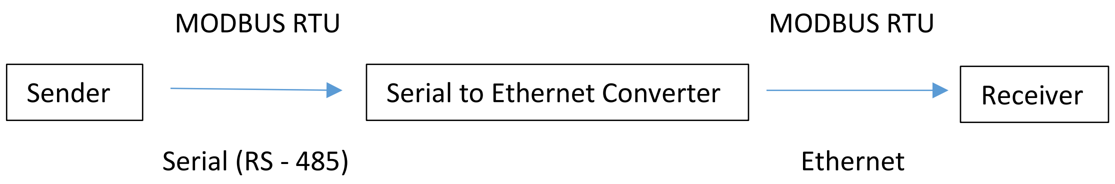
MODBUS RTU OVER TCP Server Function Block Configuration for LOGIX 5000 controllers

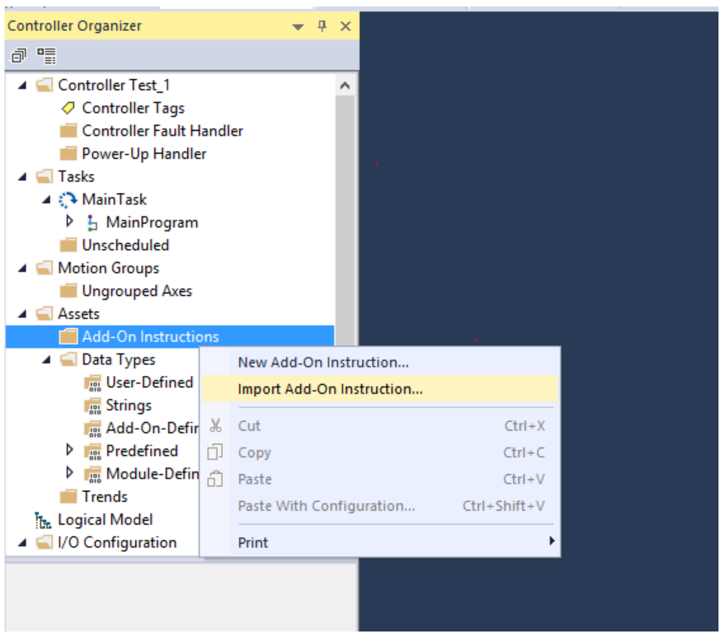
Introduction

This function block is defined to have Modbus communication with devices which are using Modbus RTU over TCP protocol (Not Modbus TCP). Here Message sent by the client is MODBUS RTU type but it is sent through Ethernet. Serial to Ethernet converters like MOXA Nport devices use this kind of communication which are not converting the RTU message to TCP. Here for this Function Block PLC act as a client and the other device act as Master.



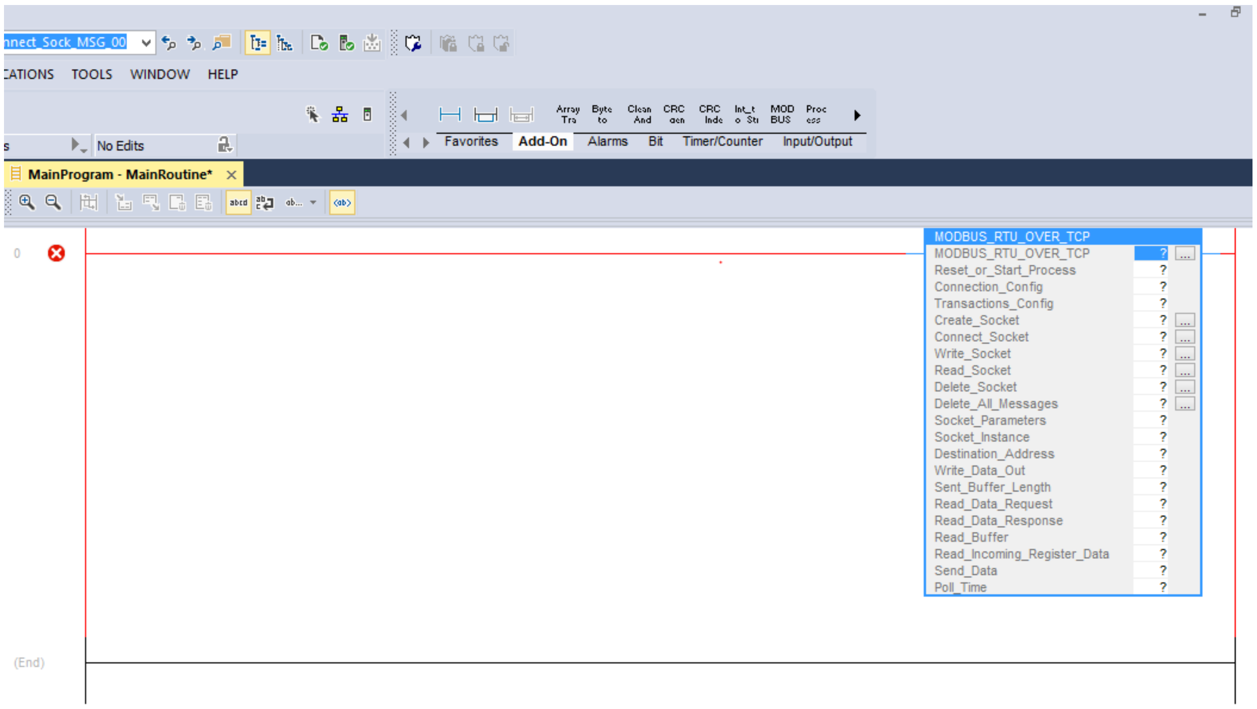
Importing the Sample Block to the Project

1. Connect the device to PLC. (Serial to Ethernet converter).
2. Connect the PLC to pc.
3. Power up devices.
4. Make sure that the PLC and the Ethernet device IP addresses are in same range.
5. Open a new Studio 5000 project with Selecting the particular PLC.
6. Then go to Assets/Add-On Instructions/right click – Import Add-On Instruction



7. Select the block and Import.

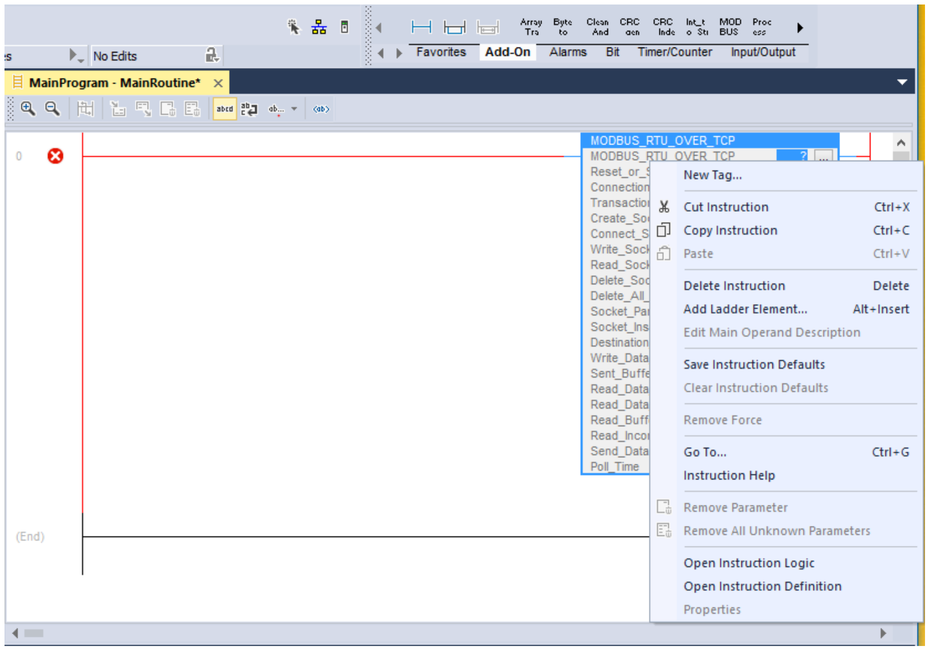
8. In the main routine select Add-On/ MODBUS_RTU_OVER_TCP



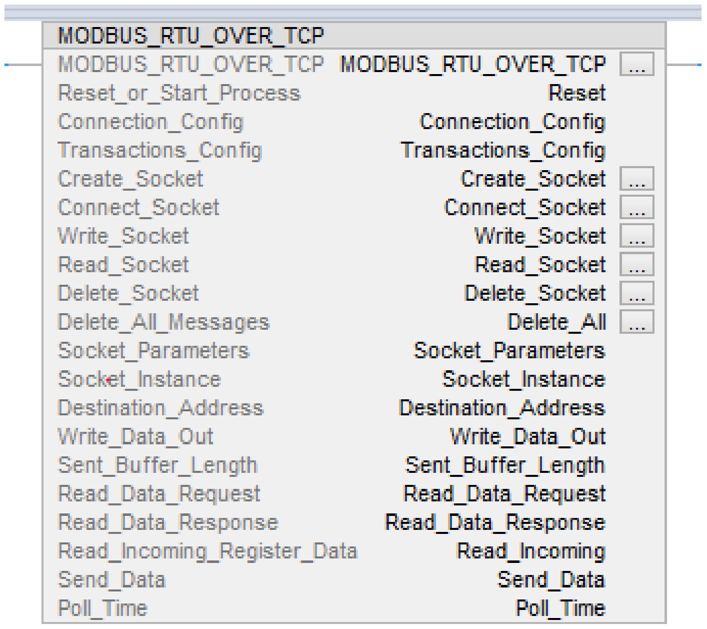
```
MODBUS_RTU_OVER_TCP
MODBUS_RTU_OVER_TCP            ?
Reset_or_Start_Process         ?
Connection_Config              ?
Transactions_Config            ?
Create_Socket                  ?
Connect_Socket                 ?
Write_Socket                   ?
Read_Socket                    ?
Delete_Socket                  ?
Delete_All_Messages            ?
Socket_Parameters              ?
Socket_Instance                ?
Destination_Address            ?
Write_Data_Out                 ?
Sent_Buffer_Length             ?
Read_Data_Request              ?
Read_Data_Response             ?
Read_Buffer                    ?
Read_Incoming_Register_Data    ?
Send_Data                      ?
Poll_Time                      ?
```

Function Block Configuration

1. Create new tag for the each parameter of the block
   Right click - New tag - Change the name – Create
   (No need to do any other modifications, just put a name for the variable and create)

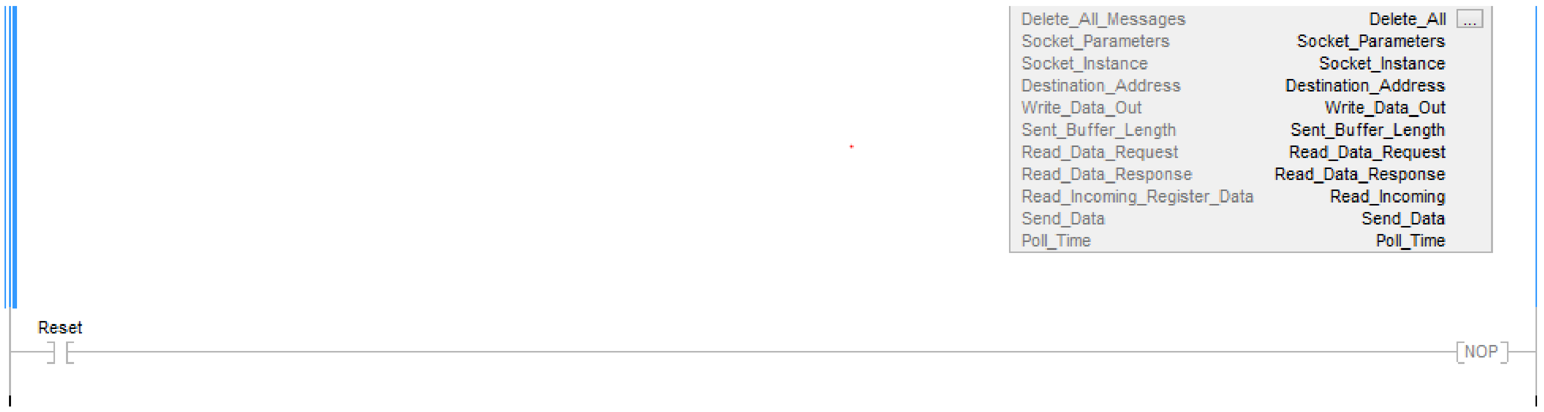


2. Do the same procedure for all the other parameters.



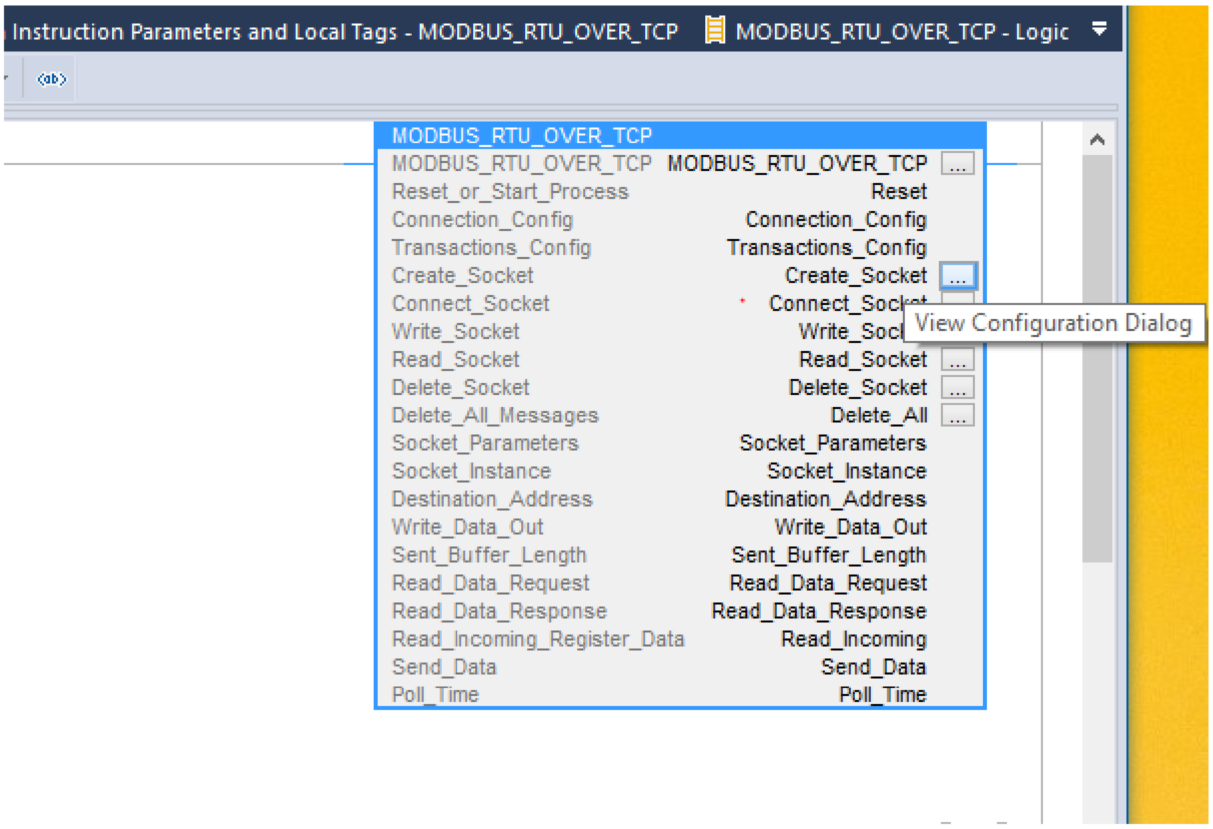
MODBUS_RTU_OVER_TCP	
MODBUS_RTU_OVER_TCP	MODBUS_RTU_OVER_TCP
Reset_or_Start_Process	Reset
Connection_Config	Connection_Config
Transactions_Config	Transactions_Config
Create_Socket	Create_Socket
Connect_Socket	Connect_Socket
Write_Socket	Write_Socket
Read_Socket	Read_Socket
Delete_Socket	Delete_Socket
Delete_All_Messages	Delete_All
Socket_Parameters	Socket_Parameters
Socket_Instance	Socket_Instance
Destination_Address	Destination_Address
Write_Data_Out	Write_Data_Out
Sent_Buffer_Length	Sent_Buffer_Length
Read_Data_Request	Read_Data_Request
Read_Data_Response	Read_Data_Response
Read_Incoming_Register_Data	Read_Incoming
Send_Data	Send_Data
Poll_Time	Poll_Time

3. Create New Rung and Put a switch with the Variable reset (Related to Reset_or_Start_Process Parameter) and add a coil and Change its type to NOP.



Delete_All_Messages	Delete_All
Socket_Parameters	Socket_Parameters
Socket_Instance	Socket_Instance
Destination_Address	Destination_Address
Write_Data_Out	Write_Data_Out
Sent_Buffer_Length	Sent_Buffer_Length
Read_Data_Request	Read_Data_Request
Read_Data_Response	Read_Data_Response
Read_Incoming_Register_Data	Read_Incoming
Send_Data	Send_Data
Poll_Time	Poll_Time

4. Now navigate to configuration of Create_Socket by clicking the tab.

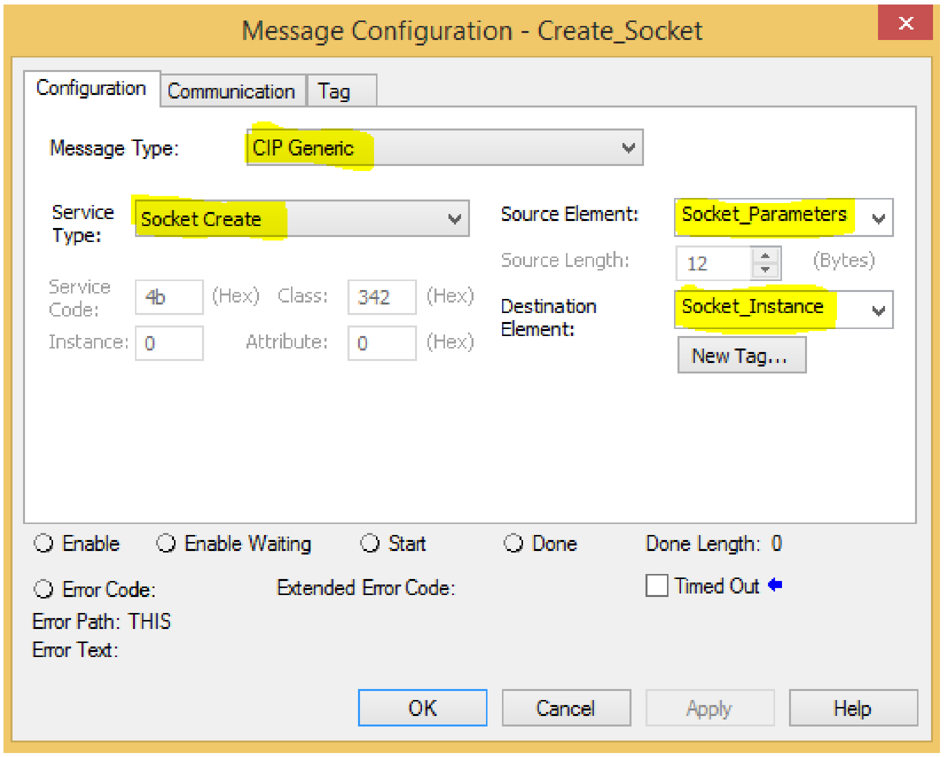


Instruction Parameters and Local Tags - MODBUS_RTU_OVER_TCP
MODBUS_RTU_OVER_TCP - Logic
View Configuration Dialog

MODBUS_RTU_OVER_TCP

MODBUS_RTU_OVER_TCP	MODBUS_RTU_OVER_TCP
Reset_or_Start_Process	Reset
Connection_Config	Connection_Config
Transactions_Config	Transactions_Config
Create_Socket	Create_Socket
Connect_Socket	Connect_Socket
Write_Socket	Write_Socket
Read_Socket	Read_Socket
Delete_Socket	Delete_Socket
Delete_All_Messages	Delete_All
Socket_Parameters	Socket_Parameters
Socket_Instance	Socket_Instance
Destination_Address	Destination_Address
Write_Data_Out	Write_Data_Out
Sent_Buffer_Length	Sent_Buffer_Length
Read_Data_Request	Read_Data_Request
Read_Data_Response	Read_Data_Response
Read_Incoming_Register_Data	Read_Incoming
Send_Data	Send_Data
Poll_Time	Poll_Time

5. Set the Configurations as in figure.(reference to the variables created)



Message Configuration - Create_Socket

Configuration | Communication | Tag

Message Type: CIP Generic

Service Type: Socket Create

Service Code: 4b (Hex) Class: 342 (Hex)

Instance: 0 Attribute: 0 (Hex)

Source Element: Socket_Parameters

Source Length: 12 (Bytes)

Destination Element: Socket_Instance

New Tag...

Enable | Enable Waiting | Start | Done | Done Length: 0

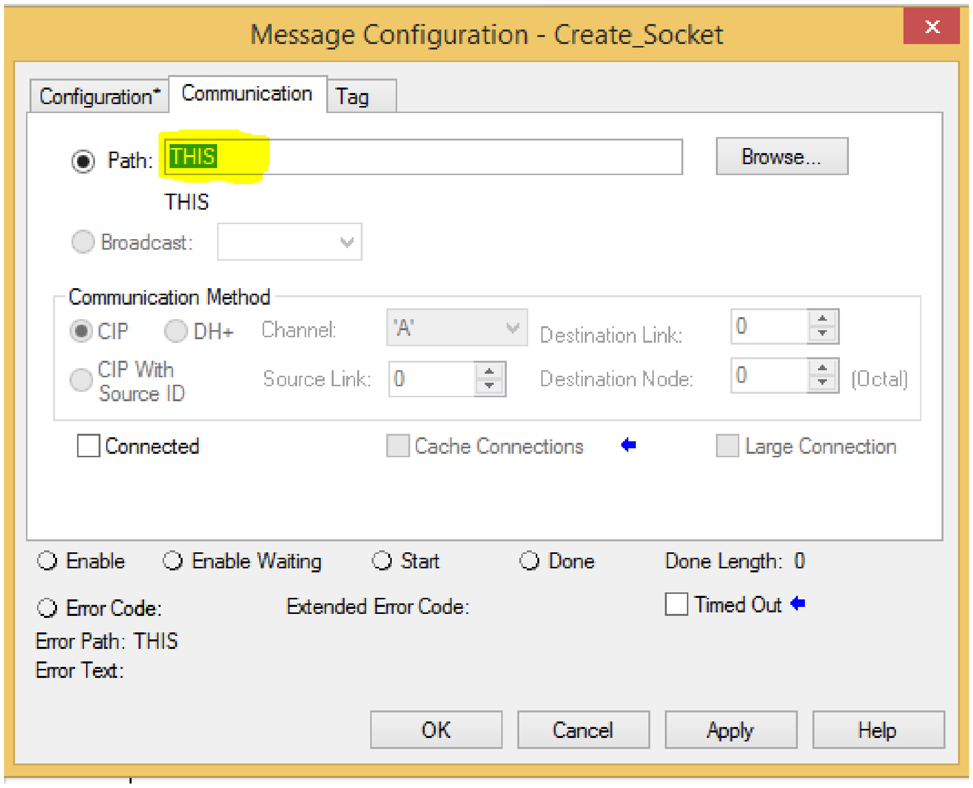
Error Code: Extended Error Code: Timed Out

Error Path: THIS

Error Text:

OK | Cancel | Apply | Help

6. Set the Communication Path as in Figure. Then give OK.



Message Configuration - Create_Socket

Configuration* | Communication | Tag

Path: THIS

THIS

Broadcast:

Communication Method

CIP | DH+ | Channel: 'A' | Destination Link: 0

CIP With Source ID | Source Link: 0 | Destination Node: 0 (Octal)

Connected | Cache Connections | Large Connection

Enable | Enable Waiting | Start | Done | Done Length: 0

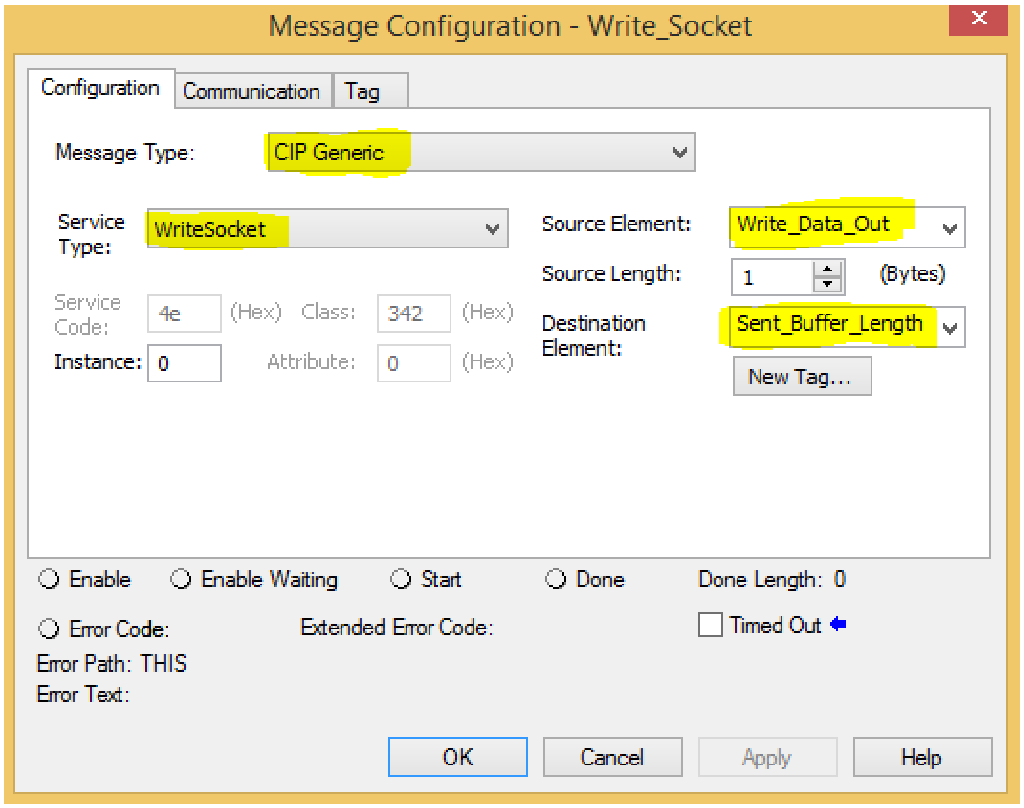
Error Code: Extended Error Code: Timed Out

Error Path: THIS

Error Text:

OK | Cancel | Apply | Help

7. As above Set all the message Instruction parameters as below. **<u>All the Communication Paths are same as above figure. For the Instance put any value.</u>**



Message Configuration - Write_Socket

Configuration | Communication | Tag

Message Type: CIP Generic

Service Type: WriteSocket

Service Code: 4e (Hex) Class: 342 (Hex)

Instance: 0 Attribute: 0 (Hex)

Source Element: Write_Data_Out

Source Length: 1 (Bytes)

Destination Element: Sent_Buffer_Length

New Tag...

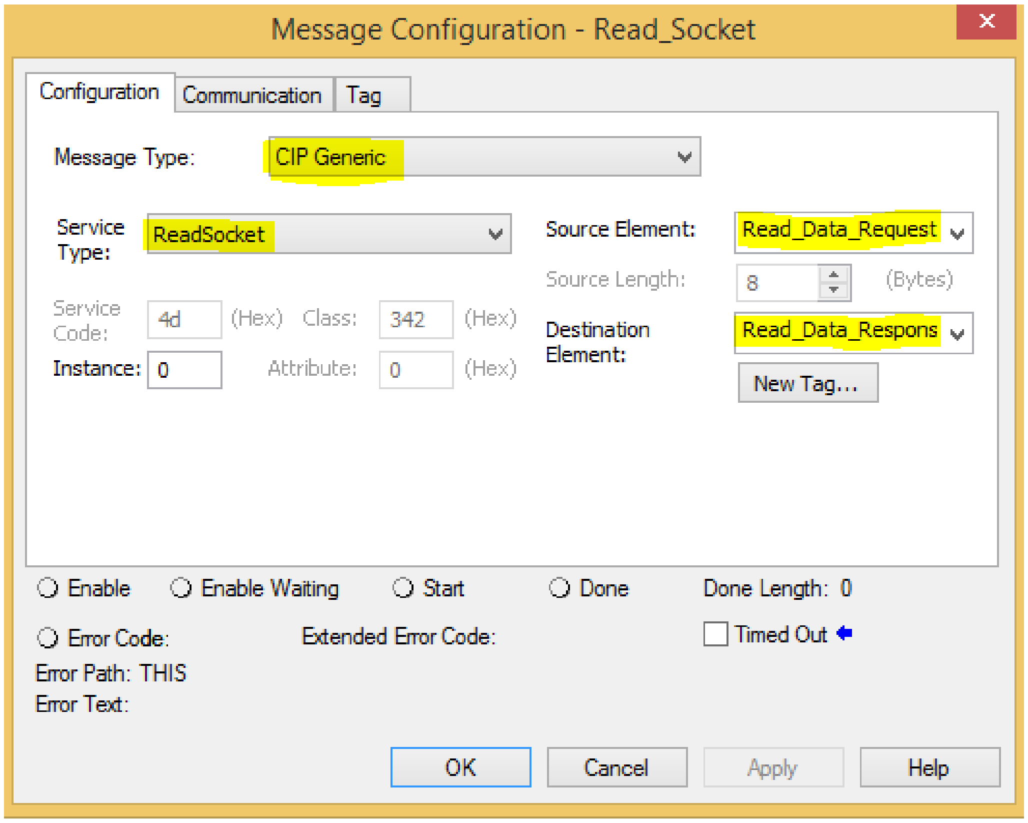
Enable | Enable Waiting | Start | Done | Done Length: 0

Error Code: Extended Error Code: Timed Out

Error Path: THIS

Error Text:

OK | Cancel | Apply | Help



Message Configuration - Read_Socket

Configuration | Communication | Tag

Message Type: CIP Generic

Service Type: ReadSocket

Service Code: 4d (Hex) Class: 342 (Hex)

Instance: 0 Attribute: 0 (Hex)

Source Element: Read_Data_Request

Source Length: 8 (Bytes)

Destination Element: Read_Data_Respons

New Tag...

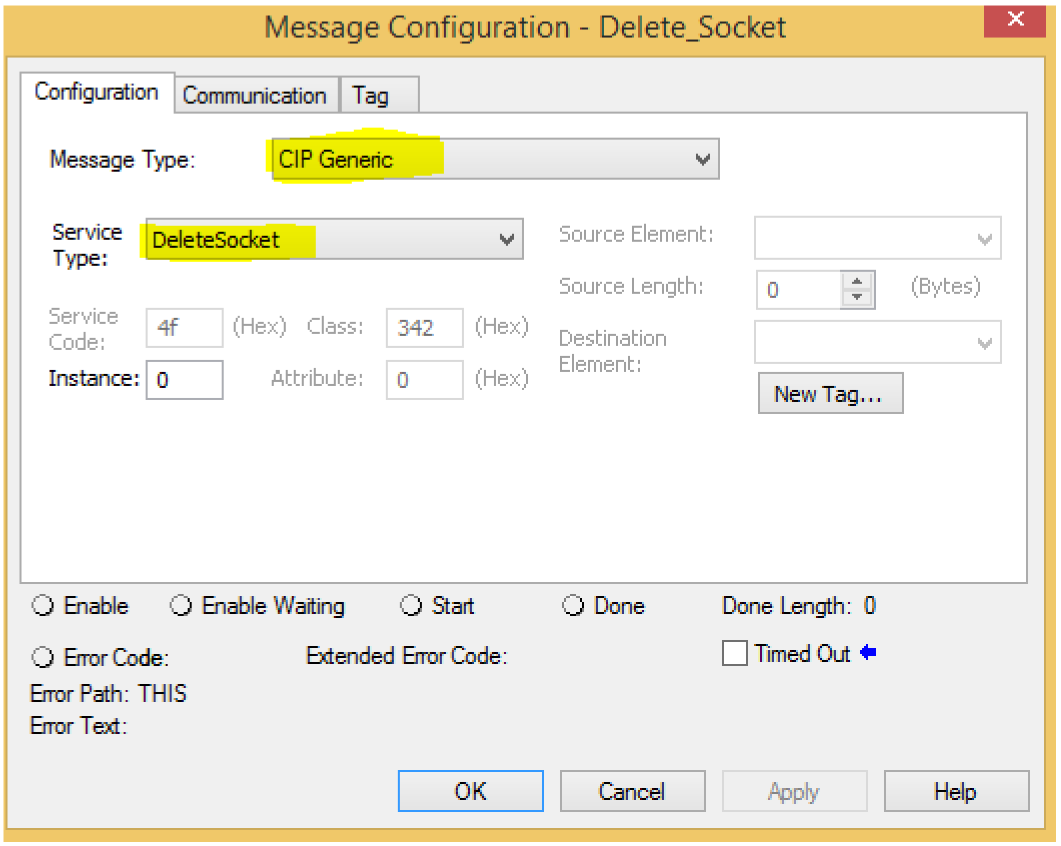
Enable | Enable Waiting | Start | Done | Done Length: 0

Error Code: Extended Error Code: Timed Out

Error Path: THIS

Error Text:

OK | Cancel | Apply | Help



Message Configuration - Delete_Socket

Configuration | Communication | Tag

Message Type: CIP Generic

Service Type: DeleteSocket

Service Code: 4f (Hex) Class: 342 (Hex)

Instance: 0 Attribute: 0 (Hex)

Source Element:

Source Length: 0 (Bytes)

Destination Element:

New Tag...

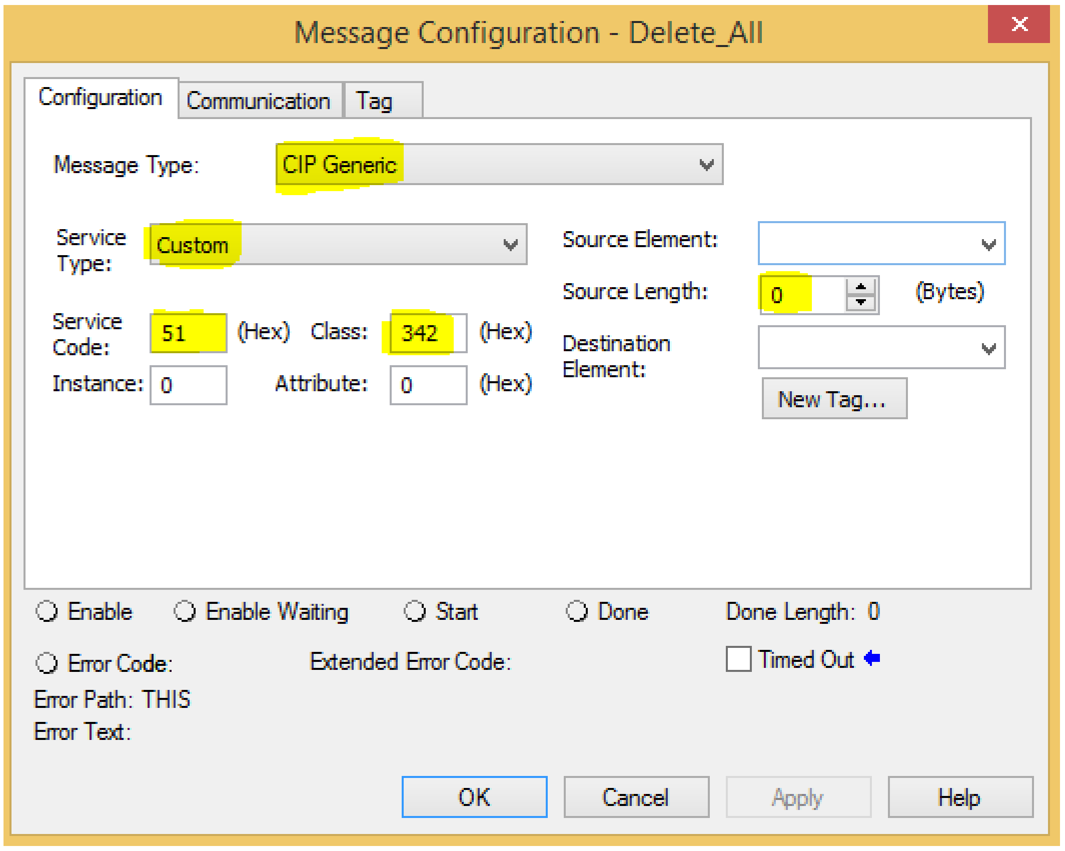
Enable Enable Waiting Start Done Done Length: 0

Error Code: Extended Error Code: Timed Out

Error Path: THIS

Error Text:

OK Cancel Apply Help



Message Configuration - Delete_All

Configuration | Communication | Tag

Message Type: CIP Generic

Service Type: Custom

Service Code: 51 (Hex) Class: 342 (Hex)

Instance: 0 Attribute: 0 (Hex)

Source Element:

Source Length: 0 (Bytes)

Destination Element:

New Tag...

Enable Enable Waiting Start Done Done Length: 0

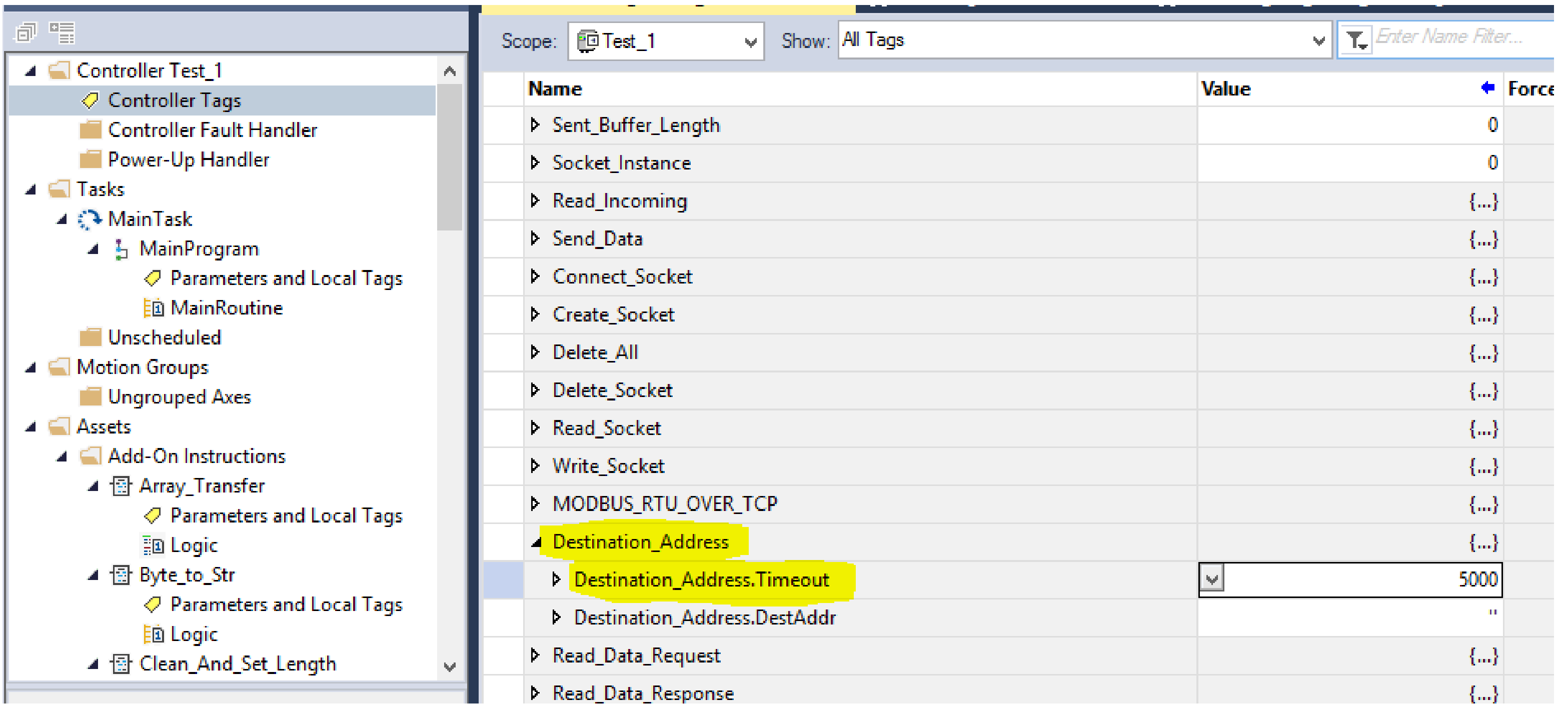
Error Code: Extended Error Code: Timed Out

Error Path: THIS

Error Text:

OK Cancel Apply Help

8. Navigate to **Controller tags – Destination_Address(variable related to Destination_Address Parameter)- Destination_Address.timeout**
Set this value to preferable value.



Controller Test_1
- Controller Tags
- Controller Fault Handler
- Power-Up Handler

Tasks
- MainTask
  - MainProgram
    - Parameters and Local Tags
    - MainRoutine
- Unscheduled

Motion Groups
- Ungrouped Axes

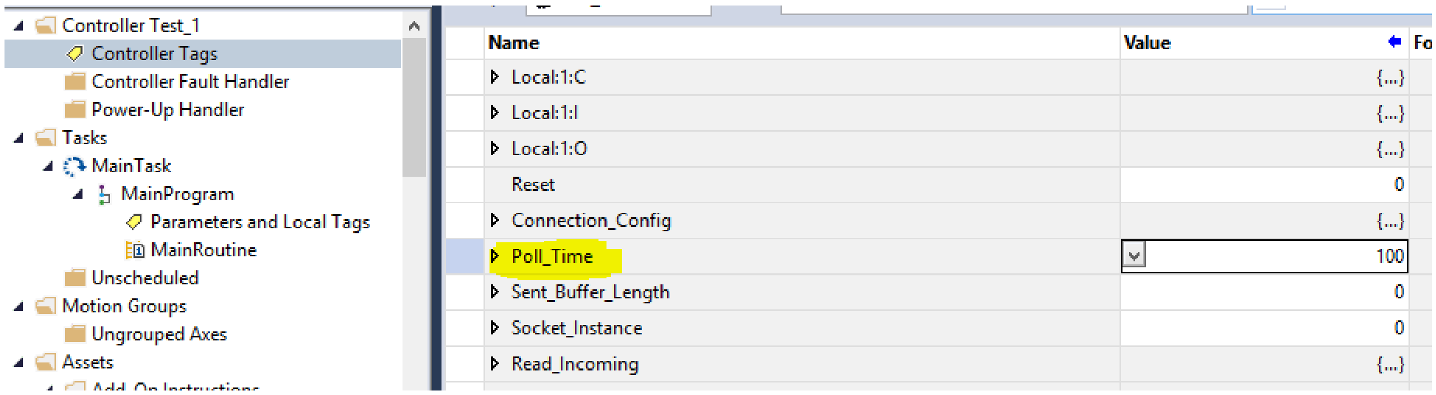
Assets
- Add-On Instructions
  - Array_Transfer
    - Parameters and Local Tags
    - Logic
  - Byte_to_Str
    - Parameters and Local Tags
    - Logic
  - Clean_And_Set_Length

Scope: Test_1 Show: All Tags

Name	Value
Sent_Buffer_Length	0
Socket_Instance	0
Read_Incoming	{...}
Send_Data	{...}
Connect_Socket	{...}
Create_Socket	{...}
Delete_All	{...}
Delete_Socket	{...}
Read_Socket	{...}
Write_Socket	{...}
MODBUS_RTU_OVER_TCP	{...}
Destination_Address	{...}
Destination_Address.Timeout	5000
Destination_Address.DestAddr	''
Read_Data_Request	{...}
Read_Data_Response	{...}

9. Navigate to **Controller tags – Poll time**

This value is the time interval between two adjacent transactions. <u>Set this value to 100 or greater than that (value >= 100(ms))</u>



Controller Test_1
- Controller Tags
- Controller Fault Handler
- Power-Up Handler

Tasks
- MainTask
  - MainProgram
    - Parameters and Local Tags
    - MainRoutine
- Unscheduled

Motion Groups
- Ungrouped Axes

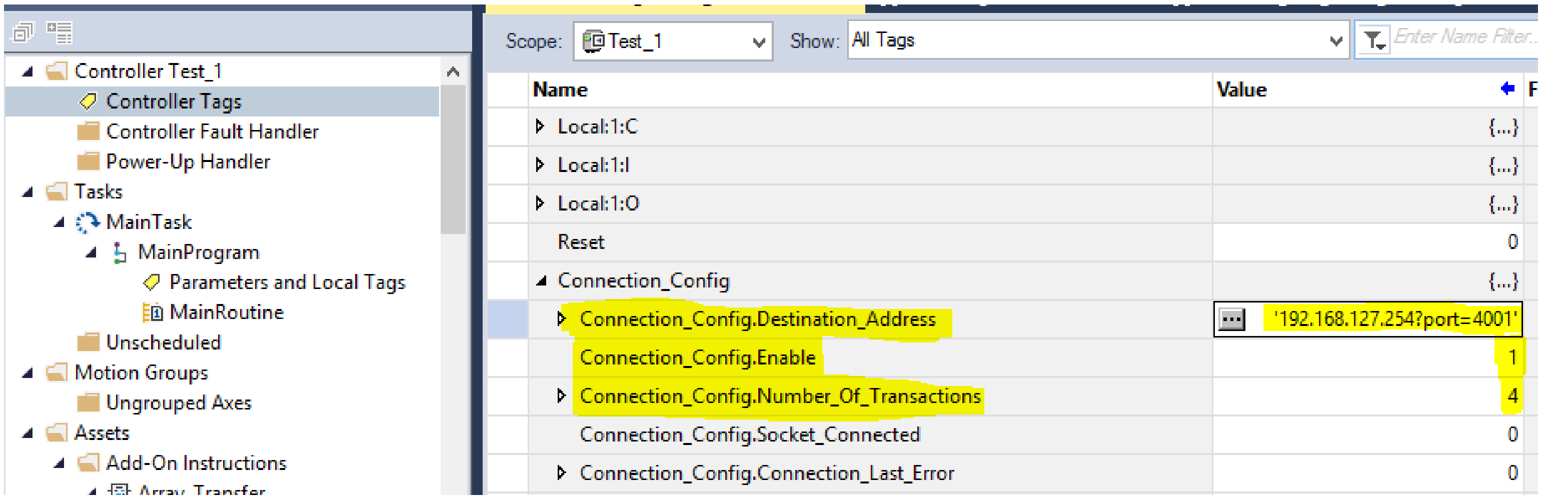
Assets

Name	Value
Local:1:C	{...}
Local:1:I	{...}
Local:1:O	{...}
Reset	0
Connection_Config	{...}
Poll_Time	100
Sent_Buffer_Length	0
Socket_Instance	0
Read_Incoming	{...}

Connection Block Configuration

1. Navigate to **Controller tags – Connection_Config(Variable related to Connection_Config Parameter)**

2. Set the values to the highlighted Variables as in figure.
   - **Destination_Address =** IP address and the Port of the Device which is connected to PLC. **Set the values Exactly as in figure**.(MOXA device or any other converter)
   - **Enable =** Set this to 1 to enable the connection. If this is 0 connection will not be established. You can break the connection by setting this to 0 while program is running.
   - **Number_Of_Transactions =** Number of Modbus devices that are connected on the other side by Serial (RS_485).



Scope: Test_1 Show: All Tags

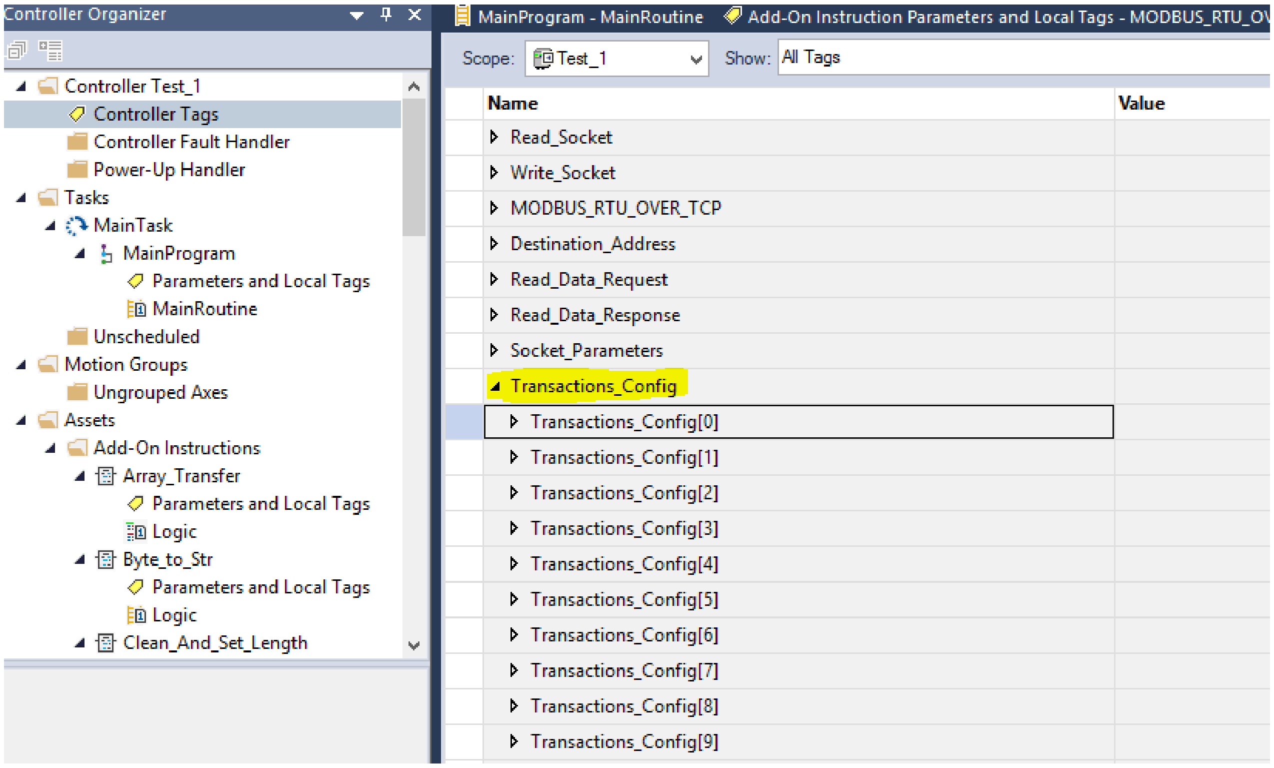
Name	Value
Local:1:C	{...}
Local:1:I	{...}
Local:1:O	{...}
Reset	0
Connection_Config	{...}
Connection_Config.Destination_Address	'192.168.127.254?port=4001'
Connection_Config.Enable	1
Connection_Config.Number_Of_Transactions	4
Connection_Config.Socket_Connected	0
Connection_Config.Connection_Last_Error	0

- Socket_Connected = This will indicated the status whether the device is connected or not.
- Connection_Last_Error = This will indicates the last error thrown by program.

Error Codes :
1 – Connection Error
2 – Write Error
3 – Read Error

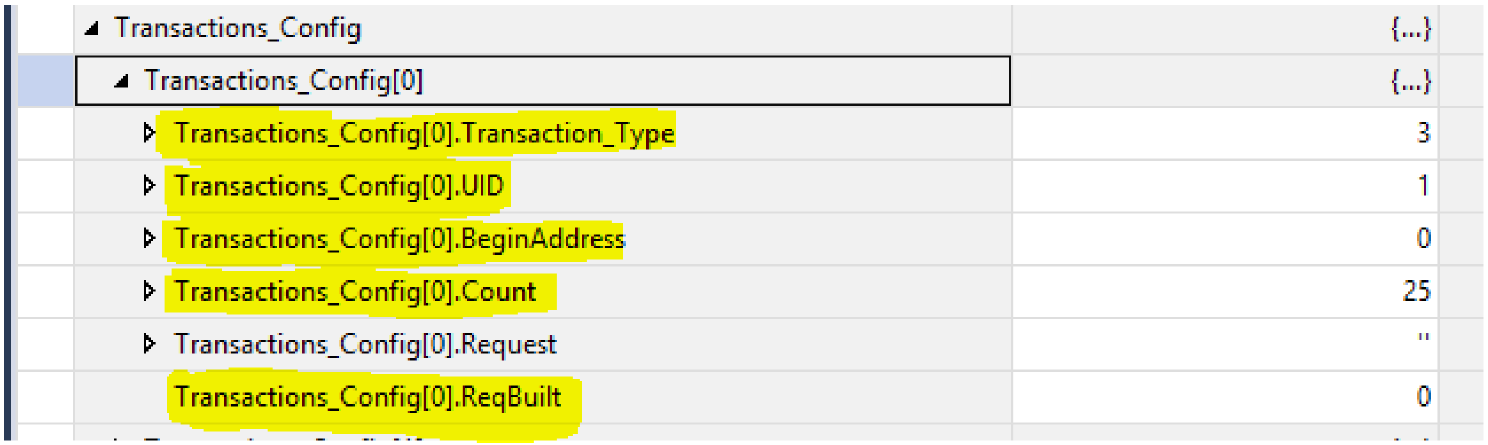
Transaction Block Configuration

1. Navigate to **Controller tags – Transactions_Config(Variable related to Transactions_Config Parameter)**



Name	Value
Read_Socket	
Write_Socket	
MODBUS_RTU_OVER_TCP	
Destination_Address	
Read_Data_Request	
Read_Data_Response	
Socket_Parameters	
Transactions_Config	
Transactions_Config[0]	
Transactions_Config[1]	
Transactions_Config[2]	
Transactions_Config[3]	
Transactions_Config[4]	
Transactions_Config[5]	
Transactions_Config[6]	
Transactions_Config[7]	
Transactions_Config[8]	
Transactions_Config[9]	

2. Configure a transaction for each Modbus device connected on other side as below. (up to 10 devices can be configured)



Transactions_Config	{...}
Transactions_Config[0]	{...}
Transactions_Config[0].Transaction_Type	3
Transactions_Config[0].UID	1
Transactions_Config[0].BeginAddress	0
Transactions_Config[0].Count	25
Transactions_Config[0].Request	''
Transactions_Config[0].ReqBuilt	0

- **Transaction_Type =** Set this to 3 – Read holding registers
  6 – Write one Register
  16 – Write multiple registers
- **UID =** Set the Modbus UID of the Modbus Device.
- **BeginAddress =** The register You want to start reading or writing (**E.g. if the Register is 40001 then put 0 to BeginAddress**) 40001 – 0, 40002 – 1, 40003 – 2,40012 – 11
- **Count =** No of Registers you want to read or write
- **ReqBuilt =** this will rebuilt the request String. Initially set this to 0. If you want to change the values while in program, **After changing the values set this variable to 0. Then it will automatically regenerate the request string**.

- **After doing above configurations download the program to PLC.**

- **While in program you can**

  - Enable or disable Connection.
  - Edit the Number of Transactions
  - Edit any detail related to transactions(After doing this set reqbuilt to 0)