

Protect^{IT} Motor and Generator Terminal REM 54_

Modbus Communication Protocol

Technical Description

Industrial^{IT}
enabled™

Industrial IT enabled products from ABB are the building blocks for greater productivity, featuring all the tools necessary for lifecycle product support in consistent electronic form.



Contents

1. About this manual	5
1.1. Copyrights	5
1.2. Trademarks	5
1.3. Document Revisions	5
2. Overview of the protocol	6
2.1. Transmission frame formats	7
2.1.1. ASCII mode	7
2.1.2. RTU mode	9
2.2. Master's queries	10
2.3. Normal responses	11
2.4. Exception responses	12
3. REM 54_ profile of Modbus	13
3.1. Supported application functions	13
3.2. Supported diagnostic subfunctions	13
3.3. Diagnostic counters	15
3.4. Possible exception codes	15
3.5. Event reporting	16
4. Setup of the Modbus interface	17
4.1. Protocol activation	17
4.2. Protocol parameters	18
5. Explanation of the REM 54_ configurations	19
5.1. General guidelines for how the REM 54_ application data is seen on the Modbus protocol	19
5.2. Modbus point lists for Modbus configurations	20
5.2.1. Interpretation of the Modbus point list	21
5.2.1.1. Coils	22
5.2.1.2. Digital inputs	23
5.2.1.3. Input registers	23
5.2.1.4. Holding registers	24
6. Appendix A: Profile checklist	25
7. Appendix B: List of used abbreviations	27

1. About this manual

1.1. Copyrights

The information in this document is subject to change without notice and should not be construed as a commitment by ABB Oy. ABB Oy assumes no responsibility for any errors that may appear in this document.

In no event shall ABB Oy be liable for direct, indirect, special, incidental or consequential damages of any nature or kind arising from the use of this document, nor shall ABB Oy be liable for incidental or consequential damages arising from use of any software or hardware described in this document.

This document and parts thereof must not be reproduced or copied without written permission from ABB Oy, and the contents thereof must not be imparted to a third party nor used for any unauthorized purpose.

The software or hardware described in this document is furnished under a license and may be used, copied, or disclosed only in accordance with the terms of such license.

Copyright © 2004 ABB Oy

All rights reserved.

1.2. Trademarks

All brand or product names are trademarks or registered trademarks of their respective holders.

1.3. Document Revisions

Version B/19.04.2004

- Chapter 3.1 updated
- Picture 3.1-1 updated
- Chapter 4 updated
- Pictures in Chapter 4.2 updated

2. Overview of the protocol

The Modbus protocol was first introduced by Modicon Inc. and is widely accepted as a communication standard for industrial device controllers and PLCs. The protocol determines how each controller connected to the Modbus network will recognize a message addressed to it. It also determines the task to be performed and extracts any data or other information contained in the message. If a reply is required, the controller will construct a reply message and send it using the Modbus protocol.

The connection of the master to the slaves can be established either directly or via modems by using the compatible serial interface. This interface defines the connector pinouts, the cabling, the signal levels, the transmission baud rates, and the parity checking.

The communication technique used in the Modbus protocol is a master-slave technique. This means that only one device can be the master and initiate transactions, while other devices connected to the network are slaves and can accordingly not initiate any transactions.

A message sent by the master to the slave is called a query (see Fig. 2.-1). The master can address a query to an individual slave or to all slaves (i.e. broadcast). If a query is broadcast, the address code 00 will be used. After the slave has received a query, it will attempt to perform the requested task. If a query has been sent to an individual slave, the slave will send a message (called a response) to the master. If it has been broadcast, however, no response will be sent. The response can be either a normal response (in case of performing the requested task) or an exception response (otherwise).

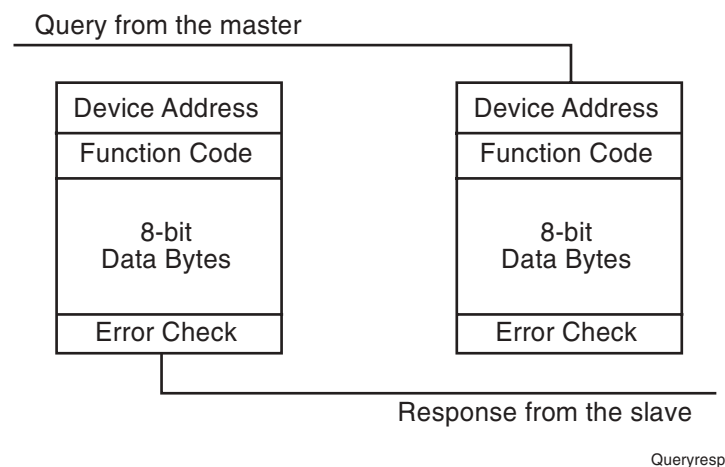


Fig. 2.-1 Query response cycle

The devices in the Modbus protocol are made up of four groups of scan or control points (Digital Inputs, Input Registers, Coils and Holding Registers). Each group consists of either scan or control points, which all have separate 16-bit addresses.

All data addresses in the Modbus protocol are referenced to zero. The first occurrence of a data item is addressed as item number zero: for example the coil known as 'coil 1' is addressed as coil 0000 in the data address field of a Modbus message and 'coil 127' as coil 007E hex (126 in decimal format).

2.1. Transmission frame formats

The format of a query in the Modbus protocol is as follows. The first byte of the frame is the address of the slave to which the query is directed. If the query is broadcast, the address will be 00. The second byte is a function code defining the requested task to be performed. The following bytes are the data to be sent, and the last two bytes form an error-checking field.

The response includes fields containing the id of the slave, confirmation of the performed task in form of a function code, any data to be returned, and an error-checking field. If an error occurs in the receipt of the message, or if the slave is unable to perform the requested task, the slave will construct an error message and send it as its response.

The Modbus protocol has two serial transmission modes: ASCII and RTU. These modes define the bit contents of the message fields transmitted in the network. They also determine how information will be packed into the message fields and decoded. The selected mode and the serial parameters must be the same for all devices in a Modbus network.

2.1.1. ASCII mode

If the Modbus ASCII (American Standard Code for Information Interchange) mode is used, each 8-bit byte in a message will be sent as two ASCII characters forming a hexadecimal number. The allowable characters are the hexadecimals 0-9 and A-F. The advantage of this mode compared to the RTU is that the ASCII mode allows time intervals of up to one second to occur between the characters without causing an error. The format of each byte is presented in Table 2.1.1-1. Fig. 2.1.1.-1 describes the bit sequence of the ASCII mode.

Table 2.1.1-1 ASCII byte format

Coding System	8-bit byte sent as two ASCII characters representing a hexadecimal number
Bits per Byte	1 start bit 7 data bits, the least significant bit sent first 1 bit for even/odd parity; no bit for no parity 1 stop bit if parity is used; 2 bits if no parity
Error Check Field	Longitudinal Redundancy Check (LRC)

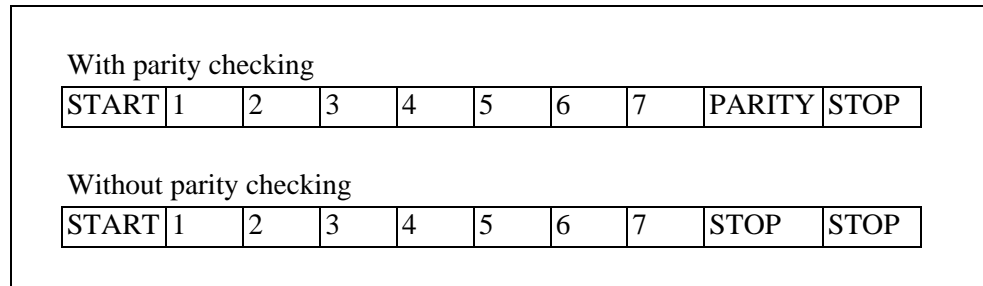


Fig. 2.1.1.-1 Description of the bit sequence for the ASCII mode

The message starts with a colon character (:), followed by ASCII 3A in hexadecimal format, and ends with a "carriage return - line feed" (CRLF) pair, i.e. ASCII 0D and 0A in hexadecimal format. The other fields in the message frame (see Fig. 2.1.1.-2) are identical in both modes of the Modbus protocol, with the exception of the error-checking field. In the ASCII mode the LRC (Longitudinal Redundancy Check) method is applied, whereas the CRC (Cyclical Redundancy Check) method is applied in the RTU mode.

When a device that has been connected to the network detects a colon character, it decodes the following field to find out whether it is the device to which the query is directed. For this reason, each device must continuously monitor the Modbus network.

If a longer interval than one second occurs between the characters, the receiving device assumes that an error has occurred. Consequently, it clears the receive buffer and starts waiting for the colon character. A typical message frame is shown in Fig. 2.1.1.-2.

Start	Address	Function	Data	LRC check	End
1 char	2 chars	2 chars	2*n chars	2 chars	2 chars (CRLF)

Fig. 2.1.1.-2 Description of the standard frame format

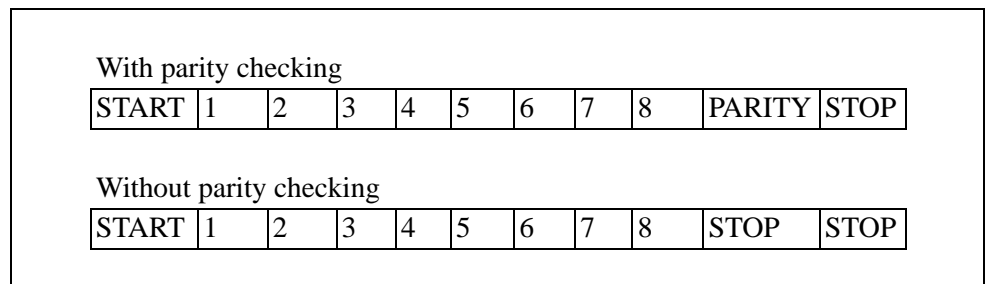
2.1.2.**RTU mode**

In the RTU (Remote Terminal Unit) mode each message byte is sent in binary format. Each byte has one start bit, eight data bits, one even, odd or no parity bit, and one or two stop bits. The number of stop bits depends on whether a parity bit is used. If odd or even parity is used, the byte has one stop bit. If parity is not used, however, there are two stop bits. All together there are eleven bits in one byte. The format of one byte is presented in Table 2.1.2-1.

Table 2.1.2-1 RTU byte format

Coding System	8-bit binary
Bits per Byte	1 start bit 8 data bits, the least significant bit sent first 1 bit for even/odd parity; no bit for no parity 1 stop bit if parity is used; 2 bits if no parity
Error Check Field	Cyclical Redundancy Check (CRC)

The messages are transmitted in the network from left to right, i.e. the Least Significant Bit (LSB) first and the Most Significant Bit (MSB) last. The description of the bit sequence for the RTU mode is presented in Fig. 2.1.2.-1.

*Fig. 2.1.2.-1 Description of the bit sequence for the RTU mode*

The beginning of each frame is marked with a silent interval of at least 3.5 character times. This is implemented as a multiple of character times at the baud rate that is being used. The end of the frame is also marked with a silent interval of at least 3.5 character times.

The entire message frame must be transmitted continuously. If there is a silent interval longer than 1.5 character times between the characters, the next byte is considered as the beginning of a new frame. If a new message begins before the 3.5 character time silent interval, the characters received will be considered as part of the old message frame.

Start	Address	Function	Data	CRC check	End
T1-T2-T3-T4	8 bits	8 bits	N x 8 bits	16 bits	T1-T2-T3-T4

Fig. 2.1.2.-2 Description of the standard frame format

2.2.**Master's queries**

The format of a master's query depends on what function is being used.

The format of a read function query (read coil status, read input status, read input registers, read holding registers) is as follows:

Mode	ASCII	RTU
Start character	':'	3.5 character time of the silent (idle) line
Address	2 characters	1 byte
Function	2 characters	1 byte
Starting data address	4 characters	2 bytes
Quantity of points	4 characters	2 bytes
Error check field	LRC 2 characters	CRC 2 bytes
End characters	CR LF (0x0C 0x0A)	3.5 character time of the silent (idle) line

The format of a force single coil or a preset single register function query is as follows:

Mode	ASCII	RTU
Start character	':'	3.5 character time of the silent (idle) line
Address	2 characters	1 byte
Function	2 characters	1 byte
Data address	4 characters	2 bytes
Data value	4 characters	2 bytes
Error check field	LRC 2 characters	CRC 2 bytes
End characters	CR LF (0x0C 0x0A)	3.5 character time of the silent (idle) line

The format of a force multiple coil or a preset multiple registers function query is as follows:

Mode	ASCII	RTU
Start character	':'	3.5 character time of the silent (idle) line
Address	2 characters	1 byte
Function	2 characters	1 byte
Data address	4 characters	2 bytes
Quantity of points	4 characters	2 bytes
Byte count	2 characters	1 byte
Data values	2*N characters	N bytes
Error check field	LRC 2 characters	CRC 2 bytes
End characters	CR LF (0x0C 0x0A)	3.5 character time of the silent (idle) line

N is equal to byte count.

Technical Description

The format of a read/write multiple registers function query is as follows:

Mode	ASCII	RTU
Start character	':'	3.5 character time of the silent (idle) line
Address	2 characters	1 byte
Function	2 characters	1 byte
Read data address	4 characters	2 bytes
Read quantity of points	4 characters	2 bytes
Write data address	4 characters	2 bytes
Write quantity of points	4 characters	2 bytes
Byte count	2 characters	1 byte
Write data values	2*N characters	N bytes
Error check field	LRC 2 characters	CRC 2 bytes
End characters	CR LF (0x0C 0x0A)	3.5 character time of the silent (idle) line

N is equal to byte count.

The format of a diagnostics function query is as follows:

Mode	ASCII	RTU
Start character	':'	3.5 character time of the silent (idle) line
Address	2 characters	1 byte
Function	2 characters	1 byte
Subfunction	4 characters	2 bytes
Data field	4 characters	2 bytes
Error check field	LRC 2 characters	CRC 2 bytes
End characters	CR LF (0x0C 0x0A)	3.5 character time of the silent (idle) line

2.3.

Normal responses

The format of a normal response to a master's query depends on what function is being used.

The format of a response to a read function query is as follows:

Mode	ASCII	RTU
Start character	':'	3.5 character time of the silent line
Address	2 characters (echo of master's query)	1 byte (echo of master's query)
Function	2 characters (echo of master's query)	1 byte (echo of master's query)
Byte count	2 characters	1 byte
Data values	2*N characters	N bytes
Error check field	LRC 2 characters	CRC 2 bytes
End characters	CR LF (0x0C 0x0A)	3.5 character time of the silent line

N is equal to byte count.

Technical Description

The format of a response to a force single coil or a preset single register function query is an echo of the query itself.

Mode	ASCII	RTU
Start character	':'	3.5 character time of the silent (idle) line
Address	2 characters	1 byte
Function	2 characters	1 byte
Data address	4 characters	2 bytes
Data value	4 characters	2 bytes
Error check field	LRC 2 characters	CRC 2 bytes
End characters	CR LF (0x0C 0x0A)	3.5 character time of the silent (idle) line

The format of a response to a force multiple coils or a preset multiple registers function query is as follows:

Mode	ASCII	RTU
Start character	':'	3.5 character time of the silent (idle) line
Address	2 characters	1 byte
Function	2 characters	1 byte
Data address	4 characters	2 bytes
Quantity of points	4 characters	2 bytes
Error check field	LRC 2 characters	CRC 2 bytes
End characters	CR LF (0x0C 0x0A)	3.5 character time of the silent (idle) line

The format of a response to a diagnostics function query is an echo of the query itself. If the request is directed to a counter, however, the slave returns the counter's value in the data field.

2.4.

Exception responses

The format of an exception response to a master's query is as follows:

Mode	ASCII	RTU
Start character	':'	3.5 character time of the silent line
Address	2 characters (echo of master's query)	1 byte (echo of master's query)
Function	2 characters (echo of master's query with MSB set)	1 byte (echo of master's query with MSB set)
Exception code	2 characters	1 byte
Error check field	LRC 2 characters	CRC 2 bytes
End characters	CR LF (0x0C 0x0A)	3.5 character time of the silent line

An application program in the master is responsible for handling exception responses. Typical processes include successive attempts to send a query, sending diagnostic messages to the slave, and notifying the operators.

Modbus Communication Protocol

Technical Description

3. REM 54_ profile of Modbus

3.1. Supported application functions

Function code (HEX)	Function description
01	Read Coil Status Reads the on/off status of discrete outputs
02	Read Input Status Reads the on/off status of discrete inputs
03	Read Holding Registers Reads contents of output registers
04	Read Input Registers Reads contents of input registers
05	Force Single Coil Sets the status of a discrete output
06	Preset Single Register Sets the value of a holding register
08	Diagnostics Checks the communication system between the master and the slave
0B	Get comm event counters Returns amount of successful read/write operations on data points
0F	Force Multiple Coils Sets the status of multiple discrete outputs
10	Preset Multiple Registers Sets the value of multiple holding registers
17	Read/write holding registers Reads a set of holding registers and writes a set of holding registers in one query

3.2. Supported diagnostic subfunctions

The implementation of the Modbus protocol in the REM 54_ supports the following subfunction codes:

Code	Name	Description
00	Return Query Data	The data in the query data field is to be returned (looped back) in the response. The entire response should be identical to the query.
01	Restart Communication Option	The slave's peripheral port is to be initialized and restarted, and all of its communication event counters are to be cleared. If the port is currently in the Listen Only Mode, no response will be sent. If the port is not currently in the Listen Only Mode, a normal response will be sent. This occurs before the restart is executed.
02	Return Diagnostic Register	The contents of the slave's diagnostic register is returned in the response.
04	Force Listen Only Mode	Forces the addressed slave to enter the Listen Only Mode for Modbus communications.
10	Clear Counters and Diagnostic Register	Clears all counters and the diagnostic register.

Technical Description

Code	Name	Description
11	Return Bus Message Count	The response data field returns the quantity of messages that the slave has detected in the communications system since its last restart, clear counters operation, or power-up.
12	Return Bus Communication Error Count	The response data field returns the quantity of CRC errors encountered by the slave since its last restart, clear counters operation, or power-up.
13	Return Bus Exception Error Count	The response data field returns the quantity of Modbus exception responses returned by the slave since its last restart, clear counters operation, or power-up.
14	Return Slave Message Count	The response data field returns the quantity of messages addressed to the slave, or broadcast that the slave has processed since its last restart, clear counters operation, or power-up.
15	Return Slave No Response Count	The response data field returns the quantity of messages addressed to the slave for which it sent no response (neither a normal response nor an exception response) since its last restart, clear counters operation, or power-up.
16	Return Slave NACK Response Count	The response data field returns the quantity of messages addressed to the slave for which it send a NACK response
17	Return Slave Busy Response Count	The response data field returns the quantity of messages addressed to the slave for which it send a Slave Busy response
18	Return Bus Character Overrun Count	The response data field returns the quantity of messages addressed to the slave that it could not handle due to a character overrun condition since its last restart, clear counters operation, or power-up

Note: Sending other subfunction codes than those listed above will cause an "illegal data value" exception response.

3.3. Diagnostic counters

Name	Meaning
Bus Message Count	The number of messages that the REM 54_ has detected in the communications system since its last restart, clear counters operation, or power-up.
Bus Communication Error Count	The number of CRC or LRC errors encountered by the REM 54_ since its last restart, clear counters operation, or power-up.
Bus Exception Error Count	The number of Modbus exception responses sent by the REM 54_ since its last restart, clear counters operation, or power-up.
Slave Message Count	The number of messages addressed to the REM 54_ or broadcast that the REM 54_ has processed since its last restart, clear counters operation, or power-up.
Slave No Response Count	The number of messages addressed to the REM 54_ for which it sent no response (neither a normal response nor an exception response) since its last restart, clear counters operation, or power-up.
Slave NACK Response Count	The number of messages addressed to the REM 54_ for which it returned a NACK response
Slave Busy Response Count	The number of messages addressed to the REM 54_ for which it returned a Slave Busy response
Bus Character Overrun Count	The number of messages addressed to the REM 54_ that it could not handle due to a character overrun condition since its last restart, clear counters operation, or power-up

3.4. Possible exception codes

The following exception codes may be generated by the REM 54_ Modbus Interface.

Code	Name	Meaning
01	ILLEGAL FUNCTION	The function code received in the query is not an allowable action for the REM 54_.
02	ILLEGAL DATA ADDRESS	The data address or number of items received in the query is not allowable or correct for the REM 54_. The REM 54_ will send this exception response if an attempt to read or write part of a multiple register database object is detected. Possible objects are time, strings and counters.
03	ILLEGAL DATA VALUE	A value contained in the query data field is out of range. The contents of the register or the status of the coil has not changed.
04	SLAVE DEVICE FAILURE	An unrecoverable error occurred while the slave was attempting to perform the requested action.
05	ACKNOWLEDGE	The slave has accepted the request and is processing it, but a long duration of time will be required to do so. This response is returned to prevent a timeout error from occurring in the master. The master can next issue a Poll Program Complete message to determine if processing is completed.
06	SLAVE DEVICE BUSY	The slave is engaged in processing a long-duration program command. The master should retransmit the message later when the slave is free.

07	NEGATIVE ACKNOWLEDGE	The slave cannot perform the program function received in the query. This code is returned for an unsuccessful programming request using function code 13 or 14 decimal. The master should request diagnostic or error information from the slave.
08	MEMORY PARITY ERROR	The slave attempted to read extended memory, but detected a parity error in the memory. The master can retry the request, but service may be required on the slave device.

Note: If an "Illegal data value" exception response is generated when attempting to preset multiple registers, the contents of the register to which an illegal value has been imposed and of the following registers will not be changed. Registers that have already been preset will not be restored.

3.5. Event reporting

Event reporting function based on proprietary data format and procedures as defined in REM 54_ is not supported in REM 54_. Instead, momentary change detect points associated with digital input points can be used.

An example of momentary change detect is illustrated in the figure below. A host device monitors a physical input bit 1. The figure illustrates the physical signal transitions of input 1. At each rising edge/falling edge transition, the status of the Modbus digital input 1xxxx addresses are listed. The dotted line arrows indicate the poll received by the REM 54_ device and the state of both the status bit and the momentary indication bit. Note that the even bit (momentary change detect) resets itself to a zero state after the poll.

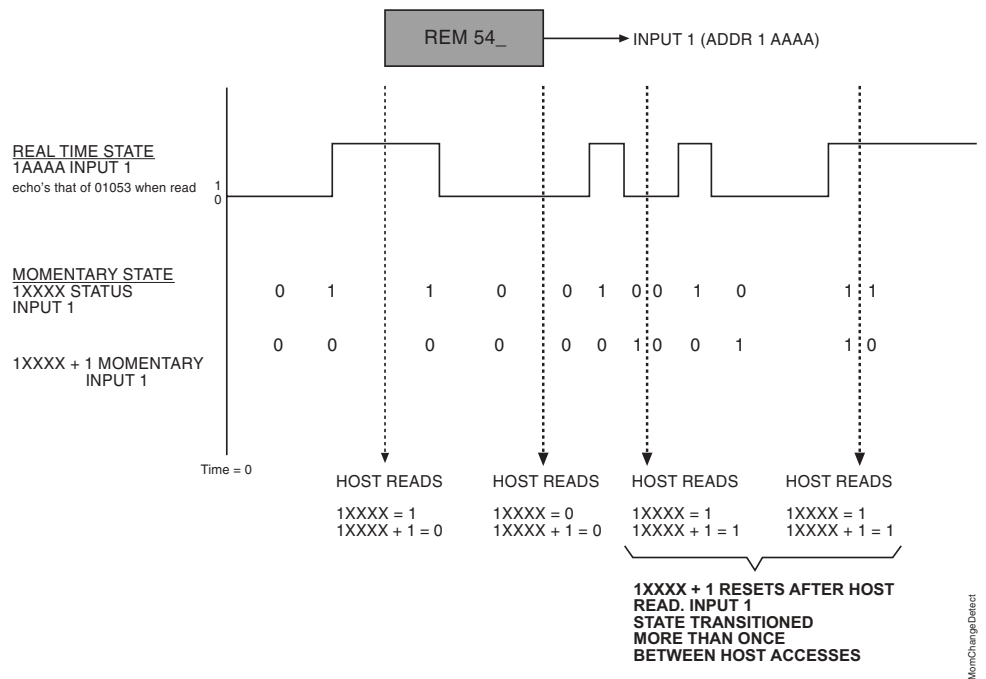


Fig. 3.5.-1 Momentary change detect example

4. Setup of the Modbus interface

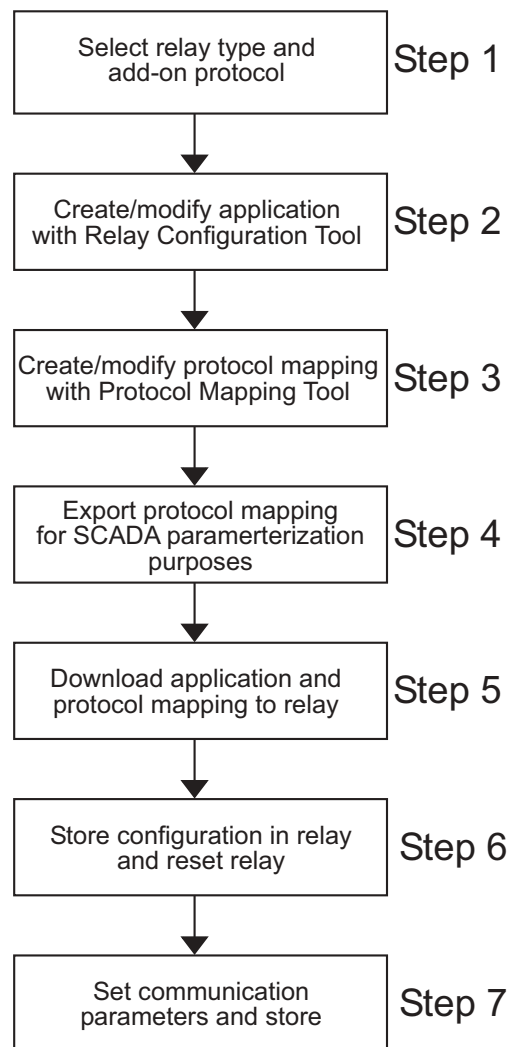
4.1. Protocol activation

This section describes the communication parameters required to configure the REM 54_ to communicate using the Modbus protocol.

The Modbus protocol can be used only when the protocol is properly configured. The protocol must be selected in CAP 505 as an Add-on protocol. For additional information, refer to the CAP 505 Operator's Manual.

When the protocol is selected and the relay configuration is created (refer to: Relay Configuration Tool in CAP 505), the protocol mapping must be created or modified using the Protocol Mapping Tool (PMT). The protocol parameters (as described in chapter 4.2.) are available only after the protocol is first selected and then activated.

After the protocol mapping is downloaded and stored in the relay, a reset of the relay activates the protocol.



Modbus_interfacefg_a

Fig. 4.1.-1 Interface configuration

Technical Description

If the application is changed, start over from step 2.

If you wish to keep the existing protocol mapping, you should select a new name for the protocol mapping or skip step 3.

If you create a new protocol mapping with the wizard, the protocol mapping addresses will be changed.

Application downloading overwrites existing add-on protocol parameters and protocol mapping.

4.2.

Protocol parameters

The protocol and link parameters of the Modbus interface can be programmed by means of the local HMI via \Communication\Modbus. Following parameters are available:

Parameter	Comment	Default value
Unit address	Modbus unit address 1...247	1
Baud Rate	Baud rates: 300/1200/ 2400/4800/9600 bps	9600
Modbus Mode	Link mode: 0 = ASCII 1 = RTU	1
Password	For future use	
No of data bits	7 - ASCII Mode 8 - RTU Mode	8
No of stop bits	1...2	1
Parity	0 - No parity 1 - Odd parity 2 - Even parity	2
Next character time-out	Maximum allowed time gap between received characters of the same frame (in milliseconds) 0 - Not in Use	0
End of frame timeout	Minimum idle time following the frame transmission to the REM 54_ (in milliseconds)	10
CRC Order	Order of CRC bytes in the RTU link mode 0 - Low High 1 - High Low	0



When a new configuration is downloaded to the relay (see “Protocol activation” on page 17), the protocol parameters will be set to their default values. The default values of the parameters are listed in the table above.

5. Explanation of the REM 54_ configurations

5.1. General guidelines for how the REM 54_ application data is seen on the Modbus protocol

The figure and the table in this section describe shortly how the process data in the REM 54_ device is seen on the Modbus protocol.

The mapping of the function block data into Modbus process data is done by the Modbus Protocol Mapping. Protocol mapping can be referred to as Protocol Object Dictionary or POD in REC 523 and REX 521 product documentation.

In the application example below, all the possible process data is present. The grey boxes show to which Modbus data category the signals belong.

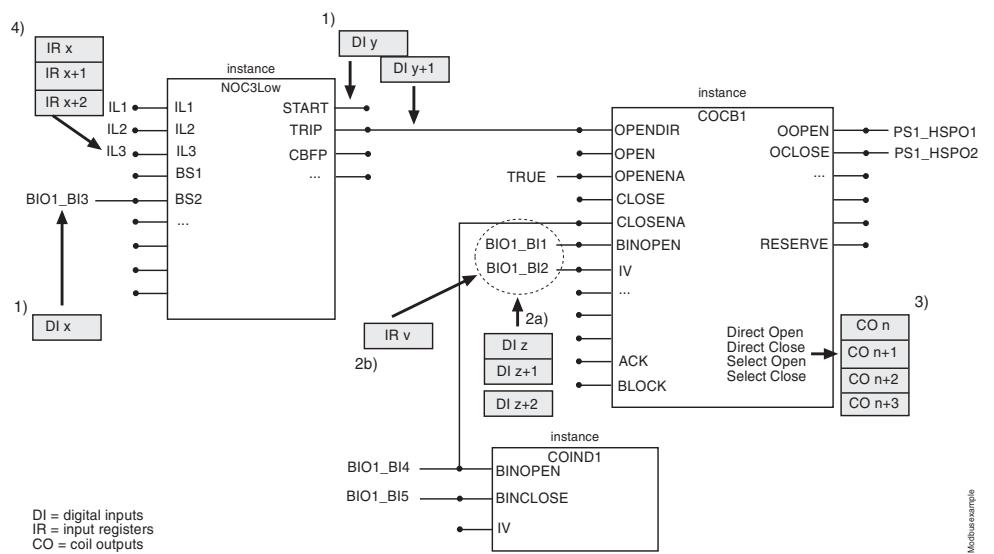


Fig. 5.1.-1 Application example

The figure is explained in the table on the next page.

Explanations to Fig. 5.1.-1			
No	Application data type	Explanation	Modbus data type
1)	One Bit Input	Binary input to function block, e.g. blocking input.	Digital input (1x references)
1)	One Bit Output	Binary output from function block, e.g. START or TRIP signals.	Digital input (1x references)
2a)	Two Bit Input	Binary position data coded in two bits (OPEN, CLOSE)	Digital inputs (1x references) coded in three bits: Bit 1: OPEN Bit 2: CLOSE Bit 3: Validity (1 if OPEN value = CLOSE value)
2b)	Two Bit Input	In addition to 2a) the OPEN and the CLOSE bit values are also coded as least significant bits in an input register. (One register per object)	Input Registers (3x references) Values: 1 = CLOSE 2 = OPEN 3 = Undefined 0 = Undefined
3)	Control output points	Outputs controlled from the Modbus host.	Coils (0x references)
4)	Measurement inputs	Measurement inputs to the function blocks	Input Registers (3x references)
Not visible in the figure	Parameters, settings etc.	Some parameters of the device and function blocks may be adjustable (look in the Modbus point list of the Modbus configuration)	Holding Registers (4x references) • If the data can be both read and written Input Registers (3x references) • If data is read only type

5.2.

Modbus point lists for Modbus configurations

The point list is a cross-reference between the function block's signals and the Modbus protocol. The list tells on which addresses the process data is located when the REM 543 is viewed from the protocol's side.

Along with the Modbus configurations on the "REM 543 Modbus Configurations" CD-ROM (1MRS151023) are also the Modbus point lists. Each Modbus configuration has its own list. The lists are saved in HTML format (See Fig. 5.2.-1.).

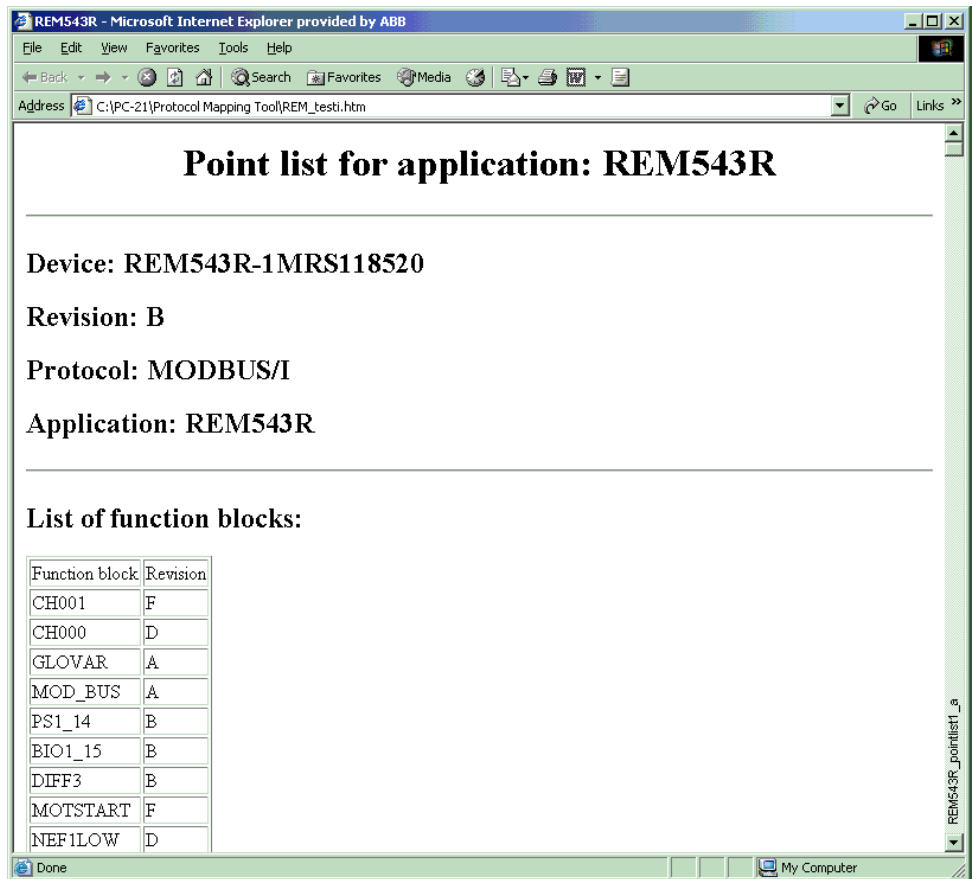


Fig. 5.2.-1 Example of a Modbus point list

The lists start with a header identifying the Modbus configuration and the revision letter. Hereafter follows a list of function blocks in the application. Also the revision letters of the function blocks are shown.

After this follows the description of the process data: Coils, Digital inputs, Input registers and Holding registers. Process data will be described further in the following section.

5.2.1.

Interpretation of the Modbus point list

The point list shows how the application data of a certain Modbus configuration is organized on the Modbus protocol. Modbus basic data areas are:

- Coils (0x references),
- Digital inputs (1x references),
- Input registers (3x references) and
- Holding registers (4x references).

The basic data areas have further been divided into subtypes called:

- Basic range,
- Extended range,
- Slowly changing,

- Control (possible in Coils and Holding registers area only) and
- Diagnostics.

Subtypes are not necessarily used in all the basic data areas which depends on the Modbus configuration.

In the REM 54_ Modbus configurations the Modbus basic data areas are used as follows:

	Coils	Digital inputs	Input registers	Holding registers
Basic range		Start at 0001H	Start at 0001H	
Extended range		Start at 0800H		
Slowly changing		Start at 1000H		
Control	Start at 2000H			
Diagnostics			Start at 3000H	Start at 3000H

5.2.1.1.

Coils

Coils are outputs used for control or acknowledgement purposes. Coil outputs start from coil address 2000H. The coils in the REM 54_ can be written but not read. Reading will cause an exception response from the REM 54_.

If more than one coil is written at a time (write multiple coils) it will result in exception response 05. This response indicates that the request has been accepted by the slave and processing has started. Still, a long duration of time is required. The master will hereafter issue a Poll Program Complete message to determine if processing is completed. See the Modicon technical publications for more information. Do not use multiple coil writing unless your host can interpret the 05 exception response correctly.

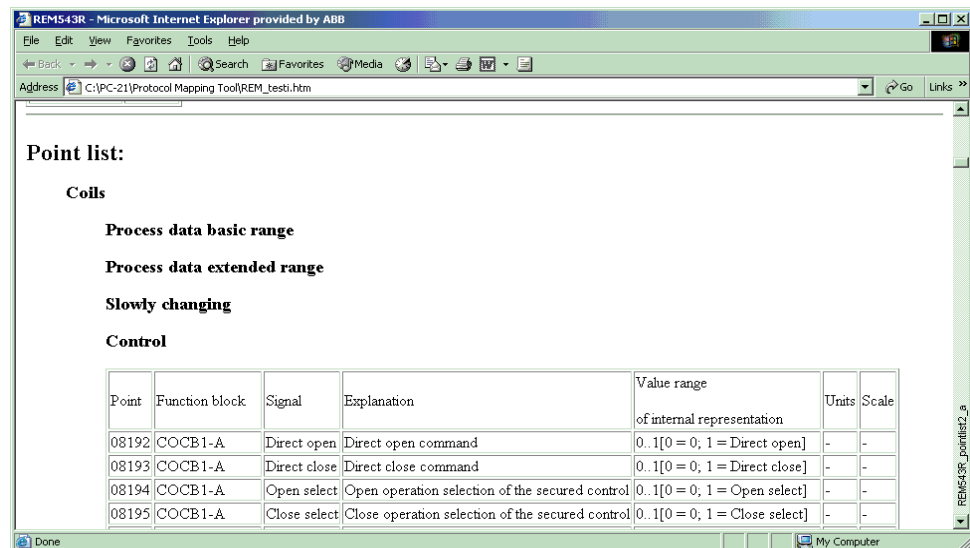


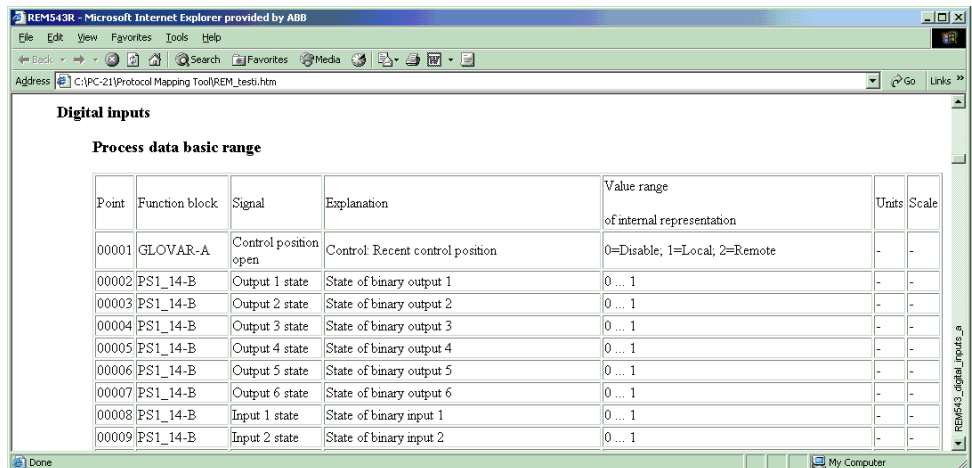
Fig. 5.2.1.1.-1 Example of a Coils section in a point list

5.2.1.2.

Digital inputs

Digital inputs are used for monitoring discrete process inputs (e.g. breaker position and locking inputs). Process outputs (like START or TRIP signals) are also available in this area. The difference between basic range data and extended range data is that the latter contains both momentary positions and change detect bits, while the former contains only momentary positions of the digital inputs. Basic range data starts from point address 0001H and extended range data from point address 0800H.

Slowly changing data means such data that can be scanned by the master with a "slower cycle" than the basic and extended range data. Slowly changing data starts from point address 1000H.



Point	Function block	Signal	Explanation	Value range of internal representation	Units	Scale
00001	GLOVAR-A	Control position open	Control Recent control position	0=Disable; 1=Local; 2=Remote	-	-
00002	PS1_14-B	Output 1 state	State of binary output 1	0 ... 1	-	-
00003	PS1_14-B	Output 2 state	State of binary output 2	0 ... 1	-	-
00004	PS1_14-B	Output 3 state	State of binary output 3	0 ... 1	-	-
00005	PS1_14-B	Output 4 state	State of binary output 4	0 ... 1	-	-
00006	PS1_14-B	Output 5 state	State of binary output 5	0 ... 1	-	-
00007	PS1_14-B	Output 6 state	State of binary output 6	0 ... 1	-	-
00008	PS1_14-B	Input 1 state	State of binary input 1	0 ... 1	-	-
00009	PS1_14-B	Input 2 state	State of binary input 2	0 ... 1	-	-

Fig. 5.2.1.2.-1 Example of a Digital inputs section in a point list

5.2.1.3.

Input registers

Measurement data is located in the Input register basic data area, starting from point address 0001H. In the Modbus point list the value type (signed or unsigned) and the predefined scaling are shown for each process value.

All the two-bit position data (breakers, disconnectors) found in the digital input area are also located in the Input register area. This data is always mapped so that one single input register contains the OPEN/CLOSE bit positions of one object in the two least significant bit locations.

The Diagnostics area includes values which need not to be continuously scanned by the master but rather read on demand. This data starts from point address 3000H.

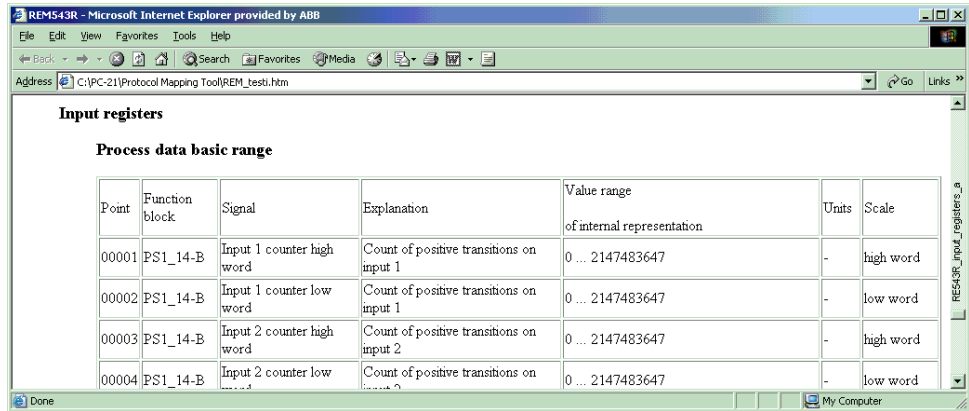


Fig. 5.2.1.3.-1 Example of a Input registers section in a point list

5.2.1.4.

Holding registers

In this area resides data which may be read or written on demand. This data starts from point address 3000H.

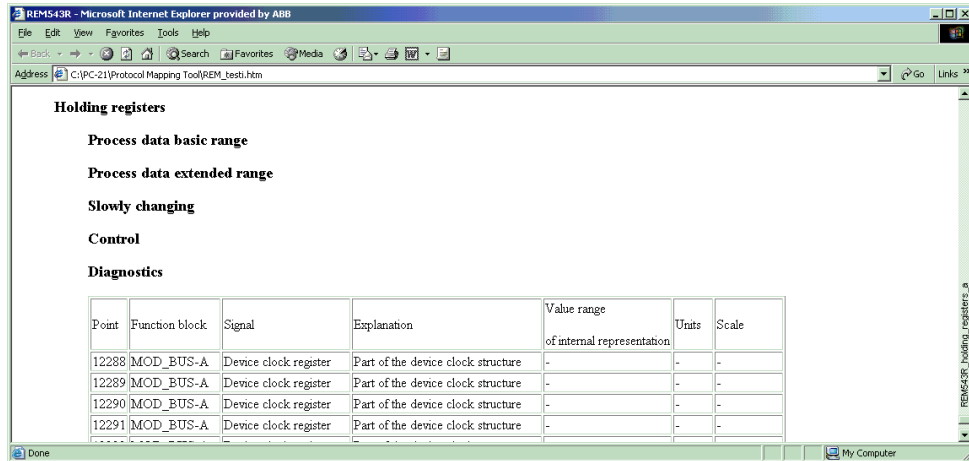


Fig. 5.2.1.4.-1 Example of a Holding registers section in a point list

Modbus Communication Protocol

Technical Description

6.

Appendix A: Profile checklist

MODBUS
DEVICE PROFILE DOCUMENT
Vendor Name: ABB Oy Distribution Automation
Device Name: REM 54_
Device Function: Slave
Modes: <input checked="" type="checkbox"/> RTU <input checked="" type="checkbox"/> ASCII

Supported function codes

Code (HEX)	Function	Supported
01	Read coil Status	Yes
02	Read Input Status	Yes
03	Read Holding Register	Yes
04	Read Input Registers	Yes
05	Force Single Coil	Yes
06	Preset Single Register	Yes
07	Read Exception Status	No
08	Diagnostics	Yes
0B	Fetch Comm Event Counter	Yes
0C	Fetch Comm Event Log	No
0F	Force Multiple Coils	Yes
10	Preset Multiple Registers	Yes
11	Report Slave ID	No
14	Read General Reference	No
15	Write General Reference	No
16	Mask Write 4x Register	No
17	Read/Write 4x Registers	No

Supported diagnostic subfunction codes

Code (HEX)	Name	Supported
00	Return Query Data	Yes
01	Restart Communication Option	Yes
02	Return Diagnostic Register	Yes
03	Change ASCII Delimiter	No
04	Force Listen Only Mode	Yes
0A	Clear Counters and Diagnostics Register	Yes
0B	Return Bus Message Count	Yes
0C	Return Bus Communication Error Count	Yes
0D	Return Bus Exception Error Count	Yes
0E	Return Slave Message Count	Yes
0F	Return Slave No Response Count	Yes
10	Return Slave NAK Count	No
11	Return Slave Busy Count	No
12	Return Bus Character Overrun Count	No
13	Return IOP Overrun Count	No
14	Clear Overrun Counter Counter and Flag	No
15	Get / Clear Modbus Plus Statistics	No

Supported exception responses

Code	Name	Supported
01	ILLEGAL FUNCTION	Yes
02	ILLEGAL DATA ADDRESS	Yes
03	ILLEGAL DATA VALUE	Yes
04	SLAVE DEVICE FAILURE	Yes
05	ACKNOWLEDGE	Yes
06	SLAVE DEVICE BUSY	Yes
07	NEGATIVE ACKNOWLEDGE	No
08	MEMORY PARITY ERROR	No

Supported data types

Name	Supported
Digital input	Yes
Coil	Yes
Input register	Yes
Holding register	Yes
General reference	No

Supported event reporting methods

Name	Supported
Momentary change detect on digital input	Yes

7. Appendix B: List of used abbreviations

ASCII	American Standard Code for Information Interchange
CRC	Cyclic Redundancy Check
CTS	Clear To Send
DCD	Data Carrier Detect
LRC	Longitudinal Redundancy Check
PLC	Programmable Logic Controller
POD	Protocol Object Dictionary
RTS	Request To Send
RTU	Remote Terminal Unit



ABB Oy

Distribution Automation

P.O. Box 699

FI-65101 Vaasa

FINLAND

Tel. +358 10 22 11

Fax. +358 10 224 1094

www.abb.com/substationautomation