DNx-AI-207

User Manual

16-Channel, 18-bit, Sequential Sampling, Differential Analog Input Board with Cold-Junction Compensation for the PowerDNA Cube and RACK Series Chassis

March 2025

PN Man-DNx-AI-207

CE

Contacting United Electronic Industries

Mailing Address:
249 Vanderbilt Avenue
Norwood, MA 02062
U.S.A.

Shipping Address:
24 Morgan Drive
Norwood, MA 02062
U.S.A.

For a list of our distributors and partners in the US and around the world, please contact a member of our support team:

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: uei.support@ametek.com
Website: www.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 / AMETEK 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 / AMETEK product for that purpose is null and void and United Electronic Industries / AMETEK 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.

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

Table of Contents

Table of Figures

Chapter 1 Introduction

This document outlines the feature set and use of the DNx-AI-207, 16-channel analog input boards.

The following sections are provided in this chapter:

- Organization of this Manual (Section 1.1)
- The AI-207 Board Overview (Section 1.2)
- Features (Section 1.3)
- Specification (Section 1.4)
- Device Architecture (Section 1.5)
- Indicators (Section 1.6)
- Connectors and Wiring (pinout) (Section 1.7)
- Data Representation (Section 1.8)

1.1 Organization of this Manual

This AI-207 User Manual is organized as follows:

- **Introduction**
  Chapter 1 provides an overview of DNx-AI-207 features, device architecture, connectivity, and logic.
- **Programming with the High-Level API**
  Chapter 2 provides an overview of the how to create a session, configure the session, and interpret results with the Framework API.
- **Programming with the Low-Level API**
  Chapter 3 is an overview of low-level API commands for configuring and using the AI-207 series board.
- **Appendix A - Accessories**
  This appendix provides a list of accessories available for use with the DNx-AI-207 board.
- **Index**
  This is an alphabetical listing of the topics covered in this manual.

NOTE: A glossary of terms used with the PowerDNA Cube/RACK and I/O boards can be viewed or downloaded from www.ueidaq.com.

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**.”

Bold typeface will also represent field or button names, as in “Click **Scan Network**.”

Text formatted in `fixed` typeface generally represents source code or other text that should be entered verbatim into the source code, initialization, or other file.

Examples of Manual Conventions

Before plugging any I/O connector into the Cube or RACKtangle, be sure to remove power from all field wiring. Failure to do so may cause severe damage to the equipment.

Usage of Terms

Throughout this manual, the term “Cube” refers to either a PowerDNA Cube product or to a PowerDNR RACKtangle™ rack mounted system, whichever is applicable. The term DNR is a specific reference to the RACKtangle, DNA to the PowerDNA I/O Cube, and DNx to refer to both.

1.2 The AI-207 Board Overview

The DNx-AI-207 is a 16-channel sequential sampling A/D board that features 18-bit resolution and 12 software-selectable input gain ranges.

DNA-AI-207, DNR-AI-207, and DNF-AI-207 boards are compatible with the UEI Cube, RACKtangle, and FLATRACK chassis respectively. These board versions are electronically identical and differ only in mounting hardware. The DNA version is designed to stack in a Cube chassis. The DNR/F versions are designed to plug into the backplane of a RACK chassis.

The DNx-AI-207 is pin compatible with UEI's DNx-AI-217 board, which offers more resolution, higher sample rates and simultaneously sampling inputs.

1.2.1 ADC Input Configuration

Each AI-207 channel is sequentially sampled with a maximum board throughput of 16 kS/s (16 kHz). This configuration allows an application with a single channel enabled to sample that channel at up to 16 kS/s. For applications monitoring multiple channels, each additional channel must share the 16 kHz maximum bandwidth (e.g. maximum of 1 kS/s for 16 channels enabled, 2 kS/s for 8 channels enabled, etc.).

Additionally, the DNx-AI-207 provides a dedicated CJC channel that can be used for reading from the built-in CJC sensor on UEI's DNA-STP-AI-U or AI-207TC terminal panels. When used with DNA-STP-AI-U or 207TC panel, the DNx-AI-207 also features a direct connection to thermocouples (with open TC detection). The software included will perform all required TC linearization and CJC compensation and return data in °C or °F if desired.

Another key feature of the DNx-AI-207 is the oversampling engine, allowing the board to automatically acquire as many samples as possible for the given gain/speed and average them, which dramatically improves the noise floor.

1.2.2 Compatibility

If input cross talk and channel settling time issues are a problem, even when connected to high impedance signal sources, consider a pin-compatible simultaneously sampling analog input board, such as the AI-217.

1.2.3 Environmental Conditions

As with all UEI PowerDNA boards, the DNx-AI-207 can be operated in harsh environments and has been tested at 5*g* vibration, 50*g* shock, -40 to +85°C temperature. Each board provides 350 V_{rms} isolation between the board and its enclosure, or any other installed boards.

1.2.4 Software Support

Software included with the DNx-AI-207 provides a comprehensive yet easy to use API that supports all popular operating systems including Windows, Linux, real-time operating systems such as QNX, RTX, VxWorks and more. The UEIDAQ framework comes with bindings for various programming languages such as C, C++, C#, VB.NET and scientific software packages such as LabVIEW and Matlab, as well as supporting OPC servers.

1.3 Features

The AI-207 analog input board offers the following features:

- 16 fully differential channels; additional dedicated CJC channel
- Maximum sampling rate of 1 kHz per channel when all channels are enabled
- 18-bit resolution
- ±10 V input range
- Programmable gains: 1, 2, 4, 8, 10, 20, 40, 80, 100, 200, 400, 800
- Overvoltage protection (-40V to +55V)
- Dynamic autozero support
- Embedded averaging engine

1.4 Specification

The technical specifications for the DNx-AI-207 are listed in **Table 1-1** below.

Table 1-1 DNx-AI-207 Technical Specifications

Number of channels:	16 fully differential plus 1 single-ended dedicated CJC channel
Programmable DIO line	1 (external trigger)
ADC resolution	18 bits
Sampling rate	1 S/s - 16 kS/s per channel; 16 kS/s max aggregate for entire board
FIFO size	512 samples
Input bias current	±5nA max, ±0.5nA typical
Input impedance	10MΩ
Gains	1,2,4,8,10,20,40,80,100,200,400,800
Frontend bandwidth	48kHz @ -3dB
Common mode rejection	100dB typical
Oversampling ratio	2 to 8192, selected automatically
Accuracy	±287.59 µV at ±10 V input range. Please see the DNx-AI-207 Datasheet for detailed tables and graphs.
Isolation	350 Vrms
Overvoltage protection	-40V to +55V
Power consumption	1.4W (stand-by); 2.2W max
Operating temp. (tested)	-40°C to +85°C
Operating humidity	95%, non-condencing
Vibration *IEC 60068-2-6* *IEC 60068-2-64*	5 g, 10-500 Hz, sinusoidal 5 g (rms), 10-500Hz, broadband random
Shock *IEC 60068-2-27*	50 g, 3 ms half sine, 18 shocks @ 6 orientations 30 g, 11 ms half sine, 18 shocks @ 6 orientations
MTBF	637,000 hours

1.5 Device Architecture

As shown in **Figure 1-1**, the DNX-AI-207 board has multiplexed inputs with a single 18-bit converter.

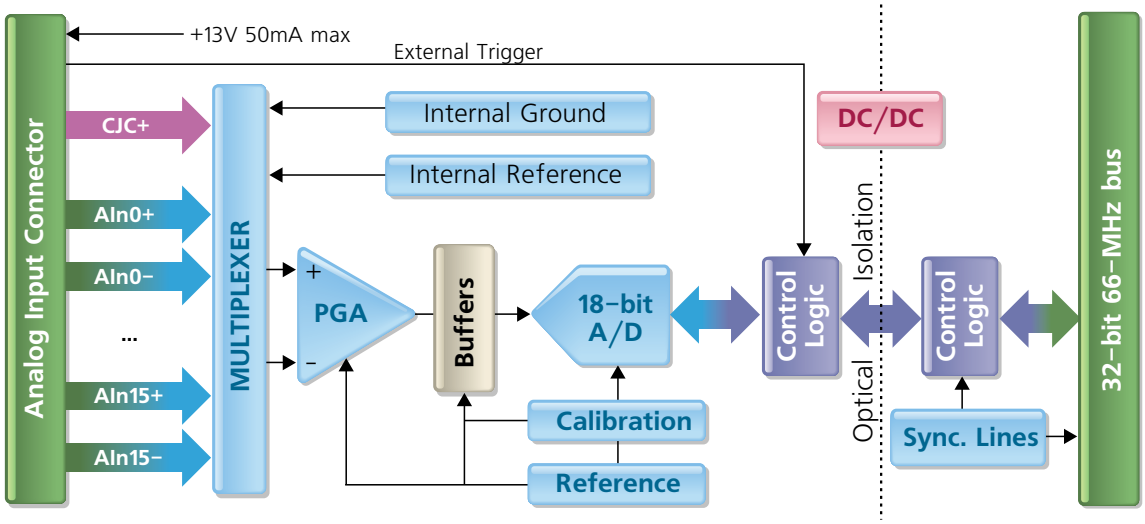


Figure 1-1 Block Diagram of DNx-AI-207 I/O Board

The DNx-AI-207 analog input board features 16 differential input channels designed for mid- to low-speed high-resolution signal measurement.

The DNx-AI-207 provides a dedicated CJC channel that can be used for reading from a built-in CJC sensor on the DNA-STP-AI-U accessory terminal panel. Static CJC compensation may be used when no CJC sensor is available (such as when the I/O board is connected to a DNA-STP-37 accessory panel).

The DNx-AI-207 has an input multiplexer that sequentially selects each of the available input signals for input to a cascaded fully differential programmable gain amplifier (PGA) and then to an 18-bit successive approximation (SAR) A/D converter. Logic on the isolated side controls channel switching, settling time delays, and the conversion process. Also, it reads data from the converter at the maximum possible rate and sends it over the isolation barrier to the non-isolated logic for the further processing. The following additional channels, which are used to improve quality of the acquired signal, are internally connected to the multiplexers: internal ground, internal 2.5000V reference, and CJC channel.

Users configure channels (e.g., enable/disable, gains, etc.) via a channel list, which gets processed by the PowerDNA firmware. Every channel is allocated a required delay for best possible settling of the analog signal. Longer settling times are required for greater gains. The rest of the time between channel sampling is used for oversampling a single channel. The number of averages used in this computation varies from 2 to 8096, depending on the selected acquisition rate and, because of the complexity of channel list processing, is set automatically by the firmware.

1.5.1 Multiplexer & Programmable Gain

The analog input lines (Ain+ and Ain- in **Figure 1-1**) enter through the DB-37 connector pins into high performance, fault-protected multiplexers, which sequentially mux in each channel that is user-enabled. Channels are sampled at the user-programmed sample rate. When the CJC is enabled, the multiplexer also switches in the CJC channel, and when the autozero feature is enabled, the multiplexer additionally switches in internal ground to provide an autozero reference for each gain setting selected.

The AI-207 supports 12 user-programmable gains: 1, 2, 4, 8, 10, 20, 40, 80, 100, 200, 400, and 800x gains.

1.5.2 Maximum Sample Rate

When acquiring thermocouple data, UEI recommends a maximum sample rate of 10 Hz.

For non-thermocouple applications, several configuration options affect the maximum allowable sample rate for a channel: the number of analog input channels enabled, the gain selected, and whether or not autozero is enabled. (See Section 1.5.3 for more information about autozero functionality).

The number of input channels passing through the input multiplexer will affect the maximum allowable sample rate per channel:

- For example, if all 16 channels are enabled, with no CJC and no autozero, the result is 16 inputs to the multiplexer. With a maximum aggregate sampling rate of 16 kS/s (16 kHz) for the board, the maximum per channel will be 1 kS/s (1 kHz).
- If only 8 channels are enabled, with no CJC and no autozero, that results in 8 inputs to the multiplexer. With a maximum sampling rate of 16 kS/s for the board, the maximum per channel will be 2 kS/s.

Note that each gain uses a gain-specific initial settling time to ensure measurement accuracy, which can limit the sampling rate of the channel. The Minimum Settling Time (see **Table 1-2**) is the shortest time that the firmware allows a channel to settle. When the sample rate and channel configuration are programmed, the firmware allocates the minimum time for each channel depending on the gain selected.

Table 1-2 Gains vs Settling Times and Maximum Sampling Rates

Input Range	Gain	Min Settling Time (us)	Maximum Sampling Freq (Hz)
±10V	1	60	Max sampling rate (16 kHz)
±5V	2	70	14,285
±2.5V	4	80	12,500
±1.25V	8	90	11,111
±1V	10	100	10,000
±500mV	20	120	8,333
±250mV	40	150	6,666
±125mV	80	160	6,250

Table 1-2 Gains vs Settling Times and Maximum Sampling Rates

Input Range	Gain	Min Settling Time (us)	Maximum Sampling Freq (Hz)
±100mV	100	180	5,555
±50mV	200	200	5,000
±25mV	400	240	4,166
±12.5mV	800	280	3,571

Enabling autozero also affects the maximum sample rate of the channels. The following section (Section 1.5.3) describes this.

1.5.3 Autozero

The DNx-AI-207 provides an automated offset autozero feature, which removes any offset fluctuations for every signal reading when running the device over temperature and over time. This reduces temperature drift to a few microvolts over the full specified range.

When autozero is enabled, internal references are sampled and measurements are stored for the autozeroing calculation. The firmware schedules the sampling of the internal reference for the different gains programmed through the channel multiplexer. This scheduling will lower the maximum sample rate of channels.

1.5.4 Oversampling Engine

Another feature, the oversampling engine, permits the DNx-AI-207 to acquire as many samples as possible for the given gain/speed and automatically average them, dramatically improving noise rejection. This happens under firmware control and is not programmable by the user.

1.5.5 Data Storage and Timestamping

Acquired samples are stored in a 512 word FIFO, in the order that channels are enabled in the channel list.

Users can optionally timestamp the acquired samples. The timestamp will be stored as the first or last entry in the user-defined list of channels to acquire. Whether to store the timestamp in the first position or in the last position is user-programmable.

1.6 Indicators

The DNx-AI-207 indicators are described in **Table 1-3** and illustrated in **Figure 1-2**.

Table 1-3 AI-207 Indicators

LED Name	Description
RDY	Indicates board is powered up and operational
STS	Indicates which mode the board is running in: • **OFF**: Configuration mode, (e.g., configuring channels, running in point-by-point mode) • **ON**: Operation mode, (e.g., running in VMap or ACB mode)

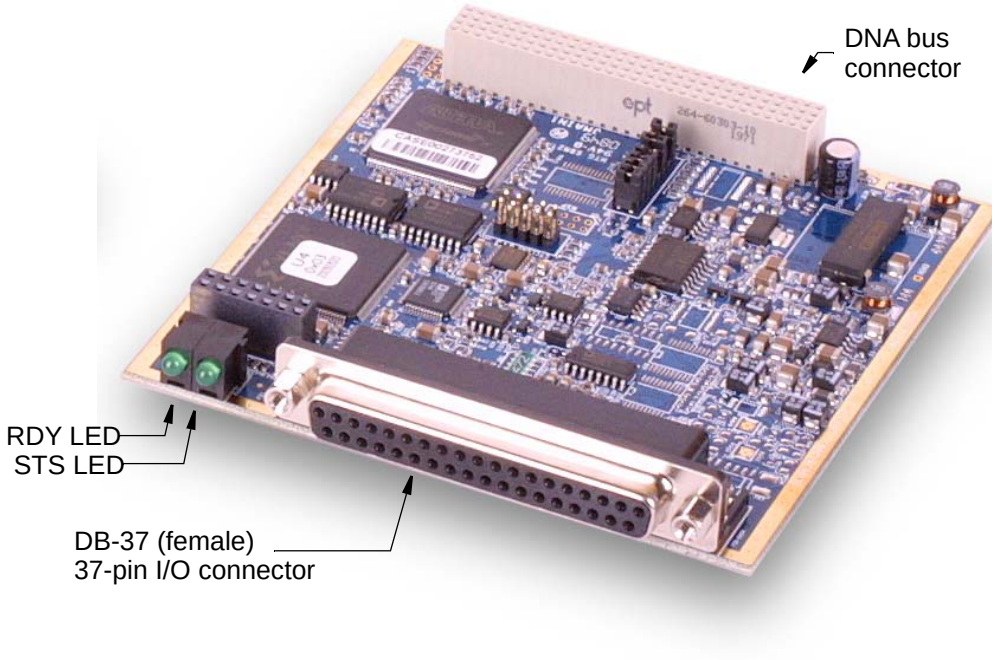


Figure 1-2 Photo of DNA-AI-207 Analog Input Board

1.7 Connectors and Wiring (pinout)

The AI-207 analog input board uses a 37-pin female D-Sub connector with the following pinout:

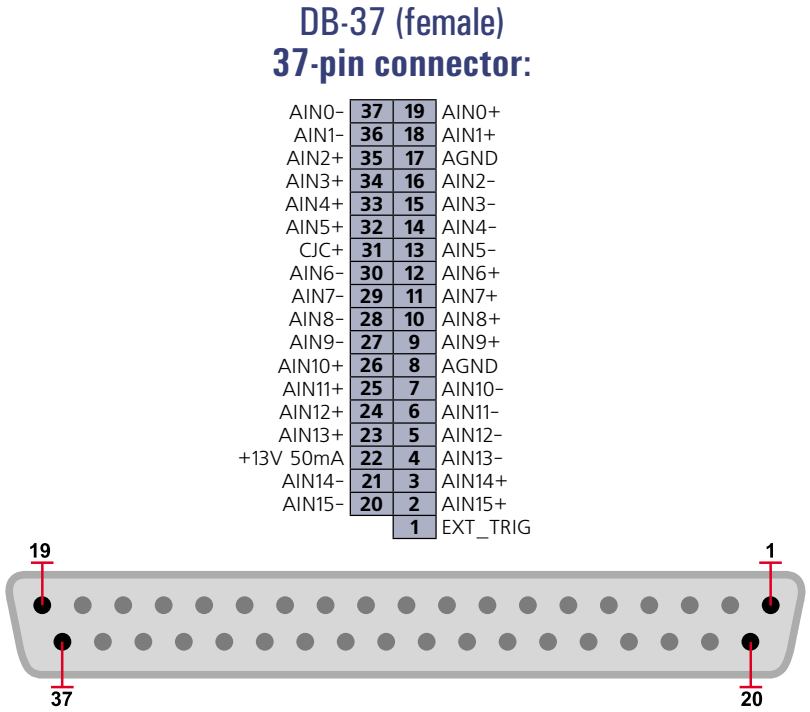


Figure 1-3 Pinout of the DNx-AI-207 Analog Input Board

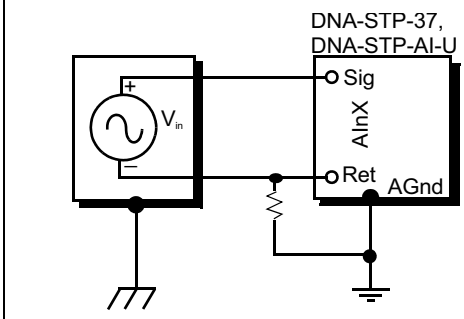
The AI-207 uses a 37-pin D-sub connector. The following signals are located at the connector:

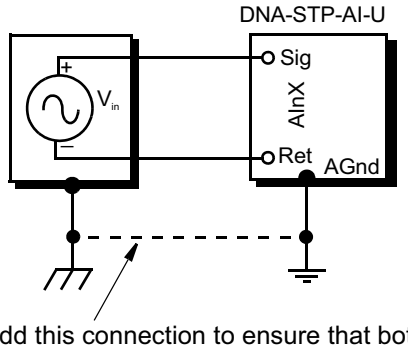
- AIN0± – AIN15± input channels, differential mode

  The AI-207 measures inputs between AIn+ and AIn- as long as common mode voltage is within limits. For single ended connections, AIn- should be connected to AGND
- +13V 50mA provides current
- CJC+ cold junction compensation return line from DNA-STP-AI-U
- AGND board analog ground, isolated from system ground
- EXT_TRIG accepts an external trigger signal to the board

1.7.1 Analog Input Ground Connections

To avoid errors caused by common mode voltages on analog inputs, follow the recommended grounding guidelines in **Figure 1-4** below.

<table>
<tr><th rowspan="3">Input Configuration</th><th colspan="2">Type of Input</th></tr>
<tr><th>Floating</th><th>Grounded</th></tr>
<tr><th>Typical Signal Sources:
Thermocouples
DC Voltage Sources
Instruments or sensors
with isolated outputs</th><th>Typical Signal Sources:
Instruments or sensors
with non-isolated outputs</th></tr>
<tr><td>Differential</td><td> 

The resistor (10k < R < 100k) provides a return path to ground for bias current.</td><td> 

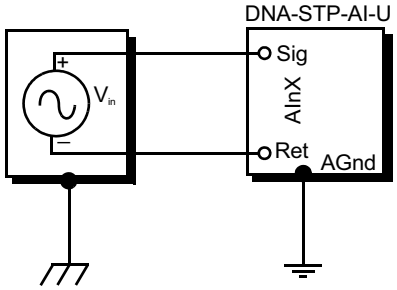
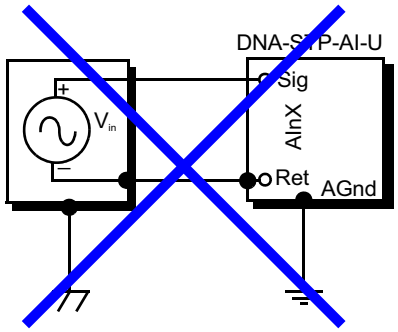
Add this connection to ensure that both grounds are at the same potential.</td></tr>
<tr><td>Single-Ended, Ground Referenced</td><td> 
</td><td>NOT RECOMMENDED 
</td></tr>
</table>

Figure 1-4 Recommended Ground Connections for Analog Inputs

Because all analog input channels in AI-201/202/207/208/225 boards are isolated as a group, you can connect board AGND to the ground of the signal source and eliminate the resistors shown in **Figure 1-4** for floating differential input signals.

1.8 Data Representation

The AI-207 board is designed with 18-bit A/D converters. The AI-207 board can return 18-bit straight binary data in 32-bit words.

The 18-bit data is represented as follows:

Bit	Name	Description	Reset State
17-0	ADCDATA	Upper 18 bits of data, straight binary	<pos>

<pos> represents a position in the output buffer. Upon reset, every entry in the output buffer is filled with its relative position number. If you read sequential data, it could mean the ADC failed to start.

To convert data into floating point, use the following formula:

$$\text{Volts} = \frac{(\text{Raw\&3FFFF}) \times \left(\frac{20\text{V}}{2^{18}-1}\right) - 10\text{V}}{(\text{gain factor})}$$

1.8.1 CJC Data

Raw CJC Voltage from the AI-207 may be represented as:

$$T_{\text{Kelvin}} = \frac{\text{CJCVoltage}}{0.00295}$$

For example, if the voltage read from Channel 33 (the CJC channel) is 0.87, the CJC Temperature is:

Temp. Scale	Calculation	CJC Temperature
Kelvin	0.87/0.00295	294.9 °K
Celsius	294.9 – 273.15	21.75 °C
Fahrenheit	1.8 x 21.75 + 32	71.75 °F

Chapter 2 Programming with the High-Level API

This chapter provides the following information about using the UeiDaq high-level Framework API to program the DNx-AI-207:

- About the High-level Framework (Section 2.1)
- Creating a Session (Section 2.2)
- Configuring Channels (Section 2.3)
- Configuring Autozero (Section 2.4)
- Configuring the Timing (Section 2.5)
- Reading Data (Section 2.6)
- Cleaning-up the Session (Section 2.7)

2.1 About the High-level Framework

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

UeiDaq Framework is bundled with examples for supported programming languages. Examples are located under the UEI programs group in:

- *Start » Programs » UEI » Framework » Examples*

The following sections focus on the C++ API, but the concept is the same no matter which programming language you use.

Please refer to the “UeiDaq Framework User Manual” for more information on use of other programming languages.

2.2 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;
```

2.3 Configuring Channels

UeiDaq 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 such as:

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

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

2.3.1 Voltage Measurement

To program the analog input circuitry, configure the channel list using the session’s object method “`CreateAIChannel`”.

For example, the following resource string selects analog input channels 0,2,3,4 on device 1 at IP address 192.168.100.2: “pdna://192.168.100.2/Dev1/Ai0,2,3,4”

The gain applied on each channel is specified by using low and high input limits.

For example, the AI-207 available gains are 1, 2, 4, 8, 10, 20, 40, 80,100, 200, 400, 800 and the maximum input range is [-10V, 10V].

To select the gain of 100, you need to specify input limits of [-0.1V, 0.1V].

```
// Configure channels 0,1 to use a gain of 100 in differential mode
session.CreateAIChannel("pdna://192.168.100.2/Dev0/Ai0,1",
            -0.1, 0.1,
            UeiAIChannelInputModeDifferential);
```

2.3.2 Thermocouple Measurement

Thermocouple measurements are configured using the Session object's method "CreateTCChannel".

The measurements will be scaled in the unit specified by the "temperature scale" parameter.

Depending on your hardware, you can specify whether the scaling calculation will use a constant cold-junction temperature or whether you will measure it from a sensor built in the DNx-STP-AIU connector block

```
// Add 4 channels (0 to 3) to the channel list and configure
// them to measure a temperature between 0.0 and 1000.0 degrees C
// from type J thermocouples, scale temperatures in Celsius
// degrees and using CJC built-in compensation from the STP-AI-U,
// in differential mode.

session.CreateTCChannel("pdna://192.168.100.2/dev0/Ai0:3",
                        0, 1000.0,
                        UeiThermocoupleTypeJ,
                        UeiTemperatureScaleCelsius,
                        UeiColdJunctionCompensationBuiltIn,
                        25.0,
                        "",
                        UeiAIChannelInputModeDifferential);
```

2.4 Configuring Autozero

The AI-207 offers an offset autozero, which removes any offset fluctuations over the temperature range and/or run time for every acquired signal reading. This reduces temperature drift to a few microvolts over the full specified range.

The autozero feature is enabled on a per board basis. By enabling a single channel, all channels will use this feature.

Autozero must be configured before the session start. Use the following code to enable autozero:

```
CUeiAIChannel* aiChan =
                 dynamic_cast<CUeiAIChannel*>(session.GetChannel(0));

aiChan->EnableAutoZero(true);
```

Alternatively, pass a `false` condition to `EnableAutoZero()` to disable the feature.

2.5 Configuring the Timing

You can configure the AI-207 to run in simple mode (point by point) or buffered mode (ACB mode).

In simple mode, the delay between samples is determined by software on the host computer.

In buffered mode, the delay between samples is determined by the AI-207 on-board clock.

The following sample shows how to configure the simple mode. Please refer to the “UeiDaq Framework User Manual” to learn how to use the other timing modes.

```
session.ConfigureTimingForSimpleIO();
```

2.6 Reading Data

Reading data from the AI-207 is done using a reader object. There is a reader object to read raw data coming straight from the A/D converter. There is also a reader object to read data already scaled to volts or mV/V.

The following sample code shows how to create a scaled reader object and read samples.

```
// Create a reader and link it to the session’s stream
CueiAnalogScaledReader reader(session.GetDataStream());

// read one scan, the buffer must be big enough to contain
// one value per channel
double data[2];
reader.ReadSingleScan(data);
```

2.7 Cleaning-up the Session

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

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

Chapter 3 Programming with the Low-Level API

This chapter provides the following information about programming the AI-207 using the low-level API:

- About the Low-level API (Section 3.1)
- Low-level Functions (Section 3.2)
- Low-level Programming Techniques (Section 3.3)
- Programming the AI-207 (Immediate Mode) (Section 3.4)
- Programming Scan Rate (Section 3.5)
- Configuring Channels & Scan Rate in ACB Mode (Section 3.6)

3.1 About the Low-level API

The low-level API provides direct access to the DAQBIOS protocol structure and registers in C. The low-level API is intended for speed-optimization, when programming unconventional functionality, or when programming under Linux or real-time operating systems.

When programming in Windows OS, however, we recommend that you use the UeiDaq high-level Framework API (see **Chapter 2**). The Framework extends the low-level API with additional functionality that makes programming easier and faster.

For additional information regarding low-level programming, refer to the “PowerDNA API Reference Manual” located in the following directories:

- On Linux systems:
  <PowerDNA-x.y.z>/docs
- On Windows systems:
  Start » All Programs » UEI » PowerDNA » Documentation

3.2 Low-level Functions

Table 3-1 provides a summary of AI-207-specific functions. All low-level functions are described in detail in the PowerDNA API Reference Manual.

Table 3-1 Summary of Low-level API Functions for DNx-AI-207

Function	Description
`DqAdv207Read`	Configures and reads data (in point-to-point mode) On first call, configures AI-207 with user-programmed channel list parameters. The default sample rate is set to 10 Hz; however, users can reprogram the rate after the first call of this function. Upon subsequent calls, reads data from AI-207 enabled channels and converts data using channel-specific configuration parameters, such as channel gain.
`DqAdv207ReadChannel`	Returns raw measurements of internal channel references
`DqAdv207SetAutozero`	Enables/disables autozeroing capability. By default, autozero is OFF

3.3 Low-level Programming Techniques

Application developers are encouraged to explore the existing source code examples when first programming the AI-207. Sample code provided with the installation is self-documented and serves as a good starting point.

Code examples are located in the following directories:

- On Linux systems: <PowerDNA-x.y.z>/src/DAQLib_Samples
- On Windows: *Start » All Programs » UEI » PowerDNA » Examples*

Sample code for data acquisition modes have the name of the mode and the name of the I/O boards being programmed embedded in the sample name. For example, SampleVMap207 contains sample code for running the an AI-207 using VMap data acquisition mode. Note that immediate mode samples are named Sample<I/O board name>, (i.e., Sample207).

3.3.1 Data Collection Modes

The AI-207 supports the following acquisition modes:

- Immediate (point-to-point): Designed to provide easy access to a single I/O board at a non-deterministic pace. Acquires a single data point per channel. Runs at a maximum of 100 Hz.
- rtDMAP/VMAP: Designed for closed-loop (control) applications. Users set up a “map” of I/O boards and channels from which to acquire data. Input data is stored in I/O board FIFOs at a rate paced by its hardware clock; a single API call (refresh) paced by the user application causes data to be collected directly from I/O board FIFOs and packed for delivery to the user application.
  - rtDMAP collects only 1 data sample per channel
  - VMAP (rtVMAP) can collect multiple samples per channel
- aDMAP/aVMAP: designed with the same concept as rtDMAP/VMAP except that timebase for transfers is controlled by the CPU board in the UEI chassis (IOM) instead of paced by the user application.
  For example, data transfers to the user application can be initiated by a timeout of a user-programmed counter on the IOM.
- ACB: Designed for data acquisition applications that require all channel data to be delivered without gaps but can accept slight delays in delivery. Data is acquired at the full speed of the I/O board and then stored in an advanced circular buffer (ACB) on the IOM CPU. Data from the ACB is transferred to the user application based on user-programmed events (e.g., data storage crosses a frame boundary of the circular buffer).
  Not available on UEIPAC products.

API that implement data acquisition modes and additional mode descriptions are provided in the *PowerDNA API Reference Manual*.

3.4 Programming the AI-207 (Immediate Mode)

The following sections provide an overview of how to set up and use your AI-207 in Immediate Mode using the low-level API.

For best results, use this overview in conjunction with actual sample code, (i.e., Sample207, Sample207_Thermocouple). This overview does not address typical initialization or error handling. Refer to Section 3.3 for sample code location.

3.4.1 Setting up Channel Configuration & Timestamps

The AI-207 channels are configured by setting up an array of channels you wish to acquire data for and their configuration settings: channel number, gain setting, differential vs single-ended.

The following code uses `uint cl[CHAN_LIST_SZ]` to hold the channel list and sets channels 0 through `CHAN_LIST_SZ-1` to be configured with a gain of 1 and set to differential mode.

```
for (i = 0; i < CHAN_LIST_SZ; i++) {
        cl[i] = i | DQ_LNCL_GAIN(DQ_AI208_GAIN_1) | DQ_LNCL_DIFF;
}
```

`DQ_LNCL_GAIN()` is a macro that uses `*208*GAIN` defines -- these are AI-208 #defines that also map to the supported AI-207 gain levels: `DQ_AI208_GAIN_1` (±10v), `DQ_AI208_GAIN_2` (±5v), through `DQ_AI208_GAIN_800` (±0.0125v).

NOTE: For each gain setting, firmware uses a precalculated initial delay to allow the channel to settle before sampling. When the scan rate and channel list are programmed, the firmware allocates the minimum settle time for each channel depending on the gain selected; this settle time can limit the maximum sample rate the channel can run at.
Refer to **Table 1-2** on page 6 for the minimum settling time for each gain setting.

If timestamping is enabled, the timestamp value will be stored in the FIFO with each scan. Timestamps can be configured as the first item in the list or the last item in the list:

```
if (TIMESTAMP) {
        cl[CHAN_LIST_SZ-1] = DQ_LNCL_TIMESTAMP;
}
```

3.4.2 Configuring the Channel List

After the `cl` array is set up, the array is passed to firmware for hardware/system configuration. Note that the channel list (`cl`) setup is the same for all of the data acquisition modes; however, the API that configures the firmware/hardware based on the channel list is different for the different modes. Detailed descriptions of API and data acquisition modes are provided in the *PowerDNA API Reference Manual*.

In immediate mode, AI-207 configuration requires an initial call to the `DqAdv207Read` API. This read passes configuration information between the user application and firmware; data acquired from this first read should be disregarded:

```
DqAdv207Read(hd0, DEVN, CHAN_LIST_SZ, cl, NULL, NULL);
```

where

- `hd0` is the handle to the IOM
- `DEVN` is the AI-207 position in the chassis
- `CHAN_LIST_SZ` is the number of channels (including timestamp or CJC) that you are programming
- `cl` is the array of channel configuration data
- Note that the final two parameters would be pointers to arrays to hold acquired data, but in this first read, that data is disregarded

The `DqAdv207Read` function returns status flags, one of which indicates the configuration status. Your program should not advance until `DqAdv207Read` is passing back settled data. The `DQ_DATA_NOTREADY` flag signifies new data has not yet settled on the IOM side, and you should continue waiting until the data is ready.

```
int ret;
do {
        ret = DqAdv207Read(hd, DEVN, CHAN_LIST_SZ, cl, NULL, NULL);
        UeiPalSleep(50);
} while ((ret == DQ_DATA_NOTREADY) && (ret >= 0)); //wait until data is valid
```

NOTE: As an alternative, you can use the UEI-provided error-checking macro, `Chk4Err,` which assigns the returned status to the `ret` variable. Refer to UEI sample code for usage.

3.4.3 Adding CJC Channel to the Channel Configuration

If your application uses CJC compensation for thermocouple measurements, add the CJC channel to the channel list to acquire CJC readings. This must get set up before the initial call to `DqAdv207Read`:

```
// Add CJC sensor channel, the CJC sensor returns the temperature in Kelvin
// multiplied by 0.00295. Ambient temperature gives 0.8V so we can use gain
// to improve measurements.
cl[CHANNELS-1] = DQL_FE207_CJC | DQ_LNCL_GAIN(DQ_AI208_GAIN_4);
```

Note that `DQL_FE207_CJC` is defined as 33, the AI-207 CJC channel number.

3.4.4 Setting up the AutoZero Feature

The AI-207 can be programmed to continuously sample an internal reference and use that reference to autozero any offset fluctuations over temperature and over time.

To enable or disable autozero, use the `DqAdv207SetAutozero` API:

```
DqAdv207SetAutozero(hd0, DEVN, &auto_zero);
```

where `&auto_zero` is a pointer to a Boolean TRUE or FALSE.

Note that firmware schedules the sampling of references through the multiplexer when autozero is enabled; this could affect the maximum sample rate for channels, as the maximum AI-207 aggregate sample rate for the board is 16 kHz.
Refer to Section 1.5.3 for more information.

Also note that the `DqAdv207SetAutozero()` function should be called before the first time `DqAdv207ReadDqAdv207Read()` is called.

3.4.5 Setting the Scan Rate

The maximum sample rate in immediate mode is 100 Hz. (The default sample rate is 10 Hz).

To change the scan rate in immediate mode (i.e., to 100 Hz in this example), you can use the following API:

```
// Program the scan clock (aka CL clock) to acquire 100.0 scan/s.
// The function DqCmdSetClock() can program clocks on several I/O boards
//    at once.
// Use the following to program one I/O board
    clkEntries = 1;
    clkSet.dev = DEVN | DQ_LASTDEV; // last (and only) device in the list
    clkSet.ss = DQ_SS0IN; // program input subsystem clock
    clkSet.clocksel = DQ_LN_CLKID_CLIN; // Program the scan clock, convert
                                        // clock will be set automatically
    clkSet.frq = 100.0;
    DqCmdSetClock(hd0, &clkSet, &actualClkRate, &clkEntries);
```

where `clkSet` is a structure of type `DQSETCLK` and `actualClkRate` is returned to the user application as the actual rate programmed.

To acquire > 100 Hz, you will need to use a different mode (Section 3.3.1).

An overview of setting the scan rate in different data acquisition modes is provided in Section 3.5 and detailed descriptions of API and modes are provided in the *PowerDNA API Reference Manual*.

3.4.6 Reading Data & Timestamps

The `DqAdv207Read` API is used to read acquired data and the timestamp, if the timestamp is enabled. Returned data is in the order setup in the channel list.

```
DqAdv207Read(hd0, DEVN, CHAN_LIST_SZ, cl, data, fdata);
```

where

- `hd0` is the handle to the IOM
- `DEVN` is the AI-207 position in the chassis

- `CHAN_LIST_SZ` is the number of channels (including timestamp or CJC) that you are programming
- `cl` is the array of channel configuration data
- `data` is the raw binary data received from the device (uint32)
- `fdata` is the converted floating point voltage value

If reading thermocouple data, you can use the following API to convert the CJC temperature reading to the correct temperature scale, and then convert the acquired channel data to the thermocouple temperature.

Note that thermocouple constants can be found in `TCConversion.h`, which is provided in the same directory with the `Sample207_Thermocouple` sample code.

```
// convert CJC channel voltage to temperature (Kelvin)
cjcTemp = fdata[CHANNELS-1]/0.00295;

// convert Kelvin to the temperature scale needed to convert
//   thermocouple voltage.
UeiDaqConvertTempScale(UeiTemperatureScaleKelvin, cjcTemp,
                       UeiTemperatureScaleCelsius,
                       &cjcTemp);

//convert voltages to temperature:
UeiDaqConvertVoltToThermocoupleTemp(UeiThermocoupleTypeK,
                                    UeiTemperatureScaleCelsius,
                                    cjcTemp,
                                    fdata[i],
                                    &scaledTemp);
```

3.5 Programming Scan Rate

The sample rate is set with different API, depending on the data acquisition mode chosen:

- Immediate Mode: `DqCmdSetClock` (see Section 3.4.5)
- *VMAP: `DqRtVmapSetScanRate`
- ACB: `DqAcbInitOps` (see Section 3.6)

UEI Clock Nomenclature Note: Throughout UEI sample code and documentation, you will see references to CL and CV clocks.

In general, CL clocks (also called channel list clocks or scan clocks) are used in I/O boards with multiplexed inputs. Clocks for muxed inputs are serially sequenced with the order/control based on the channel list, to control the sampling of each channel through the mux. The CV clocks, also called conversion clocks, are typically referred to with simultaneously sampled I/O boards.

For the AI-207, you will use the CL clock, (for example when using ACB mode):

- CLCLKSRC0 programs a clock generated locally on the I/O board, divided down from the 66 MHz system clock on the board.
- CLCLKSRC1 specifies a clock generated external to the board and routed in over the chassis SYNC bus (for example, a 1PPS-synchronized clock generated on the chassis CPU board).

Refer to Section 3.6 for information about setting clock sources using ACB mode. Refer to sample code and the *PowerDNA API Reference Manual* to learn more about setting clocks (and programming the AI-207) in ACB mode and the different MAP modes. Refer to the *DNx 1PPS Sync Interface Manual* to learn more about programming and routing chassis-wide synchronized clocks to all I/O boards.

3.6 Configuring Channels & Scan Rate in ACB Mode

In ACB mode, the channel list (`cl`) array is configured the same way as described in Section 3.4.1 with some additional configuration. The `cl` is passed to chassis firmware via the `DqAcbInitOps()` function, along with other parameters including an additional `Config` parameter and the sample rate.

The `Config` parameter is built with the following flags:

```
// 207 unit configuration word
#define CFG207          (DQ_LN_ENABLED \
                        |DQ_LN_ACTIVE \
                        |DQ_LN_IRQEN \
                        |DQ_LN_CLCKSRC0 \
                        |DQ_LN_STREAMING \
                        |DQ_AI208_MODEFIFO \
                        |DQ_LN_RAW32)
uint32 Config = CFG207;
```

- `DQ_LN_ENABLE` enables all board operations.
- `DQ_LN_CLCKSRC0` selects the internal channel list clock (`cl`) source as a time base. To select a clock generated external to the board and routed in over the chassis SYNC bus (for example, a 1PPS-synchronized clock generated on the CPU), select `DQ_LN_CLCKSRC1`. (Refer to the previous section for more information.)

The channel list and other configuration parameters get passed to `DqAcbInitOps`:

```
DqAcbInitOps(bcb,
             &Config,
             0,                 //TrigSize
             NULL,              //pDQSETTRIG TrigMode
             &fCLClk,
             0,                 //float*  fCVClk, not using on this board
             &CHAN_LIST_SZ,
             cl,
             0,
             &acb);
```

where

- `bcb` points to the buffer control block: each new advanced circular buffer will have a BCB structure allocated to it
- `fCLKClk` is a float set to the sample rate
- `CHAN_LIST_SZ` is the number of channels (including timestamp) that you are programming
- `cl` is the array of channel configuration
- `acb` is a structure of type DQACBCFG that configures ACB parameters. See **Table 3-2**.

Refer to the *PowerDNA API Reference Manual* for a full list of flags and descriptions of ACB API and parameters.

DQACBCFG **Structure Element**	**Description**
uint32 samplesz	raw sample , bytes
uint32 scansz	scan , samples
uint32 framesize	number of scans in the frame, max
uint32 frames	frames in the buffer
uint32 ppevent	packets per DQ_ePacketDone event
uint32 mode	mode of operations: Single,Cycle,Recycled,error handling
uint32 dirflags	transfer direction and additional flags
uint32 maxpkt	how much data to accumulate in the packet before sending (0=default)
uint32 hwbuf	how much data to keep on the cube (0 = default)
uint32 hostringsz	number of packets in the host ring buffer (0 = default)
uint32 wtrmark	percent of the ring buffer queue packets kept in case IOM reports an error

Table 3-2 DQACBCFG Structure

Appendix A

A.1 Accessories

The following cables and STP boards are available for the AI-207 board.

DNA-CBL-37

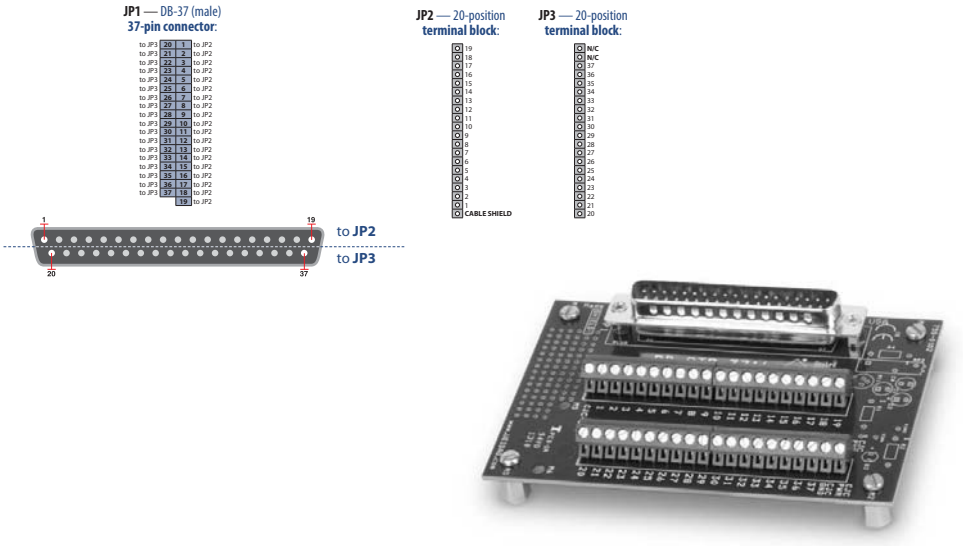
This is a 37-conductor flat ribbon cable with 37-pin male D-sub connectors on both ends. The length is 3ft and the weight is 3.4 ounces or 98 grams.

DNA-CBL-37S

This is a 37-conductor round shielded cable with 37-pin male D-sub connectors on both ends. It is made with round, heavy-shielded cable; 3 ft (90 cm) long, weight of 10 ounces or 282 grams; also available in 10ft and 20ft lengths.

DNA-STP-37

The DNA-STP-37 provides easy screw terminal connections for all DNA and DNR series I/O boards which utilize the 37-pin connector scheme. The DNA-STP-37 is connected to the I/O board via either DNA-CBL-37 or DNA-CBL-37S series cables. The dimensions of the STP-37 board are 4.2w x 2.8d x1.0h inch or 10.6 x 7.1 x 7.6 cm (with standoffs). The weight of the STP-37 board is 2.4 ounces or 69 grams.



DNA-STP-AI-U

Universal screw-terminal panel with embedded cold-junction compensation (CJC).

DNA-STP-AI-207TC

Screw terminal panel for use with the DNx-AI-207 and thermocouples. The panel provides open thermocouple detection as well as the CJC measurement.

Index