ARTISAN®
TECHNOLOGY GROUP

Your **definitive** source for quality pre-owned equipment.

Artisan Technology Group
(217) 352-9330 | sales@artisantg.com | artisantg.com

Full-service, independent repair center
with experienced engineers and technicians on staff.

We buy your excess, underutilized, and idle equipment
along with credit for buybacks and trade-ins.

Custom engineering
so your equipment works exactly as you specify.

- Critical and expedited services
- In stock / Ready-to-ship
- Leasing / Rentals / Demos
- ITAR-certified secure asset solutions

Expert team | Trust guarantee | 100% satisfaction

All trademarks, brand names, and brands appearing herein are the property of their respective owners.

Find the *RTD Embedded Technologies DM35838HR* at our website: *Click HERE*

DM35838HR

PCI Express Isolated Digital I/O Board

User's Manual

BDM-610010054 Rev B



RTD Embedded Technologies, Inc.
AS9100 and ISO 9001 Certified

RTD Embedded Technologies, Inc.
103 Innovation Boulevard
State College, PA 16803 USA
Telephone: 814-234-8087
Fax: 814-234-5218

www.rtd.com

sales@rtd.com
techsupport@rtd.com

Revision History

Rev A	Initial Release
Rev B	Section 2.1: Added MTBF

Advanced Analog I/O, Advanced Digital I/O, aAIO, aDIO, a2DIO, Autonomous SmartCal, "Catch the Express", cpuModule, dspFramework, dspModule, expressMate, ExpressPlatform, HiDANplus, "MIL Value for COTS prices", multiPort, PlatformBus, and PC/104EZ are trademarks, and "Accessing the Analog World", dataModule, IDAN, HiDAN, RTD, and the RTD logo are registered trademarks of RTD Embedded Technologies, Inc (formerly Real Time Devices, Inc.). PS/2 is a trademark of International Business Machines Inc. PCI, PCI Express, and PCIe are trademarks of PCI-SIG. PC/104, PC/104-Plus, PCI-104, PCIe/104, PCI/104-Express and 104 are trademarks of the PC/104 Embedded Consortium. All other trademarks appearing in this document are the property of their respective owners.

Failure to follow the instructions found in this manual may result in damage to the product described in this manual, or other components of the system. The procedure set forth in this manual shall only be performed by persons qualified to service electronic equipment. Contents and specifications within this manual are given without warranty, and are subject to change without notice. RTD Embedded Technologies, Inc. shall not be liable for errors or omissions in this manual, or for any loss, damage, or injury in connection with the use of this manual.

Copyright © 2020 by RTD Embedded Technologies, Inc. All rights reserved.

Table of Contents

1 Introduction **9**

1.1 Product Overview 9

1.2 Board Features 9

1.3 Ordering Information 10

1.4 Contact Information 10
1.4.1 Sales Support 10
1.4.2 Technical Support 10

2 Specifications **11**

2.1 Operating Conditions 11

2.2 Electrical Characteristics 11

2.1 Functional Characteristics 12

3 Board Connection **13**

3.1 Board Handling Precautions 13

3.2 Physical Characteristics 13

3.3 Connectors and Jumpers 14
3.3.1 Bus Connectors 14
CN1(Top) & CN2(Bottom): PCIe Connector 14
3.3.2 External I/O Connectors 15
CN3 & CN4: Isolated Digital Input Connector 15
CN5: Isolated Output 16
Other Connectors 16
3.3.3 Jumpers 17
User ID Jumper 17

3.4 Steps for Installing 18

4 IDAN Connections **19**

4.1 Module Handling Precautions 19

4.2 Physical Characteristics 19

4.3 Connectors 20
4.3.1 External I/O Connectors 20
P2 & P3: Isolated Input Connector 20
P4: Isolated Output Connector 21
4.3.2 Bus Connectors 22
CN1(Top) & CN2(Bottom): PCIe Connector 22

4.4 Steps for Installing 22

5 Functional Description **24**

5.1 Block Diagram 24

5.2 PCIe Switch 24

5.3 FPGA with DMA Engine 24

5.4 Opto-Isolators 24

5.5 Advanced Digital Inputs 25
DMA 25
Advanced Interrupts 25

5.6 Advanced Digital Outputs 25
DMA 25
Peripheral Select 25

5.7 Quad Pulse Width Modulator 25

5.8 Incremental Encoder 26
5.9 External Clocking 27
5.10 Programmable Clock 27

6 Register Address Space **28**
Register Types 28
Clock Source 28
Function Block Mapping 29

6.1 BAR0: General Board Control 30
6.1.1 GBC_FMT (Read-Only) 30
6.1.2 GBC_REV (Read-Only) 30
6.1.3 GBC_BRD_RST (Read/Write) 30
6.1.4 GBC_PDP (Read-Only) 30
6.1.5 GBC_BUILD (Read-Only) 30
6.1.6 GBC_SYS_CLK_FREQ (Read Only) 31
6.1.7 GBC_USER_ID (Read Only) 31
6.1.8 GBC_IRQ_STATUS (Read/Clear) 31
6.1.9 GBC_DIRQ_STATUS (Read/Clear) 31
6.1.10 GBC_EOI (Read/Clear) 31
6.1.11 FBn_ID (Read-Only) 31
6.1.12 FBn_Offset (Read-Only) 31
6.1.13 FBn_Offset_DMA (Read-Only) 32

6.2 BAR2: Functional Block Standard DMA 33
6.2.1 FB_DMAm_Action (Read/Write) 33
6.2.2 FB_DMAm_Last_Action (Read/Write) 33
6.2.3 FB_DMAm_Setup (Read/Write) 34
6.2.4 FB_DMAm_Stat_Used (Read/Write) 34
6.2.5 FB_DMAm_Stat_Invalid (Read/Write) 34
6.2.6 FB_DMAm_Stat_Overflow (Read/Write) 34
6.2.7 FB_DMAm_Stat_Underflow (Read/Write) 34
6.2.8 FB_DMAm_Stat_Complete (Read/Write) 34
6.2.9 FB_DMAm_Current_Buffer (Read-Only) 34
6.2.10 FB_DMAm_COUNT (Read-Only) 34
6.2.11 FB_DMAm_RD_FIFO_CNT(Read-Only) 34
6.2.12 FB_DMAm_WR_FIFO_CNT(Read-Only) 35
6.2.13 FB_DMAm_ADDRESSn (Read/Write) 35
6.2.14 FB_DMAm_SIZEn (Read/Write) 35
6.2.15 FB_DMAm_CTRLn (Read/Write) 35
6.2.16 FB_DMAm_STATn (Read/Clear) 35

6.3 BAR2: Functional Block Header Registers 36
6.3.1 FB_ID (Read-Only) 36
6.3.2 FB_DMA_CHANNELS (Read -Only) 36
6.3.3 FB_DMA_BUFFERS (Read-Only) 36

6.4 BAR2: Digital Input Functional Block 37
6.4.1 FB_CONTROL_SIZE (Read -Only) 38
6.4.2 FB_CHANNEL_SIZE (Read-Only) 38
6.4.3 Mode_Status (Read/Write, Read-Only) 38
6.4.4 CLK_SRC (Read/Write) 38
6.4.5 START_TRIG (Read/Write) 38
6.4.6 STOP_TRIG (Read/Write) 39
6.4.7 CLK_DIV (Read/Write) 39
6.4.8 CLK_DIV_CNTR (Read Only) 39
6.4.9 PRE_START_COUNT (Read/Write) 39
6.4.10 POST_STOP_COUNT (Read/Write) 39
6.4.11 SAMPLE_CNT (Read Only) 39
6.4.12 INT_ENA (Maskable Read/Write) 39
6.4.13 INT_STAT (Read/Clear) 39
6.4.14 CLK_BUSn 39
6.4.15 ADV_INT_MODE (Read/Write) 40
6.4.16 ADV_INT_MASK (Read/Write) 40
6.4.17 ADV_INT_COMP (Read/Write) 40
6.4.18 ADV_INT_CAPT (Read/Write) 40
6.4.19 MODE_CONFIG (MASKABLE READ/WRITE) 41

6.4.20 CHn_FIFO_DATA_CNT (Read) 41
6.4.21 CHn_MAX_FIFO_SIZE (Read) 41
6.4.22 DIG_IN (Read Only) 41
6.4.23 CHn_FIFO_ACCESS (Read/Write) 41

6.5 BAR2: Digital Out Functional Block 42
6.5.1 FB_CONTROL_SIZE (Read -Only) 43
6.5.2 FB_CHANNEL_SIZE (Read-Only) 43
6.5.3 Mode_Status (Read/Write, Read-Only) 43
6.5.4 CLK_SRC (Read/Write) 43
6.5.5 START_TRIG (Read/Write) 43
6.5.6 STOP_TRIG (Read/Write) 44
6.5.7 CLK_DIV (Read/Write) 44
6.5.8 CLK_DIV_CNTR (Read Only) 44
6.5.9 POST_STOP_COUNT (Read/Write) 44
6.5.10 CONVERSION_CNT (Read Only) 44
6.5.11 INT_ENA (Maskable Read/Write) 44
6.5.12 INT_STAT (Read/Clear) 44
6.5.13 CLK_BUSn 44
6.5.14 PERIPH_SEL_BIT_n (Read/Write) 45
6.5.15 CHn_FIFO_DATA_CNT (Read) 45
6.5.16 CHn_MAX_FIFO_SIZE (Read) 45
6.5.17 DIG_OUT (Read/Write) 45
6.5.18 CHn_FIFO_ACCESS (Read/Write) 45

6.6 BAR2: Quad PWM Functional Block 46
6.6.1 FB_CONTROL_SIZE (Read -Only) 47
6.6.2 FB_CHANNEL_SIZE (Read-Only) 47
6.6.3 Mode_Status (Read/Write, Read-Only) 47
6.6.4 CLK_SRC (Read/Write) 47
6.6.5 START_TRIG (Read/Write) 47
6.6.6 STOP_TRIG (Read/Write) 47
6.6.7 CLK_DIV (Read/Write) 48
6.6.8 CLK_DIV_CNTR (Read Only) 48
6.6.9 PERIOD_COUNT (Read Only) 48
6.6.10 INT_ENA (Maskable Read/Write) 48
6.6.11 INT_STAT (Read/Clear) 48
6.6.12 CLK_BUSn 48
6.6.13 CHn_FIFO_DATA_CNT (Read) 48
6.6.14 CHn_MAX_FIFO_SIZE (Read) 48
6.6.15 CHn_PWM_WIDTH (Read/Write) 49
6.6.16 CHn_FIFO_ACCESS (Read/Write) 49

6.7 BAR2: Incremental Encoder Functional Block 50
6.7.1 FB_CONTROL_SIZE (Read -Only) 50
6.7.2 FB_CHANNEL_SIZE (Read-Only) 50
6.7.3 Mode_Status (Read/Write, Read-Only) 50
6.7.4 CLK_SRC (Read/Write) 51
6.7.5 START_TRIG (Read/Write) 51
6.7.6 STOP_TRIG (Read/Write) 51
6.7.7 CLK_DIV (Read/Write) 51
6.7.8 CLK_DIV_CNTR (Read Only) 51
6.7.9 PRE_START_COUNT (Read/Write) 51
6.7.10 POST_STOP_COUNT (Read/Write) 52
6.7.11 SAMPLE_CNT (Read Only) 52
6.7.12 INT_ENA (Maskable Read/Write) 52
6.7.13 INT_STAT (Read/Clear) 52
6.7.14 CLK_BUSn 52
6.7.15 MODE_CONFIG (MASKABLE READ/WRITE) 52
6.7.16 FIFO_DATA_CNT (Read) 53
6.7.17 MAX_FIFO_SIZE (Read) 53
6.7.18 THRESH_LOW (Read/Write) 53
6.7.19 THRESH_HIGH (Read/Write) 53
6.7.20 INCENC_VALUE (Read/Write) 53
6.7.21 FIFO_ACCESS (Read) 53

6.8 BAR2: External Clocking Functional Block 54

6.8.1 EXT_CLK_IN (Read-Only) 54
6.8.2 EXT_CLK_GATE_IN (Read Only) 54
6.8.3 EXT_CLK_DIR (Read/Write) 54
6.8.4 EXT_CLK_EDGE (Read/Write) 55
6.8.5 EXT_CLKn_PW (Read/Write) 55
6.8.6 EXT_CLKn_CFG (Read/Write) 55
6.9 Programmable Clock Functional Block 55
6.9.1 FB_CONTROL_SIZE (Read -Only) 56
6.9.2 FB_CHANNEL_SIZE (Read-Only) 56
6.9.3 Mode_Status (Read/Write, Read-Only) 56
6.9.4 CLK_SRC (Read/Write) 56
6.9.5 START_TRIG (Read/Write) 56
6.9.6 STOP_TRIG (Read/Write) 57
6.9.7 CLK_DIV (Read/Write) 57
6.9.8 CLK_DIV_CNTR (Read Only) 57
6.9.9 COUNT (Read Only) 57
6.9.10 INT_ENA (Maskable Read/Write) 57
6.9.11 INT_STAT (Read/Clear) 57
6.9.12 CLK_BUSn 57
6.9.13 MODE_CONFIG (MASKABLE READ/WRITE) 58
6.9.14 CHn_FIFO_DATA_CNT (Read) 58
6.9.15 CHn_MAX_FIFO_SIZE (Read) 58
6.9.16 INTERVAL_CLK_DIV (Read/Write) 58
6.9.17 LAST_INTERVAL_COUNT (Read-Only) 58
6.9.18 CHn_FIFO_ACCESS (Read/Write) 58
7 Troubleshooting 59
8 Additional Information 60
8.1 PC/104 Specifications 60
8.2 PCI and PCI Express Specification 60
9 Limited Warranty 61

Table of Figures

Figure 5: Board Dimensions 13
Figure 2: Board Connections 14
Figure 3: Isolated Input Circuit 16
Figure 4: Isolated Output Circuit 16
Figure 5: Example 104™ Stack 18
Figure 6: IDAN Dimensions 19
Figure 7: Example IDAN System 23
Figure 8: DM35838HR Block Diagram 24
Figure 9: Incremental Encoder Signals 26

Table of Tables

Table 1: Ordering Options 10
Table 2: Operating Conditions 11
Table 3: Electrical Characteristics 11
Table 4: Functional Characteristics 12
Table 5: CN3 Isolated Input 15
Table 6: CN4 Isolated Input 15
Table 7: CN5 Isolated Output 16
Table 8: JP8 – User ID Jumper 17
Table 9: P2 and P3 Pin Assignments 20
Table 10: P4 Pin Assignments 21
Table 11: Incremental Encoder Inputs 26
Table 12: Function Block Mapping 29
Table 13: Base Functional Block 30
Table 14: DMA Registers 33
Table 15: Functional Block Header 36
Table 16: Digital Input Functional Block 37
Table 17: Digital Out Functional Block 42
Table 18:PWM Functional Block 46
Table 19: Incremental Encoder Functional Block 50
Table 20:External Clocking Functional Block 54
Table 21: Programmable Clock Functional Block 55

1 Introduction

1.1 Product Overview

The DM35838HR is a PC/104 isolated digital I/O module designed for applications requiring isolation between the computer and the external signals it is monitoring and controlling. This module provides 48 opto-isolated digital inputs and 16 opto-isolated outputs to interface with a variety of devices. The inputs allow for a broad range of input voltages, and are compatible with 12V and 24V systems. The outputs provide the ability to drive LEDs and relays. The DM35838HR also provides many enhanced digital I/O features such as pulse width modulators (PWM), incremental encoders, programmable clocks, and advanced interrupts.

1.2 Board Features

- PCI Express Bus:
  - PCIe/104 Universal Board
    - Interfaces with Type 1 or Type 2 bus
    - Lane repopulation allows maximum system expansion
  - Provides 2.5 Gbps in each direction
  - In-band interrupts and messages
  - Message Signaled Interrupt (MSI) support
- Isolated inputs:
  - 48 Opto-isolated digital input
  - Two 50 position 0.1" DIL header
  - Two Input Ranges Available
    - +Standard version:
      - Guaranteed On at 4V
      - Maximum input 15V
    - +24V version:
      - Guaranteed On at 18V
      - Maximum input 32V
  - Custom settings available
  - Reserve-polarity protection
  - 3000 $VAC_{(RMS)}$ isolation
  - Advance Interrupt
    - Interrupt on Match, Change or Strobe
  - 4K-Word FIFOs with DMA
  - Maximum input frequency: 10Khz
- Isolated Outputs
  - 16 Opto-isolated open-collector digital outputs
  - One 50 position 0.1" DIL header
  - 4.75K ohm pull-up resistor
  - Maximum output current: 16mA
  - Maximum output voltage: 80V
  - 3000 $VAC_{(RMS)}$ isolation
- Incremental Encoders
  - Four Incremental Encoder channels
  - Single-ended or Pseudo-differential inputs
  - Variable frequency input filtering
  - 32-bit resolution
  - FIFO with DMA channel available for position sampling
- Pulse Width Modulators
  - 2 – Quad PWM outputs
  - Single-ended or Differential outputs
  - 32-bit resolution
  - FIFO with DMA channel
- External Clocking
  - Provides 6 external clock that can be configured as inputs or outputs
  - Can be used as external trigger
  - External gate provided for each clock input
- Programmable Clock
  - Four programmable clocks
  - Maximum frequency of 50MHz

- Can be started or stopped by an interrupt or another clock
- Continuous or One-Shot Operation
- Can be cascaded
- Event Counter
- Interval Counter based on a 50Mhz Clock
  - Minimal interval: 20ns
  - Maximum interval: 85 seconds
  - FIFO with DMA channel

1.3 Ordering Information

The DM35838HR is available with the following options:

Table 1: Ordering Options

Part Number	Description
DM35838HR-12V	PCIe/104 High Density Isolated Digital I/O Module for 12V systems
DM35838HR-24V	PCIe/104 High Density Isolated Digital I/O Module for 24V systems
IDAN-DM35838HR-D	PCIe/104 High Density Isolated Digital I/O Module in IDAN enclosure for 12V systems
IDAN-DM35838HR-24V-D	PCIe/104 High Density Isolated Digital I/O Module in IDAN enclosure for 24V systems

The Intelligent Data Acquisition Node (IDAN®) building block can be used in just about any combination with other IDAN building blocks to create a simple but rugged 104™ stack. This module can also be incorporated in a custom-built RTD HiDAN® or HiDANplus® High Reliability Intelligent Data Acquisition Node. Contact RTD sales for more information on our high reliability systems.

1.4 Contact Information

1.4.1 Sales Support

For sales inquiries, you can contact RTD Embedded Technologies sales via the following methods:

Phone: 1-814-234-8087 Monday through Friday, 8:00am to 5:00pm (EST).
E-Mail: sales@rtd.com

1.4.2 Technical Support

If you are having problems with your system, please try the steps in the Troubleshooting section of this manual on page 59.

For help with this product, or any other product made by RTD, you can contact RTD Embedded Technologies technical support via the following methods:

Phone: 1-814-234-8087 Monday through Friday, 8:00am to 5:00pm (EST).
E-Mail: techsupport@rtd.com

2 Specifications

2.1 Operating Conditions

Table 2: Operating Conditions

Symbol	Parameter	Test Condition	Min	Max	Unit
V_{cc5}	5V Supply Voltage		4.75	5.25	V
V_{cc3}	3.3V Supply Voltage		n/a	n/a	V
V_{cc12}	12V Supply Voltage		n/a	n/a	V
V_{cc-12}	-12V Supply Voltage		n/a	n/a	V
T_a	Operating Temperature		-40	+85	C
T_s	Storage Temperature		-55	+125	C
RH	Relative Humidity	Non-Condensing	0	90%	%
MTBF	Mean Time Before Failure	Telcordia Issue 2 30°C, Ground benign, controlled		763,215	Hours

2.2 Electrical Characteristics

Table 3: Electrical Characteristics

Symbol	Parameter	Test Condition	Min	Typical	Max	Unit
P	Power Consumption	V_{cc5} = 5.0V Active (All Outputs Open) Active (All Outputs Closed)		 2 3		W
I_{cc5}	5V Input Supply Current	Active (All Outputs Open) Active (All Outputs Closed)		0.4 0.6		A
PCIe/104 Bus						
	Differential Output Voltage		0.8		1.2	V
	DC Differential TX Impedance		80		120	Ω
	Differential Input Voltage		0.175		1.2	V
	DC Differential RX Impedance		80		120	Ω
	Electrical Idle Detect Threshold		65		175	mV
Isolated Inputs						
	Input Voltage	12V 24V	-15 -32		15 32	V_{DC}
	Guaranteed Turn-On Voltage	12V 24V	4 16.4	 18		V_{DC}
	Turn-On Threshold Current				2.5	mA
	Isolation Voltage				3000	$V_{AC(RMS)}$
	Max Frequency				10	KHz
Isolated Outputs						
V_{CE}	Collector-Emitter Voltage		0		80	V_{DC}
V_{EC}	Emitter-Collector Voltage				7	V_{DC}
$V_{CE(SAT)}$	Collector-Emitter Saturation Voltage				0.4	V_{DC}
	Max Pull-up Voltage	Collector Voltage = 0V			22	V_{DC}
	Output Current				20	mA
	Isolation Voltage				3000	$V_{AC(RMS)}$
	Max Frequency				10	KHz

2.1 Functional Characteristics

Table 4: Functional Characteristics

Symbol	Parameter	Value	Unit
GBC_SYS_CLK_FREQ	System Clock Frequency	50	MHz
	Digital Input FIFO Size	4095	D-Words
	Digital Output FIFO Size	4095	D-Words
	PWM FIFO Size	511	D-Words
	Incremental Encoders	511	D-Words
	Programmable Clock	511	D-Words

3 Board Connection

3.1 Board Handling Precautions

To prevent damage due to Electrostatic Discharge (ESD), keep your board in its antistatic bag until you are ready to install it into your system. When removing it from the bag, hold the board at the edges, and do not touch the components or connectors. Handle the board in an antistatic environment, and use a grounded workbench for testing and handling of your hardware.

3.2 Physical Characteristics

STEP model is available upon request; contact RTD Tech Support for more information.

- Weight: Approximately 99.8 g (0.22 lbs) with heatsink
- Dimensions: 90.17 mm L x 95.89 mm W (3.550 in L x 3.775 in W)

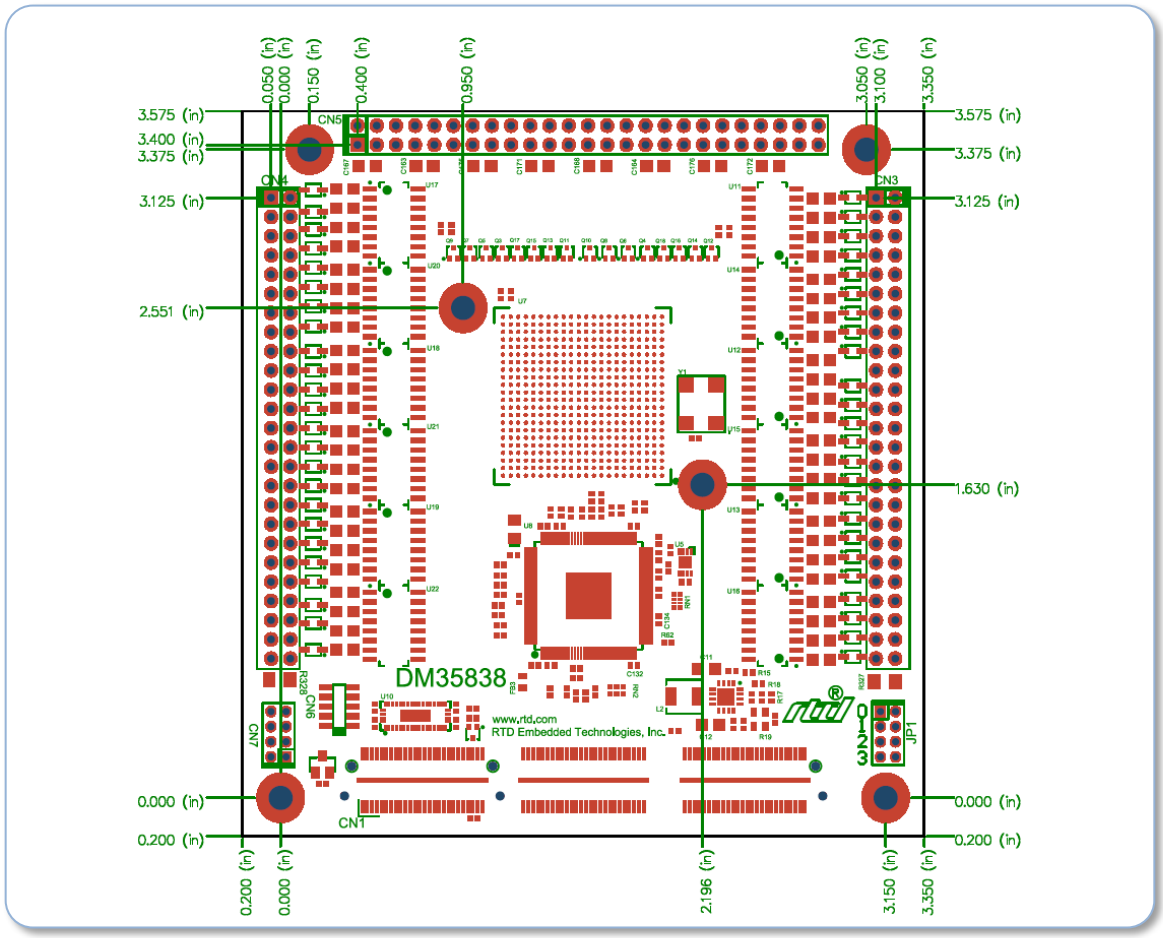


Figure 1: Board Dimensions

3.3 Connectors and Jumpers

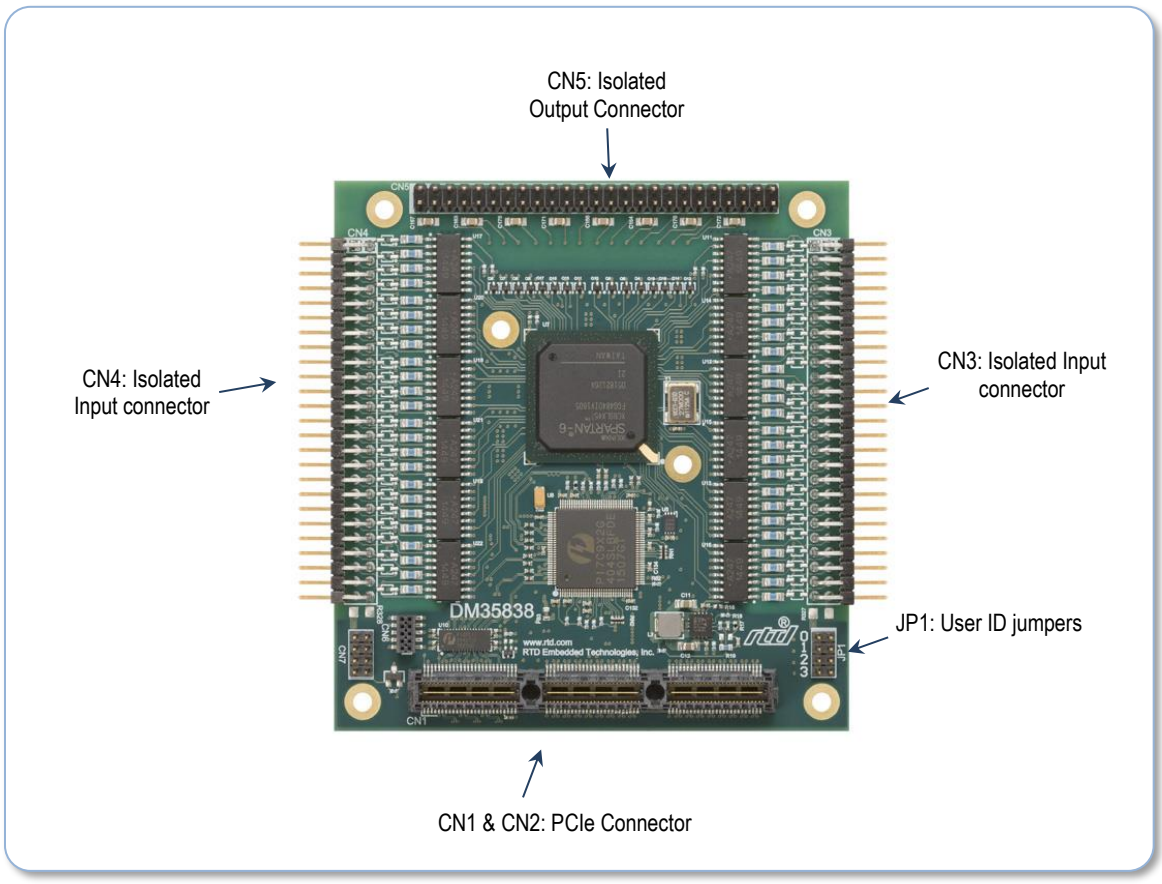


Figure 2: Board Connections

3.3.1 Bus Connectors

CN1(Top) & CN2(Bottom): PCIe Connector

The PCIe connector is the connection to the system CPU. The position and pin assignments are compliant with the PCI/104-Express Specification. (See PC/104 Specifications on page 60)

The DM35838HR is a “Universal” board, and can connect to either a Type 1 or Type 2 PCIe/104 connector.

3.3.2 EXTERNAL I/O CONNECTORS

CN3 & CN4: Isolated Digital Input Connector

Connector Part #: SAMTEC TSW-125-08-S-D-RA **Mating Connector:** FCI 65043-012 LF

The Isolated Inputs are on CN3 and C4N4. The pin assignments are shown in the Table below. The Figure below shows the input circuit.

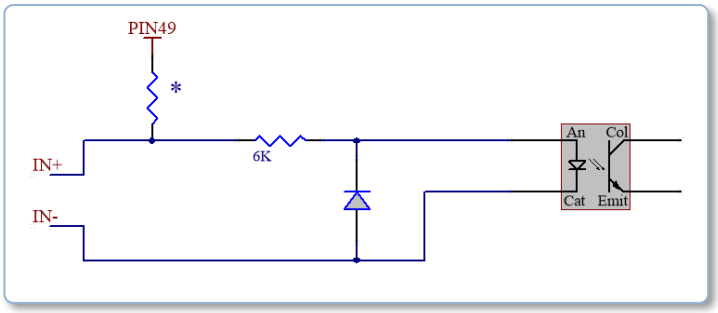
Pin 49	Pin 47	Pin 45	Pin 43	Pin 41	Pin 39	Pin 37	Pin 35	Pin 33	Pin 31	Pin 29	Pin 27	Pin 25	Pin 23	Pin 21	Pin 19	Pin 17	Pin 15	Pin 13	Pin 11	Pin 9	Pin 7	Pin 5	Pin 3	**Pin 1**
Pin 50	Pin 48	Pin 46	Pin 44	Pin 42	Pin 40	Pin 38	Pin 36	Pin 34	Pin 32	Pin 30	Pin 28	Pin 26	Pin 24	Pin 22	Pin 20	Pin 18	Pin 16	Pin 14	Pin 12	Pin 10	Pin 8	Pin 6	Pin 4	Pin 2

Table 5: CN3 Isolated Input

Pin Name	#	#	Pin Name
DI0.0-	2	1	DI0.0+
DI0.1-	4	3	DI0.1+
DI0.2-	6	5	DI0.2+
DI0.3-	8	7	DI0.3+
DI0.4-	10	9	DI0.4+
DI0.5-	12	11	DI0.5+
DI0.6-	14	13	DI0.6+
DI0.7-	16	15	DI0.7+
DI0.8-	18	17	DI0.8+
DI0.9-	20	19	DI0.9+
DI0.10-	22	21	DI0.10+
DI0.11-	24	23	DI0.11+
DI0.12-	26	25	DI0.12+
DI0.13-	28	27	DI0.13+
DI0.14-	30	29	DI0.14+
DI0.15-	32	31	DI0.15+
DI0.16-	34	33	DI0.16+
DI0.17-	36	35	DI0.17+
DI0.18-	38	37	DI0.18+
DI0.19-	40	39	DI0.19+
DI0.20-	42	41	DI0.20+
DI0.21-	44	43	DI0.21+
DI0.22-	46	45	DI0.22+
DI0.23-	48	47	DI0.23+
n.c.	50	49	n.c.[1]

Table 6: CN4 Isolated Input

Pin Name	#	#	Pin Name
DI1.0-	2	1	DI1.0+
DI1.1-	4	3	DI1.1+
DI1.2-	6	5	DI1.2+
DI1.3-	8	7	DI1.3+
DI1.4-	10	9	DI1.4+
DI1.5-	12	11	DI1.5+
DI1.6-	14	13	DI1.6+
DI1.7-	16	15	DI1.7+
DI1.8-	18	17	DI1.8+
DI1.9-	20	19	DI1.9+
DI1.10-	22	21	DI1.10+
DI1.11-	24	23	DI1.11+
DI1.12-	26	25	DI1.12+
DI1.13-	28	27	DI1.13+
DI1.14-	30	29	DI1.14+
DI1.15-	32	31	DI1.15+
DI1.16-	34	33	DI1.16+
DI1.17-	36	35	DI1.17+
DI1.18-	38	37	DI1.18+
DI1.19-	40	39	DI1.19+
DI1.20-	42	41	DI1.20+
DI1.21-	44	43	DI1.21+
DI1.22-	46	45	DI1.22+
DI1.23-	48	47	DI1.23+
n.c.	50	49	n.c.[2]



[1] Optional pull-up resistor can be added to the IN+ of each channel. This pull-up will allow user to apply a voltage to pin 49 of CN3 to configure the inputs to operate with switches. Note, this will negate channel to channel isolation of any channel that have pulls up. Contact RTD Sales for more information.
[2] Optional pull-up resistor can be added to the IN+ of each channel. This pull-up will allow user to apply a voltage to pin 49 of CN4 to configure the inputs to operate with switches. Note, this will negate channel to channel isolation of any channel that have pulls up. Contact RTD Sales for more information.

Figure 3: Isolated Input Circuit

CN5: Isolated Output

Connector Part #: SAMTEC TSW-125-16-S-D **Mating Connector:** FCI 65043-012 LF

The Isolated Outputs are on CN5. The pin assignments are shown in the Table below. The Figure below shows the output circuit.

Pin 49	Pin 47	Pin 45	Pin 43	Pin 41	Pin 39	Pin 37	Pin 35	Pin 33	Pin 31	Pin 29	Pin 27	Pin 25	Pin 23	Pin 21	Pin 19	Pin 17	Pin 15	Pin 13	Pin 11	Pin 9	Pin 7	Pin 5	Pin 3	**Pin 1**
Pin 50	Pin 48	Pin 46	Pin 44	Pin 42	Pin 40	Pin 38	Pin 36	Pin 34	Pin 32	Pin 30	Pin 28	Pin 26	Pin 24	Pin 22	Pin 20	Pin 18	Pin 16	Pin 14	Pin 12	Pin 10	Pin 8	Pin 6	Pin 4	Pin 2

Table 7: CN5 Isolated Output

Pin Name	#	#	Pin Name
Collector 0	2	1	Pull-up 0
Pull-up 1	4	3	Emitter0
Emitter 1	6	5	Collector 1
Collector 2	8	7	Pull-up 2
Pull-up 3	10	9	Emitter 2
Emitter 3	12	11	Collector 3
Collector 4	14	13	Pull-up 4
Pull-up 5	16	15	Emitter 4
Emitter 5	18	17	Collector 5
Collector 6	20	19	Pull-up 6
Pull-up 7	22	21	Emitter 6
Emitter 7	24	23	Collector 7
Collector 8	26	25	Pull-up 8
Pull-up 9	28	27	Emitter 8
Emitter 9	30	29	Collector 9
Collector 10	32	31	Pull-up 10
Pull-up 11	34	33	Emitter 10
Emitter 11	36	35	Collector 11
Collector 12	38	37	Pull-up 12
Pull-up 13	40	39	Emitter 12
Emitter 13	42	41	Collector 13
Collector 14	44	43	Pull-up 14
Pull-up 15	46	45	Emitter 14
Emitter 15	48	47	Collector 15
n.c.	50	49	n.c.

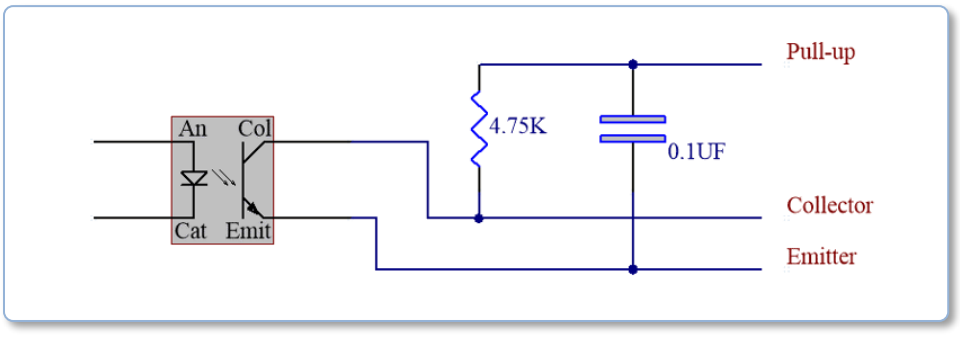


Figure 4: Isolated Output Circuit

Other Connectors

CN6 and CN7 are for factory use only

3.3.3 JUMPERS

User ID Jumper

The User ID Jumper is a four position, user defined jumper block. It can be used to uniquely identify multiple boards in the system. The jumper settings can be read from the USER_ID register. An installed jumper results in a logic high, and an open jumper results in a logic low.

Table 8: JP8 – User ID Jumper

Position	Description
1-2	User ID bit 0
3-4	User ID bit 1
5-6	User ID bit 2
7-8	User ID bit 3

3.4 Steps for Installing

1. Always work at an ESD protected workstation, and wear a grounded wrist-strap.
2. Turn off power to the PC/104 system or stack.
3. Select and install stand-offs to properly position the module on the stack.
4. Remove the module from its anti-static bag.
5. Check that pins of the bus connector are properly positioned.
6. Check the stacking order; make sure all of the busses used by the peripheral cards are connected to the cpuModule.
7. Hold the module by its edges and orient it so the bus connector pins line up with the matching connector on the stack.
8. Gently and evenly press the module onto the PC/104 stack.
9. If any boards are to be stacked above this module, install them.
10. Attach any necessary cables to the PC/104 stack.
11. Re-connect the power cord and apply power to the stack.
12. Boot the system and verify that all of the hardware is working properly.

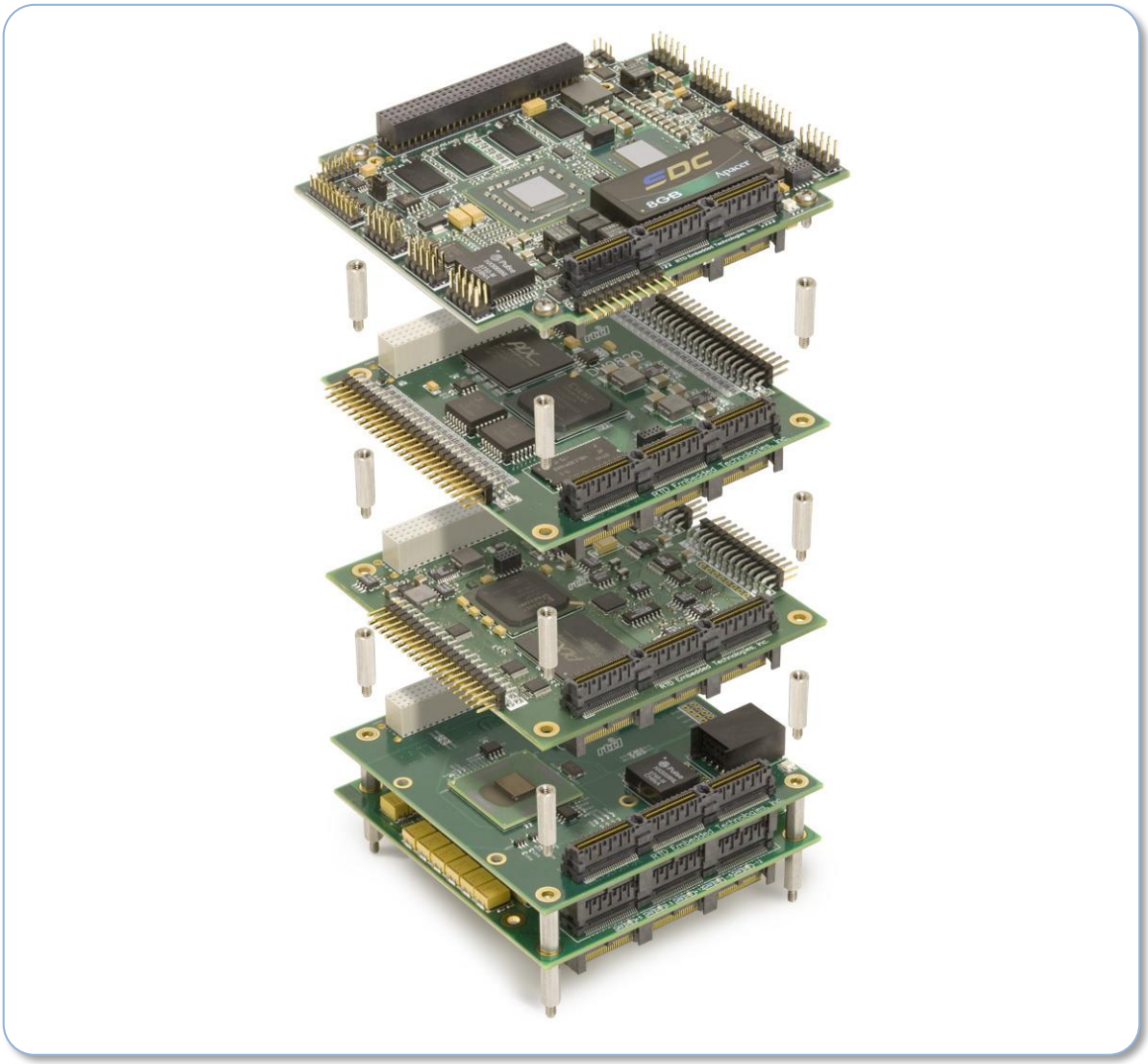


Figure 5: Example 104™ Stack

4 IDAN Connections

4.1 Module Handling Precautions

To prevent damage due to Electrostatic Discharge (ESD), keep your module in its antistatic bag until you are ready to install it into your system. When removing it from the bag, hold the module by the aluminum enclosure, and do not touch the components or connectors. Handle the module in an antistatic environment, and use a grounded workbench for testing and handling of your hardware.

4.2 Physical Characteristics

- Weight: Approximately 0.42 Kg (0.92 lbs.)
- Dimensions: 152mm L x 130mm W x 34mm H (5.983" L x 5.117" W x 1.339" H)

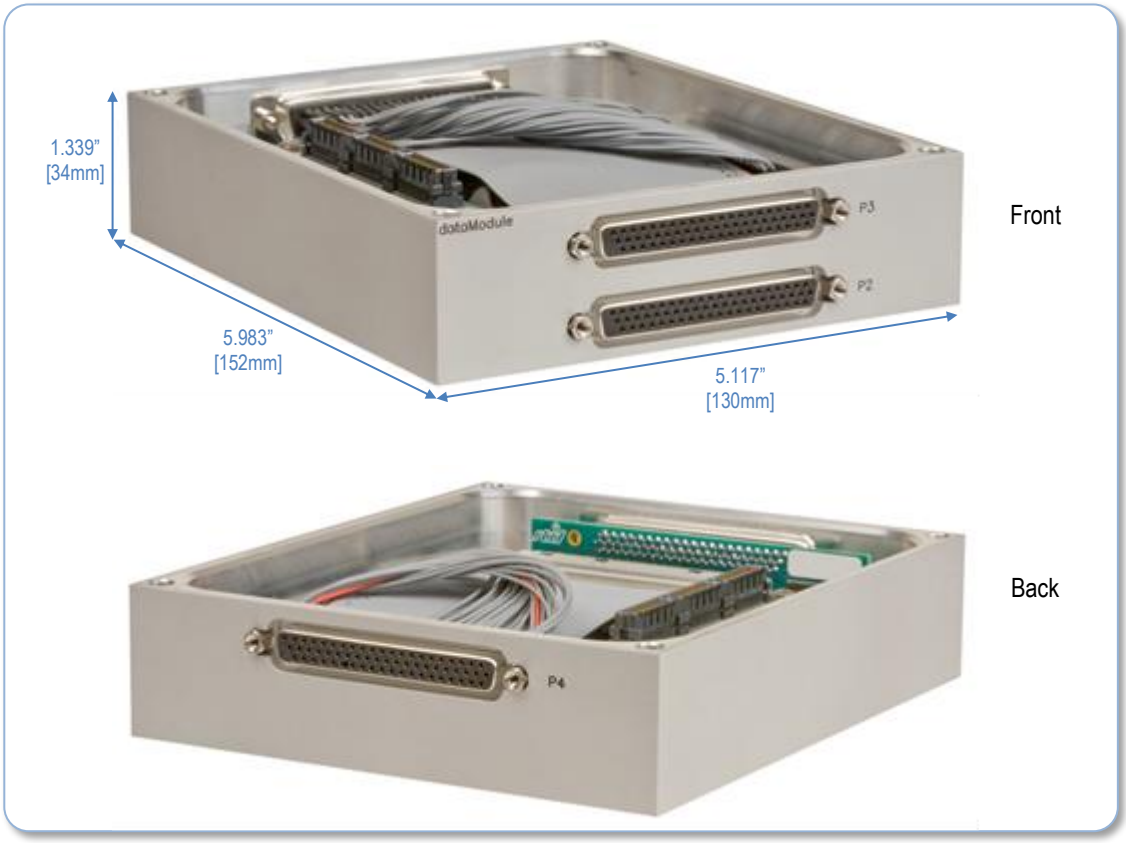


Figure 6: IDAN Dimensions

4.3 Connectors

4.3.1 External I/O Connectors

P2 & P3: Isolated Input Connector

Connector Part #: VALCONN HDB-62S **Mating Connector:** VALCONN HDB-62P

The Isolated Inputs are on P2 and P3, which are connected to CN3 and CN4 respectively. The pin assignments are shown in the Table below.

Table 9: P2 and P3 Pin Assignments

IDAN P2 Pin				CN3
Row 1	Row 2	Row 3	Signal	Pin
1			DI0.0+	1
	22		DI0.0-	2
		43	DI0.1+	3
2			DI0.1-	4
	23		DI0.2+	5
		44	DI0.2-	6
3			DI0.3+	7
	24		DI0.3-	8
		45	DI0.4+	9
4			DI0.4-	10
	25		DI0.5+	11
		46	DI0.5-	12
5			DI0.6+	13
	26		DI0.6-	14
		47	DI0.7+	15
6			DI0.7-	16
	27		DI0.8+	17
		48	DI0.8-	18
7			DI0.9+	19
	28		DI0.9-	20
		49	DI0.10+	21
8			DI0.10-	22
	29		DI0.11+	23
		50	DI0.11-	24
9			DI0.12+	25
	30		DI0.12-	26
		51	DI0.13+	27
10			DI0.13-	28
	31		DI0.14+	29
		52	DI0.14-	30
11			DI0.15+	31
	32		DI0.15-	32
		53	DI0.16+	33
12			DI0.16-	34
	33		DI0.17+	35
		54	DI0.17-	36
13			DI0.18+	37
	34		DI0.18-	38
		55	DI0.19+	39
14			DI0.19-	40
	35		DI0.20+	41
		56	DI0.20-	42
15			DI0.21+	43
	36		DI0.21-	44
		57	DI0.22+	45
16			DI0.22-	46
	37		DI0.23+	47

IDAN P3 Pin				CN4
Row 1	Row 2	Row 3	Signal	Pin
1			DI1.0+	1
	22		DI1.0-	2
		43	DI1.1+	3
2			DI1.1-	4
	23		DI1.2+	5
		44	DI1.2-	6
3			DI1.3+	7
	24		DI1.3-	8
		45	DI1.4+	9
4			DI1.4-	10
	25		DI1.5+	11
		46	DI1.5-	12
5			DI1.6+	13
	26		DI1.6-	14
		47	DI1.7+	15
6			DI1.7-	16
	27		DI1.8+	17
		48	DI1.8-	18
7			DI1.9+	19
	28		DI1.9-	20
		49	DI1.10+	21
8			DI1.10-	22
	29		DI1.11+	23
		50	DI1.11-	24
9			DI1.12+	25
	30		DI1.12-	26
		51	DI1.13+	27
10			DI1.13-	28
	31		DI1.14+	29
		52	DI1.14-	30
11			DI1.15+	31
	32		DI1.15-	32
		53	DI1.16+	33
12			DI1.16-	34
	33		DI1.17+	35
		54	DI1.17-	36
13			DI1.18+	37
	34		DI1.18-	38
		55	DI1.19+	39
14			DI1.19-	40
	35		DI1.20+	41
		56	DI1.20-	42
15			DI1.21+	43
	36		DI1.21-	44
		57	DI1.22+	45
16			DI1.22-	46
	37		DI1.23+	47

Table 9: P2 and P3 Pin Assignments

IDAN P2 Pin			Signal	CN3 Pin
Row 1	Row 2	Row 3		
		58	DI0.23-	48
17			*Reserved*	49
	38		*Reserved*	50
		59	*Reserved*	
18			*Reserved*	
	39		*Reserved*	
		60	*Reserved*	
19			*Reserved*	
	40		*Reserved*	
		61	*Reserved*	
20			*Reserved*	
	41		*Reserved*	
		62	*Reserved*	
21			*Reserved*	
	42		*Reserved*	

IDAN P3 Pin			Signal	CN4 Pin
Row 1	Row 2	Row 3		
		58	DI1.23-	48
17			*Reserved*	49
	38		*Reserved*	50
		59	*Reserved*	
18			*Reserved*	
	39		*Reserved*	
		60	*Reserved*	
19			*Reserved*	
	40		*Reserved*	
		61	*Reserved*	
20			*Reserved*	
	41		*Reserved*	
		62	*Reserved*	
21			*Reserved*	
	42		*Reserved*	

P4: Isolated Output Connector

Connector Part #: VALCONN HDB-62S **Mating Connector:** VALCONN HDB-62P

The Isolated Outputs are on P4, which is connected to CN5. The pin assignments are shown in the Table below.

Table 10: P4 Pin Assignments

IDAN P4 Pin			Signal	C5 Pin
Row 1	Row 2	Row 3		
1			Pull-up 0	1
	22		Collector 0	2
		43	Emitter 0	3
2			Pull-up 1	4
	23		Collector 1	5
		44	Emitter 1	6
3			Pull-up 2	7
	24		Collector 2	8
		45	Emitter 2	9
4			Pull-up 3	10
	25		Collector 3	11
		46	Emitter 3	12
5			Pull-up 4	13
	26		Collector 4	14
		47	Emitter 4	15
6			Pull-up 5	16
	27		Collector 5	17
		48	Emitter 5	18
7			Pull-up 6	19
	28		Collector 6	20
		49	Emitter 6	21
8			Pull-up 7	22
	29		Collector 7	23
		50	Emitter 7	24
9			Pull-up 8	25
	30		Collector 8	26
		51	Emitter 8	27
10			Pull-up 9	28
	31		Collector 9	29
		52	Emitter 9	30
11			Pull-up 10	31
	32		Collector 10	32

Table 10: P4 Pin Assignments

IDAN P4 Pin			Signal	C5 Pin
Row 1	Row 2	Row 3		
		53	Emitter 10	33
12			Pull-up 11	34
	33		Collector 11	35
		54	Emitter 11	36
13			Pull-up 12	37
	34		Collector 12	38
		55	Emitter 12	39
14			Pull-up 13	40
	35		Collector 13	41
		56	Emitter 13	42
15			Pull-up 14	43
	36		Collector 14	44
		57	Emitter 14	45
16			Pull-up 15	46
	37		Collector 15	47
		58	Emitter 15	48
17			*Reserved*	49
	38		*Reserved*	50
		59	*Reserved*	
18			*Reserved*	
	39		*Reserved*	
		60	*Reserved*	
19			*Reserved*	
	40		*Reserved*	
		61	*Reserved*	
20			*Reserved*	
	41		*Reserved*	
		62	*Reserved*	
21			*Reserved*	
	42		*Reserved*	

4.3.2 BUS CONNECTORS

CN1(Top) & CN2(Bottom): PCIe Connector

The PCIe connector is the connection to the system CPU. The position and pin assignments are compliant with the *PCI/104-Express Specification*. (See PC/104 Specifications on page 60)

The DM35838HR is a “Universal” board, and can connect to either a Type 1 or Type 2 PCIe/104 connector.

4.4 Steps for Installing

1. Always work at an ESD protected workstation, and wear a grounded wrist-strap.
2. Turn off power to the IDAN system.
3. Remove the module from its anti-static bag.
4. Check that pins of the bus connector are properly positioned.
5. Check the stacking order; make sure all of the busses used by the peripheral cards are connected to the cpuModule.
6. Hold the module by its edges and orient it so the bus connector pins line up with the matching connector on the stack.
7. Gently and evenly press the module onto the IDAN system.
8. If any boards are to be stacked above this module, install them.
9. Finish assembling the IDAN stack by installing screws of an appropriate length.
10. Attach any necessary cables to the IDAN system.
11. Re-connect the power cord and apply power to the stack.
12. Boot the system and verify that all of the hardware is working properly.

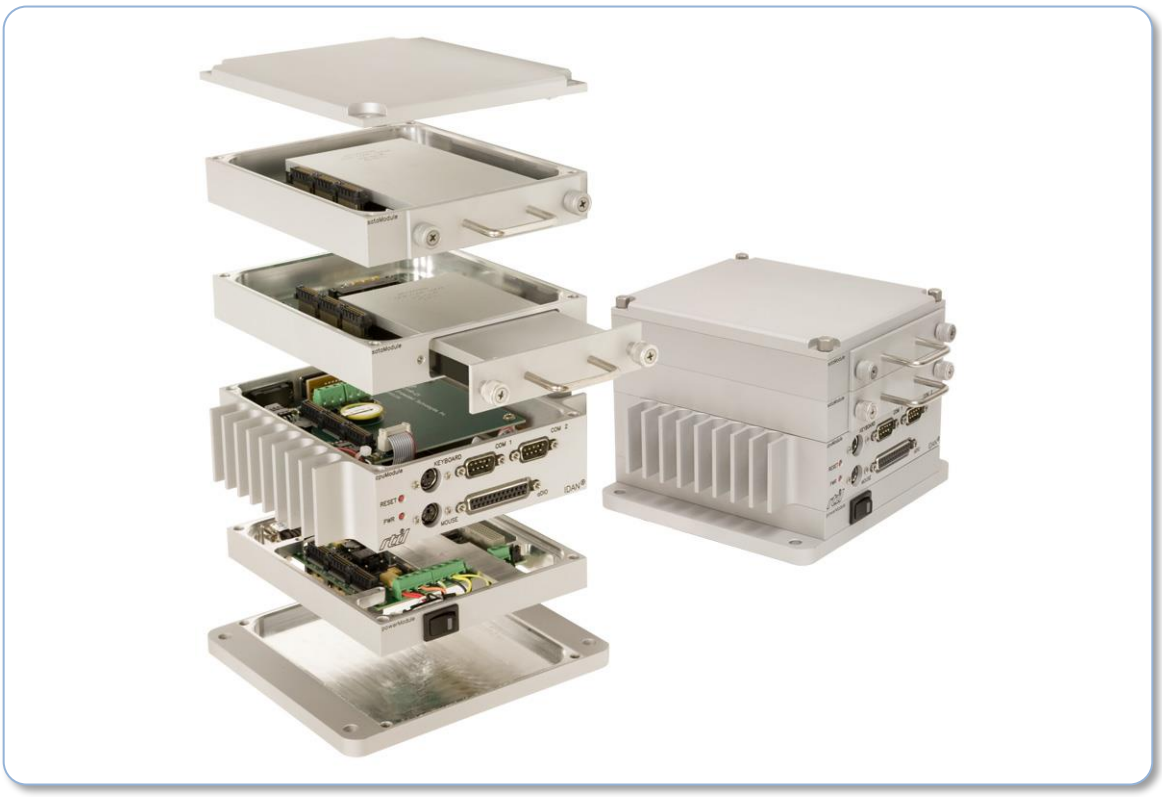


Figure 7: Example IDAN System

5 Functional Description

5.1 Block Diagram

The Figure below shows the functional block diagram of the DM35838HR. The various parts of the block diagram are discussed in the following sections.

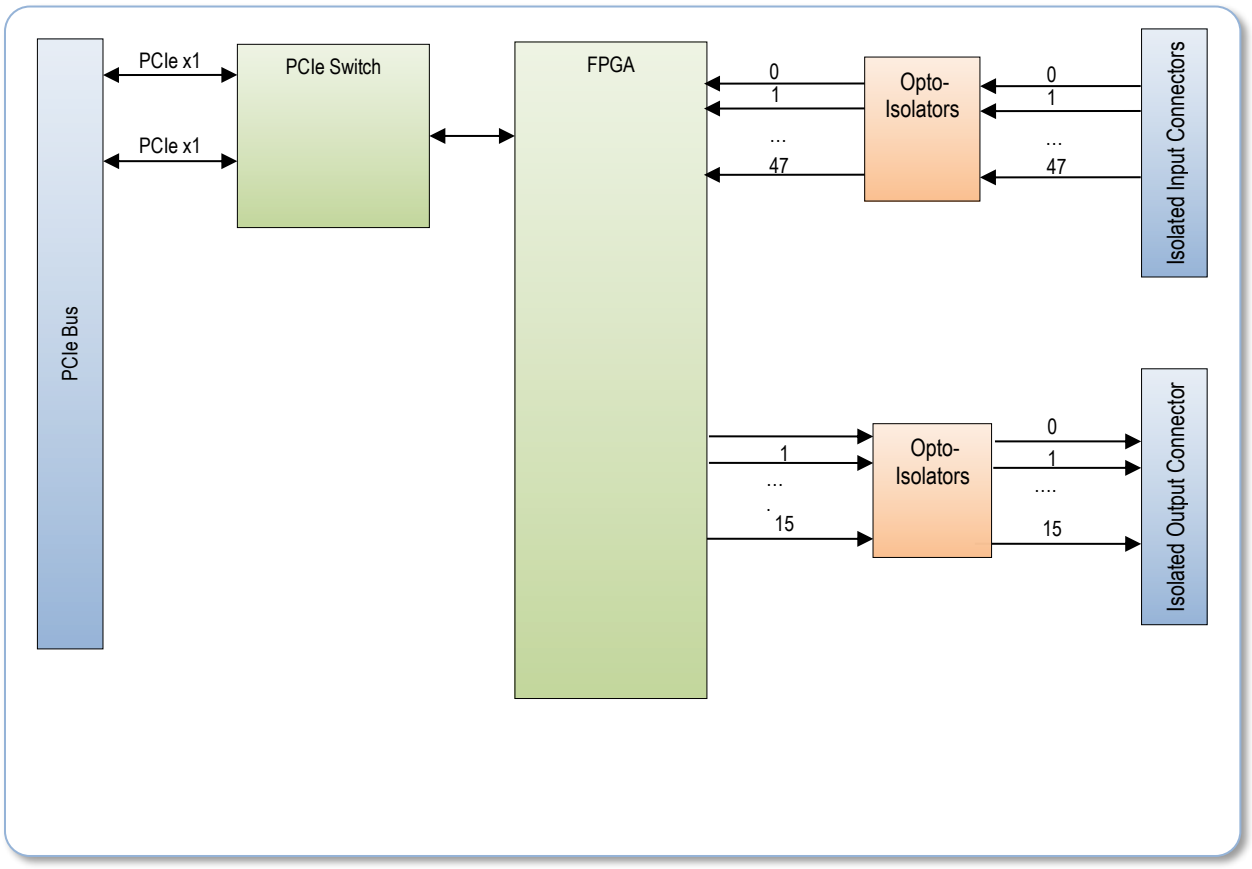


Figure 8: DM35838HR Block Diagram

5.2 PCIe Switch

The PCIe Switch provides lane repopulation. One lane is used as the uplink to the CPU. On the opposite side of the board it is replaced by a lane from the switch. This allows a virtually limitless number of boards to be added to the system.

5.3 FPGA with DMA Engine

The DM35838HR features a FPGA with a built in PCI Express interface and DMA engine. The FPGA controls all communication between the bus and the control logic on the board. The FPGA also features small FIFOs for use with DMA, which is needed for continuous data transfer.

Each DMA channel can be programmed to transfer data from FPGA to PCIe bus or from the PCIe bus to the FPGA. Each DMA channel also features a 64-bit PCI addressing and a 16MB maximum buffer for memory accessing.

5.4 Opto-Isolators

The opto-isolators provide the isolation and input/output circuit for the digital I/O.

5.5 Advanced Digital Inputs

The DM35838HR features two 24-bit digital input function blocks with DMA and advance interrupts.

DMA

The DM35838HR has a DMA channel. Each channel has a 4095 sample FIFO for DMA. Each sample is packed into 32 bits.

Advanced Interrupts

The DM35838HR has an advanced interrupt block that can generate an interrupt on a match, event or strobe. The interrupts are across all 24-digital input of the function block. The bits can be individually selected.

When an interrupt is generated, the data on all the ports is latched into the Capture registers. Bits are tested regardless of if a pin is an input or output.

A Match interrupt is generated when all un-masked bits in the Compare register match the input value of the port. An Event interrupt is generated when any un-masked input port bit changes. A strobe interrupt is generated every event of the CLK_SRC for the function block.

5.6 Advanced Digital Outputs

The DM35838HR features one 16-bit digital output function block with DMA and peripheral select.

DMA

The DM35838HR has a DMA channel. Each channel has a 4095 sample FIFO for DMA. Each sample is packed into 32 bits.

Peripheral Select

The DM35838HR has multipurpose output pins. The outputs can be configured as standard outputs, PWM, or external clocks. The outputs are configurable by bit to maximum flexible of all the features.

Please refer to PERIPH_SEL_BIT_n section on page 45 on how to configure the register for a particular output mode.

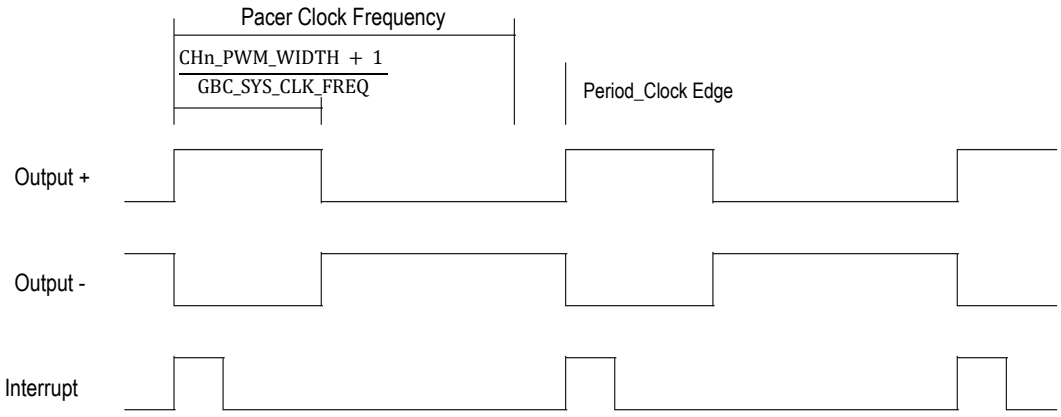
5.7 Quad Pulse Width Modulator

The DM35838HR features 2 quad PWM output functional block with DMA. The PWM is available with the output pin peripherals are configured as PWM.

All the PWM outputs have the same period. The pulse width of each of the four outputs is individually adjustable.

An interrupt is generated at the beginning of every period.

The width register is checked at the beginning of every period. If the width register is modified in the middle of a period, the output will not be affected until the next period.



5.8 Incremental Encoder

The DM35838HR features 4 32-bit incremental encoder functional block with DMA.

An Incremental Encoder is used to detect the relative position of a shaft or linear actuator. A typical implementation is a slotted wheel with two optical sensors positioned such that when one sensor is positioned over a slot, the other is positioned between slots. The output of the optical sensors is shown in Figure 9, with one sensor named "A," and the other named "B." At every edge of the "A" or "B" input, the counter either increments or decrements. The direction can be interpreted from the state of the signals, i.e. which signal leads.

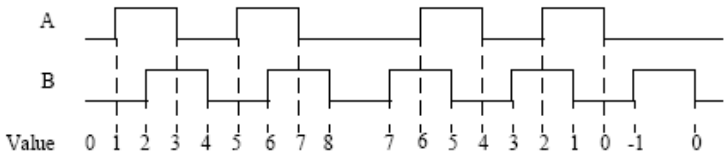


Figure 9: Incremental Encoder Signals

The encoders include a "Phase Filter" that prevents the counter from counting on certain transitions. This allows the encoders to count pulses, and other specialized applications.

Encoder inputs can be configured as single ended or pseudo-differential. In pseudo-differential mode, the "+" and "-" inputs must be the inverse of each other in order for the encoder to see a change.

Separate interrupts are generated for positive and negative rollover. Positive rollover occurs when the counter is at its maximum value, and receives a signal to count up. Negative rollover occurs when the counter is at 0, and receives a signal to count down. Because separate interrupts are generated, the counter can be easily expanded in software.

The Incremental Encoder inputs are show in the table below.

Table 11: Incremental Encoder Inputs

Pin	Encoder 0		Encoder 1		Encoder 2		Encoder 3	
	CN3 IN+ Pin Number	CN3 IN- Pin Number	CN3 IN+ Pin Number	CN3 IN- Pin Number	CN4 IN+ Pin Number	CN4 IN- Pin Number	CN4 IN+ Pin Number	CN4 IN- Pin Number
A+	25	26	37	38	1	2	13	14
A-	27	28	39	40	3	4	15	16
B+	29	30	41	42	5	6	17	18
B-	31	32	43	44	7	8	19	20
Index +	33	34	45	46	9	10	21	22
Index -	35	36	47	48	11	12	23	24

5.9 External Clocking

The DM35838HR features an external clocking function block. This feature allows the user to input a clock to drive a CLK_BUSn signal or output a CLK_BUSn signal. The CLK_SRCn are used to drive the CLK_BUSn signal which are part of the FPGA function block Clock Source. Clock sources are used as either sample clocks for function blocks or triggers for starting and stopping them.

There are 6 available CLK_BUSn, each is associated with a pin on CN3/CN4. Each can be configured to be either an input or an output. As an input to a CLK_BUSn, the max input clock frequency is ½ system clock frequency. This value can be found in GBC_SYS_CLK_FREQ on page 31. As an output CLK_BUSn will generated on the associated pin. By default, this signal will be a pulse that is high for 20ns when the CLK_BUSn signal goes high. The width of this pulse can be increased using EXT_CLKn_PW on page 55.

5.10 Programmable Clock

The DM35838HR features 4 programmable clock functional block.

The programmable clocks use aa master clock and divide it down by an integer. The result of this clock can then be placed on the CLK_BUS and be used as a clock source or used as a stop or start trigger for any other functional blocks.

The programmable clock functional block also features an interval counter and event counter register. These can be used to capture how many events are occur on a particular CLK_BUSn and how many events occur at particular interval of the system clock.

6 Register Address Space

This product was built as a modular design, which allows each board function to have its own Functional Block (FB). Each functional block is designed to work independently. For this reason, we provide individual DMA channels, interrupts, clocks, and FIFOs to each functional block.

The registers are described by their PCIe Base Address Register (BAR), which is defined in the PCI configuration space for this board. The configuration space is generally handled by the operating system. For more information on how to use the configuration space, consult the *PCI Local Bus Specification, Revision 3.0* from the PCI-SIG.

Register Types

There are several different types of registers that are referred to in this section. A description of each type is below.

- Read/Write Registers: The value that is written to this register can also be read back.
- Maskable Registers: This is a 32 bit register that consists of 16-bit data field in the upper word and a 16-bit mask value in the lower word. For each bit in the data field, it is only written to the register if the corresponding bit in the mask field is '1'.
- Sticky Registers: This is a status read register. When a bit in this register has a value of '1', a '1' needs written to that bit to reset the register to '0'. This is typically used for interrupt status registers.
- Read Only: This register can only be read.



NOTE: Writing to Read-Only registers may have unexpected results.

Clock Source

Clock sources can serve as either sample clocks for function blocks, or triggers for starting and stopping them. Function blocks can drive a CLK_SRCn with a CLK_BUSn (see the register descriptions for details on the possible values for CLK_BUSn), and other function blocks then trigger from that clock. This is what lets multiple function blocks start at the same time, or stop on the same trigger.

Below is the list of clock sources and the register value needed to select the source:

0x00: System clock/immediate
0x01: Never
0x02: CLK_BUS2
0x03: CLK_BUS3
0x04: CLK_BUS4
0x05: CLK_BUS5
0x06: CLK_BUS6
0x07: CLK_BUS7
0x08: Reserved
0x09: Reserved
0x0A: CLK_BUS2 Inverted
0x0B: CLK_BUS3 Inverted
0x0C: CLK_BUS4 Inverted
0x0D: CLK_BUS5 Inverted
0x0E: CLK_BUS6 Inverted
0x0F: CLK_BUS7 Inverted

Function Block Mapping

Below is the Function Block slot mapping for the DM35838HR.

Table 12: Function Block Mapping

Slot	ID	Description	Connector
FB0	0x01003011	Digital Input	CN3
FB1	0x01003011	Digital Input	CN4
FB2	0x01003012	Digital Out	CN5
FB3	0x01003100	Quad PWM	CN5 (PWM0)
FB4	0x01003100	Quad PWM	CN5 (PWM1)
FB5	0x01003200	Incremental Encoder	CN3 (Encoder 0)
FB6	0x01003200	Incremental Encoder	CN3 (Encoder 1)
FB7	0x01003200	Incremental Encoder	CN4 (Encoder 2)
FB8	0x01003200	Incremental Encoder	CN4 (Encoder 3)
FB9	0x00000003	External Clocking	CN3/CN5
FB10	0x00000004	Programmable Clock	N/A
FB11	0x00000004	Programmable Clock	N/A
FB12	0x00000004	Programmable Clock	N/A
FB13	0x00000004	Programmable Clock	N/A

6.1 BAR0: General Board Control

This section contains the board-level interrupt control, as well as a description of all the Functional Blocks in BAR2. Most of the registers in this section are Read-Only. Although a full implementation requires BAR0 to be 0x400 in length, most implementations will have a shorter length.

Table 13: Base Functional Block

Offset	0x03	0x02	0x01	0x00
0x00	GBC_BRD_RST	GBC_EOI	GBC_REV	GBC_FMT
0x04	GBC_PDP			
0x08	GBC_BUILD			
0x0C	*Reserved*	GBC_USER_ID	GBC_SYS_CLK_FREQ	
0x10	GBC_IRQ_STATUS			
0x14				
0x18	GBC_DIRQ_STATUS			
0x1C				
0x20	FB0_ID			
0x24	FB0_OFFSET			
0x28	FB0_OFFSET_DMA			
0x2C	reserved			
0x30	FB1_ID			
0x34	FB1_OFFSET			
0x38	FB1_OFFSET_DMA			
0x3C	reserved			
…				
0x20+0x10*n	FBn_ID			
0x24+0x10*n	FBn_OFFSET			
0x28+0x10*n	FBn_OFFSET_DMA			
0x2C+0x10*n	reserved			
…				
0x3F0	FB61_ID			
0x3F4	FB61_OFFSET			
0x3F8	FB61_OFFSET_DMA			
0x3FC	reserved			

6.1.1 GBC_FMT (READ-ONLY)

This register contains the format ID that is used in this board.

- B[7:4] Major Format – Changes to this value indicate changes that are not backward compatible.
  - 0 – Initial Major Format
- B[3:0] Minor Format – Changes to this value indicate additional features that are backward compatible.
  - 1 – Initial Minor Format
  - 2 – Addition of FB_DMAm_CTRLn[Pause] flag and Early DMA Buffer Termination.

6.1.2 GBC_REV (READ-ONLY)

This register contains the PDP revision for this board. A=1, B=2, etc.

6.1.3 GBC_BRD_RST (READ/WRITE)

Write a value of 0xAA to this register to reset the board.

6.1.4 GBC_PDP (READ-ONLY)

This register contains the PDP number for this board in decimal.

6.1.5 GBC_BUILD (READ-ONLY)

Unique 32-bit build number for every FPGA build.

6.1.6 GBC_SYS_CLK_FREQ (READ ONLY)

The measured frequency of the system clock. Units are 10 kHz, i.e. (Frequency in Hertz) = (GBC_SYS_CLK_FREQ * 10 kHz). This is measured once after any reset, including GBC_BRD_RST. This value is not available (will read 0) until 100us after reset.

6.1.7 GBC_USER_ID (READ ONLY)

This value is set by jumpers on the board, and is used to identify individual board when there are multiple in the same system.

6.1.8 GBC_IRQ_STATUS (READ/CLEAR)

This is a 64-bit interrupt status register for non-DMA interrupts. Each bit in this register corresponds to one of the Function Blocks; bit 0 corresponds to FB0 (whose ID and OFFSET are at 0x020), etc. Bits 60 through 63 are reserved. This is a "sticky" register, and the driver clears it by writing a '1' to the appropriate bit. The interrupts signaled by this registered are generally handled by a User Mode Delayed Procedure Call. Any status information required to service the interrupt must be latched in the Function Block. This typically means that there will also be a sticky status register in the Function Block.

The Interrupt Service Routine must be able to clear the interrupt without accessing any registers in the Function Block. This means that if a persistent interrupt condition occurs (i.e. FIFO Empty) the GBC_IRQ_STATUS register will initially be set. The ISR will then clear the bit in GBC_IRQ_STATUS and launch the DPC. In this state the interrupt condition still exists, but the status bit is cleared (because we are servicing it). Depending on implementation, that Function Block may not be able to issue another interrupt while in this state. In this state the full status of the board cannot be determined just from the GBC_IRQ_STATUS register. The Function Block should also maintain a status register that will indicate if the interrupt condition still exists, which will be cleared by the DPC. By doing an End-Of-Interrupt (see Section 6.1.10 GBC_EOI (Read/Clear) below) any "hidden" interrupts will be generated again.

6.1.9 GBC_DIRQ_STATUS (READ/CLEAR)

This is a 64-bit interrupt status register for DMA interrupts. Each bit in this register corresponds to one of the Function Blocks; bit 0 corresponds to FB0 (whose ID and OFFSET are at 0x020), etc. Bits 60 through 63 are reserved. This is a "sticky" register, and the driver clears it by writing a '1' to the appropriate bit. The interrupts signaled by this registered are generally handled by the Kernel Mode ISR. This interrupt is only for the Function Block Standard DMA in Section 6.2 on page 33.

6.1.10 GBC_EOI (READ/CLEAR)

This register is used to acknowledge an interrupt. It is used to safeguard against missing an interrupt. At the end of the ISR, write a 0x01 to this register. If there is another interrupt pending in the status registers, the interrupt line is toggled (Legacy Mode), or another interrupt is sent (MSI Mode).

6.1.11 FBN_ID (READ-ONLY)

This is a 32-bit value that identifies the type of Function Block in slot 'n'. Any changes to a Function Block result in a new Block ID. The mappings for the Functional Block ID can be found on page 29. The ID consists of three fields:

- B[31:24] Type Revision – This is used to indicate the addition of features to the function block. Any checks in the software should be of the form (Type Revision > NN) so that the driver doesn't fail with a newer FPGA. Any changes to the FPGA that would break existing software must change the Type, not the Type Revision.
- B[23:16] Subtype – A Subtype is a variant of the Type with one of the following differences. This allows the driver and library to use the same functions, but have different enumerations.
- B[15:0] Type – The Type of the Function Block is used to define the register map. Any time that the positions of registers change, or registers are added or removed require a new Type number. This is loosely separated into
    - B[15:8] Family – i.e. ADC, DAC, Digital I/O, etc.
    - B[7:0] Member

6.1.12 FBN_OFFSET (READ-ONLY)

This is the offset from the beginning of the Functional Block section (typically the offset in BAR2) that this Functional Block resides in.

6.1.13 FBn_Offset_DMA (Read-Only)

This is the offset from the beginning of the Functional Block section (typically the offset in BAR2) that the Functional Block DMA Registers resides in.

6.2 BAR2: Functional Block Standard DMA

This section describes a standard DMA engine implementation that may be used by the Functional Blocks. Any Functional Block that uses DMA shall use this implementation so that the kernel mode driver may be used to service it, resulting in better performance.

Table 14: DMA Registers

Offset	0x03	0x02	0x01	0x00
D + 0x00	FB_DMAm_Stat_Underflow	FB_DMAm_Stat_Overflow	FB_DMAm_Setup	FB_DMAm_Action
D + 0x04	FB_DMAm_Current_Buffer	FB_DMAm_Count		
D + 0x08	FB_DMAm_RD_FIFO_CNT		FB_DMAm_WR_FIFO_CNT	
D + 0x0C	FB_DMAm_Last_Action	FB_DMAm_Stat_Complete	FB_DMAm_Stat_Invalid	FB_DMAm_Stat_Used
D + 0x10	FB_DMAm_CTRL0	FB_DMAm_STAT0	Reserved	
D + 0x14	Reserved	FB_DMAm_SIZE0		
D + 0x18	FB_DMAm_ADDRESS0			
D + 0x1C				
D + 0x20	FB_DMAm_CTRL1	FB_DMAm_STAT1	Reserved	
D + 0x24	Reserved	FB_DMAm_SIZE1		
D + 0x28	FB_DMAm_ADDRESS1			
D + 0x2C				
…				
D + 0x10 + (0x10 * n)	FB_DMAm_CTRLn	FB_DMAm_STATn	Reserved	
D + 0x14 + (0x10 * n)	Reserved	FB_DMAm_SIZEn		
D + 0x18 + (0x10 * n)	FB_DMAm_ADDRESSn			
D + 0x1C + (0x10 * n)				

6.2.1 FB_DMAm_Action (Read/Write)

After writing to the Action register, the user should poll the Last_Action register (below) until it reads the same value. This shows that the action has been performed by the DMA state machine. This is especially important when entering and exiting the Clear state.

- 0x00 = Clear: Clear the Current Buffer field, the internal offset counters, and the FIFO. DMA is stopped.
- 0x01 = Go: Starts DMA
- 0x02 = Pause: DMA transfers are stopped, but all internal registers maintain their state. During PAUSE you will still receive Stat_Underflow and Stat_Overflow interrupts. After PAUSE, you may transition to GO or CLEAR.
- 0x03 = Halt: Buffer has been filled that has the HALT bit set, or has attempted to use a buffer with the Valid bit cleared. After HALT, you must transition to CLEAR.

6.2.2 FB_DMAm_Last_Action (Read/Write)

The DMA Engine writes the value of FB_DMAm_Action to this register after it has completed this channel. This indicates to the user that the last command has been processed. It specifically aids the transition to Clear. When transitioning to Clear, the user should wait until FB_DMAm_Last_Action indicates that the Clear has been processed before allowing any other Action changes.

The user may also write a value to this register and then poll the register to see when the value changes. This method can be used to detect when the DMA engine services the channel without an Action change.

6.2.3 FB_DMAm_Setup (Read/Write)

- B0:IntEna: Set to '1' to enable the DMA engine to generate interrupts on completion of a buffer.
- B1:ErrIntEna: Set to '1' to enable the DMA engine to generate interrupts on error.
- B2:Direction (RO or R/W): Set to '1' to transfer from the FPGA to the PCI bus. Clear to '0' to transfer from the PCI bus to the FPGA.
- B3:IgnoreUsed: Set to '1' to prevent an error condition when accessing a buffer with the Used bit set. When "IgnoreUsed" is set, the hardware will still set the Used_Desc bit (below) if the next buffer has the Used bit set. However, it will not generate an interrupt, and will not PAUSE the DMA engine. Examples are continuous output from a DAC, or very large Pre-trigger buffering using system memory.

6.2.4 FB_DMAm_Stat_Used (Read/Write)

This register is used to determine the source of a DMA interrupt. The bits are cleared by writing 0x00 to the byte. Stat_Used will be set regardless of having ErrIntEna set to '1', and regardless of the state of the IgnoreUsed bit (above).

- B0: Used_Desc (R/C). Set to '1' by the DMA engine if it attempting to use a descriptor with the Used bit set.

6.2.5 FB_DMAm_Stat_Invalid (Read/Write)

This register is used to determine the source of a DMA interrupt. The bits are cleared by writing 0x00 to the byte. Stat_Invalid will be set regardless of having ErrIntEna set to '1'.

- B0: Invalid_Desc (R/C). Set to '1' by the DMA engine if it attempting to use a descriptor with the Valid bit cleared.

6.2.6 FB_DMAm_Stat_Overflow (Read/Write)

This register is used to determine the source of a DMA interrupt. The bits are cleared by writing 0x00 to the byte. Stat_Overflow will be set regardless of having ErrIntEna set to '1'.

- B0: Overflow (R/C). Set to '1' by the DMA engine if an overflow occurred on the FIFO.

6.2.7 FB_DMAm_Stat_Underflow (Read/Write)

This register is used to determine the source of a DMA interrupt. The bits are cleared by writing 0x00 to the byte. Stat_Underflow will be set regardless of having ErrIntEna set to '1'.

- B0: Underflow (R/C). Set to '1' by the DMA engine if an underflow occurred on the FIFO.

6.2.8 FB_DMAm_Stat_Complete (Read/Write)

This register is used to determine the source of a DMA interrupt. The bits are cleared by writing 0x00 to the byte.

- B0: Buffer_Complete (R/C). Set to '1' by the DMA engine when a buffer is filled that has the Interrupt bit set.

6.2.9 FB_DMAm_Current_Buffer (Read-Only)

This is the ID for the buffer that will be used for the next access. The user may use this to track the progress of the DMA activity.

6.2.10 FB_DMAm_COUNT (Read-Only)

This is the offset in the DMA buffer for the next access. The user may use this to track the progress of the DMA activity.

6.2.11 FB_DMAm_RD_FIFO_CNT(Read-Only)

- B[9:0] This is the amount of data available in the read FIFO in bytes. Software can use this to determine when the FIFO is empty. A value of 0x3FC indicates that there are 1020 or more bytes available.
- B15: RD_EMPTY- '1' indicates that the read FIFO is empty

6.2.12 FB_DMAm_WR_FIFO_CNT(Read-Only)

- B[9:0] This is the amount of space available in the write FIFO in bytes. Software can use this to determine when the FIFO is empty. A value of 0x3FC indicated that there are 1020 or more bytes available.
- B15: WR_FULL- '1' indicates that the read FIFO is full

6.2.13 FB_DMAm_ADDRESSn (Read/Write)

This is the 64-bit PCI address for DMA Channel m, buffer n. It must be double-word aligned (i.e. b[1:0] are reserved).

6.2.14 FB_DMAm_SIZEn (Read/Write)

This is the size in bytes of the buffer for DMA Channel m, buffer n. It must be an integer number of double-words (i.e. b[1:0] are reserved). The actual size is FB_DMAm_SIZEn + 4 Bytes. The maximum buffer size is 16MB.

6.2.15 FB_DMAm_CTRLn (Read/Write)

- B0: Valid: Sets this bit to '1' to indicate that this contains valid information. The DMA engine will set the error bit and halt if it is ready to use this descriptor and it is not valid.
- B1: Halt: Sets this bit to '1' to halt the DMA engine after this buffer is full.
- B2: Loop: Sets this bit to '1' to start back at descriptor 0 after this buffer is full. This has a higher priority than the HALT bit.
- B3: Interrupt: Sets this bit to '1' to generate an interrupt after this buffer is full.
- B4: Pause: Sets this bit to '1' to pause the DMA engine after this buffer is full.

If the last buffer is reached, and the HALT and LOOP bits are both '0', the DMA engine will loop.

If the last buffer is reached, and the HALT and LOOP bits are both '1', the DMA engine will halt and the Current_Buffer will be set to 0.

6.2.16 FB_DMAm_STATn (Read/Clear)

- B0: Used (R/C): DMA engine sets to '1' to indicate that it has completely used this descriptor. The user must clear this bit when it is ready to be used again. The DMA engine will set the error bit and PAUSE if it is ready to use this descriptor and the Used bit is set, unless the IgnoreUsed bit is set. The bits are cleared by writing 0x00 to the byte.
- B1: Terminated (R/C): DMA engine sets to '1' to indicate that this buffer was terminated early at the request of the associated Function Block.

6.3 BAR2: Functional Block Header Registers

Each Functional Block has a standard header as follows.

Table 15: Functional Block Header

Offset	0x03	0x02	0x01	0x00
FB + 0x00	FB_ID			
FB + 0x04	FB_DMA_BUFFERS	FB_DMA_CHANNELS	Reserved	Reserved

6.3.1 FB_ID (READ-ONLY)

This is a repeat of the ID from the General Board Control section. This can be used to validate that the correct Function Block is being accessed.

6.3.2 FB_DMA_CHANNELS (READ -ONLY)

This register contains the number of DMA Channels in this Function Block.

6.3.3 FB_DMA_BUFFERS (READ-ONLY)

This register contains the number of Buffer Descriptors in each DMA Channel.

6.4 BAR2: Digital Input Functional Block

C in the register offset stands for channel number

Table 16: Digital Input Functional Block

	Offset	0x03	0x02	0x01	0x00
Header	FB + 0x00	FB_ID			
	FB + 0x04	FB_DMA_BUFFERS	FB_DMA_CHANNELS	FB_CONTROL_SIZE	FB_CHANNEL_SIZE
	FB + 0x08	STOP_TRIG	START_TRIG	CLK_SRC	MODE_STATUS
	FB + 0x0C	CLK_DIV			
	FB + 0x10	CLK_DIV_CNTR			
	FB + 0x14	PRE_START_COUNT (limited by FIFO size)			
	FB + 0x18	POST_STOP_COUNT			
	FB + 0x1C	SAMPLE_CNT			
	FB + 0x20	INT_ENA (Conversion, Start, Stop, Error, Channel)			
	FB + 0x24	INT_STAT		*Reserved*	
	FB + 0x28	CLK_BUS3	CLK_BUS2	*Reserved*	*Reserved*
	FB + 0x2C	CLK_BUS7	CLK_BUS6	CLK_BUS5	CLK_BUS4
	FB + 0x30	*Reserved*			ADV_INT_MODE
	FB + 0x34	ADV_INT_MASK			
	FB + 0x38	ADV_INT_COMP			
	FB + 0x3C	ADV_INT_CAPT			
	FB + 0x40	MODE_CONFIG (Maskable register – 16-bit)			
	FB + 0x44	*Reserved*			
	FB + 0x48	*Reserved*			
	FB + 0x4C	*Reserved*			
	FB + 0x50	*Reserved*			
	FB + 0x54	*Reserved*			
	FB + 0x58	*Reserved*			
	FB + 0x5C	*Reserved*			
	FB + 0x60	*Reserved*			
Digital I/O Channels	FB + 0x08 + (FB_CONTROL_SIZE * 0x04) + (FB_CHANNEL_SIZE * 0x04 * c)	CH0_FIFO_DATA_CNT			
	FB + 0x0C + (FB_CONTROL_SIZE * 0x04) + (FB_CHANNEL_SIZE * 0x04 * c)	CH0_MAX_FIFO_SIZE			
	FB + 0x10 + (FB_CONTROL_SIZE * 0x04) + (FB_CHANNEL_SIZE * 0x04 * c)	DIG_IN			
	FB + 0x14 + (FB_CONTROL_SIZE * 0x04) + (FB_CHANNEL_SIZE * 0x04 * c)	CH0_FIFO_ACCESS (DIG_IN)			
	FB + 0x18 + (FB_CONTROL_SIZE * 0x04) + (FB_CHANNEL_SIZE * 0x04 * c)	*Reserved*			
	FB + 0x1C + (FB_CONTROL_SIZE * 0x04) + (FB_CHANNEL_SIZE * 0x04 * c)	*Reserved*			
	FB + 0x20 + (FB_CONTROL_SIZE * 0x04) + (FB_CHANNEL_SIZE * 0x04 * c)	*Reserved*			
	FB + 0x24 + (FB_CONTROL_SIZE * 0x04) + (FB_CHANNEL_SIZE * 0x04 * c)	*Reserved*			
	FB + 0x28 + (FB_CONTROL_SIZE * 0x04) + (FB_CHANNEL_SIZE * 0x04 * c)	*Reserved*			
	FB + 0x2C + (FB_CONTROL_SIZE * 0x04) + (FB_CHANNEL_SIZE * 0x04 * c)	*Reserved*			
	FB + 0x30 + (FB_CONTROL_SIZE * 0x04) + (FB_CHANNEL_SIZE * 0x04 * c)	*Reserved*			

Table 16: Digital Input Functional Block

	Offset	0x03	0x02	0x01	0x00
	FB + 0x34 + (FB_CONTROL_SIZE * 0x04) + (FB_CHANNEL_SIZE * 0x04 * c)	*Reserved*			

6.4.1 FB_CONTROL_SIZE (READ -ONLY)

This register contains the total number of DWORDs register in control section of the function block. This can be used to calculate the offset for each register in a channel.

6.4.2 FB_CHANNEL_SIZE (READ-ONLY)

This register contains the total number of DWORDs registers in each channel of the function block. This can be used to calculate the offset for each register in a channel.

6.4.3 MODE_STATUS (READ/WRITE, READ-ONLY)

Selects the current mode of operation and indicates its triggering status.

B[3:0]: Mode

- 0x04: Uninitialized. This is the power-on state. No converter initialization has taken place. Sampling is stopped, and all counters are reset and the triggering state machine is reset. Transition to any of the other Modes will start converter initialization (sampling will not start until initialization is complete).
- 0x00: Reset. Sampling is stopped. All counters are reset and the triggering state machine is reset.
- 0x01: Paused. Sampling is stopped, but the counters and triggering state machine maintain their state.
- 0x02: Go, Single-Shot. After filling the buffer with the Post-Stop samples, capturing stops. The Mode must be set back to RESET in order to capture more samples.
- 0x03: Go, Re-arm. After filling the buffer with the Post-Stop samples and the FIFO is empty, the triggering state machine is restarted, i.e. FIFO is filled with Pre-Start samples and waits for a start trigger.

B[7:4]: Status

- 0x08: Uninitialized – The status when in the “Uninitialized” mode and the converter requires initialization.
- 0x09: Initializing
- 0x00: Stopped – The status when in the “Reset” mode, or in the “Uninitialized” mode and the converter does not require initialization.
- 0x01: Filling Pre-Trigger buffer
- 0x02: Waiting for start trigger
- 0x03: Sampling/Waiting for stop trigger
- 0x04: Filling Post-Stop buffer
- 0x05: Wait to re-arm – Waiting until local FIFO is empty so the pre-trigger buffer can be filled.
- 0x07: Done capturing

6.4.4 CLK_SRC (READ/WRITE)

Selects the source for the clock of this function block.

Refer to Clock Source section on page 28, for list of valid values.

6.4.5 START_TRIG (READ/WRITE)

Selects the start trigger from the clock bus. CLK_DIV will start counting after the start trigger, unless PRE_START_COUNT is non-zero in which case CLK_DIV will start counting immediately.

Refer to Clock Source section on page 28, for list of valid values.

In addition to the clock source list the follow values are valid:

- 0x08: Advanced Interrupt

6.4.6 STOP_TRIG (READ/WRITE)

Selects the stop trigger from the clock bus.

Refer to Clock Source section on page 28, for list of valid values.

In addition to the clock source list the follow values are valid:

- 0x08: Advanced Interrupt

6.4.7 CLK_DIV (READ/WRITE)

Divider for the pacer clock. Pacer Clock Frequency = (Clk_Src_Frequency) / (1 + CLK_DIV). If synchronizing with the pacer clock from another Function Block (by using one of the CLK_BUS signals), this is typically set to 0.

6.4.8 CLK_DIV_CNTR (READ ONLY)

The current value of the Clock Divide Counter. This counter starts at a value of CLK_DIV, and counts down. When it reaches zero, a sample is taken. This is useful when using a slow sample clock.

6.4.9 PRE_START_COUNT (READ/WRITE)

Number of samples to collect before the Start Trigger. The length is limited by the FIFO size – writing a value larger than the FIFO size will have indeterminate results.

6.4.10 POST_STOP_COUNT (READ/WRITE)

Number of samples to collect after the Stop Trigger.

6.4.11 SAMPLE_CNT (READ ONLY)

Total number of samples collected. This does not increment in while in the “Waiting For Start Trigger” state. It also continues counting after a Re-Arm.

6.4.12 INT_ENA (MASKABLE READ/WRITE)

Each bit corresponds to an interrupt source. A value of ‘1’ enables the source, and a value of ‘0’ disables it. See below for a description of the sources.

6.4.13 INT_STAT (READ/CLEAR)

Each bit corresponds to an interrupt source. Reading a value of ‘1’ indicates that an event has occurred. Reading a value of ‘0’ indicates that the event has not occurred. Writing a ‘1’ will clear that bit.

B0: Sample – A sample has been taken.

B1: Advanced Interrupt – Interrupt generated from advanced interrupt mode.

B2: Pre-Start Buffer Filled

B3: Start Trigger

B4: Stop Trigger

B5: Post-Stop Buffer Filled

B6: Sampling has completed and the FIFO is empty (all data transferred to host)

B7: Pacer – The pacer clock has ticked.

B8: Reserved

B9: Reserved

B10: Strobe – Positive edge detected (Must have strobe enable,MODE_CONFIG(Strobe) = ‘1’)

6.4.14 CLK_BUSN

Selects a source to drive onto Clock Bus N. That clock bus can then be used by a different function block as a clock source or trigger.

A function block can drive multiple different Clock Buses. However, a Clock Bus N should not be driven by more than one function block at the same time or the clock signal will be undefined.

B[7:0]:

- 0x00: Disabled (Default)
- 0x80: Sample – A sample has been taken.
- 0x81: Advanced Interrupt – Interrupt generated from advanced interrupt mode.
- 0x82: Pre-Start Buffer Filled
- 0x83: Start Trigger
- 0x84: Stop Trigger
- 0x85: Post-Stop Buffer Filled
- 0x86: Sampling has completed and the FIFO is empty (all data transferred to host)
- 0x87: Pacer – The pacer clock has ticked.
- 0x88: Reserved
- 0x89: Reserved
- 0x8A: Strobe – Positive edge detected (Must have strobe enable,MODE_CONFIG(Strobe) = '1')

6.4.15 ADV_INT_MODE (READ/WRITE)

Set the current mode for the advance interrupts.

B[1:0]: Mode

- 0x0: Disabled. This is the power-on state. In this mode no advanced interrupts will occur.
- 0x1: Match. A Match interrupt is generated when all un-masked bits in the Compare register match the input value of the port. This is when the following expression is true:

  ((DIG_IN xor ADV_INT_COMP) and not ADV_INT_MASK) = '0'
- 0x2: Event Mode. An Event interrupt is generated when any un-masked input port bit changes. This is when the following expression is:

  ((DIG_IN xor ADV_INT_ CAPT) and not ADV_INT_MASK) = '1'

  The ADV_INT_CAPT register is updated at every advanced interrupt or event.
- 0x3: Strobe. A Strobe interrupt is generated every rising edge of the CLK_SRC for the function block:

  The ADV_INT_CAPT register is updated at every advanced interrupt

6.4.16 ADV_INT_MASK (READ/WRITE)

This register determines if a bit is checked for the advanced interrupts.

0 = Bit is used for match/event 1 = Bit is ignored

6.4.17 ADV_INT_COMP (READ/WRITE)

The compare register is used for the Match interrupt. When all selected bits in this register match all selected bits on the DIG_IN register, an interrupt is generated.

6.4.18 ADV_INT_CAPT (READ/WRITE)

The Capture register latches the input ports when an interrupt is generated. All values are latched, regardless of the ADV_INT_MASK register, or DIO_DIRECTION.

This register can be written to when ADV_INT_MODE is set to Disabled.

6.4.19 MODE_CONFIG (MASKABLE READ/WRITE)

This register provides configuration for Digital Input Function Block.

B[2]: Strobe: Configures the most significant bit of the DIN function block as a strobe input
- '0' = Normal input
- '1' = Strobe input

B[1:0]: Reserved

6.4.20 CHN_FIFO_DATA_CNT (READ)

This register shows the current sample count that is available in the PWM channel FIFO.

6.4.21 CHN_MAX_FIFO_SIZE (READ)

This register shows the max number of samples that the PWM channel FIFO can hold.

6.4.22 DIG_IN (READ ONLY)

This register provides the current value on the Digital In lines. The bits in the register correspond with the Digital In pins as follows:

Bit	CN3/CN4 IN+ Pin Number	CN3/CN4 IN- Pin Number	Signal Name
23	47	48	DIn.23
22	45	46	DIn.22
21	43	44	DIn.21
20	41	42	DIn.20
19	39	40	DIn.19
18	37	38	DIn.18
17	35	36	DIn.17
16	33	34	DIn.16
15	31	32	DIn.15
14	29	30	DIn.14
13	27	28	DIn.13
12	25	26	DIn.12

Bit	CN3/CN4 IN+ Pin Number	CN3/CN4 IN- Pin Number	Signal Name
11	23	24	DIn.11
10	21	22	DIn.10
9	19	20	DIn.9
8	17	18	DIn.8
7	15	16	DIn.7
6	13	14	DIn.6
5	11	12	DIn.5
4	9	10	DIn.4
3	7	8	DIn.3
2	5	6	DIn.2
1	3	4	DIn.1
0	1	2	DIn.0

This is the same value that is written into the DMA INPUT FIFO.

6.4.23 CHN_FIFO_ACCESS (READ/WRITE)

This register provides direct access to the DMA FIFO. It can be used to access the data without the use of the DMA engine. The DMA engine for this channel must be set to "Pause." Each register access advances to the next sample.

6.5 BAR2: Digital Out Functional Block

C in the register offset stands for channel number

Table 17: Digital Out Functional Block

	Offset	0x03	0x02	0x01	0x00
Header	FB + 0x00	FB_ID			
	FB + 0x04	FB_DMA_BUFFERS	FB_DMA_CHANNELS	FB_CONTROL_SIZE	FB_CHANNEL_SIZE
	FB + 0x08	STOP_TRIG	START_TRIG	CLK_SRC	MODE_STATUS
	FB + 0x0C	CLK_DIV			
	FB + 0x10	CLK_DIV_CNTR			
	FB + 0x14	*Reserved*			
	FB + 0x18	POST_STOP_COUNT			
	FB + 0x1C	CONVERSION_CNT			
	FB + 0x20	INT_ENA (Conversion, Start, Stop, Error, Channel)			
	FB + 0x24	INT_STAT		*Reserved*	
	FB + 0x28	CLK_BUS3	CLK_BUS2		*Reserved*
	FB + 0x2C	CLK_BUS7	CLK_BUS6	CLK_BUS5	CLK_BUS4
	FB + 0x30	*Reserved*			*Reserved*
	FB + 0x34	*Reserved*			
	FB + 0x38	*Reserved*			
	FB + 0x3C	*Reserved*			
	FB + 0x40	*Reserved*			
	FB + 0x44	PERIPH_SEL_BIT_3	PERIPH_SEL_BIT_2	PERIPH_SEL_BIT_1	PERIPH_SEL_BIT_0
	FB + 0x48	PERIPH_SEL_BIT_7	PERIPH_SEL_BIT_6	PERIPH_SEL_BIT_5	PERIPH_SEL_BIT_4
	FB + 0x4C	PERIPH_SEL_BIT_11	PERIPH_SEL_BIT_10	PERIPH_SEL_BIT_9	PERIPH_SEL_BIT_8
	FB + 0x50	PERIPH_SEL_BIT_15	PERIPH_SEL_BIT_14	PERIPH_SEL_BIT_13	PERIPH_SEL_BIT_12
	FB + 0x54	*Reserved*	*Reserved*	*Reserved*	*Reserved*
	FB + 0x58	*Reserved*	*Reserved*	*Reserved*	*Reserved*
	FB + 0x5C	*Reserved*	*Reserved*	*Reserved*	*Reserved*
	FB + 0x60	*Reserved*	*Reserved*	*Reserved*	*Reserved*
Digital I/O Channels	FB + 0x08 + (FB_CONTROL_SIZE * 0x04) + (FB_CHANNEL_SIZE * 0x04 * c)	*Reserved*			
	FB + 0x0C + (FB_CONTROL_SIZE * 0x04) + (FB_CHANNEL_SIZE * 0x04 * c)	*Reserved*			
	FB + 0x10 + (FB_CONTROL_SIZE * 0x04) + (FB_CHANNEL_SIZE * 0x04 * c)	*Reserved*			
	FB + 0x14 + (FB_CONTROL_SIZE * 0x04) + (FB_CHANNEL_SIZE * 0x04 * c)	*Reserved*			
	FB + 0x18 + (FB_CONTROL_SIZE * 0x04) + (FB_CHANNEL_SIZE * 0x04 * c)	CH1_FIFO_DATA_CNT			
	FB + 0x1C + (FB_CONTROL_SIZE * 0x04) + (FB_CHANNEL_SIZE * 0x04 * c)	CH1_MAX_FIFO_SIZE			
	FB + 0x20 + (FB_CONTROL_SIZE * 0x04) + (FB_CHANNEL_SIZE * 0x04 * c)	DIG_OUT			
	FB + 0x24 + (FB_CONTROL_SIZE * 0x04) + (FB_CHANNEL_SIZE * 0x04 * c)	CH1_FIFO_ACCESS (DIG_OUT)			
	FB + 0x28 + (FB_CONTROL_SIZE * 0x04) + (FB_CHANNEL_SIZE * 0x04 * c)	*Reserved*			
	FB + 0x2C + (FB_CONTROL_SIZE * 0x04) + (FB_CHANNEL_SIZE * 0x04 * c)	*Reserved*			
	FB + 0x30 + (FB_CONTROL_SIZE * 0x04) + (FB_CHANNEL_SIZE * 0x04 * c)	*Reserved*			

Table 17: Digital Out Functional Block

	Offset	0x03	0x02	0x01	0x00
	FB + 0x34 + (FB_CONTROL_SIZE * 0x04) + (FB_CHANNEL_SIZE * 0x04 * c)	*Reserved*			

6.5.1 FB_CONTROL_SIZE (READ -ONLY)

This register contains the total number of DWORDs register in control section of the function block. This can be used to calculate the offset for each register in a channel.

6.5.2 FB_CHANNEL_SIZE (READ-ONLY)

This register contains the total number of DWORDs registers in each channel of the function block. This can be used to calculate the offset for each register in a channel.

6.5.3 MODE_STATUS (READ/WRITE, READ-ONLY)

Selects the current mode of operation and indicates its triggering status.

B[3:0]: Mode

- 0x04: Uninitialized. This is the power-on state. No converter initialization has taken place. Converting is stopped, and all counters are reset and the triggering state machine is reset. Transition to any of the other Modes will start converter initialization (converter will not start until initialization is complete).
- 0x00: Reset. Converting is stopped. All counters are reset and the triggering state machine is reset.
- 0x01: Paused. Converting is stopped, but the counters and triggering state machine maintain their state.
- 0x02: Go, Single-Shot. Converter starts. When stop trigger occurs, post-stop conversions take place and then converting stops. The Mode must be set back to RESET in order to start again.
- 0x03: Go, Re-arm. Converter starts. When stop trigger occurs, post-stop conversions take place and then the triggering state machine is restarted, i.e. waits for a start trigger.

B[7:4]: Status

- 0x08: Uninitialized – The status when in the “Uninitialized” mode and the converter requires initialization.
- 0x09: Initializing
- 0x00: Stopped – The status when in the “Reset” mode, or in the “Uninitialized” mode and the converter does not require initialization.
- 0x01: *Reserved*
- 0x02: Waiting for start trigger
- 0x03: Running/Waiting for stop trigger
- 0x04: Output Post-Stop buffer
- 0x05: Wait to re-arm.
- 0x07: Done

6.5.4 CLK_SRC (READ/WRITE)

Selects the source for the clock of this function block.

Refer to Clock Source section on page 28, for list of valid values.

6.5.5 START_TRIG (READ/WRITE)

Selects the start trigger from the clock bus. CLK_DIV will start counting after the start trigger, unless PRE_START_COUNT is non-zero in which case CLK_DIV will start counting immediately.

Refer to Clock Source section on page 28, for list of valid values.

6.5.6 STOP_TRIG (READ/WRITE)

Selects the stop trigger from the clock bus.

Refer to Clock Source section on page 28, for list of valid values.

6.5.7 CLK_DIV (READ/WRITE)

Divider for the pacer clock. Pacer Clock Frequency = (Clk_Src_Frequency) / (1 + CLK_DIV). If synchronizing with the pacer clock from another Function Block (by using one of the CLK_BUS signals), this is typically set to 0.

6.5.8 CLK_DIV_CNTR (READ ONLY)

The current value of the Clock Divide Counter. This counter starts at a value of CLK_DIV, and counts down. When it reaches zero, a conversion is made. This is useful when using a slow clock.

6.5.9 POST_STOP_COUNT (READ/WRITE)

Number of values to convert after the Stop Trigger.

6.5.10 CONVERSION_CNT (READ ONLY)

Total number of conversions executed. This does not increment in while in the "Waiting For Start Trigger" state. It also continues counting after a Re-Arm.

6.5.11 INT_ENA (MASKABLE READ/WRITE)

Each bit corresponds to an interrupt source. A value of '1' enables the source, and a value of '0' disables it. See below for a description of the sources.

6.5.12 INT_STAT (READ/CLEAR)

Each bit corresponds to an interrupt source. Reading a value of '1' indicates that an event has occurred. Reading a value of '0' indicates that the event has not occurred. Writing a '1' will clear that bit.

B0: Conversion – The DIGOUT pins have been updated with a new value.

B1: *Reserved*

B2: *Reserved*

B3: Start Trigger

B4: Stop Trigger

B5: Post-Stop Buffer Output

B6: Conversions completed and the FIFO is empty (all data transferred)

B7: Pacer – The pacer clock has ticked.

6.5.13 CLK_BUSN

Selects a source to drive onto Clock Bus N. That clock bus can then be used by a different function block as a clock source or trigger.

A function block can drive multiple different Clock Buses. However, a Clock Bus N should not be driven by more than one function block at the same time or the clock signal will be undefined.

B[7:0]:

- 0x00: Disabled (Default)
- 0x80: Conversion – A new value is on the DIGOUT pins.
- 0x81: *Reserved*
- 0x82: *Reserved*
- 0x83: Start Trigger
- 0x84: Stop Trigger
- 0x85: Post-Stop Buffer Output
- 0x86: Conversions completed and the FIFO is empty (all data transferred)
- 0x87: Pacer – The pacer clock has ticked.

6.5.14 PERIPH_SEL_BIT_N (READ/WRITE)

This register selects the peripheral for the outputs pin of this function block.

Bit	CN5 Pin Number	PERIPH_SEL_BIT_n		
		0x00	0x01	0x02
15	47	DIG_OUT (15)	PWM1_N (3)	
14	44	DIG_OUT (14)	PWM1_P (3)	
13	41	DIG_OUT (13)	PWM1_N (2)	
12	38	DIG_OUT (12)	PWM1_P (2)	
11	35	DIG_OUT (11)	PWM1_N (1)	
10	32	DIG_OUT (10)	PWM1_P (1)	
9	29	DIG_OUT (9)	PWM1_N (0)	
8	26	DIG_OUT (8)	PWM1_P (0)	
7	23	DIG_OUT (7)	PWM0_N (3)	
6	20	DIG_OUT (6)	PWM0_P (3)	
5	17	DIG_OUT (5)	PWM0_N (2)	EXT_CLK_7
4	14	DIG_OUT (4)	PWM0_P (2)	EXT_CLK_6
3	11	DIG_OUT (3)	PWM0_N (1)	EXT_CLK_5
2	8	DIG_OUT (2)	PWM0_P (1)	EXT_CLK_4
1	5	DIG_OUT (1)	PWM0_N (0)	EXT_CLK_3
0	2	DIG_OUT (0)	PWM0_P (0)	EXT_CLK_2

6.5.15 CHN_FIFO_DATA_CNT (READ)

This register shows the current sample count that is available in the FIFO.

6.5.16 CHN_MAX_FIFO_SIZE (READ)

This register shows the max number of samples that FIFO can hold.

6.5.17 DIG_OUT (READ/WRITE)

The last value sent to the Digital Output. For each bit, a '0' causes the output to be open, and a value '1' causes the output to be closed. The bits in the register correspond with the Digital Out pins as follows:

Bit	CN5 Collector Pin Number	CN5 Emitter Pin Number	Signal Name
15	47	48	DO15
14	44	45	DO14
13	41	42	DO13
12	38	39	DO12
11	35	36	DO11
10	32	33	DO10
9	29	30	DO9
8	26	27	DO8

Bit	CN5 Collector Pin Number	CN5 Emitter Pin Number	Signal Name
7	23	24	DO7
6	20	21	DO6
5	17	18	DO5
4	14	15	DO4
3	11	12	DO3
2	8	9	DO2
1	5	6	DO1
0	2	3	DO0

This register defaults to all 0 at power on.

If the current Mode is "Reset" or the associated DMA engine is set to "Clear", a write to this register will immediately update the Digital I/O Output. Bit assignments are the same as above.

6.5.18 CHN_FIFO_ACCESS (READ/WRITE)

This register provides direct access to the DMA FIFO. It can be used to access the data without the use of the DMA engine. The DMA engine for this channel must be set to "Pause." Each register access advances to the next sample.

6.6 BAR2: Quad PWM Functional Block

C in the register offset stands for channel number

Table 18:PWM Functional Block

	Offset	0x03	0x02	0x01	0x00
Header	FB + 0x00	FB_ID			
	FB + 0x04	FB_DMA_BUFFERS	FB_DMA_CHANNELS	FB_CONTROL_SIZE	FB_CHANNEL_SIZE
PWM Control	FB + 0x08	STOP_TRIG	START_TRIG	CLK_SRC	MODE_STATUS
	FB + 0x0C	CLK_DIV			
	FB + 0x10	CLK_DIV_CNTR			
	FB + 0x14	Reserved			
	FB + 0x18	Reserved			
	FB + 0x1C	PERIOD_COUNT			
	FB + 0x20	INT_ENA (Conversion, Start, Stop, Error, Channel)			
	FB + 0x24	INT_STAT		*Reserved*	
	FB + 0x28	CLK_BUS3	CLK_BUS2	*Reserved*	*Reserved*
	FB + 0x2C	CLK_BUS7	CLK_BUS6	CLK_BUS5	CLK_BUS4
Channel 0	FB + 0x08 + (FB_CONTROL_SIZE * 0x04) + (FB_CHANNEL_SIZE * 0x04 * c)	CH0_FIFO_DATA_CNT			
	FB + 0x0C + (FB_CONTROL_SIZE * 0x04) + (FB_CHANNEL_SIZE * 0x04 * c)	CH0_MAX_FIFO_SIZE			
	FB + 0x10 + (FB_CONTROL_SIZE * 0x04) + (FB_CHANNEL_SIZE * 0x04 * c)	CH0_PWM_WIDTH			
	FB + 0x14 + (FB_CONTROL_SIZE * 0x04) + (FB_CHANNEL_SIZE * 0x04 * c)	CH0_FIFO_ACCESS			
Channel 1	FB + 0x08 + (FB_CONTROL_SIZE * 0x04) + (FB_CHANNEL_SIZE * 0x04 * c)	CH1_FIFO_DATA_CNT			
	FB + 0x0C + (FB_CONTROL_SIZE * 0x04) + (FB_CHANNEL_SIZE * 0x04 * c)	CH1_MAX_FIFO_SIZE			
	FB + 0x10 + (FB_CONTROL_SIZE * 0x04) + (FB_CHANNEL_SIZE * 0x04 * c)	CH1_PWM_WIDTH			
	FB + 0x14 + (FB_CONTROL_SIZE * 0x04) + (FB_CHANNEL_SIZE * 0x04 * c)	CH1_FIFO_ACCESS			
Channel 2	FB + 0x08 + (FB_CONTROL_SIZE * 0x04) + (FB_CHANNEL_SIZE * 0x04 * c)	CH2_FIFO_DATA_CNT			
	FB + 0x0C + (FB_CONTROL_SIZE * 0x04) + (FB_CHANNEL_SIZE * 0x04 * c)	CH2_MAX_FIFO_SIZE			
	FB + 0x10 + (FB_CONTROL_SIZE * 0x04) + (FB_CHANNEL_SIZE * 0x04 * c)	CH2_PWM_WIDTH			
	FB + 0x14 + (FB_CONTROL_SIZE * 0x04) + (FB_CHANNEL_SIZE * 0x04 * c)	CH2_FIFO_ACCESS			
Channel 3	FB + 0x08 + (FB_CONTROL_SIZE * 0x04) + (FB_CHANNEL_SIZE * 0x04 * c)	CH3_FIFO_DATA_CNT			
	FB + 0x0C + (FB_CONTROL_SIZE * 0x04) + (FB_CHANNEL_SIZE * 0x04 * c)	CH3_MAX_FIFO_SIZE			
	FB + 0x10 + (FB_CONTROL_SIZE * 0x04) + (FB_CHANNEL_SIZE * 0x04 * c)	CH3_PWM_WIDTH			
	FB + 0x14 + (FB_CONTROL_SIZE * 0x04) + (FB_CHANNEL_SIZE * 0x04 * c)	CH3_FIFO_ACCESS			

6.6.1 FB_CONTROL_SIZE (READ -ONLY)

This register contains the total number of DWORDs register in control section of the function block. This can be used to calculate the offset for each register in a channel.

6.6.2 FB_CHANNEL_SIZE (READ-ONLY)

This register contains the total number of DWORDs registers in each channel of the function block. This can be used to calculate the offset for each register in a channel.

6.6.3 MODE_STATUS (READ/WRITE, READ-ONLY)

Selects the current mode of operation and indicates its triggering status.

B[3:0]: Mode

- 0x04: Uninitialized. This is the power-on state. No initialization has taken place. Running is stopped, and all counters are reset and the triggering state machine is reset. Transition to any of the other Modes will start converter initialization (conversions will not start until initialization is complete).
- 0x00: Reset. Running is stopped. All counters are reset and the triggering state machine is reset.
- 0x01: Paused. Running is stopped, but the counters and triggering state machine maintain their state.
- 0x02: Go, Single-Shot. After sending the Post-Stop values, running stops. The Mode must be set back to RESET in order to send more values.
- 0x03: Go, Re-arm. After sending the Post-Stop values and the FIFO is empty, the triggering state machine is restarted.

B[7:4]: Status

- 0x08: Uninitialized – The status when in the “Uninitialized” mode and the converter requires initialization.
- 0x09: Initializing
- 0x00: Stopped – The status when in the “Reset” mode, or in the “Uninitialized” mode and the converter does not require initialization.
- 0x01: Reserved
- 0x02: Waiting for start trigger
- 0x03: Running/Waiting for stop trigger
- 0x04: Reserved
- 0x05: Reserved
- 0x07: Done

6.6.4 CLK_SRC (READ/WRITE)

Selects the source for the clock of this function block.

Refer to Clock Source section on page 28, for list of valid values.

6.6.5 START_TRIG (READ/WRITE)

Selects the start trigger from the clock bus. CLK_DIV will start counting after the start trigger.

Refer to Clock Source section on page 28, for list of valid values.

6.6.6 STOP_TRIG (READ/WRITE)

Selects the stop trigger from the clock bus.

Refer to Clock Source section on page 28, for list of valid values.

6.6.7 CLK_DIV (READ/WRITE)

Divider for the pacer clock. Pacer Clock Frequency = (Clk_Src_Frequency) / (1 + CLK_DIV). If synchronizing with the pacer clock from another Function Block (by using one of the CLK_BUS signals), this is typically set to 0.

6.6.8 CLK_DIV_CNTR (READ ONLY)

The current value of the Clock Divide Counter. This counter starts at a value of CLK_DIV, and counts down. When it reaches zero, a conversion is made. This is useful when using a slow speed clock.

6.6.9 PERIOD_COUNT (READ ONLY)

Total number of periods executed. This does not increment in while in the "Waiting For Start Trigger" state. It also continues counting after a Re-Arm.

6.6.10 INT_ENA (MASKABLE READ/WRITE)

Each bit corresponds to an interrupt source. A value of '1' enables the source, and a value of '0' disables it. See below for a description of the sources.

6.6.11 INT_STAT (READ/CLEAR)

Each bit corresponds to an interrupt source. Reading a value of '1' indicates that an event has occurred. Reading a value of '0' indicates that the event has not occurred. Writing a '1' will clear that bit.

B0: Period – A period occurred.
B1: Reserved
B2: Reserved
B3: Start Trigger
B4: Stop Trigger
B5: Post-Stop Buffer Empty
B6: Reserved
B7: Pacer – The pacer clock has ticked.

6.6.12 CLK_BUSN

Selects a source to drive onto Clock Bus N. That clock bus can then be used by a different function block as a clock source or trigger.

A function block can drive multiple different Clock Buses. However, a Clock Bus N should not be driven by more than one function block at the same time or the clock signal will be undefined.

B[7:0]:

- 0x00: Disabled (Default)
- 0x80: Period – A period occurred.
- 0x81: Reserved
- 0x82: Reserved
- 0x83: Start Trigger
- 0x84: Stop Trigger
- 0x85: Post-Stop Buffer Empty
- 0x86: Reserved
- 0x87: Pacer – The pacer clock has ticked.

6.6.13 CHN_FIFO_DATA_CNT (READ)

This register shows the current sample count that is available in the PWM channel FIFO.

6.6.14 CHN_MAX_FIFO_SIZE (READ)

This register shows the max number of samples that the PWM channel FIFO can hold.

6.6.15 CHN_PWM_WIDTH (READ/WRITE)

Sets the width of output of the pulse width modulator. Pulse Width = (1 + CHn_PWM_WIDTH) / (System Clock). The width is defined as the time that the non-inverted output is high, and the inverted output is low.

The width register is checked at the beginning of every period. If the width register is modified in the middle of a period, the output will not be affected until the next period.

Note: that with Pacer Clock Frequency sets the maximum value for this register, a 100% duty cycle is not possible. This will result in the non-inverted output to be low, and the inverted output to be high.

The function block Mode needs to be set to: Go, Single-Shot or Go, Re-arm for the PWM to work. When Status is "Stopped", non-inverted output is low, and the inverted output is high.

6.6.16 CHN_FIFO_ACCESS (READ/WRITE)

This register provides direct access to the DMA FIFO. It can be used to access the data without the use of the DMA engine. The DMA engine for this channel must be set to "Pause." Each register access advances to the next sample.

6.7 BAR2: Incremental Encoder Functional Block

C in the register offset stands for channel number

Table 19: Incremental Encoder Functional Block

	Offset	0x03	0x02	0x01	0x00
Header	FB + 0x00	FB_ID			
	FB + 0x04	FB_DMA_BUFFERS	FB_DMA_CHANNELS	FB_CONTROL_SIZE	FB_CHANNEL_SIZE
Control	FB + 0x08	STOP_TRIG	START_TRIG	CLK_SRC	MODE_STATUS
	FB + 0x0C	CLK_DIV			
	FB + 0x10	CLK_DIV_CNTR			
	FB + 0x14	PRE_START_COUNT (limited by FIFO size)			
	FB + 0x18	POST_STOP_COUNT			
	FB + 0x1C	SAMPLE_CNT			
	FB + 0x20	INT_ENA (Sample, Start, Stop, Threshold, Pacer Tick, etc)			
	FB + 0x24	INT_STAT		*Reserved*	
	FB + 0x28	CLK_BUS3	CLK_BUS2	*Reserved*	*Reserved*
	FB + 0x2C	CLK_BUS7	CLK_BUS6	CLK_BUS5	CLK_BUS4
Channel 0	FB + 0x08 + (FB_CONTROL_SIZE * 0x04) + (FB_CHANNEL_SIZE * 0x04 * c)	MODE_CONFIG (Maskable register – 16-bit)			
	FB + 0x0C + (FB_CONTROL_SIZE * 0x04) + (FB_CHANNEL_SIZE * 0x04 * c)	FIFO_DATA_CNT			
	FB + 0x10 + (FB_CONTROL_SIZE * 0x04) + (FB_CHANNEL_SIZE * 0x04 * c)	MAX_FIFO_SIZE			
	FB + 0x14 + (FB_CONTROL_SIZE * 0x04) + (FB_CHANNEL_SIZE * 0x04 * c)	THRESH_LOW			
	FB + 0x1C + (FB_CONTROL_SIZE * 0x04) + (FB_CHANNEL_SIZE * 0x04 * c)	THRESH_HIGH			
	FB + 0x20 + (FB_CONTROL_SIZE * 0x04) + (FB_CHANNEL_SIZE * 0x04 * c)	INCENC_VALUE			
	FB + 0x24 + (FB_CONTROL_SIZE * 0x04) + (FB_CHANNEL_SIZE * 0x04 * c)	FIFO_ACCESS			

6.7.1 FB_CONTROL_SIZE (READ -ONLY)

This register contains the total number of DWORDs register in control section of the function block. This can be used to calculate the offset for each register in a channel.

6.7.2 FB_CHANNEL_SIZE (READ-ONLY)

This register contains the total number of DWORDs registers in each channel of the function block. This can be used to calculate the offset for each register in a channel.

6.7.3 MODE_STATUS (READ/WRITE, READ-ONLY)

Selects the current mode of operation and indicates its triggering status.

B[3:0]: Mode

- 0x04: Uninitialized. This is the power-on state. No converter initialization has taken place. Sampling is stopped, and all counters are reset and the triggering state machine is reset.

Transition to any of the other Modes will start converter initialization (sampling will not start until initialization is complete).

- 0x00: Reset. Sampling is stopped. All counters are reset and the triggering state machine is reset.
- 0x01: Paused. Sampling is stopped, but the counters and triggering state machine maintain their state.
- 0x02: Go, Single-Shot. After filling the buffer with the Post-Stop samples, capturing stops. The Mode must be set back to RESET in order to capture more samples.
- 0x03: Go, Re-arm. After filling the buffer with the Post-Stop samples and the FIFO is empty, the triggering state machine is restarted, i.e. FIFO is filled with Pre-Start samples and waits for a start trigger.

B[7:4]: Status

- 0x08: Uninitialized – The status when in the "Uninitialized" mode and the converter requires initialization.
- 0x09: Initializing
- 0x00: Stopped – The status when in the "Reset" mode, or in the "Uninitialized" mode and the converter does not require initialization.
- 0x01: Filling Pre-Trigger buffer
- 0x02: Waiting for start trigger
- 0x03: Sampling/Waiting for stop trigger
- 0x04: Filling Post-Stop buffer
- 0x05: Wait to re-arm – Waiting until local FIFO is empty so the pre-trigger buffer can be filled.
- 0x07: Done capturing

6.7.4 CLK_SRC (READ/WRITE)

Selects the source for the clock of this function block.

Refer to Clock Source section on page 28, for list of valid values.

6.7.5 START_TRIG (READ/WRITE)

Selects the start trigger from the clock bus. CLK_DIV will start counting after the start trigger, unless PRE_START_COUNT is non-zero in which case CLK_DIV will start counting immediately.

Refer to Clock Source section on page 28, for list of valid values.

6.7.6 STOP_TRIG (READ/WRITE)

Selects the stop trigger from the clock bus.

Refer to Clock Source section on page 28, for list of valid values.

6.7.7 CLK_DIV (READ/WRITE)

Divider for the pacer clock. Pacer Clock Frequency = (Clk_Src_Frequency) / (1 + CLK_DIV). If synchronizing with the pacer clock from another Function Block (by using one of the CLK_BUS signals), this is typically set to 0.

6.7.8 CLK_DIV_CNTR (READ ONLY)

The current value of the Clock Divide Counter. This counter starts at a value of CLK_DIV, and counts down. When it reaches zero, a sample is taken. This is useful when using a slow sample clock.

6.7.9 PRE_START_COUNT (READ/WRITE)

Number of samples to collect before the Start Trigger. The length is limited by the FIFO size – writing a value larger than the FIFO size will have indeterminate results.

6.7.10 POST_STOP_COUNT (READ/WRITE)

Number of samples to collect after the Stop Trigger.

6.7.11 SAMPLE_CNT (READ ONLY)

Total number of samples collected. This does not increment in while in the "Waiting For Start Trigger" state. It also continues counting after a Re-Arm.

6.7.12 INT_ENA (MASKABLE READ/WRITE)

Each bit corresponds to an interrupt source. A value of '1' enables the source, and a value of '0' disables it. See below for a description of the sources.

6.7.13 INT_STAT (READ/CLEAR)

Each bit corresponds to an interrupt source. Reading a value of '1' indicates that an event has occurred. Reading a value of '0' indicates that the event has not occurred. Writing a '1' will clear that bit.

B0: Sample – A sample has been taken.
B1: Reserved
B2: Pre-Start Buffer Filled
B3: Start Trigger
B4: Stop Trigger
B5: Post-Stop Buffer Filled
B6: Sampling has completed and the FIFO is empty (all data transferred to host)
B7: Pacer – The pacer clock has ticked
B8: Positive rollover - Indicates channel has transitioned from 0xFFFFFFFF to 0x00000000.
B9: Negative rollover - Indicates channel has transitioned from 0x00000000 to 0xFFFFFFFF.
B10: High threshold crossed.
B11: Low threshold crossed.

6.7.14 CLK_BUSN

Selects a source to drive onto Clock Bus N. That clock bus can then be used by a different function block as a clock source or trigger.

A function block can drive multiple different Clock Buses. However, a Clock Bus N should not be driven by more than one function block at the same time or the clock signal will be undefined.

B[7:0]:

- 0x00: Disabled (Default)
- 0x80: Sample – A sample has been taken.
- 0x81: Reserved
- 0x82: Pre-Start Buffer Filled
- 0x83: Start Trigger
- 0x84: Stop Trigger
- 0x85: Post-Stop Buffer Filled
- 0x86: Sampling has completed and the FIFO is empty (all data transferred to host)
- 0x87: Pacer – The pacer clock has ticked.
- 0x88: Positive rollover - Indicates channel has transitioned from 0xFFFFFFFF to 0x00000000.
- 0x89: Negative rollover - Indicates channel has transitioned from 0x00000000 to 0xFFFFFFFF.
- 0x8A: High threshold crossed.
- 0x8B: Low threshold crossed.

6.7.15 MODE_CONFIG (MASKABLE READ/WRITE)

This register provides configuration for Incremental Encoder.

B[12]: Differential: Selects single-ended or differential mode

- '0' = Single Ended. Only "+" inputs are used

- '1' = Pseudo-Differential

B[11]: Reserved
B[10]: Index: When enabled, a high input on the Index input clears the counter
- '0' = Index Inputs Disabled
- '1' = Index Input Enabled

B[9]: Hold: When enabled, the encoder continues counting in the background, but the INCENC_VALUE register remains constant.
- '0' = INCENC_VALUE registers are not held
- '1' = INCENC_VALUE registers are held

B[8]: Enable: Enable for this incremental encoder
- '0' = Encoder is disabled
- '1' = Encoder is enabled

B[7:0]: Phase Filter: Selects if a particular state transition will cause the encoder counter to change.
For each bit:
- '0' = Transition will change counter
- '1' = Transition will not change counter.

The bit assignments for the transitions are:

Bit	Previous State [B:A]	Current State [B:A]	Direction
7	00	10	Down
6	10	11	Down
5	11	01	Down
4	01	00	Down
3	10	00	Up
2	11	10	Up
1	01	11	Up
0	00	01	Up

6.7.16 FIFO_DATA_CNT (READ)

This register shows the current sample count that is available in the Incremental Encoder FIFO.

6.7.17 MAX_FIFO_SIZE (READ)

This register shows the max number of samples that the FIFO can hold.

6.7.18 THRESH_LOW (READ/WRITE)

Unsigned 32-bit value indicating the low threshold. If the count value goes below the threshold value, an interrupt will be generated. The interrupt will only be generated once the threshold value is crossed.

6.7.19 THRESH_HIGH (READ/WRITE)

Unsigned 32-bit value indicating the high threshold. If the count value goes above the threshold value, an interrupt will be generated. The interrupt will only be generated once the threshold value is crossed.

6.7.20 INCENC_VALUE (READ/WRITE)

The unsigned 32-bit current value of the incremental encoder channel. This is the same value that is written to the DMA FIFO.

This register does not update when MODE_CONFIG[HOLD] = 1.

This register can only be written to when MODE_CONFIG[Enable] = 0. This allows the counter to be pre-loaded with a known position value.

6.7.21 FIFO_ACCESS (READ)

This register provides direct access to the DMA FIFO. It can be used to access the data without the use of the DMA engine. The DMA engine for this channel must be set to "Pause." Each register access advances to the next sample.

If the MODE_CONFIG[HOLD] = 1, the last value before HOLD was enable will be written to the FIFO on each clock pulse, until HOLD is disabled.

6.8 BAR2: External Clocking Functional Block

This function block provides an interface to the External Clocking. It is used to input a clock onto a Clock Bus N, or to output a Clock Bus N.

Table 20:External Clocking Functional Block

	Offset	0x03	0x02	0x01	0x00
Header	FB + 0x00	FB_ID			
	FB + 0x04	FB_DMA_BUFFERS	FB_DMA_CHANNELS	*Reserved*	*Reserved*
	FB + 0x08	EXT_CLK_EDGE	EXT_CLK_DIR	EXT_CLK_GATE_IN	EXT_CLK_IN
	FB + 0x0C	EXT_CLK2_PW			
	FB + 0x10	EXT_CLK3_PW			
	FB + 0x14	EXT_CLK4_PW			
	FB + 0x18	EXT_CLK5_PW			
	FB + 0x1C	EXT_CLK6_PW			
	FB + 0x20	EXT_CLK7_PW			
	FB + 0x24	EXT_CLK3_CFG	EXT_CLK2_CFG	*Reserved*	*Reserved*
	FB + 0x28	EXT_CLK7_CFG	EXT_CLK6_CFG	EXT_CLK5_CFG	EXT_CLK4_CFG

6.8.1 EXT_CLK_IN (READ-ONLY)

This register provides the current value on the External Clocking lines. The bits in the register correspond with the External Clocking pins as follows:

Bit	CN3 + Pin Number	CN3 - Pin Number	CLK_BUS	Signal
5	11	12	7	EXT_CLK_7
4	9	10	6	EXT_CLK_6
3	7	8	5	EXT_CLK_5
2	5	6	4	EXT_CLK_4
1	3	4	3	EXT_CLK_3
0	1	2	2	EXT_CLK_2

6.8.2 EXT_CLK_GATE_IN (READ ONLY)

This register provides the current value on the External Clocking Gates lines. External Clocking Gates can only be used when inputting an external clock. The bits in the register correspond with the External Clocking Gates pins as follows:

Bit	CN3 + Pin Number	CN3 - Pin Number	CLK_BUS	Signal
5	23	24	7	EXT_CLK_GATE7
4	21	22	6	EXT_CLK_GATE6
3	19	20	5	EXT_CLK_GATE5
2	17	18	4	EXT_CLK_GATE4
1	15	16	3	EXT_CLK_GATE3
0	13	14	2	EXT_CLK_GATE2

6.8.3 EXT_CLK_DIR (READ/WRITE)

Selects the direction of the External Clocking bits. 0=input, 1=output.

All pins default to inputs at power-up.

6.8.4 EXT_CLK_EDGE (READ/WRITE)

Selects which edge detect to trigger on. This is a bit settable register.

0= Rising Edge Detect, 1= Fall Edge Detect.

6.8.5 EXT_CLKN_PW (READ/WRITE)

This register is used to increase the pulse width of the clock when it is set as an output.

$$Pulse\ Width\ =\ (1\ +\ EXT_CLKn_PW)\ /\ (System\ Clock)$$



NOTE: If EXT_CLKn_PW is set to be wider than the EXT_CLKn_CFG Clock Frequency the signal will just stay high.

6.8.6 EXT_CLKN_CFG (READ/WRITE)

Selects clocking method.

B[7:0]:

0x00: Disables External Clocking.

0x80: Not Gated: CLK_BUSn will be inputted/outputted independent of the EXT_CLK_GATEn corresponding gate value.

0x81: Clock Gated (High): CLK_BUSn will be inputted when the EXT_CLK_GATEn corresponding gate value is high, this doesn't affect when outputting a clock.

0x82: Clock Gated (Low): CLK_BUSn will be inputted when the EXT_CLK_GATEn corresponding gate value is low, this doesn't affect when outputting a clock.

6.9 Programmable Clock Functional Block

Table 21: Programmable Clock Functional Block

	Offset	0x03	0x02	0x01	0x00
Header	FB + 0x00	FB_ID			
	FB + 0x04	FB_DMA_BUFFERS	FB_DMA_CHANNELS	FB_CONTROL_SIZE	FB_CHANNEL_SIZE
Control	FB + 0x08	STOP_TRIG	START_TRIG	CLK_SRC	MODE_STATUS
	FB + 0x0C	CLK_DIV			
	FB + 0x10	CLK_DIV_CNTR			
	FB + 0x14	Reserved			
	FB + 0x18	Reserved			
	FB + 0x1C	COUNT			
	FB + 0x20	INT_ENA (Conversion, Start, Stop, Error, Channel)			
	FB + 0x24	INT_STAT		*Reserved*	
	FB + 0x28	CLK_BUS3	CLK_BUS2		*Reserved*
	FB + 0x2C	CLK_BUS7	CLK_BUS6	CLK_BUS5	CLK_BUS4
	FB + 0x30	CONFIG (Maskable register – 16-bit)			
	FB + 0x08 + (FB_CONTROL_SIZE * 0x04) + (FB_CHANNEL_SIZE * 0x04 * c)	CH0_FIFO_DATA_CNT			
	FB + 0x0C + (FB_CONTROL_SIZE * 0x04) + (FB_CHANNEL_SIZE * 0x04 * c)	CH0_MAX_FIFO_SIZE			
	FB + 0x10 + (FB_CONTROL_SIZE * 0x04) + (FB_CHANNEL_SIZE * 0x04 * c)	INTERVAL_CLK_DIV			
	FB + 0x14 + (FB_CONTROL_SIZE * 0x04) + (FB_CHANNEL_SIZE * 0x04 * c)	LAST_INTERVAL_COUNT			

	Offset	0x03	0x02	0x01	0x00
	FB + 0x1C + (FB_CONTROL_SIZE * 0x04) + (FB_CHANNEL_SIZE * 0x04 * c)	CH0_FIFO_ACCESS			

6.9.1 FB_CONTROL_SIZE (READ -ONLY)

This register contains the total number of DWORDs register in control section of the function block. This can be used to calculate the offset for each register in a channel.

6.9.2 FB_CHANNEL_SIZE (READ-ONLY)

This register contains the total number of DWORDs registers in each channel of the function block. This can be used to calculate the offset for each register in a channel.

6.9.3 MODE_STATUS (READ/WRITE, READ-ONLY)

Selects the current mode of operation and indicates its triggering status.

B[3:0]: Mode

- 0x04: Uninitialized. This is the power-on state. No converter initialization has taken place. Sampling is stopped, and all counters are reset and the triggering state machine is reset. Transition to any of the other Modes will start converter initialization (sampling will not start until initialization is complete).
- 0x00: Reset. Counting is stopped. All counters are reset and the triggering state machine is reset.
- 0x01: Paused. Counting is stopped, but the counters and triggering state machine maintain their state.
- 0x02: Go, Single-Shot. After filling the buffer with the Post-Stop samples, capturing stops. The Mode must be set back to RESET in order to capture more samples.
- 0x03: Go, Re-arm. After filling the buffer with the Post-Stop samples and the FIFO is empty, the triggering state machine is restarted, i.e. FIFO is filled with Pre-Start samples and waits for a start trigger.

B[7:4]: Status

- 0x08: Uninitialized – The status when in the “Uninitialized” mode and the converter requires initialization.
- 0x09: Initializing
- 0x00: Stopped – The status when in the “Reset” mode, or in the “Uninitialized” mode and the converter does not require initialization.
- 0x01: Reserved
- 0x02: Waiting for start trigger
- 0x03: Running/Waiting for stop trigger
- 0x04: Reserved
- 0x05: Reserved
- 0x07: Done capturing

6.9.4 CLK_SRC (READ/WRITE)

Selects the source for the clock of this function block.

Refer to Clock Source section on page 28, for list of valid values.

6.9.5 START_TRIG (READ/WRITE)

Selects the start trigger from the clock bus. CLK_DIV will start counting after the start trigger, unless PRE_START_COUNT is non-zero in which case CLK_DIV will start counting immediately.

Refer to Clock Source section on page 28, for list of valid values.

6.9.6 STOP_TRIG (READ/WRITE)

Selects the stop trigger from the clock bus.

Refer to Clock Source section on page 28, for list of valid values.

6.9.7 CLK_DIV (READ/WRITE)

Divider for the pacer clock. Pacer Clock Frequency = (Clk_Src_Frequency) / (1 + CLK_DIV). If synchronizing with the pacer clock from another Function Block (by using one of the CLK_BUS signals), this is typically set to 0.

6.9.8 CLK_DIV_CNTR (READ ONLY)

The current value of the Clock Divide Counter. This counter starts at a value of CLK_DIV, and counts down. When it reaches zero, a sample is taken. This is useful when using a slow sample clock.

6.9.9 COUNT (READ ONLY)

Total number of events that have occurred. This does not increment in while in the "Waiting For Start Trigger" state. It also continues counting after a Re-Arm.

When MODE_CONFIG(1) = '1' this register will reset this register to 0 every interval clock pulse.

6.9.10 INT_ENA (MASKABLE READ/WRITE)

Each bit corresponds to an interrupt source. A value of '1' enables the source, and a value of '0' disables it. See below for a description of the sources.

6.9.11 INT_STAT (READ/CLEAR)

Each bit corresponds to an interrupt source. Reading a value of '1' indicates that an event has occurred. Reading a value of '0' indicates that the event has not occurred. Writing a '1' will clear that bit.

B0: Count – The count has changed.
B1: Interval Counter – Interval count has been reached
B2: Reserved
B3: Start Trigger
B4: Stop Trigger
B5: Reserved
B6: Reserved
B7: Reserved

6.9.12 CLK_BUSN

Selects a source to drive onto Clock Bus N. That clock bus can then be used by a different function block as a clock source or trigger.

A function block can drive multiple different Clock Buses. However, a Clock Bus N should not be driven by more than one function block at the same time or the clock signal will be undefined.

B[7:0]:

- 0x00: Disabled (Default)
- 0x80: Count – The count has changed.
- 0x81: Interval Counter – Interval count has been reached.
- 0x82: Reserved
- 0x83: Start Trigger
- 0x84: Stop Trigger
- 0x85: Reserved
- 0x86: Reserved
- 0x87: Reserved

6.9.13 MODE_CONFIG (MASKABLE READ/WRITE)

This register provides configuration for Programmable Clock function block.

B[1]: Interval Count Reset: When enabled this causes the Count register to reset to 0 every interval clock pulse.
- '0' = Interval Count Reset is disabled
- '1' = Interval Count Reset is enabled

B[0]: Interval Counter Enable: Used to enable/disable the interval counter.
- '0' = Interval Counter is disabled
- '1' = Interval Counter is enabled

6.9.14 CHN_FIFO_DATA_CNT (READ)

This register shows the current sample count that is available in the interval counter FIFO.

6.9.15 CHN_MAX_FIFO_SIZE (READ)

This register shows the max number of samples that the interval counter FIFO can hold.

6.9.16 INTERVAL_CLK_DIV (READ/WRITE)

Divider for the interval clock. Interval Clock Frequency = (System Clock) / (1 + INTERVAL_CLK_DIV). When the interval counter is enabled, every interval clock pulse will update the LAST_INTERVAL_COUNT and generate an interval counter interrupt.

6.9.17 LAST_INTERVAL_COUNT (READ-ONLY)

The last count value taken from the COUNT register. This is the same value that is written to the DMA FIFO.

6.9.18 CHN_FIFO_ACCESS (READ/WRITE)

This register provides direct access to the DMA FIFO. It can be used to access the data without the use of the DMA engine. The DMA engine for this channel must be set to "Pause." Each register access advances to the next sample.

7 Troubleshooting

If you are having problems with your system, please try the following initial steps:

- **Simplify the System** – Remove modules one at a time from your system to see if there is a specific module that is causing a problem. Perform you troubleshooting with the least number of modules in the system possible.

- **Swap Components** – Try replacing parts in the system one at a time with similar parts to determine if a part is faulty or if a type of part is configured incorrectly.

If problems persist, or you have questions about configuring this product, contact RTD Embedded Technologies via the following methods:

Phone: +1-814-234-8087
E-Mail: techsupport@rtd.com

Be sure to check the RTD web site (http://www.rtd.com) frequently for product updates, including newer versions of the board manual and application software.

8 Additional Information

8.1 PC/104 Specifications

A copy of the latest PC/104 specifications can be found on the webpage for the PC/104 Embedded Consortium:

www.pc104.org

8.2 PCI and PCI Express Specification

A copy of the latest PCI and PCI Express specifications can be found on the webpage for the PCI Special Interest Group:

www.pcisig.com

9 Limited Warranty

RTD Embedded Technologies, Inc. warrants the hardware and software products it manufactures and produces to be free from defects in materials and workmanship for one year following the date of shipment from RTD Embedded Technologies, Inc. This warranty is limited to the original purchaser of product and is not transferable.

During the one year warranty period, RTD Embedded Technologies will repair or replace, at its option, any defective products or parts at no additional charge, provided that the product is returned, shipping prepaid, to RTD Embedded Technologies. All replaced parts and products become the property of RTD Embedded Technologies. Before returning any product for repair, customers are required to contact the factory for a Return Material Authorization (RMA) number.

This limited warranty does not extend to any products which have been damaged as a result of accident, misuse, abuse (such as: use of incorrect input voltages, improper or insufficient ventilation, failure to follow the operating instructions that are provided by RTD Embedded Technologies, “acts of God” or other contingencies beyond the control of RTD Embedded Technologies), or as a result of service or modification by anyone other than RTD Embedded Technologies. Except as expressly set forth above, no other warranties are expressed or implied, including, but not limited to, any implied warranties of merchantability and fitness for a particular purpose, and RTD Embedded Technologies expressly disclaims all warranties not stated herein. All implied warranties, including implied warranties for merchantability and fitness for a particular purpose, are limited to the duration of this warranty. In the event the product is not free from defects as warranted above, the purchaser's sole remedy shall be repair or replacement as provided above. Under no circumstances will RTD Embedded Technologies be liable to the purchaser or any user for any damages, including any incidental or consequential damages, expenses, lost profits, lost savings, or other damages arising out of the use or inability to use the product.

Some states do not allow the exclusion or limitation of incidental or consequential damages for consumer products, and some states do not allow limitations on how long an implied warranty lasts, so the above limitations or exclusions may not apply to you.

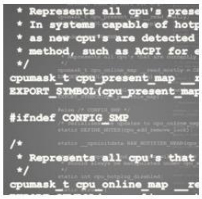
This warranty gives you specific legal rights, and you may also have other rights which vary from state to state.

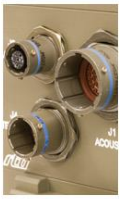
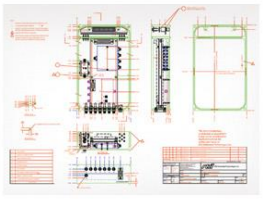
RTD Embedded Technologies, Inc.
103 Innovation Boulevard
State College, PA 16803 USA
Telephone: 814-234-8087
Fax: 814-234-5218

www.rtd.com

sales@rtd.com
techsupport@rtd.com






Copyright 2020 by RTD Embedded Technologies, Inc. All rights reserved.

Artisan Technology Group is an independent supplier of quality pre-owned equipment

Gold-standard solutions

Extend the life of your critical industrial, commercial, and military systems with our superior service and support.

We buy equipment

Planning to upgrade your current equipment? Have surplus equipment taking up shelf space? We'll give it a new home.

Learn more!

Visit us at **artisantg.com** for more info on price quotes, drivers, technical specifications, manuals, and documentation.

Artisan Scientific Corporation dba Artisan Technology Group is not an affiliate, representative, or authorized distributor for any manufacturer listed herein.

We're here to make your life easier. How can we help you today?

(217) 352-9330 | sales@artisantg.com | artisantg.com

ARTISAN® TECHNOLOGY GROUP