RELION® PROTECTION AND CONTROL

REX615

Modbus Communication Protocol Manual

Document ID: 2NGA001860

Issued: 2024-04-23

Revision: A

Product version: 1

Open Source Software

This product contains open source software. For license information refer to product documentation at *www.abb.com*.

Warranty

Please inquire about the terms of warranty from your nearest ABB representative.

www.abb.com/mediumvoltage

Disclaimer

The data, examples and diagrams in this manual are included solely for the concept or product description and are not to be deemed as a statement of guaranteed properties. All persons responsible for applying the equipment addressed in this manual must satisfy themselves that each intended application is suitable and acceptable, including that any applicable safety or other operational requirements are complied with. In particular, any risks in applications where a system failure and/or product failure would create a risk for harm to property or persons (including but not limited to personal injuries or death) shall be the sole responsibility of the person or entity applying the equipment, and those so responsible are hereby requested to ensure that all measures are taken to exclude or mitigate such risks.

This product has been designed to be connected and communicate data and information via a network interface which should be connected to a secure network. It is the sole responsibility of the person or entity responsible for network administration to ensure a secure connection to the network and to take the necessary measures (such as, but not limited to, installation of firewalls, application of authentication measures, encryption of data, installation of antivirus programs, etc.) to protect the product and the network, its system and interface included, against any kind of security breaches, unauthorized access, interference, intrusion, leakage and/or theft of data or information. ABB is not liable for any such damages and/or losses.

This document has been carefully checked by ABB but deviations cannot be completely ruled out. In case any errors are detected, the reader is kindly requested to notify the manufacturer. Other than under explicit contractual commitments, in no event shall ABB be responsible or liable for any loss or damage resulting from the use of this manual or the application of the equipment. In case of discrepancies between the English and any other language version, the wording of the English version shall prevail.

Conformity

This product complies with following directive and regulations.

Directives of the European parliament and of the council:

- Electromagnetic compatibility (EMC) Directive 2014/30/EU
- Low-voltage Directive 2014/35/EU
- RoHS Directive 2011/65/EU
- RoHS Directive (EU) 2015/863 amending Annex II

UK legislations:

- Electromagnetic Compatibility Regulations 2016
- Electrical Equipment (Safety) Regulations 2016
- The Restriction of the Use of Certain Hazardous Substances in Electrical and Electronic Equipment Regulations 2012

These conformities are the result of tests conducted by the third-party testing in accordance with the product standard EN / BS EN 60255-26 for the EMC directive / regulation, and with the product standards EN / BS EN 60255-1 and EN / BS EN 60255-27 for the low voltage directive / safety regulation.

The product is designed in accordance with the international standards of the IEC 60255 series.

Contents

1 Introduction

1.1 This manual

The communication protocol manual describes a communication protocol supported by the protection relay. The manual concentrates on vendor-specific implementations.

1.2 Intended audience

This manual addresses the communication system engineer or system integrator responsible for pre-engineering and engineering the communication setup in a substation from a protection relay's perspective.

The system engineer or system integrator must have a basic knowledge of communication in protection and control systems and thorough knowledge of the specific communication protocol.

1.3 Product documentation

1.3.1 Product documentation set

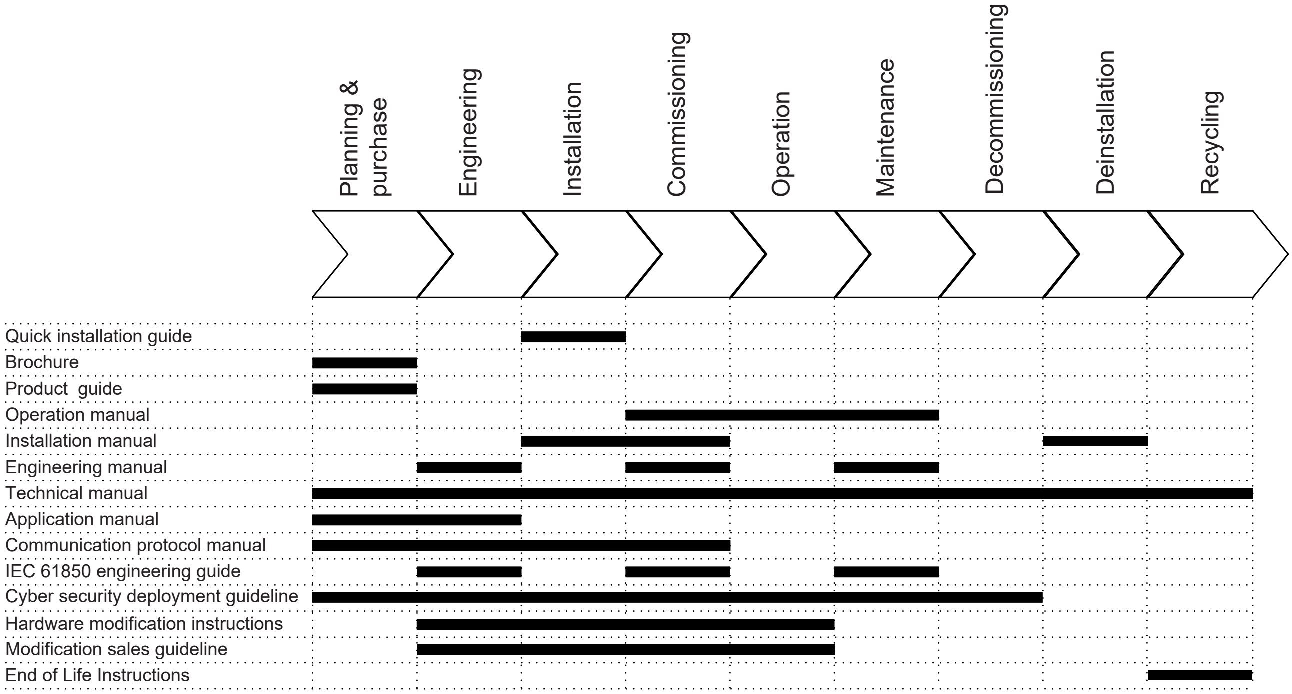

Figure 1: The intended use of documents during the product life cycle

1.3.2 Document revision history

Document revision/date	Product release	History
A/2024-04-23	PCL1	First release

1.3.3 Related documentation

Download the latest documents from the ABB Web site *www.abb.com/mediumvoltage*.

1.4 Symbols and conventions

1.4.1 Symbols

The caution icon indicates important information or warning related to the concept discussed in the text. It might indicate the presence of a hazard which could result in corruption of software or damage to equipment or property.

The information icon alerts the reader of important facts and conditions.

The tip icon indicates advice on, for example, how to design your project or how to use a certain function.

Although warning hazards are related to personal injury, it is necessary to understand that under certain operational conditions, operation of damaged equipment may result in degraded process performance leading to personal injury or death. Therefore, comply fully with all warning and caution notices.

1.4.2 Document conventions

A particular convention may not be used in this manual.

- Abbreviations and acronyms are spelled out in the glossary. The glossary also contains definitions of important terms.
- Menu paths are presented in bold.

  Select **Main menu** > **Settings**.
- Parameter names are shown in italics.

  The function can be enabled and disabled with the *Operation* setting
- Parameter values are indicated with quotation marks.

  The corresponding parameter values are "On" and "Off".
- Input/output messages and monitored data names are shown in Courier font.

  When the function starts, the `START` output is set to TRUE.
- Values of quantities are expressed with a number and an SI unit. The corresponding imperial units may be given in parentheses.
- This document assumes that the parameter setting visibility is "Advanced".

2 Modbus overview

2.1 Modbus standard

Modbus is a communication protocol developed by the Modicon company in the 1970's. Originally it was used for communication in PLCs and RTU devices. Later on the Modbus protocol was used in a variety of different device applications. Today the Modbus protocol is mainly used over serial communication networks and Ethernet.

The Modbus serial communication and the Ethernet-based Modbus TCP/IP communication in this protection relay follow the specifications maintained by Modbus Organization.

Modbus communication reference guides are downloadable from Technical Resources at *www.modbus.org*.

2.1.1 Serial communication

Modbus is a client-server protocol when it is used over serial communication networks. This protection relay implements the server side of the protocol. Depending on the chosen physical serial interface it is possible to build multidrop networks or point-to-point communication connections.

There can only be one Modbus client unit on a Modbus serial network. The Modbus client unit communicates with one Modbus server unit at a time. Usually the client reads, or scans, data from the servers cyclically. The client can also write data or give commands to the server units. Each server unit has a unique unit address. Thus, the client can identify the server with which it communicates. The Modbus standard also supports client broadcast transmissions.

Modbus serial protocol uses two link modes: Modbus RTU and Modbus ASCII. Both modes are supported by this protection relay.

2.1.2 Ethernet communication

Modbus communication over Ethernet TCP/IP is of client-server type. This protection relay operates as a Modbus server.

The Modbus TCP/IP connection is established when the Modbus client opens a TCP socket connection to the Modbus server. The socket port 502 on the TCP/IP stack is reserved for Modbus. If the connection request is accepted by the server, the client can start communicating with the server unit.

Protection relays can usually accept several simultaneous Modbus TCP/IP client connections even though the number of connections is limited. The protection relay can be configured to only accept socket connection requests from known client IP addresses.

2.1.3 Application data implementation

This protection relay is designed to operate with a wide range of different Modbus clients and servers. The Modbus memory map shows the protection relay's internal process data in a simple I/O map which is mainly aimed at PLC clients and other process automation devices. Time-tagged, chronological event lists and fault records can be read over the Modbus interface. This data is more suitable for SCADA Modbus clients.

The Modbus standard defines four main memory areas for mapping protection relay's process data. Due to its open nature, the Modbus standard does not define exactly what type of data should be mapped to each memory area. The Modbus mapping approach of the protection relay ensures that the same process data are readable from as many Modbus memory areas as possible. The users may then choose the memory areas that are most suitable for their Modbus client systems.

2.1.4 Terms and definitions

Modbus data appear in different memory areas in the Modbus device. The four most common areas are coils, digital inputs, input registers and holding registers. These are also referred to as 0X, 1X, 3X and 4X areas respectively.

Modbus defines addressing in two ways: PLC addressing starts from address 1 and regular Modbus data addressing starts from 0. For example, a holding register at PLC address 234 can be referred to either as 4X register 234 or as 40234. The regular Modbus address, that is the PLC address decremented by one, is shown when analyzing the Modbus traffic on the physical network.

Listings and references to the Modbus data in this documentation follow the PLC addressing scheme. Addresses start from 1.

See also the Modbus protocol standard documentation that can be found for free at *www.modbus.org*.

3 Vendor-specific implementation

3.1 Protocol server instances

In this manual the counterparts of the communication exchange are referred to as "client-server".

The protection relay can communicate with several protocol clients simultaneously. Furthermore, it is possible to configure the protection relay to provide different protocol data and data outlook for different clients. A protocol server communication entity which is configured to operate with a specific client is called an instance.

There are three server instance scenarios.

1. One client - One protocol instance - One protocol mapping. The protection relay is intended to operate toward one protocol client. The default protocol data mapping or data outlook can be modified freely.
2. Several clients - Several protocol instances - One protocol mapping. The protection relay is intended to operate toward several protocol clients. All the clients are able to access the same data or similar data outlook. The default protocol mapping or data outlook can be modified freely.
3. Several clients - Several protocol instances - Several protocol mappings. The protection relay is intended to operate toward several protocol clients. Some or all the clients may access protocol data in a different manner, so several protocol mappings derived from the default protocol mapping need to be prepared.

3.1.1 Connection to clients

In the relay it is possible to activate up to five protocol server instances, each represented by a separate function block in the relay configuration. The five function blocks are named MBSLPRT1...5. For each connected client, a protocol instance has to be activated by dragging the function block into the relay configuration. When a function block is present in the configuration, its setting and monitoring parameters are visible in the HMI. An exception is protocol instance 1, which is always visible in the HMI, and which can be activated without dragging its function block instance into the relay configuration.

Figure 2: Function block

The protection relay restricts communication clients to five, regardless of the protocols that the clients use. This includes the MMS clients and other communication protocol clients.

The available five Modbus instances may be freely activated. However, it is recommended to activate the instances in chronological order. For example, instance 1 is to be used if there is only one client connection and instances 1 and 2 when there are two clients.

First setup and configuration upload

Dragging a protocol instance function block into the relay configuration is only the first step of the protocol activation. When a protocol instance is added for the first time, it is inactive and unconfigured by default, meaning that it has not been assigned to a physical link port. Neither has any Modbus data point configuration been loaded for the instance. Next step is to do these setups.

Each time the protocol instance setting parameter *Operation* is changed between "Off" and "On", a relay restart is needed. If *Operation* is set to "On" without restarting the relay, the Modbus instance responds to all Modbus requests with exception value 4.

When a configuration containing Modbus instances is uploaded to the relay, the Operation setting is "Off". After the upload is done, the Modbus instances restart, still with Operation being in "Off" state. Thereafter, Parameter Setting in PCM600 uploads the new settings to all applications, including the Operation "On" values for the protocol instances. In this case the relay must be restarted once more in order for the Modbus protocol to start operating properly.

3.1.2 Protocol server attachment to a client

After its activation, an instance should be attached to the intended client.

If the client is in a serial connection, the instance must be attached to the intended serial port.

In case of a TCP client, the instance must be first attached to the physical Ethernet port. If there are several TCP client connections, the protection relay must be able to distinguish between the clients. There are two setting parameters in an instance.

- *Client IP*: When the client makes the TCP connection, its IP address is checked. This instance is given to the client with this IP address. It is also possible to use the address “0.0.0.0” if no client IP address is to be defined. In this case, the client's IP address is ignored. Note that in case the client resides in another network segment, the relay will not be able to detect the client's IP-address.
- *TCP port*: This parameter defines the TCP socket port used by this particular instance. It can be used in conjunction with the *Client IP* setting, thus allowing only a certain Client IP address to establish the connection. Or the Client IP can alternatively be ignored with the setting "0.0.0.0".

3.1.3 Several identical client connections

If several clients access the same protocol data, the client connections must still be kept apart. Also the number of each instance used for each client must be noted so that if there are problems with the communication, the line diagnostic data for instances follows the same instance number rule.

In case of sequential event data transaction and a TCP client connection, it is essential that a reconnecting client is given back the same instance to which it was attached before disconnecting. This way, the event reading resumes from the point where the client left off, provided that no event overflow has occurred while the client was absent. If multiple client connections are used, they must be distinguished by using the *Client IP* and *TCP port* parameters.

3.1.4 Protocol data mapping to server instances

There can be N number of different data mappings for a protocol. The mappings are identified and numbered, starting from one. This number is not related to the protocol instance number.

In PCM600, it is necessary to always define the mappings to be edited or viewed.

Each protocol instance has the setting parameter *Mapping selection*, which defines the protocol mappings to be used by this instance. Several protocol instances can use the same mapping. By default, the *Mapping selection* parameter for all the instances is set to use the mapping number one.

3.2 Modbus link alternatives

Modbus communication is possible over the serial communication interface, over the Ethernet interface, or over both interfaces simultaneously.

Depending on the protection relay type, either only serial communication or only Ethernet communication may be supported.

3.2.1 Serial link

Modbus serial communication requires that the protection relay variant is equipped with a serial interface in the COM card. The serial interface card can contain one or two serial interfaces.

The Modbus link mode can be either Modbus RTU or Modbus ASCII.

Modbus serial communication can run on two separate serial ports simultaneously. The Modbus serial link characteristics can be different on the two ports. This applies also to the Modbus RTU and ASCII link modes and the unit address.

Documentation on the Modbus serial link messages and the Modbus standard can be obtained from www.modbus.org.

3.2.1.1 Modbus serial link parameters

Serial link setting parameters can be accessed with Parameter Setting tool in PCM600, via the HMI path **Configuration** > **Communication** > **Protocols** > **Modbus**.

Address

Each serial link can be given a separate unit address.

End delay

The end of message delay, or timeout, is used only in the Modbus RTU link mode. According to the Modbus standard, an idle period of 3.5 characters, that is the time it takes to transmit 3.5 characters with the used baud rate, defines the end of a Modbus RTU frame in the RTU mode. This parameter can be given with the accuracy of one character. The default setting is three characters, but the user can increase or decrease the value.

In a multidrop RS-485 Modbus network the unit may detect and receive response messages from other server units. Thus, consider the minimum silent time between the response frame and the beginning of client's next request frame when setting the end delay in Modbus RTU mode.

This parameter has no meaning in the Modbus ASCII link mode.

Start delay

The intraframe delay on serial Modbus RTU link is defined as a silent interval of 3.5 characters. The delay is essential for Modbus devices to recognize the beginning and end of each RTU frame. If the end delay is decreased in the protection relay, the response messages may be transmitted too fast for the link standard especially if slower baud rates are used. The start delay parameter adds idle characters before the transmission, thus increasing the silent interval between the Modbus RTU link frames. The start delay default setting is four idle (silent) characters.

To set the timing properly, consider also how the other server units in a multidrop RS-485 network detect the Modbus traffic between the client and this protection relay.

Port

It is possible to define which serial port is used by separate Modbus instances by the Port parameter, selecting: “COM1” or “COM2”.

If this protocol does not operate as expected, make sure that other serial protocols are not using the COM port as well.

The baud rate is defined on the serial driver side and are therefore located via the HMI paths **Configuration** > **Communication** > **COM1** and **Configuration** > **Communication** > **COM2**.

3.2.1.2 Modbus serial diagnostic counters

Modbus serial diagnostic counters can be viewed via the HMI path **Monitoring** > **Communication** > **Protocols** > **Modbus (n)**.

The counters show complete Modbus protocol link frames and Modbus errors. The serial communication drivers (COM1, COM2) maintain their own counters for lower level serial communication diagnostics.

Table 1: Serial diagnostic counters

Counter	Description
Status	Shows the value "True" if the serial instance is in use. This indicates that the Modbus client is connected and Modbus messages, which are addressed to the device, are received regularly at least with a 15 second interval or faster. In all other cases this value is "False".
Reset counters	True = Reset all diagnostic counters
Received frames	Total amount of received Modbus frames. For example, the Modbus frames that are addressed to this instance.
Transmitted frames	Total amount of transmitted Modbus responses.
Transmitted exc A	Total amount of exception responses 1 and 2. These exception responses usually reveal configuration errors on the Modbus client side. Either the client uses a request function code which is not supported or the requested Modbus point(s) does not exist.
Transmitted exc B	Total amount of exception responses 3. These exceptions usually reveal the protection relay application level rejections. That is, the protection relay application rejects the request at this moment, under the current circumstances. The exception can also mean that the value in the Modbus write request is out of range.
Checksum errors	Total amount of detected Modbus checksum errors. The Modbus instance only calculates checksums of Modbus frames that contain a proper link address. All other incoming Modbus frames are discarded.

3.2.1.3 Troubleshooting serial communication

The diagnostic capabilities can be used for investigating communication problems. If communication cannot be established to the relay, then proceed in this order.

1. Reset the serial driver and Modbus protocol diagnostic counters to make it easier to view the changes.
2. Check the serial driver diagnostic counters. If serial characters are not received, check the cable (Rx line) and the link setup parameters, also on the client side.
3. If serial characters are received, check if whole link frames are also received. Do this first on the driver side.
4. Go over to Modbus diagnostics and check if Modbus link frames are internally received. The serial driver detects every link frame on the line, but the Modbus protocol only reacts to Modbus link frames, which are addressed to its own protocol instance.
5. Check the receive and send delay settings in the relay. If link frames are not received properly there might be character timing problems.
6. Check for receive errors, checksum errors or several retransmissions. If these are found, the line may be noisy.
7. If Modbus link messages are received, check that the response messages are sent to the client.
8. Check the serial driver's transmitted character counter. If it is running, then the relay is transmitting. If the client receives nothing, then check the cable (Tx line).

3.2.1.4 Character framing in different serial link modes

According to the Modbus standard, the character length in the Modbus RTU mode should be 11 bits and in Modbus ASCII mode 10 bits. In addition, according to the standard, it is possible to freely define the character parity: even, odd or no parity. No parity means that the bit length of the serial character is reduced by one. Thus, the character is compensated with an additional stop bit.

Table 2: RTU characters

Coding system	8-bit binary
Bits per charac-ter	1 start bit 8 data bits, the least significant bit is sent first 1 bit for even/odd parity; no bit if parity is not used 1 stop bit if parity is used; 2 stop bits if parity is not used

Table 3: ASCII characters

Coding system	Two ASCII characters representing a hexadecimal number
Bits per charac-ter	1 start bit 7 data bits, the least significant bit is sent first 1 bit for even/odd parity; no bit if parity is not used 1 stop bit if parity is used; 2 stop bits if parity is not used

3.2.2 TCP/IP link

The protection relay operates as a Modbus TCP/IP server. A Modbus TCP/IP client can establish a connection to the protection relay through the standardized TCP socket port 502.

The Modbus TCP/IP interface of the protection relay can be configured to accept up to five simultaneous Modbus client connections. It is possible to grant connections only to the predefined TCP/IP clients. The write authority of the Modbus TCP/IP client is configurable.

Modbus TCP usually shares the Ethernet connection with the other Ethernet based protocols of the protection relay. The number of Ethernet based clients that can be simultaneously connected to the protection relay is restricted.

3.2.2.1 Modbus TCP/IP diagnostic counters

Modbus TCP/IP counters can be viewed via the HMI path **Monitoring** > **Communication** > **Protocols** > **Modbus (n)**.

The counters show the complete Modbus protocol link frames and Modbus errors. The Ethernet communication driver maintains its own counters for lower-level communication diagnostics.

Table 4: TCP/IP diagnostic counters

Counter	Description
Status	Shows the value "True" if the TCP/IP or serial instance is in use. This means that a Modbus client has connected to the TCP socket and Modbus TCP messages are received regularly at least at 15-second intervals or faster. In all other cases this value shows "False".
Reset counters	True = Reset all diagnostic counters
Received frames	Total amount of received Modbus frames.
Transmitted frames	Total amount of transmitted Modbus responses.
Transmitted exc A	Total amount of exception responses 1 and 2. These exception responses usually reveal configuration errors on the Modbus client's side.
Transmitted exc B	Total amount of exception responses 3. These exceptions reveal the protection relay application-level rejections.

Table 5: Common (instance independent) Modbus TCP/IP diagnostic counters

Counter	Description
CnReject no sockets	The number of connection requests that are rejected due to unavailable TCP sockets.
CnReject unregistered	The amount of connection requests that are rejected since the client is not registered.

3.3 Supported Modbus function codes

3.3.1 Application functions

Table 6: Supported application functions

Function code	Name	Description
01	Read coil status	Reads the status of discrete outputs.
02	Read digital input status	Reads the status of discrete inputs.
03	Read holding registers	Reads the contents of output registers.
04	Read input registers	Reads the 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 client and the server.

Table continues on the next page

Function code	Name	Description
15	Force multiple coils	Sets the status of multiple discrete outputs.
16	Preset multiple registers	Sets the value of multiple holding registers.
23	Read/write holding registers	Exchanges holding registers in one query.

3.3.2 Diagnostic functions

The diagnostic functions are only intended for serial communication. However, the serial diagnostic counters can be read, but not reset, via the Modbus TCP/IP interface. The serial line cannot be forced to the listen mode via the Modbus TCP/IP interface.

Table 7: Supported diagnostic subfunctions

Function code	Name	Description
00	Return query data	The data in the query data field is returned (looped back) in the response. The entire response is identical to the query.
01	Restart communication option	The server's peripheral port is initialized and restarted and the communication event counters are cleared. Before this, a normal response is sent provided that the port is not in the listen-only mode. If the port is in the listen only mode, no response will be sent.
04	Force listen only mode	The server is forced to enter the listen only mode for Modbus communication.
10	Clear counters and diagnostic register	All counters and the diagnostic register are cleared.
11	Return bus message count	The response returns the number of messages in the communication system detected by the server since its last restart, clear counters operation or power up.
12	Return bus communication error count	The response returns the number of CRC errors encountered by the server since its last restart, clear counters operation or power up.
13	Return bus exception error count	The response returns the number of Modbus exception responses sent by the server since its last restart, clear counters operation or power up.

Table continues on the next page

Function code	Name	Description
14	Return server message count	The response returns the number of messages addressed to the server or broadcast which the server has processed since its last restart, clear counters operation or power up.
15	Return server no response count	The response returns the number of messages addressed to the server for which a response (a normal response or an exception response) has not been sent since its last restart, clear counters operation or power-up.
16	Return server NACK response count	The response returns the number of messages addressed to the server for which a negative acknowledgement response has been sent.
18	Return bus character overrun count	The response returns the number of messages addressed to the server for which it has not been able to send a response due to a character overrun since its last restart, clear counters operation or power up.

3.3.3 Exception codes

Table 8: Supported exception codes

Function code	Name	Description
01	Illegal function	The server does not support the requested function.
02	Illegal data address	The server does not support the data address or the number of items in the query is incorrect.
03	Illegal data value	A value contained in the query data field is out of range.
04	Server device error	This exception only occurs when a Modbus instance is activated without proper initialization. A relay restart is required.

3.4 Modbus application data

3.4.1 Modbus data implementation

The protection relay is internally modeled according to the IEC 61850 standard. The Modbus protocol is implemented on top of this model. However, not all features of the IEC 61850 data model are available through the Modbus interface.

The Modbus protocol standard defines one-bit digital data and 16-bit register data as RTU application data alternatives. The protocol does not define how this protocol application data should be used by a protection relay application. The usage depends on the protection relay implementation.

As the relay is a freely configurable device, almost all internal IEC 61850 data objects can be mapped to Modbus. The internal IEC 61850 data objects in this device are assigned with potential (empty) Modbus mappings according to the general rules based on the IEC 61850 common data classes (CDC).

Exceptions to the general mapping rules:

- The Beh and Mod attributes of every logical node and some redundant data objects within the generic function blocks are unmapped.
- LEDGGIOx LED states are pre-mapped according to rules depending on the virtual LED configuration.
- The fault record structure has a fixed mapping.

Table 9: Mapping rules

CDC	Description	Attribute [1]	Modbus data type	Areas
SPS	Singe point status	stVal	Mom+MCD readable bit pair	0x1x3x4x
SPC	Controllable single point status	stVal	Mom+MCD readable bit pair	0x1x3x4x
		Oper.ctlVal	One writable bit	0x4x
DPC	Controllable double point status	stVal	Open/Close/Fault bits	0x1x3x4x
		Oper.ctlVal	Direct and SBO writable bits	0x
ACD	Protection activation detection (Start)	general	Mom+MCD readable bit pair	0x1x3x4x
		dirGeneral [2]	Readable enumeral AI	3x4x
		phsA	Mom+MCD readable bit pair	0x1x3x4x

Table continues on the next page

[1] A data object need not contain all data attributes listed for the object class in question.
[2] The enumeral values for the ACD class dirGeneral attribute are 1=Unknown, 2=Forward, 3=Backward, 4=Both.

CDC	Description	Attribute [1]	Modbus data type	Areas
		phsB	Mom+MCD readable bit pair	0x1x3x4x
		phsC	Mom+MCD readable bit pair	0x1x3x4x
		neut	Mom+MCD readable bit pair	0x1x3x4x
ACT	Protection activation (Operate)	general	Mom+MCD readable bit pair	0x1x3x4x
		phsA	Mom+MCD readable bit pair	0x1x3x4x
		phsB	Mom+MCD readable bit pair	0x1x3x4x
		phsC	Mom+MCD readable bit pair	0x1x3x4x
		neut	Mom+MCD readable bit pair	0x1x3x4x
INS	Integer value	stVal	Readable integer AI	3x4x
INC	Controllable integer value	stVal	Readable integer AI	3x4x
		Oper.ctlVal	Writable integer AI	4x
ENS	Enumeral value	stVal	Readable integer AI	3x4x
ENC	Controllable enumeral value	stVal	Readable integer AI	3x4x
		Oper.ctlVal	Writable integer AI	4x
MV	Meas value	mag.f	Readable integer AI	3x4x
		instMag.f		
CMV	Complex meas value	cVal.mag.f instCVal.mag.f	Readable integer AI	3x4x
DEL	Phase-to-phase measurements	phsAB.instCVal.mag.f phsAB.cVal.mag.f	Readable integer AI	3x4x

Table continues on the next page

[1] A data object need not contain all data attributes listed for the object class in question.

CDC	Description	Attribute [1]	Modbus data type	Areas
		phsBC.instCVal.mag.f phsBC.cVal.mag.f	Readable integer AI	3x4x
		phsCA.instCVal.mag.f phsCA.cVal.mag.f	Readable integer AI	3x4x
WYE	Phase-to-ground measurements (filtered)	phsA.instCVal.mag.f phsA.cVal.mag.f	Readable integer AI	3x4x
		phsB.instCVal.mag.f phsB.cVal.mag.f	Readable integer AI	3x4x
		phsC.instCVal.mag.f phsC.cVal.mag.f	Readable integer AI	3x4x
		neut.instCVal.mag.f neut.cVal.mag.f	Readable integer AI	3x4x
		net.instCVal.mag.f net.cVal.mag.f	Readable integer AI	3x4x
		res.instCVal.mag.f res.cVal.mag.f	Readable integer AI	3x4x
WYE [3]	Phase-to-ground measurements (instantaneous)	phsA.instCVal.mag.f phsA.cVal.mag.f	Readable integer AI	3x4x
		phsB.instCVal.mag.f phsB.cVal.mag.f	Readable integer AI	3x4x
		phsC.instCVal.mag.f phsC.cVal.mag.f	Readable integer AI	3x4x
		neut.instCVal.mag.f neut.cVal.mag.f	Readable integer AI	3x4x
		net.instCVal.mag.f net.cVal.mag.f	Readable integer AI	3x4x
		res.instCVal.mag.f res.cVal.mag.f	Readable integer AI	3x4x

Table continues on the next page

[1] A data object need not contain all data attributes listed for the object class in question.
[3] WYE class measurands can be obtained as filtered values and, in some cases, also as instantaneous values. If values are polled fast, instantaneous values show more ripples.

CDC	Description	Attribute [1]	Modbus data type	Areas
SEQ	Sequence of components	c1.instCVal.mag.f	Readable integer AI	3x4x
		c1.instCVal.ang.f	Readable integer AI	3x4x
		c2.instCVal.mag.f	Readable integer AI	3x4x
		c2.instCVal.ang.f	Readable integer AI	3x4x
		c3.instCVal.mag.f	Readable integer AI	3x4x
		c3.instCVal.ang.f	Readable integer AI	3x4x
BCR	Binary counter	actVal	Readable integer AI	3x4x
BCS	Binary control-led step position	valWTr.posVal	Readable integer AI	3x4x

Update rate of analog and indication protocol data

Update rate of protocol data depends on multiple factors that need to be considered when communication engineering is done. This chapter describes the mechanism how process data change is updated to IEC 60870-5-103, IEC 60870-5-104, Modbus and DNP3 communication protocols to process it further.

[1] A data object need not contain all data attributes listed for the object class in question.

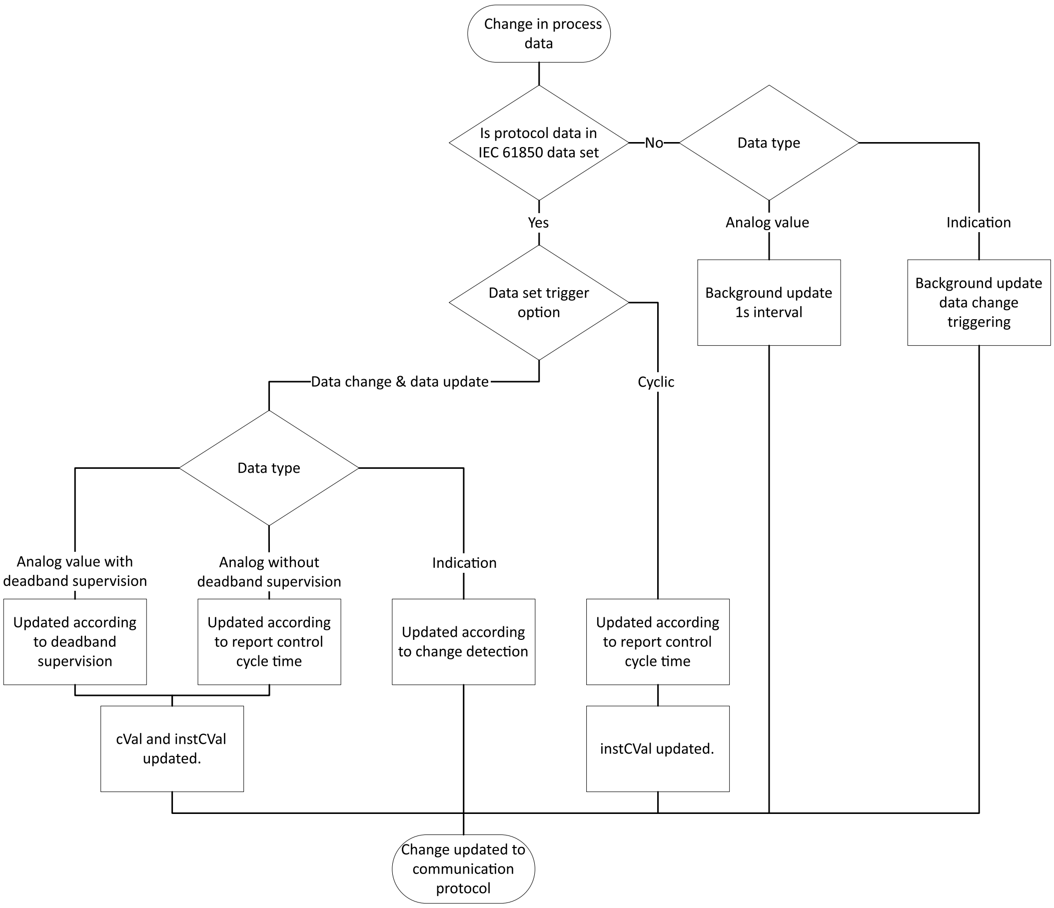

Figure 3: Data flow from process data change to communication protocol

Dataset and Report Control in IEC 61850

Data update rate is dependent of data set and report control engineering in IEC 61850 tool. In most cases default values are suitable, as typically essential measurement and indication data is by default mapped to IEC61860 dataset and report control block. However, it may be necessary to understand all dependencies, if some customization or modification to update rates is needed:

- Data set content
- Trigger options for report control
- Signals selected in communication management (instantaneous or deadband supervised value)

Engineering of dataset and report control blocks is described in IEC 61850 Engineering Guide.

Measurement data

Typically, the commonly used measurement data, such as current/voltage measurement is included in IEC61850 default dataset with data_change + data_update trigger options. In this scenario the update rate depends on the set 'deadband' setting of the measurement function itself. This can be adjusted by the function specific deadband settings, described more in the Technical Manual. In addition, if the default report control trigger options have been changed, and cyclical triggering is used instead of data update + data change, the update rate is the integrity period IntgPd of the report control block. This may be useful for cyclical update of specific measurement data.

Some measurement functions do not have internal deadband supervision in the function. In this case the datasets integrity period IntgPd (see IEC61850 Engineering Manual) defines the update speed of the data.

Third possibility is, that the measurement data does not reside in a IEC61850 dataset. In this scenario the protocol internally polls the analog value once per second. If the one second poll period is more suitable for the application, it is also possible then to remove the specific measurement data points from dataset in IEC61850 tool.

Indication data

Commonly used indication data, such as protection start/trip and other important values are by default included in a IEC61850 dataset. If the protocol has CMT mappings of indication data that does not belong to a IEC61850 dataset, the protocol adds this data automatically to an internal update mechanism that enables still a fast update for the time-critical events.

Application function deadband supervision

The deadband supervision function reports the measured value according to integrated changes over a time period. The sensitivity of reporting can be adjusted with the *X deadband* parameter of a measurement function. By default, deadband supervision defines update rate of analog values to communication protocol. Technical manual describes dead band supervision in more detail.

32-bit-wide integer data

The generic pre-mapping of integer analog values (INS, INC) is generally defined to be 16-bit-wide registers. If the user instead defines a 32-bit register, then the option “Use DA value from system level” should be used. If the source integer value range exceeds 16 bits and the user-mapped 32-bit register uses the option “Regular Modbus register value”, the high word data has initially been masked out.

Change events and time synchronization

The Modbus standard does not define event reporting or time synchronization procedures. Proprietary solutions are introduced in this protection relay to support these functionalities.

Control operations

The Modbus standard defines data types 0X for coils and 4X for holding registers to be used for control operations. This protection relay supports both data types.

Control operations include automatic checking for authorization and local and remote blockings as well as preventing simultaneous controlling by multiple clients.

Application data compatibility

This protection relay is designed to operate with a wide range of Modbus clients spanning from industrial PLCs to substation SCADA devices. The application solutions have been chosen to achieve the highest possible level of compatibility with the systems.

- Application data is readable in many different Modbus memory areas. Digital data is readable as bits or packed bits in registers.

- Primarily 16-bit register sizes are used for measurands. 32 bits are used only in some rare cases.
- The measurands can be freely rescaled by the user.
- The proprietary Modbus event buffer can be read in many different ways. A client can continuously read and log change events in real time or, for example, read an N number of latest events on demand.
- Change detection data can be used as an alternative to the event record reading to catch fast indication data transitions between the client scans.
- The Modbus fault record gives a summary of the captured max-min values and protection stages starting and possibly tripping during a fault.
- The addressing of the application data in the documentation and tools follows the so-called Modbus-PLC addressing principle, where the base address 1 is used. The application data addressing in this protection relay spans between 1 and 9999.
- The Modbus memory-mapped data in the monitoring direction is assembled into user-definable registers or bits in a specific UDR memory area. The data can then be scanned from this area.

3.4.2 Data mapping principles

The available Modbus data in the protection relay can be mapped to a Modbus user-defined area. An unmapped Modbus point does not cause any burden on the protection relay until it is taken into use.

3.4.2.1 Data in monitoring direction

All data in the monitoring direction is available through the 3X and 4X memory areas. This includes the digital indication data which is also readable in the 1X and 0X areas. All register structures are located in the 4X area.

The Modbus data may contain empty bits or registers within the sequential data areas. These bits and registers are intended for possible future expansion. Reading this data does not result in any Modbus exception response. The value in these bits or registers is always zero.

3.4.2.2 One-bit data mapping

All one-bit data in the protection relay is readable either from the 0X or 1X memory area. The Modbus bit point addresses are similar regardless of the memory area. In addition, the same one-bit data can also be read either from the 3X or the 4X area. In this case, the bit values are packed into 16-bit 3X and 4X registers. The bit locations follow a pattern similar to the 0X and 1X locations.

If a one-bit value is located in the 0X or 1X bit address 1800, the same bit value can also be found in the 3X or 4X register 112 (1800 DIV 16) at register bit 8 (1800 MOD 16). This is easier to understand when the address numbers are expressed in the hexadecimal format: 1800 = 0x708, where the register 112 = 0x70 and bit 8 = 0x8.

3.4.2.3 Data in control direction

The protection relay's controls, set points and acknowledgements are mapped in parallel to both Modbus 0X data (coils) and Modbus 4x data (registers). Control points can only be operated one by one.

3.4.3 Digital input data

As the indication signals related to protection applications often change rapidly, the Modbus client may not detect all the changes.

Momentary position and momentary change detection bits

In this protection relay, indications are shown as two adjacent Modbus bits in the Modbus memory map. The two bits represent the momentary position and the momentary change detection state of the indication.

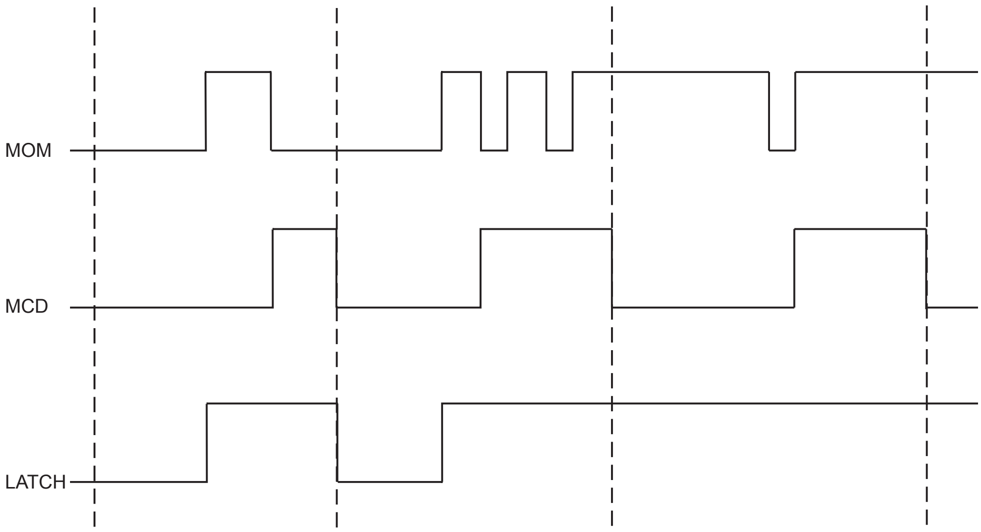

Figure 4: Change detection bit

If the momentary value of an indication bit has changed twice or more since the client last read it, the MCD bit is set to one. When the MCD bit has been read, it is reset to zero. Since the indications usually are 1 (active), it is easy to detect an indication activation by combining the MOM and MCD bits using a logical OR operation (MOM+MCD). The momentary position bit and the MCD bit of a certain indication point always occur as pairs in the Modbus memory map. The momentary bit is always located on an even bit address. MCD bit on the next odd bit address.

The MCD bit states are client-dependent. The MCD bit is only reset for the specific Modbus client that reads it. Thus, other Modbus clients may still receive value 1 from the same MCD bit when it is read.

MOM indication changes are captured in the protection relay's sequential Modbus event buffer. Additionally, the Modbus event buffer provides a time stamp and chronology of indication changes.

Latch bit

Some of the digital MOM+MCD points also incorporate a latch bit alternative. The latch bit hides the MOM and MCD bits and instead returns the result of combining the two bits with a logical OR operation. The MOM+MCD bits are generally used for this.

3.4.3.1 Multiple digital inputs mapping

Digital inputs related to two-bit DPC or DPS objects, for instance circuit breakers and disconnectors, have a multiple mapping in the Modbus address space. The objects' open and close bits are coded as MOM+MCD bit pair entities. The MCD bit reveals if the object has changed its position several times since the Modbus client last scanned it. The open and close bits are also coded using MOM values only, together with a fault bit. The fault bit is set to "1" when the object is in intermediate (00) or faulty (11) position.

Table 10: Bit treatment

Bits	Treatment
Close MOM	One 2-bit entity
Close MCD	
Open MOM	One 2-bit entity
Open MCD	
:	
Close MOM	One 1-bit entity
Open MOM	One 1-bit entity
Faulty position MOM	One 1-bit entity
:	

The MOM values are identical in each entity. The MCD bit is only reset if the MOM bit in the same entity is read.

3.4.3.2 Digital input configuration

Digital input indications are mirrored on several Modbus memory areas. Indications can be accessed in Communication Management in PCM600, under tabs: 1X – discrete inputs, 0X – read-only coils, 3X – input registers and 4X – holding registers. Digital inputs are read-only objects. Writing to the defined 0X and 4X addresses results in an exception response.

The bit address field shows the 1X and 0X Modbus memory addresses on which the data occur as default. The Modbus register address and bit within the register are shown under the 3X and 4X register views. The address field may also be empty, meaning that the object is not located in the Modbus memory at all as default. The indication objects can be taken into use in the user-definable area.

Modbus Communication Management shows only active data objects, that is, the objects that are available in the functions that are activated in the relay configuration.

Table 11: Setting columns in the Modbus CMT view

Setting column	Alternatives	Description
Bit address	Not adjustable	The 0x and 1x bit Modbus memory map address where the object resides as default. If the field is empty, the object is not visible in the Modbus memory map as default. The object can always be further mapped into the user-definable area.
Data category	None	Point does not belong to any user-definable data category.
	1...16	Point belongs to data category number N. If any object belonging to this category changes its value, bit (N-1) in SSR3 gets value 1. The SSR3 bit value is automatically reset when client reads it.
Event enable	Unchecked [1]	No Modbus events generated from this point.
	Checked	Modbus event is generated when the value changes. Accurate event time stamp is inherited from IEC 61850 level.
Rising edge only	Unchecked [1]	Modbus event is generated from both value transitions; from 0 to 1 and from 1 to 0.
	Checked	Modbus event is generated only from 0 to 1 transitions.

3.4.4 Measurand registers

The Modbus measurands are located in the Modbus register area. The measurands are readable from both 3X and 4X areas from the same register addresses.

The Modbus measurands derive from the protection relay's internal, original IEC 61850 filtered measurand values. Modbus register values in this protection relay are always in integer format. Since the internal IEC 61850 values are often represented as decimal numbers, the Modbus stack needs to scale these values to integer format. Thus, there always exists a scale Factor and an offset parameter for each Modbus register value. The user can freely configure these parameters with Communication Management in PCM600.

The formula for calculating the Modbus register value is:

$$Modbus\ value = \left(IEC61850Value \times scaleFactor\right) + Offset$$

(Equation 1)

The range of the original IEC 61850 value can be checked from specification of the function itself, in e.g. Technical Manual.

All frequently updated data are readable from a sequential data area. Additionally, there is a separate sequential data area for measurands and counters with a slow update rate.

[1] Default setting

3.4.4.1 Primary and per-unit values

Measurands originating from CT or VT measurements can be obtained from the protection relay in two ways. They can be viewed either as primary values or as per-unit values.

The primary values are represented internally as decimal numbers. The primary units are [A] for current and [kV] for voltage. The internal representation of the per-unit values is always 1.0 at nominal current or voltage. A typical range for a per-unit value is 0.00...40.00, that is 0 to 40 times nominal. With Communication Management in PCM600, the user can select how these values are presented in the Modbus register. It may be necessary to upscale or downscale the primary values to fit the register's 16-bit integer value. The register's scale factor and offset parameters can be used for this purpose. By default, this protection relay shows per-unit values multiplied by an scale factor of 1000.

If the primary value representation is selected but no CT or VT ratio parameters are configured in the protection relay, the Modbus values remain as per-unit values. Check the protection relay configuration to find out the CT or VT ratio being used.

If scaling of primary values is used, the protection relay must be rebooted if the CT/VT ratio settings are changed. Otherwise, Modbus continues using the old CT/VT settings in its internal scaling algorithms.

3.4.4.2 Register sizes

In most cases the measurands or counters are located in single 16-bit registers. The measurands are either unsigned or signed two's complement values while the counters are always unsigned values.

In some cases, the measurands or counter values can be located in two consecutive registers, thus forming a single 32-bit integer value. The 32-bit value is always coded so that the high word part, that is, the higher 16 bits, is located first in this register address. The low word part, that is, the lower 16 bits, is then always in the next register address.

Register sizes and types are clearly stated in the Modbus memory map list.

3.4.4.3 Register saturation

After a re-scaling operation the Modbus value may exceed the limit of the Modbus register representation. The Modbus value then saturates to the closest max or min value of the register size in question.

3.4.4.4 Rearranging of register value ranges

The pre-defined original Modbus register does not always fit inside the whole value range of the source value.

Example

A counter in the motor protection relay shows the running hours of the motor. The original system counter value has a range of 0…999999 hours. For Modbus, a 16-bit unsigned register is defined for this value. The default scale factor for this Modbus register is defined as x1.

The value range for the 16-bit register is only 0…65535. This means that when the original counter reaches 65535 hours (about seven and half years), the Modbus value saturates (remains locked) at 65535.

There are several ways to overcome this problem.

- It is always possible to assign the Modbus register value to a 32-bit user-definable register. Even when the original register saturates at 65535, the user-definable register continues calculating upwards from this value.
- Rescaling can also be applied on measurands and Modbus counter (integer) values. The original Modbus value can be edited to show full hours, tens of hours or days.
  - If the source hour value is divided by 10, the Modbus value shows tens of hours. This accuracy might be sufficient in many cases. Maximum Modbus register value 65535 would then actually mean 655350 hours, or approximately 75 years. Here the scale factor for Modbus registers is given as a multiplicand. Division by 10 is thus the same as multiplying it by 0.1
  - If the source hour value is divided by 24, the Modbus value shows the number of days. Division by 24 is the same as multiplying by 0.04167.

3.4.4.5 Register configuration

Measurand registers are mirrored on both 3x and 4x Modbus register areas. Registers can be accessed in the Modbus Communication Management tool in PCM600, under tabs: 3x – input registers and 4x – holding registers.

Register values are received from the IEC 61850 system level in two formats. Measurands are usually received as floating point values and counters as integer values. The Modbus register values are always presented as integer values. To make the source floating point value decimals visible in the Modbus register, the received IEC 61850 value can be multiplied with, for example, values 10, 100 or 1000. The Modbus register rounds the integer part and truncates all decimals that are left in the source value.

Table 12: Setting columns in the Modbus Communication Management tool in PCM600

Setting column	Alternatives	Description
Register address	Not adjustable	The 3x and 4x Modbus memory map addresses where the register resides as default. If the field is empty, the register is not visible in the Modbus memory map as default. The register can always be further mapped into the user-definable area.
Data category	0 [1]	Register does not belong to any data category group.
	1...16	Register belongs to data category group number N. If any object belonging to this group changes its value, bit (N-1) in SSR4 gets the value 1. The SSR4

Table continues on the next page

[1] Default setting

Setting column	Alternatives	Description
		bit value is automatically reset when client reads it.
Primary scale factor in use	Unchecked [1]	In case of current and voltage values, the value is now a PU (Per Unit) value, where 1.0 corresponds to 1.0 × Nominal value.
	Checked	In case of current and voltage values, the value is now a primary value, based on the configured CT and VT ratio values. Current is in [A] and voltage is in [kV].
Scale	Any real value	Modbus value is scaled as Value × Scale + Offset. The result is a correctly rounded integer value. If the Modbus value exceeds the register size limit after the scaling, then it saturates at the register max value.
Offset	Any real value	

The *Primary scale factor in use* setting has a meaning only for values that are related to current or voltage measurements. Usually the check box should be visible only for these kinds of values. In case the setting occurs for another type of value, it does not affect the source value.

The primary scale factor setting cannot be applied afterwards in the user-definable register. The setting must be done at the source register value, and the UDR setting *Scale value format* must be configured as “Regular Modbus register value”.

3.4.5 Control operations

The protection relay's outputs can be controlled either through the 0X coil objects or 4X holding register control structures. See the Modbus control objects' memory map for the available control objects.

The control objects in this protection relay are either single-point or double-point control objects.

Single point control object output types

Single point control objects can be either pulse outputs or persistent outputs.

The Modbus client should only write "1" to the pulse outputs. This write operation activates the control operation and there is no need for the Modbus client to write "0" to the object. However, writing "0" is not forbidden. The result is that nothing happens to the control object.

The Modbus client can write both "1" and "0" to the persistent outputs. Therefore, the persistent outputs have two defined levels: "0" and "1".

Most of the outputs in this protection relay are pulse outputs.

Double point control operation modes on IEC 61850 level

This protection relay supports two control models: direct-operate and select-before-operate. The IEC 61850 single point control objects in this protection relay are of direct-operate type. The IEC 61850 double point control objects can be configured either into the direct-operate or select-before-operate mode.

An IEC 61850 double point output cannot support both direct-operate and select-before-operate modes at the same time.

Double point control operations on Modbus level

The double-point select-before-operate mode is usually used for the circuit breaker operations. Modbus incorporates a 30-second fixed select time-out on protocol level. Four controllable objects exist on the Modbus level.

- Select open
- Select close
- Cancel selection
- Operate (=execute) selection

Direct operate of a double-point object consists of two controllable objects.

- Direct open (writing the value "1" opens the circuit breaker)
- Direct close (writing the value "1" closes the circuit breaker)

Direct operate of a double point object is always possible over Modbus. In addition, select-before-operate control is possible if the controllable object's control model is set to "sbo-with-enhanced-security."

3.4.5.1 Control functions

Generally, output objects are controlled one at a time. The protection relay accepts only functions 05 (force single coil) and 15 (force multiple coils), when the 0X coils control structure is used for control operation.

Only one control bit can be operated at a time when the 4X control structures are used.

Exception codes

Only a few exception code alternatives exist for the write coil and write register requests in Modbus:

- 01 = illegal function
- 02 = illegal address
- 03 = illegal value

The exception code 03 is also returned if a command operation is rejected due to other internal reasons. An additional internal reason code for the exception, can be found in the SSR6 register after the command operation.

Internal control rejection reasons with coils may be, for example:

- The client has no write authority.
- The protection relay is in local or OFF state.
- The control operation is already reserved by another client and thus blocked.

If a positive acknowledgement is returned, the control command has been initiated by the protection relay.

3.4.5.2 Control bits

Control bits are write-only coil (0X) data. In addition, they are assigned in parallel to holding registers (4X). Control bits can be accessed in the Modbus Communication Management tool in PCM600, under the tabs 0X – writable coils and 4X – writable registers.

The controllable write-only coil point definitions are separated from the read-only coil point definitions. It is possible to define a write-only object X and a read-only object Y to the same 0X coil bit address Z. While this may be a violation of the 0X Modbus area intention in some cases, it is safe from the relay's point of view. It means that the reading of 0X coil address Z returns the value of object Y, and writing to 0X coil address Z activates object X.

In the configuration example, a physical output on coil address 100 needs to be controlled. The state of the physical output should be readable from the same coil address 100.

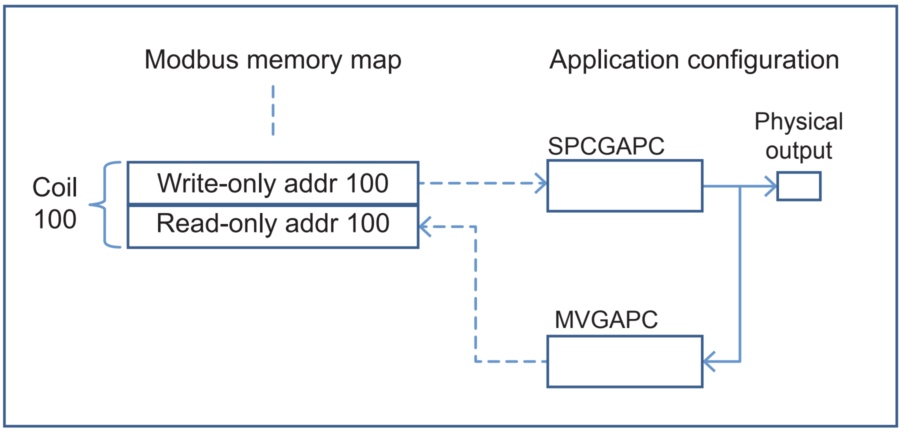

Figure 5: Write and read coil example

The state of the physical output is wired to an `MVGAPC` input. The MVGAPC “Momonly” signal alternative is chosen for the Modbus indication. The MVGAPC user-definable address is configured as “100”. The `SPCGAPC` output is wired to the physical output. The writable coil address is also adjusted to “100”. Alternatively, the physical output state (XGGIO) Modbus signal could be mapped to coil address 100. However, as this signal also contains the MCD bit, the indication would then cover two coil addresses, 100 and 101.

3.4.6 System status registers

The Modbus SSRx 16-bit system status registers are by default located at addresses 40128...40133. See the Modbus memory map for the actual locations of the SSRx registers. The SSRx registers can also be read from the 3X register area from corresponding register addresses. It is also possible to change or entirely remove the predefined SSRx addresses.

Table 13: System status registers

Register	Description	Address
SSR1	Device health	40128
SSR2	Device mode	40129
SSR3	Data available 1	40130
SSR4	Data available 2	40131
SSR5	Device alive counter	40132
SSR6	Last command result	40133

3.4.6.1 SSR1

The device health SSR1 register is located by default at address 40128. The bits in SSR1 are common for all Modbus clients. The bits in SSR1 give an overview of the protection relay's health. If a specific bit in this register is "1", it signifies a warning or an error in the hardware entity in question.

More specific warning and error codes can be read from elsewhere in the Modbus memory. See the Modbus memory map for these register locations.

Table 14: 16-bit SSR1 register

Bit	Description
0	Device global warning
1	Device global error
2	Slot G (X130) error
3	Slot F (X120) error
4	Slot E (X115) error
5	Slot D (X110) error
6	Slot C (X105) error
7	Slot B (X100) error
8	Slot A1 (X000) error
9	Slot A2 (X000) error
10...15	0 = not used

3.4.6.2 SSR2

The device mode SSR2 register is located by default at address 40129. The bit values in SSR2 are common for all Modbus clients. The bits give an overview of the protection relay's mode. For example, bit 6 is activated if the protection relay's configured time synchronization source is lost.

Table 15: 16-bit SSR2 register

Bit	Description
0	Test mode (1= Device is set into test mode)
1...2	Local/Remote states (bit 1= LSB) 00 = Remote – Modbus controls allowed 01 = Station – Modbus controls allowed 10 = Local – Modbus controls not allowed 11 = Off – Modbus controls not allowed
3...5	Active setting parameter setting group (bit 3 = LSB) 001 = Setting group 1 010 = Setting group 2 011 = Setting group 3 100 = Setting group 4 101 = Setting group 5 110 = Setting group 6
6	Protection relay time synchronization failure (1 = Failure)
7	0 = not used
8	Last reset cause (1= Power reset)
9	Last reset cause (1= Watchdog reset)
10	Last reset cause (1= Warm reset)
11...15	0 = not used

3.4.6.3 SSR3

The data available 1 SSR3 register is located by default at address 40130. The bit values in the SSR3 register are Modbus client dependent.

Bits 0 and 1 are set to "1" as long as the client in question has not read out the available Modbus event or fault records.

Bit 4 is set to "1" if any momentary bit has been updated in the Modbus memory map. The bit is reset when the client reads the register.

Bit 5 is set to "1" if any MCD bit has been set in the Modbus memory map. The bit is reset when the client reads the register.

Bit 6 is set to "1" to indicate the device restart. The bit is reset when the client reads this register.

Bit 8 is set to "1" when an event record has been loaded into registers 49252...49262. The bit is reset when the client writes the reset code 4 to the event record selection register 49251.

Bit 9 is set to "1" when a fault record has been loaded into registers starting from 49402. The bit is reset when the client writes the reset code 4 to the fault record selection register 49401.

Table 16: 16-bit SSR3 register

Bit	Description
0	Unread event records available
1	Unread fault records available
2	0 = not used
3	0 = not used
4	Any MOM bit updated
5	Any indication MCD bit set
6	Device restart bit
7	0 = not used
8	Event record ready for reading
9	Fault record ready for reading
10...15	0 = not used

3.4.6.4 SSR4

The data available in SSR4 register is located by default at address 40131. The bit values in SSR4 are Modbus client dependent.

Bits 0...15 in the SSR4 register correspond to different data categories in the regular Modbus memory map. Bit 0 corresponds to data category 1, bit 1 to data category 2 and so on.

If a bit is set to "1", some data belonging to the category in question has changed since the client last scanned the register. The SSR4 bit or bits are cleared when the register is read.

The data category number for all Modbus data is shown in the Modbus memory map. The meaning of the category number is available in a separate table. If the data have not been assigned to any category, the data category number for that data is set to "0". The data category number is freely configurable with Communication Management tool in PCM600. The table below is an example of how the categories can be divided.

Table 17: 16 bit SSR4 register

Bit	Description	Data category
0	Data in category 1 changed	1 = Physical inputs
1	Data in category 2 changed	1 = Protection function start/trip
2	Data in category 3 changed	1 = LED Alarm
3	Data in category 4 changed	1 = New disturbance record available
4	Data in category 5 changed	1 = New demand values
5	Data in category 6 changed	1 = New peak demand values
6	Data in category 7 changed	0
7	Data in category 8 changed	0
8	Data in category 9 changed	0
9	Data in category 10 changed	0
10	Data in category 11 changed	0

Table continues on the next page

Bit	Description	Data category
11	Data in category 12 changed	0
12	Data in category 13 changed	0
13	Data in category 14 changed	0
14	Data in category 15 changed	0
15	Data in category 16 changed	0

3.4.6.5 SSR5

The device alive counter SSR5 register is located by default at address 40132. SSR5 counts upwards from 0 to 65535 and then starts over. The meaning of this register is to assure that the device is actually operating.

3.4.6.6 SSR6

The last command result SSR6 register is located by default at address 40133. This client dependent SSR6 register shows the result of a specific client's last write attempt. This is especially useful if the exception code 03 appears. The client will only see its own results, not the results of other clients. A client with no write authority will receive a 0x0000 value response when reading this register.

Table 18: 16-bit SSR6 register

ClientCmdSEQNo				Cmd State		Resp Type		CMDResultCode							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

ClientCmdSEQNo

Counts the client's control operations from 0000...1111, that is 0...15, and then starts over.

CmdState

00 = No write command has ever been issued by this client

01 = Command in progress

11 = Response Ready

RespType

01 = Unsecured control response

11 = Modbus 03 exception response valid. CMDResultCode is in this case 0. The reason for the 03 exception is an invalid written value.

Table 19: CMDResultCode

Code	Description
0	OK
201	Device in local mode
202	Control operation reserved by another client

Table continues on the next page

Code	Description
203	Select-timeout or Execute/Cancel without select
204	Control operation internally blocked
205	Control operation timed out
250	Other reason

3.4.7 User-definable data

There can be several reasons for defining UD data. For example, the user may want to repack a limited amount of important data into sequential addresses and thereafter only scan this smaller set of data. Especially with serial links, this saves bandwidth and improves response times.

User-definable register can be used if more advanced rescaling and re-manipulating of the regular Modbus register is needed. Many of these features are defined for retrofit purposes and are not needed for normal installations. Some rescaling features are redundant at the moment.

3.4.7.1 User-definable registers

The Modbus register areas 3X and 4X from 1 to 511 can be compiled freely by the user. Almost any regular register data in the Modbus memory map can be made to appear as a register copy in this UDR memory area. The regular Modbus source register is not moved away from its original location and thus it can be read also from the original location.

If extended UDR range is used (addresses over 128), engineer must make sure that the assigned UDR addresses do not overlap with the SSR addresses (e.g. addresses 128-133 assigned for UDR), as this would lead to address overlapping and undefined behavior. If using extended UDR range >128, it is recommended to skip SSR addresses or relocate SSR registers in CMT tool to >512, to avoid address overlapping.

3.4.7.2 User-definable bits

The Modbus bit address areas 0X and 1X from bit 16 to 8191 can be freely compiled by the user. Almost any regular bit data in the Modbus memory map can be made to appear as a bit data copy in this UDB memory area. The regular Modbus source bit data are not moved away from their original location and thus they can be read also from the original location.

Bit 16 is the first valid bit address in the address space because the register and bit addresses overlap and the register addresses start from the register location 1. The bit address 16 is the same as register 1, bit 0.

3.4.7.3 Data exceptions

Some exceptions exist for the Modbus source data concerning the UD mapping.

- None of the system status registers or fixed register structures can appear in the UD area.
- UD registers/bits cannot act as source data for other UD data.

- Modbus source data can only be attached to one UD location.

3.4.7.4 Data properties

The UD data inherits all properties from the source data.

- The memory areas on which the source data are located
- Data pre-scaling in case of registers

3.4.7.5 Unmapped data locations

It is possible to partially scan unmapped register or bit locations, also known as gaps. No exception responses are generated. The unmapped locations always return data value 0.

3.4.7.6 UDR data configuration

It is possible to partially scan unmapped register or bit locations, also known as gaps. No exception responses are generated. The unmapped locations always return data value 0.

3.4.7.7 UDR register value manipulation

UDR values are initially copied from the source register. Thereafter the following manipulations can be applied to the UDR value.

Additional rescaling of the source data value

There are three possibilities for UDR rescaling of the Modbus source register value.

Multiplicative and divisor scaling are similar operations. The defined scale factor can in both cases be a decimal value. For example, multiplicative scaling by 0.25 is the same as divisor scaling by 4.

Ratio scaling can be used in the same manner as multiplicative or divisor scaling, but including an offset. Additionally, it is possible to define new limit ranges for the resulting register value. Value then saturates (stops) at the defined min. and max. values. This may be needed for retrofit purposes, in case the relay value must emulate exactly the same value range as the value from the replaced unit.

For multiplicative and divisor scaling, the register value always saturates at the min. and max. values for the register type in question. The value does not roll over.

Swapping high and low words within a 32-bit register

32-bit registers do not exist in the Modbus standard. A 32-bit register consists of two consecutive 16-bit registers that together form the 32-bit value. There is no official definition for the low-high word order in a 32-bit register. Most vendors, including this relay's, use the order high-low (high word on lower address) as default. If the word order is incorrect, for example in a retrofit case, it is possible to change it.

Specific retrofit UDR manipulations

- Swapping of the byte order within a register word

- Redefining the register bit size. Less than 16 bits are used for the value. The used bits can additionally be left or right justified within the register, that is, they can appear on either the most significant or least significant side of the register.

Table 20: UDR scaling alternatives

Scaling alternative	Setting	Description
No scaling		No change is made to the source-Value
Ratio scaling	UDRScaleArg1 = Min in	Uses all 4 scaling arguments UDRScaleArg1... UDRScaleArg4.
	UDRScaleArg2 = Max in	
	UDRScaleArg3 = Min out	
	UDRScaleArg4 = Max out	
Multiplicative scaling	UDRScaleArg1 = Multiplicand	Uses the argument UDRScaleArg1 (Min in)
Divisor scaling	UDRScaleArg1 = Divisor	Uses the argument UDRScaleArg1 (Min in)

Ratio scaling operation

The sourceValue is to be checked for saturation. If it is less than "Min in", the UDR result value is equal to "Min out". If it is greater than "Max in", the UDR result value is "Max out". Otherwise the UDR result value is calculated as

- X = (MaxOut-MinOut)/(MaxIn-MinIn)
- UDR_ResultValue = X × sourceValue + (MinOut - X × MinIn)

Multiplicative scaling operation

UDR_resultValue = sourceValue × multiplicand

Divisior scaling operation

UDR_resultValue = sourceValue/Divisor

3.4.7.8 UDR register configuration

UDR registers are read-only registers mirrored both on input register (3x) and holding register (4x) areas. After adding a Modbus register value into the user-definable memory area, it is possible to leave it in exactly the same format as the source Modbus value. Alternatively, it is possible to change the user-definable value outlook to better suit the Modbus client.

The UDR value configuration can be divided in three parts.

- Initial properties. Redefine the user-definable register type and how the Modbus value source is retrieved.
- Rescaling. Applies a new scaling to the UDR value.
- Presentation. Modifies the presentation of the UDR value. These modifications are only needed for special retrofit cases.

Columns in the tool do not necessarily appear in the same order as in the table.

Table 21: Configuration columns in Modbus Communication Management tool in PCM600

Setting	Alternatives	Description
Initial properties		
Scale value format	Regular Modbus register value [1]	UDR value initially inherits scaling from the source Modbus register. This includes the source value *Primary scale in use* setting. If checked, the value is automatically presented as a primary value, according to the system CT and VT ratio settings.
	DA value from system level	UDR value is initially the same as the IEC 61850 value received from system level. For current and voltage measurements this means pu value (1.0 corresponds to 1.0 × nominal value).
UDR register size (v2)	Same as the source register [1]	If the Modbus source register is a 16-bit register, then the UDR register is automatically also
		16-bit. This principle applies to 32-bit registers, too.
	16-bit	UDR register is forced to 16-bit, regardless of the source register size.
	32-bit	UDR register is forced to 32-bit, regardless of the source register size.
UDR register type	Same as the source register [1]	If the Modbus register is an unsigned register, also the UDR register is unsigned. This principle applies to signed source registers, too.
	Unsigned	The value is interpreted as a positive value only.
	Signed	The value is interpreted as a signed two's complement value.
Rescaling		

Table continues on the next page

[1] Default value

Setting	Alternatives	Description
Scaling	No scaling [1]	No rescaling of the Modbus UDR value. Scaling arguments are ignored.
	Ratio scaling	Linear scaling of the source value in the source range *Min In...Max In* into UDR value range *Min Out...Max Out*. The scaled UDR value saturates at *Min Out* and *Max Out* values.
	Multiplicative scaling	The source value is multiplied by scaling argument 1 (similar to argument in ratio *Min In*). Other scaling arguments are ignored.
	Divisor scaling	The source value is divided by the scaling argument 1 (similar to argument in ratio *Min In*). Other scaling arguments are ignored.
Min In	Any real value	Scaling argument 1. Used also as the scaling factor for multiplicative and divisor scaling.
Max In	Any real value	Scaling argument 2
Min Out	Any real value	Scaling argument 3
Max Out	Any real value	Scaling argument 4
Presentation: 32-bit register presentation		
Word swap	Unchecked [1]	32-bit register value expressed according to “de facto” standard. Word order High-Low.
	Checked	32-bit register's value word order changed to Low-High
Presentation: Special retrofit value manipulation		
Byte swap	Unchecked [1]	16-bit register value expressed according to Modbus standard. Byte order High-Low.
	Checked	16-bit register value byte order changed to Low-High.

Table continues on the next page

Setting	Alternatives	Description
Bit size	0 [1]	Register bit size reduction not in use. All bits in register used by the value.
	2...31	Register bit size reduced to this value. Saturation limits of the value according to the configured bit size, including possible sign bit.
Justification	Right [1]	After a value bit size reduction, the remaining value bits are aligned to the right (LSB) side of the register.
	Left	After a value bit size reduction, the remaining value bits are aligned to the left (MSB) side of the register.

Multiplicative and divisor scaling are similar operations, because any real value can be used as scaling argument. For example, divisor scaling by 4 is the same as multiplicative scaling by 0.25.

3.4.8 Event records

The protection relay creates a Modbus event record when a momentary digital input bit changes its value. The protection relay then stores the changed Modbus bit location and value into the Modbus event record buffer. The event time tag is also stored into the record. The time tag includes a full time stamp from a year down to milliseconds.

Modbus event generation on/off is selectable for each individual momentary bit in the Modbus memory map. It is possible to define whether events are to be generated from the rising edge or both edges' transitions of the momentary bit.

If the Modbus indication point is mapped to the user-definable Modbus area, then the possible events from this point come from the original Modbus point location. In case the UDR mapped indication point has no original Modbus point location, then the event comes from its UDR point location.

The size of a Modbus event buffer is 500 events. Modbus event buffers are instance dependent. The Modbus event buffer's behavior at overflow is configurable. Either the 500 oldest or the 500 newest events are kept in the buffer.

- In the "Keep oldest" mode, no new Modbus events are collected into the buffer after overflow. Event collecting is resumed only after the event buffer has been emptied of the amount defined by the Event backoff setting. No special overflow event is created at the overflow moment. Instead, the Modbus record sequence number is increased for every new event that was not stored into the buffer. Thus, a jump in the record sequence number reveals the overflow moment and the sequence number difference reveals how many events were lost after overflow.

- In the "Keep newest" mode, new Modbus events continuously overwrite the oldest events in the buffer. The record sequence number reveals that overflow has happened, but it is noticed in the sequence jumps for the oldest events.

Multiple clients support

Several Modbus clients can read out Modbus event records from the protection relay independently of one another. The Modbus event buffer keeps track of where in the event buffer the different clients read at any moment. Clients are identified either by the serial port from where the requests are issued or by the client's IP address in the TCP/IP network. Up to 25 different IP addresses, belonging to both registered and unregistered Modbus clients, can be memorized by the protection relay.

3.4.8.1 Single event record structure

The Modbus event record structure is located at addresses 49251...49262.

Table 22: Event record structure

Address	Register	Values	Comment
49251	Event selection	1...5 and -1...-499	Write register
49252	Sequence Number	0...65535	
49253	Unread records left	0...499	
49254	TimeStamp (Year,Month)		High byte:year, low byte:month
49255	TimeStamp (Day, Hour)		High byte:day, low byte:hour
49256	TimeStamp (Min, Sec)		High byte:min, low byte:second
49257	TimeStamp (Milliseconds)		Word: milliseconds (0...999)
49258	Event type		See separate description
49259	Data Object ID 1	0 or UID high word	See separate description
49260	Data Object ID 2	Modbus address or UID low word	
49261	Data Value	Modbus data value	Value into which object has changed
49262	Data Value		Additional data

The event record can have two different data object identification alternatives. The data object can be identified by the Modbus address where the object resides or it can be identified by a unique id which is platform dependent.

The identification alternative is selected with the Modbus parameter *Event ID*.

3.4.8.2 Single event record reading

As long as there are unread Modbus events available for a Modbus client, bit 0 of Modbus SSR3 register remains "1".

Events are read in two steps. First, the client writes a selection code to the Event selection register at location 49251. The selection code defines the type of read operation that the client wants to perform. If this register is written a second time omitting a read event transaction, this write operation does not succeed and consequentially the relay answers with a message having the exception code: illegal data value. The selected event record is loaded by the protection relay into the following 11 registers from 9252 to 9262. Second, the client reads out the 11 registers in one multiple register read operation.

Event records can be read by using two commands, function 6 for the write operation and function 3 for the read operation, or by using function 23 that includes write and read operations in the same transaction.

If event records are read by using two commands, the positive confirmation to the write select operation tells the client that an event record has been loaded for reading. Another way to detect the positive confirmation is by monitoring the state of SSR3 bit 8.

Selection code 1: Reading the oldest unread record

When writing the selection code 1, the protection relay first checks the client. If the client has read events before, the protection relay knows which internal event has been sent to this specific client during the last reading. The protection relay then loads the next event, that is the oldest unread, into the next 11 registers. If this is the first time the client reads events from the protection relay, the oldest event of the Modbus event buffer is loaded into the 11 event record registers.

Selection code 2: Reading the oldest stored record

Selection code 2 always forces the event reading to go back to the oldest event in the Modbus event buffer. The oldest event record is then loaded into the 11 event record registers. After the client has read out this record, the next record becomes the oldest unread. The client can continue with the selection code 1 by reading out the oldest unread event record again.

Selection code -1...-499

A negative selection code, that is a 16-bit two's complement value, defines how many records backwards from the newest event the event record reading is to be moved. For example, the ten latest events could be read out at any time by first selecting -10, reading out the event and then continuing with the selection code 1 to read out the nine additional event records. There can be 500 event records altogether.

Selection code 3: Resetting the event read pointer

The write selection 3 is not followed by a read operation. The selection 3 means that there are no unread records in the Modbus event buffer left for the client in question, that is, the buffer is cleared. The next new event that is logged into the Modbus event buffer becomes the first unread record for this specific client.

Selection code 4: Resetting SSR3 bit 8

The write selection 4 is not followed by a read operation. The selection code only resets bit 8 in SSR3.

If event records are read by using two commands, the client can re-read the 11 event record registers as many times as it wants. As long as no new selection write operation is performed, the contents of the 11 event record registers are not changed.

Selection code 5: Reading the newest stored record

Selection code 5 always forces the event reading to go to the newest event in the Modbus event buffer. The newest event record is then loaded into the 11 event record registers. The next new event that is logged into the Modbus event buffer becomes the first unread record for this specific client. Older events can be read with a negative selection code.

3.4.8.3 Other event record registers

Sequence number

Every Modbus event record is given a sequence number. The sequence number runs from 1 to 65535 and then rolls over to 1 again. The client can check that the sequence numbers of the recorded data are sequential. During the event buffer overflow the client can notice a jump in the sequence numbers when some event records are lost. The gap between the new and the previous sequence number reveals how many event records have been lost.

Unread records left

This register shows how many unread event records still remain for the client in question at a particular moment.

Time stamp registers

Time stamps are either in local time or UTC time. The time stamp alternative is selected with a Modbus parameter.

Time stamp registers usually hold two data values in the high and low byte of the registers. High byte value = RegisterValue DIV 256, Low byte value = RegisterValue MOD 256. The Milliseconds register is an exception as it contains the milliseconds 0...999 coded as such.

Event type

This register contains information to interpret the event data correctly.

Table 23: Information in the 16-bit register

Bit	Meaning	Values	
15	Event time stamp format	0 = Local time	1 = UTC time
14	Time stamp source	0 = Internal application	1 = Modbus stack
13	Clock not synchronized	0 = Synchronized	1 = Time not synchronized

Table continues on the next page

Bit	Meaning	Values	
12	Clock failure	0 = Clock OK	1 = Clock failure
11	Reserved	0	
10	Reserved	0	
9	Reserved	0	
8	Data object ID type	0 = Modbus address	1 = UID data
7...0	Data value identifica-tion	00000000 = One-bit indication 00000010 = Two-bit indication 00000100... 00001010 = Integer value 11111111 = re-served	00000001 = ACD in-dication 00000011 = SEC indication+add data 00001001 = reserved 00001011...

Event time stamp format bit 15

The time stamp format can be selected with a Modbus parameter via the HMI or Parameter Setting.

Event time stamp source bit 14

The time stamp can be generated by the protection relay application (accurate time) or by Modbus. If generated by Modbus, the change values are detected by the Modbus background scan task. Since there is a latency time between the value change and the time when Modbus detects the change, the time stamp is not accurate in this case.

Clock not synchronized bit 13

The quality information bit is set in the protection relay's real-time clock if the protection relay has not been synchronized.

Clock failure bit 12

The quality information bit is set in the protection relay's real-time clock if the clock has a severe failure. Do not rely on this time stamp.

Data object ID bit 8

The coding alternatives of the data object ID registers 1 and 2 are the Modbus address or UID. The coding alternatives cannot occur simultaneously in the protection relay but are selected and configured at the system setup phase. The default setting is "Modbus address".

The UID code is 32 bits wide and occupies both registers 9259 and 9260. The word order is high/low. The UID code refers to the functional design of the protection relay platform in which the object resides. Shortly, it means that the UID code is equal in all the platform protection relays in which the same functional design and the same Modbus object is used.

While UID is supported for backwards compatibility, it is not unambiguous for all indication objects.

Different Modbus indications originating from the same IEC 61850 data attribute have an identical UID. It is therefore recommended to use the Modbus address as the identification instead of the UID.

Data value identification bits 5..0

Coding of the event data value is one bit, two bits or 32 bits. The coding depends on the IEC 61850 common data class which is the origin of the Modbus data in question.

Table 24: Modbus event value alternatives

Object derived from IEC 61850 Class	Meaning	One Bit Data Value	Two Bit Data Value	32-bit Data Value
SPS	Single Point Status	X		
SPC	Single Point Status of a controllable object	X		
DPS	Dual Point Status		X	
DPC	Dual Point Status of a controllable object		X	
ACT	Trip status	X		
ACD	Start status	X		
INS/INC	Integer status			X

Table 25: Interpretation of the one-bit data value

Register 49261 binary coded value	Meaning
xxxx.xxxx.xxxx.xxx0	Object in OFF position
xxxx.xxxx.xxxx.xxx1	Object in ON position

Table 26: Interpretation of the two-bit data value

Register 49261 binary coded value	Meaning
xxxx.xxxx.xxxx.xx00	Object in intermediate position (changing)
xxxx.xxxx.xxxx.xx01	Object in ON (close) position
xxxx.xxxx.xxxx.xx10	Object in OFF (open) position
xxxx.xxxx.xxxx.xx11	Object in faulty position

In case of a DPS/DPC two-bit event value (Data value identification = 2), the data object ID registers 49259 and 49260 always refer to the Modbus address or UID of the CLOSE momentary value bit.

Table 27: Interpretation of the integer status data value

Register address	Meaning
49261	Higher 16-bit part of the 32-bit integer value
49262	Lower 16-bit part of the 32-bit integer value

Table 28: Interpretation of the ACD data

Register address	Meaning
49261	xxxx.xxxx.xxxx.xxx0 Object in OFF position
	xxxx.xxxx.xxxx.xxx1 Object in ON position
49262	xxxx.xxxx.xxxx.xx00 Start in unknown direction
	xxxx.xxxx.xxxx.xx01 Start in forward direction
	xxxx.xxxx.xxxx.xx10 Start in backwards direction
	xxxx.xxxx.xxxx.xx11 Start in both directions

Table 29: Interpretation of the SEC data

Register address	Meaning
49261	xxxx.xxxx.xxxx.x000 Unknown security violation
	xxxx.xxxx.xxxx.x001 Critical security violation
	xxxx.xxxx.xxxx.x010 Major security violation
	xxxx.xxxx.xxxx.x011 Minor security violation
	xxxx.xxxx.xxxx.x100 Warning
49262	Security violations counter, 16 bits

The original SEC cnt attribute is actually defined as a 32-bit counter. The Modbus event shows the least significant 16 bits of that counter, that is 0...65535.

3.4.8.4 Multiple event records reading

It is possible to read out up to 10 sequential event records in one event select/read transaction. The number of sequential event records to be returned for reading is written to the Num of records register (49250) in front of the selection register. The number can be written once before a select/read event transaction. If this number is never written, only one event record is returned.

If the Modbus client requests multiple event records, the returned records should also be read out by the client. One record consists of 11 registers, two records of 22 registers and so on. The read length must thus be adjusted depending on the number of records requested.

The selection/read operation is otherwise identical to the single-record read case. The next records to be returned always continue from the last record in the previous read operation.

Reading out more event records than are available in the internal event buffer

The requested amount of event records is always returned for reading. For example, if 10 event records are requested, but the protection relay only contains five event records, the last valid event record is repeated (duplicated) in the last five event records returned. The easiest way to detect the duplication is to check the sequence number of the event records. The sequence numbers remain similar to the duplicated event records.

Extended event record structure

The extended register addresses are 49250 and 49263...49361.

Table 30: Extended event record structure with the maximum of 10 event records

Address	Register	Values	Description
49250	Num of records	1...10	Write: Number of Event structures
49251	Selection		Write: Selection code
49252	Sequence Number 1		Event record 1
49253	Unread records left 1		
49254	TimeStamp 1		
49255	TimeStamp 1		
49256	TimeStamp 1		
49257	TimeStamp 1		
49258	Event Type 1		
49259	Data Object Id 1_1		
49260	Data Object Id 2_1		
49261	Data Value 1		
49262	Data Value 1		
49263	Sequence Number 2		Event record 2
49264	Unread records left 2		
49265	TimeStamp 2		
49266	TimeStamp 2		
49267	TimeStamp 2		
49268	TimeStamp 2		
49269	Event Type 2		
49270	Data Object Id 1_2		
49271	Data Object Id 2_2		
49272	Data Value 2		
49273	Data Value 2		
49274	Sequence Number 3		Event record 3
:	:	:	:
49285	Sequence Number 4		Event record 4
:	:	:	:
49296	Sequence Number 5		Event record 5
:	:	:	:
49307	Sequence Number 6		Event record 6
:	:	:	:
49318	Sequence Number 7		Event record 7
:	:	:	:

Table continues on the next page

Address	Register	Values	Description
49329	Sequence Number 8		Event record 8
:	:	:	:
49340	Sequence Number 9		Event record 9
:	:	:	:
49351	Sequence Number 10		Event record 10
49352	Unread records left 10		
49353	TimeStamp 10		
49354	TimeStamp 10		
49355	TimeStamp 10		
49356	TimeStamp 10		
49357	Event Type 10		
49358	Data Object Id 1_10		
49359	Data Object Id 2_10		
49360	Data Value 10		
49361	Data Value 10		

3.4.9 Fault records

A fault record is created by the protection relay as a set of registrations during a detected fault period. The registration includes the selected peak values and the global duration value of the protection stages, the time of recording and a sequence number for the fault record.

The size of the protection relay's internal Modbus fault record buffer is 100 records. The 100 latest fault records are at any time readable from the protection relay. The Modbus fault record is Modbus dependent and the data organization and buffer size differ from the protection relay's initial system level registrations. When the Modbus fault record buffer becomes full, the protection relay overwrites the oldest records in the buffer.

Multiple clients support

Several Modbus clients can independently of one another read out the Modbus fault records from the protection relay. The Modbus fault record buffer keeps track of where in the buffer the different clients are reading at the moment. Clients are identified either by the serial port from where the requests are issued or by the client's IP address in the TCP/IP network.

3.4.9.1 Fault record structure

The protection relay's fault record structure starts from location 49401 and consists of a fixed header part and an application data part. The application data part is always protection relay type specific. The whole fault record including the protection relay specific application data part is found in the Modbus memory map section.

Table 31: Header part of the record structure

Address	Register	Values	Comment
49401	Fault record selection	1...4 and -1...-99	Write register
49402	Sequence Number	0...65535	
49403	Unread records left	0...99	
49404	TimeStamp (Year,Month)		High byte:year, low byte:month
49405	TimeStamp (Day, Hour)		High byte:day, low byte:hour
49406	TimeStamp (Min, Sec)		High byte:min, low byte:second
49407	TimeStamp (Milliseconds)		Word: milliseconds (0...999)
49408	Time quality		
49409	From this location onwards starts the Fault record application data.		

Fault record application data part

Table 32: Application data part of the record structure

RegA	Type	Scale	IEC 61850 name	Description	Values
			LD0.FLTRFRC1		
9409	u32	1	.OpCnt.stVal	Fault record number (high)	0..999999
9410				(low word)	
9411	i16	1	.ProFnc.stVal	Protection func code 1	-128…127
			LD0.FLTRFRC2		
9412			.ProFnc.stVal	Protection func code 2	0..1
			LD0.FLTRFRC1		
9413	u16	100	.Hz.mag.f	Frequency	30.00...80.00 [Hz]
9414	u16	100	.StrDur.mag	Start duration	0...100.00 [%]
9415	u32	1000	.StrOpTm.mag	Operate time (high)	0.000...999999.999 [s]
9416				(low word)	
9417	u32	1000	.FltPtR.mag	Fault point resistance	0.00...1000000.00 [ohm]
9418				(Low word)	
9419	u32	1000	.PhReact.mag	Fault point reactance	0.0...1000000.0 [ohm]
9420				(Low word)	
9421	u32	1000	.FltDiskm.mag	Fault distance	0.00...3000.00 [x pu]
9422				(Low word)	
9423	u16	1	.ActSetGr.stVal	Active setting group	1...6

Table continues on the next page

RegA	Type	Scale	IEC 61850 name	Description	Values
9424	u16	1	.ShotPntr.stVal	AR shot pointer value	1…7
9425					0
9426					0
9427					0
9428	u16	1000	.Max50DifAA.mag	Max. diff. current phs A	0.000...80.000 [x pu]
9429	u16	1000	.Max50DifAB.mag	Max. diff. current phs B	0.000...80.000 [x pu]
9430	u16	1000	.Max50DifAC.mag	Max. diff. current phs C	0.000...80.000 [x pu]
9431	u16	1000	.Max50RstAA.mag	Max. bias current phs A	0.000…50.000 [x pu]
9432	u16	1000	.Max50RstAB.mag	Max. bias current phs B	0.000…50.000 [x pu]
9433	u16	1000	.Max50RstAC.mag	Max. bias current phs C	0.000…50.000 [x pu]
9434	u16	1000	.DifAPhsA.mag	Diff. current phs A	0.000...80.000 [x pu]
9435	u16	1000	.DifAPhsB.mag	Diff. current phs B	0.000...80.000 [x pu]
9436	u16	1000	.DifAPhsC.mag	Diff. current phs C	0.000...80.000 [x pu]
9437	u16	1000	.RstAPhsA.mag	Bias current phs A	0.000…50.000 [x pu]
9438	u16	1000	.RstAPhsB.mag	Bias current phs B	0.000…50.000 [x pu]
9439	u16	1000	.RstAPhsC.mag	Bias current phs C	0.000…50.000 [x pu]
9440	u16	1000	.DifARes.mag	Diff. current Io	0.000...80.000 [x pu]
9441	u16	1000	.RstARes.mag	Bias current Io	0.000…50.000 [x pu]
9442	u16	1000	.Max50APh-sA.mag	Max. current phs A(1)	0.000...50.000 [xIn]
9443	u16	1000	.Max50APhsB.mag	Max. current phs B(1)	0.000...50.000 [xIn]
9444	u16	1000	.Max50APhsC.mag	Max. current phs C(1)	0.000...50.000 [xIn]
9445	u16	1000	.Max50ARes.mag	Max. current Io(1)	0.000...50.000 [xIn]
9446	u16	1000	.APhsA.mag	Current phs A(1)	0.000...50.000 [xIn]
9447	u16	1000	.APhsB.mag	Current phs B(1)	0.000...50.000 [xIn]
9448	u16	1000	.APhsC.mag	Current phs C(1)	0.000...50.000 [xIn]

Table continues on the next page

RegA	Type	Scale	IEC 61850 name	Description	Values
9449	u16	1000	.ARes.mag	Current Io(1)	0.000...50.000 [xIn]
9450	u16	1000	.AResClc.mag	Current Io-Calc(1)	0.000...50.000 [xIn]
9451	u16	1000	.APsSeq.mag	Current Pos-Seq(1)	0.000...50.000 [xIn]
9452	u16	1000	.ANgSeq.mag	Current Neg-Seq(1)	0.000...50.000 [xIn]
			LD0.FLTRFRC2		
9453	u16	1000	.Max50APh-sA.mag	Max. current phs A(2)	0.000...50.000 [xIn]
9454	u16	1000	.Max50APhsB.mag	Max. current phs B(2)	0.000...50.000 [xIn]
9455	u16	1000	.Max50APhsC.mag	Max. current phs C(2)	0.000...50.000 [xIn]
9456	u16	1000	.Max50ARes.mag	Max. current Io(2)	0.000...50.000 [xIn]
9457	u16	1000	.APhsA.mag	Current phs A(2)	0.000...50.000 [xIn]
9458	u16	1000	.APhsB.mag	Current phs B(2)	0.000...50.000 [xIn]
9459	u16	1000	.APhsC.mag	Current phs C(2)	0.000...50.000 [xIn]
9460	u16	1000	.ARes.mag	Current Io(2)	0.000...50.000 [xIn]
9461	u16	1000	.AResClc.mag	Current Io-Calc(2)	0.000...50.000 [xIn]
9462	u16	1000	.APsSeq.mag	Current Pos-Seq(2)	0.000...50.000 [xIn]
9463	u16	1000	.ANgSeq.mag	Current Neg-Seq(2)	0.000...50.000 [xIn]
			LD0.FLTRFRC3		
9464	u16	1000	.Max50APh-sA.mag	Max. current phs A(3)	0.000...50.000 [xIn]
9465	u16	1000	.Max50APhsB.mag	Max. current phs B(3)	0.000...50.000 [xIn]
9466	u16	1000	.Max50APhsC.mag	Max. current phs C(3)	0.000...50.000 [xIn]
9467	u16	1000	.Max50ARes.mag	Max. current Io(3)	0.000...50.000 [xIn]
9468	u16	1000	.APhsA.mag	Current phs A(3)	0.000...50.000 [xIn]
9469	u16	1000	.APhsB.mag	Current phs B(3)	0.000...50.000 [xIn]
9470	u16	1000	.APhsC.mag	Current phs C(3)	0.000...50.000 [xIn]
9471	u16	1000	.ARes.mag	Current Io(3)	0.000...50.000 [xIn]
9472	u16	1000	.AResClc.mag	Current Io-Calc(3)	0.000...50.000 [xIn]
9473	u16	1000	.APsSeq.mag	Current Pos-Seq(3)	0.000...50.000 [xIn]
9474	u16	1000	.ANgSeq.mag	Current Neg-Seq(3)	0.000...50.000 [xIn]
			LD0.FLTRFRC1		
9475	u16	1000	.PhVPhsA.mag	Voltage phs A(1)	0.000...4.000 [xUn]
9476	u16	1000	.PhVPhsB.mag	Voltage phs B(1)	0.000...4.000 [xUn]

Table continues on the next page

RegA	Type	Scale	IEC 61850 name	Description	Values
9477	u16	1000	.PhVPhsC.mag	Voltage phs C(1)	0.000...4.000 [xUn]
9478	u16	1000	.PPVPhsAB.mag	Voltage phs AB(1)	0.000...4.000 [xUn]
9479	u16	1000	.PPVPhsBC.mag	Voltage phs BC(1)	0.000...4.000 [xUn]
9480	u16	1000	.PPVPhsCA.mag	Voltage phs CA(1)	0.000...4.000 [xUn]
9481	u16	1000	.VRes.mag	Voltage Uo(1)	0.000...4.000 [xUn]
9482	u16	1000	.VZro.mag	Voltage Zero-Seq(1)	0.000...4.000 [xUn]
9483	u16	1000	.VPsSeq.mag	Voltage Pos-Seq(1)	0.000...4.000 [xUn]
9484	u16	1000	.VNgSeq.mag	Voltage Neg-Seq(1)	0.000...4.000 [xUn]
			LD0.FLTRFRC2		
9485	u16	1000	.PhVPhsA.mag	Voltage phs A(2)	0.000...4.000 [xUn]
9486	u16	1000	.PhVPhsB.mag	Voltage phs B(2)	0.000...4.000 [xUn]
9487	u16	1000	.PhVPhsC.mag	Voltage phs C(2)	0.000...4.000 [xUn]
9488	u16	1000	.PPVPhsAB.mag	Voltage phs AB(2)	0.000...4.000 [xUn]
9489	u16	1000	.PPVPhsBC.mag	Voltage phs BC(2)	0.000...4.000 [xUn]
9490	u16	1000	.PPVPhsCA.mag	Voltage phs CA(2)	0.000...4.000 [xUn]
9491	u16	1000	.VRes.mag	Voltage Uo(2)	0.000...4.000 [xUn]
9492	u16	1000	.VZro.mag	Voltage Zero-Seq(2)	0.000...4.000 [xUn]
9493	u16	1000	.VPsSeq.mag	Voltage Pos-Seq(2)	0.000...4.000 [xUn]
9494	u16	1000	.VNgSeq.mag	Voltage Neg-Seq(2)	0.000...4.000 [xUn]
			LD0.FLTRFRC1		
9495	u16	100	.MaxTmpRl.mag	PTTR thermal level	0.00...99.99
9496	u16	100	.AMaxNgPs.mag	PDNSPTOC1 ratio I2/I1	0.00...999.99 [%]
9497	i16	100	.DifA-NAngVN.mag	Angle Uo-Io(1)	-180.00…180.00 [deg]
9498	i16	100	.Di-fAAAngVBC.mag	Angle UBC-IA(1)	-180.00…180.00 [deg]
9499	i16	100	.DifABAngV-CA.mag	Angle UCA-IB(1)	-180.00…180.00 [deg]
9500	i16	100	.DifACAng-VAB.mag	Angle UAB-IC(1)	-180.00…180.00 [deg]
			LD0.FLTRFRC2		
9501	i16	100	.DifA-NAngVN.mag	Angle Uo-Io(2)	-180.00…180.00 [deg]
9502	i16	100	.Di-fAAAngVBC.mag	Angle UBC-IA(2)	-180.00…180.00 [deg]

Table continues on the next page

RegA	Type	Scale	IEC 61850 name	Description	Values
9503	i16	100	.DifABAngV-CA.mag	Angle UCA-IB(2)	-180.00…180.00 [deg]
9504	i16	100	.DifACAng-VAB.mag	Angle UAB-IC(2)	-180.00…180.00 [deg]
			LD0.FLTRFRC1		
9505	i16	100	.HzRteChg.mag	Frequency gradi-ent	-10.00...10.00 [Hz/s]
9506	i16	100	.CondNeut.mag	Conductance Yo	-1000.00...1000.00 [mS]
9507	i16	100	.SusNeut.mag	Susceptance Yo	-1000.00...1000.00 [mS]
9508	i16	100	.PPLoopRis.mag	Fault loop resist-ance	-1000.00...1000.00 [ohm]
9509	i32	100		(Low word)	
9510	i32	100	.PPLoo-pReact.mag	Fault loop reac-tance	-1000.00...1000.00 [ohm]
9511				(Low word)	
9512	u16	1000	.CBClrTm.mag	Breaker clear time	0.000...3.000 [s]

3.4.9.2 Fault record reading

As long as there are unread fault records available for the Modbus client in question, bit 1 of the Modbus SSR3 register remains "1".

The fault record reading is done in two steps. First, the client writes a selection code to the Fault record selection register at location 49401. The selection code defines the type of read operation that the client wants to do. The selected fault record is loaded by the protection relay into the following N registers (49402- NNNN). Second, the client reads out these registers in one multiple register read operation.

The fault records can be read by using two commands, function 6 for the write operation and function 3 for the read operation, or by using function 23 that includes write and read operations in the same transaction.

If the fault records are read by using two commands, the positive confirmation to the write select operation tells the client that a fault record has been loaded for reading. Another way to detect the positive confirmation is by monitoring the state of SSR3 bit 9.

Fault record structure length

Since the application data part is protection relay type dependent, the length of the fault record structures varies in different types of protection relays. A client can read out more Modbus registers than are actually coded in one structure when reading out the data structures. The maximum read amount is 80 Modbus registers. The additional trailing registers contain the value 0. The Modbus protocol gives an exception response if the client tries to read out too few registers from the fault record structure.

Selection code 1: Reading the oldest unread record

When writing the selection code 1, the protection relay first checks the client. If the client has been reading fault records before, the protection relay knows which internal fault record was sent to this specific client during the last reading. The protection relay then loads the next fault record, that is the oldest unread, into the registers following the selection register. If this is the first time the client reads fault records from the protection relay, the oldest fault record of the Modbus fault record buffer is given to the client.

Selection code 2: Reading the oldest stored record

The selection code 2 always forces the fault record reading to go back to the oldest fault record stored in the buffer. The oldest fault record is then loaded into the registers following the selection register. After the client has read out this record, the next record becomes the oldest unread. The client can continue by reading out the oldest unread fault records again with the selection code 1.

Selection code -1...-99

A negative selection code, that is a 16 bit two's complement value, defines how many records backwards from the newest fault record the reading is to be moved. For example, the ten latest fault records can be read out at any time by first selecting -10, reading out the record and then continuing with the selection code 1 to read out the nine additional records.

Selection code 3: Resetting the fault record read pointer

The write selection code 3 is not followed by a read operation. The selection 3 means that there are no unread records in the Modbus fault record buffer left for the client in question, that is, the buffer is cleared. The next new fault record that is logged into the Modbus fault record buffer becomes the first unread record for this specific client.

Selection code 4: Resetting SSR3 bit 9

The write selection 4 is not followed by any read operation. The selection code only resets bit 9 in SSR3.

If the fault records are read by using two commands, the client can re-read the given fault record registers as many times as it wants. As long as no new selection write operation is performed, the contents of the fault record registers are not changed.

3.4.9.3 Other fault record registers

Sequence number

Every fault record is given a sequence number. The sequence number runs from 1 to 65535 and then rolls over to one again. The client can check that the sequence numbers of the recorded data are sequential. During the fault record buffer overflow the client can notice a jump in the sequence numbers when some fault records are lost. The gap between the new and the previous sequence number reveals exactly how many records have been lost.

Unread records Left

This register shows how many unread fault records still remain unread for the client in question at a particular moment.

Time stamp registers

The time stamp registers usually hold two data values in the high and low byte of the registers: High byte value = RegisterValue DIV 256, Low byte value = RegisterValue MOD 256. An exception is the milliseconds register which contains the milliseconds 0...999 coded as such. Time stamp also contains a time quality register.

Time quality

Table 33: Information in the 16 bit (bits 15..0) register

Bit	Meaning	Values	
15	Event time stamp format	0 = Local time	1 = UTC time
14	Time stamp source	0 = Internal application	1 = Modbus stack
13	Clock not synchronized	0 = Synchronized	1 = Time not synchronized
12	Clock failure	0 = Clock OK	1 = Clock failure
11...0	Reserved	0	

Event time stamp format bit 15

The time stamp format can be selected with a Modbus parameter via the HMI or Parameter Setting.

Event time stamp source bit 14

The time stamp can be generated by the protection relay application, that is accurate time, or by Modbus. If generated by Modbus, the change values are detected by the Modbus background scan task. Since there is a latency time between the value change and the time when Modbus detects the change, in this case the time stamp is not accurate.

Clock not synchronized bit 13

The quality information bit is set in the protection relay's real-time clock if the protection relay has not been synchronized.

Clock failure bit 12

The quality information bit is set in the protection relay's real-time clock if the clock has a severe failure. Do not rely on this time stamp.

3.4.10 Parameter setting group selection

In the current protection relay version, parameter setting group selection and reading is defined through regular Modbus registers. Formerly this was achieved by a special 4x register 9231. The new register has the default address 2301. See the protection relay documentation for the number of available setting groups. Exception response 3 is given if the written value is out of range or the setting group changing is blocked.

3.4.11 Time synchronization

The real-time clock inside the protection relay runs in UTC time. However, the local time is also known by the protection relay through the time parameter settings. With Modbus the protection relay time can be viewed and set either in local time or UTC time.

Two identical time structures are available in the Modbus memory map: the protection relay's local time at location 49201...49208 and the internal UTC time at the location 49211...49218.

Time synchronization can be given either to the local time structure or to the UTC time structure.

The protection relay accepts Modbus time synchronization only if the *Synch source* setting is set to "Modbus". The parameter can be set via **Configuration** > **Time** > **Synchronization** > **Synch source**.

3.4.11.1 Real-time clock structure

Table 34: Modbus real-time clock structure

Modbus address		Register contents	Values
Local Time	UTC Time		
49201	49211	Control register	0...2
49202	49212	Year	2000...9999
49203	49213	Month	1...12
49204	49214	Day	1...31
49205	49215	Hour	0...23
49206	49216	Minutes	0...59
49207	49217	Seconds	0...59
49208	49218	Milliseconds	0...999

3.4.11.2 Writing to real-time structures

The Modbus time synchronization can be done in several ways. Over the serial interface, the host's synchronization write can be given with the Modbus broadcast address "0". Thus, all protection relays in the same serial network can be synchronized at the same time.

Method 1: Synchronization in one step

Registers 49201...49208 (49211...49218) should be written in one multiple registers preset request (function 16) by a Modbus TCP/IP client or by a serial interface client. The protection relay's Modbus address or the Modbus broadcast address can be used with the serial interface. If the clock is written in one step, the write value of the register 49201 (49211) is not checked by the protection relay.

Method 2: Synchronization in three steps

1. The client reserves the time synchronization by writing value "1" to the register 49201 (49211). If necessary, check that the reservation value is zero at the beginning. If the time synchronization writing is already reserved by another client, the protection relay returns the exception response 03.
2. The client writes the time structure to the protection relay. This can be done in one transaction or alternatively each register can be written separately.
3. The client sets the clock by writing "2" into the register 49201 (49211). When the value "2" is written, the timesync registers are latched onto the protection relay's internal clock and the reservation in 49201 (49211) is released.

The Modbus broadcast address cannot be used with the synchronization method 2.

There is an internal timeout for the clock setting. The time synchronization reservation is released if the clock is not set within two minutes. The client can abort the time synchronization at any time by writing "0" into the register 49201 (49211). In that case the real-time clock is not set at all.

Other Modbus clients can read the currently running real-time clock even if the time writing is reserved by another client.

3.4.12 Reset time structure

The time and cause of the protection relay's last reset are stored into this structure. The reset time is taken directly from the protection relay's RTC at the startup. The clock might not be accurate and the data can be corrupted.

Table 35: Reset time structure

Address	Register	Values	Comment
49221	TimeStamp (Year,Month)		High byte:year, low byte:month
49222	TimeStamp (Day,Hour)		High byte:day, low byte:hour
49223	TimeStamp (Min,Sec)		High byte:min, low byte:seconds
49224	TimeStamp (Millisec-onds)		Word: milliseconds
49225	Time Quality	See Time quality table	
49226	Cause of reset	1 = Power reset	
		2 = Watchdog reset	
		3 = Warm reset	

Table 36: Time quality

Bit	Meaning	Values
15	Time format	0 = Local time
		1 = UTC time
14	Time source	0 = Internal (RTC)
13	RTC not synchronized	0 = RTC synchronized
		1 = Not synchronized
12	RTC Failure	0 = RTC OK
		1 = RTC failure
11...0	Not used	0

3.5 SPA application data

3.5.1 SPA protocol

The Modbus protocol includes an internal Modbus ASCII to SPA protocol converter. The SPA protocol is available as a resident extension to the Modbus ASCII protocol and it operates only through the asynchronous serial interface. This interface provides connection to gateway products requiring SPA.

The SPA protocol reuses the settings available for the Modbus ASCII link. The link characteristics are similar in the SPA protocol and Modbus ASCII protocol standards (1 start bit, 7 data bits, even parity, 1 stop bit).

The Modbus unit number setting is reused as the SPA server number. No additional protocol mode parameter exists. The protection relay's Modbus ASCII link detects the incoming client messages and automatically adjusts itself according to the protocol. This switching happens seamlessly, restarting the protection relay is not required.

The SPA conversion from or to Modbus ASCII is done according to the predefined rules. The basic principle is that all data available for the Modbus interface is also available for the SPA protocol. If data is not available through the Modbus interface, it is not available for the SPA protocol either.

3.5.2 Supported SPA data

The protection relay supports general SPA data.

- The protection relay responds with its device type to SPA fiction reading (RF:).
- The protection relay responds to SPA event reading (RL:) and SPA event backup reading (RB:).
- SPA time synchronization messages are accepted and the protection relay's real-time clock can be synchronized from this source. Both WT: and WD: messages are supported.
- The protection relay accepts the WC:0 acknowledge messages from the client.

3.5.3 Reading of SPA data

SPA data derives directly from the protection relay's Modbus data. All Modbus data in monitoring direction can be read through Modbus input or holding registers. Modbus registers are theoretically addressed 1…65535, but in practice the highest available address is 9999.

16-bit wide Modbus registers can contain either one analog value or a set of maximum 16 packed indication bits. In some cases two consecutive Modbus registers have been defined to contain one 32-bit analog value.

The SPA channel number corresponds to the Modbus register address in the SPA read messages.

The SPA data type and number define the value response format.

3.5.3.1 Reading of one register

The examples below are from reading the register 138. The SPA server number is 25. The actual value in register 138 is assumed to be 52342 (decimal). It does not matter from SPA point of view if the register value is formed from one measurand or if it is a register packed with indication bits. The SPA stack does not know the origin of the register value. The system engineer selects the most appropriate read method.

(I1/16) Read the register as separate SPA bits:

```
>25R138I1/16:CC <25D:1/0/0/1/1/1/1/0/0/0/1/1/0/0/1/1:CC
```

(I30) Read the register as one signed 16-bit decimal value:

```
>25R138I30:CC <25D:-13193:CC
```

(I31) Read the register as one unsigned 16-bit decimal value:

```
>25R138I31:CC <25D:52342:CC
```

(I32) Read the register as one 16-bit hexadecimal value:

```
>25R138I32:CC <25D:CC76:CC
```

It is also possible to read several consecutive 16-bit registers in one SPA read message. The register values are all in the same format. For example:

```
>25R138/140I32:CC <25D:CC76/C845/C772:CC
```

3.5.3.2 Reading of two registers

Since some Modbus analog values can reside in two consecutive register pairs, it is possible to read these values in one read message. For example, registers 146 and 147 could contain one 32-bit value (0xF025A476). The read message should always be directed to the first register address of the register pair.

(I40) Reading the registers as one 32-bit signed decimal value:

```
>25R146I40:CC <25D:-265968522:CC
```

(I41) Reading the registers as one 32-bit unsigned decimal value:

```
>25R146I41:CC <25D:4028998774:CC
```

(I42) Reading the registers as one 32-bit hexadecimal value:

```
>25R146I42:CC <25D:F025A476:CC
```

The SPA implementation cannot check that the two Modbus registers actually belong together as one value. Any two independent consecutive 16-bit registers can be read freely in one 32-bit data value packet.

3.5.3.3 Special reading of indication bits

Many Modbus indications (one bit data) in the device are coded as MOM and MCD bit pairs. The client detects a fast indication 0→1→0 change, if the two bits are combined with logical OR operation. The SPA protocol automatically includes this combination of MOM and MCD bits.

Reading of 16-bit registers

As an example, register 223 contains 8 MOM + MCD bit pairs in 16 separate bits. The register can be read as a regular 16-bit register revealing every bit.

Example:

```
>25R223I1/16:CC <25D:1/0/0/1/1/1/1/0/0/0/1/1/0/0/1/1:CC
```

(I21/28) It is also possible to read the logical OR operation result of every bit pair. Since there are 8 pairs, the OR results are 8 bits:

```
>25R223I21/I28:CC <25D:1/1/1/1/0/1/0/1:CC
```

(I51) Reading is also possible to perform in unsigned decimal form:

```
>25R223I51:CC <25D:175:CC
```

(I52) Or in hexadecimal form:

```
>25R223I52:CC <25D:AF:CC
```

Reading of 32-bit registers

If the two consecutive 16-bit registers (32 bits) contain all 16 MOM + MCD bit pairs, then the logical OR operation result (=16 bits) can be read out in one query. Assuming the registers are 223 and 224.

(I61) In unsigned decimal form:

```
>25R223I61:CC <25D:43567:CC
```

(I62) In hexadecimal form:

```
>25R223I62:CC <25D:AA2F:CC
```

3.5.4 Writing of SPA data

Writing SPA data refers here either to writing to a Modbus coil (one bit data) or to a writable Modbus register (up to 16-bit data). SPA write operation can only be performed to one Modbus object at a time.

There are rules for writing SPA data.

- When writing to coils, SPA channel 0 is used. Data type is O, and the data number corresponds to the coil address to be written. Value can be “0” or “1” (depends on the object).
- When writing to registers, SPA channel 1 is used. Data type is O, and the data number corresponds to the register address to be written. Value can be 0...65535 (depends on the object).

If the written data address does not exist in the protection relay, there is a negative SPA response (NAK) number 6. If the value is rejected by the Modbus object then the negative SPA response number is 8. A successful writing is positively acknowledged.

Writing the value “1” to the Modbus coil address 2052 (if the SPA channel is 0, the channel number is omitted in the command message):

```
>25WO2052:1:CC <25A:CC
```

Writing the value “7” to the Modbus register address 9051:

```
>25W1O9051:7:CC <25A:CC
```

3.5.5 SPA events

The MOM + MCD bits available on the Modbus interface in the protection relay detect the fast indication data transitions. It is possible to receive also SPA events from the protection relay interface. The SPA events are derived from user enabled Modbus events.

Every Modbus indication data can separately be enabled to produce a time tagged event on either both its transitions (ON-OFF) or only on the activating transition (ON). If this is done, then the Modbus events are automatically converted to SPA events for the SPA interface. The Modbus/SPA events need not to be enabled for the MOM + MCD operations to work. The Modbus/SPA event generation can be optimized to include only the required events.

3.5.5.1 Event outlook

Events include a normal SPA time stamp, in seconds and milliseconds. The event channel is the Modbus register address on which the indication bit resides. The event code is E0…E31 depending on the bit in the register indication (0…15) and the state into which the indication has changed (0…1).

For example, an event code like 23.543 35E23 means that the event occurred at the time 23.543 and that register 35 bit 11 (23 div 2) changed to value “1” (23 mod 2). Register 35 bit 11 corresponds to bit address 571 (35 × 16 + 11).

If there are several SPA events pending, the interface responds with maximum 5 events every time.

3.5.6 SPA time synchronization

The client has to send at least one complete time synchronization message WD: before the protection relay starts accepting shorter time synchronization messages WT:. The protection relay accepts time synchronization on either the SPA broadcast address 900 or on its own address.

3.5.7 SPA ZC-302 configuration

The following chapters contain information on how to set up Modbus data for SPA-polling when using SPA ZC-302. Knowledge of the organization of Modbus data objects in the protection relay is a prerequisite.

3.5.7.1 Use of Modbus user-defined area for SPA

The protection relay's Modbus data can be relocated to the user-defined Modbus memory area. Therefore it is possible to build up the packed set of measurands and indications. The user-defined register area can also be accessed from the SPA protocol.

The SPA protocol standard defines that the highest possible SPA channel number is 999. This corresponds to Modbus register address 999. Some SPA client applications can however access higher channel numbers than 999. If this is the case, then the SPA server application responds with channel numbers higher than 999. In Modbus data mapping of this relay there are measurand registers that are as default located on higher register addresses than 999. If this causes problems in the

SPA communication, the desired measurands can be remapped to a user-defined register area having lower register addresses.

When packing MOM + MCB indication bit pairs into Modbus user-defined registers, it is also possible to map MOM-only bits into the same register. To poll the register using the automatic MOM + MCD bit OR-ing feature, map the MOM-only bit to an even bit location in the register, leaving the corresponding MCD location unmapped. The OR-ing is done for this bit pair, but since the unused MCD bit is always 0, the result is always according to the MOM bit state.

If the Modbus indication point is mapped to the user-defined Modbus area, then the SPA events from this point come from the original Modbus point location. In case the UDR mapped indication point has no original Modbus point location, then the event comes from its UDR point location.

3.5.7.2 Modbus user definable area set up for SPA ZC-302 polling

In this example phase currents IL1...IL3 are mapped into user-definable register addresses 0xB…0xC (11…13). Some indication bits that have been assembled into register 0xD (14), LEDPTRC1.Str.general and LEDPTRC1.Op.general, correspond to the Start/Operate LED states in the protection relay. CB1 Open, Close and Fault are all Circuit Breaker1 position data. Furthermore it is assumed that this data is mapped in Profibus offset addresses 4 and onwards.

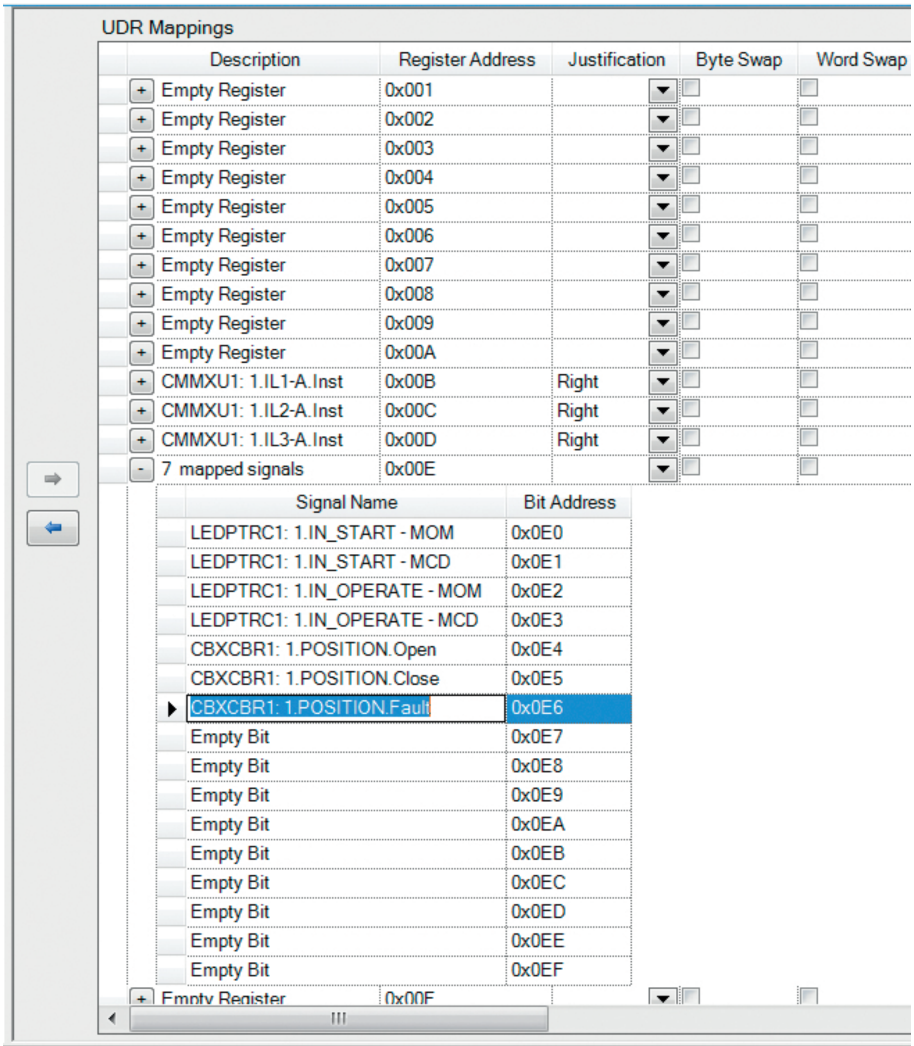

Figure 6: User definable area mappings

The SPA commands for poll in values (SPA address 10) to fetch ZC-302 Profibus octet-offset values into 4 and onwards.

- `>10R11I31`: Read register 11 as an unsigned integer … map to Profibus offset 4...5
- `>10R12I31`: Read register 12 as an unsigned integer … map to Profibus offset 6...7
- `>10R13I31`: Read register 13 as an unsigned integer … map to Profibus offset 8...9

The bits in register 14 can be fetched as an integer value.

- `>10R14I31`: Read register 14 as an unsigned integer … map to Profibus offset 10...11

Profibus offsets 4...11 values assembled sequentially are achieved more efficiently by reading two Modbus registers at a time using only two SPA polls.

- `>10R11I42:CC` Read registers 11 and 12 as a 32-bit hex value, into Profibus offsets 4...7
- `>10R13I42:CC` Read registers 13 and 14 as a 32-bit hex value, into Profibus offsets 8...11

The value format was changed into hex values. This creates a shorter SPA response data value which saves SPA communication bandwidth. I41 (unsigned 32-bit integer) also can be used. SPA ZC-302 accepts both of these value types.

It does not matter how the separate data values have been polled into its Profibus memory area from the SPA ZC-302 point of view. It is not necessary to poll each object in one by one. This means that the User Definable Modbus area is sequentially filled up with the Modbus values to be transferred to Profibus offset octets. Then the data can be polled into SPA ZC-302 using two register reads at a time.

3.6 Advanced protocol customization

Different advanced customization features are supported by the protocol; these features can be enabled via the Prtl Customization configuration parameter.

The Prtl Customization parameter is a 31-bit bitmask. Currently, bit zero (0) is in use. The setting value for each customization feature is calculated by the following formula: 2^(bit number).

3.6.1 Modbus Advanced customization features

The following table summarizes each of the advanced features supported by the protocol.

Table 37: Modbus Advanced customization features

Parameter	Value (Range)	Unit	Step	Default	Description		
Prtl Customization	0	-	1	0	Bit number (n)	Value (2*n)	Feature
						0	All customization is disabled
					0	1	Disables TCP keep-alive messages sending
					1 - 30	2 - 1073741824	Reserved for future use

In order to enable multiple features is it necessary to calculate the value for each required feature and add all of the afterwards.

The use of advanced customization features can be monitored in **Monitoring** > **Communication** > **Protocols** > **Modbus (n)**.

4 Modbus parameters and diagnostics

4.1 Parameter list

The Modbus parameters can be accessed with PCM600 or via the HMI path **Configuration** > **Communication** > **Protocols** > **Modbus (n)**.

Some parameters are not visible in the “Basic” setting visibility mode. To view all parameters use “Advanced” setting visibility mode in Parameter Setting in PCM600 and HMI.

4.1.1 Modbus settings

Table 38: Non group settings

Parameter	Values (Range)	Unit	Step	Default	Description
Operation	1=on 5=off			5=off	Enable or disable this protocol instance
Port	1=COM 1 2=COM 2 3=Ethernet - TCP 1			3=Ethernet - TCP 1	Port selection for this protocol instance. Select between serial and Ethernet based communication.
Mapping selection	1...2		1	1	Chooses which mapping scheme will be used for this protocol instance.
Address	1...254		1	1	Unit address
Link mode	1=RTU 2=ASCII			1=RTU	Selects between ASCII and RTU mode. For TCP, this should always be RTU.
TCP port	1...65535		1	502	Defines the listening port for the Modbus TCP server. Default = 502.
Parity	0=none 1=odd 2=even			2=even	Parity for the serial connection.
Start delay	0...20		1	4	Start delay in character times for serial connection
End delay	0...20		1	4	End delay in character times for serial connections
CRC order	0=Hi-Lo 1=Lo-Hi			0=Hi-Lo	Selects between normal or swapped byte order for checksum for serial connection. Default: Hi-Lo.

Table continues on the next page

Parameter	Values (Range)	Unit	Step	Default	Description
Client IP				0.0.0.0	Sets the IP address of the client. If set to zero, connection from any client is accepted.
Write authority	0=Read only 1=Disable 0x write 2=Full access			2=Full access	Selects the control authority scheme
Time format	0=UTC 1=Local			1=Local	Selects between UTC and local time for events and timestamps.
Event ID selection	0=Address 1=UID			0=Address	Selects whether the events are reported using the MB address or the UID number.
Event buffering	0=Keep oldest 1=Keep newest			0=Keep oldest	Selects whether the oldest or newest events are kept in the case of event buffer overflow.
Event backoff	1...500		1	200	Defines how many events have to be read after event buffer overflow to allow new events to be buffered. Applicable in "Keep oldest" mode only.
ControlStructPWd 1				****	Password for control operations using Control Struct mechanism, which is available on 4x memory area.
ControlStructPWd 2				****	Password for control operations using Control Struct mechanism, which is available on 4x memory area.
ControlStructPWd 3				****	Password for control operations using Control Struct mechanism, which is available on 4x memory area.
ControlStructPWd 4				****	Password for control operations using Control Struct mechanism, which is available on 4x memory area.
ControlStructPWd 5				****	Password for control operations using Control Struct mechanism, which is available on 4x memory area.
ControlStructPWd 6				****	Password for control operations using Control Struct mechanism, which is available on 4x memory area.
ControlStructPWd 7				****	Password for control operations using Control Struct mechanism, which

Table continues on the next page

Parameter	Values (Range)	Unit	Step	Default	Description
					is available on 4x memory area.
ControlStructPWd 8				****	Password for control operations using Control Struct mechanism, which is available on 4x memory area.
Prtl Customization	0...2147483647		1	0	Customization parameter. Please, refer to the protocol manual.

4.2 Monitored data

The Modbus monitored data can be accessed with Parameter Setting in PCM600 or via the HMI path **Monitoring** > **Communication** > **Protocols** > **Modbus (n)**.

4.2.1 MODBUS Monitored data

Table 39: Monitored data

Name	Type	Values (Range)	Unit	Description
Customization Mode	Enum	0=Off/Normal 1=By Parameter 2=By File		Protocol Customization Mode
Reset counters	BOOLEAN	0=False 1=True		Reset counters
Received frames	INT32	-1...2147483646		Number of received frames
Transmitted frames	INT32	-1...2147483646		Number of transmitted frames
Transmitted exc A	INT32	-1...2147483646		Number of transmitted exception responses 01 and 02
Transmitted exc B	INT32	-1...2147483646		Number of transmitted exception responses 03
Checksum errors	INT32	-1...2147483646		Number of checksum errors
CnReject no sockets	INT32	-1...2147483646		Number of rejected connections due to no sockets available
CnReject unregistered	INT32	-1...2147483646		Connection rejected due to unregistered client

5 Glossary

ACD	Start/pickup status
ACT	1. Application configuration tool in PCM600 2. Trip status in IEC 61850
AI	Analog input
ASCII	American standard code for information interchange
Cnt	Counter
CT	Current transformer
DPC	Double-point control
DPS	Double-point status
EMC	Electromagnetic compatibility
Ethernet	A standard for connecting a family of frame-based computer networking technologies into a LAN.
HMI	Human-machine interface
IEC	International electrotechnical commission
IEC 61850	International standard for substation communication and modeling
INS/INC	Integer status
IP	Internet protocol
IP address	A set of four numbers between 0 and 255, separated by periods. Each server connected to the Internet is assigned a unique IP address that specifies the location for the TCP/IP protocol.
LED	Light-emitting diode
LSB	Least significant bit
MCD	Momentary change detect
MMS	1. Manufacturing message specification 2. Metering management system
Modbus	A serial communication protocol developed by the Modicon company in 1979. Originally used for communication in PLCs and RTU devices.
Modbus ASCII	Modbus link mode. Character length 10 bits.
Modbus memory map	Allocation of accessible protocol data
Modbus RTU	Modbus link mode. Character length 11 bits.
Modbus TCP/IP	Modbus RTU protocol which uses TCP/IP and Ethernet to carry data between devices
MOM	Momentary position
PCM600	Protection and control IED manager
PLC	Programmable logic controller

PU	Protection unit
RS-485	Serial link according to EIA standard RS485
RTC	Real-time clock
RTU	Remote terminal unit
Rx	Receive/Received
SBO	Select-before-operate
SCADA	Supervision, control and data acquisition
SEC	Security violation
SI	Sensor input
SPA	Strömberg protection acquisition. ABB proprietary serial client-server protocol used in substation automation for point-to-point communication.
SPC	Single-point status of a controllable object
SPS	Single-point status
SSR1	System status register for device health
SSR2	System status register for device mode
SSR3	System status register for data available 1
SSR4	System status register for data available 2
SSR5	System status register for device alive counter
SSR6	System status register for last command result
SSRx	System status register
TCP	Transmission control protocol
TCP/IP	Transmission control protocol/Internet protocol
Tx	Transmit/Transmitted
UD	User-definable
UDB	User-definable bit
UDR	User-definable register
UID	Unique ID
UTC	Coordinated universal time
VT	Voltage transformer

ABB Distribution Solutions
Digital Substation Products
P.O. Box 699
FI-65101 VAASA, Finland
Phone +358 10 22 11

abb.com/mediumvoltage
abb.com/relion
go.abb/digitalsubstations

2NGA001860 A