# Application Note: Using Modbus With the Conext™ CL Series

976-0317-01-01
Rev A

# Important Safety Instructions

## READ AND SAVE THESE INSTRUCTIONS - DO NOT DISCARD

This document contains important safety instructions that must be followed during the installation and maintenance of the Conext CL three phase transformerless grid tie inverters. Read and keep this document for future reference.

Read these instructions carefully and look at the equipment to become familiar with the device before trying to install, operate, service or maintain it. The following special messages may appear throughout this bulletin or on the equipment to warn of potential hazards or to call attention to information that clarifies or simplifies a procedure.

This is the safety alert symbol. It is used to alert you to potential personal injury hazards. Obey all safety messages that follow this symbol to avoid possible injury or death.

The addition of this symbol either to a “Danger” or “Warning” safety label indicates that an electrical hazard exists which will result in personal injury if the instructions are not followed.

**⚠ ⚠ DANGER**

DANGER indicates an imminently hazardous situation, which, if not avoided, will result in death or serious injury.

**⚠ WARNING**

WARNING indicates a potentially hazardous situation, which, if not avoided, can result in death or serious injury.

Schneider Electric

**⚠ CAUTION**

CAUTION indicates a potentially hazardous situation, which, if not avoided, can result in moderate or minor injury.

***NOTICE***

NOTICE is used to address practices not related to physical injury. The safety alert symbol shall not be used with this signal word.

**⚠ ⚠ DANGER**

**RISK OF FIRE, ELECTRIC SHOCK, EXPLOSION, AND ARC FLASH**

This Application Note is in addition to, and incorporates by reference, the installation and operation manual for the Conext™ CL 18000NA, 25000NA, 20000E and 25000E photovoltaic grid tie inverters. Before referring this Application Note you must read the relevant product manuals. 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.

**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 CL 18000NA (part number PVSCL18NA)

Conext CL 25000NA (part number PVSCL25NA)

Conext CL 20000E (part number PVSCL20E)

Conext CL 25000E (part number PVSCL25E)

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 register definition. 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 and Ethernet. 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. The Master can be a datalogger or any monitoring solution. In this scheme, information and data are transferred between a Modbus Master and up to 31 Slave devices.

The Master device initiates and controls all the 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 the 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 the packets transmitted by a Slave are responses.

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

The Conext™ CL 18000NA, 25000NA, 20000E and 25000E photovoltaic grid tie inverters support only the Modbus/RTU protocol.

### Related Documents

**Table 1** Document references

| Document reference | Document title | Document number | Version |
|---|---|---|---|
| 1 | Modbus Application Protocol Specification | From www.modbus.org | 1.1b |
| 2 | Conext CL 18000NA, 25000NA, 20000E and 25000E Installation and Operation Manual | 990-5058-001;<br>990-9687-001 | A |
| 3 | Modbus Mesaging on TCP/IP Implementation Guide | From www.modbus.org | 1.0b |
| 4 | Modbus Map: Conext™ CL | 503-0266-01-01 | A |
| 5 | SunSpec Alliance Interoperability Specification Common Models NA 1.5 | | |
| 6 | SunSpec Alliance Interoperability Specification Inverter Models NA 1.1 | | |
| 7 | SunSpec-Alliance-M-MPPT-Specification-T-1 NA TEST1 | | |

# Modbus Physical Layer

The Conext CL inverter supports the Modbus communication protocol via two physical interfaces: RS485 and Ethernet - Modbus TCP/IP. The location of the connectors on the communication card are as shown in Figure 1.

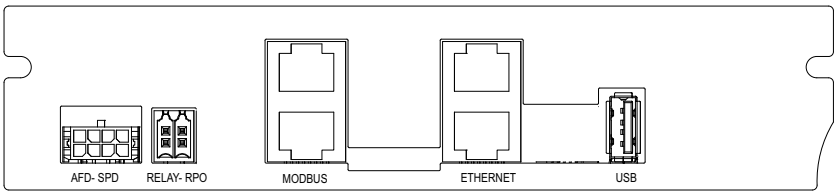


**Figure 1** Communication card connector locations

Note:

- Ethernet connection does not support daisy chaining. For details on Ethernet connection, refer to the Conext CL Installation and Operation manual.
- Modbus TCP/IP shares the same register definitions as Modbus RS485.
- Modbus TCP/IP uses default port 502.
- For TCP/IP protocol details refer to the Modbus Messaging on TCP/IP Implementation Guide V1.0b.

This application note describes the RS485 connection and protocol with examples.

## Modbus RS485 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 Modbus (RJ-45) connection are shown in Table 1-1.

***NOTICE***

**RISK OF EQUIPMENT DAMAGE**

Make sure the other end of the Modbus (RS485)/Ethernet connection is also Modbus (RS485)/ Ethernet. Connection to any other type of communication port, such as Ethernet, may result in an equipment damage.

**Failure to follow these instructions can result in equipment damage.**

Note:

- Using incorrect pin out for RS-485 cable and interchanging the GND pins, results in discontinuity on the network and poor communication.
- It is recommended to use the shielded Cat5 cable 24 AWG.

**Table 1-1** RJ-45 pin details

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

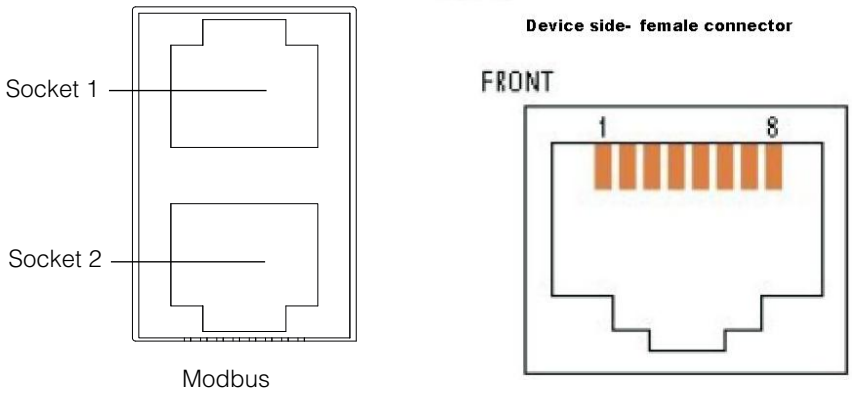


**Figure 2** Modbus RS485 connection and pin details

## Communication parameters

Table 2 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 2** Data format for the RS485 connection

| Parameter | Value |
|---|---|
| Baud rate | 19200 (default), 9600, 38400, 57600, 115200 |
| Data bits | 8 |
| Stop bits | 1 (default), 2 |
| Parity | None (default), Odd, Even |

# 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. The Inverter ID is selected using the Select button as shown in Figure below. Once the desired inverter ID is selected, press Enter to confirm the ID. The Inverter ID can be any number between 1 and 254.

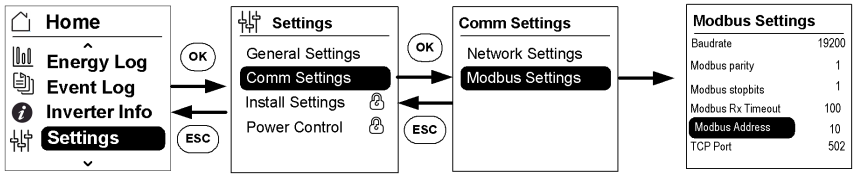


**Figure 3** Setting the Inverter ID

## Reviewing the Modbus Slave Address (Inverter ID)

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

From the main menu, select **Inverter Information**.

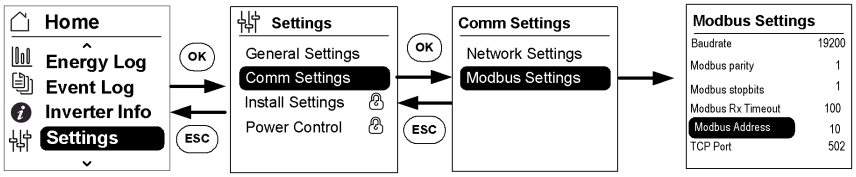


**Figure 4** Review the Inverter ID

# 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.<br>• In the tables that show the packet structure, white background denotes the DATA field of the packet. |

**Table 3** 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.

## Packet communications

This section describes the Modbus functions supported by the inverter.

### Modbus functions supported by the inverter

**Table 4** Data format for the RS485 connection

| Function (Decimal/Hex) | Meaning | Action | See this section: |
|---|---|---|---|
| 03/03h | Read Holding Registers | Reads a value from one or more consecutive holding registers in the inverter. | “Function 03: Read Holding Registers” on page 1–9 |
| 06/06h | Write Holding Registers | Writes a value into one holding register in the inverter. | Function 06h |

**Table 4** Data format for the RS485 connection (Continued)

| Function (Decimal/Hex) | Meaning | Action | See this section: |
|---|---|---|---|
| 16/10h | Write Multiple Registers | Writes a value into one or more consecutive holding registers in the inverter. | "Function 16: Write Multiple Registers" on page 1–10 |
| 43/2Bh | Read Device Identifier | Reads the Manufacturer, Model, and Version information for the device. | "Function 43: Read Device Identifier" on page 1–11 |

## Function 03: Read Holding Registers

To read the 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).

Note: Addresses are 0 based ("on the wire" addressing) and not 1 based ("traditional" addressing).

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

**Table 5** 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 the grid voltage. This parameter is made available in the Modbus map at address 0x17F8 with a scaling factor of 10.

**Table 6** Request packet

| Slave | Function | Start register | | # of registers (3) | | CRC checksum | |
|---|---|---|---|---|---|---|---|
| 01 | 03 | 17 | F8 | 00 | 01 | 00 | 4F |

**Table 7** Response packet

| Slave | Function | Byte count | Register 1 | | CRC checksum | |
|---|---|---|---|---|---|---|
| 01 | 03 | 02 | 12 | 27 | F4 | FE |

The master retrieves the data from the response:

Register 0x17F8: 0x1227 = 4647(scaled 464.7 for NA Model)

## Function 16: Write Multiple Registers

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

A Write 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 8** 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 register data fields, the Preset Registers Response packet has the same fields as the Read Registers Request packet.

Example:

Write to the “Reactive power mode select” register of the inverter at Modbus address 01 (0x01) to set "cos phi as a function of P”. The reactive power mode select register is at address 0xFA60, and it must be set to “2” (0x0002) to set the "cos phi as a function of P”.

**Table 9** Request packet

| Slave | Function (Hex) | Start register | | # of registers | Byte count | Register 1 | | CRC checksum | |
|---|---|---|---|---|---|---|---|---|---|
| 01 | 10 | FA | 60 | 01 | 02 | 00 | 02 | 25 | A9 |

**Table 10** Response packet

| Slave | Function | Start register | | # of registers | | CRC checksum | |
|---|---|---|---|---|---|---|---|
| 01 | 10 | FA | 60 | 00 | 01 | 31 | 0F |

## Function 43: Read Device Identifier

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 three pieces of data, as shown in Table 11.

**Table 11** 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 In the case of the 18KW inverter (Conext 18000NA), it will report the following:

- Manufacturer Name: Schneider Electric
- Product Identification: PVSCL18NA
- Product Version Number: 000.000

Example:

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

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

**Table 12** 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 CL inverter only supports ID code 01. |

The value of Object ID determines the items in the response, as shown in Table 13.

**Table 13** 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 CL 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 | 0A |
|---|---|

Specifies the length of the following object (in bytes). 0x0A = 10 bytes.

| Object Data | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 50 | 56 | 53 | 43 | 4c | 31 | 38 | 4e | 41 |
| P | V | S | C | L | 1 | 8 | N | A |

| 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 | A7 | 2E |
|---|---|---|

### Broadcasts

Broadcast request packets from the master are supported. Broadcasts are valid only with Function 16 and 06h are triggered by setting the slave address to zero (0). All slaves will receive and execute the request, but will not respond.

Note:

After the broadcast it is recommended to read back the registers and confirm the values written are valid.

## 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 14** 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

The 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 the Modbus registers.

The 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 15** 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 the 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 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 16.

**Table 16** 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.<br>Note: Refer to table 4 for supported function |
| 03 | Illegal Data Value | The Modbus Request Packet contains an illegal number of bytes in the data field.<br>Note: All unused address contains a value 0xFF and no exception occurs in Conext CL. |
| 04 | Slave Device Failure | An unrecoverable error occurred while the inverter (slave) was attempting to perform the requested action |

## Modbus Error Response Example: Illegal Function code

A Modbus master uses the function code 02(illegal function code) to read the registers at an address 01 with a registers size of 18 bytes.

**Table 17** Request packet

| Slave | Function | Start registers | | # of registers (1) | | CRC checksum | |
|---|---|---|---|---|---|---|---|
| 01 | 02 | 00 | 01 | 00 | 09 | E9 | CC |

In this case the function code 02 is not supported in Conext CL and is mentioned as an illegal code with the response below.The response packet is shown in Table 18.

**Table 18** Response packet

| Slave | Function | Error code | CRC checksum | |
|---|---|---|---|---|
| 01 | 82 | 01 | 81 | 60 |

## Modbus Error Response Example: Illegal address

A modbus master requests 25 registers at an address 0x0001-0x0009 using the following query.

**Table 19** Request packet

| Slave | Function | Start registers | Number of registers | CRC |
|---|---|---|---|---|
| 01 | 03 | 00 01 | 00 19 | D5C0 |

In general other product will give an Error “Illegal Address” but the Conext CL doesn't give any Illegal address since unused locations will provide value of FF.



**Date:** Novembert 2014 **Revision:** Rev A **Document Number:** 976-0317-01-01

**Contact Information:** solar.schneider-electric.com.

For country-specific details, please contact your local Schneider Electric Sales Representative or visit the Schneider Electric Solar Business website at http://solar.schneider-electric.com/tech-support/