GFK-2992C

October 2017

Serial Communication Module EP-5261
Function Block FB_RSTI_EP_5261
Handling Function Block FB_MBM_RTU_Master

Serial Communication Module EP-5261

The RSTi-EP Serial Communication (EP-5261) module provides extended communication options. For example, devices (such as barcode scanners or printers) can be integrated consistently in RSTi-EP systems using an RS-232, RS-485, or RS-422 interface. The EP-5261 module presents a solution for connecting the control cabinet to the field.

The data transfer rate can be parameterised between 300 and 11,5200 bps. The process data length can be parameterised to be 8 or 16 byte. A terminating resistor can be parameterised for the RS-485 and RS-422 interface respectively.

The communication status is indicated by two LEDs on the respective plug.

The module electronics supply the connected data terminal device with power from the input current path (I_{IN}) either with 5 V dc or 24 V dc. Both supply voltage outputs are protected against over-current.

The module features a type plate, which includes identification information, the key technical specifications, and a block diagram. Additionally, a QR code allows for direct online access to the associated documentation. The software for reading the QR code must support inverted QR codes.

Markers are available as accessories for labelling the equipment. Each I/O module can be labelled to ensure clear identification when replacing individual modules or electronic units.

The RSTi-EP station is usually installed on a horizontally positioned DIN-rail. Installation on vertically positioned DIN rails is also possible.

Modules should to be allowed to de-energize for a minimum 10 seconds after power down, prior to starting any maintenance activity.

Refer to the *RSTi-EP Slice I/O User Manual* (GFK-2958) for additional information.

For power-feed (a software utility available on PME V9.00) requirements, refer to the *RSTi-EP Power Supply Reference Guide*.

Serial Communication Module EP-5261

EP-5261 Module Features

- *Spring style* technology for ease of wiring
- DIN-rail mounted
- Double-click installation for positive indication of correct installation
- Supports indirect firmware update through the network monitor
- Supports replacement without shutting down the system

Ordering Information

Module	Description
EP-5261	RSTi-EP Slice I/O Serial Communication module

Specifications

Item	EP-5261 Specification
System Data	
Data	Process, parameter, and diagnostic data depend on the network adapter used
Interface	RSTi-EP system bus
System bus transfer rate	48 Mbps
Serial Interfaces	
Number	1
Type	RS-232, RS-485, RS-422, parameterisable
Transfer Rate	300 – 11,5200 Bps, parameterisable
Supply voltage	5 V dc or 24 V dc
Current of power supply output	500 mA max
Standards RS232	DIN 66020, DIN 66259, EIA-RS232C, CCITT V.24 / V.28
Standards RS485/RS422	DIN 66259 part 1 and 3, EIA-RS422/485, CCITT V.11
Terminating resistor RS485/RS422	120 Ω, parameterisable
Short-circuit-proof	Yes
Module diagnosis	Yes
Individual channel diagnosis	Yes
Supply	
Supply voltage	20.4 V – 28.8 V
Current consumption from system current path I_{SYS}	8 mA
Current consumption from input current path I_{IN}	16 mA + Load
General Data	
Operating temperature	-20 to +60°C (-4 to +140 °F)
Storage temperature	-40 to +85°C (-40 to +185 °F)
Air humidity (operation/transport)	5 to 95%, noncondensing as per IEC 61131-2
Width	11.5 mm (0.45 in)
Depth	76 mm (2.99 in)
Height	120 mm (4.72 in)
Weight	92 g (3.25 oz)

Current Demand for Analog Output Modules

Product	I_{SYS}	I_{IN}	I_{OUT}	I_S	I_L
EP-5261	8 mA	16 mA + Load	--	--	--

I_{SYS}	Current consumption from the system current path
I_{IN}	Power consumption from input current path
I_{OUT}	Power consumption from output current path
I_S	Current demand of the connected sensors
I_L	Current demand of the connected actuators
x	Must be included when calculating the power supply

LEDs

LED	EP-5261 Status
Module Status	**Green:** Communication over the system bus **Red:** Collective error diagnostic
1.1	**Yellow:** RS-232 parameterised **Yellow flashing:** Data is being received
1.2	**Yellow:** RS-232 parameterised **Yellow flashing:** Data is being transmitted
1.3	--
1.4	--
2.1	--
2.2	--
2.3	--
2.4	--
3.1	3.1 - 3.4 **Yellow:** RS-422 parameterised 3.1 + 3.2 **Off**, 3.3 + 3.4 **Yellow:** RS-485 parameterised
3.2	
3.3	**Yellow flashing:** Data are being recieved
3.4	**Yellow flashing:** Data are being transmitted
4.1	**Green:** Supply voltage +5 V dc
4.2	--
4.3	**Green:** Supply voltage +24 V dc
4.4	--

Overview of Editable Parameters

Description	Options[1]	Default
Operating mode	Disabled (0) / RS232 (1) / RS485 (2) / RS422 (3)	Disabled
Data bits[2]	7 Bit (0) / 8 Bit (1)	8 Bit
Baud rate	300 (0) / 600 (1) / 1200 (2) / 2400 (3) / 4800 (4) / 9600 (5) / 14400 (6) / 19200 (7) / 28800 (8) / 38400 (9) / 57600 (10) / 115200 (11)	9600
Stop bit	1 Bit (0) / 2 Bit (1)	1 Bit
Parity	None (0) / Even (1) / Odd (2)	None
Flow control	None (0) / CTS/RTS (1) / XON/XOFF (2)	None
XON character	0 … 255	17
XOFF character	0 … 255	19
Terminating resistor RS-485/422	Off (0) / On (1)	Off
Process data length	16 Byte (1)	16 Byte
1) Values in brackets for EtherCAT and Modbus-TCP 2) Option *7 Bit* works only in combination with a parity (*even* or *odd*)		

Diagnostic Data

Name	Bytes	Bit	Description	Default
Error indicator	0	0	Module error	
		1	Internal error	
		2	External error	
		3	Channel error	0
		4	Error	
		5	Reserved	0
		6	Reserved	0
		7	Parameter error	
Module type	1	0	Module Type 0x05	1
		1		0
		2		1
		3		0
		4	Reserved	0
		5	Reserved	0
		6	Reserved	0
		7	Reserved	0
Error byte 2	2	0-7	Reserved	0
Error byte 3	3	0-2	Reserved	0
		3	Internal diagnostic FIFO full	0
		4-7	Reserved	0
Channel type	4	0	Channel type 0x79	1
		1		0
		2		0
		3		1
		4		1
		5		1
		6		1
		7		0
Diagnostic bits per channel	5		Number of diagnostic bit per channel	0
Number of channels	6		Number of similar channels per module	1
Channel error	7-10	0-31	Reserved	0
Time stamp	43-46		Time stamp [µs] (32 bit)	

Data Transfer

The process data length can be parameterized to be 8 or 16 Bytes. Byte 0 is used for status and diagnosis, Byte 1 for the data segment length, and the remaining 6 or 14 Bytes are user data.

Process input data: The data sent from the serial device are written into the receive memory of the module. As soon as the SPS request results in that RX_CNT is not equal RX_CNT_ACK, the data will be sent in segments via the fieldbus coupler to the PLC. The successfully data transfer will be acknowledged to the module.

The receive memory can safe a maximum of 255 Bytes. A software handshake (XON/XOFF) or a hardware handshake (RTS/CTS) can be parameterised using the flow control, so that an alarm will warn against a buffer overflow.

Process output data: The data sent from the PLC via the fieldbus coupler are written into the transmission memory of the module. The module is continously checking if data is ready to be sent or a data transfer to the device has completed successfully. Not until then will the next data will be transferred.

Process Input Data

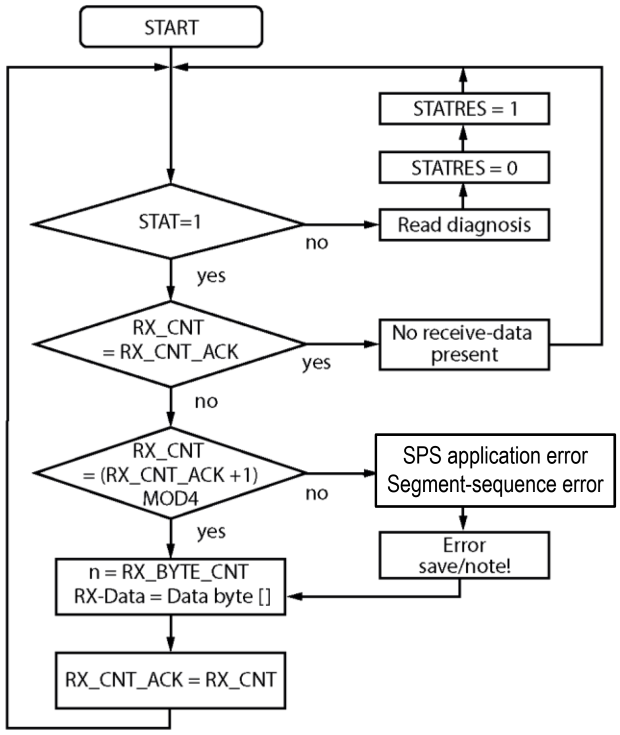
Byte	Format	Name	Bit	Description	Notes
IB0	Word	Status and Diagnosis	IX0.0	Data in the receive memory	RX = 0: Receive memory is empty RX = 1: A telegram or telegram segment in the receive memory is ready for transmission.
			IX0.1	Receive memory nearly full	Only 10 characters are left in the receive memory. XOFF will be set if parametrerised.
			IX0.2	Not used	
			IX0.3	RX_CNT	The RX_CNT value is assigned to each data segment of the process input data while transmission. The sequence or the RX_CNT values is: Binary: 00, 01, 10, 11, 00, ... Decimal: 0, 1, 2, 3, 0, ... A faulty data seqence indicates missing data segments.
			IX0.4	RX_CNT	
			IX0.5	TX_CNT_ACK	The TX_CNT_ACK value is a copy of the TX_CNT value, which has been transferred together with the last data segment of the process output data. TX_CNT_ACK acknowledges that the data has been taken over successfully.
			IX0.6	TX_CNT_ACK	
			IX0.7	STAT	STAT = 1: Communication with the device is without fault. STAT = 0: Faulty communication with the device.
IB1	Word	Length of the data segment / of the subsequent diagnosis data		RX	Length of the data / diagnosis data in this frame
IB 2 ... IB 7 or IB 2 ... IB 15		Received data		User data of the transferred telegram segment	

Process Output Data

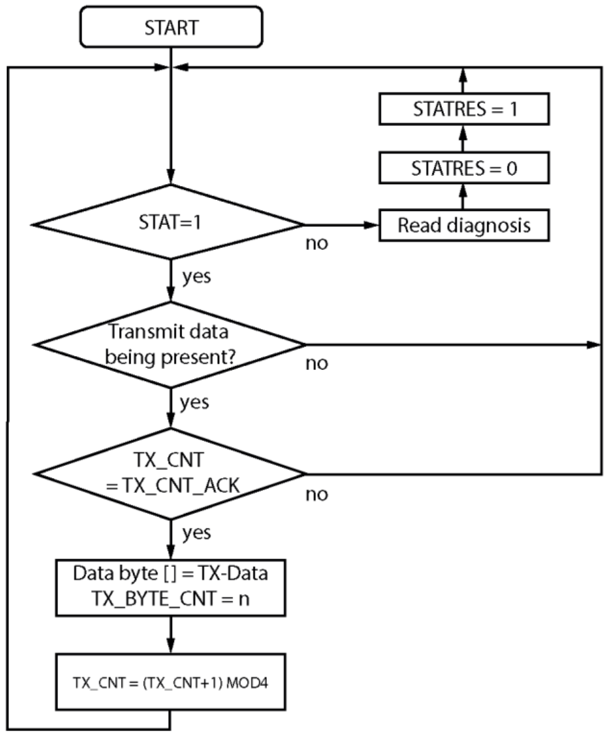
Byte	Format	Name	Bit	Description	Remarks
QB0	Word	Status and Diagnosis	IX0.0	RXBUF FLUSH	Bit 0: RXBUF FLUSH The receive memory can be scrubbed using this bit. STATRES = 1: A requirement with RXBUF FLUSH = 1 will be ignored. STATRES = 0: The receive memory will be scrubbed with RXBUF FLUSH = 1.
			IX0.1	TXBUF FLUSH	Bit 1: TXBUF FLUSH The emission memory can be scrubbed using this bit. STATRES = 1: A requirement with TXBUF FLUSH = 1 will be ignored. STATRES = 0: The emission memory will be scrubbed with TXBUF FLUSH = 1.
			IX0.2	RX_HWBUFFER	Bit 2: DisableSend_TX_HWBUFFER This bit controls the hardware emission memory: DisableSend_TX_HWBUFFER = 0: The hardware emission memory is released. A character (Byte) will be sent as soon as it reaches the buffer. DisableSend_TX_HWBUFFER = 1: The hardware emission memory is locked. Characters (Bytes) will only be sent, when DisableSend_TX_HWBUFFER is set to 0 again.
			IX0.3	TX_CNT	The TX_CNT value is assigned to each data segment of the process output data. The sequence or the TX_CNT values is: Binary: 00–>01–>10–>11–>00... Decimal: 0–>1–>2–>3–>0... A faulty data sequence indicates missing data segments.
			IX0.4	TX_CNT	
			IX0.5	RX_CNT_ACK	RX_CNT_ACK must include a copy of the RX_CNT value. The RX_CNT value has been transferred together with the last data segment of the process input data. RX_CNT_ACK must be set in analogy with RX_CNT (in the status byte). It indicates that the data segment has been transferred successfully by using RX_CNT and enables to receive new data.
			IX0.6	RX_CNT_ACK	
			IX0.7	Communication Status	The input data status bit STAT will be reset using this bit. When changing from 1 to 0 (falling edge) STAT will be reset from 0 to 1. STAT = 0: All changes in the data fields TX_BYTE_CNT, TX_CNT and RX_CNT_ACK will be ignored. The receive or emission memory can be scrubbed using RXBUF FLUSH or TXBUF FLUSH respectively. STAT = 1 or changing from 0 to 1: The buffers cannot be scrubbed.
QB1	Word	Length of the data segment			
QB 2 ... QB 7 or QB 2 ... QB 15		Transmission data		User data of the transferred telegram segment	

Enabling the Data Transfer

There are ways to announce the communication module to the control. Using the test mode, you only copy the input data into the output data of the module so the received data will be sent again. Or, select one of the function blocks provided by your engineering tool. For programming, refer to the following schemes showing the sequences for receiving and transmission.



Scheme of the Receiving Sequence



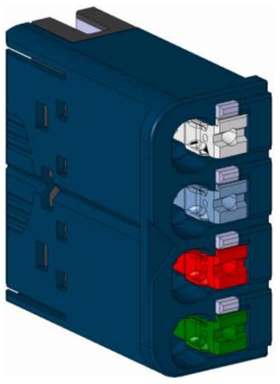
Scheme of the Transmission Sequence

The status and control word values during various states of communication are provided in the following table.

Action	Input Byte 0 (Status) of Module								Input Byte 1 (length of RX byte seg.)	Output byte 0 (control) off the module								Output byte 1 (length of TX byte seg.)	Notes
	7	6	5	4	3	2	1	0		7	6	5	4	3	2	1	0		
	Stat	TX_CNT_ACK		RX_CNT						STATRES	RX_CNT_ACK		TX_CNT						
Init/ Startup	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	After powerup, module is ready for communi-cation
Activate communi-cation	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	PLC is ready for communi-cation (response)
Receive data	1	0	0	0	1	0	0	0	N (1 ..14)	1	0	0	0	0	0	0	0	0	Module has received bytes
	1	0	0	0	1	0	0	0	N	1	0	1	0	0	0	0	0	0	RX acknow-ledge after data taken over
Send data	1	0	0	0	1	0	0	0	X	1	0	1	0	1	0	0	0	N (1..14)	Before changing TX-CNT, set TX bytes
	1	0	1	0	1	0	0	0	X	1	0	1	0	1	0	0	0	N	TX acknow-ledge after module sent data

Field Wiring

The connection frame can take up to four connectors (shown in the following figure), and four wires can be connected to each connector. The *Spring style* technology allows either finely stranded or solid wire conductors with crimped wire-end ferrules or ultrasonically welded wires, each with a maximum cross-section of 1.5 mm² (16 guage), to be inserted easily through the opening in the clamping terminal without having to use tools. To insert fine stranded wires without wire-end ferrules, the pusher must be pressed in with a screwdriver and released to latch the wire.



Connector Block with Four Wire Connectors

Connector Specifications:

- Conductor cross-section 0.14 to 1.5 mm² (26 – 16 guage)
- 10 A max amperes
- 4-pole

The modules do not have a fused sensor/activator power supply. All cables to the connected sensors/actuators must be fused corresponding to their conductor cross-sections (as per Standard DIN EN 60204-1, section 12).

Connection Cables for the Serial Device

Use shielded cables, because electromagnetic interferences from the surroundings have to be assumed. The maximum permissable cable length depends on the cable capacitance and the baud rate.

Connecting a RS-232 Device

Maximum Cable Length RS-232

Cable Capacitance	Maximum Cable Length
≤ 2500 pF	15 m (49 ft), shielded
55 pF/m	45 m (147 ft)

Connecting RS-485 or RS-422 Device

The serial device must be connected using a twisted pair cable (U/UTP, Type Cat- 3 or J-2YY-2x2x0,6).

Maximum Cable Length RS-422/485

Baud Rate in kbps	Maximum Cable Length
≤ 19200	1200 m (3937 ft), shielded
38400	500 m (1640 ft)
57600	250 m (820 ft)
115200	200 m (656 ft)

- RS-485: Use one core pair for Data+/Data–; use any wire for the ground singal GND COM. The remaining free wires should be connected to ground.
- RS-422: Connect the wires for transmitting signals TXD+/TXD– and those for receiving signals RXD+/RXD– in pairs respectively. Use any wire for the ground singal GND COM. The remaining free wires should be connected to ground.

Refer to the *RSTi-EP Slice I/O User Manual* (GFK-2958) for additional information.
Technical assistance is available at http://support.ge-ip.com.

Installation in Hazardous Areas

- EQUIPMENT LABELED WITH REFERENCE TO CLASS I, GROUPS A, B, C & D, DIV. 2 HAZARDOUS AREAS IS SUITABLE FOR USE IN CLASS I, DIVISION 2, GROUPS A, B, C, D OR NON-HAZARDOUS AREAS ONLY

WARNING - EXPLOSION HAZARD - SUBSTITUTION OF COMPONENTS MAY IMPAIR SUITABILITY FOR CLASS I, DIVISION 2;

WARNING - EXPLOSION HAZARD - WHEN IN HAZARDOUS AREAS, TURN OFF POWER BEFORE REPLACING OR WIRING MODULES; AND

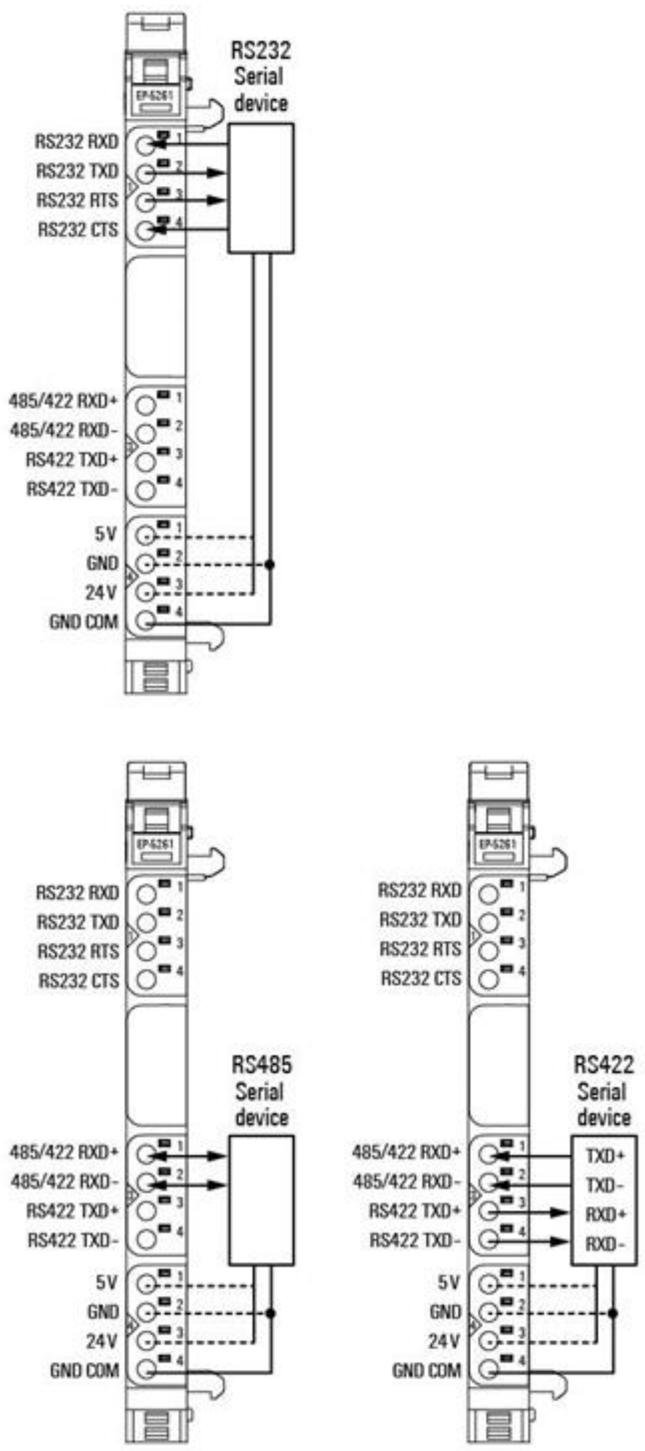
WARNING - EXPLOSION HAZARD - DO NOT CONNECT OR DISCONNECT EQUIPMENT UNLESS POWER HAS BEEN SWITCHED OFF OR THE AREA IS KNOWN TO BE NONHAZARDOUS.

ATEX Marking

II 3 G Ex nA IIC T4 Gc

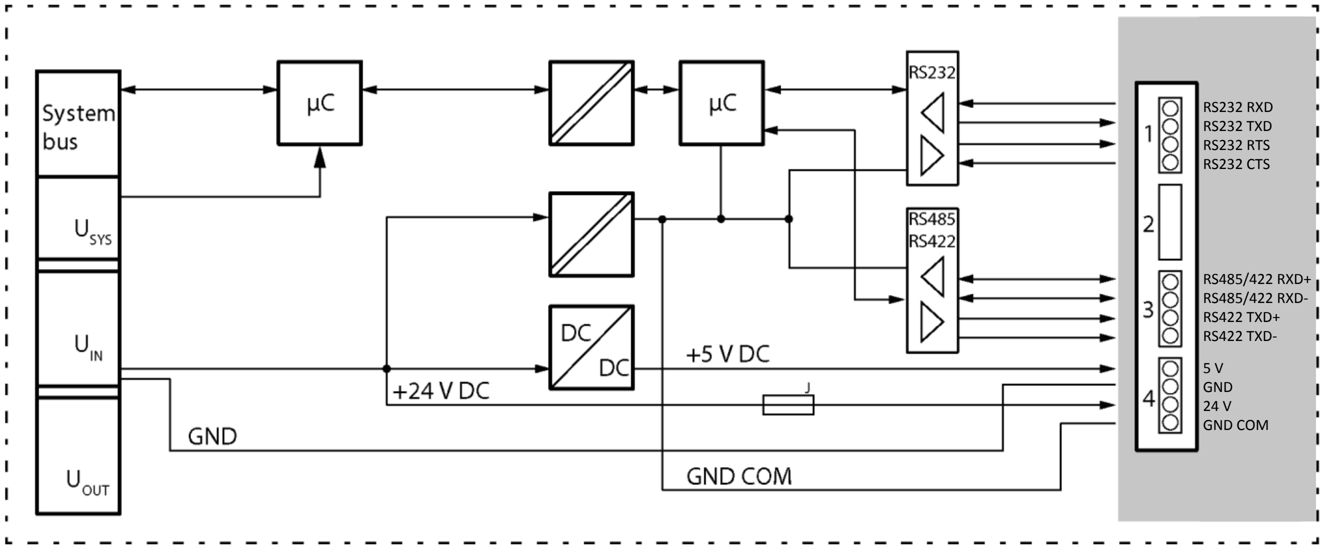
Ta: -20 to +60°C (-4 to +140 °F)

Connection Diagrams



EP-5261 Module Connections

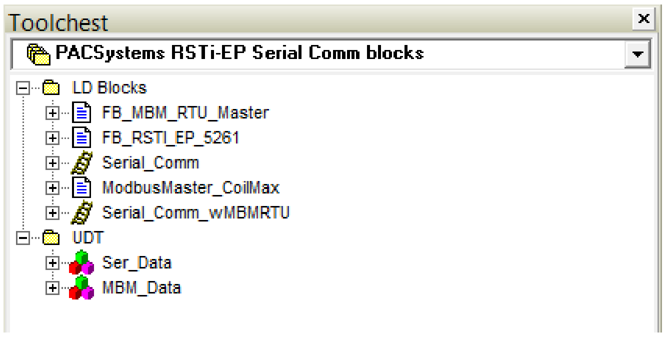
Connection Block Diagrams



EP-5261 Block Diagram

Function Blocks for Enabling Serial Communication

The serial communication module uses 8 registers of input memory and 8 registers of output memory to send and receive the data to any serial device. Function blocks are provided as part of the toolchest drawer to enable faster implementation of serial communication with less efforts on application development.



Toolchest Structure

Note: The toolchest drawer is available for download from the support site at http://geautomation.com.

Note: The import of the Toolchest drawer (RSTi-EP xxxx.ZDRW) in Proficy Machine Edition project, throws an warning message – “Missing blocks MODtimeOut and Inst_FB_EP5261”. Ignore this warning and contimue to import the blocks.

The function block FB_RSTI_EP_5261 is used to send and receive byte strings of length upto 400 bytes. The function block can send 400 bytes of data and receive 400 bytes of data. The length of the data to be received and sent is determined by the input parameter of the function block.

The function block FB_MBM_RTU_Master is used to send and receive Modbus messages with the support of the FB_RSTI_EP_5261 function block. The RTU master function block reads the user inputs and creates Modbus messages. Tthe byte string is then input to the function block FB_RSTI_EP_5261 which in turn outputs the bytes on the serial bus. The response is received by FB_RSTI_EP_5261 and passed as input to the RTU master to parse the response and update the data to internal controller memory.

The function block ModbusMaster_CoilMax is internally used by the RTU master function block.

Note: Refer to the respective function block section in this document for more information on input and output parameters of the serial and RTU master function blocks.

The blocks *Serial_Comm* and *Serial_Comm_wMBMRTU* are pre-defined blocks that call to FB_RSTI_EP_5261 and FB_MBM_RTU_Master. Additionally, the symbolic variables are assigned to the inputs and outputs of the function blocks FB_RSTI_EP_5261 and FB_MBM_RTU_Master.

➢ **To invoke serial and RTU master functions blocks**

If only serial communication is required for the application, perform the following steps:

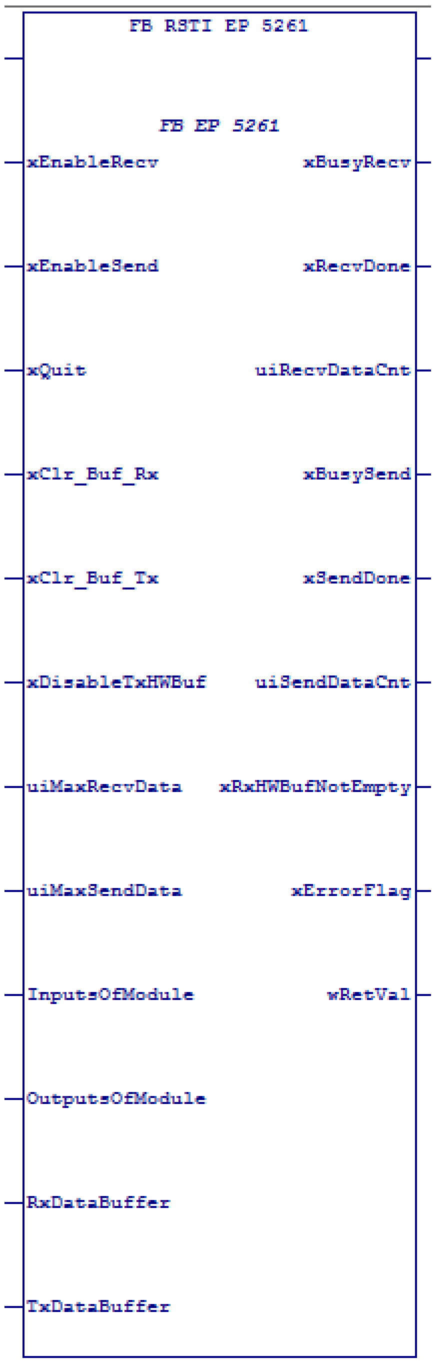
1. Import the UDT *Ser_Data* to the controller target.
2. Import the block *FB_RSTI_EP_5261* to the controller target.
3. Import the block *Serial_Comm*.
4. Call the block *Serial_Comm* in the application logic.

If Modbus RTU communication is to be used,

1. Import the UDT’s *Ser_Data* and *MBM_Data* into the controller target.
2. Import the blocks in the following sequence as follows:
    a. ModbusMaster_CoilMax
    b. FB_MBM_RTU_Master

   c. FB_RSTI_EP_5261
   d. Serial_Comm_wMBMRTU
3. Call the block *Serial_Comm_wMBMRTU* in the application logic.

Function Block FB_RSTI_EP_5261

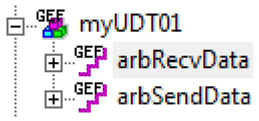


Function Block FB_RSTi_EP-5261

The RSTi-EP Function Block (FB_RSTI_EP_5261) can be used for the data handling of the serial interface module RSTi EP-5261 (1 Channel Serial Communications, RS-232, 422, 485). It supports simultaneous transmitting and receiving of data (full duplex mode). For example, using RS-232 mode is possible. As only the input and output data is evaluated, this block can be chosen regardless of the type of interface used.

To enable data transmission using FB_RSTI_EP_5261, a User Defined Type *Ser_Data* must be imported.

Additionally, each array must to be limited to 400 bytes. To use the variables, an instance of this UDT must be built.



The FB_RSTI_EP_5261 block retrieves the process input data (such as InputOfModule, %AI0001) and stores it on the data buffer (RxDataBuffer) of a UDT. The size and the location of the data buffer are user-determined. Additionally, the user can define the number of bytes within a telegram (uiMaxRecvData). The same process applies for data transmission.

FB_RSTi_EP-5261 Input Variables

Variable	*Type*	*Description*
xEnableRecv	BOOL	Enables receive data: 0 = Receive disabled, 1 = Receive enabled
xEnableSend	BOOL	Enables send data: 0 = Send disabled, 1 = Send enabled
xQuit	BOOL	Acknowledgement of errors
xClr_Buf_Rx	BOOL	Flushes receive buffer : 0 to 1 and Quit = 1
xClr_Buf_Tx	BOOL	Flushes the send buffer : 0 to 1 and Quit = 1
xDisableTxHWBuffer	BOOL	Disables the hardware (HW) transmit buffer: 0 = released , 1 = disabled
uiMaxRecvData	UINT	Maximum number of the data byte to be received within one telegram. Can be changed before a new job according to the expected telegram length. ***Note:*** Must be > 0. If not, data will not be received.
uiMaxSendData	UINT	Maximum number of the data byte to be transmitted within one telegram. Can be changed before a new job according to the expected telegram length. ***Note:*** Must be > 0. If not, data will not be transmitted.
InputOfModule	ARRAY [0..7] OF INT	Process input data of the module . Data type is INT.
OutputOfModule	ARRAY [0..7] OF INT	Process input data of the module. Data type is INT.
RxDataBuffer	ARRAY OF BYTE	Address of the buffer that receives data within the PLC. Array of n elements of data type BYTE. Currently fixed to 400 bytes.
TxDataBuffer	ARRAY OF BYTE	Address of the buffer that transmits data within the PLC. Array of n elements of data type BYTE. Currently fixed to 400 bytes.

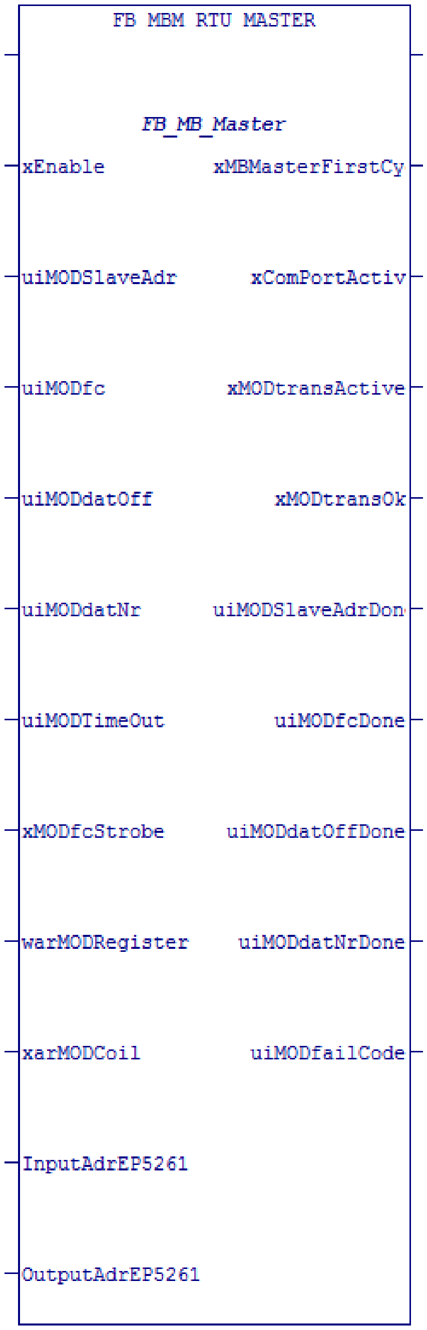
FB_RSTi_EP-5261 Output Variables

Variable	*Type*	*Description*
xBusyRecv	BOOL	Displays an active data reception
xRecvDone	BOOL	Displays finished data reception. Remains TRUE until *xEnableRecv* is TRUE.
uiRecvDataCnt	UINT	Counter for the received data bytes
xBusySend	BOOL	Displays active data transmission
xSendDone	BOOL	Displays finished data transmission. Remains TRUE until *xEnableSend* is TRUE.
uiSendDataCnt	UINT	Counter for transmitted data bytes
xRxHWBufNotEmpty	BOOL	Indicates that the hardware receive buffer is not empty
xErrorFlag	BOOL	Displays a general error. Can be reset by acknowledging the variable *xQuit*.
wRetVal	WORD	Return Value: value is > 8000h to error
Possible Output Variable	***RetVal***	***Description***
Warnings	16#0000	No error
	16#0001	Receive buffer not empty
	16#0002	Handshake (CTS or XOFF) ON
	16#0003	Receive buffer not empty and handshake (CTS or XOFF) ON
Errors	16#8000	Module not ready for communication. Check the address of variable *InputOfMdule*.
	16#8008	Indicates a parameter fault. Check the parameter choice in the PLC configuration.
	16#8010	Indicates a hardware fault. Replace the hardware.
	16#8020	Indicates fault data flow control. Check the parameter in the PLC configuration.
	16#8040	Indicates frame fault. Check the parameter choise in the PLC configuration.
	16#8080	Indicates (receive)buffer overflow of EP-5261 module. Check the communication.
	16#80C0	Indicates (receive)buffer overflow and frame fault of EP-5261 module. Check the communication.
	16#8101	Size of receive buffer is > maximum number of received bytes ==> FB abort.
	16#8201	Size of send buffer is > maximum number of bytes to be sent ==> FB abort.

Modbus RTU Master Function Block (FB_MBM_RTU_Master), Version 1.001

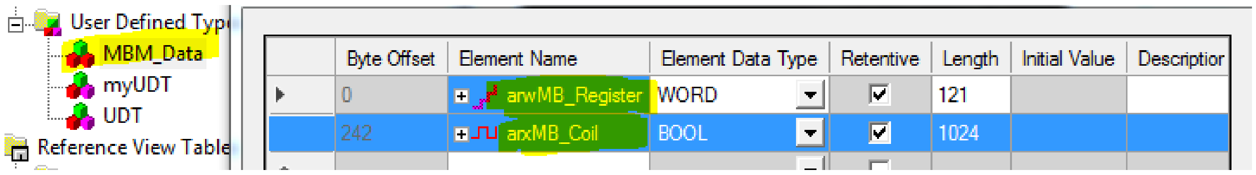
The modbus RTU master function block FB_MBM_RTU_Master is a MODBUS RTU Master block used *in combination with the Serial Communication Interface module RTSi EP-5261 and the* function block *FB_RSTI_EP_5261*. Using this block, a maximum of 120 registers / 1024 coils per job per message can be exchanged.

The function block FB_MBM_RTU_Master Version 1.001 is using/calling the function block FB_RSTI_EP_5261 for handling the serial module EP5261. All needed data and variables will be automatically used internally. Additionally the auxiliary function "ModbusMaster_CoilMax"is also called from the function block which is included in the library.



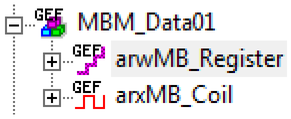
Modbus RTU Master Function Block (FB_MBM_RTU_Master-1.001)

To enable MODBUS data access, a User Defined Types *MBM_Data* must be imported.
An USER DEFINED TYPES data base has to be created for the MODBUS data access or could get from the library.



	Byte Offset	Element Name	Element Data Type	Retentive	Length	Initial Value	Description
▶	0	arwMB_Register	WORD	✓	121		
	242	arxMB_Coil	BOOL	✓	1024		

To use the variables, an instance of this UDT must be built.



Every data transfer on the bus is initiated and controlled by the master. A high level on the input varible *xMODfcStrobe* starts the job, and its function is determined by the function code entered at the input variable *usiMODfc*. Communication with the MODBUS station, for which an address has been parameterized on the input variable *usiMODslaveAdr*, is accepted. Depending on the function code, the MODBUS input parameters are interpreted as listed in the following table.

Function Code	***uiMODdatOff***	***uiMODdatNr***	***warMODRegister***	***xarMODCoil***
FC 1	Coil source offset in slave	Number of requested coils	---	Coil target buffer in master
FC 2	Input source offset in slave	Number of requested inputs	---	Input target buffer in master
FC 3	Holding register source offset in slave	Number of requested registers	Holding register target buffer in master	---
FC 4	Input register source offset in slave	Number of requested registers	Input register target buffer in master	---
FC 5	Coil target offset in slave	---	---	Coil value in index [1]
FC 6	Holding register target offset in slave	---	Register value in index [1]	---
FC 8	---	---	Test value in index [1]	---
FC 15	Coil target offset in slave	Number of coils to be sent	---	Coil source buffer in master
FC 16	Holding register target offset in slave	Number of registers to be sent	Holding register source buffer in master	---

Buffers *warMODRegister* and *xarMODCoil* are always Read or Written by the master from the first index (MODBUS to 1). The MODBUS diagnostic outputs *usi_MODslaveAdrDone*, *usiMODfcDone*, *uiMODdatOffDone*, and *uiMODdatNrDone*, always mirror the communication state during a malfunction, meaning they acknowledge the respective inputs or provide information concerning the content of the slave response. With function codes 5, 6 and 8, *uiMODdatNrDone* describes the written value. The output *xMODtransActive* is TRUE during execution, until the slave response is received. After the job has been completed, the output returns to FALSE and *xMODtransOk* is set to a value of 1 if no telegram fault, protocol fault, data fault, or other communication fault has occurred. During a fault, *xMODtransOk* remains at FALSE and the fault is described with the *usiMODfailCode* exception code. If the slave does not answer within the set *tMODtimeOut* timeout threshold, an error message is generated. The outputs are refreshed each time a new job is started.

Supported Function Codes

Function Code	Description	Output
1	Read multiple coil status	Reading of bit variables (Coils)
2	Read multiple input status	Reading of bit variables (Inputs)
3	Read multiple holding registers	Reading of word variables (Register)
4	Read multiple input registers	Reading of word variables (Inputs)
5	Force single coil	Writing of a bit variable
6	Force single register	Writing of a word variable
8	Loop back diagnostic test (00: return query data)	Connection test
15	Force multiple coils	Writing multiple bit variables (Coils)
16	Force multiple registers	Writing multiple word variables (Register)

FB_MBM_RTU_Master Input Variables

Variable	Type	Description
xEnable	BOOL	Enable COM port for MODBUS communication and reset/initiate the module EP-5261 while toggle the flag.
usiMODSlaveAdr	BYTE	Address of MODBUS Slave. Default = 0.
usiMODfc	BYTE	Function codes 1, 2, 3, 4, 5, 6, 8, 15, 16 are supported, depending of the MODBUS slave
uiMODdatOff	WORD	Offset address (begin) of MODBUS register that will be accessed
uiMODdatNr	WORD	Number of MODBUS register of the access
tMODtimeOut	TIME	Timeout for MODBUS slave answer. Default = 10s, value in second.
xMODfcStrobe	BOOL	Start of a MODBUS request 0 to 1
warMODRegister	Array 0..120 of WORD	MODBUS data: Buffer for receive or send register, e.g. MBM_Data01.arwMB_Register
xarMODCoil	Array 0..512 of BOOL	MODBUS data: Buffer for receive or send BOOL(coil) variables, e.g. MBM_Data01.arxMB_Coil
InputAdrEP5261	Array 0..7 of INT	Hardware address of the inputs of EP5261
OutputAdrEP5261	Array 0..7 of INT	Hardware address of the outputs of EP5261

FB_MBM_RTU_Master Output Variables

Variable	*Type*	*Description*
xMBMasterFirstCycle	BOOL	Flag first cycle: set to a value of 1 after the first cycle
xComPortActiv	BOOL	Status of COMport, MODBUS is activated
xMODtransActive	BOOL	MODBUS transmission is active
xMODtransOk	BOOL	MODBUS request is answered OK
usiMODslaveAdrDone	BYTE	Last addressed MODBUS slave
usiMODfcDone	BYTE	Last Function Code
uiMODdatOffDone	WORD	Last offset addr. of MODBUS register
uiMODdatNrDone	WORD	Last number of MODBUS register
usiMODfailCode	BYTE	MODBUS fail code
xEnableRx	BOOL	Responding Bit of FB_RSTI_EP_5261: Start/Enable Receive
Possible Output Variable	***usiMODfailCode***	***Description***
Error code	0	No error
	1	MODBUS: Illegal Function Code
	2	MODBUS: Error Data address
	3	MODBUS: Error Data value
	4	MODBUS: Error telegram length
	10	Parameter error of the function block
	11	COMport open/initializing failed
	12	MODBUS: CRC Error
	13	MODBUS: Time out
	14	MODBUS: Error slave address
	15	MODBUS: Slave indicates incorect Function Code
	16	MODBUS: Slave indicates incorrect number or address of register

Note: FB_MBM_RTU_Master uses and calls the auxiliary function block *ModbusMaster_CoilMax* (included in the library).

Release History

Serial Communication Module

Catalog Number	*Firmware Version*	*Date*	*Comments*
EP-5261-AB	01.00.12	Oct 2017	Release for firmware enhancements and addressing issue in PLC Stop handling.
EP-5261	01.01	Jun 2015	Initial release

PLC Application Function Blocks

Firmware Version	*Firmware Version*	*Date*	*Comments*
FB_RSTI_EP_5261	01.001	Oct 2017	Release for firmware enhancements and provding simplified Block structure.
FB_MBM_RTU_Master			
FB_RSTI_EP_5261	01.00	Aug 2016	Initial release
FB_MBM_RTU_Master			

Important Product Information for this Release

Updates

N/A

Functional Compatibility

N/A

Problems Resolved by this Release

Subject	**Description**
PLC Stop Handling	Release for firmware enhancements and addressing issue in PLC Stop handling.

New Features and Enhancements

Subject	**Description**
FB_MBM_RTU_Master	Simplified Function Block structure with Ver 01.001. Refer “Modbus RTU Master Function Block (FB_MBM_RTU_Master), Version 1.001” in this document for details.

Known Restrictions and Open Issues

None

Operational Notes

None

Product Documentation

RSTi-EP Slice I/O Module User Manual (GFK-2958)

RSTi-EP Slice I/O Functional Safety Module User Manual (GFK-2956)

RSTi-EP Slice I/O Speciality Modules IPI (GFK-2962)

1-800-433-2682
1-434-978-5100
www.geautomation.com

For public disclosure