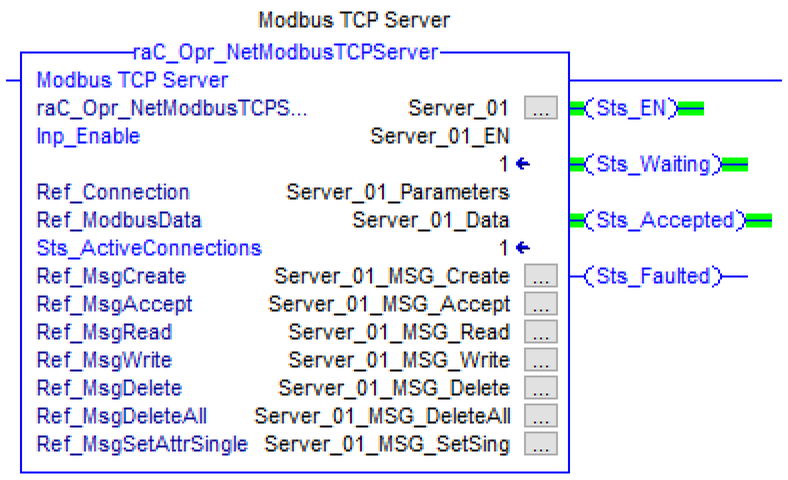
Modbus TCP Server Add-On Instruction based code for ControlLogix® and CompactLogix® controllers

Introduction

This document describes the application and use of the Modbus TCP Server Add-On Instruction.

Modbus TCP Server Add-On Instruction (AOI) allows users to implement Modbus TCP Server functionality into the Logix family of controllers. AOIs can be used standalone or can be added to an existing application following the directions outlined below.



```
Modbus TCP Server
raC_Opr_NetModbusTCPServer
Modbus TCP Server
raC_Opr_NetModbusTCPS...          Server_01 ...      (Sts_EN)
Inp_Enable                     Server_01_EN
                                          1 ←        (Sts_Waiting)
Ref_Connection            Server_01_Parameters
Ref_ModbusData                  Server_01_Data       (Sts_Accepted)
Sts_ActiveConnections                     1 ←
Ref_MsgCreate             Server_01_MSG_Create ...   (Sts_Faulted)
Ref_MsgAccept             Server_01_MSG_Accept ...
Ref_MsgRead                 Server_01_MSG_Read ...
Ref_MsgWrite               Server_01_MSG_Write ...
Ref_MsgDelete             Server_01_MSG_Delete ...
Ref_MsgDeleteAll       Server_01_MSG_DeleteAll ...
Ref_MsgSetAttrSingle   Server_01_MSG_SetSing ...
```

Contents

Requirements

Hardware Requirements

The Modbus TCP Server code requires a ControlLogix or CompactLogix controller with an EtherNet/IP module that supports Logix Sockets functionality. See Knowledgebase technote 470690 for complete list of controllers and modules.
https://rockwellautomation.custhelp.com/app/answers/detail/a_id/470690

Software Requirements

The Modbus TCP Server AOI code supports Logix controller revisions 20 and higher.

Memory Requirements

First instance of the Modbus TCP Server AOI uses about 123 Kbytes of memory.
Each additional AOI instance requires about 40 Kbytes of memory.
These estimates based on the ControlLogix 5570 family of controllers.
Please note that some Compactlogix controllers have a starting memory size as low as 384Kbytes. This code can take a significant amount of memory in smaller CompactLogix controllers.

Functional Requirements and Description

Supported Modbus Function Codes

Bit Level Commands

Function Code	Name	Description	Supported Values	Modbus Range
01	Read Coils	This function code is used by the Client to read contiguous status of coils in a remote device (0xxxx addresses). The coils in the response message are packed as one coil per bit of the data field.	Start Address: 0 to 2047 Length: 1 to 120 coils	00001-02048
02	Read Discrete Inputs	This function code is used by the Client to read contiguous status of discrete inputs in a remote device (1xxxx addresses). The inputs in the response message are packed as one coil per bit of the data field.	Start Address: 0 to 2047 Length: 1 to 120 Inputs	10001-12048
05	Write Single Coil	This function code is used by the Client to write a single coil. to either ON or OFF in a remote device. (0xxxx addresses).	Start Address: 0 to 2047	00001-02048
15	Write Multiple Coils	This function code is used by the Client to write one or more coils in a sequence of coils to either ON or OFF in a remote device. (0xxxx addresses).	Start Address: 0 to 2047 Length: 1 to 120 coils	00001-02048

Word Level Commands

Function Code	Name	Description	Supported Values	Modbus Range
03	Read Holding Registers	This function code is used by the Client to read the contents of a contiguous block of holding registers (4xxxx addresses) in a remote device.	Start Address: 0 to 1023 Length: 1 to 120 registers	40001-41024
04	Read Input Registers	This function code is used by the Client to read the contents of a contiguous block of input registers (3xxxx addresses) in a remote device.	Start Address: 0 to 1023 Length: 1 to 120 input registers	30001-31024
06	Write a Single Holding Register	This function code is used by the Client to write a single holding register (4xxxx addresses) in a remote device	Start Address: 0 to 1023	40001-41024
16	Write Multiple Holding Registers	This function code is used by the Client to write contiguous holding registers (4xxxx addresses) in a remote device.	Start Address: 0 to 1023 Length: 1 to 120 registers	40001-41024

Implementation

Modbus TCP Sever AOI implementation

Using Periodic Task

It’s recommended to add AOIs into a Periodic task with Rate of 10ms (or higher).
Slower rates will reduce controller load and reduce performance.
Faster task rates will increase performance but will add a significant load to the controller.

See Performance Data section for details.

Rung Import and tag naming changes

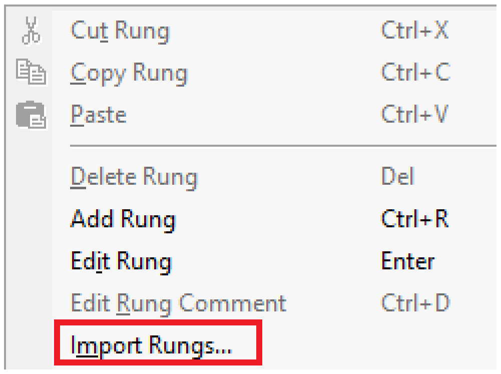
The pre-configured Add-On Instructions are supplied in a Rung format.

The Rung Import format must be used to implement the AOI.

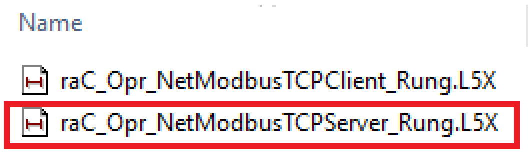
Important:
Use only the Rung Import process.
Do not use Copy/Paste functionality or add these AOIs using Instructions tool bar. Doing this will remove configurations from pre-configured message instructions, making AOIs non-functional.

Rung Import process for Modbus TCP Server AOI:

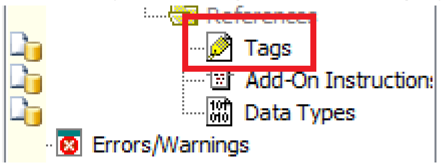
1. Open a Ladder Routine within your application
2. Right click on any empty area and select ***Import Rungs***



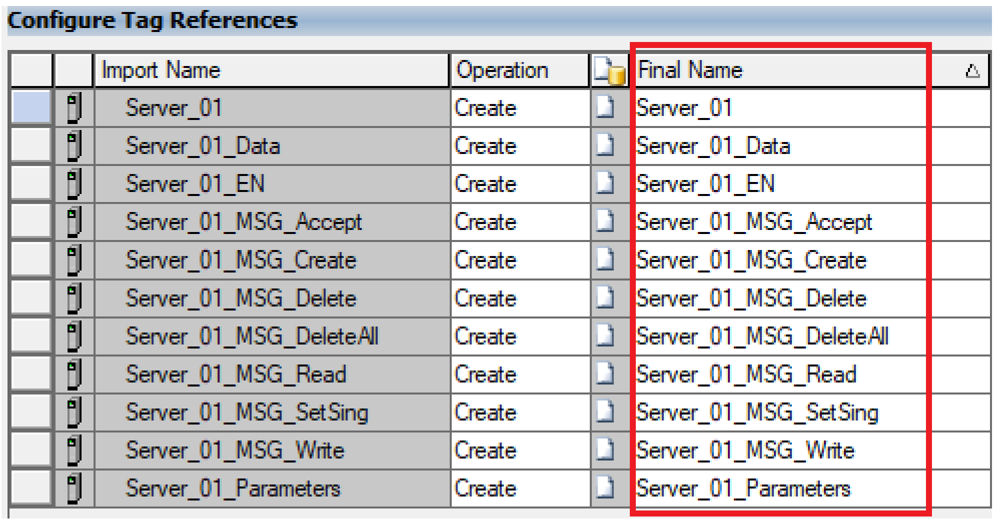
3. Select ***raC_Opr_NetModbusTCPServer_Rung.L5X*** file and click ***Import.***


Name
raC_Opr_NetModbusTCPClient_Rung.L5X
raC_Opr_NetModbusTCPServer_Rung.L5X

4. When Import Configuration Dialog opens, select ***Tags***



5. You can leave final names as-is or change them to accommodate your application.

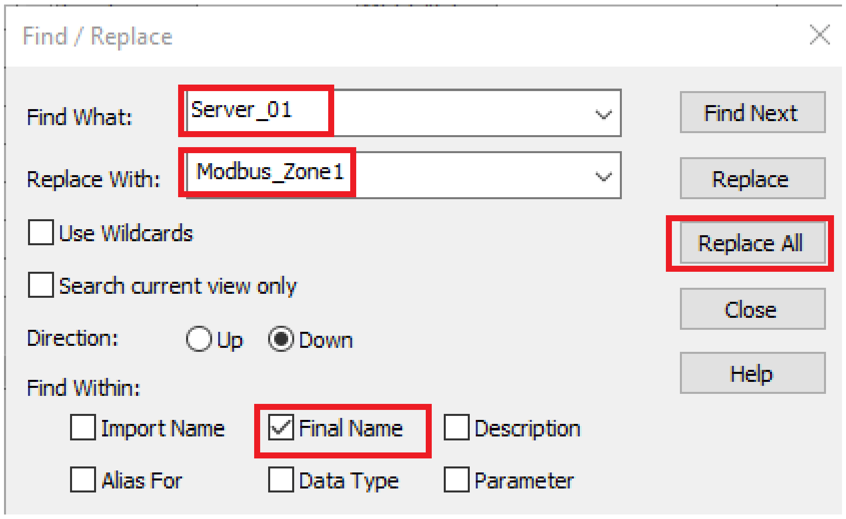


Configure Tag References

		Import Name	Operation		Final Name	
		Server_01	Create		Server_01	
		Server_01_Data	Create		Server_01_Data	
		Server_01_EN	Create		Server_01_EN	
		Server_01_MSG_Accept	Create		Server_01_MSG_Accept	
		Server_01_MSG_Create	Create		Server_01_MSG_Create	
		Server_01_MSG_Delete	Create		Server_01_MSG_Delete	
		Server_01_MSG_DeleteAll	Create		Server_01_MSG_DeleteAll	
		Server_01_MSG_Read	Create		Server_01_MSG_Read	
		Server_01_MSG_SetSing	Create		Server_01_MSG_SetSing	
		Server_01_MSG_Write	Create		Server_01_MSG_Write	
		Server_01_Parameters	Create		Server_01_Parameters	

6. To change Final Names click ***Find/Replace*** button

When Dialog opens, replace default name **Server_01** with desired prefix, verify that Final Names Box is checked then click ***Replace All***



Find / Replace

Find What: Server_01

Replace With: Modbus_Zone1

☐ Use Wildcards

☐ Search current view only

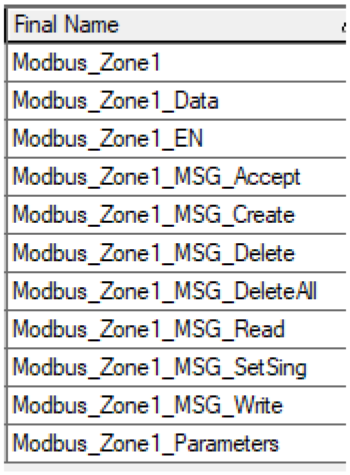
Direction: ○ Up ● Down

Find Within:

☐ Import Name ☑ Final Name ☐ Description

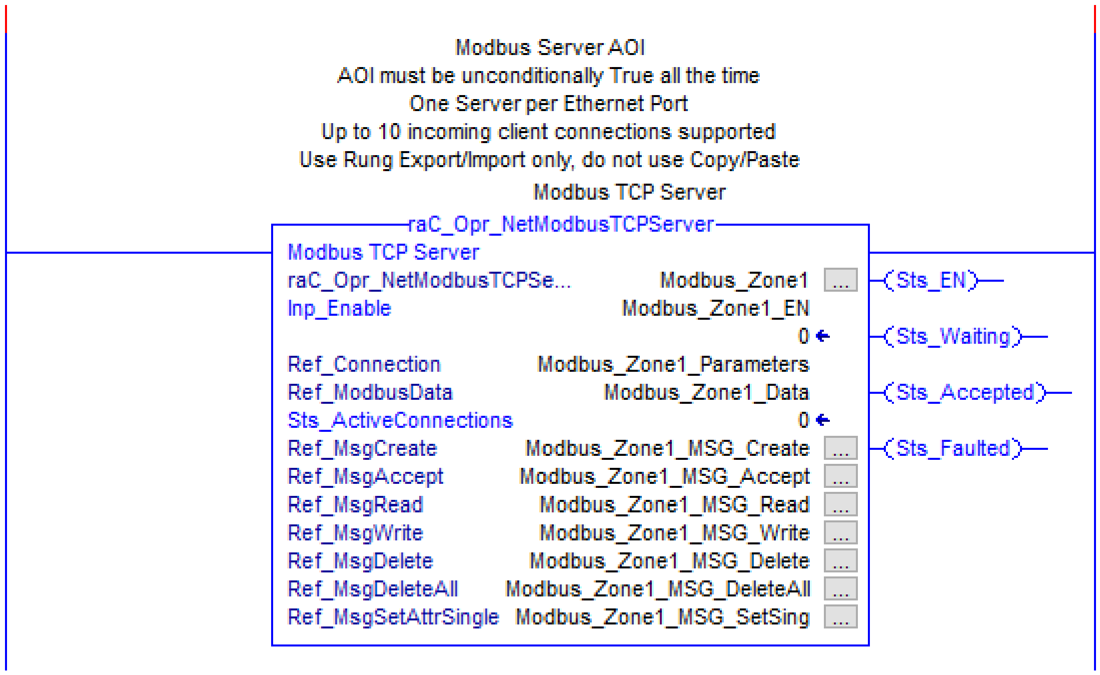
☐ Alias For ☐ Data Type ☐ Parameter

Close Find/Replace dialog and verify **Final Names**



Final Name
Modbus_Zone1
Modbus_Zone1_Data
Modbus_Zone1_EN
Modbus_Zone1_MSG_Accept
Modbus_Zone1_MSG_Create
Modbus_Zone1_MSG_Delete
Modbus_Zone1_MSG_DeleteAll
Modbus_Zone1_MSG_Read
Modbus_Zone1_MSG_SetSing
Modbus_Zone1_MSG_Write
Modbus_Zone1_Parameters

Click ***Ok*** to finish the Import process
New rung should look like shown below without any errors.



Modbus Server AOI
AOI must be unconditionally True all the time
One Server per Ethernet Port
Up to 10 incoming client connections supported
Use Rung Export/Import only, do not use Copy/Paste
Modbus TCP Server

raC_Opr_NetModbusTCPServer
Modbus TCP Server

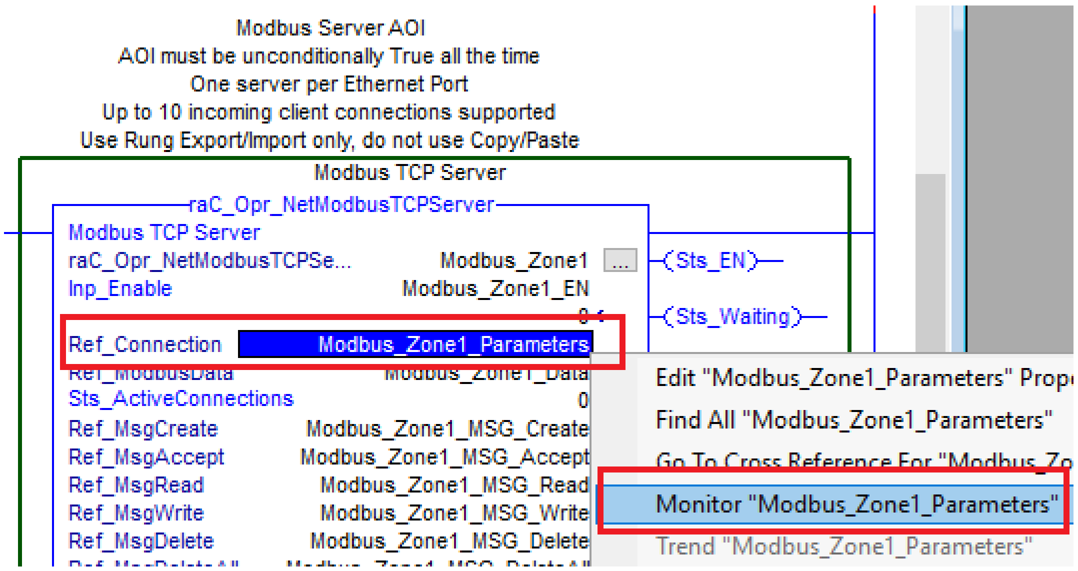
raC_Opr_NetModbusTCPSe...	Modbus_Zone1
Inp_Enable	Modbus_Zone1_EN
	0 ←
Ref_Connection	Modbus_Zone1_Parameters
Ref_ModbusData	Modbus_Zone1_Data
Sts_ActiveConnections	0 ←
Ref_MsgCreate	Modbus_Zone1_MSG_Create
Ref_MsgAccept	Modbus_Zone1_MSG_Accept
Ref_MsgRead	Modbus_Zone1_MSG_Read
Ref_MsgWrite	Modbus_Zone1_MSG_Write
Ref_MsgDelete	Modbus_Zone1_MSG_Delete
Ref_MsgDeleteAll	Modbus_Zone1_MSG_DeleteAll
Ref_MsgSetAttrSingle	Modbus_Zone1_MSG_SetSing

Configure Operational Parameters

Modbus TCP Server requires a local EtherNet/IP module that supports Logix Sockets. See Requirements section for details.

In this section we will link Modbus TCP Server AOI to this EtherNet/IP module.

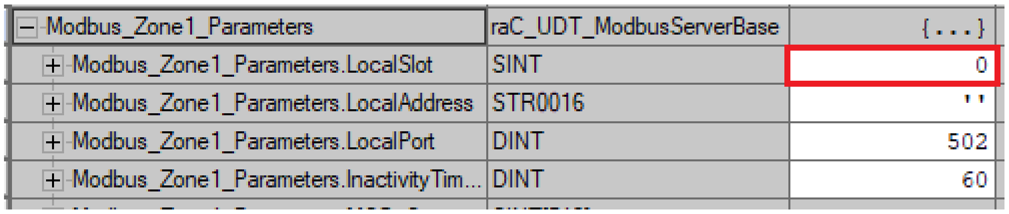
1. Right Click on the tag attached to the **Ref_Connection** parameter and select ***Monitor "…"***



Modbus Server AOI
AOI must be unconditionally True all the time
One server per Ethernet Port
Up to 10 incoming client connections supported
Use Rung Export/Import only, do not use Copy/Paste

Modbus TCP Server
raC_Opr_NetModbusTCPServer
Modbus TCP Server
raC_Opr_NetModbusTCPSe... Modbus_Zone1
Inp_Enable Modbus_Zone1_EN
Ref_Connection Modbus_Zone1_Parameters
Ref_ModbusData Modbus_Zone1_Data
Sts_ActiveConnections 0
Ref_MsgCreate Modbus_Zone1_MSG_Create
Ref_MsgAccept Modbus_Zone1_MSG_Accept
Ref_MsgRead Modbus_Zone1_MSG_Read
Ref_MsgWrite Modbus_Zone1_MSG_Write
Ref_MsgDelete Modbus_Zone1_MSG_Delete
Sts_EN
Sts_Waiting
Edit "Modbus_Zone1_Parameters" Prop
Find All "Modbus_Zone1_Parameters"
Go To Cross Reference For "Modbus_Zo
Monitor "Modbus_Zone1_Parameters"
Trend "Modbus_Zone1_Parameters"

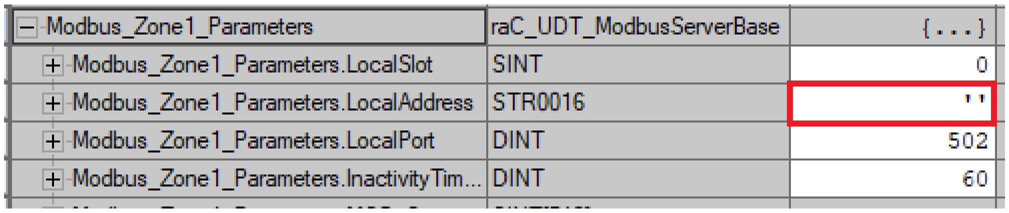
2. Expand Parameters tag. Specify the slot of the Local EtherNet/IP module.



Modbus_Zone1_Parameters	raC_UDT_ModbusServerBase	{...}
Modbus_Zone1_Parameters.LocalSlot	SINT	0
Modbus_Zone1_Parameters.LocalAddress	STR0016	''
Modbus_Zone1_Parameters.LocalPort	DINT	502
Modbus_Zone1_Parameters.InactivityTim...	DINT	60

For 1756 ControlLogix controllers specify the actual slot of desired 1756-EN2T(R) module.
For 1756-L8xE controllers using the built in Ethernet port specify the 1756-L8xE controller slot.
For CompactLogix 5370, 5380, 5480 controllers set the "**.LocalSlot"** to 0.

3. Specify the **".LocalAddress"** of the EtherNet/IP module.

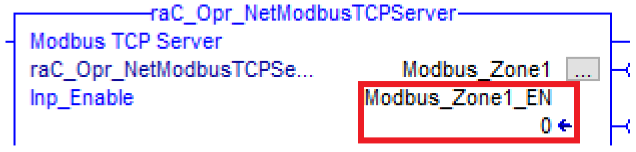


Modbus_Zone1_Parameters	raC_UDT_ModbusServerBase	{...}
Modbus_Zone1_Parameters.LocalSlot	SINT	0
Modbus_Zone1_Parameters.LocalAddress	STR0016	''
Modbus_Zone1_Parameters.LocalPort	DINT	502
Modbus_Zone1_Parameters.InactivityTim...	DINT	60

For CompactLogix 5380 and 5480 controllers **in Dual IP mode** only, specify the IP Address of the Local Ethernet connection used for Modbus TCP communications. **Leave this field blank for all other cases.**

4. Leave Default Modbus TCP port at 502. This value is Modbus TCP protocol standard.
5. If you change any of these Parameters during operation be sure to reset and then set the AOI **Inp_Enable** parameter tag.

6. Start Modbus TCP Server by setting tag attached to **Inp_Enable** parameter to 1.



Implementation Restrictions

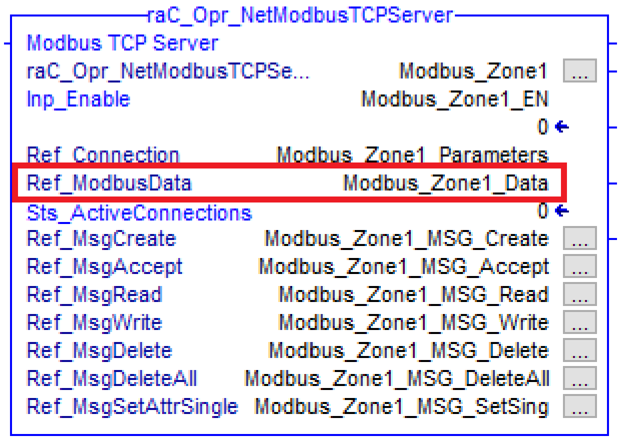
1. Implementation must be done using Import Rung function only to preserve Message instruction configurations. Do not use Copy/Paste as it will not bring complete Message instruction configurations and tags. Do not use Search/Replace tags once rung is implemented. All replacement can be done only during rung import.

2. Only one Server AOI is supported per CompactLogix controller (5370, 5380, 5480). ControlLogix Controllers (1756) can have one server per each 1756-EN2T(R) module used, but each instance must use own set of data tags.

3. Modbus TCP Server and Modbus TCP Client AOIs can reside in the same program. However Server applications may cause a temporary Client disconnection due to the shared Logix Sockets object.

Monitoring Modbus TCP Server operations

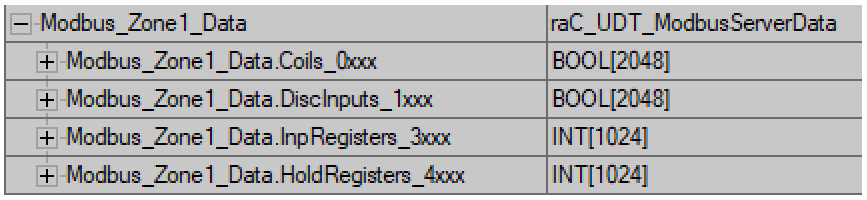
1. Modbus Tags are located under **Ref_ModbusData** parameter tag.



raC_Opr_NetModbusTCPServer
Modbus TCP Server

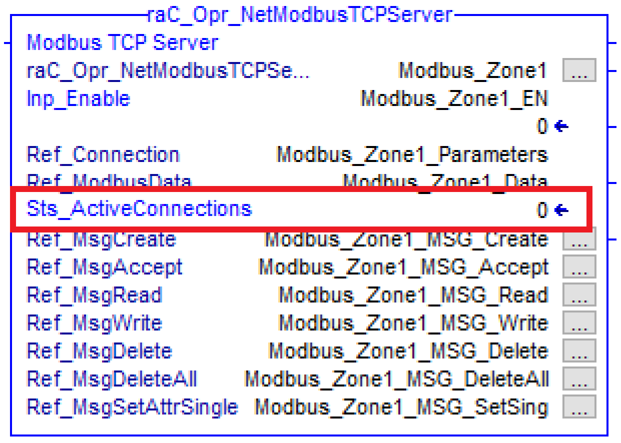
raC_Opr_NetModbusTCPSe...	Modbus_Zone1
Inp_Enable	Modbus_Zone1_EN
	0
Ref_Connection	Modbus_Zone1_Parameters
Ref_ModbusData	Modbus_Zone1_Data
Sts_ActiveConnections	0
Ref_MsgCreate	Modbus_Zone1_MSG_Create
Ref_MsgAccept	Modbus_Zone1_MSG_Accept
Ref_MsgRead	Modbus_Zone1_MSG_Read
Ref_MsgWrite	Modbus_Zone1_MSG_Write
Ref_MsgDelete	Modbus_Zone1_MSG_Delete
Ref_MsgDeleteAll	Modbus_Zone1_MSG_DeleteAll
Ref_MsgSetAttrSingle	Modbus_Zone1_MSG_SetSing

This tag contains four separated data areas for coils (0xxxx), discrete inputs (1xxxx), input registers (3xxxx) and holding registers (4xxxx). These tag values can be read and populated by the user application without any restrictions.



Modbus_Zone1_Data	raC_UDT_ModbusServerData
Modbus_Zone1_Data.Coils_0xxx	BOOL[2048]
Modbus_Zone1_Data.DiscInputs_1xxx	BOOL[2048]
Modbus_Zone1_Data.InpRegisters_3xxx	INT[1024]
Modbus_Zone1_Data.HoldRegisters_4xxx	INT[1024]

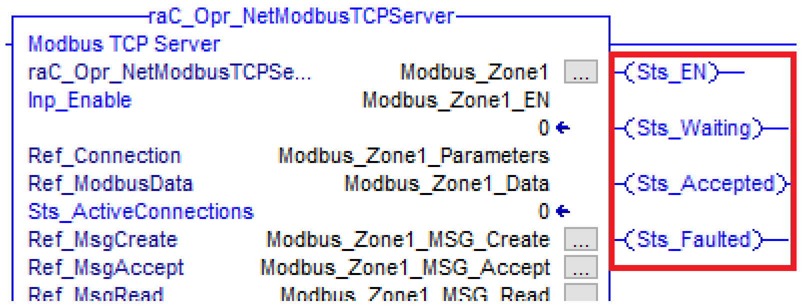
2. The **Sts_ActiveConnections** parameter indicates the number of currently active Client connections.



raC_Opr_NetModbusTCPServer
Modbus TCP Server

raC_Opr_NetModbusTCPSe...	Modbus_Zone1
Inp_Enable	Modbus_Zone1_EN
	0
Ref_Connection	Modbus_Zone1_Parameters
Ref_ModbusData	Modbus_Zone1_Data
Sts_ActiveConnections	0
Ref_MsgCreate	Modbus_Zone1_MSG_Create
Ref_MsgAccept	Modbus_Zone1_MSG_Accept
Ref_MsgRead	Modbus_Zone1_MSG_Read
Ref_MsgWrite	Modbus_Zone1_MSG_Write
Ref_MsgDelete	Modbus_Zone1_MSG_Delete
Ref_MsgDeleteAll	Modbus_Zone1_MSG_DeleteAll
Ref_MsgSetAttrSingle	Modbus_Zone1_MSG_SetSing

3. Status Bits



```
raC_Opr_NetModbusTCPServer
Modbus TCP Server
raC_Opr_NetModbusTCPSe...          Modbus_Zone1 ...       -(Sts_EN)-
Inp_Enable                         Modbus_Zone1_EN
                                              0 ←         -(Sts_Waiting)-
Ref_Connection                Modbus_Zone1_Parameters
Ref_ModbusData                     Modbus_Zone1_Data      -(Sts_Accepted)-
Sts_ActiveConnections                         0 ←
Ref_MsgCreate                 Modbus_Zone1_MSG_Create ... -(Sts_Faulted)-
Ref_MsgAccept                 Modbus_Zone1_MSG_Accept ...
Ref_MsgRead                   Modbus_Zone1_MSG_Read
```

a. **Sts_EN** output indicates that Modbus TCP Server functionality is enabled.
b. **Sts_Waiting** output indicates that Server is ready to accept an incoming connection request from a Client.
c. **Sts_Accepted** output indicates that at least one Client connection request is accepted and servicing data requests.
d. **Sts_Faulted** output indicates that one of the message instructions is faulted.

Performance data

Modbus Server performance can be affected by many factors including: periodic task rate, performance of the Client device, speed of Server controller, how busy the Server controller is, network performance, network card, number of Clients connected to the Server, the number of active transactions etc.

The Sever can affect the performance of the data delivery in the Client based on the following factors:

- Number of connected Clients
- Number of active Transactions per Client
- Server Periodic Scan Rate
- It requires two Periodic task scans to service each message

The Server can delay data delivery by as much as the following formula:

(Total number of Transaction) x 2 Scans x Periodic task rate

Example:

Assumptions:

- The Server has one Client connection with 3 active transactions
- The Server has a second Client connection with 4 active transactions
- The Server Periodic Task rate is the default 10 mS

(Client 1, 3 transactions + Client 2, 4 Transactions) x 2 Scans per transaction x 10 mS Periodic Rate

(3 + 4) x 10 x 2 = 140msec

So in this example the Server can delay the actual data transmission by as much as 140 msec.

Revision History

1. Revision 1.02 – Initial Release in Ladder Program format. If you are currently using this code in an existing application, you may continue to do so.

2. Server Revision 2.00.00 – Initial Release in Add-On Instruction format. This version is recommended for use in all new applications.
   - 2.1. Enhancements
     - 2.1.1. Re-written code in Add-On instruction format
     - 2.1.2. Reduced memory requirements
     - 2.1.3. Simplified implementation and configuration
   - 2.2. Corrected Anomalies
     - 2.2.1. None
   - 2.3. Known Anomalies
     - 2.3.1.None

3. Server Revision 2.00.01 - Update
   - 3.1. Enhancements
     - 3.1.1. None
   - 3.2. Corrected Anomalies
     - 3.2.1. Minor logic correction related to the local IP Address
   - 3.3. Known Anomalies
     - 3.3.1.None