W5-JDC4

Revision 1.02

4-Channel DeviceNet Serial Gateway

User's Manual

Western Reserve Controls

Although every effort has been made to insure the accuracy of this document, all information is subject to change without notice. Western Reserve Controls Inc. assumes no liability for any errors or omissions in this document or for direct, indirect, incidental or consequential damage resulting from the use of this document.

Document PUB 32.0
Rev 1.02
March 2003

Copyright © 2003 Western Reserve Controls

Western Reserve Controls
1485 Exeter Rd.
Akron, Ohio 44306

http://www.wrcakron.com

DeviceNet is a trademark of the Open DeviceNet Vendor Association ("ODVA").
All other trademarks are property of their respective companies.

TABLE OF CONTENTS

1 OVERVIEW 1

1.1 FEATURES 2
1.2 TYPICAL APPLICATIONS 3
1.3 BASIC OPERATION 3
1.3.1 Polled I/O 3
1.3.2 Cyclic Input Message 3
1.3.3 Change-of-State, or C.O.S. 4
1.3.4 Explicit Messages 4
1.4 MAJOR OPTION SELECTIONS 4
1.4.1 Maximum Transmit and Receive Characters Parameters 4
1.4.2 Master-Slave Handshake vs. Immediate Option 4

2 QUICK START 5

2.1 HOW TO INSTALL AND ESTABLISH DEVICENET COMMUNICATIONS 5
2.2 DEFAULT SETTINGS 6
2.3 HOW TO INSTALL A SERIAL NETWORK 7
2.4 HOW TO READ SERIAL DEVICE DATA FROM THE W5-JDC4 7
2.4.1 Example Assembly for Serial Receive 8
2.4.2 The Receive Max Character Length 8
2.5 HOW TO WRITE SERIAL OUTPUT DATA TO THE W5-JDC4 9
2.5.1 Example Assembly for Serial Transmit 9
2.5.2 The Transmit Max Character Length 9

3 GENERAL SPECIFICATIONS 10

4 HARDWARE INSTALLATION AND SET-UP 12

4.1 OVERVIEW 12
4.2 LED OPERATION 13
4.2.1 DeviceNet LEDs 13
4.2.2 Serial Port LEDs 14
4.3 SERIAL PORT CONNECTOR 14
4.4 ROTARY SWITCHES 15
4.5 DEVICENET CONFIGURATION 15
4.5.1 Network Termination 16
4.5.2 DeviceNet Connection Wiring 16

5 SOFTWARE CONFIGURATION AND SET-UP 18

5.1 DEVICE PARAMETERS 18
5.2 SETTING UP THE SERIAL LINK 18
5.3 RECEIVING SERIAL DATA FROM THE ASCII DEVICE 18
5.3.1 Overview 18
5.3.2 Setting up the Receive Character Buffer Length 18
5.3.3 Setting up and Using Pad Mode 19
5.3.4 Setting Up and Using the Swap Bytes Mode 20
5.3.5 Setting Up and Using Delimiter Operation 21
5.3.6 Receive String Data Type 21
5.3.7 Setting Up the Scanner I/O Receive Size 22
5.3.8 Explicit Messages to Receive the Serial Data String 22
5.3.9 Status Byte Description 22
5.4 TRANSMITTING SERIAL DATA TO THE ASCII DEVICE 25

5.4.1 Overview *25*
5.4.2 Setting up the Transmit Character Buffer Length *25*
5.4.3 Setting Up and Using the Transmit Delimiter *25*
5.4.4 Setting up and using the TX Byte Swap Mode *25*
5.4.5 Transmitting from the Master to the W5-JDC4 .... *26*
5.4.6 Transmit String Data Type *26*
5.4.7 Transmitting Serial Data *26*
5.4.8 Setting Up the Scanner I/O Transmit Size *26*
5.4.9 Master-Slave Handshake vs. Immediate Mode *27*
5.4.10 Explicit Messages to Transmit Serial Data String *28*
5.5 SETTING UP DEVICENET COMMUNICATIONS 28
5.5.1 Polled I/O *28*
5.5.2 Cyclic and Change-of-State I/O *28*
5.5.3 Setting up the DeviceNet I/O Connections *29*
5.5.4 Setting up the Connection Timer (EPR) *29*
5.5.5 Setting up the DeviceNet Baudrate *29*

6 ASSEMBLY OBJECT FORMATS 30

6.1 SHORT STRING DATA TYPE ASSEMBLIES 30
6.1.1 Immediate Mode *30*
6.1.2 Handshake Mode *30*
6.2 ARRAY (DATA BLOCK) DATA TYPE ASSEMBLIES 31
6.2.1 Immediate Mode *31*
6.2.2 Handshake Mode *31*
6.3 STRING DATA TYPE ASSEMBLIES 31
6.3.1 Immediate Mode *31*
6.3.2 Handshake Mode *31*

7 THEORY OF OPERATION 33

7.1 THE TRANSMIT RECORD ALGORITHM 33
7.1.1 Basic Theory of operation *33*
7.2 THE RECEIVE RECORD ALGORITHM 33
7.2.1 Basic Theory of operation *33*

8 DEVICENET PROFILE, OBJECTS AND SERVICES 36

8.1 W5-JDC4 DEVICENET PROFILE 36
8.2 IDENTITY OBJECT (CLASS 1) 37
8.3 DEVICENET OBJECT (CLASS 3) 38
8.4 ASSEMBLY OBJECT (CLASS 4) 40
8.4.1 Instance 101 (Consume) Assembly Object Description *40*
8.4.2 Instance 102 (Produce) Assembly Object Description *41*
8.4.3 Disabling channels *41*
8.5 CONNECTION (CLASS 5) OBJECT 42
8.6 ACKNOWLEDGE (CLASS 43) OBJECT 42
8.7 SERIAL PORT OBJECT CLASS 112 (70_{HEX}) 42
8.7.1 Serial Port Class Attributes *42*
8.7.2 Serial Port Class Services *42*
8.7.3 Serial Port Instance Attributes *42*
8.7.4 Instance Services *44*
8.8 TRANSMIT RECORD OBJECT CLASS 113 (71_{HEX}) 44
8.8.1 Transmit Record Object Class Attributes *44*
8.8.2 Transmit Record Object Class Services *44*
8.8.3 Transmit Record Object Instance Attributes *45*

8.8.4 Transmit Record Object Instance Services *47*
8.9 RECEIVE RECORD OBJECT CLASS 114 (72 $_{HEX}$) 47
8.9.1 Receive Record Object Class Attributes *47*
8.9.2 Receive Record Object Class Services *47*
8.9.3 Receive Record Object Instance Attributes *47*
8.9.4 Receive Record Object Instance Services *51*
8.10 COMMON DEVICENET SERVICES 51

TABLE OF FIGURES

FIGURE 1-1 W5-JDC4 1
FIGURE 4-1 W5-JDC4 OUTLINE DRAWING – TOP VIEW 12
FIGURE 4-2 W5-JDC4 OUTLINE DRAWING – RIGHT SIDE VIEW 13
FIGURE 4-3 DEVICENET CONNECTORS 17
FIGURE 5-1 RECEIVE ARRAY DATA FORMAT 22
FIGURE 5-2 RECEIVE SHORT_STRING DATA FORMAT 22
FIGURE 5-3 RECEIVE STRING DATA FORMAT 22
FIGURE 7-1 TRANSMIT RECORD ALGORITHM FUNCTIONAL FLOWCHART 33
FIGURE 7-2 RECEIVE RECORD ALGORITHM FUNCTIONAL FLOWCHART 34

LIST OF TABLES

TABLE 1-1 I/O MESSAGE TYPES....3
TABLE 2-1 INSTANCES OF THE SERIAL CHANNELS....5
TABLE 2-2 DEFAULT INPUT (SERIAL RECEIVE) ASSEMBLY FORMAT 8
TABLE 2-3 DEFAULT OUTPUT (SERIAL TRANSMIT) ASSEMBLY FORMAT 9
TABLE 4-1 MODULE STATUS LED (LABELED MS)....13
TABLE 4-2 NETWORK STATUS LED (LABELED NS) 14
TABLE 4-3 RS232 CONNECTOR SIGNALS....14
TABLE 4-4 BAUD RATE SWITCH (RATE) 15
TABLE 4-5 DEVICENET ADDRESS SWITCHES....15
TABLE 4-6 MAXIMUM NETWORK CABLE LENGTHS....16
TABLE 5-1 SERIAL STATUS BYTE....23
TABLE 5-2 W5-JDC4 CONSUME ASSEMBLY OBJECT WITH HANDSHAKE MODE 27
TABLE 6-1 IMMEDIATE MODE CONSUME SHORT STRING....30
TABLE 6-2 IMMEDIATE MODE PRODUCE SHORT STRING....30
TABLE 6-3 HANDSHAKE MODE CONSUME SHORT STRING....30
TABLE 6-4 HANDSHAKE MODE PRODUCE SHORT STRING....30
TABLE 6-5 IMMEDIATE MODE CONSUME DATA BLOCK 31
TABLE 6-6 IMMEDIATE MODE PRODUCE DATA BLOCK....31
TABLE 6-7 HANDSHAKE MODE CONSUME DATA BLOCK 31
TABLE 6-8 HANDSHAKE MODE PRODUCE DATA BLOCK....31
TABLE 6-9 IMMEDIATE MODE CONSUME STRING....31
TABLE 6-10 IMMEDIATE MODE PRODUCE STRING 31
TABLE 6-11 HANDSHAKE MODE CONSUME STRING....31
TABLE 6-12 HANDSHAKE MODE PRODUCE STRING 32
TABLE 8-1 DEVICENET OBJECTS....36
TABLE 8-2 IDENTITY OBJECT CLASS ATTRIBUTES (INSTANCE 0)....37
TABLE 8-3 IDENTITY OBJECT INSTANCE ATTRIBUTES (INSTANCE 1) 37
TABLE 8-4 IDENTITY OBJECT COMMON SERVICES 38
TABLE 8-5 CLASS 3 CLASS ATTRIBUTES....38
TABLE 8-6 CLASS 3 INSTANCE ATTRIBUTES....39
TABLE 8-7 INSTANCE 101 CONSUME DATA (ASCII TRANSMIT STRING)....40
TABLE 8-8 INSTANCE 102 PRODUCE DATA (ASCII RECEIVE STRING) 41
TABLE 8-9 CHANNEL 2 DISABLED....41
TABLE 8-10 CLASS 112 (70_{HEX}) SERIAL PORT OBJECT INSTANCE ATTRIBUTES....42
TABLE 8-11 CLASS 113 (71_{HEX}) TRANSMIT RECORD OBJECT INSTANCE ATTRIBUTES 45
TABLE 8-12 CLASS 114 (72_{HEX}) RECEIVE RECORD OBJECT INSTANCE ATTRIBUTES....47

1 Overview

The W5-JDC4 is a panel-mounted DeviceNet-to-serial link communications gateway that provides a flexible DeviceNet interface to as many as four different channels of ASCII devices. A wide variety of serial ASCII devices can be easily connected, with independent setups for each channel.

The W5-JDC4 does not interpret the data being transmitted across it, and so the transferred messages may contain data of any nature or definition. This allows you to use the same gateway for many different serial protocols.

Using the W5-JDC4 you may communicate with the connected peripheral devices in the same fashion as the other DeviceNet products in the system. Data may be read/written using either I/O or explicit messaging. Typically real-time data is read and written as I/O by the DeviceNet Master via Polled, Change-of-State or Cyclic I/O and parameters are read and written with the Explicit Messaging technique. However, you may also read and write serial data via explicit messages.

To read and write the serial ASCII devices connected to the serial ports, you read and/or write to all 4 channels simultaneously. You can use either an I/O Message or Explicit Message.

The W5-JDC4 is defined as a Communications Adapter device on the DeviceNet system. It has four 9-pin D-sub connectors for connection to the RS232 interface port on your devices and two 5-pin "micro" connectors for connections to the DeviceNet network. The W5-JDC4 has one assigned DeviceNet address, which is set by two 10-position rotary switches on the unit. There is also one rotary switch for the DeviceNet baud rate. Other W5-JDC4 parameters are software-configurable. Each W5-JDC4 has 2 standard green/red DeviceNet LED's for module status and network status and two green LED's for each serial port to indicate RS232 transmit and receive activity.

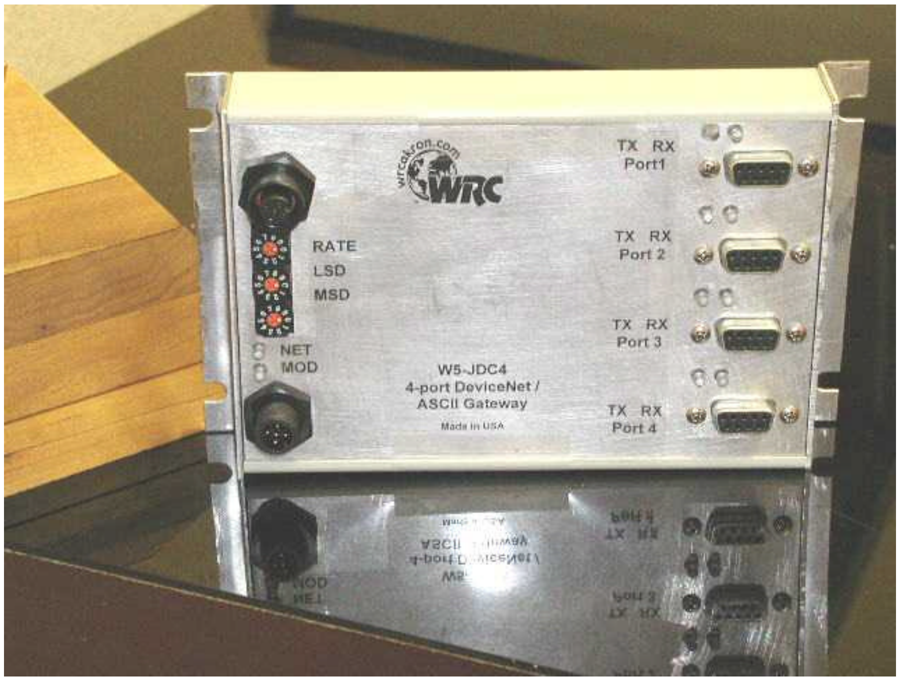


Figure 1-1 W5-JDC4

1.1 Features

The W5-JDC4 has the following features:

- Translates messages and data between DeviceNet and a serial peripheral device
- Up to 4 serial devices can be connected simultaneously
- ODVA Group 2 Only Slave
- ODVA Conformance tested to DeviceNet Spec 2.0
- Defined as a DeviceNet Communications Device Profile 12 (C_{hex})
- Autobaud operation
- I/O Messaging of Serial Data
  - Poll
  - COS
  - Cyclic
- Explicit Messaging
  - Serial data
  - Configuration data
- Pad mode option
- Byte swapping option
- Master-Slave handshake option
- Individual software configurable parameters for each serial port
- DeviceNet address and baud rate selection via DIP switches
- Panel mount
- 2 micro DeviceNet connectors – 1 male and 1 female
- 4 DB9 RS-232 Connectors – male (DTE)
- 2 standard DeviceNet module and network status LED's
- 8 serial transmit and receive LED's
- Powered from DeviceNet 11-25 Vdc network power
- ASCII string length up to 128 bytes
- Serial port baud rate up to 115.2k baud

1.2 Typical Applications

- Weigh scales
- Bar code readers and scanners
- Display panels
- Robots
- Drives
- Motion controllers
- Operator stations / HMI
- Magnetic code readers

1.3 Basic Operation

The W5-JDC4 operates as the DeviceNet front-end to the serial device(s). The DeviceNet Master can receive and send data to and from the W5-JDC4 via the methods described in this section. It sends the data to the device and likewise accepts responses from the device, which are passed back to the DeviceNet system as required.

The W5-JDC4 has one DeviceNet address (MacID). All DeviceNet messages to the W5-JDC4 itself (to read / write its internal data) are sent to this address. DeviceNet messages to and from the serial device can be sent to the W5-JDC4 DeviceNet assembly objects using either I/O or explicit messaging.This allows you to define the specific operation of each W5-JDC4. These objects include all the set-up required for the serial communications link.

The following chart and section defines the various messaging methods used for "typical" data types at your serial device and a brief explanation follows.

Table 1-1 I/O Message Types

Typical Data	Polled	Cyclic	Bit-Strobe	Change-of-State	Explicit Message
Commands	Ō	Ō		Ō	Ō
Status	Ō	Ō		Ō	Ō
Parameters					Ō

1.3.1 Polled I/O

The DeviceNet Master uses the W5-JDC4's predefined polled IO connection to send serial input and output data to the W5-JDC4. When a poll is received and the record has changed since the last poll was sent, the W5-JDC4 sends the associated transmit data out the serial port to the remote ASCII device. When the W5-JDC4 receives serial data from a device on the serial link, the poll response data to the Master contains up to 50 bytes of received data.

1.3.2 Cyclic Input Message

Cyclic I/O is the function by which a slave device sends its input data to the master at a specific time period without the host explicitly requesting it. When the specified time interval (defined by you) elapses, the most recent input data from the serial port are transmitted to the master. This data is the same format as a poll response.

1.3.3 Change-of-State, or C.O.S.

C.O.S. I/O is the function by which a slave device sends its input data to the master when defined input data changes without the host explicitly requesting it. In the case of the W5-JDC4, this occurs when the delimiter character is asynchronously received from the serial device, when the defined number of characters is received or when the internal buffer is filled. This data is the same format as a poll response.

1.3.4 Explicit Messages

Explicit messages are typically used to read and write configuration data. This data allows the W5-JDC4 to change its internal operating parameters such as baudrate and parity. In addition the user can use explicit messages to read and write the serial port data.

1.4 Major Option Selections

The W5-JDC4 has several different operating modes. Some of these are available only in certain combinations. These option selections are described here briefly and in more detail later.

1.4.1 Maximum Transmit and Receive Characters Parameters

These parameters are both defaulted from the factory to 20 for each channel. For any channel, setting both parameters to zero will cause the W5-JDC4 to fully eliminate the channel's header and data from the consume and produce assembly. See Section 8.4.3 for details.

1.4.2 Master-Slave Handshake vs. Immediate Option

If DeviceNet Master-Slave **Handshake Mode** is selected, the DeviceNet Master can inhibit the W5-JDC4 from sending new ASCII data until the Master is ready to receive and process the new data. This option is used only with a Poll I/O or Explicit Message.

The W5-JDC4 will indicate to the Master that new data is available by setting the New Data Flag in the Status byte of the Produce message. When the Master is ready to receive new serial data, it sets a new number in the new record number byte of the next poll command message. Note that this applies only to data being sent from the W5-JDC4 to the Master.

In **Immediate Mode** this handshaking is not active and the W5-JDC4 sends new data as soon as it is received from the ASCII device. (Default.) See Section 5.4.9 for details.

2 Quick Start

To quickly install your W5-JDC4 in your DeviceNet system, follow the instructions below. For more details, see Section 4. Where the *Instance* ***ch*** notation is used, **ch** is the serial **ch**annel.

Table 2-1 Instances of the Serial Channels

Class			Instance	Serial Channel
70 $_{hex}$(112)	71 $_{hex}$(113)	72 $_{hex}$(114)	1	1
70 $_{hex}$(112)	71 $_{hex}$(113)	72 $_{hex}$(114)	2	2
70 $_{hex}$(112)	71 $_{hex}$(113)	72 $_{hex}$(114)	3	3
70 $_{hex}$(112)	71 $_{hex}$(113)	72 $_{hex}$(114)	4	4

2.1 How to Install and Establish DeviceNet Communications

1. Connect your DeviceNet network cable to a 5-pin female (or male) micro-style connector according to DeviceNet cable wiring specifications
2. Make sure that the DeviceNet network is properly terminated.
3. The W5-JDC4 Node Address (MacID) is set to 63 at the factory. Make sure no other device on the network is set to 63, or change the W5-JDC4 address to one that is not currently used. See Section 4.4 for MacID rotary switch settings.
4. The W5-JDC4 baud rate is set to Autobaud operation at the factory. No baud rate setting is required. If a fixed baudrate is needed, set the baudrate rotary switch to the required baudrate. See Section 4.4 for actual settings.
5. Make sure that there is power on the DeviceNet network and plug the cable into the W5-JDC4.
6. The W5-JDC4 will undergo its initialization sequence, flashing both LED's red and green. After approximately 5 seconds, the Module Status LED (labeled "MS") will flash green. The Network Status LED (labeled "NS") will remain off. This condition occurs while the W5-JDC4 is attempting to synchronize to the network baudrate.
7. The Module Status LED ("MS") will go on solid after the Device successfully determines the network baudrate. This requires devices on the network attempting to communicate with each other. The Network Status LED (labeled "NS") will begin to flash green. If it turns solid red, check for a duplicate MacID on the network. It will remain off until the W5-JDC4 receives a valid DeviceNet message from which it will set its baud rate.
8. The W5-JDC4 is now operating on the network.
9. You may now map the W5-JDC4 into your scanner. The data format may be found in Sections 8.4.1 and 8.4.2.
10. Once the Master recognizes the unit on the link and allocates the connection (initiates communications), The Network Status LED will be solid green. The device is now being actively scanned.

2.2 Default Settings

The following list shows the default set-up for the parameters of each channel.

Serial Port	**Default Operation**
Serial Character Framing	7N2
Serial Port Comm Speed	9600 baud
Serial Port Receive from ASCII Device	**Default Operation**
Max Number of Receive Chars	20
Receive Record Start Mode	Not enabled
Receive Start Delimiter	Colon
Receive Record End Mode	Enabled and include with data string
Receive End Delimiter	Carriage return
Gateway Send (Produce) on DeviceNet to Master	**Default Operation**
Receive String Data Type	Short_string: data with preceding 1 byte length
Pad Mode	Enabled (1)
Pad Character	null character (0)
Receive Swap Mode	Off
DeviceNet Handshake Mode	Off
Serial Data	--
Actual Received Data Size	0
Receive Record Number	0
Serial Port Transmit to ASCII Device	**Default Operation**
Max Number of Transmit Chars	20
Transmit End Delimiter Mode	Include delimiter with
Transmit End Delimiter Character	Include
Gateway Receive (Consume) on DeviceNet from Master	Carriage return
Transmit String Data Type	Short_String: data with 1 preceding length byte
Transmit Swap Mode	Off
Record Header Mode	Active; record header bytes precede data string
Explicit Messages from EDS Editor	**Default Operation**

Actual Serial Data String to Send to ASCII Device	--
Transmit Serial Data Size	0
Transmit Record Number	0

2.3 How to Install a Serial Network

1. The communication between your serial device(s) and the W5-JDC4 is an RS232 3-wire network. Connect an appropriate cable to your device.
2. Connect the other end of the cable to the W5-JDC4 using the 9-pin DB9 connector. See Section 4.3.
3. Turn on power to the serial device and the W5-JDC4.
4. Set up the ASCII buffer sizes on the W5-JDC4. (The defaults are 20 and 20). If more than 20 bytes are required for the transmit or receive buffers, set the appropriate parameters in your configuration file to the buffer size you need for your ASCII data.

   NOTE: This will modify the IO message size. You will need to reconfigure the poll / C.O.S. / cyclic transmit and receive data sizes if you modify the ASCII buffer size from the default value. In many configuration tools, this will un-map the data in your scanner's scan table. They must be remapped in order to be able to process the data in your PLC or PC software. These values are displayed in Table 8-11 and Table 8-12.

2.4 How to Read Serial Device Data from the W5-JDC4

1. Connect to the W5-JDC4 from your DeviceNet Configuration Manager tool.
2. Connect the serial side of the W5-JDC4 to your computer's serial port or another serial device.
3. Go to the device configuration screen in the Configuration Manager.
4. Set the baudrate and framing format of the serial port to the baudrate and framing format of the serial device that you are using.
5. Put the Configuration tool in to monitor mode.
6. Direct the device that you are communicating with to send data. For example, if you are connected to a computer terminal program, type a message into the terminal. When you hit enter, the module will update the data with the message that you typed, and increment the record number.
7. The default assembly format of the poll message is shown below.

2.4.1 Example Assembly for Serial Receive

Table 2-2 shows the produced assembly for 20 characters of ASCII data (default) with padding enabled.

Table 2-2 Default Input (Serial Receive) Assembly Format

Byte 0	**Byte 1**	**Byte 2**	**Byte 3**	**Byte 4 to 23**
Record Number Channel 1	Status	Reserved	Length	ASCII Data
Byte 24	**Byte 25**	**Byte 26**	**Byte 27**	**Byte 28 to 47**
Record Number Channel 2	Status	Reserved	Length	ASCII Data
Byte 48	**Byte 49**	**Byte 50**	**Byte 51**	**Byte 52 to 71**
Record Number Channel 3	Status	Reserved	Length	ASCII Data
Byte 72	**Byte 73**	**Byte 74**	**Byte 75**	**Byte 76 to 95**
Record Number Channel 4	Status	Reserved	Length	ASCII Data

Note: The default data type is Short_String, which includes one length byte.

2.4.2 The Receive Max Character Length

The parameter is found in the Receive Record Object (Class **114**, Instance **ch**, Attribute **7**). Changing this parameter to zero (0) will eliminate only the data, the header will still be produced and sent to the master.

2.5 How to Write Serial Output Data to the W5-JDC4

1. Do steps 1-6 of Section 2.4 above.
2. Each channel has a factory default of 20 characters maximum. If more is needed, see section 8.9.3.7 and 8.8.3.5 for details.
3. Enter the serial data that you wish to send in the transmit data parameter. See Section 8.8.3.3
4. Change the Length of the data in the length byte to reflect the length you wish to send.
5. Change the Record Number.
6. The W5-JDC4 will generate the characters that you typed in on the computer screen.
7. The assembly formats of these messages are configurable and are covered in Section 6.

2.5.1 Example Assembly for Serial Transmit

Table 2-3 shows the produced assembly for 20 characters of ASCII data (default) with padding enabled.

Table 2-3 Default Output (Serial Transmit) Assembly Format

Byte 0	**Byte 1**	**Byte 2**	**Byte 3**	**Bytes 4 to 23**
Reserved	Record Number Channel 1	Reserved	Length	ASCII Data
Byte 24	**Byte 25**	**Byte 26**	**Byte 27**	**Bytes 28 to 47**
Reserved	Record Number Channel 2	Reserved	Length	ASCII Data
Byte 48	**Byte 49**	**Byte 50**	**Byte 51**	**Bytes 52 to 71**
Reserved	Record Number Channel 3	Reserved	Length	ASCII Data
Byte 72	**Byte 73**	**Byte 74**	**Byte 75**	**Bytes 76 to 95**
Reserved	Record Number Channel 4	Reserved	Length	ASCII Data

Note: The default data type is Short_String, which includes one length byte.

Note: If using a delimiter, you must use the actual length of the string, including the delimiter, for the data size, whether or not it fills the entire data block.

2.5.2 The Transmit Max Character Length

The parameter is found in the Transmit Record Object (Class **113**, Instance **ch**, Attribute **5**). Changing this parameter to zero (0) will eliminate only the data, the header will still be consumed and should be sent by the master.

3 General Specifications

Product:	W5-JDC4 Device-Serial Gateway
Description:	Communications gateway between a serial capable device over an RS232 or RS485 interface and a DeviceNet network.
Device Type:	Communications Adapter, 12 (0C$_{hex}$)
Device Profile:	Identity Object Message Router Object DeviceNet Object Assembly Object – 2 instances Connection Object Acknowledge Object Serial I/O Object (vendor-specific) – 4 instances Transmit Serial Object (vendor-specific) – 4 instances Receive Serial Object (vendor-specific) – 4 instances
Product Revision:	1.07
Product Code:	730 (2DA$_{hex}$)
Vendor ID:	9 Western Reserve Controls Inc.
DeviceNet Conformance:	Designed to conform to the ODVA DeviceNet Specification Volume I and II, Version 2.0.
DeviceNet Communications:	Predefined Master/Slave Connection Set, Group 2 Only Server
DeviceNet:	**Baud rate selection:** 125k, 250k and 500k baud, and autobaud – switch selectable **Address selection:** Address number 0 to 63 – switch selectable (default = 63) **Cable Connection**: W5-JDC4: 2, 5-contact standard DeviceNet micro connectors (one male, one female) DeviceNet Cable: 5-contact standard DeviceNet micro connectors **Status Indicators:** Module Status: green/red bi-color LED Network Status: green/red bi-color LED
Serial port:	**Baud rate**: 1200, 2400, 4800, 9600, 19.2k, 38.4k, 57.6k, 115.2k baud (software selectable) **Parity**: Odd/even/none (software selectable) **Data bits**: 7 or 8 (software selectable)

	Serial port connection: W5-JDC4: DB9-M (male, 9-pin D-sub) Amp p/n 745182-2 or equivalent **Status Indicators**: Transmit Active: green LED Receive Active: green LED
Network Isolation:	500V
Max Power:	3.75 watts: 340 mA @ 11 Vdc – 150 mA @ 25 Vdc unregulated power supply
Mounting:	DIN rail mount, EN 50022
Size:	• Length: 6.00" (152,4 mm) • Width: 4.00" (101,6 mm) • Height: 2.08" (52,7 mm)
Operating Temp:	0-60 ^{o}C
Humidity:	0-95% RH, non-condensing

4 Hardware Installation and Set-Up

4.1 Overview

The W5-JDC4 consists of an IP20, aluminum, panel-mounted enclosure.

The W5-JDC4 contains two LED's to indicate the status of the device and the status of the network. The device can be connected to the main DeviceNet trunk line or to a drop line via a 5-pin female plug-style connector. It also has eight (8) green LED's to indicate the presence of activity on the four (4) sets of RS-232 transmit and receive lines.

All power for the W5-JDC4 is derived from the DeviceNet power.

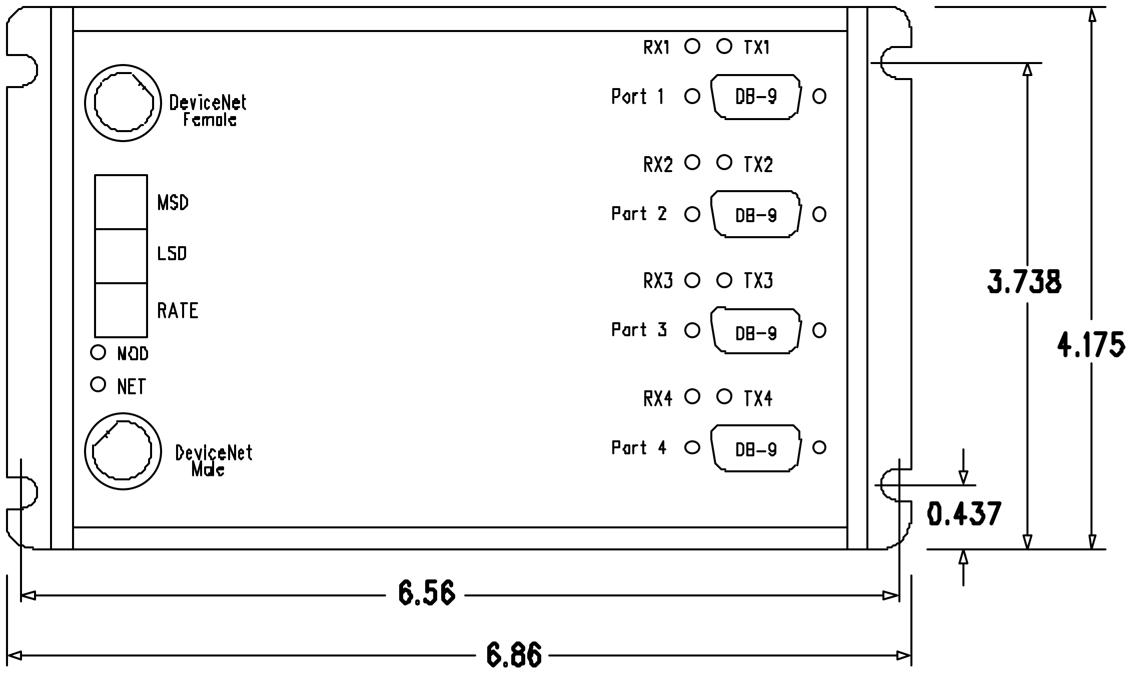


Figure 4-1 W5-JDC4 Outline Drawing – Top View

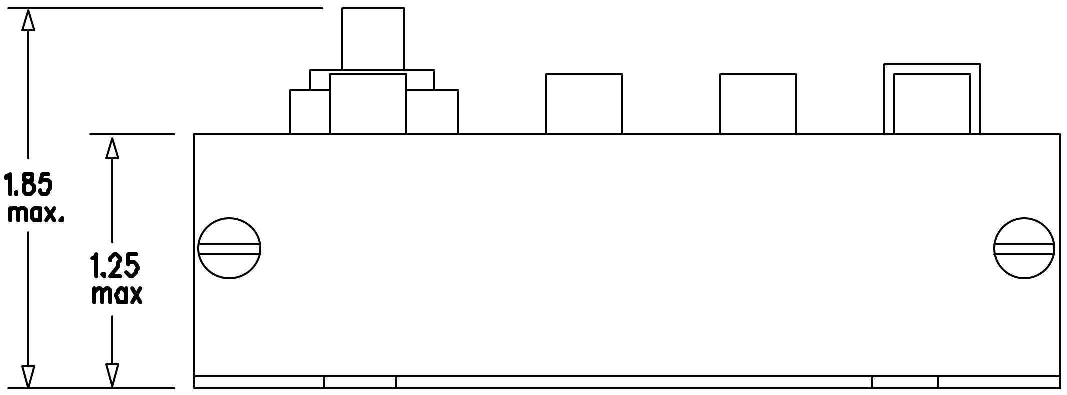


Figure 4-2 W5-JDC4 Outline Drawing – Right Side View

4.2 LED Operation

4.2.1 DeviceNet LEDs

The W5-JDC4 has two LEDs that provide visual status information to the user about the product and the DeviceNet network. See Tables 5-1 and 5-2 that follow below for how to interpret LED status indications.

Table 4-1 Module Status LED (labeled MS)

LED State	Module Status	Meaning
OFF	No Power	There is no power through DeviceNet.
Green	Device Operational	W5-JDC4 is operating normally.
Flashing Green	Device in Standby	W5-JDC4 needs commissioning (e.g. attempting autobaud).
Flashing Red	Minor Fault	Recoverable fault.
Red	Unrecoverable Fault	W5-JDC4 may need replaced.
Flashing Red/Green	Device Self-Testing	W5-JDC4 is in self-test mode.

Table 4-2 Network Status LED (labeled NS)

LED State	Network Status	Meaning
OFF	No Power / Not on-line	W5-JDC4 has no power or has not completed the Dup_MAC_ID test.
Flashing Green	On-line, not connected	W5-JDC4 is on-line but is not allocated to a Master.
Green	On-line	W5-JDC4 is operating normally.
Flashing Red	Connection time-out	One or more I/O connections are timed out.
Red	Critical link failure	W5-JDC4 has detected an error that makes it incapable of communicating on the link. (Bus off or Duplicate MAC ID).

4.2.2 Serial Port LEDs

The W5-JDC4 has two (2) RS-232 activity LEDs for each of the 4 channels: one for transmit (TX) and one for receive (RX). These LEDs are electrically tied to the serial data lines and will illuminate when there is data signals active on the respective data lines and the W5-JDC4 has power.

4.3 Serial Port Connector

The ASCII devices are connected to the W5-JDC4 via a 3-wire communications cable. See your ASCII device's User Manual for details on the proper connections.

The RX and TX designators are referenced with respect to the W5-JDC4.

Table 4-3 RS232 Connector Signals

DB9 Pin #	RS232 Designator	RS232 Signal
2	TX	Transmit
3	RX	Receive
5	GND	Ground

Note: The RS232 max distance spec is 50 feet (15m).

4.4 Rotary Switches

There are 2 10-position rotary switches for device address (MacID) and 1 10-position rotary switch for DeviceNet baud rate.

Table 4-4 Baud Rate Switch (RATE)

Position	Definition
0	125k baud
1	250k baud
2	500k baud
3-9	Use values stored in non-volatile memory.

Table 4-5 DeviceNet Address Switches

MSB Position	LSB Position	Address
0	0	0
0	1	1
•	•	•
6	3	63
6	4-9	Use Values stored in non-volatile memory
>6	0-9	

4.5 DeviceNet Configuration

DeviceNet specifications provide for a maximum network distances for the main trunk line and drop lines, depending upon the baud rate used on the network. See Table 4-6

Table 4-6 Maximum Network Cable Lengths

	Trunk Line Length		Drop Length			
	Maximum Distance		Maximum		Cumulative	
Baud Rate	**Meters**	**Feet**	**Meters**	**Feet**	**Meters**	**Feet**
125k baud	500 m*	1640 ft	6 m	20 ft	156 m	512 ft.
250k baud	250 m*	820 ft	6 m	20 ft	78 m	256 ft.
500k baud	100 m	328 ft	6 m	20 ft	39 m	128 ft.

*Note: Thick cable only. Thin cable is 100m max at any baud rate.

4.5.1 Network Termination

A DeviceNet system **must be terminated at each end of the trunk line**. The host controller and the **last** W5-JDC4 or other DeviceNet device on the network must always be terminated to eliminate reflections, even if only two nodes are present. The DeviceNet specifications for the terminating resistor are:

- 121 ohm
- 1% metal film
- 1/4 Watt

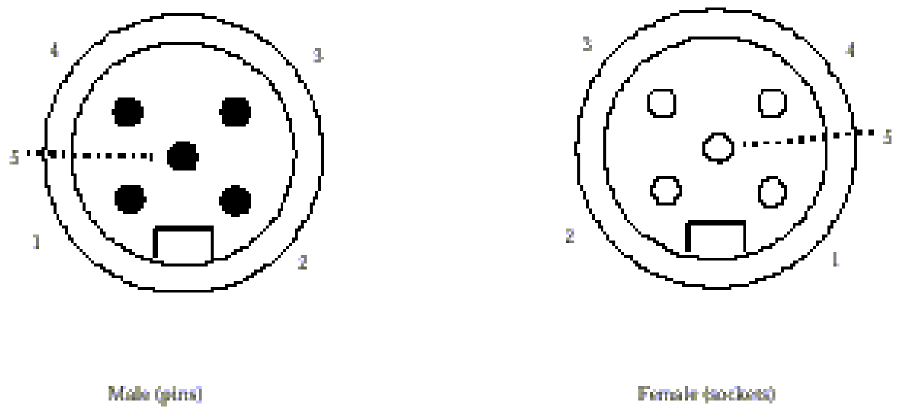
IMPORTANT: **Per the DeviceNet spec -- do not terminate devices on drop lines.**

NOTE: If you feel you are having DeviceNet communications errors, check your network terminations.

With power removed from the network, measure the dc resistance with an ohmmeter. It should measure ~60 ohms. If it measures 121 ohms, add another 121 ohm terminating resistor. If it measures ~40 ohms, then you have 3 terminators and must remove one.

4.5.2 DeviceNet Connection Wiring

The W5-JDC4 uses two (2) 5-pin standard micro-style DeviceNet connector, one of which has male pins and one of which has female sockets. These connectors are physically connected internally to the device.



1 - Drain	bare
2 - V+	red
3 - V-	black
4 - CAN_H	white
5 - CAN_L	blue

Figure 4-3 DeviceNet Connectors

5 SOFTWARE Configuration and Set-Up

The W5-JDC4 is an easy device to set up and configure. Using features like the EDS sheets for configuration can expedite the process if you use a network configuration tool that supports them. They provide a graphical interface to the device's parameters and allow the addition of helpful text descriptions in setting up your device. The current EDS file is available on our website, WRC web site. If your configuration tool does not support the EDS device profiles, the set up of a DeviceNet device requires a little more understanding of DeviceNet and its operation. This section is designed to fully describe the features of the W5-JDC4 and to help you set them up.

Note: Refer to Section 2.2 above for a list of the default settings for all the configurable parameters.

5.1 Device Parameters

The operation of each of the 4 serial channels is identical in function and independent in operation from each other.

5.2 Setting Up the Serial Link

The first two Parameters in each channel group allow you to define the serial link communications options. You must set up each channel individually to match the baud rate and framing characteristics of the particular serial device connected to the W5-JDC4.

5.3 Receiving Serial Data from the ASCII Device

5.3.1 Overview

The W5-JDC4 receives a number of characters and transmits these to the DeviceNet Master via

- I/O – poll, COS, Cyclic
- Explicit Message

The received character string is captured when

- the specific number of bytes defined (Receive Character Buffer Length) is received, or
- the defined End-of-String Terminator character is detected.

When either of these events occur the W5-JDC4 stores the received message string into its internal buffer and will then transmit (Produce) it onto DeviceNet at the next poll request or appropriate opportunity.

Note: Instance **ch** = channel number 1,2,3, or 4

5.3.2 Setting up the Receive Character Buffer Length

The receive character buffer length is the number of characters that the W5-JDC4 can receive from your I/O device into its buffer at one time. The length of the data string sent to the DeviceNet Master is less than or equal to this size, plus 4 for the header size.

If the W5-JDC4 receives more characters that this number, it will internally generate an overflow and force the data into the W5-JDC4 DeviceNet transmit buffer to be sent to the Master. The subsequent received characters will then be received into the buffer and handled as the start of next incoming message string. The overflow bit in the status byte will be set as well.

Caution: Incoming characters could be missed in the process of handling a string longer than the defined max length.

This value can be set and retrieved by using the standard set and get services on Class **114**, instance **ch**, attribute **7**.

5.3.3 Setting up and Using Pad Mode

Pad Mode operation is the method used by the W5-JDC4 that adds extra characters to the end of its received data string (after the delimiter character) from the external I/O device before sending the string to the DeviceNet scanner (Master) as an I/O Response. The quantity added is such that the data string returned to the scanner is always a constant length, and that length is the number specified in the receive_character_length parameter (See Section 8.9.3.5) plus 4 bytes for header. The quantity of pad characters sent can vary from message to message, depending upon the size of the incoming string.

5.3.3.1 Pad Mode Selection

Pad mode is included with our device for compatibility with Scanners that cannot **receive** variable length I/O messages. (Notable examples include many Allen-Bradley's scanners at the time of this printing). For such scanners, Pad mode must be ON (a value of 1). Turning Pad mode ON will not harm scanners that do support variable length receive messages. The default value for Pad Mode is ON. If your scanner does support variable I/O messaging lengths, you may turn the Pad Mode option OFF (a value of 0) to conserve some network bandwidth. Using pad mode will also make received data easier to parse since the data position in memory will be pre-determined.

The selection of Pad Mode is valid only for the DeviceNet message that the W5-JDC4 produces. It has no effect on DeviceNet messages sent from the Scanner to the W5-JDC4. To select or deselect pad mode, use Class **114**, Instance **ch**, Attribute **5**.

5.3.3.2 Pad Mode Character

The W5-JDC4 allows you to specify the character that pad mode uses to pad the received serial data. This can be set to any valid I/O value (0-127 in 7 bit modes, 0-255 in 8 bit modes). To change pad character, use Class **114**, Instance **ch**, Attribute **6**.

5.3.4 Setting Up and Using the Swap Bytes Mode

This option may be helpful if the W5-JDC4 is connected to a DeviceNet scanner that organizes the data string characters into data type elements that are larger than 1 byte each. An example is many Allen Bradley PLC's, such as the SLC500. In such cases the bytes of the data in the Master's memory organization can be reversed from the order in which they are sent or received on the DeviceNet and the serial link to the ASCII device. This may cause problems in some cases.

Thus, the message received or desired "ABCDEFGH" string may appear in memory as "BADCFEHG" for 2-byte (16 bit) word organization, and "DCBAHGFE" for 4-byte (32 bit) word organization.

5.3.4.1 Transmit Byte Swapping

If Class **113**, Instance **ch**, Attribute **9** is non-zero, the bytes from the Master will be swapped by the W5-JDC4 before transmitting the string to the ASCII device.

5.3.4.2 Receive Byte Swapping

If Class **114**, Instance **ch**, Attribute **12** is non-zero, the W5-JDC4 will re-order the bytes received from your ASCII device before sending the string to the Master.

5.3.4.3 Rules for Usage

1. This feature is set for both transmit and receive independently.
2. The Byte Swapping works better if the string length is an even multiple of the byte-swap size.
3. If a delimiter is received, then
   - All characters up to and including the defined delimiter are sent to the DeviceNet Master.
   - If Pad Mode = 1, then the W5-JDC4 will fill the Poll Response data with the Pad Char up to the defined size.
   - If Pad Mode = 0, then the W5-JDC4 will send only the data up to and including the delimiter
4. If no delimiter is received, then
   - The W5-JDC4 will receive up to Max_Number_of_Receive_Chars, and then send this string to DeviceNet with an overflow error.
   - It will continue to receive and send strings of size Max_Number_of_Receive_Chars, along with the overflow error, until a delimiter is received. This could continue indefinitely if your I/O device does not transmit the specified delimiter.

5.3.5 Setting Up and Using Delimiter Operation

When receiving data strings from your serial device, the W5-JDC4 can take advantage of both Start and Stop (End) delimiters. The **Start Delimiter is the start-of-string indicator** and the **End Delimiter is the end-of-string indicator.** This allows you further control over exactly which characters are sent to the Master.

When you select **Start Delimiter** operation, you define a character that prompts the W5-JDC4 to start storing the incoming data string. All characters up to this Start Delimiter (after the previous message was completed) are ignored. Once the Start Delimiter is received, all characters are stored until either the **End Delimiter** is received or the Max Receive Char Length is reached. Once the End Delimiter is reached, the data string is captured and prepared to send to the DeviceNet Master.

If either delimiter is used, you also can elect whether or not to include those characters in the string sent to the Master.

5.3.6 Receive String Data Type

This is the format of the data – array, short_string or string – you will send to the DeviceNet Master. These are shown below: Which one you pick depends on your application, and will modify the format of the data field.

This is the format of the data you will send to the W5-JDC4 – array, short_string or string. These are shown below: Which one you pick depends on your application, and will modify the format of the data field.

The **Array** data type does not have a length associated with it. It is equivalent to specifying a length of zero using a string or short string data type.

The **Short_String** data type is the default data type of the device. This will suffice for most applications. The Short_String data type has only one byte of length, and the rest of the data bytes are appended after the length.

The **String** data type has two bytes of length. The String data type is useful in talking to some PLC's or other devices that have a data file specifically made to handle this data type. The length is little endian (low byte, high byte), and the high order byte should always be set to zero. The W5-JDC4 will only receive up to 128 bytes of information, so the extra byte, although required for this data type, is always 0.

Note: This data length information is never sent to the ASCII device. Therefore, if the Short_String or String data type is used, the W5-JDC4 uses this information internally and does *not* send it to the ASCII device.

Data Byte 1	**Data Byte 2**	. . . . .	**Data Byte N**

Figure 5-1 Receive Array Data Format

Length Byte	**Data Byte 1**	**Data Byte 2**	. . . . .	**Data Byte N**

Figure 5-2 Receive Short_String Data Format

Len. Byte 0	**Len. Byte 1**	**Data Byte 1**	**Data Byte 2**	. . . . .	**Data Byte N**

Figure 5-3 Receive String Data Format

5.3.7 Setting Up the Scanner I/O Receive Size

The W5-JDC4 automatically calculates the number of bytes it will send the DeviceNet Master. Its value is determined by a combination of the incoming data and the options you have selected. The Produce size in the DeviceNet Object reflects the size of the DeviceNet message to be sent by the W5-JDC4 to the Master.

IMPORTANT: If you are using a Scanner that must receive a constant message length (such as an A-B xxxx-DNB scanner), you must set its input value to this number of bytes.

5.3.8 Explicit Messages to Receive the Serial Data String

Class **114**, Instance **ch**, Attribute **2**,**3**,**7**, and **14** contain the record number, data, size, and status of the most recent incoming serial data string. You can use the data to read your device's ASCII data via the Explicit Messaging technique.

Attribute **3** holds the most recent received data.

As explained in Section 5.3.6, Attribute **7** defines the size, in bytes, of the DeviceNet message to be sent by the W5-JDC4 to the Master.

Attribute **2** holds the record number of the data string in Attribute **3**.

5.3.9 Status Byte Description

The status byte is an OR'd bitfield of a number of status and exceptions.

Table 5-1 Serial Status Byte

Bit	Exception
0	TX FIFO Overflow
1	Rx FIFO Overflow
2	Rx Parity Error
3	Data in TX FIFO
4	Data in RX FIFO
5	Non-Delimited Record
6	Handshake Error
7	New Data Flag

5.3.9.1 TX FIFO Overflow

The transmit queue has overflowed resulting in a loss of data. The transmit I/O is full of data waiting to be transmitted. Some of the data added has been lost. When space becomes available in the TX I/O, this bit will be reset.

5.3.9.2 Rx FIFO Overflow

The receive queue has overflowed resulting in a loss of data. The receive I/O is full of data waiting to be processed. The data has been lost. When space becomes available in the RX I/O, this bit will be reset.

5.3.9.3 Rx Parity Error

If this bit is set, then a parity error occurred while processing the current record.

5.3.9.4 Data in TX FIFO

If this bit is set, there is data in the transmit FIFO.

5.3.9.5 Data in RX FIFO

If this bit is set, there is data in the receive FIFO.

5.3.9.6 Non-Delimited Record

This bit signifies the currently displayed record was not the result of encountering a delimiter. This bit does not necessarily specify an error. It is up to the application to determine what the source of the error is. This bit, along with the length specifier, creates an easy source to determine what caused the currently shown record. If this bit is not set, a delimiter encountered in the parsed ASCII data caused the record change. If this bit is set, you must compare the length byte to the maximum length. If the Maximum length is equal to the length of the buffer, then the record was caused by a fill event. If the length is not equal to the maximum length parameter, then the record was caused by a timeout event.

5.3.9.7 Handshake Error

This error will occur only in Master-Slave Handshake Mode. It indicates that the Master has requested a new data record from the W5-JDC4, but the W5-JDC4 has not indicated new data is available to be sent.

5.3.9.8 New Data

This bit is used only when the Master-Slave Handshake option is active. When the W5-JDC4 receives a new data string into its serial port, it sets this flag in its DeviceNet response message. The bit will remain set for 2 produce messages after the Master requests the new data. It then will be reset.

5.4 Transmitting Serial Data to the ASCII Device

5.4.1 Overview

The W5-JDC4 transmits a number of characters from the DeviceNet Master to your serial device via

- I/O Messaging
- Explicit Messaging

The received character string is transmitted when

- The specific number of bytes defined (Transmit Character Buffer Length) is received, or
- The defined End-of-String Terminator character is detected.

When either of these events occurs the W5-JDC4 stores the DeviceNet string data into its internal buffer and will then transmit it out its serial port.

In order to transmit data to your serial device, the data must first be sent to the W5-JDC4 and then the W5-JDC4 must send the data to the serial device. The options for transmitting from the Master to the W5-JDC4 are discussed first.

Note: Instance **ch** = channel number 1,2,3, or 4

5.4.2 Setting up the Transmit Character Buffer Length

The Transmit character buffer length is the number of characters that the W5-JDC4 can receive in its transmit buffer from the DeviceNet system. This size contributes to the I/O Consume Size. This size can be found in Class **113**, Instance **ch**, Attribute **5**.

5.4.3 Setting Up and Using the Transmit Delimiter

The transmit delimiter is an **end-of-string** character which is used by the W5-JDC4 to determine how many bytes to transmit over the serial link to your W5-JDC4 device. This Transmit Delimiter will be used if the Transmit Buffer Length equals 0. If the buffer length is not 0 the W5-JDC4 will ignore the transmit delimiter.

The W5-JDC4 will transmit up to and including the delimiter when the above condition is met. The transmit delimiter can be set to any valid I/O character that can be received over the link. Be very careful not to set the delimiter to a value outside of the valid range for your data bits (Note: A data bit size setting of 7 will only allow you a delimiter range of 0-127 dec., 00-7Fhex). If you do not have a valid delimiter, or the delimiter is never received, the device will only update the output buffer on detection of an overflow condition. These values can be set and retrieved by using the standard set and get services on Class **113**, Instance **ch**, Attributes **6** and **7**.

5.4.4 Setting up and using the TX Byte Swap Mode

This option may be helpful if the W5-JDC4 is connected to a DeviceNet scanner that organizes the data string characters into data type elements that are larger than 1 byte each. An example is many Allen Bradley PLC's, such as the SLC500. In such cases the bytes of the data in the Master's memory organization can be reversed from the order in which they are sent or received on the DeviceNet and the serial link to the ASCII device. This may cause problems in some cases.

See Section 5.3.4 above, Setting Up and Using the Swap Bytes Mode.

5.4.5 Transmitting from the Master to the W5-JDC4

You can transmit data to the W5-JDC4 from the Master using 2 methods – Poll I/O or Explicit Messages. Both require understanding and setting up some parameters.

5.4.6 Transmit String Data Type

This is the format of the data you will send from the Master – **array**, **short_string** or **string** – to the W5-JDC4. These are shown below: Which one you pick depends on your application, and **will modify the format of the data field**.

See Section 5.3.6 above, Receive String Data Type for details.

5.4.7 Transmitting Serial Data

The length of the string set determines the use of a delimiter in transmitting data to a serial device from the W5-JDC4.

If the string length is zero, or the data type is type Array:

The W5-JDC4 receives data sent from the DeviceNet Master and **uses the delimiter** to determine how much data to send to the serial device. The W5-JDC4 will compute the length and then store this as the new length in the string attribute. (This will not show up if the data type is array, you will just see the string truncated, and the length will be in the background)

If a delimiter is contained within the string, then

- All characters up to and including the defined delimiter are stored.

If no delimiter is contained within the string, then

- The W5-JDC4 will store all the data received.

If the string length > 0 or the data type is String or Short_String,

The W5-JDC4 receives data sent from the DeviceNet Master ignoring any embedded terminator. It will store the number of characters defined in Max_Number_of_Transmit_Chars, or the total sent by the Master, whichever is less.

Now, the W5-JDC4 will send the data immediately if Handshake Mode parameter Class **113**, Instance **ch**, Attribute **8** is set to 1 (See below). You can always cause this data to be transmitted by incrementing the record counter.

5.4.8 Setting Up the Scanner I/O Transmit Size

The W5-JDC4 automatically calculates the number of bytes it will receive from the DeviceNet Master. Its value is determined by a combination of the incoming data and the options you have selected. The Consume size in the DeviceNet Object defines the size of the DeviceNet message to be sent to the W5-JDC4 from the Master and should be set as the Output size in your Scanner's I/O set-up.

IMPORTANT: You must set your scanner's Output value to this number of bytes.

5.4.9 Master-Slave Handshake vs. Immediate Mode

If DeviceNet Master-Slave **Handshake Mode** is selected, the DeviceNet Master can inhibit the W5-JDC4 from sending new ASCII data across DeviceNet to the Master until the Master is ready to receive and process the new data. This option is only with a Poll I/O or Explicit Message.

In this mode there two data required for the complete transaction:

- A "New Data Available" Flag is set by the W5-JDC4 in the status byte.
    - This informs the Master that the W5-JDC4 has received a new data string and is waiting for the OK to send it. See Table 5-1.
- An additional "Ready for New Data" byte is pre-pended to the message the Master sends to the W5-JDC4. (The W5-JDC4's Consume Object.)
    - This New Data byte is used to indicate to the W5-JDC4 that the Master is ready to receive the new data. (The W5-JDC4's Produce Object does not change format.)

Table 5-2 W5-JDC4 Consume Assembly Object with Handshake Mode

Byte 0	Byte 1	Byte 2	Byte 3	Bytes 4-X (X < 54)
New Data Record Byte	Record Number Byte	Reserved	Length	ASCII Data

The Master monitors this new data flag and when the Master is ready to receive new serial data, it sets a new number in the new record number byte of the next poll command message. Note that this applies only to **data being sent from the W5-JDC4** to the Master.

The operation proceeds as follows:

1. The W5-JDC4 receives a new data string
2. The W5-JDC4 sets the New Data Flag in the Status byte of its next produce message.
3. The Master sends out messages to the W5-JDC4 in the normal fashion. If the Master is not ready to receive new data, the New Record Data byte remains constant.
4. When the Master is ready to receive the new data string, it changes the New Record Data byte to any value different than what it had been sending.
5. The W5-JDC4 will send the new data upon receipt of a record from the Master in which the New Record Data byte has been changed.
6. If the W5-JDC4 receives an updated New Record Data byte and has no new, it will set the Handshake Error bit in its Produce Status byte.

In **Immediate Mode,** this handshaking is not active and the W5-JDC4 sends new data as soon as it is received from the ASCII device. It is the Master's responsibility to be ready to accept and process the new data string when it is presented.

5.4.10 Explicit Messages to Transmit Serial Data String

You can use Class **113**, Instance **ch**, Attributes **2** and **3** to set up the serial data string and send it to your ASCII device via the Explicit Messaging technique.

Attribute 3 will hold the data you wish to send.

Attribute 2 holds the record number of the data string in Attribute 3. Changing the record will cause the data in Attribute 3 to be transmitted immediately from the W5-JDC4 to the ASCII device.

5.5 Setting up DeviceNet Communications

The W5-JDC4 supports 4 modes of data transfer of the serial buffer. They are:

- Polled I/O
- Change-of-State I/O
- Cyclic I/O
- Explicit Message

5.5.1 Polled I/O

The polled connection is the only manner in which you can send serial output data to the I/O and, therefore, to your I/O device. The DeviceNet Master initiates the polled connection transfer. The Master sends the W5-JDC4 its serial output buffer along with a Record Number and length byte. The W5-JDC4 monitors the Record Number and if the Record Number changes, then the W5-JDC4 transmits the data buffer on its serial link. If the Record Number does not change, then the device does not transmit the data buffer.

After the device has transmitted its data out to the serial link, the W5-JDC4 then takes any information that is stored in its current serial input buffer and sends this data to the DeviceNet Master. It sends all characters up to and including the received delimiter, padding the remaining bytes if the serial string is smaller than the maximum receive bytes for that channel.

When the W5-JDC4 receives a new message (either with a delimiter or with an overflow condition without a delimiter) the device then increments the receive record, updates the length byte, and copies the new information from the last receive delimiter into the buffer. If an overflow occurs, the W5-JDC4 indicates so in its receive status bit. The receive status byte also reflects parity errors in the device.

5.5.2 Cyclic and Change-of-State I/O

The Cyclic connection initiates a transmission every time the connection timer expires. This is explained below in the Section 5.5.4. The cyclic connection can only send data from the W5-JDC4. If you need to transmit on the I/O link, you will need to use the polled connection to do so. The polled and cyclic connections are not exclusive, so both can exist at the same time. The manner in which cyclic connection reports its data is the same as the polled connection. The cyclic connections transmit buffer is the same as the polled connections transmit buffer, so overflows and received delimiters act the same over any connection.

The Change of State (COS) connection is the same as the cyclic connection except that as well as triggering communications on the expiration of the timer, the COS connection also initiates a transfer on a receive of the delimiter or an overflow. The COS connection is mutually exclusive with the cyclic connection, but can coexist with the polled connection. The COS connection operation is very useful in conserving bandwidth, and provides the Master with the most current data as fast or faster than a poll connection. The COS connection automatically turns on the COS mechanism when the connection is created.

5.5.3 Setting up the DeviceNet I/O Connections

It is useful to first set up your serial lnk before setting up your connection. To set up the communications with your network configuration tool, it is often necessary to know the connection input and output sizes. Instructions for setting up your serial connection are provided above. See the sections on receive and transmit sizes.

If you are using a network configuration tool with some type of scanner or scanning software, you must direct your scanner to set up the connections for you. This often requires some information about the device, such as input and output sizes. The input and output sizes are computed from the transmit size and the receive sizes. These sizes are defined in the transmit and receive objects of your device. The transmit size of the poll connection is computed by adding 4 to the Transmit Buffer Size on the W5-JDC4. The Transmit size for the change of state and cyclic connections are set to 0, because these connections do not initiate a transmission on the serial link. The receive size of all three connections is computed by adding the 4 bytes of the header to the receive buffer size.

Important: Remember to re-map the data (if necessary) after you set the sizes, because many configuration tools will automatically un-map your data when you change the connection sizes. If you are not using such a software package, it is probably not necessary to set up the transmit and receive sizes.

5.5.4 Setting up the Connection Timer (EPR)

EPR stands for Expected Packet Rate. This is the value that the W5-JDC4 sets the connection timer to for the cyclic and polled connection. This is also the value it uses in the connections to calculate the time the device should wait before signaling a timeout. If you have a scanner or scanning software, you must configure it with the EPR that you want the W5-JDC4 to be scanned with. The **scanner will then configure the EPR** in the W5-JDC4 at the beginning of communications. Consult your scanner's manuals on how to configure the EPR (the EPR is sometimes referred to as the "scan rate").

Note: If you need to set up the EPR, it can be done manually by performing a set (Service 10_{hex}) on the connection class (Class 5) attribute 9. The polled connection uses instance 2, where as the COS and cyclic connections use instance 4. This must be done after allocating the connection.

5.5.5 Setting up the DeviceNet Baudrate

Autobaud is the mechanism that allows the DeviceNet device, in this case the W5-JDC4, to automatically determine the baudrate that is operational on the network to which the W5-JDC4 is connected and to adjust its DeviceNet speed to match. The W5-JDC4 is shipped with autobaud as its default baudrate.

If you wish to change the baudrate to a fixed speed, you can set it in two different places. 1) The first is in the DeviceNet object (Class 3). This is where most configuration tools will look to change the baudrate. 2) Because autobaud is not supported by the standard DeviceNet Object, the W5-JDC4 provides a parameter to allow the autobaud selection. These values can be set and retrieved by using the standard Set and Get services.

6 Assembly Object Formats

The **produce** (respond to master's request) and **consume** (receive a master's request) assembly object formats vary according to 2 specific parameter options: the Data Type and the Handshaking / Immediate Mode operation.

For this to work correctly in many PLC's with a DeviceNet scanner, the outgoing poll message must be constructed in a buffer, and when complete, should then be copied whole to the scanner's file. Making the header portion always 4 bytes facilitates this. This makes mapping the device into a scanners memory always on either 16 or 32 bit boundaries.

If using a more direct mode, it is advisable to construct the whole message, leaving the Transaction ID alone, then change the Record Number when ready to have it sent by the W5-JDC4.

For the tables below

- TID = Record Number, 0-255
- Status = definition per Table 5-1.

6.1 Short String Data Type Assemblies

6.1.1 Immediate Mode

Table 6-1 Immediate Mode Consume Short String

Reserved	TID	Reserved	Length	String byte #1	String byte #2	String byte #3	String byte N

Table 6-2 Immediate Mode Produce Short String

TID	Status	Reserved	Length	String byte #1	String byte #2	String byte #3	String byte N

6.1.2 Handshake Mode

Table 6-3 Handshake Mode Consume Short String

Received TID	TID	Reserved	Length	String byte #1	String byte #2	String byte #3	String byte N

Table 6-4 Handshake Mode Produce Short String

TID	Status	Reserved	Length	String byte #1	String byte #2	String byte #3	String byte N

6.2 Array (Data Block) Data Type Assemblies

6.2.1 Immediate Mode

Table 6-5 Immediate Mode Consume Data block

Reserved	TID	Reserved	Reserved	String byte #1	String byte #2	String byte #3	String byte N

Table 6-6 Immediate Mode Produce Data block

TID	Status	Reserve d	Reserved	String byte #1	String byte #2	String byte #3	String byte N

6.2.2 Handshake Mode

Table 6-7 Handshake Mode Consume Data block

Receive TID	TID	Reserved	Reserved	String byte #1	String byte #2	String byte #3	String byte N

Table 6-8 Handshake Mode Produce Data block

TID	Status	Reserved	Reserved	String byte #1	String byte #2	String byte #3	String byte N

6.3 String Data Type Assemblies

6.3.1 Immediate Mode

Table 6-9 Immediate Mode Consume String

Reserved	TID	LSB of Length	MSB of Length	String byte #1	String byte #2	String byte #3	String byte N

Table 6-10 Immediate Mode Produce String

TID	Status	LSB of Length	MSB of Length	String byte #1	String byte #2	String byte #3	String byte N

6.3.2 Handshake Mode

Table 6-11 Handshake Mode Consume String

Received TID	TID	LSB of Length	MSB of Length	String byte #1	String byte #2	String byte #3	String byte N

Table 6-12 Handshake Mode Produce String

TID	Status	LSB of Length	MSB of Length	String byte #1	String byte #2	String byte #3	String byte N

7 Theory of Operation

7.1 The Transmit Record Algorithm

7.1.1 Basic Theory of operation

The Transmit record object addresses several issues of utilizing a DeviceNet network device in the process of communicating via a serial data stream. The device takes a block of data passed by the I/O and transmits this data out over the DeviceNet link. This data can be protected via a record counter for allowing a DeviceNet I/O connection to allow a device to control the transmission of this data from the W5-JDC4 by changing it.

The data flow and algorithm are shown below for reference.

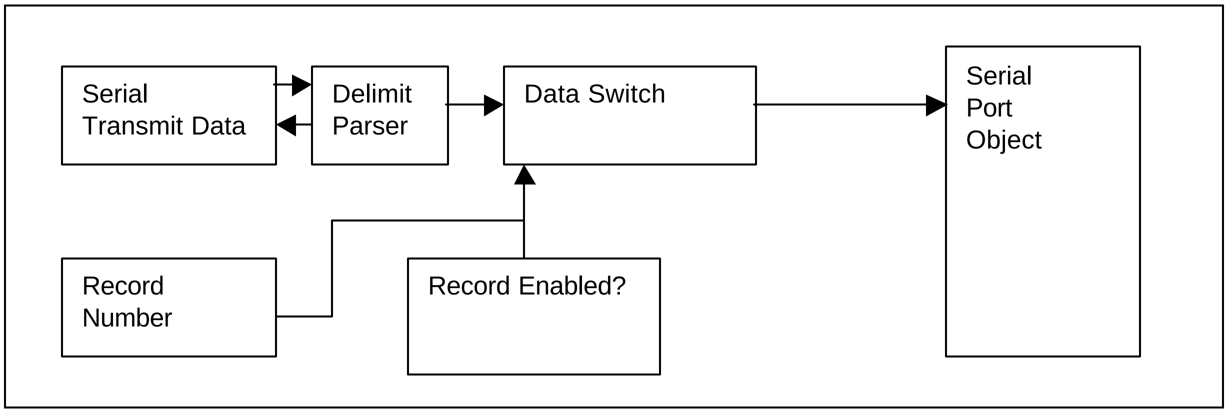


Figure 7-1 Transmit Record Algorithm Functional Flowchart

As diagramed above, the serial data is set and then parsed if the serial data length is set to zero. The serial data parse then sets the length on the serial data. If Record is disabled, the data is immediately added to the serial port's transmit queue. Also, whenever the record number is changed, the serial transmit data is sent to the serial port object.

The delimit parse determines the length of the string by searching the string for a delimiter character. The algorithm can be configured to ignore, include or exclude the character.

7.2 The Receive Record Algorithm

The receive record algorithm was developed in order to enable the receipt of data from a remote serial device and the subsequent transmission of data over a DeviceNet I/O connection. The Receive Record Algorithm controls when data is acquired and when that acquired data is presented to the connection. This Section will help you understand the algorithm in order to efficiently utilize all of the capabilities of the W5-JDC4 to make your job easier.

7.2.1 Basic Theory of operation

The Basic function of the W5-JDC4 is to address the problem of receiving and transmitting serial data over DeviceNet. Serial data is a data stream, in which data is presented to a device one byte at a time and does not have a beginning or an end. DeviceNet however, is not stream oriented; data is presented in chunks of defined sizes and bounds. The Receive Record algorithm formats the streaming data into a data block. It uses user-defined events on the receive serial link to determine where this data block starts and ends. It then uses a status byte to notify the controlling W5-JDC4 when new data is available. The W5-JDC4 then increments the record number and the new data are presented.

The Receive Record Algorithm creates a DeviceNet object and presents the data in an ordered fashion to the DeviceNet I/O connection. The Receive Record object may be reached over DeviceNet in Class 114. If you are going to be using explicit messaging to communicate with the device, we recommend that you access the data from this point.

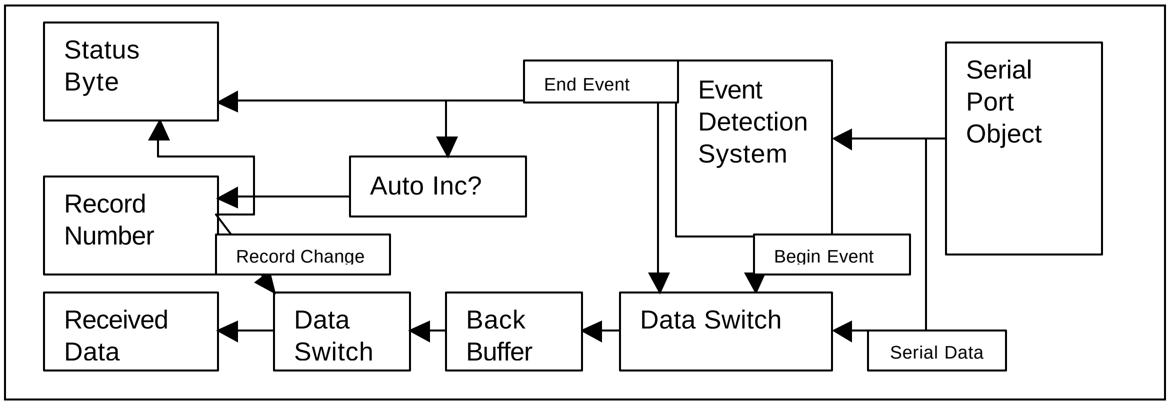


Figure 7-2 Receive Record Algorithm Functional Flowchart

The Receive Record Algorithm is diagramed above. The serial stream originates from the serial port object (Class 112). It is looked at a byte at a time by the event detection system. If the event detection system detects a beginning event (the beginning delimiter is received or disabled), it opens the first data switch, allowing the serial data to accumulate in the back buffer. Once an end event is received (the End delimiter is received, or the back buffer is full) the event detection system will close the data switch and set a new data status. When the new data status is set, if the device is set up in auto increment mode, the object will automatically increment the record number and clear the new data bit. In this situation, the user will not be allowed to set the Receive Record Number, and it will not be included in the poll request assembly.

The W5-JDC4 now includes functionality to allow you to specify the data type of the string. The string may be set to the DeviceNet data types of STRING, SHORT_STRING or ARRAY. The difference between the data types is how the length of the string is reported. The STRING data type has a two-byte indicator for length. This data type may be used to directly map your data into an A-B controller that supports the string data file.

The SHORT_STRING data type is the classic data type from the W5-JDC4 revisions 5 and under. This data type has a 1-byte length and data space savings.

The ARRAY data type does not have a length field and may be used if you do not need to know the length of your received data.

The W5-JDC4 also supports scanners that are not fully DeviceNet compliant and do not support short poll responses. These short poll responses save bandwidth on the DeviceNet network, but many scanners do not support this functionality and will not allow communication with a device that responds in this manner. The W5-JDC4 supports this non-compliant behavior through pad mode. This is a function where characters are appended to the end of the serial data in order to fill out the poll response. The default for pad mode is ON, and the default for the pad character is 0 (NULL).

Note: Turn pad mode ON if you receive errors indicating that the I/O data response is too short.

The W5-JDC4 also supports byte-reordering to support non-string data type I/O's. The DeviceNet network data-ordering scheme is little endian, meaning that the low byte of a multi-byte messages transmitted first. This means that if you are using an I/O that is using a 16 bit or larger word size and you map the I/O's data directly into this space without string support, your data bytes will show up swapped. The W5-JDC4 implements byte reordering (swapping) in order to ease use on these PLC's. The swapping mechanism re-orders the I/O data so that is appears correctly on I/O's that use Little endian byte order and greater than 8 bit word sizes. To Implement byte swapping, determine the number of bytes in your data size. Subtract 1 from this number, and set the Receive Record object's byte swapping attribute to this value. Now, you must set the length of your message size to a multiple of 1+ this value, or the W5-JDC4 will not be able to swap the bytes correctly. The data will now be ordered properly in your W5-JDC4.

8 DeviceNet Profile, Objects and Services

This section, along with the DeviceNet specification Volumes 1 and 2 Errata 5 completely describes the public operation of the device. Factory self-test and factory configuration options are not described in this manual.

8.1 W5-JDC4 DeviceNet Profile

This section describes the DeviceNet Objects present in the I/O. The I/O conforms to a Type 12, Communications Adapter Device.

Table 8-1 DeviceNet Objects

Object	DeviceNet Object Class	# of Instances
Identity	1	1
Message Router	2	1
DeviceNet	3	1
Assembly	4	2 (Consume Assembly) (Produce Assembly)
Connection	5	3 (Explicit Msg) (Polled I/O) (COS/Cyclic)
Acknowledge	43 ($2B_{hex}$)	1
Serial Port	112 (70_{hex})	4
Serial Transmit	113 (71_{hex})	4
Serial Receive	114 (72_{hex})	4

8.2 Identity Object (Class 1)

Instances 0 and 1 exist in the W5-JDC4.

Table 8-2 Identity Object Class Attributes (Instance 0)

Attribute ID	Access Rule	Name	DeviceNet Data Type	Description of Attribute	Value
1	Get	Revision	UINT	Revision of this object	1
2	Get	Max. Object Instance	UINT	Maximum instance number of an object currently	1
6	Get	Max. Class Attribute ID	UINT	Attribute ID number of the last class attribute of the class definition implemented in the device	7
7	Get	Max. Instance Attributes ID	UINT	Attribute ID number of the last instance attribute of the class definition implemented in the device	101

Table 8-3 Identity Object Instance Attributes (Instance 1)

Attribute ID	Access Rule	Name	DeviceNet Data Type	Description of Attribute	Value
1	Get	Vendor	UINT	ODVA Vendor Number for this product	9 = WRC
2	Get	Device Type	UINT	ODVA Communications Device Type	12 = Comm. Adapter
3	Get	Product Code	UINT	Unique Product Code Number	730 ($2DA_{hex}$)
4	Get	Revision	STRUCT of:	Revision of this device	
		Major Revision	USINT		1
		Minor Revision	USINT		>=6
5	Get	Status	WORD	Summary status of device	See DeviceNet Spec
6	Get	Serial Number	UDINT	Unique Device Serial Number	Varies per Device
7	Get	Product Name	SHORT STRING	ASCII Name of product	W5-JDC4
10	Get/Set	Heartbeat Interval	USINT	The interval in second that the device generates a heartbeat message. A value of 0 disables heartbeat generation.	0
101	Get	Build Number	UDINT	The Build Number is a WRC Specific Metric that helps maintain firmware revision control. It is unique to the specific build of firmware.	TBD

Table 8-4 Identity Object Common Services

Service Code	Class	Instance	Service Name	Description of Service
O5 $_{hex}$	Yes	Yes	Reset	Invokes the Reset Service for the device.
OE $_{hex}$	Yes	Yes	Get_Attribute_Single	Returns the contents of the specified attribute.
10 $_{hex}$	No	Yes	Set_Attribute_Single	Modifies an attribute value.

8.3 DeviceNet Object (Class 3)

The behavior of the DeviceNet object does not deviate and will have no extensions to the basic DeviceNet specification.

Table 8-5 Class 3 Class Attributes

Attribute ID	Access Rule	Name	DeviceNet Data Type	Description of Attribute	Semantics of Value
1	Get	Revision	UINT	Revision of this object	The current value assigned to this is two (2).

The following instance attributes are implemented in the DeviceNet object on the W5-JDC4

Table 8-6 Class 3 Instance Attributes

Attribute ID	Access Rule	Name	DeviceNet Data Type	Description of Attribute	Semantics of Value
1	Get/(*Set) (NV)	MacId	USINT	Mac Id of the device	0-63
2	Get/(*Set) (NV)	Baud rate	USINT	Baud rate that the device participates in bus activities with,	0-2.
3	Get/Set	Bus Off Interrupt	BOOL	Defines how a device processes a bus off interrupt	0= Hold in bus off condition 1 = reset the bus off condition and continue operating on the network if possible
4	Get/Set	Bus Off Counter	USINT	Number of times can went to the bus off state	0-255
5	Get	Allocation Choice Byte	USINT	This is a bit field of the currently allocated Group 2 Connection Set.	
6	Get	MACID Switch Changed	BOOL	This indicates if the switch has been changed science the last power cycle	0=Switch has not changed 1=Switch has changed science the last power cycle
7	Get	Baudrate Switch Changed	BOOL	This indicates if the switch has been changed science the last power cycle	0=Switch has not changed 1=Switch has changed science the last power cycle
8	Get	MacId Switch value	USINT	The actual value of the MacID switch	0-63 – MacID hardware settable 64-99 – MacID software settable
9	Get	Baudrate Switch Value	USINT	The actual value of the Baud Rate BCD switch.	0: hardware (125K) 1: hardware (250K) 2: hardware (500K) 3-9: software selectable.
100	Get/(*Set)(NV)	Autobaud	BOOL	Use the autobaud algorithm to determine the network Baudrate	0=Do not AutoBaud 1=Automatically Detect the baudrate
101	Get	Consume Assembly Size	UINT	Size of the Consume Assembly associated with the poll and COS connections	Size in bytes
102	Get	Produce Assembly Size	UINT	Size of the produce assembly associated with the poll and COS connections	Size in Bytes

* This is only settable if the MacID / Baudrate switch is in the software settable range, else this will represent the hardware set options. The hardware set options will be stored in non-volatile storage. The baudrate will represent the actual baudrate of the network in this case.

The autobaud algorithm will operate only if it's attribute value is set to 1. If the baudrate attribute (attribute 2) is set, autobaud (attribute 100) will be disabled. If the autobaud attribute is set to zero, on the next power cycle, the unit will use the baudrate specified by the baudrate attribute (attribute 2). If the switches are set to values 0-2 on power-up, the baudrate attribute will be set to the selected baudrate, and the autobaud attribute will be set to disabled. If the switches are set to 3-9, the values set in non-volatile memory will be used.

8.4 Assembly Object (Class 4)

The assembly objects are static; meaning only attribute 3 of the instances is Get/Set. In the case of the produce assembly, the access rule of attribute 3 is Get Only.

8.4.1 Instance 101 (Consume) Assembly Object Description

Table 8-7 Instance 101 Consume Data (ASCII Transmit String)

Byte	Character	Description	When Used
0	New Data Record Number	Integer value changed to indicate OK to send new data	Master-Slave Handshake Mode
1	T/ID #	Record Number, Integer value 0 – 255 (0 = Initialized State)	Normal Header Mode
2	Length	Number of bytes to transmit. 0 indicates transmit delimited mode, in which the device transmits up to and including the transmit delimiter character.	String - LSB
3	Length	Number of bytes to transmit. 0 indicates transmit delimited mode, in which the device transmits up to and including the transmit delimiter character.	String - MSB Short_String
4	1st Char	ASCII Character	Always
5	2nd Char	ASCII Character	Always
6	3rd Char	ASCII Character	Always
•			Always
•			Always
•			Always
•	Last Char	ASCII Character	Always
Length + 4	Terminator	End-of-Text Character	Include End Delimiter

8.4.2 Instance 102 (Produce) Assembly Object Description

Table 8-8 Instance 102 Produce Data (ASCII Receive String)

Byte	Character	Description	When Used
0	T/Id #	Record Number, Integer value 0 – 255 (0 = Initialized State)	Always
1	Status	Status / Error Value	Always
2	Length	Length of valid data in bytes	String - LSB
3	Length	Length of valid data in bytes	String - MSB Short String
4	1st Char	ASCII Character	Always
5	2nd Char	ASCII Character	Always
•			Always
•			Always
•			Always
•	Last Char	ASCII Character	Always
•	Terminator	End-of-Text Character	Include End Delimiter
•	Any	Pad character (present if characters received is less than Max Receive Chars value)	Pad Mode Enabled
•	Any	Pad character (present if characters received is less than Max Receive Chars value)	Pad Mode Enabled
Max Rx Char + 4	Any	Pad character (present if characters received is less than Max Receive Chars value)	Pad Mode Enabled

8.4.3 Disabling channels

Channels are disabled by setting the channel's Transmit Max Characters (Class **113**, Instance **ch**, Attribute **5**) and Receive Max Characters (Class **114**, Instance **ch**, Attribute **7**) parameters to zero. This will eliminate the channel's header and data from the produce assembly, and must be eliminated from the scanner's message to the JDC4.

8.4.3.1 Example of Channel Disabling

If: 113,2,5 = 0 and 114,2,7 = 0 **then** the order of both consume and produce is:

Table 8-9 Channel 2 disabled

Channel 1 Header & Data	Channel 3 Header & Data	Channel 4 Header & Data

8.5 Connection (Class 5) Object

This object is implemented as per ODVA DeviceNet Specification Volume1: Revision 2 :Errata 5.

8.6 Acknowledge (Class 43) Object

This object is implemented as per ODVA DeviceNet Specification Volume1: Revision 2 :Errata 5.

8.7 Serial Port Object Class 112 (70 $_{hex}$)

There are 4 instances of the Serial Port Object on the W5-JDC4. Each Instance of the Serial Port Object corresponds to a physical port on the W5-JDC4 board.

8.7.1 Serial Port Class Attributes

There are no Class Attributes associated with the serial port object.

8.7.2 Serial Port Class Services

There are no serial port services associated with this device.

8.7.3 Serial Port Instance Attributes

Table 8-10 Class 112 (70$_{hex}$) Serial Port Object Instance Attributes

Parameter	**Attribute**	**Access**	**Description**	**Parameter Choices**		**Default Setting**	**Default Value**	**Data Type**
Status	1	Get	Serial port status	**Bit 0**-Tx FIFO overflow **Bit 1**-Rx FIFO overflow **Bit 2**-Rx parity error **Bit 3**-Tx FIFO has data **Bit 4**-Rx FIFO has data		OK	0	BYTE
Reserved	2	Get/Set	Flow Control Status					
Reserved	3	Get/Set	Flow Control Type					
Serial Character Framing Format	4	Get/Set/ NV	Character framing	0 = 7N2 1 = 7E1 2 = 7O1 3 = 8N1 4 = 8N2	5 = 8E1 6 = 8O1 7 = 7E2 8 = 7O2	7N2	0	USINT
Serial Baud Rate	5	Get/Set/ NV	I/O communications speed	0 =1200 1 = 2400 2 = 4800 3 = 9600	4 = 19.2k 5 = 38.4k 6 = 57.6k 7 = 115.2k	9600 baud	3	USINT
Notify Rx Path	6	Get	Object to notify when receive events happen	EPATH		NULL		EPATH
Notify TX	7	Get	Object to	EPATH		NULL		EPATH

Path			notify when transmit events happen				

8.7.3.1 Status

The Status of the Serial Port is an Array of bits that correspond to the current status of the serial object FIFO and/or the next character in the serial object buffer. There are currently 5 defined bits:

1. TX FIFO overflow
   The Transmit FIFO has overflowed, causing the loss of Transmitted data. Send less data.
2. RX FIFO overflow
   The Receive FIFO has overflowed, causing the loss of received data.
3. RX Parity Error
   The Next Character in the Receive FIFO has a parity error associated with it.
4. TX FIFO has data
5. RX FIFO has data

8.7.3.2 Flow Control Status

This Parameter is reserved.

8.7.3.3 Flow Control Type.

This Parameter is reserved for future expansion

8.7.3.4 Serial Character Framing Format.

This attribute defines the framing format of the character. It includes Data Bits, Parity Selection And Stop Bits.

8.7.3.5 Serial Baud Rate

This Attribute defines the baudrate of the associated serial port.

8.7.3.6 Notify TX path

This attribute defines the endpoint for the transmit FIFO service. The transmit FIFO service expects the path defined to accept 3 bytes of data. The first 2 bytes represent the instance # of the reporting object. The third byte represents the state of the object. These states include:

0. FIFO is Empty
1. FIFO has data.
2. FIFO has overflowed

These status messages will only be relayed when the status of the object instance changes. If the path is null, no object will be notified. On the W5-JDC4, This value is hard coded to NULL and is get only.

8.7.3.7 Notify RX path

This attribute defines the endpoint for the transmit FIFO service. The transmit FIFO service expects the path defined to accept 3 bytes of data. The first 2 bytes represent the instance # of the reporting object. The third byte represents the state of the object. These states include:

0. FIFO is empty
1. FIFO has data without error.
2. FIFO has overflowed
3. FIFO has data with error.

These status messages will only be relayed when the status of the object instance changes. If the path is null, no object will be notified. On the W5-JDC4, This value is hard coded to NULL and is get only.

8.7.4 Instance Services

The instance supports the standard Get, Set and Reset services, along with 2 service specific services, Getc (4b $_{hex}$) and Putc (4C $_{hex}$).

8.7.4.1 GetC (4B $_{hex}$)

GetC takes no arguments. It returns 1 character out of the Receive FIFO. If there are no characters in the receive FIFO, GetC will return the general error code "No Stored Attribute Data" (18 $_{hex}$). You should not call this function if another Application Object that sends serial Data is using the Serial Port. On the W5-JDC4, the service should be considered 'in use' if the Max Receive Serial Character Parameter on the Receive Formatting Object is set to a non-zero value.

8.7.4.2 PutC (4C $_{hex}$)

PutC takes a short string data type as an argument. It adds this string into the transmit FIFO of the port.

8.8 Transmit Record Object Class 113 (71 $_{hex}$)

8.8.1 Transmit Record Object Class Attributes

There are no class attributes for the Transmit Record Object

8.8.2 Transmit Record Object Class Services

There are no Class services for the Transmit Record Object. Any service directed at the transmit record object will return the DeviceNet error code SERVICE NOT SUPPORTED

8.8.3 Transmit Record Object Instance Attributes

Table 8-11 Class 113 (71 $_{hex}$) Transmit Record Object Instance Attributes

Parameter	Attribute	Access	Description	Parameter Choices	Default Setting	Default Value	Data Type
Hardware Interface Instance	1	Get	Serial port number	Serial ports 1, 2, 3, 4	Instance Number	Instance Number	USINT
TX Record Number	2	Get/Set	Record number assigned to the data string to be sent to the serial device	0-255	0	0	USINT
Tx Data	3	Get/Set	Last transmitted serial data	N/a	None	0	USINT
Data Type	4	Get/Set/ NV	Format of data	**0** = Array **1** = Short String **2** = String	Array	0	USINT
Max Number of Tx Chars	5	Get/Set/ NV	Maximum number of characters the 1782-I/O expects to receive into its I/O port from the serial device	0 – 128	20 chars	20	USINT
Transmit End Delimiter	6	Get/Set/ NV	Character which identifies the end of the data string from the I/O device when the length is specified as 0	Any valid standard I/O character (0 – 127, 0-255)	Carriage return	D$_{hex}$	USINT
Transmit Delimiter Mode	7	Get/Set/ NV	Selects if the Tx delimiter is used, included or excluded in the resultant data string	**0** =No Delimiter **1** = Exclude the delimiter **2** = Include The delimiter	Include	2	USINT
DeviceNet Handshake Mode	8	Get/Set	Defines if the received string is sent immediately (in poll mode) or after a change in the Tx Record Number	**0** = Disabled **1** = Enabled	No hand shake	0	USINT
Swap Mode	9	Get/Set/ NV	If enabled, the position of the bytes in the serial messages will be swapped every 2,3, or 4 bytes	**0** = Disabled **1** = 16-bit Swap Enabled **2** = 24-bit Swap Enabled **3** = 32-bit Swap Enabled	Disabled	0	USINT
EDS Editor Data	100	Get/Set	Data entry point for EDS editors to allow the separating of the string length for the user.	Data of the Transmit string	NULL	0	Short String

EDS Editor Size	101	Get/Set	Data entry point for EDS editors to allow the separating of the string length for the user.	Length of the transmit string	No Data	0	USINT

8.8.3.1 Hardware Interface Instance

This is the instance of the serial port object that this object gets it's data from. On the W5-JDC4, this is attribute is get only and defined as its own attribute number.

8.8.3.2 Transmit Record Number

This is the transmit record number. Changing this number causes a transmit of the data stored in attribute 3, according to the delimiter parsing rules.

8.8.3.3 Transmit Data

This is the data you wish to send to the remote serial device.

8.8.3.4 Data String Type

This is the data type.

8.8.3.5 Maximum Transmit String Length

This attribute is stored in non-volatile memory. This attribute has get and set access. This attribute defines the maximum length allowed for a transmit string. This attribute is used in computing the location and size of the port in assembly objects. If this attribute is set to 0, the object is considered disabled, and no processing associated with this object occurs. Also, if this attribute is set to zero, the port will not appear in the consume assembly, including the ports header!

8.8.3.6 Transmit End Delimiter

This attribute is get/Set and is stored in non-volatile memory. End delimiter used if the data is set to 0 and the transmit mode is not set to no delimiter. See the Transmit delimiter mode selection for details on the operation of this parameter.

8.8.3.7 Transmit Delimiter Mode

This attribute is Get/Set. This attribute is stored in non-volatile memory. The transmit delimiter mode determines the handling of the transmit delimiter in case the data length is set to zero. If the Delimiter mode is set to NO_DELIMITER (0) the delimiter is ignored. If the data is set to EXCLUDE_DELIMITER (2), then the serial port will transmit up until the first delimiter character is encountered. If this option is set to INCLUDE_DELIMITER (1) then the serial port will also transmit the delimiter with the data.

8.8.3.8 Handshaking Mode

This attribute is Get/Set. This attribute is NOT stored in non-volatile memory.

Handshaking mode is not designated non-volatile because this parameter does not make sense using an implicit I/O connection. The program must set this parameter through the explicit connection if this feature is wanted.

8.8.3.9 Swap Mode

This attribute is Get/Set. This attribute is stored in non-volatile memory. Swap mode arranges the data into little-endian form along 2, 3 or 4 byte selectable boundaries. The maximum transmit length should be a multiple of the swap mode size setting. Swap mode is used to put data into human readable form for ease of use.

8.8.3.10 Status

There is no status associated with this object. You may wish to use the TX FIFO status information from the serial port object. Class **114**, Instance **ch**, Attribute **14** (**ch** = channel 1,2,3, or 4)

8.8.4 Transmit Record Object Instance Services

Service Code	Class	Instance	Service Name	Description of Service
0E hex	Yes	Yes	Get_Attribute_Single	Returns the contents of the specified attribute.
10 hex	No	Yes	Set_Attribute_Single	Modifies an attribute value.

8.9 Receive Record Object Class 114 (72 hex)

8.9.1 Receive Record Object Class Attributes

There are no class attributes for the Transmit Record Object

8.9.2 Receive Record Object Class Services

There are no Class services for the Transmit Record Object. Any service directed at the transmit record object will return the DeviceNet error code SERVICE NOT SUPPORTED

8.9.3 Receive Record Object Instance Attributes

Table 8-12 Class 114 (72 hex) Receive Record Object Instance Attributes

Parameter	Attribute	Acces s	Description	Parameter Choices	Default Setting	Default Value	Data Type
Hardware Interface Instance	1	Get	Instance of the serial port object that we listen to	1-4	1-4	1-4	USINT
Rx Record Number	2	Get/Set	Record number assigned to the last received data string	0-255	0	0	USINT
Rx Data	3	Get	Last received serial data	N/a	None	0	USINT
Rx Data Format	4	Get/Set/ NV	Format of data	**0** = Array **1** = Short String **2** = String	Short String	1	USINT
Pad Mode	5	Get	Indicates whether to pad the invalid data region after the delimiter with the pad character, or to use variable length I/O responses	**0** = Disabled **1** = Enabled	Enabled	1	USINT

Pad Character	6	Get/Set/ NV	The value to use to pad the invalid data portion of the poll response	Any valid standard I/O character (0 – 127, 0-255)	NULL	0	USINT
Max Number of Rx Chars	7	Get/Set/ NV	Maximum number of characters the gateway expects to receive into its I/O port from the serial device	0 – 128	20 chars	20	USINT
Start Delimiter Char	8	Get/Set/ NV	Character which identifies the beginning of the data string from the I/O device when the length is specified as 0	Any valid standard I/O character (0 – 127, 0-255)	Carriage return	D_{hex}	USINT
Stop Delimiter Char	9	Get/Set/ NV	Character which identifies the end of the data string from the I/O device when the length is specified as 0	Any valid standard I/O character (0 – 127, 0-255)	Carriage return	D_{hex}	USINT
Start Delimiter Mode	10	Get/Set/ NV	Selects whether or not the beginning delimiter is included in the data string	**0** = No Delimiter **1** = Exclude Delimiter **2** = Include Delimiter	No Delimiter	0	USINT
Stop Delimiter Mode	11	Get/Set	Selects whether or not the end delimiter is included in the data string	**0** =No Delimiter **1** = Exclude Delimiter **2** = Include Delimiter	Include	2	USINT
Swap Mode	12	Get/Set	If enabled, the position of the bytes in the serial messages will be swapped every 2 or 4 bytes.	**0** =Disabled **1** =16-bit Swap Enabled **2** =24-bit Swap Enabled **3** =32-bit Swap Enabled	Disabled	0	USINT
Immediate / Handshake Mode	13	Get/Set/ NV	Defines if the received string is sent immediately to master (in poll mode) or after a I/O handshake ACK	**0** =Set New Data Status Bit and wait for I/O to increment record number **1** =Automatically increment the record number and show the new data when an end event is received	Auto Increment	1	USINT

Status	14	Get	Status of the Record object	**Bit 7**-New Data Available **Bit 6-**Record State Conflict **Bit 5**-Non-Delimited Record **Bit 4-**Data In RX FIFO **Bit 3-**Data In TX FIFO **Bit 2-**Parity Error in current record. **Bit 1-**Rx FIFO overflow **Bit 0-**Tx FIFO Overflow	No Status	0	BYTE
Timeout Delay	15	Get/Set/ NV	Timeout in millisecond from the last received character until the algorithm times out and set the new data bit. A value of zero disables this feature.	Delay in ms.	Disabled	0	UINT
EDS Editor Data	100	Get/Set	Data entry point for EDS editors to allow the separating of the string length for the user.		NULL	0	SHORT STRING
EDS Editor Size	101	Get/Set	Data entry point for EDS editors to allow the separating of the string length for the user.	Data of the transmit string	No Data	N/A	USINT
EDS Editor Receive Record Number	102	Get/Set	Will not complain on a set of the record number when not in handshake mode. Otherwise, the behavior is the same.	Length of the transmit string	No Data	0	USINT

8.9.3.1 Hardware Interface Instance

This is the instance of the serial port object that this object gets it's data from. On the W5-JDC4, this is attribute is get only and defined as its own attribute number.

8.9.3.2 Receive Record Number

This attribute is Get/Set. This attribute is not stored in non-volatile memory. The Record number, when changed will cause new data to be put in to the received data attribute. The record number will change automatically on the set of the new data bit if the Handshake mode is set to Auto increment (1). When the received record number is changed, either by the algorithm or by the PLC, the new data bit will be reset. You may not se the new data bit toggle if the Handshake mode is set to auto-increment.

8.9.3.3 Received Data

This is the data received from the remote device. This attribute has get only access.

8.9.3.4 Received Data Format

This attribute defines the format of the received data. Please see the description of the choices on the transmit record object for behavior and allowable values.

8.9.3.5 Pad Mode

Pad mode is factory defaulted to 1 to allow the assembly objects to properly line up data. In the W5-JDC4 this parameter is get/set. See Attribute 5 in Table 8-12.

8.9.3.6 Pad Character

This attribute is get/Set. This attribute is stored in non-volatile memory.

The pad character is the character used to pad the string the maximum received length when the received string is shorter than this length. Legal values are 0-255.

8.9.3.7 Maximum Number of received characters

This attribute is Get/Set. This attribute is stored in non-volatile memory.

This attribute defines the maximum number of received characters of the received data parameter. Once the n'th character is inserted into the received data string, the new data bit will be set. The non-delimited record status bit will be set on the change to this record.

8.9.3.8 Start Delimiter Character

This attribute is Get/Set. This attribute is stored in non-volatile memory.

This attribute defines the start delimiter used in the Receive Record Formatting algorithm.

8.9.3.9 Stop Delimiter Character

This attribute is Get/Set. This attribute is stored in non-volatile memory.

This attribute defines the stop delimiter used in the Receive Record Formatting algorithm.

8.9.3.10 Start Delimiter Mode

This attribute is Get/Set. This attribute is stored in non-volatile memory.

Controls the start of the string. Characters received before the start character are dropped. Characters received after are placed into the string until the new data flag is set. Valid selections are NO_DELIMITER (0) (always acquiring characters), INCLUDE_DELIMITER, (1) include the character in the string), and EXCLUDE_DELIMITER (2) do not include the received delimiter in the string).

8.9.3.11 Stop Delimiter Mode

This attribute is Get/Set. This attribute is stored in non-volatile memory.

Controls the end of the string. Once the stop character is received, the new data flag is set. . Valid selections are NO_DELIMITER (0) (do not use delimiters to set the string), INCLUDE_DELIMITER,

(1) include the character in the string), and EXCLUDE_DELIMITER (2) do not include the received delimiter in the string).

8.9.3.12 Swap Mode

This attribute is Get/Set. This attribute is stored in non-volatile memory.

Swap mode arranges the data into little-endian form along 2, 3 or 4 byte selectable boundaries. The maximum transmit length should be a multiple of the swap mode size setting. Swap mode is used to put data into human readable form for ease of use.

8.9.3.13 Status

May be read to determine status of transmit or receive records.

8.9.3.14 Timeout Delay

This attribute is Get/Set. This attribute is stored in non-volatile memory.

The Timeout Delay is a new addition to the WRC block-formatting algorithm. This attribute allows implementers to define a timeout delay. If the unit does not receive a character before this timer expires, the unit will then indicate that new data is available. The unit will restart this timer every time a character is received and placed into the buffer that does not trigger the new data bit. The new data bit may be acted on immediately, if the device's handshake mode is set to Immediate. A change in record that was caused by a timeout of a buffer overflow will set the non-delimited record bit.

8.9.4 Receive Record Object Instance Services

Service Code	Class	Instance	Service Name	Description of Service
0E $_{hex}$	Yes	Yes	Get_Attribute_Single	Returns the contents of the specified attribute.
10 $_{hex}$	No	Yes	Set_Attribute_Single	Modifies an attribute value.

8.10 Common DeviceNet Services

DeviceNet is divided into logical functional blocks called objects, which provide services that allow for control over the hardware and routines that those objects contain. To allow for multiple similar functions, the objects are built of multiple instances that the services of the objects act upon. A class service acts upon the entire object, allowing one service to be enacted on all of the instances. This saves time, effort and network bandwidth.

The common services are a common set of services that have been provided in most or all of the objects to allow for common functionality in creating, deleting, getting, setting and resetting the variables of the different classes and instances. We will describe two of the services here: get and set.

The get and set services have a common format for specifying what object, instance, attribute and service that the command is specifying. In order of first to last, DeviceNet specifies service, class, instance, attribute and data. The data is always little endian (low byte precedes high order byte), and the others are all one byte in length on the W5-JDC4. Note that the get service has no data.

The get service gets data from an attribute of a class or class instance. The service number of this request is 14 (H_{ex}). The class instance and attribute are all defined by which variable you want to get. The response from a get command takes on the form: service, value. The value will be little-endian and can be of variable length and bounds based on the definition of the attribute. The service will be reported as the get service with the highest bit set to indicate a response.

The set service sets data from an attribute of a class or class instance. The service number of this request is 16 (10_{hex}). The class instance and attribute are all defined by which variable you want to set. The value is little endian, and the size is defined by the attribute that you are setting. The response from a set command only echoes the service. The service will be reported as the set service with the highest bit set to indicate a response.

An error response will have the service set to 94_{hex}. This response will be followed by a two-byte error code, defining the type of fault. For a detailed list of error codes, connect to the ODVA web site at www.odva.org.