WEINTEK LABS., INC.

TCP/UDP Socket

TCP/UDP Data Transmission achieved by Network Library

Demo Project

Contents

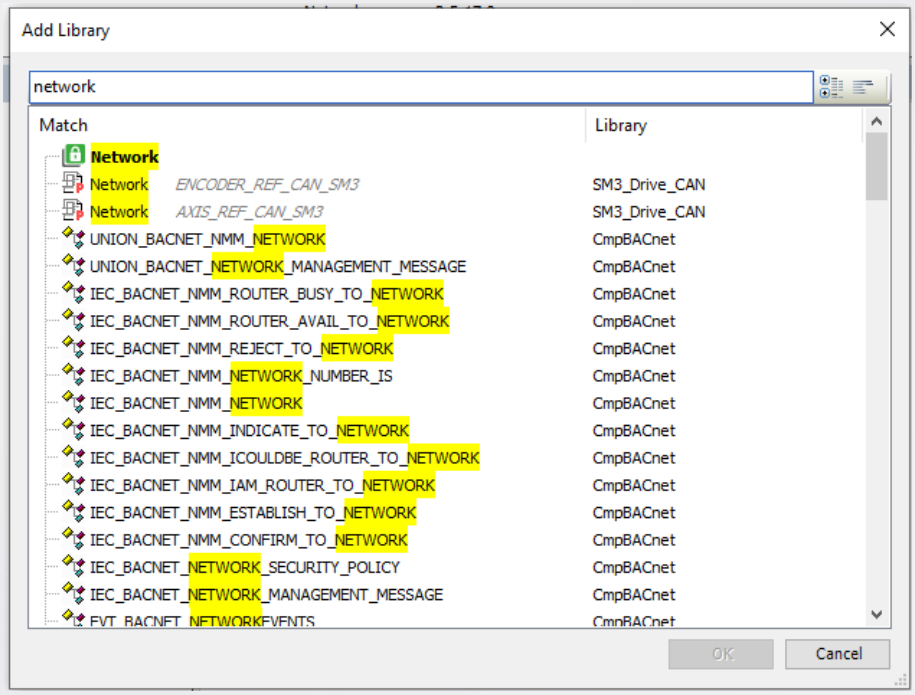
1. Overview

TCP and UDP are communication protocols widely used in the industrial field for communication between production devices or between production devices and internal management system. With Network Library supported in CODESYS, applications using TCP/UDP Socket can be achieved with TCP_Server / TCP_Client / UDP_Peer function blocks.

This demo project explains how to perform TCP & UDP data transmission using Weintek Built-in CODESYS and SocketTest software that runs on PC. Other TCP & UDP socket test software can be used to replace SocketTest on PC.

2. Network Library

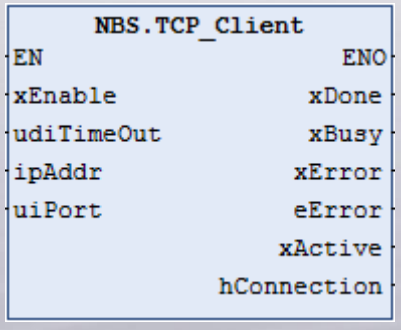
Step 1. In CODESYS software add Network Library.



※ In old versions of CODESYS, adding CAA Net Base Service & CAA Type External Library is necessary because Network Library is not supported.

Function Block

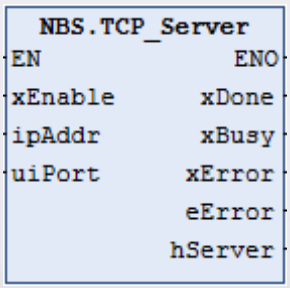
TCP_Client:



Function: Establishes connection with TCP Server.

Input/Output	Type	Description
xEnable	BOOL	See: Library Guidelines
udiTimeOut	UDINT	Defines the time (µs) after which the connection setup aborts with an error message. After the timer has expired, xError is set. With udiTimeOut=0 the timeout is deactivated.
ipAddr	NBS.IP_ADDR	The IP address to establish connection.
uiPort	UINT	The port number to establish connection.
xDone	BOOL	See: Library Guidelines
xBusy	BOOL	See: Library Guidelines
xError	BOOL	See: Library Guidelines
eError	NBS.ERROR	Error ID
xActive	BOOL	TRUE：The handle of the connection is valid. FALSE：The handle of the connection is invalid.
hConnection	CAA.HANDLE	Handle of Connection

TCP_Server:

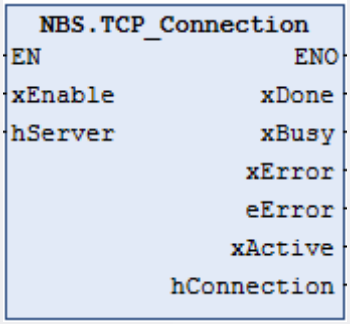


Function: Sets up a TCP Server.

Input/Output	Type	Description
xEnable	BOOL	See: Library Guidelines
udiTimeOut	UDINT	Defines the time (µs) after which the connection setup aborts with an error message. After the timer has expired, xError is set. With udiTimeOut=0 the timeout is deactivated.
ipAddr	NBS.IP_ADDR	The IP address for sending and receiving data.
uiPort	UINT	The port number for sending and receiving

		data.
xDone	BOOL	See: Library Guidelines
xBusy	BOOL	See: Library Guidelines
xError	BOOL	See: Library Guidelines
eError	NBS.ERROR	Error ID
hServer	CAA.HANDLE	Handle of Server

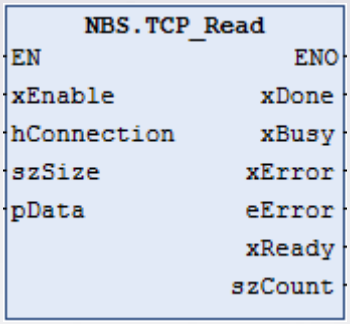
TCP_Connection



Function: Allows connection when TCP Server is active.

Input/Output	Type	Description
xEnable	BOOL	See: Library Guidelines
hServer	CAA.HANDLE	Handle of Server
xDone	BOOL	See: Library Guidelines
xBusy	BOOL	See: Library Guidelines
xError	BOOL	See: Library Guidelines
eError	NBS.ERROR	Error ID
xActive	BOOL	TRUE：The handle of the connection is valid. FALSE：The handle of the connection is invalid.
hConnection	CAA.HANDLE	Handle of Connection

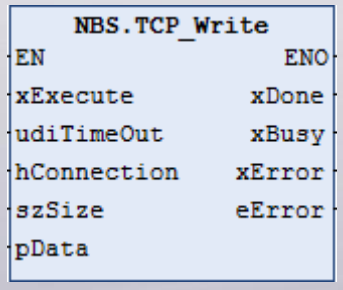
TCP_Read



Function: Reads data from the previously established connection.

Input/Output	Type	Description
xEnable	BOOL	See: Library Guidelines
hConnection	CAA.HANDLE	Handle of Connection
szSize	CAA.SIZE	Maximum number of bytes to be read; can be retrieved via operator SIZEOF.
pData	CAA.PVOID	Target address for the first byte to be read; can be retrieved via operator ADR.
xDone	BOOL	See: Library Guidelines
xBusy	BOOL	See: Library Guidelines
xError	BOOL	See: Library Guidelines
eError	NBS.ERROR	Error ID
xReady	BOOL	TRUE：Data received FALSE：No data received
szCount	CAA.SIZE	Size of the received data.

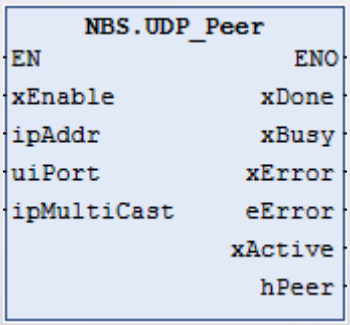
TCP_Write



Function: Writes data to the previously established connection.

Input/Output	Type	Description
xEnable	BOOL	See: Library Guidelines
udiTimeOut	UDINT	Defines the time (µs) after which the connection setup aborts with an error message. After the timer has expired, xError is set. With udiTimeOut=0 the timeout is deactivated.
hConnection	CAA.HANDLE	Handle of Connection
szSize	CAA.SIZE	Maximum number of bytes to be written; can be retrieved via operator SIZEOF.
pData	CAA.PVOID	Target address for the first byte to be written; can be retrieved via operator ADR.
xDone	BOOL	See: Library Guidelines
xBusy	BOOL	See: Library Guidelines
xError	BOOL	See: Library Guidelines
eError	NBS.ERROR	Error ID

UDP_Peer

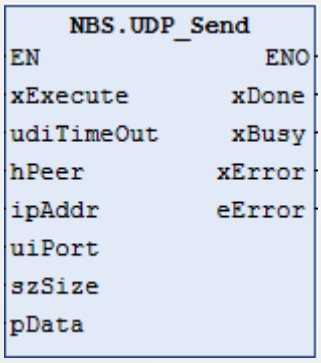


Function: Activates UDP.

Input/Output	Type	Description
xEnable	BOOL	See: Library Guidelines
ipAddr	NBS.IP_ADDR	The IP address to establish connection.
uiPort	UINT	The port number to establish connection.
ipMultiCast	IP_ADDR	Multicast address. '255.255.255.255' => INADDR_NONE
xDone	BOOL	See: Library Guidelines
xBusy	BOOL	See: Library Guidelines

xError	BOOL	See: Library Guidelines
eError	NBS.ERROR	Error ID
xActive	BOOL	TRUE：Handle of peer is valid. FALSE：Handle of peer is invalid.
hPeer	CAA.HANDLE	Handle of Peer

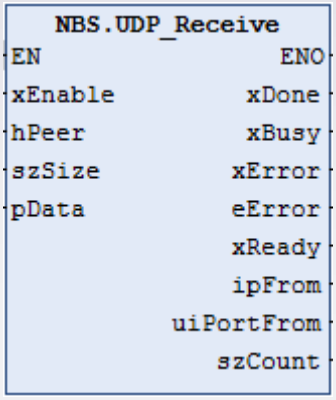
UDP_Send



Function: Sends data when UDP is activated.

Input/Output	Type	Description
xEnable	BOOL	See: Library Guidelines
hPeer	CAA.HANDLE	Handle of Peer
ipAddr	NBS.IP_ADDR	The IP address for sending and receiving data.
uiPort	UINT	The port number for sending and receiving data.
szSize	CAA.SIZE	Maximum number of bytes to be written; can be retrieved via operator SIZEOF.
pData	CAA.PVOID	Target address for the first byte to be written; can be retrieved via operator ADR.
xDone	BOOL	See: Library Guidelines
xBusy	BOOL	See: Library Guidelines
xError	BOOL	See: Library Guidelines
eError	NBS.ERROR	Error ID

UDP_Receive



Function: Receives data when UDP is activated.

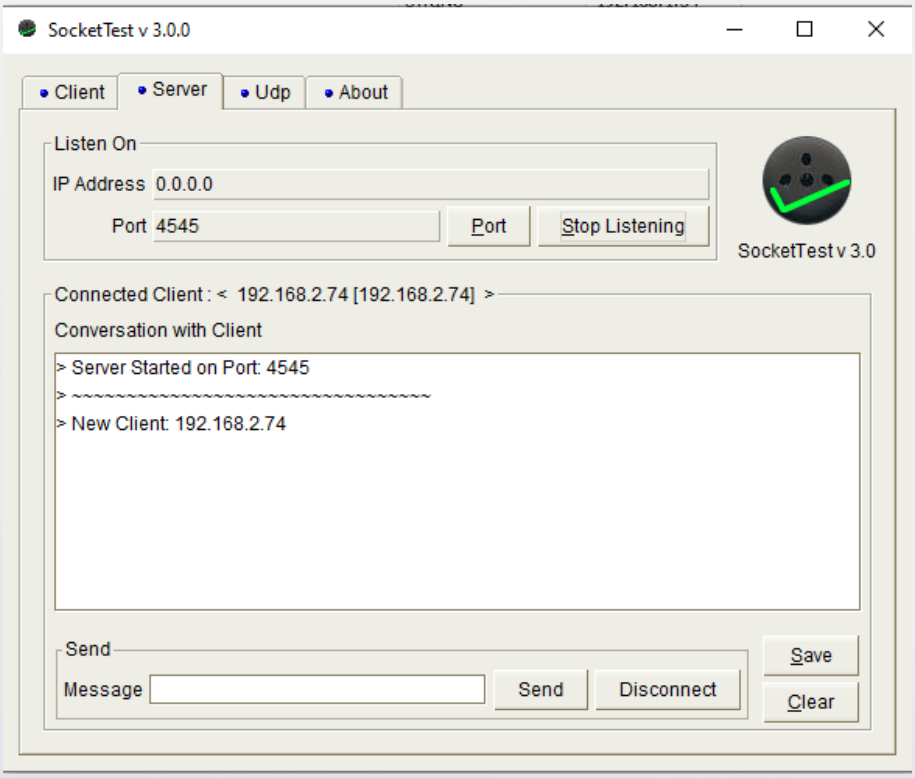
Input/Output	Type	Description
xEnable	BOOL	See: Library Guidelines
hPeer	CAA.HANDLE	Handle of Peer
szSize	CAA.SIZE	Maximum number of bytes to be read; can be retrieved via operator SIZEOF.
pData	CAA.PVOID	Target address for the first byte to be read; can be retrieved via operator ADR.
xDone	BOOL	See: Library Guidelines
xBusy	BOOL	See: Library Guidelines
xError	BOOL	See: Library Guidelines
eError	NBS.ERROR	Error ID
ipFrom	NBS.IP_ADDR	The IP address for receiving data.
uiPortFrom	UINT	The port number for receiving data.
xReady	BOOL	TRUE：Data received FALSE：No data received
szCount	CAA.SIZE	Size of the received data.

3. Operation

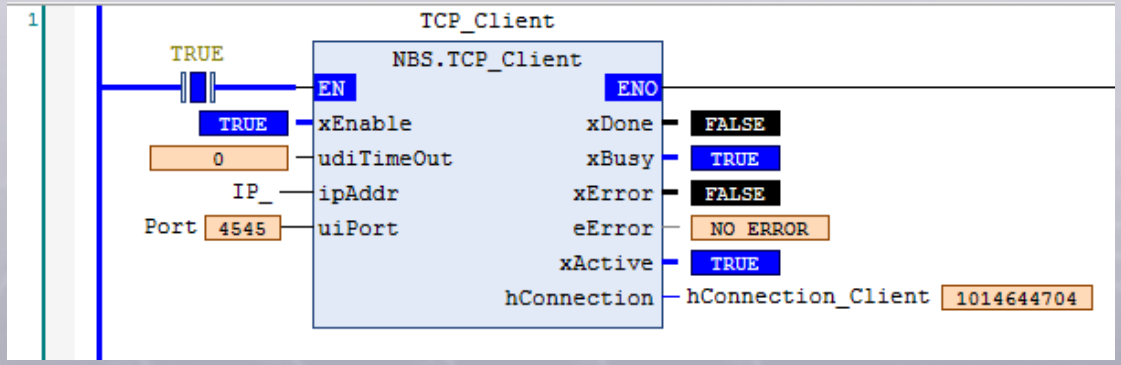
Run the demo project: CODESYS_TCP_Socket.

TCP_Client:

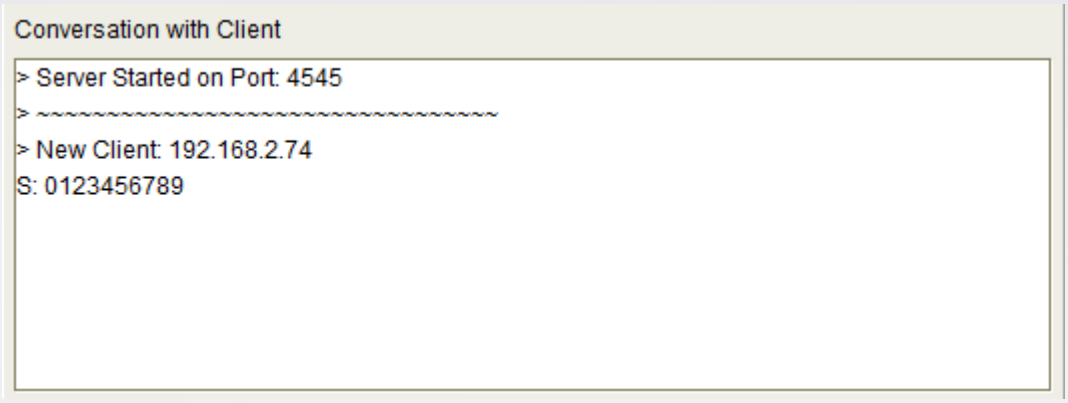
Step 1. In SocketTest software open the Server tab and then click [Start Listening].



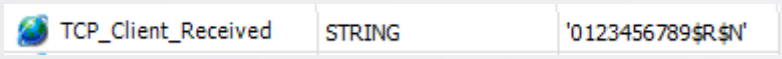
Step 2. Execute Program TCP_Client. IP address = PC's address, Port = the port number found in SocketTest's Server tab, TCP_Client.xEnable=TRUE.



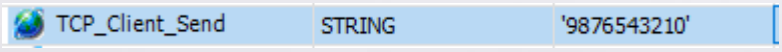
Step 3. Enter a string in the Message field in SocketTest and then click [Send].

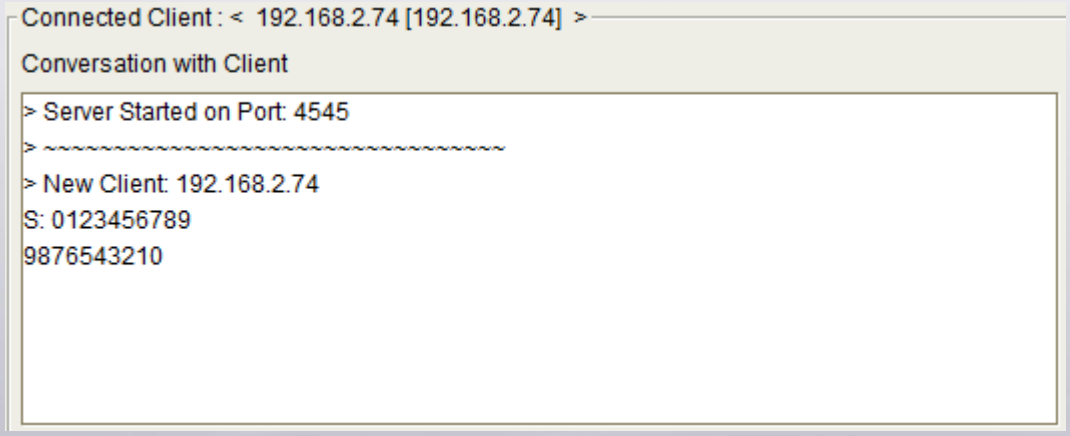


Step 4. When TCP_Read.xEnable = “True”, the string sent from SocketTest is received in GVL.TCP_Client_Received.



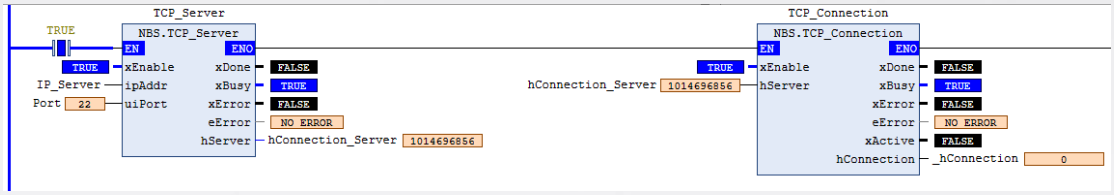
Step 5. In GVL.TCP_Client_Send enter a string, and then the string is sent to SocketTest when TCP_Write.xEnable = “True”. The data in GVL.TCP_Client_Send can then be viewed in the Message filed of SocketTest.



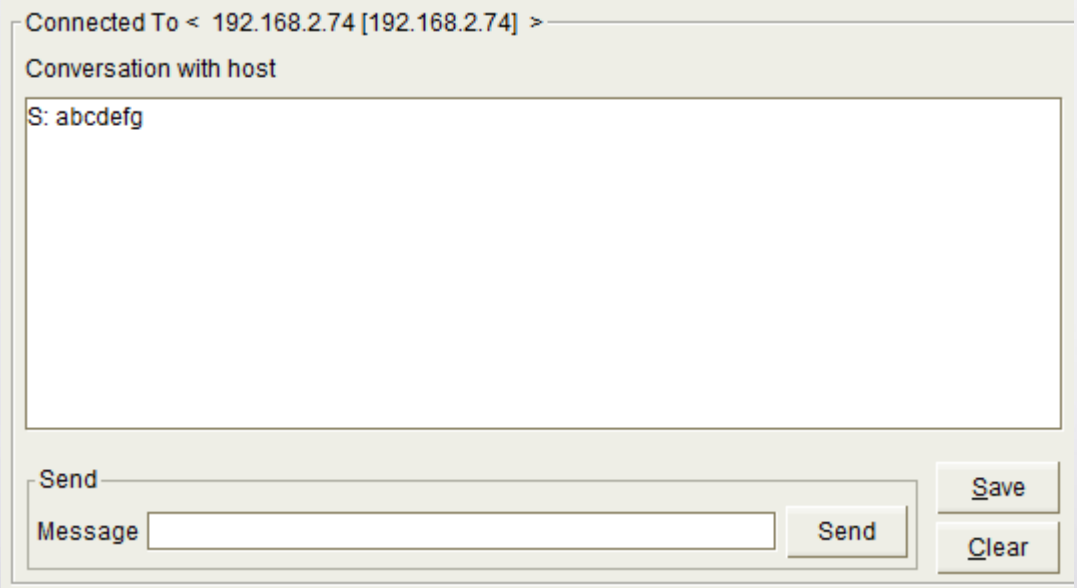


TCP Server:

Step 1. Execute Program TCP_Client. When TCP_Server.xEnable & TCP_Connection.xEnable = “TRUE”, the TCP server is activated and ready for connection.

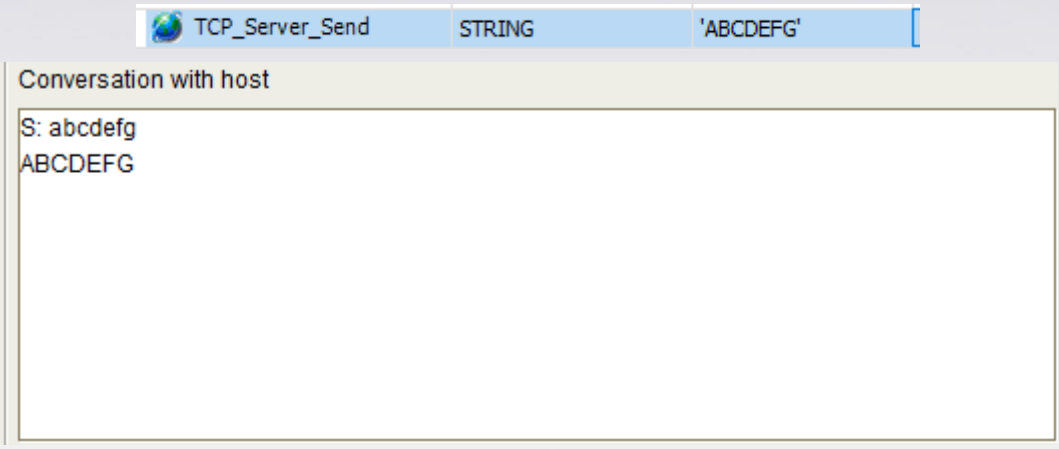


Step 2. In the Client tab in SocketTest, IP address = CODESYS address, Port = TCP_Server’s uiPort. Click [Connect], enter a string in Message filed and then click [Send].



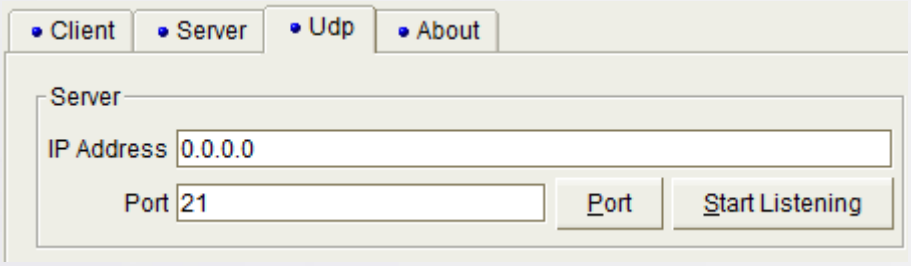
Step 3. When TCP_Read.xEnable = “TRUE”, the string sent from SocketTest is received in GVL.TCP_Server_Received.

Step 4. In GVL.TCP_Client_Send enter a string, and then the string is sent to SocketTest when TCP_Write.xEnable = “True”. The data in GVL.TCP_Client_Send can then be viewed in the Message filed of SocektTest.

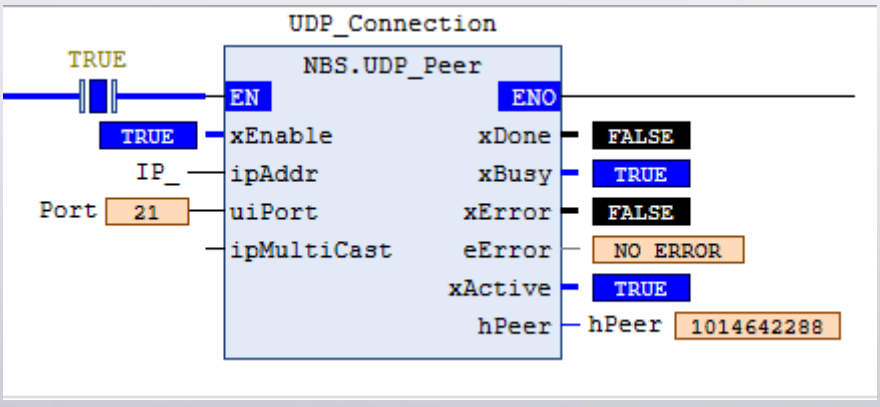


UDP

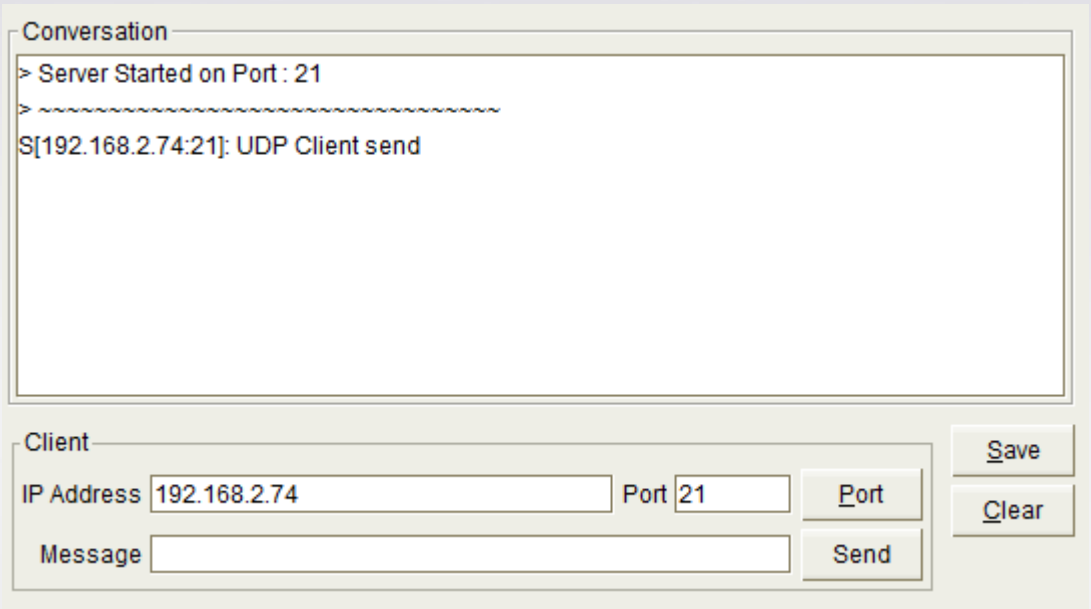
Step 1. In the UDP tab in SocketTest click [Start Listening].

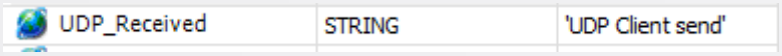


Step 2. Execute Program UDP. IP address = PC's address, Port = the port number found in SocketTest's UDP tab, UDP_Peer.xEnable = "TRUE".



Step 3. Enter a string in the Message field in SocketTest and then click [Send]. When UDP_Read.xEnable = "TRUE", the string sent from SocketTest is received in GVL.TCP_Client_Received.





Step 4. In GVL.UDP_Send enter a string, and then the string is sent to SocketTest when TCP_Write.xEnable = “True”. The data in GVL.UDP_Send can then be viewed in the Message filed of SocektTest.