United Electronic Industries

The High-Performance Alternative

DNx-VR-608

—

User Manual

Eight channel Variable Reluctance Sensor Interface
for the PowerDNA Cube, PowerDNR RACKtangle, and PowerDNF FLATRACK

October 2016

PN Man-DNx-VR-608

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.

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 of the DNR-, DNF- and DNA-VR-608 board and its use for variable reluctance sensor applications.

1.1 Organization of Manual

This DNx-VR-608 User Manual is organized as follows:

- **Introduction**
  This section provides an overview of the DNx-VR-608 board features, device architecture, and wiring.
- **Programming with the High-Level API**
  This chapter provides an overview of the how to create a session, configure the session, and format relevant data with the Framework API.
- **Programming with the Low-Level API**
  Describes low-level API commands for configuring and using the DNx-VR-608 series layer for operating modes.
- **Appendix A - Accessories**
  This appendix provides a list of accessories available for use with the DNx-VR-608 board.
- **Index**
  This is an alphabetical listing of the 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 to 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**.”

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 VR-608 Interface Board

The DNA-VR-608, DNR-VR-608, and DNF-VR-608 are 8-channel, variable reluctance sensor interface boards. The DNA/DNR/DNF-VR-608 is compatible with UEI's "Cube", RACKtangle and FlatRACK chassis respectively; the boards are electronically identical. The boards are designed for use in a wide variety of motion and rotation monitoring applications. Each channel provides a fully differential input and is capable of monitoring VR output levels as high as 250 Volts peak-to-peak. The board is also an ideal solution as a general counter unrelated to variable reluctance sensors and can be used to measure an extremely wide variety of AC signals.

The DNx-VR-608 offers a maximum input pulse rate of 300 kHz. The maximum input pulse rate is somewhat dependent on the signal level; the technical specifications on the following page show various pulse rates obtainable at various input signal levels. The minimum detectable input signal may be fixed at 50 millivolts peak-to-peak; an internal "Adaptive Peak Threshold" mode may set the threshold to 30% of a time-averaged input. A watchdog circuit resets the input to minimum threshold level if the input "drops out" for 85 milliseconds. Open sensor/disconnected cable detection is also provided on the VR-608.

The DNx-VR-608 supports the following operating modes:

- Timed Count/Frequency: counts the number of teeth detected during a specified time interval and returns the velocity in TPS, RPS, RPM
- N pulses: measure the time taken to detect N teeth and return velocity
- Z/Index pulse: measure the number of teeth and the time elapsed between two Z/Index pulses (The Z/Index tooth is usually a gap or a double tooth on the encoder wheel)

The input impedance of each channel is greater than 40 kOhms. This high impedance ensures the board is compatible with an extremely wide range of variable reluctance sensors.

Each board provides a fully differential input and four isolated TTL-level digital outputs that change state on the threshold crossing of the input waveform. This allows the board to be used as a signal conditioning front end for existing TTL based counter test systems.

The DNx-VR-608 is fully isolated from the UEI chassis as well as other I/O boards installed in the chassis. In addition, the inputs are divided into four sets of channel pairs, and each two-channel pair is isolated from the others. Each isolated channel pair is also supported with a logic level digital output.

All connections are made through 37-pin D connectors that ensure that mating cables or connectors are readily available. Users may also use the DNA-STP-37 screw terminal panel via DNA-CBL-37 series cables. The DNx-VR-608 includes software drivers supporting all popular operating systems.

Our UEIDAQ Framework provides Windows users a simple and complete software interface to all popular Windows programming languages and data acquisitions and control applications (e.g., LabVIEW, MATLAB). The board also includes a factory written (and supported), simple yet powerful API for all popular non-Windows operating systems including Linux, VxWorks, QNX, RTX and more.

1.3 Features

The common features of the DNx-VR-608 are listed below:

- DNA-VR-608 for “CUBE” chassis
- DNR-VR-608 for RACKtangle™ I/O chassis
- Wide input ranges (50mV to 250V peak-to-peak)
- Bipolar (VR) and Unipolar (Hall/TTL) input modes
- 300 kHz maximum input pulse rate
- Adaptive Threshold input mode
- True zero-cross detection inputs
- High/Low tooth index detection
- Open-circuit detection
- Conditioned signal inputs can be routed to TTL outputs
- Tested to withstand 5g Vibration, 50g Shock, -40 to +85°C Temperature, and Altitude up to 70,000 ft or 21’000 meters
- Weight of 134 g or 4.9 oz for DNA-VR-608; 146g or 5.14 oz for DNR
- UEI Framework Software API may be used with all popular Windows programming languages and most real time operating systems such as RT Linux, RTX, or QNX and graphical applications such as LabVIEW, MATLAB and any application supporting ActiveX or OPC.

1.4 Specification

The technical specification for DNx-VR-608 is provided in the table below:

Table 1-1. DNx-VR-608 Technical Specifications

Number of channels	8
Channel configuration	Differential
Channel isolation	4 isolated banks of two channels
Input Impedance	> 40 kOhm, < 250 pF
Input voltage range	Up to 250 Vp-p
Maximum pulse rate (bipolar, VR input mode)	300 kHz at $\geq$ 3.2 Vp-p 200 kHz at $\geq$ 2.1 Vp-p 100 kHz at $\geq$ 1.1 Vp-p 50 kHz at $\geq$ 0.5 Vp-p 25 kHz at $\geq$ 0.25 Vp-p 10 kHz at $\geq$ 0.125 Vp-p $\leq$ 5 kHz at $\geq$ 0.08 Vp-p
Minimum detectable input (VR input mode)	50 mVp-p, fixed input mode
Max pulse rate, unipolar mode	18 kHz (V_{low} <1V, V_{high} > 3.5 V
Measurement accuracy	0.0833 Hz or 0.1% whichever is greater
Inter-tooth timing	
Time base resolution	15.2 nS
Counter depth	32-bits
Timing measurement range	20 µS to 65 seconds
Overvoltage protection	300 Vp-p
GENERAL SPECIFICATIONS	
Power dissipation	< 3W
Isolation	350 Vrms
Operating Temp. Range	Tested -40 to +85 °C
Operating Humidity	95%, non-condensing
Vibration *IEC 60068-2-6* *IEC 60068-2-64*	5 g, 10-500 Hz, sinusoidal 5 g (rms), 10-500 Hz, broad-band 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	260,000 hours

1.5 Device Architecture

This section describes the hardware used in the DNx-VR-608 board. A block diagram of the board is shown in **Figure 1-1** and of one channel in **Figure 1-2**.

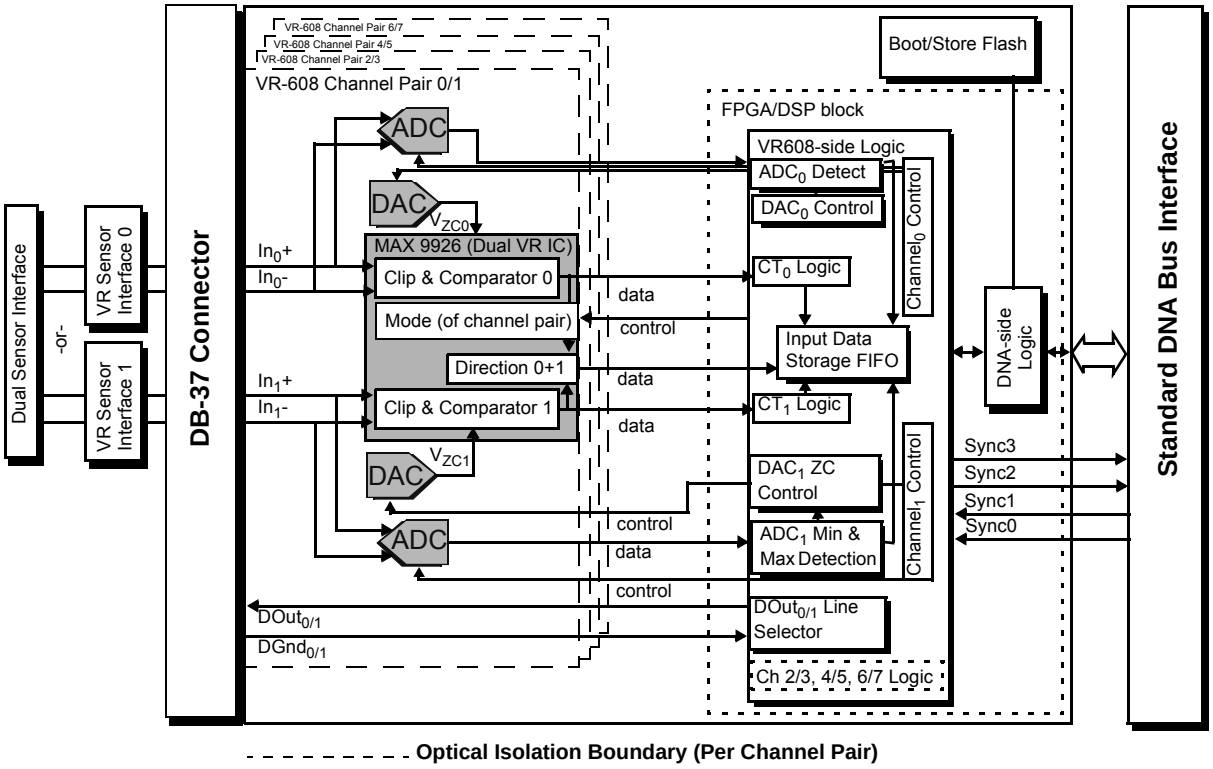


Figure 1-1. Logic Block Diagram of the VR-608 board

The front-end of the DNx-VR-608 exposes four dual-channel pair (0&1, 2&3, 4&5, 6&7) inputs and four digital output lines through the DB-37 connector. Each channel pair can connect to either two individual VR sensors (e.g. one on the In_0+/- input lines of Channel 0 and one on the In_1+/- Channel 1 input lines) or one out of phase dual VRS (e.g. on all four In_0+/- & In_1+/- lines of Chan 0&1).

The VR-608 is designed to accept the following input signal sources:

- Variable reluctance sensor (VRS) that translates teeth of a gear wheel (either with or without a zero / index / z-tooth) as an analog sine wave. Both wires (signal and return) from the VRS connect to one single VR-608 channel (e.g. In_0+/- of Channel 0).
- Hall-effect sensor that translates teeth to a 0-5V square wave. The sensor's two signal and return wires connect to a single VR-608 channel (e.g. In_0+/- of Channel 0); the sensor's power wire shall not be connected to the VR-608.

All In+/- lines of a channel pass through a limiter circuit and then enter into both a Dual VR IC and the channel's individual ADC. The ADC serves two purposes: to capture the raw voltage level that's useful for debug & open-circuit detection; and to detect the minimum & maximum voltage peaks & calculate the zero-level crossing for special cases beyond the normal ones built-in to the Dual VR IC. The Dual VR IC's purpose is to convert the VR sensor's analog signal into the digital signal / comparator circuit output. The Dual VR IC electronic chip is the Maxim MAX9926 Variable-Reluctance Sensor Interface.

After the input signal has passed the limiter circuit it enters the Dual VR IC and is clipped from ±125V down to ±5V using zener diodes as shown in **Figure 1-2**. The ±5V input is amplified by a configurable quantity. The input enters a zero-crossing detection circuit that's used to find when it should raise its comparator output. The input also enters the peak detector circuit that is used to find when it should drop its comparator output as described later.

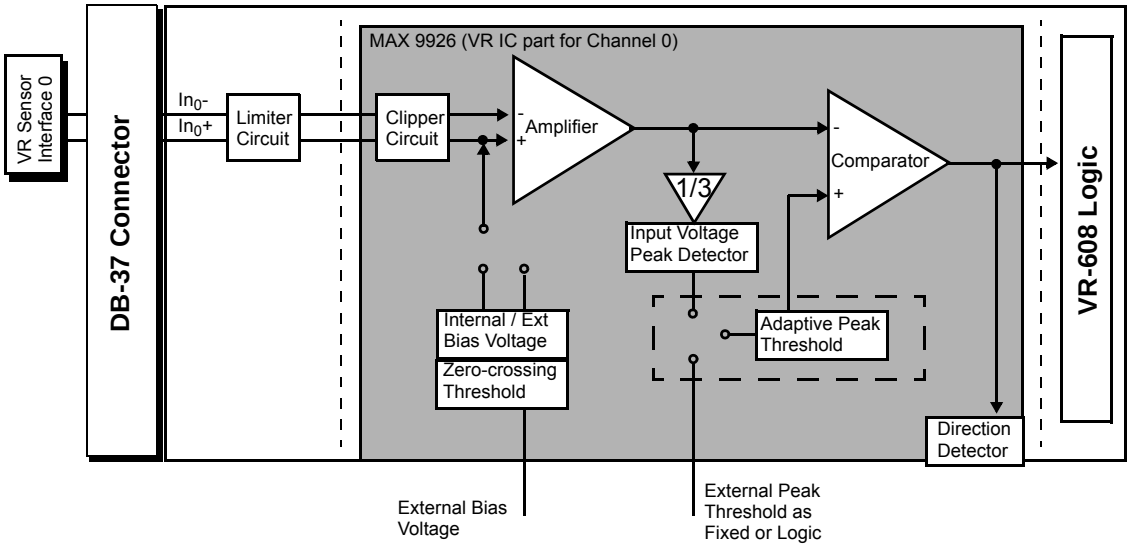


Figure 1-2. Logic Block Diagram of a VR-608 Channel

The zero-crossing (ZC) threshold can be configured in these modes:

- automatically to mid-range of the analog sine wave signal using the built-in zero-cross detector of the Dual VR IC (`VR608_ZC_ONCHIP` mode)
- fixed to a particular input level by the user (`VR608_ZC_FIXED` mode) use this mode for Hall-effect sensors

The adaptive peak threshold (APT) can be configured to:

- automatically to the peak of the analog sine wave signal using the built-in APT detector of the Dual VR IC (`VR608_APT_ONCHIP` mode)
- fixed to a particular input level by the user (`VR608_APT_FIXED` mode)

The on-chip adaptive peak threshold (APT) functionality in `VR608_APT_ONCHIP` mode is shown from the MAX9926 specification sheet in **Figure 1-3** below.

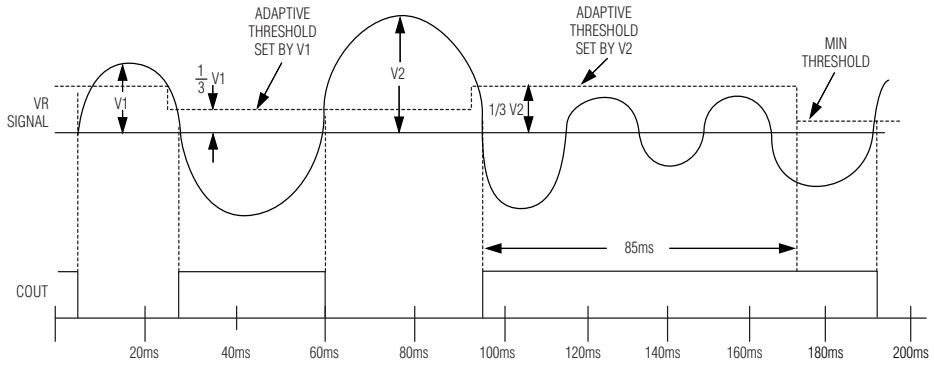


Figure 1-3. MAX9926 Adaptive Peak Threshold (APT)

Notice that if the input voltage remains lower than the adaptive threshold for 85ms, the threshold will be dropped to the minimum value, driving the comparator output low; this ensures pulse recognition in the presence of an intermittent sensor connection in as `VR608_APT_ONCHIP` mode.

For Hall-effect mode the analog peak threshold should be reconfigured to a fixed voltage as `VR608_APT_FIXED` mode. This allows you to set your own peak voltage; but fixed modeis limited to VR sensor input pulses under 18kHz.

After the ZC and APT stages the signal enters into the MAX9926's comparator. In `VR608_MODE_COUNTER` mode this indicates the presence of a tooth on the gear-wheel of each of the two independent channels of the Dual VR IC. This output passes from the Dual VR IC through optical isolation to the VR-608 Logic chip seen in **Figure 1-1**. The Dual VR IC's comparator output for each channel enters the channel's CT module in the VR-608 Logic:

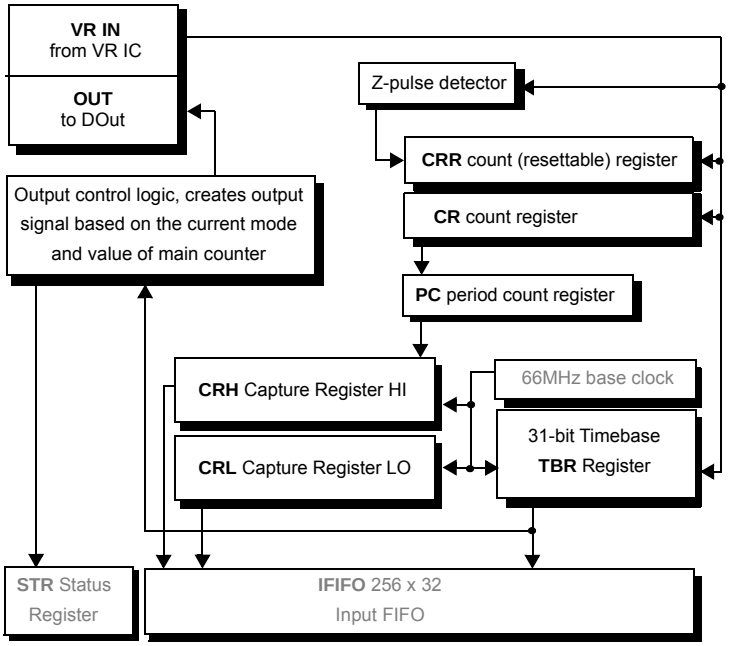


Figure 1-4. VR-608 Counter/Timer Module

The Counter/Timer registers hold the following information for the VR-608:

- CRL (number of teeth/pulses) / CRH (per time): velocity
- CR: total number of teeth/pulses
- CRR: position (count reset by the z-tooth pulse detector)

Results from the counting process (input data) can be stored in an input FIFO. The VR-608 logic can provide either the FIFO, or immediate register values of CRR, CR, CRH, CRL, TBR, and STR upon request from the firmware.

In practice, the software provides the following information from the registers:

Mode		CRL / CRH	CRR	CR
Timed (TBR)	Counter	Velocity	Position	Teeth Count
Z-pulse	Counter	Velocity	Position	Teeth Count
N-pulse	Counter	Velocity	-	-

More information about setting these modes in your software application can be found in the API Manual and the following two chapters, including function calls used to obtain the ADC FIFO contents.

The VR-608 Logic chip is responsible for gathering the data from each channel's Dual VR IC, ADCs, accumulating the digital data readings inside internal FIFOs, calculating results (velocity, position, etc) within C/T module, and buffering the data for the firmware to return to your software application.

The VR-608 Logic chip also controls four digital output lines user-selectable to reroute in hardware the following internal signals to those digital output lines:

- Tooth detector: This is the Dual VR IC comparator result for a channel. The output is a logical high when a tooth is detected.
- Z-tooth detector: This is the Counter/Timer logic module zero / index / z-tooth detector. It is a logical high when the z-tooth is detected.

The digital output lines can be set to start this way at power-up initialization. This is done through two firmware features that are completely user-configured:

- Initialization Values load from NVRAM a preset configuration
- Auto-start into operating mode that enables everything at power-up

By pre-configuring the digital output lines to a specific configuration and also selecting auto-start, as soon as the power is turned on, the VR-608 can begin detecting normal/index teeth as TTL logic levels over the digital output lines without user intervention or additional configuration equipment. Note however that the VR-608 may output transient glitches at powerup, shutdown, and/or channel enable. These transients may be interpreted as signals if you use the digital output lines as an input to other systems, such as legacy systems.

1.6 Jumpers

The DNA-VR-608 jumper location to be used with the PowerDNA Field Installation Guide is shown in **Figure 1-5** below:

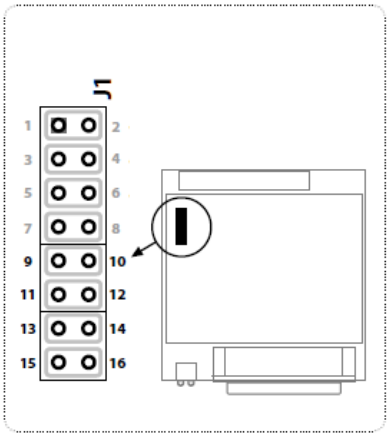


Figure 1-5. VR-608 Jumper Address Location

1.7 Indicators

A photo of the DNx-VR-608 unit is illustrated below.

The front panel has two LED indicators:

- RDY: indicates that the layer is receiving power and operational.
- STS: can be set by the user using the low-level framework.

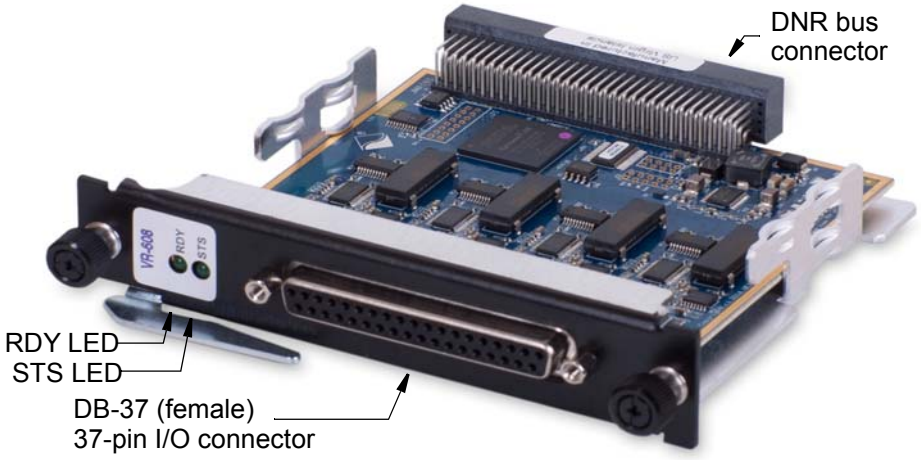


Figure 1-6. The DNR-VR-608 Layer

1.8 Wiring and Layer Connectors

Figure 1-7 below illustrates the pinout of the VR-608. The VR-608 uses a B-size 37-pin D-sub connector. The following signals are located at the connector:

- In0- to In7- and In0+ to In7+: differential analog input channel for the signal & return lines of variable resistance sensor or hall-effect sensor.
- Digital output lines (Dout x/y) and their return/ground lines (Gnd x/y)
- NC - not connected; do not connect to these pins

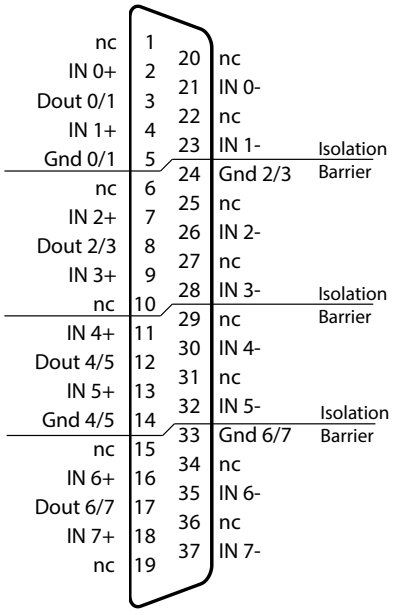


Figure 1-7. Pinout Diagram of the VR-608

Chapter 2 Programming with the High Level API

This section describes how to control the DNx-VR-608 using the UeiDaq Framework High Level API.

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.

The following section focuses on the C++ API, but the concept is the same no matter what programming language you use.

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

2.1 Creating a Session

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

```
// create a session object
CUeiSession session;
```

2.2 Configuring the Resource String

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:

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

For PowerDNA and RACKtangle, the device class is **pdna**.

For example, the following resource string selects variable reluctance lines 0,1 on device 1 at IP address 192.168.100.2: “pdna://192.168.100.2/Dev1/Vr0,1”

NOTE: In the Framework, VR channels should be configured as pairs (e.g. 0,1) since some configuration options will apply to both channels anyway.

2.3 Configuring a Session

This session type measures velocity, position and counts the teeth of a gear-toothed wheel rotating next to a variable reluctance sensor.

Each VR channel can convert the output signal of the VR sensor to a pulse train that can be counted and whose frequency can be measured.

Use the method CreateVRChannel() to configure one or more channel(s) in VR mode.

The following call configures channel 0 of a VR-608 set as device 1:

```
// Configure session to write to channel 0 on device 1
session.CreateVRChannel("pdna://192.168.100.2/Dev1/vr0", vrMode);
```

It configures the following parameters:

- **VR mode**: Selects the mode among the following:
  - `UeiVRModeCounterTimed`: Count number of teeth detected during a timed interval.
  - `UeiVRModeCounterNPulses`: Measure the time taken to detect N teeth. Number of teeth needs to be set separately.
  - `UeiVRModeZPulse`: Measure the number of teeth and the time elapsed between two Z pulses. The Z tooth is usually a gap or a double tooth on the encoder wheel.

You can configure additional parameters using the channel object methods (or a property node under LabVIEW):

- **Zero Crossing mode**: Configures the method used to detect zero crossing. Zero crossing finds the point in time where the VR sensor output voltage goes from positive to negative voltage. This point is the center of the tooth lining up with the center of the VR sensor:
  - `UeiZCModeChip`: The front-end IC will automatically calculate the ZC
  - `UeiZCModeFixed`: Provide a fixed DAC bias voltage level for the ZC

```
// Set ZC mode to ‘Chip’ mode
pVrChan-> SetZCMode(UeiZCModeChip);
```

  - **Zero Crossing level**: Configures the threshold to detect zero crossing - only when ZC mode is set to `UeiZCModeFixed`.

```
// Set ZC level to 2.0V
pVrChan-> SetZCLevel (2.0);
```

- **Adaptive Peak Threshold mode**: Configures the adaptive peak threshold (APT) mode on variable reluctance sessions. APT finds the point in time where the VR sensor output voltage falls below a certain threshold. This point marks the beginning of the gap between two teeth on the gear-wheel.
  - `UeiAPTModeChip`: The front-end IC will automatically set the peak threshold to 1/3 of the peak input voltage as described in Chapter 1
  - `UeiAPTModeFixed`: Use your own analog peak voltage as threshold

```
// Set APT mode to ‘Chip’ mode
pVrChan-> SetAPTMode(UeiAPTModeChip);
```

  - **APT threshold**: Configures the APT fixed threshold. This parameter is only used when APT mode is set to `UeiAPTModeFixed`.

```
// Set APT threshold to 4 volts
pVrChan->SetAPTThreshold(4);
```

- **Number of teeth**: Configures the the number of teeth on the encoder wheel. This parameter is required to measure the position when the mode is set to Timed or NPulses. It is ignored in ZPulse mode.

```
// Set number of teeth to 60
pVrChan->SetNumberOfTeeth(60);
```

- **Size of Z-tooth**: For gear-wheels with an index/z-tooth you can set the number of missing teeth (from -1 to -3) or long teeth (1 to 3) with this function to add them to the total number of teeth:

```
// Set width of the z-tooth to 0 teeth (gear wheel has no index tooth)
pVrChan->SetZToothSize(0);
```

- **Timed Mode Rate**: When a channel is in `UeiVRModeCounterTimed` mode the channel's counter-timer unit saves a new read point into the input buffer at a rate / time-base that you configure with SetTimedModeRate(). This is not the sampling rate (see SetADCRate call) which is the same for all channels, but rather the rate at which data is stored:

```
// Set the read frequency to 10 Hz in UeiVRModeCounterTimed mode
pVrChan->SetTimedModeRate(10);
```

2.4 Configuring the Timing

You can configure the VR-608 to run in simple mode (point by point) only. Use of ACB mode is not currently supported. In simple mode, the delay between samples is determined by software on the host computer.

The following sample shows how to configure the simple mode. Please refer to the "UeiDaq Framework User's Manual" to learn about timing modes.

```
// configure timing for point-by-point (simple mode)
session.ConfigureTimingForSimpleIO();
```

The ADC sampling rate for all channels is controlled with Get/SetADCRate():

```
// Set the configuration rate of all channels to 200kHz
session.SetADCRate(200000)
```

2.5 Reading Data

Channels on the VR-608 are read independently. You need to create a reader object for each configured channel. Each reader returns a structure (a cluster in LabVIEW) containing the measured velocity, position and total teeth count since the session started.

The following sample code shows how to create a reader object and read a measurement:

```
// Create a reader for channel 0 and link it to the session's stream

CUeiVRReader reader(vrSession.GetDataStream(), 0);

// Read one value

tUeiVRData vrData;
reader.Read(1, &vrData, &numValueRead);

std::cout << vrData.velocity << std::endl
std::cout << vrData.position << std::endl
std::cout << vrData.teethCount << std::endl
```

2.6 Cleaning-up the Session

The session object will clean itself up when it goes out of scope or when it is destroyed. To reuse the object with a different set of channels or parameters, you can manually clean up the session as follows:

```
// clean up the session

session.CleanUp();
```

Chapter 3 Programming with the Low-level API

This chapter illustrates how to program the PowerDNA cube using the low-level API. The low-level API offers direct access to PowerDNA DAQBios protocol and also allows you to access device registers directly.

However, we recommend that, when possible, you use the UeiDaq Framework High-Level API, because 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.

For additional information about low-level programming of the VR-608, please refer to the PowerDNA API Reference Manual document under:

Start » Programs » UEI » PowerDNA » Documentation

Refer to the PowerDNA API Reference Manual on how to use the following low-level functions of the VR-608, as well as others related to cube operation:

Function	Description
`DqAdv608Enable`	Start or stop a VR-608 device (call after DqAdv608SetCfg)
`DqAdv608GetCfg`	Get the presently configured parameters.
`DqAdv608SetWatermark`	Set the watermark to better control dataflow from the IOM
`DqAdv608Read`	Reads position, velocity, tooth count, timestamp inputs
`DqAdv608ReadADCFifo`	Peek into ADC channel's FIFO data of a channel pair
`DqAdv608ReadADCStatus`	Read ADC status, present and accumulated min/max values
`DqAdv608ReadFifo`	Read the channel FIFO
`DqAdv608SetCfg`	Set configuration parameters.

3.1 Function Call Tutorial

The following tutorial provides a brief overview of how to set up and use your VR-608 using the Low-Level API. It is suggested that it be used in conjunction with an actual code sample found in ../src/DAQLib_Samples/Sample608 (Linux) or %PDNAROOT%\Examples\Visual C++\Sample608 (Windows).

This tutorial explains the following topics:

- Start-up
  - Initializing the cube and enabling VR-608 board(s).
- Configuration
  - Configuring the VR-608 parameters and enabling operation.
- Operation
  - Retrieving input data.
- Stopping
  - Disabling layers cleanly.

3.1.1 Initialization

You must first get a "DAQLib handle for an IOM" by calling DqOpenIOM():

```
// Connect to the IOM and obtain a library handle for the connection
DqOpenIOM("192.168.100.2", DQ_UDP_DAQ_PORT, 1000, &hd, &DQRdCfg);
```

3.1.2 Configuration

Begin by creating a VR608_ExtCfg structure for the channel(s). The parameters for this structure are described in the API manual.

Configure the VR-608 card:

```
// Set the configuration for each channel
DqAdv608SetCfg(hd, DEVN, chnl, front_cfg, mode, &extcfg);
```

Remember that the front_cfg and mode will be set for each channel pair.

To verify that parameters were set as desired, read back the configuration as:

```
// Get the configuration for each channel
DqAdv608GetCfg(hd, DEVN, chnl, &front_cfg, &mode, &extcfg);
```

Once that all channels are configured, enable operation:

```
// Enable the channel
DqAdv608Enable(hd, DEVN, 0xFF);
```

..where 0xFF is the enable_mask for all eight channels. There is no benefit to enabling or disabling specific channels so always use 0xFF.

The VR-608 is now in operating mode.

3.1.3 Read Inputs

To read input data from channel(s) call:

```
// Read input data from a channel list

DqAdv608Read(hd, DEVN, sizeof(ch_list)/sizeof(uint32), ch_list, NULL,
NULL, data);
```

to obtain a VR608_READ_DATA structure for all channels in the ch_list array. The VR608_READ_DATA structure will contain:

- Velocity in TIMED, ZPULSE, or NPULSE counter modes.
- Position in TIMED or ZPULSE counter modes, or in decoder mode.
- Tooth Count in TIMED or ZPULSE counter modes, or in decoder mode.
- Timestamp: for any mode, if set in the ch_list parameters.
- Rd_flags: described in the API manual.
- Status: described in the API manual.
- ADC_val: described in the API manual.

Note that unconnected channels will still produce data if queried.

If position information is configured to be stored in the FIFO, you can read the FIFO with the DqAdv608ReadFifo command.

For debugging purposes, it is also possible to directly read the ADC's status with DqAdv608ReadADCStatus() and FIFO with DqAdv608ReadADCFifo().

3.1.4 Stop cleanly

To stop operation, call DqAdv608Enable() with a FALSE parameter as follows:

```
// Disable operation

DqAdv608Enable(hd, DEVN, 0x00);
```

This stops operation of the VR-608 board without changing the configuration.

3.2 Hall Effect Tutorial

The following tutorial provides a brief overview of how to set up Sample608.c to provide the equivalent of Hall Effect mode for input pulses from 0.2 to 18000 Hz for the VR-608. Hall Effect is a special mode for TTL-level-only sensors.

To achieve this effect both the adaptive peak threshold and zero-crossing features are disabled (by setting them to FIXED) configuring the front-end IC to act as a high-performance differential amplifier connected to a precision comparator (with external hysteresis to the comparator for reducing glitches). The Timer Mode Rate must be set to less than half of the lowest input frequency that the sensor is expected to output.

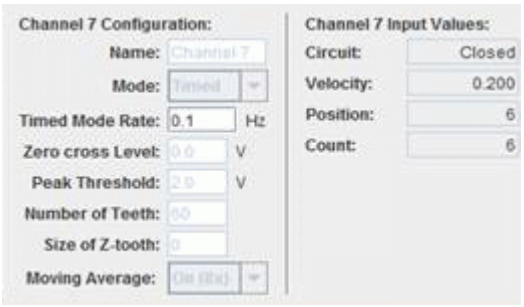
The configuration should contain the follow parameters:

```
// Counter mode
mode = DQ_VR608_MODE_COUNTER | DQ_VR608_MODE_TIMED;
// Set mode for Hall Effect
front_cfg = DQ_VR608_ZC_FIXED | DQ_VR608_APT_FIXED;
// Set extcfg flags
extcfg.ext_flags =
        DQ_VR608_CFGVAL_TMODE_RATE |
        DQ_VR608_EXTFLG_ZC_LEVEL |
        DQ_VR608_EXTFLG_NUM_TEETH |
        DQ_VR608_EXTFLG_ADC_RATE |
        DQ_VR608_EXTFLG_CLEAR |
        0;

extcfg.tmode_rate = 0.1; // Hz, always set to less than (input freq / 2)
extcfg.adc_rate = 200000;                    // Hz, set high
extcfg.apt_th = DQ_VR608_DFLT_APT_TH;        // must set for APT_FIXED
extcfg.zc_level = 2.0;                       // must set for ZC_FIXED
extcfg.adc_mv_avg = DQ_VR608_DFLT_ADC_MV_AVG; // perform averaging
extcfg.ps_param = 60;                        // divide by 60 for RPM
extcfg.z_param = 0;                          // no z-tooth
```

In Sample608.c, you must set READ_ADC_FIFO to TRUE in order to read the values directly from the FIFO.

In PowerDNA Explorer these settings are set as:



Appendix

A. Accessories

The following cables and STP boards are available for the DNx-VR-608.

DNA-CBL-37

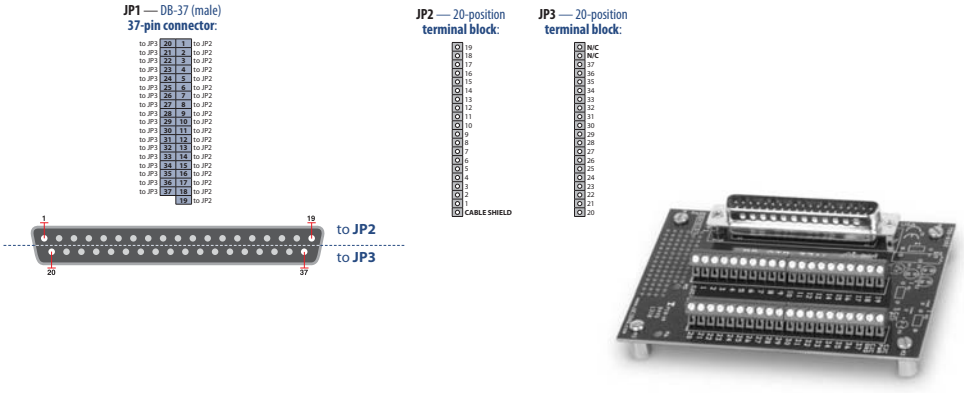
This is a 37-conductor flat ribbon cable with 37-pin male D-sub connectors on both ends. The length is 3ft and the weight is 3.4 ounces or 98 grams.

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 DNx 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 cable. 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.



Pinout and photo of DNA-STP-37 screw terminal panel

Index