INDUSTRIAL CONTROL COMMUNICATIONS, INC.

Modbus RTU Sniffer Driver Manual

April 3, 2017

TABLE OF CONTENTS

1 Modbus RTU Sniffer

1.1 Overview

The Modbus RTU sniffer driver enables fully non-intrusive insight into any existing Modbus RTU network consisting of a master and at least one slave. The driver can be configured to “sniff” the requests of the master and log the responses of the slaves into the database. Some notes of interest are:

- The Modbus sniffer driver never transmits on the Modbus network being sniffed.
- Supported Modbus functions are indicated in Table 1.

Table 1: Supported Modbus RTU Sniffer Functions

Function Code	Function
03	Read Holding Registers
04	Read Input Registers
06	Write Single Register
16	Write Multiple Registers

- Both Holding and Input Registers are independently supported in Service Objects.
- The Modbus Sniffer Service Objects are similar to those of the Modbus Master Service Objects with the exception that instead of the driver itself generating requests, it must rely on the existing Modbus master to initiate requests on its behalf. Therefore, if the master never reads or writes a certain register that is configured in a sniffer service object, the value of that register will never be updated in the internal database.

1.2 Sniffer Settings

Baud Rate

Selects the baud rate of the network.

Parity

Selects the parity and number of stop bits.

Timeout

Defines the maximum number of milliseconds for a break in network communications before a timeout event will be triggered. To disable timeout processing, set this field to 0.

1.3 Service Object Settings

The Modbus RTU Sniffer driver is passive (listen only), and uses service objects to define what registers to log values for from the network traffic. Each input register or holding register in a service object is mapped to 2 bytes in the database (the data type is fixed at 16-bit Unsigned).

Separate service objects exist for targeting holding register accesses and input register accesses. Because these service objects are largely identical, however, their settings will be addressed together. In general, the only difference is that input register service objects always sniff reads, whereas holding register service objects can selectively sniff only reads, only writes, or both reads and writes.

Description

This 32-character (max) field is strictly for user reference: it is not used at any time by the driver.

Destination Address

Indicates the destination address (0…247 for holding register service objects and 1…247 for input register service objects) of the remote slave device on the network that contains the registers to be logged by this service object.

Note that address 0 is defined by Modbus as the broadcast address: if this address is used in a holding register service object, the "Read Function" must be set to "Disabled", as slaves cannot respond to broadcast messages. Also note that using a destination address of 0 will configure the service object to only log broadcast messages; if a destination address other than 0 is used, however, both broadcast messages and requests targeted specifically at the defined destination address will be logged.

Start Register

Defines the starting register number (1…65535) for a range of registers associated with this service object.

Number of Registers

Defines the number of registers (1…123) to be targeted by this service object.

Database Address

Defines the database address where the first register of this service object will be mapped. The configuration studio will not allow entry of a starting database address that will cause the service object to run past the end of the database. The highest valid database address, therefore, will depend on the number of items to be accessed.

Multiplier

The amount that associated network values are scaled by prior to being stored into the database. Network values (logged from the device) are divided by the multiplier before being stored into the database.

Data Type

Fixed at 16-Bit Unsigned.

Read Function

Fixed at "4 (Read Input Registers)" for input register service objects. Select whether or not to log the responses to read requests from the master to the slave. Note that the Read Function and Write Function cannot both be set to "Disabled".

Write Function

Not available for input register service objects. Select whether or not to log writes from the master to the slave. When enabled, writes using both function codes 6 and 16 are logged, so that write values will be captured regardless of the function code used by the master. Note that the Read Function and Write Function cannot both be set to “Disabled”.

1.4 Diagnostics Object

Each service object can optionally include a diagnostics object for debugging and diagnostics. Note that the diagnostics object for the Modbus sniffer driver is slightly different than that of the Modbus RTU master driver, because the sniffer driver does not actually transmit any requests itself. The diagnostics information should be interpreted from the perspective of the network master (as if the master were updating the diagnostics information). For example, when the master transmits a request to read a register, the TX Counter is incremented, and when the slave responds, the RX Counter is incremented.

Diagnostics Database Address

Enter the database address at which to store the status information.

ICC

INDUSTRIAL CONTROL COMMUNICATIONS, INC.

1600 Aspen Commons, Suite 210
Middleton, WI USA 53562-4720
Tel: [608] 831-1255 Fax: [608] 831-2045

http://www.iccdesigns.com

Printed in U.S.A