Multi-PCI

Data Acquisition System

Reference Manual

Version 1.0

Another Quality Product
by Quanser Consulting

Quanser
ADVANCED CONTROL SYSTEMS

How to contact Quanser Consulting:

	(905) 940-3575	Telephone
	(905) 940-3576	Facsimile
	80 Esna Park Drive Markham, ON Canada L3R 2K8	Mail
	http://www.quanser.com	Web
	mailto://info@quanser.com	General information

MultiQ-PCI Data Acquisition System Reference Manual

Printed in Canada.

Table of Contents

Table of Contents

List of Figures

List of Tables

List of Examples

1. Introduction

The MultiQ-PCI card is a plug-and-play, multi-function I/O card. Some of its features include:

- 48 digital I/O channels.
  - 20 of the digital I/O channels have edge detection and interrupt capability.
  - 7 of the digital outputs may be used as counter overflow outputs.
- Watchdog timer with several selectable reset periods that can reset the PCI bus.
- Six 24 bit up/down counters arranged in 3 pairs with:
  - inputs that can be driven in various modes (1x, 2x, 4x) from incremental encoders inputs, digital inputs, the paired counter's overflow, the system clock or driven by software.
  - the ability to generate an interrupt on counter overflow or from encoder/digital input index pulse.
  - the ability to be pre-loaded/cleared on a counter overflow.
  - the ability for the output of the second counter to be captured on the overflow of the first.
  - the ability to be programmed as a periodic interrupt generator.
  - battery backup of the counter circuitry to prevent count loss during a power failure.
- Charge control of the backup Ni-Cad battery.
- 16 differential analog inputs with 14 bit resolution.
- 4 analog outputs with 13 bit resolution and remote sense inputs to compensate for any external output resistance.

1.1 Card Identification

The MultiQ-PCI card may be identified in the PCI system using the following identifiers:

Vendor ID:	0x1131
Device ID:	0x7146

1.2 Installation

Refer to the MultiQ-PCI User's Guide for installation instructions, as well as detailed information about the MultiQ-PCI terminal board. Be sure to handle the card with caution. The card is shown in Figure 1 below.

The MultiQ-PCI data acquisition card is susceptible to damage from static electricity. Always ground yourself when installing the MultiQ-PCI card in your computer!

To prevent ESD damage to the card:

- Do not remove the circuit board from its protective anti-static bag until you are ready to configure the board for installation.
- Handle the circuit board only at grounded, ESD protected stations.
- Remove power from the PCI system before installing or removing the circuit board.

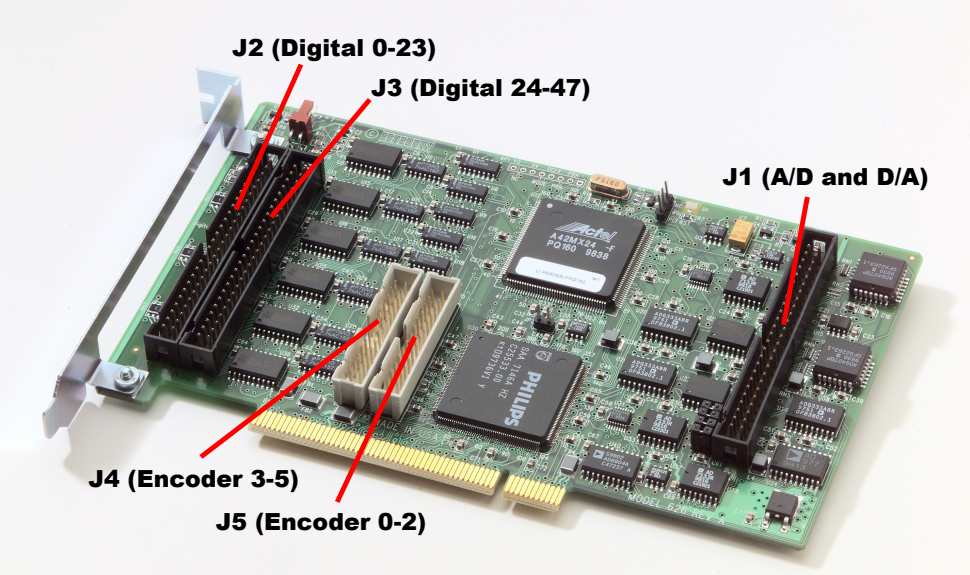

Figure 1: MultiQ-PCI Data Acquisition Card

2. Board Overview

2.1 Layout

The layout of the MultiQ-PCI card is depicted in Figure 1. The connectors are labeled to facilitate connection to the MultiQ-PCI terminal board. Detailed pin-outs for the connectors may be found in the appendices.

Although most of the functionality of the board is software-programmable, some features can be enabled or disabled using jumpers on the board. Table 1 describes the jumpers on the card.

Table 1: Jumpers

Jumper	Description
JP1	Watchdog timer PCI reset
JP2	Battery connection for battery backup
JP3-JP6	D/A sense loopback

2.2 Block Diagram

A schematic of the board is illustrated in Figure 2 on page 4. The schematic shows the functional components of the board. The board is functionally divided into five components:

- Counters.
- Digital I/O
- A/D Conversion
- D/A Conversion
- Watchdog Timer

The counters handle a broad variety of functionality, including edge capture, periodic interrupt generation, encoder inputs and more.

The different components are described in detail in the proceeding chapters.

Block Diagram

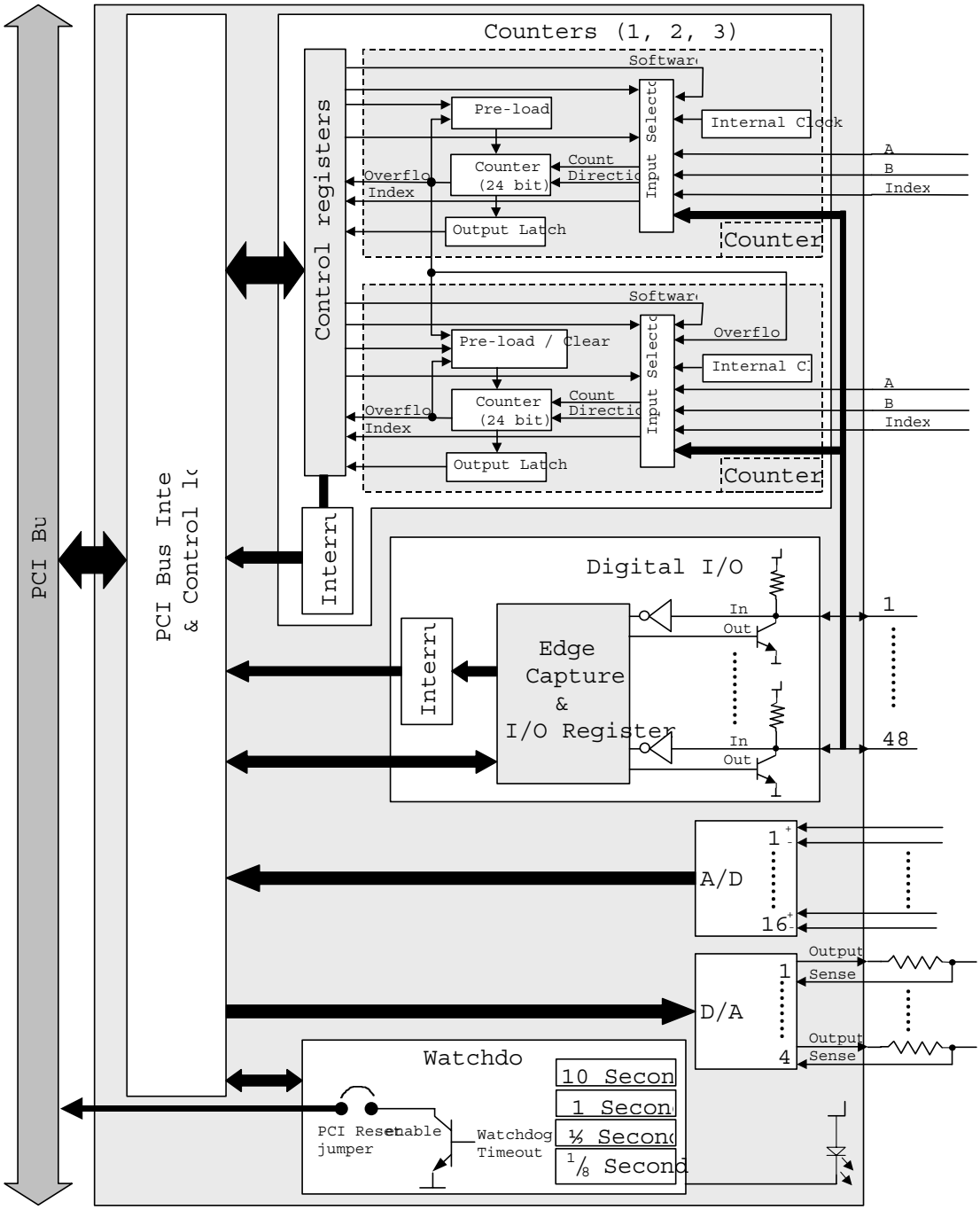

Figure 2: System Block Diagram

3. Initial States After Reset

This section describes the state of the MultiQ-PCI registers on a hardware reset. The same states result when the watchdog timer expires and the watchdog timer is enabled.

3.1 Miscellaneous Registers

Table 2: MISC1 & MISC2 Registers Initial States			
Register or I/O	**State**	**See**	**Page**
Watchdog oscillator	Oscillator 1 is selected.	Watchdog/Miscellaneous Registers	31
Watchdog	Disabled		
MISC1 Write Enable	Disabled.		
Watchdog period	$^{1}/_{8}^{th}$ second		
EDCAP	0		

3.2 Digital I/O

Table 3: Digital I/O Initial States			
Register or I/O	**State**	**See**	**Page**
Digital I/O 0-7	All inputs. (pull up resistors enabled)	Digital I/O	7
Digital I/O 8-47	Indeterminate (will depend on the inputs during reset)		
Digital I/O 0-5	Will work as digital inputs not counter overflows		
EDGSEL INTSEL CAPSEL	Cleared		

3.3 Counters

If there is a charged external battery connected to the system then most of the functionality of the counters is held through a reset/power failure. The interrupts are disabled to allow the system time to get back up and running.

Table 4: Counter Initial States			
Register or I/O	**State**	**See**	**Page**
Counter input selection	Unchanged	Counters	15
Edge detection			
Multiplier			
Clear Counter B			
Pre-load trigger			
Pre-load registers			

4. Digital I/O

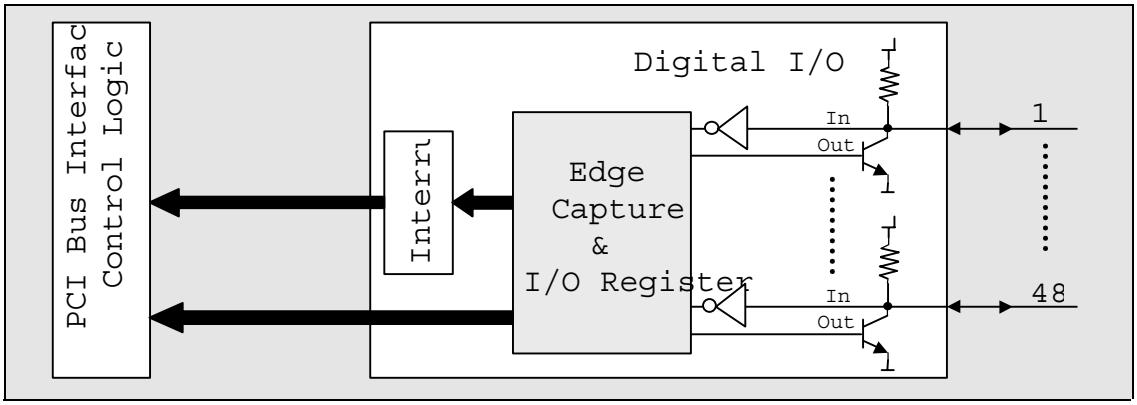

Figure 3: Digital I/O Channels

4.1 Overview

The MultiQ-PCI board provides 48 digital I/O channels. 40 of these channels (channels 0-39) offer edge detection and interrupt on edge detection. Either a positive or negative edge can be detected. The other 8 channels (40-47) have simple input/output functionality only.

Each channel can function as an input or an output.

Any channel that is to be used as an input must have a '0' written to its output control register. This ensures that the output transistor is turned off, allowing the pull-up resistor to pull the channel high. **If a '0' is not written then the transistor will have to sink high currents if the input is driven high externally. (See Figure 3 "Digital I/O Channels" above.)**

Table 5: Digital I/O Register Offsets						
Write				**Read**		
Offset (Hex)	***Register***	***Description***		***Offset (Hex)***	***Register***	***Description***
40	Not used			40	DINA	Status of digital inputs (0-15)
42	WRINTSELA	Interrupt Enable	(0-15)	42		
44	WREDGSELA	Edge Selection	(0-15)	44		
46	WRCAPSELA	Capture Enable	(0-15)	46		
48	DOUTA	Write to Digital Output	(0-15)	48	RDCAPFLGA	Edges captured so far (0-15)
4A	Not used			4A	RDINTSELA	Status of interrupt enable register A
4C				4C	RDEDGSELA	Status of edge selection register A
4E				4E	RDCAPSELA	Status of capture enable register A
50				50	DINB	Status of digital inputs (16-31)
52	WRINTSELB	Interrupt Enable	(16-31)	52		
54	WREDGSELB	Edge Selection	(16-31)	54		
56	WRCAPSELB	Capture Enable	(16-31)	56		
58	DOUTB	Write to Digital Output	(16-31)	58	RDCAPFLGB	Edges captured so far (16-31)
5A	Not used			5A	RDINTSELB	Status of interrupt enable register B
5C				5C	RDEDGSELB	Status of edge selection register B
5E				5E	RDCAPSELB	Status of capture enable register B
60				60	DINC	Status of digital inputs (32-47)
62	WRINTSELC	Interrupt Enable	(32-47)	62		
64	WREDGSELC	Edge Selection	(32-47)	64		
66	WRCAPSELC	Capture Enable	(32-47)	66		
68	DOUTC	Write to Digital Output	(32-47)	68	RDCAPFLGC	Edges captured so far (32-47)
6A	Not used			6A	RDINTSELC	Status of interrupt enable register C
6C				6C	RDEDGSELC	Status of edge selection register C
6E				6E	RDCAPSELC	Status of capture enable register C

4.2 Writing to the Digital Outputs

There are 3 digital output registers used to write to the outputs. The format of the data to be written to each register is shown below.

Table 6: Writing to Digital Outputs																	
Offset (Hex)	**Register Name**	**Output Channel Number**															
		Bit 15	***Bit 14***	***Bit 13***	***Bit 12***	***Bit 11***	***Bit 10***	***Bit 9***	***Bit 8***	***Bit 7***	***Bit 6***	***Bit 5***	***Bit 4***	***Bit 3***	***Bit 2***	***Bit 1***	***Bit 0***
48	DOUTA	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
58	DOUTB	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
68	DOUTC	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32

Writing a “0” to a bit in the output register will cause the corresponding output to turn off and thus give an output of 5V. When a ‘1’ is written then the output transistor is turned on causing the output channel to give an output of 0V.

When an I/O channel is to be used as an input then a ‘0’ must be written to the corresponding bit of that channels output register. This is to prevent high currents caused if the output transistor is turned on while a 5V input signal is being applied to the input (see Figure 3 on page 7).

4.2.1 Using Digital Outputs 0-5 as Counter Overflows

Digital outputs 0-5 can be setup to pull the corresponding output load low for a short pulse (500 nsecs) when the counters overflow. (See Table 23 "Write to MISC2 90 (hex)" on page 31)

4.3 Read the Digital Inputs

There are three digital input registers used to read the inputs. The format of the data returned by each register is shown in the table below.

Table 7: Reading Digital Inputs

Offset (Hex)	Register Name	Input Channel Number															
		Bit 15	*Bit 14*	*Bit 13*	*Bit 12*	*Bit 11*	*Bit 10*	*Bit 9*	*Bit 8*	*Bit 7*	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
40	DINA	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
50	DINB	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
60	DINC	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32

Each digital input has a 10K Ohm pull-up resistor to +5V. With a 5V input or no input connected a ‘0’ will be read from the digital input register. Pulling the input to ground will cause a ‘1’ to be read from the digital input register. The corresponding output register must have “0” written to it before it can be used as an input.

4.4 Capturing Digital Input Edges

This feature is not available on inputs 20-23 and 44-47 (the grayed out areas of the tables). Reading any edge capture data or interrupt data from these inputs will always return 0. Writing to any of edge capture or interrupt register for these inputs will have no effect.

4.4.1 Selecting Positive or Negative Edge Capture

Table 8: Selecting Positive or Negative Edge Capture																	
Offset (Hex)	Register Name	Input Channel Number															
		Bit 15	*Bit 14*	*Bit 13*	*Bit 12*	*Bit 11*	*Bit 10*	*Bit 9*	*Bit 8*	*Bit 7*	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
44	WREDSELA	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
54	WREDSELB	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
64	WREDSELC	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32

Write to the WREDSEL register with a ‘0’ to select negative edges (transition from 1 to 0) or a ‘1’ to select positive edges (transition from 0 to 1). Note that these will take effect once edge capturing for a particular channel has been enabled.

4.4.2 Interrupting on Edge Capture

A captured edge is usually set up to be accompanied by an interrupt. (See “Enabling Interrupt on Edge Capture” on page 13). If this is not done then the RDCAPFLG will need to be polled to see if an edge has occurred.

4.4.3 Enabling and Disabling Edge Capture

Table 9: Enabling & Disabling Edge Capture																	
Offset (Hex)	Register Name	Input Channel Number															
		Bit 15	*Bit 14*	*Bit 13*	*Bit 12*	*Bit 11*	*Bit 10*	*Bit 9*	*Bit 8*	*Bit 7*	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
42	WRCAPSELA	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
52	WRCAPSELB	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17
62	WRCAPSELC	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33

Before any edges may be captured, the desired channel(s) must have edge-capturing enabled. Enabling or disabling edge capture is done using the EDCAP bit of MISC1 in conjunction with the WRCAPSEL registers. When the EDCAP bit is set to '1', writing to a WRCAPSEL register will enable those channels with the corresponding bit set to 1. When the EDCAP bit is set to '0', writing to a WRCAPSEL register will disable those channels with the corresponding bit set to 1.

Thus, to enable edge capture for a set of digital input channels:

1. Write to the EDCAP bit of the MISC1 register.

   ‘1’ is used to enable one or more channels.

   ‘0’ is used to disable one or more channels.

2. Write to the WRCAPSELA, WRCAPSELB or WRCAPSELC register to apply the above command to those channels with their corresponding bits set to ‘1’ in WRCAPSELA, WRCAPSELB or WRCAPSELC.

```
void setupEdgeCapturing(int board)
{
        // Clear the EDCAP bit (disable command)
        WriteReg(board, 0x88, value & 0xE000);          // 0x88 = MISC1 register

        // Clear all previously captured events and interrupts on channels 0-15
        WriteReg(board, 0x46, 0xffff);                  // 0x46 = WRCAPSELA register

        // Set up channels 2, 9, and 14 to capture positive edges
        // The other channels will capture negative edges if enabled.
        WriteReg(board, 0x44, 0x2102);                  // 0x44 = WREDSELA register

        // Enable interrupts on channels 2 and 9 when an edge is captured
        WriteReg(board, 0x42, 0x0102);                  // 0x42 = WRINTSELA register

        // Set the EDCAP bit (enable command)
        WriteReg (board, 0x88, value | 0x1000);         // Set EDCAP bit

        // Enable edge capturing on channels 2, 9 and 14
        WriteReg(board, 0x46, 0x2102);
}

void interruptHandler(void)
{
        // When an interrupt occurs, read the RDCAPFLG register to determine which
        // channel(s) caused the interrupt. A '1' bit indicates an edge has been
        captured.
        unsigned int channels = ReadReg(board, 0x24);  // 0x24 = RDCAPFLAG register

        // Clear the EDCAP bit (disable command)
        WriteReg(board, 0x88, value & 0xE000);

        // Clear the interrupt and disable edge capturing on those channels that
        // caused the interrupt.
        WriteReg(board, 0x46, channels);

        // Since edge capturing for a channel must be disabled to clear the interrupt
        // condition, edge capturing must be re-enabled to capture more edges.

        // Set EDCAP bit (enable command)
        WriteReg(board, 0x88, value | 0x1000);

        // Enable edge triggering on channels 2, 9 and 14.
        WriteReg(board, 0x46, 0x2102);
}
```

Example 1: Capturing Positive Edges on Channels 2, 9 and 14, Interrupting on 2 and 9

4.5 Reading Edge Capture Status Registers

4.5.1 Reading which Channels have Captured Edges

Table 10: Reading Captured Edges																	
Offset (Hex)	Register Name	Input Channel Number															
		Bit 15	*Bit 14*	*Bit 13*	*Bit 12*	*Bit 11*	*Bit 10*	*Bit 9*	*Bit 8*	*Bit 7*	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
48	RDCAPFLGA	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
58	RDCAPFLGB	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
68	RDCAPFLGC	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32

RDCAPFLGA, RDCAPFLGB and RDCAPFLGC are used to determine which, if any, channels have captured edges. A ‘1’ shows an edge has been captured for the corresponding channel.

4.5.2 Reading the Status of the Interrupt Enable Registers

Table 11: Reading the Status of the Interrupt Enable Registers																	
Offset (Hex)	Register Name	Input Channel Number															
		Bit 15	*Bit 14*	*Bit 13*	*Bit 12*	*Bit 11*	*Bit 10*	*Bit 9*	*Bit 8*	*Bit 7*	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
4A	RDINTREGA	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
5A	RDINTREGB	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
6A	RDINTREGC	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32

RDINTREGA, RDCAPFLGB and RDCAPFLGC are used to determine which, if any, channels will generate an interrupt on an edge capture. A ‘1’ shows that the corresponding channel will generate an interrupt on edge capture (if the channel is enabled).

4.5.3 Reading the Status of the Edge Selection Register

Table 12: Reading the Status of the Edge Selection Registers																	
Offset (Hex)	Register Name	Input Channel Number															
		Bit 15	*Bit 14*	*Bit 13*	*Bit 12*	*Bit 11*	*Bit 10*	*Bit 9*	*Bit 8*	*Bit 7*	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
4C	RDEDGREGA	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
5C	RDEDGREGB	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
6C	RDEDGREGC	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32

RDEDGREGA, RDCAPFLGB and RDCAPFLGC are used to determine which channels will capture positive edges and which will capture negative edges(when they are enabled). A transition from 0 to 1 is considered a positive edge.

4.5.4 Reading the Status of the Capture Enable Register

Table 13: Reading the Status of the Capture Enable Registers																	
Offset (Hex)	Register Name	Input Channel Number															
		Bit 15	*Bit 14*	*Bit 13*	*Bit 12*	*Bit 11*	*Bit 10*	*Bit 9*	*Bit 8*	*Bit 7*	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
4E	RDCAPSELA	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
5E	RDCAPSELB	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
6E	RDCAPSELC	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32

RDCAPSELA, RDCAPSELB and RDCAPSELC are used to determine which, if any, channels have edge capturing enabled. A ‘1’ shows that the corresponding channel has edge capturing enabled.

4.5.5 Clearing a Captured Edge

Once an edge has been captured it can be cleared by disabling it in its WRCAPSEL register. Doing this will also clear any accompanying interrupt. Obviously it will need to be re-enabled if another edge is to be captured.

4.6 Digital Input Interrupts

4.6.1 Enabling Interrupt on Edge Capture

Table 14: Enabling Interrupt on Edge Capture																	
Offset (Hex)	Register Name	Input Channel Number															
		Bit 15	*Bit 14*	*Bit 13*	*Bit 12*	*Bit 11*	*Bit 10*	*Bit 9*	*Bit 8*	*Bit 7*	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
42	WRINTSELA	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
52	WRINTSELB	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
62	WRINTSELC	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32

Write to the correct WRINTSEL register with a ‘1’ to enable and ‘0’ to disable interrupts on that channel. An interrupt will only occur if the channel has been enabled using one of the WRCAPSEL registers and the WREDCAP register. The interrupts do not have to be enabled to capture an edge, an edge can still be captured but the RDCAPFLG registers will have to be polled to see if an edge capture has occurred.

4.6.2 Digital Interrupt Handling

There is only one main interrupt to the system from the MultiQ-PCI. Once this interrupt has been received it is up to the user to determine whether the interrupt originated from the counters (See “Counter Interrupt Handling” on page 27) or the digital inputs.

Digital Input Interrupts

The RDCAPFLG registers and the corresponding RDINTREG registers need to be logically ‘AND-ed’ together to determine whether an interrupt occurred on a particular digital input channel.

An interrupt must be cleared once it has been handled to prevent it from causing multiple interrupts. (See “Clearing a Captured Edge” on page 13).

5. Counters

5.1 Overview

The MultiQ-PCI has six counters arranged in 3 pairs : 1A, 1B, 2A, 2B, 3A and 3B. One of these pairs is shown in Figure 7 on page 20. They can be used in pairs or “stand alone”. Each counter can be used with the internal quadrature encoder interface for positioning systems, as timers using the internal clock, or as external (hardware) or internal (software) event counters. When paired, a 48 bit counter can be achieved and development of functions like frequency counting is easy, as one counter in the pair can feed the other counter either as a count, clear or latch input.

The counter features include:

- Counters can be completely software driven.
- Counters can be driven from digital inputs 0-15.
- Counters can be driven from internal timers.
- Counters can be driven from encoder inputs.
- Encoder input buffers interface directly to TTL, CMOS, or differential RS422 signals.
- Quadrature decoder logic is available to detect and convert encoder edges into clock and direction signals.
- Encoder quadrature multiplier (x1, x2, x4)
- 24-bit up/down counters that can be pre-loaded by various triggers.
- Counters can be read on the fly or captured by various triggers to be read later.
- Selectable counter direction for timer and event counting modes.
- Index input or overflow count can re-load counter with preset values.
- Programmable interrupt on rollover or index pulse.
- 5 Volt encoder power is available at encoder connector.

5.1.1 Latches

From Figure 7 "Block diagram of a counter pair" it can be seen that there is one latch per pair of counters.

Counter A or B can cause it's own count to be latched on receiving an index pulse. Or, an overflow from counter A can cause counter B's count to be latched. These latches can be accompanied with an interrupt and the value captured in the latch can be read later.

The latches can also be setup to latch as they are read. This allows the present value of the selected counter to be read.

5.1.2 Clear/Pre-load

The system can be programmed so that an overflow from Counter A will clear counter B. This is useful for frequency counting where counter B is latched & cleared at an interval determined by the time it takes for counter A to overflow.

Both counters can be programmed so that an index or overflow from a counter will pre-load itself with the contents of it's pre-load register. This feature could be used to clear the counter by loading it with 0. This feature would be used for setting up a programmable interval timer. Both the clear & pre-load can be accompanied by an interrupt.

5.1.3 Interrupts

Each counter can generate an interrupt on overflow, index or both.

5.1.4 Overflow Outputs

The overflow output of each counter can be individually connected to a digital output. When the counter overflows a single pulse will occur pulling the output load low for 2 clock cycles (500 nsec).

5.1.5 Index Inputs

Each counter has a hardware or software controlled index input that can programmed to perform clear, pre-load or latch functions. (See "Clear/Pre-load" and "Latches" above).

5.1.6 Counter Inputs

Each counter has various input modes including:

- Quadrature encoders with hardware or software controlled index.
- Single-ended event, direction and index inputs.
- System clock with hardware or software control of direction and index for event timing.

Each counter channel has two clock input phases (referred to as channels in this manual) named “A” and “B”. Depending on the application, one or both of these signals may be connected to an encoder or to a pulse source.

If both phases are used, they are assumed to be quadrature encoded, meaning that they are 90º out of phase. In this case, counters will count both up and down based on the timing relationship of the two-phase inputs.

If only one phase is used, the input is said to be single-phase. In this case, counter channels will count either up or down for each pulse on the phase input depending on the polarity of the direction input. This configuration is typically used to count pulses from devices that produce a single clock output.

Quadrature encoders for positioning systems have advantages over single-phase encoders. Counters will not accumulate errors when an encoder changes direction or dithers about a state transition boundary.

Also, it is possible to increase encoder resolution by clocking the counters at a multiple of the single-phase clock rate.

5.1.7 Power Failure

If an external NiCad battery is connected and fully charged and the Battery Enable bit of MISC2 is set (see “Battery Backup” on page 35), then all the counter registers will be held and the counters will continue to operate if power is lost, even though the rest of the system is down.

The counter interrupts are disabled when the system powers up but if an interrupt occurred during power down then **it will occur the moment the interrupts are re-enabled**.

5.1.8 Counter Registers

Table 15: Counter Registers					
Write			Read		
Offset (hex)	*Register Name*	*Description*	*Offset (hex)*	*Register Name*	*Description*
0	CR1A	Counter 1A setup register	0	CR1A	Counter 1A setup register
2	CR1B	Counter 1B setup register	2	CR1B	Counter 1B setup register
4	CR2A	Counter 2A setup register	4	CR2A	Counter 2A setup register
6	CR2B	Counter 2B setup register	6	CR2B	Counter 2B setup register
8	CR3A	Counter 3A setup register	8	CR3A	Counter 3A setup register
A	CR3B	Counter 3B setup register	A	CR3B	Counter 3B setup register
C	PRE1ALSW	Pre-loads 1A lsw	C	LATCH1ALSW	Reads Latch 1A lsw
E	PRE1AMSW	Pre-loads 1A msw	E	LATCH1AMSW	Reads Latch 1A msw
10	PRE1BLSW	Pre-loads 1B lsw	10	LATCH1BLSW	Reads Latch 1B lsw
12	PRE1BMSW	Pre-loads 1B msw	12	LATCH1BMSW	Reads Latch 1B msw
14	PRE2ALSW	Pre-loads 2A lsw	14	LATCH2ALSW	Reads Latch 2A lsw
16	PRE2AMSW	Pre-loads 2A msw	16	LATCH2AMSW	Reads Latch 2A msw
18	PRE2BLSW	Pre-loads 2B lsw	18	LATCH2BLSW	Reads Latch 2B lsw
1A	PRE2BMSW	Pre-loads 2B msw	1A	LATCH2BMSW	Reads Latch 2B msw
1C	PRE3ALSW	Pre-loads 3A lsw	1C	LATCH3ALSW	Reads Latch 3A lsw
1E	PRE3AMSW	Pre-loads 3A msw	1E	LATCH3AMSW	Reads Latch 3A msw
20	PRE3BLSW	Pre-loads 3B lsw	20	LATCH3BLSW	Reads Latch 3B lsw
22	PRE3BMSW	Pre-loads 3B msw	22	LATCH3BMSW	Reads Latch 3B msw

5.2 Encoder Connections

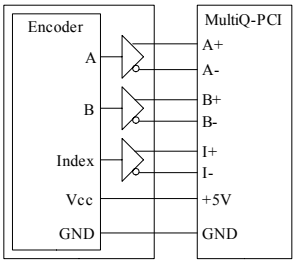

Figure 4: Differential pair encoder

(output with complement)

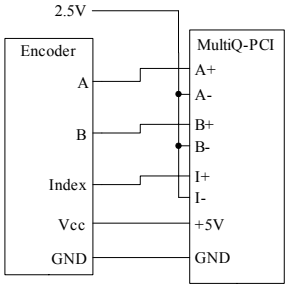

Figure 5: Single-ended encoder

(TTL or CMOS)

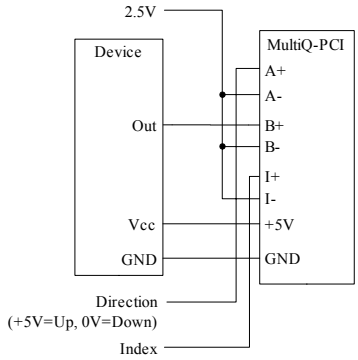

Figure 6: Single-ended event counter with external count direction and index control.

(TTL or CMOS)

Note: If the counters are driven in a single-ended mode as shown in Figure 5 and Figure 6, then the unused phase inputs do not have to be biased at 2.5V. Each encoder input has two 100K resistors that bias the input to 2.5V when no driving source is present.

The encoders are also referred to by channel numbers. Counters map to channel numbers according to the following table:

Table 16: Encoder Channels

Counter	Encoder
1A	0
2A	1
3A	2
1B	3
2B	4
3B	5

5.3 Block Diagram

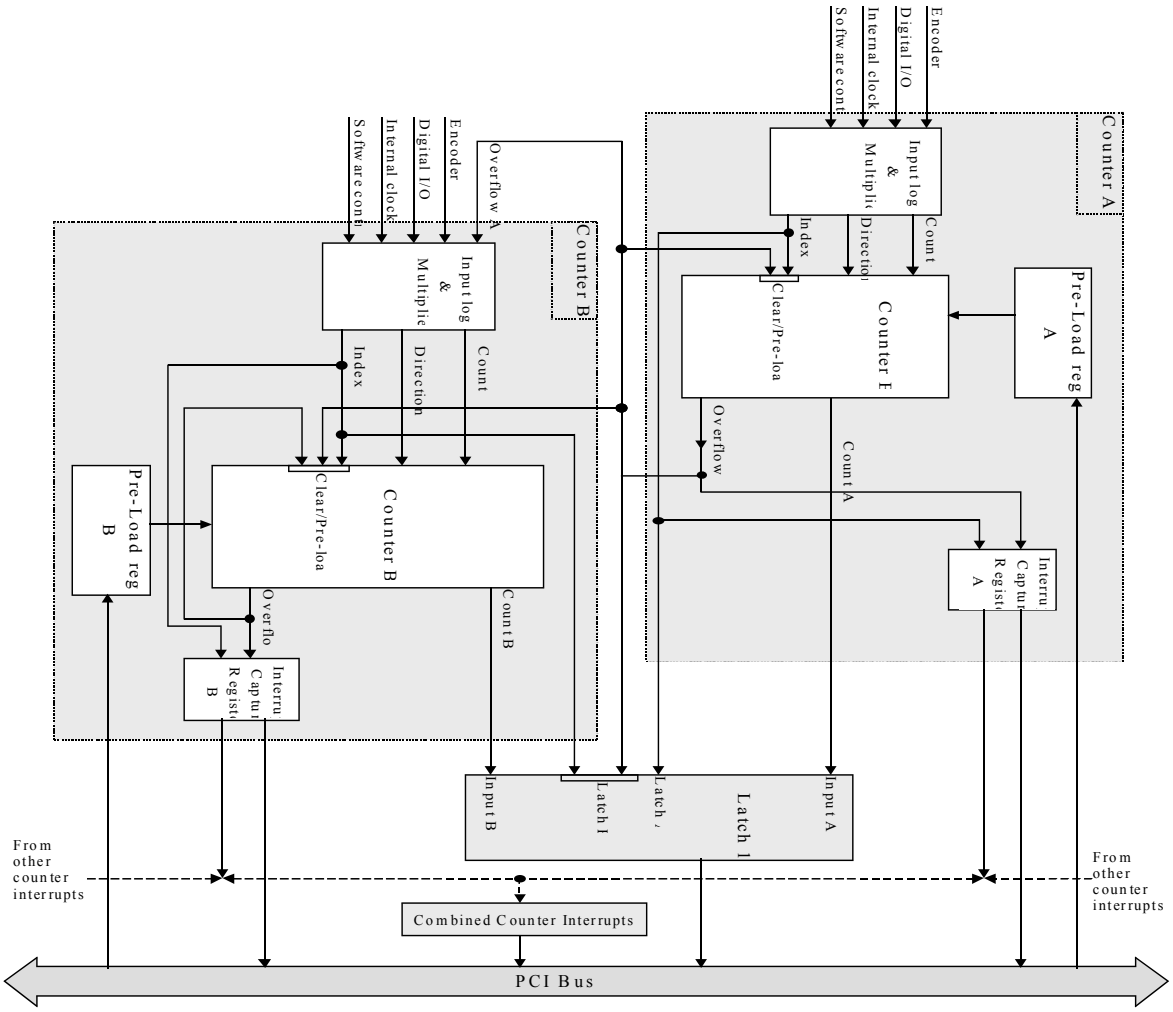

Figure 7: Block diagram of a counter pair

5.4 Programming the Counters

Table 17: CR1A 00 (hex) Read/Write

Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7
Counter B index source		Counter B source		Counter A Index edge selection	Counter A pre-load trigger source		Counter A multiplier	
00=Encoder 01=Digital inputs 10=Controlled by bit 1 of CR1B 11=Index disabled		00=Encoder 01=Digital inputs 10=Up Count with System clock 11=Down count with System clock (10 & 11 gated by bit 1 of CR1B)		0=Positive 1=Negative (Also used for software control of Counter A Index, 1=index)	00=Counter A index pulse 01=Counter A overflow 10=Disabled 11=Disabled		00=4x 01=2x 10=1x 11=Channel A is direction. Index is count pulse	

Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Counter A interrupt source		Counter A source edge selection	Counter A Index source		Counter A source	
00=None 01=Overflow only 10=Index only 11=Index & Overflow		0=Positive 1=Negative (Also software control of Counter A count pulse. 1=enable when running with the system clock)	00=Encoder 01=Digital inputs 10= Controlled by CR1A bit11 11= Index disabled		00=Encoder 01=Digital inputs 10=System clock up (gated by bit4) 11=System clock down (gated by bit4)	

Table 18: CR1B 02 (hex) Read/Write

Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
Clear Interrupt	Select Interrupt B	Select interrupt A	Counter A Enable	Counter B interrupt source		Latch source	
1=Clear Used with bit 13 or 14	1=Selected Used with bit 15	1=Selected Used with bit 15	0=Enabled 1=Only enabled if index A is high	00=None 01=Overflow only 10=Index only 11=Index & Overflow		00=Depends on latch read address 01=A's index latches A 10=B's index latches B 11=Counter A overflow captures B	

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Counter B pre-load trigger source		Clear Counter B	Counter B multiplier		Counter B Enable	Counter B index edge selection	Counter B source edge selection
00=Counter B index pulse 01=Counter B overflow 10=Counter A overflow 11=Disabled		1=Counter A overflow clears Counter B 0=Not Cleared	00=4x 01=2x 10=1x 11=Channel A is direction (Counter A overflow is count pulse)		0=Enabled 1=Only enabled if index B is high	0=Positive 1=Negative (Also software control of Counter B Index, 1=index)	0=Positive 1=Negative (Also software control of Counter B count pulse. 1=enable when running with the system clock)

	The grayed bits read back different information to that which was written to them. (See Table 19 "CR1B 02 (hex) Read" below).

Table 19: CR1B 02 (hex) Read

Bit 15	Bit 14	Bit 13
Counter B Direction 0=Down 1=Up	1=Counter A overflow routed to digital output	1=Counter B overflow routed to digital output

Counters 2 and 3 have the same register layout but with different Read/Write addresses (see Table 15 "Counter Registers" on page 18).

In the following examples groups of bits will be referred to using their name and set to a binary value. For example 'Counter B Multiplier'=10 means bit 4 =1 and bit 3=0.

'X' is used to show the value of a bit in one of the counter registers that is not important to the example being discussed. This is not to say it should be ignored. The value of these bits will depend on other functions being used and writing to any bit in the configuration example will always have an effect. Some bits have dual functions depending on the present settings of other bits.

Many of the counter examples will only deal with counter 1A. Unless otherwise stated the function being discussed can also be applied to counter 1B. The correct register & bit/s just

need to be used by looking at the applicable tables. Counters 2A, 2B, 3A and 3B will function in the same way just controlled using different register addresses.

Do not be confused by the naming of registers CR1A and CR1B. They are simply the A and B parts of the single register CR1 (Counter Register 1) and do not refer specifically to counter A and counter B.

5.5 Configuring the Counter Source

Counter A and Counter B can be driven by a differential or single-ended encoder, the digital inputs, software or from the system clock. Each counter's input logic has a count, direction and index line that may be controlled.

The 'Counter A Source' bits are used to select counter A's source (see Table 17 and 18 on page 21):

- If 'Counter A Source' is `00` then encoder inputs are selected.
- If 'Counter A Source' is `01` then the digital inputs control the counters.

Table 20: Digital inputs as Counter input controls

Digital Input Channel	Counter	Counter Input Function
0	1 A	Count or direction input
1	1 A	Count input
2	1 A	Index input
3	1 B	Count or direction input
4	1 B	Count input
5	1 B	Index input
6	2 A	Count or direction input
7	2 A	Count input
8	2 A	Index input
9	2 B	Count or direction input
10	2 B	Count input
11	2 B	Index input
12	3 A	Count or direction input
13	3 A	Count input
14	3 A	Index input
15	3 B	Count or direction input
16	3 B	Count input
17	3 B	Index input

When the digital inputs are selected as the source then the corresponding digital inputs can be used to control the counters in much the same way as when the encoder inputs are used. (See Table 20 "Digital inputs as Counter input controls" above). This mode also gives software control of the counters. This is achieved by writing to the corresponding digital output which in turn changes the digital input and thus the counter. The disadvantage to this tech-

nique is the fact that the digital output is now used for counter control and cannot be used for normal I/O functions.

In mode 00 and 10 above, the ‘Counter A Source Edge Selection’ is used to select positive ‘0’ or negative ‘1’ edge selection:

- If ‘Counter A Source’ is `10` then the counter will count down with the system clock.
- If ‘Counter A Source’ is `11` then the counter will count up with the system clock.

In both these modes the ‘Counter A Multiplier’ must be 10 and ‘Counter A Source Edge Selection’ must be 1. These two bits become part of the oscillator feedback loop. ‘Counter A Source Edge Selection’ may be used as an enable. The timer will only run when it is ’1’.

‘Counter A Enable’ is used to gate the counter under normal counting operation. When cleared, counting is enabled. when set counting is disabled. When using software to control counter A, ‘Counter A Enable’ must be ‘1’ to enable counting.

The ‘Counter A Index Source’ bits are used to select counter A’s index source:

- If ‘Counter A Index Source’ is `00` then the encoder index input will control indexing.
- If ‘Counter A Index Source’ is `01` then the digital inputs control the indexing. While in this mode, ‘Counter A Index Edge Selection’ = 0 will index on the positive edges while 1 will index on the negative edges.

5.5.1 Software Control of the Index Pulse

If ‘Counter A Index Source’ is 10 or 11 then an index is caused by toggling the ‘Counter A Index Edge Selection’ bit. When this kind of software control of the index pulse is needed it is recommended that the ‘Counter A Index Source’ be set to 10. Then setting ‘Counter A Index Edge Selection’ will set the index input to the counter and clearing it will clear the index input to the counter. If the ‘Counter A Index Source’ be set to 11 then setting ‘Counter A Index Edge Selection’ will clear the index input to the counter and clearing it will set the index input to the counter.

5.5.2 Multiplier

The ‘Counter A Multiplier’ bits are used to multiply counter A’s incoming pulses by 1, 2 or 4 or to allow software count control.

The ‘Counter B Multiplier’ bits are used to multiply counter B’s incoming pulses by 1, 2 or 4 or to allow an overflow from counter A to cause counter B to count 1.

The multiplying feature is only used when the encoder or digital inputs are used to drive the counters. If only a count and direction input is being used then the multiplier should be 1x or 2x. Channel A is the direction and channel B the count.

5.5.3 Software Control of Direction and Count Generation

This mode is only available on counter A. Direction control and a count pulse can be generated completely under software control for counter A. This mode uses the counter's index interface to generate the count pulse. As a result, while in this mode, the index cannot cause an interrupt, a latch or a pre-load of counter A. To place counter A in this mode ‘Counter A Enable’ must be 1 and ‘Counter A Multiplier’ must be 11.

When counters A and B are paired, the count direction is determined by counter A. The count direction may be controlled by the channel A encoder inputs, the channel A digital inputs or can be software controlled. To use software control for the direction, ‘Counter A Edge Selection’ must be 0. To set the count direction down ‘Counter A Source’ must be 10. To set the count direction up ‘Counter A Source’ must be 11. When using the encoder or digital inputs, channels A and B must be connected in parallel to form a single direction control.

The index channel of counter A controls counting. It can be connected using the encoder index channel, the digital input index pin or it can be software controlled. To use software control for the count pulse, it is recommended that the ‘Counter A Index Source’ be set to 10. In this case, setting ‘Counter A Index Edge Selection’ will set the index input and then clearing it will clear the index input, thus creating a count pulse. If the ‘Counter A Index Source’ is set to 11 then setting ‘Counter A Index Edge Selection’ will clear the index input and clearing it will set the index input, also creating a count.

5.6 Driving Counter B from Counter A's Overflow

This mode is only available on Counter B. Counter B can be setup so that it will count every time counter A overflows. To do this ‘Counter B Multiplier’ must be 11, which will cause the count input to counter B to come from counter A’s overflow.

Channel A of counter B will now control the count direction. It can be connected to the channel A encoder input, the channel A digital input or it can be software controlled. To use software control for the direction, ‘Counter B Source Edge Selection’ must be 0. To set the count direction to down ‘Counter B Source’ must be 11. To set the count direction to up ‘Counter B Source’ must be 10.

5.7 Triggering a Counter Load

Each counter has a pre-load register that can be loaded into the counter when triggered by various events. Each counters pre-load register is set up by simply writing the 24 bit value to the lsb and msb of the pre-load register (see Table 15 “Counter Registers" on page 18).

Pre-loading counter A can be triggered by an index pulse or overflow of counter A. Pre-loading counter B can be triggered by an index pulse or an overflow of counter A or B. A load must not be set to be triggered from counter A's index if the index is being used to run the counter under software control. Doing so will cause a load every time counter A counts (see 'Counter A & B Pre-Load Trigger Source' in Table 17 and 18 on page 21).

5.8 Clearing Counter B from Counter A's Overflow

Setting 'Clear Counter B' to 1 will allow an overflow from counter A to clear counter B. This is only available for counter B. Of course the same effect could be achieved by pre-loading the counter with 0.

5.9 Latching the Counters

Each pair of counters 1, 2 and 3 has one latch between them. (See Figure 7 "Block diagram of a counter pair" on page 20). There are two distinct sets of modes of operation for the latches.

Mode 1: 'Latch Source' = 00 allows either counter A or counter B to be captured any time. The least significant word (lsw) must be read first. This causes the entire 24 bit count to be latched allowing the most significant word (msw) to be read later without causing the count to be re-latched. The read address will decide which counter will be latched (see Table 15 "Counter Registers" on page 18 for the read addresses).

Mode 2: In the second set of modes the appropriate counter is latched by an event that would usually be programmed to also create an interrupt. This interrupt then prompts the user to read the latched count before it is overwritten by the next event. To read the count the lsw and msw of either latch A or B for the appropriate counter can be read (see Table 15 "Counter Registers" on page 18 for the read addresses). In this mode it does not matter which order the lsw & msw are read as they will have already been latched.

'Latch Source' = 01 will latch counter A when there is a counter A index.

'Latch Source' = 10 will latch counter B when there is a counter B index.

'Latch Source' = 11 will latch counter B when counter A overflows (useful when frequency counting).

5.10 Interrupts

5.10.1 Interrupts While Under Battery Backup

If there is a power failure while an external NiCad battery is connected and fully charged and the ‘Battery Enable’ bit of MISC2 was set (See Table 23 “Write to MISC2 90 (hex)” on page 31) then when power is restored the interrupts are automatically disabled. Any interrupts that occurred while the power was down will cause an interrupt the moment interrupts are re-enabled.

5.10.2 Interrupt Source Selection

If ‘Counter A (or B) Interrupt Source’ = 00 then no interrupt will occur. An index and/or an overflow can still be captured but they will not generate an interrupt. The MISC2 register will have to be polled to see if an index or overflow has occurred.

If ‘Counter A (or B) Interrupt Source’ = 01 then an interrupt will occur when the counter overflows.

If ‘Counter A (or B) Interrupt Source’ = 10 then an interrupt will occur when the counter receives an index pulse.

When ‘Counter A (or B) Interrupt Source’ = 11 then an interrupt will occur when the counter overflows or receives an index.

5.10.3 Clearing an Interrupt

There are 3 bits per counter pair that are used to clear the interrupts. . The ‘Select Interrupt A or B’ bits (bit 13 & 14) are used to select which interrupts must be cleared. A ‘1’ in either register will allow it to be cleared. Then writing a ‘1’ to the ‘Clear Interrupt’ will clear the interrupts selected with a ‘1’ in their ‘Select Interrupt A or B’ bit. If the ‘Clear Interrupt’ bit is clear it has no effect.

```
void clearInterrupt2A()
{
      // A '1' is written to interrupt A to select it and a '1' is written to the
      // interrupt command bit to clear interrupt A
      WriteReg(board, 0x06, 0b101XXXXXXXXXXXXX);
}
```

Example 2: Clearing an interrupt on Counter 2A

```
void clearInterrupt2B()
{
      // A '1' is written to interrupt B to select it and a '1' is written to the
      // interrupt command bit to clear interrupt A
      WriteReg(board, 0x06, 0b110XXXXXXXXXXXXX);
}
```

Example 3: Clearing an interrupt on Counter 2B

```
void clearInterrupt3A3B()
{
      // A '1' is written to interrupt A and B to select them and a '1' is written to the
      // interrupt command bit to clear interrupt A
      WriteReg(board, 0x0a, 0b111XXXXXXXXXXXXX);
}
```

Example 4: Clearing an interrupt on Counter 3A and 3B

5.10.4 Counter Interrupt Handling

There is only one main interrupt to the system from the MultiQ-PCI. Once this interrupt has been received it is up to the user to determine whether it originated from the counters or the digital inputs. (See “Digital Interrupt Handling” on page 13).

The ‘Counter Overflow Interrupt’ flags (COINT) and the ‘Index interrupt’ flags (INDINT) of the MISC2 register (See Table 24 “Read from MISC2 92 (hex)” on page 31) must be checked to determine which counter and what event may have caused the interrupt. An interrupt must be cleared once it has been handled to prevent it from causing multiple interrupts. (See “Clearing an Interrupt” on page 26).

5.11 Counter Configuration Examples

5.11.1 Setup Counter 1A and 1B to Count Frequency at 1B

In this example Counter 1A is set up to count down using the system clock which will cause the counter to count at 2MHz (1 count takes 500nsec). Pre-loading counter 1A with two million will have the counter count for exactly 1 second before it overflows. When it overflows it is set up to pre-load itself again, thus generating an overflow every second.

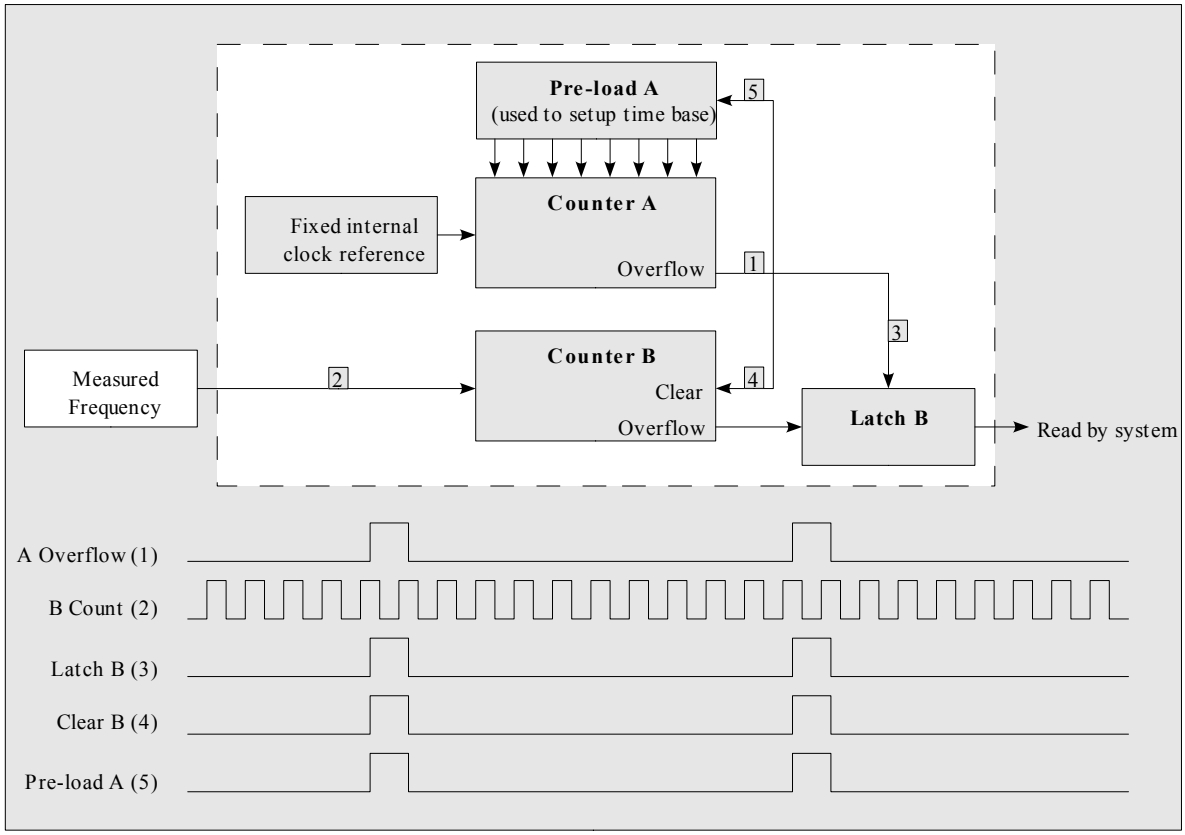

Figure 8: Frequency counter block diagram and timing

Counter 1B is set up to count the unknown frequency from an external source. The overflow from Counter 1A is set to first latch Counter 1B's count and then clear the count. This will occur every second. The value held in the latch is the number of counts that Counter 1B counted is 1 second which is a direct frequency reading (Hz - Cycles per second). This value can be read by the system at any time.

```
void setupFrequencyCounting()
{
       // Pre-load 1A with lsw of 2000000
       WriteReg(board, 0x0C, 0x8480);

       // Pre-load 1A with msw of 2000000
       WriteReg(board, 0x0D, 0x001E);

       // Setup counter's CR1A
       WriteReg(board, 0x00, 0x0373);

       // Setup counter's CR1B
       WriteReg(board, 0x02, 0xfff0);

       // The counters are now set up and running

       // To read the frequency in Hertz, tead the Latched value of counter 1B
       // as follows:

       lsw = ReadReg(board, 0x10);         // Get the lsw of the frequency
       msw = ReadReg(board, 0x12);         // Get the msw of the frequency
}
```

Example 5: Frequency counter

6. Watchdog/Miscellaneous Registers

Table 21: Write to MISC1 88 (hex)							
Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
1=Write enable	Not used	Not used	EDCAP (see pg. 10)	Not Used ISEL4 (not ISEL3)	Not Used ISEL2	Not Used ISEL1	Not Used ISEL0
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Not Used GSEL3	Not Used GSEL2	Not Used GSEL1	Not Used GSEL0	Not Used DACPOL3	Not Used DACPOL2	Not Used DACPOL1	Not Used DACPOL0

Table 22: Read from MISC1 88 (hex)							
Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
1=Write Enabled	1=Watchdog timed out	Internal Register DAC_EN	EDCAP (see pg. 10)	Internal Register ISEL4 (not ISEL3)	Internal Register ISEL2	Internal Register ISEL1	Internal Register ISEL0
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Internal Register GSEL3	Internal Register GSEL2	Internal Register GSEL1	Internal Register GSEL0	Internal Register DACPOL3	Internal Register DACPOL2	Internal Register DACPOL1	Internal Register DACPOL0

Table 23: Write to MISC2 90 (hex)							
Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
1=Clear watchdog	1=Charge Enable	Not used	Not used	Not used	Not used	0=Digital output 6	0=Digital output 5
						1=Counter 6 Overflow	1=Counter 5 Overflow
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0=Digital output 4	0=Digital output 3	0=Digital output 2	0=Digital output 1	1=Backup Battery Enabled	1=Watchdog Enabled	Watchdog period selection 00=1/8th second 01=½ second 10=1 second 11=10 seconds	
1= Counter 4 Overflow	1= Counter 3 Overflow	1= Counter 2 Overflow	1= Counter 1 Overflow				

Table 24: Read from MISC2 92 (hex)							
Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
COINT3B	COINT3A	COINT2B	COINT2A	COINT1B	COINT1A	INDINT3B	INDINT3A
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INDINT2B	INDINT2A	INDINT1B	INDINT1A	1=Backup Battery Enabled	1=Watchdog enabled	Watchdog period selection 00=1/8th second 01=½ second 10=1 second 11=10 seconds	

	The grayed bits do not read back the same registers.

6.1 Battery Charging

Bit 14 “Charge Enable” of MISC2 (92 hex) turns the trickle charging of an optional external backup battery on and off. The system will default to “Charging off” after a reset.

Before writing to any of the bits of MISC2, except bit 15 (Clear watchdog), the write enable (bit 15) of MISC1 must be set. This is to prevent accidental modification of the watchdog settings by a crashed CPU. It is a good idea to clear this bit immediately after writing to MISC2.

6.2 Watchdog Oscillator

6.2.1 Overview

Embedded systems often include a watchdog timer to regain program control following an unplanned loss of control by the PCI bus master. In such systems, the CPU is responsible for periodically clearing the watchdog timer. Should the CPU crash, the watchdog won't be cleared and will eventually overflow and cause the CPU to be reset through the PCI bus.

After a reset the watchdog timer is disabled and the watchdog oscillator selection is cleared.

To use the watchdog, it needs to be enabled and the watchdog period needs to be set.

6.2.2 Testing the Watchdog Without Resetting the Host

Under normal conditions a watchdog time-out will cause the PCI bus to be reset. By removing jumper JP1 no PCI reset will occur although the watchdog will be fully functional internally. This may be useful during development as the status of the watchdog can be obtained from bit 14 of Misc1. A ‘1’ means it has timed out. The watchdog output at JP1 is an ‘open collector’ output.

6.2.3 Watchdog Enable and Period Selection

Bits 1 and 0 of MISC2 (92 hex) are used to select a period of $^1/_{8^{th}}$, ½, 1 or 10 seconds while bit 2 is used to enable the watchdog (see Table 23 “Write to MISC2 90 (hex)” on page 31).

Before writing to any of the bits of MISC2, except bit 15 (clear watchdog), the write enable (bit 15) of MISC1 must be set. This is to prevent accidental modification of the watchdog settings by a crashed CPU. It is a good idea to clear this bit immediately after writing to MISC2.

Although the Enable and period selection bits are in the same register, first the register must be written to with the ‘Watchdog Enable’ bit = ‘1’ (bit 2) and then the register must

be written to again with bit 2 again = '1' and the period selection bits set as desired. This is because the period selection bits are held in a cleared state ($^1/_{8}^{th}$ of a second) and cannot be changed until the 'Watchdog enable' bit is = '1'.

6.2.4 Clearing the Watchdog

To prevent a PCI reset the watchdog timer must be cleared at least once within the period of the watchdog timer. To do this bit 15 ("Write Enable") of MISC1 (88 hex) must first be cleared and then bit 15 (Clear watchdog) of MISC2 (92 hex) must be set. The clear watchdog bit is automatically cleared after the watchdog has been cleared.

6.2.5 Watchdog LED Status

If the watchdog is disabled the red LED will be off continuously.

When the watchdog is enabled the LED will be off for ½ the selected watchdog period, then on for ½ the period and then the watchdog will reset the PCI bus (pulled low) if JP1 is inserted. If the jumper is not inserted then bit 14 of MISC2 will go high and the LED will flash at ½ the selected watchdog period.

6.3 Digital Outputs 0-5 Source Selection

Bits 4-9 of MISC2 (90 hex) are used to select the function of digital outputs 0-5. They can be setup either as standard digital outputs or to pull the output load low for a short pulse when the corresponding counter overflows (see Table 23 "Write to MISC2 90 (hex)" on page 31).

Before writing to any of the bits of MISC2, except bit 15 (clear watchdog), the write enable (bit 15) of MISC1 must be set. This is to prevent accidental modification of the watchdog settings by a crashed CPU. It is a good idea to clear this bit immediately after writing to MISC2.

MISC2 can not be used to read back the value that bits 4-9 were previously set to. This has to be done by reading the Counter Register B of each counter as shown below.

Table 25: Reading back Digital Outputs 0-5 Source Selection

Register	Bit	Digital I/O PIN	Function
CR1B (02 Hex)	14	0	0=Digital input, 1= Counter 1A Overflow
	13	1	0=Digital input, 1= Counter 1B Overflow
CR2B (06 Hex)	14	2	0=Digital input, 1= Counter 2A Overflow
	13	3	0=Digital input, 1= Counter 2B Overflow
CR3B (0A Hex)	14	4	0=Digital input, 1= Counter 3A Overflow
	13	5	0=Digital input, 1= Counter 3B Overflow

7. Battery Backup

A 3.6V Nickel Cadmium battery can be connected to the MultiQ-PCI using JP1. Make sure that it is connected correctly or damage will occur (see Figure 1 “MultiQ-PCI Data Acquisition Card” on page 2). It will be trickle charged at approximately 100mA if the “Charge Enable” bit of MISC2 is set (see "Battery Charging" on page 32).

While the backup battery is connected & charged the system clock can be made to run during a power failure to ensure that the counters do not loose counts. To do this the “Battery Enable” bit of MISC2 must be set to enable the backup battery. In this state the counter setup registers and the count that the counter were at when the power failed will be held and the counters will continue to count so no counts will be lost. If a counter interrupt occurs during backup (like a counter overflow) then the appropriate interrupt register will be set (counter overflow or index) but the interrupt enable is cleared when the main power to the system is restored and the system is booted. This is done to allow the user time to initialize everything. The moment the interrupts are re-enabled, the interrupt will be passed on to the system.

8. Analog Inputs

There are 16 analogue input channels (0-15). Each channel can be set to have a ±5V or ±10V range which will return a 16 bit value (including the sign) in the range ±32767.

A ‘Poll List’ is used to setup the A/D converter. This is a list containing 1-16 commands that will be executed sequentially every time the ReadADC command is used. Each command tells the converter which channel to convert (0-15) and what range to use (±5V or ±10V). The converter starts with the first command in the list and continues until it reaches a command with a ‘1’ in bit 15 of the command or until the 16th command is executed. The results are placed in the array Data[0..15]. If the same set of channel and range combinations need to be read again the user needs only to execute another ReadADC command and the old Data will be overwritten with the new. When a new poll list needs to be set up, the CloseADC command needs to be executed to close the old list and then the ResetADC needs to be executed with the details of the new poll list. This poll list approach makes reading several channels much faster (see "MultiQ-PCI Programming Guide").

Conversion takes approximately 23 µsec/channel. Once completely finished with the ADC the CloseADC command should be executed.

9. Analog Outputs

The analogue channels have a 14 bit resolution (including sign).

It takes approximately 11 µsecs to convert the digital value to an analogue output voltage.

To use the Digital to Analogue converter call the WriteDAC function and supply a channel and an output value between –8191 and +8191. This will be converted to an analogue voltage between ±10V at the specified output channel (see "MultiQ-PCI Programming Guide").

Each D/A has a sense input that should be connected to the D/A output at the end of the wire used to connect to whatever external circuit is used with the D/A. This will help keep noise down.

If this feature is not used then insert the jumpers to connect the sense input to the D/A output right on the board.

J6, J7, J8 and J9 are inserted to disable the D/A sense inputs for channels 0 -3 respectively.

Appendix A: Specifications

Table 26: General Specifications		
Interface	**Parameter**	**Description**
Bus	Type	PCI, 32-bit, 33MHz
Watchdog	Output characteristics	Open collector
	Time-out options	1/8, ½, 1 and 10 seconds
Encoders	Input characteristics	Differential pair, TTL/CMOS compatible
	Counter width	24 bits
	Number of channels	6
Digital I/O	Input characteristics	TTL/CMOS compatible
	Configuration	Each channel is programmable as an input or output
	Number of channels	48
A/D	Resolution	14 bits
	Input range	Each channel is programmable as ±5V or ±10V
	Number of channels	16
	Conversion time	Approximately (17+23*n)* µs, where *n* = # channels read
D/A	Resolution	13 bits
	Output range	±10V
	Number of channels	4
	Conversion time	Approximately 17 µs/channel
Battery	Type	3.6V Nickel Cadmium rechargeable (optional)
	Charge current	62.5 mA

Appendix B: Operating Limits

Table 27: Operating Limits					
Interface	**Parameter**	**Min.**	**Typical**	**Max.**	**Units**
Power	Operating range	4.75		5.25	V
	Operating range +12V	-11.6		15	V
	Operating range -12V	-15		-11.6	V
	Quiescent current (not charging)		450		mA
Battery	Charging current	0	50	62.5	mA
	Battery drain under backup		60		mA
	Battery drain with charger off and not in backup mode		5		mA
I/O	Output sink current			100	mA
	Captured pulse width	250			ns
Encoder	Internal timer mode			2	Mhz
	Externally driven	0		500	kHz
Watchdog	Maximum current sink			500	mA
Temperature	Operating range	0		70	C

Appendix C: Analog Connector

Table 28: J1 (50-pin IDC Ribbon Connector)					
Analog inputs and outputs					
Pin	***Function***	***User Designation***	***Pin***	***Function***	***User Designation***
1	Shield		2	GND	
3	-ADIN0		4	+ADIN0	
5	-ADIN1		6	+ADIN1	
7	-ADIN2		8	+ADIN2	
9	-ADIN3		10	+ADIN3	
11	-ADIN4		12	+ADIN4	
13	-ADIN5		14	+ADIN5	
15	-ADIN6		16	+ADIN6	
17	-ADIN7		18	+ADIN7	
19	GND		20	GND	
21	-ADIN8		22	+ADIN8	
23	-ADIN9		24	+ADIN9	
25	-ADIN10		26	+ADIN10	
27	-ADIN11		28	+ADIN11	
29	-ADIN12		30	+ADIN12	
31	-ADIN13		32	+ADIN13	
33	-ADIN14		34	+ADIN14	
35	-ADIN15		36	+ADIN15	
37	GND		38	GND	
39	GND		40	GND	
41	DAC4†		42	DAC0	
43	DAC5†		44	DAC1	
45	DAC6†		46	DAC2	
47	DAC7†		48	DAC3	
49	GND		50	Shield	

† These pins are currently sense inputs for DAC0-DAC3.

Appendix D: Digital Connectors

Table 29: J2 (50-pin IDC Ribbon Connector)		
Digital I/O 0 - 23		
Pin	***Function***	***User Designation***
1	DIO23	
3	DIO22	
5	DIO21	
7	DIO20	
9	DIO19	
11	DIO18	
13	DIO17	
15	DIO16	
17	DIO15	
19	DIO14	
21	DIO13	
23	DIO12	
25	DIO11	
27	DIO10	
29	DIO9	
31	DIO8	
33	DIO7	
35	DIO6	
37	DIO5	
39	DIO4	
41	DIO3	
43	DIO2	
45	DIO1	
47	DIO0	
49	5V	
Even pins	GND	

Table 30: J3 (50-pin IDC Ribbon Connector)		
Digital I/O 24 - 47		
Pin	***Function***	***User Designation***
1	DIO47	
3	DIO46	
5	DIO45	
7	DIO44	
9	DIO43	
11	DIO42	
13	DIO41	
15	DIO40	
17	DIO39	
19	DIO38	
21	DIO37	
23	DIO36	
25	DIO35	
27	DIO34	
29	DIO33	
31	DIO32	
33	DIO31	
35	DIO30	
37	DIO29	
39	DIO28	
41	DIO27	
43	DIO26	
45	DIO25	
47	DIO24	
49	5V	
Even pins	GND	

Appendix E: Encoder Connectors

Table 31: J4 (26-pin IDC Ribbon Connector)					
Encoders 3-5					
Pin	***Function***	***User Designation***	***Pin***	***Function***	***User Designation***
1	ENCD3 B-		2	ENCD3 B+	
3	GND		4	ENCD3 A-	
5	ENCD3 A+		6	5V	
7	ENCD3 I-		8	ENCD3 I+	
9	GND		10	ENCD4 A-	
11	ENCD4 A+		12	5V	
13	ENCD4 B-		14	ENCD4 B+	
15	GND		16	ENCD4 I-	
17	ENCD4 I+		18	5V	
19	ENCD5 B-		20	ENCD5 B+	
21	GND		22	ENCD5 A-	
23	ENCD5 A+		24	5V	
25	ENCD5 I-		26	ENCD5 I+	

Table 32: J5 (26-pin IDC Ribbon Connector)					
Encoders 0-2					
Pin	***Function***	***User Designation***	***Pin***	***Function***	***User Designation***
1	ENCD0 B-		2	ENCD0 B+	
3	GND		4	ENCD0 A-	
5	ENCD0 A+		6	5V	
7	ENCD0 I-		8	ENCD0 I+	
9	GND		10	ENCD1 A-	
11	ENCD1 A+		12	5V	
13	ENCD1 B-		14	ENCD1 B+	
15	GND		16	ENCD1 I-	
17	ENCD1 I+		18	5V	
19	ENCD2 B-		20	ENCD2 B+	
21	GND		22	ENCD2 A-	
23	ENCD2 A+		24	5V	
25	ENCD2 I-		26	ENCD2 I+	

Appendix E: Encoder Connectors

Index

Index