United Electronic Industries

The High-Performance Alternative

DNA/DNR-SL-501 Serial Line Communication Layer — User Manual

4-port RS-232 or RS-485 (Serial Port) Layer for the PowerDNA Cube and RACKtangle

December 2010 Edition

Version 3.10

PN Man-DNx-SL-501-1210

Contacting United Electronic Industries

Mailing Address:

27 Renmar Avenue
Walpole, MA 02081
U.S.A.

For a list of our distributors and partners in the US and around the world, please see http://www.ueidaq.com/partners/

Support:

Telephone: (508) 921-4600
Fax: (508) 668-2350

Also see the FAQs and online "Live Help" feature on our web site.

Internet Support:

Support	support@ueidaq.com
Web-Site	www.ueidaq.com
FTP Site	ftp://ftp.ueidaq.com

Product Disclaimer:

WARNING!

DO NOT USE PRODUCTS SOLD BY UNITED ELECTRONIC INDUSTRIES, INC. AS CRITICAL COMPONENTS IN LIFE SUPPORT DEVICES OR SYSTEMS.

Products sold by United Electronic Industries, Inc. are not authorized for use as critical components in life support devices or systems. A critical component is any component of a life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system, or to affect its safety or effectiveness. Any attempt to purchase any United Electronic Industries, Inc. product for that purpose is null and void and United Electronic Industries Inc. accepts no liability whatsoever in contract, tort, or otherwise whether or not resulting from our or our employees' negligence or failure to detect an improper purchase.

NOTE

Specifications in this document are subject to change without notice. Check with UEI for current status.

Table of Contents

List of Figures

Chapter 1 Introduction

This document outlines the feature-set and describes the operation of the DNx-SL-501 Serial Line Communication boards. The DNA- version is designed for use with a PowerDNA Cube data acquisition system. The DNR- version is designed for use with a DNR-12 RACKtangle or a DNR-6 HalfRACK system. Both versions are functionally identical. This board (layer) offers four independent serial interfaces software-configurable as RS-232 or RS-485. Please ensure that you have the PowerDNA Software Suite installed before running examples.

1.1 Organization of this manual

This DNx-SL-501 User Manual is organized as follows:

- **Introduction**
  This section provides an overview of the document content.
- **The SL-501 layer**
  This section provides an overview of the device architecture, connectivity, and logic of the layer.
- **Wiring & Connectors**
  This section provides wiring schemes, notes, and specifications.
- **Programming with the High-Level API**
  This section explains how to create a session, how to configure the serial port bus communication of the layer, and how to interpret results on the SL-501 series layer.
- **Programming with the Low-level API**
  This section describes Low-Level API commands for configuring and using the SL-501 series layer.
- **Appendices**

  A. Accessories
  This appendix provides a list of accessories available for SL-501 layer(s).
- I**ndex**
  This is an alphabetical listing of topics covered in this manual.

Manual Conventions

To help you get the most out of this manual and our products, please note that we use the following conventions:

Tips are designed to highlight quick ways to get the job done, or reveal good ideas you might not discover on your own.

NOTE: Notes alert you to important information.

CAUTION! Caution advises you of precautions to take to avoid injury, data loss, and damage to your boards or a system crash.

Text formatted in **bold** typeface generally represents text that should be entered verbatim. For instance, it can represent a command, as in the following example: “You can instruct users how to run setup using a command such as **setup.exe**.”

1.2 Frequently Asked Questions

For frequently answered questions, application notes, and support, visit us online:

http://www.ueidaq.com/faq/

1.3 The SL-501 Layer

The SL-501 layer has the following features:

- Four (4) independent ports
- Each port software-configurable as RS-232 or RS-485
- Max speed of 256Kbit/s for RS-232 and 1Mbit/s for RS-485
- Completely independent bit rate settings for every port
- 350V isolation between ports and between ports and circuitry
- Compatible with RS-422 networks when used in RS-485 mode
- Half- and full-duplex support for RS-485

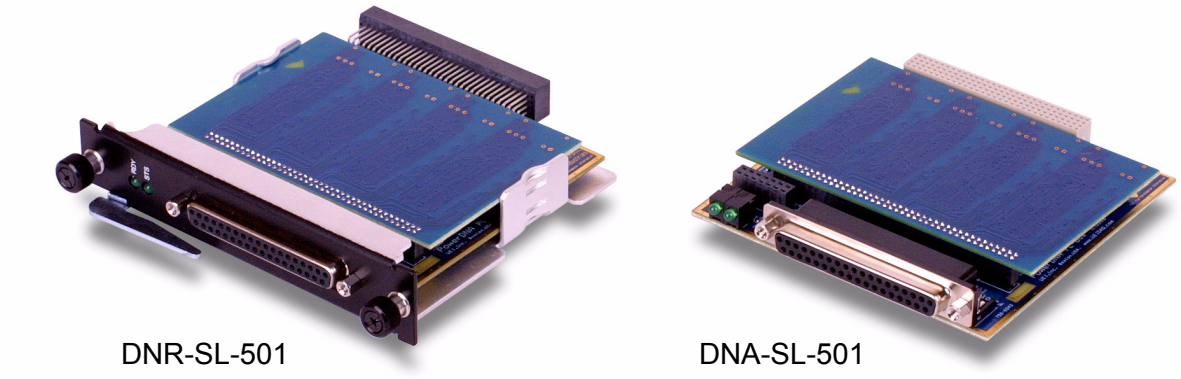


Figure 1-1 Photos of DNR-SL-501 and DNA-SL-501 Boards

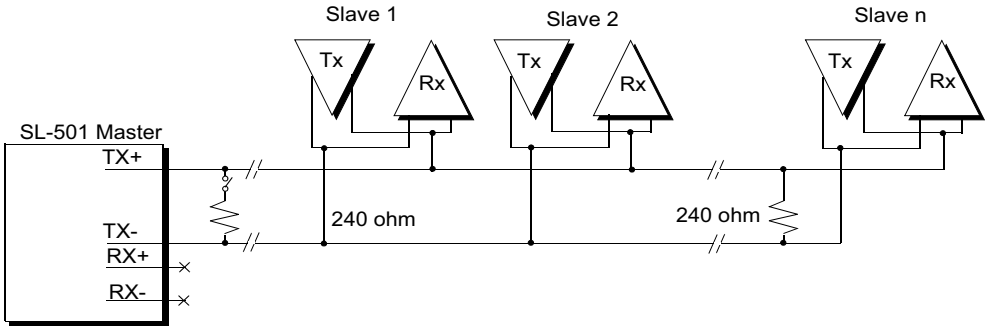
1.4 What is Serial Communication?

A serial port sends and receives data one bit at a time over one line (composed of a send, a receive, and one common ground wire).

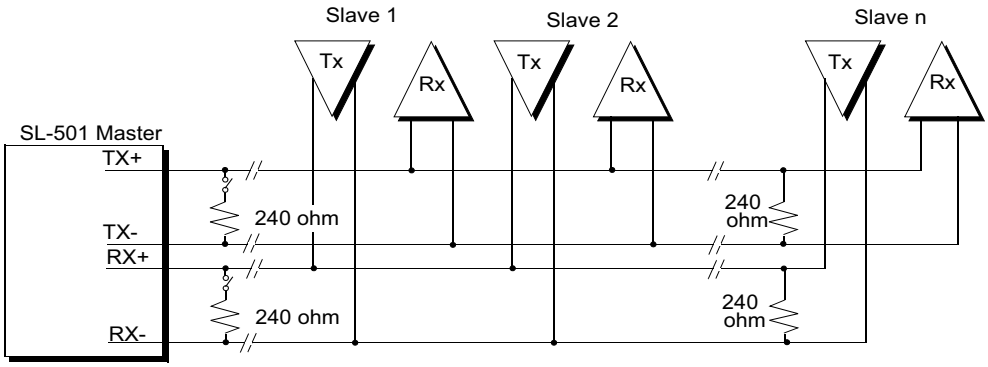
RS-232 is a standard for serial binary data interconnection between a DTE (Data terminal equipment) and a DCE (Data communication equipment) and normally operates in a bipolar range of –10 to 10V.

RS-485 (a.k.a EIA-485) is a physical layer electrical specification of a two-wire, half-duplex, multipoint serial connection. A full duplex RS-485 system can be constructed by using two twisted-pair connections (transmit/receive pairs) together as shown in **Figure 1-4**. UART data frames for RS-232 and RS-485 are shown in **Figure 1-3**.

RS- is an abbreviation for "Recommended Standard".



Two-wire Twisted-Pair Half Duplex Network with 240 ohm Terminating Resistors



Four-wire Twisted-Pair Full Duplex Network with 240 ohm Terminating Resistors

Figure 1-2 RS-485 Topologies

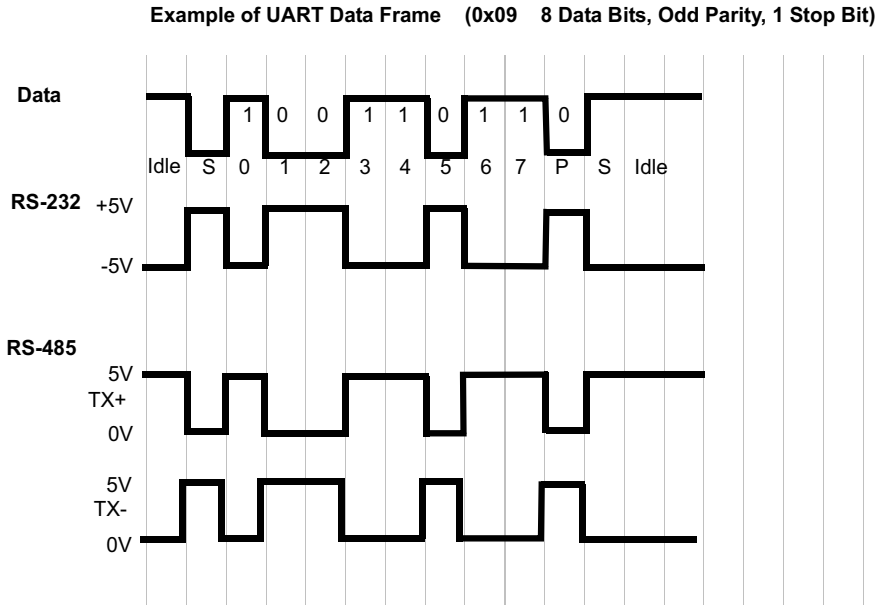


Figure 1-3 UART Data Frames for RS-232 and RS-485

1.5 Architecture

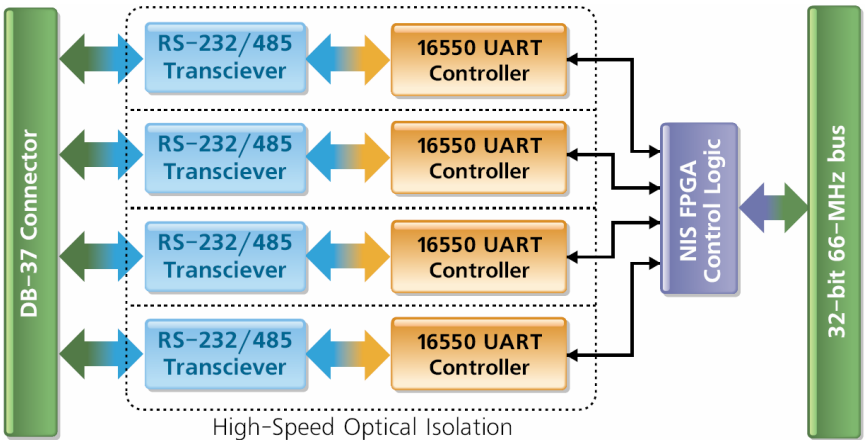


Figure 1-4. Logic Block Diagram: SL-501 Overview

An electrical signal on the RS-232/485/422 bus flows through the DB-9 connector to a DB-37 connector, and then to the MAX3160E serial transceiver chip. The MAX3160E acts as an interface to the UART16550 serial interface controller — it assists the UART16550 by handling transmission/reception to/from the serial bus.

The transceiver and controller are isolated from each other by a high-speed isolation IC capable of withstanding 350V channel-to-channel or 15kV ESD. There are four MAX3106E » isolation » UART16550 structures, one per port; isolation is per-port.

The UART16550 is in turn controlled by a FPGA Control Chip — this is the layer control chip. FPGA works in conjunction with the core module logic of the cube.

1.6 Layer capabilities

Using the RS-232 or RS-485 standard, the controller is capable of communicating at speeds of 256Kbit/s for RS-232 and 1Mbit/s for RS-485. When in RS-485 mode, the layer is compatible with RS-422 networks.

The UART16550 runs at a base-block frequency of 66MHz, with a FIFO size of 2048.

Each port has independently programmable:

- Baud/bit rate
- UART interrupt
- Timeout interrupt
- TX/RX FIFO interrupt
- Error interrupt (4 per port)

1.7 Wiring & Connectors

A DNA-CBL-COM (see **Figure 1-5**) from the 37-pin connector of the SL-501 provides four individual 9-pin ports, labeled by port as (1), (2), (3), (4).

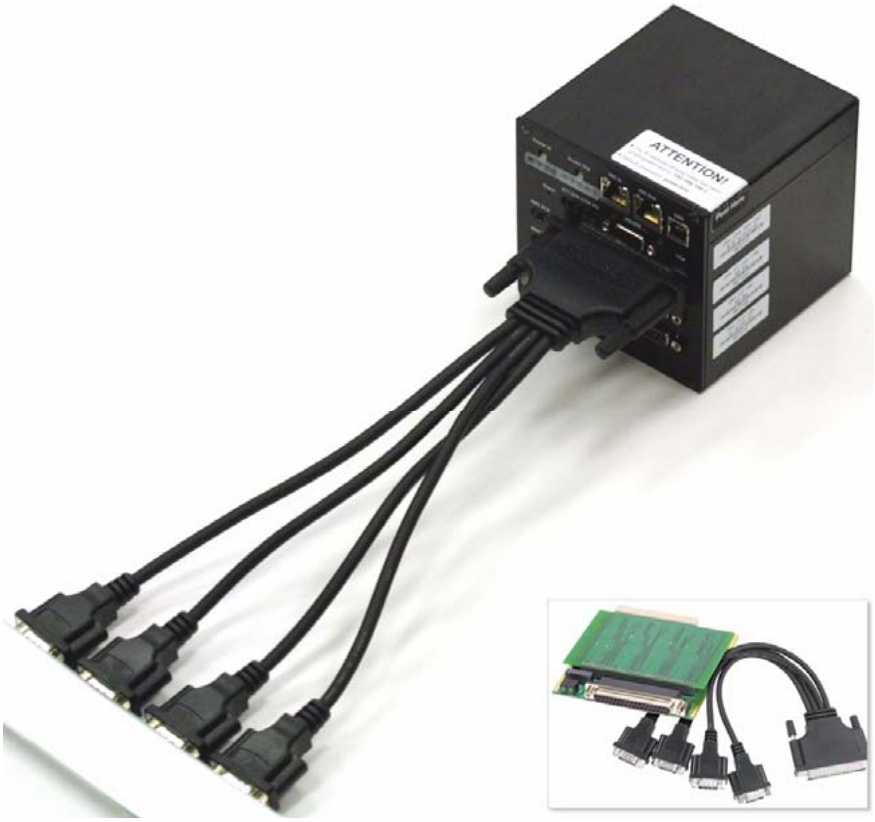


Figure 1-5. PowerDNA Cube with DNA-SL-501 and Cables

The following signals are located at the connector:

- GNDx - Isolated ground for the corresponding serial port
- TXDx/RXDx RS-232: Transmit/Receive
- RTSx/CTSx RS-232: Request to Send/Clear to Send
- TXx+/TXx- RS-485: Transmit pair
- RXx+/RXx- RS-485: Receive pair

The B-size 37-pin female D-Sub connector on the SL-501 is divided into four 9-pin D-connector serial ports by a DNA-CBL-COM cable with the following pinout:

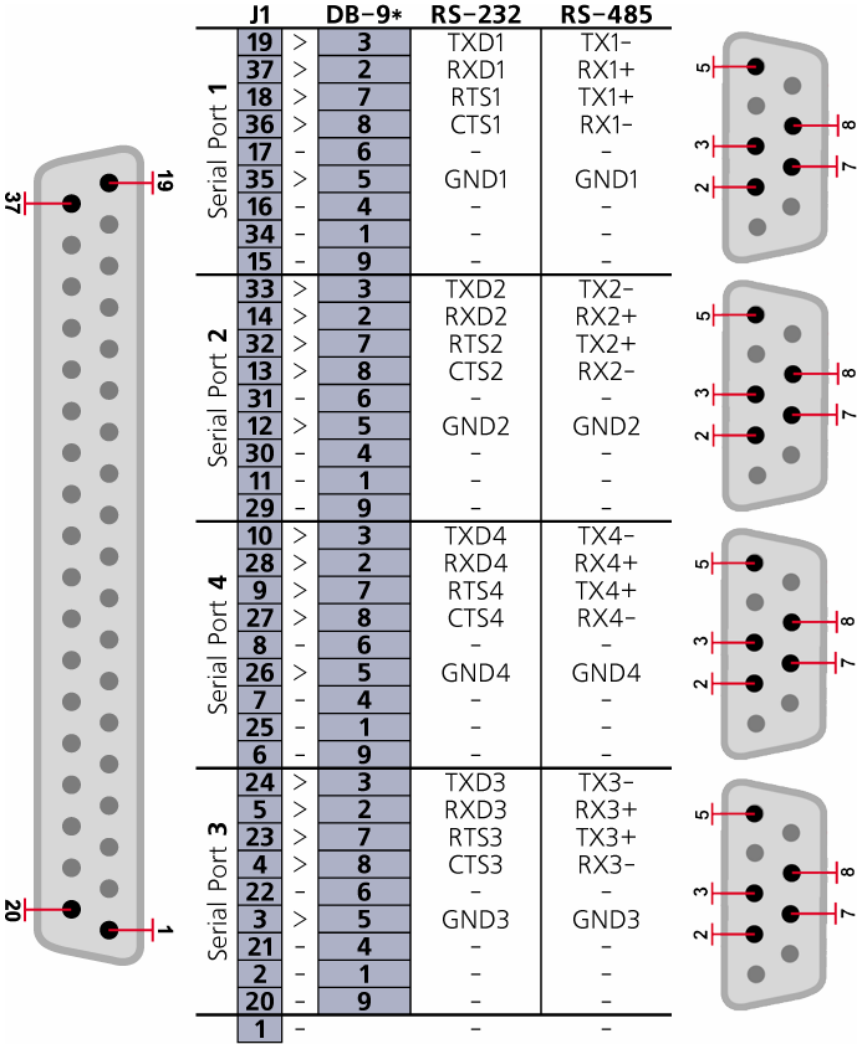


Figure 1-6 . Logic Block Diagram: DNx-SL-501 Pinout

1.8 Jumper Settings for DNA Version

The base board of a DNA-SL-501 module (layer) has a jumper block that assigns the position of the module within a PowerDNA Cube. The jumpers must be set to match the physical position of an I/O board or layer in the Cube. This function is not required with DNR version boards.

NOTE: Since all layers are assembled in Cubes before shipment to a customer, you should never have to change a jumper setting unless you change a layer from one position to another in the field.

A diagram of the jumper block is shown in **Figure 1-7**. To set the layer address, place jumpers as shown in **Figure 1-7**.

		Layer's Position as marked on the Faceplate*					
		I/O **1**	I/O **2**	I/O **3**	I/O **4**	I/O **5**	I/O **6**
Jx Pins	9-10	● ●	○ ○	● ●	○ ○	● ●	○ ○
	11-12	● ●	● ●	○ ○	○ ○	● ●	● ●
	13-14	● ●	● ●	● ●	● ●	○ ○	○ ○
	15-16	● ●	● ●	● ●	● ●	● ●	● ●

* *All I/O Layers are sequentially enumerated from top to the bottom of the Cube*

○ ○ - **Open** ● ● - **Closed**

Figure 1-7. Diagram of DNA-SL-501 Layer Position Jumper Settings

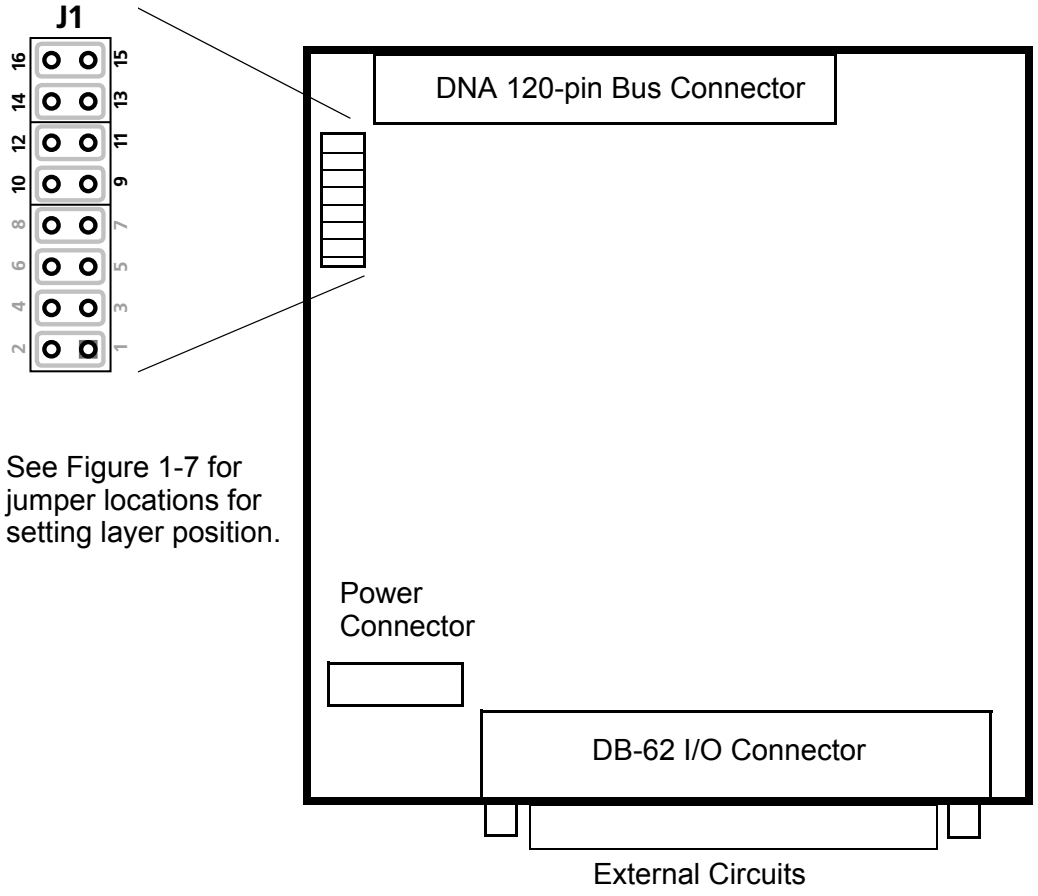


Figure 1-8. Physical Layout of DNA-SL-501 Layer Base Board (60x)

Chapter 2 Programming with UeiDaq Framework

This section describes how to program the DNx- SL-501 using the UeiDaq's Framework API.

The UeiDaq Framework is object oriented and its objects can be manipulated in the same manner in different development environments, such as Visual C++, Visual Basic, LabVIEW, or DASYLab.

Framework comes bundled with examples for supported programming languages. These are located under the UEI programs group in:

```
Start » Programs » UEI » Framework » Examples
```

The following subsections focus on the C++ API, but the concept is the same regardless of programming language.

Please refer to the "UeiDaq Framework User Manual" to get more information on using other programming languages.

2.1 Creating a Session

The Session object controls all operations on your PowerDNA device. Therefore, the first task is to create a session object:

```
CUeiSession session;
```

2.2 Configuring the Serial Ports

The framework uses resource strings to select which device, subsystem, and channels to use within a session. The resource string syntax is similar to a web URL:

```
<device class>://<IP address>/<Device Id>/
<Subsystem><Channel list>
```

For PowerDNA the device class is **pdna**.

For example, the following resource string selects serial ports 0,2,3 on device 1 at IP address 192.168.100.2:

```
"pdna://192.168.100.2/Dev1/Com0,2,3".
```

In addition to the resource, you will also configure:

- Port mode (RS-232, RS-485 half-duplex or RS-485 full duplex)
- Bit rate (bits per second)
- Number of data bits
- Parity
- Number of stop bits

```
// Configure Com ports 0, 2 and 3 on device 1
session.CreateSerialPort(pdna://192.168.100.2/
Dev1/Com0,2,3,
UeiSerialModeRS232,
UeiSerialBitsPerSecond57600,
UeiSerialDataBits8,
UeiSerialParityNone,
UeiSerialStopBits1);
```

2.3 Configuring the Timing

You need to configure the SL-501 to use the “messaging” timing mode. A message is represented by an array of bytes:

The SL-501 can be programmed to wait for a certain number of bytes to be received before notifying the session.

It is also possible to program the maximum amount of time to wait for the specified number of bytes before notifying the session.

The following sample shows how to configure the messaging I/O mode to be notified when 10 bytes have been received or every second, whichever is less. (Note that if the serial port receives less than 10 bytes per second, it will return whatever number of bytes are available every second).

```
session.ConfigureTimingForMessagingIO(10, 1.0);
```

2.4 Reading Data

Reading data from the SL-501 is done using a reader object. As there is no multiplexing of data (contrary to what’s being done with AI, DI, or CI sessions), you need to create one reader object per serial port to be able to read from each port in the port list.

The following sample code shows how to create a reader object tied to port 1 and read up to 10 bytes from the serial port.

```
// Create a reader and link it to the session’s
// stream, port 1
reader = new
CUeiSerialReader(session.GetDataStream(), 1);
// read up to 10 bytes, numBytesRead contains the
// number of bytes actually received.
Unsigned char bytes[10];
reader->Read(10, bytes, &numBytesRead);
```

2.5 Writing Data

Writing data to the SL-501 is done using a writer object. As there is no multiplexing of data (contrary to what’s being done with AO, DO, or CO sessions), you need to create one writer object per serial port to be able to write to each port in the port list.

The following sample code shows how to create a writer object tied to port 2 and send one byte to the serial port.

```
// Create a writer and link it to the session’s
// stream, port 2
writer = new
CUeiSerialWriter(session.GetDataStream(), 2);

// Write 1 byte, numBytesWritten contains the
// number of bytes actually sent
unsigned char bytes[2] = {0x23, 0};
writer->Write(1, bytes, &numBytesWritten);
```

2.6 Cleaning-up the session

The session object cleans itself up when it goes out of scope or when it is destroyed. However, you can manually clean up the session (to reuse the object with a different set of channels or parameters).

```
session.CleanUp();
```

Chapter 3 Programming with the Low-Level API

The low-level API offers direct access to PowerDNA DaqBIOS protocol and allows you to access device registers directly.

We recommend that, where possible, you use the UeiDaq Framework tool (*see Chapter 2*), as it is easier to use. You should need to use the low-level API only if you are using an operating system other than Windows.

Please refer to the `API Reference Manual` document under:

`Start » Programs » UEI » PowerDNA » Documentation`

for pre-defined types, error codes, and functions for use with this layer.

3.1 Baud Rate Generation with the SL-501

The DNx-SL-501 layers with logic revision 02.10.06or later provide three sources for baud rate generation:

- 24 MHz and 66 MHz baseclocks. These can be used to keep software compatibility with three DNx-SL-501 layers with logic revisionfrom 02.0E.05 to 02.10.05
- Per-channel PLL (Phase Locked Loop) baud rate generator — new functionality starting with logic revision 02.10.06.

Through the software, one of these 3 sources can be selected as the baseclock. The baseclock is then divided by an integer value to get the “16 times baud clock” frequency that is required by the serial transmit and receive circuitry.

Please refer to the Sample501.c program from the PowerDNA software suite, which is usually installed in the C:\Program Files\UEI\SDK\Examples\Visual C++\ folder. The standard baud rates (19200, 9600, etc.) can be set using the provided channel confiiguration helper macros where the desired rate constant DQ_SL501_BAUD_XXX) is passed as a parameter to the CFG_501() helper macro. This configuration is then passed to the DqAdv501SetChannelCfg() function.

To set customized baud rates, the baud constant DQ_SL501_BAUD_CUST is supplied to the CFG_501() helper macro and then to the DqAdv501SetChannelCfg() function. The DqAdv501BaseClock() function can be used to select whether the 66MHz or 24 MHz baseclock is used. For any given baud rate, one baseclock setting wil produce an actual rate that is closer to the desired rate. When the DqAdv501Set Baud() is used to set the rate, the value returned by the ”realbaud parameter” may be examined to determine which baseclock setting produces a baud setting closest to the ideal value. To use the variable PLL as a baseclock source, bitwise OR the constant (1L<<DQ+SL501_BAUD_PLL_SH) to the config value passed to the DqAdv501SetChannelCfg() function and set the rate using the DqAdv501SetBaud() function.

Appendices

A. Accessories

This appendix provides a list of accessories available for SL-501 layer(s).

DNA-CBL-COM Cable

A DNA-CBL-COM Cable with a DB-37 on one end and 4 DB-9 Serial Port Connectors on the other end as shown in Figure 3. See Figure 4 for pinout. One cable may be used with each SL-501 board in the PowerDNA Cube.

Index