WAGO-I/O-*PRO* 32 Library

SerComm.lib

Contents

SERCOMM 2
SERCOMM_VERSION 7
COM_ACTION (Data type) 9
COM_BAUDRATE (Data type) 10
COM_BYTESIZE (Data type) 12
COM_FLOW_CONTROL (Data type) 13
COM_PARITY (Data type) 14
COM_STOPBITS (Data type) 15

SERCOMM

WAGO-I/O-*PRO* 32 Library elements			
Category:	Communication via the configuration interface of the controller or via the serial interface modules		
Name:	SERCOMM		
Type:	Function	Function block **X**	Program
Library name:	SerComm.lib		
Applicable to:	All programmable fieldbus controllers except MODBUS		

Input parameters:	**Data type:**	**Comments:**
EN	BOOL	FB execution control. The function block is executed on a rising edge at EN.
COMPORT	BYTE	Displays the COM-Port on which one of the possible operations is performed, dependent on the 'FB_ACTION'.
BAUDRATE	COM_ BAUDRATE	Displays the baud rate with which the data is to be transmitted and received.
PARITY	COM_ PARITY	Shows if, and if yes, which parity is to be used.
STOPBITS	COM_ STOPBITS	Indicates the number of stop bits to be used.
BYTESIZE	COM_ BYTESIZE	Displays the number of bits that represent a data byte.
FLOW_CONTROL	COM_FLOW_ CONTROL	Shows if, and if yes, which type of flow control is to be used.
FB_ACTION (as from version 2.0, previously 'ACTION')	COM_ ACTION	Indicates the type of operation that is to be performed on the set COM-Port.
BYTES_TO_DO	DWORD	Indicates the number of bytes to be written or read.
IN_BUFFER	DWORD	Pointer on the buffer the data of which are to be transmitted.
OUT_BUFFER	DWORD	Pointer on the buffer that accepts the data received.

→

Output parameters:	Data type:	Comments:
ENO	BOOL	Shows that an operation on the COM-Port has been completely performed or aborted with an error code (TRUE).
ERROR	BOOL	Indicates that an error has occurred during an operation specified by 'FB_ACTION' (TRUE).
LAST_ERROR	BYTE	If an error is indicated by 'ERROR', the corresponding error code can be read out here.
BYTES_DONE	DWORD	Displays the number of bytes already transmitted during a write operation. With a read operation, the number of bytes read is returned.

Internal parameters:	Data type:	Comments:
INTERNAL_USE_DO_ NOT_MODIFY	ARRAY OF BYTE	Internal array. This array must not be modified by the user program

Header:

```
(* Communication over a serial interface *)
FUNCTION_BLOCK SERCOMM

VAR_INPUT
 EN : BOOL;                          (* Initial = FALSE *)
 COMPORT : BYTE;                     (* Initial = COM1 *)
 BAUDRATE : COM_BAUDRATE;            (* Initial = 19200 Baud *)
 PARITY : COM_PARITY;                (* Initial = even parity *)
 STOPBITS : COM_STOPBITS;            (* Initial = one stopbit *)
 BYTESIZE : COM_BYTESIZE;            (* Initial = 8 Databits *)
 FLOW_CONTROL : COM_FLOW_CONTROL;    (* Initial = No flow control *)

 FB_ACTION : COM_ACTION;             (* Initial = Open *)
 BYTES_TO_DO : DWORD;                (* Initial = 0 *)

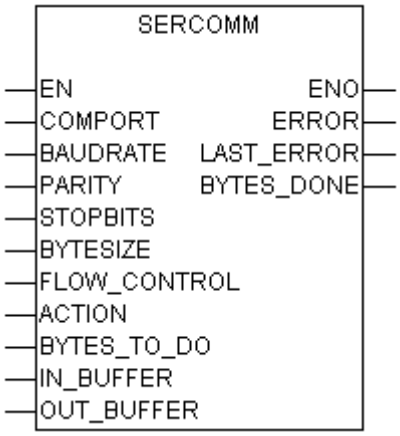
 SEND_BUFFER : DWORD;                (* Address of the send buffer *)
 RECEIVE_BUFFER : DWORD;             (* Address of the receive buffer *)
END_VAR

VAR_OUTPUT
 ENO : BOOL;
 ERROR : BOOL;                       (* Indicates an error *)
 LAST_ERROR : BYTE;                  (* Error code *)
 BYTES_DONE : DWORD;                 (* Number of write/read bytes *)
END_VAR

VAR CONSTANT
 INTERNAL_USE_DO_NOT_MODIFY : ARRAY [0..133] OF BYTE;
END_VAR
```

→

Graphic display:



Time behavior:

Function description:
This function block is intended to transmit and receive data via the controller configuration interface and via connected interface modules of type 750-650/651/653. The controller serial interface is termed COM 1. All connected interface modules of the above mentioned type can be accessed in the order as connected, by COM 2 to COM 65. Only one COM port each can be allocated to each instance of the function block 'SerComm'. In other words, an opened COM port cannot be opened a second time. Furthermore, an operation on the port is only possible with the instance of the FB used for opening that very COM port.
The following is a more detailed description of the different COM ports. COM 1: The serial interface of the controllers has a full duplex capability. This interface supports all FB communication parameters specified. Only the flow control must be deactivated (NO_FLOW_CONTROL) and 'BYTESIZE' must only have the value 'BS_8'. During data transmission, the data is re-copied block by block into an internal buffer of the firmware, from the send buffer assigned by 'SEND_BUFFER'. This means that the contents of the send buffer should not be modified during the transmission process. The variable 'BYTES_DONE' is updated at the time of sending a block. Once all data is sent, 'ENO' assumes the 'TRUE' value. During data receipt, these are written into an internal circular buffer annular buffer of 128 byte size. While receiving further data, the information in the circular buffer is written over, unless data has already been read out of the circular buffer by a read operation. Only when sending data is it possible to set the 'ENO' output to become activated only several cycles later and thereby identifying the completion of the operation. In this case it is **not necessary** to re-poll the FB until 'ENO' is set. If the COM port is closed, in the case of low baud rates, directly following the sending of the data, it might be possible that the last character will be skipped. In such a case, we recommend to wait a couple of cycles until the COM port is closed. If COM port 1 is opened, applications such as **WAGO-I/O-*PRO* 32** or **WAGO-I/O-*CHECK*** can no longer communicate via the configuration interface with the controller. Communication is possible again, once the COM ports has closed. COM 2 - 65: The supported communication parameters depend on the interface module by which this COM port is represented. For details, please refer to the individual data sheets of the bus modules. If a bus module does not assist the temporary setting of communication parameters, the communication parameters of the FB must coincide with those of the bus module, otherwise it is not possible to open the COM port. All four possible operations (OPEN, CLOSE, READ, WRITE) permit setting the 'ENO' output several cycles later. In this case it is **necessary** to poll the FB several times until this output assumes the 'TRUE' value. Compared to 'COM1', a separate action is executed internally for each action (OPEN, CLOSE, READ, WRITE) at 'COM2 – 65'. A status change partially requires the termination of a PLC cycle. Therefore, each instance of the FB must be polled only **once** per cycle and **per action**! After an operation (ENO = TRUE) has been terminated, the internal actions are executed first when a negative edge at the 'EN' input has been recognised. The internal states are reset to the COM port following a successful 'CLOSE' operation, so that an operation not yet terminated (ENO = FALSE) can be aborted. During a repeated poll of the FB to perform an operation, all communication parameters can be modified, except 'ACTION' and 'COMPORT'. Analog to COM 1, the following holds true: the contents of the send buffer should not be modified during transmission. If the number of bytes contained in the input buffer of the module is bigger than the number

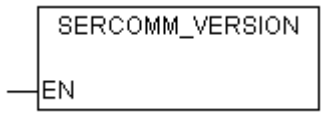
of data bytes to be read, the (maximum) next whole-numbered multiple of the data width in the process image of the module is returned in bytes. This must be taken into account when handling the input buffer.
If an attempt is made to open a COM port simultaneously over two instances, there is a danger of conflict, resulting in one of the two operations being aborted with an error message. **Error codes:** 0x00: No error. 0x01: This library is not supported by the firmware. 0x02: COM port outside of the valid area. 0x03: This function block instance has not yet been assigned a COM port. 0x04: This function block instance has already been assigned a different COM port. 0x05: COM port is already opened. 0x06: COM port is already closed. 0x07: COM port is not opened. 0x08: A write operation is still active (COM1). 0x09: These transfer parameters are not supported by the COM port. 0x0A: The current settings of the bus module could not be read. 0x0B: This library version does not yet support the temporary setting of communication parameters. 0x0C: Bus module could not be initialised. 0x0D: Error when writing data into the FIFO memory of the bus module. 0x0E: The contents of the FIFO memory was not sent (continuous sending). 0x0F: Internal error

SERCOMM_VERSION

WAGO-I/O-*PRO* 32 Library elements			
Category:	Communication via the controller configuration interface or via serial interface modules		
Name:	SERCOMM_VERSION		
Type:	Function **X**	Function block	Program
Library name:	SerComm.lib		
Applicable to:	All programmable fieldbus controllers except MODBUS		
Input parameters:	**Data type:**	**Comments:**	
EN	BOOL	This function is executed on a rising edge at EN.	
Return value:	**Data type:**	**Comments:**	
SERCOMM_VERSION	WORD	Library version	
Header:			

```
FUNCTION SERCOMM_VERSION : WORD
VAR_INPUT
 EN: BOOL;
END_VAR
```

Graphic display:



Time behavior:



→

Function description:

The SERCOMM_VERSION function returns the current version number of the library. This function can be used for information during program development. Furthermore, running time related version conflicts can be avoided.

Version	Description
1.0	First version. The temporary setting of COM parameters is not yet supported.
2.0	The instance variable 'ACTION' of the function block 'SERCOMM' was renamed as follows: 'FB_ACTION'. Use the SerComm.lib of version 2.0 or better as from the WAGO-I/O-*PRO* 32 version 2.2.
2.1	During operation on COM1, the corresponding error code is now sent back when a failure occures. Failure removal for terminal blocks supporting continuous transmission
3.0	The function block "SERCOMM" now is edge triggered
3.1	As far as the communication parameters are concerned, it is possible to use the default-value of the module.
3.2	In the event of a failure, the exact number of the data used will be indicated during a write operation.
4.0	The communication parameters of configurable modules are set when the COM port opens and reset when it closes.
4.1	Activation of several interfaces as the same time no longer leads to resource conflicts and is now possible without any problems.
4.2	It is now possible to send sero bytes via COM1. With open COM port (2-65) the user can no more change the module parameters accidentally by external tools.
5.0	Compatible with the WAGO-I/O-PRO 32 version 2.2 SR5. The library works with the onlinechange feature.
5.1	When using modules that allow setting of parameters, data from the send buffer is no longer resent if the interface is opened or closed.

COM_ACTION (Data type)

WAGO-I/O-*PRO* 32 Library elements			
Category:	Communication via the controller configuration interface or via serial interface modules.		
Name:	COM_ACTION		
Type:	Data type **X**		
Library name:	SerComm.lib		
Applicable to:	SERCOMM		
Elements:	**Value:**	**Comments:**	
OPEN	0	COM port is opened.	
CLOSE	1	COM port is closed.	
READ	2	Read operation on a COM port.	
WRITE	3	Write operation on a COM port.	
Header:			

```
TYPE COM_ACTION :
(
 OPEN  := 0,
 CLOSE := 1,
 READ  := 2,
 WRITE := 3
);
END_TYPE
```

Data type description:
This data type is an enumeration describing the operation being performed on a COM port.

COM_BAUDRATE (Data type)

WAGO-I/O-*PRO* 32 Library elements			
Category:	Communication via the controller configuration interface or via serial interface modules		
Name:	COM_BAUDRATE		
Type:	Data type **X**		
Library name:	SerComm.lib		
Applicable to:	SERCOMM		
Elements:	**Value:**	**Comments:**	
BAUD_110	11	110 bit/s	
BAUD_300	30	300 bit/s	
BAUD_600	60	600 bit/s	
BAUD_1200	120	1200 bit/s	
BAUD_2400	240	2400 bit/s	
BAUD_4800	480	4800 bit/s	
BAUD_9600	960	9600 bit/s	
BAUD_12000	1200	12000 bit/s	
BAUD_14400	1440	14400 bit/s	
BAUD_19200	1920	19200 bit/s	
BAUD_28800	2880	28800 bit/s	
BAUD_38400	3840	38400 bit/s	
BAUD_57600	5760	57600 bit/s	
BAUD_62500	6250	62500 bit/s	
BAUDRATE_ TERMINAL_DEFAULT	16#FFFF	Default-value of the module	
			→

Header:

```
TYPE COM_BAUDRATE :
(
 BAUD_110   :=   11,
 BAUD_300   :=   30,
 BAUD_600   :=   60,
 BAUD_1200  :=  120,
 BAUD_2400  :=  240,
 BAUD_4800  :=  480,
 BAUD_9600  :=  960,
 BAUD_12000 := 1200,
 BAUD_14400 := 1440,
 BAUD_19200 := 1920,
 BAUD_28800 := 2880,
 BAUD_38400 := 3840,
 BAUD_57600 := 5760,
 BAUD_62500 := 6250,
 BAUDRATE_TERMINAL_DEFAULT := 16#FFFF
);
END_TYPE
```

Function description:

This data type is an enumeration describing the number of bits per second with which data is sent or received.

COM_BYTESIZE (Data type)

WAGO-I/O-*PRO* 32 Library elements			
Category:	Communication via the controller configuration interface or via serial interface modules		
Name:	COM_BYTESIZE		
Type:	Data type **X**		
Library name:	SerComm.lib		
Applicable to:	SERCOMM		
Elements:	**Value:**	**Comments:**	
BS_7	7	7 Data bits per byte	
BS_8	8	8 Data bits per byte	
BYTESIZE_ TERMINAL_DEFAULT	16#FFFF	Default-value of the module	
Header:			

```
TYPE COM_BYTESIZE :
(
 BS_7 :=7,
 BS_8 :=8,
 BYTESIZE_TERMINAL_DEFAULT : =16#FFFF
);
END_TYPE
```

Data type description:
This data type is an enumeration describing the number of bits by which a data byte is represented.

COM_FLOW_CONTROL (Data type)

WAGO-I/O-*PRO* 32 Library elements			
Category:	Communication via the controller configuration interface or via serial interface modules		
Name:	COM_FLOW_CONTROL		
Type:	Data type **X**		
Library name:	SerComm.lib		
Applicable to:	SERCOMM		
Elements:	**Value:**	**Comments:**	
NO_FLOW_CONTROL	0	No flow control	
XON_XOFF	1	Software flow control	
RTS_CTS	2	Hardware handshake	
FULLDUPLEX	3	RS485	
HALFDUPLEX	4	RS485	
FLOW_CONTROL_ TERMINAL_DEFAULT	16#FFFF	Default-value of the module	
Header:			

```
TYPE COM_FLOW_CONTROL :
(
 NO_FLOW_CONTROL :=0,
 XON_XOFF :=1,
 RTS_CTS :=2,
 FULLDUPLEX :=3,
 HALFDUPLEX :=4,
 FLOW_CONTROL_TERMINAL_DEFAULT :=16#FFFF
);
END_TYPE
```

Data type description:
This data type is an enumeration describing whether, and if yes, which type of flow control is supported.

COM_PARITY (Data type)

WAGO-I/O-*PRO* 32 Library elements			
Category:	Communication via the controller configuration interface or via serial interface modules		
Name:	COM_PARITY		
Type:	Data type **X**		
Library name:	SerComm.lib		
Applicable to:	SERCOMM		
Elements:	**Value:**	**Comments:**	
PARITY_NO	0	No parity bit	
PARITY_ODD	1	The parity bit is set with an even number of bits in the data byte.	
PARITY_EVEN	2	The parity bit is set with an odd number of bits in the data byte.	
PARITY_ TERMINAL_DEFAULT	16#FFFF	Default-value of the module	
Header:			

```
TYPE COM_PARITY :
(
 PARITY_NO:=0,
 PARITY_ODD:=1,
 PARITY_EVEN:=2,
 PARITY_TERMINAL_DEFAULT:=16#FFFF
);
END_TYPE
```

Data type description:
This data type is an enumeration indicating whether a parity bit is inserted.

COM_STOPBITS (Data type)

WAGO-I/O-*PRO* 32 Library elements			
Category:	Communication via the controller configuration interface or via serial interface modules		
Name:	COM_STOPBITS		
Type:	Data type **X**		
Library name:	SerComm.lib		
Applicable to:	SERCOMM		
Elements:	**Value:**	**Comments:**	
STOPBITS_1	1	1 Stop bit per data byte	
STOPBITS_2	2	2 Stop bits per data byte	
STOPBITS_ TERMINAL_DEFAULT	16#FFFF	Default-value of the module	
Header:			

```
TYPE COM_STOPBITS :
(
 STOPBITS_1:=1,
 STOPBITS_2:=2,
 STOPBITS_TERMINAL_DEFAULT:=16#FFFF
);
END_TYPE
```

Data type description:
This data type is an enumeration indicating the number of stop bits to be inserted after each data byte.