

RELION® PROTECTION AND CONTROL

# 620 series ANSI

## Modbus Communication Protocol Manual







Document ID: 1MAC458836-IB  
Issued: 2019-05-29  
Revision: C  
Product version: 2.0

© Copyright 2019 ABB. All rights reserved

# Copyright

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

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

## **Trademarks**

ABB and Relion are registered trademarks of the ABB Group. All other brand or product names mentioned in this document may be trademarks or registered trademarks of their respective holders.

## **Warranty**

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

[www.abb.com/substationautomation](http://www.abb.com/substationautomation)

## 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 anti virus 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.

## Conformity

This product complies with the directive of the Council of the European Communities on the approximation of the laws of the Member States relating to electromagnetic compatibility (EMC Directive 2014/30/EU) and concerning electrical equipment for use within specified voltage limits (Low-voltage directive 2014/35/EU). This conformity is the result of tests conducted by ABB in accordance with the product standards EN 50263 and EN 60255-26 for the EMC directive, and with the product standards EN 60255-1 and EN 60255-27 for the low voltage directive. The product is designed in accordance with the international standards of the IEC 60255 series and ANSI C37.90. This product complies with the UL 508 certification.

---

## Table of contents

<b>Section 1</b>	<b>Introduction.....</b>	<b>5</b>
	This manual.....	5
	Intended audience.....	5
	Product documentation.....	6
	Product documentation set.....	6
	Document revision history.....	6
	Related documentation.....	7
	Symbols and conventions.....	7
	Symbols.....	7
	Document conventions.....	7
<b>Section 2</b>	<b>Modbus standard.....</b>	<b>9</b>
	Serial communication.....	9
	Ethernet communication.....	9
	Application data implementation .....	10
	Terms and definitions.....	10
	Documentation.....	11
<b>Section 3</b>	<b>Vendor-specific implementation.....</b>	<b>13</b>
	Modbus link alternatives.....	13
	Serial link.....	13
	Modbus serial link parameters.....	13
	Modbus serial diagnostic counters.....	15
	Character framing in different serial link modes.....	15
	TCP/IP link.....	16
	TCP/IP interface configuration.....	16
	Modbus TCP/IP diagnostic counters.....	18
	Supported function codes.....	19
	Application functions.....	19
	Diagnostic functions.....	19
	Exception codes.....	20
	Application data.....	21
	Modbus data objects.....	21
	Modbus data implementation.....	21
	Data mapping principles.....	22
	Data in monitoring direction.....	22

# Table of contents

---

One-bit data mapping.....	23
Data in control direction.....	23
Digital input data.....	23
Multiple digital inputs mapping.....	24
Measurand registers.....	25
Primary and per-unit values.....	25
Register sizes.....	26
Time of update.....	26
Control operations.....	27
Control functions.....	28
Control operations through 4X register structures.....	28
Additional control operation features.....	30
System status registers .....	31
SSR1.....	32
SSR2 .....	32
SSR3 .....	33
SSR4 .....	34
SSR5 .....	35
SSR6 .....	35
User-definable data.....	36
User definable registers.....	36
User definable bits.....	36
Data exceptions.....	37
Data properties.....	37
Unmapped data locations.....	37
UDR data configuration.....	37
UDR register value manipulation.....	37
Event records .....	38
Single event record structure.....	39
Single event record reading.....	40
Other event record registers.....	41
Multiple event records reading.....	45
Fault records .....	47
Fault record structure.....	48
Fault record reading.....	48
Other fault record registers.....	50
Parameter setting group selection.....	51
Time synchronization .....	51
Real-time clock structure.....	52
Writing to real-time structures.....	52

---

Device information.....	53
ASCII character coding.....	54
ASCII string syntax.....	54
Reset time structure.....	54
Accessing of non-protocol-mapped data.....	55
<b>Section 4 Modbus parameters and diagnostics.....</b>	<b>57</b>
Parameter list.....	57
Monitored data.....	58
<b>Section 5 Glossary.....</b>	<b>61</b>



---

## Section 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

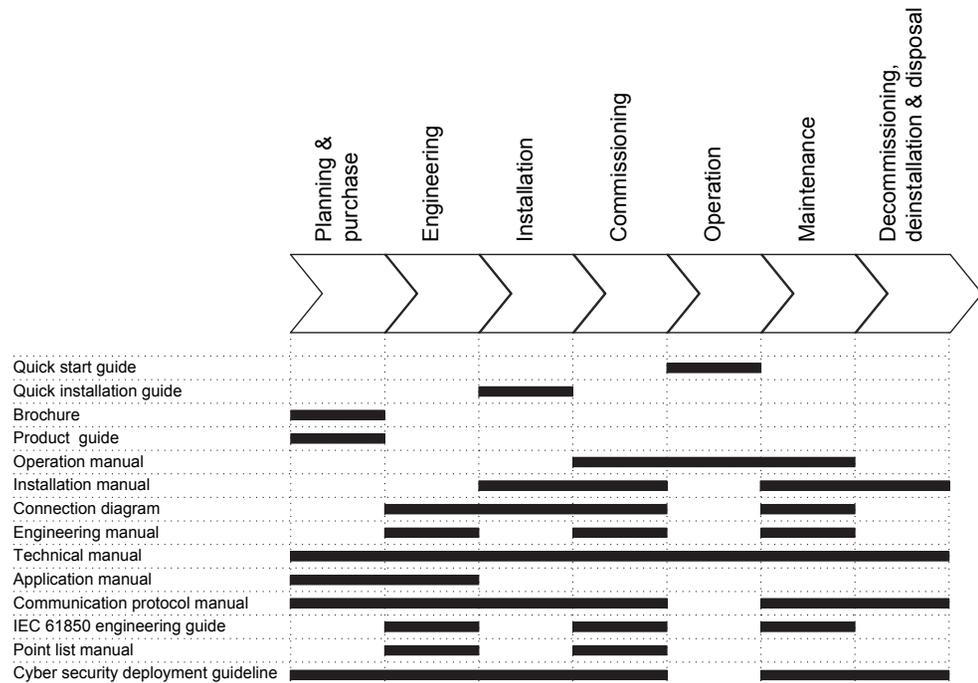


Figure 1: The intended use of documents during the product life cycle



Product series- and product-specific manuals can be downloaded from the ABB Web site <http://www.abb.com/relion>.

### 1.3.2 Document revision history

Document revision/date	Product series version	History
A/2012-10-31	2.0	First release
B/2019-05-17	2.0	Content updated
C/2019-05-29	2.0	Content updated



Download the latest documents from the ABB Web site  
<http://www.abb.com/substationautomation>.

### 1.3.3 Related documentation

Product-specific point list manuals and other product series- and product-specific manuals can be downloaded from the ABB Web site  
<http://www.abb.com/substationautomation>.

## 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.
- Push button navigation in the LHMI menu structure is presented by using the push button icons.

- 
- To navigate between the options, use  and .
  - Menu paths are presented in bold.  
Select **Main menu/Settings**.
  - WHMI menu names are presented in bold.  
Click **Information** in the WHMI menu structure.
  - LHMI messages are shown in Courier font.  
To save the changes in nonvolatile memory, select Yes and press .
  - 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 "Enabled" and "Disabled".
  - Input/output messages and monitored data names are shown in Courier font.  
When the function picks up, the PICKUP output is set to TRUE.
  - Dimensions are provided both in inches and mm. If it is not specifically mentioned, the dimension is in mm.

---

## Section 2 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 has been 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](http://www.modbus.org).

### 2.1 Serial communication

Modbus is a master-slave protocol when it is used over serial communication networks. This protection relay implements the slave 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 master unit on a Modbus serial network. The Modbus master unit communicates with one Modbus slave unit at a time. Usually the master reads, or scans, data from the slaves cyclically. The master can also write data or give commands to the slave units. Each slave unit has a unique unit address. Thus, the master can identify the slave with which it communicates. The Modbus standard also defines the possibility for Master broadcast transmissions.

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

### 2.2 Ethernet communication

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

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. It is possible to configure the protection relay to only accept socket connection requests from known client IP addresses.

## 2.3 Application data implementation

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

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 master systems.

## 2.4 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 addressing, 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.

---

Refer also to the Modbus protocol standard documentation that can be found for free at [www.modbus.org](http://www.modbus.org).

## 2.5 Documentation

The ANSI Modbus point list manuals cover all the 620 series variants.



A newer SW version of the same 620 series configuration may contain additional Modbus points.



---

## Section 3 Vendor-specific implementation

### 3.1 Modbus link alternatives

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

#### 3.1.1 Serial link

Modbus serial communication requires that the protection relay variant is equipped with a serial interface card at the slot X000. 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 concerning the Modbus serial link messages and the Modbus standard can be obtained from [www.modbus.org](http://www.modbus.org).

#### 3.1.1.1 Modbus serial link parameters

Serial link setting parameters can be accessed with PST, WHMI or via the LHMI path **Configuration/Communication/Modbus**.



In the LHMI parameter names end either with number 1 or with number 2. The numbers refer to two separate serial ports, that is the instances 1 and 2.

#### 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 slave units. Thus, consider the minimum silent time between the response frame and the beginning of master'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 this IED, the response messages may be transmitted too fast according to the link standard especially true with slower baud rates. 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 slave units in a multidrop RS-485 network detect the Modbus traffic between the master and this IED.

### Serial port

It is possible to define which serial port is used for separate Modbus serial instances: COM1 or COM2. The serial communication instance is not active if this parameter is set to "Not in use."



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



Baud rate is defined on the serial driver side and are therefore located via the LHMI paths **Configuration/Communication/COM1** and **Configuration/Communication/COM2**.

### 3.1.1.2

#### Modbus serial diagnostic counters

Modbus Serial diagnostic counters can be viewed via the LHMI path **Monitoring/Communication/Modbus/Serial**.

Counters related to the possible Modbus serial instances 1...2 have the suffixes 1...2 (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
Received frames N	Total amount of received Modbus frames. For example, the Modbus frames that are addressed to this instance.
Transmitted frames N	Total amount of transmitted Modbus responses.
Cksm Err N	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.
Transmitted Exc A N	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 N	Total amount of exception responses 3. These exceptions usually reveal the IED application level rejections. That is, the IED 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.
Status N	Shows the value "True" if the serial instance is in use. This indicates that the Modbus client is connected. The status is checked every second, and when the Modbus client disconnects, the Modbus server waits for 15 seconds to update the status to "False". Resetting of all diagnostic counters in the instance N is done by writing value "True" into this same Status N object.

### 3.1.1.3

#### 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. 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 character	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 character	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

### 3.1.2

#### 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.

#### 3.1.2.1

#### TCP/IP interface configuration

The Modbus TCP/IP protocol uses the Ethernet interface. The general setup parameters of Ethernet, for example the protection relay's own IP address, are found via the LHMI path **Configuration/Communication/Ethernet**.

#### Client connections

The Modbus TCP/IP server accepts as many simultaneous client connections as defined with the *Max TCP/IP clients* parameter:

- The setting range for the parameter is 0...5.
- If the parameter value is zero, the Modbus TCP/IP server connection is not in use.
- The parameter works in conjunction with the parameters of a registered Modbus TCP/IP client.

When client X reconnects, the old connection of that client is disconnected and the new connection is accepted to avoid zombie clients. When the maximum number of clients are connected, a new connection request is handled as follows:

- If there are unregistered clients connected, the one with the longest silent period is disconnected and a new connection is accepted.
- If there are only registered clients connected, the new connection request is rejected.

It is possible to predefine the client or clients which are always granted Modbus TCP/IP connections by registering the clients' IP addresses. For example, if four concurrent connections are allowed and three of them are registered, they are seen as Client connection 1...Client connection 3. These three registered connections are then dedicated to certain clients only and the fourth connection is available to other clients.

### Client IP addresses

There are five Modbus setting parameters for Modbus client IP addresses. The parameter value "0.0.0.0" indicates that the client IP address is not defined.

If there are, for example, four available TCP/IP connections defined and one of the connections is to be dedicated for a certain client X, enter the client X's IP address to the *Client IP1* parameter. The IP addresses of the Modbus clients 2..4 can be set to "0.0.0.0". The setting of the Modbus client5 IP address has no meaning in this example as the connection is not in use. In this example, the TCP/IP session 1 is dedicated to the client X which means that this registered client X is always able to connect to the protection relay. Unregistered clients can connect to sessions 2...4. However, an unregistered client connection request can be rejected if sessions 2...4 are already occupied. The write authority can also be assigned differently for registered TCP/IP clients.

### Client's write authority

The registering of a Modbus client affects the client's write authority and the reading of latched Modbus data.

The *TCP write authority* parameter can be set to three different states:

- 0 = No write authority for any Modbus TCP/IP client
- 1 = Write authority only for registered Modbus TCP/IP clients
- 2 = Write authority for all Modbus TCP/IP clients

The possible blocking of write operation does not include the selection write operation that has to be done to read out Modbus event and fault record structures.

### 3.1.2.2 Modbus TCP/IP diagnostic counters

Modbus TCP/IP counters can be viewed via the LHMI path **Monitoring/Communication/Modbus/Ethernet**.

Counters related to the possible Modbus TCP/IP instances 1...5 have the suffixes 1...5 (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
Received frames N	Total amount of received Modbus frames.
Transmitted frames N	Total amount of transmitted Modbus responses.
Transmitted Exc A N	Total amount of exception responses 1 and 2. These exception responses usually reveal configuration errors on the Modbus client's side.
Transmitted Exc B N	Total amount of exception responses 3. These exceptions reveal the IED application level rejections.
Status N	Shows the value "True" if the TCP/IP instance is in use. This indicates that the Modbus client has connected to the TCP socket. The status is checked every second, and when the Modbus client disconnects, the Modbus server waits for 15 seconds to update the status to "False". Resetting of all diagnostic counters in the instance N is done by writing the value "False" into this same Status N object.

The counters and status of an instance N are reset when the client makes a TCP socket disconnection or if the TCP socket connection keep alive times out.

The common (instance independent) Modbus TCP/IP diagnostic counters can be reset separately by writing the value 0 into them.

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

Counter	Description
CnReject No sockets	The amount of connection requests that are rejected due to unavailable TCP sockets.
CnReject Not reg	The amount of connection requests that are rejected since the client is not registered.

## 3.2 Supported function codes

### 3.2.1 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.
06	Preset single register	Sets the value of a holding register.
08	Diagnostics	Checks the communication system between the master and the slave.
16	Preset multiple registers	Sets the value of multiple holding registers.
23	Read/write holding registers	Exchanges holding registers in one query.

### 3.2.2 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 slaves peripheral port is initialized and restarted and the communication event counters are cleared. Before this, a normal response will be 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 slave 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 slave since its last restart, clear counters operation or power up.

Table continues on next page

Function code	Name	Description
12	Return bus communication error count	The response returns the number of CRC errors encountered by the slave since its last restart, clear counters operation or power up.
13	Return bus exception error count	The response returns the number of Modbus exception responses sent by the slave since its last restart, clear counters operation or power up.
14	Return slave message count	The response returns the number of messages addressed to the slave or broadcast which the slave has processed since its last restart, clear counters operation or power up.
15	Return slave no response count	The response returns the number of messages addressed to the slave for which a response (neither a normal response nor an exception response) has not been sent since its last restart, clear counters operation or power up.
16	Return slave NACK response count	The number of messages addressed to the slave for which a negative acknowledgement response has been sent is returned in the response.
18	Return bus character overrun count	The response returns the number of messages addressed to the slave 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.

### 3.2.3

### Exception codes

*Table 8: Supported exception codes*

Function code	Name	Description
01	Illegal function	The slave does not support the requested function.
02	Illegal data address	The slave 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.

## 3.3 Application data

### 3.3.1 Modbus data objects

The Modbus protocol in the protection relays of this product series is built on top of the internal IEC 61850 data model. Thus, the Modbus application data objects, proprietary events and MCD bits are derived from IEC 61850 data objects and data set reporting. The protection relays have a predefined IEC 61850 data set configuration. In other words, it is predefined which internal data object changes the protection relays detect.

The available Modbus indications in the protection relays of this product series are generally selected from the IEC 61850 indications residing in data sets. Objects that do not reside in any data set are updated to the Modbus database slower. This concerns, for example, some measurand register values. Fast changes in these object values may not be detected or propagated to the Modbus database. However, the latest value of these objects is always found in the Modbus database.



For a list of the available data objects, see the point list manual.

### 3.3.2 Modbus data implementation



The numeric register locations used in this section are for example purposes only. The real Modbus register locations are in the protection relay's memory map.

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 IEC61850 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 exactly how this protocol application data should be used by an protection relay application. The usage depends on the protection relay implementation.

#### 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.

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 masters 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.
- Both 16- and 32-bit register sizes are used for measurands.
- The proprietary Modbus event buffer can be read in many different ways. A master 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 master scans.
- The Modbus fault record gives a summary of the captured analog quantities and protection stages picking up 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.

## 3.3.3 Data mapping principles

Modbus data is organized sequentially. This is the most efficient organization method since the master normally scans the Modbus data in blocks.

### 3.3.3.1 Data in monitoring direction

All data in the monitoring direction is available through the 4X memory area. 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.

---

### 3.3.3.2 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. One-bit data is also available in 4X register memory area as bit-packed 16-bit registers. The bit locations follow a pattern similar to the 0X and 1X locations.

### 3.3.3.3 Data in control direction

Protection relay controls, set points and acknowledgements are mapped to Modbus 0X data (coils). Coils can only be operated one by one.

Currently the ANSI implementation of controls via the Modbus protocol are restricted to the 4X registers map.

Some control bits are packed bits in the 4X control register structures. The 4X control structure contains a password which has to be given before starting control operations.

## 3.3.4 Digital input data

As the indication signals related to protection applications often change rapidly, the Modbus master 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.

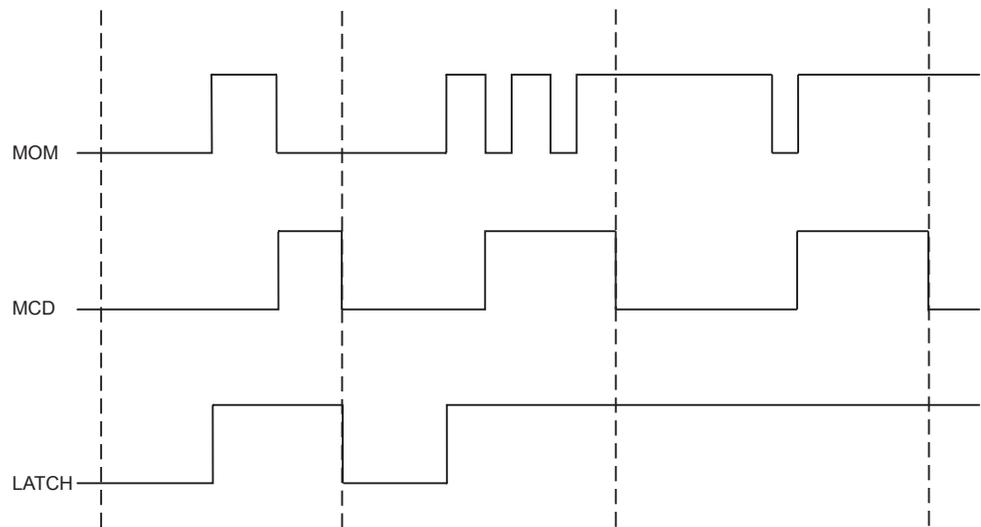


Figure 2: Change detection bit

If the momentary value of an indication bit has changed two times or more since the master 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 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.

### 3.3.4.1 Multiple digital inputs mapping

Digital inputs related to two-bit DPC or DPS objects, for instance circuit breaker 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 bits reveals if the object has changed its position several times since the Modbus master last scanned it. In addition, the open and close bits are also coded using MOM values only, among with a fault bit. The fault bit is set to "1" when the object is in intermediate (00) or faulty (11) position.

**Table 9:** *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.

### 3.3.5 Measurand registers

The Modbus measurands are located in the Modbus register area. The measurands are readable from 4X areas.

The formula for calculating the Modbus register value is:

$$\text{Modbus value} = (\text{IEC61850Value} \times \text{scaleFactor}) + \text{Offset}$$

(Equation 1)

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.

#### 3.3.5.1 Primary and per-unit values

Measurands originating from CT 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. The internal representation of the per-unit values is always 1.0 at nominal current. A typical range for a per-unit value is 0.00...40.00, that is 0 to 40 times nominal.



If the primary value representation is selected but no CT 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 ratio being used.

### 3.3.5.2 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.

### 3.3.5.3 Time of update

Some Modbus values may have a time structure attached to their values in the Modbus memory map. This is often the case with demand measurement values. The time structure shows the time when the value was last updated.

**Table 10:** *Time structure data*

Address	Register	Values	Comment
N	TimeStamp (Year,Month)		High byte:year, low byte:month
N+1	TimeStamp (Day,Hour)		High byte:day, low byte:hour
N+2	TimeStamp (Min,Sec)		High byte:min, low byte:seconds
N+3	TimeStamp (Milliseconds)		Word: milliseconds
N+4	Time quality	See the table about time quality register	

**Table 11:** *Time quality register*

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

### 3.3.6 Control operations

Refer to the Modbus control objects' memory map for the available control objects. Control operations are supported via the 4X registers.

The control objects in this IED 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 will happen 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 IED are pulse outputs.

#### Control operation modes on IEC 61850 level

This IED supports two control models: direct-operate and select-before-operate. The IEC 61850 single point control objects in this IED 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.

#### Control operations on Modbus level

The double point select-before-operate mode is usually used for the circuit breaker operations. 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 a "1" value opens the circuit breaker)
- Direct close (writing a "1" value closes the circuit breaker)



Direct operate of a double point object is only possible if the controllable object control model is set to "Direct-with-normal-security".

For PLC compatibility, the direct control points accept both values "1" (=normal direction) and "0" (=inverted direction). For example, if the Direct close point is controlled with value "0", it will "Open" the breaker and vice versa.

### 3.3.6.1

#### Control functions

Only controls made through 4X register structures are supported in this protection relay. The circuit breaker can be operated via Modbus by using a function codes 06,16 or 23.

Multiple control bits 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.

### 3.3.6.2

#### Control operations through 4X register structures

The control outputs can be operated through the control structures in the 4X register area. This means that the control output is also located as a bit within the value and bit mask registers of the 4X control structure. Although usually less, there may be up to 16 control structures defined in the IED.

The control structure operations can be controlled with passwords. Each password is shared by two consecutive control structures, that is, the first two control structures share

the first password, the next two control structures share the second password, and so on. As a default, no passwords exist for the structures. Any four character ASCII string can be used as a password. The password string “\*\*\*\*” with four asterisks, that is ASCII code 42, indicates that a password is not used.

**Table 12:** *Single control structure*

Location	Meaning
4x Reg N	Execute register
4x Reg N+1	Password register 1 high, two ASCII characters
4x Reg N+2	Password register 2 low, two ASCII characters
4x Reg N+3	Control register
4x Reg N+4	Confirmation register

With the control operations the client must assemble the control structure register values and write them into the IED.

### **Execute register**

The control step is executed when value "1" is written into this register.

### **Password register 1**

If a password is defined, the first two ASCII characters of the four character password are written into this register: the first character into the higher byte and the second character into the lower byte of the register. If no password is defined for the control structure, this register is not checked by the IED.

### **Password register 2**

If a password is defined, the last two ASCII characters of the four character password are written into this register: the third character into the higher byte and the fourth character into the lower byte of the register. If no password is defined for the control structure, this register is not checked by the IED.

### **Control register**

Set the register bit corresponding to the output to the proper write value. For pulse type outputs the value is always "1".

### **Confirmation register**

Set the register bit corresponding to the object to be operated to "1". All other bits must be set to zero.

---

### Control structure register assembling order

The Modbus client can assemble all the control structure registers and write them in one multiple registers write function 16 request.

The Modbus client can also write the registers in several separate transactions or even one by one using registers write function 06. The execute register has to be written last and no more than 15 seconds may occur between the separate register writes. The control structure operation will time out after 15 seconds after the last register write.



If several clients are allowed to perform control operations simultaneously, this method should not be used by more than one of the multiple clients in question.

### Exception codes

Only a few exception code alternatives exist for control structures:

- 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.

The primary internal rejection reasons for control structure write operations may be for example:

- The Modbus control structure write has timed out (15 sec).
- The client has no write authority.
- The IED is in the local or OFF state.
- The control operation is blocked, that means already reserved, by another client.

If a positive acknowledgement is returned, the control command has been initiated inside the IED.

#### 3.3.6.3

### Additional control operation features

#### Normal or enhanced security operations

Control objects on protection relay system level (IEC 61850 level) always follow a control model. Control model alternatives are referred to as normal-security or enhanced-security. Some control objects has a fixed control model. Other objects' control models are configurable. On Modbus level this means:

---

### Normal security object

- Positive confirmation means that the control has been activated and the application behind the control point has performed successfully.
- Exception 03 response from a normal-security object means that either the control is not activated, or the control is activated, but the application behind the control point does not perform successfully.

### Enhanced security object

- Positive confirmation means that control has been activated. The application behind the control point has started, but has not yet finished. SSR6 state is set into 'In progress'.
- Exception 03 response means that the control is not activated. SSR6 reason code is updated.

After a positive confirmation, SSR6 state is set to 'Ready' when the application control eventually is terminated. SSR6 reason code is updated with either a positive or a negative reason code.

### Impact on master's logic

Only one control sequence can be performed at a time by the protection relay. A new Modbus control command cannot be accepted by the protection relay after an enhanced security object control, until the SSR6 state is set to 'Ready'.

Enhanced security objects are in practice always Double Point objects. For example, in the case of a control made to a motor-controlled disconnecter, the control sequence lasts 10 seconds. Master can monitor the command progress.

- By polling the SSR6 register and examine the state bits. Control can be in state 'In progress' for 10 seconds.
- Double Point object .stSeld attribute is set to '1' while the control operation is in progress. This also lasts for 10 seconds.
- The control should result in some input data eventually changing position. This input data could be monitored to determine that the control operation is over. This should also take 10 seconds.

## 3.3.7

### System status registers

See Modbus ANSI point list manuals for specific register locations.

**Table 13:** *System status registers*

Register	Description	Address
SSR1	Device health	4xxxx + 1
SSR2	Device mode	4xxxx + 2
SSR3	Data available 1	4xxxx + 3
SSR4	Data available 2	4xxxx + 4
SSR5	Device alive counter	4xxxx + 5
SSR6	Last command result	4xxxx + 6

### 3.3.7.1

#### SSR1

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	Meaning
0	Device global warning
1	Device global error
2	Slot 0 (X130) warning or error
3	Slot 1 (X120) warning or error
4	Slot 2 (X110) warning or error
5	Slot 3 (X100) warning or error
6	Slot 4 (X000) warning or error
7...15	0 = not used

### 3.3.7.2

#### SSR2

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	Meaning
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

### 3.3.7.3

### SSR3

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 recorded. The bit is reset when the client writes the reset code 4 to the event record selection register.

Bit 9 is set to "1" when a fault record has been recorded. The bit is reset when the client writes the reset code 4 to the fault record selection register.

**Table 16:** 16 bit SSR3 register

Bit	Meaning
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

### 3.3.7.4

### SSR4

The bit values in SSR4 are Modbus client dependent.

Bits 0...15 in the SSR4 registers 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 each 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".



See the point list manuals for data categories specific to ANSI protection relays.

**Table 17:** 16 bit SSR4 register

Bit	Meaning	Data category
0	Data in category 1 changed	1 = Physical inputs
1	Data in category 2 changed	1 = Protection function pickup/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
Table continues on next page		

Bit	Meaning	Data category
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
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

**3.3.7.5****SSR5**

SSR5 is a device alive counter. SSR5 simply counts upwards from 0 to 65535 and then starts over. The meaning of this register is to assure that the device is actually operating.

**3.3.7.6****SSR6**

SSR6 is a last command register. 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 or if the command initiates a secured control operation. 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

10 = Secured 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	Meaning
0	OK
201	Device in local mode
202	Control operation reserved by another client
203	Select-timeout or Execute/Cancel without select
204	Control operation internally blocked
205	Control operation timed out
250	Other reason

### 3.3.8 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.

#### 3.3.8.1 User definable registers

The Modbus register areas 3X and 4X from 1 to 127 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.

#### 3.3.8.2 User definable bits

The Modbus bit address areas 0X and 1X from bit 16 to 2047 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.



The 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.

### 3.3.8.3 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 themselves cannot act as source data for other UD data.
- Modbus source data can only be attached to one UD location.

### 3.3.8.4 Data properties

The UD data inherits all properties from the source data. This applies to:

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

### 3.3.8.5 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.

### 3.3.8.6 UDR data configuration

UDR definitions can be created using the Communication Management tool in PCM600.

### 3.3.8.7 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.
- Swapping high/low words within the 32 bit register types.
- Changing the saturation points of the UDR values, that is, the bit-length of the source value can be redefined. Also the justification of the redefined bits within the Modbus register can be defined.
- Swapping high/low bytes within the 16 bit register types.

**Table 20: UDR scaling alternatives**

Scaling alternative	Setting	Description
No scaling		No change is made to the sourceValue
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

- $X = (\text{MaxOut} - \text{MinOut}) / (\text{MaxIn} - \text{MinIn})$
- $\text{UDR\_ResultValue} = X \times \text{sourceValue} + (\text{MinOut} - X \times \text{MinIn})$

### Multiplicative scaling Operation

$\text{UDR\_resultValue} = \text{sourceValue} \times \text{multiplicand}$

### Divisor scaling Operation

$\text{UDR\_resultValue} = \text{sourceValue} / \text{Divisor}$

## 3.3.9

### 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.

Modbus events can also be generated from selected Modbus registers. This concerns registers containing status information. In this case events would be generated each time the register's integer value changes.

The size of the protection relay's internal Modbus event record buffer is 500 events. The 500 latest events are at any time readable from the protection relay. When the Modbus event record buffer becomes full, the protection relay overwrites the oldest event records in the buffer.

### Multiple clients support

Several Modbus clients can independently of one another read out Modbus event records from the protection relay. The Modbus event buffer keeps track of where in the event 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. Up to 25 different IP addresses, belonging to both registered and unregistered Modbus clients, can be memorized by the protection relay.

#### 3.3.9.1

### Single event record structure

See Modbus ANSI point list manuals for specific event record structure mappings.

**Table 21:** *Event record structure*

Address	Register	Values	Comment
4xxx + 1	Event selection	1...4 and -1...-499	Write register
4xxx + 2	Sequence Number	0...65535	
4xxx + 3	Unread records left	0...499	
4xxx + 4	TimeStamp (Year,Month)		High byte:year, low byte:month
4xxx + 5	TimeStamp (Day, Hour)		High byte:day, low byte:hour
4xxx + 6	TimeStamp (Min, Sec)		High byte:min, low byte:second
4xxx + 7	TimeStamp (Milliseconds)		Word: milliseconds (0...999)
4xxx + 8	Event type		See separate description
4xxx + 9	Data Object ID 1	0 or UID high word	See separate description
4xxx + 10	Data Object ID 2	Modbus address or UID low word	
4xxx + 11	Data Value	Modbus data value	Value into which object has changed
4xxx + 12	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 on which 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*.

### 3.3.9.2 Single event record reading

As long as there are unread Modbus events available for the Modbus client in question, 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. The selection code defines the type of read operation that the client wants to perform. The selected event record is loaded by the protection relay into the following 11 registers. 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 the 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.

### 3.3.9.3

#### 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 exactly how many event records have been lost.

##### Unread records left

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

##### Time stamp registers

Time stamp is 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 22:** Information contained by 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
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 identification	00000000 = One bit indication      00000001 = ACD indication 00000010 = Two bit indication      00000011 = SEC indication+add data 00000100...                              00001001 = reserved 00001010 = Integer value              00001011... 11111111 = reserved

#### Event time stamp format bit 15

Event time stamp format.

#### 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. 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.

### 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 23:** *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	Pickup status	X		
INS/INC	Integer status			X

**Table 24:** *Interpretation of the one-bit data value*

Register 4xxx binary coded value	Meaning
xxxx.xxxx.xxxx.xxx0	Object in OFF position
xxxx.xxxx.xxxx.xxx1	Object in ON position

**Table 25:** *Interpretation of the two-bit data value*

Register 4xxxx 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 always refer to the Modbus address or UID of the CLOSE momentary value bit.

**Table 26:** *Interpretation of the integer status data value*

Register address <sup>1)</sup>	Meaning
4xxxx	Higher 16 bit part of the 32 bit integer value
4xxxx + 1	Lower 16 bit part of the 32 bit integer value

1) See Modbus ANSI point list manual for specific event record structure mappings.

**Table 27:** *Interpretation of the ACD data*

Register address <sup>1)</sup>	Meaning
4xxxx	xxxx.xxxx.xxxx.xxx0 Object in OFF position
	xxxx.xxxx.xxxx.xxx1 Object in ON position
4xxxx + 1	xxxx.xxxx.xxxx.xx00 Pickup in unknown direction
	xxxx.xxxx.xxxx.xx01 Pickup in forward direction
	xxxx.xxxx.xxxx.xx10 Pickup in backwards direction
	xxxx.xxxx.xxxx.xx11 Pickup in both directions

1) See Modbus ANSI point list manual for specific event record structure mappings.

**Table 28:** *Interpretation of the SEC data*

Register address <sup>1)</sup>	Meaning
4xxxx	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
4xxxx + 1	Security violations counter, 16 bits

1) See Modbus ANSI point list manual for specific event record structure mappings.

---

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.

#### 3.3.9.4 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 shall be written to the Num of records register in front of the selection register. This number can be written once or it can be rewritten for each select/read 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 exactly similar to the single-record read case. The next records to be returned always continues 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

**Table 29:** *Extended event record structure with the maximum of 10 event records*

Address <sup>1)</sup>	Register	Values	Description
4xxx	Num of records	1...10	Write: Number of Event structures
4xxx + 1	Selection		Write: Selection code
4xxx + 2	Sequence Number 1		Event record 1
4xxx + 3	Unread records left 1		
4xxx + 4	TimeStamp 1		
4xxx + 5	TimeStamp 1		
4xxx + 6	TimeStamp 1		
4xxx + 7	TimeStamp 1		
4xxx + 8	Event Type 1		
4xxx + 9	Data Object Id 1_1		
4xxx + 10	Data Object Id 2_1		
4xxx + 11	Data Value 1		
4xxx + 12	Data Value 1		
4xxx + 13	Sequence Number 2		Event record 2
:	:	:	
4xxx + 23	Data Value 2		Event record 3
4xxx + 24	Sequence Number 3		
:	:	:	
4xxx + 34	Data Value 3		Event record 4
4xxx + 35	Sequence Number 4		
:	:	:	Event record 5
4xxx + 45	Data Value 4		
4xxx + 46	Sequence Number 5		
:	:	:	Event record 6
4xxx + 56	Data Value 5		
4xxx + 57	Sequence Number 6		Event record 7
:	:	:	
4xxx + 67	Data Value 6		
4xxx + 68	Sequence Number 7		Event record 7
:	:	:	
4xxx + 78	Data Value 7		

Table continues on next page

Address <sup>1)</sup>	Register	Values	Description
4xxxx + 79	Sequence Number 8		Event record 8
:	:	:	
4xxxx + 89	Data Value 8		
4xxxx + 90	Sequence Number 9		Event record 9
:	:	:	
4xxxx + 100	Data Value 9		
4xxxx + 101	Sequence Number 10		Event record 10
4xxxx + 102	Unread records left 10		
4xxxx + 103	TimeStamp 10		
4xxxx + 104	TimeStamp 10		
4xxxx + 105	TimeStamp 10		
4xxxx + 106	TimeStamp 10		
4xxxx + 107	Event Type 10		
4xxxx + 108	Data Object Id 1_10		
4xxxx + 109	Data Object Id 2_10		
4xxxx + 110	Data Value 10		
4xxxx + 111	Data Value 10		

1) See Modbus ANSI point list manual for specific event record structure mapping.

### 3.3.10

## 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.

### 3.3.10.1 Fault record structure

The protection relay's fault record structure 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 30:** Header part of the record structure

Address <sup>1)</sup>	Register	Values	Comment
4xxxx	Fault record selection	1...4 and -1...-99	Write register
4xxxx + 1	Sequence Number	0...65535	
4xxxx + 2	Unread records left	0...99	
4xxxx + 3	TimeStamp (Year,Month)		High byte:year, low byte:month
4xxxx + 4	TimeStamp (Day, Hour)		High byte:day, low byte:hour
4xxxx + 5	TimeStamp (Min, Sec)		High byte:min, low byte:second
4xxxx + 6	TimeStamp (Milliseconds)		Word: milliseconds (0...999)
4xxxx + 7	Time quality		
4xxxx + 8	From this location onwards starts the Fault record application data.		

1) See Modbus ANSI point list manual for specific event record structure mapping.

### Fault record application data part

The data in the application section are protection relay type dependent. The description of the data is found in the Modbus fault record section of the Modbus memory map.

### 3.3.10.2 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. 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 (4xxxx-NNNN). See Modbus ANSI point list manual for specific fault record structure mapping. Second, the client reads out these registers in one multiple register read operation.



The fault records can be read by using two commands, the function 6 for the write operation and the function 3 for the read operation, or by using the 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 vary 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 will give 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 has been 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.

### 3.3.10.3

### 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 31:** Information contained by the 16 bit (bits 15..0) register

Bit	Meaning	Values
15	Event record 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 LHMI or the parameter setting tool.

#### **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.

### **3.3.11**

#### **Parameter setting group selection**

The active parameter setting group can be changed by writing the new setting group number to 4X register. See Modbus ANSI point list manual for specific mapping. 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.

### **3.3.12**

#### **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 and the internal UTC time.

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**.

### 3.3.12.1 Real-time clock structure

*Table 32: Modbus real-time clock structure*

Modbus address <sup>1)</sup>		Register contents	Values
Local Time	UTC Time		
4xxx	4xxx + 19	Control register	0...2
4xxx + 1	4xxx + 10	Year	2000...9999
4xxx + 2	4xxx + 11	Month	1...12
4xxx + 3	4xxx + 12	Day	1...31
4xxx + 4	4xxx + 13	Hour	0...23
4xxx + 5	4xxx + 14	Minutes	0...59
4xxx + 6	4xxx + 15	Seconds	0...59
4xxx + 7	4xxx + 16	Milliseconds	0...999

1) See Modbus ANSI point list manual for specific event record structure mapping.

### 3.3.12.2 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

The real-time clock structure registers should be written in one multiple registers preset request (function 16) by a Modbus TCP/IP client or by a serial interface master. 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 control register 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 control register. 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 control register. When the value "2" is written, the timesync registers are latched onto the protection relay's internal clock and the reservation of the control register 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 control register. 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.

### 3.3.13

#### Device information

The device information of the protection relay can be read from the Modbus registers.



If the information data are in practice shorter, the trailing registers in the response are filled with the value "0".

The Modbus device information is based on the internal IEC 61850 device information model of the protection relay. All internal descriptions are coded as ASCII strings. The Modbus device information ASCII string includes the information from the protection relay.

- Protection relay model (max. 12 characters)
- Protection relay type (max. 6 characters)
- Protection relay's serial number (max. 12 characters)
- Protection relay's location information (max. 34 characters)
- CPU card SW and HW revision numbers
- HMI card SW and HW revision numbers
- Slot 0 (X130) card SW and HW revision numbers
- Slot 1 (X120) card SW and HW revision numbers
- Slot 2 (X110) card SW and HW revision numbers
- Slot 3 (X100) card SW and HW revision numbers
- Slot 4 (X000) card SW and HW revision numbers



The protection relay does not need to contain cards in all slots nor does a specific card need to include a CPU. The SW revision information is simply omitted from the information string.

### 3.3.13.1 ASCII character coding

*Table 33: The 8 bit ASCII character coding in the Modbus registers*

Modbus register	ASCII character
Register 1 High byte	= ASCII character 1
Register 1 Low byte	= ASCII character 2
Register 2 High byte	= ASCII character 3
:	:

### 3.3.13.2 ASCII string syntax

#### Syntax

C (model; type; serialNo; location; swRev; hwRev) H (swRev; hwRev)  
 ) 0 (swRev; hwRev) 1 (swRev; hwRev) 2 (swRev; hwRev) 3 (swRev; hwRev)  
 ) 4 (swRev; HwRev)

- Parenthesis and semicolon ASCII characters are used as delimiters inside the string.
- Section C ( . . . ) contains protection relay information and CPU version information.
- Section H ( . . . ) contains version information of the LHMI card.
- Sections 0 ( . . . ) to 4 ( . . . ) contain version information of the additional HW cards (slots 0...4).
- If an additional card does not include any version information, it is signaled with a "-" (minus) character in the swRev field. If both swRev and hwRev are signalled with "-" signs, the card in question does not exist in the protection relay.

The data within the C section is restricted to certain maximum lengths. For example, the user-definable protection relay location is here restricted to a maximum of 34 characters. If the protection relay location information on system level contains more characters, only the 34 first characters are displayed.

### 3.3.14 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 34:** *Reset time structure*

Address <sup>1)</sup>	Register	Values	Comment
4xxxx	TimeStamp (Year,Month)		High byte:year, low byte:month
4xxxx + 1	TimeStamp (Day,Hour)		High byte:day, low byte:hour
4xxxx + 2	TimeStamp (Min,Sec)		High byte:min, low byte:seconds
4xxxx + 3	TimeStamp (Milliseconds)		Word: milliseconds
4xxxx + 4	Time Quality	See Time quality table	
4xxxx + 5	Cause of reset	1 = Power reset	
		2 = Watchdog reset	
		3 = Warm reset	

1) See Modbus ANSI point list manual for specific event record structure mapping.

**Table 35:** *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

### 3.3.15

## Accessing of non-protocol-mapped data

The protection relay application includes a number of general-purpose I/O data. By default, these data are mapped to this protocol. See the point list manual for the exact mappings.

The general-purpose objects can be connected to any internal object in the protection relay configuration application using the Application Configuration or Signal Matrix tool. This gives additional opportunities for the protocols.

---

### Example 1

Due to security reasons, protocols do not contain mappings for the direct control of physical outputs. This way, the client cannot accidentally write a change to a physical output.

It is possible to connect general-purpose outputs to physical outputs using the Application Configuration tool. The general-purpose output can also be controlled from the protocol.

### Example 2

The legacy protocol default mappings are a selection of the most important signals produced by the IEC 61850-based protection relay applications. The manufacturer's selection of important signals may not always serve every customer.

Any non-protocol-mapped internal signal can be freely connected to a general-purpose input object via the Application Configuration tool. This object can then be accessed by the legacy protocol as regular protocol application data.

### Example 3

The basic IEC 61850 application model of the protection relay produces a great amount of information. In some cases, this is more than what is feasible to transport through a legacy protocol. Via the PCM600/Communication Management tools, unnecessary data objects can be excluded from the legacy protocol.

However, in some cases a better solution is to OR together several internal signals into one general signal. This OR output can be connected to a general-purpose input and accessed by the legacy protocol as regular protocol application data.

## Section 4 Modbus parameters and diagnostics

### 4.1 Parameter list

The Modbus parameters can be accessed with PCM600 or via the LHMI path  
**Configuration/Communication/Modbus.**

**Table 36:** *Modbus settings*

Parameter	Values (Range)	Unit	Step	Default	Description
Serial port 1	0=Not in use 1=COM 1 2=COM 2			0=Not in use	COM port for Serial interface 1
Parity 1	0=none 1=odd 2=even			2=even	Parity for Serial interface 1
Address 1	1...255			1	Modbus unit address on Serial interface 1
Link mode 1	1=RTU 2=ASCII			1=RTU	Modbus link mode on Serial interface 1
Start delay 1	0...20	char		4	Start frame delay in chars on Serial interface 1
End delay 1	0...20	char		3	End frame delay in chars on Serial interface 1
Serial port 2	0=Not in use 1=COM 1 2=COM 2			0=Not in use	COM port for Serial interface 2
Parity 2	0=none 1=odd 2=even			2=even	Parity for Serial interface 2
Address 2	1...255			2	Modbus unit address on Serial interface 2
Link mode 2	1=RTU 2=ASCII			1=RTU	Modbus link mode on Serial interface 2
Start delay 2	0...20			4	Start frame delay in chars on Serial interface 2
End delay 2	0...20			3	End frame delay in chars on Serial interface 2
MaxTCPclients	0...5			5	Maximum number of Modbus TCP/IP clients
TCPWriteAuthority	0=No clients 1=Reg. clients 2=All clients			2=All clients	Write authority setting for Modbus TCP/IP clients
EventID	0=Address 1=UID			0=Address	Event ID selection
TimeFormat	0=UTC 1=Local			1=Local	Time format for Modbus time stamps

Table continues on next page

Parameter	Values (Range)	Unit	Step	Default	Description
ClientIP1				000.000.000.000	Modbus Registered Client 1
ClientIP2				000.000.000.000	Modbus Registered Client 2
ClientIP3				000.000.000.000	Modbus Registered Client 3
ClientIP4				000.000.000.000	Modbus Registered Client 4
ClientIP5				000.000.000.000	Modbus Registered Client 5
CtlStructPWd1				****	Password for Modbus control struct 1 and 2 <sup>1)</sup>
CtlStructPWd2				****	Password for Modbus control struct 3 and 4
CtlStructPWd3				****	Password for Modbus control struct 5 and 6
CtlStructPWd4				****	Password for Modbus control struct 7 and 8
CtlStructPWd5				****	Password for Modbus control struct 9 and 10
CtlStructPWd6				****	Password for Modbus control struct 11 and 12
CtlStructPWd7				****	Password for Modbus control struct 13 and 14
CtlStructPWd8				****	Password for Modbus control struct 15 and 16

1) The amount of available control structures may vary depending on the IED type.

## 4.2 Monitored data

The Modbus serial monitored data can be accessed with PST or via the LHMI path **Monitoring/Communication/Modbus/Serial**.

*Table 37: Modbus serial*

Parameter	Values (range)	Description
Received frames N <sup>1)</sup>	0...2147483648	Received Modbus frames by instance N
Transmitted frames N	0...2147483648	Transmitted Modbus frames by instance N
Cksm Err N	0...2147483648	Checksum errors detected by instance N
Transmitted Exec A N	0...2147483648	Transmitted exception responses 1 and 2 by instance N
Transmitted Exec B N	0...2147483648	Transmitted exception responses 3 by instance N
Status N	True, False	Status of communication, instance N
		Reset of diagnostic counters instance N

1) N = instance number 1, 2...

The Modbus Ethernet monitored data can be accessed with PST or via the LHMI path **Monitoring/Communication/Modbus/Ethernet**.

**Table 38:** *Modbus TCP/IP*

Parameter	Values (range)	Description
Received frames N <sup>1)</sup>	0...2147483648	Received Modbus frames by instance N
Transmitted frames N	0...2147483648	Transmitted Modbus frames by instance N
Transmitted Exec A N	0...2147483648	Transmitted exception responses 1 and 2 by instance N
Transmitted Exec B N	0...2147483648	Transmitted exception responses 3 by instance N.
Cn Rejects No socket	0...2147483648	Connection rejections due to no free sockets
Cn Rejects Not reg	0...2147483648	Connection rejections due to not registered client IP.
Status N	True, False	Status of communication, instance N
		Reset of diagnostic counters instance N

1) N = instance number 1, 2...



---

## Section 5      Glossary

<b>ACD</b>	Start/pickup status
<b>ACT</b>	1. Application Configuration tool in PCM600 2. Trip status in IEC 61850
<b>ANSI</b>	American National Standards Institute
<b>ASCII</b>	American Standard Code for Information Interchange
<b>Cnt</b>	Counter
<b>CPU</b>	Central processing unit
<b>CT</b>	Current transformer
<b>Data set</b>	The content basis for reporting and logging containing references to the data and data attribute values
<b>DPC</b>	Double-point control
<b>DPS</b>	Double-point status
<b>EMC</b>	Electromagnetic compatibility
<b>Ethernet</b>	A standard for connecting a family of frame-based computer networking technologies into a LAN
<b>HMI</b>	Human-machine interface
<b>HW</b>	Hardware
<b>IEC 61850</b>	International standard for substation communication and modeling
<b>IED</b>	Intelligent electronic device
<b>INS/INC</b>	Integer status
<b>IP address</b>	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.
<b>LED</b>	Light-emitting diode
<b>LHMI</b>	Local human-machine interface
<b>LSB</b>	Least significant bit
<b>MCD</b>	Momentary change detect

---

<b>Modbus</b>	A serial communication protocol developed by the Modicon company in 1979. Originally used for communication in PLCs and RTU devices.
<b>Modbus ASCII</b>	Link mode using 7-bit ASCII characters
<b>Modbus memory map</b>	Allocation of accessible protocol data
<b>Modbus RTU</b>	Link mode using 8-bit binary characters
<b>Modbus TCP/IP</b>	Modbus RTU protocol which uses TCP/IP and Ethernet to carry data between devices
<b>MOM</b>	Momentary position
<b>PCM600</b>	Protection and Control IED Manager
<b>PLC</b>	Programmable logic controller
<b>PST</b>	1. Parameter Setting tool in PCM600 2. Product Selection Tool
<b>RS-485</b>	Serial link according to EIA standard RS485
<b>RTC</b>	Real-time clock
<b>RTU</b>	Remote terminal unit
<b>SCADA</b>	Supervision, control and data acquisition
<b>SEC</b>	Security violation
<b>SPC</b>	Single-point status of a controllable object
<b>SPS</b>	Single-point status
<b>SSR1</b>	System status register for device health
<b>SSR2</b>	System status register for device mode
<b>SSR3</b>	System status register for data available 1
<b>SSR4</b>	System status register for data available 2
<b>SSR5</b>	System status register for device alive counter
<b>SSR6</b>	System status register for last command result
<b>SW</b>	Software
<b>TCP</b>	Transmission Control Protocol
<b>TCP/IP</b>	Transmission Control Protocol/Internet Protocol
<b>UD</b>	User-definable
<b>UDB</b>	User-definable bit
<b>UDR</b>	User-definable register

---

<b>UID</b>	Unique ID
<b>UL</b>	Underwriters Laboratories
<b>UTC</b>	Coordinated universal time
<b>WHMI</b>	Web human-machine interface







---

**ABB Distribution Solutions**  
**Distribution Automation**

P.O. Box 699  
FI-65101 VAASA, Finland  
Phone +358 10 22 11

**ABB Inc.**

655 Century Point  
Lake Mary, FL 32746, USA  
Phone +1-800-222 1946

**[www.abb.com/mediumvoltage](http://www.abb.com/mediumvoltage)**  
**[www.abb.com/relion](http://www.abb.com/relion)**  
**[www.abb.com/substationautomation](http://www.abb.com/substationautomation)**