MOTOROLA

Land Mobile Products Sector

Driver V2.00 of the DNP V3.00 (Slave)

Land Mobile Products Sector
16, Kremenetski Street, Tel Aviv 67899

User Guide

Printed in Israel
FVN4651B

COMMERCIAL WARRANTY (STANDARD)

Motorola radio communications products are warranted to be free from defects in material and workmanship for a period of ONE (1) YEAR, (except for crystals and channel elements which are warranted for a period of ten (10) years), from the date of shipment. Parts, including crystals and channel elements, will be replaced free of charge for the full warranty period but the labor to replace defective parts will only be provided for one Hundred-Twenty (120) days from the date of shipment. Thereafter purchaser must pay for the labor involved in repairing the product or replacing the parts at the prevailing rates together with any transportation charges to or from the place where warranty service is provided. This express warranty is extended by Motorola Communications and Electronics Inc., 1301 E. Algonquin Road, Schaumburg, Illinois 60196, to the original purchaser only, and only to those purchasing for purpose of leasing or solely for commercial, industrial, or governmental use.

THIS WARRANTY IS GIVEN IN LIEU OF ALL OTHER WARRANTIES EXPRESS OR IMPLIED WHICH ARE SPECIFICALLY EXCLUDED, INCLUDING WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. IN NO EVENT SHALL MOTOROLA BE LIABLE FOR INCIDENTAL OR CONSEQUENTIAL DAMAGES TO THE FULL EXTENT SUCH MAY BE DISCLAIMED BY LAW.

In the event of a defect, malfunction or failure to conform to specifications established by seller, or if appropriate, to specifications accepted by Seller in writing, during the period shown, Motorola, at its option, will either repair or replace the product or refund the purchase price thereof, and such action on the part of Motorola shall be the full extent of Motorola's liability hereunder.

This warranty is void if:

a. the product is used in other than its normal and customary manner;

b. the product has been subject to misuse, accident neglect or damage;

c. unauthorized alterations or repairs have been made, or unapproved parts used in the equipment.

This warranty extends only to individual products, batteries are excluded, but carry their own separate limited warranty. Because each radio system is unique, Motorola disclaims liability for range, coverage, or operation of the system as a whole under this warranty except by a separate written agreement signed by an officer of Motorola.

Non-Motorola manufactured products are excluded from this warranty, but subject to the warranty provided by their manufacturers, a copy of which will be supplied to you on specific written request.

In order to obtain performance of this warranty, purchaser must contact its Motorola salesperson or Motorola at the address first above shown, attention Quality Assurance Department.

This warranty applies only within the United States.

COMPUTER SOFTWARE COPYRIGHTS

The Motorola products described in this instruction manual may include copyrighted Motorola computer programs stored in semi conductor memories or other media. Laws in the United States and other countries preserve for Motorola certain exclusive rights for copyrighted computer programs including the exclusive right to copy or reproduce in any form the copyrighted computer program. Accordingly, any copyrighted Motorola computer programs contained in the Motorola products described in this instruction manual may not be copied or reproduced in any manner without the express written permission of Motorola. Furthermore, the purchase of Motorola products shall not be deemed to grant either directly or by implication, estoppel, or otherwise, any license under the copyrights, patents or patent applications of Motorola, except for the normal non-exclusive, royalty free license to use that arises by operation of law in the sale of a product.

Table of Contents

Glossary

MDLC	Motorola Data Link Communication Protocol
DNP	Distributed Network Protocol
RTU	Remote Terminal Unit
SCADA	Supervisory Control and Data Acquisition
MOSCAD	Motorola SCADA
MOSCAD-L	Motorola SCADA-Light
DNP+	Distributed Network Protocol with a set of MDLC features
IED	Intelligent Equipment Device
IIN	Internal Indication

Applicable Documentation

The MOSCAD system includes the following manuals:

- MOSCAD Programming Toolbox for Windows - Getting Started, Motorola publication no. 68P02949C80
- MOSCAD Programming Toolbox for Windows - User Guide, Motorola publication no. 68P02949C85
- MOSCAD Programming Toolbox for Windows - Special Functions, Motorola publication no. 68P02949C90
- MOSCAD Programming Toolbox for Windows - Third Party Protocols Support, Modbus and Allen Bradley, Motorola publication no. 68P02952C50
- MOSCAD Programming Toolbox for Windows - AGA8 Gas Flow Calculations, Motorola publication no. 68P02946C25
- MOSCAD RTU Service manual, Motorola publication no. 68P02991G90
- MOSCAD RTU Owner's manual, Motorola publication no. 68P02994G10
- MCP/M User’s Manual, Motorola publication no. 68P02945C05-0.

References

The following is a list of abbreviations which refer to the titles of the different manuals which comprise the MOSCAD Programming Toolbox guide.

These abbreviations are used throughout the book. For a complete description, refer to the list below:

Abbreviation	Manual Title
User Guide	MOSCAD Programming Toolbox - User Guide - Motorola publication no. 68P02949C85
DNP Guide	DNP V3.00 Subset Definitions, Version 1.00 - The network protocol written by Harris Canada, Inc.

Terms and Conventions

The MOSCAD RTU is shipped in two versions, MOSCAD RTU and MOSCAD-L RTU. Most of the features described in this manual are common to MOSCAD and MOSCAD-L. Throughout the documentation the terms "RTU" and "MOSCAD" refer to the "generic" system. Differences are indicated by specific references to MOSCAD-L.

Overview

This manual describes the Distributed Network Protocol (DNP) V3.00 as implemented in the DNP driver (slave) for MOSCAD. The driver conforms to the Level 3 specifications provided in *DNP V3.00 Subset Definitions* by Harris Canada, Inc. (document number P009-0IG.SUB).

This manual covers the DNP implementation for both MOSCAD V3.70 or later and MOSCAD-L V1.00 or later. It explains how the DNP protocol functions in relation to reading from and writing to the RTU database. The manual is, therefore, intended for MOSCAD application programmers.

The driver described in this manual represents an implementation of DNP V3.00 Level 3 for communicating between a master and an Intelligent Equipment Device (IED), such as RTU). The driver satisfies the requirements of the DNP V3.00 Level 3.

Two drivers are supported: DNP and DNP+.

The DNP+ driver enhances the standard DNP 3.0 protocol with a set of features that are currently included only as a part of the MDLC protocol. Some of these features are: *Remote diagnostics* of the RTUs, efficient *time synchronization*, *remote monitoring* of applications, and *uploading* and *downloading* updated parameters. This allows the MOSCAD ToolBox user to communicate via MDLC-coded massages over the physical channel, with the MDLC-enabled capabilities of the MOSCAD. DNP-aware IEDs use that same physical channel to communicate with other DNP-aware devices. The DNP+ *time synchronization* is very accurate but in that port time synchronization can be done only via the DNP 3.0 protocol. . The DNP+ driver implements the DNP 3.0 channel access mechanism, while the DNP driver does not.

The DNP+ protocol includes advanced features which are defined in the DNP 3.0 and MDLC protocols. To achieve seamless communication over the RF network, the combined system uses the DNP 3.0 type channel access mechanism and time synchronization for both the MOSCAD and the DNP 3.0 type components. Unique features such as remote diagnostics of RTUs, remote monitoring of applications, upload and download of updated parameters, use the MDLC protocol, and these features are available only for MOSCAD devices.

Since the DNP+ driver uses the DNP 3.0 type channel access, any MOSCAD in the DNP+ system should be loaded with the DNP+ driver, even if it uses the MDLC protocol only.

The DNP and the DNP+ drivers implement the DNP protocol standards in the same way. They differ only in the port declarations.

Unless otherwise indicated, this document applies to both DNP and DNP+. The term DNP stands for both DNP and DNP+, unless an express reference is made to the specific driver.

Driver of the DNP V3.00 (Slave)

This chapter covers the DNP driver for the slave side. The driver is written as a 'C' block and is not part of the MOSCAD system; but it is incorporated into a new or existing MOSCAD application using the MOSCAD Toolbox software. The driver itself is downloaded as a 'C' block to the flash memory and is activated by a ladder process.

The User Interface

The DNP V3.00 driver functionality is incorporated into the MOSCAD application using the regular MOSCAD Toolbox tools. The driver interfaces to the application via:

- Site configuration parameters
- Database tables
- Rungs

You will use the Site Configuration and Application Programmer tools to define the DNP port (Site Configuration) and the database and ladder rungs (Application Programmer) in order to merge driver functionality with the overall MOSCAD application. Therefore, you can easily incorporate the DNP driver into new or existing MOSCAD applications, by simply defining a port, adding the DNP-specific tables, and developing the necessary ladder rungs.

DNP supports the following data types ("I/O points"):

- Binary Input
- Binary Output
- Analog Input
- Analog Output
- Counter
- Freeze counter
- Floating-point

Site Configuration Definitions

Site configuration is a two-step process: defining the port and setting its advanced parameters.

The DNP driver is implemented on RS-232/RS-485 ports. The DNP+ driver is implemented on RS-232(Radio Darcom or RS232 Multi-drop fiber optic)/RS-485 ports. Up to two ports

of the RTU can be defined as a slave DNP port, each of them is handled by a different driver.

Defining the Port

To set the DNP port, start the Site Configuration module and define:

- Port 1 as RS-485 User Port (Ladder Controlled)

or

- Port 1, 2, or 3 as RS-232 User Port (Ladder Controlled)

To set the DNP+ port, start the Site Configuration module and define:

- Port 1 as RS-485 Multi-drop or as RS-232 Async RTU-to-RTU

or

- Port 2 or 3 as RS-232 External Modem Async (Multi-drop half-duplex without CD or Darcom) or as RS-232 Async RTU-to-RTU

Port 2 is applicable only to MOSCAD, and not to MOSCAD-L.

Up to two DNP master ports can function simultaneously.

The port's associated link name should be passed as a permanent parameter via the Permanent Configuration Parameters table discussed later in this manual.

NOTE

For RS-485 the protocol does not support channel access functionality in user ports. This means that the unsolicited ("burst" in MOSCAD terms) frames mechanism may work but data collision may occur if multiple RTUs try to communicate simultaneously.

Setting the Advanced Parameters

To implement DNP on the user port, you will set several advanced parameters to the values provided below.

- **Dynamic RAM size**

1. Open the Advanced menu.
2. Choose General System Parameters.
3. Choose System Values.
4. Set the value to 30.

RAM size for dynamic allocation<0-4000>K-bytes.....[2]:**30**

30K is set aside to enable the driver to allocate the necessary RAM for managing the database and communications via the selected DNP port. For two drivers define 60K. For a larger database, you can increase the dynamic RAM size, depending on the memory available on the RTU. You can always add more memory to the RTU using a RAM expansion board.

- **Error Logger RAM Size**

1. Open the Advanced menu.
2. Choose General System Parameters.
3. Choose Buffers/Queues Size.
4. Set the value to 3000.

Error logger buffer size <100-2000>bytes .....[300]:**3000**

- **User port buffer size**

1. Open the Advanced menu.
2. Choose MDLC and User Port Heaps.
3. Choose More.
4. For the RS232 port set the value to 100, and for RS485 set the value to 200.

Ladder-Diagram user port buffer size<10-1000>bytes [50]:**100**

This parameter should be set according to the Ladder Main Process scan time. Eight 100-byte buffers are available for reception (each buffer has 16 bytes overhead). These buffers should be sufficient to store all characters received during one scan time. For example, under a baud rate of 9600, each character takes about 1 msec to arrive. If the scan time is greater than 8*84 msec, increase this parameter.

This parameter is applicable for DNP only.

- **Unformatted (Adapter) buffers**

1. Open the Advanced menu.
2. Choose MDLC and User Port Heaps.
3. For the RS232 port set the value to 100, and for RS485 set the value to 200.

Unformatted (Adapter) buffer size<10-1000>bytes .....[50]:**100**

The size should be set according to the Ladder Main Process scan time. Eight 100-byte buffers are available for reception (each buffer has 16 bytes overhead). These buffers should be sufficient to store all characters received during one scan time. For example, under a baud rate of 9600, each character takes about 1 msec to arrive. If the scan time is greater than

8*84 msec, increase this parameter. In addition, if you get the #1952 error or unexpected CRC errors, you should increase this value.

This parameter is applicable for DNP+ only.

- **Number of idle to announce 'End of RX'**

1. Open the Advanced menu.
2. Choose Ports.
3. Choose Port x (x stands for the port number).
4. Choose Advanced Physical Layer Parameters.
5. Set the value to 50.

Number of idle to announce 'End of RX' <1-200>.....[80]:**50**

Announce the end of reception after 50 idles. For example, under a baud rate of 9600 each character takes about 1 msec to arrive. Therefore end of RX is announced after about 50 msec.

This parameter is applicable for DNP only.

- **User tasks for 'C' blocks**

See the note in section *Permanent Configuration Parameters Table* page *11*.

- **I2Phys function table size**

1. Open the Advanced menu.
2. Choose General System Parameters.
3. Choose System Values.
4. Choose More.
5. Choose More.
6. Set the value to 3 for one driver, 6 for two drivers.

Size of I2Phys function table <0-255>.....[0]:**3**

This parameter is applicable for DNP+ only.

Application Programmer Definitions

At the application level, you will configure the DNP by defining user tables and ladder processes.

You define the following DNP-specific user tables:

- Permanent Configuration Parameters table
- Variable Configuration Parameters table
- Statistics table
- Object 12 Variation 1 table (optional)
- Binary Select Before Operate table (optional)
- Discrete Output Status table (optional)
- I/O tables
- Unsolicited Response table(s).

NOTE

The table and variable names provided here are arbitrary and have no impact on the application.

Permanent Configuration Parameters Table

The Permanent Configuration Parameters table is used to store values that may not change during the lifetime of the application. For example, the DNP port number, the master address, or the number of Binary Input points are permanent parameters.

It should be defined as a single-column table of Integer Parameter type and must contain at least 44 rows. The following figure illustrates such a table.

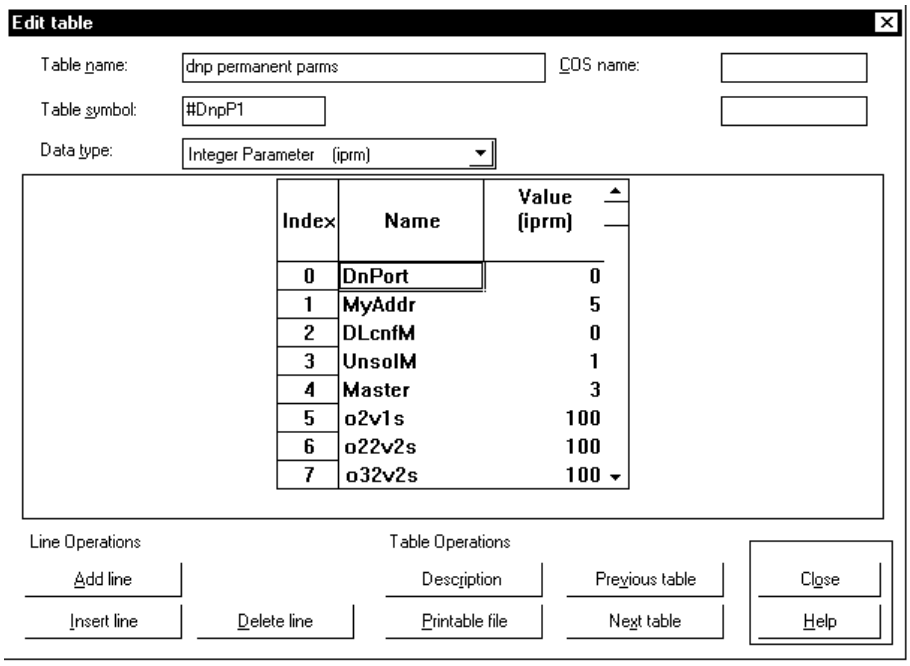

Figure 1: Permanent Configuration Parameters Table

Not all of the 44 rows are used by DNP and some rows are reserved for future use. But you must define at least 44 rows all the same.

The Permanent Configuration parameters are as follows:

- **DnPort**

Port number or port's link ID, depending on the driver.

For the DNP driver, this is the communication port number that corresponds to the logical name defined in the Site Configuration.

The following chart lists the valid values and their meanings.

Value	Meaning
0	User1
1	User2
2	User3

NOTE

For the DNP driver, it is highly recommended to work with ports 2 or 3, because they support flow control (define the port with the option of DTR/CTS). Working with port 1 requires enlarging the buffers or the baud rate.

For the DNP+ driver, this is the link ID that corresponds to the communication port as defined in the Site Configuration. The following chart lists the valid values and their meanings.

Value	Meaning
21 to 29	LINE 1 - LINE 9 respectively
51 to 69	RSLink 1 - RSLink 19 respectively
80 to 99	LINE 10 - LINE 29 respectively

- **MyAddr**

The DNP address of the driver.

The address can be a number from 0 to 65534.

The following chart lists the valid values and their meanings.

Value	Meaning
0	The DNP address is identical to its MOSCAD Site ID (SelfID).
1 to 65534	An address different from the RTU's actual Site ID

- **DLcnfM**

Data link confirm mode. This parameter allows you to reduce communications by limiting confirmation messages.

The following chart lists the valid values and their meanings.

Value	Meaning
0	Never confirm.
1	Issue confirmations only on multi-frame fragments.
2	Always confirm.

- **UnsolM**

Unsolicited response mode. A slave RTU issues an unsolicited message to report an unusual event. This parameter allows you to disable or enable this feature.

The following chart lists the valid values and their meanings.

Value	Meaning
0	Disable unsolicited responses
1	Enable unsolicited responses

- **Master**

The address of the master.

The valid values are 0 through 65535.

- **o2v1s**

Number of Binary events without time stamp that can be handled simultaneously.

The buffer allocated for storing the Binary events without time stamp requires about 6 bytes per event. It is, therefore, recommended to consider the size carefully.

The valid values range from 0 to 1000.

- **o22v2**

Number of Counter events without time stamp that can be handled simultaneously.

The buffer allocated for storing the Counter events without time stamp requires about 8 bytes per event. It is, therefore, recommended to consider the size carefully.

The valid values range from 0 to 1000.

- **o32v2s**

Number of Analog events without time stamp that can be handled simultaneously.

The buffer allocated for storing the Analog events without time stamp requires about 8 bytes per event. It is, therefore, recommended to consider the size carefully.

The valid values range from 0 to 1000.

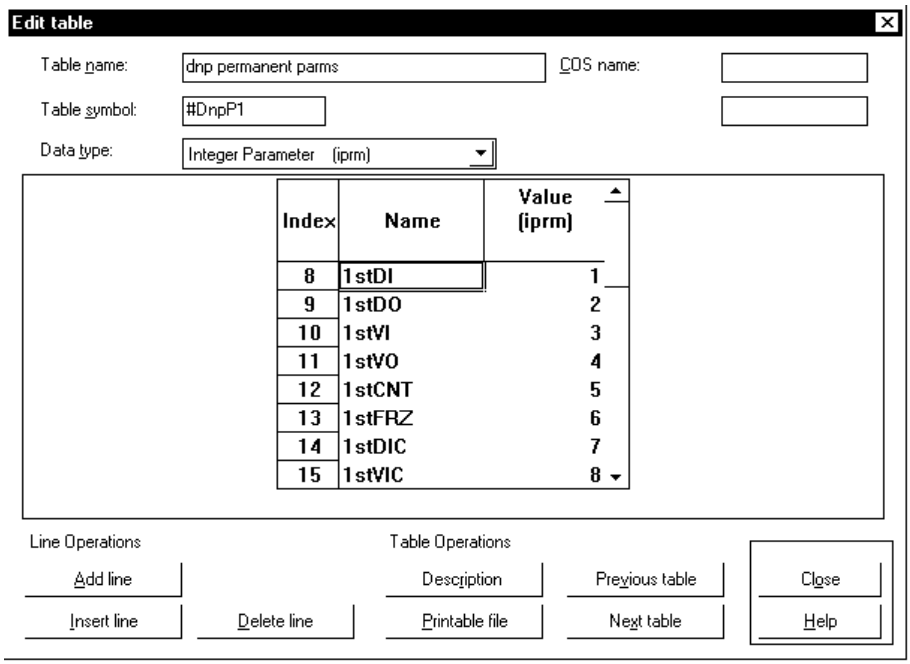

Figure 2: Permanent Configuration Parameters Table (cont.)

- **1stDI**

Number (index) of the first Binary Inputs table.

The driver must know the size of the Binary Input tables in the database, in order to relate to the variables correctly. As you may not always know the size of the table while developing the application (or when adding DNP support to an existing application), you will use this and the nDI (Number of Binary Input points) parameter to set the number (index) of the first Binary Input Table and the number of the input points. These parameters set the size of the table.

The valid values are 0 through 126 (the maximum number of user tables). To ensure correct reference to variables, all the Binary Inputs tables must be recorded in consecutive order (one after the other).

- **1stDO**

Same as 1stDI for Binary Outputs table.

- **1stVI**

Same as 1stDI for Analog Inputs table.

- **1stVO**

Same as 1stDI for Analog Outputs table.

- **1stCNT**

Same as 1stDI for Counters table.

- **1stFRZ**

Same as 1stDI for Freeze Counters table.

- **1stDIC**

Same as 1stDI for Binary Input Classes table.

- **1stVIC**

Same as 1stDI for Analog Input Classes table.

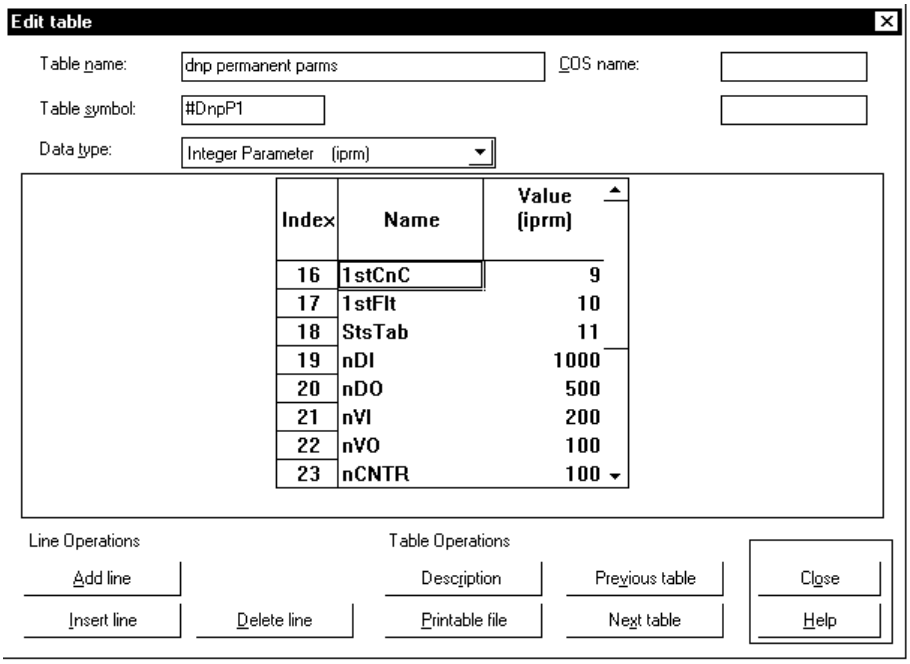

Figure 3: Permanent Configuration Parameters Table (cont.)

- **1stCnC**

Same as 1stDI for Counter Classes table.

- **1stFlt**

Same as 1stDI for Floating-Points table.

- **StsTab**

Number (index) of the Statistics table.

- **nDI**

Number of Binary Input points (0 through 10000). See explanations for 1stDI.

- **nDO**

Number of Binary Output points (0 through 10000). See explanations for 1stDI.

- **nVI**

Number of Analog Input points (0 through 3000). See explanations for 1stDI.

- **nVO**

Number of Analog Output points (0 through 3000). See explanations for 1stDI.

- **nCNTR**

Number of Counter points (0 through 3000). See explanations for 1stDI.

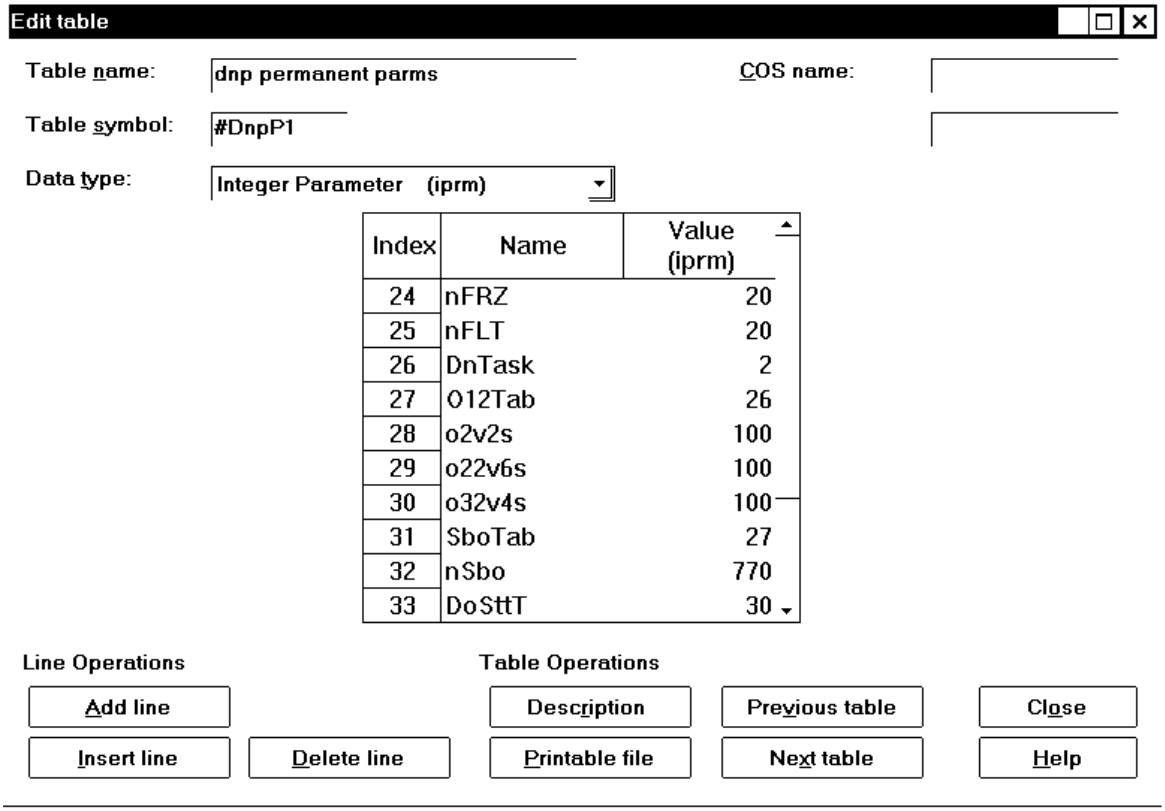

Figure 4: Permanent Configuration Parameters Table (cont.)

- **nFRZ**

Number of FREEZE Counter points (0 through 3000). See explanations for 1stDI.

nFRZ should be less than or equal to nCNTR. If it is less than nCNTR, then:

- The FREEZE function will freeze the first nFRZ counters. The rest of the counters will be ignored.
- The FREEZE and CLEAR functions will freeze and clear the first nFRZ counters. The rest of the counters will be only cleared (their values will be lost).

- **nFLT**

Number of floating-point points (0 through 3000)

- **DnTask**

The internal driver task number. The following chart lists the valid values and meanings.

Value	Meaning
0	Default value: Task Priority B for DNP, Task Priority E for DNP+.
1	Task Priority A (Applicable to DNP only)
2	Task Priority B (Applicable to DNP only)
3	Not used
4	Not used
5	Task Priority E
6	Task Priority F
7	Task Priority G
8	Task Priority H
9	Task Priority I
10	Task Priority J
11	Task Priority K
12	Task Priority L
13	Task Priority M
14	Task Priority N

NOTE

1. If two ports are defined as slave DNP ports, they should have different DnTask values.
2. Do not build a process (in the ladder diagram) or other C Block applications with the same priority as DnTask.
3. Usage of Task Priority E - M requires changing the following advanced parameter in the Site Configuration:

Number of User tasks for 'c' blocks

1. Open the Advanced menu.
2. Choose General System Parameters.
3. Choose System Values and click on the More button.
4. Set the value to 1 for task priority E, 2 for task priority F, etc...

- **O12Tab**

Number (index) of the Object 12 Variation 1 Information table. See *Object 12 Variation 1 Information Table* on page 28.

The valid values are -1 through 126 (the maximum number of user tables). The following chart lists the valid values and their meanings.

Value	Meaning
-1	No table is configured. PULSE_ON and PULSE_OFF are not supported.
0 to 126	A table number.

- **o2v2s**

Number of Binary events with time stamp that can be handled simultaneously.

The buffer allocated for storing the Binary events with time stamp requires about 12 bytes per event. It is, therefore, recommended to consider the size carefully.

The valid values range from 0 to 1000.

- **o22v6s**

Number of Counter events with time stamp that can be handled simultaneously.

The buffer allocated for storing the Counter events with time stamp requires about 14 bytes per event. It is, therefore, recommended to consider the size carefully.

The valid values range from 0 to 1000.

- **o32v4s**

Number of Analog events with time stamp that can be handled simultaneously.

The buffer allocated for storing the Analog events with time stamp requires about 14 bytes per event. It is, therefore, recommended to consider the size carefully.

The valid values range from 0 to 1000.

- **SboTab**

Same as 1stDI for Binary Select Before Operate table, except that it may also take the value -1. See *Binary Select Before Operate* Table on page 30.

The valid values are -1 through 126 (the maximum number of user tables). The following chart lists the valid values and their meanings.

Value	Meaning
-1	No table is configured. Binary Select before Operate is not supported.

0 to 126	A table number. Binary Select before Operate is supported for Binary Output points.

- **nSbo**

Number of Binary Select Before Operate points (0 through 10000). See explanations for 1stDI. This number should be less or equal to **nDI**.

- **DoSttT**

Same as 1stDI for DO Status table except that it may also take the value -1. The size and amount of DO status tables should be set according to **nDO**. See *Discrete Output (DO) Status Tables* on page 32.

The valid values are -1 through 126 (the maximum number of user tables). The following chart lists the valid values and their meanings.

Value	Meaning
-1	No table is configured. PULSE_ON and PULSE_OFF are not supported.
0 to 126	A table number.

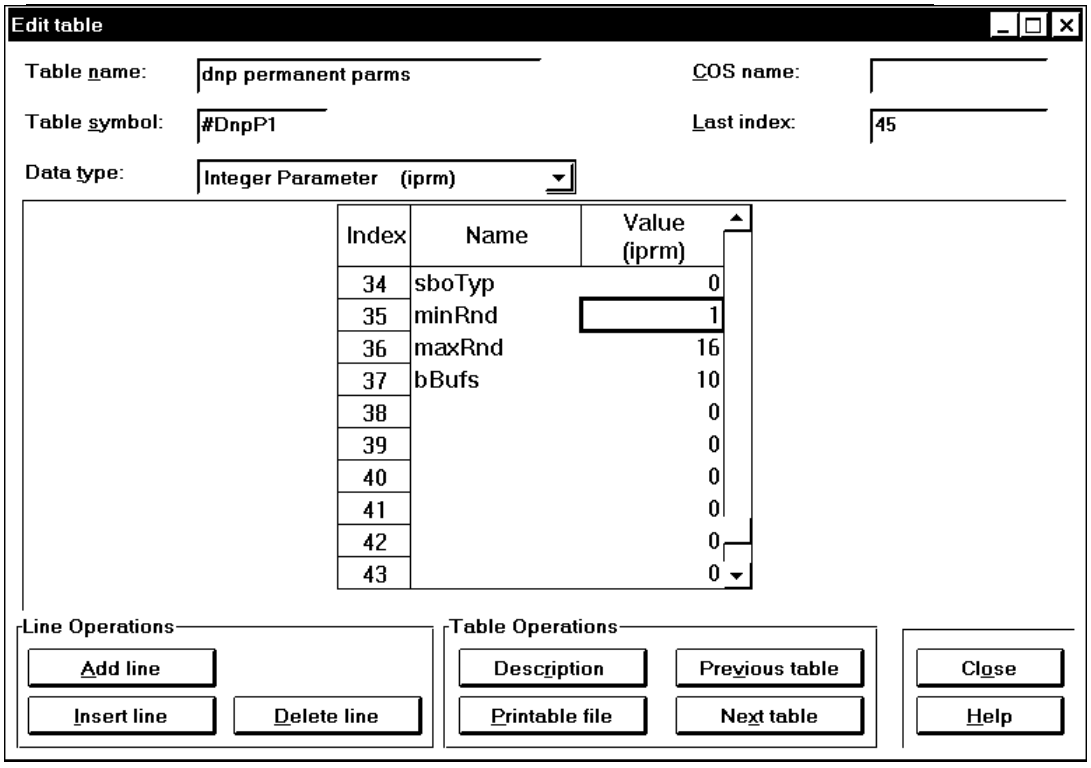

Figure 5: Permanent Configuration Parameters Table (cont.)

- **sboTyp**

Select before Operate implementation. Two implementations are available:

1. The driver implements the selection by closing output relays. The user application cannot interfere in the selection success during run-time, only by linking the appropriate bits to the appropriate I/Os. It is recommended to use this implementation when I/O modules are available.

2. The user application is responsible for setting which points should be selected, at run-time. This implementation does not require any I/O module.

See section *Binary Select Before Operate Tables* on page 30 for more details.

The following chart lists the valid values and their meanings.

Value	Meaning
0	The driver implements the selection by closing output relays.
1 to 65534	The driver implements the selection according the user application. This value is used as the time that the driver waits after it marks the selected point (step 3 above), in 100 milliseconds units.

- **minRnd**

Minimum random number for transmission. Applicable to the DNP+ driver only.

To avoid collisions, each station must wait a random quiet time. In this case, a collision may occur only if two stations decide to wait for the same amount of time. For each transmission, the driver uses a new random number. **minRnd** and **maxRnd** limit the minimum and the maximum values of this number.

- **maxRnd**

Maximum random number for transmission. See explanation for **minRnd.** Applicable to the DNP+ driver only.

- **nBufs**

Number of formatted buffers that can be received and stored until the driver task handles them. Applicable to the DNP+ driver only.

It is recommended to use the default value (10), unless working with a radio

Variable Configuration Parameters Table

The Variable Configuration Parameters table stores the parameters that can be modified during the lifetime of the application.

The table should be defined as single-column of Integer Parameter type. It must contain at least 21 rows. The following figure illustrates a Variable Configuration Parameters table.

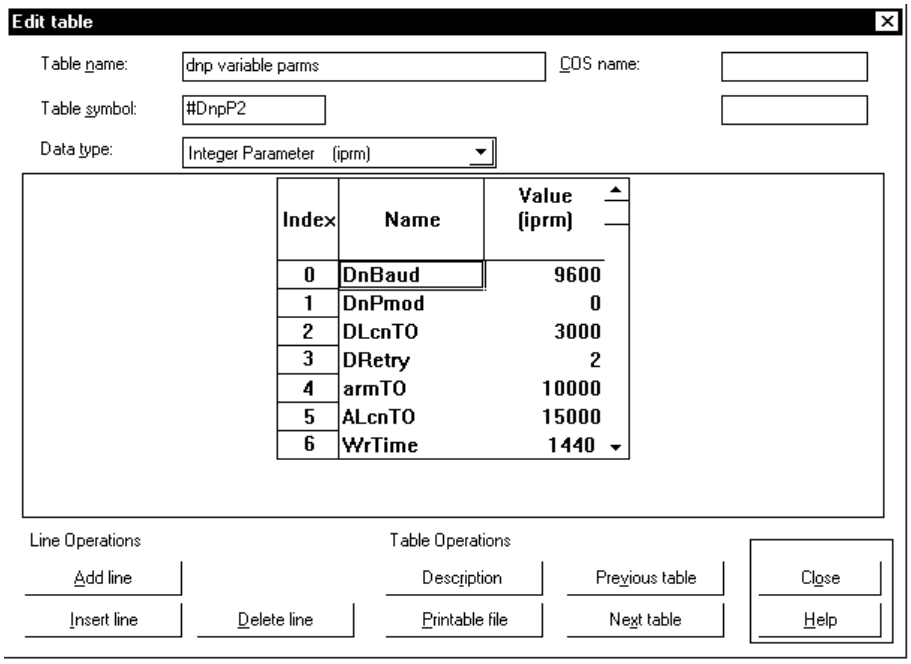

Figure 6: Variable Configuration Parameters Table

The Variable Configuration parameters are as follows:

- **DnBaud**

DNP port baud rate (600, 1200, 2400, 4800, 9600, 19200, 38400, 57600). Applicable to the DNP driver only.

- **DnPmod**

Communication port mode. Applicable to the DNP driver only. The valid values are 0 through 7. The following chart lists the valid values and their meanings.

Value	Meaning
0	8 Bit, No parity, 1 Stop bit
1	7 Bit, Even parity, 2 Stop bits
2	7 Bit, Odd parity, 2 Stop bits
3	7 Bit, Even parity, 1 Stop bit
4	7 Bit, Odd parity, 1 Stop bit
5	8 Bit, No parity, 1 Stop bit

Value	Meaning
6	8 Bit, Even parity, 2 Stop bits
7	8 Bit, Odd parity, 1 Stop bit

- **DLcnTO**

Data link confirm timeout, in milliseconds. This is the time the slave RTU waits for the master RTU's data link layer to confirm the last frame sent before starting the retries.

Define a timeout greater than 0 if the permanent DLcnfM parameter was set to 1 or 2. Otherwise, set it to 0.

For DNP+, always define 0. The link level time-outs are part of the MDLC parameters.

Valid values are 0 through 65535.

- **DRetry**

Data Link retry. This is the number of times the slave will retransmit a frame if the master does not confirm it.

Define a value greater than 0 if the permanent DLcnfM parameter was set to 1 or 2. Otherwise, set it to 0.

Valid values are 0 through 255.

- **armTO**

SELECT/OPERATE arm timer duration, in milliseconds. This is the time the slave RTU is allowed wait for the OPERATE command after receiving the SELECT command. After this time, the OPERATE command will not be carried out.

Valid values are 0 through 65535.

- **ALcnTO**

Application Layer confirm timeout, in milliseconds. This is the time the Application Layer waits for the confirmation to arrive. This time must be longer than the total retry time allocated to the Data Link layer.

Define a timeout greater than 0 if the permanent DLcnfM parameter was set to 1 or 2. Otherwise set it to 0.

Valid values are 0 through 65535.

- **WrTime**

Write time interval, in minutes. This is the time interval between slave device's update requests from the master ("need time" IIN bit should be set to 1).

Set this value to 0 if the device will not support report by exception, or if the slave device does not have any reason to synchronize its time with the master's (no time stamped data reported).

Valid values are 0 through 65535; 0 means "never update".

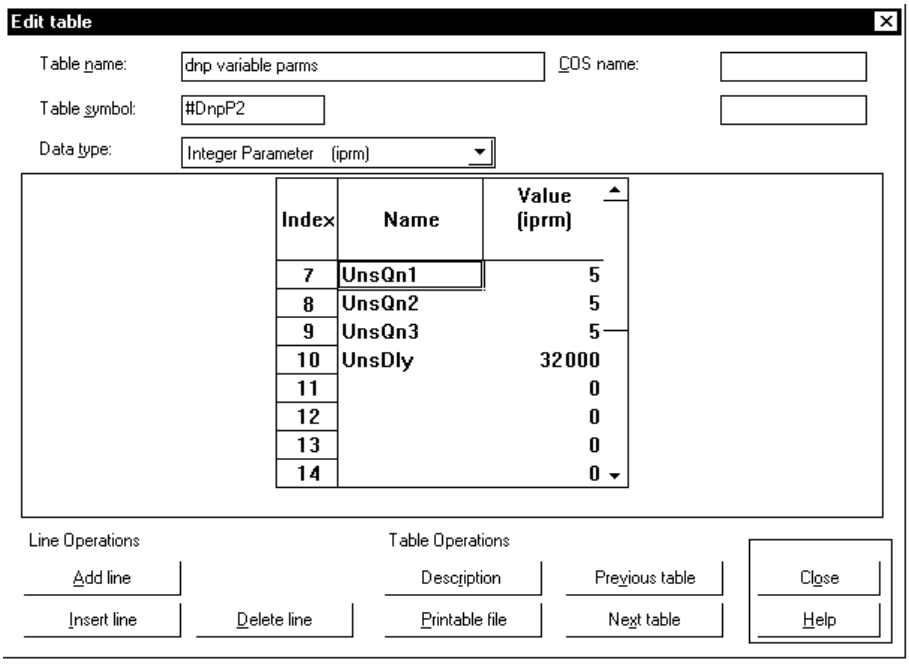

Figure 7: Variable Configuration Parameters Table (cont.)

- **UnsQn1**

Unsolicited response minimum quantity for class 1. This is the minimum number of events in class 1 that generate an unsolicited response. For example, if you set the parameter to 4, a burst occurs after four unusual events. See UnsDly parameter.

This parameter is relevant if the permanent UnsolM parameter is set to 1.

Valid values are 1 through 255.

- **UnsQn2**

Same as UnsQn1 for class 2.

- **UnsQn3**

Same as UnsQn1 for class 3.

- **UnsDly**

Unsolicited response notification delay, in milliseconds. This parameter allows the RTU to issue unsolicited responses before the minimum number of events, as defined via UnsQn1, UnsQn2 and UnsQn3, take place. You allow this by setting a time limit to wait for the minimum number to be reached.

Valid values are 0 through 65535. The following chart lists the valid values and their meanings.

Value	Meaning
0	Do not issue an unsolicited message unless the minimum number of events accumulate.
1 through 65535	Wait the indicated number of milliseconds and then issue the unsolicited message even if the minimum number of events has not been reached yet.

Statistics Table

The Statistics table stores valuable statistical and diagnostic data about driver behavior. You will use this table to obtain information about such unusual occurrences as too many retries or framing errors which reduce efficiency.

To use the Statistics table, define the first index of the table via the permanent StsTab parameters.

The Statistics table should be defined as single-column of Integer Parameter type. It must contain at least 25 rows. Figure 8 illustrates the table.

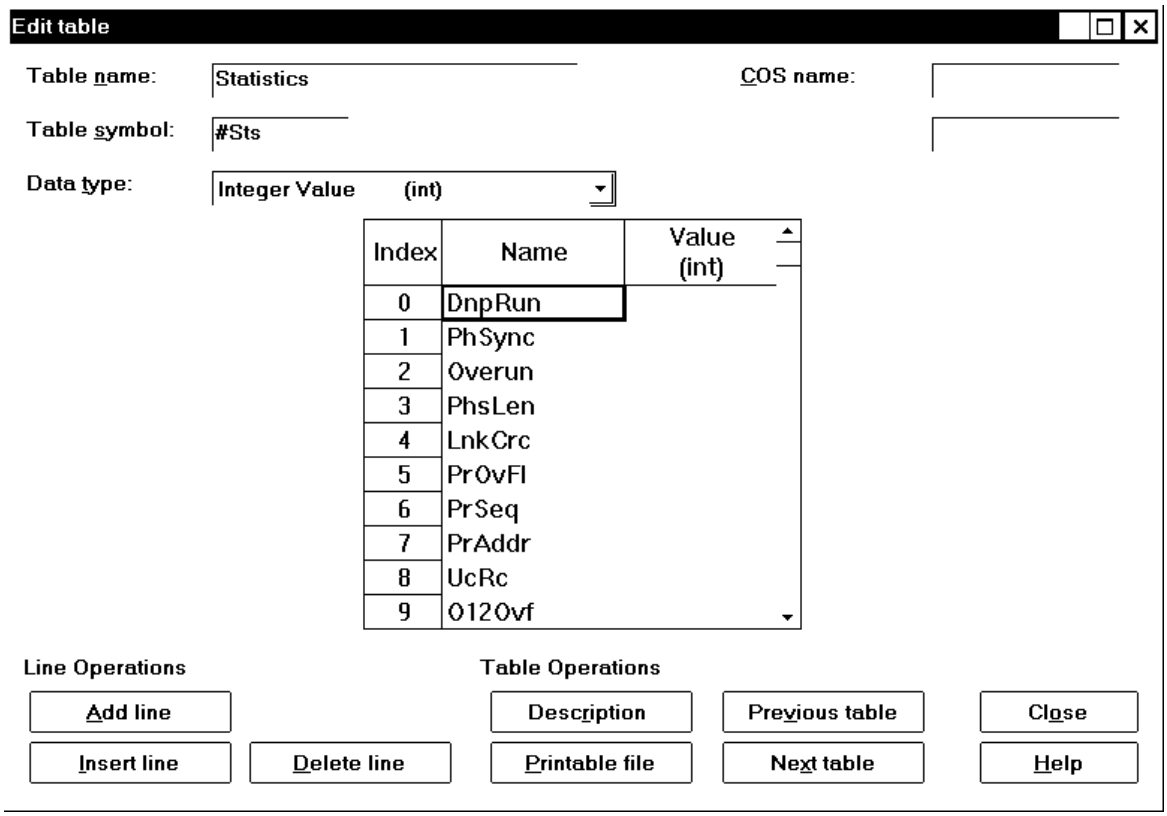

Figure 8: Statistics Table

The Statistics parameters are as follows:

- **DnpRun**

This flag indicates whether the DNP driver (task) is running (1) or not (0).

- **PhSync**

Number of times that extra bytes were received before the start bytes (0x05, 0x64; Physical Layer).

- **Overrun**

Number of times that the mainline Data Link routine could not read serial receive data before it was overwritten (Physical Layer).

- **PhsLen**

Number of times that the invalid length byte was received (Physical Layer).

- **LnkCrc**

Number of frame received with a CRC error (Link Layer).

- **PrOvFl**

Number of times that the Application Layer receive fragment buffer was too small (Transport Layer).

- **PrSeq**

Number of times that the sequence number of multi-frame request fragment has not incremented correctly (Transport Layer).

- **PrAddr**

Source addresses in consecutive multi-frames do not match (Transport Layer).

- **UcRc**

UCALLs Return code. The value is 0 for proper execution. Applicable for some UCALLs. See section *UCALLs Return Codes*.

- **O12Ovf**

Object 12 variation 1 Information table overflow. The value is 0 for proper execution.

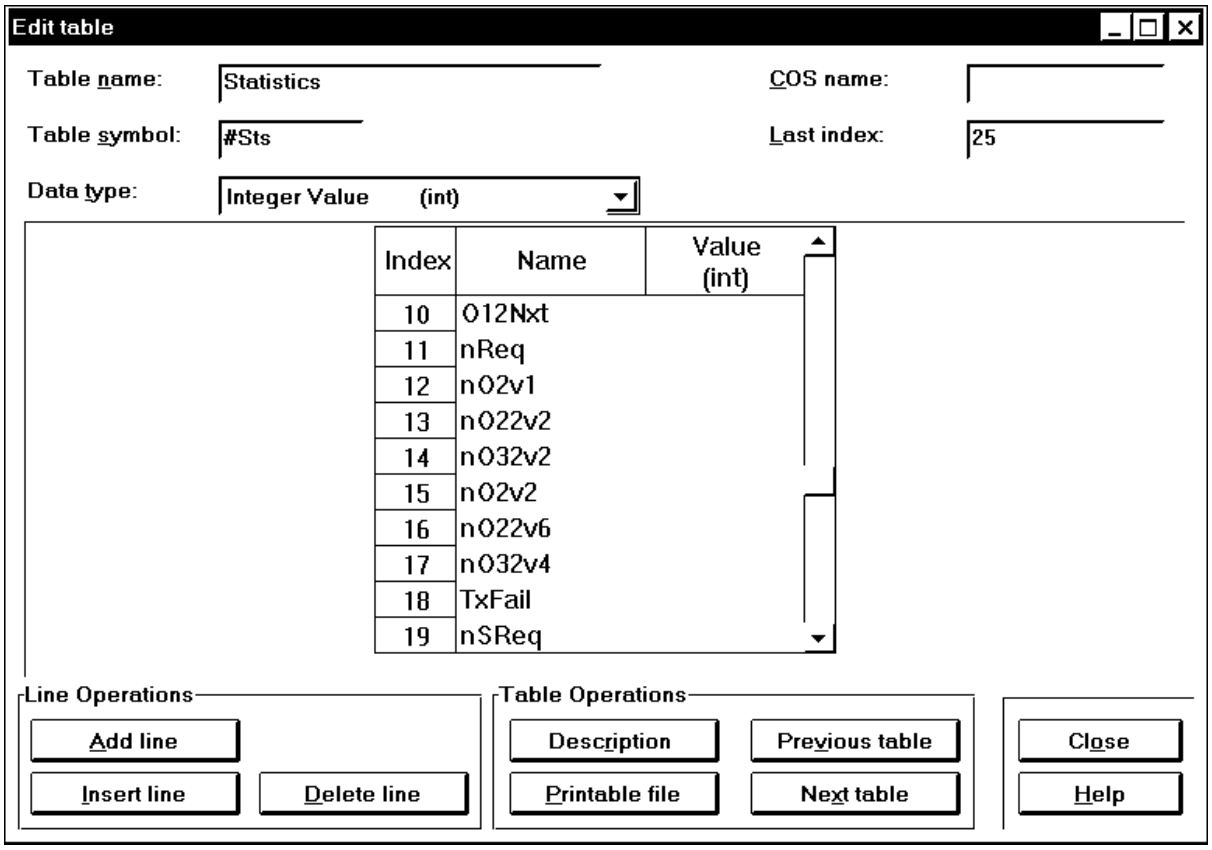

Figure 9: Statistics Table (Cont.)

- **O12Nxt**

Next row to fill in Object 12 variation 1 Information table.

- **nReq**

Number of requests that have been received and processed.

- **nO2v1**

Number of events currently stored in the Binary events without time stamp buffer.

- **nO22v2**

Number of events currently stored in the Counter events without time stamp buffer.

- **nO32v2**

Number of events currently stored in the Analog events without time stamp buffer.

- **nO2v2**

Number of events currently stored in the Binary events with time stamp buffer.

- **nO22v6**

Number of events currently stored in the Counter events with time stamp buffer.

- **nO32v4**

Number of events currently stored in the Analog events with time stamp buffer.

- **txFail**

Number of times that transmission failed.

- **nSReq**

Number of Select before Operate requests that have been received and processed.

Object 12 Variation 1 Information Table

The Object 12 Variation 1 Information table stores the object 12 variation 1 controls 1 (Pulse On) and 2 (Pulse Off), while controls 3 (Latch On) and 4 (Latch Off) are stored directly in the Binary Output tables.

This table is handled as a cyclic queue. The driver writes the requests one after the other and the ladder handles this information and marks -1 in the first column, to indicate that this row is free again.

The driver stores the next free row to be written in. You can see this value in O12Nxt, in the Statistic table, but you cannot change it.

When the driver receives a new request with object 12 variation 1 control 1 or 2, it first checks whether the next row to fill is free (-1 in the first column). If it is, the driver fills that row and increments (cyclically) the next row to write in, or else it marks overflow in O12Ovf in the Statistic table.

To use the Object 12 Variation 1 Information table, define the first index of the table via the permanent O12Tab parameter.

The table should be defined as multi-column. The following figure illustrates the table.

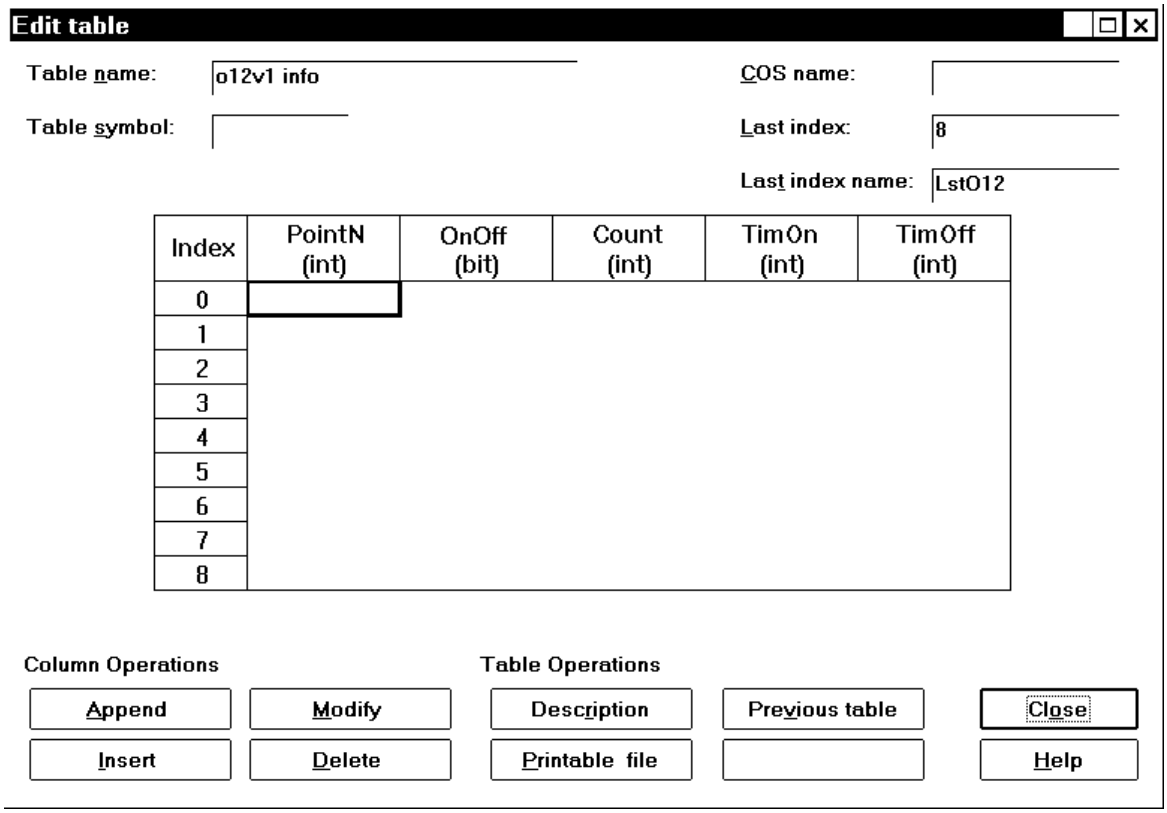

Figure 10: Object 12 Variation 1 Information table

The Object 12 Variation 1 Information includes:

- **PointN**

Point number

- **OnOff**

Control byte. The valid values are:

Value	Meaning
0	Pulse off. The point is turned off for the specified TimOff, and then turned on for the specified TimOn, as many as Count times. After Count times, it is left in on state.
1	Pulse on. The point is turned on for the specified TimOn, and then turned off for the specified TimOff, as many as Count times. After Count times, it is left in off state.

- **Count**

Number of times to count.

- **TimOn**

On-Time in milliseconds. Values greater than 65535 are truncated.

- **TimOff**

Off-Time in milliseconds. Values greater than 65535 are truncated.

Binary Select Before Operate Tables

The Binary Select Before Operate tables are used to mark the Binary Output points that have been selected for operating by a select request. Each table represents up to 1000 points. If **nSbo** is greater that 1000, append more tables.

The table should be defined as multi-column. Its structure is mandatory and is dependent on the Binary Select Before Operate implementation, as fixed by the **sboTyp** parameter. Its structure is based on the I/O tables structure (See section *I/O Tables* on page 33). Also, it contains columns couples. Each column couple represents up to 250 points. If **sboTyp** is greater than 0, all the columns are bit columns. If **sboTyp** is 0, the first column of each couple is a Discrete Output column, and the second is a Discrete Input. If **nSbo** is less than 250, define two columns and specify the exact number of rows. For example, if **nSbo** is 50, define two columns (as described above) and specify 49 as last index. If **nSbo** is greater than 250, you must define multiples of 250 entries. For example, if **nSbo** is 260 points, define 2 couples and specify 249 as last index. The first 250 Binary Output point are represented by the first couple, the next 250 Binary Output points are represented by the next couple, and so on. If **nSbo** is greater than 1000, then additional tables are required (append the tables). Each such table represents up to 1000 points and should have an even number of columns.

The following figure illustrates the table.

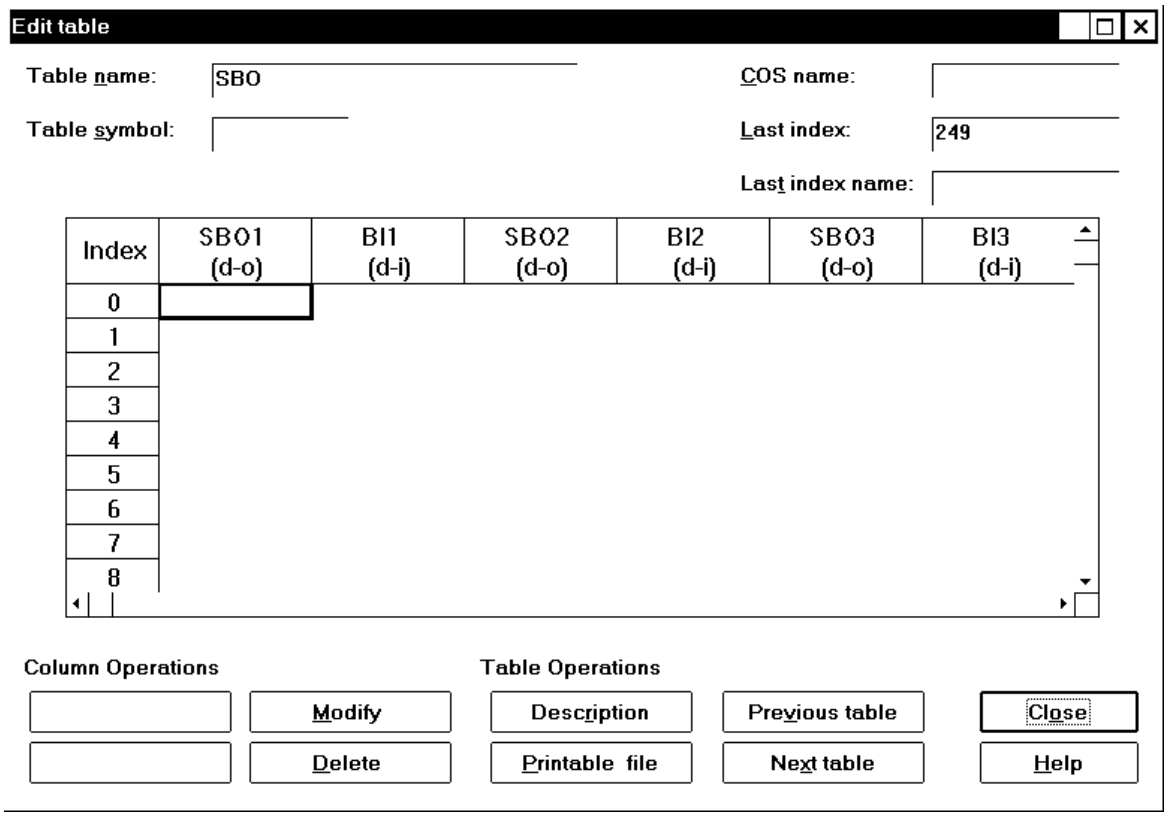

Figure 11: Binary Select Before Operate table

If **sboTyp** is 0, the driver implements the selection by closing output relays. Therefore, for each Binary Output, you need to connect the two entries that represent it, one in a Discrete Output column and the other in a Discrete Input column, to the same Discrete Output I/O. The first connection is the relay itself and the second connection is its BI (back indication). You may disable the selection of a certain point by setting its disable field.

If **sboTyp** is 1, it is not applicable since the user application is responsible for setting the BI columns.

When the driver receives a new request with a Select function Object 12 Variation 1, it does the following:

- If the appropriate field is ?? in the DO Status table, return failure in the appropriate status bytes and do not mark the selection. See section *Discrete Output (DO) Status Tables* page 32.
- Otherwise, set the appropriate entries in the Discrete Output columns.

After marking all the points to be selected:

If **sboTyp** is 0,

- Scan out (activate) the data of all the Discrete Output columns that have been modified.
- Wait a bit, until the physical BIs are updated.
- Scan in (read BI) all the parallel Discrete Input columns.

If **sboTyp** is greater than 0,

- Increase the nSReq in the Statistics table to mark that a new successful Select Before Operate request has been received.
- Wait the time that is determined according the **sboTyp** parameter. During this time, the user application should set/clear the appropriate bits in the BI columns.

- Check that the appropriate entries in the Discrete Input columns are set. If they are, return success in all the status bytes. Otherwise, return failure in the appropriate status bytes and clear the selection.

To use the Binary Select Before Operate tables, define the index of the first table via the permanent SboTab parameter and the number of points to select via the permanent nSbo parameter.

Discrete Output (DO) Status Tables

The DO Status tables are used to enable the user to avoid the Select, Direct Operate and Operate commands on DO points. Any value other than 0 represents the error that should be returned whenever a Select, a Direct Operate or an Operate request is received for a DO point.

The tables should be defined as multi-column and their structure is mandatory. The structure is based on the I/O tables structure (see below). Their size is determined according to the DO tables, but their columns are of Integer type. Each table represents up to 2000 points.

When the driver receives a new request with a Select, a Direct Operate or an Operate function Object 12 Variation 1, it does the following:

- If the appropriate field in the DO Status table is not 0, return failure in the status line.
- Otherwise, perform the request as expected.

The following figure illustrates the table.

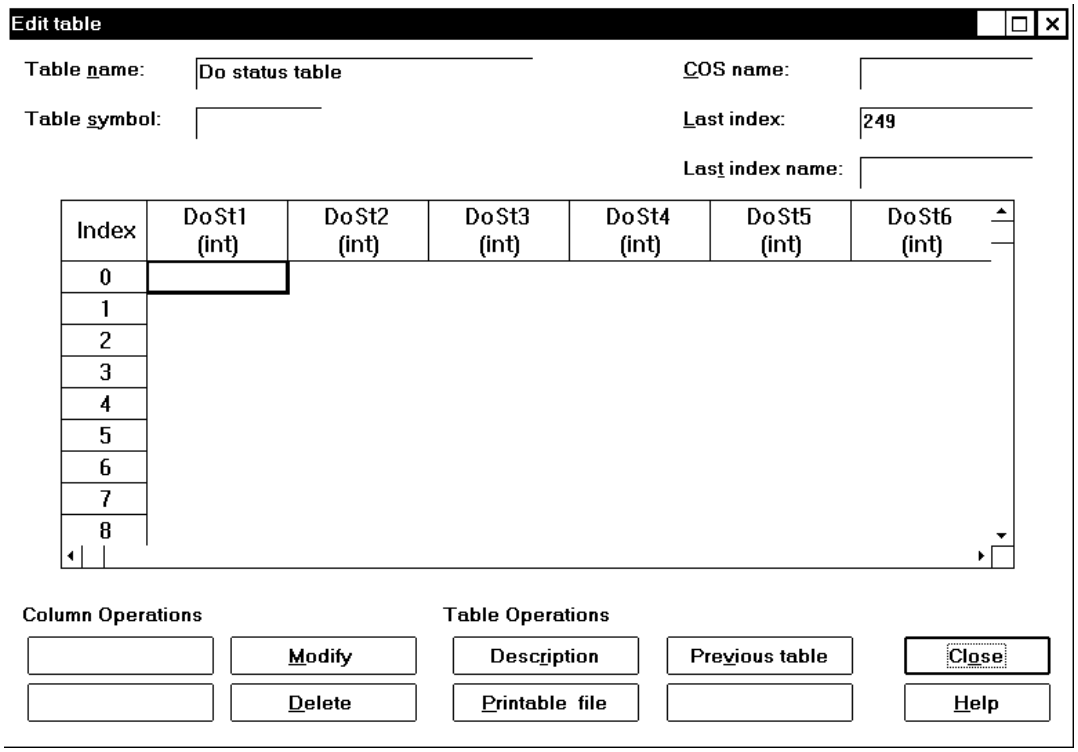

Figure 12 Discrete Output Status Table

/O Tables

You must define at least one I/O table for each data type you use for DNP. If you need to define less than 250 entries, specify the exact number of rows. For example, if you need 50 entries, specify 49 as last index. If you need more than 250 entries, you must define multiples of 250 entries. For example, if you need 260 entries, define two columns with last index of 249.

The data type of the columns are determined according the I/O data type, as follows:

- For Binary Inputs and Binary Outputs tables, define Discrete (bit) columns (in fact, you can use any “bit” data type supported in the Database Builder and mix different bit types if this better serves your purposes).
- For Analog Inputs, Analog Outputs, Counters and Freeze Counters tables, define Integer Value (int) columns (or any other data type with the same size or integer value — 2 bytes).
- For Floating-Point tables, define Real Value (real) columns (or any other data type with the same size — 4 bytes).

Classes Tables

Classes are groups of points. Classes allow you to refer to a group of points globally.

Four classes are supported: Class 0 represents the static points (the entire database) and Classes 1 through 3 are used for events.

The Classes table is where you map the points to the classes. The columns correspond to the variables and the values in the rows determine the class affiliation of the points.

Class tables should be defined for the following data types:

- Binary Inputs
- Analog Inputs
- Counters

Each type requires its own class tables, the size of which should be identical to the corresponding I/O table. Their columns should be defined as Integer Parameter (iprm) columns. Each point value in the class table, according its order. The values that associate a point to a class are listed in the chart below. See also Figure 20.

Value	Meaning
2	Class 1
4	Class 2
8	Class 3

Freeze Counter Tables

If you use counters, define Counter tables, set nCNTR to a value greater than 0, and define appropriate Freeze Counter tables. The Freeze Counter tables should have the same size as (or less than) the Counter tables. Each point in the Freeze Counter tables stores the frozen value of the corresponding (same order) counter.

The index of the first tables of each data type should be specified in the Permanent Configuration Parameters table.

The following figures illustrate the required I/O tables.

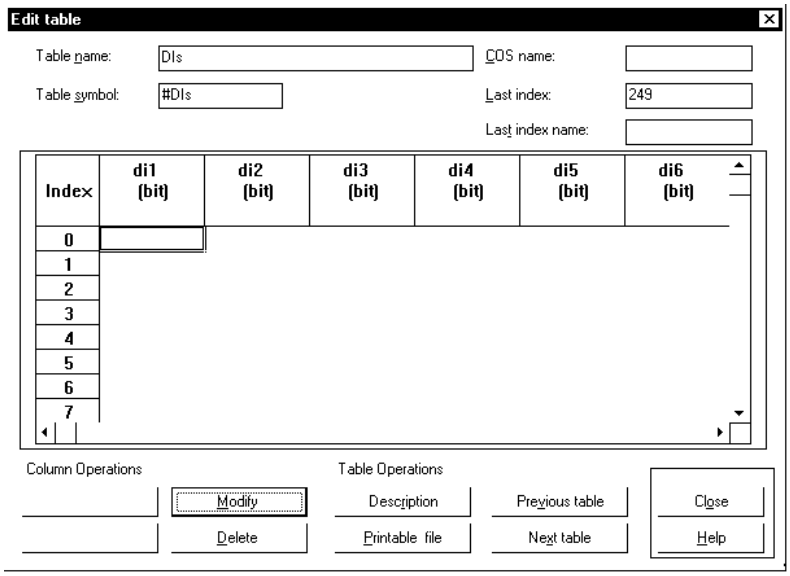

Figure 13: Binary Inputs table

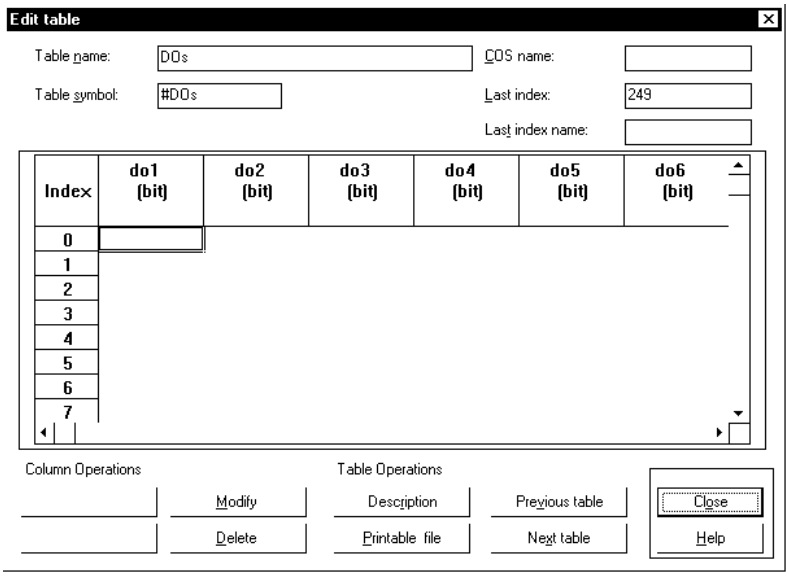

Figure 14: Binary Outputs table

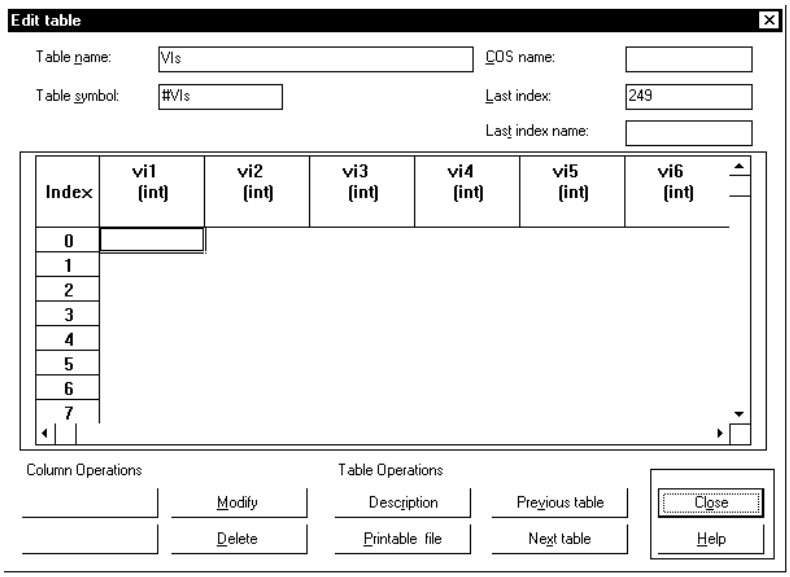

Figure 15: Analog Inputs table

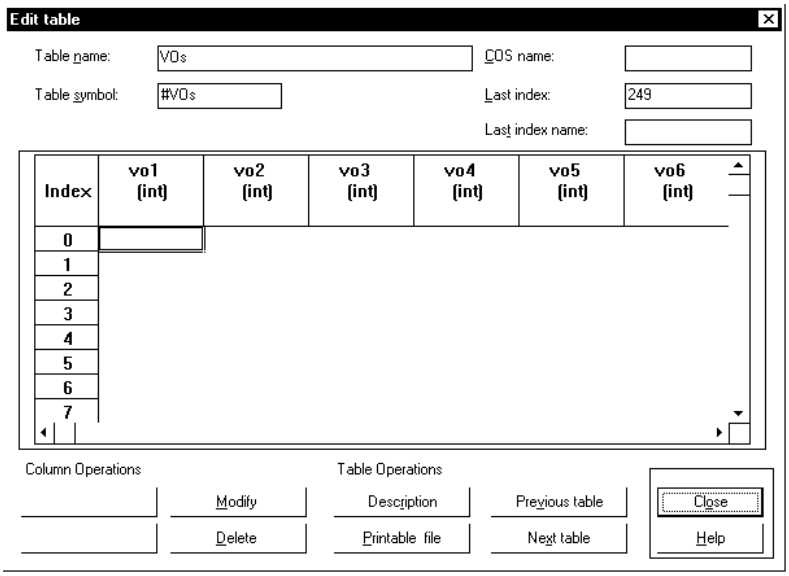

Figure 16: Analog Outputs table

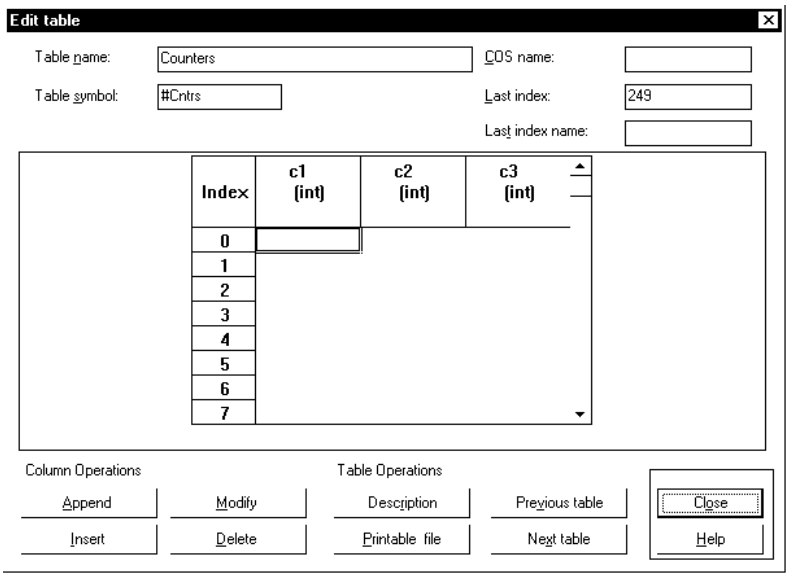

Figure 17:Counters table

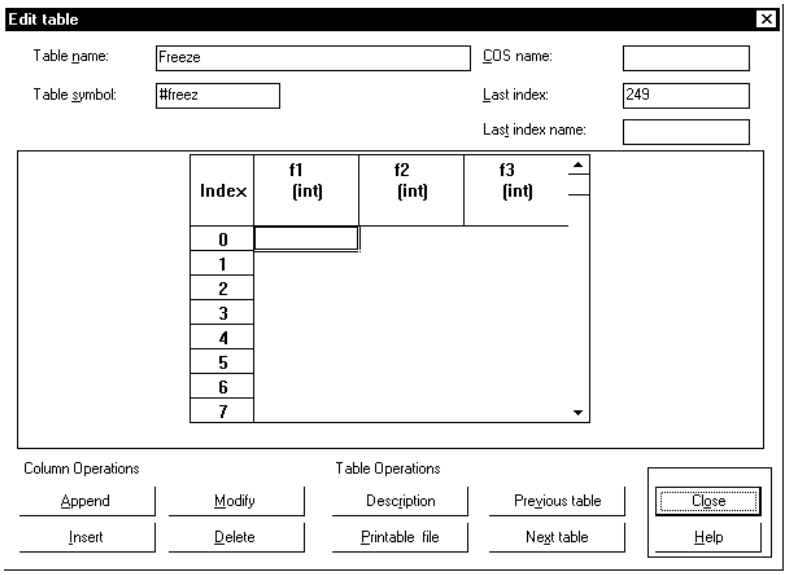

Figure 18: Freeze Counters table

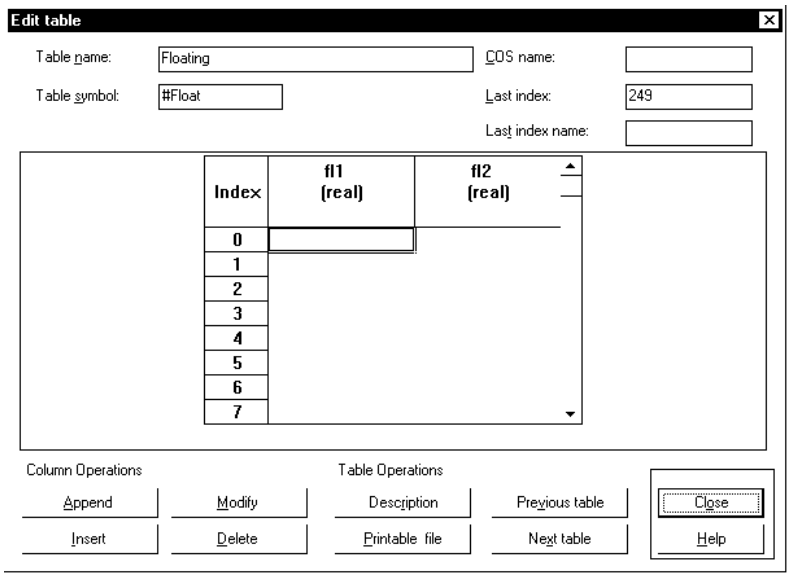

Figure 19: Floating-point table

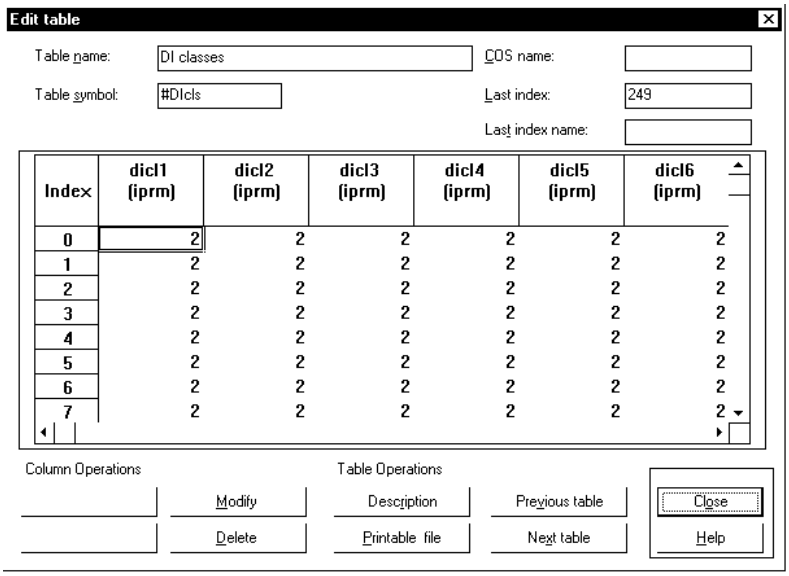

Figure 20: Classes table

The Ladder Process

This section describes the UCALLs that should be used to run the DNP driver in a MOSCAD application.

NOTE

All the UCALL functions that use a table number as parameter, expect a *constant* integer (it is recommended to use the value that appears in the Table Symbol field). A variable integer will cause unpredictable results.

DSstrt/DS2str UCALL

The DNP driver is launched by calling the DSstrt/DS2str UCALL.

This UCALL must be called from the **Main Process**, in each ladder scan. Actually, it should be called only after power-up or after the 'C' block has been downloaded. However, since there is no indication whether the DNP driver 'C' block has been downloaded, call the function in each scan; an internal mechanism decides whether or not to run the DNP driver task.

This function takes 2 constant parameters:

1. Number of the permanent DNP configuration parameters table. The table structure is described in *Permanent Configuration Parameters Table* on page 11.

2. Number of the variable DNP configuration parameters table. The table structure is described in section *Variable Configuration Parameters Table* on page 11.

DSparm/DS2prm UCALL

Modifies the variable DNP configuration parameters.

This function expects a single constant parameter, the number of the variable DNP configuration parameters table. See *Variable Configuration Parameters Table* on page 11.

This UCALL may be called a few times. It is, however, recommended to use the variable parameters that were set when the task was run, and to avoid this UCALL.

DSunsl/DS2uns UCALL

Stores the information on an event and sends an unsolicited response, if the driver is configured to work in unsolicited mode.

This function expects a single constant parameter, a table number containing the information for *one* event. The table structure is explained below. It is possible to use different tables for this UCALL. However, using the same table saves time. This event information is stored in the driver's events buffers (buffers are defined in the Permanent Parameters table) and will be sent as a response to a poll request. If the permanent UnsolM parameter is set to 1, this information will be sent as an unsolicited response if all the conditions to send is were

fulfilled before any poll request was received. The conditions to send unsolicited request are defined by the variable parameters **UnsQn1**, **UnsQn2**, **UnsQn3** and **UnsDly**.

You must call this function from a task with higher priority than the driver priority

The Unsolicited Response Table: This table contains the information required for generating an event.

The table should be defined as single-column of Integer Parameter type and must contain at least 21 rows. The following figure illustrates an Unsolicited Response table.

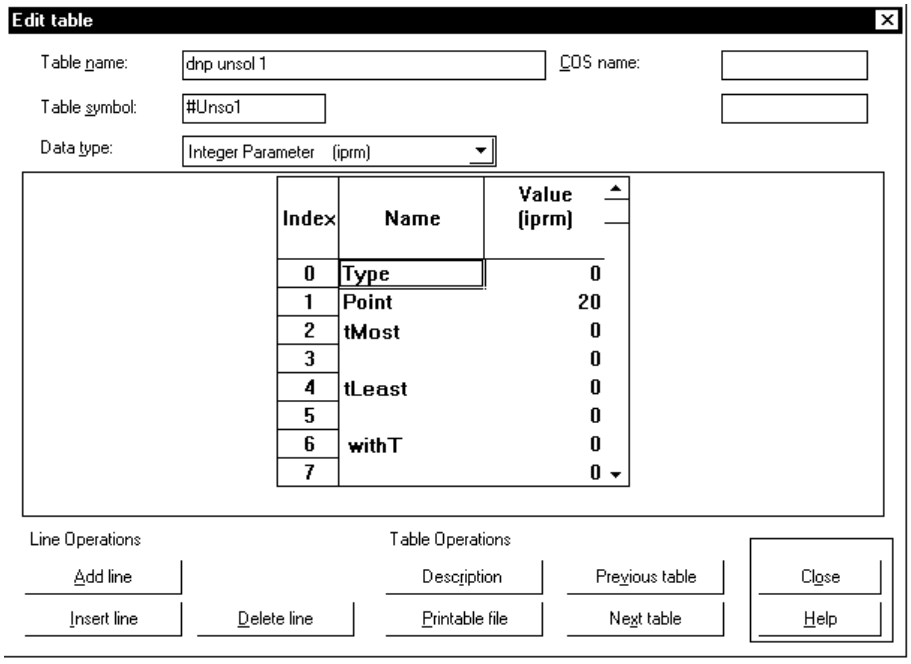

Figure 21: Unsolicited Response table

The Unsolicited Response parameters are as follows:

- **Type**

Type of the reported point.

The following chart lists the valid values and their meanings.

Value	Meaning
0	Binary Input
1	Analog Input
2	Counter

- **Point**

Point number. The value stored in this point will be taken from the relevant table and will be sent as unsolicited response.

The following two parameters take up 8 bytes that represent the time stamp according to the TmMost and TmLeast structures. TmMost and TmLeast are fields in the PRMEVENT table (in the system tables). They are built as follows:

- **tMost**

Day	Month	Year	Hour
0..31	1..12	0..99	0..23

- **tLeast**

Minutes	Seconds	Milliseconds
0..59	0..59	0..999

- **withT**

With time stamp indication. This parameter determines if the unsolicited message will be sent with time stamp or not.

The following chart lists the valid values and their meanings.

Value	Meaning
0	Send without time stamp
1	Send with time stamp

NOTE

The recommended way to fill in the tMost and tLeast parameters is as follows:

- Call the Time function. It will fill in the current time in the TmMost and TmLeast fields of the PRMEVENT table, according to the above structure.
- Call the CPY function to copy four bytes from TmMost to tMost.
- Call the CPY function to copy four bytes from TmLeast to tLeast.

DSclr/DS2clr UCALL

Clears the Statistics table.

This function has no parameters.

DSdefv/DS2dfv UCALL

Sets the default variation per object, to which the driver should respond on a request with variation 0.

This function takes 2 constant parameters:

1. The object.
2. The default variation to be set for the object passed in the first parameter. This variation should be supported. See section

Supported Objects, Variations and Qualifiers in page 45.

UCALLs Return Codes

The UCALLs return codes are applicable to: DSunsl/DS2uns.

The following chart lists the possible return codes:

Code	Meaning
0	No error
1	Problem in getting table information
2	Events buffer was not allocated for this data type, or no point of this data type was defined.
3	Events buffer overflow
4	Point is out of range
5	Unknown data type
6	Events queue busy
7	Object variation not supported

NOTE

If code 3 is returned, no new events will be inserted into the events buffer until old ones are cleared.

Supported Functions

This section lists the supported DNP functions.

These are the supported request function codes:

1 - Read
2 - Write
3 - Select. The driver is limited to only one frame in a select request.
4 - Operate
5 - Direct Operate
6 - Direct Operate, No Acknowledgment

7 - Immediate Freeze
8 - Immediate Freeze, No Acknowledgment
9 - Freeze and Clear
10 - Freeze and Clear, No Acknowledgment
13 - Cold Restart (Restart the RTU including the DNP driver)
14 - Warm Restart (Restart only the DNP driver)
20 - Enable Unsolicited Messages
21 - Disable Unsolicited Messages
23 - Delay Measurement

The following are the supported response function codes:

129 - Response
130 - Unsolicited Message

Supported Objects, Variations and Qualifiers

The following table lists all the objects, function codes and qualifiers supported by MOSCAD.

Important: * means “Parse only”.

OBJECT			REQUEST (Master->Slave)		RESPONSE (Slave->Master)	
Obj	**Var**	**Description**	**Func Code (dec)**	**Qual Code (hex)**	**Func Code (dec)**	**Qual Code (hex)**
1	0	Binary Input - All Variations	1	00,01,06		
1	1	Binary Input	1	00,01, 06	129	00,01
1	2	Binary Input with Status	1	00,01, 06	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 with 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	00,01, 06		
10	1	Binary Output	1	00,01, 06	129	00,01
10	2	Binary Output Status	1	00,01, 06	129	00,01
12	0	Control Block - All Variations				
12	1	Control Relay Output Block	3,4,5,6	17,28	129	echo of req
12	2*	Pattern Control Block	5,6	17,28		
12	3*	Pattern Mask	5,6	00,01		
20	0	Binary Counter - All Variations	1,7,8,9, 10	00,01,06		
20	1*	32-Bit Binary Counter	1	00,01, 06		
20	2*	16-Bit Binary Counter	1	00,01, 06		
20	3*	32-Bit Delta Counter	1	00,01, 06		

OBJECT			REQUEST (Master->Slave)		RESPONSE (Slave->Master)	
Obj	**Var**	**Description**	**Func Code (dec)**	**Qual Code (hex)**	**Func Code (dec)**	**Qual Code (hex)**
20	4*	16-Bit Delta Counter	1	00,01, 06		
20	5*	32-Bit Binary Counter without Flag	1	00,01, 06		
20	6	16-Bit Binary Counter without Flag	1	00,01, 06	129	00,01
20	7*	32-Bit Delta Counter without Flag	1	00,01, 06		
20	8*	16-Bit Delta Counter without Flag	1	00,01, 06		
21	0	Frozen Counters - All Variations	1	00,01,06		
21	1*	32-Bit Frozen Counter	1	00,01, 06		
21	2*	16-Bit Frozen Counter	1	00,01, 06		
21	3*	32-Bit Frozen Delta Counter	1	00,01, 06		
21	4*	16-Bit Frozen Delta Counter	1	00,01, 06		
21	9*	32-Bit Frozen Counter without Flag	1	00,01, 06		
21	10	16-Bit Frozen Counter without Flag	1	00,01, 06	129	00,01
22	0	Counter Change Event - All Variations	1	06,07, 08		
22	1*	32-Bit Counter Change Event without Time	1	06,07, 08		
22	2	16-Bit Counter Change Event without Time	1	06,07, 08	129, 130	17,28
22	3*	32-Bit Delta Counter Change Event without Time	1	06,07, 08		
22	4*	16-Bit Delta Counter Change Event without Time	1	06,07, 08		
22	6	16-Bit Counter Change Event with Time	1	06,07, 08	129, 130	17,28
23	0*	Frozen Counter Event - All Variations	1	06,07, 08		
23	1*	32-Bit Frozen Counter Event without Time	1	06,07, 08		
23	2*	16-Bit Frozen Counter Event without Time	1	06,07, 08		
23	3*	32-Bit Frozen Delta Counter Event	1	06,07,		

OBJECT			REQUEST (Master->Slave)		RESPONSE (Slave->Master)	
Obj	**Var**	**Description**	**Func Code (dec)**	**Qual Code (hex)**	**Func Code (dec)**	**Qual Code (hex)**
		without Time		08		
23	4*	16-Bit Frozen Delta Counter Event without Time	1	06,07, 08		
30	0	Analog Input - All Variations	1	00,01, 06		
30	1*	32-Bit Analog Input	1	00,01, 06		
30	2	16-Bit Analog Input	1	00,01, 06	129	00,01
30	3*	32-Bit Analog Input without flag	1	00,01, 06		
30	4	16-Bit Analog Input without flag	1	00,01, 06	129	00,01
32	0	Analog Change Event - All Variations	1	06,07, 08		
32	1*	32-Bit Analog Change Event without Time	1	06,07, 08		
32	2	16-Bit Analog Change Event without Time	1	06,07, 08	129, 130	17,28
32	4	16-Bit Analog Change Event with Time	1	06,07, 08	129, 130	17,28
40	0	Analog Output Status - All Variations	1	00,01, 06		
40	1*	32-Bit Analog Output Status	1	00,01, 06		
40	2	16-Bit Analog Output Status	1	00,01, 06	129	00,01
41	1*	32-Bit Analog Output Block	3,4,5,6	17,28		
41	2	16-Bit Analog Output Block	3,4,5,6	17,28	129	echo of req
50	0	Time and Date - All Variations	1	07		
50	1	Time and Date	2	07		
			1	07	129	07
52	2**	Time Delay Fine			129	07

** Applicable only as a response to a Delay Measurement request

OBJECT			REQUEST (Master->Slave)		RESPONSE (Slave->Master)	
Obj	**Var**	**Description**	**Func Code (dec)**	**Qual Code (hex)**	**Func Code (dec)**	**Qual Code (hex)**
60	1	Class 0 Data	1	06		
60	2	Class 1 Data	1	06,07, 08		
			20,21	06		
60	3	Class 2 Data	1	06,07, 08		
			20,21	06		
60	4	Class 3 Data	1	06,07, 08		
			20,21	06		
80	1*	Internal Indication	1	00,01		
			2	00,01		
100	1	Short Floating Point	1	00,01, 06		
No Object			13			
No Object			23			

Some of the object/variations have a status byte which indicates the status of the point. Some other object/variations have a flag byte which indicates the status of the point, as well. These bytes are different in each case. Currently, we always return On-Line (0x01).