RVT communication

How to use RS485 – USB – Ethernet RVT connections

Power and productivity
for a better world™

Table of contents

1 Introduction

1.1 Intended audience

This manual is intended for programmers, commissioning people, supervision people who need to start communication, access data, and to develop supervision software which will interact with the Power^{IT} Power Factor Controller RVT.

1.2 Before you start

This manual describes the RVT data table. These data are accessible through RS485, USB, or Ethernet, using Modbus RTU and TCP protocols or PQ-Link software.

All information available from the keyboard of the RVT will be available through the data table. Addresses, access levels and protocol information are of concerns.

To be able to access data of the Power^{IT} Power Factor Controller RVT consistently, a basic knowledge of it is needed. Functionality of the RVT, meaning of various measurements, logging of data are some particular aspects that should be familiar. Look in the RVT operating manual to know more about it.

1.3 How to use this manual

Chapter 2 gives details concerning the Modbus protocol.

Chapter 3 describes Modbus functions and how Modbus is implemented in the controller.

Chapter 4 describes USB / TCP/IP protocol and how it is implemented in the controller.

Chapter 5 contains the table reference and formats to access measurements / settings data.

Chapter 6 describes the Windows DLL to handle USB / TCP/IP requests in a user specific application.

1.4 Software protocols and physical interface

The RVT Power Factor Controller supports three communication protocols:
Modbus RTU and Modbus TCP (Chapter 2 and 0) and PQ-Link protocol (Chapter 6).

Three physical connections are available:
RS485 with the Modbus Adapter option (Chapter 2), USB (Chapter 4), Ethernet-RJ45 (Chapter 4).

The table below resumes the availability of communication protocols depending on the RVT type and the connection provided.

Communication protocol		**Available connection**		
		RS485 Modbus Adapter (option)	USB	Ethernet-RJ45
RVT Type	RVT 6	Modbus RTU	PQ-Link	Not available
	RVT 12	Modbus RTU	PQ-Link	Not available
	RVT 12-3P	Modbus RTU	PQ-Link	PQ-Link and Modbus TCP

2 Modbus protocol overview

2.1 Overview

MODBUS RTU

MODBUS RTU is a non-proprietary serial communications protocol that is widely used in the process control industry. The protocol was developed by Modicon for PLC communications and later released for public use.

This protocol is available in all major Human Machine Interface (HMI) software packages and terminals. Many of the major controller and PLC manufacturers also offer MODBUS protocol as a standard or optional protocol in their instrumentation.

The hardware over which MODBUS RTU communications are performed is not defined by the protocol. MODBUS RTU is supported on RS-232, RS-422, RS-485, Ethernet and other electrical standards. It should be noted that MODBUS RTU, MODBUS ASCII and MODBUS Plus are unique communication formats, and are not compatible with each other. This document will discuss MODBUS RTU only.

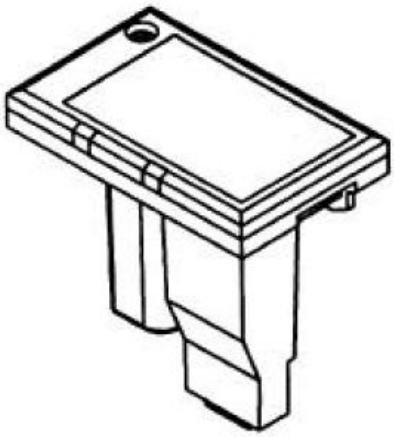
MODBUS TCP

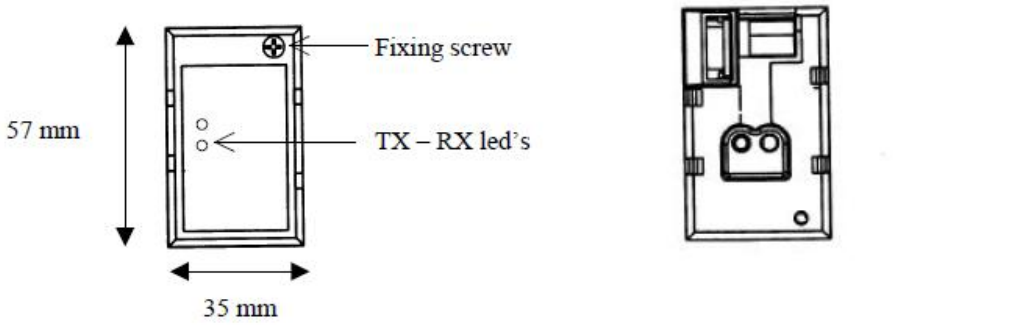
MODBUS TCP is a MODBUS RTU message transmitted with a TCP/IP wrapper and sent over an Ethernet network instead of serial lines. The Server does not have a SlaveID as in RTU since it uses an IP Address instead.

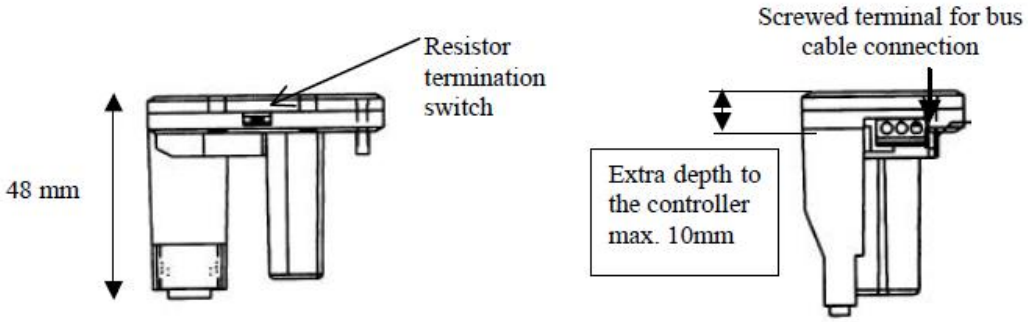
2.2 RS485 Modbus Adapter

The Modbus protocol communicates with the instrumentation by means of an industry standard serial interface. This interface may be RS-232, RS-422 or RS-485. Some systems may also support the protocol over other busses or networks, such as Ethernet. An RS-232 interface allows only two devices to be connected together. RS-422 supports 1 driver and up to 10 receivers on a single network. For bi-directional communications, special tri-state circuitry is provided. RS-485 supports up to 32 driver/receiver pairs. With special hardware, the RS-422 and RS-485 limits can be expanded to allow as many as 248 devices on a single network. Each device on a network must have a unique address, which may be soft configured. Address zero is reserved for broadcast messages from the host to all slaves.

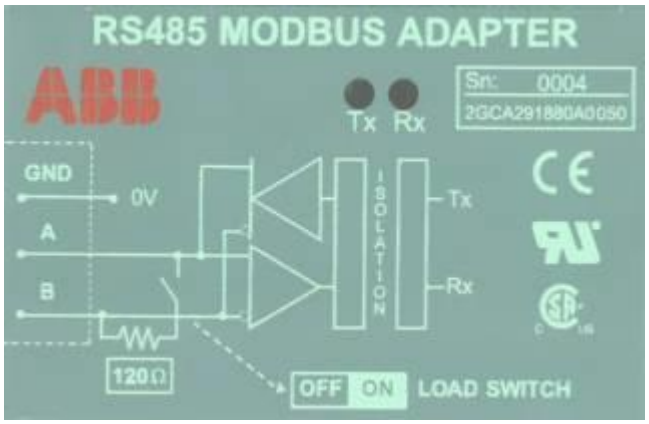
The RS485 Modbus Adapter is an option to the RVT. It enables the connection of the RVT controller to an RS485 Modbus network.

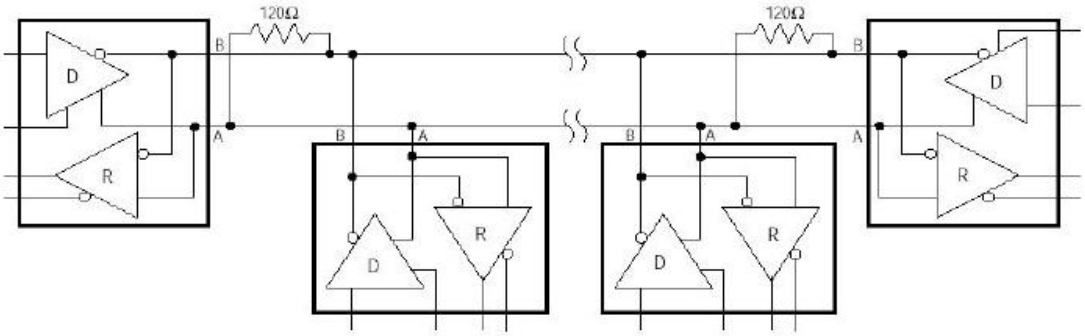






CAUTION: Be careful that the RS485 MODBUS ADAPTER is the one with a GREEN text colour (3.3V power supply). The one with a WHITE text colour is reserved for the old model (5V power supply).





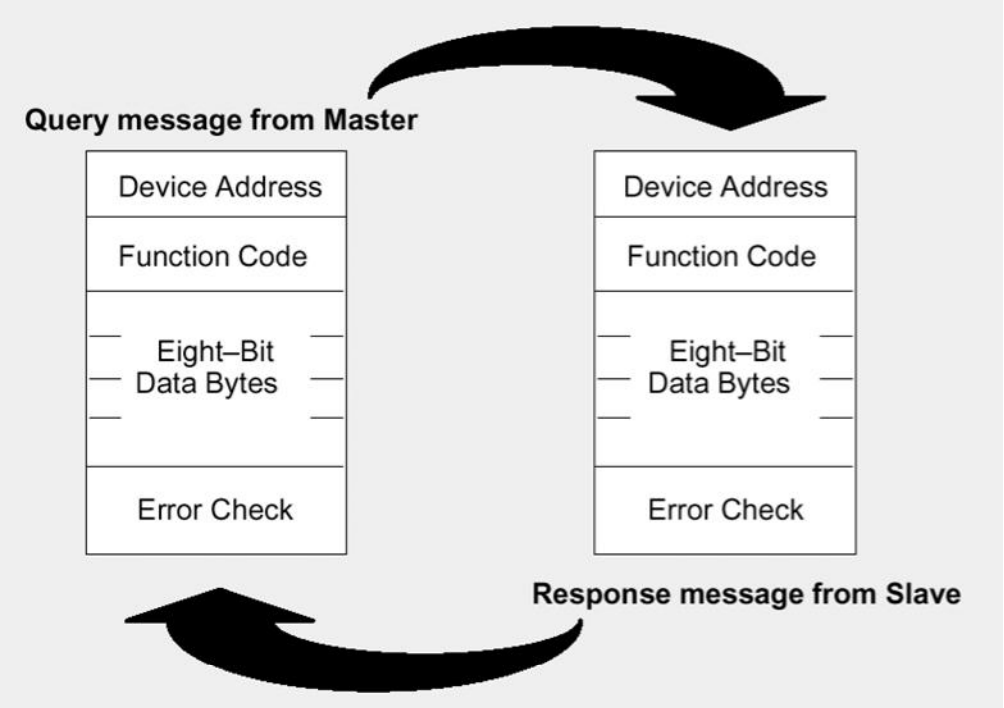
2.3 Transactions on Modbus Networks

Modbus protocol uses a master–slave technique, in which only one device (the master) can initiate transactions (called ‘queries’). The other devices (the slaves) respond by supplying the requested data to the master, or by taking the action requested in the query. Typical master devices include host processors and programming panels. Typical slaves include programmable controllers.

The master can address individual slaves, or can initiate a broadcast message to all slaves.

Slaves return a message (called a ‘response’) to queries that are addressed to them individually. Responses are not returned to broadcast queries from the master.

The Modbus protocol establishes the format for the master’s query by placing into it the device (or broadcast) address, a function code defining the requested action, any data to be sent, and an error–checking field. The slave’s response message is also constructed using Modbus protocol. It contains fields confirming the action taken, any data to be returned, and an error–checking field. If an error occurred in receipt of the message, or if the slave is unable to perform the requested action, the slave will construct an error message and send it as its response.



The Query:

The function code in the query tells the addressed slave device what kind of action to perform. The data bytes contain any additional information that the slave will need to perform the function.

The data field must contain the information telling the slave which register to start at and how many registers to read.

The error check field provides a method for the slave to validate the integrity of the message contents.

The Response:

If the slave makes a normal response, the function code in the response is an echo of the function code in the query. The data bytes contain the data collected by the slave, such as register values or status. If an error occurs, the function code is modified to indicate

that the response is an error response, and the data bytes contain a code that describes the error.

The error check field allows the master to confirm that the message contents are valid.

2.4 Serial Transmission Mode

The transmission mode defines the bit contents of message fields transmitted serially on the networks. It determines how information will be packed into the message fields and decoded.

Modbus defines two transmission modes: ASCII or RTU.

Only RTU mode will be used here. The mode and serial parameters must be the same for all devices on a Modbus network.

RTU Mode

The main advantage of this mode is that its greater character density allows better data throughput than ASCII for the same baud rate.

Each message must be transmitted in a continuous stream.

The format for each byte in RTU mode is:

Bits per Byte:

1 start bit

8 data bits, least significant bit sent first

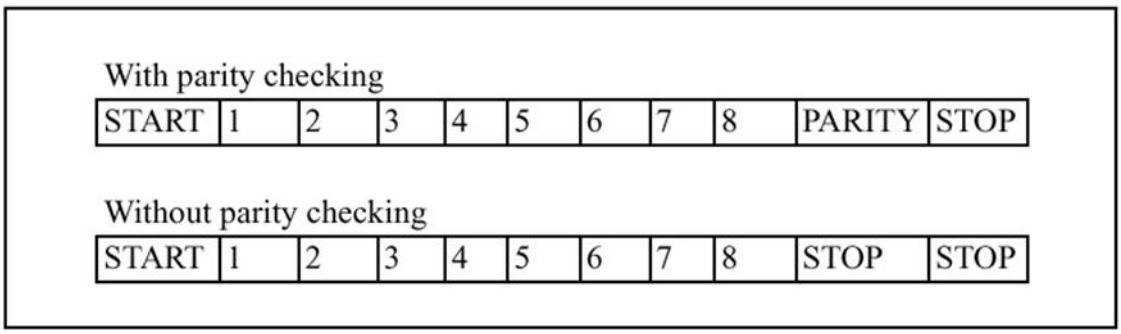
1 bit for even/odd parity; no bit for no parity

1 stop bit if parity is used; 2 bits if no parity

Error Check Field: Cyclical Redundancy Check (CRC)

The messages are transmitted in the network from left to right, i.e. the Least

Significant Bit (LSB) first and the Most Significant Bit (MSB) last.



Description of the bit sequence for the RTU mode

2.5 Modbus Message Framing

A Modbus message is placed by the transmitting device into a frame that has a known beginning and ending point. This allows receiving devices to begin at the start of the message, read the address portion and determine which device it is, and to know when the message is completed.

Partial messages can be detected and errors can be set as a result.

Modbus RTU Framing

In RTU mode, messages start with a silent interval of at least 3.5 character times.

This is most easily implemented as a multiple of character times at the baud rate that is being used on the network (shown as T1–T2–T3–T4 in the figure below).

Another factor to consider is that each device has its own response time. This response time can be anywhere from a few milliseconds to a few hundred milliseconds. The Host must be configured to allow adequate time for the slowest device to respond.

The first field then transmitted is the device address.

Networked devices monitor the network bus continuously, including during the 'silent' intervals. When the first field (the address field) is received, each device decodes it to find out if it is the addressed device.

Following the last transmitted character, a similar interval of at least 3.5 character times marks the end of the message. A new message can begin after this interval.

The entire message frame must be transmitted as a continuous stream. If a silent interval of more than 1.5 character times occurs before completion of the frame, the receiving device flushes the incomplete message and assumes that the next byte will be the address field of a new message.

Similarly, if a new message begins earlier than 3.5 character times following a previous message, the receiving device will consider it is a continuation of the previous message. This will set an error, as the value in the final CRC field will not be valid for the combined messages. A typical message frame is shown below.

START	ADDRESS	FUNCTION	DATA	CRC CHECK	END
T1–T2–T3–T4	8 BITS	8 BITS	*n* x 8 BITS	16 BITS	T1–T2–T3–T4

For a complete description of the Modbus protocol, please look at the Modicon

Modbus Protocol Reference Guide (PI–MBUS–300 Rev. J).

Modbus TCP Framing

Modbus TCP/IP (also Modbus-TCP) is simply the Modbus RTU protocol with a TCP interface that runs on Ethernet.

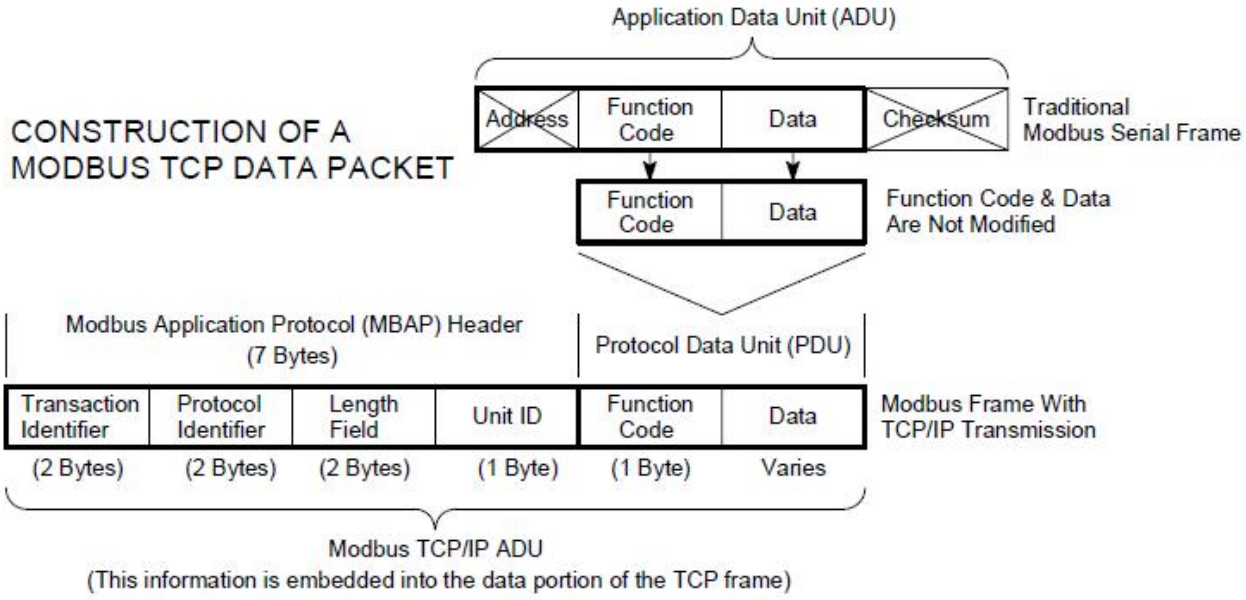
The Modbus messaging structure is the application protocol that defines the rules for organizing and interpreting the data independent of the data transmission medium.

TCP/IP refers to the Transmission Control Protocol and Internet Protocol, which provides the transmission medium for Modbus TCP/IP messaging.

Simply stated, TCP/IP allows blocks of binary data to be exchanged between computers. It is also a world-wide standard that serves as the foundation for the World Wide Web. The primary function of TCP is to ensure that all packets of data are received correctly, while IP makes sure that messages are correctly addressed and routed. Note that the TCP/IP combination is merely a transport protocol, and does not define what the data means or how the data is to be interpreted (this is the job of the application protocol, Modbus in this case).

So in summary, Modbus TCP/IP uses TCP/IP and Ethernet to carry the data of the Modbus message structure between compatible devices. That is, Modbus TCP/IP combines a physical network (Ethernet), with a networking standard (TCP/IP), and a standard method of representing data (Modbus as the application protocol). Essentially, the Modbus TCP/IP message is simply a Modbus communication encapsulated in an Ethernet TCP/IP wrapper.

In practice, Modbus TCP embeds a standard Modbus data frame into a TCP frame, without the Modbus checksum, as shown in the following diagram.



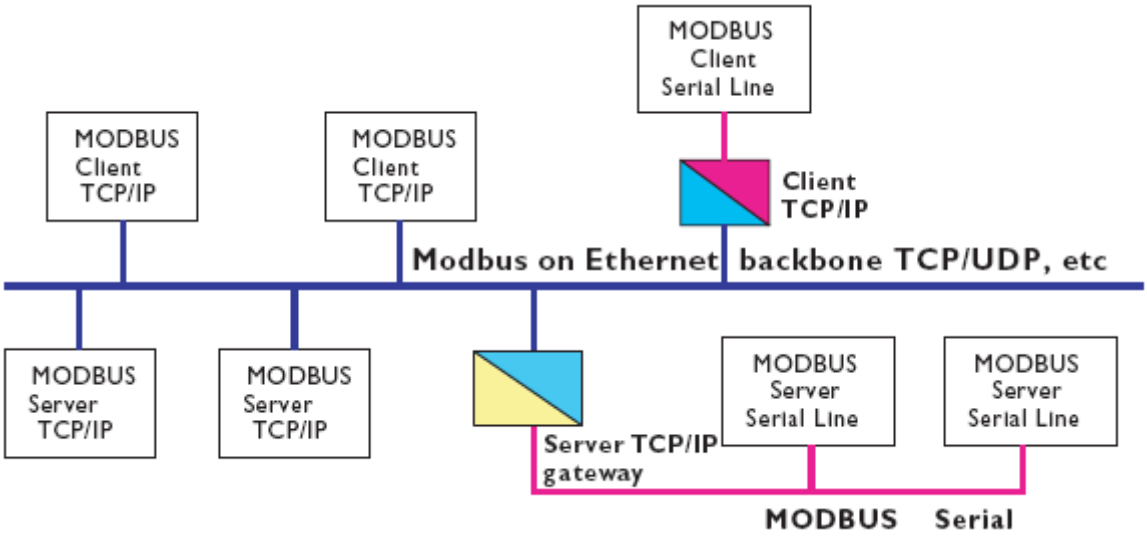
The complete Modbus TCP/IP Application Data Unit is embedded into the data field of a standard TCP frame and sent via TCP to well-known system port 502, which is specifically reserved for Modbus applications. Modbus TCP/IP clients and servers listen and receive Modbus data via port 502.

Modbus TCP must establish a connection before transferring data, since it is a connection-based protocol. The Master (or Client in Modbus TCP) establishes a connection with the Slave (or Server). The Server waits for an incoming connection from the Client. Once a connection is established, the Server then responds to the queries from the Client until the client closes the connection.

The number of Clients connected to 1 specific RVT is limited to 5.

In summary:

- Modbus TCP allows the user to connect to a RVT12-3P through Ethernet or Internet using Modbus standard protocol (with HMI, SCADA…)
- The slave address of Modbus RTU specification is replaced by the IP address via TCP port 502.
- Multiple Clients may access multiple RVT Servers.


MODBUS
Client
Serial Line
MODBUS
Client
TCP/IP
MODBUS
Client
TCP/IP
Client
TCP/IP
Modbus on Ethernet backbone TCP/UDP, etc
MODBUS
Server
TCP/IP
MODBUS
Server
TCP/IP
MODBUS
Server
Serial Line
MODBUS
Server
Serial Line
Server TCP/IP
gateway
MODBUS Serial

3 Modbus function codes

3.1 Data Addresses in Modbus Messages

Modbus defines 4 address spaces: 2 address spaces for bit addressable data and 2 address spaces for 16 bits addressable data.

Address space	Data	Readable/writable	Modbus name
OXXXX	Output bit	Read & write	Coil status
1XXXX	Input bit	Read	Input status
3XXXX	Input word	Read	Input register
4XXXX	Output word	Read & write	Holding register

Input register address space will be mainly used for measurements.
Holding register address space will contain settings.
All data addresses in Modbus messages are referenced to zero.

For example:

The coil known as ‘coil 1’ in a programmable controller is addressed as coil 0000 in the data address field of a Modbus message.
Coil 127 decimal is addressed as coil 007E hex (126 decimal).
Holding register 40001 is addressed as register 0000 in the data address field of the message.
The function code field already specifies a ‘holding register’ operation. Therefore the ‘4XXXX’ reference is implicit.
Holding register 40108 is addressed as register 006B hex (107 decimal).

3.2 Supported function codes

The following table gives the Modbus functions which are implemented and supported.

The code is the one used in function field of the Modbus message.

The address space concerned and the purpose of the function are given below.

Code	Function	Address range/remark
1	Read coil status	0XXXX reads the on/off status of discrete outputs
2	Read input status	1XXXX reads the on/off status of discrete inputs
3	Read holding registers	4XXXX reads contents of output registers
4	Read input registers	3XXXX reads contents of input registers
5	Force single coil	0XXXX sets the status of a discrete output
6	Preset single register	4XXXX sets the value of a holding register
7	Read exception status	Device specific
8	Diagnostics	Checks the communication system between the master and the slave
11	Fetch comm. event ctr.	Returns the amount of successful read/write operations on data points
12	Fetch comm. event log	Returns log registers of communication events
15	Force multiple coils	0XXXX sets the status of multiple discrete outputs
16	Preset multiple registers	4XXXX sets the value of multiple holding registers
17	Report slave ID	Device specific
22	Mask write 4X registers	4XXXX and/or write of a holding register
23	Read/write 4X registers	4XXXX reads a set of holding registers and writes a set of holding registers in one query

Remark: please note that for security reasons broadcast is not supported by the RVT.

3.3 Master's queries and Slave's responses

When a master device sends a query to a slave device it expects a normal response. One of four possible events can occur from the master's query:

- If the slave device receives the query without a communication error, and can handle the query normally, it returns a normal response.
- If the slave does not receive the query due to a communication error, no response is returned. The master program will eventually process a timeout condition for the query.
- If the slave receives the query, but detects a communication error (parity or CRC), no response is returned. The master program will eventually process a timeout condition for the query.
- If the slave receives the query without a communication error, but cannot handle it (for example, if the request is to read a non–existent coil or register), the slave will return an exception response informing the master of the nature of the error.

3.4 Reads and writes to Modbus addresses (functions 1, 2, 3, 4, 5, 6, 15, 16, 22, 23)

The format of a read function (read coil status (01), read input status (02), read input registers (04), read holding registers (03)) is as follows:

QUERY	
Slave address	1 byte
Function	1 byte
Starting data address	2 bytes
Quantity of points	2 bytes
Error check field CRC	2 bytes

RESPONSE	
Slave address	1 byte (echo of master's query)
Function	1 byte (echo of master's query)
Byte count	1 byte
Data values	N bytes
Error check field CRC	2 bytes

The format of a force single coil (05) or a preset single register (06) function is as follows:

QUERY	
Slave address	1 byte
Function	1 byte
Data address	2 bytes
Data value	2 bytes
Error check field CRC	2 bytes

RESPONSE	
Slave address	1 byte (echo of master's query)
Function	1 byte (echo of master's query)
Data address	2 bytes
Data value	2 bytes
Error check field CRC	2 bytes

The format of a force multiple coil (15) or a preset multiple registers (16) function is as follows:

QUERY	
Slave address	1 byte
Function	1 byte
Data address	2 bytes
Quantity of points	2 bytes
Byte count	1 byte
Data values	N bytes
Error check field CRC	2 bytes

RESPONSE	
Slave address	1 byte (echo of master's query)
Function	1 byte (echo of master's query)
Data address	2 bytes
Quantity of points	2 bytes
Error check field CRC	2 bytes

The format of a read/write multiple registers (23) function is as follows:

QUERY	
Slave address	1 byte
Function	1 byte
Read data address	2 bytes
Read quantity of points	2 bytes
Write data address	2 bytes
Write quantity of points	2 bytes
Byte count	1 byte
Write data values	N bytes
Error check field CRC	2 bytes

RESPONSE	
Slave address	1 byte (echo of master's query)
Function	1 byte (echo of master's query)
Byte count	1 byte
Data values	N bytes
Error check field CRC	2 bytes

The format of a Mask/write register (22) function is as follows:

QUERY	
Slave address	1 byte
Function	1 byte
Data address	2 bytes
And mask	2 bytes
Or mask	2 bytes
Error check field CRC	2 bytes

RESPONSE	
Slave address	1 byte (echo of master's query)
Function	1 byte (echo of master's query)
Data address	2 bytes
And mask	2 bytes
Or mask	2 bytes
Error check field CRC	2 bytes

3.5 Fetch comm event counter (function 11)

The controller's event counter is incremented once for each successful message completion. It is not incremented for exception responses, poll commands, or fetch event counter commands. It returns amount of successful read/write operations on data points.

The format of a Fetch comm event counter (11) function query is as follows:

QUERY	
Slave address	1 byte
Function	1 byte
Error check field CRC	2 bytes

RESPONSE	
Slave address	1 byte (echo of master's query)
Function	1 byte (echo of master's query)
Status word	2 bytes (0)
Event counter	2 bytes
Error check field CRC	2 bytes

3.6 Fetch comm event log (function 12)

Returns a status word, the comm event counter (see function 11), the bus message counter (see function 08 subfunction 11), and a field of event bytes from the slave.

The format of a Fetch comm event log (12) function query is as follows:

QUERY	
Slave address	1 byte
Function	1 byte
Error check field CRC	2 bytes

RESPONSE	
Slave address	1 byte (echo of master's query)
Function	1 byte (echo of master's query)
Byte count	1 byte
Status word	2 bytes (0)
Event counter	2 bytes
Bus message counter	2 bytes
Event log buffer	N bytes
Error check field CRC	2 bytes

The 64 bytes wide Event log buffer is filled with communication events. The most recent communications event is shown in the Event 0 byte.

Event bytes are stored in the Even log buffer for 4 different reasons.

The bit will be set to logic ‘1’ if the corresponding condition is TRUE.

Slave Modbus Receive Event

This type of event byte is stored by the slave when a query message is received.

It is stored before the slave processes the message.

Bit	Contents
0	Not Used
1	Communications Error
2	Not Used
3	Not Used
4	Character Overrun
5	Currently in Listen Only Mode
6	Broadcast Received
7	1

Slave Modbus Send Event

This type of event byte is stored by the slave when it finishes processing a query message.

It is stored if the slave returned a normal or exception response, or no response.

Bit	Contents
0	Read Exception Sent (Exception Codes 1-3)
1	Slave Abort Exception Sent (Exception Code 4)
2	Not used
3	Not used
4	Write Timeout Error Occurred
5	Currently in Listen Only Mode
6	1
7	0

Slave Entered Listen Only Mode

This type of event byte is stored by the slave when it enters the Listen Only Mode.

The event is defined by a content of ‘04’ hex.

Slave Initiated Communication Restart

This type of event byte is stored by the slave when its communications port is restarted. The slave can be restarted by the Diagnostics function (code 08), with subfunction Restart Communications Option (code 01).

The event is defined by a contents of ‘00’ hex.

3.7 Diagnostics function and subfunctions (function 8)

The format of a diagnostics (08) function query is as follows:

QUERY	
Slave address	1 byte
Function	1 byte
Subfunction	2 bytes
Data field	2 bytes
Error check field CRC	2 bytes

The format of a response to a diagnostics function query is an echo of the query itself.

If the request is directed to a counter, however, the slave returns the counter's value in the data field.

00 Return Query Data

The data in the query data field is to be returned (looped back) in the response. The entire response should be identical to the query.

01 Restart Communication Option

The slave's peripheral port is to be initialized and restarted, and all of its communication event counters are to be cleared. If the port is currently in the Listen Only Mode, no response will be sent. If the port is not currently in the Listen Only Mode, a normal response will be sent. This occurs before the restart is executed.

02 Return Diagnostic Register (Not supported)

03 (Not supported)

04 Force Listen Only Mode

Forces the addressed slave to enter the Listen Only Mode for Modbus communications.

10 Clear Counters and Diagnostic Register

Clears all counters and the diagnostic register.

11 Return Bus Message Count

The response data field returns the total quantity of messages that the slave has detected in the communications system since its last restart, clear counters operation, or power-up.

12 Return Bus Communication Error Count

The response data field returns the quantity of CRC errors encountered by the slave since its last restart, clear counters operation, or power-up.

13 Return Bus Exception Error Count

The response data field returns the quantity of Modbus exception responses returned by the slave since its last restart, clear counters operation, or power-up.

14 Return Slave Message Count

The response data field returns the quantity of messages addressed to the slave, or broadcast that the slave has processed since its last restart, clear counters operation, or power-up.

15 Return Slave No Response Count

The response data field returns the quantity of messages addressed to the slave for which it sent no response (neither a normal response nor an exception response) since its last restart, clear counters operation, or power-up.

16 Return Slave NACK Response Count (Not supported)

17 Return Slave Busy Response Count (Not supported)

18 Return Bus Character Overrun Count

The response data field returns the quantity of messages addressed to the slave that it could not handle due to a character overrun condition since its last restart, clear counters operation, or power-up

19 (Not supported)

20 (Not supported)

21 (Not supported)

Diagnostic counters

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

3.8 Exception responses

Exception responses are sent when the slave device cannot handle the query. The format of an exception response to a master's query is as follows:

01 ILLEGAL FUNCTION	The function code received in the query is not an allowable action for the slave device (see paragraph 2.2).
02 ILLEGAL DATA ADDRESS	The data address or number of items received in the query is not allowable or correct for the slave device. The slave device will send this exception response if an attempt to read or write part of a multiple register database object is detected. Possible objects are time, strings and counters
03 ILLEGAL DATA VALUE	A value contained in the query data field is out of range. The contents of the register or the status of the coil has not changed (see paragraph 4.3).

04 SLAVE DEVICE ABORT	An unrecoverable error occurred while the slave was attempting to perform the requested action. This may happen when the access level for changing a parameter is not reached (see paragraph 4.2).
05 ACKNOWLEDGE	Not supported
06 SLAVE DEVICE BUSY	Not supported
07 NEGATIVE ACKNOWLEDGE	Not supported
08 MEMORY PARITY ERROR	Not supported

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

3.9 CRC generation

The Cyclical Redundancy Check (CRC) field is two bytes, containing a 16–bit binary value. The CRC value is calculated by the transmitting device, which appends the CRC to the message. The receiving device recalculates a CRC during receipt of the message, and compares the calculated value to the actual value it received in the CRC field. If the two values are not equal, an error results.

Placing the CRC into the Message:

When the 16–bit CRC (two 8–bit bytes) is transmitted in the message, the low-order byte will be transmitted first, followed by the high-order byte.

Example: here is an example of calculating directly the CRC.

```
/*-----------------------------------------------------------------
FUNCTION        : This routine calculates the crc high and low byte of a message.
  -----------------------------------------------------------------
INPUT PARAMETERS    :    buf  ->  Array containing message to be sent to controller
                         start ->  Start of loop in crc counter, usually 0.
                         cnt  ->  Amount of bytes in message being sent to controller
  -----------------------------------------------------------------
OUTPUT              :    temp  ->  Returns crc byte for message.
  -----------------------------------------------------------------
*/
word crc(byte *buf,word start,word cnt)
{
word        i,j;
word        temp,temp2,flag;
    temp=0xFFFF;
    for (i=start; i<cnt; i++)
    {
            temp=temp ^ buf[i];
            for (j=1; j<=8; j++)
```

```
            {
                flag=temp & 0x0001;
                temp=temp >> 1;
                if (flag) temp=temp ^ 0xA001;
            }
        }
        /* Reverse byte order. */
        temp2=temp >> 8;
        temp=(temp << 8) | temp2;
        temp &= 0xFFFF;
        return(temp);
}
```

4 Ethernet / RJ45 and USB connections for PQ-Link protocol

4.1 General overview

The Data of the RVT can be accessed by different means:

- TCP/IP connection from a local client or from a remote client

- USB seen as a USB UART interface

The server will allow local and distant access to the RVT. Different access levels will be implemented to restrict certain functionality to given users. A login and password will therefore be required.

The format of the messages transferred via those two medium will be the same.

4.2 Physical layer

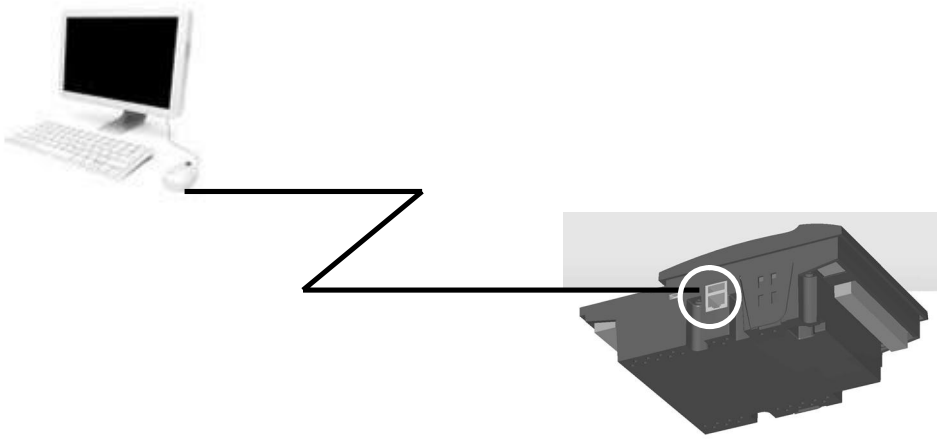
4.2.1 TCP/IP

TCP/IP connections can be indifferently initiated locally or remotely. As the local connection is used by the UI, it will have extended access rights to parameters compared to a remote connection.

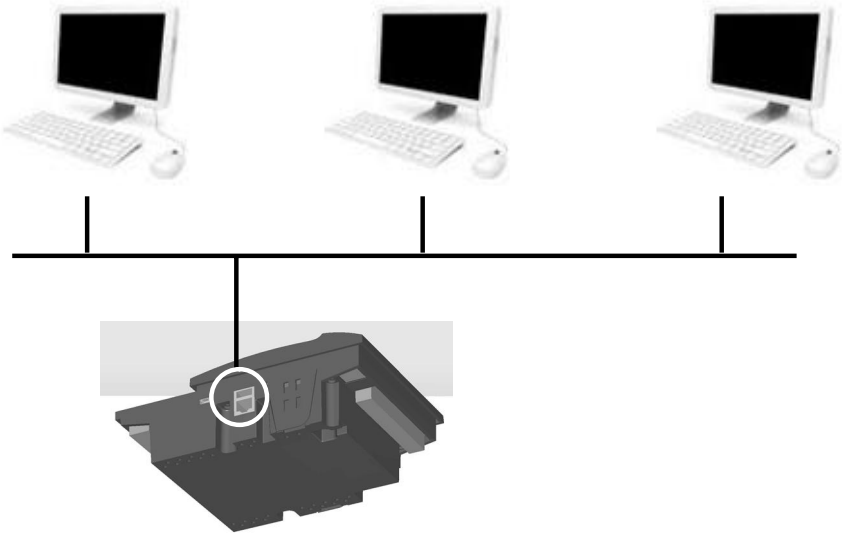
The TCP port used by default is 4250.

The maximum number of TCP/IP clients to the RVT is 2.

The connection to the RVT is an RJ45 Cat5e Ethernet cable



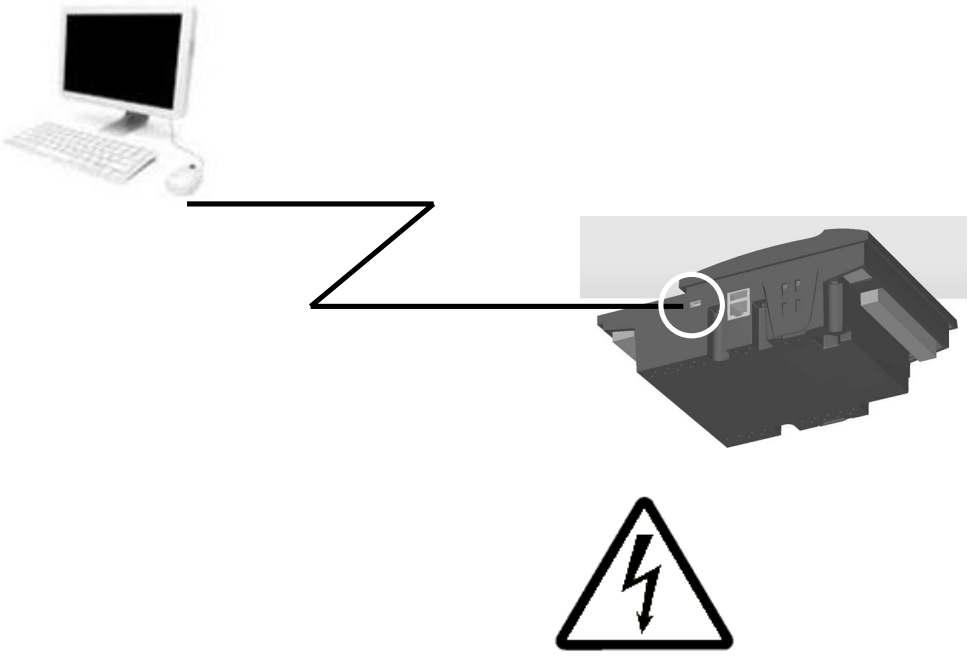
The RVT can be connected directly to a LAN or through Internet



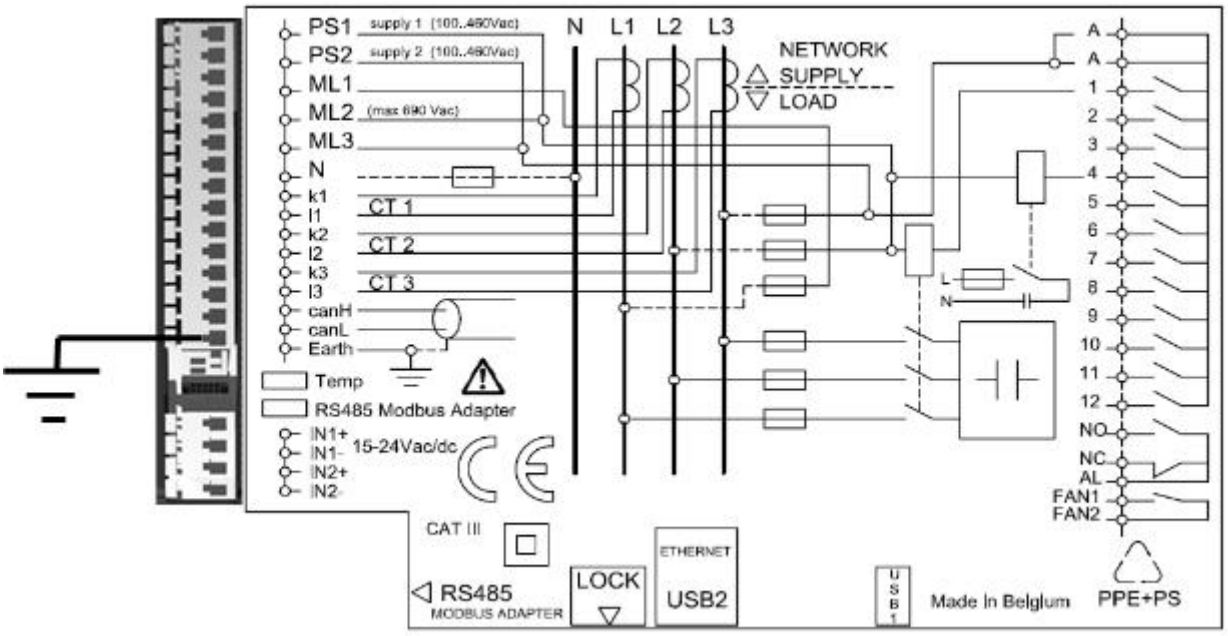
4.2.2 USB

The USB interface is used to present the RVT as a serial interface on its USB port.

The computer is connected through a USB-A male to USB-Mini B male.



Caution: The USB connection to the RVT is not isolated. It is mandatory to connect the protective EARTH connection when using the USB.



4.3 Framing layer & Command layer

The data can be accessed by different means:

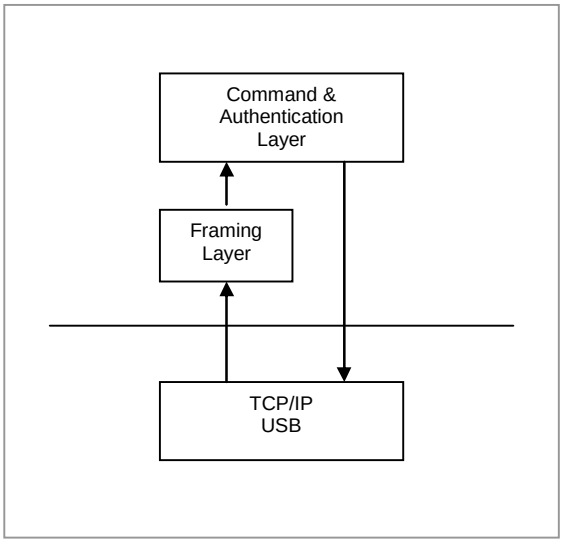
- TCP/IP connection from a local client or from a remote client
- USB seen as a USB UART interface

The server will allow local and distant access to the RVT. Different access levels will be implemented to restrict certain functionality to given users. A login and password will therefore be required.

The format of the messages transferred via those two medium will be the same.

Two layers will be put on top of them:

Framing Layer	This layer is taking care of receiving the frames. The byte stream is decoded and frames generated are passed to the layers above.
Command Layer	This is the upper layer taking care of the commands. It will also take care of the authentication of the client during the connection via some specific commands.



Those two layers will use the same format regardless of the actual “transmission medium” used.

The data is arranged in packets with integrated error checking.

The Windows DLL (chapter 6) incorporates all framing and command issues needed to communicate with the RVT.

5 Data table

5.1 Overview

There's quite an extensive set of parameters available in the RVT.

The individual parameters have been put together in groups to ease manipulation and transfers between the different layers of the application.

The parameters won't be individually accessible to the application. Only groups of parameters will be exchanged between the application and the lower layers.

This will allow the lower layer to be quite independent from the parameters contained within the group.

The parameter groups will be split in two types:

- Groups that are needed by the lower layers. They will have known and fixed group IDs in all different applications.
- Groups which are specific to an application. Their IDs will be known by the applicative layer. The lower-layers won't know the internals of these parameter groups.

5.2 Parameter Types

There are three basic types of parameter groups:

- Configuration parameters are defining the behavior of the system.
- Measurement parameters are generated as data.
- Info parameters are kind of internal information.

The parameters IDs allowed are set as follows:

Type	Sub-type	Access type	Allowed IDs	
			Lower	Upper
Configuration	Universal	R/W	0x0000	0x0FF
	Access protected	R/W	0x0100	0x07FF
	Application Specific	R/W	0x0800	0x0FFF
Measurement	Application Specific	R	0x1000	0x1FFF
Info	Universal	R	0x2000	0x20FF
	Application Specific	R	0x2100	0x2FFF

The access type is given from the perspective of the local or distant user interface.

The "Configuration – Universal" parameters are considered as system parameters. As such, they are only modifiable by users that have at least "Administrator" rights.

The "Configuration – Access Protected" parameters can be modified by users that have at least "Configurator" rights so that their value can be reset or modified.

The "Configuration – Application Specific" parameters can be modified by users that have at least "Configurator" rights.

The other read-only parameters are accessible by any kind of user.

Values of single parameters within a group can be of different types. Here is a list of these types and the associated memory usage:

Type	Size in bytes	Remarks
Byte	1	Unsigned
Signed char	1	Signed
Word	2	Unsigned
Dword	4	Unsigned
64-bits	8	Unsigned
Time	6	Same format as used in the RTC parameter
Float	4	IEEE-754 floating-point number
String	Variable	NULL-terminated ASCII string

5.3 Parameter Changes and access

Please note that the RVT is fitted with some locking function, independently than the administrator rights.

- MODE: the RVT must be in SET Mode to allow parameters settings modifications.
- LOCK SWITCH: the lock switch have to be released
- BANK SETTINGS: the parameter Bank Settings must be set to Unlocked.
- The parameter COMMUNICATION LOCK is used to add an access level to users. When locked, all parameter settings modifications (except the Communication lock item setting) from the RVT touchscreen are forbidden. Parameters may meanwhile be modified by the communication access only (provided all others access levels are fulfilled).

Variable	Group ID	locked	Unlocked
Mode	0x0802	0 : AUTO 1 : MAN	2 : SET
Lock switch status	0x0014	0 : Lock switch pushed	1 : Lock switch released
Bank Settings	0x0802	1 : Bank Settings are locked	0 : Bank Settings are unlocked
Communication Lock	0x0809	1 : Communication lock	0 : Communication unlocked

Note: The RVT returns automatically to AUTO mode when the touch screen is not pressed for more than five minutes.

5.4 Parameter Groups

5.4.1 Configuration

5.4.1.1 Universal

These are the different groups with their size, type and assigned group IDs:

Group ID	Description	Size (in bytes)	Modbus Base address
0x0000	Real-time Clock	6	40000
0x0001	Modbus Data	5	40100
0x0002	Ethernet Data	21	31700
0x0004	Touch screen Calibration Data	17	40400
0x0013	Backlight Settings	1	41900
0x0014	Input Information	5	31400

5.4.1.2 Access Protected

These are the different groups with their size, type and assigned group IDs:

Group ID	Description	Size (in bytes)	Modbus Base Address
0x0100	Event Logging L1-L2	88	30000
0x0101	Event Logging L2-L3	88	30100
0x0102	Event Logging L3-L1	88	30200
0x0103	Event Logging Total	158	30300
0x0104	Event Logging Temperature	160	30400
0x0105	Energy	80	42000
0x0106	Installation Settings 2	48	42100
0x0107	Status information	146	42200
0x0109	Alarm Logging	36	42300

5.4.1.3 RVT specific

These are the different groups with their size, type and assigned group IDs:

Group ID	Description	Size (in bytes)	Modbus Base Address
0x0801	Bank Settings	25	42800
0x0802	I/O	5	42900
0x0803	Protection (Alarm Relay no 1)	62	43000
0x0804	Warning (Alarm Relay no 2)	61	43100
0x0805	Event Logging Settings	140	43200
0x0806	Installation Settings 1	12	43300
0x0807	User Settings	31	43400
0x0808	ID	57	43500
0x0809	Status information	9	32500

5.4.1.4 Measurement

These are the different groups with their size, type and assigned group IDs:

Group ID	Description	Size (in bytes)	Modbus Base Address
0x1000	Voltages	104	30500
0x1001	Line Currents	56	30600
0x1002	Temperature	36	30700
0x1003	Powers	88	30800
0x1005	PFC Control Data	24	30900
0x1006	Status information	32	31000

5.4.1.5 Universal

These are the different groups with their size, type and assigned group IDs:

Group ID	Description	Size (in bytes)	Modbus Base Address
0x2001	Ethernet current settings	20	32600
0x2081	LED control	1	31100

5.5 Parameter List

The parameter list is organized in several group of parameter.

Each group of parameter is identified by a Group ID.

A few data specifies how and where the data is available or can be programmed.

Parameters settings values have a limited range. If a written value exceeds the minimum and maximum allowable values, the written group of parameter will be omitted.

1/ Byte offset - Offset in bytes of the data into the Group of parameters

2/ Description - General description

3/ Units - Units depending of the type of data

4/ RVT - Applicable for the RVT 6 or 12. Some data are not available for the basic standard RVT

5/ RVT3P - Applicable for the RVT 12 with 3 phase measurement. All data are available into this full version of the RVT.

6/ Data type - Format

7/ Size in bytes - Depending on the data type

8/ Default value - Default value programmed as factory settings

9/ Min value - Minimum level allowed by this data

10/ Max value - Maximum level allowed by this data

11/ Modb @ - Base Modbus address where the data is located while accessing through Modbus protocol

5.5.1 Configuration (Universal)

5.5.1.1 Real Time Clock (GroupID = 0x0000)

Byte Offset	Description	Units	RVT	RVT3P	Data type	Size in bytes	Default value	Min value	Max value	Modb @
0	Hours	Hour		*	Byte	1	0	0	23	40001
1	Minutes	Minutes		*	Byte	1	0	0	59	40002
2	Seconds	Seconds		*	Byte	1	0	0	59	40003
3	Year			*	Byte	1	109	0	255	40004
4	Month			*	Byte	1	1	1	12	40005
5	Day			*	Byte	1	1	1	31	40006

The Year 0 is defined as 1900 i.e. the year 2010 will be encoded as 110.

5.5.1.2 Modbus Data (GroupID = 0x0001)

Byte Offset	Description	Units	RVT	RVT3P	Data type	Size in bytes	Default value	Min value	Max value	Modb @
0	Modbus address		*	*	Byte	1	1	1	247	40101
1	NOT USED		*	*	Byte	1	0			40102
2	Modbus baud rate	Bit/second	*	*	Byte	1	9	1	9	40103
3	parity		*	*	Byte	1	0	0	2	40104
4	stop bits		*	*	Byte	1	0	0	1	40105

The ‘Baud rate’ is defined as follows:

Value	Description
1	300 bauds
2	600 bauds
3	1200 bauds
4	2400 bauds
5	4800 bauds
6	9600 bauds
7	19200 bauds
8	38400 bauds
9	57600 bauds

The ‘Parity’ is defined as follows:

Parity	Signification
0	No parity
1	Even
2	Odd

The ‘Stop Bits’ are defined as follows:

Value	Description
0	1 stop bit
1	2 stop bit

5.5.1.3 Ethernet Data (GroupID = 0x0002)

Byte Offset	Description	Units	RVT	RVT3P	Data type	Size in bytes	Default value	Min value	Max value	Modb @
0	Static IP address			*	Dword	4	192.168.1.40	0	0xFFFFFFFF	
4	Static IP mask			*	Dword	4	255.255.255.0	0	0xFFFFFFFF	
8	Static Gateway IP address			*	Dword	4	0	0	0xFFFFFFFF	
12	NOT USED			*	Dword	4	0			
16	NOT USED			*	Dword	4	0			
20	DHCP client enabled			*	Byte	1	1	0	0X01	

The ‘DHCP client enabled’ is defined as follows:

Value	Description
0	DHCP disabled
1	DHCP enabled

The IP addresses are expected to be provided in network order (big endian).

5.5.1.4 Touchscreen Calibration Data (GroupID = 0x0004)

Byte Offset	Description	Units	RVT	RVT3P	Data type	Size in bytes	Default value	Min value	Max value	Modb @
0	X factor 0		*	*	Dword	4	3868	0	0xFFFFFFFF	40401
4	Y factor 0		*	*	Dword	4	3686	0	0xFFFFFFFF	40403
8	X factor 1		*	*	Dword	4	162	0	0xFFFFFFFF	40405
12	Y factor 1		*	*	Dword	4	334	0	0xFFFFFFFF	40407
16	Calibration Done		*	*	Byte	1	1	0	1	40409

The ‘Calibration Done’ is defined as follows:

Value	Description
0	Undone
1	Done

5.5.1.5 Backlight settings (GroupID = 0x0013)

Byte Offset	Description	Units	RVT	RVT3P	Data type	Size in bytes	Default value	Min value	Max value	Modb @
0	Backlight percentage	%	*	*	Byte	1	100	10	100	41901

5.5.1.6 Input information (GroupID = 0x0014)

Byte Offset	Description	Units	RVT	RVT3P	Data type	Size in bytes	Min value	Max value	Modb @
0	NOT USED		*	*	Word	2			31401
2	NOT USED		*	*	Word	2			31402
4	Lock switch status		*	*	Byte	1	0	1	31403

The 'Lock switch' parameter is defined as follows:

Value	Description
1	Unlocked
0	Locked

5.5.2 Configuration (Access Protected)

5.5.2.1 Event Logging L1 (GroupID = 0x0100)

Note : - Voltage loggings refers to ML1-ML2 in case of Connection type Line to Line and ML1-N in case of Connection type L-N

- Current loggings refers to CT1 input

- Powers loggings refers to L1

Please refer to Appendix A7 of the RVT Manual for more information.

Byte Offset	Description	Units	RVT	RVT3P	Data type	Size in bytes	Default value	Min value	Max value	Modb @
0	Urms peak	V		*	Float	4	0	0	9e6	30001
4	Accumulated peak Urms duration	s		*	Time / 6 bytes	6	0:0:0:0:0:0	0:0:0:0:0:0	255:12:30:23:59:59	30003
10	Urms Lowest	V		*	Float	4	0	0	9e6	30006
14	Irms peak	A	*	*	Float	4	0	0	9e6	30008
18	Accumulated peak Irms duration	s	*	*	Time / 6 bytes	6	0:0:0:0:0:0	0:0:0:0:0:0	255:12:30:23:59:59	30010
24	Irms Lowest	A	*	*	Float	4	0	0	9e6	30013
28	peak active power	W		*	Float	4	0	-1e9	1e9	30015
32	Accumulated peak active power duration	s		*	Time / 6 bytes	6	0:0:0:0:0:0	0:0:0:0:0:0	255:12:30:23:59:59	30017
38	peak reactive power	var		*	Float	4	0	-1e9	1e9	30020
42	Accumulated peak reactive power duration	s		*	Time / 6 bytes	6	0:0:0:0:0:0	0:0:0:0:0:0	255:12:30:23:59:59	30022
48	peak missing reactive power	var		*	Float	4	0	-1e9	1e9	30025
52	Accumulated peak missing reactive power duration	s		*	Time / 6 bytes	6	0:0:0:0:0:0	0:0:0:0:0:0	255:12:30:23:59:59	30027
58	peak apparent power	VA		*	Float	4	0	-1e9	1e9	30030
62	Accumulated peak apparent power duration	s		*	Time / 6 bytes	6	0:0:0:0:0:0	0:0:0:0:0:0	255:12:30:23:59:59	30032
68	peak THDU	%		*	Float	4	0	0	1000	30035

72	Accumulated peak THDU duration	s		*	Time / 6 bytes	6	0:0:0:0:0:0	0:0:0:0:0:0	255:12:30:23:59:59	30037
78	peak THDI	%	*	*	Float	4	0	0	1000	30040
82	Accumulated peak THDI duration	s	*	*	Time / 6 bytes	6	0:0:0:0:0:0	0:0:0:0:0:0	255:12:30:23:59:59	30042

The communication from the user to the RVT is limited to the reset of data, it means that all the Group ID is transmitted with some data set to 0 and 0:0:0:0:0:0.

5.5.2.2 Event Logging L2 (GroupID = 0x0101)

Note : - Voltage loggings refers to ML2-ML3 in case of Connection type Line to Line and ML2-N in case of Connection type L-N

- Current loggings refers to CT2 input

- Powers loggings refers to L2

Please refer to Appendix A7 of the RVT Manual for more information.

Byte Offset	Description	Units	RVT	RVT3P	Data type	Size in bytes	Default value	Min value	Max value	Modb @
0	Urms peak	V	*	*	Float	4	0	0	9e6	30101
4	Accumulated peak Urms duration	s	*	*	Time / 6 bytes	6	0:0:0:0:0:0	0:0:0:0:0:0	255:12:30:23:59:59	30103
10	Urms Lowest	V	*	*	Float	4	0	0	9e6	30106
14	Irms peak	A		*	Float	4	0	0	9e6	30108
18	Accumulated peak Irms duration	s		*	Time / 6 bytes	6	0:0:0:0:0:0	0:0:0:0:0:0	255:12:30:23:59:59	30110
24	Irms Lowest	A		*	Float	4	0	0	9e6	30113
28	peak active power	W		*	Float	4	0	-1e9	1e9	30115
32	Accumulated peak active power duration	s		*	Time / 6 bytes	6	0:0:0:0:0:0	0:0:0:0:0:0	255:12:30:23:59:59	30117
38	peak reactive power	var		*	Float	4	0	-1e9	1e9	30120
42	Accumulated peak reactive power duration	s		*	Time / 6 bytes	6	0:0:0:0:0:0	0:0:0:0:0:0	255:12:30:23:59:59	30122
48	peak missing reactive power	var		*	Float	4	0	-1e9	1e9	30125
52	Accumulated peak missing reactive power duration	s		*	Time / 6 bytes	6	0:0:0:0:0:0	0:0:0:0:0:0	255:12:30:23:59:59	30127
58	peak apparent power	VA		*	Float	4	0	-1e9	1e9	30130
62	Accumulated peak apparent power duration	s		*	Time / 6 bytes	6	0:0:0:0:0:0	0:0:0:0:0:0	255:12:30:23:59:59	30132
68	peak THDU	%	*	*	Float	4	0	0	1000	30135
72	Accumulated peak THDU duration	s	*	*	Time / 6 bytes	6	0:0:0:0:0:0	0:0:0:0:0:0	255:12:30:23:59:59	30137
78	peak THDI	%		*	Float	4	0	0	1000	30140
82	Accumulated peak THDI duration	s		*	Time / 6 bytes	6	0:0:0:0:0:0	0:0:0:0:0:0	255:12:30:23:59:59	30142

The communication from the user to the RVT is limited to the reset of data, it means that all the Group ID is transmitted with some data set to 0 and 0:0:0:0:0:0.

5.5.2.3 Event Logging L3 (GroupID = 0x0102)

Note : - Voltage loggings refers to ML3-ML1 in case of Connection type Line to Line and ML3-N in case of Connection type L-N

- Current loggings refers to CT3 input

- Powers loggings refers to L3

Please refer to Appendix A7 of the RVT Manual for more information.

Byte Offset	Description	Units	RVT	RVT3P	Data type	Size in bytes	Default value	Min value	Max value	Modb @
0	Urms peak	V		*	Float	4	0	0	9e6	30201
4	Accumulated peak Urms duration	s		*	Time / 6 bytes	6	0:0:0:0:0:0	0:0:0:0:0:0	255:12:30:23:59:59	30203
10	Urms Lowest	V		*	Float	4	0	0	9e6	30206
14	Irms peak	A		*	Float	4	0	0	9e6	30208
18	Accumulated peak Irms duration	s		*	Time / 6 bytes	6	0:0:0:0:0:0	0:0:0:0:0:0	255:12:30:23:59:59	30210
24	Irms Lowest	A		*	Float	4	0	0	9e6	30213
28	peak active power	W		*	Float	4	0	-1e9	1e9	30215
32	Accumulated peak active power duration	s		*	Time / 6 bytes	6	0:0:0:0:0:0	0:0:0:0:0:0	255:12:30:23:59:59	30217
38	peak reactive power	var		*	Float	4	0	-1e9	1e9	30220
42	Accumulated peak reactive power duration	s		*	Time / 6 bytes	6	0:0:0:0:0:0	0:0:0:0:0:0	255:12:30:23:59:59	30222
48	peak missing reactive power	var		*	Float	4	0	-1e9	1e9	30225
52	Accumulated peak missing reactive power duration	s		*	Time / 6 bytes	6	0:0:0:0:0:0	0:0:0:0:0:0	255:12:30:23:59:59	30227
58	peak apparent power	VA		*	Float	4	0	-1e9	1e9	30230
62	Accumulated peak apparent power duration	s		*	Time / 6 bytes	6	0:0:0:0:0:0	0:0:0:0:0:0	255:12:30:23:59:59	30232
68	peak THDU	%		*	Float	4	0	0	1000	30235
72	Accumulated peak THDU duration	s		*	Time / 6 bytes	6	0:0:0:0:0:0	0:0:0:0:0:0	255:12:30:23:59:59	30237
78	peak THDI	%		*	Float	4	0	0	1000	30240
82	Accumulated peak THDI duration	s		*	Time / 6 bytes	6	0:0:0:0:0:0	0:0:0:0:0:0	255:12:30:23:59:59	30242

The communication from the user to the RVT is limited to the reset of data, it means that all the Group ID is transmitted with some data set to 0 and 0:0:0:0:0:0.

5.5.2.4 Event Logging Total (GroupID = 0x0103)

Note : - Voltage loggings refers to the averaging of the voltage measurements on the 3 phases. In case of Connection type where only one voltage measurement is present, please refer to the above Event Logging tables.

- Current loggings refers to the averaging of the current measurements on the 3 phases. In case of Connection type where only one current measurement is present, please refer to the above Event Logging tables.

- Powers loggings refers to the power calculations on all phases.

Please refer to Appendix A7 of the RVT Manual for more information.

Byte Offset	Description	Units	RVT	RVT3P	Data type	Size in bytes	Default value	Min value	Max value	Modb @
0	Urms L-L peak	V		*	Float	4	0	0	9e6	30301
4	Accumulated peak Urms L-L duration	S		*	Time / 6 bytes	6	0:0:0:0:0:0	0:0:0:0:0:0	255:12:30:23:59:59	30303
10	Urms L-L Lowest	V		*	Float	4	0	0	9e6	30306
14	Irms peak	A		*	Float	4	0	0	9e6	30308
18	Accumulated peak Irms duration	S		*	Time / 6 bytes	6	0:0:0:0:0:0	0:0:0:0:0:0	255:12:30:23:59:59	30310
24	Irms Lowest	A		*	Float	4	0	0	9e6	30313
28	peak active power	W	*	*	Float	4	0	-1e9	1e9	30315
32	Accumulated peak active power duration	S	*	*	Time / 6 bytes	6	0:0:0:0:0:0	0:0:0:0:0:0	255:12:30:23:59:59	30317
38	peak reactive power	var	*	*	Float	4	0	-1e9	1e9	30320
42	Accumulated peak reactive power duration	S	*	*	Time / 6 bytes	6	0:0:0:0:0:0	0:0:0:0:0:0	255:12:30:23:59:59	30322
48	peak missing reactive power	var	*	*	Float	4	0	-1e9	1e9	30325
52	Accumulated peak missing reactive power duration	S	*	*	Time / 6 bytes	6	0:0:0:0:0:0	0:0:0:0:0:0	255:12:30:23:59:59	30327
58	peak apparent power	VA	*	*	Float	4	0	-1e9	1e9	30330
62	Accumulated peak apparent power duration	S	*	*	Time / 6 bytes	6	0:0:0:0:0:0	0:0:0:0:0:0	255:12:30:23:59:59	30332
68	peak THDU	%		*	Float	4	0	0	1000	30335
72	Accumulated peak THDU duration	S		*	Time / 6 bytes	6	0:0:0:0:0:0	0:0:0:0:0:0	255:12:30:23:59:59	30337
78	peak THDI	%		*	Float	4	0	0	1000	30340
82	Accumulated peak THDI duration	S		*	Time / 6 bytes	6	0:0:0:0:0:0	0:0:0:0:0:0	255:12:30:23:59:59	30342
88	peak Supplied Active energy	W		*	Float	4	0	0	1e9	30345
92	Accumulated peak Supplied Active energy duration	S		*	Time / 6 bytes	6	0:0:0:0:0:0	0:0:0:0:0:0	255:12:30:23:59:59	30347
98	peak Consumed Active energy	W		*	Float	4	0	0	1e9	30350
102	Accumulated peak Consumed Active energy duration	S		*	Time / 6 bytes	6	0:0:0:0:0:0	0:0:0:0:0:0	255:12:30:23:59:59	30352
108	peak Inductive Reactive energy	var		*	Float	4	0	0	1e9	30355
112	Accumulated peak Inductive Reactive energy duration	S		*	Time / 6 bytes	6	0:0:0:0:0:0	0:0:0:0:0:0	255:12:30:23:59:59	30357

118	peak Capacitive Reactive energy	var		*	Float	4	0	0	1e9	30360
122	Accumulated peak Capacitive Reactive energy duration	S		*	Time / 6 bytes	6	0:0:0:0:0:0	0:0:0:0:0:0	255:12:30:23:59:59	30362
128	peak Total Apparent energy	VA		*	Float	4	0	0	1e9	30365
132	Accumulated peak Total Apparent energy duration	S		*	Time / 6 bytes	6	0:0:0:0:0:0	0:0:0:0:0:0	255:12:30:23:59:59	30367
138	peak frequency max	Hz	*	*	Float	4	40	40	70	30370
142	Accumulated peak frequency max duration	S	*	*	Time / 6 bytes	6	0:0:0:0:0:0	0:0:0:0:0:0	255:12:30:23:59:59	30372
148	peak frequency min	Hz	*	*	Float	4	70	40	70	30375
152	Accumulated peak frequency min duration	S	*	*	Time / 6 bytes	6	0:0:0:0:0:0	0:0:0:0:0:0	255:12:30:23:59:59	30377

The communication from the user to the RVT is limited to the reset of data, it means that all the Group ID is transmitted with some data set to 0 and 0:0:0:0:0:0.

5.5.2.5 Event Logging Temperature (GroupID = 0x0104)

Byte Offset	Description	Units	RVT	RVT3P	Data type	Size in bytes	Default value	Min value	Max value	Modb @
0	peak Temperature max T1	°C/°F	*	*	Float	4	-40	-40	150	30401
4	Accumulated Temperature max T1 duration	S	*	*	Time / 6 bytes	6	0:0:0:0:0:0	0:0:0:0:0:0	255:12:30:23:59:59	30403
10	peak Temperature min T1	°C/°F	*	*	Float	4	150	-40	150	30406
14	Accumulated Temperature min T1 duration	°C/°F	*	*	Time / 6 bytes	6	0:0:0:0:0:0	0:0:0:0:0:0	255:12:30:23:59:59	30408
20	peak Temperature max T2	S	*	*	Float	4	-40	-40	150	30411
24	Accumulated Temperature max T2 duration	°C/°F	*	*	Time / 6 bytes	6	0:0:0:0:0:0	0:0:0:0:0:0	255:12:30:23:59:59	30413
30	peak Temperature min T2	°C/°F	*	*	Float	4	150	-40	150	30416
34	Accumulated Temperature min T2 duration	S	*	*	Time / 6 bytes	6	0:0:0:0:0:0	0:0:0:0:0:0	255:12:30:23:59:59	30418
40	peak Temperature max T3	°C/°F	*	*	Float	4	-40	-40	150	30421
44	Accumulated Temperature max T3 duration	S	*	*	Time / 6 bytes	6	0:0:0:0:0:0	0:0:0:0:0:0	255:12:30:23:59:59	30423
50	peak Temperature min T3	°C/°F	*	*	Float	4	150	-40	150	30426
54	Accumulated Temperature min T3 duration	S	*	*	Time / 6 bytes	6	0:0:0:0:0:0	0:0:0:0:0:0	255:12:30:23:59:59	30428
60	peak Temperature max T4	°C/°F	*	*	Float	4	-40	-40	150	30431
64	Accumulated Temperature max T4 duration	S	*	*	Time / 6 bytes	6	0:0:0:0:0:0	0:0:0:0:0:0	255:12:30:23:59:59	30433
70	peak Temperature min T4	°C/°F	*	*	Float	4	150	-40	150	30436
74	Accumulated Temperature min T4 duration	S	*	*	Time / 6 bytes	6	0:0:0:0:0:0	0:0:0:0:0:0	255:12:30:23:59:59	30438
80	peak Temperature max T5	°C/°F	*	*	Float	4	-40	-40	150	30441
84	Accumulated Temperature max T5 duration	S	*	*	Time / 6 bytes	6	0:0:0:0:0:0	0:0:0:0:0:0	255:12:30:23:59:59	30443
90	peak Temperature min T5	°C/°F	*	*	Float	4	150	-40	150	30446
94	Accumulated Temperature min T5 duration	S	*	*	Time / 6 bytes	6	0:0:0:0:0:0	0:0:0:0:0:0	255:12:30:23:59:59	30448
100	peak Temperature max T6	°C/°F	*	*	Float	4	-40	-40	150	30451
104	Accumulated Temperature max T6 duration	S	*	*	Time / 6 bytes	6	0:0:0:0:0:0	0:0:0:0:0:0	255:12:30:23:59:59	30453

110	peak Temperature min T6	°C/°F	*	*	Float	4	150	-40	150	30456
114	Accumulated Temperature min T6 duration	S	*	*	Time / 6 bytes	6	0:0:0:0:0:0	0:0:0:0:0:0	255:12:30:23:59:59	30458
120	peak Temperature max T7	°C/°F	*	*	Float	4	-40	-40	150	30461
124	Accumulated Temperature max T7 duration	S	*	*	Time / 6 bytes	6	0:0:0:0:0:0	0:0:0:0:0:0	255:12:30:23:59:59	30463
130	peak Temperature min T7	°C/°F	*	*	Float	4	150	-40	150	30466
134	Accumulated Temperature min T7 duration	S	*	*	Time / 6 bytes	6	0:0:0:0:0:0	0:0:0:0:0:0	255:12:30:23:59:59	30468
140	peak Temperature max T8	°C/°F	*	*	Float	4	-40	-40	150	30471
144	Accumulated Temperature max T8 duration	S	*	*	Time / 6 bytes	6	0:0:0:0:0:0	0:0:0:0:0:0	255:12:30:23:59:59	30473
150	peak Temperature min T8	°C/°F	*	*	Float	4	150	-40	150	30476
154	Accumulated Temperature min T8 duration	S	*	*	Time / 6 bytes	6	0:0:0:0:0:0	0:0:0:0:0:0	255:12:30:23:59:59	30478

The communication from the user to the RVT is limited to the reset of data, it means that all the Group ID is transmitted with some data set to 0 and 0:0:0:0:0:0.

5.5.2.6 Energy (GroupID = 0x0105)

Byte Offset	Description	Units	RVT	RVT3P	Data type	Size in bytes	Min value	Max value	Modb @
0	Supplied Active energy L1	Wh		*	Float	4	-1e12	1e12	42001
4	Supplied Active energy L2	Wh		*	Float	4	-1e12	1e12	42003
8	Supplied Active energy L3	Wh		*	Float	4	-1e12	1e12	42005
12	Supplied Active energy	Wh		*	Float	4	-1e12	1e12	42007
16	Consumed Active energy L1	Wh		*	Float	4	-1e12	1e12	42009
20	Consumed Active energy L2	Wh		*	Float	4	-1e12	1e12	42011
24	Consumed Active energy L3	Wh		*	Float	4	-1e12	1e12	42013
28	Consumed Active energy	Wh		*	Float	4	-1e12	1e12	42015
32	Total Active energy	Wh		*	Float	4	-1e12	1e12	42017
36	Inductive Reactive energy L1	varh		*	Float	4	-1e12	1e12	42019
40	Inductive Reactive energy L2	varh		*	Float	4	-1e12	1e12	42021
44	Inductive Reactive energy L3	varh		*	Float	4	-1e12	1e12	42023
48	Capacitive Reactive energy L1	varh		*	Float	4	-1e12	1e12	42025
52	Capacitive Reactive energy L2	varh		*	Float	4	-1e12	1e12	42027
56	Capacitive Reactive energy L3	varh		*	Float	4	-1e12	1e12	42029
60	Total Reactive energy	varh		*	Float	4	-1e12	1e12	42031
64	Total Apparent energy L1	VAh		*	Float	4	-1e12	1e12	42033
68	Total Apparent energy L2	VAh		*	Float	4	-1e12	1e12	42035
72	Total Apparent energy L3	VAh		*	Float	4	-1e12	1e12	42037
76	Total Apparent energy	VAh		*	Float	4	-1e12	1e12	42039

The communication from the user to the RVT is limited to the reset of data, it means that all the Group ID is transmitted with some data set to 0 and 0:0:0:0:0:0.

5.5.2.7 Installation Settings 2 (GroupID = 0x0106)

Byte Offset	Description	Units	RVT	RVT3P	Data type	Size in bytes	Default value	Min value	Max value	Modb @
0	C/k 1Ph	A		*	Float	4	1	0.01	5	42101
4	C/k 3Ph	A	*	*	Float	4	1	0.01	5	42103
8	Qstep 1Ph	var		*	Float	4	50000	10	100e6	42105
12	Qstep 3Ph	var	*	*	Float	4	50000	10	100e6	42107
16	Phase Shift	°	*	*	Float	4	90	-179	180	42109
20	size output 1	Step	*	*	Byte	1	1	0	9	42111
21	size output 2	Step	*	*	Byte	1	1	0	9	42112
22	size output 3	Step	*	*	Byte	1	1	0	9	42113
23	size output 4	Step	*	*	Byte	1	1	0	9	42114
24	size output 5	Step	*	*	Byte	1	1	0	9	42115
25	size output 6	Step	*	*	Byte	1	1	0	9	42116
26	size output 7	Step	*	*	Byte	1	1	0	9	42117
27	size output 8	Step	*	*	Byte	1	1	0	9	42118
28	size output 9	Step	*	*	Byte	1	1	0	9	42119
29	size output 10	Step	*	*	Byte	1	1	0	9	42120
30	size output 11	Step	*	*	Byte	1	1	0	9	42121
31	size output 12	Step	*	*	Byte	1	1	0	9	42122
32	Status output 1		*	*	Byte	1	1	0	5	42123
33	Status output 2		*	*	Byte	1	1	0	5	42124
34	Status output 3		*	*	Byte	1	1	0	5	42125
35	Status output 4		*	*	Byte	1	1	0	5	42126
36	Status output 5		*	*	Byte	1	1	0	5	42127
37	Status output 6		*	*	Byte	1	1	0	5	42128
38	Status output 7		*	*	Byte	1	1	0	5	42129
39	Status output 8		*	*	Byte	1	1	0	5	42130
40	Status output 9		*	*	Byte	1	1	0	5	42131
41	Status output 10		*	*	Byte	1	1	0	5	42132
42	Status output 11		*	*	Byte	1	1	0	5	42133
43	Status output 12		*	*	Byte	1	1	0	5	42134
44	RedirectCTInput L1		*	*	Byte	1	0	0	2	42135
45	RedirectCTInput L2			*	Byte	1	1	0	2	42136
46	RedirectCTInput L3			*	Byte	1	2	0	2	42137
47	Phase rotation		*	*	Byte	1	0	0	1	42138

Bidirectional update is necessary to let the user set the data in manual mode and retrieve calculated value after auto commissioning.

The ‘Output status’ parameter is defined as follows:

Value	Description
0	Fixed OFF
1	1Ph L1
2	1Ph L2
3	1Ph L3
4	3Ph
5	Fixed ON

5.5.2.8 Status information (GroupID = 0x0107)

Byte Offset	Description	Units	RVT	RVT3P	Data type	Size in bytes	Default value	Min value	Max value	Modb @
0	Add step status 1Ph L1			*	Byte	1	0	0	1	42201
1	Add step status 1Ph L2			*	Byte	1	0	0	1	42202
2	Add step status 1Ph L3			*	Byte	1	0	0	1	42203
3	Add step status 3Ph		*	*	Byte	1	0	0	1	42204
4	Remove step status 1Ph L1			*	Byte	1	0	0	1	42205
5	Remove step status 1Ph L2			*	Byte	1	0	0	1	42206
6	Remove step status 1Ph L3			*	Byte	1	0	0	1	42207
7	Remove step status 3Ph		*	*	Byte	1	0	0	1	42208
8	Next relay status		*	*	Word	2	0	0	0x3FFF	42209
10	Step size in MAN mode 1Ph L1	Steps	*	*	Word	2	0	0	500	42210
12	Step size in MAN mode 1Ph L2	Steps	*	*	Word	2	0	0	500	42211
14	Step size in MAN mode 1Ph L3	Steps	*	*	Word	2	0	0	500	42212
16	Step size in MAN mode 3Ph	Steps	*	*	Word	2	0	0	500	42213
18	Address Low External temperature probe 1		*	*	64 bits	8	0	0	0xFFF…	42214
26	Address High External temperature probe 1		*	*	64 bits	8	0	0	0xFFF…	42218
34	Address Low External temperature probe 2		*	*	64 bits	8	0	0	0xFFF…	42222
42	Address High External temperature probe 2		*	*	64 bits	8	0	0	0xFFF…	42226
50	Address Low External temperature probe 3		*	*	64 bits	8	0	0	0xFFF…	42230
58	Address High External temperature probe 3		*	*	64 bits	8	0	0	0xFFF…	42234
66	Address Low External temperature probe 4		*	*	64 bits	8	0	0	0xFFF…	42238
74	Address High External temperature probe 4		*	*	64 bits	8	0	0	0xFFF…	42242
82	Address Low External temperature probe 5		*	*	64 bits	8	0	0	0xFFF…	42246
90	Address High External temperature probe 5		*	*	64 bits	8	0	0	0xFFF…	42250
98	Address Low External temperature probe 6		*	*	64 bits	8	0	0	0xFFF…	42254
106	Address High External temperature probe 6		*	*	64 bits	8	0	0	0xFFF…	42258
114	Address Low External temperature probe 7		*	*	64 bits	8	0	0	0xFFF…	42262

122	Address High External temperature probe 7		*	*	64 bits	8	0	0	0xFFF…	42266
130	Address Low External temperature probe 8		*	*	64 bits	8	0	0	0xFFF…	42270
138	Address High External temperature probe 8		*	*	64 bits	8	0	0	0xFFF…	42274

The ‘Add step status’ parameter is defined as follows:

Value	Description
0	No change
1	Ask for at least 1 more step

The ‘Remove step status’ parameter is defined as follows:

Value	Description
0	No change
1	Ask for at least 1 more step

These step statuses are updated according to the Auto mode or Manual mode.

Value	Description
0	Relay to open
1	Relay to close

The ‘Next relay Status’ parameter is defined as follows:

Bit used	Output relay
Bit 0	1
Bit 1	2
Bit 2	3
Bit 3	4
Bit 4	5
Bit 5	6
Bit 6	7
Bit 7	8
Bit 8	9
Bit 9	10
Bit 10	11
Bit 11	12
Bit 12	Alarm
Bit 13	Fan
Bit 14	Not used
Bit 15	Not used

5.5.2.9 Alarm Logging (GroupID = 0x0109)

Byte Offset	Description	Units	RVT	RVT3P	Data type	Size in bytes	Default value	Min value	Max value	Modb @
0	Buffer 0	Alarm Type	*	*	Byte	1	0	0	9	42301
1	Time stamp of alarm in buffer 0	s		*	Time / 6 bytes	6	0:0:0:0:0:0	0:0:0:0:0:0	255:12:30:23:59:59	42302
7	Buffer 1	Alarm Type	*	*	Byte	1	0	0	9	42305
8	Time stamp of alarm in buffer 1	s		*	Time / 6 bytes	6	0:0:0:0:0:0	0:0:0:0:0:0	255:12:30:23:59:59	42306
14	Buffer 2	Alarm Type	*	*	Byte	1	0	0	9	42309
15	Time stamp of alarm in buffer 2	s		*	Time / 6 bytes	6	0:0:0:0:0:0	0:0:0:0:0:0	255:12:30:23:59:59	42310
21	Buffer 3	Alarm Type	*	*	Byte	1	0	0	9	42313
22	Time stamp of alarm in buffer 3	s		*	Time / 6 bytes	6	0:0:0:0:0:0	0:0:0:0:0:0	255:12:30:23:59:59	42314
28	Buffer 4	Alarm Type	*	*	Byte	1	0	0	9	42317
29	Time stamp of alarm in buffer 4	s		*	Time / 6 bytes	6	0:0:0:0:0:0	0:0:0:0:0:0	255:12:30:23:59:59	42318
35	Buffer index		*	*	Byte	1	0	0	5	42321

This table contains the alarm messages and the time stamp of their occurrences.

There is a circular buffer where both information are stored

- Kind of alarm logged.
- Time stamp.

This buffer may contain until 5 consecutive alarms.

A buffer index points to the eldest alarm logged.

When the buffer is full, the eldest alarm in the buffer is overwritten with the new one and the index is incremented.

The alarm type is defined as follows:

Value	Type	Description
0	No alarm	This alarm buffer is unused for now
1	Protection cos j	Insufficient available reactive power
2	Protection Temp Sensor	Temperature sensor lost while monitoring
3	Protection U Max	Overvoltage detection
4	Protection T Max	Internal temperature threshold reached
5	Protection External T Max	Temperature sensor threshold reached
6	Protection I Max	Over current detection
7	Protection THDU	THDU threshold reached
8	Protection External	External input protection activated
9	Protection U Min	Under voltage detection

5.5.3 Configuration (RVT Specific)

5.5.3.1 Bank Settings (GroupID = 0x0801)

Byte Offset	Description	Units	RVT	RVT3P	Data type	Size in bytes	Default value	Min value	Max value	Modb @
0	V nominal	V	*	*	Float	4	400	10	9e6	42801
4	V scale		*	*	Float	4	1.0	0.1	10000	42803
8	1Ph/3Ph/3Ph-N		*	*	Byte	1	1	0	7	42805
9	Lin / Circ		*	*	Byte	1	1	0	1	42806
10	Prog/direct		*	*	Byte	1	1	0	1	42807
11	Normal / Integral		*	*	Byte	1	1	0	1	42808
12	NOT USED		*	*	Byte	1	0			42809
13	delay ON	Sec	*	*	Dword	4	40	1	64800	42810
17	delay OFF	Sec	*	*	Dword	4	40	1	64800	42812
21	Delay reset	Sec	*	*	Dword	4	3	1	64800	42814

The ‘1Ph/3Ph/3Ph-N’ parameter is defined as follows:

Value	Description
0	1Ph-1LL1
1	3Ph-1LL1
2	3Ph-1LN1
3	3Ph-3LL3
4	3Ph-3LL2
5	3Ph-3LN3
6	3Ph-1LL3
7	3Ph-1LN3

The ‘Linear/Circular’ parameter is defined as follows:

Value	Description
0	Linear
1	Circular

The ‘Progressive/Direct’ parameter is defined as follows:

Value	Description
0	Progressive
1	Direct

The ‘Normal/Integral’ parameter is defined as follows:

Value	Description
0	Normal
1	Integral

5.5.3.2 I/O (GroupID = 0x0802)

Byte Offset	Description	Units	RVT	RVT3P	Data type	Size in bytes	Default value	Min value	Max value	Modb @
0	Language		*	*	Byte	1	0	0	4	42901
1	Temperature unit		*	*	Byte	1	0	0	1	42902
2	Mode		*	*	Byte	1	0	0	2	42903
3	Bank settings lock	Unlocked/ Locked	*	*	Byte	1	0	0	1	42904
4	NOT USED		*	*	Byte	1	100			42905

The ‘Language parameter’ is defined as follows:

Value	Description
0	English
1	French
2	Deutsch
3	Spanish
4	Chinese

The ‘Temperature unit’ parameter is defined as follows:

Value	Description
0	°C
1	°F

The ‘Mode’ parameter is defined as follows:

Value	Description
0	Auto
1	Manual
2	Set

The ‘Bank Settings Lock’ parameter is defined as follows:

Value	Description
0	Unlocked
1	Locked

5.5.3.3 Protection (alarm relay n°1) (GroupID = 0x0803)

Byte Offset	Description	Units	RVT	RVT3P	Data type	Size in bytes	Default value	Min value	Max value	Modb @
0	V min prot.	V	*	*	Float	4	90	5	9e6	43001
4	V max prot.	V	*	*	Float	4	480	5	9e6	43003
8	T1 max prot.	°C or °F	*	*	Float	4	40	-40	150	43005
12	T2 max prot.	°C or °F	*	*	Float	4	40	-40	150	43007
16	T3 max prot.	°C or °F	*	*	Float	4	40	-40	150	43009
20	T4 max prot.	°C or °F	*	*	Float	4	40	-40	150	43011
24	T5 max prot.	°C or °F	*	*	Float	4	40	-40	150	43013
28	T6 max prot.	°C or °F	*	*	Float	4	40	-40	150	43015
32	T7 max prot.	°C or °F	*	*	Float	4	40	-40	150	43017
36	T8 max prot.	°C or °F	*	*	Float	4	40	-40	150	43019
40	THDV max prot.	%	*	*	Float	4	10	0.5	1000	43021
44	Irms max prot.	Ä	*	*	Float	4	1	0.1	9e6	43023
48	Ext. prot.		*	*	Byte	1	0	0	1	43025
49	V min Enable		*	*	Byte	1	0	0	1	43026
50	V max Enable		*	*	Byte	1	0	0	1	43027
51	T1 max Enable.		*	*	Byte	1	0	0	1	43028
52	T2 max Enable.		*	*	Byte	1	0	0	1	43029
53	T3 max Enable.		*	*	Byte	1	0	0	1	43030
54	T4 max Enable.		*	*	Byte	1	0	0	1	43031
55	T5 max Enable.		*	*	Byte	1	0	0	1	43032
56	T6 max Enable.		*	*	Byte	1	0	0	1	43033
57	T7 max Enable.		*	*	Byte	1	0	0	1	43034
58	T8 max Enable.		*	*	Byte	1	0	0	1	43035
59	THDV max Enable.		*	*	Byte	1	0	0	1	43036
60	Irms max Enable.		*	*	Byte	1	0	0	1	43037
61	Ext. Prot. Alarm		*	*	Byte	1	0	0	1	43038

The External protection parameter is defined as follows:

Value	Description
0	Disabled
1	Enabled

The External protection Alarm parameter is defined as follows:

Value	Description
0	Disconnection & alarm
1	Disconnection only

5.5.3.4 Warning (warning/fan relay) (GroupID = 0x0804)

Byte Offset	Description	Units	RVT	RVT3P	Data type	Size in bytes	Default value	Min value	Max value	Modb @
0	V min warning	V	*	*	Float	4	90	5	9e6	43101
4	V max warning	V	*	*	Float	4	480	5	9e6	43103
8	T1 max start fan	°C or °F	*	*	Float	4	40	-40	150	43105
12	T2 max start fan	°C or °F	*	*	Float	4	40	-40	150	43107
16	T3 max start fan	°C or °F	*	*	Float	4	40	-40	150	43109
20	T4 max start fan	°C or °F	*	*	Float	4	40	-40	150	43111
24	T5 max start fan	°C or °F	*	*	Float	4	40	-40	150	43113
28	T6 max start fan	°C or °F	*	*	Float	4	40	-40	150	43115
32	T7 max start fan	°C or °F	*	*	Float	4	40	-40	150	43117
36	T8 max start fan	°C or °F	*	*	Float	4	40	-40	150	43119
40	THDV max warning	%	*	*	Float	4	10	0.5	300	43121
44	Irms max warning	A	*	*	Float	4	1	0.1	9e6	43123
48	NOT USED		*	*	Byte	1	0			43125
49	V min Enable		*	*	Byte	1	0	0	1	43126
50	V max Enable		*	*	Byte	1	0	0	1	43127
51	T1 max Enable.		*	*	Byte	1	0	0	1	43128
52	T2 max Enable.		*	*	Byte	1	0	0	1	43129
53	T3 max Enable.		*	*	Byte	1	0	0	1	43130
54	T4 max Enable.		*	*	Byte	1	0	0	1	43131
55	T5 max Enable.		*	*	Byte	1	0	0	1	43132
56	T6 max Enable.		*	*	Byte	1	0	0	1	43133
57	T7 max Enable.		*	*	Byte	1	0	0	1	43134
58	T8 max Enable.		*	*	Byte	1	0	0	1	43135
59	THDV max Enable.		*	*	Byte	1	0	0	1	43136
60	Irms max Enable.		*	*	Byte	1	0	0	1	43137

5.5.3.5 Event Logging Settings (GroupID = 0x0805)

Byte Offset	Description	Units	RVT	RVT3P	Data type	Size in bytes	Default value	Min value	Max value	Modb @
0	Vrms Threshold	V	*	*	Float	4	1000	10	9e6	43201
4	Irms Threshold	A	*	*	Float	4	1000	0	9e6	43203
8	Total active power threshold	W	*	*	Float	4	1e6	0	1e9	43205
12	Line active power threshold	W	*	*	Float	4	1e6	0	1e9	43207
16	Total reactive power threshold	var	*	*	Float	4	1e6	0	1e9	43209
20	Line reactive power threshold	var	*	*	Float	4	1e6	0	1e9	43211
24	Total missing reactive power threshold	var	*	*	Float	4	1e6	0	1e9	43213
28	Line missing reactive power threshold	var	*	*	Float	4	1e6	0	1e9	43215
32	Total apparent	VA	*	*	Float	4	1e6	0	1e9	43217

	power threshold									
36	Line apparent power threshold	VA	*	*	Float	4	1e6	0	1e9	43219
40	THDV threshold	%	*	*	Float	4	30	0	1000	43221
44	THDI threshold	%	*	*	Float	4	100	0	1000	43223
48	Supplied Active energy threshold	Wh		*	Float	4	1e6	0	1e9	43225
52	Consumed Active energy threshold	Wh		*	Float	4	1e6	0	1e9	43227
56	Inductive reactive energy threshold	varh		*	Float	4	1e6	0	1e9	43229
60	Capacitive reactive energy threshold	varh		*	Float	4	1e6	0	1e9	43231
64	Apparent energy threshold	VAh		*	Float	4	1e6	0	1e9	43233
68	frequency max threshold	Hz	*	*	Float	4	65	40	70	43235
72	frequency min threshold	Hz	*	*	Float	4	45	40	70	43237
76	temperature T1 threshold max	°C or °F	*	*	Float	4	150	-40	150	43239
80	temperature T1 threshold min	°C or °F	*	*	Float	4	-40	-40	150	43241
84	temperature T2 threshold max	°C or °F	*	*	Float	4	150	-40	150	43243
88	temperature T2 threshold min	°C or °F	*	*	Float	4	-40	-40	150	43245
92	temperature T3 threshold max	°C or °F	*	*	Float	4	150	-40	150	43247
96	temperature T3 threshold min	°C or °F	*	*	Float	4	-40	-40	150	43249
100	temperature T4 threshold max	°C or °F	*	*	Float	4	150	-40	150	43251
104	temperature T4 threshold min	°C or °F	*	*	Float	4	-40	-40	150	43253
108	temperature T5 threshold max	°C or °F	*	*	Float	4	150	-40	150	43255
112	temperature T5 threshold min	°C or °F	*	*	Float	4	-40	-40	150	43257
116	temperature T6 threshold max	°C or °F	*	*	Float	4	150	-40	150	43259
120	temperature T6 threshold min	°C or °F	*	*	Float	4	-40	-40	150	43261
124	temperature T7 threshold max	°C or °F	*	*	Float	4	150	-40	150	43263
128	temperature T7 threshold min	°C or °F	*	*	Float	4	-40	-40	150	43265
132	temperature T8 threshold max	°C or °F	*	*	Float	4	150	-40	150	43267
136	temperature T8 threshold min	°C or °F	*	*	Float	4	-40	-40	150	43269

5.5.3.6 Installation Settings 1 (GroupID = 0x0806)

Byte Offset	Description	Units	RVT	RVT3P	Data type	Size in bytes	Default value	Min value	Max value	Modb @
0	CT scaling		*	*	Float	4	1	0.1	10000	43301
4	NOT USED			*	Float	4	1			43303
8	NOT USED			*	Float	4	1			43305

5.5.3.7 User Settings (GroupID = 0x0807)

Byte Offset	Description	Units	RVT	RVT3P	Data type	Size in bytes	Default value	Min value	Max value	Modb @
0	Target cos j		*	*	Float	4	1	0.7	1.3	43401
4	Night cos j		*	*	Float	4	1	0.7	1.3	43403
8	Regenerative cos j		*	*	Float	4	-1	-1.3	-0.7	43405
12	Night Reg cos j		*	*	Float	4	-1	-1.3	-0.7	43407
16	alarm delay	Sec	*	*	Dword	4	360	1	64800	43409
20	alarm reset delay	Sec	*	*	Dword	4	1	1	64800	43411
24	alarm cos j		*	*	Float	4	1	0.7	1.3	43413
28	Night cos j Enable		*	*	Byte	1	0	0	1	43415
29	Regenerative cos j Enable		*	*	Byte	1	0	0	1	43416
30	NOT USED		*	*	Byte	1	0			43417

The Power factor or cos j format is defined as follows:

P >= 0:	0.0 to 0.5 to 1.0 inductive	value = 0.0 …0.5… 1.0
	1.0 to 0.5 to 0.0 capacitive	value = 1.0 …1.5… 2.0
P < 0:	-0.0 to -0.5 to -1.0 inductive	value = -0.0 …-0.5… -1.0
	-1.0 to -0.5 to -0.0 capacitive	value = -1.0 …-1.5… -2.0

Positive values are for passive loads. Negative values represent regenerative mode.

5.5.3.8 ID (GroupID = 0x0808)

Byte Offset	Description	Units	RVT	RVT3P	Data type	Size in bytes	Default value	Min value	Max value	Modb @
0	Serial number low		*	*	Dword	4		0	0xFFFFFFFF	43501
4	Serial number high		*	*	Dword	4		0	0xFFFFFFFF	43503
8	Article number		*	*	Dword	4		0	0xFFFFFFFF	43505
12	RVT relay number		*	*	Byte	1		6	12	43507
13	RVT model (RVT/RVT3P)		*	*	Byte	1		0	1	43508
14	ABB IDnr1		*	*	Byte	1		0	0xFF	43509
15	ABB IDnr2		*	*	Word	2		0	0xFFFF	43510
17	ABB IDnr3		*	*	Dword	4		0	0xFFFFFFFF	43511
21	Soft Version		*	*	Dword	4		0	0xFFFFFFFF	43513
25	Internal Temperature protection		*	*	Float	4	85	0	120	43515
29	Product ID 0		*	*	Word	2		0	0xFFFF	43517
31	Product ID 1		*	*	Word	2		0	0xFFFF	43518
33	Product ID 2		*	*	Word	2		0	0xFFFF	43519
35	Product ID 3		*	*	Word	2		0	0xFFFF	43520
37	Product ID 4		*	*	Word	2		0	0xFFFF	43521
39	Product ID 5		*	*	Word	2		0	0xFFFF	43522
41	Product ID 6		*	*	Word	2		0	0xFFFF	43523
43	Product ID 7		*	*	Word	2		0	0xFFFF	43524
45	Product ID 8		*	*	Word	2		0	0xFFFF	43525
47	Product ID 9		*	*	Word	2		0	0xFFFF	43526
49	Product ID 10		*	*	Word	2		0	0xFFFF	43527
51	Product Type 0		*	*	Word	2		0	0xFFFF	43528
53	Product Type 1		*	*	Word	2		0	0xFFFF	43529
55	Product Type 2		*	*	Word	2		0	0xFFFF	43530

This group ID is a read-only group of parameters.

The ‘RVT model’ parameter is defined as follows:

Value	Description
0	RVT version
1	RVT 3-phase version

5.5.3.9 Status information (GroupID = 0x0809)

Byte Offset	Description	Units	RVT	RVT3P	Data type	Size in bytes	Default value	Min value	Max value	Modb @
0	Communication lock status		*	*	Byte	1	0	0	1	32501
1	NOT USED		*	*	Byte	1	0			32502
2	NOT USED		*	*	Byte	1	0			32503
3	NOT USED		*	*	Byte	1	0			32504
4	NOT USED		*	*	Byte	1	0			32505
5	NOT USED		*	*	Byte	1	0			32506
6	NOT USED		*	*	Byte	1	0			32507
7	NOT USED		*	*	Byte	1	0			32508
8	NOT USED		*	*	Byte	1	0			32509

The ‘Communication lock status’ parameter is defined as follows:

Value	Description
0	Unlocked
1	Locked

5.5.4 Measurement

5.5.4.1 Voltage (GroupID = 0x1000)

The data provided in this table depends on the connection type of the voltage measurement inputs to the RVT. Non connected inputs will give not applicable (n.a.) results. Please refer to Appendix A7 of the RVT Manual for more information.

Connection type: 1Ph-1LL1, 3Ph-1LL1, and 3Ph-1LL3

Byte Offset	Description	Units	RVT	RVT3P	Data type	Size in bytes	Min value	Max value	Modb @
0	n.a.								30501
4	RMS voltage L-L	V	*	*	Float	4	10	9,00E+06	30503
8	n.a.								30505
12	n.a.								30507
16	n.a.								30509
20	n.a.								30511
24	n.a.								30513
28	n.a.								30515
32	n.a.								30517
36	Fundamental voltage L-L	V	*	*	Float	4	10	9,00E+06	30519
40	n.a.								30521
44	n.a.								30523
48	n.a.								30525
52	n.a.								30527
56	n.a.								30529
60	n.a.								30531
64	n.a.								30533
68	Voltage THD L-L	%	*	*	Float	4	0	1000	30535
72	n.a.								30537
76	n.a.								30539
80	n.a.								30541
84	n.a.								30543
88	n.a.								30545
92	n.a.								30547
96	Frequency	Hz	*	*	Float	4	45	75	30549
100	n.a.								30551

Connection type: 3Ph-1LN1

Byte Offset	Description	Units	RVT	RVT3P	Data type	Size in bytes	Min value	Max value	Modb @
0	n.a.								30501
4	RMS voltage L-N	V	*	*	Float	4	10	9,00E+06	30503
8	n.a.								30505
12	n.a.								30507
16	n.a.								30509
20	n.a.								30511
24	n.a.								30513
28	n.a.								30515
32	n.a.								30517
36	Fundamental voltage L-N	V	*	*	Float	4	10	9,00E+06	30519
40	n.a.								30521
44	n.a.								30523
48	n.a.								30525
52	n.a.								30527
56	n.a.								30529
60	n.a.								30531
64	n.a.								30533
68	Voltage THD L-N	%	*	*	Float	4	0	1000	30535
72	n.a.								30537
76	n.a.								30539
80	n.a.								30541
84	n.a.								30543
88	n.a.								30545
92	n.a.								30547
96	Frequency	Hz	*	*	Float	4	45	75	30549
100	n.a.								30551

Connection type: 3Ph-3LL3, 3Ph-3LL2, and 3Ph-3LN3

Byte Offset	Description	Units	RVT	RVT3P	Data type	Size in bytes	Min value	Max value	Modb @
0	RMS voltage L1-L2	V		*	Float	4	10	9e6	30501
4	RMS voltage L2-L3	V		*	Float	4	10	9e6	30503
8	RMS voltage L3-L1	V		*	Float	4	10	9e6	30505
12	Total RMS voltage L-L	V		*	Float	4	10	9e6	30507
16	RMS voltage L1-N	V		*	Float	4	10	9e6	30509
20	RMS voltage L2-N	V		*	Float	4	10	9e6	30511
24	RMS voltage L3-N	V		*	Float	4	10	9e6	30513
28	Total RMS voltage L-N	V		*	Float	4	10	9e6	30515
32	Fundamental voltage L1-L2	V		*	Float	4	10	9e6	30517
36	Fundamental voltage L2-L3	V		*	Float	4	10	9e6	30519
40	Fundamental voltage L3-L1	V		*	Float	4	10	9e6	30521
44	Total Fundamental voltage L-L	V		*	Float	4	10	9e6	30523
48	Fundamental voltage L1-N	V		*	Float	4	10	9e6	30525
52	Fundamental voltage L2-N	V		*	Float	4	10	9e6	30527
56	Fundamental voltage L3-N	V		*	Float	4	10	9e6	30529
60	Total Fundamental voltage L-N	V		*	Float	4	10	9e6	30531
64	Voltage THD L1-L2	%		*	Float	4	0	1000	30533
68	Voltage THD L2-L3	%		*	Float	4	0	1000	30535
72	Voltage THD L3-L1	%		*	Float	4	0	1000	30537
76	Total THD L-L	%		*	Float	4	0	1000	30539
80	Voltage THD L1-N	%		*	Float	4	0	1000	30541
84	Voltage THD L2-N	%		*	Float	4	0	1000	30543
88	Voltage THD L3-N	%		*	Float	4	0	1000	30545
92	Total THD L-N	%		*	Float	4	0	1000	30547
96	Frequency	Hz		*	Float	4	45	75	30549
100	Voltage imbalance	%		*	Float	4	0	300	30551

Connection type: 3Ph-1LN3

Byte Offset	Description	Units	RVT	RVT3P	Data type	Size in bytes	Min value	Max value	Modb @
0	n.a.								30501
4	n.a.								30503
8	n.a.								30505
12	n.a.								30507
16	n.a.								30509
20	RMS voltage L-N	V		*	Float	4	10	9,00E+06	30511
24	n.a.								30513
28	n.a.								30515
32	n.a.								30517
36	n.a.								30519
40	n.a.								30521
44	n.a.								30523
48	n.a.								30525
52	Fundamental voltage L-N	V		*	Float	4	10	9,00E+06	30527
56	n.a.								30529
60	n.a.								30531
64	n.a.								30533
68	n.a.								30535
72	n.a.								30537
76	n.a.								30539
80	n.a.								30541
84	Voltage THD L-N	%		*	Float	4	0	1000	30543
88	n.a.								30545
92	n.a.								30547
96	Frequency	Hz		*	Float	4	45	75	30549
100	n.a.								30551

5.5.4.2 Line Currents (GroupID = 0x1001)

Byte Offset	Description	Units	RVT	RVT3P	Data type	Size in bytes	Min value	Max value	Modb @
0	RMS line current L1	A	*	*	Float	4	0	9e6	30601
4	RMS line current L2	A		*	Float	4	0	9e6	30603
8	RMS line current L3	A		*	Float	4	0	9e6	30605
12	Total RMS line current	A		*	Float	4	0	9e6	30607
16	RMS neutral current	A		*	Float	4	0	9e6	30609
20	Fundamental line current L1	A	*	*	Float	4	0	9e6	30611
24	Fundamental line current L2	A		*	Float	4	0	9e6	30613
28	Fundamental line current L3	A		*	Float	4	0	9e6	30615
32	Total Fundamental line current	A		*	Float	4	0	9e6	30617
36	Line current THD L1	%	*	*	Float	4	0	1000	30619
40	Line current THD L2	%		*	Float	4	0	1000	30621
44	Line current THD L3	%		*	Float	4	0	1000	30623
48	Total Line current THD	%		*	Float	4	0	1000	30625
52	Current imbalance	%		*	Float	4	0	300	30627

5.5.4.3 Temperatures (GroupID = 0x1002)

Byte Offset	Description	Units	RVT	RVT3P	Data type	Size in bytes	Min value	Max value	Modb @
0	Internal Temperature	°C or °F	*	*	Float	4	-40	150	30701
4	External temperature probe 1	°C or °F	*	*	Float	4	-40	150	30703
8	External temperature probe 2	°C or °F	*	*	Float	4	-40	150	30705
12	External temperature probe 3	°C or °F	*	*	Float	4	-40	150	30707
16	External temperature probe 4	°C or °F	*	*	Float	4	-40	150	30709
20	External temperature probe 5	°C or °F	*	*	Float	4	-40	150	30711
24	External temperature probe 6	°C or °F	*	*	Float	4	-40	150	30713
28	External temperature probe 7	°C or °F	*	*	Float	4	-40	150	30715
32	External temperature probe 8	°C or °F	*	*	Float	4	-40	150	30717

5.5.4.4 Powers (GroupID = 0x1003)

Byte Offset	Description	Units	RVT	RVT3P	Data type	Size in bytes	Min value	Max value	Modb @
0	Active power L1	W		*	Float	4	-1e9	1e9	30801
4	Active power L2	W		*	Float	4	-1e9	1e9	30803
8	Active power L2	W		*	Float	4	-1e9	1e9	30805
12	Total Active power	W	*	*	Float	4	-1e9	1e9	30807
16	Mean Active power over 15min	W		*	Float	4	-1e9	1e9	30809
20	Reactive power L1	var		*	Float	4	-1e9	1e9	30811
24	Reactive power L2	var		*	Float	4	-1e9	1e9	30813
28	Reactive power L3	var		*	Float	4	-1e9	1e9	30815
32	Total Reactive power	var	*	*	Float	4	-1e9	1e9	30817
36	Apparent power L1	VA		*	Float	4	-1e9	1e9	30819
40	Apparent power L2	VA		*	Float	4	-1e9	1e9	30821
44	Apparent power L3	VA		*	Float	4	-1e9	1e9	30823
48	Total Apparent power	VA	*	*	Float	4	-1e9	1e9	30825
52	Power factor L1			*	Float	4	-2	2	30827
56	Power factor L2			*	Float	4	-2	2	30829
60	Power factor L3			*	Float	4	-2	2	30831
64	Total Power factor		*	*	Float	4	-2	2	30833
68	Displacement power factor (cos φ) L1			*	Float	4	-2	2	30835
72	Displacement power factor (cos φ) L2			*	Float	4	-2	2	30837
76	Displacement power factor (cos φ) L3			*	Float	4	-2	2	30839
80	Total displacement power factor (cos φ)		*	*	Float	4	-2	2	30841
84	NOT USED			*	Float	4	-2	2	30843

See above the Power factor or cos j format (GroupID 0x0807)

5.5.4.5 PFC Control Data (GroupID = 0x1005)

Byte Offset	Description	Units	RVT	RVT3P	Data type	Size in bytes	Min value	Max value	Modb @
0	Missing reactive power L1	var		*	Float	4	0	1e9	30901
4	Missing reactive power L2	var		*	Float	4	0	1e9	30903
8	Missing reactive power L3	var		*	Float	4	0	1e9	30905
12	Missing reactive power	var	*	*	Float	4	0	1e9	30907
16	Missing steps L1	steps		*	Word	2	0	500	30909
18	Missing steps L2	steps		*	Word	2	0	500	30910
20	Missing steps L3	steps		*	Word	2	0	500	30911
22	Missing steps	steps	*	*	Word	2	0	500	30912

5.5.4.6 Status information (GroupID = 0x1006)

Byte Offset	Description	Units	RVT	RVT3P	Data type	Size in bytes	Min value	Max value	Modb @
0	Relay status		*	*	Word	2	0	0x3FFF	31001
2	External input status		*	*	Byte	1	0	3	31002
3	Temperature status		*	*	Byte	1	0	1	31003
4	Alarm status		*	*	Byte	1	0	1	31004
5	NOT USED		*	*	Byte	1	0	0XFF	31005
6	NOT USED		*	*	Byte	1	0	0xFF	31006
7	NOT USED		*	*	Byte	1	0	0xFF	31007
8	NOT USED		*	*	Byte	1	0	0xFF	31008
9	NOT USED		*	*	Byte	1	0	0xFF	31009
10	NOT USED		*	*	Byte	1	0	0xFF	31010
11	NOT USED		*	*	Byte	1	0	0xFF	31011
12	NOT USED		*	*	Byte	1	0	0xFF	31012
13	NOT USED		*	*	Byte	1	0	0xFF	31013
14	NOT USED		*	*	Byte	1	0	0xFF	31014
15	NOT USED		*	*	Byte	1	0	0xFF	31015
16	NOT USED		*	*	Byte	1	0	0xFF	31016
17	NOT USED		*	*	Byte	1	0	0xFF	31017
18	NOT USED		*	*	Byte	1	0	0xFF	31018
19	NOT USED		*	*	Byte	1	0	0xFF	31019
20	NOT USED		*	*	Dword	4	0	0xFFFFFFFF	31020
24	NOT USED		*	*	Dword	4	0	0xFFFFFFFF	31022
28	NOT USED		*	*	Dword	4	0	0xFFFFFFFF	31024

The ‘Relay Status’ parameter is defined as follows:

Value	Description
0	Relay open
1	Relay closed

Bit used	Output relay
Bit 0	1
Bit 1	2
Bit 2	3
Bit 3	4
Bit 4	5
Bit 5	6
Bit 6	7
Bit 7	8
Bit 8	9
Bit 9	10
Bit 10	11
Bit 11	12

Bit 12	Alarm
Bit 13	Fan
Bit 14	Not used

The ‘External input status’ parameter is defined as follows:

Value	Description
0	External input reset
1	External input set

Bit used	External Input Number
Bit 0	1
Bit 1	2
Bit 2	Not used
Bit 3	Not used
Bit 4	Not used
Bit 5	Not used
Bit 6	Not used
Bit 7	Not used

The ‘Temperature status’ parameter is defined as follows:

Value	Description
0	Reset
1	Set

The ‘Alarm status’ parameter is defined as follows:

Value	Description
0	Reset
1	Set

5.5.5 Info – Universal

5.5.5.1 Ethernet current configuration (GroupID = 0x2001)

Byte Offset	Description	Units	RVT	RVT3P	Data type	Size in bytes	Min value	Max value	Modb @
0	Current IP address			*	Dword	4	0	0xFFFFFFFF	32601
4	Current IP mask			*	Dword	4	0	0xFFFFFFFF	32603
8	Current Gateway IP address			*	Dword	4	0	0xFFFFFFFF	32605
12	NOT USED			*	Dword	4	0	0xFFFFFFFF	32607
16	NOT USED			*	Dword	4	0	0xFFFFFFFF	32609

These variables are providing information about the current network configuration.

When DHCP is disabled, those values will be the same as the one from the Ethernet Data parameter.

When DHCP is enabled, those values will be different than the static ones provided in the Ethernet Data parameter.

5.5.5.2 LED control (GroupID = 0x2081)

Byte Offset	Description	Units	RVT	RVT3P	Data type	Size in bytes	Min value	Max value	Modb @
0	Led frequency Status		*	*	Byte	1	0	4	31101

The 'LED frequency status' parameter is defined as follows:

Value	Blinking rate
0	Always Off
1	0.5 second
2	1 second
3	2 seconds
4	Always On

5.6 Source IDs

Here are the different curves available on the RVT and their associated ID:

Byte Offset	Description	Units	RVT	RVT3P	Data type	Size in bytes	# elts	Min value	Max value
0	Voltage Spectrum L1-L2	V		*	Float	4	49	-9e6	9e6
1	Voltage Spectrum L2-L3	V	*	*	Float	4	49	-9e6	9e6
2	Voltage Spectrum L3-L1	V		*	Float	4	49	-9e6	9e6
3	Current Spectrum L1	A	*	*	Float	4	49	-9e6	9e6
4	Current Spectrum L2	A		*	Float	4	49	-9e6	9e6
5	Current Spectrum L3	A		*	Float	4	49	-9e6	9e6
6	Current Spectrum N	A		*	Float	4	49	-9e6	9e6
7	Voltage Spectrum L1-L2	%V1		*	Word	2	49	0	300
8	Voltage Spectrum L2-L3	%V1	*	*	Word	2	49	0	300
9	Voltage Spectrum L3-L1	%V1		*	Word	2	49	0	300
10	Current Spectrum L1	%A1	*	*	Word	2	49	0	300
11	Current Spectrum L2	%A1		*	Word	2	49	0	300
12	Current Spectrum L3	%A1		*	Word	2	49	0	300
13	Current Spectrum Neutral	%A1		*	Word	2	49	0	300
14	Samples L1-L2	V		*	Float	4	128	-9e6	9e6
15	Samples L2-L3	V	*	*	Float	4	128	-9e6	9e6
16	Samples L3-L1	V		*	Float	4	128	-9e6	9e6
17	Samples L1-N	V		*	Float	4	128	-9e6	9e6
18	Samples L2-N	V		*	Float	4	128	-9e6	9e6
19	Samples L3-N	V		*	Float	4	128	-9e6	9e6
20	Samples I1	A	*	*	Float	4	128	-9e6	9e6
21	Samples I2	A		*	Float	4	128	-9e6	9e6
22	Samples I3	A		*	Float	4	128	-9e6	9e6
23	Samples INeutral	A		*	Float	4	128	-9e6	9e6

These IDs will have to be used in the TCP/IP server/client command requesting to gather some curve information.

6 Windows Communication DLL for PQ-Link protocol

6.1 Introduction

This document describes the interface of the Windows Communication DLL.

The interface is heavily based on the protocol described Chapter 4 of this document.

This document will focus on the differences with the protocol. Indeed, to ease the life of DLL user, some code has been added to handle annoying parts of the protocol.

Moreover, to be compatible with Visual Basic 6.0, some types have had to be modified as VB does not support certain types (unsigned 16-bits integer for example).

The DLL allows communicating with the RVT through:

- a TCP/IP network connection
- the RVT USB serial interface

6.2 Interface

6.2.1 Introduction

All exported library functions follow some similar syntax:

```
COMMANDCLIENTDLL_API int __stdcall CommandClient_Fct(…);
```

The first item is defined as follows:

```
#ifdef COMMANDCLIENTDLL_EXPORTS
#define COMMANDCLIENTDLL_API __declspec(dllexport)
#else
#define COMMANDCLIENTDLL_API __declspec(dllimport)
#endif
```

The “COMMANDCLIENTDLL_EXPORTS” define is used within the DLL to make sure that the functions get exported. When the DLL is used in an external program, it is not defined and the functions are then imported from the DLL.

All functions return an integer providing an error code to the calling layer.

The “__stdcall” specifier is used to indicate that the calling convention to be used is the standard one.

Special care has been taken so that the function names exported by the DLL are the same as the ones defined above.

WARNING: in the rest of this chapter, “COMMANDCLIENTDLL_EXPORTS” and “__stdcall” have been removed from the documentation to ease the reading. They are naturally always present in the header file.

6.2.2 Opening and Closing

6.2.2.1 CommandClient_Init

This function opens the connection to the RVT.

It has the following prototype:

```
int CommandClient_Init(char *RVTAddress,
                       int   ConnectionType,
                       void ( __stdcall *Callback_ConnectionReset)(void));
```

The RVTAddress parameter is an IP address in the case of a TCP/IP connection e.g. “192.168.1.40” or a COM port in the case of a serial connection e.g. “COM11”.

The ConnectionType can take two values:

- 0 or CONNECTION_TYPE_TCPIP for a TCP/IP connection
- 1 or CONNECTION_TYPE_SERIAL for a serial connection

The Callback_ConnectionReset is a pointer to a function that will be called if a loss of connection with the RVT is detected.

If one does not wish to use the callback mechanism, this parameter can be set to NULL. Unwanted disconnections can then be detected when any of the DLL function call returns the “RVT_SYS_SOCKET_DISCONNECTED” error code.

When a disconnection is happening, it is necessary to call the clean function to free the PC resources used and try to connect to the RVT again.

6.2.2.2 CommandClient_Clean

This function closes the connection to the RVT.

It has the following prototype:

```
int CommandClient_Clean();
```

The closing allows to free resources allocated both on the PC side and on the RVT side.

6.2.3 Authentication

Please note that the user account covered here are applicative user account handled by the RVT.

Once the TCP/IP, USB or Modbus connection has been established to the server, the client must authenticate itself.

As the system can be accessed remotely, some basic authentication is put in place. This allows distinguishing users and granting them certain rights to do things.

This authentication is mentioned as the first point of the command layer for the very important reason that no command will be accepted before an authentication is performed. The only exception is for the local administrator user that is automatically detected based on its local connection to the server.

6.2.3.1 CommandClient_Authenticate

This function allows authenticating as a given user on the RVT.

It has the following prototype:

```
int CommandClient_Authenticate(char          *Login,
                               char          *Password,
                               unsigned char *AccessLevel);
```

The password is here given as a string and transformed by the DLL to be used in the protocol.

The Login and Password parameters are used to authenticate oneself and the AccessLevel is returned to indicate the associated access level.

§ **Access levels**

The following access levels are defined:

Level	Name	Description
0x00	Monitor	Very basic user allowing only reading measurements and curves from the device.
0x10	Configurator	More advanced user allowed modifying measurement parameters.
0x40	Administrator	User allowed modifying any parameter of the device and to create and delete users.
0x80	Local application	Special user created for internal use only. It has the same rights as an Administrator.
0xFF	Reserved	Reserved.

6.2.3.2 CommandClient_CreateUser

This function allows creating a new user account on the RVT.

It has the following prototype:

```
int CommandClient_CreateUser(char          *Login,
                             char          *Password,
                             unsigned char AccessLevel);
```

The password is here given as a string and transformed by the DLL to be used in the protocol.

The AccessLevel parameter specifies what access level to associate with the new user account.

6.2.3.3 CommandClient_DeleteUser

This function allows deleting a user account on the RVT.

It has the following prototype:

```
int CommandClient_DeleteUser(char *Login);
```

6.2.4 Parameter access

6.2.4.1 CommandClient_GetParameter

This function allows getting a parameter group from the RVT.

It is restricted to configuration, measurement and information parameters.

The parameter is read from the actual parameters memory.

Note that the parameters that have been set in the shadow memory but not yet applied are not returned when a “Get Parameter” command is issued.

For a reason of transmission efficiency, it is only possible to transfer parameter by groups of parameters; in other words, transmission of individual parameters is not foreseen.

One must have at least Monitor rights to perform this task.

It has the following prototype:

```
int CommandClient_GetParameter(int           ParamGroupID,
                               unsigned char *Value,
                               int           *Size);
```

The ParamGroupID is specifying which parameter group to get.

The Value and Size parameters are pointers to the buffer where the parameter group values will be stored and to the actual size of this parameter group.

6.2.4.2 CommandClient_SetParameter

This function allows setting a parameter group in the RVT.

It is restricted to configuration parameters.

The modification of application specific parameters is applied in a shadow memory. Parameter changes will only be copied to parameters memory after an “Apply Parameter Changes” command is issued.

The modification of universal parameters is applied directly i.e. no “Apply Parameter Changes” command is required.

For a reason of transmission efficiency, it is only possible to transfer parameter by groups of parameters; in other words, transmission of individual parameters is not foreseen.

One must have at least Configurator rights to set application specific parameters and at least administrator rights to set universal parameters.

It has the following prototype:

```
int CommandClient_SetParameter(int            ParamGroupID,
                               unsigned char *Value,
                               int            Size);
```

The ParamGroupID is specifying which parameter group to set.

The Value parameter is a pointer to the buffer where the data to write is stored and the Size is specifying the amount of data to be written.

6.2.4.3 CommandClient_ApplyParameterChanges

This function applies the shadowed parameter changes in the parameter memory of the RVT.

One must have at least Monitor rights to perform this task.

It has the following prototype:

```
int CommandClient_ApplyParameterChanges();
```

6.2.4.4 CommandClient_ConvertRVTtoVB

This helper function allows retrieving a single parameter value from a parameter group.

It has the following prototype:

```
int CommandClient_ConvertRVTtoVB(unsigned char *ParamGroupValue,
                                 int            Offset,
                                 unsigned char  ValueType,
                                 char          *OutputString,
                                 int           *OutputStringSize);
```

The ParamGroupValue are the parameter group value as returned by the CommandClient_GetParameter command. This is a buffer of unsigned char values.

The Offset and ValueType are specifying where to find the wanted data and what is the type of the wanted value.

The OutputString and OutputStringSize are specifying where the string should be placed and what its size is.

One should make sure that there’s enough space in the output buffer for the string. Currently, the biggest parameters defined for the RVT are 180 bytes big and are of the string type. Numbers are converted into much smaller strings.

6.2.4.5 CommandClient_ConvertVBtoRVT

This helper function allows setting a single parameter value into a parameter group.

It has the following prototype:

```
int CommandClient_ConvertVBtoRVT(char          *InputString,
                                 unsigned char *ParamGroupValue,
                                 int           Offset,
                                 unsigned char  ValueType);
```

The InputString is the value to be converted and stored in the parameter group.

The ParamGroupValue is the parameter group value as returned by the CommandClient_GetParameter command. This is a buffer of unsigned char values.

The Offset is specifying where to store the converted data and the ValueType is specifying what type of data should be written to the parameter group.

One should make sure that the Offset and ValueType will not cause writing out of the ParamGroupValue buffer.

6.2.5 Curve access

6.2.5.1 CommandClient_RequestCurveEx (future use)

This function makes a request to get a curve available.

Wanted curve is identified by their Source ID.

One must have at least Monitor rights to perform this task.

The curves will be too big to be retrieved with a single command. As a consequence, the data will be split in chunk. The server is specifying the size of those chunks and is warning the client of how many of those chunks will need to be transferred to get the complete curve.

It has the following prototype:

```
int CommandClient_RequestCurveEx(int            SourceID,
                                 unsigned char  NumberOfPeriods,
                                 unsigned char  TriggerEnable,
                                 int            TriggerSource_ID,
                                 int            TriggerSource_Offset,
                                 unsigned char  TriggerComparator,
                                 unsigned char  TriggerValue_Type,
                                 char          *TriggerValue_ValueString,
                                 int           *CurveDescriptor);
```

The CurveDescriptor is made out of the NumberOfChunks and ChunkSize as defined in the protocol.

The Trigger Value to be used is here provided as a string and is converted by the DLL to the type specified by TriggerValue_Type.

It can take some time between the moment of the request and the time where the first data gets available.

IMPORTANT NOTE:

- Number of periods is “1” by default
- Trigger function is reserved for future use : “0” by default

6.2.5.2 CommandClient_RequestCurve

This is a simplified version of the CommandClient_RequestCurveEx command where triggering is disabled.

It has the following prototype:

```
int CommandClient_RequestCurve(int            SourceID,
                               unsigned char  NumberOfPeriods,
                               int           *CurveDescriptor);
```

6.2.5.3 CommandClient_ReleaseCurve

This function is telling the RVT to stop getting data for the given curve.

If the curve was requested by a single client, the curve is not made available anymore for the client and the slot is freed.

If the curve was requested by multiple clients, the freeing is only made when the last client releases the curve.

Note that the effect of this command may not be immediate.

One must have at least Monitor rights to perform this task.

It has the following prototype:

```
int CommandClient_ReleaseCurve(int SourceID);
```

6.2.5.4 CommandClient_GetCurve

This function is getting curve data from the RVT.

It has the following prototype:

```
int CommandClient_GetCurve(int            SourceID,
                           int            CurveDescriptor,
                           unsigned char *Value,
                           int           *Size);
```

The SourceID and CurveDescriptor are identifying the curve.

The Value is a buffer where the curve will be stored and the Size is the actual size of the curve returned. One should make sure that the buffer is big enough to contain the curve.

The curves are transferred by chunks.

This function is taking care of downloading the number of chunks required to get the complete curve. It is also taking care that all chunks returned do belong to the same curve set.

One must have at least Monitor rights to perform this task.

6.2.5.5 CommandClient_GetCurveIDs

This function is getting all the SourceIDs currently in use by the DLL.

One must have at least Monitor rights to perform this task.

It has the following prototype:

```
int CommandClient_GetCurveIDs(int *SourceIDs,
                              int *Size);
```

The Size is specifying how many SourceIDs have been returned and copied to the location pointed by the SourceIDs pointer. One must ensure that the location is big enough to contain the maximum number of curves allowed by the system.

6.2.5.6 CommandClient_GetCurveCharacteristics

This function is getting the characteristics of a given curve.

It has the following prototype:

```
int CommandClient_GetCurveCharacteristics(int            SourceID,
                                          unsigned char *NumberOfPeriods,
                                          unsigned char *TriggerEnable,
                                          int           *TriggerSource_ID,
                                          int           *TriggerSource_Offset,
                                          unsigned char *TriggerComparator,
                                          unsigned char *TriggerValue_Type,
                                          char  *TriggerValue_ValueString,
                                          int   *TriggerValue_ValueStringSize,
                                          int           *CurveDescriptor);
```

This function can be seen as a way to get back the parameters that were passed at the moment of requesting the curve.

This function along with the GetCurveIDs function allow for an easy re-populating of the user interface. The interface does not have to store the characteristics of all the curves it currently manages; it can just ask it back to the RVT.

One must have at least Monitor rights to perform this task.

6.2.6 Miscellaneous

§ Reset

This function allows to remotely restarting the RVT.

It has the following prototype:

```
int CommandClient_Reset();
```

After calling this command, the connection should be closed using the CommandClient_Clean function and should be re-established again.

One needs at least Configurator rights to perform this task.

6.3 Important considerations

6.3.1 Visual Basic 6.0 support

This DLL has been built with support for VB6.0 in mind.

6.3.2 Multi-threading

The DLL is not coded for multi-threaded application.

The first consequence is that all calls to the DLL should be called from a single thread. Calling from different thread could be possible but protection should then be implemented outside of the DLL.

The second consequence is that all calls to the DLL are blocking. In usual cases it is not a problem but when the connection gets lost for example, it could take a short amount of time to return from a called function.

The only alternative is to go for non-blocking behaviour but then the programming of the application will become more complex as a request issued would not have a direct answer with data to process but the answer would come at a later asynchronous stage.

6.3.3 Sequence of actions

The following sequence of action should be followed when using this DLL to communicate with a RVT:

1. Connect to the RVT
2. Authenticate on the RVT
3. Perform wanted actions (Get / Set parameters, Get curves…)
4. Go back to step 3. while the connection should be active
5. Disconnect from the RVT

One should not forget the authentication phase otherwise no subsequent action will be possible and the connection will be closed by the RVT. Moreover, the authentified user's access level will make it possible to perform certain actions or not.

Unwanted TCP/IP or serial disconnections with the RVT can be monitored through the callback provided at initialization time or through the returning o of the "RVT_SYS_SOCKET_DISCONNECTED" code. When that is happening, it is necessary to call the clean function and try to connect to the RVT again.

6.4 Error codes

The following error codes can be returned by the DLL:

Error code define	Value
RVT_SUCCESS	0
RVT_SYS_MEMORY	1
RVT_SYS_TASK_CREATE	2
RVT_SYS_SEMAPHORE_CREATE	3
RVT_SYS_SEMAPHORE_FAILURE	4
RVT_SYS_PERIPHERAL_IO	5
RVT_MEMORY_CORRUPT	6
RVT_SYS_SOCKET_OPEN	10
RVT_SYS_SOCKET_BIND	11
RVT_SYS_SOCKET_LISTEN	12
RVT_SYS_SOCKET_CONNECT	13
RVT_SYS_SOCKET_DISCONNECTED	14
RVT_SYS_INVALID_OBJECT	16
RVT_SYS_BUFFER_OVERFLOW	17
RVT_SYS_BUFFER_TOO_SMALL	18
RVT_NO_MORE_OBJECT_ALLOWED	20
RVT_OBJECT_NOT_FOUND	21
RVT_OBJECT_OPEN_ERROR	22
RVT_OBJECT_IO_ERROR	23
RVT_NO_MORE_OBJECT	24
RVT_OBJECT_DISABLED	25
RVT_OBJECT_ALREADY_USED	26
RVT_PKT_MALFUNCTION	29
RVT_SECL_UNKNOWN_LOGIN	30

RVT_SECL_INVALID_LOGIN	31
RVT_SECL_LOGIN_FAILURE	32
RVT_SECL_AUTH_FAILURE	33
RVT_SECL_TOO_MANY_USERS	34
RVT_SECL_NOT_LOGGED_IN	35
RVT_SECL_NOT_ENOUGH_RIGHTS	36
RVT_PMDB_INVALID_ID	40
RVT_PMDB_UNAVAILABLE	41
RVT_PMDB_IO_FAILURE	42
RVT_PMDB_INVALID_NVRAM	43
RVT_PMDB_TYPE_MISMATCH	44
RVT_PMDB_OPERATION_DENIED	45
RVT_PMDB_INVALID_SIZE	46
RVT_PMDB_INVALID_CONTENT	47
RVT_FCT_INVALID_PARAMETER	50
RVT_FCT_NEEDS_INITIALIZATION	51
RVT_FCT_OPERATION_FAILED	52
RVT_FCT_OPERATION_DENIED	53
RVT_FCT_OPERATION_TIMEDOUT	54
RVT_FCT_INVALID_RESPONSE	55
RVT_CMD_UNKNOWN	60
RVT_CMD_UNSUPPORTED	61
RVT_CMD_MISMATCH	62
RVT_LOG_SYSLOG_INVALID_ADDR	110
RVT_LOG_OPERATION_DENIED	111

6.5 Example codes

6.5.1 Visual Basic 6.0 project

The project is built around a single form for the user interface and a module to define the interface to the DLL.

The form code contains necessary initialization steps to make the DLL available from another directory. It obviously also contain the code for the user interface and associated calls to the DLL functions.

The module contains the necessary constant and function declarations. It also contains some helper functions to be used when using helper functions to convert parameter; these functions take care of memory allocation for the returned strings.

7 Appendices

A1 List of abbreviations

ASCII	American Standard Code for Information Interchange
Baud rate	Unit for measuring transmission speed in bits/s;
Bit	A binary digit, representing a one or zero
Bus	An electrical circuit over which data is transmitted
Byte	A whole number value represented by eight bits (0 to 255)
Chassis or Chassis Ground	A connection to an electrically conductive housing or frame of a device. It may or may not be connected to Earth Ground.
Coil	The telegram structure for Modbus transmission is implemented in registers (WORD) or coils (BOOL). A coil may be either 8 or 16 bits in length.
Common	The voltage reference point of a circuit. It may or may not be connected to earth ground, though it is generally assumed to be at zero volts, unless otherwise indicated. In floating circuits, the common is sometimes at a relatively high potential. This term is sometimes used interchangeably with the term “Ground” or GND
CRC	Cyclic Redundancy Check. Complex error checking on a message block.
CTS	ClearToSend hardware handshaking signal. Used with RequestToSend.
DHCP	is an autoconfiguration protocol used on IP networks. Computers that are connected to IP networks must be configured before they can communicate with other computers on the network. DHCP allows a computer to be configured automatically, eliminating the need for intervention by a network administrator
DLL	Dynamic-link library is Microsoft’s implementation of the shared library concept in the Microsoft Windows and OS/2 operating systems. These libraries usually have the file extension DLL, OCX (for libraries containing ActiveX controls), or DRV (for legacy system drivers). The file formats for DLLs are the same as for Windows EXE files — that is, Portable Executable (PE) for 32-bit and 64-bit Windows. As with EXEs, DLLs can contain code, data, and resources, in any combination.
Earth or Earth Ground	Global zero voltage reference point. Physical connection is made to the earth through a grounding rod, water pipe or other reliable connection.
Ethernet	is a family of frame-based computer networking technologies for local area networks (LANs). It defines a number of wiring and signalling standards for the Physical layer of the OSI networking model as well as a common addressing format and Media Access Control at Data Link Layer.
Ground Voltage reference point of a circuit.	It may or may not be connected to earth ground, though it is generally assumed to be at zero volts. Sometimes used interchangeably with the term “Common”.

Handshaking	method of data flow control for serial communications
Hexadecimal or HEX	A number system using a decimal 16 as its base. A single digit number in HEX ranges from 0 to 15, represented by 0 to 9 and A to F.
HMI	Human-Machine Interface (formerly MMI)
Industrial^{IT}	Umbrella concept for ABB's vision for enterprise automation.
Industrial^{IT} Architecture	The architecture of the Industrial IT system. The architecture defines how the system is built, in terms of basic concepts, underlying technologies, system topology, modularity, and mechanisms for interaction between different parts of the system. It also defines concepts, rules, and guidelines that a component must comply with in order to fit in the Industrial IT system. A central feature of the IIT architecture is that information and functions are centred on Aspect Objects.
Industrial^{IT} Enabled	A product that is Industrial IT enabled has been verified according to the process of Industrial IT certification. It has the right to use the "Industrial IT enabled" symbol.
IP address	An Internet Protocol (IP) address is a numerical label that is assigned to devices participating in a computer network that uses the Internet Protocol for communication between its nodes. An IP address serves two principal functions: host or network interface identification and location addressing.
Loopback	A test used for checking functionality of a serial port, utilizing a test plug that connects send, receive and handshaking signals
Long Integer	Analog value consisting of two consecutive 16-bit registers
LRC	Longitudinal Redundancy Check
MAC address	In computer networking, a Media Access Control address (MAC address) is a unique identifier assigned to most network adapters or network interface cards (NICs) by the manufacturer for identification, and used in the Media Access Control protocol sub-layer. If assigned by the manufacturer, a MAC address usually encodes the manufacturer's registered identification number.
Measurement	A measurement is a value computed by the controller through its analog and digital inputs. Measurements can be read from the RVT front plate, or through the Modbus protocol.
Modbus adapter	It is an optional small interface module through which the RVT is connected to an external Modbus serial communication bus. It performs an optical to RS485 conversion. The communication with the Modbus adapter is activated with an RVT parameter.
OPC	OLE™ for Process Control. OPC is Plug-n-Play in the field of Automation and HMI. OLE™ for Process Control (OPC™) is the most standard way for connecting hardware and data devices with HMI client applications. OPC is a concept agreed upon by a committee of members from the OPC foundation. Most automation companies in the market place including ABB are members of this foundation. OPC uses state-of-the art technologies like COM, DCOM, ActiveX of Microsoft and makes development and programming easier.

	In the OPC world, there are two major types of applications: OPC Servers and OPC Clients.
OPC Servers	OPC Server applications are used to collect data from the data sources like hardware devices. At the bottom level, the servers are mainly for reading inputs and writing outputs of the data sources. At the upper level, the servers make the data available in a standard way to the OPC client applications.
OPC Clients	The OPC Client applications can communicate directly with the OPC servers and get the data. This way OPC enhances the interface between client and server applications by providing a standard mechanism to communicate data from a data source to any client application.
Parameter	A parameter is an operating data for the controller. Parameters can be read and programmed with the RVT front plate, or through the Modbus protocol.
Parity	Simple method of data error checking performed at the byte level. May be user-specified as Odd, Even or None with most equipment and software.
PC	Personal Computer
Power^{IT} Power Factor Controller	
	Microprocessor based controller from the ABB industrial controller range. The Power^{IT} Power Factor Controller RVT is intended to switch capacitor in order to compensate the power factor of the electrical network.
Receive	Incoming communication signal. (Rx)
RTS	RequestToSend hardware handshaking signal. Used with ClearToSend.
RVT	see Power^{IT} Power Factor Controller RVT
Rx	See Receive
PLC	Programmable Logic Controller
RTS	Request To Send
RTU	Remote Terminal Unit
Time-out	Parameter specifying the max. wait time in ms. Waiting for a response in the range 0..10000 ms.
Signed Integer	Whole number value represented by 16 bits (-32768 to 32767)
SMTP	Simple Mail Transfer Protocol (SMTP) is an Internet standard for electronic mail (e-mail) transmission across Internet Protocol (IP) networks. SMTP is specified for outgoing mail transport.
	While electronic mail servers and other mail transfer agents use SMTP to send and receive mail messages, user-level client mail applications typically use only SMTP for sending messages to a mail server for relaying. For receiving messages, client applications usually use either the Post Office Protocol (POP) or the Internet Message Access Protocol (IMAP) or a proprietary system (such as Microsoft Exchange or Lotus Notes/Domino) to access their mail box accounts on a mail server.
SNMP	Simple Network Management Protocol (SNMP) is a UDP-based network protocol. It is used mostly in network management systems to monitor network-attached devices for conditions that warrant administrative attention. SNMP is a component of the Internet Protocol Suite. It consists of a set of

	standards for network management, including an application layer protocol, a database schema, and a set of data objects
TCP/IP	The Internet Protocol Suite is the set of communications protocols used for the Internet and other similar networks. It is commonly also known as TCP/IP, named from two of the most important protocols in it: the Transmission Control Protocol (TCP) and the Internet Protocol (IP), which were the first two networking protocols defined in this standard.
Transmit	Outgoing communication signal. (Tx)
Tri-State	The ability of a communications transmitter to turn its circuitry off, reducing the load on the network
Tx	see Transmit
Unsigned Integer	Positive whole number value represented by 16 bits (0 to 65535)
USB	Universal Serial Bus is a specification to establish communication between devices and a host controller (usually personal computers).
Word	A group of 16 bits
Xon/Xoff	Software implementation of data flow control

A2 References

- 2GCS212013A0050 - RS485 adapter-Installation and start-up guide.pdf
- 2GCS214013A0050 - RS485 adapter- User guide.pdf
- 2GCS215016A0050_RVT Manual EN.pdf
- 2GCS220012A0050_Quick start.pdf
- 2GCS221012A0050_ABB Power Quality Link.pdf
- Modicon Modbus Protocol Reference Guide (PI–MBUS–300 Rev. J).

A3 Description of open ports

Port	State
502/TCP	Open
4250/TCP	Open
10022/TCP	Open

A4 Cyber Security Disclaimer note

This product is designed to be connected to and to communicate information and data via a network interface. It is User's sole responsibility to provide and continuously ensure a secure connection between the product and User's network or any other network (as the case may be). The User shall establish and maintain any appropriate measures (such as but not limited to the installation of firewalls, application of authentication measures, encryption of data, installation of anti-virus programs, etc) to protect the product, the network, its system and the interface against any kind of security breaches, unauthorized access, interference, intrusion, leakage and/or theft of data or information.
ABB Ltd and its affiliates are not liable for damages and/or losses related to such security breaches, any unauthorized access, interference, intrusion, leakage and/or theft of data or information.
Please note that an ssh account exists for maintenance & development purposes.

Contact us

s.a. ABB n.v.
Power Quality Products
Allée Centrale 10
Zoning Industriel de Jumet
B-6040 Charleroi (Jumet), Belgium
Phone: +32(0) 71 250 811
Fax: +32 (0) 71 344 007
E-Mail: power.quality@be.abb.com

http://new.abb.com/high-voltage/capacitors/lv