

pbsSoftLogic DNP3 Slave Driver Configuration

October 2025

Ver. 3.0

Kamjoo Bayat

Kb@pbscontrol.com

1 – Introduction

This document describes how to configure the pbsSoftLogic DNP3 slave driver.

The DNP3 Slave driver supports the IEC 62351 and TLS layers, and this document explains all the configuration details for both layers.

IEC62351 is about authentication and TLS is about encryption of DNP3 frames.

The IEC62351 layer is named SA layer. SA stands for Security and Authentication of frames.

2 – Adding New DNP3 Slave Driver

Create a new project in pbsSoftLogic and name it DNP3Slave. Click on Project setting and select Controller type. Let's say we want to use ECU-150 RTU from Advantech Company. Set RTU IP and save the configuration. (Figure 1)

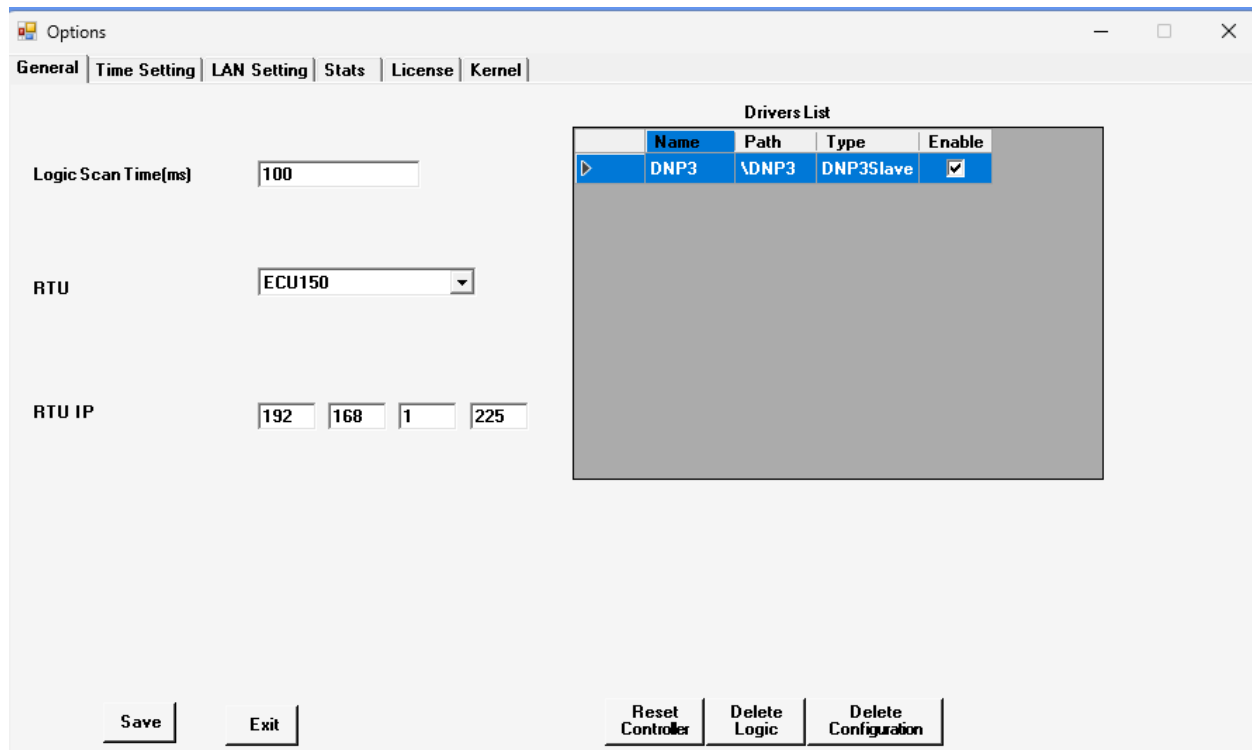


Figure 1

Right click on the list of drivers and add a new driver of type "DNP3Slave" to the project. You can use any name for the driver but it must be unique in the project. Let's say we use "DNP3" as the driver name. Keep the instance number at 1. The concept of Instance has been removed in pbsSoftLogic 2025. (Figure 2)

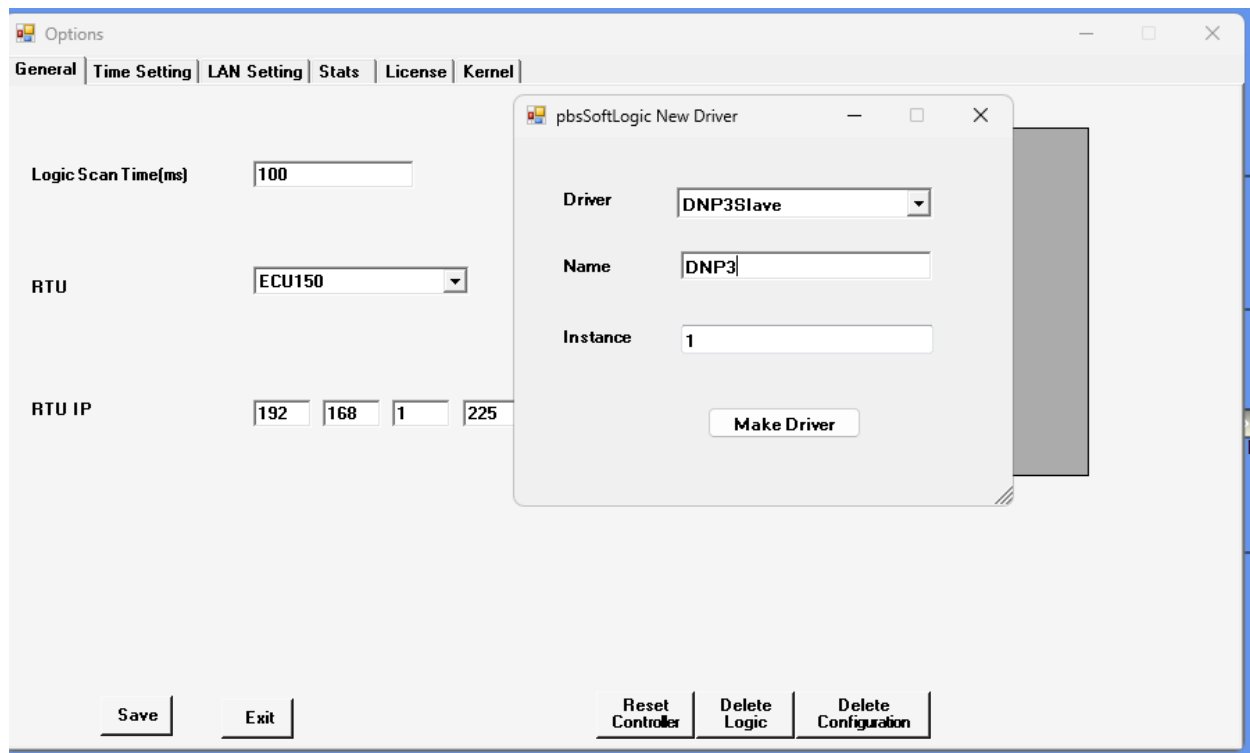


Figure 2

Click the “Make Driver” button, the initial settings and tags for the DNP3 Slave driver will be added to the project. (Figure 3)

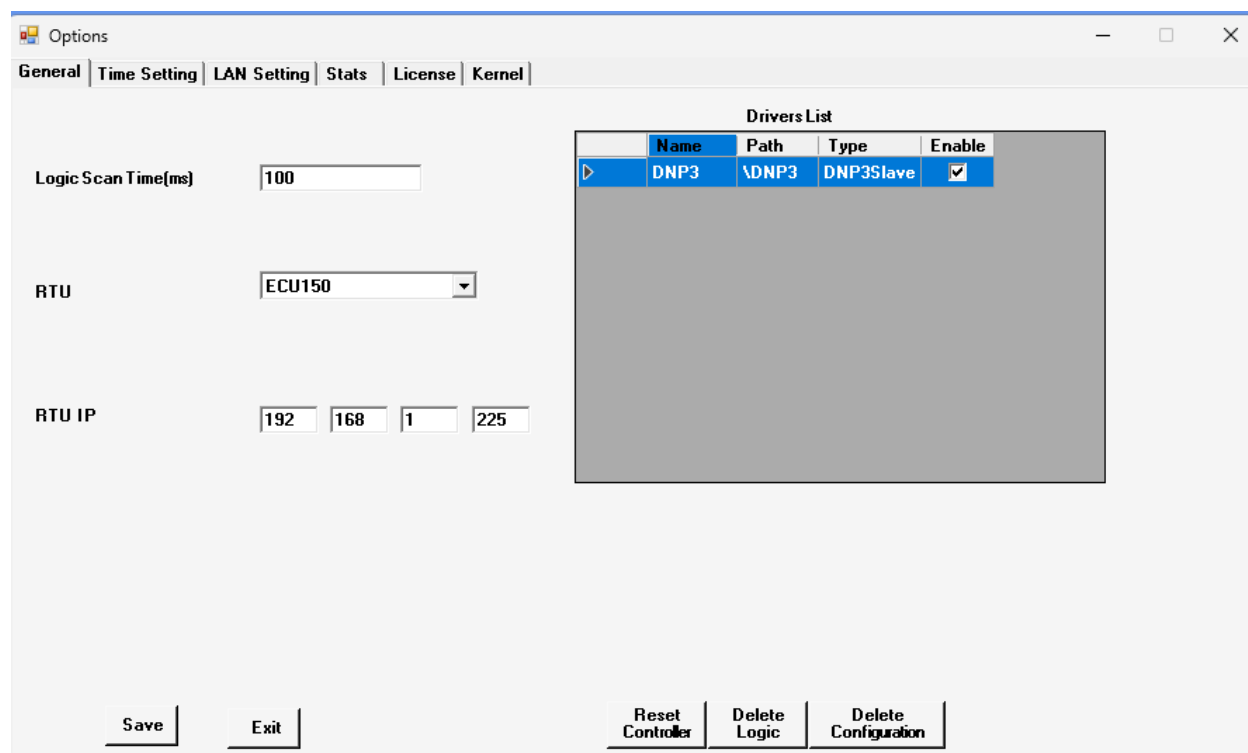


Figure 3

In the project folder you can see a folder for each driver. After adding "DNP3" to the project, the project folder is as shown in Figure 4.

	DNP3	04/08/2025 11:04	File folder	
	LIO	04/08/2025 11:04	File folder	
	DNPSlave2.c11	03/10/2025 10:07	C11 File	3 KB
	DNPSlave2.cfg	03/10/2025 10:26	Configuration Sou...	3 KB
	DNPSlave2.lx	03/10/2025 10:07	LX File	44 KB
	DNPSlave2.xml	03/10/2025 10:27	Microsoft Edge H...	17 KB
	upload.zip	03/10/2025 10:07	Compressed (zipp...	23 KB

Figure 4

Two files have been created in the "DNP3" folder. (Figure 5)



 DNP3Tags.xml	09/09/2025 13:30	Microsoft Edge H...	31 KB
 options.xml	09/09/2025 13:30	Microsoft Edge H...	10 KB

Figure 5

The default DNP3 tags are placed in the DNP3Tags.xml file. You can change the list of tags in this file. All other driver parameters are placed in the options.xml file.

By double-clicking on the "DNP3" driver or by right-clicking on it and using the "Edit" menu, the DNP3 Slave configuration tool will be launched. (Figure 6)

The DNP3 Slave Editor has several tabs:

- Physical Layer: Setup the physical layer (Serial or TCP for DNP3)
- DNP: Set up the DNP3 parameters.
- SA Layer: IEC62351 parameters
- TLS: Configure the TLS layer
- Others: Other options such as debug parameters.
- Default variations: Setup DNP3 default variation for Groups
- Tags: DNP3 tags definition.

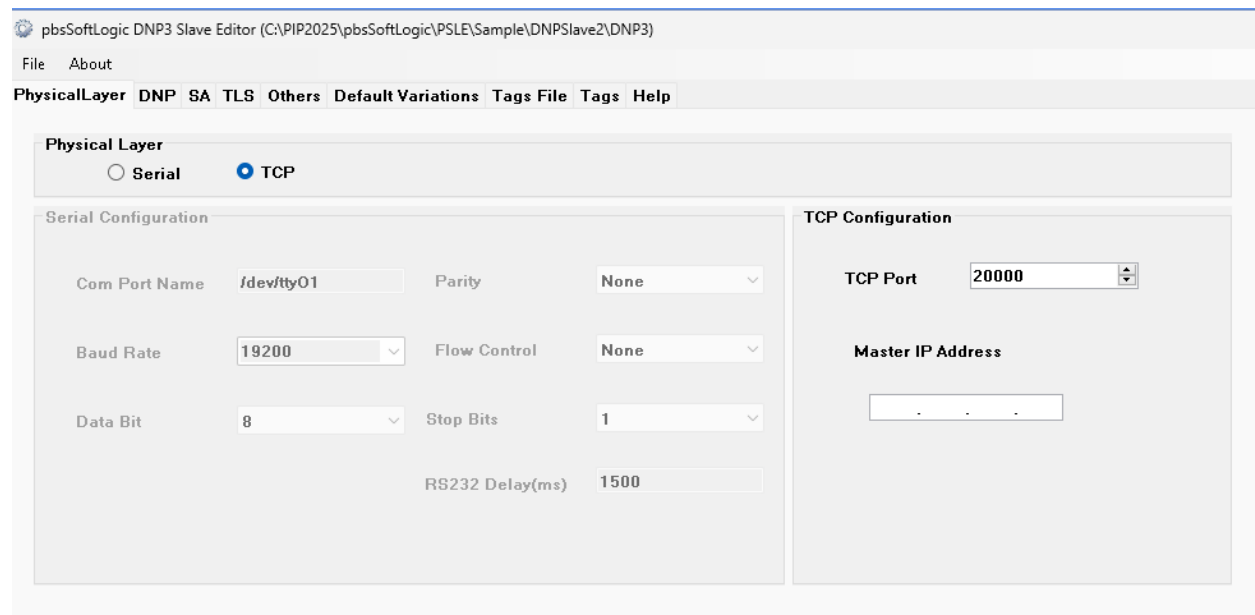


Figure 6

TCP Port: By default, the port for DNP3 is 20000. If you add more DNP3 Slave drivers to the project, you must use a different port.

Master IP Address: If you want the RTU to connect only to a specific Master with a specific IP address, you need to set the Master IP here. If the Master IP address is blank, the RTU will connect to any Master and will not check the Master IP.

To utilize the serial port as a physical layer, the following parameters must be configured:

Com Port Name: Specify the complete name of the COM port, such as /dev/ttyO1. The full COM port name can be found on RTU manual.

ECU1251 COM1 port: /dev/ttyO1

ECU1251 COM2 port: /dev/ttyO1

ECU150 COM1 port: /dev/ttyAP1

ECU150 COM2 port: /dev/ttyAP2

Beagle Bone Black Serial port name for UART1: /dev/ttyO1

Beagle Bone Black Serial port name for UART2: /dev/ttyO2

Raspberry PI Serial Port name for UART1: /dev/ttyAMA0

Raspberry PI Serial Port name for UART2: /dev/ttyS0

Banana PI Serial Port name for UART0: /dev/ttyS0

Banana PI Serial Port name for UART1: /dev/ttyS1

Banana PI Serial Port name for UART2: /dev/ttyS2

According to the DNP3 Standard, the data bits, parity, and stop bits are set to 8-N-1.

RS232 Delay (Mili Second): Some modems have limitations with sending and receiving data, requiring a delay between transmission and reception.

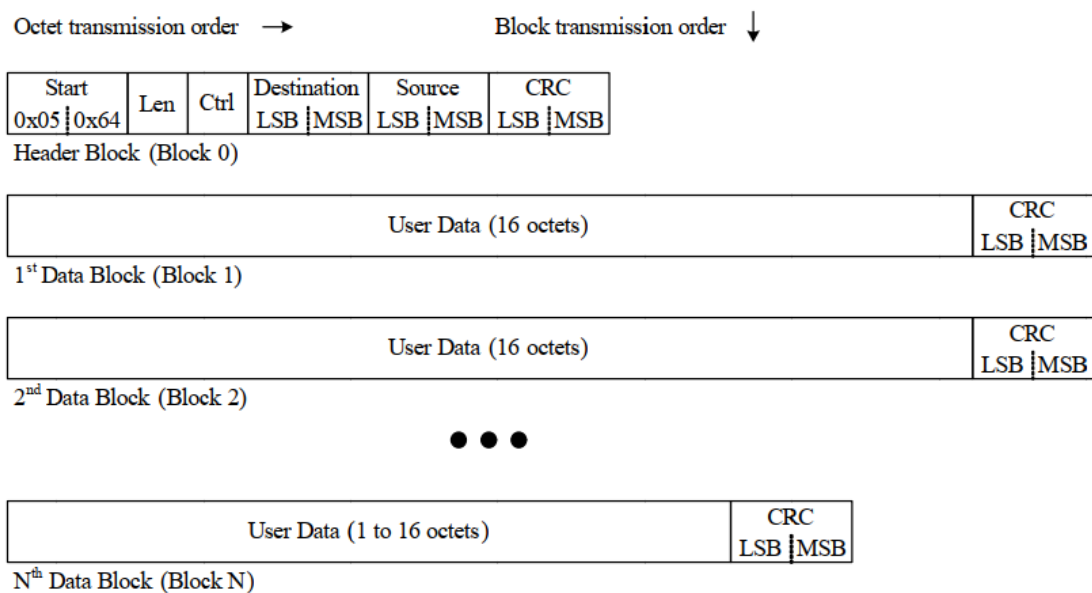
DNP Parameters (Figure 7)

Master and Slave Address: In the DNP3 standard, master and slave must have different addresses at the Data Link Layer.

The Data Link Layer frame format includes specific fields for addressing communication between devices. Each frame consists of two bytes dedicated to the Destination address and two bytes for the Source address.

When the Master device initiates communication by sending a frame, the Destination address is set to the RTU address, while the Source address is assigned as the Master address. This ensures that the RTU recognizes the frame as coming from the Master.

Conversely, when the RTU sends a frame back to the Master, the addressing is reversed: the Destination address becomes the Master address, and the Source address is set to the RTU address. This structure clearly defines the roles of sender and receiver in every frame exchanged at the Data Link Layer.



DNP3 Data Link Frame Format

Select Before Operate Timeout (Sec)

The Select Before Operate (SBO) Timeout specifies the time, in seconds, that the Remote Terminal Unit (RTU) will wait for an Operate command after receiving a Select command from the Master. When the Master device issues a Select command, the RTU enters a waiting state. If the corresponding Operate command is not received before the SBO Timeout period elapses, the RTU will automatically disable the Select state. This mechanism ensures that only timely and coordinated control actions proceed, preventing unintended operations due to communication delays or command loss.

No Communication Timeout (Keep Alive Timeout)

The No Communication Timeout, also referred to as the Keep Alive Timeout, is a crucial parameter in the operation of the Remote Terminal Unit (RTU). This timeout defines the maximum period the RTU will wait at the physical layer without receiving any frames. If the RTU does not receive any communication within the specified No Communication Timeout interval, it will automatically terminate the existing connection. Once the connection is disconnected, the RTU will enter a standby mode, waiting for a new connection request from the Master device. This mechanism ensures that inactive or disrupted sessions are promptly closed, maintaining the reliability and integrity of the communication system.

Parameter	Value	Parameter	Value
Master Address	1	Slave Address	3
Select Before Operate Timeout (s)	10	No Communication Timeout (s)- Keep Alive Timeout	60
Confirmation Timeout (s)	30	Clock Valid Period (s)	3600
Send Unsolicited At Startup	False	Number of Unsolicited Send Retries	3
Unsolicited Send of Analog Values	True	TimeZone	LocalTime
Application Layer Frame Size (Bytes)	1900	Close/Trip Operation	Any/Same
Class0Qualifire	0x28		

Figure 7

Confirmation Timeout

The Confirmation Timeout, measured in seconds, is a parameter at the application layer that determines how long the Remote Terminal Unit (RTU) will wait for a response from the Master after sending a frame. When the RTU transmits a frame to the Master and is expecting confirmation, it enters a wait state. If the RTU does not receive the expected response from the Master within the specified Confirmation Timeout period, it will automatically exit the wait state and revert to the idle state. This mechanism ensures that the RTU does not remain indefinitely in waiting condition, thereby maintaining efficient communication and system responsiveness.

Clock Valid Period

The Clock Valid period (Sec) parameter determines the duration, in seconds, for which the Remote Terminal Unit (RTU) considers its internal clock to be valid. If this parameter is set to a value greater than zero, the RTU will set the IIN1.4 bit to 1 after passing this period. This action serves as an indication that the RTU requires time synchronization from the Master device. In this state, the RTU signals to the Master that its clock may no longer be accurate and should be updated to maintain proper time alignment within the system.

Unsolicited Transmission of Analogue Values

When the option for Unsolicited Send of Analogue Values is enabled (set to True), the Remote Terminal Unit (RTU) will transmit both digital and analogue signals to the Master device whenever unsolicited messaging is active. This configuration ensures that the Master receives updates on all relevant signal changes, including both digital and analogue inputs, without waiting for a specific request.

If this feature is not enabled (set to False), the RTU will only transmit digital signal changes to the Master in unsolicited mode. In this scenario, analogue signal changes will not be communicated unless explicitly requested by the Master, thereby limiting unsolicited updates to digital inputs only.

Close and Trip Operation Settings

The configuration for Close and Trip operations determines how the Master device can issue digital output commands to the Remote Terminal Unit (RTU). There are two selectable modes for this operation: "Any/Same" and "Even/Odd".

"Any/Same" Mode

When the "Any/Same" setting is enabled, the Master is permitted to send digital output commands for both Close and Trip operations to any address within the RTU. This mode provides maximum flexibility, allowing the Master to control Close and Trip functions across all address points without restriction.

"Even/Odd" Mode

If the "Even/Odd" option is selected, the Master's ability to send commands is segmented based on address parity. In this configuration, Close commands can only be issued to even-numbered addresses, while Trip commands are restricted to odd-numbered addresses. This approach enforces a structured method for controlling outputs, ensuring that specific addresses are designated for either Close or Trip operations as defined by the system configuration.

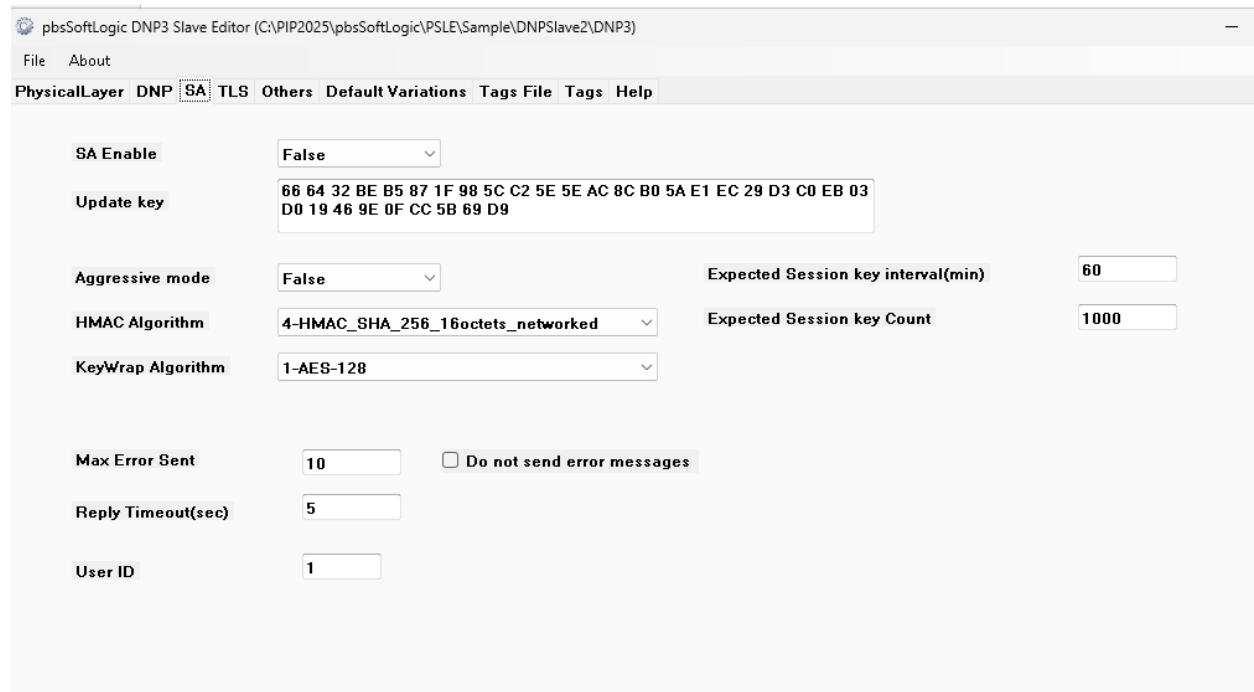
IEC62351 Parameters

With the SA layer tab, Figure 8, you can set authentication parameters for the RTU.

SA Enable : You can enable or disable the IEC62351 layer.

The Update key is the same between master and slave. You can generate a new update key by pbsFDT – DNP3 Tester software. Or generate a new key using other DNP3 testers. The key length should be 16 bytes or 32 bytes. You need to include a space between every two bytes and the bytes are in hex format like in Figure 8.

Aggressive Mode : Aggressive mode: If enabled, the master must send aggressive mode for commands. In this mode, the commands and authentication data are placed in a single frame. If false, the master must use challenge replay mode for authentication.



pbsSoftLogic DNP3 Slave Editor (C:\PIP2025\pbsSoftLogic\PSLE\Sample\DNPSlave2\DNP3)

File About

PhysicalLayer DNP **SA** TLS Others Default Variations Tags File Tags Help

SA Enable False

Update key 66 64 32 BE B5 87 1F 98 5C C2 5E 5E AC 8C B0 5A E1 EC 29 D3 C0 EB 03 D0 19 46 9E 0F CC 5B 69 D9

Aggressive mode False Expected Session key interval(min) 60

HMAC Algorithm 4-HMAC_SHA_256_16octets_networked Expected Session key Count 1000

KeyWrap Algorithm 1-AES-128

Max Error Sent 10 ☐ Do not send error messages

Reply Timeout(sec) 5

User ID 1

Figure 8

HMAC Algorithm: HMAC is used to authenticate frames. The pbsSoftLogic DNP3 slave driver supports SHA-256_8_OCTS and SHA-256_16_OCTS. When the master is initializing SA with the RTU, the RTU informs the master about the HMAC supported by the RTU.

KeyWrap Algorithm : Key wrap is used to send the session key from the master to the RTU. The DNP3 slave driver supports AES-128 format for Key Wrap algorithm. When the Master is initializing the SA with the RTU, the RTU informs the Master about the Key wrap supported by the RTU.

Max Error Sent : The number of error frames sent by the RTU to the master. After this limit, the RTU will not send an error message.

KCL: Key Challenge data Length

The number of random data sent between the Master and the RTU to change the session key between them. This random data is used by the HMAC algorithm to authenticate both parties.

CLN: Authentication Challenge data Length

When the master sends a critical command to the RTU, the RTU sends an authentication challenge to the master. In the authentication challenge, the RTU sends random data to the master and the CLN is the length of this random data. A maximum of 64 random bytes is allowed.

User ID : In the DNP3 Slave driver, one user is supported. This parameter shows the active user ID.

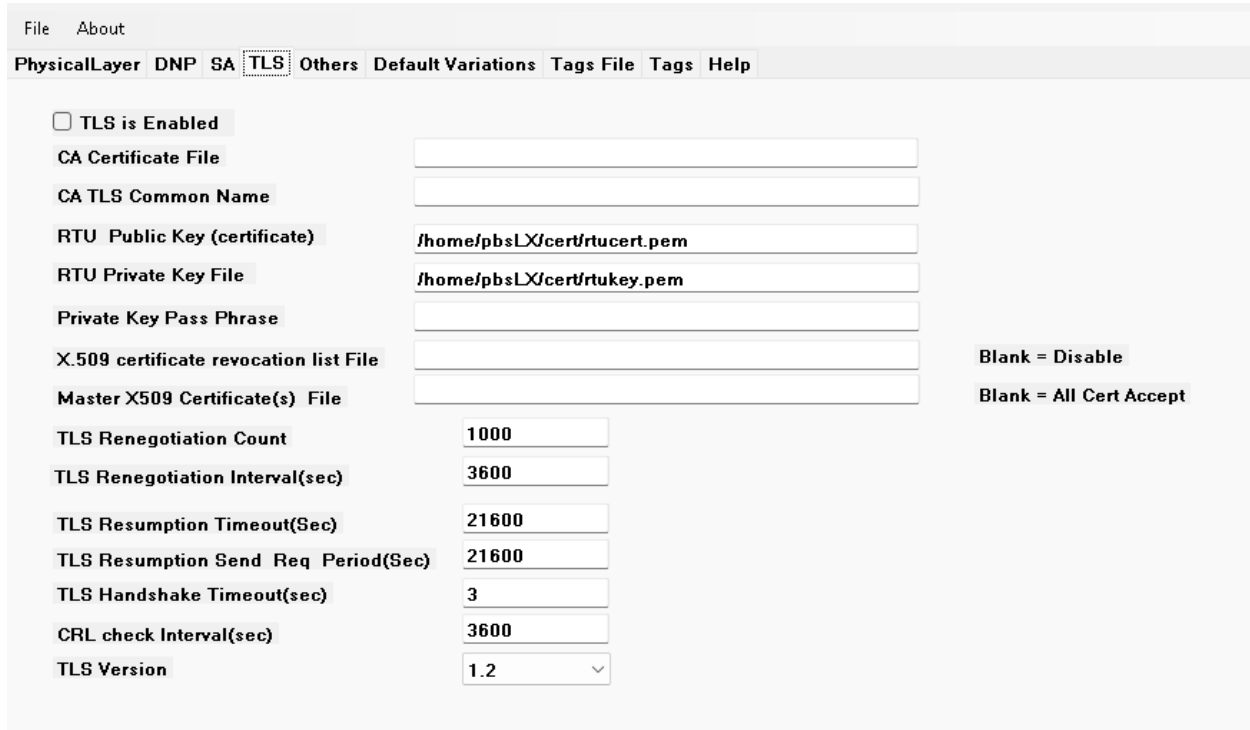
Replay Timeout (Sec): When RTU sends SA frames like challenge authentication, and then RTU waits for response from master side. This parameter shows the timeout value in seconds.

Expected Session Key Interval (Sec): When the master is initializing SA with the RTU, it sends a session key to the RTU. After this time elapses, the RTU waits for the new session key and changes the key state to Not Initialized.

Expected session Key count : Like the expected session key interval, but if the number of critical messages exceeds this parameter, the RTU waits for a new session key.

TLS Parameters

The TLS layer is used to encrypt DNP3 frames. The TLS layer runs over the TCP connection. In Figure 9 you can see the TLS parameters for the DNP3 slave driver.



File About

PhysicalLayer DNP SA **TLS** Others Default Variations Tags File Tags Help

☐ TLS is Enabled

CA Certificate File

CA TLS Common Name

RTU Public Key (certificate) /home/pbsLX/cert/rtucert.pem

RTU Private Key File /home/pbsLX/cert/rtukey.pem

Private Key Pass Phrase

X.509 certificate revocation list File

Master X509 Certificate(s) File

Blank = Disable

Blank = All Cert Accept

TLS Renegotiation Count 1000

TLS Renegotiation Interval(sec) 3600

TLS Resumption Timeout(Sec) 21600

TLS Resumption Send Req Period(Sec) 21600

TLS Handshake Timeout(sec) 3

CRL check Interval(sec) 3600

TLS Version 1.2

Figure 9

To establish a TLS connection, you must use certificate files for the Master and RTU. The easiest way to create certificate files is to use the free XCA tool. Please refer to <https://www.hohnstaedt.de/xca/index.php/download>

With the XCA tool, you must first create a root CA certificate, and then generate the RTU and tester certificates and primary key files.

CA Certificate file : Set the CA certificate file path on the RTU. Assume you have moved the certificate files to the /home/pbsLX/cert folder.

CA Common Name: When you create the CA certificate file, you can set the common name for the certificate. If you set the common name here, the driver compares the certificate common name with the configured name. If they are not the same, the connection is not allowed. If it is empty, the common name is not checked.

RTU certificate file

RTU private key file

Private key pass phrase

Set the full path of the RTU certificate file and private key file here, and if you have encrypted the private key file with a passkey, you must set the passkey here.

X509 revocation list file

You can create a revocation list with the XCA tool. The revocation certificate is not allowed to connect to the RTU. If the revocation list is empty, the revocation list is disabled.

Master certificate file: If empty, the RTU does not check the master certificate, otherwise the RTU compares the master certificate received in the handshake process with the configured certificate path.

The definition and function of other parameters are quite clear.

Other parameters

In Figure 10 you can see other parameters of the IEC104 slave driver.

Diagnostic mode

If you enable diagnostic mode, you can view frames in the RTU console if you manually run the pbsSoftLogic execution core.

To run the pbsSoftLogic runtime on the RTU, connect to the RTU using an SSH client tool and run the following commands:

```
cd /home/pbsLX
```

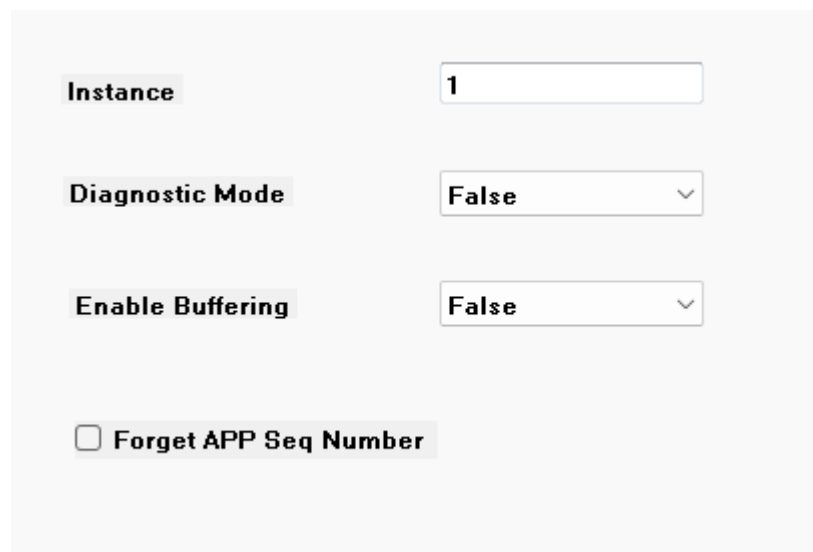
```
pskill pbsSLKLX
```

```
./pbsSLKLX
```

Enable Buffering

Buffering is a very important and practical concept in pbsSoftLogic. Please refer to the specific manual dedicated to buffering concepts in pbsSoftLogic.

Forget APP Seq Number: Some masters not managing Application layer Seq number properly. by this option Seq Number is not checked in the RTU.



Instance	1
Diagnostic Mode	False
Enable Buffering	False
<input type="checkbox"/> Forget APP Seq Number	

Figure 10

Default Variations

DNP3 tags in the pbsSoftLogic Slave Driver are organized according to the protocol's Group and Variation structure. Each tag is categorized based on its assigned group and relevant variation, which determines the type of data and behavior associated with it. The slave driver in pbsSoftLogic supports a specific set of DNP3 groups, ensuring compatibility and effective communication with master devices.

The following groups are supported in the pbsSoftLogic Slave Driver:

Group 1: Binary input

Group 2: Binary input Event

Group 3: Double Bit Binary input

Group 4: Double Bit Binary Input Event

Group 10: Binary Output (Status)

Group 12: Binary Output Command

Group 20: Counters

Group 21: Frozen Counters

Group 22: Counter Events

Group 23: Frozen Counter Events

Group 30: Analog input

Group 31: Analog input Events

Group 40: Analog Output Status

Group 41: Analog Output command

Group 50: Date Time

Group 60: class Object

Group 80: Internal indication

Group 120: authentication

Group 121: security Statistics

Group 122: security Statistics Events

DNP3 Tags

When you add a new DNP3 Slave driver to the project, default parameters and tags are added to the project. The following DNP3 data types are supported in the slave driver.

Digital Input (DI): Group 1: Binary input, Group 2: Binary input Event

Double Bit input (DPI): Group 3: Double Bit Binary input, Group 4: Double Bit Binary Input Event

Digital Output (DO): Group 10: Binary Output (Status)

Digital Output Command (DOB): Group 12: Binary Output Command

Counter (CNT):

- Group 20 Counters,
- Group 21 Frozen Counters
- Group 22 Counter Events
- Group 23 Frozen Counter Events

Analog Input (AI): Group 30: Analog input, Group 31: Analog input Events

Analog Output (AO): Group 40: Analog Output Status

Analog Output Command (AOB): Group 41: Analog Output command

Following properties can be defined for each DNP3 tag:

- Tag Name
- Tag Type
- Init Value
- Address
- Class
- Period
- Log
- Deadband
- DOBMask

pbsSoftLogic Tag Types are as following:

DI : 1–2

DO: 10

AI : 30-31

AO :40

CNT : 20-21-22-23

FI : 30-31 Var 5,6,7,8

FO : 40 Var 3,4

DPI : 3 -4

DOB : 12

AOB : 41 Var 1,2, 3,4

Address : DNP3 Tags Address , for AI , FI using same addressing space .

Class: DNP3 Tag Class = 1,2,3

Period: If set to a non-zero value, the RTU will send the tag with a cyclic transmission .

The value is in seconds. If you set Period to 5 seconds, the RTU will send the signal to the master every 5 sec .

Log: If set for a signal, especially set point signals sent by the master to the RTU, the last value of the signal is stored in the RTU flash and when the RTU is restarted, the last value for the set point is loaded.

Deadband : if set for AI and FI , RTU will send data when signal change is bigger than deadband .
Suppose deadband is 0.2 , previous signal value is 12.23 and current value is 12.24 , so RTU will not send data to the master . RTU will send data if $\text{abs}(\text{LastValue} - \text{CurrentValue}) \geq \text{deadband}$

DOBMask : It is enabled only for DOB tags . With help of DOBMask you can disable some functions for DOB Tags by setting DOBMask Bits as following :

DOBMask Bit 0 : Pulse Disable

DOBMask Bit 1 : Latch Disable

DOBMask Bit 2 : Close Disable

DOBMask Bit 3 : Trip Disable

DOBMask Bit 4 : Direct operate Disable

DOBMask Bit 5 : SBO Disable

By Default all functions are enabled.

DOBC and DOBT Type :

For DOB tags you can define DOBC and DOBT tags like following sample :

```
<Tag Name="DOBTag10" Type="DOB" Class="1" Init="0" Address="10" Log="0" Period="0" />
```

```
<Tag Name="DOBTag10.C" Type="DOBC" Class="1" Init="0" Address="10" Log="0" Period="0" />
```

```
<Tag Name="DOBTag10.T" Type="DOBT" Class="1" Init="0" Address="10" Log="0" Period="0" />
```

If you define DOBC or DOBT tags for a DOB signal , with same DOB Address , then when Master is writing to DOB for Closing , then the signal with DOBC type will change to 1 and DOBT change to 0 .

If master is sending DOB Trip , then Tag with DOBT type will change to 1 and Tag with DOBC change to 0 .

System Tags:

```
<Tag Name="MasterOnline" Type="SYS" Class="0" Init="0" Address="0" Log="0" Period="0" />  
<Tag Name="RemoteLocal" Type="SYS" Class="1" Init="0" Address="1" Log="0" Period="0" />  
<Tag Name="CounterClearCmd" Type="SYS" Class="1" Init="0" Address="2" Log="0" Period="0" />
```

MasterOnline : When DNP3 Master is connected and communicating data with RTU , this signal will set to 1 .

RemoteLocal : you can link this signal to a local key at site , then when Enabled , SCADA Master Can not send command to the site .

CounterClearCommand : When master is sending a Freeze Command with clear to RTU , a five sec pulse will generate for this signal . You can use it for clearing external counters .

Suppose RTU is reading counters from external devices by Modbus protocol . when Master Send Freeze with command , CounterClearCommand will change to 1 for 5 sec and coming back to 0 . you need to use this signal and clear counters at external device by modbus .

Sample Tag definition in DNPTags.xml

```
<Tag Name="DITag1" Type="DI" Class="1" Init="0" Address="100" Log="0" Period="0" />
<Tag Name="DITag2" Type="DI" Class="1" Init="0" Address="101" Log="0" Period="0" />
<Tag Name="DITag3" Type="DI" Class="1" Init="0" Address="3" Log="0" Period="0" />
<Tag Name="DITag4" Type="DI" Class="1" Init="0" Address="4" Log="0" Period="0" />
<Tag Name="DITag5" Type="DI" Class="1" Init="0" Address="5" Log="0" Period="0" />
<Tag Name="DITag6" Type="DI" Class="1" Init="0" Address="6" Log="0" Period="0" />
<Tag Name="DITag7" Type="DI" Class="1" Init="0" Address="7" Log="0" Period="0" />
<Tag Name="DITag8" Type="DI" Class="1" Init="0" Address="8" Log="0" Period="0" />

<Tag Name="AITag1" Type="AI" Class="1" Init="0" Address="1" Log="0" Period="0" Deadband="5.0" DOBMask="0" />
<Tag Name="AITag2" Type="AI" Class="1" Init="0" Address="2" Log="0" Period="0" />
<Tag Name="AITag3" Type="AI" Class="1" Init="0" Address="3" Log="0" Period="0" />
<Tag Name="AITag4" Type="AI" Class="1" Init="0" Address="4" Log="0" Period="0" />
<Tag Name="AITag5" Type="AI" Class="1" Init="0" Address="5" Log="0" Period="0" />
<Tag Name="AITag6" Type="AI" Class="1" Init="0" Address="6" Log="0" Period="0" />
<Tag Name="AITag7" Type="AI" Class="1" Init="0" Address="7" Log="0" Period="0" />
<Tag Name="AITag8" Type="AI" Class="1" Init="0" Address="8" Log="0" Period="0" />
<Tag Name="FITag129" Type="FI" Class="1" Init="0" Address="129" Log="0" Period="0" Deadband="0.5" />
<Tag Name="FITag130" Type="FI" Class="1" Init="0" Address="130" Log="0" Period="0" />
<Tag Name="FITag131" Type="FI" Class="1" Init="0" Address="131" Log="0" Period="0" />
<Tag Name="FITag132" Type="FI" Class="1" Init="0" Address="132" Log="0" Period="0" />
<Tag Name="CNTTag1" Type="CNT" Class="1" Init="0" Address="1" Log="0" Period="0" />
<Tag Name="CNTTag2" Type="CNT" Class="1" Init="0" Address="2" Log="0" Period="0" />
<Tag Name="CNTTag3" Type="CNT" Class="1" Init="0" Address="3" Log="0" Period="0" />
<Tag Name="CNTTag4" Type="CNT" Class="1" Init="0" Address="4" Log="0" Period="0" />
<Tag Name="DOBTag1" Type="DOB" Class="1" Init="0" Address="1" Log="0" Period="0" />
<Tag Name="DOBTag2" Type="DOB" Class="1" Init="0" Address="2" Log="0" Period="0" />
<Tag Name="DOBTag3" Type="DOB" Class="1" Init="0" Address="3" Log="0" Period="0" />
<Tag Name="DOBTag4" Type="DOB" Class="1" Init="0" Address="4" Log="0" Period="0" />

<Tag Name="CNTTag4" Type="CNT" Class="1" Init="0" Address="4" Log="0" Period="0" />
<Tag Name="DOBTag1" Type="DOB" Class="1" Init="0" Address="1" Log="0" Period="0" />
<Tag Name="DOBTag2" Type="DOB" Class="1" Init="0" Address="2" Log="0" Period="0" />
<Tag Name="DOBTag3" Type="DOB" Class="1" Init="0" Address="3" Log="0" Period="0" />
<Tag Name="DOBTag4" Type="DOB" Class="1" Init="0" Address="4" Log="0" Period="0" />

<Tag Name="DOBTag10" Type="DOB" Class="1" Init="0" Address="10" Log="0" Period="0" />
<Tag Name="DOBTag10.C" Type="DOBC" Class="1" Init="0" Address="10" Log="0" Period="0" />
<Tag Name="DOBTag10.T" Type="DOBT" Class="1" Init="0" Address="10" Log="0" Period="0" />

<Tag Name="DOBTag11" Type="DOB" Class="1" Init="0" Address="11" Log="0" Period="0" />
<Tag Name="DOBTag12" Type="DOB" Class="1" Init="0" Address="12" Log="0" Period="0" />
<Tag Name="DOBTag13" Type="DOB" Class="1" Init="0" Address="13" Log="0" Period="0" />

<Tag Name="DOBTag14" Type="DOB" Class="1" Init="0" Address="14" Log="0" Period="0" />
<Tag Name="DOBTag14_C" Type="DOBC" Class="1" Init="0" Address="14" Log="0" Period="0" />
<Tag Name="DOBTag14_T" Type="DOBT" Class="1" Init="0" Address="14" Log="0" Period="0" />

<Tag Name="DOBTag15" Type="DOB" Class="1" Init="0" Address="15" Log="0" Period="0" DOBMask="4" />
<Tag Name="DOBTag16" Type="DOB" Class="1" Init="0" Address="16" Log="0" Period="0" />
<Tag Name="AOBTag1" Type="AOB" Class="1" Init="0" Address="1" Log="0" Period="0" />
<Tag Name="AOBTag2" Type="AOB" Class="1" Init="0" Address="2" Log="0" Period="0" />
<Tag Name="AOBTag3" Type="AOB" Class="1" Init="0" Address="3" Log="0" Period="0" />
<Tag Name="DOTag1" Type="DO" Class="1" Init="0" Address="1" Log="0" Period="0" />
<Tag Name="DOTag2" Type="DO" Class="1" Init="0" Address="2" Log="0" Period="0" />
<Tag Name="DOTag3" Type="DO" Class="1" Init="0" Address="3" Log="0" Period="0" />
<Tag Name="DOTag4" Type="DO" Class="1" Init="0" Address="4" Log="0" Period="0" />
<Tag Name="AOTag1" Type="AO" Class="1" Init="0" Address="1" Log="0" Period="0" />
<Tag Name="AOTag2" Type="AO" Class="1" Init="0" Address="2" Log="0" Period="0" />
<Tag Name="AOTag3" Type="AO" Class="1" Init="0" Address="3" Log="0" Period="0" />
<Tag Name="AOTag4" Type="AO" Class="1" Init="0" Address="4" Log="0" Period="0" />
<Tag Name="AOTag5" Type="AO" Class="1" Init="0" Address="5" Log="0" Period="0" />
<Tag Name="AOTag6" Type="AO" Class="1" Init="0" Address="6" Log="0" Period="0" />
<Tag Name="AOTag7" Type="AO" Class="1" Init="0" Address="7" Log="0" Period="0" />
```

```
<Tag Name="FOTag33" Type="FO" Class="1" Init="0" Address="33" Log="0" Period="0" />
<Tag Name="FOTag34" Type="FO" Class="1" Init="0" Address="34" Log="0" Period="0" />
<Tag Name="FOTag35" Type="FO" Class="1" Init="0" Address="35" Log="0" Period="0" />

<Tag Name="DBITag1" Type="DPI" Class="1" Init="0" Address="1" Log="0" Period="0" />
<Tag Name="DBITag2" Type="DPI" Class="1" Init="0" Address="2" Log="0" Period="0" />
<Tag Name="DBITag3" Type="DPI" Class="1" Init="0" Address="3" Log="0" Period="0" />
<Tag Name="DBITag4" Type="DPI" Class="1" Init="0" Address="4" Log="0" Period="0" />
<Tag Name="DBITag5" Type="DPI" Class="1" Init="0" Address="5" Log="0" Period="0" />
```

You need to modify the default tag list based on your project needs. You can use Notepad++ editor and create new tag list based on your project requirement.

FB Programming

After defining DNP3 tags and setting parameters, you can use the tags in your Function Block program.

In Figure 11, a simple program that writes the output of a pulse generator to a DI tag is shown.

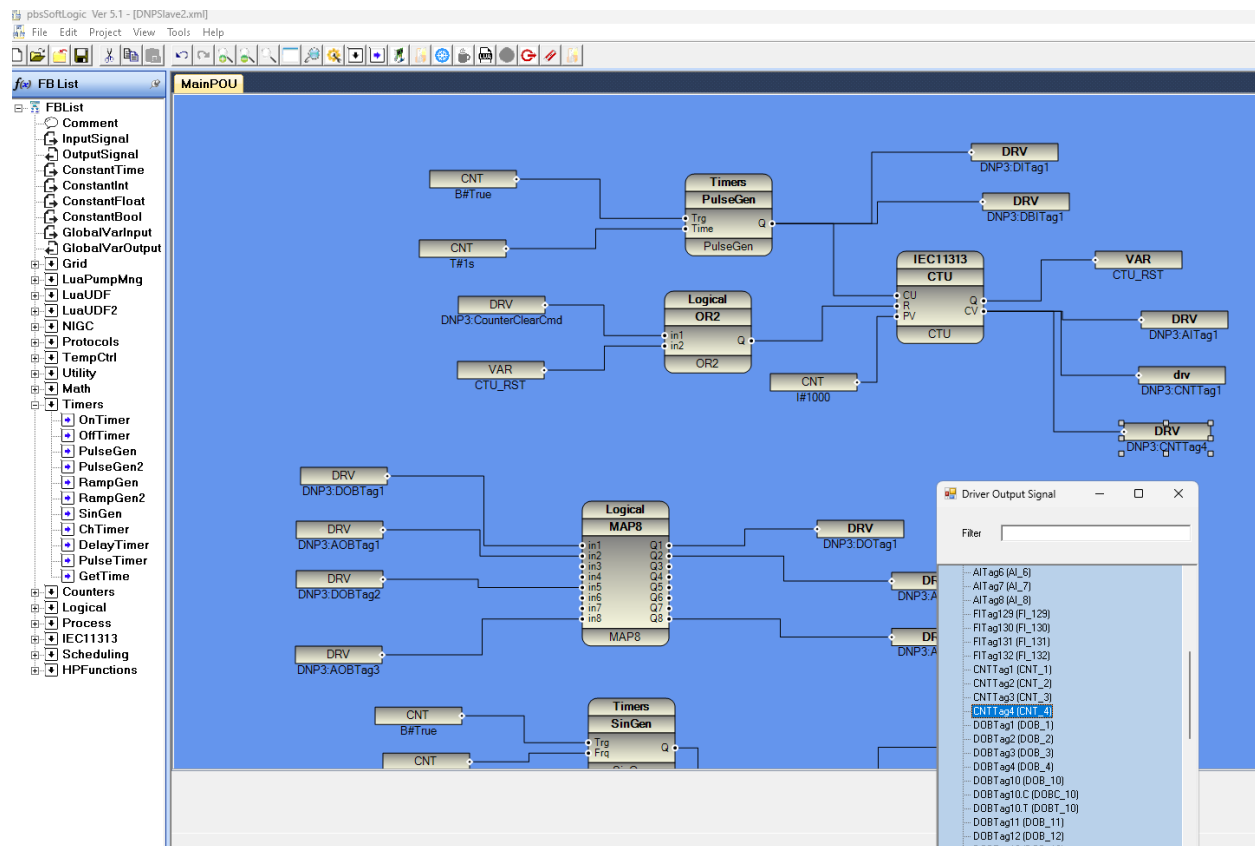


Figure 11

All input signals (DI, AI, DPI, CNT,AO,DO) must be placed on the right side of a Function Block to write a value to the driver.

All Output signals (DOB,AOB) must be placed on the left side of a Function Block to read value from driver.

In Figure 12, the pulse generator output is counted by a CTU function block and written to an DNP3 tag AI with address 1 and on the same time on the counters 1 and 4 . Also PulseGen Output is linked to DI tag with Address 1- and Double-Point input with Address1 .

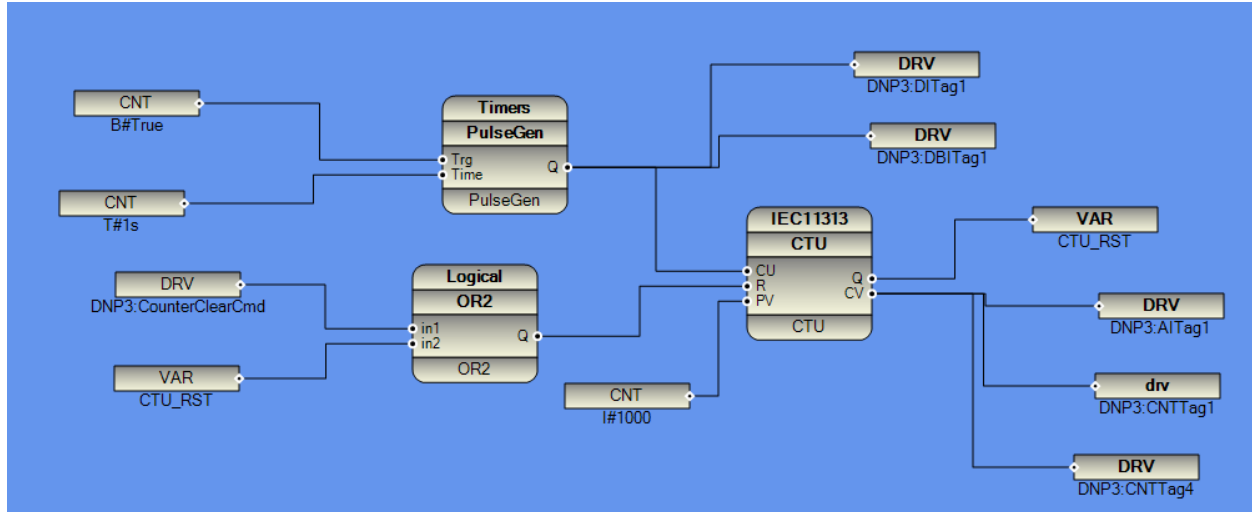


Figure 12

In Figure 12, you can see How we use CounterClearCmd tag . When Master Send Counter Freeze with Clear , CounterClearCMD will clear CTU function output to 0 .

In Following figure you can see how we can read Maser commands DOBTag1, AOBTAG1 and AOBTAG3 and map values to the DOTag1 , AOTag1 and AOTag3. Also you can see how to write a Sin Signal on FITTag129 .

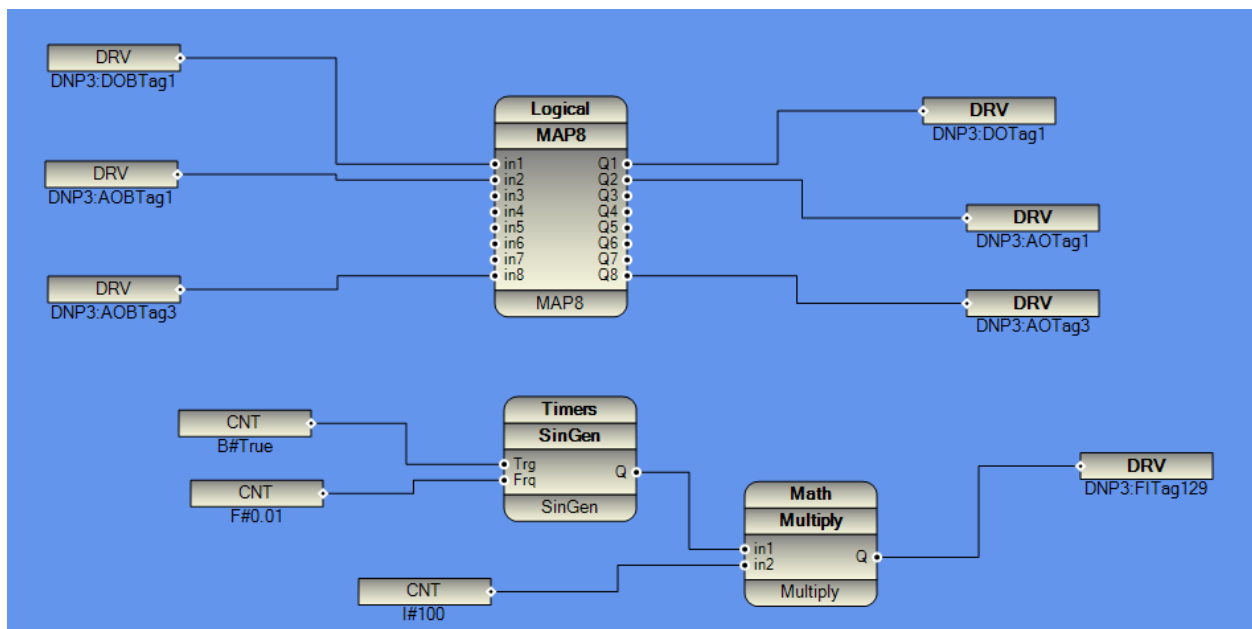


Figure 13

Transfer the logic and configuration to the RTU and reset the RTU. Then connect to the RTU with pbsSoftLogic and monitor the logic. Figure 14

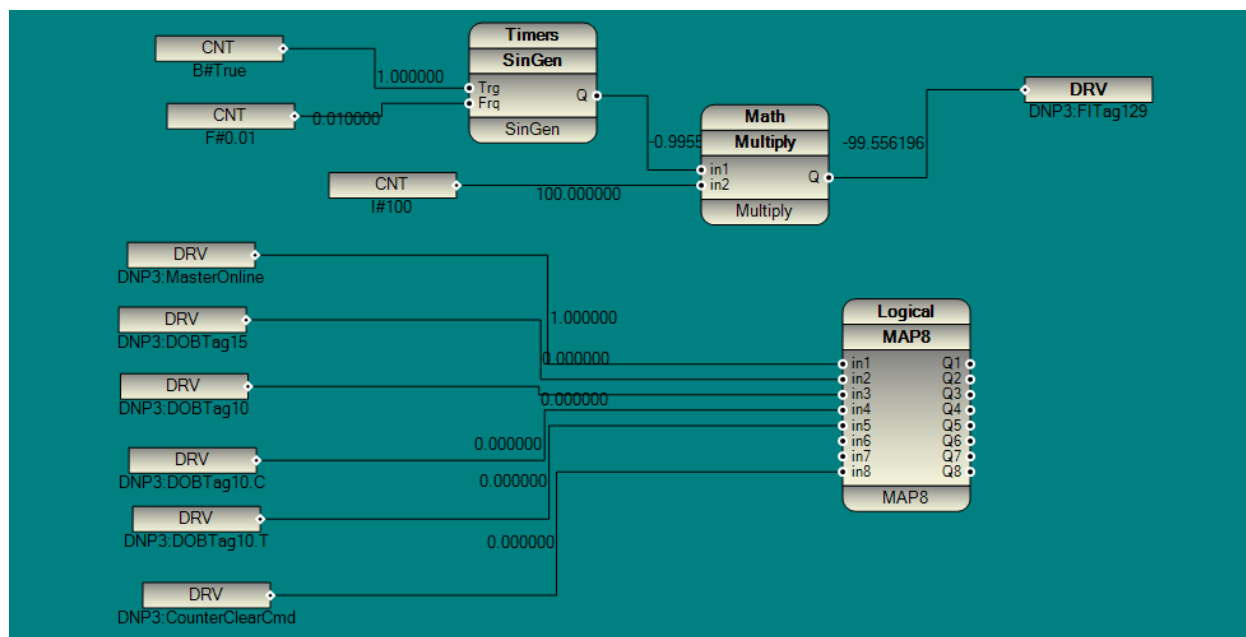


Figure 14

You can use pbsFDT or any other DNP3 tester and connect to the RTU. We use pbsFDT and send a AOB command to the AOBTag1 to the RTU. Figure 15

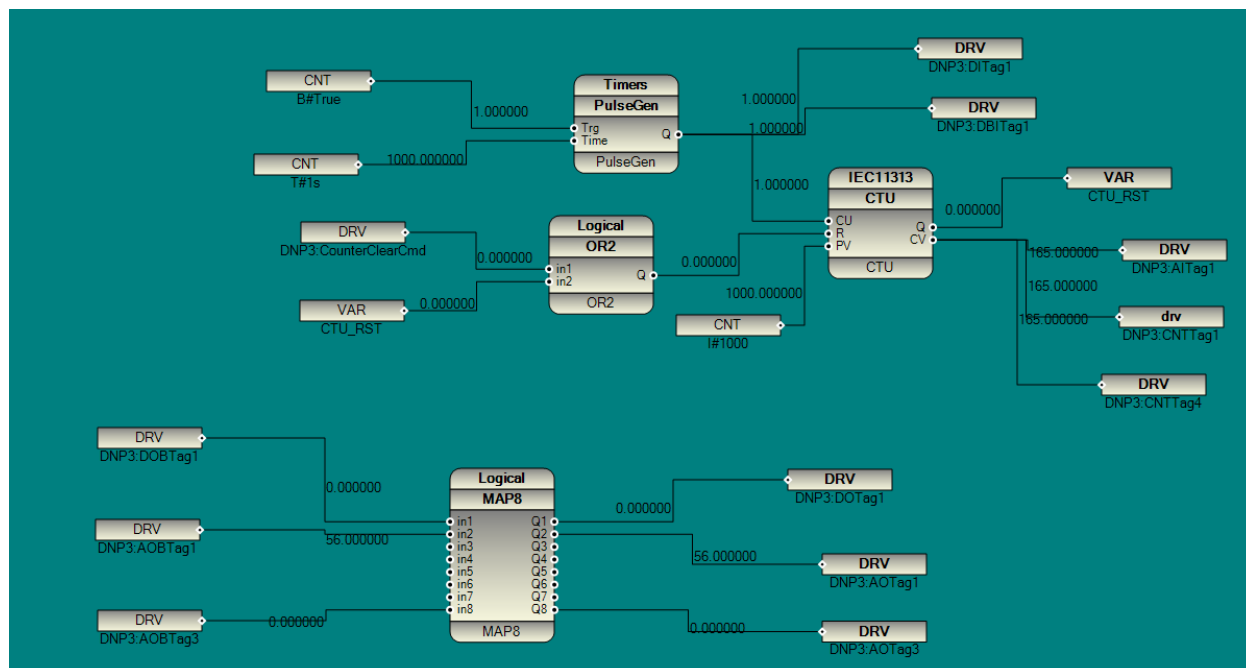


Figure 15

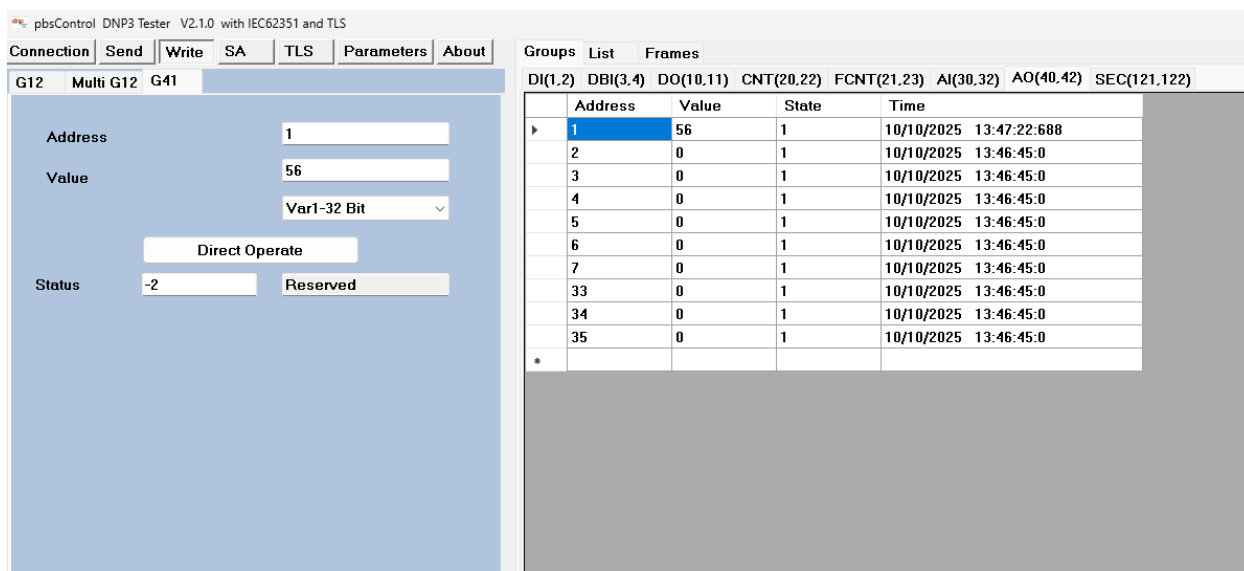


Figure 16

In Figure 17, Master is send DOB Pulse close for tag DOB Tag 10. as you can see the value of the signal and Its DOBC signal are change value to 1.

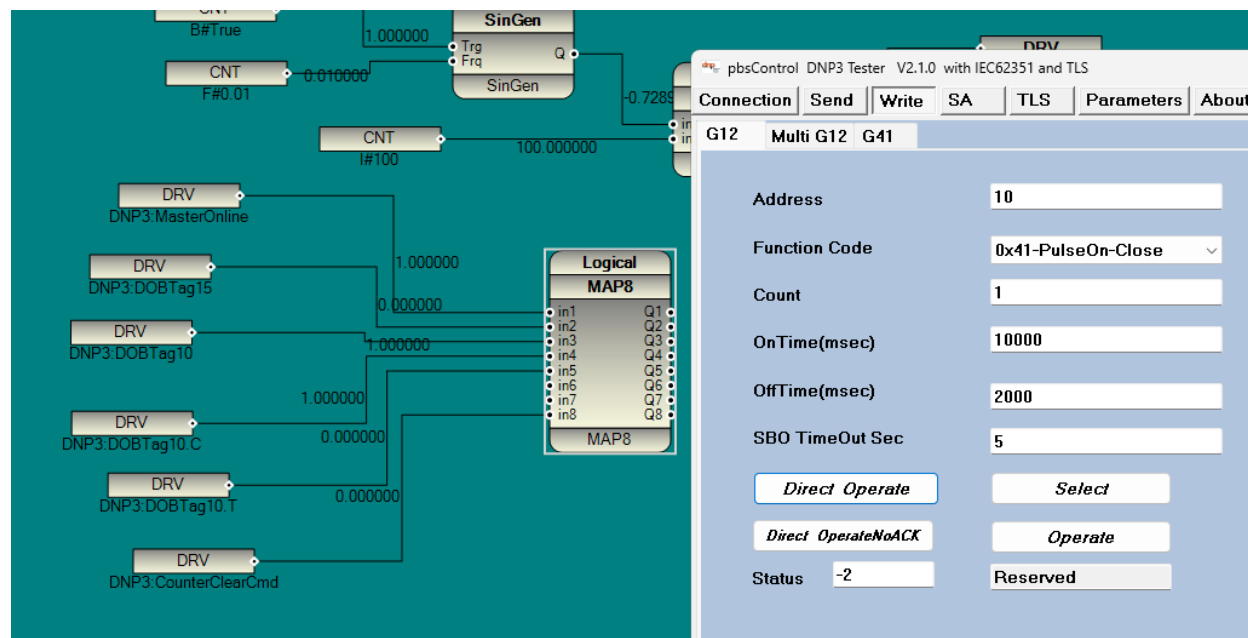


Figure 17

In figure 18 , master is send DOB Trip Command to DOB tag with address 10. DOBTag10 and DOBT tag are change to 1 .

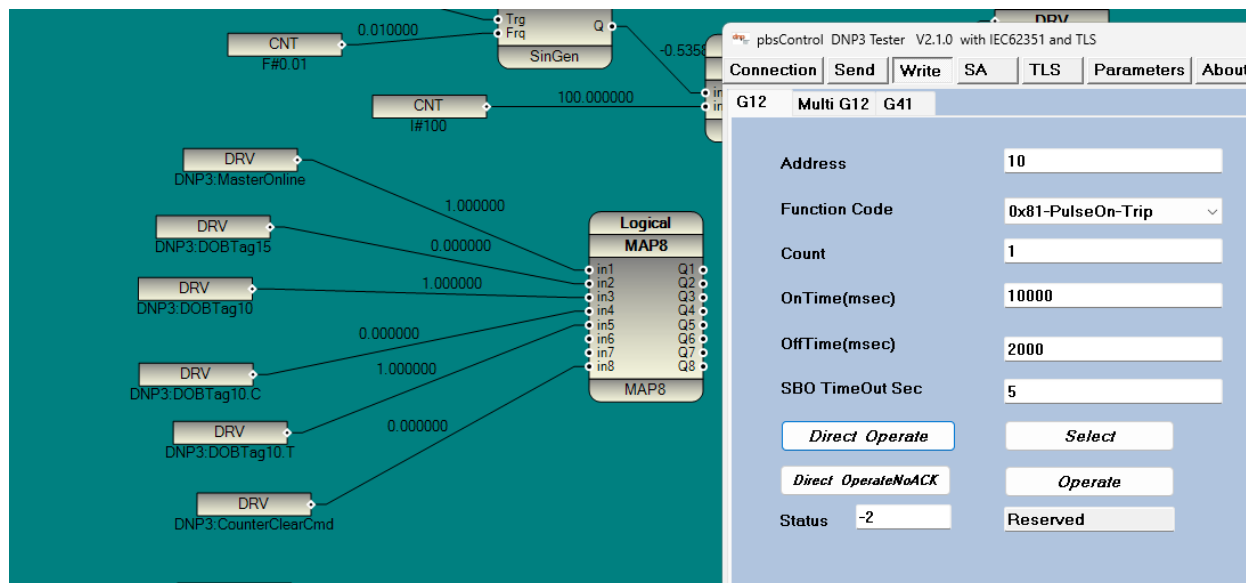


Figure 18

End of Document