Modbus Slave Connector

Devices compatible with Modbus protocol

- Name: **ModbusSlave**
- Version: 2.0.0.9
- Protocol: MODBUS RTU, ASCII and TCP
- Interface: TCP/IP or RS-232 Serial
- Runtime: Multiplatform
- Configuration:
    - Devices / Protocols

Master Slave Nomenclature
Module Configuration
Channels Configuration
Nodes Configuration
Point Configuration
Troubleshoot
Driver Revision History

Overview

This document has only the specific information about the device connection settings and its address syntax. Refer to the User Guide for more information on the Device Module.

- Modbus Master Protocol. (this driver)
- Modbus Slave Protocol
- Schneider Electric Control Expert Unity Pro PLCs. Relies on Modbus protocol, but it was its own Import Wizards.

The configuration for this protocol can be created using The Tag Import CSV tools, or copy-paste directly to UNS Tags table.

Master Slave Nomenclature

The terminology Master and Slave refers to the original Modbus terms. The Modbus Organization announced the replacement of "master-slave" terminology with "client-server" in its communications protocol. The new nomenclature reflects the specific roles of the devices within the Modbus network. Here's the definitions:

1. **Modbus Master (Client):** In the context of the Modbus protocol, the "master" is the device that initiates communications. It's called a Client because it requests data from other devices. The master actively controls when and how data is requested and collected in the network.
2. **Modbus Slave (Server)**: Slave devices are referred to as Servers because they serve data to the master upon request. They passively wait for requests and respond by sending the requested data or performing actions based on the commands received. They do not initiate communication; they only respond to the master's requests.

In the product User Interface, as will in the documentation URLs, we opted to keep the original historical nomenclature, in order to ensure compatibility, as well to avoid potential confusion with the terms Server and Client which already have other meanings in the context of the software platform.

Module Configuration

This document has only information about the device connection settings and its address syntax.For further Information, see:

- Devices Module
- Devices Module Reference

Modbus Server driver implements communication with Client devices compatibles with Modbus Open Standard protocol. It operates as a Server on TCP/IP or serial networks.

Communication Driver Information	
Driver name	Modbus Server
Assembly Name	T.ProtocolDriver.ModbusSlave
Assembly Version	2.0.0.9
Available for Linux	False
Devices supported	Any Client Modbus device
Manufacturer	N/A (Not Applicable)
Protocol	MODBUS RTU, ASCII and TCP
Interface	TCP/IP and Serial
Protocol Options	Message Format (ASCII, RTU or RTU TCP), SlaveID
Multi-threading	One thread with a pooling cycle for each Client connected

Max number of nodes	One node for each channel
PC Requirements	Standard PC Ethernet interface board, RS485 or RS232 port

Supported Operands

Operand	Read	Write	Data Type	Address size
0 – Coils	**Yes**	**Yes**	Bit	1 bit
1 – Input Status	**Yes**	-	Bit	1 bit
3 – Input Registers	**Yes**	-	Word	2 bytes
4 – Holding Registers	**Yes**	**Yes**	Word	2 bytes

Channels Configuration

Protocol Options

- **SlaveID:** Defines the driver Server address in the Modbus Network.
- **Encoding**: Determines how the information will be packed into the message fields and decoded. The options are:
  - **RTU**: Remote Terminal Unit mode, where each 8–bit byte in a message contains two 4–bit hexadecimal characters
  - **ASCII**: The message is encoded in ASCII mode, where each 8–bit byte in a message is sent as two ASCII characters
  - **RTU TCP**: The default transmission mode when the message is carried on a MODBUS TCP/IP network. It contains information to allow the recipient to recognize message boundaries even if the message has been split into multiple packets

Settings

Serial and MultiSerial channels:

- Default configuration for ASCII mode :
  - **DataBits**: 7
  - **StopBits**: 1 if parity is used, 2 if no parity
- Default configuration for RTU mode :
  - **DataBits**: 8
  - **StopBits**: 1 if parity is used, 2 if no parity

Set the other fields according to your Serial or MultiSerial port configuration

TCP/IP channels:

- **Listening Port**: Defines the TCP port where the driver will be listening for the connections, the default TCP port for the Modbus Network is **502** .

> **NOTE**
>
> You may need to configure your firewall to open the listening port.

Nodes Configuration

Station Configuration

There is no station configuration for the Modbus Server driver.

Point Configuration

The syntax for the Modbus communication points is: ***<Operand><Address>***

Where:

- ***<Operand>***: indicates the memory area
- ***<Address>***: indicates the data address in the memory area, from **1** to **65535**

Valid values for *<Operand>*
0 – Coils
1 – Input Status
3 – Input Registers
4 – Holding Registers

E.g.: ***400001*** (Operand = **Holding Register**, Address = ***1***)

Troubleshoot

Modbus protocol error codes		
Error	**Name**	**Description**
1	ILLEGAL FUNCTION	The function code received in the query is not allowable.
2	ILLEGAL DATA ADDRESS	The data address received in the query is not allowable.
3	ILLEGAL DATA VALUE	A value contained in the query data field is not allowable.
4	SERVER DEVICE FAILURE	Error while attempting to perform the requested action.
5	ACKNOWLEDGE	Request accepted, but a long duration of time will be required.
6	SERVER DEVICE BUSY	The server is engaged in a long–duration program command.
7	NEGATIVE ACKNOWLEDGE	Cannot perform the program function received in the query.
8	MEMORY PARITY ERROR	Parity error in the extended memory.

Driver Revision History

ModbusServer Revision History	
Version	**Notes**
2.0.0.9	Fixed: a Modbus Slave node with no protocol options configured (the default for a newly added node) failed to initialize; it now starts with the default message format.
2.0.0.7	Initial release on new documentation standards.

In this section...
