Cavium Networks NITROX® PX CN16XX Processor Family Hardware Manual

Contents of this document are subject to change without notice

CN16XX-HM-1.1

Cavium Networks Proprietary and Confidential DO NOT COPY

May 2009

PUBLISHED BY
Cavium Networks
805 East Middlefield Road
Mountain View, CA 94043
Phone: 650-623-7000
Fax: 650-625-9751
Email: sales@caviumnetworks.com
Web: http://www.caviumnetworks.com

© 2003-2009 by Cavium Networks

All rights reserved. No part of this manual may be reproduced in any form, or transmitted by any means, without the written permission of Cavium Networks.

Cavium Networks makes no warranty about the use of its products, and reserves the right to change this document at any time, without notice. Whereas great care has been taken in the preparation of this manual, Cavium Networks, the publisher, and the authors assume no responsibility for errors or omissions.

NITROX® is a registered trademark, and NITROX® PX is a trademark of Cavium Networks in the United States, or other countries, or both.

All other trademarks or service marks referred to in this manual are the property of their respective owners.

Table of Contents

List of Figures

List of Tables

Preface

This document describes the Cavium Networks NITROX® PX CN16XX Processor Family, the next generation security processor from Cavium Networks, providing a PCI Express interface, as well as introducing new data and security processing capabilities to the NITROX® Line.

The NITROX PX line also includes the Cavium Networks NITROX® PX CN15XX Processor Family, which provides new data and security processing capabilities to users of the NITROX line in a package pin-compatible with the NITROX 10XX processor family.

This Introduction provides the following information about this document:

Content Overview

This manual consists of the following parts:

Audience

- This document is intended to be of use to engineers, systems designers and managers tasked with using the Cavium Networks NITROX® PX CN16XX Processor Family of processors to design systems..

Revision History

The revision history is shown in Table 1.

Table 1 Revision History

Version	Date	Changes
1.1	5/2009	• in Section 2.2.5, corrected the reset value of EEPROM byte 35, and corrected the file names of bytes 40 and 41 • in Section 3.4.1, added a new step 3 to soft-reset initialization procedure • in Table 4–1, added resistor values to EXP_BAND_RES and EXP_COMP_RES. • in Table 4–3, modified the Min level of V_{HIGH} • in Section 4.3.3, added a delay time to the power-down note • in Table 4–12, removed power numbers for one-core part and added numbers for six-core part • in Section 5.2.1, corrected the description of the REQ-PFIX field of the COMMAND register

Prerequisites

- This manual and related documentation from Cavium Networks strives to be as comprehensive and easy-to-use as possible. However, we try not to reproduce information that almost all of our readers will already be familiar with. As such these documents assume that the reader has a basic understanding of
- The concepts of asymmetric (public/private key) and symmetric (secret key) cryptography.
- The SSL protocol and its related terminology. One important reference in this regard is RFC.#2246 from the Internet Engineering Task Force (IETF) and updates to it.

Related Cavium Networks Documentation

Other documents from Cavium Networks that contain information useful to you as a user of this manual and related Cavium Networks products include:

- NITROX® PX Family Hardware Manuals
- NITROX® and NITROX® PX Family Software Manuals
- NITROX® PX Instruction Set (Microcode) Manuals
- NITROX® PX Software Architecture and Programmer's Guide
- NITROX® PX Macro Instruction API

Reference Documentation

Documents from other sources that contain information that might be useful to you as a user of this manual and related Cavium Networks products include:

- All relevant Internet Engineering Task Force (IETF) RFCs, including RFC #2246 and updates to it.
- PCI Express Base Specification, V1.1.

Getting Support from Cavium Networks

For questions and comments regarding this or any other Cavium Networks product, you can reach us via our website at http://www.caviumnetworks.com.

Chapter 1

Introduction to the NITROX® PX CN16XX Family

This chapter introduces the Cavium Networks NITROX® PX CN16XX Processor family, the next generation security processor from Cavium Networks. The CN16XX provides a PCI Express interface, as well as introducing new data and security processing capabilities to the NITROX® Line. The NITROX PX line also includes the Cavium Networks NITROX® PX CN15XX Processor family, which provides new data and security processing capabilities to users of the NITROX line in a package pin-compatible with the NITROX 10XX processor family. The NITROX PX family products are available with varying number of GigaCypher cores in order to offer a wide range of software compatible performance choices. The NITROX PX family comes in two-core, four-core, six-core and eight-core versions. All processors operate at 350 MHz except the eight-core version, which operates up to 400 MHz.

This chapter has the following sections:

1.1 Overview

As mentioned above, the Cavium Networks NITROX® PX CN16XX Processor Family is the next generation security processor from Cavium Networks. The NITROX® PX CN16XX offers a PCI Express interface (with up to 4-Lane) as well as new data processing capabilities (such as support for AES-GCM, KASUMI, and SHA-256/384/512 algorithms and up to 4096-bit modular exponentiation), besides those already available in Cavium Networks' NITROX Family of processors.

Cavium Networks NITROX® PX CN16XX Processor Family devices perform modular exponentiation, random number generation, and hash processing. It also executes protocol-specific complex instructions to support the SSL/TLS or IPsec/ IKE security protocols.

The NITROX Family of processors is the industry's first processor exclusively targeted towards IPsec, SSL, Wireless LAN (WLAN), and 3G Wireless applications protocol acceleration rather than ordinary cipher acceleration. NITROX Family macro processors are custom designed and provide much better performance-to-area/cost/power ratio when compared to ASIC-based security chips. These processors are optimized for SSL, IPsec, WLAN acceleration, or 3G Wireless by loading required microcode binaries available from Cavium Networks. The detailed instruction sets for the processors are available as separate documents.

The heart of a NITROX® PX CN16XX processor is its eight micro-programmed GigaCipher Processor Cores. The cores provide design flexibility by simultaneously supporting cryptographic operations and accelerating protocol functions. Each core supports Macro processing and allows future upgrades.

Macro processing is unique to NITROX® PX CN16XX processors and allows systems to offload high-level SSL or IPsec protocol commands that reduce the host I/O traffic and system processor to increase the total system throughput. This also frees system processor resources for other functions, increasing overall system performance.

The GigaCipher Processor Cores, or execution units, provide powerful adaptive processing capability, a unique feature of the NITROX technology, which provides flexible system response to dynamic loads. Dynamic Adaptive processing is enabled by the GigaCipher's ability to accelerate both the asymmetric algorithms used for tunnel establishment and the symmetric ciphers + hashing algorithms used in bulk data encryption. This allows vendors to build balanced systems that can handle dynamic traffic conditions. For example, a VPN gateway with 10,000 established IPsec tunnels requires high bulk data encryption/ decryption with only a few hundred IKE sessions/sec. However, in the event of a VPN gateway failure, the back-up gateway must re-establish these tunnels, before resuming the bulk encryption/decryption operations, thus requiring 10,000 IKE sessions/sec. The design flexibility afforded by the NITROX® PX CN16XX allows it to seamlessly adapt to either condition.

1.2 Features

Cavium Networks NITROX® PX CN16XX Processor Family offers

- High scalability with both hardware and software scalability
  - From 100 Mbps - 2.5 Gbps
  - 200 Mbps, true-random number generator
- High flexibility with a variety of protocols and interfaces
  - AES-GCM, KASUMI, and SHA-256/384/512, in addition to the algorithms mentioned below as part of NITROX technology
  - PCI Express interface, four lanes
- Highest performance Public Key Processor
  - Up to 32K 180-bit Diffie Hellman (groups 1, 2, 5, 14, 15, 16)
  - Up to 17K 1024-bit RSA operations/second
- Highest performance Bulk Data Encryption
  - Up to 2.5 Gbps Bulk Data Encryption + Hashing (SSL, IPSec, or CCMP)
- The NITROX® PX CN16XX Family also provides current users of earlier generations of Cavium Networks products a migration path that allows them to adopt the PCI Express standard without the need for major redesign of existing software and hardware. It supports up to a four-lane PCI Express link (i.e. support for one, two, or four lanes).
- Hardware support for simultaneous use of multiple protocols (via four execution unit groups)
- Low power
  - Less than 4W

The Cavium Networks NITROX® PX CN16XX Processor Family provides the following enhanced options beyond NITROX technology to further help with design and speed development and time-to-market:

- Microcode store can now hold 12K instructions

The Cavium Networks NITROX® PX CN16XX Processor Family also delivers the powerful feature set of the underlying NITROX technology

- The world's first Security Macro Processor developed using custom CPU design techniques
  - Single chip solution that accelerates all cryptographic operations and the SSL, IPsec / IKE, and CCMP protocols
- Multi-Algorithm Support
  - RSA and Diffie-Hellman (groups 1,2,5)
  - DES/3DES, AES, ARC4
  - MD5, SHA-1, HMAC-MD5, HMAC-SHA-1
- High number of concurrent IPsec SAs or SSL Contexts supported in Host Memory
- On-Chip, true random number generator
- High performance native interfaces
  - PCI Express four-lane (Target mode)
- Small footprint
  - The CN16XX family of processors comes in a 233 PBGA package.

1.3 Product Selection Guide

Note: *Exact part numbers and package definition is in the process of being finalized. The interface configuration is subject to revision.*

Part Number	Number of Cores	Bus	Target Application Performance		Package
			MAX RSA 1024-bit Exponent	Full IPsec or SSL Processing Throughput Mbps (with AES + SHA)	
CN1605	2	PCI Express x4	4K	500 Mbps	233 BGA
CN1610	4		8K	1.0 Gbps	
CN1615	6		13K	1.5 Gbps	
CN1620	8		17K	2.5 Gbps	

Chapter 2

NITROX® PX CN16XX Family Architecture

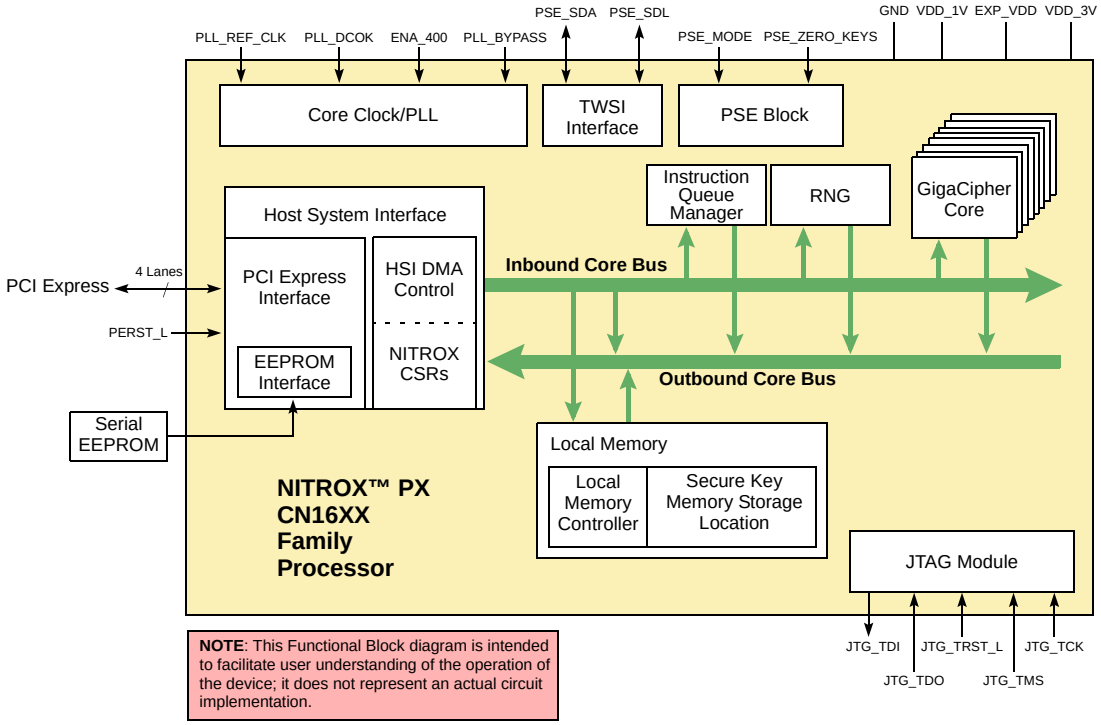
This chapter presents the block diagram of the Cavium Networks NITROX® PX CN16XX Processor. Detailed descriptions are provided for each major architectural block within the device. This includes descriptions of primary block functionality, block inputs and outputs, connectivity to other blocks in the chip and/or to the external chip interfaces, and a brief description of the configuration, control, and/or status registers for the block. This chapter should provide you with a fundamental understanding of each functional block of the device, and a basic understanding of how these blocks interact with each other and the outside world during device operation.

This chapter has the following sections:

2.1 Block-Level Architecture

The diagram below shows CN16XX functional blocks.

Figure 2–1 Functional Block Diagram for a CN16XX Family Processor



The following sections describe the funtional blocks shown in Figure 2–1.

2.2 Host System Interface Block

The Host System Interface (HSI) Port is the primary interface between CN16XX internal blocks and the host system that the processor is designed into. The Host System Interface block provides the following functions:

- Complete four-lane PCI Express ports, including control logic and CSRs.
- DMA control for all DMA transfers between the host system and the Instruction Queue Manager (IQM), and GigaCipher Processor Cores. CN16XX acts as a DMA master for all DMA operations.
- CN16XX internal configuration and status registers. The Host System Interface block implements these registers, and they are read/write-able from the Host System Interface port.
- “Any-Endian” support/configuration logic.

2.2.1 PCI Express Interface

The PCI Express interface consists of sets of differential pairs, each comprising a Lane for signaling. The NITROX® PX CN16XX family of products supports up to four Lanes. The interface operates at 2.5 Gigabits per second per Lane per direction.

NITROX® PX CN16XX devices implement a complete PCI Express-compliant port. This includes all physical interface pins and circuitry, as well as all PCI Express configuration/status registers. Although basic PCI Express register descriptions are included in this document, the reader is encouraged to refer to the PCI Express Base specifications V1.1 for a description of the PCI Express architecture and links.

For a list and description of PCI Express configuration/status registers, refer to Chapter 5, "Registers". For a list and description of PCI Express signals, refer to Section 4.1, "Signal Descriptions".

2.2.2 Host System Interface DMA Controller

Decodes, stores, and executes DMA commands targeted to the Host System Interface from the Outbound Core Bus. This provides the mechanism for CN16XX-controlled movement of data between the host system memory and the IQM, and GigaCipher Processor Cores.

An asynchronous outbound DMA Command FIFO buffers DMA commands received from the Outbound Core Bus. DMA Commands identify the initiator and target of the DMA operation, DMA Type (read or write), DMA address, DMA size, and DMA Data (DMA write only). An asynchronous inbound DMA read data FIFO buffers data received on the Host System Interface signal pins, and destined for the Inbound Core Bus during DMA reads.

DMA Commands received by the Host System Interface outbound FIFO are executed on the Host System Interface bus in the order received, however split-transaction DMA reads do not block subsequent DMA read or write operations. The DMA controller keeps track of split transactions in order to insure that split read data is returned to the correct destination on the Inbound Core Bus.

Host System Interface master-mode DMA operations can burst from 1 to 2048 bytes per transaction. However, DMA operations are issued under microcode control, so the maximum DMA size may be controlled and thus reduced by microcode.

Also under microcode control, CN16XX hardware can perform either Normal or Scatter/Gather mode DMA operations. For more information about DMA modes and sizes, please refer to the one of the Instruction Set manuals listed in the appendix of this document.

2.2.3 Configuration/Status Registers

Implements all configuration and status registers in the device (not including PCIe configuration space registers. These registers are part of the PCIe interface described in Section 2.2.1). Target memory read and write operations on the Host System Interface are decoded to generate read or write operations to these registers. These registers are offset from the address in the BAR0 register in PCIe configuration register space. PCIe configuration must be performed before accessing these BAR0 registers.

All NITROX® PX CN16XX Configuration and Status registers are four bytes wide, and are addressed on eight-byte boundaries. Therefore, all PCIe target-mode transfers are single-register transactions aligned to eight-byte boundaries, with valid data placed on bytes 3 to 0.

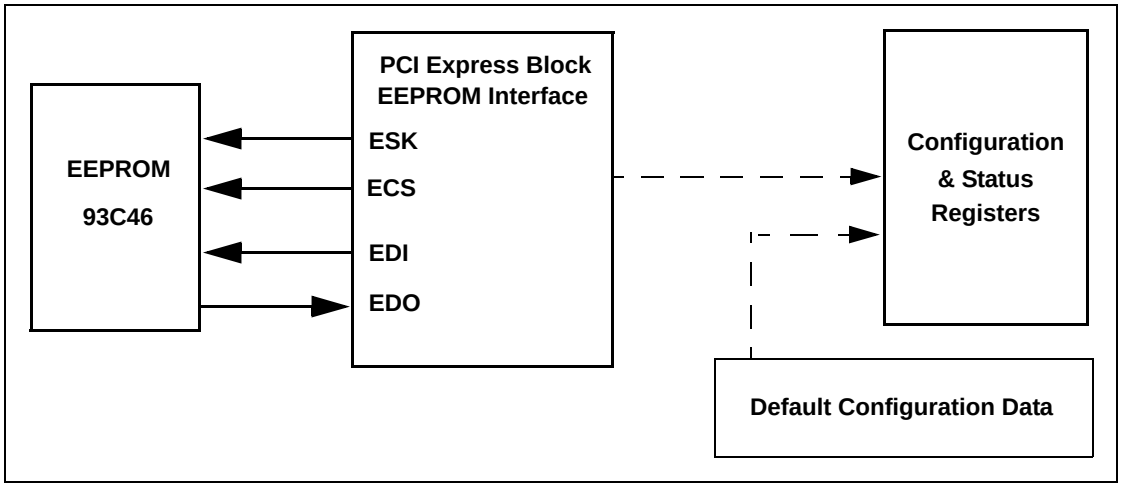
2.2.4 “Any-Endian” Support/Configuration Logic

Because host systems may be implemented big or little Endian, the Host System Interface block also includes “Any-Endian” data swapping logic that will byte-swap between the Host System Interface and the rest of the CN16XX core, in several configuration variations. For a detailed description of available Endian configuration variations, refer to the Inbound_Data_Swap and Outbound_Data_Swap register fields of the Command/Status Register. See Section 5.2, “Detailed Register Descriptions”.

2.2.5 PCI Express Configuration EEPROM Interface

The PCI Express block provides an interface to an optional external serial EEPROM, used to store configuration data. Upon CN16XX power-up, system configuration data is read from the EEPROM into the configuration registers. The presence of this external EEPROM is detected automatically; if no EEPROM is connected, default values for configuration space will be used as described in Section 5.2, “Detailed Register Descriptions”. and the ID fields will be set to their default values. The configuration registers are read-only from the perspective of the Host CPU.

Figure 2–2 Configuration EEPROM Operation



Once the presence of the external EEPROM is detected by the EEPROM controller in the Host System Interface block, an address stream is sent to the EEPROM. The EEPROM responds with a start bit (0), followed by configuration data.

The EEPROM serial clock signal (ESK) from the interface to the EEPROM runs until loading of data from the EEPROM is complete. It then automatically shuts down to save power, and does not start again unless CN16XX is reset.

The EEPROM interface supports a standard type 93C46 (64k x 16) serial EEPROM. Devices of this type can be obtained from several semiconductor vendors.

Table 2–1 PCIe Configuration EEPROM Field Map

EEPROM Word	PCIe Config Address	Reset Value	PCIe Config Register Name	Description
0	0x000	0x177D	Vendor ID[15:0]	Vendor ID, bits[15:0]
1		0x0010	Device ID[15:0]	Device ID, bits[31:16]
2	0x008	0x0000	Class Code /Revision ID	Revision ID[7:0], bits[7:0]; Class Code[7:0], bits[15:8]
3		0x1000		Class Code[23:8], bits[31:16]
4	0x00C	0x0000	BIST/Header Type/ Latency Timer/ Cache Line Size	Cache Line Size, bits[7:0]; RAZ, bits[15:8]
5		0x0000		Header Type[7:0], bits[23:16]; BIST Code[3:0], bits[27:24]; Reserved, bits[29:28]; BIST Request/Busy, bit 30; Reserved, bit 31
6	0x010	0x0000	Reserved	Not supported. Must be 0x0000
7		0x0000		Not supported. Must be 0x0000
8	0x014	0x0000	Reserved	Not supported. Must be 0x0000
9		0x0000		Not supported. Must be 0x0000
10	0x018	0x0000	Reserved	Not supported. Must be 0x0000
11		0x0000		Not supported. Must be 0x0000
12	0x01C	0x0000	Reserved	Not supported. Must be 0x0000
13		0x0000		Not supported. Must be 0x0000
14	0x020	0x0000	Reserved	Not supported. Must be 0x0000
15		0x0000		Not supported. Must be 0x0000
16	0x024	0x0000	Reserved	Not supported. Must be 0x0000
17		0x0000		Not supported. Must be 0x0000
18	0x028	0x0000	Reserved	Not supported. Must be 0x0000
19		0x0000		Not supported. Must be 0x0000
20	0x02C	0x177D	Subsystem Vendor ID[15:0]	Subsystem Vendor ID, bits[15:0]
21		0x0001	Subsystem ID[15:0]	Subsystem ID, bits[31:16]
22	0x030	0x0000	Reserved	Not supported. Must be 0x0000
23		0x0000		Not supported. Must be 0x0000
24	0x034	0x0040	Capabilities Pointer	Capabilities pointer [7:0], Reserved [15:8]
25		0x0000		Reserved [31:16]
26	0x03C	0x01FF	INT/ARB/Latency	Interrupt Line[7:0] bits[7:0]; Interrupt Pin [7:0] bits[15:8]
27		0x0000		Minimum Grant[7:0], bits[23:16]; Maximum Latency[7:0], bits[31:24]
28	0x040	0x5001	Power Manage-ment Capabilities	Power Management Capability ID[7:0], bits[7:0]; Next Capability Pointer[7:0], bits [15:8]
29		0x0003		Version[2:0], bits[18:16]; PME Clock, bit 19; Device Specific Initialization, bit 21; Reserved bit 20; AUX Current[2:0], bits[24:22]; D1 Support, bit 25; D2 Support, bit 26; PME Support[4:0], bits[31:27]
30	0x044	0x0000	Power Manage-ment Control	Power State[1:0], bits[1:0]; Reserved, bit 2; Default No Soft Reset, bit 3; Reserved, bits[7:4]; PME En, bit 8; Reserved, bits[14:9], PME Status, bit 15
31		0x0000		Reserved, bits[21:16]; B2/B3 Support, bit 22; BPCC En, bit 23; PME Data[7:0], bits[31:24]
32	0x050	0x7005	MSI Capabilities	MSI Capability ID[7:0], bits[7:0]; Next Capability Pointer[7:0], bits[15:8]
33		0x0080		MSI Enable, bit 16; Multiple Message Capable[2:0], bits[19:17]; Multiple Message Enable[2:0], bits[22:20]; 32/64 Bit Message, bit 23; Reserved, bits[31:24]
34	0x070	0x0010	PCIe Capabilities	PCIe Capability ID[7:0], bits[7:0]; Next Capability Pointer[7:0], bits[15:8]
35		0x0000		PCIe Capability Version, bits[19:16]; Device Port Type, bits[23:20]; slot implemented, bit 24; Interrupt Message Number, bits[29:25]; Reserved, bits[31:30]

Table 2–1 PCIe Configuration EEPROM Field Map

EEPROM Word	PCIe Config Address	Reset Value	PCIe Config Register Name	Description
36	0x074	0x8241	Device Capabilities	Max Payload Size Supported, bits[2:0]; Reserved, bits[4:3]; Extended Tag Field Supported, bit 5; Endpoint L0s Acceptable Latency, bits[8:6]; Endpoint L1 Acceptable Latency, bits[11:9]; RAZ, bits[14:12]; Role-Base Error Reporting, bit 15
37		0x0000		Reserved, bits[17:16]; Captured Slot Power Limit Value, bits[25:18]; Captured Slot Power Limit Scale, bits[27:26]; Reserved, bits[31:28]
38	0x07C	0x3C41	Link Capabilities	Maximum Link Speed, bits[3:0]; Maximum Link Width, bits[9:4]; Active State Link PM Support, bits[11:10];L0s Exit Latency, bits[14:12]; L1 Exit Latency[0], bit 15
39		0x0000		L1 Exit Latency[2:1], bits[17:16]; Clock Power Management, bit 18; Reserved, bit 19; Data Link Layer Active Reporting Capable, bit 20; Reserved, bits[23:21]; Port Number, bita[31:24]
40	0x080	0x0000	Link Control	Active State Link PM Control, bits[1:0]; Reserved, bit 2; Read Completion Boundary, bit 3; Link Disable, bit 4; Retrain Link, bit 5; Common Clock Config, bit 6; Extended Synch, bit 7; Enable Clk Power Management, bit 8; Reserved, bits[15:9]
41		0x1001	Link Status	Link Speed, bits[19:16]; Negotiated Link Width, bits[25:20]; Undefined, bit 26; Link Training, bit 27; Slot Clock Config, bit 28; Data Link Layer Active, bit 29; Reserved, bits[31:30]
42	0x700	0x0024	ACK Latency Timer	Round Trip Latency Time Limit, bits[15:0]
43		0x006D	Replay Timer	Replay Time Limit, bits[31:16]
44	0x704	0x0000	Other Message	Other Message, bits[31:0]
45		0x0000		
46	0x708	0x0004	Port Force Link	Reserved, bits [14:0]; ForceLink, bit 15
47		0x0700		Link State, bits[21:16]; Reserved, bits[23:22]; Low Power Entrance Count, bits[31:24]
48	0x070C	0x8000	ACK Frequency	ACK frequency, bits[7:0]; N_FTS, bits[15:8]
49		0x0080		N_FTS When Common Clock, bits[23:16]; L0s Entrance Latency, bits[26:24]; L1 Entrance Latency, bits [29:27]; Reserved, bits[31:30]
50	0x710	0x0120	Port Link Control	Other Message Request, bit 0; Scramble Disable, bit 1; Loopback Enable, bit 2; MBZ, bit 3; Reserved, bit 4; DLL Link Enable, bit 5; Reserved, bit 6; Fast Link Mode, bit 7; Reserved, bits[15:8]
51		0x0007		Link Mode Enable, bits[21:16]; Reserved, bits[24:22]; Enable Corrupted CRC, bit 25; Reserved, bits [30:26]
52	0x714	0x0000	Lane Skew	Insert Lane Skew for Transmit[15:0], bits[15:0]
53		0x0000		Insert Lane Skew for Transmit[23:16], bits[23:16]; Flow Control Disable, bit 24; ACK/NAK Disable, bit 25; Reserved, bits[30:26]; Disable Lane-to-Lane Deskew, bit 31
54	0x718	0x03AA	Symbol Number	Number of TS Symbols, bits[3:0]; Reserved, bits[7:4]; Number of SKP Symbols, bits[10:8]; Reserved, bits[13:11]; Timer Modifier for Replay Timer[1:0], bits[15:14]
55		0x2001		Timer Modifier for Replay Timer[4:2], bits[18:16]; Timer Modifier for ACK/NAK Latency Timer, bits[23:19]; Timer Modifier for Flow Control Watchdog Timer, bits[28:24]; Reserved, bits[31:29]
56	0x71C	0x0280	Symbol Timer /Filter Mask	SKP Interval Value, bits[10:0]; Reserved, bits[14:11]; Disable FC Watchdog Timer, bit 15
57		0x0000		Mask Function Mismatch Filtering, bit16; Mask Poisened TLP Filtering, bit17; Mask BAR Match Filtering, bit18; Mask Type 1 Config Request Filtering, bit19; Mask Locked Request Filtering, bit 20; Mask Tag Error Rules, bit 21; Mask Requested ID Mismatch Error, bit 22; Mask Requester ID Mismatch Error, bit 23; Mask Traffic Class Mismatch Error, bit 24; Mask Attributes Mismatch Error, bit 25; Mask Length Mismatch Error, bit 26; Mask ECRC Error Filtering, bit 27; Mask ECRC Error Filtering for Completions, bit 28; Send Decoded Message on the SII, Then Drop the Message TLPs, bit 29; Reserved, bits[31:30]
58	0x748	0x0000	VC0 Posted Receive Queue Control	VC0 Posted Data Credits, bits[11:0]; VC0 Posted Header Credits[3:0], bits[15:12]
59		0x0000		VC0 Posted Header Credits[7:4], bits[19:16]; Reserved, bits[31:20]

Table 2–1 PCIe Configuration EEPROM Field Map

EEPROM Word	PCIe Config Address	Reset Value	PCIe Config Register Name	Description
60	0x74C	0x0000	VC0 Nonposted Re-ceive Queue Control	VC0 Nonposted Data Credits, bits[11:0]; VC0 Posted Header Credits[3:0], bits[15:12]
61		0x0000		VC0 Nonposted Header Credits[7:4], bits[19:16]; Reserved, bits[31:20]
62	0x750	0x0000	VC0 Completion Re-ceive Queue Control	VC0 Completion Data Credits, bits[11:0]; VC0 Posted Header Credits[3:0], bits[15:12]
63		0x0000		VC0 Completion Header Credits[7:4], bits[19:16]; Reserved, bits[31:20]

2.3 Local Memory Block

The Local Memory block consists of on-chip Secure Key Memory and associated DMA logic.

2.3.1 Secure Key Memory

The Secure Key Memory consists of 8 kbytes of secure on-chip memory, shared by all GigaCipher Processor Cores. It is used by the GigaCipher Processor Cores for secure internal storage of Asymmetric Private Keys. Data is written to or read from this memory by the GigaCipher Processor Core over the Core Bus (Inbound/ Outbound).

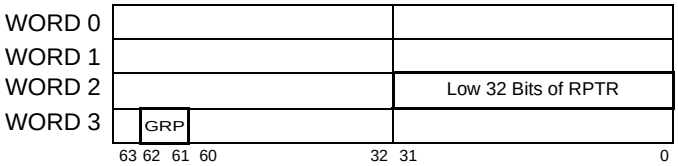
Secure Key Memory is not directly accessible from either the external Host interface or the external PSE interface.

2.4 Instruction-Queue Manager

During normal runtime operation, the CN16XX device fetches and executes macro-instructions stored in an instruction queue in host system memory. CN16XX instructions are 32 bytes long, and contain an opcode, instruction parameters, and instruction address pointers. The exact format of the 32-byte instructions is largely determined by the microcode loaded into the NITROX®, not the NITROX® hardware itself.

Figure 2–3 indicates the instruction fields that the NITROX hardware uses. In the case of a hardware error in one of the GigaCipher cores, the NITROX hardware captures the “low 32 bits of RPTR” from the instruction executing during the error in the CORE ERROR ADDRESS register (described in Section 5.2.1). The NITROX® hardware also determines which GigaCipher cores may execute the instruction using the GRP field. Refer to Section 2.5.1.

Figure 2–3 NITROX Hardware Instruction Fields



The Instruction-Queue Manager (IQM) is responsible for fetching instructions from the host system and dispatching them to GigaCipher Processor Cores. There are actually four independent Instruction-Queue Managers (IQM0-IQM3) within CN16XX, enabling up to four independent instruction queues in host memory.

An IQM performs the following operations:

- Maintaining address & size of external instruction queue(s).

- Maintaining a count of instructions in external instruction queue(s).
- Fetching instructions from the external instruction queue(s) and writing them to internal queue(s).
- Maintaining GigaCipher Processor Core availability status, and determining next available GigaCipher Processor Core for instruction assignment.
- Dispatching (writing) the next instruction in the internal queue(s) to the next available GigaCipher Processor Core.
- Maintaining group participation and scheduling instructions for GigaCipher Processor Cores.

2.4.1 Interfaces

The IQM's interface to both the Outbound Core Bus and the Inbound Core Bus. The combination of these two interfaces provides access to the Host System via the Host System Interface, and to the GigaCipher Processor Cores directly.

2.4.2 Functional Blocks

The IQM block consists of four completely independent instruction-queue managers (IQMs), as well as shared control logic. Four independent queue managers allow for interfacing with up to four host processors through independent instruction queues. In addition, it is also possible for a single host processor to manage up to four independent instruction queues, however there is no requirement to do so to obtain full system performance.
CORE ENABLE[IQM enable] enables each IQM.

Shared Control Logic

This logic maintains status of GigaCipher Processor Core availability, and services instructions for GigaCipher Processor Cores and Core Groups from all IQMs as described herein.

When a GigaCipher Processor Core is finished with its current instruction and ready for the next one, it informs the IQM's shared control logic that it is ready to accept another instruction. This is done by submitting a command to the IQM over the Outbound Core Bus. The shared control logic processes that command, and maintains a table of available GigaCipher Processor Cores and Core Groups.

Shared control logic also services each of the independent instruction queues by assigning to them the next available GigaCipher Processor Core in the execution group the instruction indicates. The queue managers are serviced by the shared control logic in a round robin fashion.

Independent Instruction-Queue Managers (IQMs)

Each of the IQMs is composed of the following logical blocks

- Host Memory Instruction-Queue Control

  Read Pointer, Start Address and Length registers identify next available instruction in the Instruction Queue in Host system memory (Section 5.2.1 describes the IQM0-3 QUEUE BASE ADDRESS *, IQM0-3 QUEUE SIZE *, IQM0-3_NEXT_ADDR_*, and IQM0-3_CMD_CNT_* registers.).

- Internal Instruction Queue

Contains GigaCipher Processor Core instructions read from associated host memory instruction queue, but not yet written to GigaCipher Processor Cores for execution. Each internal queue can store up to 32 instructions. Each queue can burst up to 16 instructions per DMA read command from its associated external queue in host system memory.

- Internal Instruction-Queue Write Pointer & Control

  Manages writes of new instructions into internal instruction queue, as read from associated external instruction queue over Host Interface.

- Internal Instruction-Queue Read Pointer & Control

  Manages reads of next instruction(s) in internal queue, and associated writes of those instructions to GigaCipher Processor Cores. Instructions from internal instruction queues are sent to GigaCipher Processor Cores on the Inbound Core Bus.

- Doorbell Register

  Written by host, to indicate number of new instructions added to end of associated external instruction queue. The value written to the doorbell register is added to the current value of the Pending-Instruction Counter described below (Section 5.2.1 describes the IQM0-3 DOORBELL register).

- Pending Instruction Counter

  Maintains the count of instructions to be read from its associated external queue into internal queue. This count is the sum of all values written to the doorbell register, minus the number of instructions read from the external queue. Its value can be read by reading the DOORBELL register (Section 5.2.1 describes the IQM0-3 DOORBELL register).

2.4.3 Functional Overview

After initialization of IQM external pointers & internal counters over the Host interface, each IQM waits for its respective Doorbell Register to be written with the number of new instructions added to its associated external queue. The IQM then adds the new Doorbell-Register value to its Pending-Instruction Counter. When the Pending-Instruction Counter is greater than zero, and there is empty space in the internal queue, the IQM submits a DMA read operation to the Host Interface over the Outbound Core Bus. When read data is returned from the Host Interface to the IQM over the Inbound Core Bus, the internal-queues write pointer is incremented.

When the IQM read pointer is less than its associated write pointer, the queue contains an instruction, and is ready to be serviced. Shared Control Logic services IQMs one at a time, in a round robin fashion. For each IQM in turn, an available GigaCipher Processor Core is identified, the next instruction in the currently serviced queue is popped and written to the GigaCipher Processor Core over the Outbound Core Bus, and the IQM's Internal Read Pointer is incremented.

2.5 Execution Units (GigaCipher Cores)

The collection of up tp eight execution units (also referred to as the GigaCipher Processor Cores or GPCs) is the heart of the CN16XX device. Each unit is an independent processor that provides logic to perform asymmetric and symmetric cryptography, as well as general arithmetic and logic functions.

Each execution unit supports a 3Kbyte register file that serves as a temporary store for data during packet processing. All of the computational units of the execution unit can access the data that is stored in the register file. The register file is used to store packet data from the Input Buffer Logic.

Each execution unit consists of dedicated computational blocks that perform hash, encryption, modular exponentiation, or generic ALU functions. The Hash Logic supports MD5, SHA-1, SHA-256 and SHA-384 and SHA-512 hash algorithms, as well as the Galois Field Multiplication needed for AES-GCM and other protocols. The Encryption Logic supports symmetric cryptography functions, such as DES/3DES, AES, RC4, and KASUMI. The AES engine supports 128-bit, 192-bit, and 256-bit key variants. The Modular Exponentiation Logic performs modular exponentiation up to 4096 bits to support symmetric cryptography functions such as RSA and Diffie-Hellman (groups 1,2,5). The Chinese Remainder Theorem (CRT) can be employed, via microcode, to speed up the modular exponentiation functions. The operand size and use of the CRT are controlled by the associated microcode.

The ALU logic performs addition/subtraction, exclusive-or, comparison, byte manipulation, and byte accumulation functions. It is used for general purpose and protocol pre/post processing.

Note that the designer does not have direct visibility into the register file or the Microcode Mem logic of the Execution Units. Consequently, each execution unit employs Built-In-Self-Test (BIST) on its internal memory structures in order to verify their functionality. The BIST is executed after Power-on. This necessitates that microcode be downloaded into the appropriate Execution Units after each hard or soft reset function.

The exact function of the execution units is largely determined by the microcode that is loaded into them (see “GigaCipher Processor Core (GPC) Instruction Set” for additional information). The Ucode Load Register is used to load microcode into the units. When downloading microcode into CN16XX, all disabled units receive the microcode. The CORE ENABLE Register enables/disables individual units. The number of bits set in the EPC_EFUS_CORE_EN Register indicates the number of available cores. The available cores are always numbered 0 .. N–1.

2.5.1 CN16XX Execution Unit Allocation

CN16XX allocates execution units based on the group of the required operation. CN16XX supports four groups. Each execution unit may support any group or any combination of groups.

This structure allows for maximum flexibility and highest performance. The microcode size can be effectively increased because different execution units can contain different microcodes to execute different operations. Section 2.4 describes how the instruction indicates the core group

Each instruction selects a particular core group (see the GRP field in Figure 2–3). The REG_EXEC_GROUP register indicates which cores participate in each group. (“REG_EXEC_GROUP Register” on page 72 describes the REG_EXEC_GROUP register.)

2.6 Random Number Generator (RNG)

The Random Number Generator produces random data at a rate of more than 200 Mbps. The block is comprised of an entropy source, a Linear Feedback Shift Register (LFSR), and a SHA-1 hash engine. The entropy source consists of a large number of high-jitter free-running oscillators. Entropy output is fed to the LFSR to further mix the data. The LFSR output is fed to the SHA-1 block, which performs hash processing on the mixed data. This multi-stage processing path for entropy data ensures high cryptographic quality of the random number.

Random data is accessible by the host system via the **Random_Number** instruction in the GigaCipher Processor Core instruction set. Up to 2KB of random data can be retrieved in a single **Random_Number** instruction. For details of the **Random_Number** instruction, refer to the NITROX Family Instruction Set Manual.

64-bit random numbers are continually generated by the RNG block, and offered to waiting GigaCipher Processor Cores on the Inbound Core Bus. If there are no GigaCipher Processor Cores waiting for random data, the current 64-bit random number is discarded by the RNG. Hardware ensures that the 64-bit random numbers are shared amongst waiting GigaCipher Processor Cores, and prevents multiple cores from consuming the same random data.

The entropy data source output is enabled via the **RND_Entropy_Enable** bit in the Command/Status register. On power-up/reset this bit is cleared to zero, disabling the entropy source from feeding the LFSR and SHA-1 engine, allowing for RNG block output predictability. In addition to **RND_Entropy_Enable**, the **RND_Disable** bit disables the random number generator from delivering any new random numbers to the GigaCipher Processor Cores, and resets the LFSR mixing function back to its reset value (00…1). These two control register bits are used to provide for RNG block testability.

Note that when in PSE Safe Mode (See PSE_MODE pin description), the entropy source is automatically enabled (PSE Safe Mode operation overrides the state of the **RND_Entropy_Enable** bit).

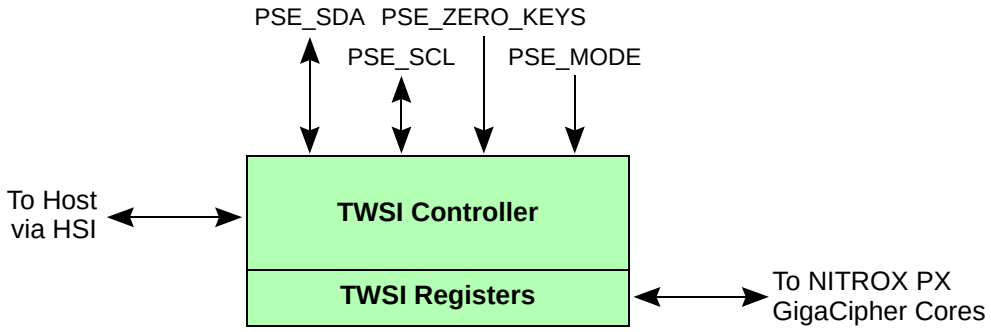
2.7 Personal Security Environment (PSE) Block

The Personal Security Environment (PSE) block allows CN16XX implementations to be designed for compliance to various high-security standards such as FIPS (Federal Information Protection Standard). The PSE block is comprised of three elements, shown in Figure 2–4

- A **PSE_MODE** input pin that, when asserted, forces CN16XX into high-security mode of operation, where direct access to most of the internal state and functionality from the Host System Interface is prohibited.
- A **PSE_ZERO_KEYS** input pin that, when asserted, causes CN16XX to clear all data in its internal and external local memories, including GigaCipher Processor Core register files, and Internal Secure Memory. For additional information, see the description of the **PSE_ZERO_KEYS** pin in Section 4.1.

- A Two-Wire Serial Interface (TWSI) bus and associated bus controller. This interface is compatible with industry-standard two-wire serial bus protocol. The PSE_SDA pin is a bidirectional data line, and the PSE_SCL pin is a bidirectional clock line. Detailed functionality of the TWSI interface and associated control logic is described later in this document. An overview is included in Section 2.8.

Figure 2–4 PSE Block Diagram



2.8 Two-Wire Serial Interface (TWSI)

The CN16XX Two-Wire Serial Interface (TWSI) is a general purpose serial bus interface compatible to the industry-standard two wire serial bus protocol. This interface provides master and/or slave mode communications (read/write), between CN16XX's internal TWSI controller and any device on the external TWSI bus. The TWSI controller is managed by, and communicates with the internal GigaCipher Processor Cores, and/or the Host System Interface logic. Thus, the TWSI provides a simple, inexpensive, and private communications port that can be used to connect CN16XX's GigaCipher Processor Core and/or Host System Interface to any industry-standard two wire serial bus-compatible device such as a serial EEPROM or microcontroller.

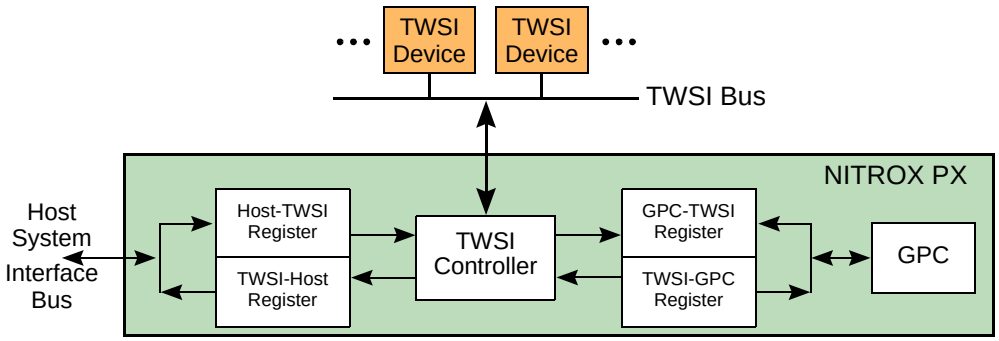
The TWSI provides two primary functions in a CN16XX system design:

- Enables the connection of I2C-compatible slave devices (such as EEPROMs) to the TWSI. These devices can be used to store system configuration and/or status information, which can be read or written by the host system over the Host System Interface, or by the GigaCipher Processor Cores.
- Provides for the connection of I2C-compatible master devices (such as microcontrollers) to the TWSI. These devices can be used for any desired type of system configuration and/or maintenance.

Data Paths

Figure 2–5 shows the data paths between the TWSI interface, the Host System Interface, and the GigaCipher Processor Cores.

Figure 2–5 Datapath Through the TWSI Interface



The host system sends commands and data to, and reads status and data from the TWSI bus logic via a pair of TWSI bus control registers. These registers, the **HOST-TWSI** and **TWSI-HOST** registers, are mapped to and accessible by both the host and the TWSI bus control logic.

A corresponding set of control registers is shared between the GigaCipher Processor Cores and the TWSI logic. The **GPC-TWSI** and **TWSI-GPC** registers are mapped to and accessible by both the GigaCipher Processor Cores and the TWSI bus control logic. These registers are not accessible from the Host System Interface.

The **HOST-TWSI** and **TWSI-HOST** registers, as well as the **GPC-TWSI** and **TWSI-GPC** registers are described in detail in Section 5.3.1, “TWSI Commands and Operations”.

For TWSI bus writes, the Host System Interface or GigaCipher Processor Core writes a TWSI command and associated write data to their respective ***-TWSI** control register, and then polls their respective **TWSI-*** control register for completion status. For TWSI bus reads, the Host System Interface or GigaCipher Processor Core writes a TWSI command to their respective ***-TWSI** control register, and then polls the corresponding **TWSI-*** register for completion status and associated read data.

The ***-TWSI** and **TWSI-*** control registers also provide a path into the command, control, and status registers of the internal TWSI controller logic. The internal TWSI controller registers are not directly accessible by the Host System Interface or the TWSI bus directly. They are not directly mapped to the Host System Interface, the GigaCipher Processor Cores, or the TWSI signal interface. Instead they are indirectly accessed using TWSI commands, which are sent to the controller over the ***-TWSI** bus control registers.

CN16XX’s TWSI interface operates in one of two modes: *master-slave*, or *slave-only*. The TWSI mode is selected using the **TWSI Mode** control register located within the internal TWSI control logic. The mode determines the way in which the TWSI interface reacts to writes to the **Host-TWSI** or **GPC-TWSI** registers.

For a detailed description of the registers and functionality of the TWSI interface, refer to Section 5.3, “Two-Wire Serial Interface Controller Details”. For TWSI timing descriptions and diagrams, see Section 4.2, “AC Characteristics”.

Common Uses of theTWSI Interface

The intended use of a TWSI master device connected to CN16XX is to enable use of CN16XX's PSE Safe Mode. In this mode (enabled by setting the **PSE_MODE** input pin), microcode for CN16XX internal processor cores (GPCs) cannot be loaded over the Host System Interface under host control. It can be loaded by, and under the control of an intelligent master device on the TWSI, or by existing GigaCipher cores via access to the TWSI interface. This feature prevents a processor subsystem from being commandeered by software running on the host system. This assists in enabling CN16XX to be used in a high security environment such as that required of FIPS140 specifications. PSE Safe Mode limitations are described with the **PSE_MODE** signal description in Section 4.1. GigaCipher Processor Core microcode functions required to enable PSE Safe Mode implementations, are described in detail in the instruction set and programmers guide documents listed in the appendix of this document.

Other general-purpose uses of the TWSI include user-defined and implemented functions. Examples include interfacing CN16XX to a smart card controller, for applications such as key storage and retrieval over an isolated and trusted path.

2.8.1 PSE Safe Mode Microcode Download

Secure downloading of CN16XX micro-code MAY be performed using a PSE master device on the TWSI bus. Below is a brief description of the process. PSE functionality is a function of three primary elements; GigaCipher Processor Core microcode, software executed in the external TWSI microcontroller (a TWSI master device), and software elements of the driver and API. Additional details of PSE Mode implementation are described in more detail in the NITROX Family Instruction Set and Programmers Guide manuals.

Cavium Networks implements the needed GigaCipher Processor Core microcode to enable this process. Host CPU and TWSI master and slave device reference software MAY also be offered by Cavium as reference. However it is assumed herein that implementation of the Host CPU and TWSI master device software and communications mechanism is an implementation detail to be implemented by the user. Thus it is not specified in this document.

After reset and system-boot, the following steps are performed to load microcode into GigaCipher Processor Cores using PSE-Safe Mode:

1. Host CPU initializes the CN16XX TWSI controller's internal registers (enable TWSI interface, enable slave receive, program proper TWSI clock value and program the slave address).
2. Host CPU sends a command to the TWSI master device on the TWSI bus, to load microcode stored in the TWSI environment, into GigaCipher Processor Cores.
3. TWSI master device loads admin microcode into the GigaCipher Processor Cores.
4. TWSI master device informs Host CPU that GigaCipher Processor Core micro-code loading is complete.
5. Host CPU enables GigaCipher Processor Cores via the Core Enable register.
6. Admin microcode executes the GigaCipher Processor Core loading protocol (specified by the GigaCipher Processor Core micro-code instruction set). Based on the desired PSE protocol it either reads GigaCipher Processor Core microcode from the TWSI interface or from the Host System Interface. In the case of micro-code load over the Host System Interface, the code should be stored in a signed container, to be verified by the GigaCipher Processor Core microcode. GigaCipher Processor Cores then load the micro-code to all (disabled) GigaCipher Processor Cores.
7. GigaCipher Processor Cores load any secure keys from TWSI master or slave devices as needed.
8. GigaCipher Processor Cores inform TWSI master device about the completion of the key and microcode loading tasks.
9. TWSI master device informs Host CPU about the completion of micro-code loading task.
10. Host CPU enables the GigaCipher Processor Cores.

2.9 JTAG Module

The CN16XX processor incorporates a serial boundary scan test access port (TAP). This port operates in accordance with IEEE standard 1149.1-1990. For a complete description, refer to the IEEE Standard Test Access Port specification document (IEEE Std. 1149.1-1990).

The standard JTAG interface consists of four basic elements:

- Test Access Port (TAP) signal pins
- TAP controller
- Instruction Register (IR)
- Data Registers (DR), including:
  - Bypass Register
  - Boundary Scan Register
  - Device ID Register

2.9.1 JTAG Register Summary

The registers used in the JTAG block are accessible by, and relevant to the JTAG port only. These registers are not accessible from the Host System Interface port of the CN16XX macro processor, and are therefore not included in Chapter 4, “Software Interface”.

Table 2–2 JTAG Register Summary

Register Size (Bits)	Register Name
4	JTAG Instruction Register
1	JTAG Bypass Register
32	JTAG Device ID Register
TBD	JTAG Boundary Scan Register

2.9.2 Register Details

JTAG Instruction Register

Description	Bit(s)	Type
The following are CN16XX-supported JTAG Instructions. These instructions are written to the JTAG Instruction Register over the JTAG TAP. • 0000-EXTEST Forces contents of the boundary scan cells onto the device outputs. Places the boundary scan register (BSR) between TDI and TDO • 0001-SAMPLE/PRELOAD Place the boundary scan register (BSR) between TDI and TDO. SAMPLE allows data from device inputs and outputs. to be captured in the boundary scan cells and shifter serially through TDO. PRELOAD allows data to be input serially into the boundary scan cells via the TDI • 0010-DEVICE_ID Loads the JTAG ID register with the vendor ID code and places the register between TDI and TDO • 0011-HIGHZ Places the bypass register (BYR) between TDI and TDO. Forces all device output drivers to a high-Z state. Truncates the boundary scan register as a single bit in length	4	W

JTAG Bypass Register

TBD

JTAG Device ID Register

The following provides a description of the JTAG ID register contents.

Description	Bit(s)	Type
Reserved for revision number (0x0)	31:28	R
Cavium Part Number (0x700)	27:12	R
Unique Manufacturer's ID for Cavium (0x1cc)	11:1	R
Indicates presence of an ID register (1)	0	R

JTAG Boundary Scan Register

Cavium provides a Boundary Scan Descriptive Language (BSDL) file for the device upon request. Contact your local Cavium sales representative for details.

Chapter 3

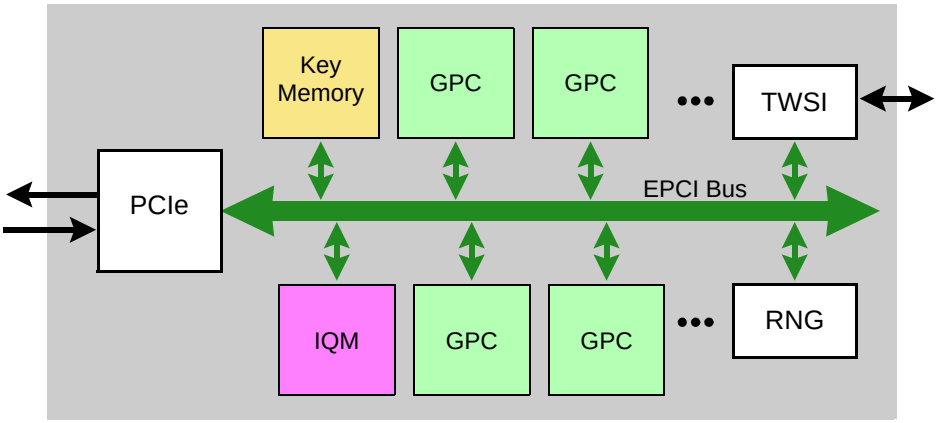
Theory of Operation

This section describes the typical flow for Cavium Networks NITROX® PX CN16XX Processor macro processor instruction processing. This includes a detailed flow description of a typical instruction or program execution flow. Chip power-on/initialization flow, and microcode download process flow are also described briefly. This understanding, and an understanding of the basic system concepts and processor architecture provided by the first two chapters of this document, should provide you a strong grasp of the basic theory of operations of the CN16XX macro processor.

3.1 Overview

Figure 3–1, below, presents the flow of operation for a CN16XX device.

Figure 3–1 Operational Block Diagram for CN16XX Family Devices



The execution units are the micro-programmable portion of Nitrox PX. The microcode can access the internal resources of the execution units and also can utilize the following via EPCI bus transactions:

- DMA read and write operations to host memory via the PCI bus
- Key memory (8KB) read and write operations
- Request unit delivery and receipt of new requests
- Random number receipt
- Request and receive PSE/TWSI interface transactions or PSE/TWSI interface secure communications

3.2 Typical Instruction Execution

Once the CN16XX Macro Processor has been initialized it is ready to execute instructions. The following is a description of the typical flow of instruction execution.

To execute a Macro Processor instruction, host software (an API and Driver) performs the following operations:

1. In a multi-threaded or multi-process system, acquires access to one of four instruction queues in host memory using an appropriate synchronization mechanism.
2. Adds the desired Macro Processor instruction(s) to the end of the host instruction queue and increments the head (write) pointer. Detailed descriptions of instruction format are included in the GPC Instruction Set description section of this document. Instruction queue entries are 32 bytes in size.
3. Releases access to the host instruction queue that was locked in step 1 above.
4. Writes the number of new instructions added to the host queue, to the associated doorbell register in the Instruction Queue Manager (IQM). In this flow example only one instruction will be added to the queue, so a 0x00000001 would be written to the Doorbell Register.

5. Waits for the instruction(s) to complete, as indicated by polling of a valid completion code at the instruction's completion code address in host system memory.

In response to having its doorbell register written, the Macro Processor performs the following operations:

1. The IQM adds the desired number of entries written in its doorbell register, to the IQM's count of valid host queue entries to be read.
2. The IQM issues a DMA READ command to the HSI block over the Outbound Core Bus. This DMA read will 'pull' instructions from the host instruction queue into the associated internal instruction queue. The number of instructions read during this read operation is dependent on the value in the IQM's count of valid host queue entries in step 1 above), and available space in the internal instruction queue. The IQM controls this flow, and up to eight instructions can be burst from host memory into the internal IQM queue in one DMA operation.
3. When the IQM-issued DMA READ operation of the previous step reaches the head of the HSI's DMA operation queue, the HSI issues a READ operation on the Host Interface Port. This HSI DMA operation reads a block of one or more host instruction queue entries from the HSI port.
4. Data read from the HSI Port is transferred to the read FIFO in the HSI block, which forwards it to the IQM internal instruction queue over the Inbound Core Bus. Upon reading a number of instructions from the external queue to the internal queue, the IQM decrements its count of valid host queue entries by that same number.
5. The IQM writes the next available instruction in its internal instruction queue to the register file of the next available GPC.
6. The GPC begins processing the instruction. It reads input data from host memory at the address specified in the instruction, by issuing DMA READ commands to the HSI block over the Outbound Core Bus. The GPC then receives the corresponding input data over the Inbound Core Bus.
7. As the GPC produces output data during instruction execution, it issues DMA WRITE commands to the HSI, writing output data from the GPC to host memory at the address specified in the instruction.
8. The GPC informs the IQM that it is available to process another instruction (and has finished the previous one).

The host software driver thread, which has been polling the instructions' destination buffer, sees the write completion and passes the output data to the application.

RNG instructions are a small exception to the above flow, because the RNG block is independent of the GPC. The IQM assigns the RNG instruction to the next available GPC. The GPC issues a DMA READ instruction to the RNG block over the Outbound Core Bus, and Random data is returned to the GPC over the Inbound Core Bus. When the GPC has collected the required number of bits, it transfers the result to the host memory address specified in the RNG instruction via DMA WRITEs, in the same manner as above.

As described earlier, there are multiple independent GPCs in the Macro Processor. The IQM assigns instructions to the GPCs in the order received into its queue. However because different types of instructions may take different amounts of time to complete, instructions may complete out of order. For example, an RNG instruction may be performed much faster than a modular exponentiation. Even two modular exponentiation operations may take different times to complete. Driver and API software can be written to maintain processing order in the host system. Refer to the NITROX® PX CN16XX Software Programmers Guide and API Specification document.

3.3 Device Initialization

Upon power-on/reset, CN16XX macroprocessor registers are preconfigured according to the default values shown in the Detailed Register Descriptions section of this document. The process of bringing the CN16XX Macroprocessor out of this reset configuration includes setting Base Address registers (BARs), and configuring memory-mapped Configuration and Status registers.

3.3.1 Microcode Download

After basic register initialization is completed as described above, microcode must be downloaded to the GPCs, by writing to the UCODE LOAD register. The procedure for loading microcode is decribed in the NITROX® PX Instruction Set (Microcode) Manuals, NITROX® PX CN16XX Family SSL Instruction Set Manual, or the NITROX® PX CN16XX Family IPSEC/IKE Instruction Set Manual, referred to in the Preface of this document.

Microcode download can be done in either Normal Mode or PSE Safe Mode. PSE Safe Mode microcode download uses the CN16XX TWSI interface as well as the PSE_MODE pin, to control the microcode download process for GPCs. This PSE Safe Mode method allows for a protected and secure download of microcode utilizing cryptographic methodologies to provide user and microcode authenticity and integrity. PSE Safe Mode microcode download is used to assist in achieving FIPS140-2 compliance in a security system implementation.

3.3.2 Core Discovery

To load microcode into all the cores and establish core groups, first determine the number of cores present. Count the number of 1s in the EPC_EFUS_CORE_EN Register. The available cores are always numbered 0 .. N–1.

3.3.3 Soft Reset Usage

CN16XX can be reset by the host CPU (soft-reset) by writing to the SOFT_RESET bit in the COMMAND Register. Performing soft-reset while CN16XX cores are currently executing commands may cause unknown results in system software. Therefore, Cavium Networks recommends that CN16XX be soft-reset only after software has made certain that no instructions are currently being executed by any cores. Software can determine that the cores are idle by reading the INTERNAL STATUS Register.

Once core discovery has been completed, perform the following steps to safely perform a soft reset:

1. Disable the IQM and enable all available cores by writing the appropriate core-mask value to the CORE ENABLE Register. This enables all available cores, but stop the IQM from fetching any new instructions from host memory.
2. Verify that all existing cores are in an idle state as follows:
   - Write a 0x7CC to the INTERNAL STATUS register, then read this same register. Bits [19:12] represent the bit mask for cores 7–0. For each core present and idle, the corresponding mask bit will be set. Continue polling this register until every mask bit corresponding to an existing core is set.

     When all mask bits associated to known present processor cores are set, all processor cores have finished processing instructions, and are waiting for a new instruction to be issued by the IQM. Because the IQM is disabled, no new instructions will be started and the chip remains in a quiescent or idle state.
3. Check for HSI Outbound FIFO Empty.
   - Write a 0x783 to the INTERNAL STATUS register. Read this register to poll for bit [21] to be set. When set, no more outbound data traffic corresponding to instructions will be generated by the CN16XX processor.
4. Allow all outstanding PCIe read/write accesses to complete.
5. Perform a soft reset.
   - Set bit [5] in COMMAND Register. The reset takes 20 internal core-clock cycles to complete. A soft reset is the equivalent of a hard reset for the CN16XX, with the exception of the PCIe core and DMA logic.
6. Perform the CN16XX initialization procedure (i.e. set all CN16XX configuration registers, reload microcode, re-enable GPCs, etc).

3.4 PSE Safe Mode Operations

3.4.1 Key Zeroing

Clearing all keys to zero in both internal and external local RAM is accomplished by driving the PSE_ZERO_KEYS pin high for a minimum zeroing time, provided the Cores contain the appropriate microcode. Refer to the Pin Descriptions section of this document for a complete description of the behavior of the PSE_ZERO_KEYS pin.

When this pin is asserted (active high), it acts as a non-maskable interrupt for all the GPCs. Also, while this pin is asserted all GPC clocks are forced to an enabled condition. As a result, all internal cores jump to a hardwired control store address.

Chapter 4

Hardware Interface

This section contains a detailed reference for information required to implement a board level logic design. This includes detailed signal descriptions, AC signal timing diagrams and parameters, and DC characteristics of the device. With the addition of the physical package mechanical information included at the end of the document, this section constitutes everything a designer should need to design the NITROX® PX CN16XX™ onto a circuit board, and interface it to the rest of the hardware in the system.

4.1 Signal Descriptions

This section contains signal descriptions for all signals used in the NITROX® PX CN16XX family devices. The signal-type key below provides a description of signal type variations. Signal AC and DC characteristics are described later in this chapter.

I	=	Input
O	=	Output
I/O	=	Bidirectional
CMOS33	=	3.3V CMOS signal
SERDES	=	SERDES differential signal
Supply	=	Power/GND supply pins

Table 4–1 Signal Descriptions Arranged by Function

Signal Name	Type*	Up/ Down[1]	Description
PCI Express Port Interface Signals			
EXP_TX_[3:0]_P	O, SERDES		Differential output data (pos). Refer to Section 4.2.2.2.
EXP_TX_[3:0]_N	O, SERDES		Differential output data (neg). Refer to Section 4.2.2.2.
EXP_RX_[3:0]_P	I, SERDES		Differential input data (pos). Refer to Section 4.2.2.2.
EXP_RX_[3:0]_N	I, SERDES		Differential input data (neg). Refer to Section 4.2.2.2.
EXP_BAND_RES	I, Analog		Bandgap precision resistor between the signal pin and GND with a value of 2.37kΩ ± 1%.
EXP_COMP_RES	I, Analog		Compensation precision resistor between the signal pin and GND with a value of 50Ω ± 1%.
EXP_REF_CLK_P	I, SERDES		Ref clock input (pos) for PLL Reference Clock (Differential 100 MHz Clock). Refer to Section 4.2.2.1.
EXP_REF_CLK_N	I, SERDES		Ref clock input (neg) for PLL Reference Clock (Differential 100 MHz Clock). Refer to Section 4.2.2.1.
EXP_REVERSE _LANES	I, CMOS33	DN	Pull up to reverse the order of the express lanes. This signal is pulled down internally. • 0 = EXP*0* is lane 0, EXP*1* is lane 1, EXP*2* is lane 2, EXP*3* is lane 3 • 1 = EXP*0* is lane 3, EXP*1* is lane 2, EXP*2* is lane 1, EXP*3* is lane 0
PERST_L	I, CMOS33		PCI Express reset
PCI CFG EEPROM Interface			
EPR_SK	O, CMOS33		EEPROM Shift Clock
EPR_CS	O, CMOS33		EEPROM Chip Select
EPR_DI	O, CMOS33		EEPROM Data In (CN16XX Data Out)
EPR_DO	I, CMOS33	UP	EEPROM Data Output (CN16XX Data In). This signal is pulled up internally.
PSE Interface			
PSE_SDA	I/O,CMOS33,OD		TWSI Data. This signal should be pulled low if unused.
PSE_SCL	I/O,CMOS33,OD		TWSI Clock. This signal should be pulled low if unused.
PSE_ZERO_KEYS	I, CMOS33	DN	Upon detection of a rising edge, writes all 0s to onchip memory. This signal is internally pulled down. May be left open, tied to GND, or tied to a test point if unused.
PSE_MODE	I, CMOS33	DN	0: Normal Mode operation 1: PSE Safe Mode operation This signal is pulled down internally. May be left open, tied to GND, or tied to a test point if unused. In PSE Safe Mode: - the UCODE_DOWNLOAD register is not accessible from the Host System Interface (writes are ignored, reads return unknown data) - The RNG entropy source is forced to the enabled state - Access to the UCODE_DOWNLOAD register over the TWSI bus is enabled.

Signal Name	Type*	Up/ Down[1]	Description
JTAG Interface			
JTG_TRST_L	I, CMOS33	DN	JTAG Test Reset - Pulled down internally. May be left open, tied to GND, or tied to a test point if not used.
JTG_TDI	I, CMOS33	UP	JTAG Test Data In - Pulled up internally. May be left open, tied to VDD33, or tied to a test point if not used.
JTG_TDO	O, CMOS33		JTAG Test Data Out
JTG_TCK	I, CMOS33	UP	JTAG Test Clock - Pulled up internally. May be left open, tied to VDD33, or tied to a test point if not used.
JTG_TMS	I, CMOS33	UP	JTAG Test Mode Select - Pulled up internally. May be left open, tied to VDD33, or tied to a test point if not used.
Core-Clock Interface			
CLK_OUT	O, CMOS33		Core clock output (350/400 MHz). When enabled by CLKOUT_EN, the internal core clock is driven on this output pin for debug purposes. Tie to test point.
CLKOUT_EN	I, CMOS33	DN	Enable signal for CLK_OUT. This signal is pulled down internally. Cavium recommends that CLKOUT_EN be pulled low during normal operation, or not connected.
PLL_BYPASS	I, CMOS33	DN	Core clock PLL bypass control signal used for testability. 0: Normal core clock PLL and multiplier operation. 1: Core clock PLL and multiplier will be bypassed. The clock will be fed directly from the PLL input reference signal. Intended for debug purposes only. Pulled down internally. May be left open, tied to GND, or tied to a test point if unused.
PLL_DCOK	I, CMOS33		Indicates to internal circuitry that power and reference clock are stable. Refer to Section 4.3.3, "Power Sequencing", for usage details. Rise time must be less than 200 ns.
PLL_REF_CLK	I, CMOS33		PLL reference clock (50 MHz)
Power Supply			
GND	Supply		Ground
EXP_GND	Supply		Ground for PCI Express interface
VDD_1V	Supply		GPC power-supply voltage
VDD_3V	Supply		3.3V I/O-supply voltage
PLL_VDD_3V	Supply		Core-clock PLL-supply voltage
EXP_VDD	Supply		PCIe interface power-supply voltage
EXP_VDDBAND	Supply		PCIe bandgap-supply voltage
EXP_VDDPLL	Supply		PCIe PLL supply voltage
Miscellaneous			
ENA_400	I, CMOS33	DN	Pull up for 400 MHz, pull down for 350 MHz.
TEST_PD	I	UP	Pull down for normal operation with 1kΩ to 5kΩ resistor. This signal is pulled up internally.
RSVD			Reserved for internal use. Must be left unconnected.
NC	Unused		No Internal Connection

1. If this column has a value, it specifies that this signal pin has an integrated "x"-Ω pull-up or pull-down resistor. Any pin that includes an internal pull-up/down resistor can be left unconnected. Unconnected pins with internal pull-up resistors are set high by default, and those with pull-down resistors are pulled low.

4.2 AC Characteristics

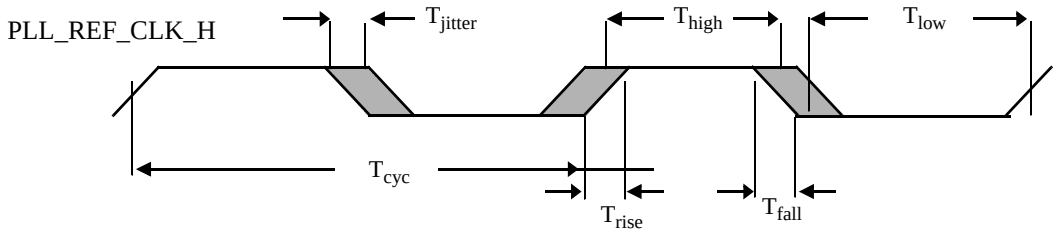
4.2.1 Core-Clock (PLL_REF_CLK) Input

Table 4–2 Core-Clock Timing Parameters

Parameter	Description	PLL_REF_CLK = 50MHz			Units
		Min	Typ	Max	
T_{cyc}	Clock Cycle Time	—	20	—	ns
T_{high}	Clock High Time	8	10	12	ns
T_{low}	Clock Low Time	8	10	12	ns
T_{rise}	Clock Rise Time[1]	—	—	2	ns
T_{fall}	Clock Fall Time[2]	—	—	2	ns
T_{jitter}	Clock Jitter Time[3]	—	—	100	ps

1. 10%-90%
2. 90%-10%
3. cycle to cycle

Figure 4–1 PCI_REF_CLK_H Core Clock Input Timing Waveform



4.2.2 PCI Express Interface (SERDES Input and Output Signals)

4.2.2.1 PCIe Clock Input Timing

Table 4–3 lists CN16XX's EXP reference-clock input electrical characteristics.

Table 4–3 EXP Reference-Clock Input

Clock Name	Frequency	Frequency Tolerance[1]	Duty Cycle	Rise Time /Fall Time (max)[2]	Phase Jitter (peak-to -peak)[3]	V_{HIGH} Levels			V_{LOW} Levels		
						Min	Nom	Max	Min	Nom	Max
EXP_REF_CLKN/P	100 MHz	±300 ppm	40/60	840 ps	80 ps	625mV	700mV	850mV	–150mV	0V	150mV

1. In links where a spread-spectrum-modulated reference clock is used, both ends of the link must share the same master reference-clock source. (A master reference clock may also be referred to as a bit-rate clock source.)
2. Measured from 20 to 80%.
3. Measured at a BER ≤ 1e–12

4.2.2.2 PCIe Signal Timing

Table 4–4 shows the PCIe TX specifications, and Table 4–5 shows the PCIe RX specifications.

Table 4–4 PCIe TX Specifications

Parameter [1]	Min	Typical	Max	Units
$V_{TX\text{-}DIFF\ pp}$	0.8	1	1.1	V
$V_{TX\text{-}DE\text{-}RATIO}$ [2]	–3.1	–3.5	–3.9	dB
$T_{TX\text{-}EYE}$	0.75	—	—	UI (400 ps)
$T_{TX\text{-}RISE}$	0.125	—	—	UI (400 ps)
$T_{TX\text{-}FALL}$	0.125	—	—	UI (400 ps)
$V_{TX\text{-}CM\text{-}ACp}$	—	—	20	mV
$V_{TX\text{-}CM\text{-}DC\text{-}ACTIVE\text{-}IDLE\text{-}DELTA}$	—	—	100	mV
$V_{TX\text{-}CM\text{-}DC\text{-}LINE\text{-}DELTA}$	—	—	20	mV
$V_{TX\text{-}IDLE\text{-}DIFF\ pp}$	—	—	20	mV
$V_{TX\text{-}RCV\text{-}DETECT}$	—	—	0.6	V
$V_{TX\text{-}DC\text{-}CM}$	0	—	2	V
$RL_{TX\text{-}DIFF}$ [3]	TBD	—	—	dB
$RL_{TX\text{-}CM}$ [3]	TBD	—	—	dB
$Z_{TX\text{-}DIFF\text{-}DC}$	80	100	120	Ω
$L_{TX\text{-}SKEW}$	—	—	500 + 2UI	ps
C_{TX}	75	—	200	nF

1. For an explanation of the PCIe parameters, refer to the PCI Express Base Specification, Rev. 1.1, pp 222-230.
2. Programmable.
3. Measured 50 MHz to 1.25 GHz.

Table 4–5 PCIe RX Specifications

Parameter [1]	Min	Typical	Max	Units
$V_{RX\text{-}DIFF\ pp}$	0.1	—	1.2	V
$T_{RX\text{-}EYE}$	0.4	—	—	UI
$V_{RX\text{-}CM\text{-}ACp}$	—	—	150	mV
$RL_{RX\text{-}DIFF}$ [2]	TBD	—	—	dB
$RL_{RX\text{-}CM}$ [2]	TBD	—	—	dB
$Z_{RX\text{-}DIFF\text{-}DC}$	80	100	120	Ω
$Z_{RX\text{-}DC}$	40	50	60	Ω
$Z_{RX\text{-}HIGH\text{-}IMP\text{-}DC}$	300k	—	—	Ω
$V_{RX\text{-}IDLE\text{-}DET\text{-}DIFF\ pp}$	65	—	175	mV
$L_{RX\text{-}SKEW}$	—	—	20	ns

1. For an explanation of the PCIe parameters, refer to the PCI Express Base Specification, Rev. 1.1, pp 222-230.
2. Measured 50 MHz to 1.25 GHz.

4.2.3 Two-Wire Serial Interface

This section provides bus cycle timing for all TSWI bus cycles implemented by NITROX® PX CN16XX. The first two figures below both display the same Master Mode WRITE operation. The first figure represents the bus cycle in the form of a signal waveform. The second figure represents the same bus cycle, but in an encoded format, to add clarity. All subsequent TWSI cycle timing figures in this section are shown only in the encoded format.

For TWSI interface I/O signal timing parameters (i.e. setup and hold times), refer to the CMOS33 signal timing section below.

Figure 4–2 Master Write Operation (7-bit Slave Address)

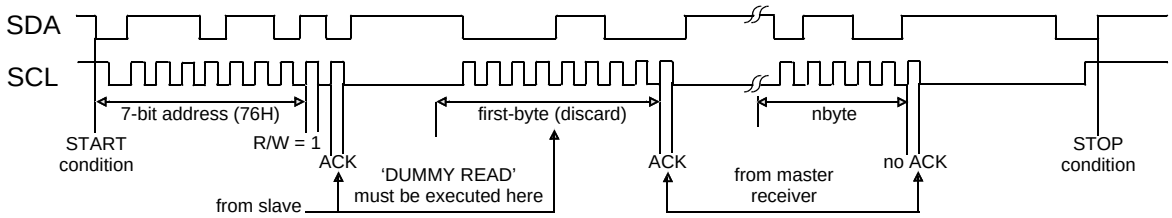


Figure 4–3 Master Write Operation - Encoded Representation

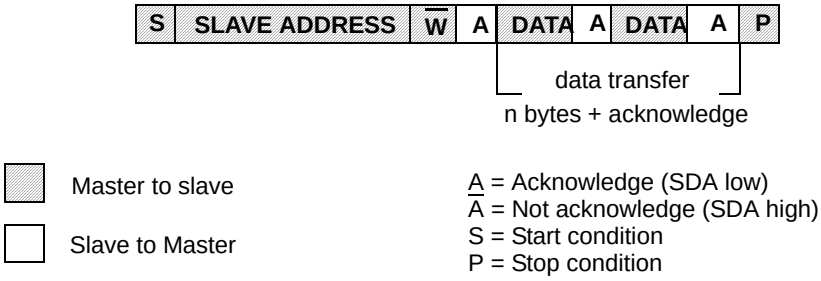


Figure 4–4 Master Read Operation (7-bit Slave Address)

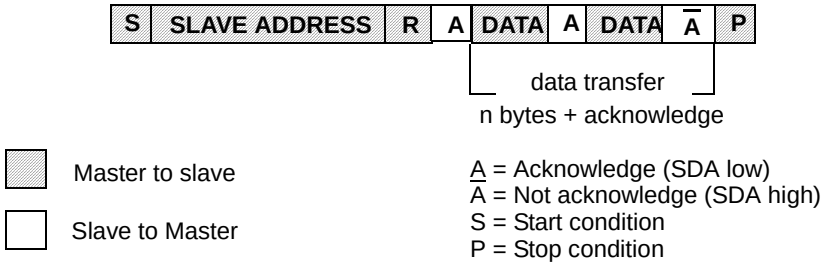


Figure 4–5 Combined Master Write

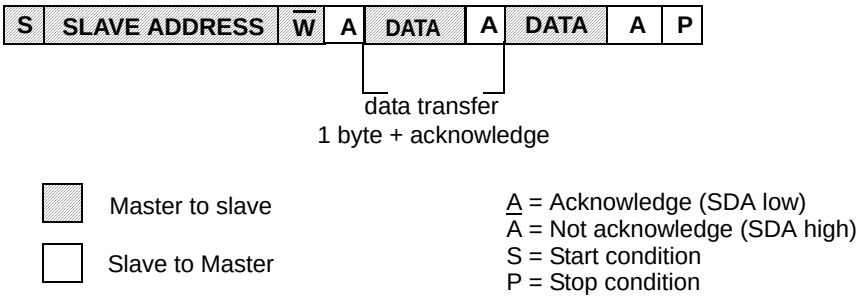


Figure 4–6 Combined Master Read

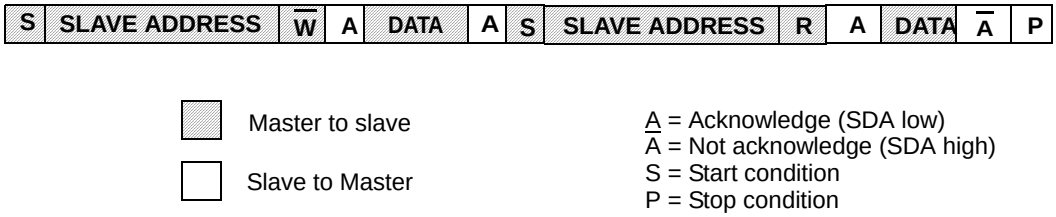


Figure 4–7 Master Write (10-bit)

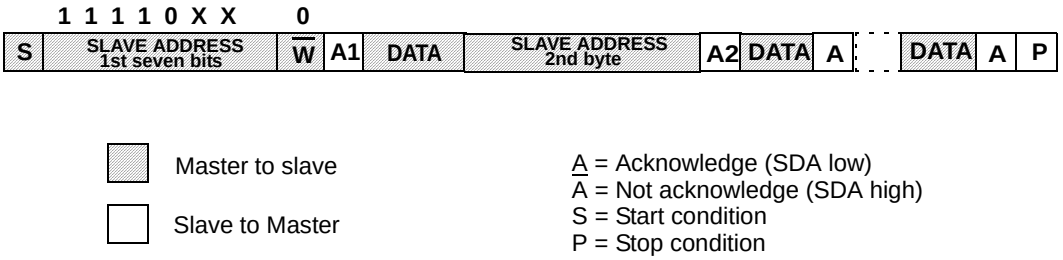
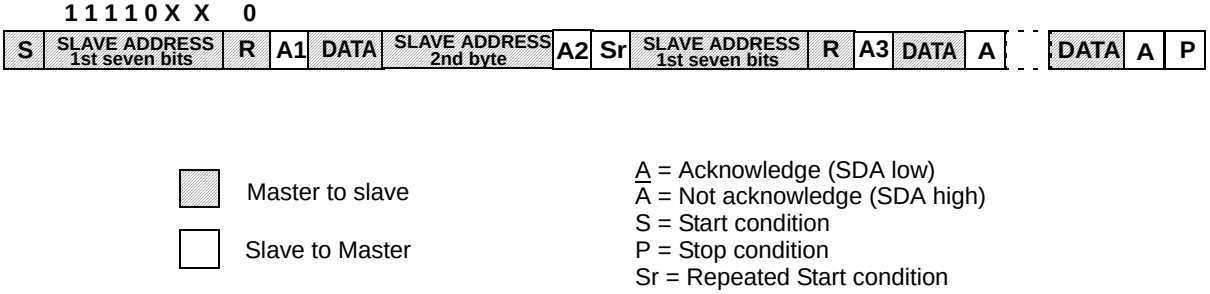


Figure 4–8 Master Read (10-bit)

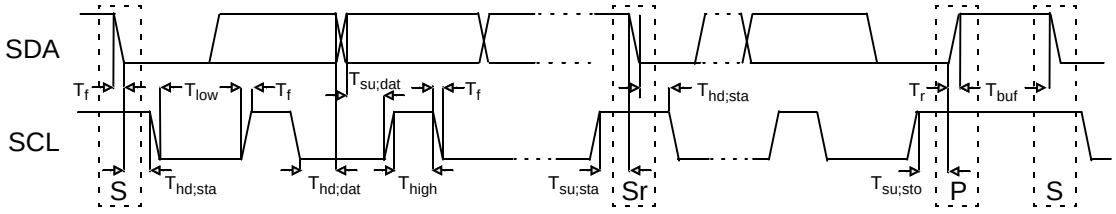


TWSI I/O Signal Timing

Table 4–6 TWSI Signal I/O Timing Parameters

Param	Description	Standard-Mode		Fast-Mode		Units	Notes
		Min	Max	Min	Max		
F_{scl}	SCL clock frequency	0	100	0	400	kHz	
$T_{hd;sta}$	Hold time (repeated) START condition. After this period, the first clock pulse is generated	4.0	—	0.6	—	µs	
T_{low}	LOW period of the SCL clock	4.7	—	1.3	—	µs	
T_{high}	HIGH period of the SCL clock	4.0	—	0.6	—	µs	
$T_{su;sta}$	Set-up time for a repeated START condition	4.7	—	0.6	—	µs	
$F_{hd;dat}$	Data hold time	0	—	0	—	µs	
$T_{su;dat}$	Data set-up time	250	—	100	—	ns	
T_r	Rise time of both SDA and SCL signals	—	1000	20 + 0.1C_b	300	ns	
T_f	Fall time of both SDA and SCL signals	—	300	20 + 0.1C_b	300	ns	
$T_{su;sto}$	Set-up time for STOP condition	4.0	—	0.6	—	µs	
T_{buf}	Bus free time between a STOP and START condition	4.7	—	1.3	—	µs	

Figure 4–9 TWSI Signal Output Timing Diagram



4.2.4 JTAG Interface

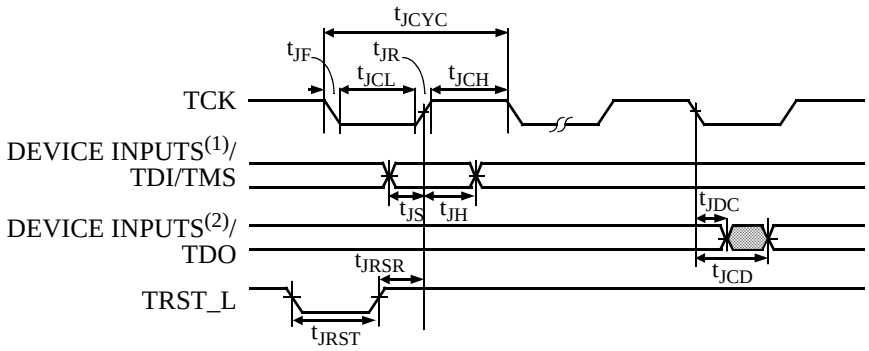
Signal Input/Output Timing

Table 4–7 JTAG Signal I/O Timing Parameters

Parameter	Description	Min	Max	Units	Notes
t_{JCYC}	TCK Cycle time	100	—	ns	
t_{JCH}	TCK Clock HIGH	40	—	ns	
t_{JCL}	TCK Clock LOW	40	—	ns	
t_{JR}	TCK rise time	—	5	ns	1
t_{JF}	TCK fall time	—	5	ns	1
t_{JRST}	TRST_L assert time	80	—	ns	2
t_{JRSR}	TRST_L recovery time	80			
t_{JCD}	TCK to output data valid	5	80	ns	3
t_{JDC}	TCK to output data hold	5	—	ns	1,3
t_{JS}	TCK to input setup time	80	—	ns	4
t_{JH}	TCK to input hold time	80	—	ns	4

1. Guaranteed by design and characterization.
2. RST is an asynchronous level sensitive signal. Setup time is for test purpose only.
3. Output = All device outputs including TDO.
4. Input = All device inputs include TDI and TMS.

Figure 4–10 JTAG TCK Waveform



1. Device inputs = All device inputs except TDI, TMS, and TRST_L.
2. Device outputs = All device outputs except TDO.

4.2.5 AC Test Conditions

Figure 4–11 Output Loading Circuit for AC Timing

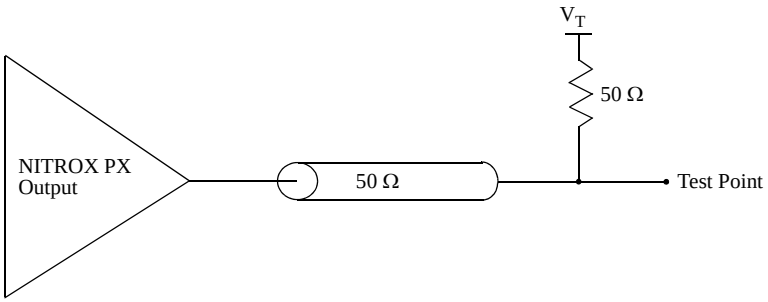


Table 4–8 AC Test Conditions

Parameter	CMOS33
Input pulse levels	0 to3V
Input rise/fall	1ns
Input reference level	1.5V
Output reference levels	1.5V
V_T	1.5V

4.3 DC Characteristics

4.3.1 Absolute Maximum Ratings

Absolute Maximum Ratings are those which if exceeded may cause permanent device failure. Device functionality at or above these limits is not implied. Exposure to Absolute Maximum Ratings for extended periods may affect device reliability.

Table 4–9 Absolute Maximum Ratings

Parameter	Description	Min	Max	Unit
VDD_1V	Core supply voltage	-0.3	1.15	V
EXP_VDD	Express supply voltage	-0.3	1.26	V
EXP_VDDBAND	Express bandgap supply voltage	-0.3	2.625	V
EXP_VDDPLL	Express PLL supply voltage	-0.3	1.25	V
VDD_3V	CMOS33 supply voltage	-0.3	3.6	V
PLL_VDD_3V	PLL supply voltage	-0.3	3.6	V
V_{IN}-3.3	CMOS33 input voltage	-0.3	3.97	V
$T_{STORAGE}$	Storage temperature	–40	125	°C

4.3.2 Recommended Operating Conditions

Recommended Operating Conditions define the limits between which the functionality of the device is guaranteed. Table 4–10 shows the recommended operationg temperatures, and Table 4–11 shows the recommended supply voltages.

Table 4–10 Recommended Operating Temperatures

Parameter	Description	Min	Max	Units
T_J	Junction temperature	N/A	110	°C
T_C	Case temperature (commercial – 350/400 MHz)	0	80	°C
T_C	Case temperature (industrial – 350 MHz)	–40	85	°C

Table 4–11 Recommended Operating Supply Voltages

Parameter	Description	Min	Typ	Max	Units
VDD_1V	Core-voltage supply for 350 MHz	0.95	1.0	1.05	V
VDD_1V	Core-voltage supply for 400 MHz	1.0	1.05	1.1	V
EXP_VDD	PCI Express interface voltage	1.15	1.2	1.25	V
VDD_3V	CMOS33 supply voltage	3.14	3.3	3.46	V
PLL_VDD_3V	PLL-supply voltage	3.14	3.3	3.46	V
EXP_VDDPLL	PCI Express PLL voltage	1.15	1.2	1.25	V
EXP_VDDBAND	PCI Express bandgap voltage	2.375	2.5	2.625	V

4.3.3 Power Sequencing

The CN16XX uses 1/1.05V core technology. Supply voltages must be brought on-line or off-line in a specific sequence to ensure correct device functionality, stability, and reliability.

Power Up:

1. Enable PLL_REF_CLK before or at the same time that VDD_3V is powered. PLL_DCOK must be deasserted low, and PERST_L must be asserted low during powerup.
2. Once VDD_3V is enabled, all PLL voltages, EXP_REF_CLK_P/N, EXP_VDDBAND, and EXP_VDD can be enabled in any order. This may be done before, coincident with, or after VDD_1V is powered up.
3. Power up VDD_1V within 5 ms once VDD_3V is stable.
4. Assert signal PLL_DCOK at least 3 ms after the power supplies are stable. CN16XX samples ENA_400 on the rising edge of PLL_DCOK to determine the internal core-clock frequency.
5. Deassert signal PERST_L high at least 1 ms after PLL_DCOK is asserted.

Figure 4–12 shows the power-up sequence.

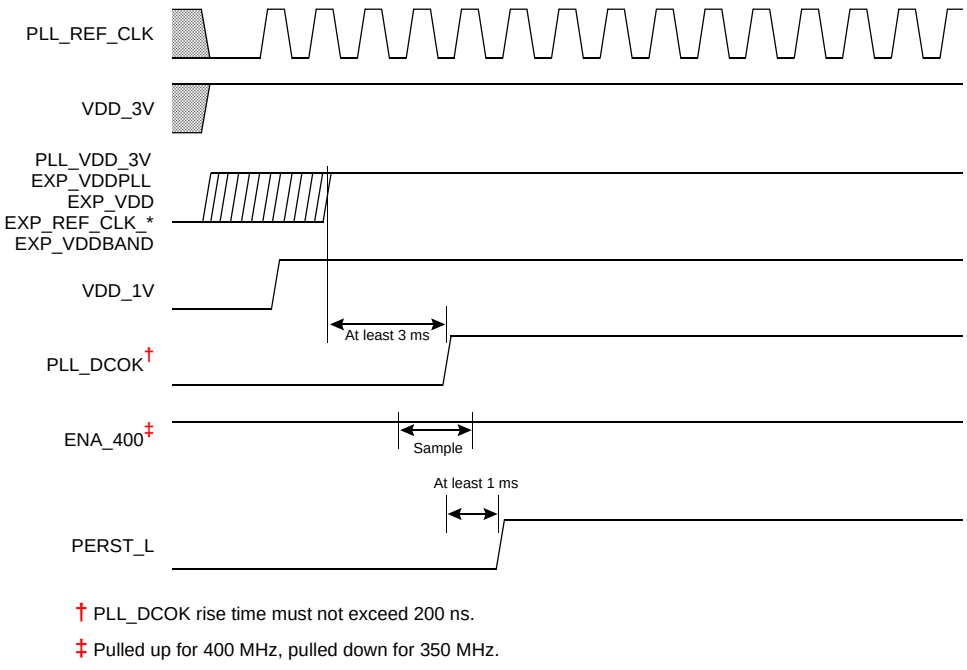


Figure 4–12 Power Sequencing

Power Down:

Deassert PLL_DCOK first. Then the power supplies and reference clocks can be removed in any order without any restrictions.

NOTE: When powering down, PLL_REFC_CLK and EXP_REF_CLK_N/P must not be stopped until .5 ms after PLL_DCOK has been deasserted.

4.3.4 Power Consumption

Table 4–12 lists the CN16XX power-supply specifications.

Table 4–12 CN16XX Core Power-Supply Specification

Frequency @ Voltage	Parameter	Description	Number of Cores				Units
			2	4	6	8	
350MHz @1V (CN1605/10/15)	ICC	ICC supply for VDD_1V	1.8	2.2	2.7	2.8	A
	ICC_{Rst}	ICC supply for VDD_1V (Reset)	1.5	2	2	2	A
400MHz @1.05V (CN1620)	ICC	ICC supply for VDD_1V	2	2.5	3.0	3.2	A
	ICC_{Rst}	ICC supply for VDD_1V (Reset)	2	2.4	2.4	2.4	A

Table 4–13 lists the PCI Express interface power-supply specifications.

Table 4–13 CN16XX I/O Power

I/Os	Voltage	Current
PCIe (4x lanes) (EXP_VDD)	1.2 V	320 mA
EXP_VDDBAND	2.5 V	5 mA
EXP_VDDPLL	1.2 V	75 mA

Table 4–14 lists additional power-supply specifications.

Table 4–14 CN16XX Additional Power Supplies

Description	Voltage	Current
PLL_VDD_3V	3.3 V	20 mA
VDD_3V	3.3 V	10 mA

4.3.5 DC Electrical Characteristics

CMOS33 I/O

Table 4–15 lists CN16XX's 3.3V CMOS bidirectional I/O for the miscellaneous interfaces' electrical characteristics.

Table 4–15 3.3V CMOS33 I/O DC Characteristics

Symbol	Parameter	Min	Nom	Max	Unit
VDD_3V	Interface supply voltage	2.97	3.3	3.63	V
VIH (DC)	DC input logic high	0.5 × VDD_3V	—	VDD_3V + 0.5	V
VIL (DC)	DC input logic low	-0.5	—	0.35 × VDD_3V	V
VOL (DC)	DC output logic low	—	—	0.1 × VDD_3V	V
VOH (DC)	DC output logic high	0.9 × VDD_3V	—	—	V
ILEAK (DC)	DC input leakage current	—	—	1	µA
Rout[1]	Driver output impedance (DC)	20	—	30	Ω

1. The source and sink currents for a driver can be computed for a specified load, and the driver supply voltage.

4.3.6 Package Thermal Specifications

Table 4–16 lists CN16XX's thermal packaging specifications, based on a maximum ambient temperature of 70ºC.

Table 4–16 Thermal Package Specification for CN16XX

Parameter	Description	Max	Units
θ_{JA0}[1]	Thermal resistance – junction to ambient at 0 m/s or 0 LFM	17.9	°C/W
θ_{JA1}[1]	Thermal resistance – junction to ambient at 1 m/s or 200 LFM	16.0	°C/W
θ_{JA2}[1]	Thermal resistance – junction to ambient at 2 m/s or 400 LFM	14.6	°C/W
θ_{JC}	Thermal resistance – junction to case	5.8	°C/W
θ_{JB}	Thermal resistance – junction to board	8.9	°C/W

1. Based on a maximum ambient temperature of 70°C.

Package Thermal Management Requirements

NOTE: Depending on the maximum power dissipated, this device may require the use of a heat sink.

The management of thermal energy due to a microelectronic device's power dissipation is important to receive the best possible performance. The temperature at which a microelectronic device operates determines, among other things, the speed and reliability of the product. Therefore careful consideration of the factors affecting the device's operating temperature is recommended to achieve the best possible results.

The most important factors affecting device-operating temperature are power dissipation, air temperature, package construction, and cooling mechanisms. The combinations of these factors determine the temperature at which the product will operate.

Thermal Definitions:

- **T_A: Ambient temperature (°C)**. Temperature of the ambient air around the device.

  $$T_A = T_C - P \times (\theta_{JA} - \theta_{JC})$$

- **T_C: Case temperature (°C)**. Temperature of the case of the device package. The measurement is made by help of thermocouples placed on the warmest point of the package (in the middle of the upper cover).

  $$T_C = T_A + P \times (\theta_{JA} - \theta_{JC})$$

- **P: Total power dissipation of the device (W)**. The sum of power dissipation from all device power supplies.

- **θ_{JA}:** Thermal resistance from the junction to the environment (°C/W)

  $$\theta_{JA} = \theta_{JC} + \theta_{CA}$$

- **θ_{JB}:** Thermal resistance from the junction to the board (°C/W)

- **θ_{JC}:** Thermal resistance from the junction to the case (°C/W)

- **θ_{CA}:** Thermal resistance from the surface of the cover to the environment (°C/W)

  $$\theta_{CA} = (T_C - T_A) / P$$

- **θ_{SA}:** Thermal resistance from the heat sink to the ambient air (°C/W)

  $$\theta_{SA} = \theta_{CA} - \theta_{CS}$$

- **θ_{CS}:** Thermal resistance of the gluing compound between the case and the heat sink (e.g. thermal compound) (°C/W)

Chapter 5

Registers

This section provides a detailed reference for much of the information required to implement a host software design for NITROX® PX CN16XX. This information includes an overview and map of all internal registers, a detailed description of each register and its individual fields, and an introduction to the basic CN16XX Instruction Set syntax. The CN16XX Instuction Set (opcodes) is implemented in microcode that is downloaded at system boot time, and so the instruction set is variable. The details of the various flavors of CN16XX instruction sets are not documented in this manual. For details of about these instruction sets, please refer to the appropriate Instruction Set Manual, listed in the appendix of this document.

This chapter introduces:

5.1 Register Summary

This section provides a quick reference list of all registers in the Host System Interface address map.

5.1.1 BAR-Address-Mapped Registers

The registers in this section are addressed by adding their specific register-offset value to the base-address value set in the corresponding PCIe configuration space BAR0 register. All BAR0 registers are accessed using PCIe memory space commands. The BAR-mapped registers are shown in Table 5–1.

Table 5–1 BAR-Mapped Registers

Offset	Register Name
BAR0 + 00h	COMMAND/STATUS Register
BAR0 + 08h	PLL_BIST Register
BAR0 + 10h	CORE ENABLE Register
BAR0 + 18h	UCODE LOAD Register
BAR0 + 20h	INTERRUPT ENABLE Register
BAR0 + 28h	INTERRUPT STATUS Register
BAR0 + 30h	CORE ERROR ADDRESS Register
BAR0 + 38h	CORE ERROR STATUS Register
BAR0 + 40h	REF_CLK_CNTR_HIGH Register
BAR0 + 48h	REF_CLK_CNTR_LOW Register
BAR0 + 50h	Reserved
BAR0 + 58h	TWSI-HOST Data Register
BAR0 + 60h	HOST-TWSI Data Register
BAR0 + 68h	INTERNAL STATUS Register
BAR0 + 70-F8h	Reserved
BAR0 + 100h	IQM_0 BASE ADDRESS HIGH Register
BAR0 + 108h	IQM_0 BASE ADDRESS LOW Register
BAR0 + 110h	IQM_0 QUEUE SIZE Register
BAR0 + 118h	IQM_0 DOORBELL Register
BAR0 + 120h	IQM_1 QUEUE BASE ADDRESS HIGH Register
BAR0 + 128h	IQM_1 QUEUE BASE ADDRESS LOW Register
BAR0 + 130h	IQM_1 QUEUE SIZE Register
BAR0 + 138h	IQM_1 DOORBELL Register
BAR0 + 140h	IQM_2 QUEUE BASE ADDRESS HIGH Register
BAR0 + 148h	IQM_2 QUEUE BASE ADDRESS LOW Register
BAR0 + 150h	IQM_2 QUEUE SIZE Register
BAR0 + 158h	IQM_2 DOORBELL Register
BAR0 + 160h	IQM_3 QUEUE BASE ADDRESS HIGH Register
BAR0 + 168h	IQM_3 QUEUE BASE ADDRESS LOW Register
BAR0 + 170h	IQM_3 QUEUE SIZE Register
BAR0 + 178h	IQM_3 DOORBELL Register

Table 5–1 BAR-Mapped Registers (Continued)

Offset	Register Name
BAR0 + 180h	IQM WATCHDOG TIMEOUT CONFIG Register
BAR0 + 188–1F8h	Reserved
BAR0 + 200h	IQM0_NEXT_ADDR_HIGH Register
BAR0 + 208h	IQM0_NEXT_ADDR_LOW Register
BAR0 + 210h	IQM1_NEXT_ADDR_HIGH Register
BAR0 + 218h	IQM1_NEXT_ADDR_LOW Register
BAR0 + 220h	IQM2_NEXT_ADDR_HIGH Register
BAR0 + 228h	IQM2_NEXT_ADDR_LOW Register
BAR0 + 230h	IQM3_NEXT_ADDR_HIGH Register
BAR0 + 238h	IQM3_NEXT_ADDR_LOW Register
BAR0 + 240h	IQM0_CMD_CNT_HIGH Register
BAR0 + 248h	IQM0_CMD_CNT_LOW Register
BAR0 + 250h	IQM1_CMD_CNT_HIGH Register
BAR0 + 258h	IQM1_CMD_CNT_LOW Register
BAR0 + 260h	IQM2_CMD_CNT_HIGH Register
BAR0 + 268h	IQM2_CMD_CNT_LOW Register
BAR0 + 270h	IQM3_CMD_CNT_HIGH Register
BAR0 + 278h	IQM3_CMD_CNT_LOW Register
BAR0 + 280h	GENINT_COUNT_THOLD Register
BAR0 + 288h	GENINT_COUNT_INT_TIME Register
BAR0 + 290h	GENINT_COUNT Register
BAR0 + 298h	GENINT_COUNT_TIMER Register
BAR0 + 2A0h	GENINT_COUNT_SUB Register
BAR0 + 2A8h	REG_EXEC_GROUP Register
BAR0 + 300h	EPC_EFUS_RCMD Register
BAR0 + 320h	EPC_EFUS_SPR_REPAIR_SUM Register
BAR0 + 328h	EPC_EFUS_SPR_REPAIR_RES0 Register
BAR0 + 330h	EPC_EFUS_SPR_REPAIR_RES1 Register
BAR0 + 338h	EPC_EFUS_SPR_REPAIR_RES2 Register
BAR0 + 348h	EPC_ADDR_INDEX Register
BAR0 + 350h	EPC_EFUS_CORE_EN Register
BAR0 + 358h	EPC_EFUS_CHIPID Register
BAR0 + 35C–3FCh	Reserved
BAR0 + 400h	PCIE_CTL Register
BAR0 + 408h	PCIE_INT_SUM Register
BAR0 + 410h	PCIE_INT_ENB Register
BAR0 + 418h	PCIE_TO_STATUS Register
BAR0 + 420h	PCIE_CPL_ERR_STATUS1 Register
BAR0 + 428h	PCIE_CPL_ERR_STATUS2 Register

Table 5–1 BAR-Mapped Registers (Continued)

Offset	Register Name
BAR0 + 430h	PCIE_CPL_ERR_STATUS3 Register
BAR0 + 438h	PCIE_BIST_STATUS Register
BAR0 + 440–4C8h	PCIE_DBG_DATA Registers

5.1.2 PCI Express Configuration/Status Registers

Table 5–2 shows the set of PCIe configuration-space registers. These registers are only accessible using PCIe configuration-space commnands.

Table 5–2 PCI Express Configuration/Status Registers

Address	Byte 3	Byte 2	Byte 1	Byte 0
00h	Device ID		Vendor ID	
04h	Status		Command	
08h	Class Code			Revision ID
0Ch	BIST	Header Type	Latency Timer	Cache LineSize
10h	Base Address Register 0			
14h				
18h	Read as 00000000h			
1Ch				
20h	Read as 00000000h			
24h				
28h	Read as 0000h			
2Ch	SubSystem ID		SubSystem Vendor ID	
30h	Read as 0000h			
34h	Reserved			Capabilities Ptr
38h	Reserved			
3Ch	MAX Latency	MIN Grant	Interrupt PIN	Interrupt Line
40h	Power Capabilities Register			
44h	Power Control Register			
48h–4Ch	Reserved			
50h	MSI Capabilities Register			
54h	MSI Lower Address Register			
58h	MSI Upper Address Register			
5Ch	MSI Message Data Register			
60h–6Ch	Reserved			
70h	PCI Express Capabilities Register			
74h	Device Capabilities Register			
78h	Device Control Register		Device Status Register	
7Ch	Link Capabilities Register			
80h	Link Control Register		Link Status Register	
84h	Slot Capabilities Register			

Table 5–2 PCI Express Configuration/Status Registers (Continued)

Address	Byte 3	Byte 2	Byte 1	Byte 0
88h	Slot Status Register		Slot Control Register	
8Ch–C0h	Reserved			
100h	PCI Express Enhanced Capability Header			
104h	Uncorrectable Error Status Register			
108h	Uncorrectable Error Mask Register			
10Ch	Uncorrectable Error Severity Register			
110h	Correctable Error Status Register			
114h	Correctable Error Mask Register			
118h	Advanced Error Capabilities/Control Register			
11Ch	Header Log Register (First DWORD)			
120h	Header Log Register (Second DWORD)			
124h	Header Log Register (Third DWORD)			
128h	Header Log Register (Fourth DWORD)			
12Ch–6FCh	Reserved			
700h	Ack Latency Timer/Replay Timer Register			
704h	Other Message Register			
708h	Port Force Link Register			
70Ch	Ack Frequency Register			
710h	Port Link Control Register			
714h	Lane Skew Register			
718h	Symbol Number Register			
71Ch	Symbol Timer Register and Filter Mask Register 1			
720h	Filter Mask Register 2			
724h	Reserved			
728h	Debug Register 0			
72Ch	Debug Register 1			
730h	Transmit Posted FC Credit Status Register			
734h	Transmit Non-Posted FC Credit Status Register			
738h	Transmit Completion FC Credit Status Register			
73Ch	Queue Status Register			
740h–744h	Reserved			
748h	VC0 Posted Receive Queue Control Register			
74Ch	VC0 Non-Posted Receive Queue Control Register			
750h	VC0 Completion Receive Queue Control Register			

5.2 Detailed Register Descriptions

Table 5–3 defines the register-bit access type. .

Table 5–3 Register-Bit Access

Type	Meaning
R	**Read Only** - Read returns current value. Write is ignored.
W	**Write Only** - Write modifies the value. Read returns 0.
R/W	**Read, Write** - Read returns the current value. Write modifies the value.
W1C	**Read, Write 1 to clear -** Read returns current status. Writing 1s clears the bits to 0; writing 0s has no effect.
WdR	**Write disabled, Read -** Write only modifies register value if the corresponding processor core is disabled. Read returns current value.

A reset value of X means undefined.

5.2.1 BAR-Mapped Register Details

The registers in this section are addressed by adding their specific register-offset value to the base-address value set in the PCI Express configuration space BAR0 register.

COMMAND Register

Offset: BAR0 + 00h

Description	Bits	Type	Reset Value
Reserved	31:26	R	X
REQ-PFIX - When set '1' the priority for servicing the IQMs (0-3) will be fixed where 0 has highest-priority and 3 has lowest. When clear '0' the 4 request engines will be serviced in a round-robin fashion.	25	R/W	0
RND_DISABLE - Disables the RNG unit. When this bit is set, the RNG unit is reset and disabled. When cleared, RNG unit is enabled.	24	R/W	0
REVISION[7:0] - The revision number of the device. The default value is 08h for CN15XX or 00h for CN16XX.	23:16	R	00h
Reserved	15:12	R	X
KEY MEMORY PARITY INVERT - When set, inverts the Key Memory Parity (forces bad parity for test purposes).	11	R/W	0
RESERVED	10	R/W	0
RND ENTROPY ENABLE - When enabled, the entropy source feeds random data to the Random Number Generator. When disabled, the entropy source feeds zero to the Random Number Generator. 0 = Disabled 1 = Enabled The PSE_MODE input pin overrides this bit. When the PSE_MODE input pin is set, the ***RND ENTROPY ENABLE*** bit is internally forced to one.	9	R/W	0
NOSNOOP - The value of this bit is driven onto the NS attribute field during all PCIe-initiated transactions. 0 = Snoop Enabled 1 = Snoop Disabled *Restrictions:* This bit must be set before enabling the processor cores and IQM in the CORE ENABLE Register.	8	R/W	0
Relaxed Order Writes - Sets the Relaxed Order bit during PCIe writes initiated by CN16XX *Restrictions:* This bit must be set before enabling the processor cores and IQM.	7	R/W	0
Relaxed Order Reads - Sets the Relaxed Order bit during PCIe reads initiated by CN16XX *Restrictions:* This bit must be set before enabling the processor cores and IQM.	6	R/W	0
Soft Reset - Resets the core clock domain systems of NITROX® PX CN16XX, bringing the chip to a reset state, without affecting the Host System Interface. This bit always reads as 0.	5	W	0
Reserved	4	R	X
Outbound Data Swap - The 64-bit outbound data bus can be viewed as either a single 64-bit lane or as 2x 32-bit lanes, which are either byte or long-word (LW) swapped, based on the following encodings. For all cases, assume the original 64-bit data is represented as 8 byte lanes labeled [A-B-C-D-E-F-G-H] 00 = PASS_THRU Mode [A-B-C-D-E-F-G-H] Data is passed through (no swap) 01 = 64b_BYTE_SWAP Mode [H-G-F-E-D-C-B-A] The single QW (64b) lane is byte-swapped. 10 = 32b_BYTE_SWAP Mode [D-C-B-A-H-G-F-E] The 2 LW (32-bit) lanes are byte-swapped. 11 = 32b_LW_SWAP Mode [E-F-G-H-A-B-C-D] The 2x-32b lanes are LW-swapped. *Restrictions:* This bit must be set before enabling the processor cores and IQM. **NOTE**: These registers control data swapping only for PCIe bus mastering DMA operations, including reading of Instructions and access to Dptr, Rptr, and Cptr data. PCIe target operations and endian of the bus are not affected by these registers	[3:2]	R/W	00b

Description	Bits	Type	Reset Value
Inbound Data Swap - The 64-bit inbound data bus can be viewed as either a single 64-bit lane or as 2x 32-bit lanes, which are either byte or long-word (LW) swapped, based on the following encoding. For all cases, assume the original 64b data is represented as 8 byte lanes labeled [A-B-C-D-E-F-G-H] 00 = PASS_THRU Mode [A-B-C-D-E-F-G-H] Data is passed through (no swap) 01 = 64b_BYTE_SWAP Mode [H-G-F-E-D-C-B-A] The single QW (64-bit) lane is byte-swapped. 10 = 32b_BYTE_SWAP Mode [D-C-B-A-H-G-F-E] The 2 LW (32-bit) lanes are byte-swapped. 11 = 32b_LW_SWAP Mode [E-F-G-H-A-B-C-D] The 2x-32-bit lanes are LW-swapped. *Restrictions:* This bit must be set before enabling the processor cores and IQM. **NOTE**: These registers control data swapping only for PCIe bus mastering DMA operations, including reading of Instructions and access to Dptr, Rptr, and Cptr data. PCIe target operations and endian of the bus are not affected by these registers	[1:0]	R/W	00b

PLL BIST Register

Offset: BAR0 + 08h

Description	Bits	Type	Reset Value
Reserved	31:18	R	0
RC_ENB - Reference- and Core-Clock Counter Enable - When 1, the REF_CLK_CNTR_HIGH/LOW and ECLK_CLK_CNTR_HIGH/LOW registers increment every cycle.	17	RW	0
BW_CTL - Core-PLL Control Register	16:12	RW	0
RESERVED	11:4	R	0
RH_BIST - Request High Memory BIST Status	3	R	X
RL_BIST - Request Low Memory BIST Status	2	R	X
KH_BIST - Key High Memory BIST Status	1	R	X
KL_BIST - Key Low Memory BIST Status	0	R	X

CORE ENABLE Register

Offset: BAR0 + 10h

Description	Bits	Type	Reset Value
IQM ENABLE[3:0] - When any bit in this field is set, the corresponding Instruction Queue Manager (IQM) is enabled and NITROX PX can begin to process instructions fed to this IQM. IQMs can be disabled and re-enabled. The base address, size, and watchdog configurations for an IQM should only be written when the IQM is disabled.	31:28	R/W	0
Reserved	27:8	R	0
CORE ENABLE[7:0] - When any bit in this field is set, the corresponding processor core is enabled and ready to process instructions. When a bit in this register is cleared, the corresponding processor core is reset and disabled. In addition, writes to the UCODE LOAD register will write the control store of any/all processor cores for which this bit is cleared. The quantity and identity (core number) of processor cores present in any given CN16XX device is part number and manufacturing lot-specific. Contact your Cavium sales or technical support representative for details of how to identify available cores in a CN16XX device. The NITROX® PX CN16XX CORE ENABLE register should not be written (enabling or disabling cores) until CN16XX has reached a completely idle, or quiescent state. As this core mask corresponds to usable cores, the bit positions in the core mask do not correspond to the bit positions in EPC_EFUS_CORE_EN[EN]. Usable cores are always contiguously numbered 0 .. N-1.	7:0	R/W	0000000h

UCODE LOAD Register

Offset: BAR0 + 18h

The UCODE LOAD register loads microcode into the internal microsequencer memory of any disabled GigaCipher Processor Core. (see CORE ENABLE Register). In general, once microcode is written, it cannot be read back. However when read, this register will return the most recently written value.

Software must ensure that back-to-back writes of this register do not overrun internal logic. After writing this register, wait 40 or more core clock cycles (PLL_REF_CLK × C_CLKMUL) before writing this register again. The UCODE LOAD register is not accessible from the Host System Interface when the chip is operating in PSE Mode (controlled by the PSE_MODE pin). It is accessible from the TWSI interface in PSE Mode however.

Description	Bits	Type	Reset Value
Reserved	31	R	X
MCODE ADDRESS[13:0] - Word (16-bit) address microcode instruction word to be written. Only address 0x0000 to 0x2FFF are valid (12K Words). Addresses 0x3000 to 0x3FFF should not be used.	30:17	W	000h
MCODE PARITY - Even parity over MCode Data [15:0].	16	W	0
MCODE DATA[15:0] - 16-bit microcode instruction word to be written to microcode memory.	15:0	W	0000h

INTERRUPT ENABLE Register

Offset: BAR0 + 20h

This register is used to enable/disable individual CN16XX interrupt conditions indicated by bits in the INTERRUPT STATUS register (BAR0 + 28h). This register works in conjunction with the PCIE_INT_ENB Register/PCIE_INT_SUM Register.

An interrupt is enabled when the corresponding interrupt enable bit is set. Disabling of an interrupt does not stop the corresponding interrupt status bit from being set in the corresponding status register, but does inhibit CN16XX from causing an interrupt on the Host System Interface. Therefore it is recommended to clear all interrupt status bits before enabling interrupts via this register.

A detailed description of the interrupt bits corresponding to their respective interrupt enable bit, is provided in the respective status register.

CN16XX interrupts are reported through PCIe INTA emulation or through the MSI scheme (Message Signaled Interrupt) if enabled.

Description	Bits	Type	Reset Value
Reserved. Read as 0.	31:23	R	X
INTERRUPT STATUS[GI_TIM] Interrupt Enable (i.e bit <18>)	22	R/W	0
INTERRUPT STATUS[GI_CNT] Interrupt Enable (i.e bit <17>)	21	R/W	0
IQM PARITY ERROR Interrupt Enable	20	R/W	0
RESERVED	19:14	R/W	0h
KEY MEMORY ZEROED Interrupt Enable	13	R/W	0
IQM[3:0] COUNTER OVERFLOW Interrupt Enable	12:9	R/W	0h
IQM[3:0] WATCH DOG TIMEOUT Interrupt Enable	8:5	R/W	0h
CORE WATCH DOG TIMEOUT Interrupt Enable	4	R/W	0
MICROCODE GENERAL INTERRUPT Interrupt Enable	3	R/W	0
KEY MEMORY PARITY ERROR Interrupt Enable	2	R/W	0
CORE INSTRUCTION PARITY ERROR Interrupt Enable	1	R/W	0
CORE DATA PARITY ERROR Interrupt Enable	0	R/W	0

INTERRUPT STATUS Register

When an interrupt condition occurs, the corresponding bit in this register will be set. This register can then be read to determine the cause of the interrupt. Writing a 1 to any bit in this register will clear the associated interrupt condition.

Offset: BAR0 + 28h

Description	Bits	Type	Reset Value
Reserved	31:19	R	X
GI_TIM - Set by the HW whenever the GENINT_COUNT_TIMER register is equal to or larger than the GENINT_COUNT_INT_TIME register. Writing a 1 to this bit clears the GENINT_COUNT_TIMER CSR.	18	W1C	0
GI_CNT - Set by the hardware whenever the GENINT_COUNT register is equal to or larger than the GENINT_COUNT_THOLD register.	17	W1C	0
IQM PARITY ERROR - A parity error has been detected during a read of an IQM's internal instructruction-queue memory.	16	W1C	0
RESERVED	15	W1C	0
RESERVED	14	W1C	0
KEY MEMORY ZEROED - indicates that PSE_ZERO_KEYS input signal is asserted.	13	W1C	0
IQM DOORBELL COUNTER OVERFLOW[3:0] - An instruction queue has detected an Instruction Queue counter overflow, caused by the most recent write to its respective doorbell register. This error implies that host software has lost count of the number of outstanding instructions to be processed. Refer to the DOORBELL register description for further information about IQM Doorbell counters.	12:9	W1C	0h
IQM WATCH DOG TIMEOUT[3:0] - An instruction queue has timed out while awaiting completion of an instruction Read operation on the Host System Interface. The timer/counter timeout period is set and enabled in the IQM WATCHDOG TIMEOUT CONFIG register.	8:5	W1C	0h
CORE WATCH DOG TIMEOUT - A processor core has timed out while awaiting completion of an event such as a read or write DMA operation on the Host System Interface. This error indicates that there is probably an associated error elsewhere in the device or system software.	4	W1C	0
MICROCODE GENERAL INTERRUPT - Indicates that one or more GPC cores have set their interrupt bit under microcode control. A GPC core may set this bit when it detects an error, or when it completes execution of an instruction if microcode supports it. Refer to the Nitrox microcode Instruction Set Manual for the microcode being used, for a detailed description of how this bit is used in that microcode.	3	W1C	0
KEY MEMORY PARITY ERROR - Internal Key memory has detected a parity error during a read operation.	2	W1C	0
CORE INSTRUCTION PARITY ERROR - A processor core has detected a parity error during a microcode control store read operation. See CORE ERROR STATUS and CORE ERROR ADDRESS register descriptions below for additional information.	1	W1C	0
CORE DATA PARITY ERROR - A processor core has detected a parity error during a register file read operation. See CORE ERROR STATUS and CORE ERROR ADDRESS register descriptions below for additional information. If this error is detected during a HSI (DMA) Write of data that has been read from a processor core register file, the complete DMA Write will be returned to the CPU Host, with the data written as an "all E" pattern from the point of error through the entire negotiated <byte_count> of write data. The Host CPU can identify the specific register file read in error by interrogating the returned write data to find the "all E' pattern.	0	W1C	0

CORE ERROR ADDRESS Register

Offset: BAR0 + 30h

Description	Bits	Type	Reset Value
CORE ERROR ADDRESS[31:0] - equals the status of the lower 32-bits of the 64-bit instruction Rptr, in a processor core at the time that a hardware error or general microcode interrupt was detected and signaled by that core. The identity of the processor core that detected the error can be identified via the CORE ERROR STATUS Register (see below). If multiple cores have detected a hardware error, this register will contain the error address of the most recent hardware error. Refer to Figure 2–3.	31:0	R	X

CORE ERROR STATUS Register

Offset: BAR0 + 38h

Description	Bits	Type	Reset Value
Reserved	31:8	R	X
CORE ERROR STATUS[7:0] - identifies any processor core that has detected a core hardware error (core BIST error, core instruction or data memory read parity error, or general microcode interrupt). When errors occur, an error address will be written to the CORE ERROR ADDRESS register. If multiple bits are set in this register, multiple cores have had hardware errors. When this occurs there is no deterministic way to detect which error is associated to the value written to the Error Address register.	7:0	W1C	0000000h

REF_CLK_CNTR_HIGH Register

Offset: BAR0 + 40h

Description	Bits	Type	Reset Value
CNT_H - contains the most-significant byte of the reference clock counter. This value is incremented every ref-clock cycle when RC_ENB bit 17 of the PLL_BIST register is set to 1.This register is set to 00000000h when reset is asserted or the STAT_CNTRL (0xC8) bit[0] is set to 1.	31:0	R	00000000h

REF_CLK_CNTR_LOW Register

Offset: BAR0 + 48h

Description	Bits	Type	Reset Value
CNT_L - contains the least-significant byte of the reference clock counter. This value is incremented every ref-clock cycle when RC_ENB bit 17 of the PLL_BIST register is set to 1.This register is set to 00000000h when reset is asserted or the STAT_CNTRL (0xC8) bit[0] is set to 1.	31:0	R	00000000h

TWSI-HOST Register

Offset: BAR0 + 58h

This register is used to transfer information from the Two-Wire-Serial-Interface (TWSI) to the NITROX® PX CN16XX Host System Interface. For a detailed description of this register, and how it is used for TWSI-HSI communications, see section 5.3.1, “TWSI Commands and Operations”, later in this chapter.

Description	Bits	Type	Reset Value
See section 5.3.1, “TWSI Commands and Operations” for detailed description of this register.	31:0	R	00000000h

HOST-TWSI Register

Offset: BAR0 + 60h

This register is used to transfer information and commands from the Host System Interface to the Two-Wire-Serial-Interface (TWSI). For a detailed description of this register, and how it is used for TWSI-HSI communications, see section 5.3.1, "TWSI Commands and Operations", later in this chapter.

Description	Bits	Type	Reset Value
See Section 5.3.1, "TWSI Commands and Operations" for details of this field.	31:0	R/W	00000000h

INTERNAL STATUS Register

Offset: BAR0 + 68h

This register is reserved for Cavium Networks internal use only. It indicates the state of various internal functional blocks within NITROX® PX CN16XX.

Description	Bits	Type	Reset Value
Reserved	31:29	R	0h
INTERNAL STATUS[16:0] - Returns the state of functional blocks internal to CN16XX, as selected by the ***FUNCTION SELECT*** field of this register.	28:12	R	00000h
Reserved. Set to 0 always.	11:10	R/W	0h
FUNCTION SELECT[9:0] - selects the functional block within the device to be returned on the ***INTERNAL STATUS*** field of this register. Write to this field to define the field read in INTERNAL STATUS above. The fields are as follows: 000h–782h = Reserved 783h = bit 12 = HSI Outbound FIFO Empty 784h–7CBh: Reserved 7CCh: bits 7–0 = Instr. pending Core idle status for GPC 7–0	9:0	R/W	3A0h

After enabling all GigaCipher processor cores, but before enabling the IQM Unit, setting FUNCTION SELECT to 0x7CC can be used to determine which GigaCipher processor cores are present during initialization.

- Write 0x7CC to FUNCTION SELECT
- Read INTERNAL STATUS to detect if GigaCipher processor cores 7–0 are idle and present. It is a corresponding bit mask, and a bit set means the core is idle and ready.

This is used in initialization and for soft reset. Before performing a soft reset, software first ensures that no instructions are pending completion (check that the cores idle and ready). Then verify that the output FIFO is empty, indicating that NITROX is in a quiescent state as follows:

- Write 0x783 to FUNCTION SELECT.
- Wait for INTERNAL STATUS[12] = 1

IQM_0-3 BASE ADDRESS HIGH Register

The Instruction Queue Manager (IQM) BASE ADDRESS HIGH register contains the upper 32-bits of the pointer to the Instruction Queue located in host memory. Software must initialize this register during power-up. This register must be set *before* enabling the GigaCipher Processor Cores.

Offset: BAR0 + 100h
BAR0 + 120h
BAR0 + 140h
BAR0 + 160h

Description	Bits	Type	Reset Value
IQM BASE ADDRESS[63:32]	31:0	WdR	X

IQM_0-3 QUEUE BASE ADDRESS LOW Register

The Instruction Queue Manager BASE ADDRESS LOW Register contains the lower 32-bits of the pointer to the Instruction Queue located in host memory. Software must initialize this register as part of power-up sequence. This register must be set *before* enabling theGigaCipher Processor Cores.

Offset: BAR0 + 108h
BAR0 + 128h
BAR0 + 148h
BAR0 + 168h

Description	Bits	Type	Reset Value
IQM BASE ADDRESS[31:5]	31:5	WdR	X
IQM BASE ADDRESS[4:0]	4:0	R	00h

IQM_0-3 QUEUE SIZE Register

The IQM QUEUE SIZE Registers are used by NITROX® PX CN16XX to determine the ending address of the Instruction Queues in Host Memory. These registers must be initialized as part of the power-up sequence. They must be set *before* enabling the GigaCipher Processor Cores.

Offset: BAR0 + 110h
BAR0 + 130h
BAR0 + 150h
BAR0 + 170h

Description	Bits	Type	Reset Value
Reserved	31:14	R	00000h
IQM SIZE[13:0] - The number of 32-byte instruction-Entries in the Instruction Queue.	13:0	WdR	X

IQM_0-3 DOORBELL Register

Offset: BAR0 + 118h
BAR0 + 138h
BAR0 + 158h
BAR0 + 178h

The IQM_*n* Doorbell Register is actually a 14-bit counter, containing the total number of outstanding instructions to be fetched from the Host Instruction Queue to NITROX® PX CN16XX internal instruction queue for execution. The host uses this register to notify the corresponding IQM(n) that there are new instructions in host memory to be executed. The host system also uses this register to identify the number of outstanding instructions yet to be read from the external host instruction queue.

When this register is read, all 14 bits of the counter are returned to the Host. When written, the lower 7-bits are added to the current counter value. If the current 14-bit value, plus the newly written 7-bit value, exceeds the 14 bits of precision of the internal counter, an IQM(n) Doorbell Counter Overflow error status bit will be set.

Writing this register with a value other than 0 instructs the IQM to start fetching instructions from System Memory. The IQM's address and size register must be set *before* writing this register.

Description	Bits	Type	Reset Value
Reserved	31:14	R	00000h
DOORBELL[7:13] - Upper 7 bits of the Doorbell register/counter. When read, returns the current value of the upper 7 bits of the internal Doorbell register/counter. This field is read-only.	13:7	R	00h
DOORBELL[6:0] - Lower 7 bits of the Doorbell register/counter. When read, returns the current value of the lower 7 bits of the internal Doorbell register/counter. When written, this 7-bit field will be added to the current value of the internal 14-bit Doorbell register/counter.	6:0	R/W	00h

IQM WATCHDOG TIMEOUT CONFIG Register

Offset: BAR0 + 180h

Description	Bits	Type	Reset Value
IQM TIMEOUT ENABLE - Enables or disables the IQM instruction read watchdog timers. When enabled, An IQM's watchdog timer will start counting down from its initialization value the moment that the IQM issues a read operation to the Host System Interface logic. When a counter reaches zero the corresponding Watchdog Timeout Error status bit will be set in the interrupt status register. 1: Watchdog Timers are enabled for all four IQMs. 0: Watchdog Timers are disabled for all four IQMs.	31	R/W	0
IQM TIMEOUT INIT[30:0] - All IQMs use the value programmed to this register as the initialization value for their respective Watchdog Timers. This value is loaded into the timer when the IQM issues an instruction read operation to the Host System Interface. The counter is then counted down once for each tick of the NITROX® PX CN16XX internal core clock. The counter timeout period is calculated as: $T_{timeout}$= ***(IQM TIMEOUT INIT[31:0])***/*CCLK* where *CCLK* is the frequency of the internal core clock.	30:0	R/W	X

IQM#(0-3)_NEXT_ADDR_HIGH Register

Offset: BAR0 + 200h
BAR0 + 210h
BAR0 + 220h
BAR0 + 230h

Description	Bits	Type	Reset Value
ADDR - Bits [63:32] of the address where the next instruction for Instruction Engine # will be fetched.	31:0	R	X

IQM#(0-3)_NEXT_ADDR_LOW Register

Offset: BAR0 + 208h
BAR0 + 218h
BAR0 + 228h
BAR0 + 238h

Description	Bits	Type	Reset Value
ADDR - Bits [32:0] of the address where the next instruction for Instruction Engine # will be fetched.	31:0	R	X

IQM#(0-3)_CMD_CNT_HIGH Register

Offset: BAR0 + 240h
BAR0 + 250h
BAR0 + 260h
BAR0 + 270h

Description	Bits	Type	Reset Value
CNT - Bits [63:32] of the number of instructions the IQM has attempted to read.	31:0	R	X

IQM#(0-3)_CMD_CNT_LOW Register

Offset: BAR0 + 248h
BAR0 + 258h
BAR0 + 268h
BAR0 + 278h

Description	Bits	Type	Reset Value
CNT - Bits [32:0] of the number of instructions the IQM has attempted to read.	31:0	R	X

GENINT_COUNT_THOLD Register

Offset: BAR0 + 280h

Description	Bits	Type	Reset Value
COUNT - When the value in the GENINT_COUNT register is equal to or greater than this value the GI_CNT bit of the Interrupt Status register will be set.	31:0	RW	FFFFFFFFh

GENINT_COUNT_INT_TIME Register

Offset: BAR0 + 288h

Description	Bits	Type	Reset Value
TIME - When the value in the GENINT_COUNT_TIMER register exceeds or is equal to the value in this register, the GI_TIM bit of the Interrupt Status register will be set.	31:0	RW	FFFFFFFFh

GENINT_COUNT Register

Offset: BAR0 + 290h

Description	Bits	Type	Reset Value
COUNT - This register is incremented by 1 each cycle that any Core asserts the GENINT wire. The value in this register is decreased by using the GENINT_COUNT_SUB register. The instruction and the microcode running on the Cores determine when a Core asserts the GENINT wire. For example, the Cores may pulse GENINT once to indicate instruction completion.	31:0	R	0h

GENINT_COUNT_TIMER Register

Offset: BAR0 + 298h

Description	Bits	Type	Reset Value
COUNT - The value in this register increments every core-clk cycle, when there is a non-zero value in the GENINT_COUNT register. This timer gets cleared back to 0 when INTERRUPT_STATUS[GI_TIM] (bit [18]) is written with a 1.	31:0	R	0h

GENINT_COUNT_SUB Register

Offset: BAR0 + 2A0h

Description	Bits	Type	Reset Value
SUB_VAL - The value written to this register is subtracted from the value in the GENINT_COUNT register.	31:0	RW	0h

REG_EXEC_GROUP Register

Offset: BAR0 + 2A8h

Description	Bits	Type	Reset Value
E7_G30 - Controls which Instruction groups (3,2,1,0) can be processed by Core 7. When a bit is set in this field the cooresponding group will be able to be processed by Core 7.	31:28	RW	0xF
E6_G30 - Controls which Instruction groups (3,2,1,0) can be processed by Core 6. When a bit is set in this field the cooresponding group will be able to be processed by Core 6.	27:24	RW	0xF
E5_G30 - Controls which Instruction groups (3,2,1,0) can be processed by Core 5. When a bit is set in this field the cooresponding group will be able to be processed by Core 5.	23:20	RW	0xF
E4_G30 - Controls which Instruction groups (3,2,1,0) can be processed by Core 4. When a bit is set in this field the cooresponding group will be able to be processed by Core 4.	19:16	RW	0xF
E3_G30 - Controls which Instruction groups (3,2,1,0) can be processed by Core 3. When a bit is set in this field the cooresponding group will be able to be processed by Core 3.	15:12	RW	0xF
E2_G30 - Controls which Instruction groups (3,2,1,0) can be processed by Core 2. When a bit is set in this field the cooresponding group will be able to be processed by Core 2.	11:8	RW	0xF
E1_G30 - Controls which Instruction groups (3,2,1,0) can be processed by Core 1. When a bit is set in this field the cooresponding group will be able to be processed by Core 1.	7:4	RW	0xF
E0_G30 - Controls which Instruction groups (3,2,1,0) can be processed by Core 0. When a bit is set in this field the cooresponding group will be able to be processed by Core 0.	3:0	RW	0xF

EPC_EFUS_RCMD Register

Offset: BAR0 + 300h

Description	Bits	Type	Reset Value
Reserved. Read as 0.	31:24	R	0
DAT - 8 bits of fuse data.	23:16	R	0
Reserved. Read as 0.	15:13	R	0
PEND - SW sets this bit on a write to start FUSE read.	12	RW	0
Reserved. Read as 0.	11:9	R	0
EFUSE - When set, return data from the efuse storage rather than the local storage for the 320 hardware fuses.	8	RW	0
Reserved. Read as 0.	7:5	R	0
ADDR - The byte address of the fuse to read.	4:0	RW	0

EPC_EFUS_SPR_REPAIR_SUM Register

Offset: BAR0 + 320h

Description	Bits	Type	Reset Value
Reserved. Read as 0.	31:1	R	0
TOO_MANY - Too Many Defects - cannot repair - bad part.	0	RO	0

EPC_EFUS_SPR_REPAIR_RES*n* Register, *n* =(0..2)

Offset: BAR0 + 328, 330, 338h

Description	Bits	Type	Reset Value
Reserved. Read as 0.	31:14	R	0
REPAIR_RES - SPR BISR Results.	13:0	RO	0

EPC_ADDR_INDEX Register

Offset: BAR0 + 348h

Description	Bits	Type	Reset Value
Reserved. Read as 0.	31:17	R	0
ADDR_SWAP - When set to 1, the value in [ADDR] (next field) is used for address bits [63:48] of DMA Read/Write operations generated by the Cores. When clear to 0, the address sent by the Cores will be used.	16	RW	0x1
ADDR - Address bits [63:48] of DMA Read/Write operations generated by the Cores. See [ADDR_SWAP] field above.	15:0	RW	0

EPC_EFUS_CORE_EN Register

Offset: BAR0 + 350h

Description	Bits	Type	Reset Value
Reserved. Read as 0.	31:8	R	0
EN - Indicates which physical Cores are fuse enabled. The number of available Cores is the number of set bits. As this core mask corresponds to physical cores, some of which may not be usable, the bit positions in the mask do not correspond to the bit positions in CORE ENABLE <7:0>. Usable cores are always contiguously numbered 0 .. N-1.	7:0	RO	X

EPC_EFUS_CHIPID Register

Offset: BAR0 + 358h

Description	Bits	Type	Reset Value
Reserved. Read as 0.	31:8	R	000000h
ID - revision ID. The default value is 08h for CN15XX or 00h for CN16XX.	7:0	RO	00h

PCIE_CTL Register

Offset: BAR0 + 400h

Description	Bits	Type	Reset Value
Reserved.	31:16	RO	0000h
ENB_PCIE_RST - Enable PCIE reset. This bit is reset by the assertion of the chip reset (PERST_L). When 1, it allows the PCIE logic to be reset when a Hot-Reset or down-link condition occurs in the PCIe Core. Should always be set.	15	RW	1

Description	Bits	Type	Reset Value
ENB_RS_RST - Enable Reset Sync reset. This bit is reset by the assertion of the chip reset (PERST_L). When 1, it allows the chip to be reset when a Hot-Reset or down-link condition occurs in the PCIe Core. Should always be set.	14	RW	1
APP_RDY_ENTR_L23 - input to PCI Express core telling the core that the application is ready to enter the L23 power state. Must be 1.	13	RW	1
DIAG_BUS_CTL[1:0]. When [1] is 1, invert the LSB of LCRC. When [0] is 1, invert the LSB of ECRC.	12:11	RW	0h
FUN_NUM - Function Number. Must be 0h.	10:8	RW	0h
TC - Traffic Class. Must be 0h.	7:5	RW	0h
VECTOR - Unused. Must be 00h.	4:0	RW	00h

PCIE_INT_SUM Register

Offset: BAR0 + 408h

This register is used with the PCIE_INT_ENB Register to enable/disable individual CN16XX interrupt conditions. This register works in conjunction with the Interrupt Enable Register/Interrupt Status Register.

Description	Bits	Type	Reset Value
Reserved.	31:16	R	0000h
LRST - Set when the PCEC asserts link reset. This bit's initial value is controlled only by the chip-reset and does not respond to other reset conditions.	15	W1C	0
TRST- Set when the PCEC asserts training reset. This bit's initial value is controlled only by the chip-reset and does not respond to other reset conditions.	14	W1C	0
Reserved.	13	R	0
ILL_CMD - Set when an unexpected command arrives on the outbound EPC.	12	W1C	0
ILL_ROMA- Set when a ROM access is detected from the PCI Express Core.	11	W1C	0
ILL_IOA- Set when an I/O access is detected from the PCI Express core.	10	W1C	0
PEC_PF- Set when the P2E EPC Control FIFO pushes a full FIFO.	9	W1C	0
PEC_PE- Set when the P2E EPC Control FIFO pops an empty FIFO.	8	W1C	0
ERC_PF- Set when the E2P Register Control FIFO pushes a full FIFO.	7	W1C	0
ERC_PE- Set when the E2P Register Control FIFO pops an empty FIFO.	6	W1C	0
PRC_PF- Set when the P2E Register Control FIFO pushes a full FIFO.	5	W1C	0
PRC_PE- Set when the P2E Register Control FIFO pops an empty FIFO.	4	W1C	0
PIC_PF- Set when the P2E Inbound Control FIFO pushes a full FIFO.	3	W1C	0
PIC_PE- Set when the P2E Inbound Control FIFO pops an empty FIFO.	2	W1C	0
CPL_ERR - Set when the completion status is not normal.	1	W1C	0
CPL_TO - Set when the completion timeout occurs.	0	W1C	0

PCIE_INT_ENB Register

Offset: BAR0 + 410h

Description	Bits	Type	Reset Value
Reserved.	31:16	RO	0000h
LRST - When set, the corresponding bit in the PCIE_INT_SUM register when set generates an interrupt.	15	R/W	0
TRST- When set, the corresponding bit in the PCIE_INT_SUM register when set generates an interrupt.	14	R/W	0
Reserved.	13	RO	0
ILL_CMD - When set, the corresponding bit in the PCIE_INT_SUM register when set generates an interrupt.	12	R/W	0
ILL_ROMA - When set, the corresponding bit in the PCIE_INT_SUM register when set generates an interrupt.	11	R/W	0
ILL_IOA - When set, the corresponding bit in the PCIE_INT_SUM register when set generates an interrupt.	10	R/W	0
PEC_PF - When set, the corresponding bit in the PCIE_INT_SUM register when set generates an interrupt.	9	R/W	0
PEC_PE - When set, the corresponding bit in the PCIE_INT_SUM register when set generates an interrupt.	8	R/W	0
ERC_PF - When set, the corresponding bit in the PCIE_INT_SUM register when set generates an interrupt.	7	R/W	0
ERC_PE - When set, the corresponding bit in the PCIE_INT_SUM register when set generates an interrupt.	6	R/W	0
PRC_PF - When set, the corresponding bit in the PCIE_INT_SUM register when set generates an interrupt.	5	R/W	0
PRC_PE - When set, the corresponding bit in the PCIE_INT_SUM register when set generates an interrupt.	4	R/W	0
PIC_PF - When set, the corresponding bit in the PCIE_INT_SUM register when set generates an interrupt.	3	R/W	0
PIC_PE - When set, the corresponding bit in the PCIE_INT_SUM register when set generates an interrupt.	2	R/W	0
CPL_ERR - When set, the corresponding bit in the PCIE_INT_SUM register when set generates an interrupt.	1	R/W	0
CPL_TO - When set, the corresponding bit in the PCIE_INT_SUM register when set generates an interrupt.	0	R/W	0

PCIE_TO_STATUS Register

Offset: BAR0 + 418h

Description	Bits	Type	Reset Value
Reserved.	31:28	R	0h
TAG - When a completion timeout occurs this field contains the tag of the request that timed out.	27:20	R	00h
FUN_NUM - When a completion timeout occurs this field contains the function-number of the request that timed out.	19:17	R	0h
TC - When a completion timeout occurs this field contains the traffic-class of the request that timed out.	16:14	R	0h
ATTR - When a completion timeout occurs this field contains the attribute of the request that timed out.	13:12	R	0h
LEN - When a completion timeout occurs this field contains the length of the request that timed out.	11:0	R	000h

PCIE_CPL_ERR_STATUS1 Register

Offset: BAR0 + 420h

Description	Bits	Type	Reset Value
REQ_ID - When a completion error occurs this field contains the Request-ID of the request that reported the completion error status	31:16	R	0000h
CPL_ID - When a completion error occurs this field contains the Completion-ID of the request that reported the completion error status.	15:0	R	0000h

PCIE_CPL_ERR_STATUS2 Register

Offset: BAR0 + 428h

Description	Bits	Type	Reset Value
TAG - When a completion error occurs this field contains the tag of the request that reported the completion error status.	31:24	R	00h
EP - When a completion error occurs this field contains the EP field of the Header of the request that reported the completion error status.	23	R	0
LOW_ADDR - When a completion error occurs this field contains the low-address bits of the request that reported the completion error status.	22:16	R	00h
BYTE_CNT - When a completion error occurs this field contains the byte-count of the request that reported the completion error status.	15:4	R	0h
TC - When a completion error occurs this field contains the traffic-class of the request that reported the completion error status.	3:1	R	0h
TD - When a completion error occurs this field contains the TD field of the Header of the request that reported the completion error status.	0	R	0

PCIE_CPL_ERR_STATUS3 Register

Offset: BAR0 + 430h

Description	Bits	Type	Reset Value
Reserved.	31:17	RO	0000h
LEN_DW - When a completion error occurs this field contains the DW -Length of the completion that reported the completion error status.	16:7	RO	000h
STATUS - When a completion error occurs this field contains the status of the request that reported the completion error status.	6:4	RO	0h
BCM - When a completion error occurs this field contains the BCM field of the header of the request that reported the completion error status.	3	RO	0
LAST - When a completion error occurs this field is set if this completion is the last of (1 or multiple) completions for the request that reported the completion error status.	2	RO	0
ATTR - When a completion error occurs this field contains the attribute of the request that reported the completion error status.	1:0	RO	0h

PCIE_BIST_STATUS Register

Offset: BAR0 + 438h

Description	Bits	Type	Reset Value
Reserved.	31:5	RO	0000000h
PCM_DATA - The BIST status of the field named memory.	4	RO	0
PCEC_RBUFF - The BIST status of the field named memory.	3	RO	0
PCEC_QHDR - The BIST status of the field named memory.	2	RO	0
PCEC_QDATA - The BIST status of the field named memory.	1	RO	0
PCB_LMEM - The BIST status of the field named memory.	0	RO	0

PCIE_DBG_DATA Register

Contains a variety of PCI Express core debug signals.

Offset: BAR0 + 440h through BAR0 + 4C8h

Description	Bits	Type	Reset Value
Reserved.	31:0	RO	0x0

5.2.2 PCI Express Configuration/Status Register Details

These registers are described here for reference only. They are compliant with the PCI Express specifications, respectively, listed in the appendix of this document. The reader should refer to that specification for a detailed description of this register and the corresponding bits or bit fields within it.

These registers are accessed only by PCI Express configuration bus cycles, and their addresses are therefore zero-relative. For a detailed description of the configuration bus cycles, see the relevant specification listed in the appendix.

VENDOR/DEVICE Register

Offset: Configuration Address 00h

Description	Bits	Type	Reset Value
DEVICE ID[15:0] - Identifies the Type of Device. Assigned by the device manufacturer (Cavium). Returns a value of 0x00010 when read.	31:16	R	00010h
VENDOR ID[15:0] - (assigned by PCI SIG) Identifies the manufacturer of the device(Cavium). Returns a value of 0x177D when read.	15:0	R	177Dh

COMMAND/STATUS Register

Offset: Configuration Address 04h

Description	Bits	Type	Reset Value
DETECTED PARITY ERROR (Status[15])	31	W1C	0
SIGNALED SYSTEM ERROR (Status[14])	30	W1C	0
RECEIVED MASTER ABORT (Status[13])	29	W1C	0
RECEIVED TARGET ABORT (Status[12])	28	W1C	0
SIGNALED TARGET ABORT (Status[11])	27	W1C	0
Read as Zero (Status[10:9])	26:25	R	0h
MASTER DATA PARITY ERROR (Status[8])	24	W1C	0
Read as Zero (Status[7])	23	R	0
Reserved (Status[6])	22	R	0
Read as Zero (Status[5])	21	R	0
CAPABILITIES LIST ENABLE (Status[4])	20	R	1
INTERRUPT STATUS (Status[3])	19	R	0
Reserved (Status[2:0])	18:16	R	X
Reserved (Command[15:11])	15:11	R	00h
DISABLE INTERRUPT Disables the device/function from asserting INTx_L.	10	R/W	0
Read as Zero (Command[9])	9	R/W	0
SYSTEM ERROR ENABLE (Command[8])	8	R/W	0
Read as Zero (Command[7])	7	R	0
PARITY ERROR RESPONSE (Command[6])	6	R/W	0
Read as Zero (Command[5:3])	5:3	R	0h
MASTER ENABLE (Command[2]). Should be 1.	2	R/W	0
MEMORY SPACE ACCESS ENABLE (Command[1]). Should be 1.	1	R/W	0
I/O SPACE ACCESS ENABLE (Command[0]). Must be 0.	0	R/W	0

CLASS CODE/REVISION ID Register

Offset: Configuration Address 08h

Description	Bits	Type	Reset Value
CLASS CODE[23:0] - (Network Encryption/Decryption Class)	31:8	R	100000h
REVISION ID[7:0]	7:0	R	00h

BIST/Header Type/Latency Timer/Cache Line Size Register

Offset: Configuration Address 0Ch

Description	Bits	Type	Reset Value
BIST CAPABLE (BIST[7]) - Not supported.	31	R	0
BIST REQUEST/BUSY (BIST[6]) - Not Supported	30	R	0
Reserved (BIST[5:4])	29:28	R	0h
BIST CODE[3:0] (BIST[3:0]) - Not Supported.	27:24	R	0h
HEADER TYPE[7:0] - NITROX® PX CN16XX is configured for Type 0 PCI configuration header type.	23:16	R	00h
Read as Zero	15:8	R	00h
CACHE LINE SIZE	7:0	R	00h

Base Address Register 0 (BAR0)

Offset: Configuration Address 10h

Description	Bits	Type	Reset Value
BAR0[31:12] - Memory Base Address #0	31:12	RW	000000h
Read as Zero	11:4	R	01h
PREFETCHABLE	3	R	1
TYPE - Locate anywhere in 64-bit address space.	2:1	R	2h
Read as Zero	0	R	0

SUBSYSTEM ID/SUBSYSTEM VENDOR ID Register

Offset: Configuration Address 2Ch

Description	Bits	Type	Reset Value
SUBSYSTEM ID[15:0]	31:16	R	0001h
SUBSYSTEM VENDOR ID[15:0]	15:0	R	177Dh

CAPABILITIES POINTER Register

Offset: Configuration Address 34h

Description	Bits	Type	Reset Value
Reserved	31:8	R	000000h
CAPABILITIES POINTER[7:0] - NITROX® PX CN16XX implements extended capabilities (required) as allowed in the PCIe specification. This register returns a value of 40h, pointing to capabilities configuration register space starting at address 40h.	7:0	R	40h

INT/ARB/LATENCY Register

Offset: Configuration Address 3Ch

Description	Bits	Type	Reset Value
MAXIMUM LATENCY[7:0]	31:24	R	00h
MINIMUM GRANT[7:0]	23:16	R	00h
INTERRUPT PIN[7:0]. 0h = no interrupt, 1h = INTA, 2h = INTB, 3h = INTC, 4h = INTD	15:8	R	01h
INTERRUPT LINE[7:0]	7:0	R/W	FFh

POWER MANAGEMENT CAPABILITIES Register

Offset: Configuration Address 40h

Description	Bits	Type	Reset Value
PME SUPPORT[4:0] (D0 to $D3_{cold}$)	31:27	R	00h
D2_SUPPORT	26	R	0
D1_SUPPORT	25	R	0
AUX_CURRENT[2:0] (0 to 375mA)	24:22	R	0h
DEVICE SPECIFIC INITIALIZATION	21	R	0
Reserved	20	R	0
PME CLOCK - Hardwired to 0	19	R	0
VERSION[2:0] - Indicates the version of the PCI Bus Power Management Interface specification with which the core complies. 3h: Complies with *PCI Bus Power Management Interface Specification*, Revision 1.2	18:16	R	3h
NEXT CAPABILITY POINTER[7:0]	15:8	R	50h
POWER MANAGEMENT CAPABILITY ID[7:0] Power Management Capability = 01h	7:0	R	01h

POWER MANAGEMENT Control/Status Register

Offset: Configuration Address 44h

Description	Bits	Type	Reset Value
PME_DATA[7:0] - Power Management data input from application. Not supported.	31:24	R	00h
BPCC_EN - bus power/clock control enable. Hardwired to 0	23	R	0
B2_B3#, B2/B3 SUPPORT FOR D3$_{hot}$. Hardwired to 0	22	R	0
Reserved	21:16	R	00h
PME_STATUS sticky bit. Indicates if a previously enabled PME event occurred or not.	15	W1C	0
PME_DATA_SCALE[1:0]. Not supported.	14:13	R	0h
POWER MANAGEMENT DATA_SELECT[3:0]. Not supported.	12:9	R	0h
PME_EN	8	R/W	0
Reserved	7:4	R	00h
DEFAULT NO SOFT RESET	3	R	00h
Reserved	2	R	00h
POWER STATE - (D0 to D3)	1:0	R/W	00h

MSI CAPABILITIES Register

Offset: Configuration Address 50h

Description	Bits	Type	Reset Value
Reserved	31:24	R	0
32/64 B MESSAGE: 0: 32-bit message, 1: 64-bit message	23	R	1
MULTIPLE MESSAGE ENABLE[2:0] - (1,2,4,8,16,32)	22:20	R/W	0h
MULTIPLE MESSAGE CAPABLE[2:0] - (1,2,4,8,16,32)	19:17	R	0h
MSI ENABLE	16	R/W	0
NEXT CAPABILITY POINTER[7:0]	15:8	R	70h
MSI CAPABILITY ID - Message Signaled Interrupt Capability = 05h	7:0	R	05h

MSI LOWER ADDRESS Register

Offset: Configuration Address 54h

Description	Bits	Type	Reset Value
MSI ADDRESS[31:2]	31:2	R/W	00000000h
Reserved	1:0	R	X

MSI UPPER ADDRESS Register

Offset: Configuration Address 58h

Description	Bits	Type	Reset Value
MSI ADDRESS[63:32]	31:0	R/W	00000000h

MSI MESSAGE DATA Register

Offset: Configuration Address 5Ch

Description	Bits	Type	Reset Value
Reserved	31:16	R	X
MSI MESSAGE DATA[15:0]	15:0	R/W	00h

PCI Express Capabilities Register

Offset: Configuration Address 70h

Description	Bits	Type	Reset Value
Reserved	31:30	R	0h
INTERRUPT MESSAGE NUMBER	29:25	R	
Slot implemented.	24	R	0
DEVICE PORT TYPE	23:20	R	
PCI EXPRESS CAPABILITY VERSION	19:16	R	01h
NEXT CAPABILITY POINTER	15:8	R	00h
PCI EXPRESS CAPABILITY ID	7:0	R	10h

Device Capabilities Register

Offset: Configuration Address 74h

Description	Bits	Type	Reset Value
Reserved	31:28	R	0h
CAPTURED SLOT POWER LIMIT SCALE	27:26	R	0h
CAPTURED SLOT POWER LIMIT VALUE	25:18	R	0h
Reserved	17:16	R	0h
ROLE-BASE ERROR REPORTING	15	R	1
Read as Zero	14:12	R	0h
ENDPOINT L1 ACCEPTABLE LATENCY (maximum is 2µs)	11:9	R	1h
ENDPOINT L0S ACCEPTABLE LATENCY (up to 128 ns)	8:6	R	1h
EXTENDED TAG FIELD SUPPORTED	5	R	0
PHANTOM FUNCTION SUPPORTED. Not supported.	4:3	R	0h
MAX PAYLOAD SIZE SUPPORTED (256B)	2:0	R	1h

Device Control/Status Register

Offset: Configuration Address 78h

Description	Bits	Type	Reset Value
Reserved	31:22	R	000h
Transaction Pending - Set to 1 when Non-Posted Requests are not yet completed and clear when they are completed	21	R	0h
Aux Power Detected - Set to 1 if Aux power detected.	20	R	0
Unsupported Request Detected - Errors are logged in this register regardless of whether error reporting is enabled in the Device Control register	19	W1C	0
Fatal Error Detected - Errors are logged in this register regardless of whether error reporting is enabled in the Device Control register.	18	W1C	0
Non-Fatal Error detected - Errors are logged in this register regardless of whether error reporting is enabled in the Device Control register.	17	W1C	0
Correctable Error Detected - Errors are logged in this register regardless of whether error reporting is enabled in the Device Control register.	16	W1C	0
Reserved	15	R	0
Max Read Request Size	14:12	R/W	2h
Enable No Snoop	11	R/W	1
AUX Power PM Enable	10	R/W	0
Phantom Function Enable	9	R/W	1
Extended Tag Field Enable	8	R/W	0
Max Payload Size	7:5	R/W	0h
Enable Relaxed Ordering	4	R/W	1
Unsupported Request Reporting Enable	3	R/W	0
Fatal Error Reporting Enable	2	R/W	0
Non-Fatal Error Reporting Enable	1	R/W	0
Correctable Error Reporting Enable	0	R/W	0

Link Capabilities Register

Offset: Configuration Address 7Ch

Description	Bits	Type	Reset Value
Port Number	31:24		00h
Reserved	23:21		0h
Data Link Layer Active Reporting Capable	20	R	0
Surprise Down Error Reporting Capable - Not supported, hardwired to 0.	19	R	0
Clock Power Management	18	R	0
L1 Exit Latency - < 1 µs	17:15	R	0h
L0s Exit Latency - 256–512 ns	14:12	R	3h
Active State Link PM Support - L0s and L1 supported	11:10	R	3h
Maximum Link Width - ×4 port link	9:4	R	04h
Maximum Link Speed - 1h for 2.5 Gbps Link.	3:0	R	1h

Link Control/Status Register

Offset: Configuration Address 80h

Description	Bits	Type	Reset Value
Reserved	31:30	R	0h
***Data Link Layer Active* -** hardwired to 0	29	R	0
Slot Clock Configuration - Indicates that the component uses the same physical reference clock that the platform provides on the connector.	28	R	1
Link Training - hardwired to 0	27	R	0
Undefined	26	R	0
Negotiated Link Width - Set automatically by hardware after Link initialization.	25:20	R	X
Link Speed - the negotiated Link speed: 2.5 Gbps	19:16	R	1h
Reserved	15:9	R	00h
***Enable Clock Power Management* -** Hardwired to 0 if Clock Power Management is disabled in the Link Capabilities register.	8	R/W	0
Extended Synch	7	R/W	0
Common Clock Configuration	6	R/W	0
Retrain Link - Hardwired to 0.	5	R	0
Link Disable - Hardwired to 0.	4	R	0
Read Completion Boundary (RCB)	3	R/W	0
Reserved	2	R/W	0
Active State Link PM Control	1:0	R/W	0h

Slot Capabilities Register

Offset: Configuration Address 84h

Description	Bits	Type	Reset Value
Physical Slot Number	31:19	R	0000h
No Command Complete Support	18	R	0
Electromechanical Interlock Present	17	R	0
Slot Power Limit Scale	16:15	R	0h
Slot Power Limit Value	14:7	R	0h
Hot-Plug Capable	6	R	0
Hot-Plug Surprise	5	R	0
Power Indicator Present	4	R	0
Attention Indicator Present	3	R	0
MRL Sensor Present	2	R	0
Power Controller Present	1	R	0
Attention Button Present	0	R	0

Slot Control/Status Register

Offset: Configuration Address 80h

Description	Bits	Type	Reset Value
Reserved	31:25	R	00h
Data Link Layer State Changed - Hardwired to 0.	24	R	0
Electromechanical Interlock Status	23	R	0
Presence Detect State	22	R	0
MRL Sensor State	21	R	0
Command Completed	20	W1C	0
Presence Detect Changed	19	W1C	0
MRL Sensor Changed	18	W1C	0
Power Fault Detected	17	W1C	0
Attention Button Pressed	16	W1C	0
Reserved	15:13	R	0h
Data Link Layer State Changed Enable - Hardwired to 0.	12	R	0
Electromechanical Interlock Control	11	R/W	0
Power Controller Control	10	R/W	0
Power Indicator Control	9:8	R/W	0
Attention Indicator Control	7:6	R/W	0
Hot-Plug Interrupt Enable	5	R/W	0
Command Completed Interrupt Enable	4	R/W	0
Presence Detect Changed Enable	3	R/W	0
MRL Sensor Changed Enable	2	R/W	0
Power Fault Detected Enable	1	R/W	0
Attention Button Pressed Enable	0	R/W	0

PCI Express Enhanced Capability Header Register

Offset: Configuration Address 100h

Description	Bits	Type	Reset Value
Next Capability Offset	31:20	R	001h
Capability Version	19:16	R	1h
PCI Express Extended Capability ID - Advanced Error Reporting.	15:0	R	0001h

Advanced Error Reporting: Uncorrectable Error Status Register

Offset: Configuration Address 104h

Description	Bits	Type	Reset Value
Reserved	31:21	R	000h
UNSUPPORTED REQUEST ERROR STATUS	20	W1C	0
ECRC ERROR STATUS	19	W1C	0
MALFORMED TLP STATUS	18	W1C	0
RECEIVER OVERFLOW STATUS	17	W1C	0
UNEXPECTED COMPLETION STATUS	16	W1C	0
COMPLETER ABORT STATUS	15	W1C	0
COMPLETION TIMEOUT STATUS	14	W1C	0
FLOW CONTROL PROTOCOL ERROR STATUS	13	W1C	0
POISONED TLP STATUS	12	W1C	0
Reserved	11:6	R	00h
SURPRISE DOWN ERROR STATUS - Not supported.	5	R	0
DATA LINK PROTOCOL ERROR STATUS	4	W1C	0
Reserved	3:1	R	0h
Undefined	0	R	0

Advanced Error Reporting: Uncorrectable Error Mask Register

Offset: Configuration Address 108h

Description	Bits	Type	Reset Value
Reserved	31:21	R	000h
UNSUPPORTED REQUEST ERROR MASK	20	R/W	0
ECRC ERROR MASK	19	R/W	0
MALFORMED TLP MASK	18	R/W	0
RECEIVER OVERFLOW MASK	17	R/W	0
UNEXPECTED COMPLETION MASK	16	R/W	0
COMPLETER ABORT MASK	15	R/W	0
COMPLETION TIMEOUT MASK	14	R/W	0
FLOW CONTROL PROTOCOL ERROR MASK	13	R/W	0
POISONED TLP MASK	12	R/W	0
Reserved	11:6	R	00h
SURPRISE DOWN ERROR MASK - Not supported.	5	R	0
DATA LINK PROTOCOL ERROR MASK	4	R/W	0
Reserved	3:1	R	0h
Undefined	0	R	0

Advanced Error Reporting: Uncorrectable Error Severity Register

Offset: Configuration Address 10Ch

Description	Bits	Type	Reset Value
Reserved	31:21	R	000h
Unsupported Request Error Severity	20	R/W	0
ECRC Error Severity	19	R/W	0
Malformed TLP Severity	18	R/W	1
Receiver Overflow Severity	17	R/W	1
Unexpected Completion Severity	16	R/W	0
Completer Abort Severity	15	R/W	0
Completion Timeout Severity	14	R/W	0
Flow Control Protocol Error Severity	13	R/W	1
Poisoned TLP Severity	12	R/W	0
Reserved	11:6	R	00h
Surprise Down Error Severity - Not supported.	5	R	1
Data Link Protocol Error Severity	4	R/W	0
Reserved	3:1	R	0h
Undefined	0	R	0

Advanced Error Reporting: Correctable Error Status Register

Offset: Configuration Address 110h

Description	Bits	Type	Reset Value
Reserved	31:14	R	00000h
Advisory Non-Fatal Error Status	13	W1C	0
Reply Timer Timeout Status	12	W1C	0
Reserved	11:6	R	00h
REPLAY_NUM Rollover Status	8	W1C	0
Bad DLLP Status	7	W1C	0
Bad TLP Status	6	W1C	0
Reserved	5:1	R	0h
Receive Error Status	0	W1C	0

Advanced Error Reporting: Correctable Error Mask Register

Offset: Configuration Address 114h

Description	Bits	Type	Reset Value
Reserved	31:14	R	00000h
Advisory Non-Fatal Error Mask	13	R/W	1
Reply Timer Timeout Mask	12	R/W	0
Reserved	11:6	R	00h
REPLAY_NUM Rollover Mask	8	R/W	0
Bad DLLP Mask	7	R/W	0
Bad TLP Mask	6	R/W	0
Reserved	5:1	R	0h
Receive Error Mask	0	R/W	0

Advanced Error Reporting: Advanced Capabilities/Control Register

Offset: Configuration Address 118h

Description	Bits	Type	Reset Value
Reserved	31:9	R	000000h
ECRC Check Enable	8	R/W	0
ECRC Check Capable	7	R	1
ECRC Generation Enable	6	R/W	0
ECRC Generation Capability	5	R	1
First Error Pointer	4:0	R	00h

Advanced Error Reporting: Header Log Register (First DWORD)

Offset: Configuration Address 11Ch

Description	Bits	Type	Reset Value
First DWORD	31:0	R	00000000h

Advanced Error Reporting: Header Log Register (Second DWORD)

Offset: Configuration Address 120h

Description	Bits	Type	Reset Value
Second DWORD	31:0	R	00000000h

Advanced Error Reporting: Header Log Register (Third DWORD)

Offset: Configuration Address 124h

Description	Bits	Type	Reset Value
THIRD DWORD	31:0	R	00000000h

Advanced Error Reporting: Header Log Register (Fourth DWORD)

Offset: Configuration Address 128h

Description	Bits	Type	Reset Value
FOURTH DWORD	31:0	R	00000000h

Ack Latency Timer/Replay Timer Register

Offset: Configuration Address 700h

Description	Bits	Type	Reset Value
REPLAY TIME LIMIT - The replay timer expires when it reaches this limit. The CN16XX initiates a replay upon reception of a Nak or when the replay timer expires. The replay time limit is updated based on the Negotiated Link Width and Max_Payload_Size.	31:16	R/W	006Dh
ROUND TRIP LATENCY TIME LIMIT - The Ack/Nak latency timer expires when it reaches this limit. The round trip latency time limit is then updated based on the Negotiated Link Width and Max_Payload_Size.	15:0	R/W	0024h

Other Message Register

Offset: Configuration Address 704h

Description	Bits	Type	Reset Value
OTHER MESSAGE REGISTER - This register can be used for either of the following purposes: • To send a specific PCI Express message, the application writes the payload of the message into this register, then sets bit 0 of the Port Link Control Register to send the message. • To store a corruption pattern for corrupting the LCRC on all TLPs, the application places a 32-bit corruption pattern into this register and enables this function by setting bit 25 of the Port Link Control Register. When enabled, the transmit LCRC result is XOR'd with this pattern before inserting it into the packet.	31:0	R/W	00000000h

Port Force Link Register

Offset: Configuration Address 708h

Description	Bits	Type	Reset Value
LOW POWER ENTRANCE COUNT - The Power Management state waits for this many clock cycles for the associated completion of the CfgWr to D-state register to go low-power. This register is intended for applications that do not let the core handle a completion for configuration request to the PMCSCR register.	31:22		07h
Reserved	23:22		0h
LINK STATE - The Link state that the CN16XX is forced to when ***FORCE LINK*** is set. State encoding is undefined.	21:16	R/W	00h
FORCE LINK - Forces the Link to the state specified by ***LINK STATE***. The Force Link pulse triggers Link renegotiation. **NOTE:** As the The Force Link is a pulse, writing a 1 to it does trigger the forced link state event, even thought reading it always returns a 0.	15	R/W	0
Reserved	14:8		00h
LINK NUMBER - Not used.	7:0	R/W	4h

Ack Frequency Register

Offset: Configuration Address 70Ch

Description	Bits	Type	Reset Value
Reserved	31:30	R/W	0h
L1 Entrance Latency - Values correspond to: 000: 1 μs 100: 16 μs 001: 2 μs 101: 32 μs 010: 4 μs 110: 64 μs 011: 8 μs 111: 64 μs	29:27	R/W	0h
L0s Entrance Latency - Values correspond to: 000: 1 μs 100: 5 μs 001: 2 μs 101: 6 μs 010: 3 μs 110: 7 μs 011: 4 μs 111: 7 μs	26:24	R/W	0h
N_FTS WHEN COMMON CLOCK IS USED - The number of Fast Training Sequence ordered sets to be transmitted when transitioning from L0s to L0. The maximum number of FTS ordered-sets that a component can request is 255. **NOTE:** The core does not support a value of 0h; a value of 0h can cause the LTSSM to go into the recovery state when exiting from L0s.	23:16	R/W	80h
N_FTS - The number of Fast Training Sequence ordered sets to be transmitted when transitioning from L0s to L0. The maximum number of FTS ordered-sets that a component can request is 255. **NOTE:** The core does not support a value of 0h; a value of 0h can cause the LTSSM to go into the recovery state when exiting from L0s.	15:8	R/W	80h
ACK FREQUENCY - The EP Core accumulates the number of pending Acks specified here (up to 255) before sending an Ack.	7:0	R/W	00h

Port Link Control Register

Offset: Configuration Address 710h

Description	Bits	Type	Reset Value
Reserved	31:26	R	00h
Enable Corrupted CRC - Causes the CN16XX to corrupt the LCRC for TLPs when set, using the pattern contained in the Other Message register. **NOTE:** This is a test feature, not to be used in normal operation.	25	R/W	0
Reserved	24:22	R	0h
Link Mode Enable - 000001: ×1 000011: ×2 (not supported) 000111: ×4 001111: ×8 011111: ×16 111111: ×32 (not supported) 001111: ×8 011111: ×16 111111: ×32 (not supported)	21:16	R/W	07h
Reserved	15:12	R	0h
Reserved	11:8	R	1h
Fast Link Mode - Sets all internal timers to fast mode for simulation purposes.	7	R/W	0
Reserved	6	R	
DLL Link Enable - Enables Link initialization. If DLL Link Enable = 0, the CN16XX does not transmit InitFC DLLPs and does not establish a Link	5	R/W	1
Reserved	4	R	0
Must be Zero	3	R/W	0
Loopback Enable - Turns on loopback.	2	R/W	0
Scramble Disable - Turns off data scrambling.	1	R/W	0
Other Message Request - When software writes a 1 to this bit, the CN16XX transmits the message contained in the Other Message register.	0	R/W	0

Lane Skew Register

Offset: Configuration Address 714h

Description	Bits	Type	Reset Value
Disable Lane-to-Lane Deskew - Causes the CN16XX to disable the internal lane-to-lane deskew logic.	31	R/W	0
Reserved	30:26	R	00h
Ack/Nak Disable - Prevents the CN16XX from sending Ack and Nak DLLPs.	25	R/W	0
Flow Control Disable - Prevents the CN16XX from sending FC DLLPs.	24	R/W	0
Insert Lane Skew for Transmit (not supported for ×16) **-** Optional feature that causes the CN16XX to insert skew between Lanes for test purposes. There are three bits per Lane. The value is in units of one symbol time. For example, the value 010b for a Lane forces a skew of two symbol times for that Lane. The maximum skew value for any Lane is 5 symbol times.	23:0	R/W	000000h

Symbol Number Register

Offset: Configuration Address 718h

Description	Bits	Type	Reset Value
Reserved	31:29	R	1h
Timer Modifier for Flow Control Watchdog Timer - Increases the timer value for the Flow Control watchdog timer, in increments of 16 125-MHz clock cycles.	28:24	R/W	0h
Timer Modifier for Ack/Nak Latency Timer - Increases the timer value for the Ack/Nak latency timer, in increments of 64 125-MHz clock cycles.	23:19	R/W	0h
Timer Modifier for Replay Timer - Increases the timer value for the replay timer, in increments of 64 clock cycles.	18:14	R/W	04h
Reserved	13:11	R	0h
Number of SKP Symbols	10:8	R/W	3h
Reserved	7:4	R	Ah
Number of TS Symbols - Sets the number of TS identifier symbols that are sent in TS1 and TS2 ordered sets.	3:0	R/W	Ah

Symbol Timer Register and Filter Mask Register 1

Offset: Configuration Address 71Ch

Description	Bits	Type	Reset Value
Reserved	31:30	R/W	0h
Send decoded Message on the SII, then drop the Message TLPs 0: Drop MSG TLP (except for Vendor MSG) 1: Do not Drop MSG (except for Vendor MSG)	29	R/W	0
Mask ECRC error filtering for Completions 0: Discard TLPs with ECRC errors for CPL type 1: Allow TLPs with ECRC errors to be passed up for CPL type	28	R/W	0
Mask ECRC error filtering 0: Discard TLPs with ECRC errors 1: Allow TLPs with ECRC errors to be passed up	27	R/W	0
Mask Length mismatch error for received Completions 0: Enforce length match for rcvd CPL TLPs; infraction results in cpl_abort, and possibly AER of unexp_cpl_err 1: MASK length match for rcvd CPL TLPs	26	R/W	0
Mask Attributes mismatch error for received Completions 0: Enforce attribute match for rcvd CPL TLPs; infraction results in cpl_abort, and possibly AER of unexp_cpl_err,cpl_rcvd_ur,cpl_rcvd_ca 1: Mask attribute match for rcvd CPL TLPs	25	R/W	0
Mask Traffic Class mismatch error for received Completions 0: Enforce Traffic Class match for rcvd CPL TLPs; infraction results in cpl_abort, and possibly AER of unexp_cpl_err, cpl_rcvd_ur, cpl_rcvd_ca 1: Mask Traffic Class match for rcvd CPL TLPs	24	R/W	0
Mask Requester ID mismatch error for received Completions 0: Enforce function match for rcvd CPL TLPs; infraction results in cpl_abort, and possibly AER of unexp_cpl_err, cpl_rcvd_ur, cpl_rcvd_ca 1: Mask function match for rcvd CPL TLPs	23	R/W	0

Description	Bits	Type	Reset Value
MASK REQUESTER ID MISMATCH ERROR FOR RECEIVED COMPLETIONS 0: Enforce Req. Id match for rcvd CPL TLPs; infraction result in cpl_abort, and possibly AER of unexp_cpl_err,cpl_rcvd_ur,cpl_rcvd_ca 1: Mask Req. Id match for rcvd CPL TLPs	22	R/W	0
MASK TAG ERROR RULES FOR RECEIVED COMPLETIONS 0: Enforce Tag Error Rules for rcvd CPL TLPs; infraction result in cpl_abort, and possibly AER of unexp_cpl_err, cpl_rcvd_ur, cpl_rcvd_ca 1: Mask Tag Error Rules for rcvd CPL TLPs	21	R/W	0
MASK LOCKED REQUEST FILTERING 0: Treat locked Read TLPs as UR 1: Treat locked Read TLPs as Supported	20	R/W	0
MASK TYPE 1 CONFIGURATION REQUEST FILTERING 0: Treat CFG type1 TLPs as UR 1: Treat CFG type1 TLPs as Supported	19	R/W	0
MASK BAR MATCH FILTERING 0: Treat out-of-bar TLPs as UR 1: Treat out-of-bar TLPs as Supported Requests	18	R/W	0
MASK POISONED TLP FILTERING 0: Treat poisoned TLPs as UR 1: Treat poisined TLPs as Supported Requests	17	R/W	0
MASK FUNCTION MISMATCH FILTERING FOR INCOMING REQUESTS 0: Treat Function MisMatched TLPs as UR 1: Treat Function MisMatched TLPs as Supported	16	R/W	0
DISABLE FC WATCHDOG TIMER	15	R/W	0
Reserved	14:11		0h
SKP INTERVAL VALUE - The number of double-symbol times to wait between transmitting SKP ordered sets. Note that the CN16XX actually waits the number of double-symbol times in this register plus 2 between transmitting SKP ordered sets. For example, if 1536 we're programmed into this register, then the CN16XX will actually transmit Skp ordered sets once overy 3074 symbol times.	10:0	R/W	280h

Filter Mask Register 2

Offset: Configuration Address 720h

Description	Bits	Type	Reset Value
Reserved		R/W	00000000h
MASK VENDOR MSG TYPE 1 DROPPED SILENTLY - must be zero 0: Vendor MSG1 is silently dropped 1: Vendor MSG1 is passed to the TRGT1 interface	1	R/W	0
MASK VENDOR MSG TYPE 0 DROPPED WITH UR ERROR REPORTING - must be zero 0: Vendor MSG0 is dropped and treated as UR 1: Vendor MSG0 is passed to the TRGT1 interface	0	R/W	0

Debug Register 0

Offset: Configuration Address 728h

Description	Bits	Type	Reset Value
Undefined debug information.	31:0	R	00000000h

Debug Register 1

Offset: Configuration Address 72Ch

Description	Bits	Type	Reset Value
Undefined debug information.	31:0	R	00000000h

Transmit Posted FC Credit Status Register

Offset: Configuration Address 730h

Description	Bits	Type	Reset Value
Reserved	31:20	R	000h
TRANSMIT POSTED HEADER FC CREDITS - The Posted Header credits advertised by the receiver at the other end of the Link, updated with each UpdateFC DLLP.	19:12	R	00h
TRANSMIT POSTED DATA FC CREDITS - The Posted Data credits advertised by the receiver at the other end of the Link, updated with each UpdateFC DLLP.	11:0	R	000h

Transmit Non-Posted FC Credit Status Register

Offset: Configuration Address 734h

Description	Bits	Type	Reset Value
Reserved	31:20	R	000h
TRANSMIT NON-POSTED HEADER FC CREDITS - The Non-Posted Header credits advertised by the receiver at the other end of the Link, updated with each UpdateFC DLLP.	19:12	R	00h
TRANSMIT NON-POSTED DATA FC CREDITS - The Non-Posted Data credits advertised by the receiver at the other end of the Link, updated with each UpdateFC DLLP.	11:0	R	000h

Transmit Completion FC Credit Status Register

Offset: Configuration Address 738h

Description	Bits	Type	Reset Value
Reserved	31:20	R	000h
TRANSMIT COMPLETION HEADER FC CREDITS - The Completion Header credits advertised by the receiver at the other end of the Link, updated with each UpdateFC DLLP.	19:12	R	00h
TRANSMIT COMPLETION DATA FC CREDITS - The Completion Data credits advertised by the receiver at the other end of the Link, updated with each UpdateFC DLLP.	11:0	R	000h

Queue Status Register

Offset: Configuration Address 73Ch

Description	Bits	Type	Reset Value
Reserved	31:3	R	00000000h
RECEIVED QUEUE NOT EMPTY - Indicates there is data in one or more of the receive buffers.	2	R	0
TRANSMIT RETRY BUFFER NOT EMPTY - Indicates that there is data in the transmit retry buffer.	1	R	0
RECEIVED TLP FC CREDITS NOT RETURNED - Indicates that the EP Core has sent a TLP but has not yet received an UpdateFC DLLP indicating that the credits for that TLP have been restored by the receiver at the other end of the Link.	0	R	0

VC0 Posted Receive Queue Control Register

Offset: Configuration Address 748h

Description	Bits	Type	Reset Value
Unused	31	R	0
Unused	30	R	0
Reserved	29:24	R	00h
Unused	23:21	R	0h
Reserved	20		0
VC0 POSTED HEADER CREDITS - The number of initial Posted header credits for VC0.	19:12	R	00h
VC0 POSTED DATA CREDITS - The number of initial Posted data credits for VC0.	11:0	R	000h

VC0 Non-Posted Receive Queue Control Register

Offset: Configuration Address 74Ch

Description	Bits	Type	Reset Value
Reserved	31:24	R	00h
Unused	23:21	R	0h
Reserved	20	R	0
VC0 NON-POSTED HEADER CREDITS - The number of initial Non-Posted header credits for VC0.	19:12	R	00h
VC0 NON-POSTED DATA CREDITS - The number of initial Non-Posted data credits for VC0.	11:0	R	000h

VC0 Completion Receive Queue Control Register

Offset: Configuration Address 750h

Description	Bits	Type	Reset Value
Reserved	31:24	R	00h
Unused	23:21	R	0h
Reserved	20	R	0
VC0 COMPLETION HEADER CREDITS - The number of initial Completion header credits for VC0.	19:12	R	00h
VC0 COMPLETION DATA CREDITS - The number of initial Completion data credits for VC0.	11:0	R	000h

5.3 Two-Wire Serial Interface Controller Details

TWSI interface logic contains a complete embedded Two-Wire-Serial-Interface controller, including a set of configuration and status registers, and command/control interface.

The architecture of the TWSI interface consists of three blocks:

- TWSI controller logic and associated configuration and status registers
- TWSI-to-Host System Interface communications mailbox registers
- TWSI-to-GigaCipher Processor Core communications mailbox registers.

The TWSI controller logic manages and performs all communication with the TWSI bus interface signals themselves, compliant to the I2C protocol. TWSI controller logic consists of a finite state machine and a set of configuration/status registers (CSRs). The state machine executes TWSI commands, and CSRs store controller configuration and status information. These commands and registers are described later in this section.

TWSI to Host System Interface communications mailbox registers are a pair of registers used to pass TWSI commands and CSR information between the Host System Interface and the TWSI controller. These registers are the HOST-TWSI and TWSI-HOST registers. They are part of the Host System Interface general register map described earlier in this chapter. They are accessible by both the Host System Interface and the TWSI controller logic. Details of how these registers are used is described below.

TWSI-to-GigaCipher Processor Core communications mailbox registers are a pair of registers used to pass TWSI commands and CSR information between the GigaCipher Processor Cores and the TWSI controller. These are the GPC-TWSI and TWSI-GPC registers. These registers are not mapped to the Host System Interface, and are only accessible by the GigaCipher Processor Cores and the TWSI controller. They are also described below.

5.3.1 TWSI Commands and Operations

There are four basic operations that can be performed with the TWSI logic.

- Reads or writes of external target devices connected to the NITROX® PX CN16XX TWSI bus signal pins, by devices connected to NITROX® PX CN16XX on the Host System Interface, or by the NITROX® PX CN16XX GigaCipher Processor Cores.
- Reads or writes of the TWSI controller's internal registers by devices connected to NITROX® PX CN16XX on the Host System Interface, or by the NITROX® PX CN16XX GigaCipher Processor Cores.
- Reads or writes to external devices connected to NITROX® PX CN16XX on the Host System Interface or to the NITROX® PX CN16XX GigaCipher Processor Cores by external TWSI master devices.
- Writes to the NITROX® PX CN16XX UCODE LOAD register by an external device connected to NITROX® PX CN16XX on the TWSI bus, or by the GigaCipher Processor Core.

Reads/Writes of External Devices on the TWSI bus

The Host System Interface or GigaCipher Processor Core writes to external devices on the TWSI bus as follows:

1. Write the desired TWSI bus write command to the HOST-TWSI or GPC-TWSI registers' ***OPC***, ***IAD***, and ***DATA*** fields. The ***OPC*** field specifies the type of write operation, the ***IAD*** field identifies the external device to be written, and ***DATA*** is the information to be written.
2. Poll the ***VALID*** bit of the of the HOST-TWSI or GPC-TWSI register to ensure that the command was accepted sucessfully.
3. Poll the ***VALID*** and ***STATUS*** bits of the TWSI-HOST or TWSI-GPC register to verify that the command has completed successfully.

The Host System Interface or GigaCipher Processor Core reads from an external device on the TWSI bus as follows:

1. Write the desired TWSI bus read command to the HOST-TWSI or GPC-TWSI register's ***OPC*** and ***IAD*** fields respectively. The ***OPC*** field specifies the type of read operation, and ***IAD*** is identifies the TWSI device to be read.
2. Poll the ***VALID*** bit of the of the HOST-TWSI or GPC-TWSI register to ensure that the command was accepted sucessfully.
3. Poll the ***VALID***, ***STATUS***, and ***DATA*** fields of the TWSI-HOST or TWSI-HOST register to verify that the command completed successfully, and to obtain the associated read data.

Read/Write of TWSI Controller Internal Registers

The Host System Interface or GigaCipher Processor Core writes to most of the TWSI Controller's internal registers as follows:

1. Write the TWSI controller register write command (1100b) to the ***OPC*** field of the HOST-TWSI or GPC-TWSI register respectively, with the ***DATA*** field containing information to be written.
2. Poll the ***VALID*** bit of the of the HOST-TWSI or GPC-TWSI register to ensure that the command was accepted sucessfully.
3. Poll the ***VALID*** and ***STATUS*** bits of the TWSI-HOST or TWSI-GPC register to verify that the command has completed successfully.

The Host System Interface or GigaCipher Processor Core reads most TWSI Controller internal registers as follows:

1. Write the TWSI controller register read command (1101b) to the ***OPC*** field of the HOST-TWSI or GPC-TWSI register respectively.
2. Poll the ***VALID*** bit of the of the HOST-TWSI or GPC-TWSI register to ensure that the command was accepted sucessfully.
3. Poll the ***VALID***, ***STATUS***, and ***DATA*** fields of the TWSI-HOST or TWSI-HOST register to verify that the command completed successfully, and to obtain the associated read data.

TWSI Master Device Access to Host System Interface or GigaCipher Processor Cores

NITROX® PX CN16XX does not provide a direct read or write path from a master device on the TWSI bus to an external Host System Interface device or to the GigaCipher Processor Cores. TWSI masters can only write to or read from the Host System Interface or GigaCipher Processor Cores indirectly via their respective mailbox register pairs. Host System Interface devices or the GigaCipher Processor Cores are required to poll the TWSI-HOST or TWSI-GPC registers to complete a TWSI master read or write access.

These operations require the Host System Interface software and/or GigaCipher Processor Core microcode to proactively access the TWSI communications mailbox registers under program control. Creation of Host System Interface software to perform these operations is an implementation detail left to the user. If implemented, GigaCipher Processor Core microcode to perform these operations will be documented in the NITROX® PX CN16XX Instruction Set and Programmers Guide manuals included in the references section of this document.

Combined Mode Accesses

An external device on the TWSI bus *must always* access NITROX® PX CN16XX over the TWSI interface in what is defined as "Combined-Mode Access". In this mode, the first byte transferred from the device on TWSI bus is always a write of the Combined-Mode Command.

For writes to NITROX® PX CN16XX, the external device follows the Combined-Mode Command by writing the data, which is sent to the register addressed by the ***ID*** field in the Combined-Mode Command.

For reads, the external device follows the Combined-Mode Command by switching to read mode, to read the specified number of data bytes from the register addressed by the the ***ID*** field in the Combined-Mode Command. If an external TWSI device reads without first writing the Combined-Mode Command, the data from the previously selected register is transferred.

The format of the Combined-Mode Command sent on the TWSI bus by an external device is shown in Table 5–4.

Table 5–4 Combined-Mode Command Format

Description	Bits
VALID: Must always be written '1' for a valid command	7
RESERVED. Write as 0.	6:5
MICROCODE LOAD**:** Only valid for write operations with the ***ID***[1:0] field set to 01b (TWSI to GPC) and the PSE_Mode signal active. A '1' written to this bit indicates micro-code load command. Next four bytes transferred from the external device is treated as microcode load command and maps exactly like micro-code load register.	4
SIZE - Size of data access (read/write) requested: 0: 8-bit data access 1: 32-bit data access	3
TYPE **-** indicates the type of the data available 0: Result of the previous GPC command 1: Data from external device to GPC	2
ID[1:0] **-** Indicates the NITROX® PX CN16XX register that the external device is trying to access (read/write): 00 = GPC-TWSI register (only read access allowed) 01 = TWSI-GPC register (read/write allowed) 10 = HOST-TWSI register (only read access allowed) 11 = TWSI-HOST register (read/write allowed)	1:0

Writing the NITROX® PX CN16XX UCODE LOAD register from an external master device on the TWSI bus.

TBD

5.3.2 TWSI Mailbox Register Detailed Descriptions

The TWSI interface communicates with the Host System Interface through a pair of mailbox registers called the HOST-TWSI and TWSI-HOST registers. A similar set of mailbox registers is used for communication between the GigaCipher Processor Cores and the TWSI interface. These are called the GPC-TWSI and TWSI-GPC registers. This section provides a detailed description of both pairs of mailbox registers.

HOST-TWSI Register

This register is part of the register space defined in Section 5.2, “Detailed Register Descriptions”. It is addressed by the Host System Interface at BAR0 + 60h. The host system uses this register to perform any of the following operations:

- Transfer a byte to a TWSI device connected on the TWSI bus
- Read a byte from the TWSI device connected on the TWSI bus
- Perform a read/write operation to the TWSI Controller's internal registers.
- Set/Clear mode of operation (master/slave only).
- Get/Set TWSI Controller's clock period.

Whenever the host writes a command to this register to communicate with the TWSI device, the action taken by chip is based on the TWSI mode. If set to master-slave mode, the command written to the HOST-TWSI register is performed by the TWSI controller automatically. If set to slave-only mode, the command is held in the HOST-TWSI register until read by the TWSI master device present on the bus. In master-slave mode, if the command is to write data to a TWSI device, data is written to the device and the return code (failure/success) is available through TWSI-HOST register. In master-slave mode, if the command is to read data from the TWSI device, the read result is available through the TWSI-HOST register. All the bytes are transferred in big-endian format.

Below is a description of the fields within the HOST-TWSI register.

Description	Bits	Type
VALID - Begins and returns status of TWSI commands. When set to '1' by the host, the TWSI controller will begin the TWSI operation specified in the ***OPC*** field of this register. Before setting this bit, the host should verify that it is a ‘0’. The TWSI controller clears this bit to ‘0’ to indicate that a previously submitted command has completed, and thus it is ready to accept another command.	31	R/W
MODE **-** specifies the type of TWSI bus read or write command as master/slave or slave-only. The TWSI controller will compare the value of this bit with the current mode bit in the internal TWSI controller’s MODE register. The result of a miscompare is <TBD>. For TWSI bus reads or writes from the Host System Interface: 0: TWSI Controller uses slave-only mode for this command 1: TWSI Controller uses master-slave mode for this command For TWSI controller register reads or writes this bit is a don’t care.	30	W

Description	Bits	Type
OPC[3:0] **-** Operation/Command field - specifies the type of operation to be performed by the current TWSI command. 0000: Single address byte write using 7-bit addressing 0001: Single address byte read using 7-bit addressing 0100: Single address byte write using10-bit addressing 0101: Single address byte read using 10-bit addressing 0010: Combined format byte write using 7-bit addressing 0011: Combined format byte read using 7-bit addressing 0110: Combined format byte write using 10-bit addressing 0111: Combined format byte read using 10-bit addressing 1000: Write TWSI Master Clock register 1001: Read TWSI Master Clock register 1010: Write TWSI mode register 1011: Read TWSI mode register 1100: Write TWSI Controller's internal registers (at ***IAD*** offset) 1101: Read TWSI Controller's internal registers (at ***IAD*** offset) Combined format addressing is described earlier in this chapter	29:26	W
DID[9:0] **-** For TWSI bus reads or writes, specifies the ID (address) of the TWSI device to be addressed by the host. In 7-bit addressing mode, only the lower 7 bits are used (bits [9:8] are are don't cares). For reads or writes of internal TWSI controller registers this field is a don't care.	25:16	W
IAD[7:0] **-** For TWSI bus reads or writes, specifies the Internal Address of the external TWSI device addressed by the ***DID***[9:0] field of the current command. For accesses of TWSI controller internal registers, bits [2:0] specify the register addressed by this command.	15:8	W
DATA[7:0] **-** For TWSI bus writes this field contains the data to be written to the TWSI interface. For writes to internal TWSI controller registers this field contains the data to be written. For reads from the internal TWSI controller registers, returns resulting read data.	7:0	R/W

TWSI-HOST Register

The TWSI-HOST register is written by external TWSI master devices during TWSI-to-Host write operations, or by TWSI slave devices in response to Host-to-TWSI read operations. It is read by the Host System Interface, and is part of the register space defined in section 5.2, “Detailed Register Descriptions”. It is addressed by the Host System Interface at BAR0 + 68h.

This register is used to perform two functions:

- Transfer data bytes between a device on the TWSI bus and the Host System Interface.
- Transfer the results of a TWSI interface Read or Write command, issued to the TWSI controller by the Host System Interface via the HOST-TWSI register.

From the perspective of the Host System Interface, the TWSI-HOST register appears as follows:

Description	Bits	Type
VALID **-** The host polls this field to determine whether it has data to read from the TWSI bus. 1: Valid data for the host to read. When the host reads this register, this bit automatically clears to zero. 0: No data to read.	15	R
STATUS - Indicates the status of the previous command issued by the host through the HOST-TWSI register. Must always be written as ‘1’. 1: Command/read successful. If the ***VALID*** bit is set, the command result or read data will be present in the ***DATA*** field. 0: TWSI bus error; the ***DATA*** field of this register contains an 8-bit code indicating the current status of the internal TWSI controller.	14	R
RESERVED **-** Read as 0	13:11	R
TYPE - Indicates the type of the data available in the ***DATA*** field. 0: Result (read value or result code) of the previous read or write command from the Host to a TWSI device. 1: Data written to the Host from an external TWSI device	10	R
ID[1:0] **-** Indicates NITROX® PX CN16XX register that the external TWSI device is trying to access. For a valid access to TWSI-HOST register this value must be '11'.	9:8	R
DATA[7:0] - Data written by the TWSI device, to be read by the host. A Read/ Write operation by the host in master-slave mode returns the result in these bits (read value or result code).	7:0	R

GPC-TWSI Register

This register is not part of the Host System Interface register space, and is not accessible from the host system. It is accessible only by the GigaCipher Processor Cores and/or the TWSI interface bus. Use of this register is controlled by GigaCipher Processor Core microcode, and is therefore not programmable by the end user. A description of functionality is included below for informational purposes only.

The APC-TWSI register is used by Admin Processor Cores in a similar manner to use of the HOST-TWSI register by the Host system software. The Admin Processor Core uses this register to perform any of the following operations:

- Transfer a byte to a TWSI device connected on the TWSI bus
- Read a byte from the TWSI device connected on the TWSI bus
- Perform a read/write operation to the TWSI Controller's internal registers.
- Set/Clear mode of operation (master/slave only).
- Get/Set TWSI Controller's clock period.

Below is a description of the fields within the GPC-TWSI register.

Description	Bits	Type
VALID - Begins and returns status of TWSI commands. When set to '1' by the GPC, the TWSI controller will begin the TWSI operation specified in the ***OPC*** field of this register. Before setting this bit, the GPC should verify that it is a '0'. The TWSI controller clears this bit to '0' to indicate that a previously submitted command has completed, and thus it is ready to accept another command.	63	R/W
MODE **-** Specifies the type of TWSI bus read or write command as master/slave or slave-only. The TWSI controller will compare the value of this bit with the current mode bit in the internal TWSI controller's MODE register. The result of a mis-compare is TBD. For TWSI bus reads or writes from the GPC: 0: TWSI Controller uses slave-only mode for this command 1: TWSI Controller uses master-slave mode for this command For TWSI controller register reads or writes this bit is a don't care.	62	R/W
UCODE LOAD - When NITROX® PX CN16XX is in PSE mode (See PSE_Mode input signal description), set this bit to write to the UCODE LOAD register from the GPC. 1: writes data in the corresponding ***DATA***[31:0] field of this register to the UCODE LOAD register in the Host System Interface register set. When set, the TWSI bus controller does not execute a TWSI bus operation when this register is written. If PSE_Mode pin is low, this signal is ignored and normal GPC to TWSI command sequences will be performed. 0: Causes normal Admin Processor Core to TWSI command sequences will be performed.	61	W

Description	Bits	Type
OPC[4:0] **-** Operation/Command field - defines the operation to be performed by this command. 00000 Single address byte write using 7-bit addressing 00001 Single address byte read using 7-bit addressing 00010 Combined format byte write using 7-bit addressing 00011 Combined format byte read using 7-bit addressing 10000 Single address word write using 7-bit addressing 10001 Single address word read using 7-bit addressing 10010 Combined format word write using 7-bit addressing 10011 Combined format word read using 7-bit addressing 00100 Single address byte write using10-bit addressing 00101 Single address byte read using 10-bit addressing 00110 Combined format byte write using 10-bit addressing 00111 Combined format byte read using 10-bit addressing 10100 Single address word write using 10-bit addressing 10101 Single address word read using10-bit addressing 10110 Combined format word write using 10-bit addressing 10111 Combined format word read using 10-bit addressing 01000 Write TWSI Master Clock register 01001 Read TWSI Master Clock register 01010 Write TWSI mode register 01011 Read TWSI mode register 01100 Write TWSI Controller's internal registers (at ***IAD*** offset) 01101 Read TWSI Controller's internal registers (at ***IAD*** offset) Byte read/writes use Data[7:0], Word read/writes use Data[31:0]. Combined format addressing is described earlier in this chapter	60:56	W
RESERVED - Read as 0.	50:55	R
DID[9:0] **-** For TWSI bus reads or writes, specifies the ID (address) of the TWSI device to be addressed by the host. In 7-bit addressing mode, only the lower 7 bits are used (bits [9:8] are are don't cares). For reads or writes of internal TWSI controller registers this field is a don't care.	49:40	W
IAD[7:0] **-** For TWSI bus reads or writes, specifies the Internal Address of the external TWSI device addressed by the ***DID***[9:0] field of the current command. For accesses of TWSI controller internal registers, bits [2:0] specify the register addressed by this command.	39:32	W
DATA[7:0] **-** For TWSI bus writes this field contains the data to be written to the TWSI interface. For writes to internal TWSI controller registers this field contains the data to be written. For reads from the internal TWSI controller registers, returns resulting read data. Byte read/writes use ***DATA***[7:0], Word read/writes use ***DATA***[31:0].	31:0	W

TWSI-GPC Register

The TWSI-GPC register is written by external TWSI master devices during TWSI-to-Admin Processor Core write operations, or by TWSI slave devices in response to Admin Processor Core-to-TWSI read operations. It is read by the GPCs.

The TWSI-GPC register is used to perform two functions:

- Transfer data bytes between a device on the TWSI bus and the Admin Processor Cores.
- Transfer the results of a TWSI interface Read or Write command, issued to the TWSI controller by the Admin Processor Cores via the GPC-TWSI register.

From the perspective of the Admin Processor Cores, the TWSI-GPC register appears as follows:

Description	Bits
VALID - The GPC polls this field to determine whether it has valid data to read from the TWSI bus. 1: Valid read data available in Data[31:0]. When the GPC reads this register, this bit automatically clears to zero. 0: No read data available.	39
Reserved - read as 0	38:37
UCODE LOAD - When NITROX® PX CN16XX is in PSE mode (see PSE_Mode input signal description), set this bit to write to the UCODE LOAD register from the TWSI bus. 1: writes data in the corresponding ***DATA***[31:0] field of this register to the UCODE LOAD register in the Host System Interface register set. If PSE_Mode pin is low, this signal is ignored. 0: Normal TWSI bus operation	36
SIZE - Size of current data access (read/write): 0: 8-bit data access 1: 32-bit data access	35
TYPE - Indicates the type of the data available in the ***DATA*** field. 0: Result (read value or result code) of the previous read or write command from the GPC to a TWSI device. 1: Data written to the GPC from an external TWSI device	34
ID[1:0] **-** Indicates NITROX® PX CN16XX register that the external TWSI device is trying to access. For a valid access to TWSI-GPC register this value must be '01'.	33:32
DATA[31:0] - Data written by the TWSI device, to be read by the GPC. A Read/Write operation by the GPC in master-slave mode returns the result in these bits (read value or result code). For 8-bit operations, only the lower 8 bits are valid. For 32-bit operations or micro-code load operations all 32 bits are valid.	31:0

5.3.3 TWSI Controller Internal Register Summary

The TWSI controller is accessed and managed via a set of TWSI command/control/status registers (CSRs) in the TWSI controller logic. These registers are accessed indirectly by either the Host System Interface or Admin Processor Core via commands sent to their respective *-TWSI and TWSI-* mailbox register sets.

The TWSI controller's TWSI MASTER CLOCK and TWSI MODE registers are selected by writing the appropriate TWSI controller command to the ***OPC*** field of the *-TWSI mailbox register, as shown below in Table 5–5.

Table 5–5 Accessing the TWSI Master Clock and Mode Registers

TWSI Controller Register Name	Access	OPC Value
TWSI MASTER CLOCK	Write	1000b
	Read	1001b
TWSI MODE	Write	1010b
	Read	1011b

The remaining TWSI Controller internal registers are selected through the ***IAD*** field of the HOST-TWSI or GPC-TWSI mailbox registers, as shown in Table 5–6.

Table 5–6 TWSI Controller Internal Register Summary

IAD Field	Access	TWSI Controller Register Name
000b	R/W	TWSI SLAVE ADDRESS register
001b	R/W	TWSI DATA register
010b	R/W	TWSI CONTROL register
011b	W	TWSI CLOCK CONTROL register
100b	R/W	TWSI EXTENDED SLAVE ADDRESS register
101b	-	Reserved
110b	-	Reserved
111b	R/W	TWSI SOFT RESET register

5.3.4 TWSI Controller Internal Register Detailed Descriptions

This section provides detailed descriptions of each of the TWSI controller configuration and status registers.

TWSI Master Clock Register

Offset: ***OPC*** = 8h (write), 9h (read)

Reset Value: 9Ch

Bits	Description
7:0	***TCLK HALF PERIOD*** - (THP) controls the frequency ratio between the TWSI controller's clock (TCLK) and the NITROX® PX CN16XX core clock (CCLK). TCLK is defined as: *TCLK(MHz) = CCLK(MHz)/(2 × (THP+1))*

TWSI Mode Register

Offset: ***OPC*** = Ah (write), Bh (read)

Reset Value: 00h

Bits	Type	Description
7	R	**SCL** -copy of pse_scl
6	R	**SDA** -copy of pse_sda
5	R/W	**SCL_OVR** -pse_scl override (set high to drive open-drain pse_scl low)
4	R/W	**SDA_OVR** -pse_sda override (set high to drive open-drain pse_sda low)
3:2	R	***RESERVED***. Read as 0.
1		**PSE_MODE** - Read-only copy of pse_mode/
0	R/W	***MASTER*** - Configures TWSI controller logic for Master-Slave or Slave-Only mode. This determines how the TWSI controller reacts to writes to the HOST-TWSI and GPC-TWSI registers. 0: Slave-Only mode 1: Master-Slave mode

TWSI Slave Address Register

Offset: ***OPC*** = Ch (write), Dh (read)
IAD = xxxxx000b
Reset Value: 00h

Bits	Type	Description
7:1	R/W	***SLAVE ADDRESS***[6:0] - Configures the 7-bit slave address of NITROX® PX CN16XX on the TWSI bus. NITROX® PX CN16XX will respond as a target (slave), to TWSI bus initiator (master) accesses at the address written to this field. For 7-bit addressing, legal values for this field are 00-77h, and 7C-7Fh. ***EXTENDED SLAVE ADDRESS***[9:8] - When the address received on the TWSI bus begins with 11110b, the NITROX® PX CN16XX TWSI controller recognizes this as the preamble to a 10-bit address. To configure NITROX® PX CN16XX as a 10-bit addressed TWSI slave, set ***SLAVE ADDRESS***[6:2] to the preamble value (11110b). In this configuration, ***SLAVE ADDRESS***[1:0] should be set to the desired ***EXTENDED SLAVE ADDRESS***[9:8]. ***EXTENDED SLAVE ADDRESS***[7:0] are described below.
0	R/W	***GENERAL CALL ADDRESS ENABLE*** - Configures NITROX® PX CN16XX to respond as a target (slave) to TWSI bus initiator (master) accesses at the 7-bit General Call Address 1: Respond to General Call Address cycles 0: Ignore General Call Address cycles

TWSI Extended Slave Address Register

Offset: ***OPC*** = Ch (write), Dh (read)
IAD = xxxxx100b
Reset Value: 00h

Bits	Type	Description
7:0	R/W	***EXTENDED SLAVE ADDRESS***[7:0] - Configures the lower 8 bits of the NITROX® PX CN16XX slave address on the TWSI bus, when NITROX® PX CN16XX is configured as a 10-bit addressed TWSI slave. Refer to the TWSI Slave Address register description for details about configuring NITROX® PX CN16XX for 10-bit slave addressing.

TWSI Data Register

Offset: ***OPC*** = Ch (write), Dh (read)
IAD = xxxxx001b
Reset Value: 00h

Bits	Type	Description
7:0	R/W	***DATA***[7:0] - For manual TWSI bus writes (transmit mode), this register should be written with the data byte or slave address to be transmitted. For manual TWSI bus reads (receive mode) this register returns the data byte that has just been received from the TWSI bus. In transmit mode the byte is sent MSB first; in receive mode, the first bit received will be placed in the MSB of the register.

TWSI Control Register

Offset: ***OPC*** = Ch (write), Dh (read)
IAD = xxxxx010b

Reset Value: 00h

Bit	Description
7	***CONTROLLER ENABLE*** - Enables or disables the TWSI Controller. 1: The TWSI Controller is enabled. It will respond to commands presented to it by the Host System Interface, the Admin Processor Cores, or from the TWSI bus itself, acting as a TWSI slave. The controller automatically manages the sequence of TWSI bus commands to transfer or receive data. 0: The TWSI Controller is disabled. It will not respond to commands and will not automatically manage bus cycle sequences. The bus interface can still be used by the Host System Interface or Admin Processor Core in manual mode, by manually managing individual TWSI bus sequences using TWSI Control, Status and Data registers.
6	***BUS ENABLE*** - Controls the TWSI controller's response to inputs on the bus. 1: The controller will respond to bus operations at its slave address, or to the General-Call address when ***GENERAL-CALL ADDRESS ENABLE*** is set. 0: TWSI bus inputs are ignored and the controller will not respond as a TWSI slave at any address.
5:3	Reserved. Should be written as 0s.
2	***ASSERT ACKNOWLEDGE*** - Determines what NITROX® PX CN16XX will drive on the bus during the acknowledge clock cycle. 1: An "Acknowledge" will be sent on the TWSI bus during the acknowledge cycle of a slave access, if any of the following conditions are met: – A complete and matching 7-bit slave address has been received – The first or the second byte of a matching 10-bit slave address has been received – The General-Call address has been received and accepted (because it was enabled) – A data byte has been received in master or slave mode. 0: Behavior is state-dependent. If cleared to zero while the NITROX® PX CN16XX TWSI controller is at idle, NITROX® PX CN16XX will no longer respond to TWSI slave accesses. If cleared to '0' while the TWSI controller is currently receiving data in master or slave mode, a "Not Acknowledge" will be sent on the TWSI bus during the acknowledge cycle, when a data byte is received on the TWSI bus. If cleared to '0' while NITROX® PX CN16XX TWSI controller is currently transmitting in slave mode, the current byte in the ***DATA***[7:0] register is assumed to be the 'last byte'. After this byte has been transmitted, the TWSI controller will return to the idle state and the TWSI controller will no longer respond to TWSI slave accesses.
1:0	Reserved. Should be written as 0s

TWSI Clock Control Register

This register controls the frequency at which the TWSI bus is sampled in receiver mode, and the frequency at which the TWSI clock line will be driven in transmitter mode. The value of ***TCLK*** below is dependent upon the settings in the MASTER CLOCK register described above.

Offset: ***OPC*** = Ch (write only)
IAD = xxxxx011b

Reset Value: None

Bits	Description
7	***RESERVED***. Write as 0.
6:3	***T_M***[3:0] - see description below
2:0	***T_N***[2:0] - see decription below

The TWSI input clock sampling frequency (F_{SAMP}) is calculated as:

$$F_{SAMP} = TCLK \ / \ (2^{T\_N[2:0]})$$

The TWSI output clock frequency driven in master mode (F_{OSCL}), is calculated as:

$$F_{OSCL} = (F_{SAMP}) / [10 \times (T\_M[3:0] + 1)]$$

$$= TCLK / [(2^{T\_N[2:0]}) \times (10 \times (T\_M[3:0] + 1))]$$

The use of two separately programmable dividers allows the master-mode output frequency to be set independently of the frequency at which the TWSI bus is sampled. The sampling frequency must be at least 10 times the frequency of the fastest master on the bus to ensure that START and STOP conditions are always detected. By using two programmable clock-divider stages, a high sampling frequency can be ensured while allowing the master mode output to be set to a lower frequency. This is particularly useful in multi-master systems.

In most applications, setting this register to all 0s is sufficient. This ensures the 10x factor of input clock to sampling frequency.

TWSI Soft Reset Register

Offset: ***OPC*** = Ch (write), Dh (read)
IAD = xxxxx111b

Reset Value: 00h

Bits	Type	Description
7:0	R/W	***SOFT RESET***[7:0] - Resets the TWSI interface when written with any value. A software reset forces the TWSI controller to the idle state and clears the TWSI Control Register to the reset state. After chip reset or after software reset the TWSI controller is not available for any communication with the TWSI bus for three ***TCLK*** cycles (at the default value of the TWSI MASTER CLOCK register. This corresponds to 315 core clock (CCLK) cycles)

Chapter 6

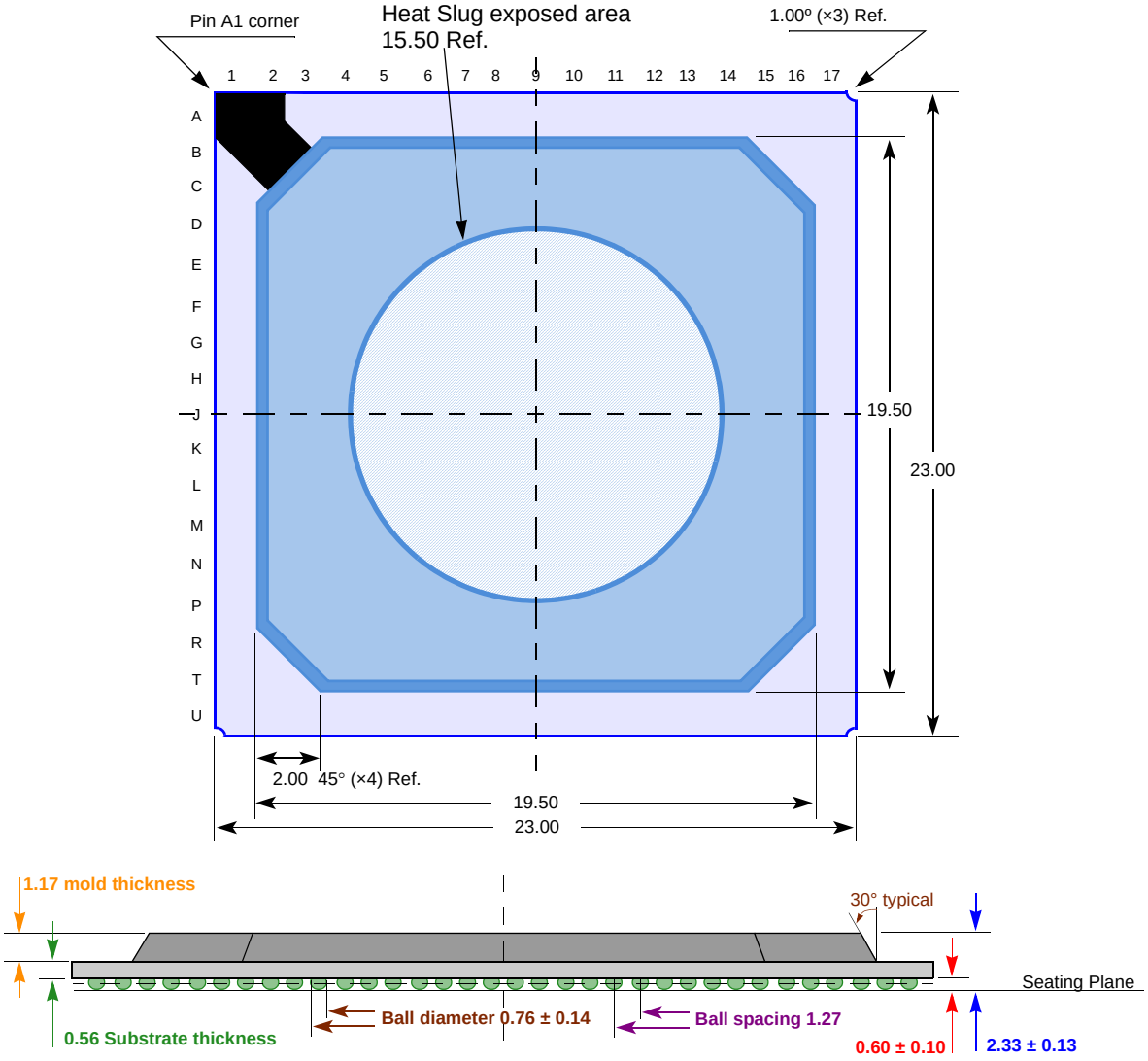
CN16XX Mechanical Interface Specifications

This section includes a detailed reference for all mechanical parameters of the device. This includes diagrams of the ball and pad grids, tables of signal assignments, and mechanical drawings of the packages.

6.1 233 HSBGA Package Diagram

Figure 6-1 shows the 233 HSBGA package diagram.

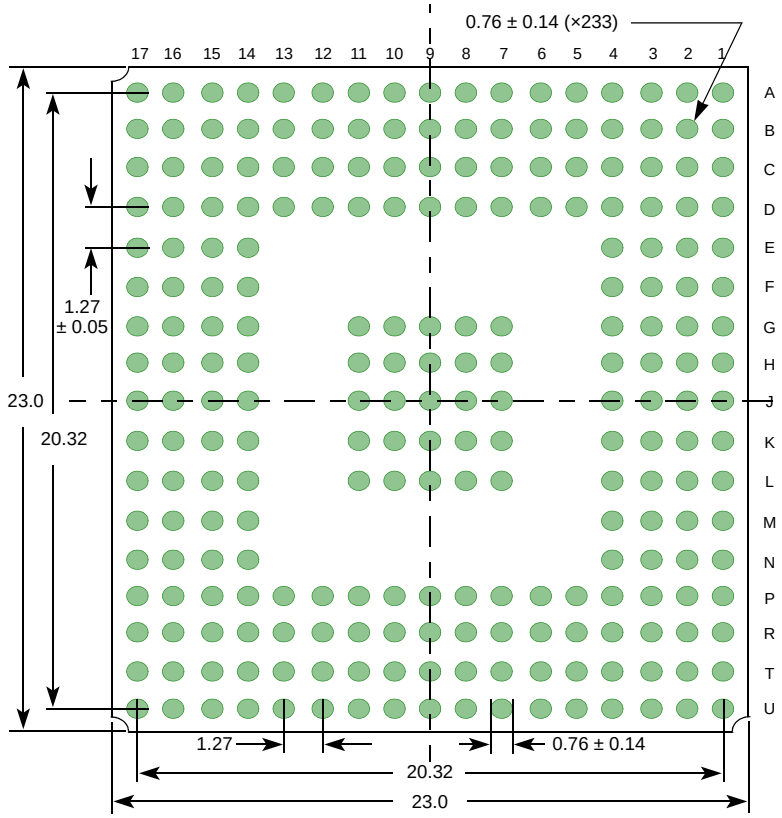
Figure 6-1 233 HSBGA Package Diagram



6.2 233 HSBGA Ball Grid Diagram - Ball (Bottom) View

Figure 6-1 shows the 233 HSBGA package diagram.

Figure 6-2 233-Pin HSBGA Pinout, Ball (bottom) View



6.1 CN16XX Signals Sorted by Signal Name

Signal Name	Ball
CLK_OUT	A3
CLKOUT_EN	A12
ENA_400	A5
EPR_CS	B10
EPR_DI	B11
EPR_DO	A11
EPR_SK	A10
EXP_BAND_RES	G16
EXP_COMP_RES	H16
EXP_GND	A17
EXP_GND	B16
EXP_GND	C15
EXP_GND	D15
EXP_GND	E14
EXP_GND	F15
EXP_GND	H15
EXP_GND	K15
EXP_GND	L14
EXP_GND	M15
EXP_GND	P15
EXP_GND	R15
EXP_GND	T16
EXP_GND	U17
EXP_GND	F17
EXP_GND	R16
EXP_GND	T17
EXP_GND	J16
EXP_REF_CLK_N	H17
EXP_REF_CLK_P	G17
EXP_REVERSE_LANES	A16
EXP_RX_0_N	C17
EXP_RX_0_P	B17
EXP_RX_1_N	D17
EXP_RX_1_P	E17
EXP_RX_2_N	M17
EXP_RX_2_P	L17
EXP_RX_3_N	N17
EXP_RX_3_P	P17
EXP_TX_0_N	D16
EXP_TX_0_P	C16
EXP_TX_1_N	E16
EXP_TX_1_P	F16

Signal Name	Ball
EXP_TX_2_N	L16
EXP_TX_2_P	K16
EXP_TX_3_N	M16
EXP_TX_3_P	N16
EXP_VDD	P16
EXP_VDD	R17
EXP_VDD	E15
EXP_VDD	F14
EXP_VDD	H14
EXP_VDD	K14
EXP_VDD	L15
EXP_VDD	M14
EXP_VDD	N15
EXP_VDD	G15
EXP_VDD	J15
EXP_VDDBAND	J17
EXP_VDDPLL	K17
GND	A1
GND	A2
GND	A4
GND	A8
GND	A13
GND	B1
GND	B2
GND	C3
GND	C5
GND	C7
GND	C11
GND	C13
GND	D4
GND	D6
GND	D8
GND	D10
GND	D12
GND	D14
GND	E3
GND	F1
GND	F4
GND	G3
GND	G7
GND	G8
GND	G9

Signal Name	Ball
GND	G10
GND	G11
GND	G14
GND	H4
GND	H7
GND	H8
GND	H9
GND	H10
GND	H11
GND	J7
GND	J8
GND	J9
GND	J10
GND	J11
GND	J14
GND	K4
GND	K7
GND	K8
GND	K9
GND	K10
GND	K11
GND	L1
GND	L3
GND	L7
GND	L8
GND	L9
GND	L10
GND	L11
GND	M4
GND	N3
GND	P4
GND	P6
GND	P8
GND	P10
GND	P12
GND	P14
GND	R1
GND	R3
GND	R5
GND	R7
GND	R11
GND	R13

Signal Name	Ball
GND	T2
GND	T8
GND	U1
GND	U5
GND	U10
GND	U14
GND	U16
JTAG_TCK	B7
JTAG_TDI	B5
JTAG_TDO	B4
JTAG_TMS	B6
JTAG_TRST_L	B8
NC	T4
NC	T15
NC	H1
NC	H2
PERST_L	M1
PLL_BYPASS	C1
PLL_DCOK	B3
PLL_REF_CLK	E2
PLL_VDD_3V	K1
PSE_MODE	F2
PSE_SCL	G2
PSE_SDA	G1
PSE_ZERO_KEYS	D1
RSVD	A6
RSVD	A7
RSVD	A14
RSVD	A15
RSVD	B12
RSVD	B13
RSVD	B14
RSVD	B15
RSVD	C2
RSVD	D2
RSVD	E1
RSVD	K2
RSVD	L2
RSVD	M2
RSVD	N1
RSVD	N2
RSVD	P1
RSVD	P2

Signal Name	Ball
RSVD	R2
RSVD	T1
RSVD	T3
RSVD	T5
RSVD	T6
RSVD	T7
RSVD	T10
RSVD	T11
RSVD	T12
RSVD	T13
RSVD	U2
RSVD	U3
RSVD	U4
RSVD	U6
RSVD	U7
RSVD	U8
RSVD	U11
RSVD	U12
RSVD	U13
RSVD	U15
TEST_PD	T14
VDD_1V	A9
VDD_1V	B9
VDD_1V	C4
VDD_1V	C6
VDD_1V	C8
VDD_1V	C9
VDD_1V	C10
VDD_1V	C12
VDD_1V	C14
VDD_1V	D3
VDD_1V	F3
VDD_1V	H3
VDD_1V	J1
VDD_1V	J2
VDD_1V	J3
VDD_1V	K3
VDD_1V	M3
VDD_1V	N14
VDD_1V	P3
VDD_1V	R4
VDD_1V	R6
VDD_1V	R8

Signal Name	Ball
VDD_1V	R9
VDD_1V	R10
VDD_1V	R12
VDD_1V	R14
VDD_1V	T9
VDD_1V	U9
VDD_3V	D5
VDD_3V	D7
VDD_3V	D9
VDD_3V	D11
VDD_3V	D13
VDD_3V	E4
VDD_3V	G4
VDD_3V	J4
VDD_3V	L4
VDD_3V	N4
VDD_3V	P5
VDD_3V	P7
VDD_3V	P9
VDD_3V	P11
VDD_3V	P13

6.2 CN16XX Signals Sorted by Associated Ball Number

Ball	Signal Name
A1	GND
A2	GND
A3	CLK_OUT
A4	GND
A5	ENA_400
A6	RSVD
A7	RSVD
A8	GND
A9	VDD_1V
A10	EPR_SK
A11	EPR_DO
A12	CLKOUT_EN
A13	GND
A14	RSVD
A15	RSVD
A16	EXP_REVERSE_LANES
A17	EXP_GND
B1	GND
B2	GND
B3	PLL_DCOK
B4	JTAG_TDO
B5	JTAG_TDI
B6	JTAG_TMS
B7	JTAG_TCK
B8	JTAG_TRST_L
B9	VDD_1V
B10	EPR_CS
B11	EPR_DI
B12	RSVD
B13	RSVD
B14	RSVD
B15	RSVD
B16	EXP_GND
B17	EXP_RX_0_P
C1	PLL_BYPASS
C2	RSVD
C3	GND
C4	VDD_1V
C5	GND
C6	VDD_1V
C7	GND
C8	VDD_1V

Ball	Signal Name
C9	VDD_1V
C10	VDD_1V
C11	GND
C12	VDD_1V
C13	GND
C14	VDD_1V
C15	EXP_GND
C16	EXP_TX_0_P
C17	EXP_RX_0_N
D1	PSE_ZERO_KEYS
D2	RSVD
D3	VDD_1V
D4	GND
D5	VDD_3V
D6	GND
D7	VDD_3V
D8	GND
D9	VDD_3V
D10	GND
D11	VDD_3V
D12	GND
D13	VDD_3V
D14	GND
D15	EXP_GND
D16	EXP_TX_0_N
D17	EXP_RX_1_N
E1	RSVD
E2	PLL_REF_CLK
E3	GND
E4	VDD_3V
E14	EXP_GND
E15	EXP_VDD
E16	EXP_TX_1_N
E17	EXP_RX_1_P
F1	GND
F2	PSE_MODE
F3	VDD_1V
F4	GND
F14	EXP_VDD
F15	EXP_GND
F16	EXP_TX_1_P
F17	EXP_GND

Ball	Signal Name
G1	PSE_SDA
G2	PSE_SCL
G3	GND
G4	VDD_3V
G7	GND
G8	GND
G9	GND
G10	GND
G11	GND
G14	GND
G15	EXP_VDD
G16	EXP_BAND_RES
G17	EXP_REF_CLK_P
H1	NC
H2	NC
H3	VDD_1V
H4	GND
H7	GND
H8	GND
H9	GND
H10	GND
H11	GND
H14	EXP_VDD
H15	EXP_GND
H16	EXP_COMP_RES
H17	EXP_REF_CLK_N
J1	VDD_1V
J2	VDD_1V
J3	VDD_1V
J4	VDD_3V
J7	GND
J8	GND
J9	GND
J10	GND
J11	GND
J14	GND
J15	EXP_VDD
J16	EXP_GND
J17	EXP_VDDBAND
K1	PLL_VDD_3V
K2	RSVD
K3	VDD_1V

Ball	Signal Name
K4	GND
K7	GND
K8	GND
K9	GND
K10	GND
K11	GND
K14	EXP_VDD
K15	EXP_GND
K16	EXP_TX_2_P
K17	EXP_VDDPLL
L1	GND
L2	RSVD
L3	GND
L4	VDD_3V
L7	GND
L8	GND
L9	GND
L10	GND
L11	GND
L14	EXP_GND
L15	EXP_VDD
L16	EXP_TX_2_N
L17	EXP_RX_2_P
M1	PERST_L
M2	RSVD
M3	VDD_1V
M4	GND
M14	EXP_VDD
M15	EXP_GND
M16	EXP_TX_3_N
M17	EXP_RX_2_N
N1	RSVD
N2	RSVD
N3	GND
N4	VDD_3V
N14	VDD_1V
N15	EXP_VDD
N16	EXP_TX_3_P
N17	EXP_RX_3_N
P1	RSVD
P2	RSVD
P3	VDD_1V
P4	GND

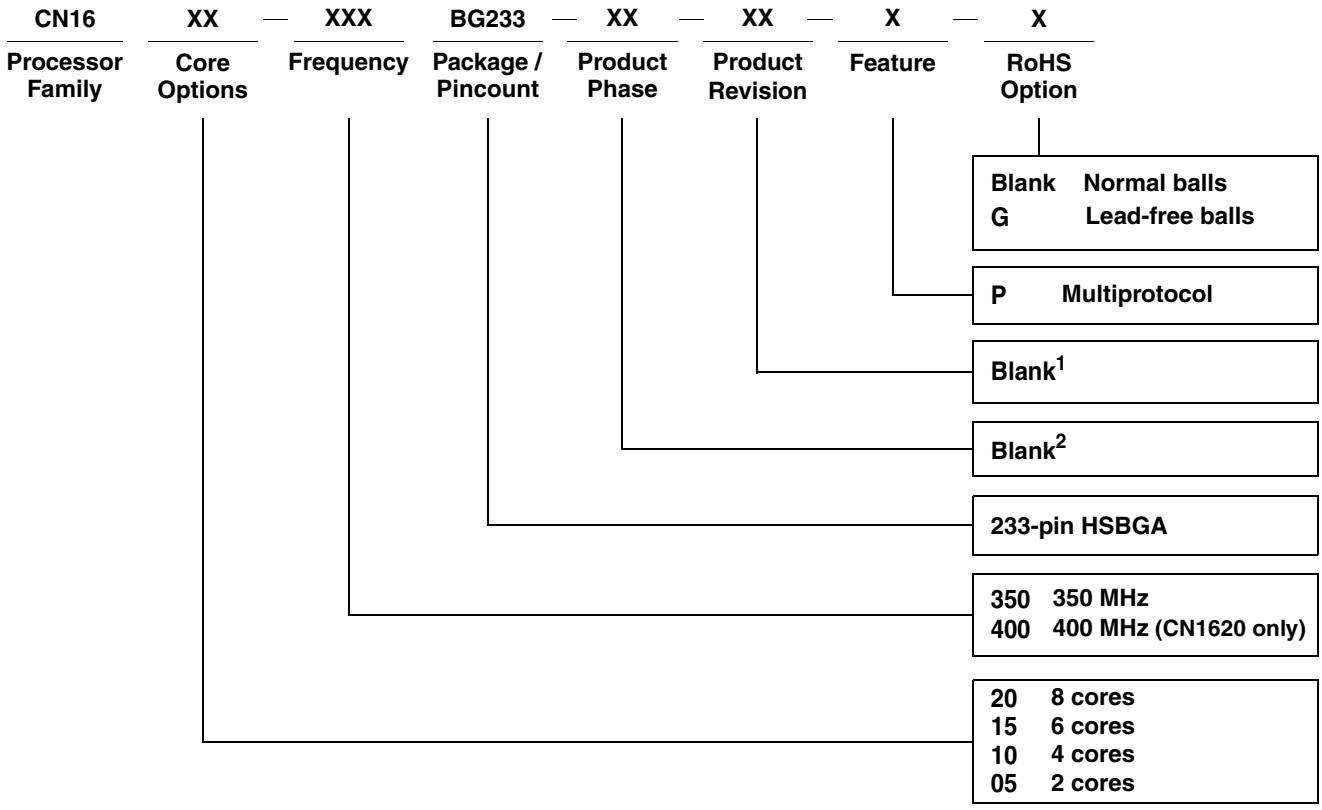
Ball	Signal Name
P5	VDD_3V
P6	GND
P7	VDD_3V
P8	GND
P9	VDD_3V
P10	GND
P11	VDD_3V
P12	GND
P13	VDD_3V
P14	GND
P15	EXP_GND
P16	EXP_VDD
P17	EXP_RX_3_P
R1	GND
R2	RSVD
R3	GND
R4	VDD_1V
R5	GND
R6	VDD_1V
R7	GND
R8	VDD_1V
R9	VDD_1V
R10	VDD_1V
R11	GND
R12	VDD_1V
R13	GND
R14	VDD_1V
R15	EXP_GND
R16	EXP_GND
R17	EXP_VDD
T1	RSVD
T2	GND
T3	RSVD
T4	NC
T5	RSVD
T6	RSVD
T7	RSVD
T8	GND
T9	VDD_1V
T10	RSVD
T11	RSVD
T12	RSVD
T13	RSVD

Ball	Signal Name
T14	TEST_PD
T15	NC
T16	EXP_GND
T17	EXP_GND
U1	GND
U2	RSVD
U3	RSVD
U4	RSVD
U5	GND
U6	RSVD
U7	RSVD
U8	RSVD
U9	VDD_1V
U10	GND
U11	RSVD
U12	RSVD
U13	RSVD
U14	GND
U15	RSVD
U16	GND
U17	EXP_GND

Appendix A

Ordering Information

The following shows the breakdown of the CN16XX family part numbers.



[1] Blank No revision suffix is required.

[2] Blank No phase suffix is required.

Example Part Numbers:

CN1615-350BG233-P-G 6 cores, 350MHz, multiprotocol, lead-free balls

CN1620-400BG233 8 cores, 400MHz, normal balls