RTU32 Series DNP3 Peer to Peer

RTU32S

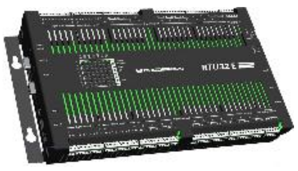
Overview of RTU32 setup for DNP3 peer to peer communications

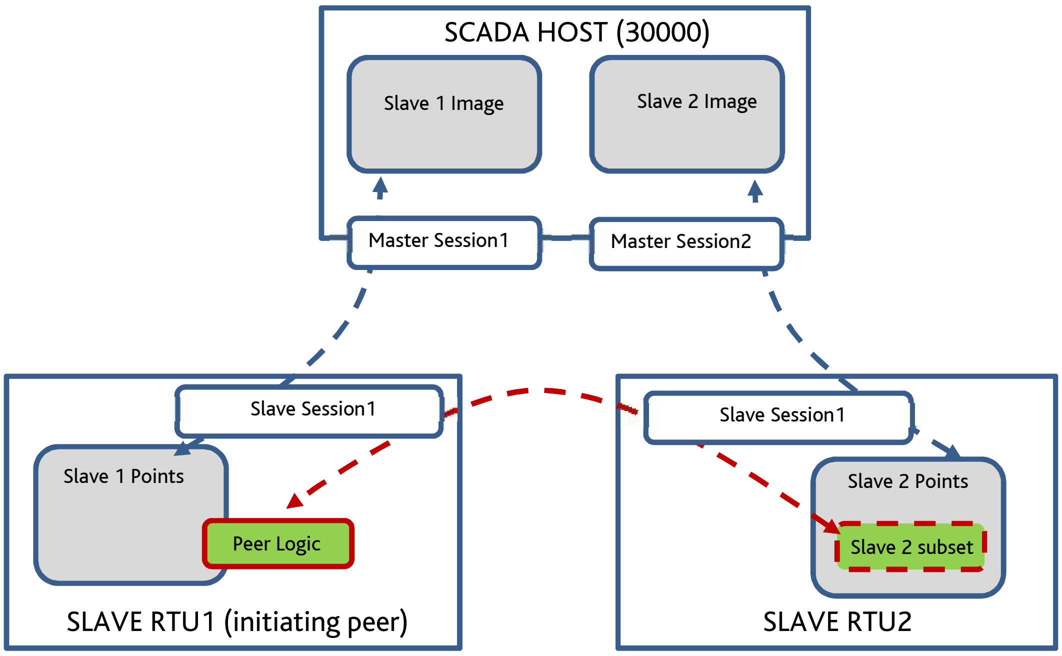
Application Note

January 2015

RTU32

RTU32R


RTU32E



Introduction

Brodersen have been manufacturing products for use in remote monitoring and control solutions for more than 40 years. Our customer base is global and our products are used in a diverse range of applications that include energy management systems, water and waste water SCADA, infrastructure monitoring, building automation and airport management systems.

This application note provides an overview of how DNP3 peer to peer communications are setup. Brodersen RTU products have been used in many applications using DNP3 protocol and are both well proven and easy to configure, use and maintain.

Setup Overview

The RTU32 Series controllers feature a 32bit CPU with on board memory and removable storage cards, serial, USB, 10/100LAN, and various IO capacities eg. RTU32 has 16x DI, 4x DO, 4x AI, 2x AO (with expansion IO capability). The RTUs include an IEC61131-3 logic engine that allows easy implementation of process control logic, with a range of inbuilt functions and tools to create advanced functions. Brodersen are able to assist with provision of example logic code for your applications that includes I/O monitoring, data logging, message handling, alarm escalation etc. The RTU32 also provides local and remote HMI interfaces via its inbuilt web server if future requirements require such functionality. The DNP3 slave protocol implementation also includes 'Peer to Peer' functionality that allows a DNP3 slave RTU32 to read/write a subset of objects from/to a remote DNP3 slave RTU32 via serial or IP interfaces (assuming there is an appropriate communications path available).

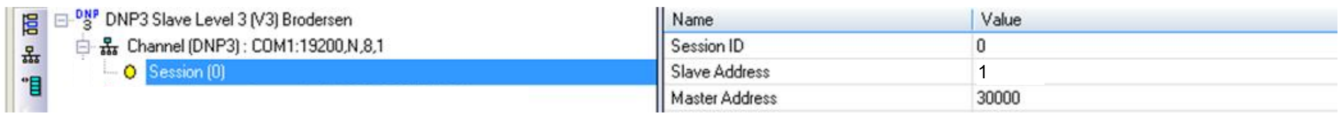
BRODERSEN
simplifying systems

System Architecture

DNP3 Peer to Peer Setup

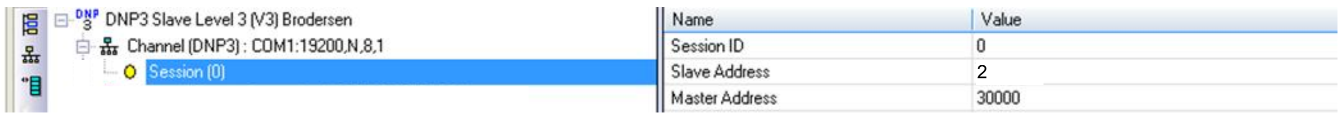
Both Slave RTUs interact with the SCADA master/host. Slave RTU1 is also acting as a peer master (initiating peer) and is able to request data values and write data values to slave RTU2.

The remote slave RTUs communications settings are setup using the Fieldbus Configurator. When communicating via serial ports the peer mode settings are the same as standard slave settings.



DNP3 Slave Level 3 (V3) Brodersen
Channel (DNP3) : COM1:19200,N,8,1
Session (0)

Name	Value
Session ID	0
Slave Address	1
Master Address	30000

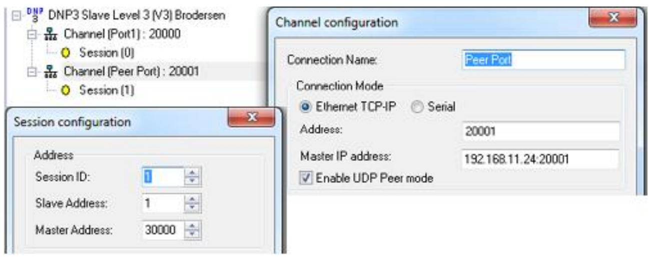


DNP3 Slave Level 3 (V3) Brodersen
Channel (DNP3) : COM1:19200,N,8,1
Session (0)

Name	Value
Session ID	0
Slave Address	2
Master Address	30000

Example Slave RTU1 and RTU2 DNP3 port setup (the same serial interface is used for the SCADA host)

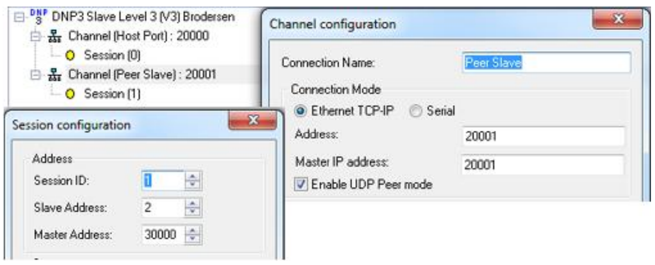
When communicating using IP ports, additional port sessions are required for each slave. Both slaves must have 'UDP Peer Mode' enabled. The initiating RTU uses the Master IP address field to define the slave's IP address.



DNP3 Slave Level 3 (V3) Brodersen
Channel (Port1) : 20000
Session (0)
Channel (Peer Port) : 20001
Session (1)

Channel configuration
Connection Name: Peer Port
Connection Mode
(● Ethernet TCP-IP) ○ Serial
Address: 20001
Master IP address: 192.168.11.24:20001
☑ Enable UDP Peer mode

Session configuration
Address
Session ID: 1
Slave Address: 1
Master Address: 30000



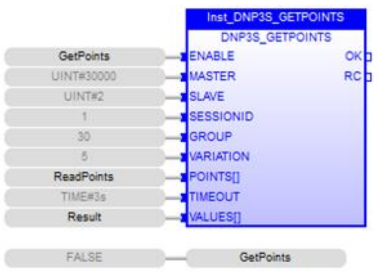
DNP3 Slave Level 3 (V3) Brodersen
Channel (Host Port) : 20000
Session (0)
Channel (Peer Slave) : 20001
Session (1)

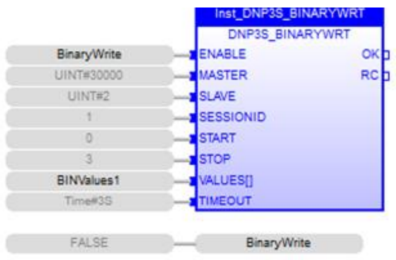
Channel configuration
Connection Name: Peer Slave
Connection Mode
(● Ethernet TCP-IP) ○ Serial
Address: 20001
Master IP address: 20001
☑ Enable UDP Peer mode

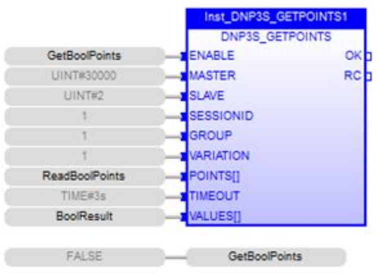
Session configuration
Address
Session ID: 1
Slave Address: 2
Master Address: 30000

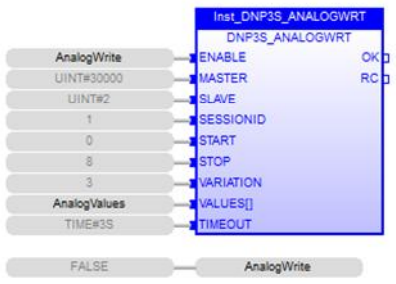
Example Slave RTU1 and RTU2 DNP3 second channel and session setup

Messages to request and send data from/to the peer slave are managed using 'Get Points' and 'Write Points' function blocks as per the example logic below. Each block allows transfer of up to 16 DNP objects/points.









Reading Groups of 1-16 Points (AI and BI shown)   *Writing 1-16 Binary and Analog Outputs*

Function Block Overview

DNP3S_GETPOINTS - Sends a read command for the specified data points

- Inputs:		
ENABLE	BOOL	Activates the function block on rising edge of 'ENABLE'
MASTER	UINT	DNP3 Master Id used in request (needs to be same as SCADA host)
SLAVE	UINT	DNP3 Slave Id used in request (remote slave address)
SESSIONID	DINT	Session Id. Function block uses the channel connection information from the specified session
GROUP	UINT	DNP3 Object group eg. 1 = BI, 30 = AI
VARIATION	UINT	DNP3 variation eg. 1 for BI, for AI 1 = 32bit with flag, 2 = 16bit with flag, 5 = float
POINTS[]	UINT	Array of point numbers to read eg. 0,1,2,3 will read 4 points starting at object 0
TIMEOUT	TIME	Request timeout
VALUES[]	ANY	Array holding the values read from the slave
- Outputs:		
OK	BOOL	Returns TRUE if request succeeded otherwise FALSE is returned.
RC	DINT	Return code. A value of 0 indicates that everything is OK, 1 = Request in progress.

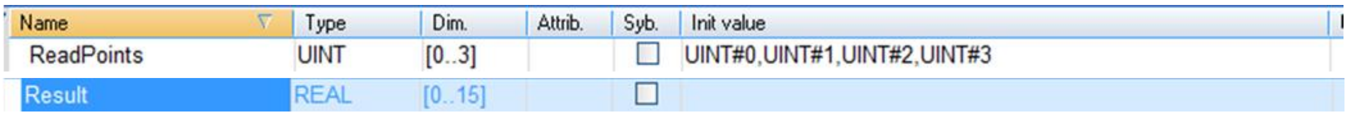
DNP3S_BINARYWRT - Sends a write command for the specified binary output points

- Inputs:		
ENABLE	BOOL	Activates the function block on rising edge of 'ENABLE'
MASTER	UINT	DNP3 Master Id used in write (needs to be same as SCADA host)
SLAVE	UINT	DNP3 Slave Id used in write (remote slave address)
SESSIONID	DINT	Session Id. Function block uses the channel connection information from the specified session
START	UINT	Start point index
STOP	UINT	Stop point index
VALUES[]	BOOL	Array holding the values to write
TIMEOUT	TIME	Write timeout
- Outputs:		
OK	BOOL	Returns TRUE if write succeeded otherwise FALSE is returned.
RC	DINT	Return code. A value of 0 indicates that everything is OK, 1 = Write in progress.

DNP3S_ANALOGWRT - Sends a write command for the specified binary output points

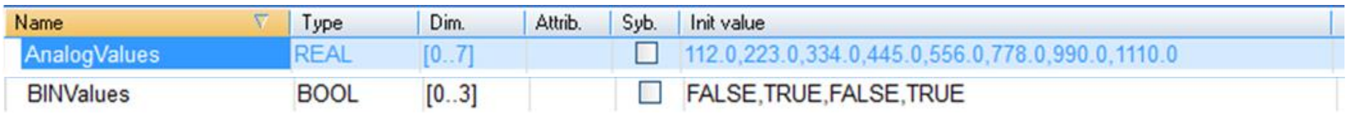
- Inputs:		
ENABLE	BOOL	Activates the function block on rising edge of 'ENABLE'
MASTER	UINT	DNP3 Master Id used in write (needs to be same as SCADA host)
SLAVE	UINT	DNP3 Slave Id used in write (remote slave address)
SESSIONID	DINT	Session Id. Function block uses the channel connection information from the specified session
START	UINT	Start point index
STOP	UINT	Stop point index
VARIATION	UINT	DNP3 AO variation 1 = 32bit, 2 = 16bit, 3 = float
VALUES[]	ANY	Array holding the values to write
TIMEOUT	TIME	Write timeout
- Outputs:		
OK	BOOL	Returns TRUE if write succeeded otherwise FALSE is returned.
RC	DINT	Return code. A value of 0 indicates that everything is OK, 1 = Write in progress.

Example showing use of Array Points for the reading of analog values (RTU requests AI objects 0, 1, 2, 3) – the response from the remote slave will be found in the Result array eg. Result[0], Result[1], Result[2], Result[3].



Name	Type	Dim.	Attrib.	Syb.	Init value
ReadPoints	UINT	[0..3]		☐	UINT#0,UINT#1,UINT#2,UINT#3
Result	REAL	[0..15]		☐	

Example showing use of Array Points for the writing of 8x analog values and 4x binary values.



Name	Type	Dim.	Attrib.	Syb.	Init value
AnalogValues	REAL	[0..7]		☐	112.0,223.0,334.0,445.0,556.0,778.0,990.0,1110.0
BINValues	BOOL	[0..3]		☐	FALSE,TRUE,FALSE,TRUE

Values can also be passed in to arrays for writing from other variables eg. BinValues[0] := Value1;