1782-JDC
DeviceNet Serial Gateway
User's Manual

™

Western Reserve Controls, Inc.

Document 25.0
Rev 5.13
September 2011

Western Reserve Controls, Inc.
1485 Exeter Road
Akron OH 44306
330-733-6662 (Phone)
330-733-6663 (FAX)
sales@wrcakron.com (Email)
http://www.wrcakron.com (Web)

TABLE OF CONTENTS

LIST OF TABLES

TABLE OF FIGURES

1. Overview

The 1782-JDC is a family of DeviceNet-to-serial link communications gateways that provide a flexible DeviceNet interface to a wide variety of ASCII devices. The JDC allows the user to easily and conveniently connect and integrate peripheral products with RS232, RS422 or RS485 serial ports into a DeviceNet system.

Using the JDC 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 polling or explicit messaging. Typically real-time data is read and written as I/O by the DeviceNet Master via Polled I/O and parameters are read and written with the Explicit Messaging technique.

The 1782-JDC is defined as a Communications Adapter device on the DeviceNet system. It has a 3-pin plug connector for connection to RS232 (JDC-1) or RS485 (JDC-2) or a 5-pin plug connector for connection to RS422/485 (JDC-4) interface port on your device and a 5-pin pluggable DeviceNet connector for connections to the DeviceNet network. The device does baud rate selection automatically when it is powered up on a network. The 1782-JDC has one assigned DeviceNet address, which is set by a 6-position DIP switch on the unit. Other JDC parameters are software-configurable and are changed from their default values by third-party DeviceNet configuration tools. Each 1782-JDC has 2 standard green/red DeviceNet LED's for module status and network status and two green LED's to indicate RS232/422/485 transmit and receive activity.

The RS232 version may be used for point-to-point connection to a single serial device. The RS422/485 version may be connected in a point-to-point fashion to a single device, or to multiple devices in the standard RS422/485 convention.

The JDC is a general-purpose gateway that is completely device-independent. The JDC 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 device for a wide variety of DeviceNet-serial interface applications.

This manual applies to 1782-JDC version 5.

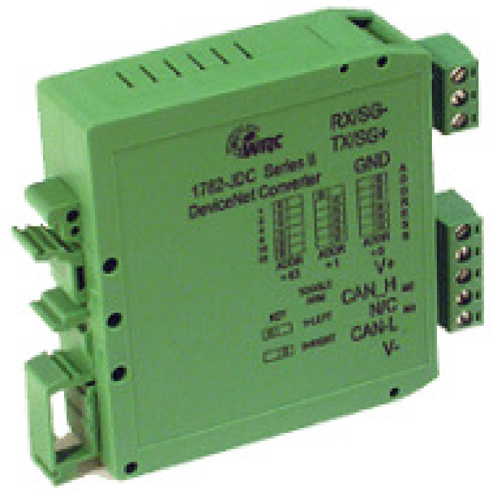

Figure 1-1 1782-JDC

1.1 Features

The 1782-JDC has the following features:

- Translates messages and data between DeviceNet and a serial peripheral device
- ODVA Group 2 Only Slave
- Defined as a DeviceNet Communications Device Profile 12 (C_{hex})
- Autobaud operation
- Polled I/O and Explicit Messaging
- Software Configurable Parameters for serial port operation
- Special mode performs byte swapping of serial message for AB PLC compatibility
- Address selection via DIP switches
- DIN rail mount
- Pluggable 5-pin DeviceNet connection
- Pluggable RS-485 2-pin connection / RS-232 3-pin connection / RS-422 5-pin connection
- 2 standard DeviceNet module and network status LED's
- 2 serial transmit and receive LED's
- Powered from DeviceNet 11-25 Vdc network power
- ASCII string length up to 124 bytes
- Serial port baud rate up to 38.4k baud
- Network electrical isolation

1.2 Product Family Selection

There are 3 members of the 1782-JDC family:

1782-JDC-1	Isolated RS-232
1782-JDC-2	Isolated RS-485
1782-JDC-4	Isolated RS-422/485

1.3 DeviceNet System Architecture

A DeviceNet network is a distributed I/O system that may contain many different products from several different vendors. Products may be configured uniformly, as clusters or as distributed clusters. Up to 64 devices, including the master may be attached to a single DeviceNet network. Any of these, except the master, may be a JDC. A typical system will include a master, such as a PLC or industrial PC, and multiple slave devices, including a 1782-JDC with connected peripheral devices.

1.4 Basic Operation

The JDC operates as the DeviceNet front-end to the serial device(s). Connecting the JDC to a single device, that device can then be assigned by the system implementer to one specific master. The DeviceNet Master can receive and send data to and from the 1782-JDC via the methods described in this section. It formats and sends the data to the device and likewise accepts responses from the device, which are reformatted and passed back to the DeviceNet system as required.

The JDC has one DeviceNet address. All DeviceNet messages to the JDC itself (to read / write its internal data) are sent to this address. All DeviceNet messages to and from the serial device are sent to the JDC DeviceNet assembly objects using poll commands.

The JDC Parameter Object allows you to define the specific operation of each JDC. These parameters include all the set-up required for the serial comm. link.

The following chart 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	√	√		√	
Status	√	√		√	
Parameters					√

1.4.1 Polled I/O

The DeviceNet Master uses the JDC's predefined polled IO connection to send input and output data to the JDC. The input data to the JDC has a one-byte record number, a one-byte length indicator and up to 124 additional bytes of transmit data. When a poll is received and the record has changed since the last poll was sent, the JDC sends the associated transmit data out the serial port to the remote ASCII device. When the JDC receives serial data from a device on the serial link, the poll response data to the Master contains a one byte record number, then a one byte status number that reflects errors or events on the bus, a one byte length indicator that is the length of the message in the response and up to 124 bytes of received data. The record number is incremented in the poll response when new data is received on the serial link. The status byte is zero if no errors occurred.

1.4.2 Explicit Messages

As mentioned, explicit messages are typically used to read and write configuration data. This data allows the JDC to change its internal operating parameters such as baudrate and parity. The JDC does not allow for direct communication to the serial ports using explicit messaging.

1.4.3 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 user-specified input data are transmitted to the master. This data is the same format as a poll response.

1.4.4 Change-of-State, or COS

COS 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 JDC, this occurs when the delimiter character is asynchronously received from the serial device. This data is the same format as a poll response.

1.5 Default Device Configuration

The 1782-JDC DeviceNet address is read from the switches and is set to 63 at the factory. All other parameters are software settable. The default settings for the 1782-JDC are provided in the discussion of the Parameter Object.

1.6 Product version and EDS

This manual applies to 1782-JDC-x version 5 and higher. An EDS (Electronic Data Sheet) for the 1782-JDC, is shipped with your device or is available on WRC's web site: http://www.wrcakron.com/.

2. Quick Start

To quickly install your 1782-JDC in your DeviceNet system, follow the instructions below. For more details, see Section 4.

2.1 How to Install and Establish DeviceNet Communications

1. Connect your DeviceNet network cable to a 5-pin female (Phoenix-type) plug according to DeviceNet cable wiring specifications
2. Make sure that the DeviceNet network is terminated properly.
3. The JDC 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 JDC address to one that is not currently used (see below).
4. The JDC baud rate is set to Autobaud operation at the factory. See below to change it to a fixed baud rate if desired.
5. Make sure that there is power on the DeviceNet network and plug the cable into the 1782-JDC.
6. The 1782-JDC 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 JDC is attempting to synchronize to the network baud rate.
7. The Module Status LED ("MS") will go on solid after the Device successfully determines the network baud rate. 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. If it remains off, make sure that there are other devices ***trying to communicate*** on the network.
8. 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.
9. The 1782-JDC is now operating on the network.

2.2 How to Change the Node Address

1. Set the 6-position DIP switch to the binary number representing the desired Node Address, 0-63. (Note: Address 0 is often reserved for a Master device.)
2. Power cycle the unit by unplugging and reconnecting the DeviceNet cable.

   NOTE: The new address will not become effective until the unit is power cycled or a Reset command is received from the Master.

2.3 How to Change the Baud Rate

The Baud Rate is set to autobaud at the factory. The baud rate can be changed through your configuration tool in its normal manner to any baud rate except autobaud. This is due to the fact that autobaud is not provided for in the DeviceNet specification. If you need to set the baud rate to autobaud, you must do it via the parameter object. The definition is included in the EDS file for easy configuration. Just use your configuration tool to access the device parameters. The baud rate is parameter 7. Select the proper baud rate and upload the parameter to the device. If your configuration tool does not support EDS parameter configuration, you will have to perform the operation manually. To do this, set the parameter Class 15 (F_{hex}), instance 7, attribute 1 to the value in Table 2-1.

NOTE: If you change the baud rate, the new baud rate will not become effective until the unit is power cycled or a reset command is received by the Identity Object (Class 1, Instance 0, reset).

Table 2-1 Baud Rate Selection

Baud Rate Value	Baud Rate
0	125k
1	250k
2	500k
3	Autobaud

2.4 How to Install a Serial Network

1. The communication between your serial device(s) and the 1782-JDC is a RS232 3-wire, RS485 2-wire or 5-wire RS422/485 differential network. Connect an appropriate cable to your device.
2. Connect the other end of the cable to the JDC using the 3-point or 5-point terminal plug provided. Note the terminal markings on the JDC case. See Figure 4-1.
3. Turn on power to the serial device and the JDC.
4. Set up the ASCII buffer sizes on the JDC. (The defaults are 20 and 20). If more than 20 bytes are required for the transmit or receive buffers, set parameters 5 and 6 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 / COS / cyclic transmit and receive data sizes if you modify the ASCII buffer size from the default value. In many configuration tools, this will unmap 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 software) These parameters can be reached in the Parameter Class 15 (F_{hex}), Instances 5 and 6, Attribute 1 if you need to configure these manually (See Table A-10).

2.5 How to Read Serial Device Input Data from the JDC

1. Set up the receive size of your connection to equal the (**Max Number of Receive Chars + 3**) in your Master's scan table. The default value should be 23 (20+3).
2. Map the data from your IO response to your scanners memory map. The device's response is byte aligned. The first byte indicates the receive record of the data. The second byte contains the JDC's serial status indicator (00 is normal). The third byte contains the length of the message being transmitted by the JDC. The subsequent bytes contain the serial buffer data from the serial input until and including the latest received

delimiter. The response is only updated when the delimiter is received or a buffer overflow occurs. Every time the buffer is updated (e.g. the delimiter received or an overflow condition) the record number will be incremented. The data is only valid up to the received delimiter in your memory map. Data after the received delimiter may contain invalid data.

3. Direct the master to begin polling the JDC. Once the first delimiter is received, the master's memory will reflect data received by the JDC. The data returned will have the following pattern:

Table 2-2 DeviceNet Consume Assembly / Serial Transmit Data String

BYTE	MEANING
Byte 1	Record Counter
Byte 2	Serial Port Status
Byte 3	String Length (number of data bytes)
Byte 4	ASCII Character 1
....	
Byte 23	ASCII Character 19
....	
Byte 127	ASCII Character 124

The status/error byte represents the status of the ASCII data record received from the ASCII device. This byte is defined as follows:

Table 2-3 Serial Receive Status/Error Byte

Byte Value	Meaning
0	No error
1	Buffer Overflow
2	Parity Error
3	Buffer Overflow and Parity Error

2.6 How to Write Serial Output Data to the JDC

1. Set up the transmit size of your connection to equal the buffer size of the ASCII transmit buffer plus 2. (Default is 22 bytes)
2. Map the IO transmit data of the JDC. The first byte of the JDC's IO transmit array is the record number. This value should be set to 0 at the beginning of communications. The second byte is a length indicator, followed by the number of data bytes. The ensuing bytes are the data that you wish to send to the JDC. If the length is set to 0 the JDC will send

all of the consecutive bytes up to and including the delimiter after the transmit record number has changed. If the length is non-zero, the JDC will send up to the number of bytes specified by the length ***without regard to the transmit delimiter and ignoring any "junk" characters after the specified amount of bytes.*** The format is:

Table 2-4 DeviceNet Consume Assembly / Transmit Data String

BYTE	MEANING
Byte 1	Record Counter
Byte 2	String Length (number of data bytes)
Byte 3	ASCII Character 1
Byte 4	ASCII Character 2
….	
Byte 21	ASCII Character 19
….	
Byte 126	ASCII Character 124

3. Begin scanning the JDC. Enter data that you want to send on the serial link. On receipt of a change to the record number, the JDC will transmit the data. If any data is in the receive buffer, the received data will be returned as a poll response. (If the sequence number is the same as the previous response, the JDC will not re-send the string.)

NOTE: If using the **Byte-Swap Mode** (Parameter 10 set to enabled) all strings being sent **must have the length set**. Using a zero length will prevent the byte swapping from occurring.

3. General Specifications

Product:	1782-JDC Device-Serial Gateway o 1782-JDC-1, Isolated RS-232 o 1782-JDC-2, Isolated RS-485 o 1782-JDC-4, Isolated RS-422/485
Description:	Communications gateway between a serial capable device over an RS232/422/485 interface and a DeviceNet network.
Device Type:	Communications Adapter, C $_{hex}$, (12)
Device Profile:	Identity Object Message Router Object DeviceNet Object Connection Object Parameter Object Serial I/O Object (vendor-specific)
Product Revision:	5
DeviceNet Communications:	Predefined Master/Slave Connection Set, Group 2 Only Server
DeviceNet:	**Baud rate selection:** Autobaud operation (default) Fixed baud (software selectable) – 125k, 250k and 500k baud
	Address selection: Address number 0 to 63, switch selectable (default = 63)
	Cable Connection: JDC: 5-pin pluggable header (male) Phoenix Contact MSTBA 2.5/5-G-5.08/AU or equivalent DeviceNet Cable: 5-contact plug (female contacts) Phoenix Contact MSTB 2.5/5-ST-5.08/AU or equivalent (included)
	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 baud (software selectable)
	Parity: Odd/even/none (software selectable)
	Data bits: 7 or 8 (software selectable)
	Cable Connection: JDC: 3-pin or 5-pin pluggable header (male) Phoenix Contact MSTBA 2.5/3-G-5.08/AU or equivalent

	Serial Cable: 3-contact or 5-contact plug (female contacts) Phoenix Contact MSTB 2.5/3-ST-5.08/AU or equivalent (included)
	Status Indicators: Transmit Active: green LED Receive Active: green LED
Network Isolation:	1000V (optional)
Max Power:	1.75 watts: 160 mA @ 11 Vdc – 70 mA @ 25 Vdc unregulated power supply
Mounting:	DIN rail mount, EN 50022 (WRC50022)
Size:	• Depth: 3.54" (90 mm) • Width: 0.98" (25 mm) • Height: 3.11" (79 mm)
Operating Temp:	0-70 ºC
Humidity:	0-95% RH, non-condensing

4. Hardware Installation and Set-Up

4.1 Overview

The JDC is mounted on an EN50022 DIN rail.

The JDC 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 two green LED's to indicate the presence of activity on the RS-232/422/485 transmit and receive lines.

All power for the JDC is derived from the DeviceNet power.

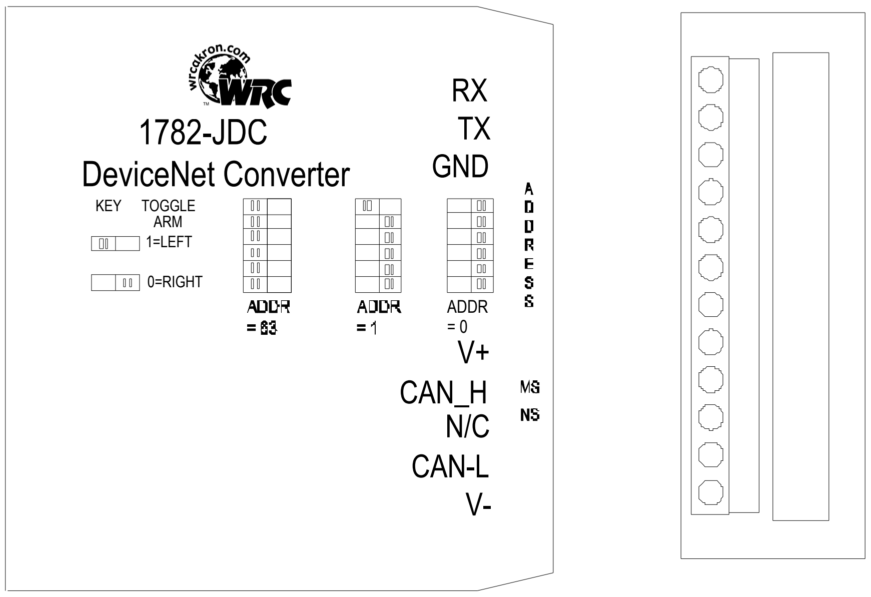

Figure 4-1 1782-JDC Outline Drawing

4.2 LED Operation

4.2.1 DeviceNet LEDs

The JDC 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	JDC is operating normally.
Flashing Green	Device in Standby	JDC needs commissioning (e.g. attempting autobaud).
Flashing Red	Minor Fault	Recoverable fault.
Red	Unrecoverable Fault	JDC may need replaced.
Flashing Red/Green	Device Self-Testing	JDC is in self-test mode.

Table 4-2 Network Status LED (labeled NS)

LED State	Network Status	Meaning
OFF	No Power / Not on-line	JDC has no power or has not completed the Dup_MAC_ID test.
Flashing Green	On-line, not connected	JDC is on-line but is not allocated to a Master.
Green	On-line	JDC is operating normally.
Flashing Red	Connection time-out	One or more I/O connections are timed out.
Red	Critical link failure	JDC 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 JDC also has two (2) RS-232/422/485 activity LEDs: one for transmit (TX) and one for receive (RX). Each of these will illuminate when there is data communications active on the respective data lines.

4.3 Serial Port Connector

The ASCII devices are connected to the JDC via a 3-wire or 5-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 JDC.

Table 4-3 RS-232/485 Connector Signals

Pin #	Designator (RS232/RS485)	RS232 Signal	RS485 Signal
3	RX/SG-	Receive	Signal -
2	TX/SG+	Transmit	Signal +
1	GND	Ground	Shield

Table 4-4 RS-422 Connector Signals

Pin #	Designator	RS422 Signal
5	TX-	Transmit -
4	TX+	Transmit +
3	GND	Ground
2	RX-	Receive -
1	RX+	Receive +

4.4 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 4-5

Table 4-5 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.

4.4.1 DeviceNet Network Termination

A DeviceNet system **must be terminated at each end of the trunk line**. The host controller and the **last** JDC 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

An appropriate terminating resistor, WRC part number RM121DN, may be purchased from WRC.

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

4.4.2 DeviceNet Connection Wiring

The JDC uses a 5-pin plug-style DeviceNet connector, which has male pins.

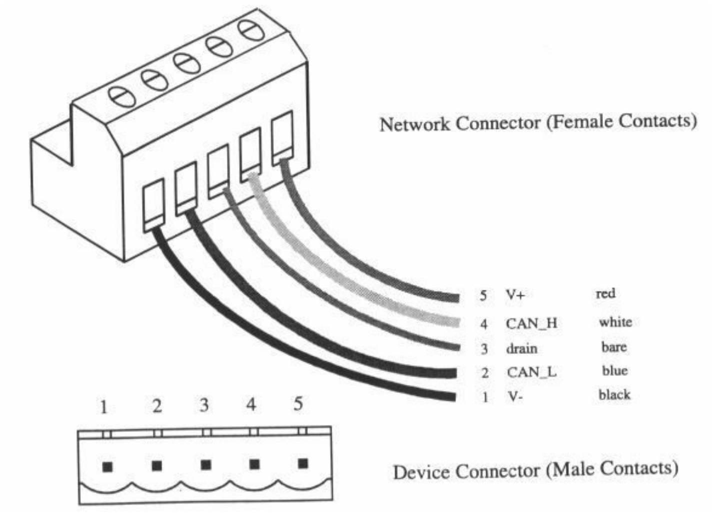
Network Connector (Female Contacts)
5 V+ red
4 CAN_H white
3 drain bare
2 CAN_L blue
1 V- black
1 2 3 4 5
Device Connector (Male Contacts)

Figure 4-2 DeviceNet cable connector

4.4.3 RS485 Network Termination (1782-JDC-2, 1782-JDC-4)

It is common practice to terminate an RS485 network system **at each end of the serial cable** to eliminate electrical reflections. An appropriate value for the terminating resistor typically is:

- 100-120 ohm, depending upon the characteristic impedance of the cable used
  - 121 ohms (same value as the DeviceNet terminator) is a standard resistor value that may be used in many cases
- 1% metal film
- 1/4 Watt

Correct cable termination is the responsibility of the user. The 1782-JDC-2 does not include an internal terminating resistor.

Note: 1782-JDC-2 products shipped with date code "0338" and earlier may have an internal 220-ohm terminating resistor installed. If you have any questions about this, please contact WRC by phone or at support@wrcakron.com.

5. SOFTWARE Configuration and Set-Up

The 1782-JDC-x 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, www.wrcakron.com. 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 1782-JDC and to help you set them up. If you have problems setting up a device, we are available to help you. We can be reached via phone at (330) 733-6662, or via the Internet at support@wrcakron.com.

The Parameters are defined in this section.

Table 5-1 Configuration Parameter List

Parameter	Param. Instance	Access	Description	Parameter Choices		Default Setting	Default Value	Data Type
Serial Character Format	1	Get/Set	Character framing	0 = 7N2 1 = 7E1 2 = 7O1	3 = 8N1 4 = 8N2 5 = 8E1 6 = 8O1	7N2	0	USINT
Serial Baud Rate	2	Get/Set	RS232/RS422/RS485 communications speed	0 =9600 1 = 1200 2 = 2400	3 = 4800 4 = 19.2k 5 = 38.4k	9600 baud	0	USINT
Receive Delimiter	3	Get/Set	Character which identifies the end of the data string from the ASCII device when the length is specified as 0	Any valid standard ASCII character (0 – 127, 0-255)		Carriage return	D_{hex}	USINT
Transmit Delimiter	4	Get/Set	Character which identifies the end of the transmit data string to the ASCII device when the length is specified as 0	Any valid standard ASCII character (0 – 127, 0-255)		Carriage return	D_{hex}	USINT
Max Number of Receive Chars	5	Get/Set	Maximum number of characters the JDC expects to receive into its ASCII port from the serial device	0 – 124		20 chars	14_{hex}	USINT
Max Number of Transmit Chars	6	Get/Set	Maximum number of characters the JDC expects to transmit out its serial port to the serial device	0 – 124		20 chars	14_{hex}	USINT
Device Net Baud Rate	7	Get/Set	The baud Rate of the device net device. The baud rate can be set here and will take effect once the JDC has been reset.	0 = 125 Kbaud 1 = 250 Kbaud 2 = 500 Kbaud 3 = Autobaud		Autobaud	3	USINT
Pad Mode	8	Get/Set	Indicates whether to pad the invalid data region after the delimiter with the pad character, or to use variable length I/O responses	0 = Pad Mode Disabled 1= Pad Mode Enabled		Enabled	1	USINT
Pad Character	9	Get/Set	The value to use to pad the invalid data portion of the poll response	Any valid standard ASCII character (0 – 127, 0-255)		NULL	0	USINT
Swap Bytes	10	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 = 32 bit Swap Enabled		Disabled	0	USINT

5.1 Setting up Serial Communications

5.1.1 Setting up the serial link baud rate

The standard baud rates are supported by the JDC, starting at 1200 baud and going to 38400 baud. The baud rate is configured at parameter 2 of the parameter object. The valid options are:

Table 5-2 Serial Baud rates

Value	Baud Rate
0	9600
1	1200
2	2400
3	4800
4	19200
5	38400

5.1.2 Setting up the receive delimiter

The receive delimiter is an **end-of-string** character. It is used by the JDC to determine that the last expected character from the ASCII device has been received. Upon receipt of this character, the JDC updates its internal data buffer on the serial receive side and transmits this new string to the DeviceNet system. It is also used by the DeviceNet sections of the device to determine when the device should generate a COS triggered message over the DeviceNet network, if COS has been selected. The receive delimiter can be set to any valid ASCII 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-$7F_{hex}$). If you do not have a valid receive delimiter, or the receive delimiter is never received, the device will only update the output buffer on detection of an overflow condition, that is, when it has received the number of characters defined in max_number_of_receive_chars, defined in Parameter 5.

These values can be set and retrieved by using the standard set and get services on class 15 (F_{hex}), instance 3, attribute 1 (See Table 5-1).

5.1.3 Setting up the transmit delimiter

The transmit delimiter is an **end-of-string** character which is used by the JDC to determine how many bytes to transmit over the serial link to your ASCII device. If the buffer length is not 0 the JDC will ignore the transmit delimiter. The JDC will transmit up to and including the delimiter when the above condition is met. The transmit delimiter can be set to any valid ASCII 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 15 (F_{hex}), instance 4, attribute 1 (See Table 5-1).

5.1.4 Setting up the Data Frame Format

The data frame format parameter defines the size of the data frame that is transmitted and received over the serial link. This parameter defines the number of data bits per character (7 or 8), the parity (odd, even or none) and the number of stop bits (1 or 2). The JDC provides several options that can be used to match the choice(s) available with your ASCII device, as the setting on both the JDC and your ASCII device must match exactly.

These values can be set and retrieved by using the standard set and get services on class 15 (F_{hex}), instance 1, attribute 1 (See Table 5-1).

5.1.5 Setting up the Receive Character Buffer Length

The receive character buffer length is the number of characters that the JDC can receive from your ASCII device into its buffer before generating an overflow and forcing the data into the JDC DeviceNet transmit buffer. This size value also determines the maximum size of the poll response to the Master from the JDC and is determined by:

produce_size = receive_character_length + 3

The valid settings for this value are 0-124. The default value is 20 (14_{hex}). This produce_size is used to set up the Poll, Cyclic and/or COS receive (response) sizes at your Master device.

These values can be set and retrieved by using the standard set and get services on class 15 (F_{hex}), instance 5, attribute 1 (See Table 5-1).

5.1.6 Setting up the Transmit Character Buffer Length

The Transmit character buffer length is the number of characters that the JDC can receive in its transmit buffer from the DeviceNet system. This size value also determines the maximum size of the poll command to the JDC from the Master (which is the JDC's Consume Size) and is determined by:

consume_size = transmit_character_length + 2

The valid settings for this value are 0-124. The default value is 20 (14_{hex}). This consume_size is used to set up the Poll transmit (command) size at your Master device.

These values can be set and retrieved by using the standard set and get services on class 15 (F_{hex}), instance 6, attribute 1 (See Table 5-1).

5.1.7 Setting up the Pad Mode

Pad Mode operation is the method by which the JDC adds extra characters to the end of its received data string (after the delimiter char) from the external ASCII 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 plus 3. The quantity of pad characters sent can vary from message to message, depending upon the size of the incoming string.

Pad mode is included with our device for compatibility with scanners that to not fully conform to the ODVA specification for **receiving** variable length I/O messages. (Notable examples include many Allen-Bradley's scanners at the time of printing). For scanners with this restriction, you must turn ON Pad mode (a value of 1). Turning Pad mode on will not harm ODVA compliant scanners; thus the default value for pad mode is OFF. If your scanner supports variable I/O messaging lengths, you may turn off pad mode off (a value of 0) to conserve some network bandwidth.

The selection of Pad Mode is valid only for the ASCII receive and DeviceNet I/O Response operation. It has no effect on DeviceNet I/O Command strings and ASCII transmit data to your ASCII device.

These values can be set and retrieved by using the standard set and get services on class 15 (F_{hex}), instance 8, attribute 1 (See Table 5-1).

5.1.8 Setting up the Pad Character

The JDC allows you to specify the character that pad mode uses to pad the received serial data with. This can be set to any valid ASCII value (0-127 in 7 bit modes, 0-255 in 8 bit modes).

These values can be set and retrieved by using the standard set and get services on class 15 (F_{hex}), instance 9, attribute 1 (See Table 5-1).

5.1.9 Using the Swap Bytes Mode

This mode is used if it is necessary to communicate through DeviceNet to an Allen Bradley PLC, such as the SLC500. When a serial message is to be transmitted or received, the bytes of the message will be reversed on the word boundary. Thus the message "ABCD" will be in memory like "BADC". This may cause problems in some cases. By setting Parameter 10 (class F_{hex}, instance 10, attribute 1) to 1, the bytes will be swapped by the JDC for both transmit and receive. *This means that all messages retain their left to right byte sequence in memory*.

For this to work, you must specify an even length for parameters 5 and 6. Messages may be any length (odd or even) that are equal to or less than parameters 5 or 6. When transmitting out from the PLC, the size of the string transmitted should be an even number. When transmitting an *odd size*, **round up the size to the next even number**.

By setting Parameter 10 to 2, the bytes will be swapped in 4 byte blocks (32-bits) for both transmit and receive, achieving the same left to right sequence as above in the 32 bit memory. This mode is intended for use with AB 32-bit PLC's.

For this to work, you must specify a length that is evenly divisible by 4 for parameters 5 and 6. Messages may be any length (odd or even) that are equal to or less than parameters 5 or 6. When transmitting out from the PLC, the size of the string transmitted should be an even number. When transmitting an *odd size*, **round up the size to the next divisible by 4 number**. Example: The length of the message is 16 bytes. Add 2 since we are using carriage return / linefeed terminators. Since 4 doesn't divide into 18 evenly, the length should be specified as 20.

Note: When mapping the JDC into a PLC and using byte swapping, be sure the message body starts on a 16 or 32 bit boundary or the results may be erroneous. It usually means mapping the record number, error (if receiving), and length bytes to the discrete input or output file or another suitable data file. This is also helpful, since the values of these numbers are available for use without a copy file command, such as needed when dealing with M0 and M1 files. The body of the ASCII string can be mapped to the M0 and M1 files.

Note: This feature is only **valid** for non-zero length strings (non-character-terminated data). It will not work with character-terminated strings when the length byte is left at 0.

Note: Termination characters are counted as part of the length of the non-zero length strings.

Note: The Pad mode should be used when operating a JDC with a PLC master.

5.1.10 Transmitting Serial Data

The use of a delimiter in transmitting data to a serial device from the JDC is determined by the Max_Number_of_Transmit_Chars parameter. The operation is defined below.

If Max_Number_of_Transmit_Chars = 0 (Param. 5),

The JDC receives data sent from the DeviceNet Master and **uses the delimiter** to determine how much data is sent to the serial device.

If a delimiter is contained within the string, then

- all characters up to and including the defined delimiter are sent to the ASCII device

If no delimiter is contained within the string, then

- the JDC will send data received from the DeviceNet I/O command message to the ASCII device

Note: If Max_Number_of_Transmit_Chars is zero, **Byte-Swapping will not work.**

If Max_Number_of_Transmit_Chars > 0 (Param. 5),

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

Note: Pad Mode is irrelevant to the ASCII transmit operation.

5.1.11 ASCII Receive Data and Pad Mode

The selection of Pad Mode is valid only for the ASCII receive and DeviceNet I/O Response operation. It has no effect on DeviceNet I/O Command strings and ASCII transmit data to your ASCII device.

The number of bytes returned to the DeviceNet Master in a Poll Response is as follows:

If Max_Number_of_Receive_Chars = 0 (Param. 5),

The expected operation is that the JDC receives characters (data bytes) **until the delimiter character is received**

If a delimiter is received, then

- all characters up to and including the defined delimiter are sent to the DeviceNet master

If no delimiter is received, then

- the JDC will receive data until the quantity of received bytes fills up its internal buffer (124 bytes), and then sends the entire string to DeviceNet with an overflow error

In this situation, the Pad Mode selection is irrelevant and no pad characters are added to the data returned.

Note: If Max_Number_of_Receive_Chars is zero, **Byte-Swapping will not work.**

If Max_Number_of_Receive_Chars > 0 (Param. 5),

The expected operation is that the JDC receives characters (data bytes) **until the delimiter character is received**

If a delimiter is received, then

- All characters up to and including the defined delimiter are sent to the DeviceNet master, and
    - If Pad Mode = 1, then the JDC will fill the Poll Response data with the Pad Char defined in Param. 9, up to the defined size
    - If Pad Mode = 0, then the JDC will send only the data up to and including the delimiter

If no delimiter is received, then

- The JDC will receive up to Max_Number_of_Receive_Chars, and then sends this string to DeviceNet with an overflow error
- If 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 ASCII device does not transmit the specified delimiter.

5.2 Setting up DeviceNet Communications

The 1782-JDC supports 3 modes of data transfer of the serial buffer. They are:

- Polled
- Change-of-state
- Cyclic

5.2.1 Polled I/O

The polled connection is the only manner in which you can send serial output data to the JDC and, therefore, to your ASCII device. The DeviceNet Master initiates the polled connection transfer. The Master sends the JDC its serial output buffer along with a record number and length byte. The JDC monitors for a change in the record number. If the record number changes, then the 1782-JDC 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 JDC 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 only if specified in the parameter object. When the JDC 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 JDC indicates so in its receive status bit. The receive status byte also reflects parity errors in the device. (See Table 2-3.)

5.2.2 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 5.2.4. The cyclic connection can only send data from the JDC. If you need to transmit on the ASCII 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.

5.2.3 Setting up the DeviceNet I/O Connections

It is useful to first set up your serial link 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 parameter object of your device. The transmit size of the poll connection is computed by adding 2 to the transmit buffer size on the 1782-JDC. 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 3 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 unmap 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.

5.2.4 Setting up the Connection Timer (EPR)

EPR stands for Expected Packet Rate. This is the value that the JDC 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 JDC to be scanned with. The **scanner will then configure the EPR** in the JDC 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_{hex}) 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.

5.2.5 Autobaud Operation

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

5.2.6 Setting up the DeviceNet Baud rate

If you wish to change the baud rate to a fixed speed, you can set it in two different places. The first is the standard location for the DeviceNet baud rate, which is in the DeviceNet object. This is where most configuration tools will look to change the baud rate. This is fine on normal DeviceNet systems, however the JDC provides some extended DeviceNet functionality above normal systems to allow the autobaud functionality. This extension is not defined by ODVA thus cannot be attached to any of the standard DeviceNet objects to ensure compatibility. You can still use the baud rate in the normal manner; it will just not allow the autobaud setting.

We have provided an extended baud rate parameter at attribute 7 of the parameter object to allow for the autobaud baud rate setting. (See Table 5-1).

These values can be set and retrieved by using the standard set and get services on class 15 (F_{hex}), instance 7, attribute 1. (See Table 5-1).

A. DeviceNet Profile, Objects and Services

A.1 1782-JDC DeviceNet Profile

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

Table A-1 DeviceNet Objects

Object	**DeviceNet Object Class**	**# of Instances**
Identity	1	1
Message Router	2	1
DeviceNet	3	1
Connection	5	3 (Explicit Msg., Polled I/O, COS/Cyclic)
Parameter	15 (F_{hex})	10

A.2 Serial I/O Polled Data Formats

Table A-2 Poll Produce Data (ASCII Receive String)

Byte	Character	Description
0	Record #	Record Counter, Integer value 1 – 255 0 = Initialized State (counter value rolls over from 255 to 1, skipping 0)
1	Status	Status / Error Value
2	Length	Length of valid data in bytes
3	1st Char	ASCII Character
4	2nd Char	ASCII Character
•		
•		
•		
length+1	Last Char	ASCII Character
length+2	terminator	End-of-Text Character
•		
•		
Max Rec. Char + 2	Any	Pad character (present if characters received is less than Max Receive Chars value)

Table A-3 Poll Consume Data (ASCII Transmit String)

Byte	Character	Description
0	Record #	Record Counter, Integer value 1 – 255 0 = Initialized State (counter value rolls over from 255 to 1, skipping 0)
1	Length	Number of bytes to transmit. 0 indicates transmit delimited mode, in which the device transmits up to and including the transmit delimiter character defined in the parameter object.
2	1st Char	ASCII Character
3	2nd Char	ASCII Character
4	3rd Char	ASCII Character
•		
•		
•		
length	Last Char	ASCII Character
length +1	Terminator	End-of-Text Character
•		
•		
Max TX Length – 1	Any	Undefined
Max TX Length	Any	Undefined (last char in this record transmitted to Master)

A.3 Identity Object, Class 1

Instances 0 and 1 exist in the 1782-JDC.

Table A-4 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	1

Table A-5 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	Set	Product Code	UINT	WRC Unique Product Code Number	701 = $2bd_{hex}$
4	Get	Revision	STRUCT of:	Revision of this device	
		Major Revision	USINT		>=5
		Minor Revision	USINT		>=1
5	Get	Status	WORD	Summary status of device	
6	Get	Serial Number	UDINT	WRC Unique Device Serial Number	
7	Get	Product Name	SHORT_STRING	ASCII Name of product	1782-JDC
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

Table A-6 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.

A.4 Parameter Object, Class F$_{hex}$ (15$_{dec}$)

There are many configurable data parameters associated with your JDC. The JDC uses a Parameter Object (a collection of these parameters) to assist you in reading and changing configurable data.

Following are the Class Attributes, Instance Attributes and Services that are supported by the JDC for the Parameter Object.

Table A-7 Parameter 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. Instance	UINT	Maximum instance number of the Parameter object	9
8	Get	Parameter class descriptor	WORD	Bits that describe parameters.	9 (supports parameter instances, params are stored in non-volatile memory)

Table A-8 Parameter Instance Attributes (Instances 1-7)

Attribute ID	Access Rule	Name	DeviceNet Data Type	Description of Attribute	Value
1	Set	Parameter Value	*data type* specified in Descriptor Data Type and Data Size.	Actual value of parameter. It can be read from or written to. This attribute is read-only if bit 4 of Attribute 4 is TRUE.	
2	Set	Link Path Size	USINT	Size of link path. If this is 0, then no link is specified.	Number of bytes
3	Set	Link Path	ARRAY of DeviceNet path:	DeviceNet path to the object from where this parameter's value is retrieved.	
4	Get	Descriptor	WORD	Description of parameter.	
5	Get	Data Type	USINT	Data type code.	
6	Get	Data Size	USINT	Number of bytes in Parameter Value	

Table A-9 Parameter Common Services

Service Code	Class	Instance	Service Name	Description of Service
O5 $_{hex}$	Yes	N/A	Reset	Resets all parameters to "out-of-the-box" values.
OE $_{hex}$	Yes	Yes	Get_Attribute_Single	Returns the contents of the specified attribute.
10 $_{hex}$	Yes	Yes	Set_Attribute_Single	Modifies an attribute value.

Table A-10 JDC Parameter Instances (Class F_{hex})

Parameter	Param. Instance	Description	Parameter Choices		Default Setting	Default Value
Serial Character Format	1	Character framing	0 = 7N2 1 = 7E1 2 = 7O1	3 = 8N1 4 = 8N2 5 = 8E1 6 = 8O1	7N2	0
Serial Baud Rate	2	RS232/ RS422/RS485 communications speed	0 =9600 1 = 1200 2 = 2400	3 = 4800 4 = 19.2k 5 = 38.4k	9600 baud	0
Receive Delimiter	3	Character which identifies the end of the data string from the ASCII device when the length is specified as 0	Any valid standard ASCII character (0 – 127, 0-255)		Carriage return	D_{hex}
Transmit Delimiter	4	Character which identifies the end of the transmit data string to the ASCII device when the length is specified as 0	Any valid standard ASCII character (0 – 127, 0-255)		Carriage return	D_{hex}
Max Number of Receive Chars	5	Maximum number of characters the JDC expects to receive into its ASCII port from the serial device	0 – 124		20 characters	14_{hex}
Max Number of Transmit Chars	6	Maximum number of characters the JDC expects to transmit out its serial port to the serial device	0 – 124		20 characters	14_{hex}
Device Net Baud Rate	7	The baud Rate of the device net device. The baud rate can be set here and will take effect once the JDC has been reset.	0 = 125 Kbaud 1 = 250 Kbaud 2 = 500 Kbaud 3 = Autobaud		Autobaud	3
Pad Mode	8	Indicates whether to pad the invalid data region after the delimiter with the pad character, or to use variable length I/O responses	0 = Pad Mode Off 1= Pad Mode Enabled		Enabled	1
Pad Character	9	The value to use to pad the invalid data portion of the poll response	Any valid standard ASCII character (0 – 127, 0-255)		NULL	0
Swap Bytes	10	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 = 32-bit Swap Enabled		Disabled	0

Parameter Instances for Table A-10

1. **Serial Character Format**: The character frame format.

2. **Baud Rate**: The baud rate of the ASCII channels. Valid baud rates are 1200, 2400, 4800, 9600, 19.2k, and 38.4k baud.

3. **End-of-Text Character**: The (printable or non-printable) byte character (value of 0-255) that signifies that the last character in a string of ASCII characters has been reached. The JDB will not return to the Master any additional characters after the EOT character, or any characters in the string after the maximum character count is reached.

4. **Transmit Delimiter**: The (printable or non-printable) byte character (value of 0-255) that signifies that the last character in a string of ASCII characters has been reached. The JDB will not transmit to the device any additional characters after the transmit delimiter character, or any characters in the string after the maximum character count is reached.

5. **Maximum Number of ASCII Input Characters**: The maximum number of ASCII characters expected to be input from the serial device connected to this port. The end-of-string, or end-of-text, character is included in this count. (A "0" implies the port does not receive data from the external device or that no device exists.)

6. **Maximum Number of ASCII Output Characters**: The maximum number of ASCII characters that will be sent from the host to the serial device connected to this port. The end-of-string, or end-of-text, character is included in this count. (A "0" implies the port does not output data to the external device or that no device exists.)

7. **DeviceNet Baud Rate**: Allows a selection of baud rates on the DeviceNet link.

8. **Pad Mode**: When enabled the pad mode will pad the serial receive data with the specified pad character. When disabled, the JDC will only return the receive data, saving some network bandwidth. The default is enabled. **Pad mode must be enabled for Allen-Bradley Scanners due to their inability to handle short poll responses**.

9. **Pad Character**: The character that the pad mode uses to lengthen the poll response with.

10. **Swap Bytes**: Exchange byte positions on word or long integer boundaries for both receive and transmitted messages. This is included for capability with Allen Bradley PLC's such as the SLC500 or Control Logic.

A.5 Common DeviceNet services

The common DeviceNet services are available through many of the common configuration tools. However, each configuration tool may implement these differently. It may be easier if you use this section in conjunction with any files or documentation that was included with your DeviceNet configuration tools while going over this section.

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 JDC. 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 (E_{hex}). 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.

B. Accessories and Other WRC Products

The following components can be used with a JDC for replacements or spare parts, or as complementary devices as a part of your DeviceNet or other CAN-Bus system.

Table B-1 WRC Replacements, Spare Parts and Other Products

Part	WRC Part Number
DIN rail	WRC 50022
Terminating resistor, axial lead	RM121DN
Discrete I/O block – 4 channels	1782-JDB4
Discrete I/O block – 8 channels	1782-JDB8
Analog Input block – 4 channels, 10-bit	1782-JDA4
Analog I/O block – 8 channels, 12-bit	1782-JDA8
DeviceNet to Serial I/O Gateway	1782-JDC
DeviceNet to Modbus Gateway	1782-JDM
DeviceNet to Optomux Gateway	1782-JDO
DeviceNet to Pamux Gateway	1782-JDP
DeviceNet to Serial Gateway, open frame	1799wr-DASCII
DeviceNet/CAN Software Utility	DNspector™
Discrete I/O block – 24 channels	W2-JDB24
Discrete I/O block – 48 channels	W2-JDB48
Discrete I/O, Analog Input block – 24 DIO, 32 AI	W2-JDA24
Discrete I/O, Analog Input block – 48 DIO, 32 AI	W2-JDA48
Analog I/O block - 32 channels	W2-JDAIO
Discrete and Analog I/O block – 24 DIO, 32 AIO	W2-JDAIO24
Discrete and Analog I/O block – 48IO, 32 AIO	W2-JDAIO48
Discrete I/O block – 8 DIs, 8 DOs, 4 AIs	W5-JDB16x
DeviceNet, CANopen Extender, DIN mount	WRC-CANX-DIN-DN
DeviceNet, CANopen Extender, DIN mount	WRC-CANX-DIN-C7
DeviceNet, CANopen Extender, NEMA box	WRC-CANX-NEM-AU
DeviceNet, CANopen Extender, NEMA box	WRC-CANX-NEM-DN
DeviceNet, CANopen Extender, Fiber Optic, NEMA box, multi-mode fiber	WRC-CANR-DF-DN
DeviceNet, CANopen Extender, Fiber Optic, NEMA box, single-mode fiber	WRC-CANR-DF-SM

C. Frequently Asked Questions

1. ***Where are the 1782-JDCx gateway User's Manuals located?***
   The WRC product User's Manuals are available from our web site Support -> Technical Data http://wrcakron.com/data.html

2. ***What are the 1782-JDCx gateways and why should I consider purchasing one for my DeviceNet network?***
   The 1782-JDCx gateways are communications adapter devices that provide you with a flexible DeviceNet interface to a wide variety of ASCII devices. The JDCx models allow you to easily and conveniently connect and integrate peripheral products with either RS-232, RS-485 or RS-422 serial ports into a DeviceNet system, enabling you to select just the right peripheral device for your application without limiting your selection to DeviceNet-cable devices only.

3. ***How can I change the DeviceNet MAC ID of my 1782-JDCx?***
   The address of the 1782-JDCx will come preset from the factory for address 63. You may change the address using the bank of 6 DIP switches on the unit: they can be set to any address from 0 to 63 in binary (111111=address 63). The MACID or address will not change if the switches are moved while the device is powered; power must be removed and reapplied when a new switch setting and MAC ID are desired or a device reset command must be issued.

4. ***What electrical connections are required for a 1782-JDCx gateway?***
   DeviceNet Cable Connection:
   1, 5-pin pluggable header (male, Phoenix Contact MSTBA 2.5/5-G-5.08/AU or equivalent)

   Serial Cable Connection: For RS-232 & RS-485
   1, 3-pin pluggable header (male, Phoenix Contact MSTBA 2.5/3-G-5.08/AU or equivalent)

   Serial Cable Connection: For RS-422
   1, 5-pin pluggable header (male, Phoenix Contact MSTBA 2.5/5-G-5.08/AU or equivalent)

5. ***How do I begin to use a 1782-JDCx gateway?***
   You can begin to use a 1782-JDCx gateway by following the User's Manual ***Quick Start*** instructions.

6. ***What are the LEDs on the 1782-JDCx indicating?***
   There are two sets of LEDs on the unit. The two green LEDs next to the serial connector are used as indicators for activity on the serial communication receive and transmit ports. The other two bi-color LEDs are used to indicate the DeviceNet module (MS) and network (NS) status. For troubleshooting or further information on status definitions corresponding to LED actions see ***LED Operation*** in the 1782-JDCx User's Manual.

7. ***On what DIN rail can I mount the 1782-JDCx?***
   The 1782-JDCx can be mounted on DIN rail mount EN50022 (WRC50022).

8. ***What are some common applications for the 1782-JDCx gateway?***
   A 1782-JDCx gateway provides DeviceNet communications for weigh scales, bar code readers and scanners, display panels, robots, drives, operator stations / HMI, magnetic code readers and other ASCII serial devices.

9. ***Are the 1782-JDCx gateways designed to "plug-n-play"?***
   Yes, the 1782-JDCx gateways are designed to "plug-n-play" using our factory default settings: serial framing of 7N2, serial baudrate of 9600, max number of receive and transmit

characters set to 20, pad mode off, etc. The full list of default settings can be found under ***Default Settings*** in the User's Manual. If you wish to change any parameters from their default settings you will need to use a DeviceNet software configuration tool. Many of these tools have EDS readers, which make the editing process easier.

10. ***Does a 1782-JDCx gateway provide electrical isolation?***
    Yes, a 1782-JDCx gateway provides 1000V network electrical isolation.

11. ***What are the currently available 1782-JDC-x gateway models?***
    1782-JDC-1: Isolated RS232 Interface
    1782-JDC-2: Isolated RS485 Interface
    1782-JDC-4: Isolated RS485/RS422 Interface
    1782-JDCE-1: Isolated RS232 Interface with Enhanced Parameters

12. ***On what DeviceNet baud rates does the 1782-JDCx communicate?***
    The unit works with 125k, 250k or 500k baud rates. It is default set to autobaud on DeviceNet. A fixed baud rate of 125k, 250k or 500k may also be assigned with a software configuration tool.

13. ***On what baud rates does the serial port communicate?***
    The serial port on the 1782-JDCx will communicate on baud rates of: 1200, 2400, 4800, 9600, 19.2k, or 38.4k baud.

14. ***How do I connect a 1782-JDCx gateway to a serial port device?***
    You connect the JDCx to a serial port device with the Phoenix connectors on the unit which are labeled for which signal goes in each port of the connector. For RS-232 or RS-485 communications the signal/pin assignments are as follows:

Pin #	Designator (RS232/RS485)	RS232 Signal	RS485 Signal
3	RX/SG-	Receive	Signal -
2	TX/SG+	Transmit	Signal +
1	GND	Ground	Shield

For RS-422 or RS-485 communications the signal/pin assignments are as follows:

Pin #	Designator (RS422/RS485)	RS422 Signal	RS485 Signal
5	TX- / GND	Transmit -	no connection
4	TX+ / GND	Transmit +	no connection
3	GND / GND	Ground	Shield
2	RX- / SG-	Receive -	Signal -
1	RX+ / SG+	Receive +	Signal +

15. ***How is the Master Handshake mode different from the Hardware handshake RTS/CTS?***
    Master Handshake mode is a data control mechanism between the DeviceNet master and the 1782-JDCx (after the JDCx receives data from serial device), communicated in IO message. The 1782-JDCx indicates new data arrival and waits for Master to issue a new record number, to be used with new data. The hardware handshaking via RTS/CTS (which

is not supported in the 1782-JDCx) requires two additional signal lines on the serial side and it controls the data before it is received.

16. ***When should I turn Pad Mode ON? Or what configuration parameter should I check when the scanner cannot stay connected to the 1782-JDCx due to an IO size error?***
    If the DeviceNet scanner or the master device needs to be pre-configured with an input (produce) size before making a connection, or if the serial device will be sending variable data size strings, you will need to turn the PAD Mode ON. This parameter should be checked if the 1782-JDCx will not stay connected due to an IO size error.

17. ***What should my master device logic be doing to see if new data is present in a poll message?***
    The Master program logic should be looking at the Receive Record Number and comparing it with the last stored record number to monitor new data arrival in Immediate (Non-Handshake) Mode. In the Master Handshake mode, the program should be looking at the New Data bit in status and if detected, should supply the new record number and wait for the 1782-JDCx to produce data along with that record number.

18. ***What does the 1782-JDCx require for power?***
    The 1782-JDCx receives all necessary power from the DeviceNet network cable. This should be between 11 and 25 Vdc. It consumes about 2 watts of power maximum.

19. ***Where should I use a 1782-JDCx gateway vs. a 1799wr-DASCII gateway?***
    The 1782-JDCx has a form factor designed for DIN rail mounting and the 1799wr-DASCII is designed for panel mounting. The 1782-JDCx models are functionally very similar to the 1799wr-DASCII. The additional parameters of the 1782-JDCE-1 are most similar to the 1799wr-DASCII and offer some additional flexibility. Also the 1799wr-DASCII is designed for RS232/422/485 serial communication, whereas the 1782-JDCx models are designed for a specific serial communication type.

20. ***How do I access the serial ASCII data from a 1782-JDCx gateway?***
    When configuring your PLC, you have the option to memory map the ASCII data from the 1782-JDCx to a location on your PC. The ASCII data can be found at this location.

21. ***When do I need to use an EDS (electronic data sheet) file for a 1782-JDCx gateway?***
    An EDS file is not *necessary*, but it makes configuring the 1782-JDCx (changing baud rate, framing, etc.) far easier than if you were to try to change the device parameters by editing the attributes of certain class objects and instances without the aid of a configuration tool. There are many software configuration tools available with EDS readers that can be used to set up the device.

22. ***What is the proper cable to use for a 1782-JDCx gateway serial connection?***
    The proper cable selection depends upon which gateway model is used, the serial communication baud rate desired and the total cable length. Cable selection guidelines:

    1782-JDC-1: RS232 – 3 wire shielded
    1782-JDC-2: RS485 – 3 wire shielded
    1782-JDC-4: RS485/RS422 – 3 wire (RS485) shielded or 5 wire (RS422) shielded
    1782-JDCE-1: RS232 - 3 wire shielded

23. ***Where can I purchase the proper cable for a 1782-JDCx gateway serial connection?***
    You can purchase the proper cable from many suppliers and fitted with the connectors required for the gateway model and serial device being used.

If the answer to your question is not found here, or you require further assistance, please contact Western
Reserve Controls at:
1485 Exeter Road
Akron, OH 44306
330-733-6662 (phone)
330-733-6663 (fax)
For product information, e-mail: sales@wrcakron.com
For technical support, e-mail: support@wrcakron.com

D. Troubleshooting

This section identifies some of the common problems that may be observed when commissioning or operating a DeviceNet and JDCx.

Problem:
JDCx does not power up; both LED's are off.
Possible Causes:
1. Power not applied to DeviceNet cable or JDC connector.
2. Cabling not properly connected.
3. The module is damaged.

Problem:
JDCx will not communicate on the network.
Module Status LED is solid Green.
Network Status LED is flashing Green.
Possible Causes:
1. Network does not have a terminating resistor. Add a 121-ohm resistor across the CAN_H and CAN_L signals at the first and last nodes.
2. Incorrect baud rate.
3. Cabling not properly connected.
4. No messaging connections are allocated.
5. Message sent to incorrect node.
6. JDCx is not in Master device's Scan List.

Problem:
JDCx will not return data.
Module Status LED is solid Green.
Network Status LED is solid Green.
Possible Causes:
1. Message sent to incorrect node.
2. JDCx is not mapped into Master's data map

Problem:
JDCx will only update the output buffer when an overflow condition is detected.
Possible Causes:
1. The receive delimiter is not a valid value. See Table 5-1, Parameter 5.
2. The receive delimiter value is never received.

Problem:
JDCx is not being scanned after changing the ASCII transmit and receive buffer

sizes from default of 20 and 20.

Possible Causes:

Scanner’s table data may have unmapped and needs to be remapped to process the data.