CDN066
DEVICENET
SPECIFICATIONS

CDN066 DEVICE

Revision History

Revision	Description	Date
1.0	Initial Release	03/15/97
1.1	Added BLOCK transfer support	05/29/97
1.2	Added Padded String support, reduced fifo sizes	06/21/97
1.3	Added Sync. Handshake control	08/04/97
3.0	Corrected errors in Xon/Xoff protocol	
4.0	Upgrade to DNet Rev. 2.0 Added support for status byte and status clear in poll Disallowed changing of items that affect produce consume size when poll connection has been established	07/08/98

DeviceNet Information

The CDN066 (DeviceNet Serial Gateway) 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 Explicit Unconnected Message Manager (UCMM).

The device provides an RS232 serial interface, allowing serial information to be sent and received from peripheral devices. The RS232 serial information is buffered in an internal 64 byte receive FIFO, allowing the unit to operate asynchronous to the DeviceNet network. Similarly, information to be transmitted to the serial device is buffered in a 64 byte serial fifo. The device may be configured to operate at a number of different baud rates and supports flow control and parity.

Units with version 2.0 firmware provide additional support for BLOCK transfer of serial data. The CDN066 may be programmed with a prefix/suffix delimiter character and will automatically parse the incoming data stream into complete packets, ensuring that polled I/O scanners only receive complete data sets.

To further support polled I/O scanners it is possible to prepend a sequence number to each transaction. The CDN066 will only transmit serial data if the sequence number of the poll request is different than the previous poll request. Each poll response packet can also include a sequence number, which auto increments on each packet sent.

Version 2.1 devices support padded DeviceNet strings in which incoming strings may be left or right justified with PAD characters to ensure that all packets are of a fixed length. In addition, the Stream object configuration (Maximum Receive Size and Maximum Transmit Size) affect the Produced and Consumed sizes of the POLL connection, allowing these to be reduced to minimize unnecessary bus traffic.

Version 4.0 added support to provide the status byte in the poll response, and allow the status information to be cleared in the poll command. Changed firmware to only allow changing of items that affect the produce and consume size when a poll connection is in the non-established state.

CDN066 DEVICE

DeviceNet Message Types

As a group 2 slave device the CDN066 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 Class Services

As a group 2 slave device the CDN066 supports the following class services and instance services.

SERVICE CODE	SERVICE NAME
05 (0x05)	Reset
14 (0x0E)	Get Attribute Single
16 (0x10)	Set Attribute Single
75 (0x4B)	Allocate Group 2 Identifier Set
76 (0x4C)	Release Group 2 Identifier Set

CDN066 DEVICE

DeviceNet Object Classes

The CDN066 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
64 (0x40)	User defined serial interface

CDN066 DEVICE

Identity Object Class Code: 01 (0x01)

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

Identity Object 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	7

Identity Object, Instance 1 Attributes

Attribute	Access	Name	Type	Value
1	Get	Vendor	UINT	59
2	Get	Product Type	UINT	12 = Communications
3	Get	Product Code	UINT	1
4	Get	Revision	STRUCT OF	
		Major Revision	USINT	4
		Minor Revision	USINT	0
5	Get	Device Status	UINT	(1)
6	Get	Serial Number	UINT	(2)
7	Get	Product Name	STRUCT OF	
		Length	USINT	6
		Name	STRING [6]	CDN066

Common Services

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

CDN066 DEVICE

(1) Device Status

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

2) Unique Serial Number

CDN066 DEVICE

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.

Router Object Class Attributes

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

Router Object, Instance 1 Attributes

Attribute	Access	Name	Type	Value
2	Get	Number of Connections	UINT	2

Common Services

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

CDN066 DEVICE

DeviceNet Object Class Code: 03 (0x03)

DeviceNet Object Class Attributes

Attribute	Access	Name	Type	Value
1	Get	Revision	UINT	2

Router Object, Instance 1 Attributes

Attribute	Access	Name	Type	Value
1	Get/Set	MACID	USINT	(1)
2	Get/Set	Baud Rate	USINT	(2)
3	Get/Set	Bus Off Interrupt	BOOL	(3)
4	Get/Set	Bus Off Counter	USINT	(4)
5	Get/Spc	Allocation Information	STRUCT of	(5)
		Choice Byte	BYTE	
		Master Node Addr.	USINT	

Common Services

Service Code	Class	Instance	Service Name
14 (0x0E)	Yes	Yes	Get_Attribute_Single
16 (0x10)	No	Yes	Set_Attribute_Single
75 (0x4B)	No	Yes	Allocate Master/Slave
76 (0x4C)	No	Yes	Release Master/Slave

(1) Settable only if the MacID switches are set to a value greater than 63. Value returned will be switch value if less than 64 or the last value set.

(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
3	Software settable

(3) Bus Off Interrupt (BOI) determines action if Bus Off state encountered. Following values are supported:

BOI	Action
0	Hold chip in OFF state (default)
1	If possible reset CAN chip

(4) Bus Off Counter will be forced to 0 whenever set regardless of the data value provided.

(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)

CDN066 DEVICE

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.

Assembly Object Class Attributes

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

Assembly Object, Instance 1 Attributes

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

Assembly Object, Instance 2 Attributes

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

Common Services

Service Code	Class	Instance	Service Name
14 (0x0E)	Yes	Yes	Get_Attribute_Single
16 (0x10)	Yes	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 64 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 64 for further information.

CDN066 DEVICE

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.

Connection Object Class Attributes

Attribute	Access	Name	Type	Value
1	Get	Revision	UINT	1

Connection Object, Instance 1 Attributes (Explicit Message)

Attribute	Access	Name	Type	Value
1	Get	State	USINT	(1)
2	Get	Instance Type	USINT	0 = Explicit Message
3	Get	Transport Class Trigger	USINT	0x83
4	Get	Production Connection	UINT	(2)
5	Get	Consumed Connection	UINT	(2)
6	Get	Initial Comm. Char.	USINT	0x21
7	Get	Production Size	UINT	67
8	Get	Consumed Size	UINT	71
9	Get/Set	Expected Packet Rate	UINT	Default 2500 msec
12	Get/Set	Timeout Action	USINT	(3)
13	Get	Prod. Path Length	USINT	0
14	Get	Production Path		(null)
15	Get	Cons. Path Length	USINT	0
16	Get	Consumed Path		(null)
17	Get	Production Inhibit	UINT	0

Connection Object, Instance 2 Attributes (POLL connection)

Attribute	Access	Name	Type	Value
1	Get	State	USINT	(1)
2	Get	Instance Type	USINT	1 = I/O Message
3	Get	Transport Class Trigger	USINT	0x82
4	Get	Production Connection	UINT	(2)
5	Get	Consumed Connection	UINT	(2)
6	Get	Initial Comm. Char.	USINT	0x1
7	Get	Production Size	UINT	See Stream Object

8	Get	Consumed Size	UINT	See Stream Object
9	Get/Set	Expected Packet Rate	UINT	Default 2500 msec
12	Get/Set	Timeout Action	USINT	(3)
13	Get	Prod. Path Length	USINT	6
14	Get	Production Path	STRUCT of	
		Log. Seg., Class	USINT	0x20
		Class Number	USINT	0x04
		Log.Seg., Instance	USINT	0x24
		Instance Number	USINT	0x01
		Log.Seg., Attribute	USINT	0x30
		Attribute Number	USINT	0x03
15	Get	Cons. Path Length	USINT	6
16	Get	Production Path	STRUCT of	
		Log. Seg., Class	USINT	0x20
		Class Number	USINT	0x04
		Log.Seg., Instance	USINT	0x24
		Instance Number	USINT	0x02
		Log.Seg., Attribute	USINT	0x30
		Attribute Number	USINT	0x03
17	Get	Production Inhibit	UINT	0

Common Services

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

(1) Connection States:

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

(2) Connection ID's:

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.

(3) Watch Dog TimeOut Activity:

0 = Timeout (Explicit Messaging default)
1 = Auto Delete
2 = Auto Reset (I/O Message default)

(4) If no data is available during the poll response a 0 length (null) packet is returned.

CDN066 DEVICE

User Defined (Serial Stream) Object Class Code: 64 (0x40)

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.

Serial Stream Class Attributes

Attribute	Access	Name	Type	Value
1	Get	Revision	UINT	2
2	Get	Max Object Instance	UINT	1
6	Get	Max Class Identifier	UINT	7
7	Get	Max Instance Attribute	UINT	22

Serial Stream Object, Instance 1 Attributes

Attribute	Access	Name	Type	Value
3	Get	Receive Data	See Notes	(3)
4	Set	Transmit Data	See Notes	(4)
5	Get/Set	Status	USINT	(5)
6	Get/Set	Baud Rate	USINT	(6)
7	Get/Set	Parity	USINT	(7)
8	Get	Data Size	USINT	(8)
9	Get	Stop Bits	USINT	(9)
10	Get/Set	Flow Control	USINT	(10)
11	Get/Set	Receive Count	USINT	(11)
12	Get/Set	Transmit Count	USINT	(12)
13	Get/Set	Maximum Receive Size	USINT	(13) *
14	Get/Set	Data Format	USINT	(14) *
15	Get/Set	Block Mode	USINT	(15) *
16	Get/Set	Receive Delimiter	USINT	(16)
17	Get/Set	Pad Char	CHAR	(17)
18	Get/Set	Maximum Transmit size	USINT	(18) *
19	Get/Set	Idle String	SHORT_STRING	(19)
20	Get/Set	Fault String	SHORT_STRING	(20)
21	Get/Set	Status Enable	USINT	(21)*
22	Get/Set	Status Clear Enable	USINT	(22)*

Common Services

CDN066 DEVICE

Service Code	Class	Instance	Service Name
5 (0x05)	No	Yes	Reset
14 (0x0E)	Yes	Yes	Get_Attribute_Single
16 (0x10)	No	Yes	Set_Attribute_Single

* Items indicated with an asterisk may affect the Produced or Consumed size of the Poll Connection.

(3) The Received Data (Attribute 3) and Transmitted Data (Attribute 4) are either an array of bytes or a DeviceNet defined SHORT_STRING, consisting of a length byte followed by the specified number of valid data bytes. The format used is determined by the Data Format parameter. Note that reading of the Transmit Data will return a single byte, indicating the last byte of the FIFO.

When a packet is received with 0 data bytes no data is transmitted. If the transmit FIFO does not have sufficient room for the packet no response packet is generated.

When data is read the response packet will be either an array of bytes or a SHORT_STRING. If no data is available either a NULL packet or an array with a length byte of 0 is returned.

(5) The Status information (Attribute 5) indicates whether data transfer errors have occurred. It is bit mapped as follows:

Bit	Interpretation
0	Transmit channel blocked
1	Transmit FIFO empty
2 *	Receive Parity error
3	Receive FIFO empty
4 *	Receive Overflow
5 *	Framing Error
6 *	Transmit FIFO Overflow
7	State of CTS signal (1 == Asserted)

* Writing any value to the Status field will clear the error bits.

(6) The Baud Rate (attribute 6) may be set by the RS232 Baud Rate switch or by software. To enable software setting the switch must be in the 'PRG' position.

Baud Rate	Interpretation
0	9600 baud
1	4800 baud
2	2400 baud
3	1200 baud
4	600 baud
5	300 baud
6	19.2 Kbaud

(7) The Parity (attribute 7) may be set by software. Note that setting the parity to 0 forces the data length size to 8. Setting the parity to non-zero forces the data length size to 7.

Parity	Interpretation
0	No parity
1	Even parity
2	Odd parity
3	N/A
4	N/A
5	Force to 1
6	Force to 0

(8) The Data size (Attribute 8) is read only. The CDN066 serial channel always processes 8 information bits. If parity is set to 0 (no parity) 8 data bits are transmitted/received. If the parity is set to a non-zero value then only 7 data bits are transmitted and the 8^{th} bit is used for the parity bit. The Data Size field is read only.

(9) The Stop bits (Attribute 9) is read only. The CDN066 serial channel always operates with 1 stop bit. The Stop Bits field is read only and fixed at 1 stop bit.

(10) Flow control (Attribute 10) may be set by software.

Flow Control	Interpretation
0	No flow control
1	X-ON/X-OFF flow control
2	Hardware flow control
3	DO NOT USE
4	CTS Detect Mode

If the flow control is set to 1 the ASCII standard X-OFF (CTRL S) character will force the transmit function to block. Characters will be buffered in the transmit FIFO until the transmitter is re-enabled using the X-ON (CTRL Q) character. Note that the CTRL-S and CTRL-Q characters will be stripped from the incoming data stream, making this protocol unsuitable for binary data transmission.

When the receive FIFO is full the CDN066 will transmit an X-OFF character. An X-ON character is transmitted when the number of characters in the receive FIFO drops below 50%.

If the flow control is set to 2 the CDN066 uses the hardware RTS and CTS signals to control flow. The unit will only transmit if the CTS signal (pin 8) is asserted (+ voltage).

If the receive FIFO has room to receive characters the RTS (pin 7) signal is asserted (+ voltage).

The CTS Detect mode allows the CDN066 to detect when the CTS signal is present. When the signal is asserted the CDN066 is free to communicate to the RS232 device. When de-asserted the CDN066 will flush the Receive FIFO, treating the RS232 device as being Off-Line. The CTS Detect mode may be used to determine if the RS232 device is connected to the CDN066.

(11) The Receive Count (Attribute 11) indicates the number of characters currently available in the receive FIFO. Writing any value will flush the receive FIFO.

(12) The Transmit Count (Attribute 12) indicates the number of characters currently in the transmit FIFO. Writing any value will flush the transmit FIFO.

(13) The Maximum Receive Size (Attribute 13) indicates the maximum number of data bytes to be returned when the receive FIFO is read (attribute 3) either using EXPLICIT messages or through the POLL connection. Setting this attribute will automatically reset the Produced Connection size as:

Connection size = Max Rcv Size (Maximum size is 64)
+ 1 (if Status Byte enabled)
+ 1 (if String Format enabled)
+ 1 (if Receive Seq. Num. enabled)

The maximum connection size is 67 bytes.
This attribute affects the produce size, and is only settable when the the poll connection is in the non-established state.

(14)The Data Format (Attribute 14) control byte determines the type of data strings transferred over the DeviceNet channel which may be either an array of bytes or a DeviceNet defined SHORT_STRING, consisting of a length byte followed by the specified number of valid data bytes. Note that the data length byte does not appear on the serial channel. The Data format control byte also determines whether the parity information is retained in the receive FIFO. If the bit is cleared then the parity information is retained. If set, the parity information is overwritten with a 0, ensuring that only valid ASCII characters (0-7FH) appear in the FIFO.

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
X	X	X	X	PADR	PL/R	Strip Parity	String Format

String Format	Interpretation
0	Process FIFO packets as SHORT_STRING variables
1	Process FIFO packets as an array of bytes. The array length implicitly defines the number of valid bytes.

Strip Parity	Interpretation
0	Retain Parity information in receive FIFO
1	Set MSB of receive FIFO data to 0

PL/R	Interpretation
0	Left justify received character string if PADR set
1	Right justify received character string if PADR set

PADR	Interpretation
0	Do not attempt to PAD received characters
1	Pad received characters strings with PADCHAR

If the PADR bit is set in block mode with the Strip Delimiter bit clear the Pad characters will be inserted between the last valid data bit and the end of the packet.

This attribute affects the produce and consume size, and is only settable when the the poll connection is in the non-established state.

(15) The Block Mode (Attribute 15) control byte determines the whether the unit preparses the RS232 serial stream, whether block sequence numbers are pre-pended to the DeviceNet packets and whether received data is retransmitted on subsequent POLL requests. The control byte has the following format:

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
X	Sync	ReSend	Enable Xmit Seq. Number	Enable Rcv. Seq.Number	Delimiter Enable	Strip Delimiter	Pre/Post Delimiter

Pre/Post Delimiter	Interpretation
0	Delimiter (if enabled) occurs at the end of the packet.
1	Delimiter (if enabled) occurs at the start of the packet. The packet length is limited to the <Max Receive Size> length. Excess characters are discarded.

Strip Delimiter	Interpretation
0	The delimiter character appears in the response packet.
1	The delimiter character is removed from the response packet.

Delimiter Enable	Interpretation
0	Disable the delimit character function
1	Enable the delimit character function

Enable Rcv.Seq.Num	Interpretation
0	Disable the receive sequence number

1 Each response packet will have a sequential number pre-pended to allow the scanner to detect new response data.

Enable Xmt.Seq.Num	Interpretation
0	Disable the transmit sequence number
1	The first byte of the poll request must contain a number different than the last request to allow the updating of the scanner data field without generating erroneous data on the RS232 data.

Resend	Interpretation
0	Valid data is only sent once
1	Valid data is resent during subsequent Poll requests until a new string of valid data is received on the RS232 serial channel.

Sync	Interpretation
0	Do not apply synchronous Hand-shake protocol
1	Apply synchronous Hand-shake protocol

When a CDN066 is configured with both <string> formatting and sequence numbers the sequence number is applied as the first byte and the string length information is contained in the second data byte.

If the delimiter function is enabled the receive packet size may have an affect on the data responses. If the Post Delimiter field is zero the CDN066 will not transmit any response data until the delimiter character is detected or until <receive data size> bytes are available. If the receive data size is set less than the number of available characters the first poll response will contain the first <receive data size> bytes and the second poll response will receive the remaining characters up to delimiter.

If the Post Delimiter field is 1 the CDN066 will not transmit any response data until a delimiter is detected AND:

1) more than <receive data size> bytes have been received, or
2) another delimiter is detected

Characters in excess of the receive data size are discarded.

If the Resend bit is set the device will resend data on subsequent Poll requests until another valid data packet has been received.

The Sync bit enables the Synchronous Hand-shake protocol which provides further control over the sequence numbers during Poll Request/Response transactions to allow the Master to determine if a) a previous Poll Request packet has been accepted and b) the current Poll Response represents a new data string.

When the Sync bit is set to 1 the Xmt.Seq.Num and Rcv.Seq.Num bits will be forced to 1.

The Xmt.Seq.Num is received in the Poll Request and is interpreted as 2 four bit numbers:

Bit Numbers 4-7	Bit Numbers 0-3
RcceiveAcknowledgeNumber	TransmitRequestNumber

The TransmitRequestNumber acts in the same way as the Xmt.Seq.Num described above. The CDN066 will ignore any data in the Poll Request until the TransmitRequestNumber is different than previously received TransmitRequestNumber. If a value of 0 is received the current data (if any) will be ignored. A 0 acts as a 'reset' function for the TransmitRequestNumber.

The ReceiveAcknowledgeNumber is compared against the ReceiveRequestNumber (see below) and if equal it releases the current receive data buffer, allowing the CDN066 to send new information. A value of 0 will reset the ReceiveRequestNumber, acting as a reset function.

The Rcv.Seq.Num is transmitted in the Poll Response and is interpreted as 2 four bit numbers:

Bit Numbers 4-7	Bit Numbers 0-3
ReceiveRequestNumber	TransmitAcknowledgeNumber

The TransmitAcknowledgeNumber will be the same value as the most recently processed TransmitRequestNumber. When a poll request packet is received the TransmitRequestNumber is compared to the last TransmitAcknowledgeNumber and if different the data contained in the poll request is transmitted (see above). The TransmitRequestNumber is then transferred to the TransmitAcknowledgeNumber to notify the Master that the transaction has been processed.

The ReceiveRequestNumber is used by the CDN066 to indicate to the Master that the poll response contains new data. The CDN066 will increment the most previous ReceiveAcknowledgeNumber (see above) and return it in the poll response if new data is available. Note that the CDN066 will generate numbers in the range 1..15, reserving 0 as the reset value.

The Sync mode is typically used with a 'Scanner' that generates continuous poll requests. During the first poll request (possibly no valid data) the Xmt.Seq.Num should be set to a value of 00, resetting the receive handshaking logic on the CDN066. If no receive data is available the poll response will have the Rcv.Seq.Num set to ?0. If data is available the CDN066 will generate a poll response with the Rcv.Seq.Num set to ?1 with the associated data contained in the response packet. Further data will be buffered until the Scanner generates a poll request with a Xmt.Seq.Num with a value of ?1, acknowledging the receipt and processing of the previous poll. The Scanner should increment the ReceiveAcknowledgeNumber after processing each poll response, wrapping from 15 to 1.

During transmission, the scanner application code may build the request message in memory and then increment the TransmitRequestNumber (1..15). This allows the background scanner function to send 'partially complete' poll requests without generating extraneous RS232 transmissions. When the scanner application code detects that the TransmitAcknowledgeNumber received as part of the poll response matches the previous TransmitRequestNumber it indicates that the scanner has successfully transmitted the previous poll data and the application may proceed to build new RS232 transmit data.

This attribute adds 1 byte to the produce and consume size, and is only settable when the the poll connection is in the non-established state.

(16)The Delimiter character (Attribute 16) determines the start or end of packet character for the RS232 channel. It is only effective if the Delimiter Enable bit in the Block Control byte is set.

(17) The Pad Char (Attribute 17) is used to pad string formatted receive data. It is typically set to ASCII <space> (020H) or an ASCII <null> (0).

(18) The Maximum Transmit Size (Attribute 18) indicates the maximum number of data bytes to be transmitted across the RS232 channel. Setting this attribute will automatically reset the Poll Consumed Connection size as:

Connection size = Max Xmt Size (Maximum value 64)
+ 1(if Status Clear enabled)
+ 1 (if String Format enabled)
+ 1 (if Transmit Seq. Num. enabled)

The maximum connection size is 67 bytes.

This attribute affects the consume size, and is only settable when the the poll connection is in the non-established state.

(19) The Idle String (Attribute 19) will be transmitted on the RS232 serial channel if the device receives a 'receive_idle' (null Poll). If the string length by is set to 0 no data will be transmitted.

(20) The Fault String (Attribute 20) will be transmitted on the RS232 serial channel if the device experiences a connection timeout. If the string length by is set to 0 no data will be transmitted.

(21) The Status Enable (Attribute 21) inserts the serial status (Class 64 Instance 1 Attribute 5) as the first byte of the poll response when set to a non-zero value.
This attribute adds 1 byte to the produce size, and is only settable when the the poll connection is in the non-established state.

(22) The Status Clear Enable (Attribute 22) allows the first byte in the poll command to clear the status byte (when status clear byte, the first byte in the poll is not equal to 0) or not change the status (status clear byte, the first byte in the poll is =0).
This attribute adds 1 byte to the consume size, and is only settable when the the poll connection is in the non-established state.