Application Note - Using Modbus With Conext™ TL Inverters

976-0301-01-01
Revision A

Contents

Schneider Electric

⚠ DANGER

HAZARD OF FIRE, ARC FLASH, OR ELECTRIC SHOCK FROM MULTIPLE SOURCES

- This Application Note is in addition to, and incorporates by reference, the installation and operation manual for the Conext™ TL 15000 E and 20000 E photovoltaic grid tie inverters. Before reviewing this Application Note you must read the Conext TL installation and operation manual. Unless specified, information on safety, specifications, installation, and operation is as shown in the primary documentation received with the product. Ensure you are familiar with that information before proceeding.
- To be installed and serviced only by qualified personnel.
- This document is intended for use by qualified installers only.
- Before servicing, disconnect all sources and wait at least 1 minute.

Failure to follow these instructions will result in death or serious injury.

Introduction

Overview

Modbus is a simple and robust open communication protocol used to provide interoperability between products from many different vendors. The purpose of this application note is to provide a brief overview of the Modbus hardware and software implementation of the Conext TL 15000 E and Conext TL 20000 E photovoltaic grid tie inverters so that you can quickly and easily interface the inverter with any third-party Modbus devices.

The inverter performs Modbus communications according to the Modbus Application Protocol v1.1. It is assumed that you are familiar with the Modbus protocol and with serial communications in general.

Key Points

The inverter is capable of communicating via the RS-485 serial communication standard. The RS-485 medium allows for multiple devices on the same serial bus network.

All communications on the network conform to a Master/Slave scheme. In this scheme, information and data is transferred between a Modbus Master device and up to 254 Slave devices.

The Master device initiates and controls all information transfer on the Modbus serial bus network. There may be **only one master** for any Modbus network.

A Slave device never initiates a communication sequence, and must remain silent unless addressed specifically by the Master.

All communication activity on the Modbus serial bus network occurs in the form of packets. A packet is a serial string of up to 255 8-bit bytes.

All packets transmitted by the Master are requests. All packets transmitted by a Slave are responses.

At most, one Slave can respond to a single request from a Master.

The Modbus protocol supports RTU and ASCII protocols. The Conext TL 15000 E and Conext TL 20000 E photovoltaic grid tie inverters support only the Modbus/RTU protocol.

Related Documents

Table 1 Related documents

Document reference	Document title	Document number	Version
1	Modbus Application Protocol Specification	From www.modbus.org	1.1b
2	Conext TL 15000 E and 20000 E Installation and Operation Manual	975-0609-01-01	AA

Modbus Physical Layer

The Conext TL inverter supports the Modbus communication protocol via an RS485 interface. Both an RJ-45 and terminal block wiring interface are supported. The module also provides an RID (Remote Inverter Disable) input and a dry (not energized) contact for signalling purposes. The module is shown in Figure 2-1.

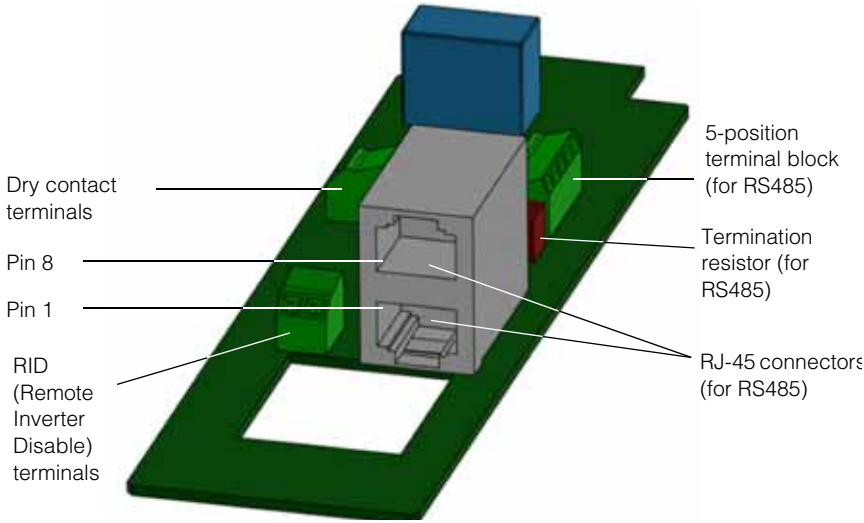


Figure 2-1 Communication module

RJ-45 Connection

The RS-485 bus is a multi-drop bus implemented as a daisy chain. The RJ-45 connector is provided with two ports to allow ease of daisy chaining. Either port can be connected to the upstream or downstream devices.

A standard Ethernet (straight-through) patch cable may be used to connect to the upstream and downstream devices. Ethernet cross-over cables **must not** be used.

The RJ45 connector provides D+, D-, and signal Ground connections.

The pin definitions of the RJ-45 connection are shown in Table 2-1. For the location of pin 8, see Figure 2-1 on page 2–5.

Table 2-1 RJ-45 pin definitions

Pin	Function
4	DATA+
5	DATA-
7	NC (Not connected)
8	Modbus ground

Terminal Block Connection

A five-position terminal block connector is also provided for Modbus interconnection. It provides the same D+, D-, and signal Ground connections that are provided on the RJ45 connector, and gives the user the flexibility to use a standard twisted pair cable to external equipment.

The pin definitions of the 5-position terminal block are shown in Table 2-2.

Table 2-2 5-position terminal block pin definitions

Pin	Function
1	DATA+
2	DATA-
3	Chassis ground
4	Modbus ground
5	NC (Not connected)

The location of the terminal block is shown in Figure 2-1 on page 2–5.

For pin numbering, see Figure 2-2.

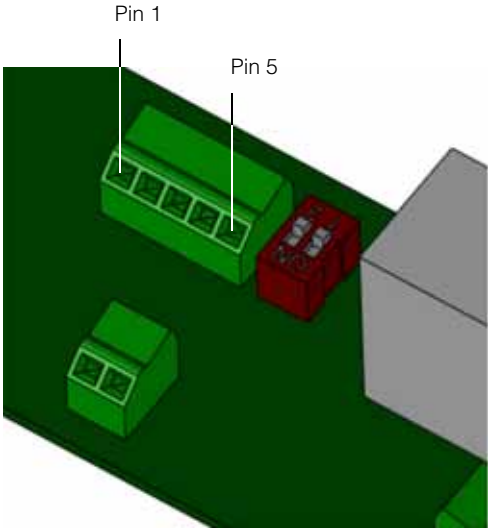


Figure 2-2 RS485 terminal block—pin numbering

Termination Resistor

You must enable the termination resistors if the inverter is on the end of the Modbus device chain. To do this, use a DIP switch on the communication interface board. If the inverter is the first or the last device of the RS485-chain, set the termination resistor to on; otherwise, set it to off.

The location of the termination resistor is shown in Figure 2-3. The settings are shown in Table 2-3.

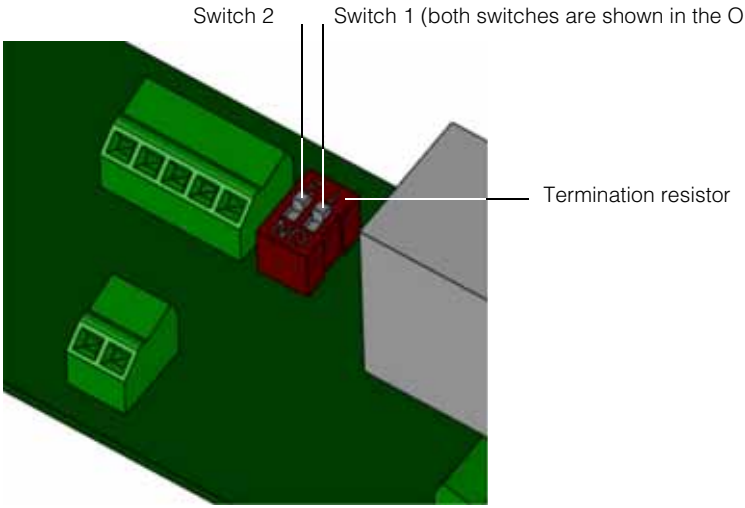


Figure 2-3 Termination resistor—switch numbering

Table 2-3 Termination resistor settings

Switch 1	Switch 2	Result
Off	Off	The termination resistor is off.
Off	On	The termination resistor is on.
On	Off	The termination resistor is on.
On	On	The termination resistor is on.

Communication parameters

Table 3 on page 9 shows the communication parameters used by the RS-485 Modbus interface on the inverter.

These parameters must be set identically on the Modbus Master device or PC program used to communicate with the inverter. To determine how to set the communication parameters of the Modbus Master device, see the documentation that accompanies the device.

Table 3 RS485 communication parameters

Parameter	Value		Parameter	Value
Baud rate	9600		Stop bits	1
Data bits	8		Parity	None

Inverter Configuration

Setting the Modbus Slave Address (Inverter ID)

The Modbus Slave address (or Inverter ID) must be unique for each device on the Modbus network. The Modbus Slave address may be read and/or modified via the front panel display of the inverter.

To change the Inverter ID:

1. From the main menu, select **Settings**, and then press OK.

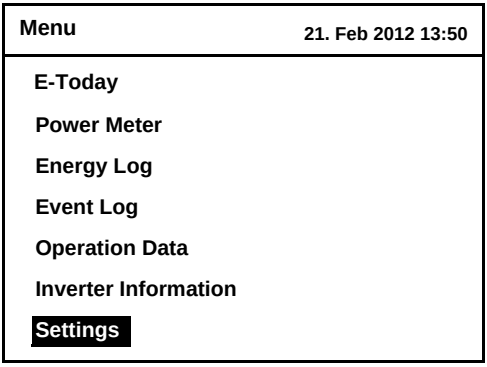


Figure 3 Main menu

2. Select **Install Settings**, and then press OK.

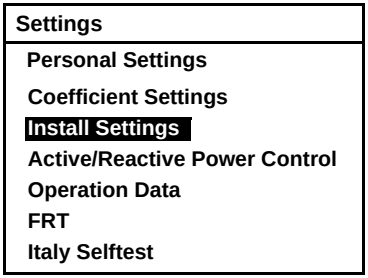


Figure 4 Settings menu

3. Use the ▼ and ▲ buttons to set each of the four digits in the password. Press OK after each digit is entered, and then press OK.
   The password for qualified installers is 5555.

Once the correct password has been entered, the Install Settings Menu is displayed, as shown in Figure 5.

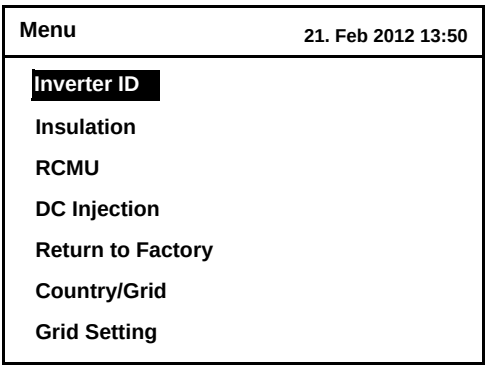


Figure 5 Install Settings menu

4. From the Install Settings Menu, select **Inverter ID**, and press OK.
5. Use the ▼ and ▲ buttons to set the Inverter ID. Press OK to confirm the new Inverter ID.
6. Press the ESC button to exit back to the main menu.

For the new Inverter ID setting to take effect, shut down and restart the inverter.

Reviewing the Modbus Slave Address (Inverter ID)

To review the current Modbus Slave Address (Inverter ID) setting:

1. From the main menu, select **Inverter Information**.
2. Press OK.
   The Inverter ID is shown at the bottom of the screen.

Modbus Logical Layer

Modbus Packet Structure

Every Modbus packet consists of four fields:

- Slave address field
- Function field
- Data field
- Error check field (checksum)

NOTICE
• The values shown in the packets are in hexadecimal format. • In the tables that show the packet structure, white background denotes the DATA field of the packet.

Table 4 Modbus packet structure

Address	Function Code	Data	Checksum

Slave address field

The slave address field of a Modbus packet is one byte in length and uniquely identifies the slave device involved in the transaction. Valid addresses range between 1 and 255.

A slave device performs the command specified in the packet when it receives a request packet with the slave address field matching its own address.

A response packet generated by the slave has the same value in the slave address field.

Function field

The function field of a Modbus request packet is one byte in length and tells the addressed slave which function to perform. Similarly, the function field of a response packet tells the master what function the addressed slave has just performed.

Data field

The data field of a Modbus request is of variable length, and depends on the function. This field contains information required by the slave device to perform the command specified in a request packet or data being passed back by the slave device in a response packet.

Data in this field is contained in 16-bit registers. Registers are transmitted in the order of high-order byte first, low-order byte second.

Example:

A 16-bit register contains the value 0x12AB. This register is transmitted:

- High order byte = 0x12
- Low order byte = 0xAB

This register is transmitted in the order 12 AB.

Error check field (checksum)

The checksum field lets the receiving device determine if a packet is corrupted with transmission errors. In Modbus RTU mode, a 16-bit Cyclic Redundancy Check (CRC-16) is used.

The sending device calculates a 16-bit value, based on every byte in the packet, using the CRC-16 algorithm. The calculated value is inserted in the error check field.

The receiving device performs the calculation, without the error check field, on the entire packet it receives. The resulting value is compared to the error check field. Transmission errors are indicated when the calculated checksum does not equal the checksum stored in the incoming packet. The receiving device ignores a bad packet.

For more information on CRC-16 calculations, see “Appendix A: CRC-16 calculation” on page 36.

Packet communications

This section describes the Modbus functions supported by the inverter.

Modbus functions supported by the inverter

Table 5 Modbus functions supported by the inverter

Function	Meaning	Action	See this section:
03	Read Holding Registers	Reads a value from one or more consecutive holding registers in the inverter.	“Function 03: Read Holding Registers” on page 13
16	Write Multiple Registers	Writes a value into one or more consecutive holding registers in the inverter.	“Function 16: Preset Multiple Registers” on page 14
43	Read Device Identifier	Reads the Manufacturer, Model, and Version information for the device.	“Function 43: Device Discovery” on page 16

Function 03: Read Holding Registers

To read inverter parameter values, a master must send the slave device (inverter) a Read Holding Registers request packet.

The Read Holding Registers request packet specifies a start register and a number of registers to read. (You can read 1 or more registers.) The start register may be from 0 to 65535 (0xFFFF).

The inverter responds with a packet containing the values of the registers in the range defined in the request.

Table 6 Read Holding Registers packet structure

Request packet (master to slave)	Response packet (slave to master)
Unit ID/slave address (1 byte)	Unit ID/slave address (1 byte)
03 (function code) (1 byte)	03 (function code) (1 byte)
Start register (sr) (2 bytes)	Byte count (2 x nr) (1 byte)
# of registers to read (nr) (2 bytes)	First register in range (2 bytes)
CRC checksum	Second register in range (2 bytes)
	…
	CRC checksum (2 bytes)

Example:

The inverter is configured as a Modbus slave device with slave address 1. The master requests to read all three voltage phases (A, B, C). These three parameters are made available in the Modbus map at address 0x17F8 (V_{ab}), 0x17F9 (V_{bc}), and 0x17FA (V_{ca}), with a scaling factor of 10. The request must read 3 registers starting at 0x17F8.

Table 7 Request packet

Slave	Function	Start register		# of registers (3)		CRC checksum	
01	03	17	F8	00	03	81	8E

Table 8 Response packet

Slave	Function	Byte count	Register 1		Register 2		Register 3		CRC checksum	
01	03	06	04	B1	04	AE	04	B3	7F	4F

The master retrieves the data from the response:

- Register 0x17F8: 0x04B1 = 1201 (scaled: 120.1)
- Register 0x17F9: 0x04AE = 1198 (scaled: 119.8)
- Register 0x17FA: 0x04B3 = 1203 (scaled: 120.3)

Function 16: Preset Multiple Registers

The Preset Multiple Registers command packet allows a Modbus master to configure or control the slave inverter.

A Preset Multiple Registers data-field request packet contains a definition of a range of registers to write to, and the values that are written to those registers.

The slave inverter responds with a packet indicating that a write was performed to the range of registers specified in the request.

The Preset Multiple Registers request and response packet formats are shown in the following example transaction.

Table 9 Preset Multiple Registers packet structure

Request packet (master to slave)	Response packet (slave to master)
Unit ID/slave address (1 byte)	Unit ID/slave address (1 byte)
16 (function code) (1byte)	16 (function code) (1 byte)
Start register (sr) (2 bytes)	Start register (sr) (2 bytes)
# of registers to write (nr) (2 bytes)	# of registers written (nr) (2 bytes)
Byte count (2 x nr) (1 byte)	CRC checksum (2 bytes)
First register in range (2 bytes)	
Second register in range (2 bytes)	
...	
CRC checksum (2 bytes)	

NOTE: Except for the function field, the Preset Registers Response packet has the same fields as the Read Registers Request packet.

Example:

Write to the “Logger-Mode” register of the inverter at Modbus address 201 (0xC9) to enable reading of event logs from the inverter. The Logger-mode register is at address 0x4705, and it must be set to “2” (0x0002) to enable reading of the event logs.

For details about the Logger-Mode register, see “Logger Mode” on page 23.

Table 10 Request packet

Slave	Function	Start register		# of registers	Byte count	Register 1		CRC checksum	
C9	10	47	05	01	02	00	02	7F	7A

Table 11 Response packet

Slave	Function	Start register		# of registers		CRC checksum	
C9	10	47	05	00	01	15	34

Function 43: Device Discovery

Function code 43 checks for the presence of a device at a specific address on the Modbus device chain. A Modbus master may request Function Code 43 data from each Modbus address. A device with the requested address must report at least 3 pieces of data, as shown in Table 12.

Table 12 Mandatory components of a reply to Function Code 43

Object ID	Object name / description	Type
0x00	Manufacturer Name	ASCII String
0x01	Product Identification	ASCII String
0x02	Product Version Number	ASCII String

In the case of the 20KW inverter (Conext TL 20000 E), it will report the following:

- Manufacturer Name: Schneider Electric
- Product Identification: PVSNVC20000
- Product Version Number: 000.000

NOTICE

The Product Version Number is not used on the Conext TL inverter. It will always report 000.000.

Example:

The Modbus master asks the device at Modbus address 01 to identify itself.

Modbus function code 43 (0x2B) uses a sub-function code to distinguish different behaviors for the function. The Conext TL inverter supports sub-function 14 (0x0E).

Table 13 Request packet

Slave	Function	Sub-function	ID Code	Object ID	CRC checksum	
01	2B	0E	01	00	70	77

The ID Code field supports four values:

- 01: request to get the basic device identification (stream access)
- 02: request to get the regular device identification (stream access)
- 03: request to get the extended device identification (stream access)
- 04: request to get one specific identification object (individual access)

NOTICE

The Conext TL inverter only supports ID code 01.

The value of Object ID determines the items in the response, as shown in Table 14.

Table 14 Contents of response, depending on Object ID

Object ID	Manufacturer name	Product identification	Product version number
0x00	✓	✓	✓
0x01		✓	✓
0x02			✓

Response Packet:

Slave ID	01	Indicates the address of the responding slave device
Function Code	2B	Indicates a function code 43 (0x2B) response
Sub Code	0E	Sub code 14 (0x0E) is the only sub code supported
Read Device ID Code	01	Same as the Read Device ID code in the request packet
Conformity Level	01	Identifies the conformity level of the device to Function Code 43. 01 = basic identification, and is the only value supported by the Conext TL inverter.
More Follows	00	If there is not enough room in the packet, this field will indicate that more data follows with FF.
Next Object ID	00	Specifies the starting ID of the object in the next response if “More Follows” is FF.
Number of Objects	03	Specifies the number of objects contained in this response.
Object ID	00	The identifier for the following object: 00 = Manufacturer
Object Length	12	Specifies the length of the following object (in bytes). 0x12 = 18 bytes.

Object Data									
53	63	68	6E	65	69	64	65	72	20
S	c	h	n	e	i	d	e	r	

Object Data							
45	6C	65	63	74	72	69	63
E	l	e	c	t	r	i	c

Object ID	01

The identifier for the following object: 01 = Product Code

Object Length	0B

Specifies the length of the following object (in bytes). 0x0B = 11 bytes.

Object Data										
50	56	53	4E	56	43	32	30	30	30	30
P	V	S	N	V	C	2	0	0	0	0

Object ID	02

The identifier for the following object: 01 = Product Code

Object Length	07

Specifies the length of the following object (in bytes). 0x07 = 7 bytes.

Object Data						
30	30	30	2E	30	30	30
0	0	0	.	0	0	0

CRC Checksum	0D	03

Broadcasts

Broadcast request packets from the master are supported. Broadcasts are valid only with Function 16 and are triggered by setting the slave address to zero (0). All slaves will receive and execute the request, but will not respond.

Modbus Data Types

This section describes the data types supported by the inverter. The available formats may vary, depending on your inverter type and firmware.

Table 15 Modbus data types

Format	Data type	Range
UINT16	16-bit unsigned integer	0 to 65,535
INT16	16-bit signed integer	-32,768 to +32,767
UINT32	32-bit unsigned integer	0 to 4,294,967,295
INT32	32-bit signed integer	-2,147,483,648 to +2,147,483,647
UINT8	8-bit unsigned character	0 to 255

16-bit integer format

Unsigned and signed 16-bit integer formats are the smallest addressable units when using the Modbus protocol. Each input register to the module corresponds to one 16-bit Modbus holding register output.

32-bit integer format

To accommodate values that can reach beyond the 16-bit range, the Modbus Slave module provides 32-bit integer format as an output option.

A 32-bit register is passed via communications as two 16-bit registers—one high-order register and one low-order register.

High-order register

- $register_{high}$=value/65536

Low-order register

- $register_{low}$= value modulus 65536
- value = $register_{high}$ x 65536 + $register_{low}$ or
- value = $register_{high}$ | $register_{low}$

Example (unsigned 32-bit):

Value 12345678 is passed in unsigned 32-bit integer format:

- 12345678 = 0x00BC614E
- $Register_{high}$ = 0x00BC (unsigned) = 188
- $Register_{low}$ = 0x614E (unsigned) = 24910
- Value = 188 x 65536 + 24910 = 12345678

In unsigned 32-bit integer format, both the high-order and low-order registers are unsigned 16-bit integers.

Example (signed 32-bit):

Value -12345678 is passed in signed 32-bit integer format:

- -12345678 = 0xFF439EB2
- $Register_{high}$ = 0xFF43 (signed) = -189
- $Register_{low}$ = 0x9EB2 (unsigned) = 40626
- value = -189 x 65536 + 40626 = -12345678

In signed 32-bit integer format, the high-order register is a signed 16-bit number, but the low-order register is unsigned.

8-bit Unsigned Character Format

The 8-bit Unsigned Character format is used to encode ASCII strings within Modbus registers.

Characters are stored in the order they occur within the string, and populate the Most Significant Byte (MSB) of the Modbus 16-bit register followed by the Least Significant Byte (LSB) of the Modbus 16-bit register. For example, the ASCII string “HELLO!” would be encoded as 3 consecutive 16-bit registers with the values 0x4845, 0x4C4C, and 0x4F21.

Table 16 Modbus ASCII string encoding example

Register 1		Register 2		Register 3	
4845		4C4C		4F21	
48	45	4C	4C	4F	21
H	E	L	L	O	!

Modbus Error Responses

If the inverter receives an unsupported Modbus request, it returns an exception response informing the Modbus master of the nature of the error.

The Modbus Error Response message has two fields that differentiate it from a normal response: Function Code Field, and Data Field.

Function Code Field

In a normal response, the inverter echoes the function code of the original request in the function code field of the response. All function codes have a most-significant bit (MSB) of 0 (their values are all below 0x80).

In an exception response, the inverter sets the MSB of the function code to 1. This makes the function code value in an exception response exactly 0x80 higher than the value would be for a normal response. For example, a normal response of 0x03 (Read Holding Registers), becomes 0x83 (Unable to Read Holding Registers).

Data Field

In an error response, the inverter uses the data field of the response packet to return an error code to the Modbus Master. Four error codes are supported, as shown in Table 17.

Table 17 Modbus error codes

Error code	Error name	Error description
01	Illegal Function	The inverter does not support the function code specified in the Modbus Request Packet.
02	Illegal Address	The address range specified in the Modbus Request Packet contains an illegal register address.
03	Illegal Data Value	The Modbus Request Packet contains an illegal number of bytes in the data field.
04	Slave Device Failure	An unrecoverable error occurred while the inverter (slave) was attempting to perform the requested action

Modbus Error Response Example

A Modbus master requests 12 registers at address 0x17F4 using the following query:

Table 18 Request packet

Slave	Function	Start registers		# of registers (1)		CRC checksum	
C9	03	17	F4	00	0C	11	C1

In this case, 0x17F4 is a valid address, but the range from 0x17F4 to 0x17FF has three missing registers at addresses 0x17F5, 0x17F6, and 0x17F7, so the inverter responds with an “Illegal Address” error response.

The response packet is shown in Table 19.

Table 19 Response packet

Slave	Function	Error code	CRC checksum	
C9	83	02	41	0F

Event Logs

Events are stored whenever they occur. The maximum rate for storing is 4 events every 1.5 ms.

Event logs are time-stamped with a resolution of one millisecond.

The Logging function manages all aspects of event logging.

This function can be configured to be in different modes via the “Logger-mode” register. By default, it is set to “Log” mode where it continues to log events.

To extract any logs, the Logger has to be configured for “read” mode; in this mode, logging is stopped.

Logger Mode

Table 20 Logger-mode register

Address	Name	Access	Default	Units / range
0x4705	Logger-mode	R/W	0	UINT16, [0..5]

The different modes of logging operation are shown in Table 21.

Table 21 Logger mode values

Logger-mode	Action
0	Log Mode: Events will be logged. Log reading and configuration are disabled.
2	Read Event Log: The requested event will be read
5	Config Mode: Allow setting of RTC time and log interval. Allow clearing of the logs

Logger State

The state of the Logging function can be read with the Logger-state register, shown in Table 22.

Table 22 Logger-state register

Address	Name	Access	Default	Units / range
0x4706	Logger-state	R	0	UINT16

The logger state is a bit field, described in Table 23. A set bit (bit = 1) indicates that status is true.

Table 23 Logger State bit indicators

Bit	Status
0	Board not ready (initialization not finished, communication error)
1	Log mode active
2	Read mode active
3	Config mode active
9	CRC error detected when reading NVRAM data. Data of the current record is corrupt.
10	Any other inverter error

Event Log Size

The number of records stored in the log can be determined by reading register 0x4701 “Number of Event Log Records”.

Table 24 Number of Event Log Records register

Address	Name	Access	Default	Units / range
0x4701	Number of Event Log Records	R	0	UINT16, [0..8192]

Log Request Registers

Table 25 describes the registers that control the reading of logs.

Table 25 Log Request registers

Address	Name	Access	Default	Units / range
0x4703	Log Request Index	R/W	0	UINT16, [0..Idx_max]
0x4704	Log Response Index	R	0	UINT16, [0..Idx_max]

The “Log Request Index” is limited to the maximum number of records in the event log log.

When “Logger-mode” is written, you must reset the “Log Request Index” to 0. The register is only writeable when “Logger-mode” is configured to “Read Event Log”.

Index 0 means that nothing is to be read from Logger.

Index 1 means the most recent record of the event log.

The “Log Response Index” is set to the same value as the “Log Request Index” as soon as the data is available in the corresponding buffer registers. This register should be polled until the expected value is read.

To read logs:

1. Set the “Logger-mode” register to 2 (to enable reading of the event logs).
2. Write the index of the record to be read to register “Log Request Index”.
3. Poll the register “Log Response Index”. When this index equals the “Log Request Index”, the data is available in the particular buffer registers.
4. Read the buffer registers of the corresponding log.
5. Set the “Logger-mode” register to 0 (to resume logging).

Event Log Record

The event log content is available from reading registers 0x4720 to 0x4726.

A record of the event log consists of 7 registers, described in Table 26.

Table 26 Event Log Record registers

Address	Name	Access	Units / range
0x4720	Timestamp High	R	UINT32, Number of seconds since Jan 01 1970
0x4721	Timestamp Low	R	
0x4722	Timestamp ms	R	UINT16, ms, [0..999]
0x4723	Event 1	R	UINT16
0x4724	Event 2	R	UINT16
0x4725	Event 3	R	UINT16
0x4726	Event 4	R	UINT16

The event log time stamp is calculated as:

(TimestampHigh * 65536) + TimestampLow + (Timestamp ms / 1000)

The event log can store a maximum of four simultaneous events. All valid events are non-zero numbers.

Clearing Logs

To clear logs:

- Write the value 0x2006 to the Clear Log register, described in Table 27.

Table 27 Clear Log register

Address	Name	Access	Default	Units / range
0xF002	Clear Log	W	0	UINT16

Conext TL Modbus Map

The Modbus map defines the location of the Conext TL registers, which can be used to retrieve status information or to control the inverter.

The Modbus map is defined as a data table, the columns of which define the register attributes. (The following sections describe each of the columns in the table.) Modbus registers are defined one per row.

The Modbus map is shown in Table 30 on page 29.

Modbus Address

Addresses in the Modbus map are zero-based and are specified in hexadecimal notation, so they correspond directly with the address field specified in the Modbus Request Packet. This makes it easier to troubleshoot when capturing data “over-the-wire”.

If you need to enter these values into the data definition file of a Modbus Master device, which is expecting a “Register Number”, you will need to convert the address to decimal (base 10), and add one to the address.

For example, the “Operational Mode State” register is at address 0x1700.

0x0x1700 = 5888 decimal.

Adding one to this gives a Register Number of 5889.

Modbus Register Description

The Description column provides brief details about the purpose of the Modbus register, for example, “Energy Since Commissioning”, or “Operational Mode State”.

Modbus Register Access type

Access types may be either “Read Only” or “R/W” (Read/Write). The values of “Read Only” registers cannot be changed via Modbus, so they cannot be used to configure or alter the behavior of the inverter in any way. “Read/Write” registers may be altered at run-time and may directly affect the operation of the inverter.

Modbus Register Units

The Units column specifies two pieces of information separated by a / symbol. The first is the units of the value contained in the register, for example, kWh, V, A, Vrms. The second is the scaling factor to convert the raw binary value contained within the register to the units specified before the / symbol.

Table 28 Scale factor in the Units column

Scale factor	Action
X1	No divide necessary: result equals raw value
X10	Divide raw value by 10 to yield the result
X100	Divide raw value by 100 to yield the result

For example, if the Units column is kWh/X10, and the raw value retrieved from the Modbus register is 24, divide this value by 10 to yield 2.4 kWh.

Modbus Register Size

The Size column specifies the total size of the data at the register address specified by the Address column.

Examples:

Table 29 Modbus Register size and total size of data

Size	Number of bytes	Number of 16-bit Modbus registers
8-bit unsigned integer x 20	20 bytes	10
32-bit unsigned integer	4 bytes	2
32-bit signed integer	4 bytes	2
16-bit signed integer	2 bytes	1
16-bit unsigned integer	2 bytes	1

Invalid Registers

In the inverter Modbus register map, there are gaps between some registers. For example, the next register after 0x17F4 is 0x17F8. Unmapped registers (0x17F5 through to 0x17F7) are **invalid**. Requests to read data from invalid registers generate an “Invalid Address” exception. When an invalid register is written, the inverter responds with an “Invalid Address” exception. For an example of this, see “Modbus Error Response Example” on page 22.

Invalid Data

If the value or data written to an address is out of range, the inverter returns ExceptionCode = 02.

For example, the valid ranges for address 0xFA61 are +80 to +100 and -80 to -100. If the value 35 or -150 is written to the address, the inverter returns ExceptionCode = 02.

Modbus Map

For a complete description of the Conext TL registers available via Modbus, see Table 30.

NOTICE

- To access the following addresses, the version of the inverter's communication software must be higher than 1.22 (the version is displayed on the Inverter Information screen, as "Comm.-Version"):
  - 0x182F (PV2 Power)
  - 0xF9FB (User Phase Angle)
  - 0xFA19 (User Active Power Limit)
  - 0xFA1B (User Reactive Power Reference)
  - From 0xFA60 (Reactive Power Mode Select) to 0xFA6E (kVar(U) Response Time)
- For "range", the sign convention we use is from the supply network (that is, utility's) perspective, looking towards the inverter. In that context, "capacitive" means leading VArs and a positive phase angle; "inductive" means lagging VArs and a negative phase angle.
- For more information on leading and lagging power factors, see page 38.

Table 30 Modbus map

Address (hex)	Description	Access	Units	Size
0x0001 - 0x0009	Product Model Designation	Read Only	"C" style null terminated ASCII string	UINT8 x 18
0x0014 - 0x001D	Serial Number	Read Only	"C" style null terminated ASCII string	UINT8 x 20
0x0082 - 0x008B	Software Part Number for Processor A	Read Only	"C" style null terminated ASCII string	UINT8 x 20
0x0096 - 0x009F	Software Part Number for Processor B	Read Only	"C" style null terminated ASCII string	UINT8 x 20
0x00AA- 0x00B3	Software Part Number for Processor C	Read Only	"C" style null terminated ASCII string	UINT8 x 20

Table 30 Modbus map (Continued)

Address (hex)	Description	Access	Units	Size
0x0802 (H word) - 0x0803 (L word)	Energy Since Commissioning	Read Only	KWhr/X10	UINT32
0x0804 (H word), 0x805 (L word)	Energy Today	Read Only	KWhr/X10	UINT32
0x0806, 0x807	Energy Today (1 day earlier)	Read Only	KWhr/X10	UINT32
0x0808, 0x809	Energy Today (2 days earlier)	Read Only	KWhr/X10	UINT32
0x080A, 0x80B	Energy Today (3 days earlier)	Read Only	KWhr/X10	UINT32
0x080C, 0x80D	Energy Today (4 days earlier)	Read Only	KWhr/X10	UINT32
0x080E, 0x80F	Energy Today (5 days earlier)	Read Only	KWhr/X10	UINT32
0x0810, 0x811	Energy Today (6 days earlier)	Read Only	KWhr/X10	UINT32
0x081E, 0x081F	Operating Hours	Read Only	Hr/x1	UINT32
0x1700	Operational Mode State	Read Only	Enum • 0x0002 Reconnecting (Conext TL Countdown) • 0x0003 Online (Conext TL ON) • 0x0014 Standby • 0x0015 No DC • 0x0016 Alarm	UINT16
0x1701	Temperature 1, NTC on control board	Read Only	C/X10	INT16

Table 30 Modbus map (Continued)

Address (hex)	Description	Access	Units	Size
0x1702	Temperature 2, Boost module 1	Read Only	C/X10	INT16
0x1703	Temperature 3, Boost module 2	Read Only	C/X10	INT16
0x1704	Temperature 4, Inverter module	Read Only	C/X10	INT16
0x17F1	Apparent Power	Read Only	kVA/x10	UINT16
0x17F4	Reactive Power	Read Only	kVAr/X10	INT16
0x17F8	Inverter Vab	Read Only	Vrms/X10	INT16
0x17F9	Inverter Vbc	Read Only	Vrms/X10	INT16
0x17FA	Inverter Vca	Read Only	Vrms/X10	INT16
0x17FB	Phase A Current	Read Only	Arms/X10	INT16
0x17FC	Phase B Current	Read Only	Arms/X10	INT16
0x17FD	Phase C Current	Read Only	Arms/X10	INT16
0x17FE	Real Power (Total)	Read Only	kW/X10	INT16
0x17FF	PV1 Voltage	Read Only	V/X10	INT16
0x1800	PV1 Current	Read Only	A/X10	INT16
0x1801	PV1 Power	Read Only	kW/X10	INT16
0x1828	Fault Code	Read Only	See the Conext TL Installation and Operation Manual	UINT16
0x180D	DC Voltage	Read Only	V/X10	UINT16
0x1818	Phase A Frequency	Read Only	Hz/x10	UINT16
0x1819	Phase B Frequency	Read Only	Hz/x10	UINT16
0x181A	Phase C Frequency	Read Only	Hz/x10	UINT16
0x181B	Phase A Real Power	Read Only	kW/x10	INT16
0x181C	Phase B Real Power	Read Only	kW/x10	INT16
0x181D	Phase C Real Power	Read Only	kW/x10	INT16
0x1829	PV2 Voltage	Read Only	V/X10	INT16

Table 30 Modbus map (Continued)

Address (hex)	Description	Access	Units	Size
0x182A	PV2 Current	Read Only	A/X10	INT16
0x182F	PV2 Power[a]	Read Only	kW/X10	INT16
0xF9FB	User Phase Angle[a] **Note:** You can use either this register or register 0xFA61 (Cos(φ); page 33) to specify the angle. The inverter uses the value from the most recently written of the two registers.	R/W	Increments of 1 degrees Range: +37 capacitive (leading) to -37 inductive (lagging) Default = 0 degrees	INT16
0xFA19	User Active Power Limit[a]	R/W	kW/X10	UINT16
0xFA1B	User Reactive Power Reference[a] **Note:** You can use either this register or register 0xFA66 (Fixed kVAr %; page 34) to specify the amount of reactive power sourced (capacitive) or sinked (inductive) by the inverter. The inverter uses the value from the most recently written of the two registers.	R/W	kVAr/X10 Positive numbers make the inverter look capacitive (leading VAr) to the grid. Negative numbers make the inverter look inductive (lagging VAr) to the grid.	INT16
0xFA60	Reactive Power Mode Select[a]	R/W	• 0 : Disable (Default) • 1 : Fixed cos(φ) • 2 : cos(φ) as function of P • 3 : Fixed kVAr • 4 : kVAr as function of U	UINT16

Table 30 Modbus map (Continued)

Address (hex)	Description	Access	Units	Size
0xFA61	Fixed Cos(φ)[a] **Note:** You can use either this register or register 0xF9FB (User Phase Angle; page 32) to specify the angle. The inverter uses the value from the most recently written of the two registers.	R/W	Increment = 1 unit Range: • Capacitive 0.80 to 1.00. Enter: +80 to +100 • Inductive: 0.80 to 1.00. Enter: -80 to -100 Default: 100	INT16
0xFA62	Cos(φ) with Power[a] Upper limit	R/W	Increment = 1 unit Range: • Capacitive 0.80 to 1.00. Enter: +80 to +100 • Inductive: 0.80 to 1.00. Enter: -80 to -100 Default: 90	INT16
0xFA63	Cos(φ) with Power[a] Lower limit	R/W	Increment = 1 unit Range: • Capacitive 0.80 to 1.00. Enter: +80 to +100 • Inductive: 0.80 to 1.00. Enter: -80 to -100 Default: -90	INT16
0xFA64	Cos(φ) with % power[a] Lower limit	R/W	Increment = 1% Range: 0~100% Default: 0%	UINT16

Table 30 Modbus map (Continued)

Address (hex)	Description	Access	Units	Size
0xFA65	Cos(φ) with % power[a] Upper limit	R/W	Increment = 1% Range: 0~100% Default: 100%	UINT16
0xFA66	Fixed kVAr %[a] **Note:** You can use either this register or register 0xFA1B (User Reactive Power Reference; page 32) to specify the reactive power either sourced or sinked by the inverter. The inverter uses the value from the most recently written of the two registers.	R/W	Increment = 1% Range: -53 to +53% Default: 0% Positive numbers make the inverter look capacitive (leading VAr) to the grid. Negative numbers make the inverter look inductive (lagging VAr) to the grid.	INT16
0xFA67	kVAr(U) as %[a] Upper limit	R/W	Increment = 1% Range: -53 to +53% Default: 44%	INT16
0xFA68	kVAr(U) as %[a] Lower limit	R/W	Increment = 1% Range: -53 to +53% Default: -44%	INT16
0xFA69	kVAr(U)[a] Vmin	R/W	Unit: 0.1 V Range: 100~264 V Default: 184 V	UINT16
0xFA6A	kVAr(U)[a] Vmax	R/W	Unit: 0.1 V Range: 100~264 V Default: 253 V	UINT16
0xFA6B	kVAr(U)[a] UAC Lower limit	R/W	Unit: 0.1 V Range: 100~264 V Default: 230 V	UINT16

Table 30 Modbus map (Continued)

Address (hex)	Description	Access	Units	Size
0xFA6C	kVAr(U)[a] UAC Upper limit	R/W	Unit: 0.1 V Range: 100~264 V Default: 230 V	UINT16
0xFA6D	kVAr(U)[a] Hysteresis	R/W	Unit: 0.1 V Range: 0~30 V Default: 0 V	UINT16
0xFA6E	kVAr(U)[a] Response Time	R/W	Unit: 0.01 sec Range: 10~60 sec Default: 10 sec	UINT16
0x4701	Number of Event Log Records	Read Only	NA	
0x4703	Log Request Index	R/W	NA	INT16
0x4704	Log Response Index	Read Only	NA	INT16
0x4705	Logger-mode	R/W	NA	INT16
0x4706	Logger-state	Read Only	NA	INT16
0x4720	Timestamp High	Read Only	NA	INT16
0x4721	Timestamp Low	Read Only	NA	INT16
0x4722	Timestamp ms	Read Only	NA	INT16
0x4723	Event 1	Read Only	NA	INT16
0x4724	Event 2	Read Only	NA	INT16
0x4725	Event 3	Read Only	NA	INT16
0x4726	Event 4	Read Only	NA	INT16
0xF002	Clear Log	Write Only	NA	INT16

a.Communication software version must be higher than 1.22.

Appendix A: CRC-16 calculation

This appendix describes how to obtain the CRC-16 error check field for a Modbus RTU frame.

A frame can be considered as a continuous, serial stream of binary data (ones and zeros). The 16-bit checksum is obtained by multiplying the serial data stream by 2^{16} (10000000000000000) and then dividing it by the generator polynomial $x^{16}+x^{15}+x^{2}+1$, which can be expressed as the 16-bit binary number 11000000000000101.

The quotient is ignored and the 16-bit remainder is the checksum, which is appended to the end of the frame.

In calculating the CRC, all arithmetic operations (additions and subtractions) are performed using MODULO TWO, or EXCLUSIVE OR operation. A step-by-step example shows how to obtain the checksum for a simple Modbus RTU frame.

To generate the CRC-16 checksum:

1. Drop the MSB (Most Significant Bit) of the generator polynomial and reverse the bit sequence to form a new polynomial. This yields the binary number 1010 0000 0000 0001, or 0xA001.
2. Load a 16-bit register with initial value 0xFFFF.
3. Exclusive OR the first data byte with the low-order byte of the 16-bit register. Store the result in the 16-bit register.
4. Shift the 16-bit register one bit to the right.
5. If the bit shifted out to the right is one (1), Exclusive OR the 16-bit register with the new generator polynomial, store the result in the 16-bit registers. Return to step 4.
6. If the bit shifted out to the right is zero (0), return to step 4.
7. Repeat steps 4 and 5 until 8 shifts have been performed.
8. Exclusive OR the next data byte with the 16-bit register.
9. Repeat steps 4 through 7 until all bytes of the frame are Exclusive OR with the 16-bit register and shifted 8 times.

The content of the 16-bit register is the checksum and is appended to the end of the frame.

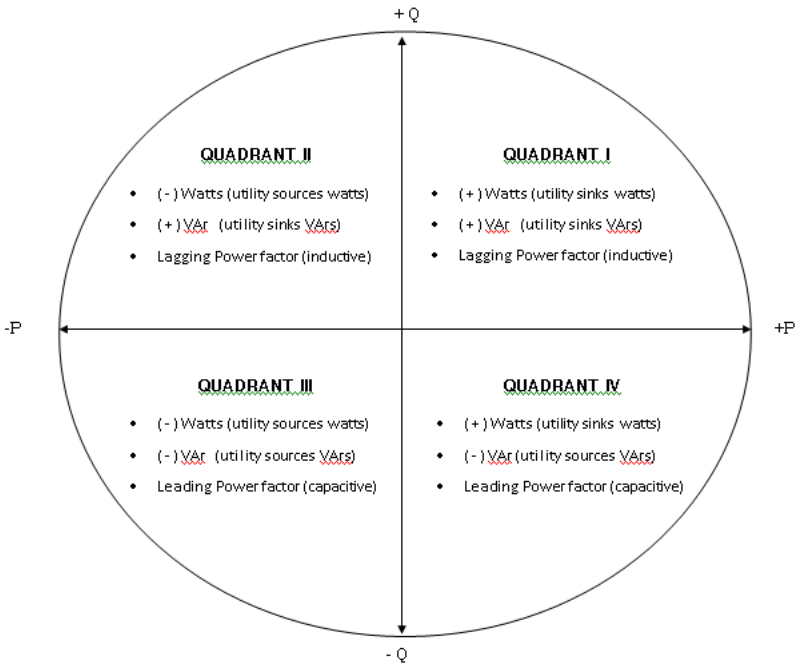
Pseudocode For CRC-16 Generation

For users familiar with computer programming, the following is the pseudocode for calculating the 16-bit Cyclic Redundancy Check.

```
Initialize a 16-bit register to 0xFFFF
Initialize the generator polynomial to 0xA001
FOR n=1 to # of bytes in packet
    BEGIN
    XOR nth data byte with the 16-bit register
    FOR bits_shifted = 1 to 8
        BEGIN
            SHIFT 1 bit to the right
            IF (bit shifted out EQUAL 1)
                XOR generator polynomial with the 16-bit register and
                store result in the 16-bit register
        END
    END
```

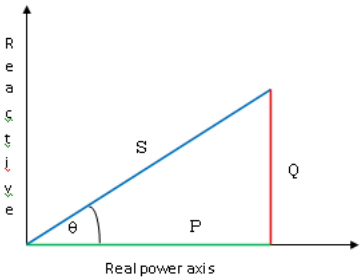
The resultant 16-bit register contains the CRC-16 checksum.

Appendix B: Leading and Lagging Power Factors



Notes on the above figure:

- All the above is as seen from the utility's perspective (looking toward the inverter).
- The direction of **reactive** power flow determines **leading** or **lagging** power factor.
- The terms **leading** and **lagging** always refer to the current with respect to the voltage.



In the above figure:

- S = Apparent power; has dimensions of volt-amps (VA)
- P = Active or real power; has dimensions of watts (W)
- Q = Reactive power; has dimensions of volt-amp-reactive (VAr)
- PF = power factor; has no units. It is defined as the cosine of angle θ (the ratio of P over S)

Exclusion for Documentation

UNLESS SPECIFICALLY AGREED TO IN WRITING, SELLER
(A) MAKES NO WARRANTY AS TO THE ACCURACY, SUFFICIENCY OR SUITABILITY OF ANY TECHNICAL OR OTHER INFORMATION PROVIDED IN ITS MANUALS OR OTHER DOCUMENTATION;
(B) ASSUMES NO RESPONSIBILITY OR LIABILITY FOR LOSSES, DAMAGES, COSTS OR EXPENSES, WHETHER SPECIAL, DIRECT, INDIRECT, CONSEQUENTIAL OR INCIDENTAL, WHICH MIGHT ARISE OUT OF THE USE OF SUCH INFORMATION. THE USE OF ANY SUCH INFORMATION WILL BE ENTIRELY AT THE USER'S RISK; AND
(C) REMINDS YOU THAT IF THIS MANUAL IS IN ANY LANGUAGE OTHER THAN ENGLISH, ALTHOUGH STEPS HAVE BEEN TAKEN TO MAINTAIN THE ACCURACY OF THE TRANSLATION, THE ACCURACY CANNOT BE GUARANTEED. APPROVED CONTENT IS CONTAINED WITH THE ENGLISH LANGUAGE VERSION WHICH IS POSTED AT WWW.SCHNEIDER-ELECTRIC.COM.

Date: March 2012 **Revision:** Revision A **Document Number:** 976-0301-01-01

Contact Information

www.schneider-electric.com

N. America	1 408 987 6255	1 925 245 1022	re.techsupport@schneider-electric.com
France	+33 (0) 825 012 999		
Deutschland	+49 (0) 180 575 3 575	+49 (0) 2102 404 7101	pv-service@de.schneider-electric.com
España	+34 902 101813	+34 93 305 5026	es-sat@es.schneider-electric.com
L'Italia	+39 035 4151111	+39 035 4153200	IT-pronto-contatto@it.schneider-electric.com
Nederland	0031 235 124124	0031 235 124111	salesassistance@nl.schneider-electric.com

For other country details please contact your local Schneider Electric Sales Representative or visit our website at:
http://www.schneider-electric.com/sites/corporate/en/support/operations/local-operations/local-operations.page