



# **ToolLink DeviceNet Gateway**

## **User's Manual**

**TABLE OF CONTENTS**

**CHAPTER 1 – OVERVIEW.....4**

    HARDWARE ..... 4

    CONFIGURATION ..... 6

    DEVICENET INTERFACE..... 6

    SERIAL INTERFACE ..... 6

**CHAPTER 2 – QUICK START GUIDE .....8**

    HARDWARE SETUP ..... 8

    SYSTEM CONFIGURATION..... 8

    CONFIGURING THE GATEWAY ..... 8

    TRANSFERRING DATA ..... 9

**CHAPTER 3 – THEORY OF OPERATION..... 10**

    DEVICENET INTERFACE..... 10

    SERIAL INTERFACE ..... 11

**CHAPTER 4 – GATEWAY CONFIGURATION ..... 13**

    CONFIGURE DEVICENET INTERFACE ..... 13

*DeviceNet Baud Rate Switch..... 13*

*MAC ID Switches..... 13*

    POWER UP GATEWAY..... 13

*DeviceNet Status LEDs ..... 13*

*Serial Channel Status LEDs..... 14*

*Register EDS File ..... 14*

**CHAPTER 5 – DEVICENET PROFILE..... 15**

*DeviceNet Message Types..... 15*

    DEVICENET OBJECT CLASSES ..... 15

    IDENTITY OBJECT CLASS CODE: 01 (0X01)..... 15

*Revision – Attribute 4..... 16*

*Device Status – Attribute 5..... 16*

*Serial Number – Attribute 6..... 16*

    ROUTER OBJECT CLASS CODE: 02 (0X02)..... 17

    DEVICENET OBJECT CLASS CODE: 03 (0X03) ..... 17

*MACID – Attribute 1..... 18*

*Baud Rate – Attribute 2..... 18*

*Allocation Information – Attribute 5..... 18*

    ASSEMBLY OBJECT CLASS CODE: 04 (0X04)..... 18

    CONNECTION OBJECT CLASS CODE: 05 (0X05) ..... 19

*State – Attribute 1 ..... 20*

*Connection ID’s – Attributes 4 and 5 ..... 20*

*Production and Consumed Sizes – Attributes 7 and 8 ..... 21*

*Watch Dog Timeout Activity – Attribute 12 ..... 21*

    USER DEFINED (SERIAL STREAM) OBJECT CLASS CODE: 100 (0X64)..... 21

*Receive Data – Attribute 3 ..... 22*

*Transmit Data – Attribute 4 ..... 22*

*Baud Rate – Attribute 6..... 23*

*Parity – Attribute 7 ..... 23*

*Flow Control – Attribute 8..... 23*

*Receive Mode – Attribute 9..... 23*

*Time-Out:..... 24*

*Start/Stop Delimiters: ..... 24*

*Stop Delimiter:..... 24*

<i>Start Delimiter:</i> .....	24
<i>No Delimiters:</i> .....	24
<i>Start Delimiter String – Attribute 11</i> .....	25
<i>Stop Delimiter String – Attribute 12</i> .....	25
<i>Rx Handshake Enable – Attribute 13</i> .....	25
<i>TX Handshake Enable – Attribute 16</i> .....	26
<i>Maximum Rx Size – Attribute 19</i> .....	26
<i>Maximum TX Size – Attribute 20</i> .....	26
<i>Byte Swapping – Attribute 21</i> .....	26
<b>APPENDIX A – PRODUCT SPECIFICATIONS</b> .....	<b>28</b>
DEVICE <sup>N</sup> ET INTERFACE .....	28
SERIAL CHANNEL .....	28
ENVIRONMENTAL .....	28
<b>APPENDIX B – ASCII CHARACTER CODES</b> .....	<b>29</b>

# Chapter 1 – Overview

This document describes how to install, configure, and operate the CDN466 series of serial to DeviceNet gateways. The following products are covered in this user manual:

Part Number	FW Rev.	Serial Channel
CDN466	1.01 or higher	RS232 full duplex

The CDN466 gateways allow you to easily interface a wide variety of serial devices to any DeviceNet industrial control network. Standard CDN466 products are tightly packaged and sealed in a rugged industrial case. Board-level and customized gateways are also available upon request.

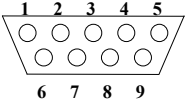
## Hardware

**Receive Status LED (RX)**

STATE	DESCRIPTION
OFF	Not receiving data
RED BLINK	Not defined
RED	Receive error
GREEN BLINK	Receiving data
GREEN	Not defined

**Transmit Status LED (TX)**

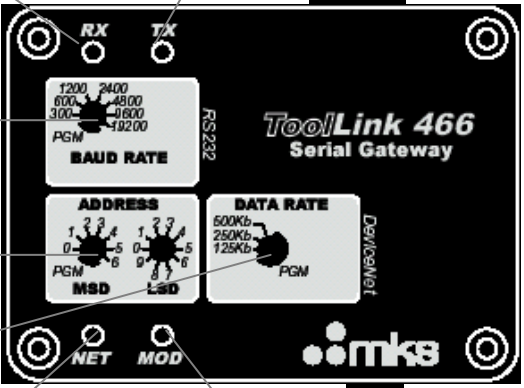
STATE	DESCRIPTION
OFF	Not transmitting data
RED BLINK	Not defined
RED	Transmit error
GREEN BLINK	Transmitting data
GREEN	Not defined



**Isolated Serial Channel  
(male DB9 connector)**

PIN	CDN466
1	nc
2	RXD
3	TXD
4	DTR/DSR*
5	SGND
6	DTR/DSR*
7	RTS
8	CTS
9	nc

\*Pins 4 and 6 connected internally.



**Serial Baud Rate Rotary Switch**

**DeviceNet Address Rotary Switches**

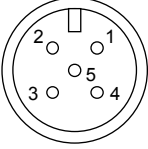
**DeviceNet Data Rate Rotary Switch**

**DeviceNet Status LED (NET)**

STATE	DESCRIPTION
OFF	No power
RED BLINK	Configuration error
RED	Unrecoverable error
GREEN BLINK	Not allocated to a master
GREEN	Allocated to a master

**Module Status LED (MOD)**

STATE	DESCRIPTION
OFF	No power
RED BLINK	Configuration error
RED	Unrecoverable error
GREEN BLINK	Not defined
GREEN	Normal operation



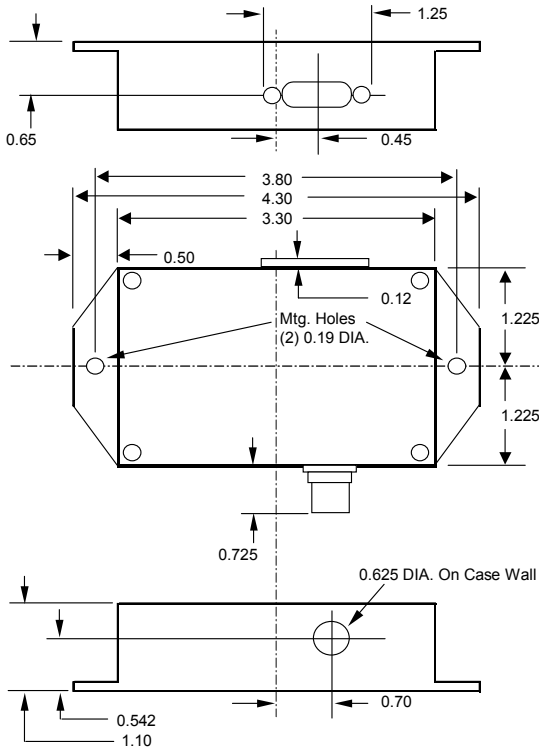
**DeviceNet Channel  
(male 5-pin micro connector)**

PIN	SIGNAL
1	SHIELD
2	V+
3	V-
4	CAN H
5	CAN L

**INSTALLATION**

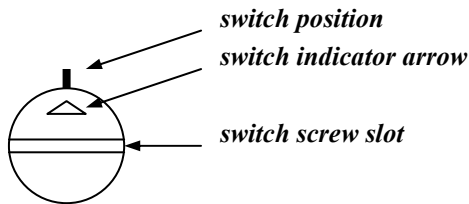
Mount the ToolLink Gateway on a horizontal or vertical surface, in a suitable location or enclosure for your application. Provide sufficient clearance and airflow to maintain 0°C to 70°C ambient operating temperature range. Fasten the ToolLink Gateway to the mounting surface using two screws (not provided) in the 0.19 inch mounting holes.

*All dimensions are inches*



**ROTARY SWITCHES**

Set the ToolLink rotary switches to the desired settings. Use a small slotted screwdriver to rotate the switches. Align the indicator arrow to the desired setting, as shown below.



Each rotary switch parameter has a **PGM** option. Setting a switch to PGM allows the parameter to be remotely set over DeviceNet. However, it must first be initialized. To initialize, set the switch to desired value and power up the gateway. The new settings are saved in its memory. Power down and change switch to PGM mode.

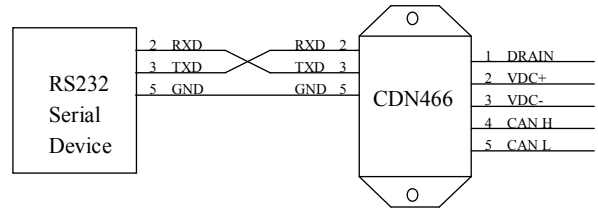
**WIRING**

The ToolLink Gateway requires two connections – one to the DeviceNet network (male 5-pin micro connector) and one to the serial device (male DB9 connector). DeviceNet and serial cables are available from a variety of industrial sources. Follow all applicable electrical codes in your area when mounting and wiring any electrical device.

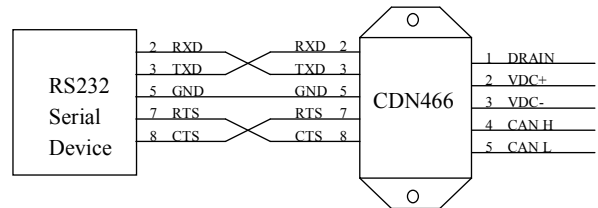
All power is received from the DeviceNet network. The ToolLink Gateway draws up to 200mA from the 24VDC power supply. Select your DeviceNet cables and power supply so that it can provide sufficient current for all networked devices at their peak operating power.

The following are typical ToolLink Gateway wiring examples. Your RS232 or RS485 interface may vary. Refer to your device's documentation for the required data and control signals.

RS232 Interface



RS232 Interface, HW Flow Control



**CONFIGURATION**

Rotary switches and software parameters configure the ToolLink Gateway's *DeviceNet Interface*, *Serial Interface*, *Serial Receive*, and *Serial Synchronization* functions. The ToolLink Gateway can be configured over its DeviceNet channel. Use your DeviceNet Configuration application program and the ToolLink EDS file to set the software parameters over the DeviceNet channel.

FUNCTION	PARAMETER	TYPE	VALUE	DESCRIPTION
DeviceNet Interface	Address	Switch	00 TO 63	Sets DeviceNet node address. MSD switch sets the most significant digit (0x to 6x). LSD switch sets the least significant digit (x0 to x9).
	Data Rate	Switch	0 = 125Kbps 1 = 250kbps 2 = 500Kbps	Sets DeviceNet data rate.
	Maximum Receive Size	Software	0 to 64 bytes	Defines the maximum receive message packet size. The total number of ToolLink input bytes is Maximum Receive Size + 4.
	Maximum Transmit Size	Software	0 to 64 bytes	Defines the maximum transmit message packet size. The total number of ToolLink output bytes is Maximum Transmit Size + 4.
	Byte-Swap Enable	Software	0 = disabled 1 = enabled	Defines how ToolLink formats its input and output data fields. When enabled, ToolLink swaps every 2 bytes in the data field.
Serial Interface	Baud Rate	Switch	0 = 19200 4 = 4800 1 = 600 5 = 9600 2 = 1200 6 = 19200 3 = 2400	Sets the serial channel baud rate.
	Parity	Software	0 = No parity 1 = Even parity 2 = Odd parity	Sets the serial channel parity mode. Received byte is tested for errors, and then parity bit is cleared before the byte is saved in RX buffer.
	Flow Control	Software	0 = None 1 = CTS/RTS	Sets the serial channel flow control.  CTS/RTS is an RS232 hardware flow control option. Gateway keeps RTS output active (low) when it can receive data. Gateway only transmits data when CTS input is active (low)
Serial Receive	Receive Mode	Software	0 = Timeout 1 = Length 2 = Delimiter	Selects how the gateway receives a complete message packet.
	Start Delimiter String	Software	String of 0-4 bytes: [Length][B1][B2][B3][B4]	Used when Receive Mode = Start/Stop Delimiter. Defines the start of a received message packet.
	Stop Delimiter String	Software	String of 0 to 4 bytes: [Length][B1][B2][B3][B4]	Used when Received Mode = Start/Stop Delimiter. Defines the end of a received message packet.
Serial Synchronization	RX Handshake Enable	Software	0 = disabled 1 = enabled	Optional receive serial message handshake protocol between ToolLink Gateway and application program.
	TX Handshake Enable	Software	0 = disabled 1 = enabled	Optional transmit serial message handshake protocol between ToolLink Gateway and application program.

**DeviceNet Interface**

The ToolLink Gateway can receive serial message packets up to 68 bytes long. The DeviceNet Output Size (Produce Size) is equal to the **Maximum Receive Size + 4 bytes of overhead**. The **Maximum Receive Size** parameter defines the Data Field size (M) for the input bytes.

**ToolLink DeviceNet Input Bytes**

STATUS	RXCTR	TXACK	LENGTH	DATA FIELD
1 byte	1 byte	1 byte	1 byte	M bytes

The ToolLink Gateway can transmit serial message packets up to 68 bytes long. The DeviceNet Input Size (Consume Size) is equal to the **Maximum Transmit Size + 4 bytes of overhead**. The **Maximum Transmit Size** parameter defines the Data Field size (N) for the output bytes.

**ToolLink DeviceNet Output Bytes**

COMMAND	RXACK	TXCTR	LENGTH	DATA FIELD
1 byte	1 byte	1 byte	1 byte	N bytes

**Serial Interface**

The **Receive Mode** parameter defines how the ToolLink Gateway receives serial message packets. The three supported modes include Timeout mode, Length mode and Delimiter mode.

When in Timeout mode, the ToolLink Gateway waits for an inter-byte delay to signal the end of a message packet. If the receiver is idle for more than 3.5 byte times (or 5 msec, whichever value is greater), then all bytes received before the timeout are grouped into a single message packet. 1 byte time =  $10 \text{ bits} \div \text{baud rate}$ .

When in Length mode, the ToolLink Gateway receives a fixed number of bytes as a complete message packet. The **Maximum Receive Size** parameter defines the message packet size (0 to 64 bytes) for the LENGTH mode.

When in delimiter mode, the ToolLink Gateway uses start and stop delimiter strings to identify the beginning and end of a message packet. The **Start Delimiter String** parameter defines the beginning of a message, and the **Stop Delimiter String** parameter defines the end of a message.

## Chapter 2 – Quick Start Guide

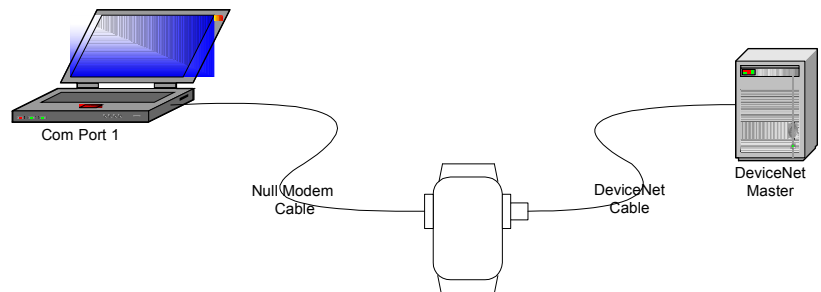
This chapter describes the setup for a simple gateway solution using a DeviceNet master and the serial port of a PC using HyperTerminal. Before beginning a basic understanding of DeviceNet and rs232 is required. Experience using explicit and poll transactions from the software provided with your DeviceNet master is essential. For more information on generating explicit and poll messages consult the DeviceNet master's software user's guide.

### Hardware Setup

Setup a gateway connection between a device net master and the serial port of a PC.

Required Hardware:

- Null modem cable
- Device net cable
- Device net master
- PC with HyperTerminal
- Serial gateway

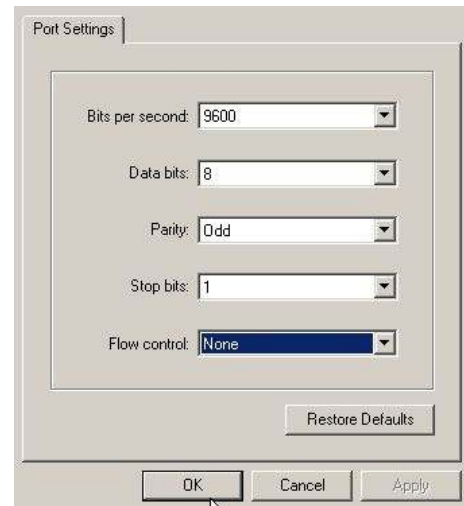


CDN466/CDN467

### System Configuration

Configure the HyperTerminal properties in the File Menu of HyperTerminal with the following parameters.

- 9600 baud
- 1 stop bit
- 8 bits of data
- Odd parity
- No flow control



### Configuring the Gateway

The major steps for configuring the gateway include setting up the Serial Stream Object, the Serial Receive Object, and the Serial Transmit Object.

Configure the gateway switches as follows:

- MACID MSD to 0
- MACID LSD to 1
- DeviceNet Data rate to 500K
- RS2323 baud rate to 19200

Once all of the hardware is setup and powered up, make sure that the master can allocate both poll and explicit connections to the gateway. Once allocated, both the net and mod LED will be solid green.



Using the DeviceNet master's software loads the parameters for the serial stream object, the serial receive object, and the serial transmit object in the tables below for the gateway through the device net connection by using explicit messaging.

**Table 1 Configure the serial stream object class 100 (0x64)**

Attribute	Access	Name	Value
6	Get	Baud Rate	9600
7	Get/Set	Parity	None
8	Get/Set	Flow Control	None
9	Get/Set	Receive Mode	Timeout
13	Get/Set	RX Handshake Enable	0 = No
16	Get/Set	TX Handshake Enable	0 = No
19	Get/Set	Max Receive Size	1
20	Get/Set	Max Transmit Size	1

**Transferring data**

The ToolLink Gateway is now set up the receive data. The Output Size (Produce Size) will be equal to 5. The 5 Byte poll response will is described below

**ToolLink DeviceNet Input Bytes**

STATUS	RXCTR	TXACK	LENGTH	DATA FIELD
1 byte	1 byte	1 byte	1 byte	1 byte

The ToolLink Gateway is now set up the Transmit data. The Input Size (Consume Size) will be equal to 5. The 5 Byte poll will is described below.

**ToolLink DeviceNet Output Bytes**

COMMAND	RXACK	TXCTR	LENGTH	DATA FIELD
1 byte	1 byte	1 byte	1 byte	1 byte

## Chapter 3 – Theory of Operation

This chapter describes how the CDN466 gateway operates. You should have a working knowledge of DeviceNet and asynchronous serial communications before continuing. The Open DeviceNet Vendors Association ([www.odva.com](http://www.odva.com)) is a good source for general DeviceNet information. Refer to your serial device documentation for its protocol information.

### DeviceNet Interface

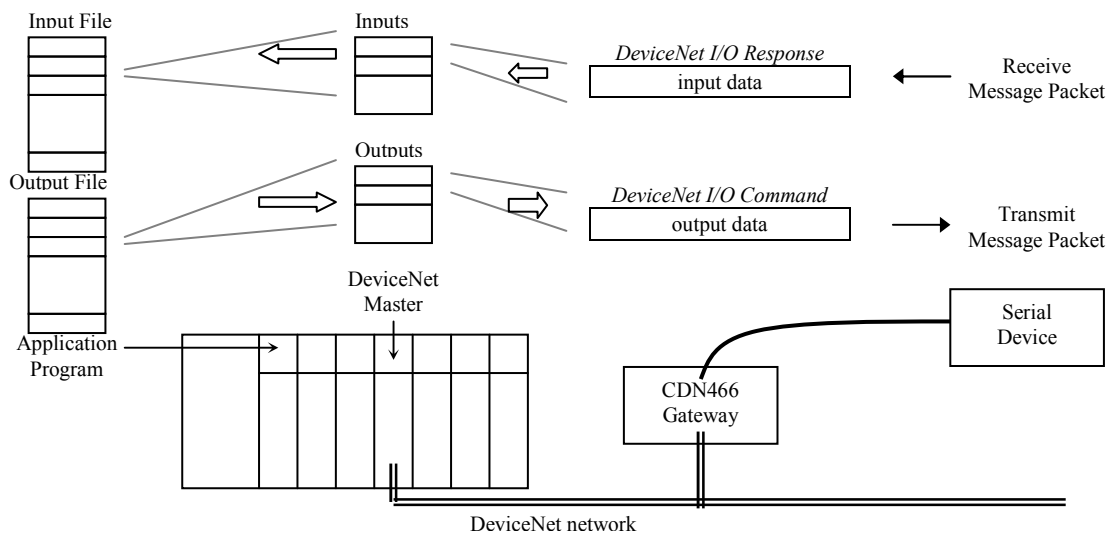
The DeviceNet Specification defines an Object Model that consists of Objects and Attributes. An Object is a predefined software process, and an Object Attribute is a data value used or created by that process. An Object can have multiple Instances, or the same process operating with different sets of Attributes or data values. For the purpose of this document, an Object Instance is an independent program or process, and its Attributes are configuration parameters and data values that are unique to that specific Object Instance.

The CDN466 gateway has six different Object Classes, or types. Five are standard objects defined by the DeviceNet Specification (*Identity, Router, DeviceNet, Assembly, Connection*). One specific object defines for the CDN466 gateway (*Serial Stream*). The *Serial Stream Object* configures the serial channel, and scans the incoming serial stream for valid message packets

The CDN466 gateway operates as a DeviceNet slave. It supports Explicit Messages and Polled I/O Messages of the predefined master/slave connection set. The Explicit Unconnected Message Manager (UCMM) is not supported. The CDN466 will be a Group 2 Only Slave device. It will support Change-of-State and Polled I/O Messages. It will also support Explicit Messaging. The DeviceNet interface will comply with the DeviceNet Physical Layer specification.

The I/O Messaging process consists of the DeviceNet master sending output data to the CDN466 in the form of a Poll/COS Command Message, and the CDN466 returning input data to the DeviceNet master in a Poll/COS Response Message. The difference between Poll and Change-of-State is Polled I/O Messaging is initiated by the DeviceNet master and responded to by the slave device

The output and input data bytes are typically mapped into data files inside the DeviceNet master. These data files are exchanged with the user application program, which acts upon the received input data and writes new output data to the DeviceNet master.



The first 4 output data bytes received from the DeviceNet master contain used to control and monitor the flow of data through the gateway. The remaining output data bytes contain serial message data to be transmitted out the serial channel.

The ToolLink Gateway can receive serial message packets up to 68 bytes long. Set the **Maximum Receive Size** equal to the size of the largest receive message packet for your application. This parameter defines the Data Field size (M) for the input bytes.

**ToolLink DeviceNet Input Bytes**

STATUS	RXCTR	TXACK	LENGTH	DATA FIELD
1 byte	1 byte	1 byte	1 byte	M bytes

The ToolLink Gateway can transmit serial message packets up to 68 bytes long. Set the **Maximum Transmit Size** equal to the size of the largest transmit message packet for your application. This parameter defines the Data Field size (N) for the output bytes.

**ToolLink DeviceNet Output Bytes**

COMMAND	RXACK	TXCTR	LENGTH	DATA FIELD
1 byte	1 byte	1 byte	1 byte	N bytes

### Serial Interface

The *Serial Stream Object* attributes configure the serial channel's baud rate, number of data bits and stop bits, parity, and flow control. This configuration applies to both the serial transmitter and receiver. The gateway has separate 128-byte serial transmit and receive FIFO buffers, allowing full duplex operation when supported by the physical layer media.

The *Serial Stream Object* is also used to configure the message packet format. A message packet is determined by one of three modes. *List* mode searches for *Pre-Delimiter* and *Post-Delimiter* byte strings at the beginning and end of a message. *Length* mode captures a specific number of message bytes, defined by *Packet Length*. *Timeout* mode uses an inter-byte delay (*Packet Timeout*) to signal the end of a message. The following examples show the three *Serial Stream Object Delimiter* modes.

When the Receive Mode is set to List, the ToolLink Gateway uses start and stop delimiter strings to identify the beginning and end of a message packet. The **Start Delimiter String** attribute defines the beginning of a message, and the **Stop Delimiter String** attribute defines the end of a message.

The **Start Delimiter String** attribute format is [length][byte1][byte2][byte3][byte4]. The length byte is 0 to 4. The remaining byte(s) define the start of a message packet, which must be a unique byte string that is not used elsewhere in the message packet. The ToolLink Gateway monitors received bytes for a match to the Start Delimiter byte string. When a match is found, the start delimiter byte(s) and all subsequent bytes are saved in the RX buffer, until a Stop Delimiter byte string is received. If the Start Delimiter String is null (length = 0), the gateway starts saving the first received byte in the RX buffer.

The **Stop Delimiter String** attribute format is [length][byte1][byte2][byte3][byte4]. The length byte is 0 to 4. The remaining byte(s) define the end of a message packet, which must be a unique byte string that is not used elsewhere in the message packet. Once a Start Delimiter String is received, the ToolLink Gateway monitors the received bytes for a match to the Stop Delimiter byte string. When a match is found, the gateway saves the stop delimiter bytes and the message packet is complete. If the Stop Delimiter String is null (length = 0), then the gateway saves the start delimiter bytes and all subsequent bytes until the **Maximum Receive Number** of bytes are received. This is a modified version of the Length Mode, using a start message delimiter to signal the start of a new fixed length message packet.

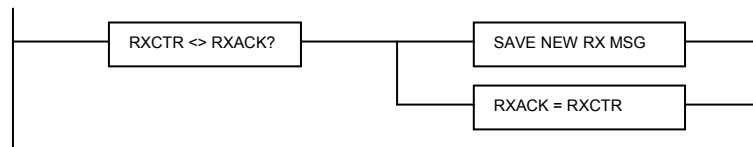
When Receive Mode = TIMEOUT, the ToolLink Gateway waits for an inter-byte delay to signal the end of a message packet. If the receiver is idle for more than 3.5 byte times (or 5 msec, whichever value is greater), then all bytes received before the timeout are grouped into a single message packet.

1 byte time = 10 bits ÷ baud rate.

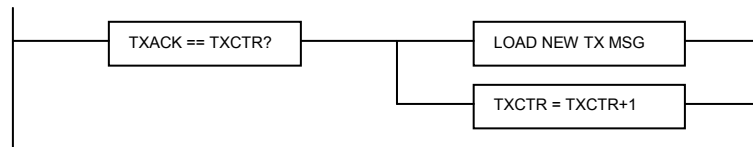
When Receive Mode = LENGTH, the ToolLink Gateway receives a fixed number of bytes as a complete message packet. The **Maximum Receive Size** parameter defines the message packet size (0 to 64 bytes) for the LENGTH mode.

Serial Synchronization

The **RX Handshake Enable** parameter selects the Receive Synchronization option. The Receive Counter (RXCTR) input byte and Receive Acknowledge (RXACK) output byte provide a handshake between the ToolLink Gateway and application program. The ToolLink Gateway always increments RXCTR (1-255) when it loads a new RX message packet into the input Data Field. When Receive Synchronization is enabled, the application must set RXACK = RXCTR to acknowledge receipt of an RX message, before the ToolLink Gateway will load the next RX message into the input Data Field. A ladder logic algorithm for Receive Synchronization is shown below.



The **TX Handshake Enable** parameter selects the Transmit Synchronization option. The Transmit Counter (TXCTR) output byte and Transmit Acknowledge (TXACK) input byte provide a handshake between the ToolLink Gateway and application program. When Transmit Synchronization is enabled, the application must increment TXCTR (1-255) when it loads a new TX message packet into the output Data Field. The ToolLink Gateway sets TXACK = TXCTR after it loads the message into the TX buffer. The application should wait for the acknowledgement before sending a new TX message, to ensure no data is lost. A ladder logic algorithm for Transmit Synchronization is shown below.



## Chapter 4 – Gateway Configuration

This chapter describes how to configure and operate the CDN466 gateway. Reading and writing attribute values over its DeviceNet interface configure the gateway. There are a variety of DeviceNet configuration tools available. Simple configuration tools use GET\_ATTRIBUTE and SET\_ATTRIBUTE explicit message commands to read and write attribute values, addressing each attribute by its *Object*, *Instance*, and *Attribute* numbers. This information is contained in Chapter 5. More sophisticated configuration tools use EDS files to simplify attribute configuration. You can configure the gateway using pull-down menus, buttons, and data entry fields from the gateway's Electronic Data sheet (EDS) file.

### Configure DeviceNet Interface

The DeviceNet Baud Rate and MAC ID Address are set using the rotary switches. Configure switches before connecting to the DeviceNet network. There is either a small triangular indicator or white indicator on the switch. Use a small screwdriver to align that indicator with the desired setting. Remove the CDN466 cover if necessary to access the rotary switches.

#### DeviceNet Baud Rate Switch

Valid settings are 125K, 250K, 500K, or PGM. When PGM is selected, the CDN466 uses the baud rate saved in its retentive memory. To save a valid baud rate in memory, set the switch to the desired baud rate and power up the CDN466 for a few seconds. Power down and set the switch to PGM. You may also write to the DeviceNet Object Baud Rate attribute.

POSITION	SETTING	POSITION	SETTING
0	125 Kbps	5	invalid
1	250 Kbps	6	invalid
2	500 Kbps	7	invalid
3	invalid	8	invalid
4	invalid	9	PGM

#### MAC ID Switches

The two MAC ID switches represent decimal numbers from 00 to 99. The LSB switch selects the *Ones* digit and the MSB switch selects the *Tens* digit. Valid MAC IDs are 00 to 63. Setting a MAC ID address greater than 63 forces the gateway to use the MAC ID saved in retentive memory. To save a valid MAC ID in memory, set the switches to the desired MAC ID and power up the CDN466 for a few seconds. Power down and set the switches to value greater than 63. You may also write to the DeviceNet Object MAC ID attribute.

MSB	LSB	Address	MSB	LSB	Address
0	0 to 9	00 to 09	6	4 to 9	stored address
1	0 to 9	10 to 19	7	0 to 3	stored address
2	0 to 9	20 to 29	8	0 to 9	stored address
3	0 to 9	30 to 39	9	0 to 9	stored address
4	0 to 9	40 to 49			
5	0 to 9	50 to 59			
6	0 to 3	60 to 63			

### Power Up Gateway

Connect the gateway to a DeviceNet network to power up the gateway.

#### DeviceNet Status LEDs

The CDN466 gateway has two bi-color status LEDs (*NET* and *MOD*) that indicate operational status. During power-up, the LEDs cycle through a sequence of alternating red and green. After power-up, the

*NET* LED should be flashing green (or solid green if allocated to a DeviceNet master) and the *MOD* LED should be solid green. If this does not occur, disconnect from DeviceNet and verify all the switch settings. See Chapter 8 for additional troubleshooting topics.

State	DeviceNet Status LED ( <i>NET</i> )
Off	No power.
Flashing Red	Configuration error. Check DeviceNet switch settings.
Solid Red	Unrecoverable error.
Flashing Green	Device not allocated to a DeviceNet master.
Solid Green	Normal runtime, device allocated as a slave.

State	Module Status LED ( <i>MOD</i> )
Off	No power.
Flashing Red	Configuration error. Check object attribute settings.
Solid Red	Unrecoverable error.
Flashing Green	Not defined.
Solid Green	Normal Operation.

### Serial Channel Status LEDs

The gateway has two bi-color LEDs to indicate serial channel activity. The *TX* LED flashes green when a packet is being transmitted. The *RX* LED flashes green when a packet is being received. A fault is indicated by solid red. After power-up, both LEDs should be off.

State	Transmit Status LED ( <i>TX</i> )
Off	No data being transmitted
Flashing Red	Not defined
Solid Red	Transmit error (parity or overrun error)
Flashing Green	Data being transmitted
Solid Green	Not defined

State	Receive Status LED ( <i>RX</i> )
Off	No data being received
Flashing Red	Not defined
Solid Red	Receive error (parity or overrun error)
Flashing Green	Data being received
Solid Green	Not defined

### Register EDS File

If using a DeviceNet configuration tool that supports Electronic Data Sheet (EDS) files, you should now register the gateway's EDS file with the software. The latest EDS file versions can be downloaded from [www.mksinst.com](http://www.mksinst.com). Select the EDS file that matches your gateway's part number and firmware version. Follow your configuration tool instructions to register EDS file. The parameters in the EDS file correspond with the Serial Stream Object attributes defined in Chapter 5.

## Chapter 5 – DeviceNet Profile

The CDN466 device operates as a slave on the DeviceNet network. The unit supports Explicit Messages and Polled I/O Messages of the predefined master/slave connection set. It does not support the Unconnected Message Manager (UCMM).

### DeviceNet Message Types

As a group 2 slave device the CDN466 supports the following message types.

CAN IDENTIFIER	GROUP 2 Message Type
10xxxxxx111	Duplicate MACID Check Message
10xxxxxx110	Unconnected Explicit Request Message
10xxxxxx101	Master I/O Poll Command Message
10xxxxxx100	Master Explicit Request Message

xxxxxx = Node Address

### DeviceNet Object Classes

The CDN466 device supports the following DeviceNet object classes.

CLASS CODE	OBJECT TYPE
01 (0x01)	Identity
02 (0x02)	Router
03 (0x03)	DeviceNet
04 (0x04)	Assembly
05 (0x05)	Connection
100 (0x64)	User Defined Serial Interface

### Identity Object      Class Code: 01 (0x01)

The Identity Object is required on all devices and provides identification of and general information about the device.

**Table 2 Class Attributes**

Attribute	Access	Name	Type	Value
1	Get	Revision	UINT	1
2	Get	Max Instance	UINT	1

**Table 3 Instance 1 Attributes**

Attr	Access		Name	Type	Value
	User	Factory			
1	Get	Get	Vendor	UINT	59
2	Get	Get	Product Type	UINT	12 = Communications Adapter
3	Get	Get	Product Code	UINT	7456
4	Get	Get	Revision	STRUCT OF	
			Major Revision	USINT	1
			Minor Revision	USINT	1
5	Get	Get	Device Status	WORD	See Below
6	Get	Get	Serial Number	UDINT	See Below
7	Get	Get	Product Name	STRUCT OF	
			Length	USINT	6
			Name	STRING [6]	CDN466

**Table 4 Common Services**

Service Code	Class	Instance	Service Name
05 (0x05)	No	Yes	Reset
14 (0x0E)	Yes	Yes	Get_Attribute_Single
50 (0x32)	No	Yes	Change_Mode

Revision – Attribute 4

MKS/CIT maintains strict revision control. The major revision number will increment as functional enhancements are implemented. The minor revision will increment if minor changes are incorporated.

Device Status – Attribute 5

bit 0	owned	0=not owned 1=owned (allocated)
bit 1	reserved	0
bit 2	configured	0
bit 3	reserved	0
bit 4-7	vendor specific	0
bit 8	minor cfg fault	0=no fault 1=minor fault
bit 9	minor dev.fault	0=no fault 1=minor device fault
bit 10	major cfg.fault	0=no fault 1=major cfg. Fault
bit 11	major dev.fault	0=no fault 1=major device fault
bit 12-15	reserved	0

Serial Number – Attribute 6

The serial number is encoded in the product during the manufacturing cycle and is guaranteed to be unique across all product lines produced by MKS/CIT. The Serial Number matches the bar codes serial number on the unit.



**Router Object Class Code: 02 (0x02)**

The Message Router Object provides a messaging connection point through which a Client may address a service to any object class or instance residing in the physical device.

**Table 5 Class Attributes**

Attribute	Access	Name	Type	Value
1	Get	Revision	UINT	1
6	Get	Max Class Attribute ID	UINT	7
7	Get	Max Instance Attribute ID	UINT	2

**Table 6 Instance 1 Attributes**

Attribute	Access	Name	Type	Value
2	Get	<b>Number of Connections</b>	UINT	2

**Table 7 Common Services**

Service Code	Class	Instance	Service Name
14 (0x0E)	Yes	Yes	Get_Attribute_Single

**DeviceNet Object Class Code: 03 (0x03)****Table 8 Class Attributes**

Attribute	Access	Name	Type	Value
1	Get	Revision	UINT	2

**Table 9 Instance 1 Attributes**

Attribute	Access	Name	Type	Value
1	Get/Set	MACID	USINT	See Below
2	Get/Set	Baud Rate	USINT	See Below
5	Get	Allocation Information	STRUCT of	See Below
		Choice Byte	BYTE	
		Master Node Addr.	USINT	

**Table 10 Common Services**

Service Code	Class	Instance	Service Name
14 (0x0E)	Yes	Yes	Get_Attribute_Single
16 (0x10)	No	Yes	Set_Attribute_Single

MACID – Attribute 1

The MACID is set using two BCD rotary switches located on the front panel. Valid MACID addresses are 0 to 63 (0 to 3F Hex). Setting the switch address to a value greater than 63 will disable the switch and allow software setting of the MACID. The software setting defaults to the last hardware setting. The switch is only read during power up.

Baud Rate – Attribute 2

Settable only if the Baud Rate switch is set to a value greater than 2. Value returned will be switch value if less than 4 or the last value set.

Switch/Value	Speed
0	125 kbits
1	250 kbits
2	500 kbits
>2	Software settable

Allocation Information – Attribute 5

Allocation_byte	
bit 0	explicit set to 1 to allocate
bit 1	polled set to 1 to allocate
bit 2	strobed (not supported)
bit 3-7	reserved (always 0)

**Assembly Object      Class Code: 04 (0x04)**

The Assembly Objects bind attributes of multiple objects to allow data to or from each object to be sent or received over a single connection.

**Table 11 Class Attributes**

Attribute	Access	Name	Type	Value
1	Get	Revision	UINT	2
2	Get	Max Instance	UINT	101

**Table 12 Instance 100 Attributes**

Attribute	Access	Name	Type	Value
3	Get	Data Stream (Input)	see notes	(1)

**Table 13 Instance 101 Attributes**

Attribute	Access	Name	Type	Value
3	Get/Set	Data Stream (Output)	see notes	(2)

**Table 14 Common Services**

Service Code	Class	Instance	Service Name
14 (0x0E)	Yes	Yes	Get_Attribute_Single
16 (0x10)	No	Yes	Set_Attribute_Single

- (1) The input data stream is structured as either an array of bytes or as a SHORT\_STRING consisting of a single byte length field and 'n' data bytes. Refer to the serial stream object class 100 for further information.
- (2) The output data stream is structured as either an array of bytes or as a SHORT\_STRING consisting of a single byte length field and 'n' data bytes. Refer to the serial stream object class overview and class 100 for further information.

**Connection Object Class Code: 05 (0x05)**

The Connection Objects manage the characteristics of each communication connection. As a Group II Only Slave device the unit supports one explicit message connection and a POLL message connection.

**Table 15 Class Attributes**

Attribute	Access	Name	Type	Value
1	Get	Revision	UINT	1

**Table 16 Instance 1 Attributes (Explicit Connection)**

Attribute	Access	Name	Type	Value
1	Get	State	USINT	See Below
2	Get	Instance Type	USINT	0 = Explicit Message
3	Get	Transport Class Trigger	USINT	0x83
4	Get	Production Connection	UINT	See Below
5	Get	Consumed Connection	UINT	See Below
6	Get	Initial Comm. Char.	USINT	0x21
7	Get	Production Size	UINT	68
8	Get	Consumed Size	UINT	68
9	Get/Set	Expected Packet Rate	UINT	default 2500 msec
12	Get/Set	Timeout Action	USINT	See Below
13	Get	Prod. Path Length	USINT	0
14	Get	Production Path		(null)
15	Get	Cons. Path Length	USINT	0
16	Get	Consumed Path		(null)

**Table 17 Instance 2 Attributes (POLL connection)**

Attribute	Access	Name	Type	Value
1	Get	State	USINT	See Below
2	Get	Instance Type	USINT	1 = I/O Message
3	Get	Transport Class Trigger	USINT	0x83
4	Get	Production Connection	UINT	See Below
5	Get	Consumed Connection	UINT	See Below
6	Get	Initial Comm. Char.	USINT	0x01
7	Get	Production Size	UINT	See Below
8	Get	Consumed Size	UINT	See Below
9	Get/Set	Expected Packet Rate	UINT	default 2500 msec
12	Get/Set	Timeout Action	USINT	See Below
13	Get	Prod. Path Length	USINT	See Below
14	Get/Set	Production Path	STRUCT of	
		Log. Seg., Class	USINT	0x20
		Class Number	USINT	0x04
		Log.Seg., Instance	USINT	0x24
		Instance Number	USINT	0x100 (default)
		Log.Seg., Attribute	USINT	0x30
		Attribute Number	USINT	0x03
15	Get	Cons. Path Length	USINT3	6
16	Get/Set	Consume Path	STRUCT of	
		Log. Seg., Class	USINT	0x20
		Class Number	USINT	0x04
		Log.Seg., Instance	USINT	0x24
		Instance Number	USINT	0x101 (default)
		Log.Seg., Attribute	USINT	0x30
		Attribute Number	USINT	0x03

**Table 18 Common Services**

Service Code	Class	Instance	Service Name
05 (0x05)	No	Yes	Reset
14 (0x0E)	Yes	Yes	Get_Attribute_Single
16 (0x10)	No	Yes	Set_Attribute_Single

State – Attribute 1

Connection States:

- 0 = non-existent
- 1 = configuring
- 3 = established
- 4 = timed out

Connection ID's – Attributes 4 and 5

Connection 1 Produced Connection ID: 10xxxxxx011  
 Connection 1 Consumed Connection ID: 10xxxxxx100

Connection 2 Produced Connection ID: 01111xxxxxx

Connection 2 Consumed Connection ID: 10xxxxxx101

xxxxxx = Node Address.

Production and Consumed Sizes – Attributes 7 and 8

The Production and Consumed sizes will change based on the maxrx and maxtx in addition to the overhead bytes.

Watch Dog Timeout Activity – Attribute 12

- 0 = Timeout (I/O Messaging default)
- 1 = Auto Delete (Explicit Messaging, fixed value)
- 2 = Auto Reset

**User Defined (Serial Stream) ObjectClass Code: 100 (0x64)**

The Serial Stream Object model supports a bi-directional serial stream of data. The object includes the transmit FIFO, the receive FIFO and the serial channel configuration attributes.

**Table 19 Serial Stream Class Attributes**

Attribute	Access	Name	Type	Value
1	Get	Revision	UINT	1
2	Get	Max Object Instance	UINT	1
6	Get	Max Class Identifier	UINT	7
7	Get	Max Instance Attribute	UINT	18

**Table 20 Serial Stream Object, Instance 1 Attributes**

Attribute	Access	Name	Type	Value
3	Get	Receive Data	Short_String	Received Message Data
4	Get/Set	Transmit Data	Short_String	Message data to transmit(4)
5	Get/Set	Status	USINT	See Below
6	Get	Baud Rate	USINT	See Below
7	Get/Set	Parity	USINT	See Below
8	Get/Set	Flow Control	USINT	See Below
9	Get/Set	Receive Mode	USINT	See Below
11	Get/Set	Start Delimiter String	Array[5]	See Below
12	Get/Set	Stop Delimiter String	Array[5]	See Below
13	Get/Set	RX Handshake Enable	USINT	1 = Yes, 0 = No
14	Get	Receive Counter	USINT	RXC Receive Message Counter (0-255)
15	Get	Receive Acknowledge	USINT	(See Below)
16	Get/Set	TX Handshake Enable	USINT	1 = Yes, 0 = No

17	Get	Transmit Counter	USINT	RXC Receive Message Counter (0-255)
18	Get	Transmit Acknowledge	USINT	See Below
19	Get/Set	Max Receive Size	USINT	Input Message Size (0-64) Default 12 bytes
20	Get/Set	Max Transmit Size	USINT	Output Message Size (0-64) Default 12 bytes
21	Get/Set	Byte Swap	USINT	1 = Yes, 0 = No

### Receive Data – Attribute 3

The Receive Data attribute returns data received from the serial connection with a 4-68 byte message formatted as follows:

Status	Rx Counter	TX Acknowledge	Length	Data
Byte 0	Byte 1	Byte 2	Byte 3	Max Rx Bytes

*Status:* Returns the state of the serial buffers and indicates if a parity error was found in the received data. Status is bit-mapped as follows:

- Bit 7 – RX Buffer Empty
- Bit 6 – RX Buffer Overflow
- Bit 5 – RX Parity Error
- Bit 4 – TX Buffer Empty
- Bit 3 – TX Buffer Overflow

*Rx Counter:* Increments from 0-255 each time a new serial packet received by the CDN466 is placed in the Data field.

*TX Acknowledge:* Increments from 0-255 each time a complete serial message has been transmitted by the CDN66.

*Length:* The number of bytes of the serial message in the Data field.

*Data:* Data received by the CDN466 from the serial connection, unused bytes are set to 0.

### Transmit Data – Attribute 4

Setting the Transmit Data attribute will allow data to be sent to the serial connection, as in the I/O command message. The Transmit Data attribute is formatted as follows:

Command	Rx Acknowledge	TX Counter	Length	Data
Byte 0	Byte 1	Byte 2	Byte 3	Max Rx Bytes

*Command:* The command byte allows the data in the receive and transmit buffers to be flushed.

- Bit 6 – Clear Rx Buffer (Set bit to clear)
- Bit 3 – Clear TX Buffer (Set bit to clear)

*Rx Counter:* If the Rx Handshaking attribute is enabled the Rx Acknowledge must be set to the value of Rx Counter (0-255) in the Receive data attribute before the data field will be updated with a new serial packet. Setting Rx Acknowledge to '0' will reset the Rx Counter to '0'. If Rx Handshaking attribute is disabled the Rx Counter is ignored and should be left at '0'.

**TX Counter:** If the TX Handshaking is enabled, the TX Counter must be incremented from its previous value (0-255) in order for the message in the data field to be transmitted. Setting TX Counter to '0' will reset the value of TX Acknowledge. If TX Handshaking is disabled, the value of TX Counter is ignored and the message in the Data field is transmitted each time it is written.

### Baud Rate – Attribute 6

Baud Rate controls the communications rate with the serial connection and must match the setting of the connected serial device.

Value	Baud Rate
0	19200
1	600
2	1200
3	2400
4	4800
5	9600
6	19200

### Parity – Attribute 7

The Parity attribute sets the format for characters transmitted and received across the serial connection. Note, that the connected serial device must be configured for an identical character format.

Value	Parity	Format
0	None	1 Start, 8 Data, 1 Stop (Default)
1	Odd	1 Start, 7 Data, Odd Parity, 1 Stop
2	Even	1 Start, 7 Data, Even Parity, 1 Stop

### Flow Control – Attribute 8

Value	Mode
0	None (default)
1	CTS/RTS (Hardware)

### Receive Mode – Attribute 9

Value	Mode
0	Time-Out (default)
1	Length
2	Start/Stop Delimiter

Time-Out:

In Time-Out mode the CDN466 measures the time delay between received serial characters. When 4 byte-times elapse between received bytes, this signals the end of the current message packet. The CDN466 automatically calculates the 4 byte-times based on current RS232 baud rate.

$$\text{Time-out} = 4 \text{ bytes} \times 9 \text{ bits/byte} \div \text{data rate (bits/second)}$$

Length:

In length mode the CDN466 is configured to receive fixed-length message packets. The length shall be software selectable from 0 to 64 bytes.

If the length is set to 0, the CDN466 will operate in a “free running” mode. For Polled I/O messaging, all RX buffer bytes are returned in response to a Poll Request message. If there are more RX bytes than will fit, then the remaining bytes are sent in the next poll/explicit transaction. For free-running mode with Change-of-State messaging, each received byte will generate a Change-of-State input message. Note that this is not an efficient use of DeviceNet bandwidth.

Start/Stop Delimiters:

In this mode the CDN466 will search for a fixed strings to mark the stop and/or end of a serial message. Behavior in this mode depends on the setting of the Start Delimiter String and Stop Delimiter String attributes.

*Start + Stop:*

If both are used, the Start and Stop delimiters will mark the beginning and end of each message. Characters outside of the delimiters will be ignored. For example, if Start Delimiter = ‘<’ and Stop Delimiter = ‘>’, the serial data stream “1<ABC>2<XY>3” would be returned in two separate DeviceNet messages as “<ABC>”, and “<XY>”

Stop Delimiter:

If only the Stop Delimiter String is used, (Start Delimiter = 0,) the CDN466 will use the first character received as the beginning of a message and the last character received as the end of message. For example, if Stop Delimiter = “!”, the serial data stream “ABCD!EFG!HI” would be returned in two separate DeviceNet messages as “ABCD!” and “EFG!”

Start Delimiter:

If only the Start Delimiter String is used (Stop Delimiter String = 0,) the CDN466 will use the Start delimiter to mark the beginning of the message and will wait until it receives the number of bytes defined by Maximum Receive Size attribute to mark the end of the message. For example, if Start Delimiter = “&” and Maximum Receive Size = 5, the data stream “1&ABCD&EFGHIJ” would return two separate DeviceNet messages “@ABCD” and “@EFGH”

No Delimiters:



If the Start Delimiter String and Stop Delimiter String are both set to zero, the CDN466 will operate in a “free running” mode. For Polled I/O messaging, all RX buffer bytes are returned in response to a Poll Request message. If there are more RX bytes than will fit, then the remaining bytes are sent in the next poll/explicit transaction. For free-running mode with Change-of-State messaging, each received byte will generate a Change-of-State input message. Note that this is not an efficient use of DeviceNet bandwidth.

Start Delimiter String – Attribute 11

The Start Delimiter String is a constant sequence of characters the CDN466 uses to identify that beginning of a received serial message when the Receive Mode attribute is in the “Start/Stop Delimiter” mode. The number of bytes in the string can be 0 (not used) to 4. The Start Delimiter set as a DeviceNet SHORT\_STRING, which includes a length byte followed by the ASCII bytes defining the string. For example, to use the four byte string “<<<<” is the SHORT STRING “4, '<', '<', '<', '<'” or “0x04, 0x3C, 0x3C, 0x3C, 0x3C”.

Stop Delimiter String – Attribute 12

The Stop Delimiter String is a constant sequence of characters the CDN466 uses to identify that end of a received serial message when the Receive Mode attribute is in the “Start/Stop Delimiter” mode. The number of bytes in the string can be 0 (not used) to 4. The Stop Delimiter set as a DeviceNet SHORT\_STRING, which includes a length byte followed by the ASCII bytes defining the string. For example, to use the four byte string “>>>>” is the SHORT STRING “4, '>', '>', '>', '>'” or “0x04, 0x3C, 0x3C, 0x3C, 0x3C”.

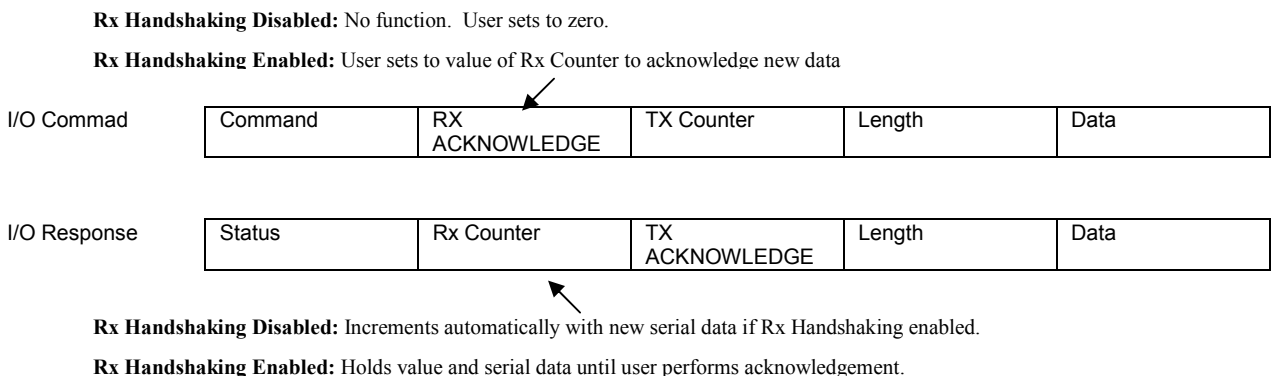
Rx Handshake Enable – Attribute 13

By default (Rx Handshaking disabled), the CDN466 updates the data received from the serial connection to the DeviceNet I/O response as soon as a complete message is received. At slow DeviceNet rates, however, there remains the possibility that more than one serial message will be received between successive polls, resulting in loss of data.

If the Rx Handshaking Enable is set, the serial data made available in the I/O response will not be overwritten by a new message until the user acknowledges that the data has been received. Once the user has acknowledged the new serial data by setting the Rx Acknowledge value equal to the Rx Counter, the I/O response will be free to update with new data.

*Important:* As the serial buffer may continue to receive data as the CDN466 is waiting for an acknowledgement, the receive buffer may reach an overflow condition if new data is not acknowledged at a sufficient enough rate.

Figure 1: Rx Handshaking



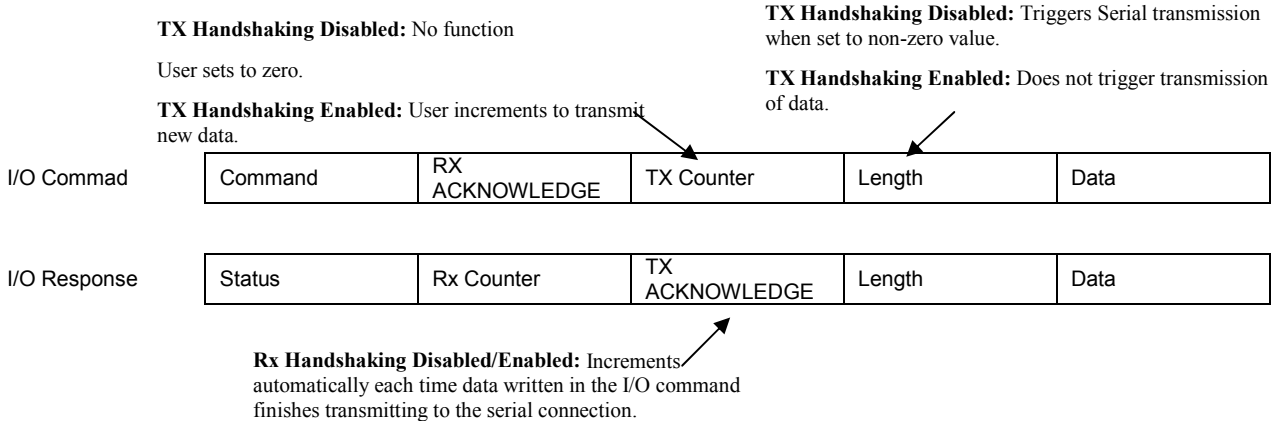
TX Handshake Enable – Attribute 16

By default (TX Handshaking disabled,) the data in the I/O command is transmitted each time the I/O command is written with a non-zero value for Length. If the user polls the CDN466 for newly received serial data, the Length field must be set to zero if no data is to be transmitted. In this case, the TX Counter performs no function and should remain at zero.

With TX Handshaking enabled, transmission of serial data is triggered from the TX Counter rather than from the Length field. The data in the I/O response is transmitted to the serial connection when the value of TX Counter is incremented from its previous state.

The value of TX Acknowledge will increment automatically to match the TX Counter after data from the I/O command has finished transmitting to the serial connection. If the user waits for TX Acknowledge to change before triggering the transmission of new data, the user should not be able to overflow the transmit buffer.

Figure 2: TX Handshaking



Maximum Rx Size – Attribute 19

Value = 0 to 64 (default 12)

Maximum Rx Size sets the length of the data field in the I/O response. It should be set greater or equal to the value of the largest message to be received from the serial connection. The produce size of the I/O connection will be configured to 4 + Maximum TX Size (4 control bytes + Maximum Rx Size data bytes.)

Maximum TX Size – Attribute 20

Value = 0 to 64 (default 12)

Maximum TX Size sets the length of the data field in the I/O command. It should be set greater or equal to the value of the largest message to be transmitted to the serial connection. The consume size of the I/O connection will be configured to Maximum TX Size + 4 (4 control bytes + Maximum TX Size data bytes).

Byte Swapping – Attribute 21

When enabled, this switches the byte positions of each byte pair in the data field poll command and poll response. This is useful with many PLC devices that reverse the positions of each byte pair in memory, garbling up string data.

Default byte positions: [1] [2] [3] [4] [5] [6][7]

Swapped positions: [2] [1] [4] [3] [6] [5][7]

Note that if the last byte has an odd-numbered position, it retains the same position with “Byte Swapping” enabled.

## Appendix A – Product Specifications

### DeviceNet Interface

Power Requirements:	11 - 28 Vdc @ 50 mA
Loss of Ground:	Yes
Reverse Polarity:	-30 Vdc
Signal Levels:	ISO11898

### Serial Channel

Isolation:	500 Volts
ESD Protection:	+/- 10 kV
Overload Protection:	+/- 30 Volts
Short Circuit:	Indefinite
RS232 Output Levels:	+/- 7.9 Volts (unloaded, typical)

### Environmental

Operating Temperature:	0° C to 70° C
Storage Temperature:	-25° C to 85° C
Size (inches):	3.25 x 2.37 x 1.08
Mounting (inches)	0.5 tabs, 3/16 diameter mounting holes
PCB Encapsulation:	RTV Silicon Compound

## Appendix B – ASCII Character Codes

Non-Printable Characters					Printable Characters								
Hex	Dec	Char	Name	Kybd	Hex	Dec	Char	Hex	Dec	Char	Hex	Dec	Char
0x00	0	NUL	Null	Ctrl @	0x20	32	Space	0x40	64	@	0x60	96	`
0x01	1	SOH	Start of heading	Ctrl A	0x21	33	!	0x41	65	A	0x61	97	a
0x02	2	STX	Start of text	Ctrl B	0x22	34	"	0x42	66	B	0x62	98	b
0x03	3	ETX	End of text	Ctrl C	0x23	35	#	0x43	67	C	0x63	99	c
0x04	4	EOT	End of transmit	Ctrl D	0x24	36	\$	0x44	68	D	0x64	100	d
0x05	5	ENQ	Enquiry	Ctrl E	0x25	37	%	0x45	69	E	0x65	101	e
0x06	6	ACK	Acknowledge	Ctrl F	0x26	38	&	0x46	70	F	0x66	102	f
0x07	7	BEL	Bell	Ctrl G	0x27	39	'	0x47	71	G	0x67	103	g
0x08	8	BS	Backspace	Ctrl H	0x28	40	(	0x48	72	H	0x68	104	h
0x09	9	HT	Horizontal tab	Ctrl I	0x29	41	)	0x49	73	I	0x69	105	i
0x0A	10	LF	Line feed	Ctrl J	0x2A	42	*	0x4A	74	J	0x6A	106	j
0x0B	11	VT	Vertical tab	Ctrl K	0x2B	43	+	0x4B	75	K	0x6B	107	k
0x0C	12	FF	Form feed	Ctrl L	0x2C	44	,	0x4C	76	L	0x6C	108	l
0x0D	13	CR	Carriage return	Ctrl M	0x2D	45	-	0x4D	77	M	0x6D	109	m
0x0E	14	SO	Shift out	Ctrl N	0x2E	46	.	0x4E	78	N	0x6E	110	n
0x0F	15	SI	Shift in	Ctrl O	0x2F	47	/	0x4F	79	O	0x6F	111	o
0x10	16	DLE	Data line escape	Ctrl P	0x30	48	0	0x50	80	P	0x70	112	p
0x11	17	DC1	Device control 1	Ctrl Q	0x31	49	1	0x51	81	Q	0x71	113	q
0x12	18	DC2	Device control 2	Ctrl R	0x32	50	2	0x52	82	R	0x72	114	r
0x13	19	DC3	Device control 3	Ctrl S	0x33	51	3	0x53	83	S	0x73	115	s
0x14	20	DC4	Device control 4	Ctrl T	0x34	52	4	0x54	84	T	0x74	116	t
0x15	21	NAK	Negative acknowledge	Ctrl U	0x35	53	5	0x55	85	U	0x75	117	u
0x16	22	SYN	Synchronous idle	Ctrl V	0x36	53	6	0x56	86	V	0x76	118	v
0x17	23	ETB	End of transmit block	Ctrl W	0x37	55	7	0x57	87	W	0x77	119	w
0x18	24	CAN	Cancel	Ctrl X	0x38	56	8	0x58	88	X	0x78	120	x
0x19	25	EM	End of medium	Ctrl Y	0x39	57	9	0x59	89	Y	0x79	121	y
0x1A	26	SUB	Substitute	Ctrl Z	0x3A	58	:	0x5A	90	Z	0x7A	122	z
0x1B	27	ESC	Escape	Ctrl [	0x3B	59	;	0x5B	91	[	0x7B	123	{
0x1C	28	FS	File separator	Ctrl \	0x3C	60	<	0x5C	92	\	0x7C	124	
0x1D	29	GS	Group separator	Ctrl ]	0x3D	61	=	0x5D	93	]	0x7D	125	}
0x1E	30	RS	Record separator	Ctrl ^	0x3E	62	>	0x5E	94	^	0x7E	126	~
0x1F	31	US	Unit separator	Ctrl _	0x3F	63	?	0x5F	95	_	0x7F	127	DEL



## WARRANTY

### Remote Monitor Unit

MKS Instruments, Inc. (**MKS**) warrants that for one year from the date of shipment the equipment described above (the "equipment") manufactured by **MKS** shall be free from defects in materials and workmanship and will correctly perform all date-related operations, including without limitation accepting data entry, sequencing, sorting, comparing, and reporting, regardless of the date the operation is performed or the date involved in the operation, provided that, if the equipment exchanges data or is otherwise used with equipment, software, or other products of others, such products of others themselves correctly perform all date-related operations and store and transmit dates and date-related data in a format compatible with **MKS** equipment. THIS WARRANTY IS **MKS'** SOLE WARRANTY CONCERNING DATE-RELATED OPERATIONS.

For the period commencing with the date of shipment of this equipment and ending one year later, **MKS** will, at its option, either repair or replace any part which is defective in materials or workmanship or with respect to the date-related operations warranty without charge to the purchaser. The foregoing shall constitute the exclusive and sole remedy of the purchaser for any breach by **MKS** of this warranty.

The purchaser, before returning any equipment covered by this warranty, which is asserted to be defective by the purchaser, shall make specific written arrangements with respect to the responsibility for shipping the equipment and handling any other incidental charges with the **MKS** sales representative or distributor from which the equipment was purchased or, in the case of a direct purchase from **MKS**, with the **MKS** home office in Andover, Massachusetts, USA.

This warranty does not apply to any equipment which has not been installed and used in accordance with the specifications recommended by **MKS** for the proper and normal use of the equipment. **MKS** shall not be liable under any circumstances for indirect, special, consequential, or incidental damages in connection with, or arising out of, the sale, performance, or use of the equipment covered by this warranty.

**MKS** recommends that all **MKS** pressure and flow products be calibrated periodically (typically every 6 to 12 months) to ensure accurate readings. When a product is returned to **MKS** for this periodic re-calibration it is considered normal preventative maintenance not covered by any warranty.

THIS WARRANTY IS IN LIEU OF ALL OTHER RELEVANT WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING THE IMPLIED WARRANTY OF MERCHANTABILITY AND THE IMPLIED WARRANTY OF FITNESS FOR A PARTICULAR PURPOSE, AND ANY WARRANTY AGAINST INFRINGEMENT OF ANY PATENT.