United Electronic Industries

The High-Performance Alternative

DNA-DIO-406 Digital Input/Output Layer User Manual

Release 1.0

December 2006 Edition

PN Man-DNA-DIO-406-1206

© Copyright 1998-2006 United Electronic Industries, Inc. All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form by any means, electronic, mechanical, by photocopying, recording, or otherwise without prior written permission.

Information furnished in this manual is believed to be accurate and reliable. However, no responsibility is assumed for its use, or for any infringement of patents or other rights of third parties that may result from its use.

All product names listed are trademarks or trade names of their respective companies.

See the UEI website for complete terms and conditions of sale:

http://www.ueidaq.com/company/terms.aspx

Contacting United Electronic Industries

Mailing Address:

611 Neponset Street
Canton, MA 02021
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:	(781) 821-2890
Fax:	(781) 821-2891

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.

Table of Contents

Appendices

© Copyright 2006 all rights reserved
United Electronic Industries, Inc.

List of Figures

List of Tables

Chapter 1 Introduction

This document outlines the feature set and use of the DNA-DIO-406 digital input/output layer when used with the PowerDNA I/O Cube. This document describes the following products:

- DNA-DIO-406 24-channel industrial digital input/output layer, pre-configured with 12 inputs and 12 outputs (configured as current sinks), with user-programmable hysteresis (deadband)

1.1 Organization of this manual

This PowerDNA DIO-406 User Manual is organized as follows:

1.1.1 Introduction

This chapter provides an overview of PowerDNA DNA-DIO-406 Digital Input/ Output board features, accessories, and what you need to get started.

1.1.2 The DIO-406 Layer

This chapter provides an overview of the device architecture, connectivity, and logic of the DNA-DIO-406 layer.

1.1.3 Programming with the High-Level API

This chapter provides a general description of the how to create a session, configure the session for digital data acquisition/output, and format relevant data.

1.1.4 Programming with the Low-Level API

Low-level API commands for configuring and using the DNA-DIO-406 layer.

Appendices

A. Accessories

This appendix describes the accessories available for use with the DIO-406 layer.

B. Layer Verification

This appendix outlines how to verify calibration for the DIO-406 layer.

Index

This is an alphabetical listing of the topics covered in this manual.

© Copyright 2006 all rights reserved
United Electronic Industries, Inc.

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! *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 you 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**.”

© Copyright 2006 all rights reserved
United Electronic Industries, Inc.

1.2 The DIO-406 Layer

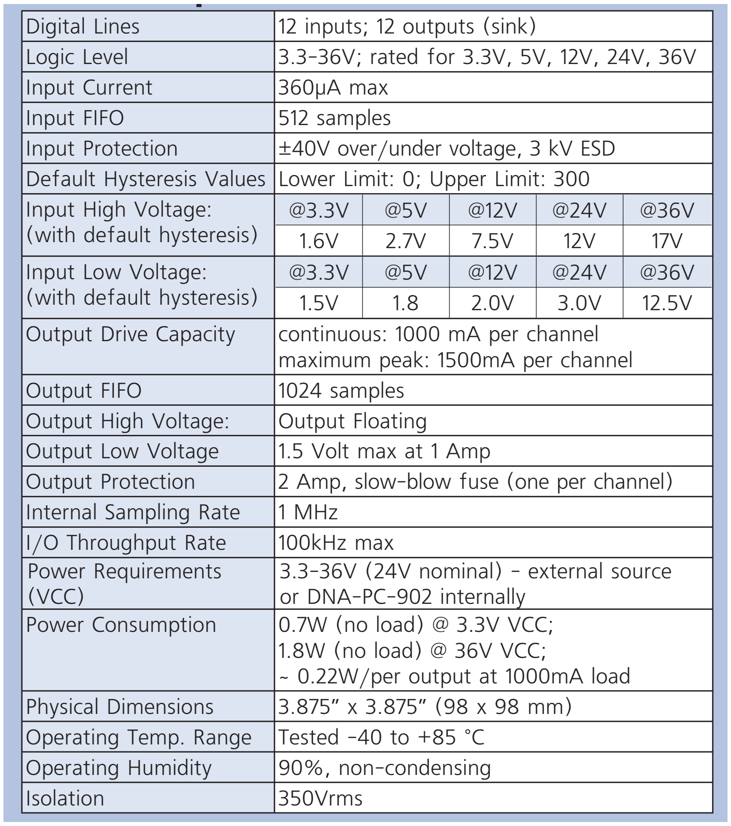
The DNA-DIO-406 is a 24-channel Industrial Input/Output Layer designed for use with a PowerDNA Cube. It has 12 digital inputs and 12 digital outputs (configured as current sinks), user-programmable hysteresis for superior noise immunity, and 350 V_{rms} isolation on the DIO lines. When powered from a user-provided 24VDC nominal power supply (3.3 to 36VDC) or from an internal 24VDC power source such as a DNA-PC-902 Power Conversion Layer, the DNA-DIO-406 can sink up to 1 Amp/channel of current from all 12 outputs, and provide readings on all 12 inputs with change-of-state detection.

Outputs feature low impedance FET transistors configured in “low side” mode (programming a logic 1 connects the output to DGND through a FET; programming a logic 0 keeps the output floating). User loads should be connected between VCC and the output. This layer can also drive solenoids and other inductive devices — each output FET is protected by a reverse diode. (For extra protection, add a reverse diode directly at the inductive load.)

The DNA-DIO-406 has a maximum update rate of 100kS/s.

The technical specifications for the DIO-406 layer are listed in **Table 1-1**.

Table 1-1 DNA-DIO-406 Technical Specification



Digital Lines	12 inputs; 12 outputs (sink)				
Logic Level	3.3-36V; rated for 3.3V, 5V, 12V, 24V, 36V				
Input Current	360µA max				
Input FIFO	512 samples				
Input Protection	±40V over/under voltage, 3 kV ESD				
Default Hysteresis Values	Lower Limit: 0; Upper Limit: 300				
Input High Voltage: (with default hysteresis)	@3.3V	@5V	@12V	@24V	@36V
	1.6V	2.7V	7.5V	12V	17V
Input Low Voltage: (with default hysteresis)	@3.3V	@5V	@12V	@24V	@36V
	1.5V	1.8	2.0V	3.0V	12.5V
Output Drive Capacity	continuous: 1000 mA per channel maximum peak: 1500mA per channel				
Output FIFO	1024 samples				
Output High Voltage:	Output Floating				
Output Low Voltage	1.5 Volt max at 1 Amp				
Output Protection	2 Amp, slow-blow fuse (one per channel)				
Internal Sampling Rate	1 MHz				
I/O Throughput Rate	100kHz max				
Power Requirements (VCC)	3.3-36V (24V nominal) - external source or DNA-PC-902 internally				

Power Consumption	0.7W (no load) @ 3.3V VCC;
	1.8W (no load) @ 36V VCC;
	~ 0.22W/per output at 1000mA load				

Physical Dimensions	3.875” x 3.875” (98 x 98 mm)				
Operating Temp. Range	Tested -40 to +85 °C				
Operating Humidity	90%, non-condensing				
Isolation	350Vrms				

© Copyright 2006 all rights reserved
United Electronic Industries, Inc.

Figure 1-1 is a photo of the DNA-DIO-406 Layer board.

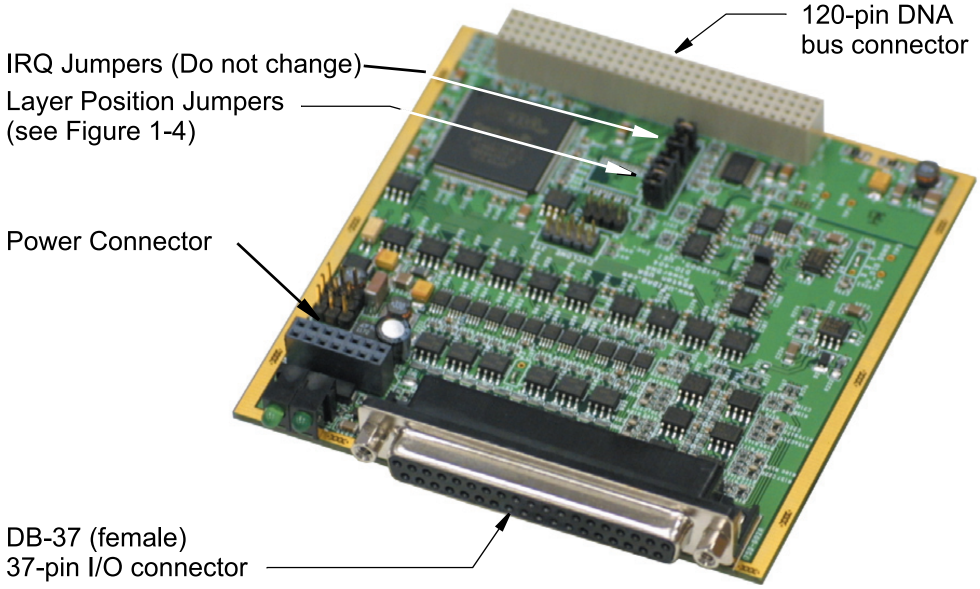


Figure 1-1 A DNA-DIO-406 Digital I/O Layer

1.3 Device architecture

The DNA-DIO-406 Layer has 12 digital inputs and 12 digital outputs (configured as current sinks). The layer can drive up to 1 Amp/channel from each output. A block diagram of the board layer is shown in **Figure 1-2**.

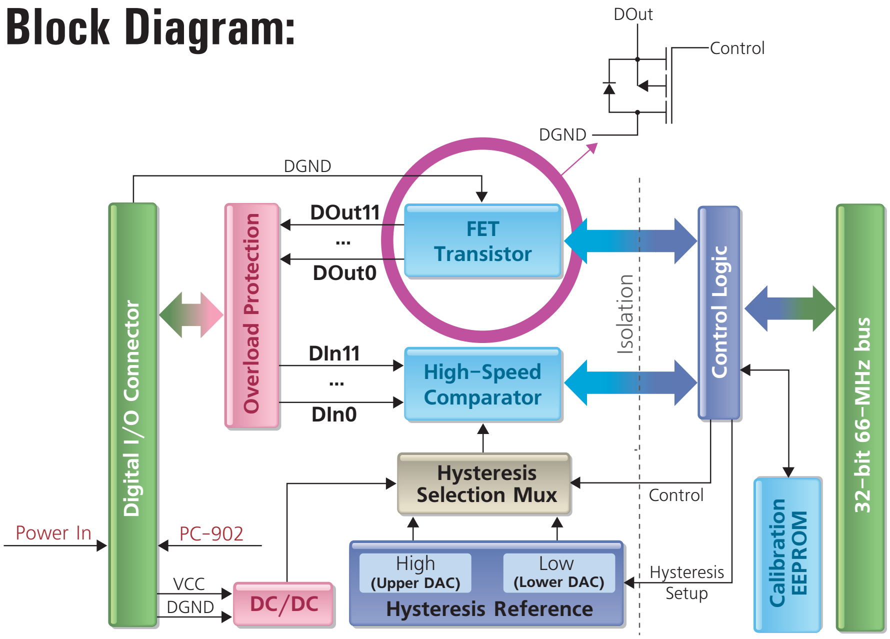


Figure 1-2 Block Diagram of DNA-DIO-406 Device Architecture

Note that the I/O part of the layer is isolated from the logic interface and that overload protection is provided on all inputs and outputs.

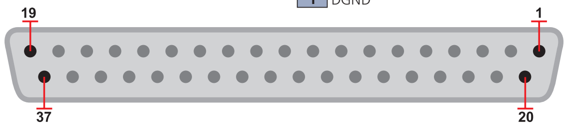
© Copyright 2006 all rights reserved
United Electronic Industries, Inc.

1.4 Layer connectors and wiring

The pinout of the DB-37 37-pin female connector for the DNA-DIO-406 Layer board is shown in **Figure 1-3**.

DB-37 (female)
37-pin connector:

VCC	37	19	**VCC**
DGND	36	18	**VCC**
DOUT11	35	17	DGND
DOUT9	34	16	DOUT10
DOUT8	33	15	DGND
DOUT6	32	14	DOUT7
DOUT5	31	13	DGND
DOUT3	30	12	DOUT4
DOUT2	29	11	DGND
DOUT0	28	10	DOUT1
DIN11	27	9	DGND
DIN9	26	8	DIN10
DIN8	25	7	DGND
DIN6	24	6	DIN7
DIN5	23	5	DGND
DIN3	22	4	DIN4
DIN2	21	3	DGND
DIN0	20	2	DIN1
		1	DGND



Note: Connect external power source to **VCC** pins. All VCC and at least 3 **DGND** pins should be used to supply external power.

Figure 1-3 DB-37 I/O Connector Pinout

Note that the DIO-406 outputs are numbered from DOut0 through DOut11 and the inputs are numbered from DIn0 to DIn11.

Also note the location of the VCC pins. Power must be supplied to the layer in either of two forms:

- By connecting an external 3.3 to 36V power source to the VCC pins directly or through the VCC pins on a DNA-STP-37, STP-37D, or DNA-DIO-O22 terminal panel that is connected to the 37-pin connector on the board.
- By using power from a PC-902 power conversion layer, which supplies power internally to the layer without breaking optoisolation.

When power is provided to the layer, the RDY LED turns on. When no power is supplied, the RDY LED is off, and the DNA-DIO-406 layer cannot operate.

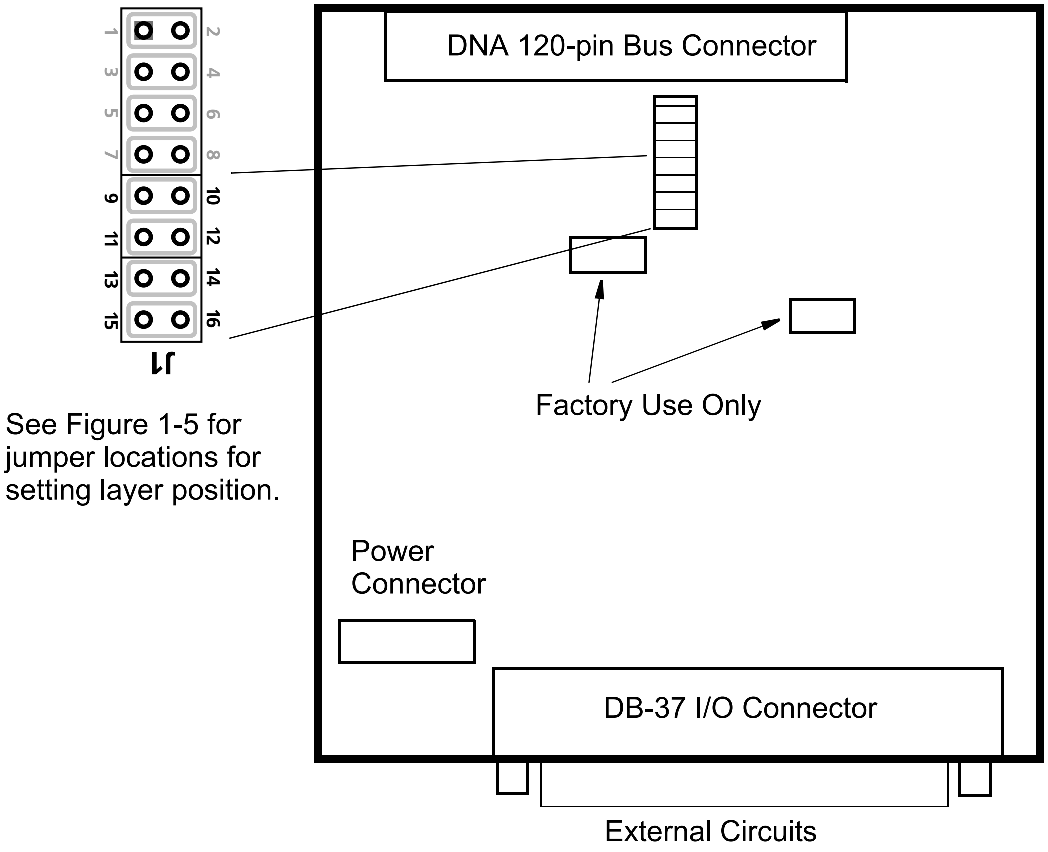


Figure 1-4. Physical Layout of DNA-DIO-406 Layer Board

1.4.0.1 Jumper Settings

A diagram of the jumper block is shown in **Figure 1-5**. To set the layer position jumpers, place jumpers as shown in **Figure 1-5**.

		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 1-5. Diagram of DNA-DIO-406 Layer Position Jumper Settings

1.4.1 Layer Capabilities

Unless a DNA-PC-902 Power Conversion Layer is installed, the inputs and outputs of this layer are powered externally. In this type of configuration, you must supply a DC voltage from 3.3 to 36 VDC to the VCC terminals. The voltage levels are shown in **Table 1-2**.

Table 1-2 DNA-DIO-406 Voltage Levels

VCC	Input “0”	Input “1”	Output “0”	Output “1”
3.3V	1.5V	1.6V	float	3.1V
5V	1.8V	2.7V	float	4.5V
12V	2.0V	7.5V	float	11.5V
24V	3.0V	12V	float	23.4V
36V	12.5V	17V	float	35.4V

1.4.1.1 Input Circuits

Each input circuit is built as shown in **Figure 1-6**.

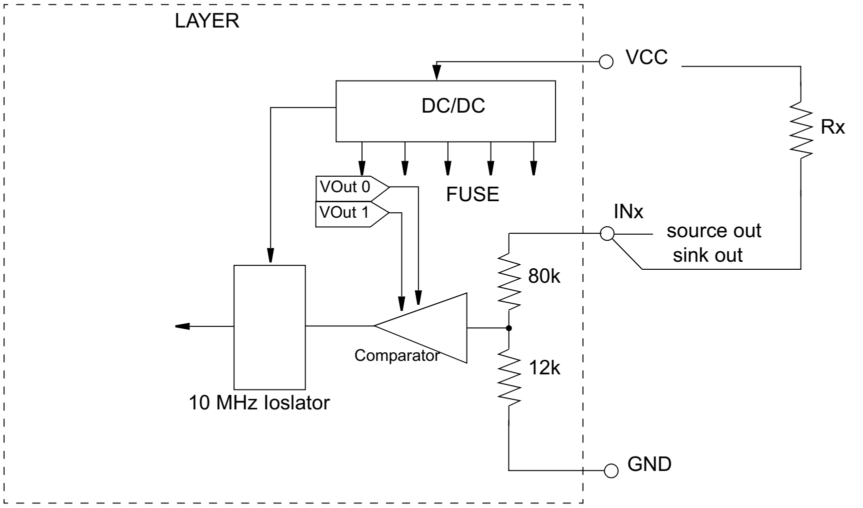


Figure 1-6. Typical Input Circuit Diagram

1.4.1.2 Output Circuits

Each output circuit is built as shown in **Figure 1-7**.

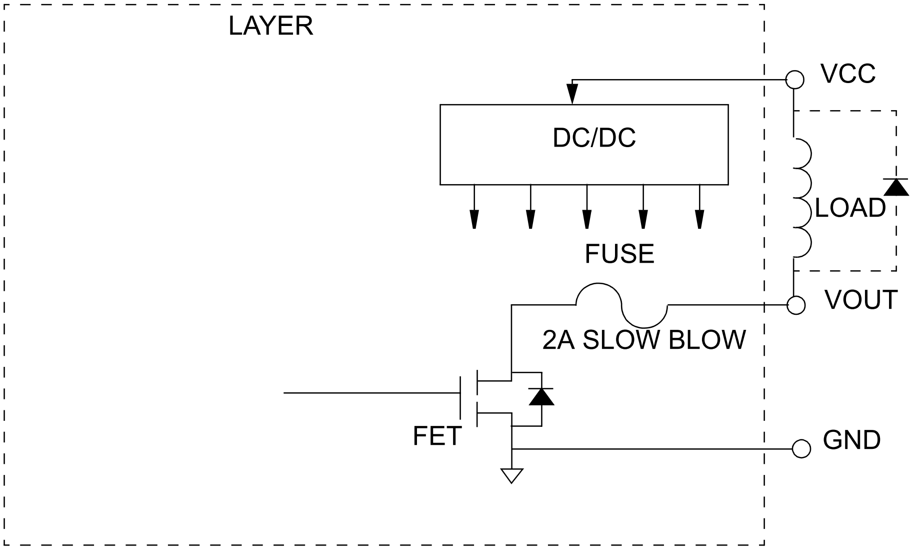


Figure 1-7 Typical Output Circuit Diagram

Chapter 2 Programming with the High Level API

This section describes how to control the PowerDNA DIO-406 with the UEIDAQ high-level API, called Framework.

Framework is object oriented; its objects can be manipulated in the same manner from a wide range of 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 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 resource string

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**.

As an example, the following resource string selects digital input channels 0,1,2,3 on device 1 at IP address 192.168.100.2:

```
"pdna://192.168.100.2/Dev1/Di0:3"
```

NOTE: In Framework, a digital channel corresponds to a physical port on the device.Note that you cannot configure a session only to access a subset of lines within a digital port.

```
//Configure session to read from port 0 on device 1

di_session.CreateDIChannel("pdna://192.168.100.2/
Dev1/Di0");

// Configure session to write to port 0 on device 1

do_session.CreateDOChannel("pdna://192.168.100.2/
Dev1/Do0");
```

NOTE: Sessions are unidirectional. If your device has both input and output ports or has bidirectional ports, you need to configure two sessions: one for input and one for output.

2.3 Configuring the timing

You can configure the DIO-406 to run in simple mode (point by point), buffered mode (ACB mode), or Dmap 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 DIO-406 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.

```
di_session.ConfigureTimingForSimpleIO();
```

2.4 Configuring hysteresis

The PowerDNA DIO-406 layer is equipped with hysteresis (deadband) circuitry that can be programmed to set low and high threshold levels, using custom property commands, as follows:

```
//“lowhysteresis”:A floating-point value
// representing the low hysteresis voltage as a
// percentage of the power supply voltage (Vcc).

// “highhysteresis”: A floating-point value
// representing the high hysteresis voltage as a
// percentage of the power supply voltage (Vcc).

// Program low threshold to 10% and high threshold
// to 90%
double lowHyst = 0.1;
double highHyst = 0.9;
di_session.SetCustomProperty(“lowhysteresis”,
sizeof(double), &lowHyst);
di_session.SetCustomProperty(“highhysteresis”,
sizeof(double), &highHyst);
```

2.5 Reading and Writing Data

Reading data from the DIO-406 is done by using a reader object.

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

```
// Create a reader and link it to the session
// stream
CUeiDigitalReader
reader(di_session.GetDataStream());

// read one scan, the buffer must be big enough to
// contain  one value per channel
uInt16 data;
reader.ReadSingleScan(&data);
```

Data Writing is done by creating a writer object. The following sample shows how to create a writer object and use it to write data.

```
// Create a writer and link it to the session
// stream
CUeiDigitalWriter
writer(do_session.GetDataStream());

// write one scan, the buffer must contain
// one value per channel
uInt16 data = 0xFEFE;
writer.WriteSingleScan(&data);
```

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) by entering:

```
di_session.CleanUp();
```

Chapter 3 Programming with the Low-Level API

This section describes how to program the PowerDNA cube using the low-level API. The low-level API offers direct access to PowerDNA DAQ Bios protocol and also allows you to access device registers directly.

We recommend that, whenever possible, you use the UeiDaq Framework High-Level API (see Chapter 2), which is easier to use.

You should need to use the low-level API only if you are working with an operating system other than Windows.

3.1 Programming hysteresis

The ground level of the inputs can be set from DGND level to VCC level in 1024 steps (increments). For the isolator circuit to set the open input level, it should be above ground level by at least 2.4V.

When programmable hysteresis mode is disabled, an input becomes “1” if the input voltage is 2.4V above selected ground level.

When programmable hysteresis mode is selected, the device logic constantly changes ground level between two programmed levels. This change of ground level occurs at 2kHz rate. Every time the logic changes ground level, it performs a “read”. The logic then produces output based on two consecutive reads at low and high ground level. **Table 3-1** summarizes the result.

Read at low	Read at high	Result
0	0	0
1	0	Keep at previous level
1	1	1

Table 3-1 Logic Level Read Result

© Copyright 2006 all rights reserved
United Electronic Industries, Inc.

The diagram in **Figure 3-1** illustrates the hysteresis feature. The logic stays at “0” until the signal crosses both low and high ground levels. If the signal falls below high ground level but never crosses below low ground level (for more than 1 ms), it remains at “1”.

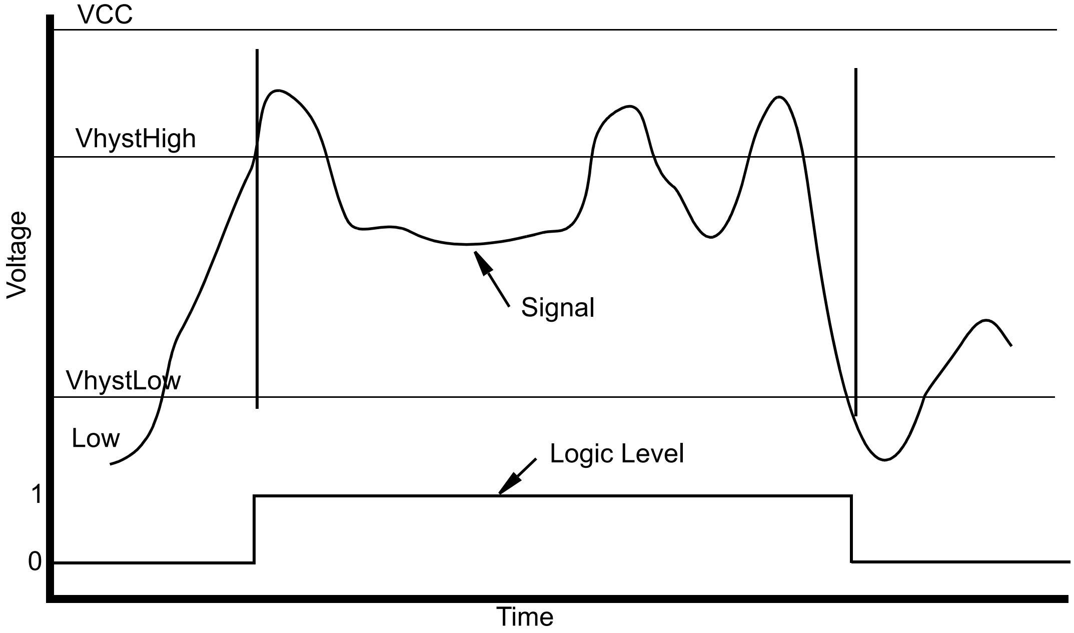


Figure 3-1 The Hysteresis Feature — Logic Level vs. Signal

Hysteresis is a specific feature of DIO-406 layer. To use this feature, you should enable it in the configuration word, as follows:

```
#define DQ_L406_HYSTEN   (1UL<<18)
// hysteresis programming is enabled
```

By default, hysteresis levels are set at 25% of VCC (low) and 75% of VCC (high). User can set hysteresis levels using the layer-specific function below:

```
DqAdv406SetHyst(int hd, int devn, uint16 level0,
uint16 level1)

//level0 and level1 are 10-bit relative values for
//low and high hysteresis levels.
```

© Copyright 2006 all rights reserved
United Electronic Industries, Inc.

3.2 Data Representation

Data representation is straightforward, as shown in**Table 3-2**.

Layer/Bits	31..24	23..12	11..0
DIO-406	Reserved	DOut23..12	DIn11..0

Table 3-2 Data Representation

Note that output lines on DIO-406 layer occupy bits 23…12. Thus, to set up all lines into one, you have to write 0x00FFF000 to DIO-406. States of bits 31…24 don't matter.

3.3 Configuration settings

Configuration setting are passed in the `DqCmdSetCfg()` function.

Note that not all configuration bits apply to the DIO-406 layer.

The following bits are used:

```
#define DQ_LN_MAPPED     (1L<<15)
// For WRRD(DMAP) devices (automatically selected)
#define DQ_LN_ACTIVE     (1L<<1)
// "STS" LED status
#define DQ_LN_ENABLED    (1L<<0)
// enable operations
```

The DIO-406 has a range of layer-specific settings as follows.

Upper part of the configuration word – DIO-406 specific:

```
#define DQ_L406_HYSTEN      (1UL<<18)
               // hysteresis programming is enabled
```

The following modes are reserved for future use:

```
#define DQ_L406_MODESCAN   (FIFO_MODESCAN)
               // single scan update mod (no buffer)
#define DQ_L406_MODEDGE    (1UL << 16)
               // edge detect mode
#define DQ_L406_MODEFIFO   (FIFO_MODEFIFO)
              // continuous acquisition with FIFO
              // (simplified buffer)
#define DQ_L406_MODECONT   (FIFO_MODECONT)
               // continuous acquisition (buffered)
```

DQ_LN_ACTIVE is needed to switch on the "STS" LED on the CPU layer.

DQ_LN_ENABLE enables all operations with the layer.

© Copyright 2006 all rights reserved
United Electronic Industries, Inc.

3.4 Channel list settings

A channel list is not required with the DIO-406.

3.5 Layer-specific commands and parameters

Two layer-specific functions are defined, as follows:

```
DqAdv406Write()
       //This function writes a 24-bit word to the
        //DIO-406 layer using DQCMD_WRCHNL.
DqAdv406Read()
       //This function reads input status using
       // DQCMD_RDCHNL.
DqAdv406SetHyst()
      //This function sets hysteresis levels.
```

These functions can be called any time in configuration and operation mode.

3.6 Using layer in ACB mode

This is a pseudo-code example that highlights the sequence of functions needed to use ACB on the 406 layer. A complete example with error checking can be found in the directory *SampleACB40x*.

Note that we use the `#defines` for a DNA-AO-406 layer.

```
#include "PDNA.h"
// unit configuration word
#define CFG406            (DQ_LN_ENABLED \
                        |DQ_LN_ACTIVE \
                        |DQ_LN_GETRAW \
                        |DQ_LN_IRQEN \
                        |DQ_LN_CVCKSRC0 \
                        |DQ_LN_STREAMING \
                        |DQ_AI30x_MODEFIFO
                        |DQ_AO30x_BI10)
                        uint32 Config = CFG406;
```

STEP 1: Start DQE engine.

```
#ifndef _WIN32
    DqInitDAQLib();
#endif
    // Start engine
    DqStartDQEngine(1000*1, &pDqe, NULL);
    // Open communication with IOM
    hd0 = DqOpenIOM(IOM_IPADDR0, DQ_UDP_DAQ_PORT,
TIMEOUT_DELAY, &RdCfg);
    // Receive IOM crucial identification data
    DqCmdEcho(hd0, DQRdCfg);

    // Set up channel list
    for (n = 0; n < CHANNELS; n++) {
            CL[n] = n;
        }
```

STEP 2: Create and initialize host and IOM sides.

```
        // Now we are going to test device
        //DqAcbCreate(pDqe, hd0, DEVN, DQ_SS0IN, &bcb);
       // Let’s assume that we are dealing with DIO-406
         //device
    dquser_initialize_acb_structure();
        // Now call the function
        DqAcbInitOps(bcb,
                     &Config,
                     0,      //TrigSize,
                     NULL,   //pDQSETTRIG TrigMode,
                     &fCLClk,
                     0,      //float*  fCVClk,
                     &CLSize,
                     CL,
                     0,      //uint32* ScanBlock,
                     &acb);
        printf("Actual clock rate: %f\n", fCLClk);
        // Now set up events
        DqeSetEvent(bcb,
    DQ_eFrameDone|DQ_ePacketLost|DQ_eBufferError|DQ_eP
    acketOOB);
        // Allocate data buffer
        datta = dquser_allocatebuffer();
        // Pre-fill ACB with raw data
        dquser_prefillbuffer(data);
        DqAcbPutScansCopy(bcb, data, // buffer
                          bufsize, // buffer size in
                                   //scans
                          bufsize, // minimum size
                          &size,   // actual copied
                                   //size (from user
                                   // buffer into ACB)
                          &avail);

                               // available free space
                                 // in buffer
```

STEP 3: Start operation.

```
        // Start operations
        DqeEnable(TRUE, &bcb, 1, FALSE);
```

STEP 4: Process data.

```
          // We will not use event notification at first
          //- just retrieve scans
          while (keep_looping) {
              DqeWaitForEvent(&bcb, 1, FALSE,
      EVENT_TIMEOUT, &events);
              if (events & DQ_eFrameDone) {
                  // fill buffer with more data
                  dquser_prefillbuffer(data);
                  DqAcbPutScansCopy(bcb, data,// buffer
                                 bufsize,  // buffer size
                                 MINRQ,   // minimum size
                                    &size,// actual
                                 //copied size from
                                 //user buffer into
                                 //ACB &avail);
                                 // available free space
                                    //in buffer
              }
          }
```

STEP 5: Stop operation.

```
          DqeEnable(FALSE, &bcb, 1, FALSE);
```

STEP 6: 6. Clean up.

```
          DqAcbDestroy(bcb);
          DqStopDQEngine(pDqe);
          DqCloseIOM(hd0);
      #ifndef _WIN32
          DqCleanUpDAQLib();
      #endif
```

3.7 Using layer in DMap mode

This example shows communication between two layers: a Layer 0 DIO-406, and a Layer 1 DIO-406.

```
For a DIO-406, DEVNIN and DEVNOUT would be the
same, and we’d assign a value only to bits 0-11 of
offset, and read bits 0-11 of ioffset.

#include "PDNA.h"
```

STEP 1: Start DQE engine

```
#ifndef _WIN32
```

```
    DqInitDAQLib();
#endif

    // Start engine
    DqStartDQEngine(1000*10, &pDqe, NULL);

    // open communication with IOM
    DqOpenIOM(IOM_IPADDR0, DQ_UDP_DAQ_PORT,
TIMEOUT_DELAY, &DQRdCfg);

    // Set hysteresis at this point
    DqAdv40xSetHyst(hd0, DEVNIN, 0x132, 0x2CA);

    // Receive IOM crucial identification data
    DqCmdEcho(hd0, DQRdCfg);

    for (i = 0; i < DQ_MAXDEVN; i++) {
        if (DQRdCfg->devmod[i]) {
            printf("Model: %x Option: %x\n",
DQRdCfg->devmod[i], DQRdCfg->option[i]);
        } else {
            break;
        }
    }
```

STEP 2: Create and initialize the host and IOM sides.

```
    DqDmapCreate(pDqe, hd0, &pBcb, UPDATE_PERIOD,
&dmapin, &dmapout);
```

STEP 3: Add channels into DMap.

```
    DqDmapSetEntry(pBcb, DEVNIN, DQ_SS0IN, 0,
DQ_ACB_DATA_RAW, 1, &ioffset);
    DqDmapSetEntry(pBcb, DEVNOUT, DQ_SS0OUT, 0,
DQ_ACB_DATA_RAW, 1, &ooffset));

    DqDmapInitOps(pBcb);
    DqeSetEvent(pBcb,
DQ_eDataAvailable|DQ_ePacketLost|DQ_eBufferError|D
Q_ePacketOOB);
```

STEP 4: Start operation.

```
    DqeEnable(TRUE, &pBcb, 1, FALSE);
```

STEP 5: Process data

```
    while (keep_looping) {
        DqeWaitForEvent(&pBcb, 1, FALSE, timeout, 
&eventsin);
        if (eventsin & DQ_eDataAvailable) {
            datarcv++;
            printf("\ndata %08x ", 
*(uint32*)ioffset);
            *(uint32*)ooffset = datarcv;
        }
    }
```

STEP 6: Stop operation.

```
    DqeEnable(FALSE, &pBcb, 1, FALSE);
```

STEP 7: Clean up.

```
    DqDmapDestroy(pBcb);
    DqStopDQEngine(pDqe);
    DqCloseIOM(hd0);
#ifndef _WIN32
    DqCleanUpDAQLib();
#endif
```

Appendices

A. Accessories

The following cables and STP boards are available for the DIO-406 layer.

DNA-PC-902

+24V power conversion layer; supplies +24V to the DIO-406 and external devices at up to 40W.

DNA-CBL-37

A 3ft, 37-way flat ribbon cable that connects the layer to a terminal panel.

DNA-DIO-O22

An external accessory for the DNA-DIO-406. The DNA-DIO-022 panel distributes 48 DIO channels into 3 groups of 16 lines that connect to three Opto-22 compatible connectors.

DNA-STP-37

37-way screw terminal panel.

B. Layer Verification

The DIO-406 layer does not require calibration.

Layer verification is performed using “simod 1” command. To access it, attach a serial interface to the PowerDNA cube and run a serial terminal program on the host PC. The DIO-406 “simod 1” command allows you to read and write a port (“r” and “w” command) as well as select a hysteresis DAC (“1” and “2”) and adjust it, using “[,],{,}” keys.

“q” or Esc causes the routine to exit.

The verification is done by setting up hysteresis levels (default are 25% for low and 75% for high ground levels) and continuously reading inputs while changing the voltage level on inputs. The easiest way to verify outputs is to attach LEDs between layer outputs and DGND, in series with proper resistors. For example, you can use 2 to 4.7 kOhm resistors to limit current flowing through LEDs.

Index