	–
rdPart3	–	P#DB1.DBX320.0 BYTE 80	–

3.6 Function

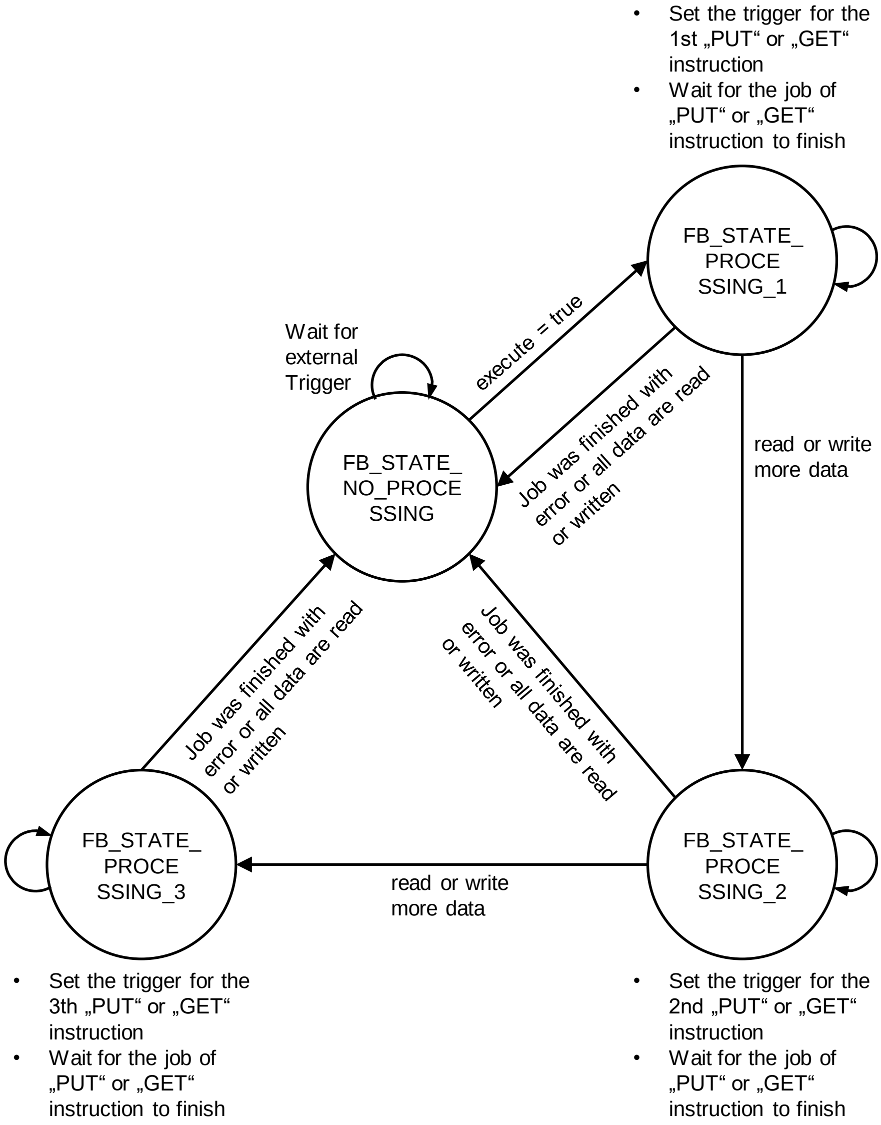
The FBs "PutDataSequence" and "GetDataSequence" are implemented as state machine. The design model of a state machine is particularly suitable for modeling more complex asynchronous processes, such as communication between partners, which can extend over several cycles.

A certain state is run through cyclically until a transition condition is fulfilled and the machine switches to the next subsequent state. This not only improves the clarity compared to conventional link control, but also makes it easier to find any errors in the program logic more quickly.

3.6.1 Overview

The following figure shows the call of the FBs "PutDataSequence" and "GetDataSequence".

Figure 3-6


• Set the trigger for the 1st „PUT“ or „GET“ instruction
• Wait for the job of „PUT“ or „GET“ instruction to finish
FB_STATE_ PROCE SSING_1
Wait for external Trigger
execute = true
FB_STATE_ NO_PROCE SSING
Job was finished with error or all data are read or written
read or write more data
Job was finished with error or all data are read or written
Job was finished with error or all data are read or written
FB_STATE_ PROCE SSING_3
read or write more data
FB_STATE_ PROCE SSING_2
• Set the trigger for the 3th „PUT“ or „GET“ instruction
• Wait for the job of „PUT“ or „GET“ instruction to finish
• Set the trigger for the 2nd „PUT“ or „GET“ instruction
• Wait for the job of „PUT“ or „GET“ instruction to finish

3.6.2 Description of the States

FB12 "PutDataSequence"

The following table describes the implemented states and the possible transitions of FB12 "PutDataSequence”.

Table 3-9

State	Description	Transition
FB_STATE_NO_ PROCESSING	In this state the FB has the following properties: • No Write job is active. • The value 16#7000 is output at the "status" output of the FB.	The FB switches to the state "FB_STATE_PROCESSING_1” when a positive edge is detected at the "execute" input of the FB.
FB_STATE_ PROCESSING_1	In this state the FB executes the following actions: • The FB starts the first Write job with the "PUT” instruction. • The FB monitors whether the "PUT" instruction has terminated successfully (DONE = true) or with an error (ERROR = true).	The state is exited when the Write job is completed successfully or with an error. • If the Write job is completed successfully, the following actions are performed: - The FB switches to the state "FB_STATE_PROCESSING_2” if more data is to be written. - The FB switches to the state "FB_STATE_NO_ PROCESSING” if the data has been transferred completely. The "done" output is set to the value "true" for one cycle. • If the Write job is completed with an error, the following actions are performed: - The "error" output is set to the value "true" for one cycle. - The status display of the "PUT" instruction is output at the "status" output. - The FB switches to the state "FB_STATE_NO_ PROCESSING"

State	Description	Transition
FB_STATE_ PROCESSING_2	In this state the FB executes the following actions: • The FB starts the second Write job with the "PUT" instruction. • The FB monitors whether the "PUT" instruction has terminated successfully (DONE = true) or with an error (ERROR = true).	The state is exited when the Write job is completed successfully or with an error. • If the Write job is completed successfully, the following actions are performed: - The FB switches to the state "FB_STATE_PROCESSING_3" if more data is to be written. - The FB switches to the state "FB_STATE_NO_ PROCESSING" if the data has been transferred completely. The "done" output is set to the value "true" for one cycle. • If the Write job is completed with an error, the following actions are performed: - The "error" output is set to the value "true" for one cycle. - The status display of the "PUT" instruction is output at the "status" output. - The FB switches to the state "FB_STATE_NO_ PROCESSING"
FB_STATE_ PROCESSING_3	In this state the FB executes the following actions: • The FB starts the third Write job with the "PUT" instruction. • The FB monitors whether the "PUT" instruction has terminated successfully (DONE = true) or with an error (ERROR = true).	The state is exited when the Write job is completed successfully or with an error. • If the Write job is completed successfully, the following actions are performed: - The "done" output is set to the value "true" for one cycle. - The FB switches to the state "FB_STATE_NO_ PROCESSING" • If the Write job is completed with an error, the following actions are performed: - The "error" output is set to the value "true" for one cycle. - The status display of the "PUT" instruction is output at the "status" output. - The FB switches to the state "FB_STATE_NO_ PROCESSING"

FB13 "GetDataSequence"

The following table describes the implemented states and the possible transitions of FB13 "GetDataSequence".

Table 3-10

State	Description	Transition
FB_STATE_NO_ PROCESSING	In this state the FB has the following properties: • No Read job is active. • The value 16#7000 is output at the "status" output of the FB.	The FB switches to the state "FB_STATE_PROCESSING_1" when a positive edge is detected at the "execute" input of the FB.
FB_STATE_ PROCESSING_1	In this state the FB executes the following actions: • The FB starts the first Read job with the "GET" instruction. • The FB monitors whether the "GET" instruction has terminated successfully (NDR = true) or with an error (ERROR = true).	The state is exited when the Read job is completed successfully or with an error. • If the Read job is completed successfully, the following actions are performed: - The FB switches to the state "FB_STATE_PROCESSING_2" if more data is to be read. - The FB switches to the state "FB_STATE_NO_ PROCESSING" if the data has been transferred completely. The "done" output is set to the value "true" for one cycle. • If the Read job is completed with an error, the following actions are performed: - The "error" output is set to the value "true" for one cycle. - The status display of the "GET" instruction is output at the "status" output. - The FB switches to the state "FB_STATE_NO_ PROCESSING"

State	Description	Transition
FB_STATE_ PROCESSING_2	In this state the FB executes the following actions: • The FB starts the second Read job with the "GET" instruction. • The FB monitors whether the "GET" instruction has terminated successfully (NDR = true) or with an error (ERROR = true).	The state is exited when the Read job is completed successfully or with an error. • If the Read job is completed successfully, the following actions are performed: - The FB switches to the state "FB_STATE_PROCESSING_3" if more data is to be read. - The FB switches to the state "FB_STATE_NO_ PROCESSING" if the data has been transferred completely. The "done" output is set to the value "true" for one cycle. • If the Read job is completed with an error, the following actions are performed: - The "error" output is set to the value "true" for one cycle. - The status display of the "GET" instruction is output at the "status" output. - The FB switches to the state "FB_STATE_NO_ PROCESSING"
FB_STATE_ PROCESSING_3	In this state the FB executes the following actions: • The FB starts the third Read job with the "GET" instruction. • The FB monitors whether the "GET" instruction has terminated successfully (NDR = true) or with an error (ERROR = true).	The state is exited when the Read job is completed successfully or with an error. • If the Read job is completed successfully, the following actions are performed: - The "done" output is set to the value "true" for one cycle. - The FB switches to the state "FB_STATE_NO_ PROCESSING" If the Read job is completed with an error, the following actions are performed: • The "error" output is set to the value "true" for one cycle. • The status display of the "GET" instruction is output at the "status" output. • The FB switches to the state "FB_STATE_NO_ PROCESSING"

3.7 Error Handling

3.7.1 FB12 "PutDataSequence"

If an error occurs in FB12 "PutDataSequence”, the cause of the error is written to the "status" output parameter.

Table 3-11

error	status	Description	Remedy
0	16#7000	FB12 "PutDataSequence" is not active.	Status information Enable FB12 "PutDataSequence” by setting the "execute" input to "true".
0	16#7001	FB12 "PutDataSequence" is initialized.	Status information
0	16#7002	Write job is running.	Status information
1	<>16#0000	Status display of the "PUT" instruction.	Detailed information is available in the STEP 7 Online Help or in the following manual: "SIMATIC STEP 7 Basic/Professional V16 and SIMATIC WinCC V16".

3.7.2 FB13 "GetDataSequence"

If an error occurs in FB13 "GetDataSequence”, the cause of the error is written to the "status" output parameter.

Table 3-12

error	status	Description	Remedy
0	16#7000	FB13 "GetDataSequence” is not active.	Status information Enable FB13 "GetDataSequence” by setting the "execute" input to "true".
0	16#7001	FB13 "GetDataSequence” is initialized.	Status information
0	16#7002	Read job is running.	Status information
1	<>16#0000	Status display of the "GET” instruction.	Detailed information is available in the STEP 7 Online Help or in the following manual: "SIMATIC STEP 7 Basic/Professional V16 and SIMATIC WinCC V16".

4 User Program of the Passive S7 CPU

No instructions for data transfer are called in the user program of the passive S7 CPU.

All you need are data blocks in which the sent and received data is stored.

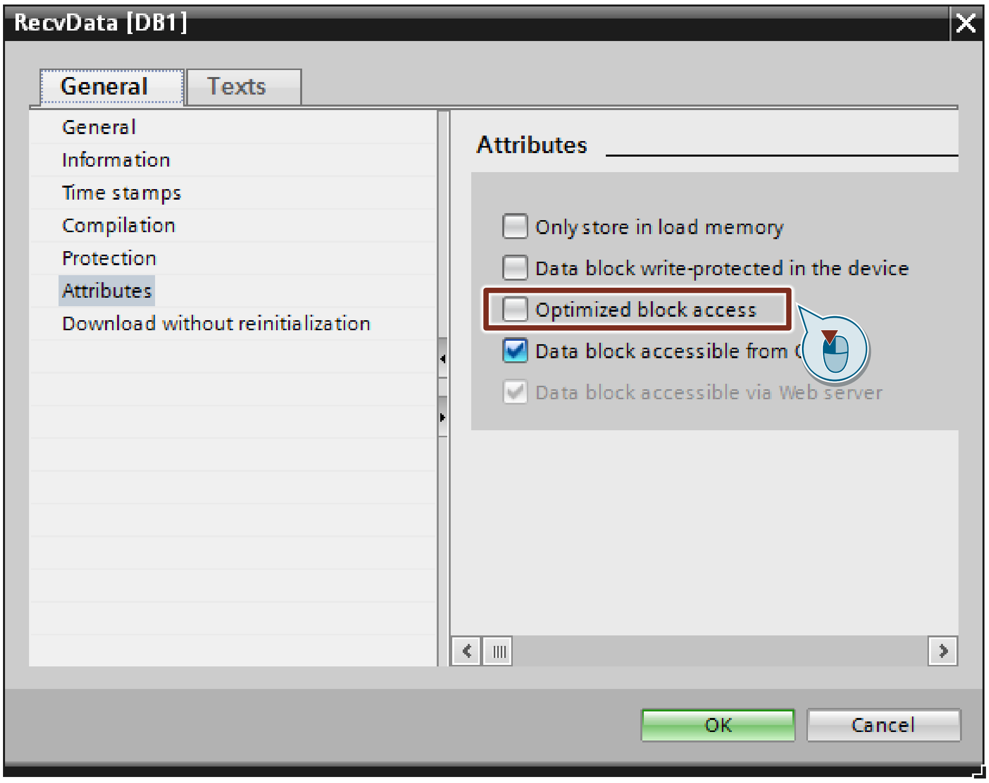
4.1 Create Send and Receive Data Areas

4.1.1 DB1 "RecvData"

The data written by the partner CPU is stored in the data block DB1 "RecvData" of the passive S7 CPU.

In the Properties of the DB1 "RecvData", under "Attributes" you disable the "Optimized block access" function.

Figure 4-1



Attributes

□ Only store in load memory

□ Data block write-protected in the device

□ Optimized block access

☑ Data block accessible from O

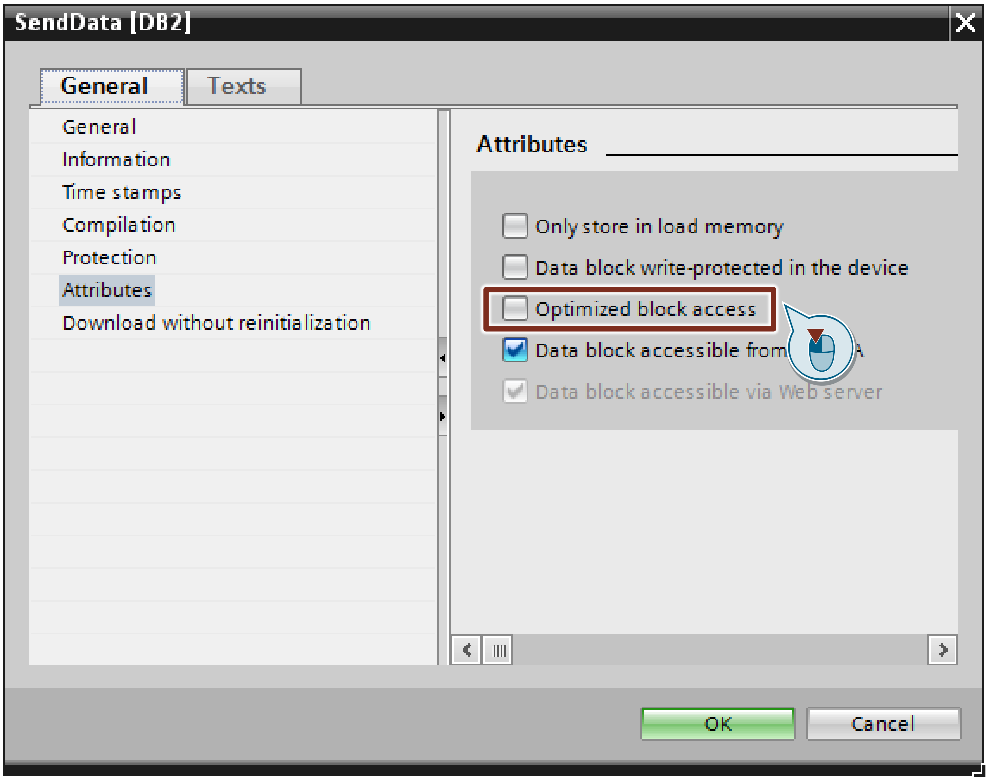
☑ Data block accessible via Web server

4.1.2 DB2 "SendData"

The data that is read by the partner CPU is stored in the data block DB2 "SendData" of the passive S7 CPU.

In the Properties of the DB2 "SendData", under "Attributes" you disable the "Optimized block access" function.

Figure 4-2



Attributes

□ Only store in load memory

□ Data block write-protected in the device

□ Optimized block access

☑ Data block accessible from A

☑ Data block accessible via Web server

5 Information

5.1 "PUT" Instruction

The "PUT" instruction is called in FB12 "PutDataSequence”. This is to be found in the "Instructions" task card under "Communication".

You use the "PUT" instruction to write data to the partner CPU.

Note

This is only possible if the function "Permit access with PUT/GET communication from remote partner" has been enabled for the partner CPU in the Properties of the S7 CPU under "Protection & Security > Connection mechanisms".

With the "PUT" instruction you cannot access blocks that have been created with the "optimized" type of access.

Detailed information about the "PUT" instruction is given in the manual entitled "SIMATIC STEP 7 Basic/Professional V16 and SIMATIC WinCC V16".

5.2 "GET" Instruction

The "GET" instruction is called in FB13 "GetDataSequence”. This is to be found in the "Instructions" task card under "Communication > S7 Communication".

You use the GET instruction to read data from the partner CPU.

Note

This is only possible if the function "Permit access with PUT/GET communication from remote partner" has been enabled for the partner CPU in the Properties of the CPU under "Protection & Security > Connection mechanisms".

With the "GET" instruction you cannot access blocks that have been created with the "optimized" type of access.

Detailed information about the "GET" instruction is given in the manual entitled "SIMATIC STEP 7 Basic/Professional V16 and SIMATIC WinCC V16".

5.3 Configuration of the "PUT" and "GET" Instructions

Note If you use FB "PutDataSequence" and FB "GetDataSequence", you specify the connection ID of the configured S7 connection at the input parameter "connectionId". The connection ID is used internally in the FB on the instructions "PUT" and "GET".

If you call the "PUT" and "GET" instructions manually in your user program, proceed as follows to configure the instructions.

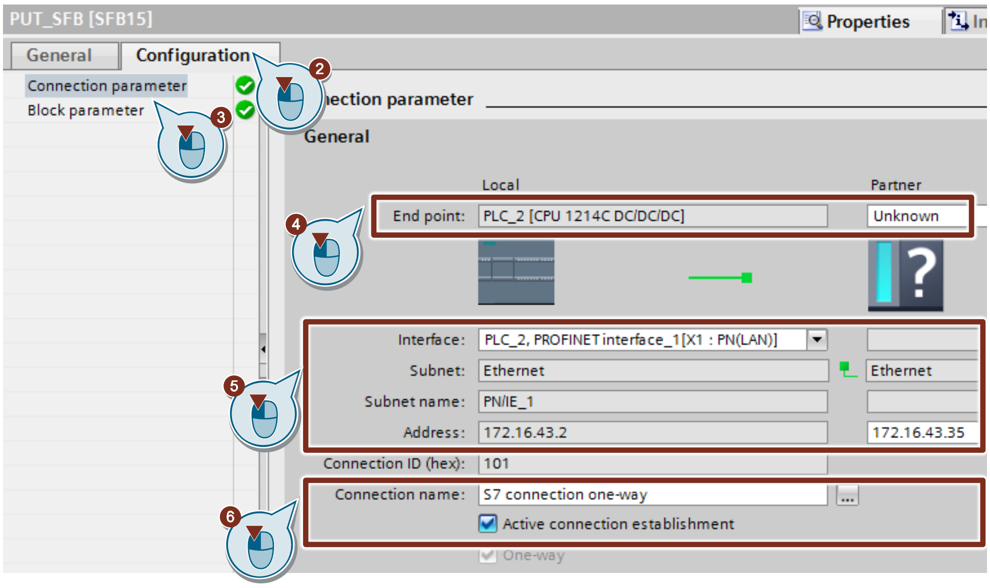
Instructions

You have the option of having the connection ID entered automatically by TIA Portal at the input parameter "ID" of the "PUT" and "GET" instructions.

You configure the "PUT" and "GET" instructions in the inspector window of the program editor. Proceed as instructed below:

1. Mark the call of the "PUT" or "GET" instruction.
2. Open the "Configuration" tab in the inspector window.
3. In the area navigation of the "Configuration" tab you select the "Connection Parameters" group. This group includes the connection parameters.
4. Define the connection endpoints of the S7 connection via which the data will be sent and received.
5. The following parameters of the communication partners are entered automatically once you have defined the connection endpoint.
   - Interface
   - Subnet
   - Subnet name
   - Address
6. Select the name of the S7 connection via which the data is to be transferred.

Figure 5-1



PUT_SFB [SFB15]

Properties | In

General | Configuration

Connection parameter ✓

Block parameter ✓

nection parameter

General

	Local	Partner
End point:	PLC_2 [CPU 1214C DC/DC/DC]	Unknown
Interface:	PLC_2, PROFINET interface_1[X1 : PN(LAN)]	
Subnet:	Ethernet	Ethernet
Subnet name:	PN/IE_1	
Address:	172.16.43.2	172.16.43.35
Connection ID (hex):	101	
Connection name:	S7 connection one-way	

☑ Active connection establishment

☑ One-way