# DNP3

# Configuration Guide

**0809-902-2301**

# Preface

## Scope of the User Guide

This guide provides basic information about the DNP3 protocol, and how to configure MOX equipment to implement it.

The guide has been organized for the operator, and it is expected that the user is an engineer, technician, electrician or similar with an understanding of the operating and programming requirements of the MOX system.

## Related Documents

A MOX system contains a collection of MOX equipments and several software packages. For this reason, a number of related documents should be read in conjunction with this user guide.

The related documents are noted below:

- MOX Unity Field Controller User Guide
- MOX IoNix Field Controller User Guide
- MoxIDE User Guide
- MoxGRAF User Guide

# Contents


**1 OVERVIEW ........ 1**

1.1 INTRODUCTION ........ 1
1.2 PROTOCOL COMPLIANCE ........ 2
1.3 PROTOCOL ARCHITECTURE ........ 3
*1.3.1 International Organisation for Standardisation (ISO) Model ........ 3*
*1.3.2 Enhanced Performance Architecture (EPA) Model ........ 3*
*1.3.3 Basic Message Structure ........ 5*
1.4 DATA COMMUNICATIONS ........ 6
*1.4.1 Quiescent Data Communications ........ 6*
*1.4.2 Unsolicited Report-by-Exception Data Communications ........ 6*
*1.4.3 Polled Report-by-Exception Data Communications ........ 6*
*1.4.4 Polled Static Data Communications ........ 6*
1.5 DATA CATEGORIES (OBJECTS) ........ 7
1.6 FLAGS/STATUS INDICATORS ........ 8
1.7 COMMUNICATION MEDIUMS ........ 9
*1.7.1 Serial Networks ........ 9*
*1.7.2 Ethernet ........ 9*
1.8 DNP3 INDEX ASSIGNMENT ........ 11

**2 DNP IMPLEMENTATION USING MOXIDE AND MOXGRAF ........ 12**

2.1 INTRODUCTION TO MOXIDE AND MOXGRAF ........ 12
*2.1.1 MoxIDE ........ 12*
*2.1.2 MoxGRAF ........ 12*
2.2 DNP OBJECT ........ 13
*2.2.1 DNP Data Types ........ 13*
*2.2.2 Class ........ 14*
*2.2.3 Map between MoxGRAF Data Types and DNP3 Data Types for Master ........ 14*
*2.2.4 Map between MoxGRAF Data Types and DNP3 Data Types for Slave ........ 15*

**3 MOX UNITY DNP3 MASTER CONFIGURATION ........ 16**

3.1 MOXIDE SETTINGS ........ 16
*3.1.1 General Tab ........ 16*
*3.1.2 Ports Tab ........ 17*
*3.1.3 DNP3 Tab ........ 18*
*3.1.4 Online Tab ........ 23*
3.2 MOXGRAF SETTINGS ........ 24
*3.2.1 Communication Control and Stat. Points ........ 27*
*3.2.2 Build and Download a MoxGRAF Project ........ 30*
3.3 MOX DNP3 MASTER REDUNDANCY ........ 31
*3.3.1 MOX DNP3 Master Path Redundancy ........ 31*
*3.3.2 MOX Unity System Redundancy ........ 32*

**4 MOX UNITY DNP3 SLAVE CONFIGURATION ........ 34**

4.1 MOXIDE SETTINGS ........ 34
*4.1.1 General Tab ........ 34*
*4.1.2 Ports Tab ........ 35*
*4.1.3 DNP3 Tab ........ 36*

*4.1.4 Online Tab* ..... *48*
4.2 MOXGRAF SETTINGS ..... 49
*4.2.1 Data Class Assignment* ..... *49*
*4.2.2 Event Object Assignment* ..... *51*
*4.2.3 Input Event Configuration* ..... *52*
*4.2.4 Periodic Event Generation* ..... *53*
*4.2.5 DNP3 Slave Point Controls* ..... *53*
*4.2.6 Build and Download a MoxGRAF Project* ..... *53*
4.3 MOX DNP3 SLAVE REDUNDANCY ..... 54
*4.3.1 Controller Redundancy* ..... *54*
*4.3.2 Data Path Redundancy* ..... *54*

**5 DNP3 TRANSFER EVENTS ..... 56**

**APPENDIX A DEVICE PROFILE (SLAVE) - MOX UNITY ..... 57**

**APPENDIX B INTEROPERABILITY TABLE (SLAVE) - MOX UNITY ..... 61**

**APPENDIX C DEVICE PROFILE (MASTER) - MOX UNITY ..... 63**

**APPENDIX D INTEROPERABILITY TABLE (MASTER) - MOX UNITY ..... 67**

**APPENDIX E PRODUCT SUPPORT ..... 69**

# Figures


Figure 1 OSI Reference Model ........ 3
Figure 2 EPA Model ........ 3
Figure 3 DNP3 Polled Sequence with Link Layer Confirmations ........ 4
Figure 4 DNP3 Message Structure ........ 5
Figure 5 MoxIDE – General Tab ........ 16
Figure 6 MoxIDE – Ports Tab ........ 17
Figure 7 MoxIDE – DNP3 Master Parallel Access Configuration ........ 18
Figure 8 MoxIDE – Editing DNP3 Master Data Paths ........ 20
Figure 9 MoxIDE – Online Tab ........ 23
Figure 10 MoxGRAF – Guide to DNP3 Master Setting ........ 24
Figure 11 MoxGRAF – DNP3 Master Devices Summary ........ 25
Figure 12 MoxGRAF – DNP3 Master Binary Input Configuration ........ 25
Figure 13 MoxGRAF – Editing Mapped Variable Parameters ........ 26
Figure 14 MoxGRAF – DNP3 Control and Stat. Points Configuration ........ 27
Figure 15 Data Path Redundancy Example 1 ........ 31
Figure 16 Data Path Redundancy Example 2 ........ 31
Figure 17 Data Path Redundancy Example 3 ........ 32
Figure 18 MOX Unity Redundancy ........ 33
Figure 19 MoxIDE – General Tab ........ 34
Figure 20 MoxIDE – Ports Tab ........ 35
Figure 21 MoxIDE DNP3 Tab ........ 36
Figure 22 MoxIDE – Enable DNP3 Slave Device ........ 37
Figure 23 MoxIDE – DNP3 Slave Ethernet Configuration ........ 38
Figure 24 UDP Server and Client Mode ........ 39
Figure 25 MoxIDE – DNP3 Slave Station Address Configuration ........ 40
Figure 26 MoxIDE – DNP3 Slave Data Paths Configuration ........ 41
Figure 27 MoxIDE – Editing DNP3 Slave Data Paths ........ 42
Figure 28 MoxIDE – DNP3 Slave Application Layer Configuration ........ 44
Figure 29 MoxIDE – DNP3 Slave Unsolicited Configuration ........ 45
Figure 30 MoxIDE – DNP3 Node Routing Table ........ 46
Figure 31 MoxIDE – Node Routing Table Configuration Dialog ........ 48
Figure 32 MoxGRAF – DNP3 Slave Setting ........ 49
Figure 33 MoxGRAF – Index Specific Settings ........ 50
Figure 34 MoxGRAF – Unsolicited Input Selction ........ 52
Figure 35 Controller Redundancy ........ 54
Figure 36 DNP Data Path Redundancy (Single Master) ........ 55
Figure 37 DNP Path Redundancy (Multiple Master) ........ 55
Figure 38 MoxGRAF – DNP3 Slave Event Object Configuration ........ 60

# 1 Overview

## 1.1 Introduction

DNP3 (Distributed Network Protocol, 3rd generation) is an industry standard communications protocol, and is used in many varying industries.

The protocol is primarily implemented using Half-Duplex procedures and Full-Duplex procedures. For Half-Duplex procedures, there are Point-to-Point and Multi-Point configurations. For Full-Duplex procedures, there are Point-to-Point, Multi-Point configurations and Dial-Up Modem.

A typical DNP3 protocol implementation describes rules for communications between points, for information such as static data requests, report by exception (RBE) data, polling regimes, and control operations. The protocol has the ability to:

- request and respond with multiple data types in single messages
- segment messages into multiple frames to ensure excellent error detection and recovery
- include only changed data in response messages
- assign priorities to data items and request data items periodically based on their priority
- respond without request (unsolicited)
- support time synchronization and a standard time format
- allow multiple masters and peer-to-peer operations
- allow user definable objects
- synchronize time and dates between different points and historical or logged data can be transferred as sequence of events (SOE).

## 1.2 Protocol Compliance

A device that is said to be DNP3 compliant should identify the year of the testing procedures, that the device was tested in compliance with, and also identify the subset of compliance.

As an example of this compliance identification, the MOX Unity is tested to DNP3-2004 Level 2 compliant: year 2004 testing procedures, level 2-subset functionality.

For DNP3 to be an open standard, function level subsets are used. This open structure is controlled by way of a minimum functional subset to which a device must adhere.

The DNP3 standard is strengthened by Subset Definitions document describing three minimum subset levels. A vendor's DNP3 implementation must be provided with a *Device Profile* document describing information required by the DNP User Group. This document includes details of the implementation of one of the three minimum subset levels, and other protocol information.

## 1.3 Protocol Architecture

### *1.3.1 International Organisation for Standardisation (ISO) Model*

The ISO (International Organisation for Standardisation) defines communication architecture that separates functions into seven layers. This architecture is called the Open System Interconnection (OSI) reference model.

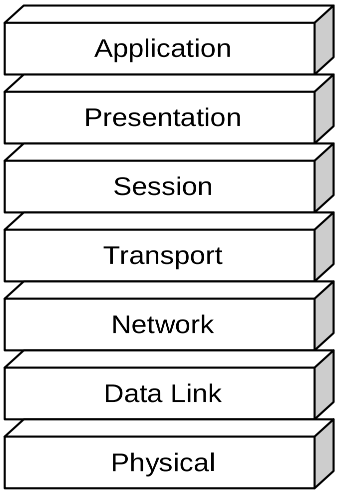



**Figure 1 OSI Reference Model**

### *1.3.2 Enhanced Performance Architecture (EPA) Model*

DNP3 protocol is based upon a simplified OSI model termed the Enhanced Performance Architecture (EPA) that consists of only three layers: Application, Transport, and Data Link.

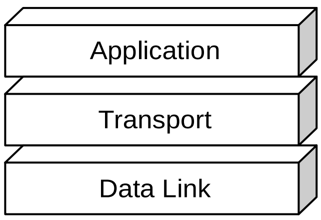



**Figure 2 EPA Model**

Each layer in the EPA model performs a set of functions required to communicate with the same layer in another device, relying on the next lower layer for more primitive functions.

The Application Layer describes the transmission procedure of the application user data.

The Transport Layer describes the quality and nature of the data delivery. As an example, this layers function would be to define if and how retransmissions will be used to ensure data delivery.

The Data Link Layer describes the logical organization of data bits transmitted on a particular medium. As an example, this layer function is to define the framing, addressing and check summing of data packets.

Figure below is an example of a typical EPA model polled data sequence, which shows the interaction between the EPA layers of both the polling device and polled device.

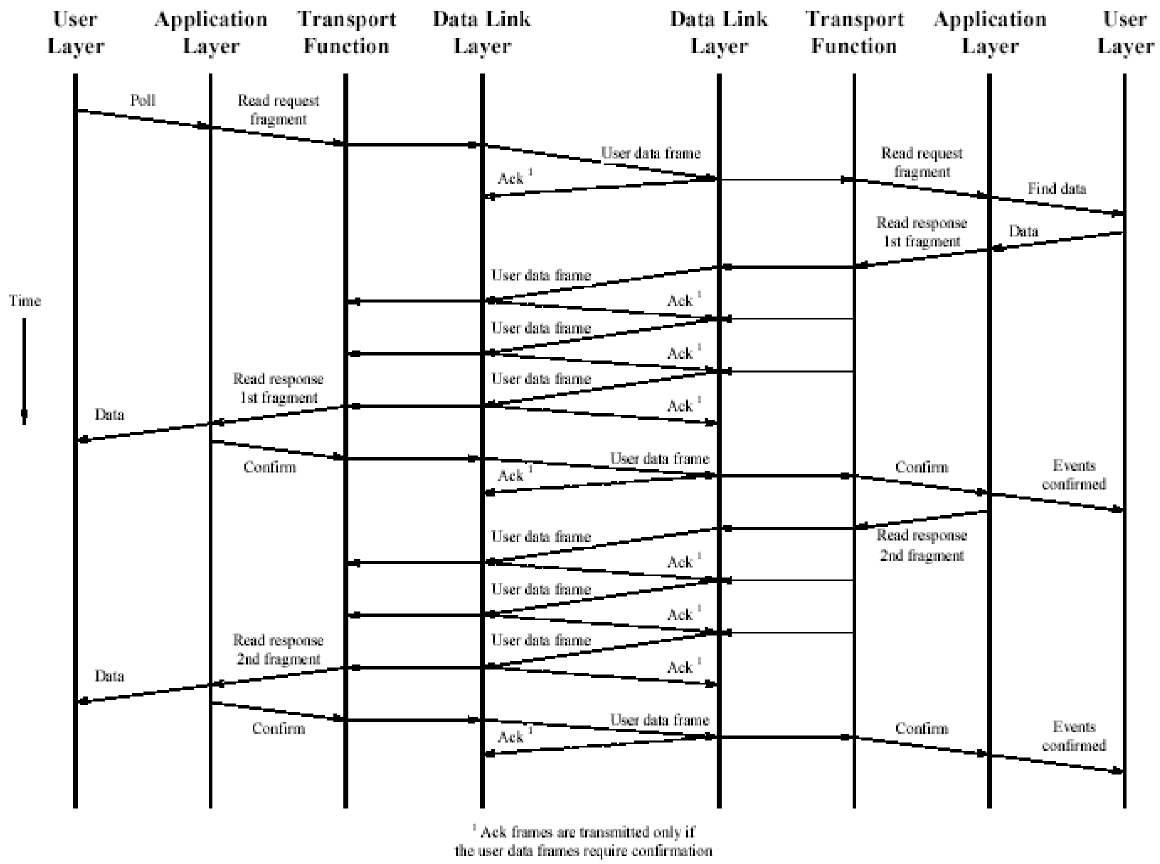



**Figure 3 DNP3 Polled Sequence with Link Layer Confirmations**

Each layer in the EPA model performs functions that are required to communicate with the same layer in another device using the EPA model. Each layer in the model relies on the next lower layer of the model for more primitive functions than itself.

At the sending device, each layer below the Application Layer receives data from the layer above for transmission. Each layer adds more information that enables the equivalent layer in the receiver to process the message.

At the receiving device, layers parse their own specific information, which has been added by the same layer at the sending site, and process the message appropriately. The layer control information is removed, and the message is passed to the next higher layer.

### 1.3.3 *Basic Message Structure*

Figure below shows a fragmented Application Layer message, segmentation of each fragment by the Transport Function, and how segments fit into Data Link Layer frames. This diagram is used to demonstrate how the higher level parts nest inside the lower layer structures. It also shows the relative positions of the Application Layer headers, the Transport Function headers and the Data Link Layer headers inside the message format.

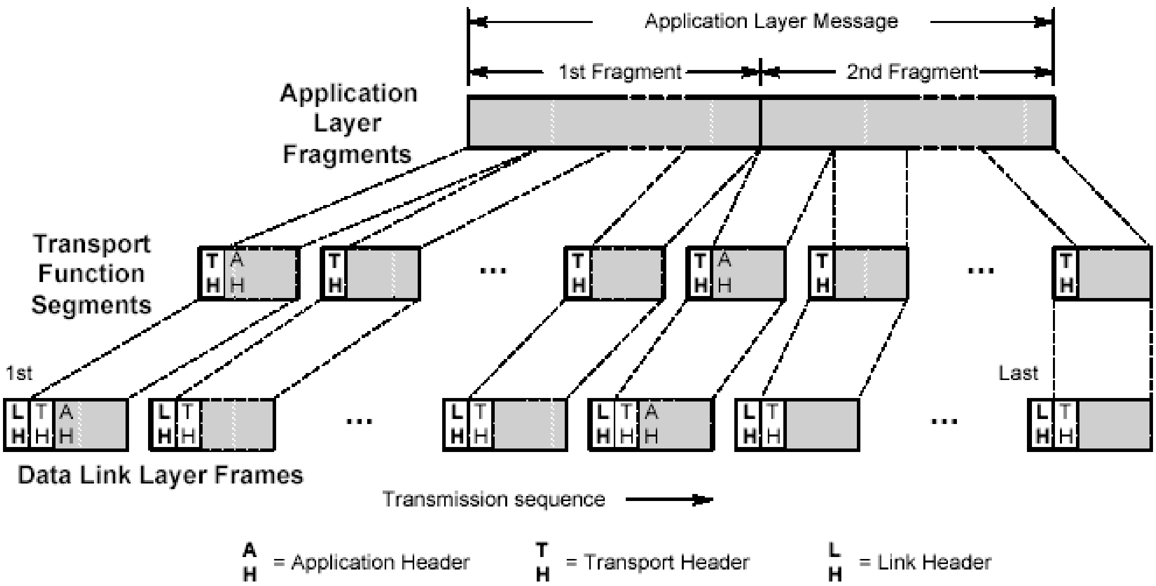



**Figure 4 DNP3 Message Structure**

## 1.4 Data Communications

DNP3 provides several different means of retrieving data.

These retrieval methods are detailed from most to least efficient:

- Quiescent
- Unsolicited Report-by-Exception
- Polled Report-by-Exception
- Polled Static

### *1.4.1 Quiescent Data Communications*

Quiescent Operation is a mode of operation in which the Master device never polls any Slave devices. The Master never sends data requests to a slave device. All communication is unsolicited report-by-exception from the slave device; however the Master still sends application layer confirmations to the Slave.

### *1.4.2 Unsolicited Report-by-Exception Data Communications*

The Unsolicited Report-by-Exception Operation is a mode of operation in which most communication is unsolicited, but the Master occasionally sends integrity polls for Class 0 Data to verify its database is up to date.

### *1.4.3 Polled Report-by-Exception Data Communications*

The Polled Report-by-Exception Operation is a mode of operation in which the Master polls frequently for event data and occasionally for Class 0 Data.

### *1.4.4 Polled Static Data Communications*

The Polled Static Operation is a mode of operation in which the Master polls only for Class 0 data or the specific data that is required.

## 1.5 Data Categories (Objects)

There are generally four different categories of data within each data type:

- Static Objects - objects that reflect the current value of the field point or software point.
- Event Objects - objects generated as a result of data changing or some other stimulant. These are historical objects reflecting the value of data at some time in the past.
- Frozen Static Objects - objects that reflect the current frozen value of the field point or software point. Data is frozen as a result of the data freeze requests.
- Frozen Event Objects - objects generated as a result of frozen data changing or some other stimulant. These are historical objects reflecting the value of changed data at some time in the past.

The MOX implementation of DNP3 only supports Static Objects and Event Objects.

## 1.6 Flags/Status Indicators

The flag/status indicator is a set of one-bit indications, each describing something about the associated data point. Each flag has a normal and an abnormal state. When all flags are in the normal state it is an indication of a valid data point value. An abnormal state in one or more flags indicates a nonstandard condition such as the value has been obtained from an alternate source or is not available.

For each data type, there are 8 indicators. The first 5 indicators are generic for all data types. The remaining 3 indicators are unique to the different data types.

The first 5 generic indicators are:

- On-line - this indicates that the data point has been read successfully. If this field is set to off-line, the value of the data point may not be correct.
    - For input data objects: If clear, the point is inactive or disabled (for example: powered-down, faulty, etc.) and unable to obtain field data.
    - For output status objects: If clear, the output point is inactive, unavailable, out-of-service, not installed or operating in local mode. The point may not be observable or may be not controllable. Commands sent to the point may fail.
- Restart – this indicates that the data has not been updated from the field since device reset.
    - For input data objects: If set, the object is in the initialization state, having a value that has never been updated from the field since restart. The bit is cleared when the object is first updated.
    - For output status objects: The Restart flag shall only be set while a device is restarting.
- Communication Lost – indicates that there is a communication failure in the path between the device where the data originates and the reporting device. This flag indicates that the value reported for the object may be stale. If set, the data value reported shall be the last value available from the originating device before communications were lost.
- Remote Forced – If set, the data value is overridden in a downstream reporting device. Only a non-originating device may set this flag. The Remote Forced or Local Forced flags are set in that object received from a downstream device. An originating device may never set this bit.
- Local Forced: If set, the data value is overridden in the device that is reporting this flag as set. This may be due to the device operating in a diagnostic or temporary mode or due to human intervention. If the value is forced in a non-originating device and overridden in a downstream device, then the non-originating device shall set both Remote Forced and Local forced flags.

In normal operation, the online flag is set (1) and the exception condition flags are all clear (0). Any other combination of the Online and exception condition flags indicates that the data value might not correctly indicate the value of the corresponding field object.

## 1.7 Communication Mediums

The communication mediums of DNP3 implementation include serial networks and Ethernet.

### *1.7.1 Serial Networks*

The protocol supports serial point-to-point communication (e.g. RS232, RS422) with limited support for half duplex serial networks (e.g. RS485).

***Full-duplex Point-to-Point***

In full-duplex point-to-point mode, both the master and the slave can transmit data at any time when needed.

***Full-duplex Multi-drop***

In full-duplex multi-drop mode, the slave's message will collide at random because there is no way for the slave to know if another station has control of the master's receive circuit. Hence whether the master can receive the slave's message, depends on the slave's data link retry.

***Full-duplex Dial-up***

In full-duplex dial-up mode, the slave's messages will require several control signals in order to transmit. The dial-up circuit is a point-to-point circuit. However, the meaning of the data carrier signal is quite different than with a direct circuit.

***Half-duplex Point-to-Point***

In half-duplex point-to-point, the slave's message will collide with the master's message. Hence whether the master can receive the slave's message depends on the slave's data link retry.

***Half-duplex Multi-drop***

In half-duplex multi-drop mode, the slave's message will collide with the master's message or another slave's message. Hence whether the master can receive the slave's message depends on the slave's data link retry.

### *1.7.2 Ethernet*

The master and slave driver supports connection-oriented (TCP) and connection-less (UDP) sockets. The protocol supports TCP and UDP communications.

***TCP***

When using TCP, the protocol can be the server end of TCP socket, the client end of TCP socket or both.

Default listen port 20000.

***UDP***

UDP has its own very robust mechanisms for insuring data delivery. The protocol's data link layer confirmation is disabled.

Default socket port 20000.

## 1.8 DNP3 Index Assignment

The MOX implementation of DNP3 supports 4 different data types:

- Binary Input (BI)
- Binary Output (BO)
- Analog Input (AI)
  - 16-Bit Analog Input
  - 32-Bit Analog Input
  - Floating Point Analog Input
- Analog Output (AO)
  - 16-Bit Analog Output
  - 32-Bit Analog Output
  - Floating Point Analog Output

Each of these 4 data types has an individual Index table. This Index is a unique number ranging from 0 to 65535.

For the Analog data types, there are some restrictions on the index assignment. For the AI data type, there are 3 formats:

- 16-Bit Analog Input
- 32-Bit Analog Input
- Floating Point Analog Input.

All AI data have one index table which is shared by 3 data formats. There cannot be any overlap between the data formats, e.g. a 16-bit AI index cannot be the same as a 32-bit AI, nor can the index be the same as a floating point AI index.

The AO data also has the same restrictions as the AI data type, e.g. a 16-bit AO index cannot be the same as a 32-bit AO, nor can the index be the same as a floating point AO index.

*Note: This restriction is valid for Citect SCADA, other SCADA applications may share the same index.*

Remember that the BI, BO, AI and AO data types have individual index tables. A BI index can be the same index as a BO, AI, or AO.

# 2 DNP Implementation using MoxIDE and MoxGRAF

## 2.1 Introduction to MoxIDE and MoxGRAF

### *2.1.1 MoxIDE*

***MoxIDE (MOX Integrated Development Environment*)** is a powerful network configuration interface that allows the user to upload and monitor device information. MoxIDE can be used to configure a communications network, alter configurations on the fly, monitor all device operations and change operational parameters of individual devices.

MoxIDE can be used to configure the DNP3 protocol settings, and to configure the MOX Unity onboard communication ports for use with DNP3.

To configure the MOX Unity for DNP3 communication, these are required configuration steps to be completed in MoxIDE:

1) Bind the MoxIDE settings to the MoxGRAF information (General).
2) Configure the communication ports on the MOX Unity (Ports).
3) Enable DNP3 Protocol on the MOX Unity (DNP 3.0).
4) Configure the DNP3 protocol settings (DNP 3.0).
5) Configure the DNP3 protocol routing tables for the DNP3 communication network (DNP 3.0).
6) Download the configuration to the MOX Unity (Online).

### *2.1.2 MoxGRAF*

***MoxGRAF*** is a development environment for designing applications without knowledge of complex, high-level computer languages. Using IEC 61131-3 programming standards, intuitive graphical and textual editors, results in robust applications developed with simplicity and in the shortest possible timeframe.

## 2.2 DNP Object

### *2.2.1 DNP Data Types*

The DNP application layer has an 8-bit object and an 8-bit variation field used to denote the data object. The 8-bit object denotes a general type of data such as static binary data. The variation of this object gives a different representation of the same data point, such as the size of the object or whether or not the object has flag information. There are generally four 5different categories of data within each data type, as outlined below:

- Static Objects: The objects that reflect the current value of the field point or software point.
- Event Objects: The objects that are generated as a result of data changing or some other stimulant. These are historical objects reflecting the value of data at some time in the past.
- Frozen Static Objects: The objects that reflect the current frozen value of the field point or software point. Data is frozen as a result of the data freeze requests. (Refer to the *DNP V3.00 Application Layer*, P009-0PD.APP.)
- Frozen Event Objects: The objects that are generated as a result of frozen data changing or some other stimulant. These are historical objects reflecting the value of changed data at some time in the past.

Each category should be represented with a different object. All the classes of a different data type should also be organized in the same range of object numbers. So far, the following groupings have been created for all traditional SCADA/DA data types and several non-traditional data types. These are as follows:

- Binary Input
- Binary Output
- 16-Bit Analog Input
- 16-Bit Analog Output
- 32-Bit Analog Output
- 32-Bit Analog Input
- Short Floating Point Analog Input
- Short Floating Point Analog Output

Supported data types for MOX Controller are as follows:

| Data Type | Size (bits) |
|---|---|
| BOOL | 8 |
| SINT | 8 |
| DINT | 32 |
| REAL | 32 |

### 2.2.2 *Class*

All DNP devices have four definable groups (classes) of data. These classes are numbered 0~3.

Class 0 is used to show their current values or static values.

Classes 1, 2, 3 are used for event objects (changes in value). Classes 1, 2, 3 allow for event objects to be separated into groups or priorities. This separation into groups then allows for prioritising of change event data.

By associating different change event data with different classes, the classes can also be requested with varying periodic rates.

### 2.2.3 *Map between MoxGRAF Data Types and DNP3 Data Types for Master*

| MoxGRAF Data Types | Direction | Mapped | DNP3 Data Types |
|---|---|---|---|
| BOOL | Input, Internal | ⇔ | Binary Input |
| BOOL | Output, Internal | ⇔ | Binary Output |
| DINT | Input, Internal | ⇔ | 16-Bit Binary Counter |
| DINT | Input, Internal | ⇔ | 32-Bit Binary Counter |
| DINT | Input, Internal | ⇔ | 16-Bit Analog Input |
| DINT | Output, Internal | ⇔ | 16-Bit Analog Output |
| DINT | Input, Internal | ⇔ | 32-Bit Analog Input |
| DINT | Output, Internal | ⇔ | 32-Bit Analog Output |
| REAL | Input, Internal | ⇔ | Short Floating Point Analog Input |
| REAL | Output, Internal | ⇔ | Short Floating Point Analog Output |

**Table 1 Map between MoxGRAF and DNP3 Data Types for Master**

### 2.2.4 Map between MoxGRAF Data Types and DNP3 Data Types for Slave

| MoxGRAF Data Types | Direction | Mapped | DNP3 Data Types |
|---|---|---|---|
| BOOL | Input, Internal | ⟺ | Binary Input |
| BOOL | Output, Internal | ⟺ | Binary Output |
| SINT | Input, Internal | ⟺ | 16-Bit Binary Counter |
| SINT, DINT | Input, Internal | ⟺ | 32-Bit Binary Counter |
| SINT | Input, Internal | ⟺ | 16-Bit Analog Input |
| SINT | Output, Internal | ⟺ | 16-Bit Analog Output |
| SINT, DINT | Input, Internal | ⟺ | 32-Bit Analog Input |
| SINT, DINT | Output, Internal | ⟺ | 32-Bit Analog Output |
| REAL | Input, Internal | ⟺ | Short Floating Point Analog Input |
| REAL | Output, Internal | ⟺ | Short Floating Point Analog Output |

**Table 2 Map between MoxGRAF and DNP3 Data Types for Slave**

# 3 MOX Unity DNP3 Master Configuration

## 3.1 MoxIDE Settings

### *3.1.1 General Tab*

The General tab is where MoxIDE configuration is bound to MoxGRAF. In order to be able to do this bind, a MoxGRAF program will need to exist.

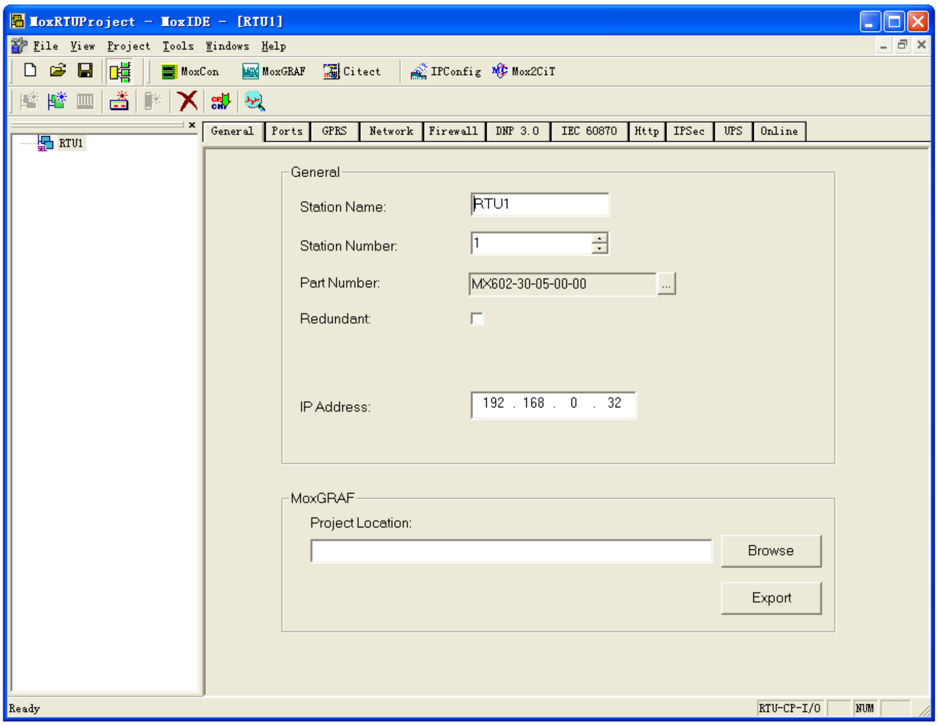



**Figure 5 MoxIDE – General Tab**

The *Station Name* is an arbitrary name given to a device for identification purpose only.

The *Station Number* is an identifier for the MOX Unity for other communication protocols.

The *Part Number* is used to select desired features of MOX Unity.

The *Redundant* option is used to configure the MOX Unity for use in a redundant configuration. This feature is not supported when using the DNP3 protocol, as the DNP3 configuration does not support multiple masters.

The *IP Address* is of the MOX Unity to where the configuration is to be downloaded.

The *MoxGRAF – File Name* is the MoxGRAF project name to link the MoxIDE project to. This field also includes the file path to where the project structure is located.

### *3.1.2 Ports Tab*

The Ports tab is where the user is able to configure the communication ports on the MOX Unity.

*COM 0* is used to configure the onboard PSTN data modem option.

*COM 1~4* are the onboard serial communications ports.

Each communication port is configurable to a type of:

- RS232 serial connection
- RS485 serial connection
- Modem connection.

The user can individually configure the baud rate, parity, stop bits, and flow of each communication port.

The MOX Unity also supports many different communication protocols.

For the MOX Unity to use the DNP3 protocol, the "DNP" needs to be selected for each port that will be communicating using the DNP3 protocol.

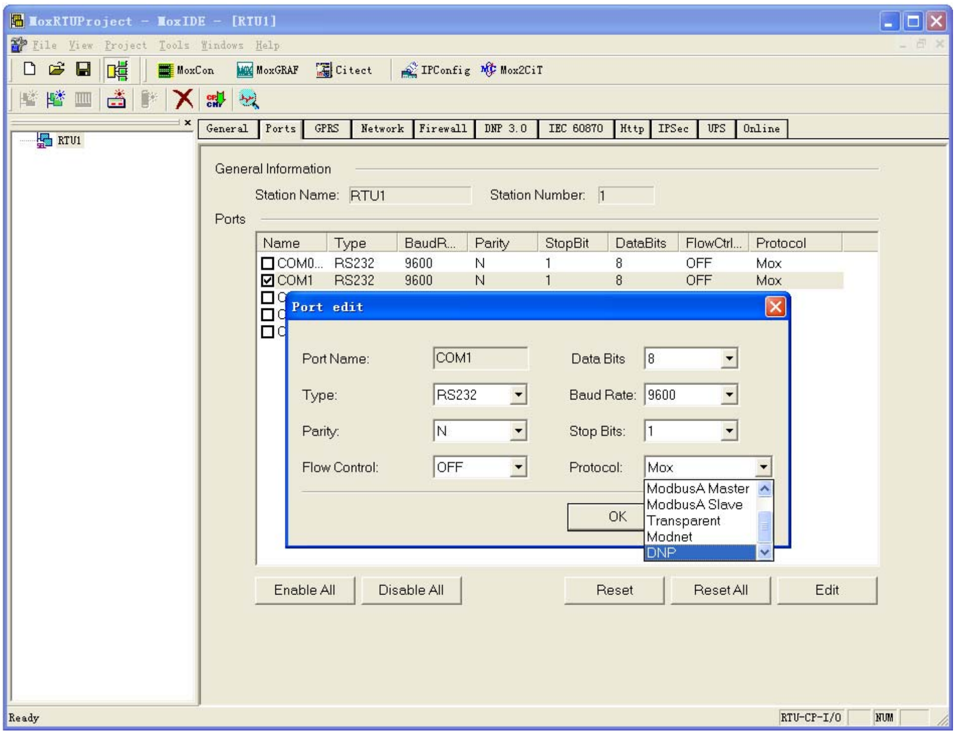



**Figure 6 MoxIDE – Ports Tab**

### 3.1.3 DNP3 Tab

***Parallel Access***

When attaching multiple masters to a single port, each device can access the port any time without waiting for another device to finish.

*Note: Only functional when the underlying media has robust collision avoidance handling implemented.*

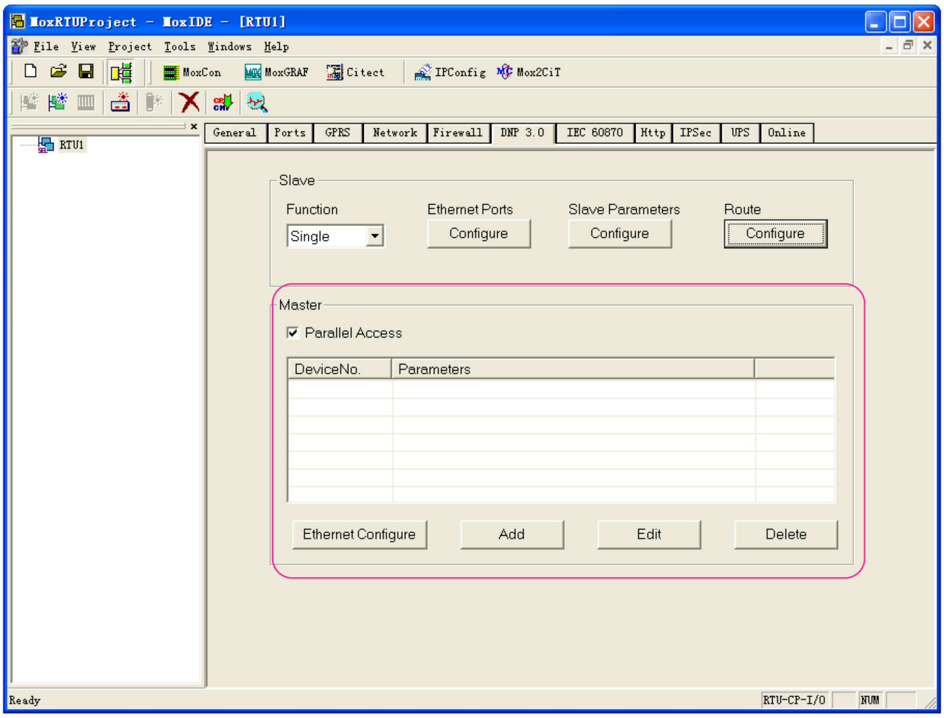
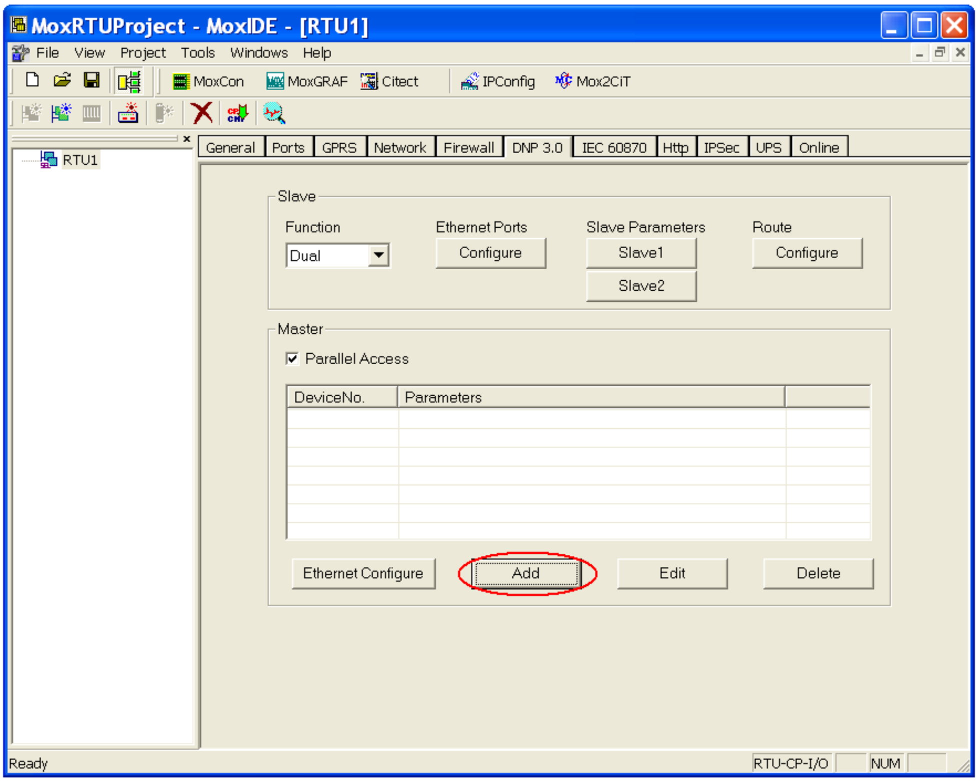
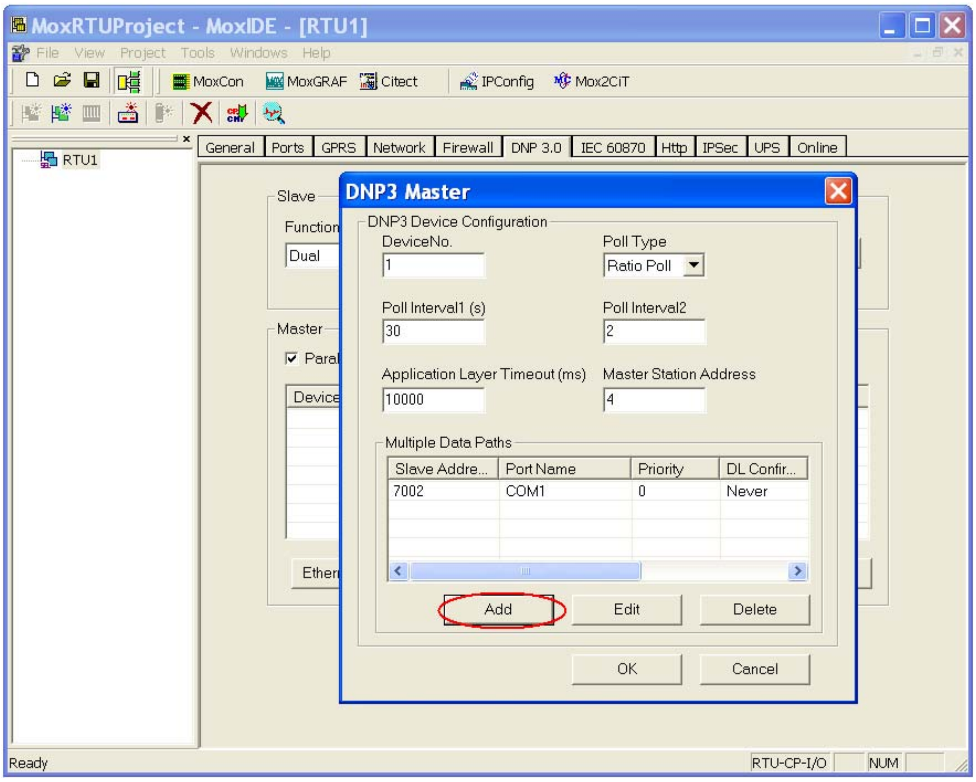



**Figure 7 MoxIDE – DNP3 Master Parallel Access Configuration**

### *Master Parameters Configuration*

Please follow the steps below:



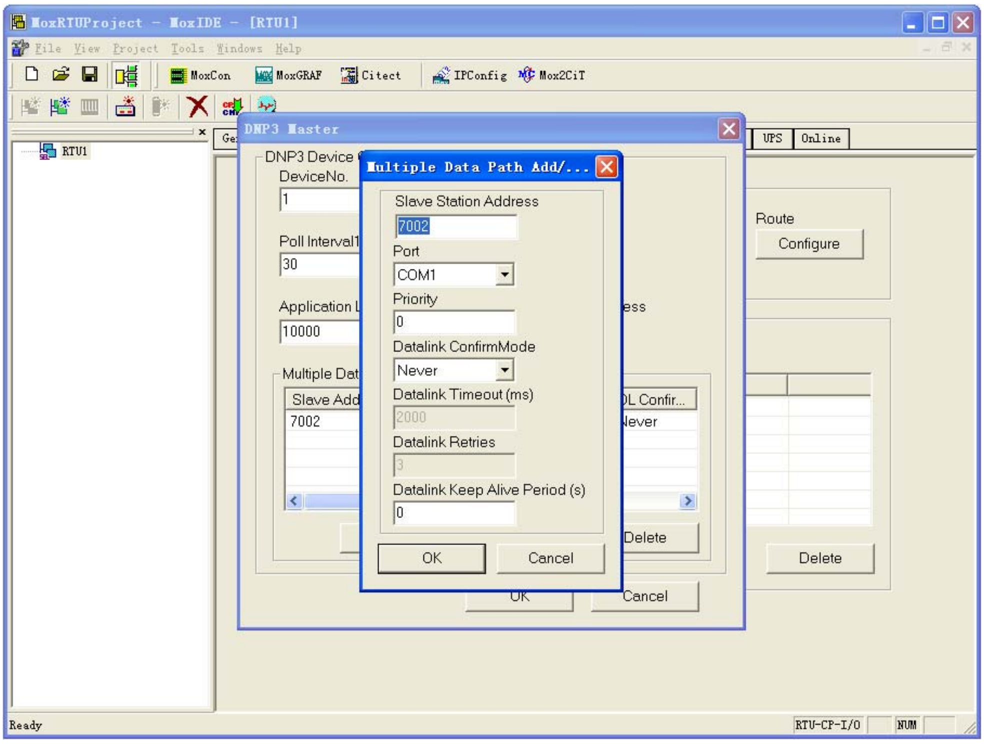
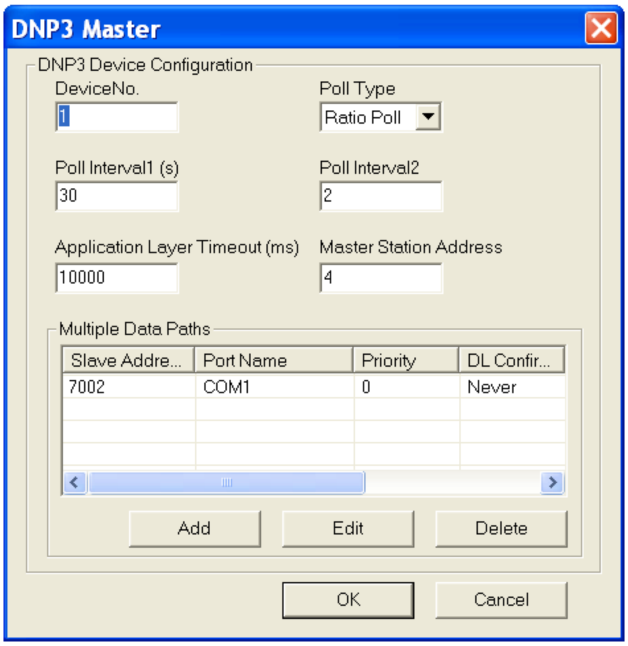


Figure 8 MoxIDE – Editing DNP3 Master Data Paths

### Multiple Data Paths Description

1) Maximum data paths is 5 for every logical DNP3 slave device
2) Multiple data paths are used for data path redundancy
3) Slave address can be different per data path

### Parameters Description



*Device No.*

The 'DeviceNo.' is used for MoxGRAF and MoxIDE to match parameters.

*Poll Type, Poll Interval 1, Poll Interval 2*

- Poll type – ratio poll
  - Poll interval 1 - specifies the interval between successive integrity polls (seconds)
  - Poll interval 2 - specifies how many event polls are issued between successive integrity polls
- Poll type – static poll
  - Poll interval 1 – specifies the interval between successive static polls (seconds)
  - Poll interval 2 – not used
- Poll type – no poll
  - Poll interval 1 – not used
  - Poll interval 2 – not used

*Application Layer Timeout*

Maximum time to wait for an application acknowledgement (ack) - 100 ~ 20,000,000 millisecond

*Master Station Address*

The DNP address of the Master - 0 ~ 65,520

*Data Path -> Slave Station Address*

The DNP address of the slave - 0 ~ 65,520

*Data Path -> Port*

Serial port or Ethernet port name

*Data Path -> Priority*

If two or more data paths are configured, the highest priority is considered the primary communication data path. All other communication data paths are standby data path. (If two data paths' priorities are the same, then the first data path will be the primary data path).

*Data Path -> Datalink ConfirmMode*

1) Never - Never request for Data Link confirmation
2) Always - Always request Data Link confirmation
3) Sometimes - Request Data Link confirmation on multi-frames only

*Data Path -> Datalink Timeout*

The maximum time to wait for a Data Link confirmation response. This parameter must be set even if the Datalink ConfirmMode is Never. Because other Data Link functions (such Request Link Status, Reset of remote link, etc) will use it - 100 ~ 20,000,000 millisecond.

*Data Path -> Datalink Retries*

The number of retries before abandoning transmission of Data Link data - 0 ~ 65,535.

*Data Path -> Datalink Keep Alive Period*

Send Request Link Status, whenever the line is idle.

- 0 disables the keep alive check
- 0 ~ 86,400 second

### 3.1.4 Online Tab

The Online tab is used to view basic configuration data that is currently running on the controller, and to download controller configuration parameters.

To download a DNP3 configuration to the controller, select the “Online” button, the Systems statistics field should populate. When the fields are populated, select the “Download” button from the Configuration section.

From the “Download” button, a new popup should appear. This popup will allow the user to download controller configuration settings.

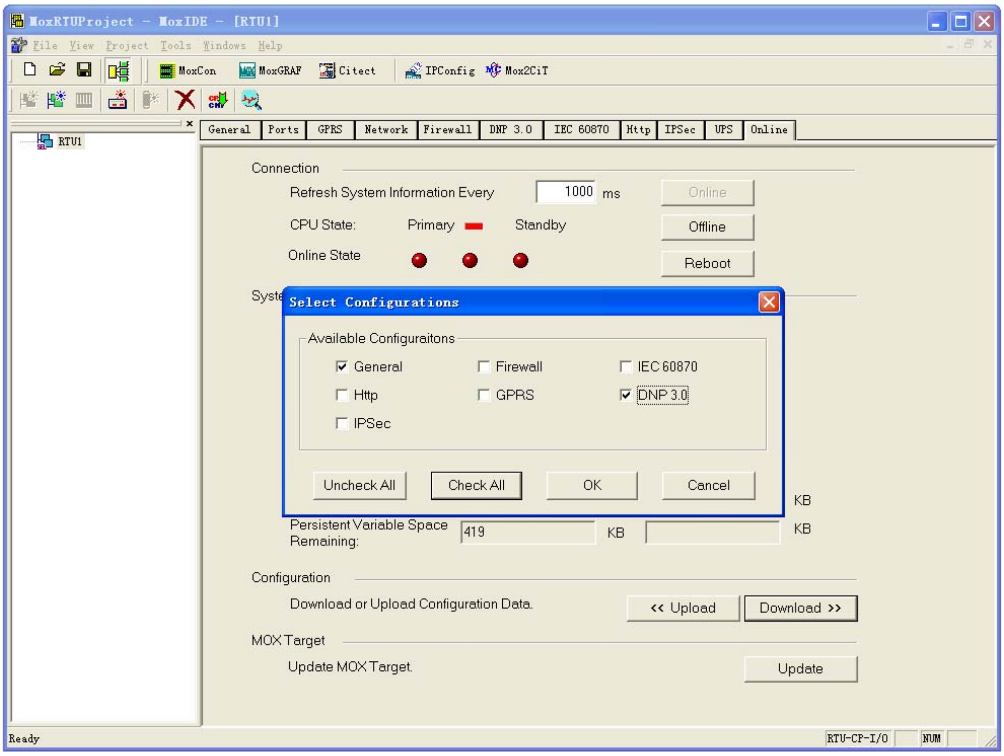



**Figure 9 MoxIDE – Online Tab**

To download the configuration, the “General” downloads the communication ports set up where the DNP3 protocol is assigned to a communication port, and the “DNP 3.0” downloads the DNP3 protocol configurations.

## 3.2 MoxGRAF Settings

MoxGRAF is used to configure the DNP3 protocol data addressing, and is used as the programming interface for the MOX Unity.

- Run MoxGRAF
- Open/Create project
- Open the link architecture page
- Right click Resource 1 page
- Select DNP3 master setting

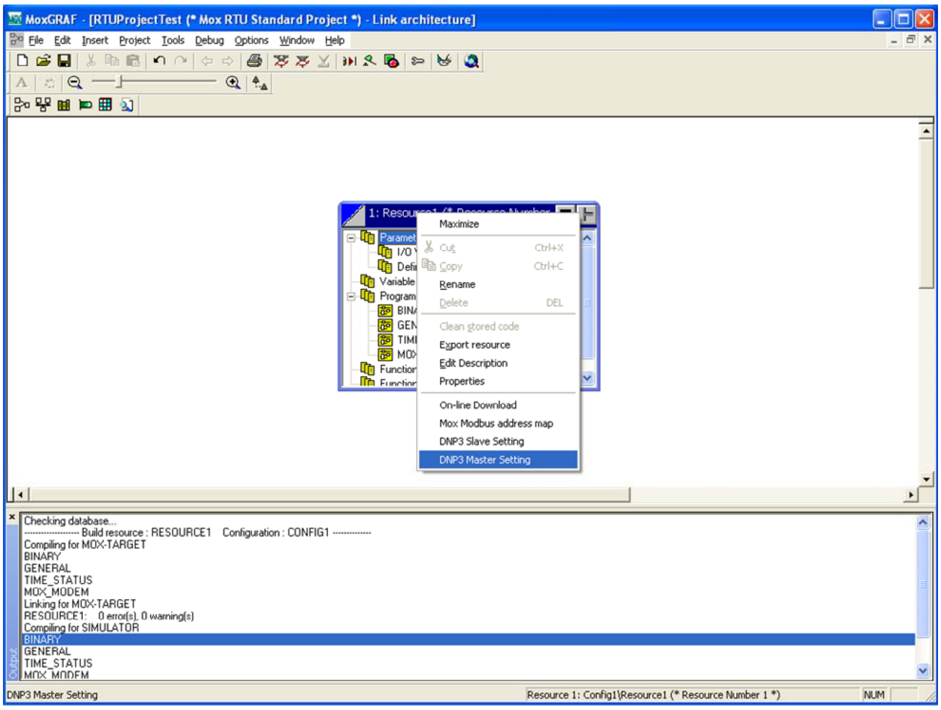



Figure 10 MoxGRAF – Guide to DNP3 Master Setting

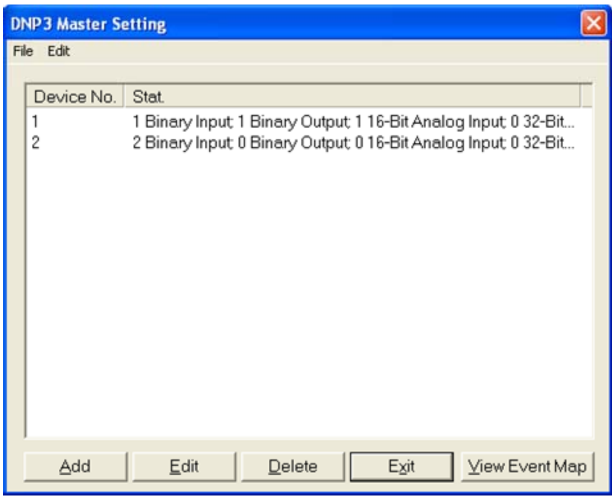



**Figure 11 MoxGRAF – DNP3 Master Devices Summary**

### *Add/Edit One DNP3 Master Device's Setting*

Click **"Add"** or **"Edit"** button, DNP3 Master variables map dialog pops up:

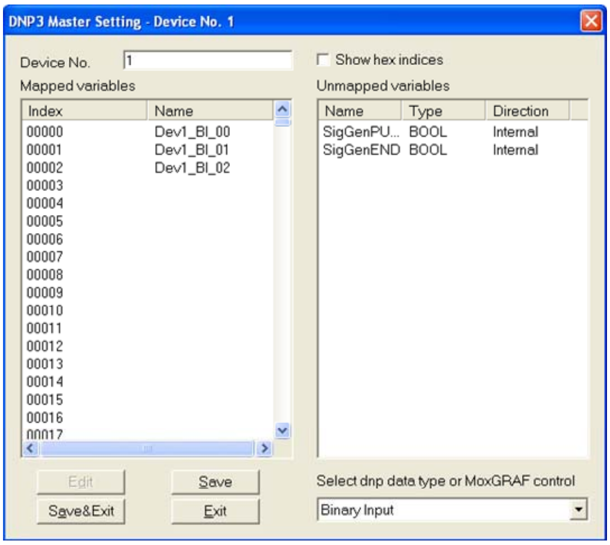



**Figure 12 MoxGRAF – DNP3 Master Binary Input Configuration**

*Device No.*

The Device No. is used for MoxGRAF and MoxIDE to match parameters so the MOX DNP3 master knows which slaves it is communicating with and the type of communication to be used.

*Mapped Variables*

Select an index in the Mapped variables list (left) then in Unmapped variables list (right) double click the variable to be mapped. Click the **"Save"** button to save to new mapped table.

*Un-mapped Variables*

In Mapped variables list (left) double click the variable to be unmapped. Click the **"Save"** button to save the new mapped table.

*Editing Mapped Variable Parameters*

In the mapped variables list (left), select the variable to edit and click the **"Edit"** button. Change the desired settings before selecting the **"Ok"** button to close edit window then click the **"Save"** button to save the new mapped table.

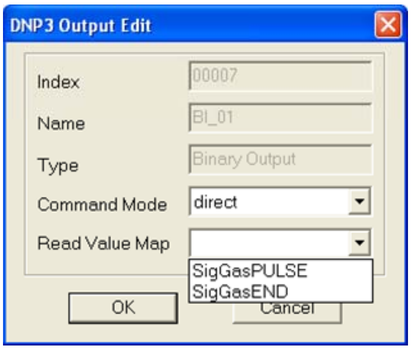



**Figure 13 MoxGRAF – Editing Mapped Variable Parameters**

*Note: Only Binary Output, 16-Bit Analog Output, 32-Bit Analog Output, Short Floating point and Analog Output mapped variables have this setting.*

**Read Value Map** is used to store the output values read from DNP3 Slave devices.

Command Mode type

- sbo -> SELECT and OPERATE (FUNCTION CODE 3 and 4)
- direct -> DIRECT OPERATE (FUNCTION CODE 5)
- directnoack -> DIRECT OPERATE - NO ACKNOWLEDGEMENT (FUNCTION CODE 6)

These variables are used for controlling the master and store statistical information.

| Function | MoxGRAF Data Type |
|---|---|
| Online Control | BOOL |
| Cold Restart | BOOL |
| Warm Restart | BOOL |
| Force Static Poll | BOOL |
| Force Event Poll | BOOL |
| Force Integrity Poll | BOOL |
| Delay Measurement | BOOL |
| Enable/Disable Unsolicited | DINT |
| Poll Counter | DINT |
| Poll Error Counter | DINT |
| Unsolicited Counter | DINT |
| Device Status | DINT |

| Function | MoxGRAF Data Type |
|---|---|
| Last IIN (Internal Indication) | DINT |
| Clear Stat. | DINT |

### 3.2.1 Communication Control and Stat. Points

These point ranges in the DNP3 Master are used to report communication status, and also to provide specific device control points for a DNP3 Slave device.

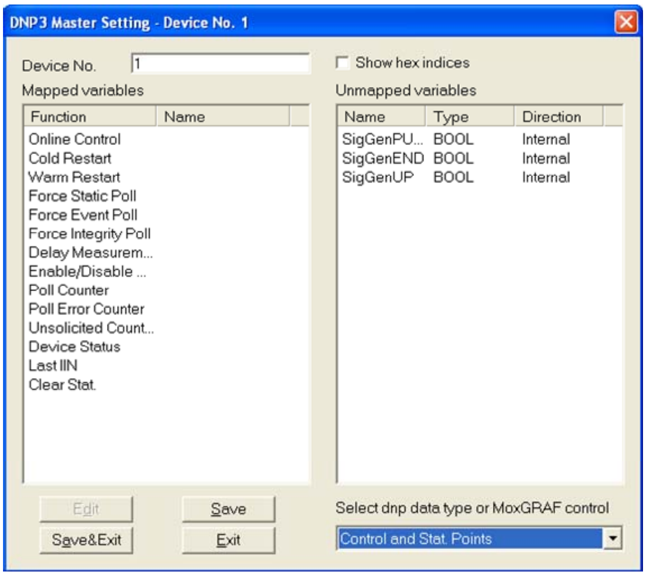



**Figure 14 MoxGRAF – DNP3 Control and Stat. Points Configuration**

#### 3.2.1.1 Digital Control Points

***Online and Offline Control***

This is a control point that can be used to either bring the DNP3 Slave device online (enable the device) or to manually put the DNP3 Slave device offline (disable the device).

- When the digital point is ON (TRUE), the appropriate DNP3 Slave device is put online.
- When the digital point is OFF (FALSE), the appropriate DNP3 Slave device is put offline.

When the DNP3 Slave device is put offline, the DNP3 Master (MOX Unity) does not initiate communication with the device.

*Note: The default value (No MoxGRAF variable is mapped) of the control point is TRUE (Online).*

***Force Cold Restart (Pulsed)***

Pulsing this point with a 'positive edge' transition will invoke a cold restart to the relevant DNP3 Slave device.

**_Force Warm Restart (Pulsed)_**

Pulsing this point with a ‘positive edge’ transition will invoke a warm restart to the relevant DNP3 Slave device.

**_Force Static Poll (Pulsed)_**

Pulsing this point with a ‘positive edge’ transition will invoke a static poll (class 0) to the relevant DNP3 Slave device.

**_Force Event Poll (Pulsed)_**

Pulsing this point with a ‘positive edge’ transition will invoke an event poll (class 1, 2, 3) to the relevant DNP3 Slave device.

**_Force Integrity Poll (Pulsed)_**

Pulsing this point with a ‘positive edge’ transition will invoke an integrity poll (class 0, 1, 2, 3) to the relevant DNP3 Slave device.

**_Delay Measurement (Pulsed)_**

Pulsing this point with a ‘positive edge’ transition will invoke delay measurement to the relevant DNP3 Slave device.

#### 3.2.1.2 Analog Communication Status Points

**_Poll Counter_**

Count of successful integrity, event, class1, class2 and class3 polls.

**_Poll Error Counter_**

Count of failed integrity, event, class1, class2 and class3 polls.

**_Unsolicited Response Counter_**

Count of unsolicited responses received by the DNP3 Master (MOX Unity) from a given DNP3 Slave device.

**_DNP3 Slave Device Status_**

Current status of the DNP3 Slave device. 0 Offline, 1 Online.

**_Last Reported IIN Flag_**

The most recently received IIN from a particular DNP3 Slave device. Use MoxGRAF DINT variable to map the IIN flag.

DINT byte order:

| | | | |
|---|---|---|---|
| **Byte 3** | **Byte 2** | **Byte 1** | **Byte 0** |

.

Byte 0 stores second octet of IIN flag. Byte 1 stores first octet of IIN flag. Byte2, Byte 3 are reserved, always 0.

## IIN Definition

| First Octet | Second Octet |
|---|---|

First Octet

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Bit number |
|---|---|---|---|---|---|---|---|---|

A one (1) in the bit position indicates the described state.

**Bit 0** - All stations message received

**Bit 1** - Class 1 data available

**Bit 2** - Class 2 data available

**Bit 3** - Class 3 data available

**Bit 4** - Time-synchronization required from the master.

**Bit 5** - Set when some or all of the Outstation's digital output points are in the Local state. That is, the Outstation's control outputs are NOT accessible through the DNP protocol.

**Bit 6** - Device trouble

**Bit 7** - Device restart

Second Octet

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Bit number |
|---|---|---|---|---|---|---|---|---|

**Bit 0** - Function code not implemented

**Bit 1** - Requested object(s) unknown.

**Bit 2** - Parameters in the qualifier, range or data fields are not valid or out of range. This is a catch-all for application request formatting errors.

**Bit 3** - Event buffer(s), or other application buffers have overflowed. For example, COS/SOE buffers have overflowed.

**Bit 4** - Request understood but requested operation is already executing.

**Bit 5** - Set to indicate that the current configuration in the Outstation is corrupt and that the master application layer should inform the user of this exception. The master may download another configuration to the Outstation. Note that sometimes a corrupt configuration will disable an Outstation, making it impossible to communicate this condition to a master station.

**Bit 6** - Reserved for use by agreement, currently always returned as zero (0).

**Bit 7** - Reserved for use by agreement, currently always returned as zero (0).

#### 3.2.1.3 Analog Control Points

***Enable and Disable Unsolicited***

This is a control point that can be used to either enable unsolicited responses or to disable unsolicited responses.

0 jumps to 1, enable unsolicited response.

0 jumps to 2, disable unsolicited response.

***Communication Status Clear (Clear Stat.)***

0 jumps to 1, clear Poll Counter

0 jumps to 2, clear Poll Error Counter

0 jumps to 3, clear Unsolicited Response Counter

### 3.2.2 Build and Download a MoxGRAF Project

After all necessary components are completed; the MoxGRAF project can be built and downloaded to the target controller. In the MoxGRAF toolbar, click Build Project/Library button to build the configuration file. The configuration file then needs to be downloaded to the controller.

Please refer to the MoxGRAF User Guide for specific details of configuring and downloading project configurations.

## 3.3 MOX DNP3 Master Redundancy

Data path redundancy on the active MOX Unity is only supported.

- If all communication paths on the active MOX Unity are lost or in error, the MOX Unity will not fail over to the standby MOX Unity. That is to say, the status of the communication paths on the active Unity will not affect the inactive Unity.

Event data synchronization between active and inactive MOX Unity is not supported.

- If the active MOX Unity fails all events are lost as none are stored in the inactive controller.

### *3.3.1 MOX DNP3 Master Path Redundancy*

Parallel communication data paths from the MOX Unity to the DNP3 Slave devices can be configured. A parallel data path ensures that if one communication data path fails, the MOX Unity can continue without interruption by using another communications data path.

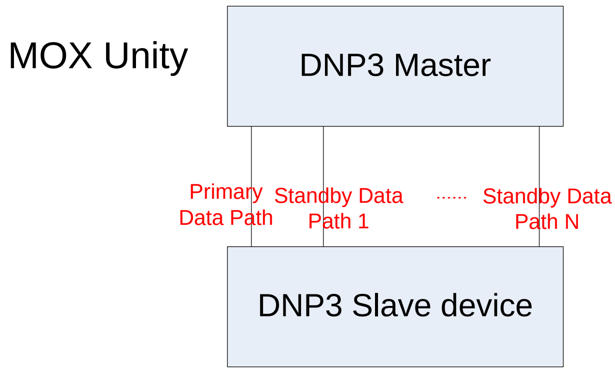



**Figure 15 Data Path Redundancy Example 1**

If the DNP3 Slave devices support peer-to-peer communication, the MOX Unity can provide the communications data path redundancy as shown here:

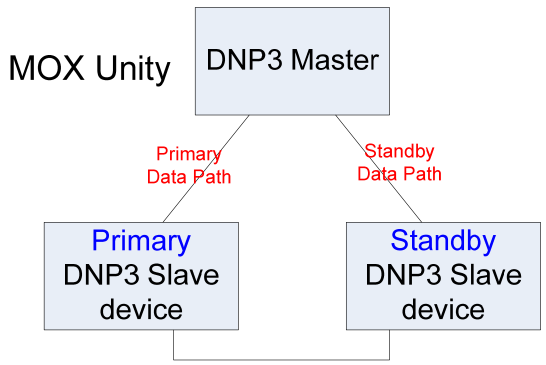



**Figure 16 Data Path Redundancy Example 2**

If the DNP3 Slave devices support peer-to-peer communication and parallel communication data paths, the MOX Unity can provide communication data path redundancy as shown here:

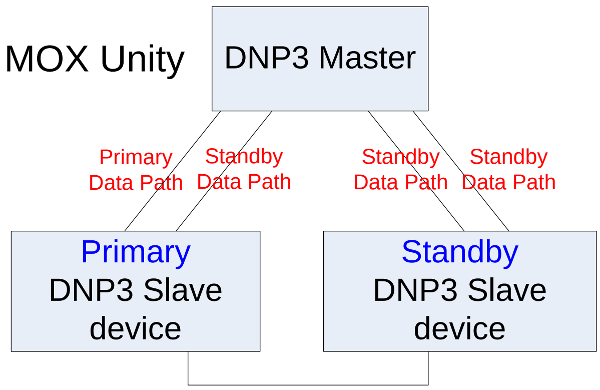



**Figure 17 Data Path Redundancy Example 3**

**Redundancy Rules:**

When MOX Unity detects that Primary Data Path fails, MOX Unity will try next Standby Data Path, if next Standby Data Path also fails, MOX Unity will try other Standby Data Paths, until the last Data Path.

When MOX Unity uses Standby Data Path, it also checks Primary Data Path. Ten minutes later after Primary Data Path is ok, MOX Unity will switch from Standby Data Path to Primary Data Path.

When MOX Unity uses Primary Data Path, it will not check Standby Data Path.

The maximum number of data paths is 5.

The primary data path is the one given the highest number in the list, e.g. 0 would be the least priority standby data path and 4 would be the highest priority primary data path.

**Note:**

1) Data Path parameters can be different.
2) Primary DNP3 Slave device and Standby DNP3 Slave device's DNP3 Slave address can be different.

### *3.3.2 MOX Unity System Redundancy*

The section is to clarify complete MOX Unity redundancy capabilities when the MOX DNP3 master driver is combined with the MOX DNP3 slave driver in MOX Unity. MOX Unity redundancy is shown here:

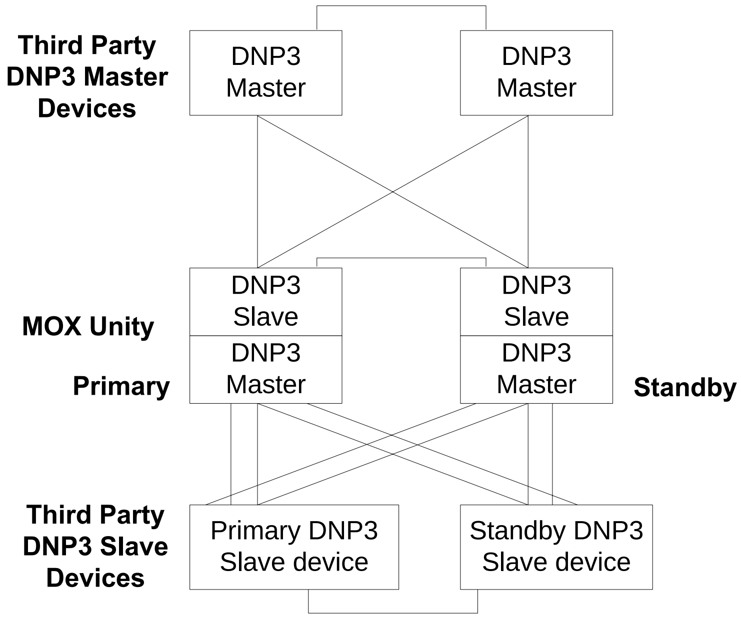



**Figure 18 MOX Unity Redundancy**

1) MOX Unity's DNP3 master driver is not redundant between primary and standby MOX Unitys.
2) Only the primary MOX Unity can communicate with the DNP3 slave devices through MOX Unity's DNP3 master driver. If DNP3 communication fails, the primary MOX Unity will not switch to the standby MOX Unity.
3) All DNP3 Slave events are stored in the primary MOX Unity, and these events will not be synchronized to standby MOX Unity.
4) Third party DNP3 master devices can always communicate with the primary MOX Unity through MOX Unity's DNP3 slave driver. You can set a flag to decide whether third party DNP3 master can communicate with standby MOX Unity. If the flag is set to TRUE, the third party DNP3 master can communicate with standby MOX Unity, but it cannot get any DNP3 slave events. If the flag is set to FALSE, the third party DNP3 master cannot communicate with standby MOX Unity.

# 4 MOX Unity DNP3 Slave Configuration

## 4.1 MoxIDE Settings

### 4.1.1 General Tab

The General tab is where MoxIDE configuration is bound to MoxGRAF. In order to be able to do this bind, a MoxGRAF program will need to exist.

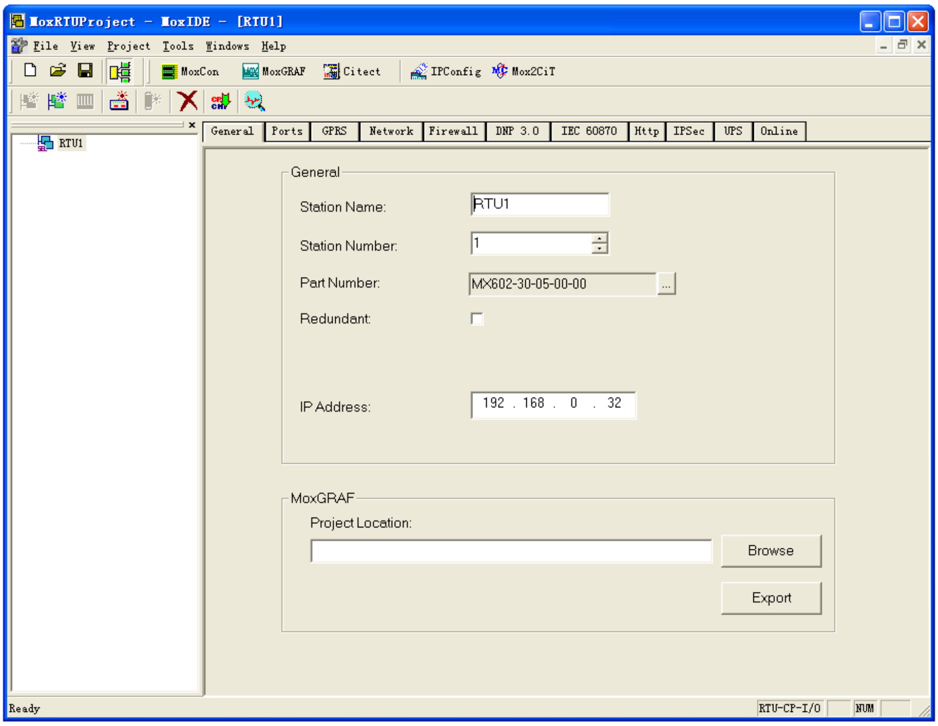



**Figure 19 MoxIDE – General Tab**

The *Station Name* is an arbitrary name given to a device for identification purpose only.

The *Station Number* is an identifier for the MOX Unity for other communication protocols.

The *Part Number* is used to select desired features of MOX Unity.

The *Redundant* option is used to configure the MOX Unity for use in a redundant configuration. This feature is not supported when using the DNP3 protocol, as the DNP3 configuration does not support multiple masters.

The *IP Address* is of the MOX Unity to where the configuration is to be downloaded.

The *MoxGRAF – File Name* is the MoxGRAF project name to link the MoxIDE project to. This field also includes the file path to where the project structure is located.

### 4.1.2 Ports Tab

The Ports tab is where the user is able to configure the communication ports on the MOX Unity.

*COM 0* is used to configure the onboard PSTN data modem option.

*COM 1~4* are the onboard serial communications ports.

Each communication port is configurable to a type of:

- RS232 serial connection
- RS485 serial connection
- Modem connection.

The user can individually configure the baud rate, parity, stop bits, and flow of each communication port.

The MOX Unity also supports many different communication protocols.

For the MOX Unity to use the DNP3 protocol, the "DNP" needs to be selected for each port that will be communicating using the DNP3 protocol.

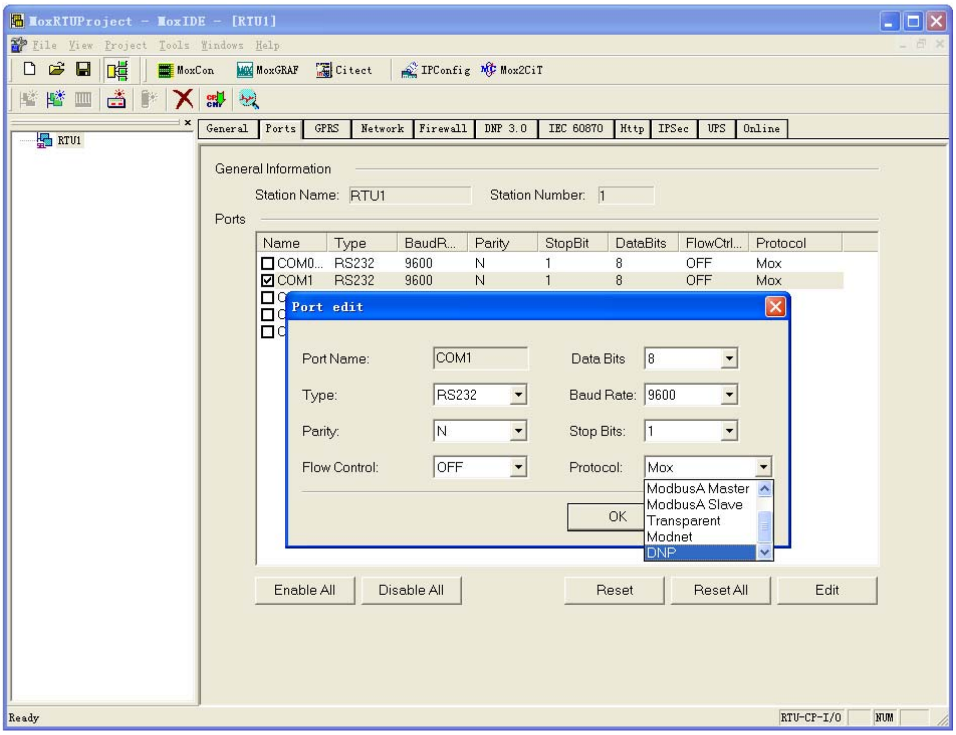



Figure 20 MoxIDE – Ports Tab

### 4.1.3 DNP3 Tab

The DNP3 tab is where the user is able to configure the device specific configuration details for DNP3 protocol support.

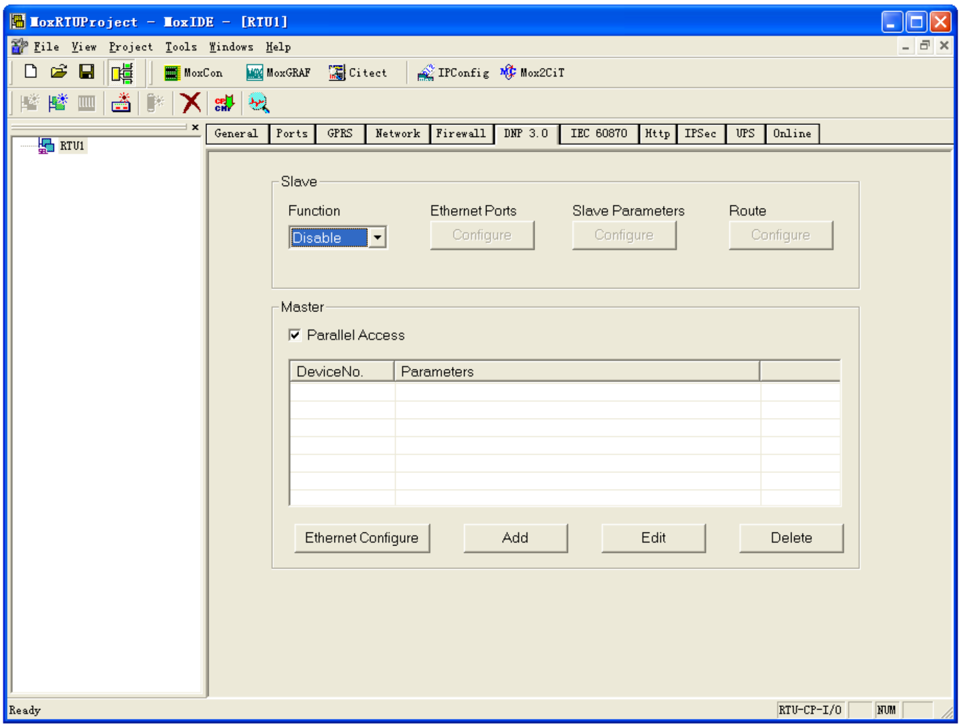



**Figure 21 MoxIDE DNP3 Tab**

From this tab, the user is able to:

- Disable DNP3 protocol support for the product.
- Enable a single DNP3 slave protocol support.
- Enable dual DNP3 slave protocol support for two communication paths (slave addresses must be different).
- Configure Ethernet ports for required DNP communications.
- Configure DNP3 protocol specific slave communication parameters.
- Configure DNP3 protocol specific master communication parameters.
- Configure DNP3 network node specific routing paths.

The MOX Unity currently supports multiple logical entities (i.e. logical controllers) within physical Unity. This is configured in MoxIDE.

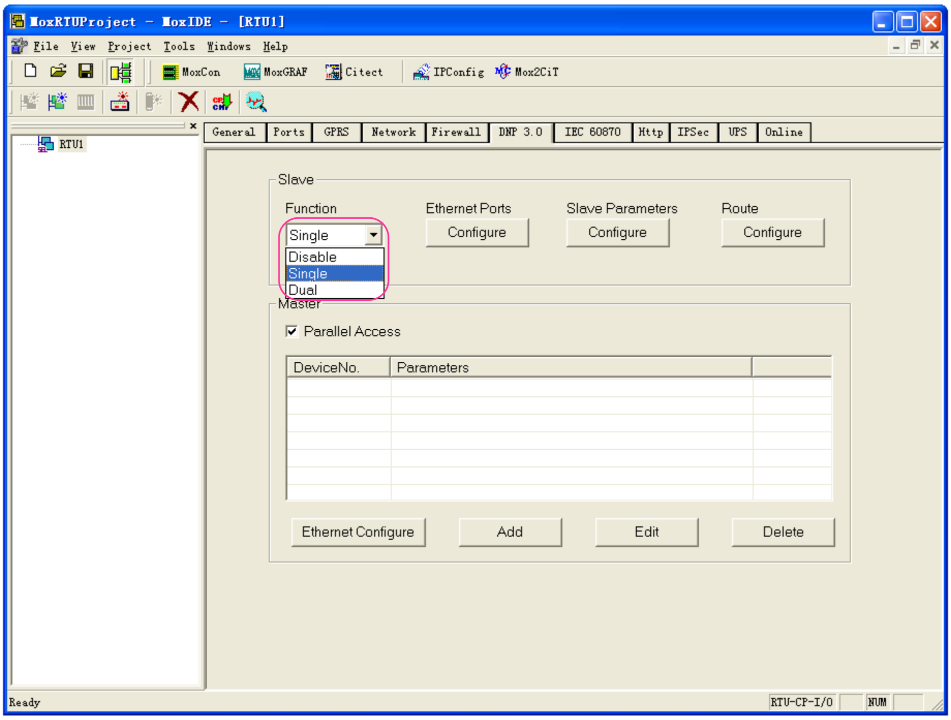
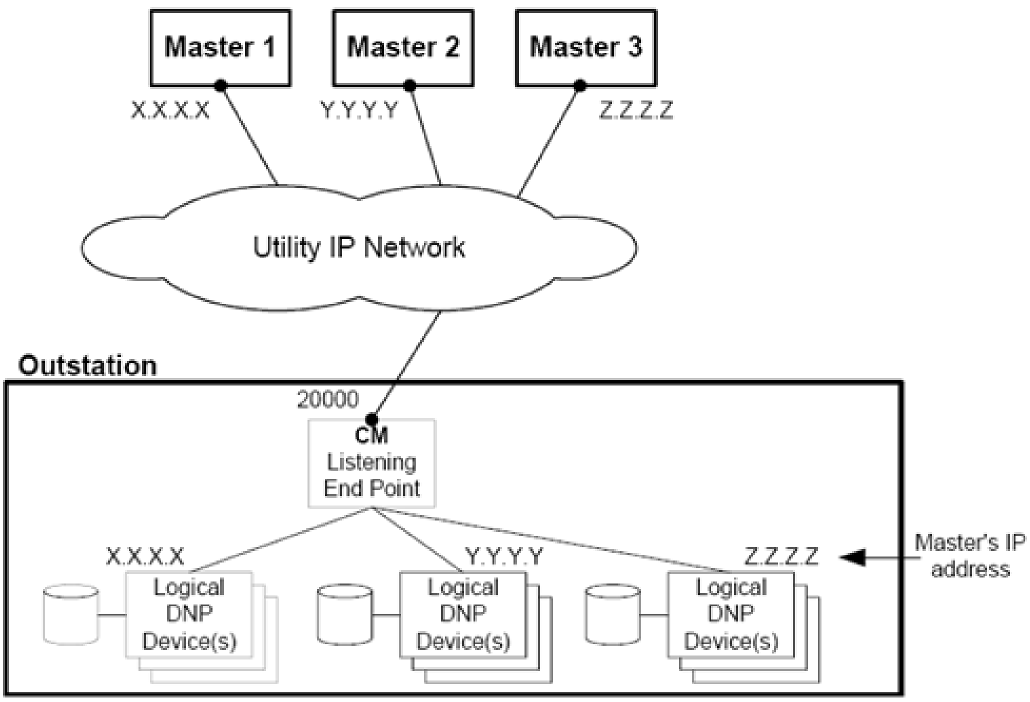



Figure 22 MoxIDE – Enable DNP3 Slave Device



#### 4.1.3.1 DNP Slave Device Start-up Rules

Upon start-up of a Slave device, the Slave device must set true (1) the DEVICE RESTART Internal Indication bit within the Internal Indication field of each subsequent response and unsolicited response. This bit must remain set until a Master device clears it with a WRITE request.

If the Slave device is configured to send unsolicited responses, it must immediately send an unsolicited response upon start-up. This initial unsolicited response must contain either:

- the response header only, containing the Access Control, Function Code and IIN fields.
- the response header plus the current state of ALL of its static data points (e.g. Binary inputs, analog inputs).

The initial unsolicited response may also contain any events objects the Slave device had generated while running previously, but had not yet reported to the Master. If such objects exist, the Slave must report them before the static data. This rule allows the Master to report data changes in chronological order by simply parsing the message from beginning to end (i.e. the most recent value of any point will be the last one reported in the message).

Upon receiving this initial unsolicited response, the Master device must send a WRITE request to clear the DEVICE RESTARTED Internal Indication object reported by the Slave.

#### 4.1.3.2 Ethernet Port Configuration for DNP3 Slave

The DNP3 slave configuration dialog allows the user to configure the Ethernet port DNP communication characteristics.

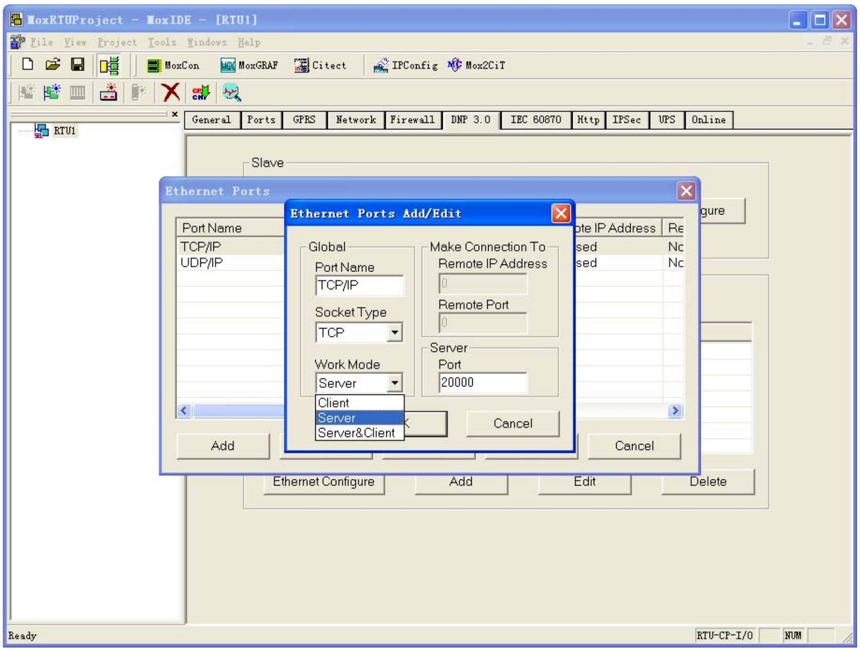



**Figure 23 MoxIDE – DNP3 Slave Ethernet Configuration**

### *TCP Server Mode*

In the TCP server mode, the MOX Unity will listen for a connection request by performing a TCP passive open operation on the Ethernet port.

### *TCP Client Mode*

In the TCP client mode, the MOX Unity initiates the connection by performing a TCP active open on the Ethernet port.

### *TCP Server & Client Mode*

In the TCP server & client mode the MOX Unity can both listen for a connection request and perform an active open on the Ethernet port.

If the master supports TCP server & client mode and both sides simultaneously initiate the connection, the connection initiated by the master will be maintained while the connection initiated by MOX Unity will be dropped.

The MOX Unity ensures that only one connection is in place to the master at any time.

### *UDP Server Mode*

In the UDP server mode, the MOX Unity provides a server port to receive all messages from the master.

### *UDP Client Mode*

In the UDP client mode, the MOX Unity sends all messages to the specified remote master's address and port.

### *UDP Server & Client Mode*

In the UDP server & client mode, the MOX Unity will send the initial unsolicited message to the configured remote port, using the source port number from subsequent requests or will continue sending all responses, solicited and unsolicited, using the configured remote port as the destination.

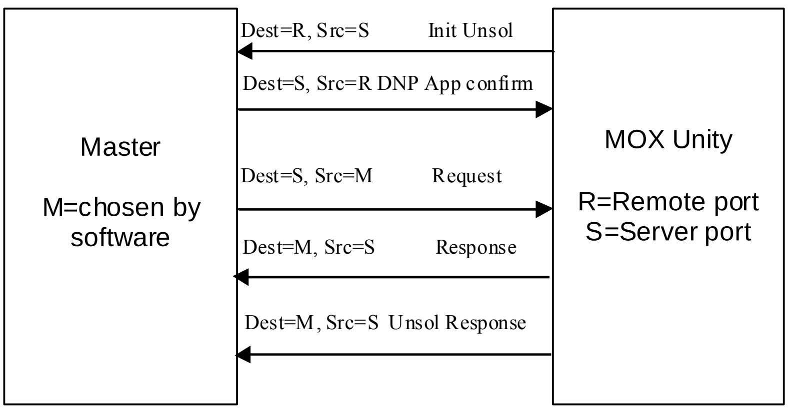



**Figure 24 UDP Server and Client Mode**

Note: *Default TCP port is 20000, default UDP port is 20000.*

### 4.1.3.3 DNP3 Slave Parameter Configuration

The DNP3 Slave configuration dialog allows the user to individually configure the protocol specific communication parameters for each DNP network node.

For each individual DNP3 network node, the user is able to configure the following:

- Master and Slave DNP3 addresses
- Communication media type (Serial or TCP/IP) to be used for the node
- The Data Link layer setup
- The Application layer setup
- Unsolicited data configuration

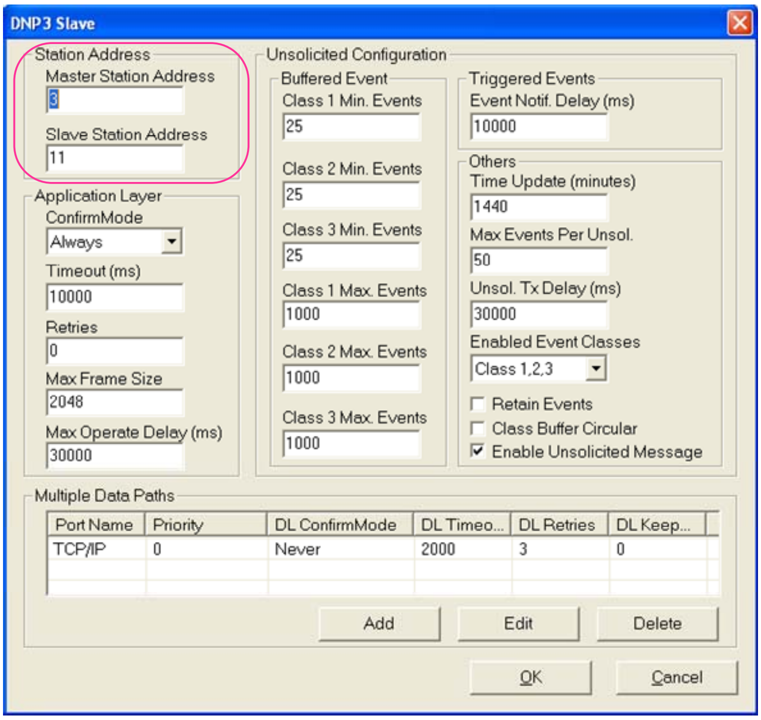



**Figure 25 MoxIDE – DNP3 Slave Station Address Configuration**

***Station Address***

The DNP3 slave device uses these two fields to:

- Uniquely identify itself within a DNP3 network (slave station address)
- Define the DNP3 master that the slave device will attempt to send unsolicited data to (master station address).

The address can be with the range from 0~65534.

*Each node within a network must have a unique address.*

### *Data Paths (Data Link Layer Configuration)*

The lowest communication layer in DNP3 is the physical channel. This channel is called the DNP data-link layer. This layer of communication is configured individually for each port to be used with the DNP3 protocol.

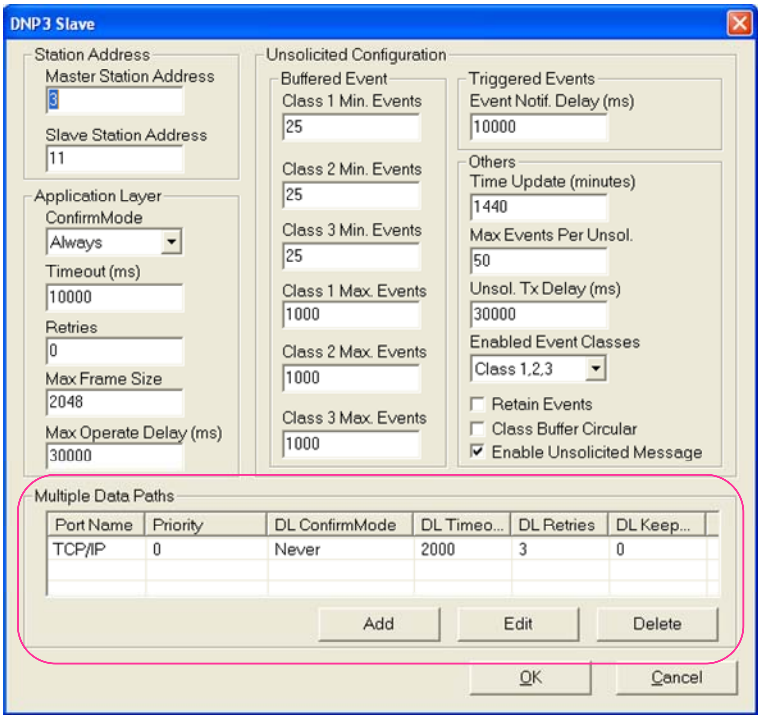



**Figure 26 MoxIDE – DNP3 Slave Data Paths Configuration**

### *Single Data Path*

Select the communication port to be configured for use and then select the *Edit* button. This will display an edit window.

*Port*

Select the port that is to be used for this slave configuration, i.e. TCP, UDP or serial communication port with DNP protocol selected.

*Priority*

If two or more data paths are configured, the highest priority data path is set as primary and the other are set as standby data paths. If two data paths' priorities are identical, the first data path in the list will be set as the primary data path.

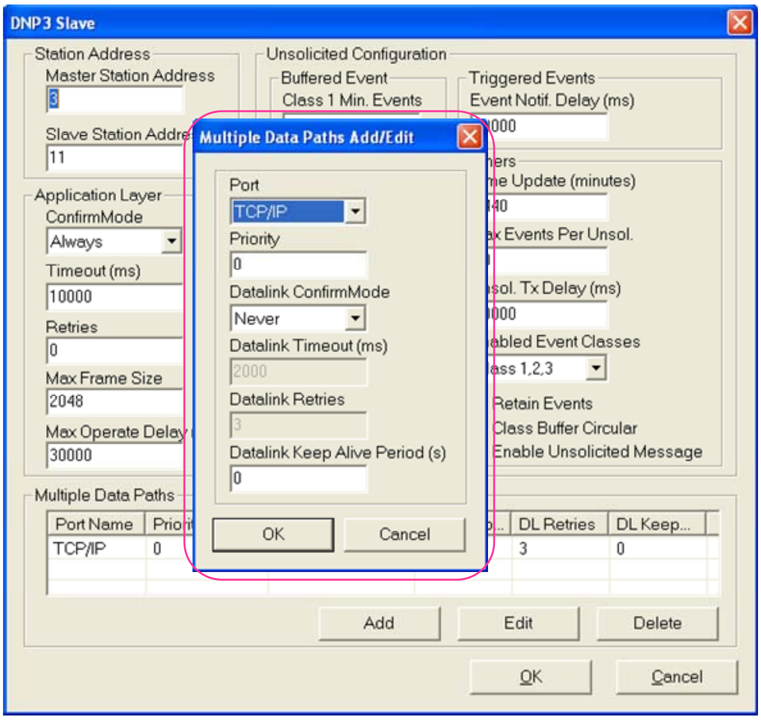



**Figure 27 MoxIDE – Editing DNP3 Slave Data Paths**

*Data Link Layer Confirm Mode*

DNP3 nodes can request data link confirmation.

By using data layer confirmations, a DNP3 device can acknowledge when messages have been successfully delivered to the destination.

A data link layer confirmation, or ACK, indicates only that the data-link frame has been received and that it passes CRC error checks.

The receiving device sends a confirmation message back to the source device when the frame from the sender's message arrives, and the message is in the correct format. Using this technique, message loss, and incorrect message formats can be minimised.

There are three different modes of Data Layer confirmation:

- Never – the sending device never sets the DNP3 "data-link confirm" required flag for a transmitted message. This is the most efficient data transmission method, but should only be used where transmission is on a highly reliable medium, as the receiver never confirms the sent message.
- Always- the sending device always requires a DNP3 data-link confirm from the receiving node. This mode is the least efficient transmission method. This mode is best suited to be used on unreliable links to improve communications.
- Sometimes – this method only requires a data-link confirm for messages where multiple framed DNP3 messages are sent. This method of sending messages is best suited on devices when a range of message frame sizes on potentially unreliable network links. The

method is useful as to be able to send individual messages rather than entire multi-framed messages again.

*Data Link Layer Confirm Timeout (msec)*

When using data link confirmations on a sent message (i.e. when using Always or Sometimes), this is the maximum period (ms) to wait for receiving the requested confirmation from the destination device.

*Data Link Layer Retries*

If the data sender requests a data-link confirmation response, and the destination does not return this response back to the sender (all within the Data Link Layer Confirm timeout period), the data-link message is resent.

The number of times that the message is resent is the number of retries set by this parameter. If the number of retries is exceeded, then the message is undeliverable, and the sender aborts the transmission of the DNP3 message. This will result in a communication failure for that message.

*Data Link Layer Keep Alive Period (s)*

Send Request Link Status, whenever the line is idle. A value of ‘0’ disables the keep-alive check parameter.

***Multiple Data Paths***

By default a single Ethernet port data path will be displayed. To add further data paths select the “*Add*” button and configure the path as required. When adding multiple data paths the following information should be noted:

- Maximum data paths is 5 for every logical DNP3 slave
- Multiple data paths are used for data path redundancy

*Application Layer Setup*

The highest communication layer in DNP3 is the Application Layer. This layer of communication is configured for all ports that use DNP3 protocol, but are used to control how the messages are generated by the controller. Messages at this layer are broken into Application Layer Fragments.

The following parameters are configurable:

- Confirmation mode
- Layer timeout
- Layer retries
- Maximum frame size
- Maximum operate delay

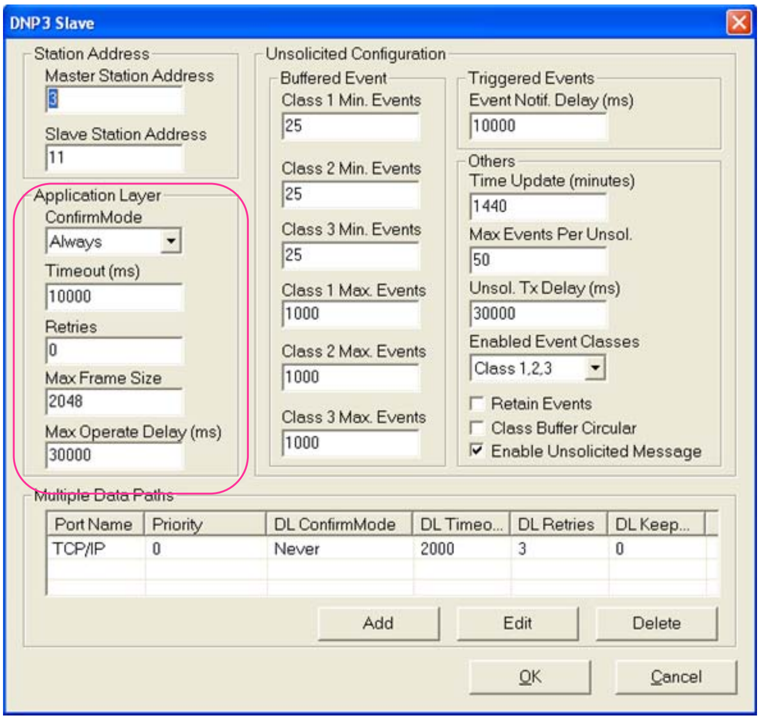



**Figure 28 MoxIDE – DNP3 Slave Application Layer Configuration**

*Application Layer Confirm Mode*

Similar to the data-link layer, application layer messages may be sent to a destination with a request for a confirmation. The confirmation from the destination indicates that a message has not only been received, but also been parsed without error.

There are three different modes of Application Layer confirmation:

- Never – the sending device never sets the DNP3 "application layer confirm" required flag for a transmitted message. The receiver never confirms the sent message content.
- Always- the sending device always requires a DNP3 application layer confirm from the receiving node. This is used to indicate to the sender that the receiver is able to parse the sent message
- Sometimes – this method only requires an application layer confirm for messages where multiple framed DNP3 messages or event data messages are sent. The method is useful as to be able to send individual frames rather than entire multi-framed messages again.

*Application Layer Confirm Timeout (msec)*

When using application layer confirmations on sent messages (i.e. when using Always or Sometimes), this is the maximum period (ms) to wait for receiving the requested confirmation from the receiver's application layer. This application layer confirmation is to confirm that the destination device has understood the message.

*Application Layer Retries*

If the data sender requests an application layer confirmation response, and the destination has does not return this response back to the sender, all within the Application Layer Confirm timeout period, the application layer message is resent. The number of times that the message is resent is the number of retries configured by this parameter. If the number of retries is exceeded, then the message is undeliverable, and the sender aborts the transmission of the message. This will result in a communication failure for that message.

*Application Layer Maximum Frame Size*

This parameter is the maximum number of octets that Application Layer can have. This value is configurable between 400~2048 octets.

*Maximum Operate Delay (ms)*

When using two-phase Select/Operate DNP3 commands, an Operate command must be received from the same master device within this period after the Select command is received. If a DNP3 Operate command for the control is not received within this period then the control operation is discarded. If an Operate command is received within this period, then the Operate command is performed on the previously selected control.

***Unsolicited Configuration***

Unsolicited information is information that is transmitted, but has not been requested.

To control how the unsolicited information is transmitted, the user is able to configure parameters that delay data transmission and group/buffer events into single data transmissions.

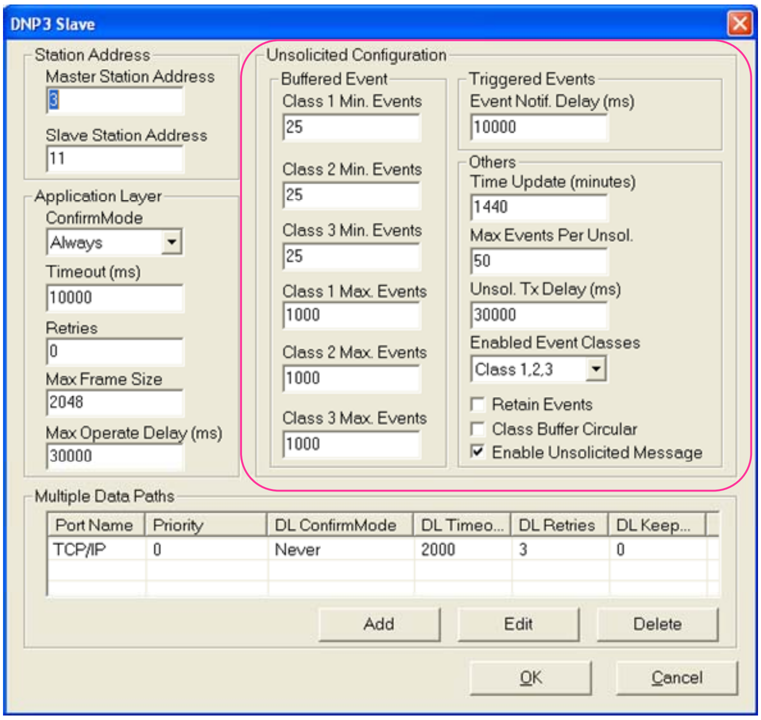



**Figure 29 MoxIDE – DNP3 Slave Unsolicited Configuration**

*Buffered Events*

Class 1, 2, 3 Minimum Events: These parameters specify the minimum number of events of a particular Class that are required to generate an unsolicited data transmission. When an unsolicited data transmission is generated, all buffered events of the enabled Classes are sent. Each Class is individually configurable.

*Triggered Events*

Event Notification Delay: This parameter sets the delay after an event occurs before the Event Object is transmitted. This delay allows the first event, and any other events that may occur in this delay period, to be sent in a single DNP Unsolicited data transmission. This parameter can be used to reduce network bandwidth requirements.

*Others*

Enable Unsolicited Message: This parameter enables or disable the generation of unsolicited data transmissions.

Unsolicited Transmission Delay: This parameter sets the minimum time between consecutive unsolicited data transmissions sent from the controller. After an unsolicited data transmission has been sent by the controller, no unsolicited data transmissions will be generated until this time period has elapsed.

Enabled Event Classes: This parameter specifies which Event Classes will be enabled to generate unsolicited data transmissions.

### 4.1.3.4 *DNP3 Slave Node Routing Table*

MOX controllers are capable of routing DNP3 protocol packets between different sections of a communication network. This is achieved by using a routing table. Each controller in the communication network can be configured to use a routing table.

Each DNP3 data-link layer frame contains both a source and destination DNP node address. This addressing method allows for peer-to-peer controller communication, and allows DNP3 data-link layer frames to be routed.

The routing table is shown below:

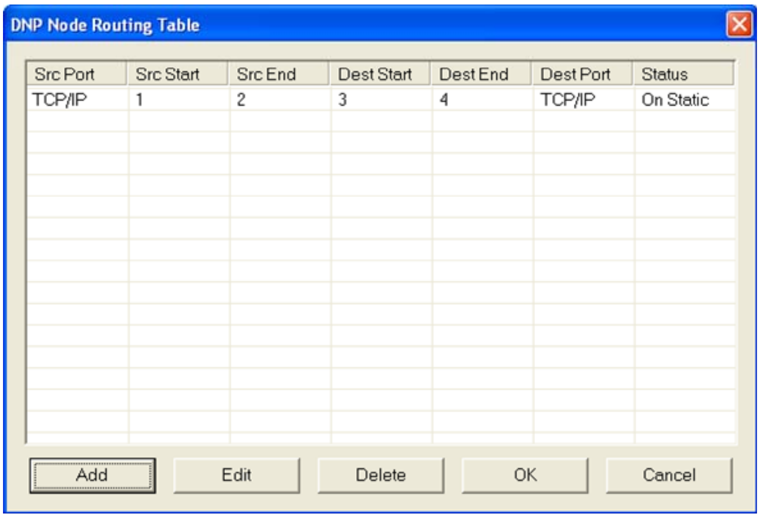
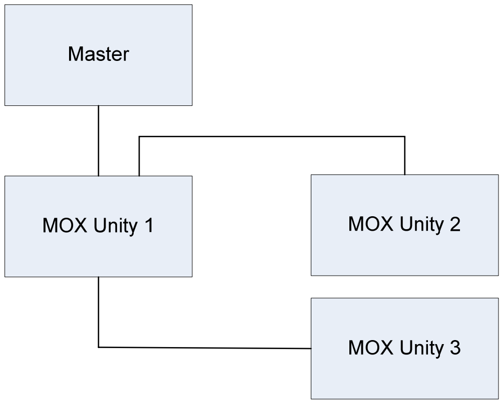



**Figure 30 MoxIDE – DNP3 Node Routing Table**




***Routing - Definition***

Routing is the process of moving a packet of data from source to destination. Routing enables messages to pass from one device, through another, and eventually reach the target device. Each intermediary device performs routing by passing along the message to the next device.

***Routing Table***

A Routing Table is used to determine if data packet are to be processed or to be forwarded to the next controller by the paths configured.

Each entry in the route table for a controller describes one scenario for routing data frames received at this controller.

***Routing Table Rules***

When a controller receives a data packet, the following rules are used to determine if the data-link layer packet is to be processed locally, or to be routed.

1) If the destination address is the address of the controller (or 65535), the frame is passed to the Transport Layer for processing (local).
2) Determine if the destination address appears in the DNP Node Routing Table. If so, the frame is passed back to the data-link layer for re-transmission according to the corresponding route-table entry.
3) If neither of these actions can be performed, the frame is discarded.

***Routing Table Entry Fields***

Each entry in the route table describes one scenario for routing data frames received at this controller.

The routing table contains the following fields:

- Src Port: This field refers to the port on which the inbound DNP frame arrived at.
- Src Start – Src End: These fields specify the range of source DNP address that this route table entry refers to.
- Dest Start – Dest End: These fields specify the range of destination DNP address that this route table entry refers to.

- Dest Port: This field specifies which port the DNP frame will be re-transmitted (forward) on.
- Status: This field indicates the status of this route table entry and maybe one of the following values:
    - On Static -- User entered route
    - Off Static -- User entered route (Disabled)

***Adding, Deleting, Editing Table Entires***

*Adding Entries*

To add an entry to the route table, simply click the “Add” button located on the DNP Node Routing Table. This action will display the entry box where the details of the route can be entered.

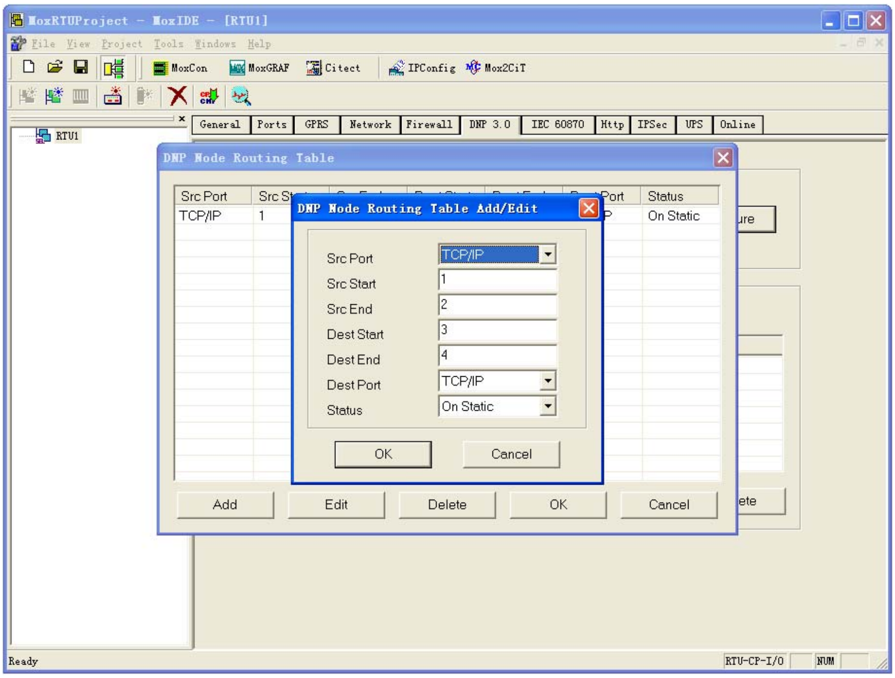



**Figure 31 MoxIDE – Node Routing Table Configuration Dialog**

*Deleting Entries*

To delete an entry in the route table, simply select the route table entry, and then click the “Delete” button located on the DNP Node Routing Table. This action will remove the selected route from the table.

*Editing Entries*

To edit an entry in the route table, simply select the route table entry, and then click the “Edit” button located on the DNP Node Routing Table. This action will display the entry box where the details of the route can be viewed and changed.

### *4.1.4 Online Tab*

Please refer to 3.1.4 Online Tab.

## 4.2 MoxGRAF Settings

To configure the MOX Unity with DNP3 slave function in MoxGRAF, it is required to complete following steps:

- Assign MoxGRAF internal variables to DNP3 index
- Assign data Classes to the individual DNP3 index
- Configure individual DNP3 address object and variation specifics and Event characteristics
- Build the project and download to the MOX Unity.

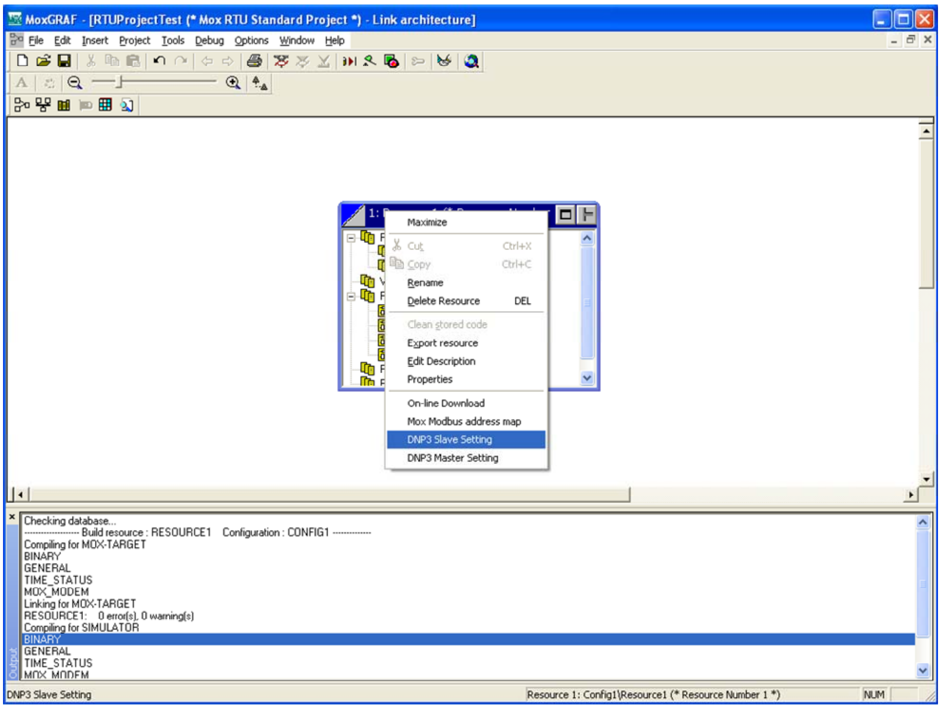



**Figure 32 MoxGRAF – DNP3 Slave Setting**

Please refer to *MoxGRAF User Guide* for variable creation and programming instructions.

### *4.2.1 Data Class Assignment*

When an *Unmapped Variable* is mapped to a Boolean or Analog Input DNP3 Index, the default Class assignment is Class 1.

If Boolean or Analog Outputs are assigned to Class 1, 2 or 3 and the Master changes any of these variables, the MOX Unity will save the variables' static objects to the corresponding class buffer for further unsolicited response.

Once the mapping of the variable has been completed, only then can the Class assignment be modified.

To modify the class assignment:

- Select the data type to be viewed from the pull down list
- Highlight the index for which the class assignment is to be changed
- Once highlighted, click on the “Edit” button to display the index specific settings.

For all data types, Class 0, 1, 2, 3 assignments are available.

*Note: Class 0 is reserved for static data objects (static data indicates the current value of the object in the slave device). Classes 1, 2 and 3 are used for event data objects (objects created as the result of data changes in the slave device).*

An event object can be assigned to either Class 1, 2 or 3.

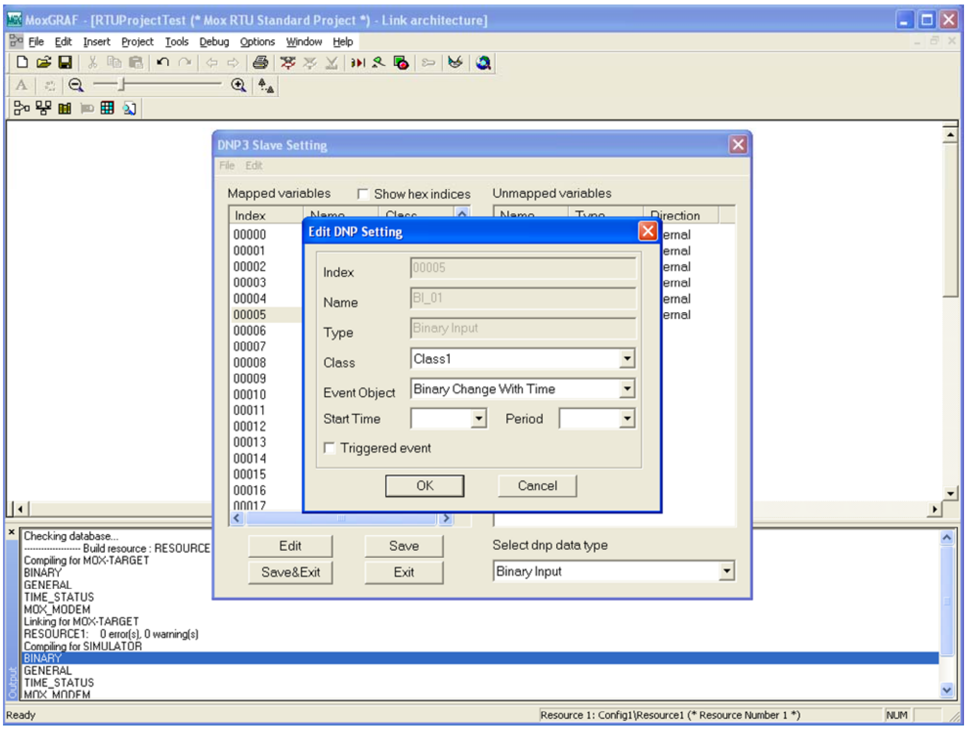



Figure 33 MoxGRAF – Index Specific Settings

### 4.2.2 Event Object Assignment

Assigning an index to Class 0 disables the Event Object configuration option for that index.

Events in DNP3 are typically associated with data changing within a slave device. Some examples of these changes are:

- Digital state changes
- Analog values exceeding some threshold

DNP3 provides the ability to report event objects with or without time.

The MOX implementation of DNP3 supports 4 different data types:

- Binary Input (BI)
- Binary Output (BO)
- Analog Input (AI)
- Analog Output (AO).

Only Input data types can be configured to represent the data in different formats to the Master device.

***Binary Inputs (BI)***

Binary Inputs objects can be configured as either:

- Binary input change of state without time and with flag
- Binary input change of state with time and with flag

***Analog Input (AI)***

Analog Inputs objects can be configured as either:

- Analog change of event without time and with flag
- Analog change of event with time and with flag

***Analog Input Deviation Configuration***

If the absolute value of the difference between the last-reported value of an analog input object and the current value of that object exceeds the deviation value, then an event is generated for that object with the current value sent as the value for the event.

*To disable this functionality enter '-1' into the deviation field.*

### 4.2.3 *Input Event Configuration*

***Unsolicited***

Unsolicited information is information that is transmitted, but has not been requested.

All configured input indexes can be selected for unsolicited communications. The configuration of the unsolicited transmissions is discussed in 1.4.2.

***Triggered***

For an input index to be a triggered event the *Triggered Event* checkbox is required to be checked. When an input is configured as a triggered event, the variable will be sent to the Master immediately (even if the class buffer is not full).

Each input index is individually configurable.

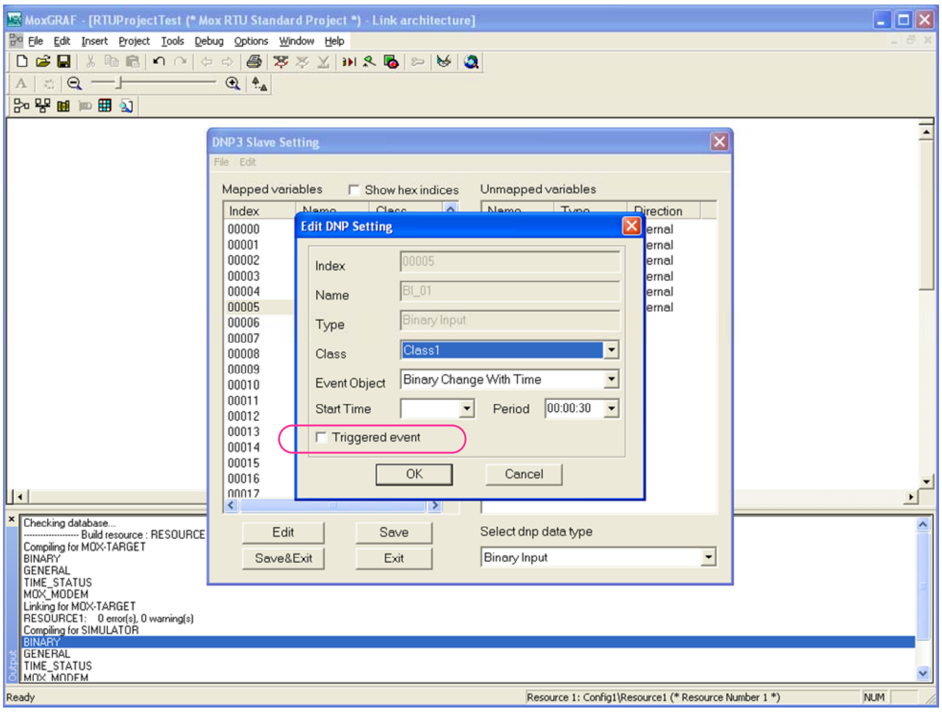



**Figure 34 MoxGRAF – Unsolicited Input Selction**

### 4.2.4 *Periodic Event Generation*

As an additional feature, events can be generated dependent on a configured time period. This is achieved via the *Start Time* and *Period* fields. Leaving both of these fields empty disables this functionality.

By entering a time value in the *Start Time* field, the MOX Unity will commence creating events at this time. If this field is left blank and the *Period* field is not empty, the MOX Unity will commence creating events on start-up.

The *Period* field is used to assign the time duration in between the periodic generation of events.

### 4.2.5 *DNP3 Slave Point Controls*

In MoxGRAF dictionary, there should be two variables that are mapped to one DNP3 Slave point.

One is used to write value to the control point, while the other is used to read back.

#### 4.2.5.1 *Analog Output Block Controls*

Once MoxGRAF detects the value read is different to the write value, the DNP3 Slave sends ‘analog output block controls’ to the DNP3 Master device.

#### 4.2.5.2 *Binary Controls – CROB*

Once MoxGRAF detects the value read is different to the write value, the DNP3 Slave sends ‘control relay output block’ to the DNP3 Master device.

### 4.2.6 *Build and Download a MoxGRAF Project*

After all necessary components are completed; the MoxGRAF project can be built and downloaded to the target controller. In the MoxGRAF toolbar, click Build Project/Library button to build the configuration file. The configuration file then needs to be downloaded to the controller.

Please refer to the MoxGRAF User Guide for specific details of configuring and downloading project configurations.

## 4.3 MOX DNP3 Slave Redundancy

MOX Unity configured as DNP3 slave supports redundancy in two levels: controller redundancy and data path redundancy.

### *4.3.1 Controller Redundancy*

MOX controller redundancy automatically backs up data between a primary slave and a standby slave.

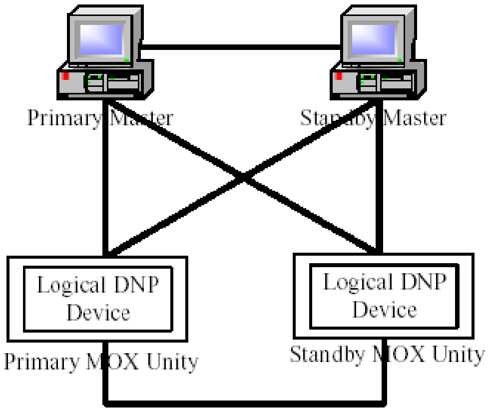



**Figure 35 Controller Redundancy**

### *4.3.2 Data Path Redundancy*

Each logical DNP3 slave supports data path redundancy.

The primary and standby data paths are determined by the data path priority parameter, configured via MoxIDE.

The highest priority data path is considered the primary data path. All other paths are standby data paths. If two data paths' priorities are identical, then the first data path will be set as the primary data path.

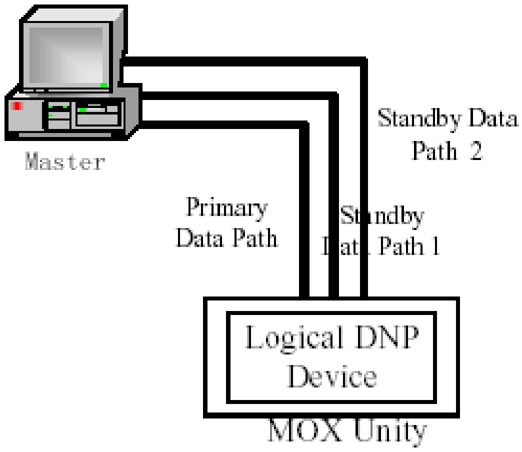



**Figure 36 DNP Data Path Redundancy (Single Master)**

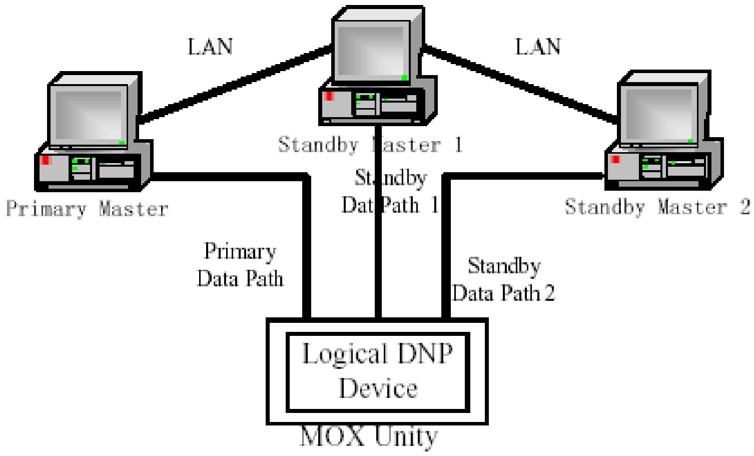



**Figure 37 DNP Path Redundancy (Multiple Master)**

**Data Path Redundancy Rules:**

When multiple data paths are configured, i.e. primary, standby 1, standby 2 data paths, in one logical DNP3 slave, the controller must operate using a set of rules for efficient communications.

1) When failure of the primary data path is detected, the controller will first try standby 1, and then standby 2 until all data paths are exhausted.
2) When the controller is using standby data path 1 or 2, it will also check the primary data path. Ten minutes after the primary data path has been detected as ok, the controller will switch from the standby data path to the primary data path.
3) When the controller uses the primary data path, it will not check the standby data paths.
4) When the controller receives data from non-active data path, the data path will be set to active. For example, the slaved controller communicates with the master via the primary data path. The controller then receives a *poll static objects* command via standby data path 2. Standby data path 2 is now set active and the controller considers primary data path to have failed.
5) The primary data path is the one given the highest value in the list, e.g. 0 would be the least priority standby data path and 4 would be the highest priority primary data path.

# 5 DNP3 Transfer Events

If one variable in MoxGRAF dictionary is mapped to both MOX Unity DNP3 Slave and Master objects, the events of the variable received by MOX Unity DNP3 Master driver will be saved to MOX Unity DNP3 Slave driver event queue.

**For instance:**

1) MoxGRAF dictionary -> Variable name ABC, type REAL.
2) DNP3 Slave driver -> ABC is mapped to Short Floating Point, Index 00001.
3) DNP3 Master driver -> ABC is mapped to Short Floating Point, Index 00005.
4) DNP3 Master driver receives an event of ABC, object 32 variation 7 index 00005 (Short Floating Point Analog Input Change Event with Time).
5) DNP3 Master driver changes the index 00005 to 00001, saves the event (object 32 variation 7 index 00001) to DNP3 Slave driver event queue.

*Note: If the DNP3 Master driver receives the events that are not mapped in DNP3 Slave driver, these events will be discarded.*

# Appendix A Device Profile (Slave) - MOX Unity

<table>
<tr><td colspan="2">DNP V3.00<br>DEVICE PROFILE DOCUMENT</td></tr>
<tr><td colspan="2">Vendor Name: MOX Group</td></tr>
<tr><td colspan="2">Device Name: MOX Unity</td></tr>
<tr><td>Highest DNP Level Supported:<br><br>For Requests Level2<br><br>For Responses Level2</td><td>Device Function:<br><br>☐ Master<br><br>√ Slave</td></tr>
<tr><td colspan="2">Notable objects, functions, and/or qualifiers supported in addition to the Highest DNP Levels Supported (the complete list is described in the attached table):<br><br>Obj 10, Var 02, Func code 130, Qualifiers 17 & 28<br>Obj 22, Var 05, Func code 129 & 130, Qualifiers 17 & 28<br>Obj 22, Var 06, Func code 129 & 130, Qualifiers 17 & 28<br>Obj 30, Var 05, Func code 129, Qualifiers 00 & 01<br>Obj 32, Var 03, Func code 129 & 130, Qualifiers 17 & 28<br>Obj 32, Var 04, Func code 129 & 130, Qualifiers 17 & 28<br>Obj 32, Var 05, Func code 129 & 130, Qualifiers 17 & 28<br>Obj 32, Var 07, Func code 129 & 130, Qualifiers 17 & 28<br>Obj 40, Var 01, Func code 129, Qualifiers 00 & 01; Func code 130, Qualifiers 17 & 28<br>Obj 40, Var 02, Func code 130, Qualifiers 17 & 28<br>Obj 40, Var 03, Func code 129, Qualifiers 00 & 01; Fun code 130, Qualifiers 17 & 28<br>Obj 41, Var 01, Request Func code 3,4,5,6, Qualifiers 17, 28, Response Func code 129<br>Obj 41, Var 03, Request Func code 3,4,5,6, Qualifiers 17, 28, Response Func code 129<br>Func code 14 (Warm Restart)</td></tr>
<tr><td>Maximum Data Link Frame Size (octets):<br><br>Transmitted ____292_______<br><br>Received (must be 292)</td><td>Maximum Application Fragment Size (octets):<br><br>Transmitted Configurable between 400 and 2048<br><br>Received 2048</td></tr>
</table>

| | |
|---|---|
| Maximum Data Link Re-tries:<br><br>☐ None<br>☐Fixed at ____________________<br>√Configurable, range 0 to 65535 | Maximum Application Layer Re-tries:<br><br>☐ None<br>√ Configurable, range _0_ to _65535_<br>(Fixed is not permitted) |
| Requires Data Link Layer Confirmation:<br><br>☐Never<br>☐ Always<br>☐ Sometimes<br>√ Configurable | |
| Requires Application Layer Confirmation:<br><br>☐ Never<br>☐ Always (not recommended)<br>☐ When reporting Event Data (Slave devices only)<br>☐ When sending multi-fragment responses (Slave devices only)<br>☐ Sometimes<br>√ Configurable | |
| Timeouts while waiting for:<br><br>Data Link Confirm ☐ None ☐ Fixed at ____ ☐ Variable √ Configurable<br>Complete Appl. Fragment √ None ☐ Fixed at ____ ☐ Variable ☐ Configurable<br>Application Confirm ☐ None ☐ Fixed at ____ ☐ Variable √ Configurable<br>Complete Appl. Response √ None ☐ Fixed at ____ ☐ Variable ☐ Configurable<br><br>Others ___________<br>Attach explanation if ' Variable ' or 'Configurable' was checked for any timeout. See Note* 8 | |
| Sends/Executes Control Operations: (see Note* 4)<br><br>WRITE Binary Outputs √ Never ☐ Always ☐ Sometimes ☐ Configurable<br>SELECT/OPERATE ☐ Never √ Always ☐ Sometimes ☐ Configurable<br>DIRECT OPERATE ☐ Never √ Always ☐ Sometimes☐ Configurable<br>DIRECT OPERATE - NO ACK ☐ Never √ Always ☐ Sometimes ☐ Configurable<br><br>Count > 1 √ Never ☐ Always ☐ Sometimes ☐ Configurable<br>Pulse On √ Never ☐ Always ☐ Sometimes ☐ Configurable<br>Pulse Off √ Never ☐ Always ☐ Sometimes ☐ Configurable<br>Latch On ☐ Never √ Always ☐ Sometimes ☐ Configurable<br>Latch Off ☐ Never √ Always ☐ Sometimes ☐ Configurable<br><br>Queue √ Never ☐ Always ☐ Sometimes ☐ Configurable<br>Clear Queue √ Never ☐ Always ☐ Sometimes ☐ Configurable<br><br>Attach explanation if 'Sometimes' or 'Configurable' was checked for any operation. | |

| FILL OUT THE FOLLOWING ITEM FOR MASTER DEVICES ONLY: | |
|---|---|
| Expects Binary Input Change Events:<br><br>☐ Either time-tagged or non-time-tagged for a single event<br>☐ Both time-tagged and non-time-tagged for a single event<br>☐ Configurable (attach explanation) | |
| **FILL OUT THE FOLLOWING ITEMS FOR SLAVE DEVICES ONLY:** | |
| Reports Binary Input Change Events when no specific variation requested:<br><br>☐ Never<br>☐ Only time-tagged<br>☐ Only non-time-tagged<br>√ Configurable to send both, one or the other (attach explanation) see Note* 1 | Reports time-tagged Binary Input Change Events when no specific variation requested:<br><br>☐ Never<br>☐ Binary Input Change With Time<br>☐ Binary Input Change With Relative Time<br>√ Configurable (attach explanation) see Note* 1 |
| Sends Unsolicited Responses:<br><br>☐ Never<br>☐ Configurable (attach explanation)<br>√ Only certain objects<br>☐ Sometimes (attach explanation)<br><br>☐ENABLE/DISABLE UNSOLICITED Function codes supported | Sends Static Data in Unsolicited Responses:<br><br>☐ Never<br>☐ When Device Restarts<br>☐ When Status Flags Change<br><br>No other options are permitted.<br>***Refer to Note1*** |
| Default Counter Object/Variation:<br><br>☐ No Counters Reported<br>√ Configurable (attach explanation)<br>☐ Default Object ______________<br>Default Variation ______________<br>☐ Point-by-point list attached<br>See Note* 2 | Counters Roll Over at:<br><br>√ No Counters Reported<br>☐ Configurable (attach explanation)<br>☐ 16 Bits<br>☐ 32 Bits<br>☐ Other Value _____________<br>☐ Point-by-point list attached |
| Sends Multi-Fragment Responses: √ Yes ☐ No | |

Note* 1
In response to an event class poll request or Obj 2 Var 0 read request, the MOX Unity will send binary events as set by the MOX Unity Binary Event Object configuration parameters (MoxGRAF DNP3 Slave Setting, refer to the following picture). Now the MOX Unity Binary Event Object configuration can be set for

- Binary Change No Time(Obj2 Var2)
- Binary Change with time(Obj2 Var1)

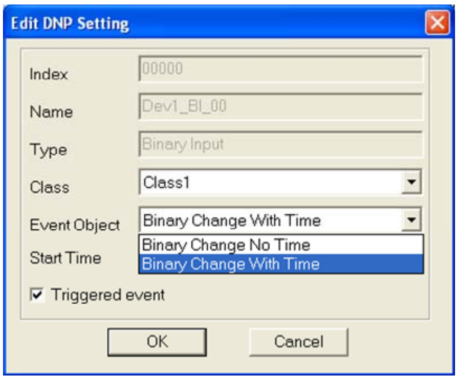



**Figure 38 MoxGRAF – DNP3 Slave Event Object Configuration**

Note* 2
MOX Unity support 32-Bit Binary Counter, 16-Bit Binary Counter, configure in MoxGRAF (DNP3 Slave Setting)

Note* 3
Self-address is not supported in MOX Unity

Note* 4
Maximum number of CROB (object 12, variation 1) objects supported in a single message the number of installed control points.
Maximum number of analog output (object 41, any variation) objects supported in a single message the number of installed control points.
. Pattern Control Block and Pattern Mask (Object 12, variations 2 and 3, respectively) supported. If so, describe any restrictions.
. CROB (object 12) and analog output (object 41) permitted together in a single message.

Note* 5
Time base drift over a 10-minute interval, 1 millisecond
Maximum delay measurement error, infinity
Maximum internal time reference error when set from protocol, 1 millisecond
Maximum response time, 60 seconds.

Note* 6
MOX Unity does not support multi-drop physical layer.

Note* 7
MOX Unity does not implement collision avoidance.

Note* 8
Configurable timeouts can be set via MoxIDE software.

# Appendix B Interoperability Table (Slave) - MOX Unity

| Object | | | REQUEST (slave must pares) | | RESPONSE (master must parse) | |
|---|---|---|---|---|---|---|
| Obj | Var | Description | Func Codes (dec) | Qual Codes (hex) | Func Code | Qual Codes (hex) |
| 1 | 0 | Binary Input - All Variations | 1 | 06 | | |
| 1 | 2 | Binary Input with status | | | 129 | 00,01 |
| | | | | | | |
| 2 | 0 | Binary Input Change – All Variations | 1 | 06,07,08 | | |
| 2 | 1 | Binary Input Change without time | 1 | 06,07,08 | 129, 130 | 17,28 |
| 2 | 2 | Binary Input Change whit Time | 1 | 06,07,08 | 129, 130 | 17,28 |
| 2 | 3 | Binary Input Change with Relative Time | 1 | 06,07,08 | | |
| | | | | | | |
| 10 | 0 | Binary Output – All Variations | 1 | 06 | | |
| 10 | 2 | Binary Output Status | | | 129 | 00,01 |
| 10 | 2 | Binary Output Status | | | *130 | 17,28 |
| | | | | | | |
| 12 | 1 | Control Relay Output Block | 3,4,5,6 | 17,28 | 129 | Echo of request |
| | | | | | | |
| 20 | 0 | Binary Counter – All Variations | 1,7,8,9,10 | 06 | | |
| 20 | 1 | 32-Bit Binary Counter | | | 129,130 | 00,01 |
| 20 | 2 | 16-Bit Binary Counter | | | 129,130 | 00,01 |
| | | | | | | |
| 21 | 0 | Frozen Counter – All Variations | 1 | 06 | | |
| 21 | 1 | 32-Bit Frozen Counter | | | 129,130 | 00,01 |
| 21 | 2 | 16-Bit Frozen Counter | | | 129,130 | 00,01 |
| | | | | | | |
| 22 | 0 | Counter Change Event – All Variations | 1 | 06,07,08 | | |
| 22 | 1 | 32-Bit Counter Change Event without Time | | | 129,130 | 17,28 |
| 22 | 2 | 16-Bit Counter Change Event without Time | | | 129,130 | 17,28 |
| 22 | 5 | 32-Bit Counter Change Event with Time | | | 129,130 | 17,28 |
| 22 | 6 | 16-Bit Counter Change Event with Time | | | 192.130 | 17,28 |
| | | | | | | |
| 30 | 0 | Analog Input – All Variations | 1 | 06 | | |
| 30 | 1 | 32-Bit Analog Input | | | 129 | 00,01 |
| 30 | 2 | 16-Bit Analog Input | | | 129 | 00,01 |
| 30 | 5 | Short Floating Point Analog Input | | | 129 | 00,01 |
| | | | | | | |
| 32 | 0 | Analog Change Event – All Variations | 1 | 06,07,08 | | |
| 32 | 1 | 32-Bit Analog Input Change Event without Time | | | 129,130 | 17,28 |
| 32 | 2 | 16-Bit Analog Input Change Event without Time | | | 129,130 | 17,28 |
| 32 | 3 | 32-Bit Analog Input Change Event with Time | | | 129,130 | 17,28 |
| 32 | 4 | 16-Bit Analog Input Change Event with Time | | | 192,130 | 17,28 |
| 32 | 5 | Short Floating Point Analog Input Change Event without Time | | | 129,130 | 17,28 |
| 32 | 7 | Short Floating Point Analog Input Change Event with Time | | | 129,130 | 17,28 |
| | | | | | | |
| 40 | 0 | Analog Output – All Variations | 1 | 06 | | |
| 40 | 1 | 32-Bit Analog Output Status | | | 129 | 00,01 |
| 40 | 1 | 32-Bit Analog Output Status | | | *130 | 17,28 |
| 40 | 2 | 16-Bit Analog Output Status | | | 129 | 00,01 |
| 40 | 2 | 16-Bit Analog Output Status | | | *130 | 17,28 |

| | | | | | | |
|---|---|---|---|---|---|---|
| 40 | 3 | Short Floating Point Analog Output Status | | | 129 | 00,01 |
| 40 | 3 | Short Floating Point Analog Output Status | | | *130 | 17,28 |
| | | | | | | |
| 41 | 1 | 32-Bit Analog Output Block | 3,4,5,6 | 17,28 | 129 | Echo of request |
| 41 | 2 | 16-Bit Analog Output Block | 3,4,5,6 | 17,28 | 129 | Echo of request |
| 41 | 3 | Short Floating Point Analog Output Block | 3,4,5,6 | 17,28 | 129 | Echo of request |
| | | | | | | |
| 50 | 1 | Time and Data | 2 | 07 where quantity =1 | 129 | 07 |
| 52 | 2 | Time Delay Fine | | | 129 | 07, quantity=1 |
| 60 | 01 | Class 0 Data | 1 | 06 | | |
| 60 | 02 | Class 1 Data | 1 | 06,07,08 | | |
| 60 | 03 | Class 2 Data | 1 | 06,07,08 | | |
| 60 | 04 | Class 3 Data | 1 | 06,07,08 | | |
| | | | | | | |
| 80 | 01 | Internal Indications | 2 | 00 index=7 | | |
| | | | | | | |
| No Object | | Cold Restart | 13 | | | |
| No Object | | Warm Restart | 14 | | | |
| No Object | | Delay Measurement | 23 | | | |

Note:

- Items highlighted grey indicates an object has not been implemented, parse only support.
- Items highlighted blue indicates an extra object, variation beyond level 2.

## Appendix C Device Profile (Master) - MOX Unity

<table>
<tr><td colspan="2">DNP V3.00<br>DEVICE PROFILE DOCUMENT</td></tr>
<tr><td colspan="2">Vendor Name: MOX Group</td></tr>
<tr><td colspan="2">Device Name: MOX Unity</td></tr>
<tr><td>Highest DNP Level Supported:<br><br>For Requests Level2<br><br>For Responses Level2</td><td>Device Function:<br><br>√ Master<br><br>☐ Slave</td></tr>
<tr><td colspan="2">Notable objects, functions, and/or qualifiers supported in addition to the Highest DNP Levels Supported (the complete list is described in the attached table):<br><br>Obj 10, Var 2, Fun code 130, Qualifier 17 & 28<br>Obj 22, Var 5, Fun code 129 & 130, Qualifier 17 & 28<br>Obj 22, Var 6, Fun code 129 & 130, Qualifier 17 & 28<br>Obj 30, Var 5, Fun code 129 & 130, Qualifier 00 & 01<br>Obj 32, Var 3, Fun code 129 & 130, Qualifier 17 & 28<br>Obj 32, Var 4, Fun code 129 & 130, Qualifier 17 & 28<br>Obj 32, Var 5, Fun code 129 & 130, Qualifier 17 & 28<br>Obj 32, Var 7, Fun code 129 & 130, Qualifier 17 & 28<br>Obj 40, Var 1, Fun code 129 & 130, Qualifier 00, 01, 17 & 28<br>Obj 40, Var 2, Fun code 130, Qualifier 17 & 28<br>Obj 40, Var 3, Fun code 129 & 130, Qualifier 00, 01, 17 & 28<br>Obj 41, Var 1, Request Fun code 3, 4, 5 & 6, Qualifier 17 & 28, Response Fun code 129<br>Obj 41, Var 3, Request Fun code 3, 4, 5 & 6, Qualifier 17 & 28, Response Fun code 129<br>Func code 14 (Warm Restart)</td></tr>
<tr><td>Maximum Data Link Frame Size (octets):<br><br>Transmitted ____292_______<br><br>Received (must be 292)</td><td>Maximum Application Fragment Size (octets):<br>Transmitted 2048<br>Received 2048</td></tr>
<tr><td>Maximum Data Link Re-tries:<br><br>☐ None<br>☐Fixed at ____________________<br>√ Configurable, range 0 to 65535</td><td>Maximum Application Layer Re-tries:<br><br>√ None<br>☐ Configurable<br>(Fixed is not permitted)</td></tr>
</table>

Requires Data Link Layer Confirmation:

☐Never
☐ Always
☐ Sometimes
√ Configurable

Requires Application Layer Confirmation:

√ Never
☐ Always (not recommended)
☐ When reporting Event Data (Slave devices only)
☐ When sending multi-fragment responses (Slave devices only)
☐ Sometimes
☐ Configurable

Timeouts while waiting for:

| | | | | |
|---|---|---|---|---|
| Data Link Confirm | ☐ None | ☐ Fixed at ____ | ☐ Variable | √ Configurable |
| Complete Appl. Fragment | ☐ None | ☐ Fixed at ____ | ☐ Variable | √ Configurable |
| Application Confirm | ☐ None | ☐ Fixed at ____ | ☐ Variable | √ Configurable |
| Complete Appl. Response | ☐ None | ☐ Fixed at ____ | ☐ Variable | √ Configurable |

Others ____________
See Note* 1

Sends/Executes Control Operations:

| | | | | |
|---|---|---|---|---|
| WRITE Binary Outputs | √ Never | ☐ Always | ☐ Sometimes | ☐ Configurable |
| SELECT/OPERATE | ☐ Never | ☐ Always | ☐ Sometimes | √ Configurable |
| DIRECT OPERATE | ☐ Never | ☐ Always | ☐ Sometimes | √ Configurable |
| DIRECT OPERATE - NO ACK | ☐ Never | ☐ Always | ☐ Sometimes | √ Configurable |
| | | | | |
| Count > 1 | √ Never | ☐ Always | ☐ Sometimes | ☐ Configurable |
| Pulse On | √ Never | ☐ Always | ☐ Sometimes | ☐ Configurable |
| Pulse Off | √ Never | ☐ Always | ☐ Sometimes | ☐ Configurable |
| Latch On | ☐ Never | √ Always | ☐ Sometimes | ☐ Configurable |
| Latch Off | ☐ Never | √ Always | ☐ Sometimes | ☐ Configurable |
| | | | | |
| Queue | √ Never | ☐ Always | ☐ Sometimes | ☐ Configurable |
| Clear Queue | √ Never | ☐ Always | ☐ Sometimes | ☐ Configurable |

See Note* 2

**FILL OUT THE FOLLOWING ITEM FOR MASTER DEVICES ONLY:**

| |
|---|
| Expects Binary Input Change Events:<br><br>√ Either time-tagged or non-time-tagged for a single event<br>☐ Both time-tagged and non-time-tagged for a single event<br>☐ Configurable (attach explanation) |
| **FILL OUT THE FOLLOWING ITEMS FOR SLAVE DEVICES ONLY:** |

| | |
|---|---|
| Reports Binary Input Change Events when no specific variation requested:<br><br>☐ Never<br>☐ Only time-tagged<br>☐ Only non-time-tagged<br>☐ Configurable to send both, one or the other (attach explanation) | Reports time-tagged Binary Input Change Events when no specific variation requested:<br><br>☐ Never<br>☐ Binary Input Change With Time<br>☐ Binary Input Change With Relative Time<br>☐ Configurable (attach explanation) |
| Sends Unsolicited Responses:<br><br>☐ Never<br>☐ Configurable (attach explanation)<br>☐ Only certain objects<br>☐ Sometimes (attach explanation)<br><br>☐ENABLE/DISABLE UNSOLICITED Function codes supported | Sends Static Data in Unsolicited Responses:<br><br>☐ Never<br>☐ When Device Restarts<br>☐ When Status Flags Change<br><br>No other options are permitted. |
| Default Counter Object/Variation:<br><br>☐ No Counters Reported<br>☐ Configurable (attach explanation)<br>☐ Default Object ______________<br>Default Variation ______________<br>☐ Point-by-point list attached<br>See Note* 3 | Counters Roll Over at:<br><br>☐ No Counters Reported<br>☐ Configurable (attach explanation)<br>☐ 16 Bits<br>☐ 32 Bits<br>☐ Other Value _____________<br>☐ Point-by-point list attached |
| Sends Multi-Fragment Responses: ☐ Yes ☐ No | |

Note* 1
Configurable timeouts can be set via MoxIDE software, refer to MoxIDE DNP3 master parameters "Data Link Timeout" and "Application Layer Timeout".

Note* 2
Configurable sends/executes control operations can be set via MoxGRAF software, refer to MoxGRAF DNP3 Master Setting's Output Edit parameter "Command Mode".

Note* 3

MOX Unity support 32-Bit Binary Counter, 16-Bit Binary Counter, configure in MoxGRAF (DNP3 Master Setting)

# Appendix D Interoperability Table (Master) - MOX Unity

| Object | | | REQUEST (slave must pares) | | RESPONSE (master must parse) | |
|---|---|---|---|---|---|---|
| Obj | Var | Description | Func Codes (dec) | Qual Codes (hex) | Func Code | Qual Codes (hex) |
| 1 | 1 | Binary Input | | | 129,130 | 00,01 |
| 1 | 2 | Binary Input with status | | | 129,130 | 00,01 |
| | | | | | | |
| 2 | 0 | Binary Input Change – All Variations | | | | |
| 2 | 1 | Binary Input Change without time | | | 129, 130 | 17,28 |
| 2 | 2 | Binary Input Change whit Time | | | 129, 130 | 17,28 |
| 2 | 3 | Binary Input Change with Relative Time | | | 129,130 | 17,28 |
| | | | | | | |
| 10 | 2 | Binary Output Status | | | 129,130 | 00,01 |
| 10 | 2 | Binary Output Status | | | 130 | 17, 28 |
| | | | | | | |
| 12 | 1 | Control Relay Output Block | 3,4,5,6 | 17,28 | 129 | Echo of request |
| | | | | | | |
| 20 | 1 | 32-Bit Binary Counter | | | 129,130 | 00,01 |
| 20 | 2 | 16-Bit Binary Counter | | | 129,130 | 00,01 |
| 20 | 3 | 32-Bit Delta Counter | | | 129,130 | 00,01 |
| 20 | 4 | 16-Bit Delta Counter | | | 129,130 | 00,01 |
| 20 | 5 | 32-Bit Binary Counter without Flag | | | 129,130 | 00,01 |
| 20 | 6 | 16-Bit Binary Counter without Flag | | | 129,130 | 00,01 |
| 20 | 7 | 32-Bit Delta Counter without Flag | | | 129,130 | 00,01 |
| 20 | 8 | 16-Bit Delta Counter without Flag | | | 129,130 | 00,01 |
| | | | | | | |
| 21 | 1 | 32-Bit Frozen Counter | | | 129,130 | 00,01 |
| 21 | 2 | 16-Bit Frozen Counter | | | 129,130 | 00,01 |
| 21 | 9 | 32-Bit Frozen Counter without Flag | | | 129,130 | 00,01 |
| 21 | 10 | 16-Bit Frozen Counter without Flag | | | 129,130 | 00,01 |
| | | | | | | |
| 22 | 1 | 32-Bit Counter Change Event without Time | | | 129,130 | 17,28 |
| 22 | 2 | 16-Bit Counter Change Event without Time | | | 129,130 | 17,28 |
| 22 | 5 | 32-Bit Counter Change Event with Time | | | 129,130 | 17,28 |
| 22 | 6 | 16-Bit Counter Change Event with Time | | | 192.130 | 17,28 |
| | | | | | | |
| 30 | 1 | 32-Bit Analog Input | | | 129,130 | 00,01 |
| 30 | 2 | 16-Bit Analog Input | | | 129,130 | 00,01 |
| 30 | 3 | 32-Bit Analog Input without Flag | | | 129,130 | 00,01 |
| 30 | 4 | 16-Bit Analog Input without Flag | | | 129,130 | 00,01 |
| 30 | 5 | Short Floating Point Analog Input | | | 129,130 | 00,01 |
| | | | | | | |
| 32 | 1 | 32-Bit Analog Input Change Event without Time | | | 129,130 | 17,28 |
| 32 | 2 | 16-Bit Analog Input Change Event without Time | | | 129,130 | 17,28 |
| 32 | 3 | 32-Bit Analog Input Change Event with Time | | | 129,130 | 17,28 |
| 32 | 4 | 16-Bit Analog Input Change Event with Time | | | 192,130 | 17,28 |
| 32 | 5 | Short Floating Point Analog Input Change Event without Time | | | 129,130 | 17,28 |
| 32 | 7 | Short Floating Point Analog Input Change Event with Time | | | 129,130 | 17,28 |
| | | | | | | |
| 40 | 1 | 32-Bit Analog Output Status | | | 129,130 | 00,01,17,28 |

| | | | | | | |
|---|---|---|---|---|---|---|
| 40 | 2 | 16-Bit Analog Output Status | | | 129,130 | 00,01 |
| 40 | 2 | 16-Bit Analog Output Status | | | 130 | 17,28 |
| 40 | 3 | Short Floating Point Analog Output Status | | | 129,130 | 00,01,17,28 |
| | | | | | | |
| 41 | 1 | 32-Bit Analog Output Block | 3,4,5,6 | 17,28 | 129 | Echo of request |
| 41 | 2 | 16-Bit Analog Output Block | 3,4,5,6 | 17,28 | 129 | Echo of request |
| 41 | 3 | Short Floating Point Analog Output Block | 3,4,5,6 | 17,28 | 129 | Echo of request |
| | | | | | | |
| 50 | 1 | Time and Date | 2 | 07 where quantity =1 | 129 | 07 |
| 51 | 1 | Time and Date CTO | | | 129,130 | 07, quantity=1 |
| 51 | 2 | Unsynchronized Time and Date CTO | | | 129,130 | 07, quantity=1 |
| 52 | 1 | Time Delay Coarse | | | 129 | 07, quantity=1 |
| 52 | 2 | Time Delay Fine | | | 129 | 07, quantity=1 |
| | | | | | | |
| 60 | 01 | Class 0 Data | 1 | 06 | | |
| 60 | 02 | Class 1 Data | 1 | 06,07,08 | | |
| 60 | 03 | Class 2 Data | 1 | 06,07,08 | | |
| 60 | 04 | Class 3 Data | 1 | 06,07,08 | | |
| | | | | | | |
| 80 | 01 | Internal Indications | 2 | 00 index=7 | | |
| | | | | | | |
| No Object | | Cold Restart | 13 | | 129 | 07, quantity=1 |
| No Object | | Warm Restart | 14 | | 129 | 07, quantity=1 |
| No Object | | Delay Measurement | 23 | | 129 | 07, quantity=1 |

Note:

- Items highlighted grey indicates an object has not been implemented, parse only support.
- Items highlighted blue indicates an extra object, variation beyond level 2.

# Appendix E Product Support

### Warranty Information

All MOX manufactured products are warranted to be free from defects in material and workmanship. Our obligation under this warranty will be limited to repairing or replacing, at our option, the defective parts within 1 year of the date of installation, or within 18 months of the date of shipment from the point of manufacture, whichever is sooner. Products may only be returned under authorization. The purchaser will prepay all freight charges to return any products with a valid return authorization number to the designated repair facility.

This limited warranty does not cover loss or damage that may occur in shipment of the goods or due to improper installation, maintenance, misuse, neglect or any cause other than ordinary commercial or industrial use. Warranty is also void if case is opened without manufacturer's consent. This limited warranty is in lieu of all other warranties whether oral or written, expressed or implied.

Liability associated with all MOX products shall not exceed the price of the individual unit that is the basis of the claim. In no event will there be liability for any loss of profits, loss of use of facilities or equipment or other indirect, incidental or consequential damages.

### Contact Details

To obtain support for MOX products, call MOX Group on the following numbers or your designated support provider and ask for MOX Support.

### E-mail addresses:

support@mox.com.au

sales@mox.com.au

### Visit our web page at:

http://www.mox.com.au

## Service Information

If you require service, contact your local MOX Group representative. A trained specialist will help you to quickly determine the source of the problem. Many problems are easily resolved with a single phone call. If it is necessary to return a unit, an RMA (Return Material Authorization) number will be provided.

All returned materials are tracked with our RMA system to ensure speedy service. You must include this RMA number on the outside of the box so that your return can be processed immediately.

Your MOX Group authorized applications engineer will complete an RMA request for you. If the unit has a serial number, we will not need detailed financial information. Otherwise, be sure to have your original purchase order number and date purchased available.

We suggest that you provide a repair purchase order number in case the repair is not covered under our warranty. You will not be billed if the repair is covered under warranty.

Please supply us with as many details about the problem as you can. The information you supply will be written on the RMA form and supplied to the repair department before your unit arrives. This helps us to provide you with the best service, in the fastest manner. Most repairs are completed within two days. During busy periods, there may be a longer delay.

If you need a quicker turnaround, ship the unit to us by airfreight. We give priority service to equipment that arrives by overnight delivery. Many repairs received by midmorning (typical overnight delivery) can be finished the same day and returned immediately.

We apologize for any inconvenience that the need for repair may cause you. We hope that our rapid service meets your needs. If you have any suggestions to help us improve our service, please give us a call. We appreciate your ideas and will respond to them.

## For Your Convenience:

Please fill in the following information and keep this manual with your MOX system for future reference:

P.O. #: __________________ Date Purchased: ___________________

Purchased From: ______________________________________________

**MOX Group**

Web: http://www.mox.com.au/
Email: info@mox.com.au

**DOC** 0809-902-2301 **VER** 1.02.04

Printed in Australia