# Non-Volatile Memory
# Host Controller Interface
# (NVMHCI) 1.0

**NVMHCI 1.0**

**April 14, 2008**

*Please send comments to Amber Huffman*
*amber.huffman@intel.com*

Non-Volatile Memory Host Controller Interface revision 1.0 specification available for download at http://www.intel.com/standards/nvmhci/index.htm. Ratified on April 14, 2008.

SPECIFICATION DISCLAIMER

THIS SPECIFICATION IS PROVIDED TO YOU "AS IS" WITH NO WARRANTIES WHATSOEVER, INCLUDING ANY WARRANTY OF MERCHANTABILITY, NON-INFRINGEMENT, OR FITNESS FOR ANY PARTICULAR PURPOSE. THE AUTHORS OF THIS SPECIFICATION DISCLAIM ALL LIABILITY, INCLUDING LIABILITY FOR INFRINGEMENT OF ANY PROPRIETARY RIGHTS, RELATING TO USE OR IMPLEMENTATION OF INFORMATION IN THIS SPECIFICATION. THE AUTHORS DO NOT WARRANT OR REPRESENT THAT SUCH USE WILL NOT INFRINGE ANY SUCH RIGHTS. THE PROVISION OF THIS SPECIFICATION TO YOU DOES NOT PROVIDE YOU WITH ANY LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS.

NVMHCI Workgroup Chair:

Amber Huffman
Intel Corporation
MS: JF2-53
2111 NE 25th Avenue
Hillsboro, OR 97124
amber.huffman@intel.com

## Table of Contents

1 INTRODUCTION ........ 6
1.1 Overview ........ 6
1.2 Scope ........ 6
1.3 Outside of Scope ........ 6
1.4 Conventions ........ 6
1.5 Definitions ........ 7
1.5.1 allocation unit ........ 7
1.5.2 command completion ........ 7
1.5.3 metadata ........ 7
1.5.4 NVM ........ 7
1.5.5 NVM device ........ 7
1.5.6 NVM page ........ 7
1.5.7 NVM page aligned ........ 7
1.5.8 NVM subsystem ........ 7
1.5.9 port ........ 7
1.5.10 sector ........ 7
1.6 Keywords ........ 7
1.6.1 mandatory ........ 7
1.6.2 may ........ 7
1.6.3 optional ........ 8
1.6.4 R ........ 8
1.6.5 reserved ........ 8
1.6.6 shall ........ 8
1.6.7 should ........ 8
1.7 Conventions ........ 8
1.8 Byte, word and Dword Relationships ........ 9
1.9 References ........ 9
2 PCI REGISTERS ........ 11
2.1 PCI Header ........ 11
2.1.1 Offset 00h: ID - Identifiers ........ 11
2.1.2 Offset 04h: CMD - Command ........ 12
2.1.3 Offset 06h: STS - Device Status ........ 12
2.1.4 Offset 08h: RID - Revision ID ........ 12
2.1.5 Offset 09h: CC - Class Code ........ 13
2.1.6 Offset 0Ch: CLS – Cache Line Size ........ 13
2.1.7 Offset 0Dh: MLT – Master Latency Timer ........ 13
2.1.8 Offset 0Eh: HTYPE – Header Type ........ 13
2.1.9 Offset 0Fh: BIST – Built In Self Test (Optional) ........ 13
2.1.10 Offset 10h: MLBAR – Memory Register Base Address, lower 32-bits ........ 13
2.1.11 Offset 14h: MUBAR – Memory Register Base Address, upper 32-bits ........ 13
2.1.12 Offset 18h: IDBAR – Index/Data Pair Register Base Address ........ 14
2.1.13 Offset 1Ch – 23h: BARS – Other Base Addresses (Reserved) ........ 14
2.1.14 Offset 24h – 27h: BAR5 – Vendor Specific ........ 14
2.1.15 Offset 28h: CCPTR – CardBus CIS Pointer ........ 14
2.1.16 Offset 2Ch: SS - Sub System Identifiers ........ 14
2.1.17 Offset 30h: EROM – Expansion ROM (Optional) ........ 14
2.1.18 Offset 34h: CAP – Capabilities Pointer ........ 14
2.1.19 Offset 3Ch: INTR - Interrupt Information ........ 14
2.1.20 Offset 3Eh: MGNT – Minimum Grant ........ 14
2.1.21 Offset 3Fh: MLAT – Maximum Latency ........ 15
2.2 PCI Power Management Capabilities ........ 15
2.2.1 Offset PMCAP: PID - PCI Power Management Capability ID ........ 15
2.2.2 Offset PMCAP + 2h: PC – PCI Power Management Capabilities ........ 16
2.2.3 Offset PMCAP + 4h: PMCS – PCI Power Management Control And Status ........ 16
2.3 Message Signaled Interrupt Capability (Optional) ........ 16
2.3.1 Offset MSICAP: MID – Message Signaled Interrupt Identifiers ........ 16

2.3.2 Offset MSICAP + 2h: MC – Message Signaled Interrupt Message Control ..... 17
2.3.3 Offset MSICAP + 4h: MA – Message Signaled Interrupt Message Address ..... 17
2.3.4 Offset MSICAP + 8h: MUA – Message Signaled Interrupt Upper Address (Optional) ..... 17
2.3.5 Offset MSICAP + Ch: MD – Message Signaled Interrupt Message Data ..... 17
2.4 Other Capability Pointers ..... 17

3 CONTROLLER REGISTERS ..... 18

3.1 Generic Host Control ..... 18
3.1.1 Offset 00h: CAP – Controller Capabilities ..... 19
3.1.2 Offset 08h: IS – Interrupt Status Register ..... 19
3.1.3 Offset 0Ch: PI – Ports Implemented ..... 19
3.1.4 Offset 10h: VS – NVMHCI Version ..... 19
3.2 Generic Host Control under AHCI ..... 19
3.3 Port Registers (one set per port) ..... 20
3.3.1 Offset 00h: PxCLB – Port x Command List Base Address ..... 21
3.3.2 Offset 04h: PxCLBU – Port x Command List Base Address Upper 32-bits ..... 21
3.3.3 Offset 10h: PxIS – Port x Interrupt Status ..... 21
3.3.4 Offset 14h: PxIE – Port x Interrupt Enable ..... 21
3.3.5 Offset 18h: PxCMD – Port x Command and Status ..... 22
3.3.6 Offset 24h: PxSIG – Port x Signature ..... 22
3.3.7 Offset 38h: PxCI – Port x Command Issue ..... 22
3.3.8 Offset 70h to 7Fh: PxVS – Vendor Specific ..... 23
3.4 Index/Data Pair registers ..... 23
3.4.1 Restrictions ..... 23
3.4.2 Register Definition ..... 23
3.4.3 Offset 00h: IDX – Index Register ..... 23
3.4.4 Offset 04h: DAT – Data Register ..... 23

4 SYSTEM MEMORY STRUCTURES ..... 24

4.1 Controller Memory Space Usage ..... 24
4.2 Port Memory Usage ..... 24
4.2.1 Command List Structure ..... 24
4.2.2 Command Table ..... 28

5 MEMORY ORGANIZATION AND COMMAND SET ..... 31

5.1 Memory Organization ..... 31
5.2 Command Status ..... 31
5.3 Command Definitions ..... 32
5.3.1 Dataset Management ..... 32
5.3.2 Flush ..... 36
5.3.3 Get Features ..... 37
5.3.4 Get Status ..... 38
5.3.5 Identify ..... 43
5.3.6 Read ..... 47
5.3.7 Set Features ..... 49
5.3.8 Write ..... 53

6 DATA TRANSFER OPERATION ..... 57

6.1 Introduction ..... 57
6.2 System Software Rules (Normative) ..... 57
6.2.1 Basic Steps when Building a Command ..... 57
6.2.2 Processing Completed Commands ..... 57
6.2.3 Data Transfer ..... 58
6.2.4 Software Examples (with PRD index fill out) ..... 58

7 ERROR REPORTING AND RECOVERY ..... 62

7.1 Error Types ..... 62
7.1.1 System Memory Errors ..... 62

7.1.2 Fatal NVM Device Errors....62
7.1.3 Status Errors ....62
7.2 Error Recovery....62
7.2.1 Host Software Error Recovery....62

8 INFORMATIVE APPENDIX ....64

8.1 Option ROM and EFI Information....64
8.1.1 EFI GUID....64
8.1.2 Version Information ....64
8.1.3 Option ROM Discovery....64
8.1.4 EFI Module Discovery ....65

# 1 Introduction

## 1.1 Overview

This specification defines the host software interface for the Non-Volatile Memory Host Controller Interface (NVMHCI). NVMHCI is a register level interface that allows host software to communicate with a platform non-volatile memory subsystem.

NVMHCI may either be a stand-alone PCI class device or it may be a port within an Advanced Host Controller Interface (AHCI) device. The specification draws heavily upon the AHCI specification.

## 1.2 Scope

NVMHCI defines a register interface for communication with a non-volatile memory subsystem. It also defines a standard command set for use with the NVM device.

## 1.3 Outside of Scope

NVMHCI is specified apart from any usage model for the NVM, but rather only specifies the communication interface to the NVM subsystem. Thus, NVMHCI does not specify whether the non-volatile memory system is used as a main memory, a cache memory, a backup memory, a redundant memory, etc. Specific usage models are outside the scope, optional, and not licensed.

NVMHCI is also specified above any non-volatile memory management, like wear leveling. Erases and other management tasks for NVM technologies like NAND are abstracted.

NVMHCI does not contain any information on caching algorithms or techniques. How the non-volatile memory is used for system level benefit is beyond the scope of this specification.

The implementation or use of other published specifications referred to in this specification, even if required for compliance with the specification, are outside the scope of this specification (for example, PCI, PCI Express and PCI-X).

## 1.4 Conventions

Hardware shall return ‘0’ for all bits and registers that are marked as reserved, and host software shall write all reserved bits and registers with the value of ‘0’.

Inside the register section, the following abbreviations are used:

| | |
|---:|---|
| **RO** | Read Only |
| **RW** | Read Write |
| **R/W** | Read Write. The value read may not be the last value written. |
| **RWC** | Read/Write ‘1’ to clear |
| **RWS** | Read/Write ‘1’ to set |
| **Impl Spec** | Implementation Specific – the controller has the freedom to choose its implementation. |
| **HwInit** | The default state is dependent on device and system configuration. The value is initialized at reset, either by an expansion ROM, or in the case of integrated devices, by a platform BIOS. |

When a register bit is referred to in the document, the convention used is “Register Symbol.Field Symbol”. For example, the configuration space PCI command register parity error response bit is referred to by the name CMD.PEE. If the register field is an array of bits, the field will be referred to as “Register Symbol.Field Symbol(array offset)”.

When a memory field is referred to in the document, the convention used is “Register Name[Offset Symbol]”.

## 1.5 Definitions

### 1.5.1 allocation unit

The smallest number of NVM pages that should be allocated by host software in file or cluster allocations.

### 1.5.2 command completion

A command is completed when the NVMHCI controller has completed processing a command and has placed valid status in the Command Status field of the command header (CH[z].CS). This status may be success or failure.

### 1.5.3 metadata

Metadata is contextual information about a particular NVM page of data. Host software may store any contextual information desired in the metadata, if provided, by the NVM subsystem.

### 1.5.4 NVM

NVM is an acronym for non-volatile memory.

### 1.5.5 NVM device

The NVM device is the non-volatile memory itself. The NVM may be on a module or soldered directly to a system board.

### 1.5.6 NVM page

The recommended minimum write granularity. This unit is an integral number of sectors.

### 1.5.7 NVM page aligned

An NVM page aligned access starts at the beginning of an NVM page (i.e. at the starting sector of an NVM page). The access size is in an NVM page granularity, meaning that the number of sectors in the access is a multiple of the NVM page size reported in the Identify command, see section 5.3.5.1.6.

### 1.5.8 NVM subsystem

The NVM subsystem includes the non-volatile memory module. This includes the memory controller, a non-volatile memory storage medium, an interface between the memory controller and memory storage medium.

### 1.5.9 port

A port is an entity that may independently execute NVMHCI commands with an NVM device. Each port has a set of registers and DMA engine with associated context.

### 1.5.10 sector

The smallest addressable data unit for Read and Write commands.

## 1.6 Keywords

Several keywords are used to differentiate between different levels of requirements.

### 1.6.1 mandatory

A keyword indicating items to be implemented as defined by this specification.

### 1.6.2 may

A keyword that indicates flexibility of choice with no implied preference.

### 1.6.3 optional

A keyword that describes features that are not required by this specification. However, if any optional feature defined by the specification is implemented, the feature shall be implemented in the way defined by the specification.

### 1.6.4 R

“R” is used as an abbreviation for “reserved” when the figure or table does not provide sufficient space for the full word “reserved”.

### 1.6.5 reserved

A keyword indicating reserved bits, bytes, words, fields, and opcode values that are set-aside for future standardization. Their use and interpretation may be specified by future extensions to this or other specifications. A reserved bit, byte, word, or field shall be cleared to zero, or in accordance with a future extension to this specification. The recipient shall not check reserved bits, bytes, words, or fields.

### 1.6.6 shall

A keyword indicating a mandatory requirement. Designers are required to implement all such mandatory requirements to ensure interoperability with other products that conform to the specification.

### 1.6.7 should

A keyword indicating flexibility of choice with a strongly preferred alternative. Equivalent to the phrase “it is recommended”.

## 1.7 Conventions

Kilobyte (KB) refers to 2^10 bytes, megabyte (MB) refers to 2^20 bytes, and gigabyte (GB) refers to 2^30 bytes.

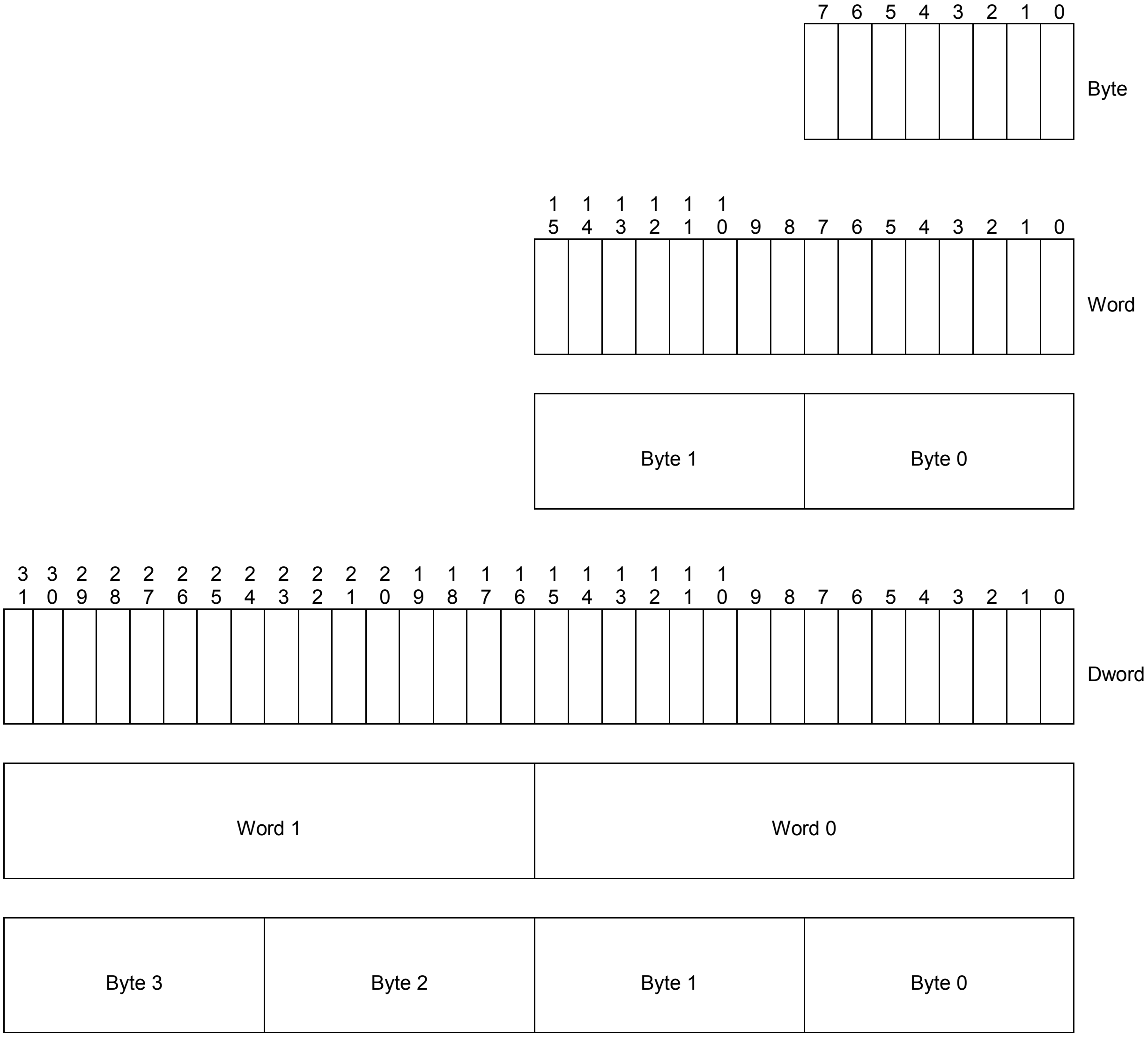
A 0-based value is a numbering scheme for which the number 0h actually corresponds to a value of 1h and thus produces the pattern of 0h = 1h, 1h = 2h, 2h = 3h, etc. In this numbering scheme, there is not a method for specifying the value of 0h.

Some parameters are defined as a string of ASCII characters. ASCII data fields shall contain only code values 20h through 7Eh. For the string “Copyright”, the character “C” is the first byte, the character “o” is the second byte, etc. The string shall be padded with spaces (ASCII character 20h) if necessary.

## 1.8 Byte, word and Dword Relationships

Figure 1 illustrates the relationship between bytes, words and Dwords. NVMHCI specifies data in a little endian format.

**Figure 1: Byte, word and Dword relationships**



## 1.9 References

Advanced Host Controller Interface specification, revision 1.3. Available from http://developer.intel.com/technology/serialata/ahci.htm.

PCI specification, revision 3.0. Available from http://www.pcisig.com.

PCI Express specification, revision 2.0. Available from http://www.pcisig.com.

PCI Power Management specification.  Available from http://www.pcisig.com.

# 2 PCI Registers

The registers specified in this section are only present if the NVMHCI device is a stand-alone PCI device. If the NVMHCI device is a port within an AHCI controller, then the PCI registers specified in the AHCI specification take precedence and are the PCI registers implemented.

This section describes the PCI register values when PCI is the system bus used. Other system buses may be used in an NVMHCI implementation, like PCI Express or PCI-X.

This section details how the PCI Header and PCI Capabilities should be constructed for an NVMHCI device. The fields shown are duplicated from the appropriate PCI specifications. The PCI documents are the normative specifications for these registers and this section details additional requirements for an NVMHCI device.

| Start | End | Name |
|---|---|---|
| 00h | 3Fh | PCI Header |
| PMCAP | PMCAP+7h | PCI Power Management Capability |
| MSICAP | MSICAP+9h | Message Signaled Interrupt Capability |

## 2.1 PCI Header

| Start | End | Symbol | Name |
|---|---|---|---|
| 00h | 03h | ID | Identifiers |
| 04h | 05h | CMD | Command Register |
| 06h | 07h | STS | Device Status |
| 08h | 08h | RID | Revision ID |
| 09h | 0Bh | CC | Class Codes |
| 0Ch | 0Ch | CLS | Cache Line Size |
| 0Dh | 0Dh | MLT | Master Latency Timer |
| 0Eh | 0Eh | HTYPE | Header Type |
| 0Fh | 0Fh | BIST | Built In Self Test (Optional) |
| 10h | 13h | MLBAR | Memory Register Base Address, lower 32-bits <BAR0> |
| 14h | 17h | MUBAR | Memory Register Base Address, upper 32-bits <BAR1> |
| 18h | 1Bh | IDBAR | Index/Data Pair Register Base Address <BAR2> |
| 1Ch | 23h | BARS | Other Base Address Registers (Reserved) <BAR3-4> |
| 24h | 27h | BAR5 | Vendor Specific |
| 28h | 2Bh | CCPTR | CardBus CIS Pointer |
| 2Ch | 2Fh | SS | Subsystem Identifiers |
| 30h | 33h | EROM | Expansion ROM Base Address (Optional) |
| 34h | 34h | CAP | Capabilities Pointer |
| 35h | 3Bh | R | Reserved |
| 3Ch | 3Dh | INTR | Interrupt Information |
| 3Eh | 3Eh | MGNT | Minimum Grant (Optional) |
| 3Fh | 3Fh | MLAT | Maximum Latency (Optional) |

### 2.1.1 Offset 00h: ID - Identifiers

| Bits | Type | Reset | Description |
|---|---|---|---|
| 31:16 | RO | Impl Spec | **Device ID (DID):** Indicates what device number assigned by the vendor. Specific to each implementation. |
| 15:00 | RO | Impl Spec | **Vendor ID (VID):** 16-bit field which indicates the company vendor, assigned by the PCI SIG. |

### 2.1.2 Offset 04h: CMD - Command

| Bit | Type | Reset | Description |
|---|---|---|---|
| 15:11 | RO | 0 | Reserved |
| 10 | RW | 0 | **Interrupt Disable (ID):** Disables the controller from generating pin-based INTx# interrupts. This bit does not have any effect on MSI operation. |
| 09 | RO | 0 | **Fast Back-to-Back Enable (FBE):** When set to '1', the controller is allowed to generate fast back-to-back cycles to different devices. Not supported by NVMHCI. |
| 08 | RO | 0 | **SERR# Enable (SEE):** When set to '1', the controller is allowed to generate SERR# on any event that is enabled for SERR# generation. Not supported by NVMHCI. |
| 07 | RO | 0 | Hardwired to 0. |
| 06 | RW / RO | 0 | **Parity Error Response Enable (PEE):** When set to '1', the controller shall generate PERR# when a data parity error is detected. If parity is not supported, then this field is read-only '0'. |
| 05 | RO | 0 | **VGA Palette Snooping Enable (VGA):** Controls how VGA compatible and graphics devices handle accesses to VGA palette registers. Not supported by NVMHCI. |
| 04 | RO | 0 | **Memory Write and Invalidate Enable (MWIE):** When set to '1', the controller is allowed to use the memory write and invalidate command. Not supported by NVMHCI. |
| 03 | RO | 0 | **Special Cycle Enable (SCE):** Controls a device's action on Special Cycle operations. Not supported by NVMHCI. |
| 02 | RW | 0 | **Bus Master Enable (BME):** Enables the controller to act as a master for data transfers. When set to '1', bus master activity is allowed. When cleared to '0', the controller stops any active DMA engines and returns to an idle condition. |
| 01 | RW | 0 | **Memory Space Enable (MSE)**: Controls access to the controller's register memory space. |
| 00 | RW | 0 | **I/O Space Enable (IOSE):** Controls access to the controller's target I/O space. |

### 2.1.3 Offset 06h: STS - Device Status

| Bit | Type | Reset | Description |
|---|---|---|---|
| 15 | RWC | 0 | **Detected Parity Error (DPE):** Set to '1' by hardware when the controller detects a parity error on its interface. |
| 14 | RO | 0 | **Signaled System Error (SSE):** Set to '1' by hardware when the controller host generates SERR#. Not supported by NVMHCI |
| 13 | RWC | 0 | **Received Master-Abort (RMA):** Set to '1' by hardware when the controller receives a master abort to a cycle it generated. |
| 12 | RWC | 0 | **Received Target Abort (RTA):** Set to '1' by hardware when the controller receives a target abort to a cycle it generated. |
| 11 | RO | 0 | **Signaled Target-Abort (STA):** Set to '1' by hardware when the controller terminates with a target abort. Not supported by NVMHCI. |
| 10:09 | RO | Impl Spec | **DEVSEL# Timing (DEVT):** Controls the device select time for the controller's PCI interface. |
| 08 | RWC | 0 | **Master Data Pariy Error Detected (DPD):** Set to '1' by hardware when the controller, as a master, either detects a parity error or sees the parity error line asserted, and the parity error response bit (bit 6 of the command register) is set to '1'. |
| 07 | RO | Impl Spec | **Fast Back-to-Back Capable (FBC):** Indicates whether the controller accepts fast back-to-back cycles. |
| 06 | RO | 0 | Reserved |
| 05 | RO | Impl Spec | **66 MHz Capable (C66):** Indicates whether the controller may operate at 66 MHz. |
| 04 | RO | 1 | **Capabilities List (CL):** Indicates the presence of a capabilities list. The controller shall support the PCI power management capability as a minimum. |
| 03 | RO | 0 | **Interrupt Status (IS):** Indicates the interrupt status of the device (1 = asserted). |
| 02:00 | RO | 0 | Reserved |

### 2.1.4 Offset 08h: RID - Revision ID

| Bits | Type | Reset | Description |
|---|---|---|---|
| 07:00 | RO | Impl Spec | **Revision ID (RID):** Indicates stepping of the controller hardware. |

### 2.1.5 Offset 09h: CC - Class Code

| Bits | Type | Reset | Description |
|---|---|---|---|
| 23:16 | RO | 01h | **Base Class Code (BCC):** Indicates the base class code as a mass storage controller. |
| 15:08 | RO | 08h | **Sub Class Code (SCC):** Indicates the sub class code as a Non-Volatile Memory controller. |
| 07:00 | RO | 01h | **Programming Interface (PI):** This field specifies the programming interface of the controller is NVMHCI 1.0. |

### 2.1.6 Offset 0Ch: CLS – Cache Line Size

| Bits | Type | Reset | Description |
|---|---|---|---|
| 07:00 | RO | 00h | **Cache Line Size (CLS):** Not supported by NVMHCI. |

### 2.1.7 Offset 0Dh: MLT – Master Latency Timer

| Bits | Type | Reset | Description |
|---|---|---|---|
| 07:00 | RO | 00h | **Master Latency Timer (MLT):** Indicates the number of clocks the controller is allowed to act as a master on PCI. Not supported by NVMHCI. |

### 2.1.8 Offset 0Eh: HTYPE – Header Type

| Bits | Type | Reset | Description |
|---|---|---|---|
| 07 | RO | Impl Spec | **Multi-Function Device (MFD):** Indicates whether the controller is part of a multi-function device. |
| 06:00 | RO | 00h | **Header Layout (HL):** Indicates that the controller uses a target device layout. |

### 2.1.9 Offset 0Fh: BIST – Built In Self Test (Optional)

The following register is optional, but if implemented, shall look as follows. When not implemented, it shall be read-only 00h.

| Bits | Type | Reset | Description |
|---|---|---|---|
| 07 | RO | Impl Spec | **BIST Capable (BC):** Indicates whether the controller has a BIST function. |
| 06 | RW | 0 | **Start BIST (SB):** Software sets this bit to ‘1’ to invoke BIST. The controller clears this bit to ‘0’ when BIST is complete. |
| 05:04 | RO | 00 | Reserved |
| 03:00 | RO | 0h | **Completion Code (CC):** Indicates the completion code status of BIST. A non-zero value indicates a failure. |

### 2.1.10 Offset 10h: MLBAR – Memory Register Base Address, lower 32-bits

This register allocates space for the memory registers defined in section 3.

| Bit | Type | Reset | Description |
|---|---|---|---|
| 31:13 | RW | 0 | **Base Address (BA):** Base address of register memory space. This represents a memory space for support of up to 32 ports. For controllers that support fewer than 32 ports, more bits are allowed to be RW, and therefore less memory space is consumed. For controllers that have vendor specific space at the end of the port specific memory space, more bits are allowed to be RO such that more memory space is consumed. |
| 12:04 | RO | 0 | Reserved |
| 03 | RO | 0 | **Prefetchable (PF):** Indicates that this range is not pre-fetchable |
| 02:01 | RO | 10 | **Type (TP):** Indicates that this range may be mapped anywhere in 64-bit address space and that the register is 64-bits wide. |
| 00 | RO | 0 | **Resource Type Indicator (RTE):** Indicates a request for register memory space. |

### 2.1.11 Offset 14h: MUBAR – Memory Register Base Address, upper 32-bits

This register specifies the upper 32-bit address of the memory registers defined in section 3.

| Bit | Type | Reset | Description |
|---|---|---|---|
| 31:00 | RW | 0 | **Base Address (BA):** Upper 32-bits (bits 63:32) of the memory register base address. |

**NOTE**: NVMHCI implementations that reside behind PCI compliant bridges, such as PCI Express endpoints, are restricted to having 32-bit assigned base address registers due to limitations on the maximum address that may be specified in the bridge for non-prefetchable memory. See the PCI Bridge 1.2 specification for more information on this restriction.

#### 2.1.12 Offset 18h: IDBAR – Index/Data Pair Register Base Address

This register specifies the Index/Data Pair base address. These registers are used to access the memory registers defined in section 3 using I/O based accesses.

| Bit | Type | Reset | Description |
|---|---|---|---|
| 31:03 | RW | 0 | **Base Address (BA):** Base address of Index/Data Pair registers that is 8 bytes in size. |
| 02:01 | RO | 0 | Reserved |
| 00 | RO | 1 | **Resource Type Indicator (RTE):** Indicates a request for register I/O space. |

#### 2.1.13 Offset 1Ch – 23h: BARS – Other Base Addresses (Reserved)

These registers allocate memory or I/O spaces for other BARs. These are reserved for future use.

#### 2.1.14 Offset 24h – 27h: BAR5 – Vendor Specific

The BAR5 register is vendor specific.

#### 2.1.15 Offset 28h: CCPTR – CardBus CIS Pointer

| Bit | Type | Reset | Description |
|---|---|---|---|
| 31:00 | RO | 0 | Points to the Card Information Structure (CIS) for the CardBus card. Not supported by NVMHCI. |

#### 2.1.16 Offset 2Ch: SS - Sub System Identifiers

| Bits | Type | Reset | Description |
|---|---|---|---|
| 31:16 | RO | HwInit | **Subsystem ID (SSID):** Indicates the sub-system identifier. |
| 15:00 | RO | HwInit | **Subsystem Vendor ID (SSVID):** Indicates the sub-system vendor identifier |

#### 2.1.17 Offset 30h: EROM – Expansion ROM (Optional)

The following register is optional. If the register is not implemented, it shall be read-only 00h.

| Bit | Type | Reset | Description |
|---|---|---|---|
| 31:00 | RW | Impl Spec | **ROM Base Address (RBA):** Indicates the base address of the controller's expansion ROM. Not supported for integrated implementations. |

#### 2.1.18 Offset 34h: CAP – Capabilities Pointer

| Bit | Type | Reset | Description |
|---|---|---|---|
| 7:0 | RO | PMCAP | **Capability Pointer (CP):** Indicates the first capability pointer offset. It points to the PCI Power management capability offset. |

#### 2.1.19 Offset 3Ch: INTR - Interrupt Information

| Bits | Type | Reset | Description |
|---|---|---|---|
| 15:08 | RO | Impl Spec | **Interrupt Pin (IPIN):** This indicates the interrupt pin the controller uses. |
| 07:00 | RW | 00h | **Interrupt Line (ILINE):** Software written value to indicate which interrupt line (vector) the interrupt is connected to. No hardware action is taken on this register. |

#### 2.1.20 Offset 3Eh: MGNT – Minimum Grant

| Bits | Type | Reset | Description |
|---|---|---|---|
| 07:00 | RO | 00h | **Grant (GNT):** Indicates the minimum grant time (in 1/4 microseconds) that the device wishes grant asserted. Not supported by NVMHCI. |

### 2.1.21 Offset 3Fh: MLAT – Maximum Latency

| Bits | Type | Reset | Description |
|---|---|---|---|
| 07:00 | RO | 00h | **Latency (LAT): I**ndicates the maximum latency (in 1/4 microseconds) that the device tolerates. Not supported by NVMHCI. |

## 2.2 PCI Power Management Capabilities

See section 5.3.2 for requirements when the PCI power management state changes.

| Start (hex) | End (hex) | Symbol | Name |
|---|---|---|---|
| PMCAP | PMCAP+1 | PID | PCI Power Management Capability ID |
| PMCAP+2 | PMCAP+3 | PC | PCI Power Management Capabilities |
| PMCAP+4 | PMCAP+7 | PMCS | PCI Power Management Control and Status |

### 2.2.1 Offset PMCAP: PID - PCI Power Management Capability ID

| Bit | Type | Reset | Description |
|---|---|---|---|
| 15:08 | RO | Impl Spec | **Next Capability (NEXT):** Indicates the location of the next capability item in the list. This may be other capability pointers (such as Message Signaled Interrupts) or it may be the last item in the list. |
| 07:00 | RO | 01h | **Cap ID (CID):** Indicates that this pointer is a PCI power management capability. |

### 2.2.2 Offset PMCAP + 2h: PC – PCI Power Management Capabilities

| Bit | Type | Reset | Description |
|---|---|---|---|
| 15:11 | RO | 0h | **PME_Support (PSUP):** Indicates the states that may generate PME#. Not supported by NVMHCI. |
| 10 | RO | 0 | **D2_Support (D2S):** Indicates support for the D2 power management state. Not supported by NVMHCI. |
| 09 | RO | 0 | **D1_Support (D1S):** Indicates support for the D1 power management state. Not supported by NVMHCI. |
| 08:06 | RO | 000 | **Aux_Current (AUXC):** This field reports the 3.3V auxiliary current requirements for the PCI function. Not supported by NVMHCI. |
| 05 | RO | Impl Spec | **Device Specific Initialization (DSI):** Indicates whether device-specific initialization is required. |
| 04 | RO | 0 | Reserved |
| 03 | RO | 0 | **PME Clock (PMEC):** Indicates that PCI clock is not required to generate PME#. |
| 02:00 | RO | Impl Spec | **Version (VS):** Indicates support for Revision 1.2 or higher revisions of the *PCI Power Management Specification.* |

### 2.2.3 Offset PMCAP + 4h: PMCS – PCI Power Management Control And Status

| Bit | Type | Reset | Description |
|---|---|---|---|
| 15 | RO | 0 | **PME Status (PMES):** Set to ‘1’ by hardware when the controller generates PME#. Not supported by NVMHCI. |
| 14:13 | RO | 0 | **Data Scale (DSC):** Indicates the scaling factor to be used when interpreting the value of the Data register. Not supported by NVMHCI. |
| 12:09 | RO | 0 | **Data Select (DSE):** Used to select which data is to be reported through the Data reigster and Data Scale field. Not supported by NVMHCI. |
| 08 | RO | 0 | **PME Enable (PMEE):** When set to ‘1’, the controller asserts the PME# signal when PMES is set to ‘1’. Not supported by NVMHCI. |
| 07:04 | RO | 0 | Reserved |
| 03 | RO | 1 | **No Soft Reset (NSFRST):** A ‘1’ indicates that the controller transitioning from D3hot to D0 because of a power state command does not perform an internal reset. |
| 02 | RO | 0 | Reserved |
| 01:00 | R/W | 00 | **Power State (PS):** This field is used both to determine the current power state of the controller and to set a new power state. The values are:<br>00 – D0 state<br>11 – D3$_{HOT}$ state<br>The D1 and D2 states are not supported for NVMHCI controllers. When in the D3$_{HOT}$ state, the controller’s configuration space is available, but the register I/O and memory spaces are not. Additionally, interrupts are blocked. |

## 2.3 Message Signaled Interrupt Capability (Optional)

| Start (hex) | End (hex) | Symbol | Name |
|---|---|---|---|
| MSICAP | MSICAP+1 | MID | Message Signaled Interrupt Capability ID |
| MSICAP+2 | MSICAP+3 | MC | Message Signaled Interrupt Message Control |
| MSICAP+4 | MSICAP+7 | MA | Message Signaled Interrupt Message Address |
| MSICAP+8 | MSICAP+B | MUA | Message Signaled Interrupt Upper Address (Optional) |
| MSICAP+C | MSICAP.D | MD | Message Signaled Interrupt Message Data |

### 2.3.1 Offset MSICAP: MID – Message Signaled Interrupt Identifiers

| Bits | Type | Reset | Description |
|---|---|---|---|
| 15:08 | RO | Impl Spec | **Next Pointer (NEXT):** Indicates the next item in the list. This may be other capability pointers or it may be the last item in the list. |
| 07:00 | RO | 05h | **Capability ID (CID):** Capabilities ID indicates this is a Message Signaled Interrupt (MSI) capability. |

### 2.3.2 Offset MSICAP + 2h: MC – Message Signaled Interrupt Message Control

| Bits | Type | Reset | Description |
|---|---|---|---|
| 15:09 | RO | 0 | Reserved |
| 08 | RO | 0 | **Per-Vector Masking Capable (PVM):** Specifies whether controller supports MSI per-vector masking. Not supported by NVMHCI. |
| 07 | RO | 1 | **64 Bit Address Capable (C64):** Specifies whether capable of generating 64-bit messages. NVMHCI controllers shall be 64-bit capable. |
| 06:04 | RW | 000 | **Multiple Message Enable (MME):** Indicates the number of messages the controller should assert. Controllers that only support single message MSI may implement this field as read-only. |
| 03:01 | RO | Impl Spec | **Multiple Message Capable (MMC):** Indicates the number of messages the controller wishes to assert. |
| 00 | RW | 0 | **MSI Enable (MSIE):** If set to '1', MSI is enabled and the traditional interrupt pins are not used to generate interrupts. If cleared to '0', MSI operation is disabled and the traditional interrupt pins are used. |

### 2.3.3 Offset MSICAP + 4h: MA – Message Signaled Interrupt Message Address

| Bits | Type | Reset | Description |
|---|---|---|---|
| 31:02 | RW | 0 | **Address (ADDR):** Lower 32 bits of the system specified message address, always Dword aligned. |
| 01:00 | RO | 00 | Reserved |

### 2.3.4 Offset MSICAP + 8h: MUA – Message Signaled Interrupt Upper Address (Optional)

| Bits | Type | Reset | Description |
|---|---|---|---|
| 31:00 | RW | 0 | **Upper Address (UADDR):** Upper 32 bits of the system specified message address. |

### 2.3.5 Offset MSICAP + Ch: MD – Message Signaled Interrupt Message Data

| Bits | Type | Reset | Description |
|---|---|---|---|
| 15:00 | RW | 0 | **Data (DATA):** This 16-bit field is programmed by system software if MSI is enabled. Its content is driven onto the lower word (PCI AD[15:0]) during the data phase of the MSI memory write transaction. |

## 2.4 Other Capability Pointers

Though not mentioned in this specification, other capability pointers may be necessary, depending upon the implementation space. Examples would be the PCI-X capability for PCI-X implementations, PCI-Express capability for PCI-Express implementations, and potentially the vendor specific capability pointer.

These capabilities are beyond the scope of this specification.

# 3 Controller Registers

The controller registers exist in non-cacheable memory space. Locked accesses are not supported. If host software attempts to perform locked transactions to the registers, indeterminate results may occur. Register accesses shall have a maximum size of 64-bits; 64-bit access shall not cross an 8-byte alignment boundary.

The registers are broken into two sections – global registers and port control. All registers that start below address 100h are global and meant to apply to the entire controller. The port control registers are the same for all ports, and there are as many register banks as there are ports. Each shall have the register set defined in section 3.3.

The global registers are only present as described when the port(s) are exposed as part of a stand-alone PCI device. If the port(s) are exposed as part of an AHCI controller, then the global registers are as defined in the AHCI specification (see section 3.2).

All registers not defined and all reserved bits within registers return '0' when read.

| Start | End | Description |
|---|---|---|
| 00h | 13h | Generic Host Control |
| 14h | 9Fh | Reserved |
| A0h | FFh | Vendor Specific registers |
| 100h | 17Fh | Port 0 port control registers |
| 180h | 1FFh | Port 1 port control registers |
| 200h | FFFh | (Ports 2 – port 29 control registers) |
| 1000h | 107Fh | Port 30 port control registers |
| 1080h | 10FFh | Port 31 port control registers |

## 3.1 Generic Host Control

The following registers apply to the entire controller when exposed as a stand-alone PCI device.

| Start | End | Symbol | Description |
|---|---|---|---|
| 00h | 03h | CAP | Controller Capabilities |
| 04h | 07h | Reserved | Reserved |
| 08h | 0Bh | IS | Interrupt Status |
| 0Ch | 0Fh | PI | Ports Implemented |
| 10h | 13h | VS | Version |

### 3.1.1 Offset 00h: CAP – Controller Capabilities

This register indicates basic capabilities of the NVM controller to host software.

| Bit | Type | Reset | Description |
|---|---|---|---|
| 31:24 | RO | Impl Spec | **Port Ready Wait (PRW):** This is the worst case time that host software shall wait for PxCMD.PRDY to be set to '1' after PxCMD.NVMP transitions from '0' to '1'. This worst case time may be experienced after an unclean shutdown; typical times are expected to be much faster. This field is in 1 second units. |
| 23:13 | RO | 0h | Reserved |
| 12:08 | RO | Impl Spec | **Number of Command Slots (NCS):** 0's based value indicating the number of command slots supported by this controller, where 0h corresponds to 1 command slot. A minimum of 1 and maximum of 32 slots per port may be supported. |
| 07:05 | RO | 0h | Reserved |
| 04:00 | RO | Impl Spec | **Number of Ports (NP):** 0's based value indicating the maximum number of ports supported by the controller silicon, where 0h corresponds to 1 port. A minimum of 1 and maximum of 32 ports may be supported. Note that the number of ports indicated in this field may be more than the number of ports indicated in the PI register. |

### 3.1.2 Offset 08h: IS – Interrupt Status Register

This register indicates which of the ports within the controller have an interrupt pending and require service.

| Bit | Type | Reset | Description |
|---|---|---|---|
| 31:0 | RWC | 0 | **Interrupt Pending Status (IPS):** If set, indicates that the corresponding port has an interrupt pending. Host software may use this information to determine which ports require service after an interrupt. Refer to the port specific PxIS and PxIE registers. |

### 3.1.3 Offset 0Ch: PI – Ports Implemented

This register indicates which ports are exposed by the controller. It is loaded by the BIOS or an expansion ROM. It indicates which ports that the controller supports are available for the host to use. For example, on a controller that supports 6 ports as indicated in CAP.NP, only ports 1 and 3 could be available, with ports 0, 2, 4, and 5 being unavailable.

Host software shall not read or write to registers within unavailable ports.

| Bit | Type | Reset | Description |
|---|---|---|---|
| 31:0 | RO | HwInit | **Port Implemented (PI):** This register is bit significant. If a bit is set to '1', the corresponding port is available for the host to use. If a bit is cleared to '0', the port is not available for the host to use. The maximum number of bits set to '1' shall not exceed CAP.NP + 1, although the number of bits set in this register may be fewer than CAP.NP + 1. At least one bit shall be set to '1'. |

### 3.1.4 Offset 10h: VS – NVMHCI Version

This register indicates the major and minor version of the NVMHCI specification that the controller implementation supports. The upper two bytes represent the major version number, and the lower two bytes represent the minor version number. Example: Version 3.12 would be represented as 00030102h. Valid versions of the specification are: 1.0.

#### 3.1.4.1 VS Value for 1.0 Compliant Controllers

| Bit | Type | Reset | Description |
|---|---|---|---|
| 31:16 | RO | 0001h | **Major Version Number (MJR):** Indicates the major version is "1" |
| 15:00 | RO | 0000h | **Minor Version Number (MNR):** Indicates the minor version is "0". |

## 3.2 Generic Host Control under AHCI

This section defines the NVMHCI global registers that exist when NVMHCI is exposed as a port(s) under an AHCI controller. The presence of these registers is indicated in bit 1 of the AHCI CAP2 register.

The following global registers are present when the NVMHCI device is exposed as a port under AHCI, at the AHCI BAR5 memory offset noted.

| Start | End | Symbol | Description |
|---|---|---|---|
| 60h | 63h | CAP | Controller Capabilities |
| 64h | 67h | Reserved | Reserved |
| 68h | 6Bh | Reserved | Reserved (Interrupt Status register in AHCI used) |
| 6Ch | 6Fh | PI | Ports Implemented |
| 70h | 73h | VS | Version |
| 74h | 9Fh | Reserved | Reserved |

The PI (Ports Implemented) register indicates the NVMHCI port(s) that are present. The AHCI PI (Ports Implemented) register shall only indicate SATA ports that are present. The AHCI IS (Interrupt Status) register shall be used for NVMHCI as well. I.e., global interrupt indications for the controller are indicated in one register and there is not a separate NVMHCI IS (Interrupt Status) register.

All other registers are as defined in section 3.1. The NVMHCI controller shall take no action based on an AHCI HBA Reset. In addition, AHCI functionality like enclosure management, LEDs, and other AHCI features do not apply to the NVMHCI ports.

### 3.3 Port Registers (one set per port)

This section describes the registers necessary to implement per exposed NVMHCI port. All ports shall have the same register mapping. The algorithm for host software to determine the offset to an NVMHCI port's registers is as follows:

- Port offset = 100h + (PI Asserted Bit Position * 80h)

This formula is valid for an NVMHCI port exposed under an AHCI controller as well as for a stand-alone PCI device. For an NVMHCI device exposed as a stand-alone PCI device, port 0 starts at 100h, port 1 starts at 180h, port 2 starts at 200h, port 3 at 280h, etc.

| Start | End | Symbol | Description |
|---|---|---|---|
| 00h | 03h | PxCLB | Port x Command List Base Address |
| 04h | 07h | PxCLBU | Port x Command List Base Address Upper 32-Bits |
| 08h | 0Fh | Reserved | Reserved |
| 10h | 13h | PxIS | Port x Interrupt Status |
| 14h | 17h | PxIE | Port x Interrupt Enable |
| 18h | 1Bh | PxCMD | Port x Command and Status |
| 1Ch | 23h | Reserved | Reserved |
| 24h | 27h | PxSIG | Port x Signature |
| 28h | 37h | Reserved | Reserved |
| 38h | 3Bh | PxCI | Port x Command Issue |
| 3Ch | 6Fh | Reserved | Reserved |
| 70h | 7Fh | PxVS | Port x Vendor Specific |

### 3.3.1 Offset 00h: PxCLB – Port x Command List Base Address

| Bit | Type | Reset | Description |
|---|---|---|---|
| 31:10 | RW | Impl Spec | **Command List Base Address (CLB):** Indicates the 32-bit base physical address for the command list for this port. This base is used when fetching commands to execute. The structure pointed to by this address range is 1KB in length. This address shall be 1KB aligned as indicated by bits 09:00 being read only. |
| 09:00 | RO | 0 | Reserved |

### 3.3.2 Offset 04h: PxCLBU – Port x Command List Base Address Upper 32-bits

| Bit | Type | Reset | Description |
|---|---|---|---|
| 31:00 | RW | Impl Spec | **Command List Base Address Upper (CLBU):** Indicates the upper 32-bits for the command list base physical address for this port. This base is used when fetching commands to execute. |

### 3.3.3 Offset 10h: PxIS – Port x Interrupt Status

This register indicates pending interrupts that require service.

| Bit | Type | Reset | Description |
|---|---|---|---|
| 31:30 | RO | 0h | Reserved |
| 29 | RWC | 0 | **Host Bus Fatal Error Status (HBFS):** Indicates that the controller encountered a host bus error that it cannot recover from, such as a bad software pointer. In PCI, such an indication would be a target or master abort. |
| 28 | RWC | 0 | **Host Bus Data Error Status (HBDS):** Indicates that the controller encountered a data error (e.g. uncorrectable ECC / parity) when reading from or writing to system memory. |
| 27 | RWC | 0 | **Fatal Error Status (FES):** Indicates that the controller encountered a fatal error. The controller stops processing on all fatal error conditions. |
| 26:02 | RO | 0h | Reserved |
| 01 | RWC | 0 | **Asynchronous Notification Status (ANS):** This bit is set to ‘1’ by hardware when the NVM subsystem has status to indicate to the host with the Get Status command. |
| 00 | RWC | 0 | **Command Completion Status (CCS):** This bit is set to ‘1’ by hardware upon completion of a command with the CH[z].I (interrupt) bit set to ‘1’. |

### 3.3.4 Offset 14h: PxIE – Port x Interrupt Enable

This register enables and disables the reporting of the corresponding interrupt to host software. When a bit is set (‘1’) and the corresponding interrupt condition is active, then an interrupt is generated. Interrupt sources that are disabled (‘0’) are still indicated in the PxIS register. This register is symmetrical with the PxIS register.

Note that these interrupts are also qualified with the PCI interrupt enable, and if operating as a port under AHCI it is qualified by the global AHCI interrupt enable.

| Bit | Type | Reset | Description |
|---|---|---|---|
| 31:30 | RO | 0h | Reserved |
| 29 | RW | 0 | **Host Bus Fatal Error Enable (HBFE):** When set and PxIS.HBFS is set, the controller shall generate an interrupt. |
| 28 | RW | 0 | **Host Bus Data Error Enable (HBDE):** When set and PxIS.HBDS is set, the controller shall generate an interrupt.. |
| 27 | RW | 0 | **Fatal Error Enable (FEE):** When set, and PxIS.FES is set, the controller shall generate an interrupt. |
| 26:02 | RO | 0h | Reserved |
| 01 | RW | 0 | **Asynchronous Notification Enable (ANE):** When set and PxIS.ANS is set, the controller shall generate an interrupt. |
| 00 | RW | 0 | **Command Completion Enable (CCE):** When set and PxIS.CCS is set, the controller shall generate an interrupt. |

### 3.3.5 Offset 18h: PxCMD – Port x Command and Status

| Bit | Type | Reset | Description |
|---|---|---|---|
| 31:16 | RO | 0 | Reserved |
| 15 | RO | 0 | **Command List Running (CR):** When this bit is set to '1', the command list DMA engine for the port is running. |
| 14:06 | RO | 0 | Reserved |
| 05 | RO | HwInit | **Cache Use Across Power Events (CUAPE):** When set to '1', it is recommended that thte host driver rely upon and use previously cached data across system power events and reboots. When cleared to '0', it is recommended that the host driver not rely upon nor use cached data across system power events and reboots. |
| 04 | RO | 0 | **NVM Critical Error (NCE):** If set to '1' then there is a critical error with the NVM device as a whole. This field is primarily used for critical errors discovered during initialization of the NVM device. If cleared to '0', there is not a critical error with the NVM device as a whole. Host software should check this bit prior to setting PxCMD.ST to '1'. If there is a critical error, host software may issue the Get Status command to obtain more information about the error condition. |
| 03 | RO | 0 | **Port Ready (PRDY):** This field indicates whether the controller is ready to process commands for an attached NVM device. Host software shall only set the ST bit to '1' if PRDY is set to '1'. Host software should wait for up to CAP.PRW seconds for PRDY to be '1' after PxCMD.NVMP is set to '1'.<br>This field is not affected by the PxCMD.ST bit. It is expected that this bit is set to and remains '1' after controller initialization is complete with an attached NVM device. |
| 02 | RWS | 0 | **Software Reset (SR):** When set to '1' by host software, this bit causes an internal reset of the port and may include a reset of the NVM devices present. All state machines that relate to data transfers shall return to an idle condition. When the controller has finished the reset action, it shall clear this bit to '0'. A host software write of '0' shall have no effect. This bit shall only be set to '1' when PxCMD.PRDY is set to '1' and PxCMD.ST is cleared to '0'. |
| 01 | RO | HwInit | **NVM Present (NVMP):** This field is set to '1' when an NVM device is attached to this port. This field is cleared to '0' when non-volatile memory is not attached to this port. |
| 00 | RW | 0 | **Start (ST):** When set, the controller may process the command list. When cleared, the controller shall not process the command list. Whenever this bit is changed from a '0' to a '1', the controller starts processing the command list at entry '0'. Whenever this bit is changed from a '1' to a '0', the PxCI register is cleared by the controller after it is in an idle state. This bit shall only be set to '1' when PRDY is set to '1''. If NCE is set to '1', the only command that host software may issue is Get Status. If this bit is cleared to '0' when commands are outstanding on the port, then software should complete a software reset before issuing new commands. |

### 3.3.6 Offset 24h: PxSIG – Port x Signature

This is a 32-bit register which identifies that this port is used for non-volatile memory. This register is important when the NVMHCI controller is a port in an AHCI controller so that the NVMHCI register interface may be distinguished.

| Bit | Type | Reset | Description |
|---|---|---|---|
| 31:00 | RO | 4E564D49h | **Signature (SIG):** Contains the non-volatile memory signature ('NVMI' in ASCII). |

### 3.3.7 Offset 38h: PxCI – Port x Command Issue

| Bit | Type | Reset | Description |
|---|---|---|---|

| | | | |
|---|---|---|---|
| 31:0 | RWS | 0 | **Commands Issued (CI):** This field is bit significant. Each bit corresponds to a command slot, where bit 0 corresponds to command slot 0. This field is set by host software to indicate to the controller that a command has been built in system memory for a command slot and may be sent to the NVM subsystem. When a command is completed (with success or error), the corresponding bit is cleared to '0' by the controller. Bits in this field shall only be set to '1' by host software when PxCMD.ST is set to '1'.<br>This field is also cleared when PxCMD.ST is written from a '1' to a '0' by host software.<br>The NVMHCI subsystem may process commands out of order. Host software shall not place commands in the list that may not be re-ordered arbitrarily. Data may not be committed to the NVM device in the order that commands are received. |

### 3.3.8 Offset 70h to 7Fh: PxVS – Vendor Specific

The registers at offset 70h to 7Fh are vendor specific.

## 3.4 Index/Data Pair registers

Index/Data Pair registers provide host software with a mechanism to access the NVMHCI memory mapped registers using I/O space based registers. On PC based platforms, host software (BIOS, Option ROMs, OSes) written to operate in 'real-mode' (8086 mode) are unable to access registers in a PCI function's address space, if the address space is memory mapped and mapped above 1MB.

The Index/Data Pair mechanism allows host software to access all of the memory mapped NVMHCI registers using indirect I/O addressing in lieu of direct memory mapped access.

### 3.4.1 Restrictions

Host software shall not alternate between Index/Data Pair based access and direct memory mapped access methods. After using direct memory mapped access to the controller registers, the Index/Data Pair mechanism shall not be used.

### 3.4.2 Register Definition

The following registers describe the registers necessary to implement Index/Data Pair.

| Start | End | Symbol | Description |
|---|---|---|---|
| 00h | 03h | IDX | Index register |
| 04h | 07h | DAT | Data register |

### 3.4.3 Offset 00h: IDX – Index Register

| Bit | Type | Reset | Description |
|---|---|---|---|
| 31:02 | RW | 0h | **Index (IDX):** This register selects the Dword offset of the memory mapped NVMHCI register to be accessed. Host software shall not set this to a value beyond the maximum register offset implemented. |
| 01:00 | RO | 0h | Reserved |

### 3.4.4 Offset 04h: DAT – Data Register

| Bit | Type | Reset | Description |
|---|---|---|---|
| 31:00 | RW | na | **Data (DAT):** This register is a "window" through which data is read or written to the memory mapped register pointed to by the Index register. A physical register is not implemented as the data is actually stored in the memory mapped registers. Since this is not a physical register, the reset value is the same as the reset value of the register that the Index register is currently pointing to. |

# 4 System Memory Structures

## 4.1 Controller Memory Space Usage

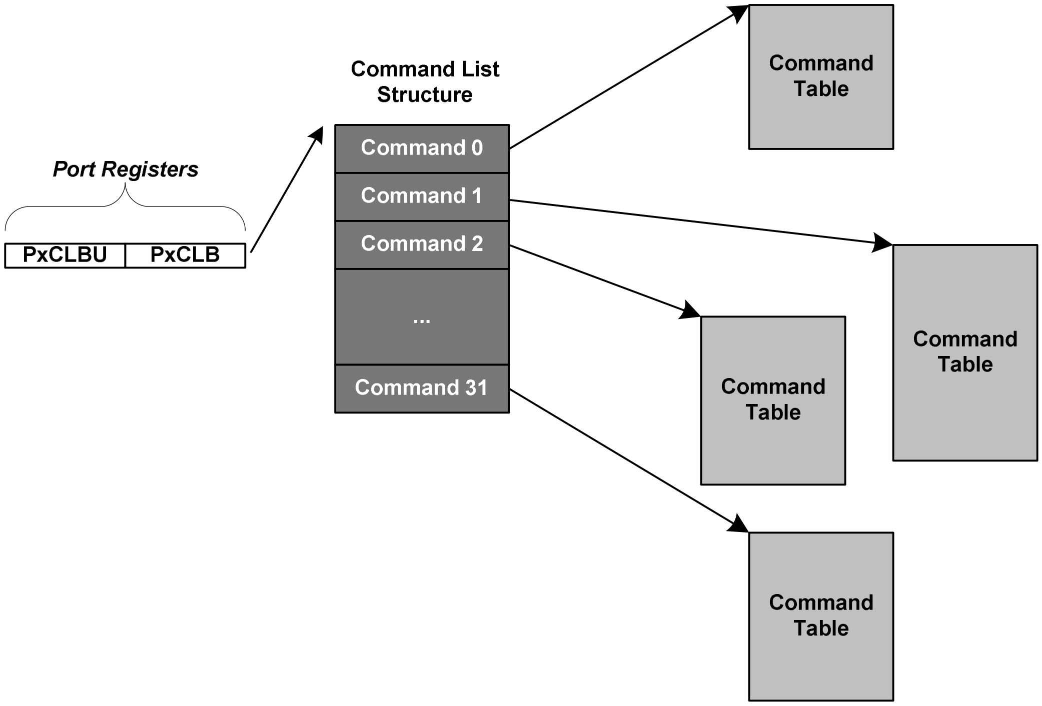
Most communication between host software and the NVM subsystem is via system memory descriptors. These descriptors describe commands to be executed, and data transfer operations that are part of those commands.

## 4.2 Port Memory Usage

The descriptor per port that is used to convey information is the Command List. The Command List contains a list of 1 to 32 commands available for a port to execute.

The base of the Command List structure is indicated by a 64-bit pointer specified in the PxCLB/PxCLBU registers. An overview of the overall structure is shown in Figure 2, and the following sections describe each area.

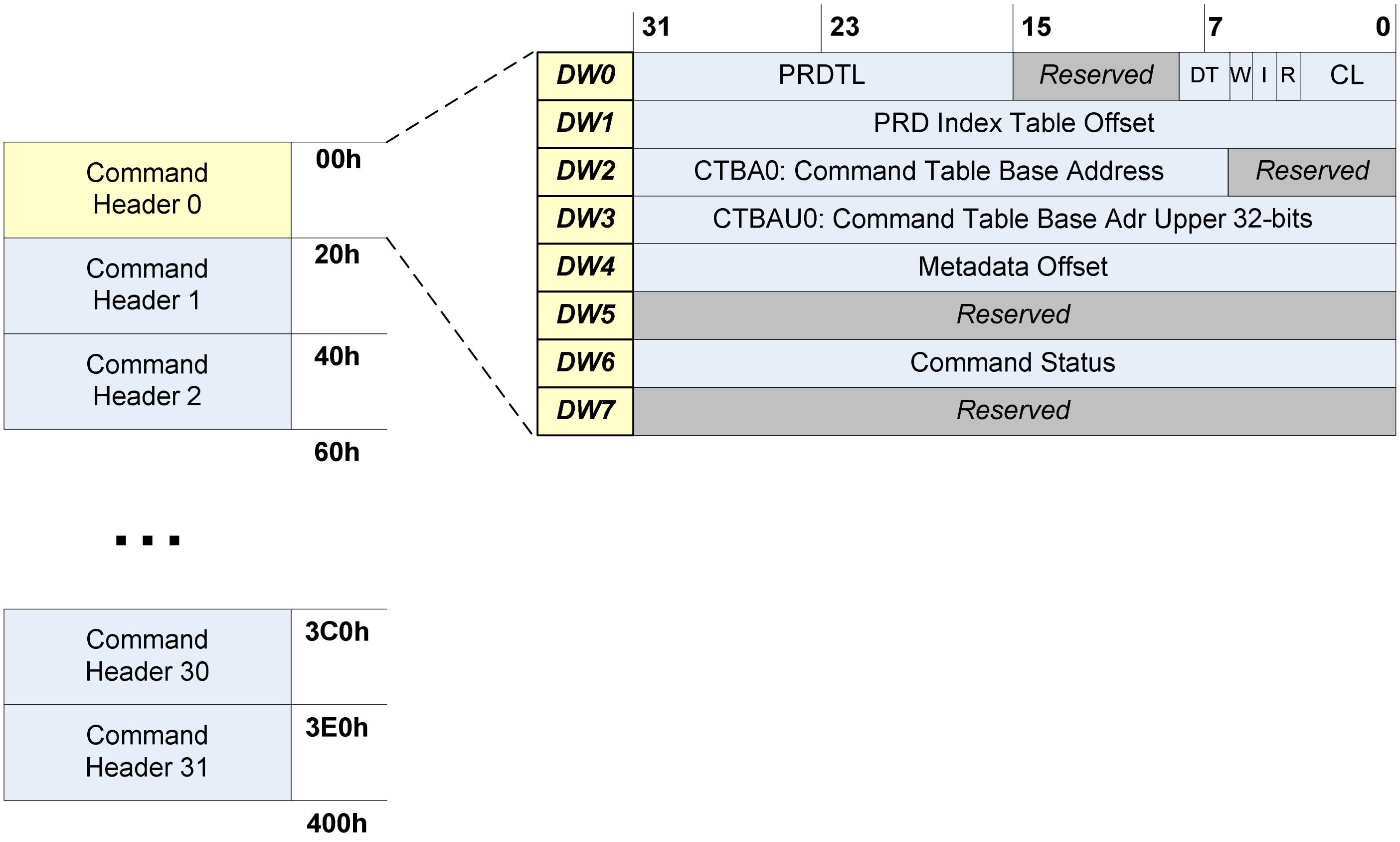
**Figure 2: Port System Memory Structures**



### 4.2.1 Command List Structure

Figure 3 shows the command list structure. Each entry contains a command header, which is a 32-byte structure that details the direction, type, and scatter/gather pointer of the command. Further details of each field are listed below.

**Figure 3: Command List Structure**



| | 31 | 23 | 15 | 7 | 0 |
|---|---|---|---|---|---|
| *DW0* | PRDTL | | *Reserved* | DT<br>W<br>I<br>R | CL |
| *DW1* | PRD Index Table Offset | | | | |
| *DW2* | CTBA0: Command Table Base Address | | | | *Reserved* |
| *DW3* | CTBAU0: Command Table Base Adr Upper 32-bits | | | | |
| *DW4* | Metadata Offset | | | | |
| *DW5* | *Reserved* | | | | |
| *DW6* | Command Status | | | | |
| *DW7* | *Reserved* | | | | |

The fields inside the command header are:

**Figure 4: DW 0 – Description Information**

<table>
<tr><th>Bit</th><th>Description</th></tr>
<tr><td>31:16</td><td><b>Physical Region Descriptor Table Length (PRDTL):</b> Length of the scatter/gather descriptor table in entries, called the Physical Region Descriptor Table. Each entry is 4 Dwords. A ‘0’ represents 0 entries, FFFFh represents 65,535 entries. The controller uses this field to know when to stop fetching PRDs. If this field is ‘0’, then no data transfer shall occur with the command (although metadata may be transferred, see DT field).</td></tr>
<tr><td>15:09</td><td>Reserved</td></tr>
<tr><td>08:07</td><td><b>Data Transfer (DT):</b> This field indicates if a data transfer occurs as part of this command, and the type of that data transfer. DT[0] if set to ‘1’ indicates that metadata shall be transferred. DT[1] if set to ‘1’ indicates that data shall be transferred.
<table>
<tr><th>Bits</th><th>Definition</th></tr>
<tr><td>00b</td><td>No data or metadata transfer</td></tr>
<tr><td>01b</td><td>Metadata transfer only</td></tr>
<tr><td>10b</td><td>Data transfer only</td></tr>
<tr><td>11b</td><td>Data and metadata transfer</td></tr>
</table></td></tr>
<tr><td>06</td><td><b>Write (W):</b> This field indicates the direction of data and/or metadata transfer. When set to ‘1’, indicates that the transfer is from system memory to the NVM device. When cleared to ‘0’, indicates that the transfer is from the NVM device to system memory. This field is set to ‘1’ for the Write, Set Features, and Dataset Management commands. This field is cleared to ‘0’ for all other commands and when there is no data transfer as indicated by DT set to a value of 00b.</td></tr>
<tr><td>05</td><td><b>Interrupt (I):</b> When set to ‘1’, hardware shall set PxIS.CCS to ‘1’ on completion of this command. When cleared to ‘0’, hardware shall not set PxIS.CCS to ‘1’ on completion of this command. This field does not affect setting of other PxIS bits by the controller.</td></tr>
<tr><td>04</td><td>Reserved</td></tr>
<tr><td>03:00</td><td><b>Command Length (CL):</b> Length of the command to be transferred. A ‘0’ represents 0 Dwords, ‘4’ represents 4 Dwords. A length of ‘0’ is illegal. The maximum value allowed is 8h, or 8 Dwords.</td></tr>
</table>

**Figure 5: DW 1 - Structure Offsets**

| Bit | Description |
|---|---|
| 31:00 | **PRD Index Table Offset (PITO):** This field contains the Dword offset within the Command Table of the PRD Index Table. The PRD Index Table may be located at a Dword aligned boundary anywhere after the PRD Table in the Command Table. The PRD Index Table is only used for Read and Write commands. |

**Figure 6: DW 2 – Command Table Base Address**

| Bit | Description |
|---|---|
| 31:07 | **Command Table Descriptor Base Address (CTBA):** Indicates the lower 32-bits of the physical address of the command table, which contains the Command, PRD Table, Metadata Region, and PRD Index Table. This address shall be aligned to a 128-byte address, indicated by bits 06:00 being reserved. |
| 06:00 | Reserved |

**Figure 7: DW 3 – Command Table Base Address Upper**

| Bit | Description |
|---|---|
| 31:00 | **Command Table Descriptor Base Address Upper 32-bits (CTBAU):** This is the upper 32-bits of the Command Table Base. |

**Figure 8: DW 4 - Metadata Offsets**

| Bit | Description |
|---|---|
| 31:00 | **Metadata Offset (MO):** When metadata is present, as indicated by the DT[0] bit being set to ‘1’ in Dword 0, this field contains the Dword offset within the Command Table of the metadata. The Metadata Region may be located at a Dword aligned boundary anywhere after the PRD Table in the Command Table. |

**Figure 9: DW 5 – Reserved**

| Bit | Description |
|---|---|

| | |
|---|---|
| 31:00 | Reserved |

**Figure 10: DW 6 – Command Status**

<table>
<tr><th>Bit</th><th>Description</th></tr>
<tr><td>31:00</td><td><b>Command Status (CS):</b> This is the completion status of the associated command. The command status field is valid after the NVM subsystem has cleared the corresponding PxCI bit to zero. Refer to section 5.2.
<table>
<tr><th>Bit</th><th>Definition</th></tr>
<tr><td>31:24</td><td><b>Vendor Specific Status (VSS):</b> Vendor specific</td></tr>
<tr><td>23:16</td><td>Reserved</td></tr>
<tr><td>15:08</td><td><b>Command Specific Status (CSS):</b> Contains the command specific status for the associated command.</td></tr>
<tr><td>07:00</td><td><b>Overall Command Status (OCS):</b> Contains the command status that is common for all commands.</td></tr>
</table>
</td></tr>
</table>

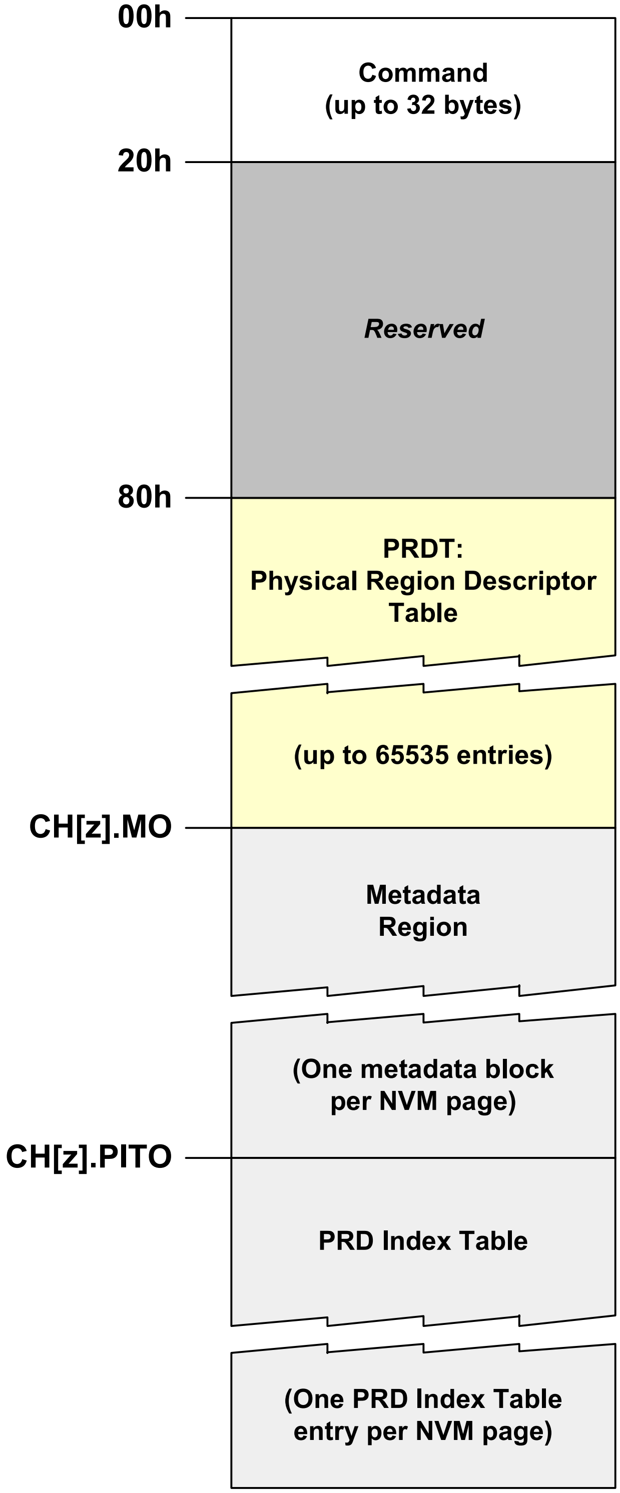
**Figure 11: DW 7 – Reserved**

| Bit | Description |
|---|---|
| 31:00 | Reserved |

### 4.2.2 Command Table

Each entry in the command list points to a structure called the command table.

**Figure 12: Command Table**



Each command contains several fields. The fields break down as follows:

#### 4.2.2.1 Command (CMD)

This is the command to execute. See section 5.3 for a description of the commands that may be executed.

#### 4.2.2.2 Physical Region Descriptor Table (PRDT)

This table contains the scatter / gather list for the data transfer. It contains a list of 0 (no data to transfer) to up to 65,535 entries.

**Figure 13: PRD Entry**

| | 31 | 23 | 15 | 7 | 0 |
|---|---|---|---|---|---|
| *DW0* | **DBA: Data Base Address** | | | | **00** |
| *DW1* | **DBAU: Data Base Addr Upper 32-bits** | | | | |
| *DW2* | ***Reserved*** | | | | |
| *DW3* | ***Reserved*** | | **DBC: Byte Count** | | **11** |

A breakdown of each field in a PRD table is shown below. Item 0 refers to the first entry in the PRD table. Item "CH[PRDTL] – 1" refers to the last entry in the table, where the length field comes from the PRDTL field in the command list entry for this command slot.

**Figure 14: DW 0 – Data Base Address**

| Bit | Description |
|---|---|
| 31:02 | **Data Base Address (DBA):** Indicates the lower 32-bits of the physical address of the data block. The block shall be Dword aligned, indicated by bits 01:00 being reserved. |
| 01:00 | Reserved |

**Figure 15: DW 1 – Data Base Address Upper**

| Bit | Description |
|---|---|
| 31:00 | **Data Base Address Upper 32-bits (DBAU):** This is the upper 32-bits of the data block physical address. |

**Figure 16: DW 2 – Reserved**

| Bit | Description |
|---|---|
| 31:00 | Reserved |

**Figure 17: DW 3 – Description Information**

| Bit | Description |
|---|---|
| 31:18 | Reserved |
| 17:00 | **Data Byte Count (DBC):** A '0' based value that indicates the length, in bytes, of the data block. A maximum of length of 256KB may exist for any entry. Bits 1:0 of this field shall be 11b to indicate Dword granularity. A value of '3' indicates 4 bytes, '7' indicates 8 bytes, etc. |

#### 4.2.2.3 Metadata Region (MR)

Each NVM page of data may have metadata associated with it. If CH[z].DT[0] is set to '1', then metadata shall be transferred to/from the Command Table at the offset indicated by CH[z].MO. When metadata is transferred as part of a Read or Write command, the amount of metadata transferred per NVM page shall be equal to the size reported in the Identify structure (see section 5.3.5.1.8). If metadata is not written as part of a Write command, then any previous metadata written for the NVM page is indeterminate. It is host software's responsibility to ensure metadata is preserved as needed for a particular NVM page.

The metadata for each NVM page is word aligned. For example, if metadata for each NVM page is 21 bytes, then each metadata entry shall consume 22 bytes.

A read of any data size (i.e. 0 sectors, 1 sector, 2 sectors, …, n sectors) may also return metadata. For writes, if the write is less than a full aligned NVM page and includes metadata, the metadata provided in the current command completely replaces all previous metadata for that NVM page.

The controller shall ensure that when out of order data transfers are performed that the metadata that is transferred corresponds to the data being transferred. For example, if there are three NVM pages of data being transferred and the third NVM page is transferred first, in this case the third set of metadata is also transferred first. The controller transfers metadata from CH[z].MO + (NVM page index * metadata size) for each NVM page transferred that includes metadata.

#### 4.2.2.4 PRD Index Table

Each data transfer for a Read or Write command has a PRD Index Table built by host software. There is no PRD Index Table associated with other commands that include a data transfer (e.g. Identify). The PRD Index Table is located in the Command Table at the offset indicated by CH[z].PITO.

**Figure 18: PRD Index Table**

| | 31 | 23 | 15 | 7 | 0 |
|---|---|---|---|---|---|
| ***NVM Page n*** | **Dword Offset** | | **PRD Entry** | | |
| ***NVM Page n+1*** | **Dword Offset** | | **PRD Entry** | | |
| ***NVM Page ...*** | **...** | | | | |
| ***NVM Page n + x*** | **Dword Offset** | | **PRD Entry** | | |

The PRD Entry field is filled with the index value of the corresponding PRD table entry where the transfer for the corresponding NVM Page of data starts. The Dword Offset field is filled with the offset in Dwords from the start of the PRD table entry where the transfer for the corresponding NVM Page of data starts.

As an example, if the first two PRD entries in the PRD table had the following sizes:

- Entry 0: Data transfer of 1KB
- Entry 1: Data transfer of 3KB

If the NVM Page size is 2KB, then NVM Page n+1 would have a PRD Entry value of 1 and a Dword offset value of 256. This corresponds to the data transfer for NVM Page n+1 starting at PRD entry 1 and offset 1KB into that system memory location.

The PRD Index Table is impacted by the sector alignment reported in the Identify data structure, see section 5.3.5.1.12. Specifically, the PRD Index Table entries are per NVM page. Software needs to consider sector alignment to determine where each NVM page starts.

The PRD Index Table enables out of order data transfer for better concurrency and thus performance. This structure avoids hardware walking the PRD table entries themselves to determine where the data transfer begins for a particular page.

# 5 Memory Organization and Command Set

## 5.1 Memory Organization

The NVM device is comprised of NVM pages. The NVM page is the recommended minimum write granularity for the NVM media. Writes that are not an NVM page size granularity or are unaligned may incur a performance penalty.

Each NVM page is comprised of some number of sectors. A sector is the smallest unit of data that may be read or written from the NVM device. The sector size, reported in bytes, is always a power of two. Sector sizes may be 512 bytes, 1KB, 2KB, 4KB, 8KB, etc. The NVM device indicates the desired sector size given the constraints of the NVM media as part of the data structure returned with the Identify command.

The NVM page may contain a single sector or a number of sectors. For example, an NVM page size of 8KB may be comprised of 1 sector that is 8KB in size or it may be comprised of 16 sectors that are 512 bytes in size.

The NVM device indicates an optimal write transfer size. The optimal write transfer size is the number of NVM pages that should be transferred as a unit for the best NVM device performance. For example, the NVM device may perform better if 64 NVM pages are written at one time.

The memory organization attributes are indicated in the Identify data structure, see section 5.3.5.

## 5.2 Command Status

Each command has a command status that is returned within the CH[z].CS field in the associated Command Header. This field is valid after the host controller has cleared the associated PxCI bit to ‘0’, indicating that the command has completed. Note that the PxCI bit is cleared to ‘0’ regardless of whether the command completed with success or in error. Host software shall examine the command status to determine whether the command completed successfully or incurred a fatal error.

The command status field is defined in Figure 10 and contains three primary fields: Overall Command Status, Command Specific Status, and Vendor Specific Status.

The Overall Command Status (OCS) field is common for all commands. The Overall Command Status field is defined in Figure 19.

**Figure 19: Overall Command Status definition**

| Bit | Definition |
|---|---|
| 07:04 | Reserved |
| 03 | **Health Status (HS):** If set to ‘1’, then the NVM device has health information to report via the Get Status method. If cleared to ‘0’, there is no health information for the NVM device to report. Reporting health information in this field may be used rather than the asynchronous interrupt when commands are already outstanding. |
| 02 | **Abort (ABRT):** If set to ‘1’, then the command was aborted. This bit is set when the command received is malformed. If cleared to ‘0’, the command was not aborted. This bit shall only be set as part of a fatal error condition. |
| 01 | **Non-fatal Error (NFE):** If set to ‘1’, then a non-fatal error occurred. The specific status for a non-fatal error is contained in the Command Specific Status field. If cleared to ‘0’, then there were no non-fatal error conditions. |
| 00 | **Fatal Error (FE):** If set to ‘1’, then the command had a fatal error and did not complete successfully. If cleared to ‘0’, then the command was successfully completed. Note that the command may have encounted non-fatal or health conditions when this bit is cleared to ‘0’. |

The Command Specific Status field is defined for each command described in section 5.3. This status field is specific to the particular command issued.

The Vendor Specific Status field may be used for vendor specific status information.

## 5.3 Command Definitions

NVMHCI includes the commands listed in Figure 20. The following subsections describe the definition for each of the commands in NVMHCI. Commands shall only be issued by host software if an NVM device is present, as indicated by PxCMD.NVMP.

All commands are executed on a port specific basis. For example, the Flush command applies only to the port that it is issued to. Thus, when removing power a Flush command would have to be issued to each individual port.

Host software should appropriately manage command timeout values. A command timeout (Timeout specified in the Command field) starts when PxCI is written for that command. The host should take into account the current workload presented to the port and whether commands have completed recently on the port before taking action due to an expired timeout. The NVMHCI controller should continue to process commands even when a timeout has passed. Host software should include a longer timeout value for a command retry to allow the NVMHCI controller to do more time consuming recovery measures.

**Figure 20: Commands and Opcodes**

| Opcode | O/M | Command |
|---|---|---|
| *No Data Transfer* | | |
| 00h | | Reserved |
| 01h | M | Flush |
| 02h – 0Fh | | Reserved |
| *Data Transfer to the Host* | | |
| 10h | M | Read |
| 11h | M | Get Features |
| 12h | M | Identify |
| 13h | M | Get Status |
| 14h – 1Fh | | Reserved |
| *Data Transfer to the NVM Device* | | |
| 20h | M | Write |
| 21h | M | Set Features |
| 22h | O | Dataset Management |
| 23h – 2Fh | | Reserved |
| Reserved Range | | |
| 30h – 7Fh | | Reserved |
| Vendor Specific Range | | |
| 80h – FFh | O | Vendor specific |

O/M: O = Optional, M = Mandatory

### 5.3.1 Dataset Management

The Dataset Management command is used by the host to indicate attributes for ranges of NVM pages. This includes attributes like frequency that data is read or written, access size, and other information that may be used to optimize performance and reliability.

#### 5.3.1.1 Command Parameters

The command layout is described in Figure 21. The command parameters are described in Figure 22.

**Figure 21: Dataset Management – Command Parameters**

| Field | 31 | 23 | 15 | 7 0 |
|---|---|---|---|---|
| ***Command*** | Timeout | | *Reserved* | Opcode: 22h |
| ***Address Low*** | *Reserved* | | | |
| ***Address High*** | *Reserved* | | | |
| ***Transfer Count*** | *Reserved* | | | # of Ranges |
| ***Parameters*** | *Reserved* | | | |
| ***Attributes*** | Command Attributes | | | |

**Figure 22: Dataset Management – Parameter Description**

| Parameter | Field | Bits | Description |
|---|---|---|---|
| Opcode | Command | 07:00 | The opcode for Dataset Management is 22h. |
| Timeout | Command | 31:16 | The relative time from command issue until the command should be completed by the NVM device. This field is in 10 millisecond units. The minimum value is 10 (corresponding to 100 milliseconds). |
| Number of Ranges | Transfer Count | 07:00 | The number of 16 byte range sets to send to the NVMHCI subsystem. This is a 0-based value. The minimum value is 0h and corresponds to 1 range. The maximum value is 255 corresponding to 256 ranges. |
| **Command Attributes** | | | |
| AR: Atomic Read | Attributes | 00 | If set to '1' then the dataset should be optimized for atomic read access. The host expects to perform operations on all ranges provided as a single object for reads. If this bit is set to '1', then the Atomic Read Range context attribute in each range shall be ignored. |
| AW: Atomic Write | Attributes | 01 | If set to '1' then the dataset should be optimized for atomic write access. The host expects to perform operations on all ranges provided as a single object for writes. If this bit is set to '1', then the Atomic Write Range context attribute in each range shall be ignored. |
| D: Deallocate | Attributes | 02 | If set to '1' then the NVM subsystem may deallocate all provided ranges. If a read occurs to a deallocated range, the NVMHCI subsystem shall return all zeros, all ones, or the last data written to the associated NVM page(s). |
| Reserved | Attributes | 31:03 | Reserved |

The data that the Dataset Management command provides is a list of ranges with context attributes. Each range consists of a starting sector address, a length of NVM pages that the range consists of, and the context attributes that should be applied for that range. The definition for the ranges is specified in Figure 23 for the maximum case of 256 ranges. Each starting sector address shall be NVM page aligned.

**Figure 23: Dataset Management – Range Definition**

| Range | Byte | Field |
|---|---|---|
| | | |
| Range 0 | 03:00 | Context Attributes |
| | 05:04 | Reserved |
| | 07:06 | Length in NVM pages |
| | 15:08 | Starting sector address |
| | | |
| Range 1 | 19:16 | Context Attributes |
| | 21:20 | Reserved |
| | 23:22 | Length in NVM pages |
| | 31:24 | Starting sector address |
| … | | |
| | | |
| Range 255 | 4083:4080 | Context Attributes |
| | 4085:4084 | Reserved |
| | 4087:4086 | Length in NVM pages |
| | 4095:4088 | Starting sector address |

#### 5.3.1.2 Command Status

This section describes the Command Specific Status field for the Dataset Management command. Figure 24 describes the Command Specific Status when the command was successful (CH[z].CS.OCS.FE = ‘0’). Figure 25 describes the Command Specific Status when the command had a fatal error (CH[z].CS.OCS.FE = ‘1’).

**Figure 24: Dataset Management – Command Specific Status, Success**

| Bits | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Command Specific Status | Reserved | | | | | | | |

**Figure 25: Dataset Management – Command Specific Status, Fatal Error**

| Bits | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Command Specific Status | Reserved | | | | | | UNAL | CONF |

CONF: Shall be set to ‘1’ if conflicting attribute settings were provided.

UNAL: Shall be set to ‘1’ if a provided range did not start on an NVM page aligned boundary.

#### 5.3.1.3 Context Attributes

The context attributes specified for each range provides information about how the range is intended to be used by host software.

**Note:** The NVM subsystem is required to maintain the integrity of data on the NVM media regardless of whether the attributes provided by host software are accurate.

**Figure 26: Dataset Management – Context Attributes**

<table>
<tr><th>Attribute</th><th>Bits</th><th>Description</th></tr>
<tr><td>AR: Atomic Read Range</td><td>00</td><td>If set to ‘1’ the dataset should be optimized for atomic read access. The host expects to perform operations on the data set as single object for reads.</td></tr>
<tr><td>AW: Atomic Write Range</td><td>01</td><td>If set to ‘1’ the dataset should be optimized for atomic write access. The host expects to perform operations on the data set as single object for writes.</td></tr>
<tr><td>RF: Read Frequency</td><td>03:02</td><td><table>
<tr><th>Value</th><th>Definition</th></tr>
<tr><td>00b</td><td>No read frequency information given.</td></tr>
<tr><td>01b</td><td>Long term storage. Read less than once on average per NVM device power cycle.</td></tr>
<tr><td>10b</td><td>User’s current working set. Read once on average every NVM device power cycle.</td></tr>
<tr><td>11b</td><td>Dynamic data. Read more than once on average per NVM device power cycle.</td></tr>
</table></td></tr>
<tr><td>WF: Write Frequency</td><td>05:04</td><td><table>
<tr><th>Value</th><th>Definition</th></tr>
<tr><td>00b</td><td>No write frequency information given.</td></tr>
<tr><td>01b</td><td>Long term storage. Written less than once on average per NVM device power cycle.</td></tr>
<tr><td>10b</td><td>User’s current working set. Written once on average every NVM device power cycle.</td></tr>
<tr><td>11b</td><td>Dynamic data. Written more than once on average per NVM device power cycle.</td></tr>
</table></td></tr>
<tr><td>RL: Read Latency</td><td>07:06</td><td><table>
<tr><th>Value</th><th>Definition</th></tr>
<tr><td>00b</td><td>No read latency information given.</td></tr>
<tr><td>01b</td><td>Idle. Longer latency acceptable.</td></tr>
<tr><td>10b</td><td>Normal. Typical latency.</td></tr>
<tr><td>11b</td><td>High. Smallest possible latency.</td></tr>
</table></td></tr>
<tr><td>WL: Write Latency</td><td>09:08</td><td><table>
<tr><th>Value</th><th>Definition</th></tr>
<tr><td>00b</td><td>No write latency information given.</td></tr>
<tr><td>01b</td><td>Idle. Longer latency acceptable.</td></tr>
<tr><td>10b</td><td>Normal. Typical latency.</td></tr>
<tr><td>11b</td><td>High. Smallest possible latency.</td></tr>
</table></td></tr>
<tr><td>WP: Write Prepare</td><td>10</td><td>If set to ‘1’ then the provided range is expected to be written in the near future.</td></tr>
<tr><td>Reserved</td><td>23:11</td><td>Reserved</td></tr>
<tr><td>Command Access Size</td><td>31:24</td><td>Number of pages expected to be transferred in a single Read or Write command from this dataset. A value of 0h indicates no Command Access Size is provided.</td></tr>
</table>

#### 5.3.1.4 Deallocate

An NVM page that has been deallocated using the Dataset Management command is no longer deallocated when any sector in the NVM page is written. Read operations do not affect the deallocation status of an NVM page.

### 5.3.2 Flush

The Flush command is used by the host to indicate that any data in temporary storage should be flushed to non-volatile memory.

This command shall be the last command issued by host software to every active port prior to any power down condition and shall be issued and have been completed before any change of the PCI power management state of the NVMHCI device, when sent the P parameter bit shall be set to ‘1’. Any command received after a Flush with the P parameter set to ‘1’ shall indicate to the NVM subsystem that the system is no longer in a power down condition. There is no requirement that Flush be sent prior to a warm reboot.

#### 5.3.2.1 Command Parameters

The command layout is described in Figure 27. The command parameters are described in Figure 28.

**Figure 27: Flush – Command Parameters**

<table>
<tr><th>Field</th><th>31</th><th>23</th><th>15</th><th></th><th>7</th><th>0</th></tr>
<tr><td><i><b>Command</b></i></td><td colspan="2">Timeout</td><td><i>Reserved</i></td><td>P</td><td colspan="2">Opcode: 01h</td></tr>
<tr><td><i><b>Address Low</b></i></td><td colspan="6"><i>Reserved</i></td></tr>
<tr><td><i><b>Address High</b></i></td><td colspan="6"><i>Reserved</i></td></tr>
<tr><td><i><b>Transfer Count</b></i></td><td colspan="6"><i>Reserved</i></td></tr>
<tr><td><i><b>Parameters</b></i></td><td colspan="6"><i>Reserved</i></td></tr>
<tr><td><i><b>Attributes</b></i></td><td colspan="6"><i>Reserved</i></td></tr>
</table>

**Figure 28: Flush – Parameter Description**

| Parameter | Field | Bits | Description |
|---|---|---|---|
| Opcode | Command | 07:00 | The opcode for Flush is 01h. |
| P | Command | 08 | If P is set to ‘1’, then the Flush is being issued immediately prior to a power down condition. If P is cleared to ‘0’, then the Flush is not being issued due to a power down condition. |
| Timeout | Command | 31:16 | The relative time from command issue until the command should be completed by the NVM device. This field is in 10 millisecond units. The minimum value is 10 (corresponding to 100 milliseconds). |

#### 5.3.2.2 Command Status

This section describes the Command Specific Status field for the Flush command. Figure 29 describes the Command Specific Status when the command was successful (CH[z].CS.OCS.FE = ‘0’). Figure 30 describes the Command Specific Status when the command had a fatal error (CH[z].CS.OCS.FE = ‘1’).

**Figure 29: Flush – Command Specific Status, Success**

| Bits | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Command Specific Status | Reserved | | | | | | | |

**Figure 30: Flush – Command Specific Status, Fatal Error**

| Bits | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Command Specific Status | Reserved | | | | | RETR | R | NME |

NME: Shall be set to ‘1’ if the data could not be flushed to the non-volatile memory due to media errors.

RETR: Shall be set to ‘1’ if the NVM subsystem requests that host software retry the request.

### 5.3.3 Get Features

The Get Features command is used to determine parameter settings for features that the NVM subsystem supports. The features that may be retrieved using Get Features are described in section 5.3.7.3.

#### 5.3.3.1 Command Parameters

The command layout is described in Figure 31. The command parameters are described in Figure 32.

**Figure 31: Get Features – Command Parameters**

| Field | 31 | 23 | 15 | 7 0 |
|---|---|---|---|---|
| ***Command*** | Timeout | | *Reserved* | Opcode: 11h |
| ***Address Low*** | *Reserved* | | | Feature |
| ***Address High*** | *Reserved* | | | |
| ***Transfer Count*** | *Reserved* | | | # of Dwords |
| ***Parameters*** | *Reserved* | | | |
| ***Attributes*** | *Reserved* | | | |

**Figure 32: Get Features – Parameter Description**

| Parameter | Field | Bits | Description |
|---|---|---|---|
| Opcode | Command | 07:00 | The opcode for Get Features is 11h. |
| Timeout | Command | 31:16 | The relative time from command issue until the command should be completed by the NVM device. This field is in 10 millisecond units. The minimum value is 10 (corresponding to 100 milliseconds). |
| Feature | Address Low | 07:00 | The Feature to retrieve settings for. |
| # of Dwords | Transfer Count | 07:00 | The number of Dwords to transfer. The maximum transfer is 255 Dwords. The number of Dwords to transfer is based on the Feature being retrieved and its associated data buffer transfer size. |

#### 5.3.3.2 Command Status

This section describes the Command Specific Status field for the Get Features command. Figure 33 describes the Command Specific Status when the command was successful (CH[z].CS.OCS.FE = ‘0’). Figure 34 describes the Command Specific Status when the command had a fatal error (CH[z].CS.OCS.FE = ‘1’).

**Figure 33: Get Features – Command Specific Status, Success**

| Bits | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Command Specific Status | Reserved | | | | | | | |

**Figure 34: Get Features – Command Specific Status, Fatal Error**

| Bits | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Command Specific Status | Reserved | | | | | | PARM | ADDR |

ADDR: Shall be set to ‘1’ if the feature address specified is not supported by the NVM subsystem.

PARM: Shall be set to ‘1’ if the parameters provided are malformed.

### 5.3.4 Get Status

Get Status returns a 64 byte structure that indicates the source of an asynchronous notification event. The host may use this command to determine health information or other details about the NVM subsystem.

#### 5.3.4.1 Command Parameters

The command layout is described in Figure 35. The command parameters are described in Figure 36.

**Figure 35: Get Status – Command Parameters**

| Field | 31 | 23 | 15 | 7 0 |
|---|---|---|---|---|
| ***Command*** | Timeout | | *Reserved* | Opcode: 13h |
| ***Address Low*** | *Reserved* | | | |
| ***Address High*** | *Reserved* | | | |
| ***Transfer Count*** | *Reserved* | | | |
| ***Parameters*** | *Reserved* | | A | Page Type |
| ***Attributes*** | *Reserved* | | | |

**Figure 36: Get Status – Parameter Description**

| Parameter | Field | Bits | Description |
|---|---|---|---|
| Opcode | Command | 07:00 | The opcode for Get Status is 13h. |
| Timeout | Command | 31:16 | The relative time from command issue until the command should be completed by the NVM device. This field is in 10 millisecond units. The minimum value is 10 (corresponding to 100 milliseconds). |
| Page Type | Parameters | 07:00 | The page type of the page to return. This field is only valid when 'A' is cleared to '0'. |
| A | Parameters | 08 | If set to '1', the host requests the next status page that the NVM subsystem has to return to the host. This may be due to an asynchronous notification or due to the NVM subsystem indicating there are more status pages to be read. |

The 64 byte data structure returned has the format in Figure 37. The first byte is the page type and defines the format of the page specific data. The second byte contains common parameters that all pages share. The rest of the page contains data specific to the type of status information being signaled from the NVM subsystem.

**Figure 37: Get Status – Data Structure**

| Byte | Description |
|---|---|
| 00h | Page Type |
| 01h | Page Parameters |
| 3Fh:02h | Page Specific Data |

**Figure 38: Get Status – Page Parameters**

| Parameter | Bits | Description |
|---|---|---|
| Status Continued | 00 | If set to '1' then there is additional status that the NVM subsystem has to return. If cleared to '0', this is the last status that the NVM subsystem has to return. |
| Reserved | 07:01 | Reserved |

**Figure 39: Get Status - Page Types**

| Page Type | Description |
|---|---|
| 00h | Reserved |
| 01h | Health Status |
| 79h:02h | Reserved |
| FFh:80h | Vendor Specific |

#### 5.3.4.2 Command Status

This section describes the Command Specific Status field for the Get Status command. Figure 40 describes the Command Specific Status when the command was successful (CH[z].CS.OCS.FE = '0'). Figure 41 describes the Command Specific Status when the command had a fatal error (CH[z].CS.OCS.FE = '1').

**Figure 40: Get Status – Command Specific Status, Success**

| Bits | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Command Specific Status | Reserved | | | | | | | |

**Figure 41: Get Status – Command Specific Status, Fatal Error**

| Bits | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Command Specific Status | Reserved | | | | NXT | RETR | PT | UNC |

UNC: Shall be set to '1' if the page could not be returned and the page requested was a valid page. This could be due to an NVM media error, a bus transfer error, or insufficient error correction capability.

PT: Shall be set to '1' if the page requested is invalid or not supported by the NVM subsystem.

RETR: Shall be set to '1' if the NVM subsystem requests that host software retry the request.

NXT: Shall be set to '1' if the host requested the next status page using the 'A' bit in the command parameters and there was no status page to return.

#### 5.3.4.3 Asynchronous Notification Generation

The Get Status command may be issued by the host due to the receipt of an asynchronous notification event, due to a time based policy implemented (polling) or due to an error returned from a previous operation (e.g. Read, Write, etc).

Because polling is typically not an efficient mechanism (from a power consumption and CPU usage perspective) and because there are scenarios where the NVM subsystem could encounter an error that may be of immediate interest to the host, it is recommended that the host and NVM subsystem support the generation (NVM subsystem) and detection (host) of asynchronous notification events.

To prevent the NVM subsystem from causing 'interrupt storms', the following conditions shall result in an asynchronous notification (if supported by the NVM subsystem):

- When any event specified in Health Status – Critical Error Status, transitions from '0' to '1'.

- When any event specified in the Health Status – Overall NVM Device Status, transitions from ‘0’ to ‘1’.
- A vendor specific event has occurred that the host should be made aware of.

Note: The asynchronous notification interrupt (setting the PxIS.ANS bit to ‘1’), is only generated on these transitions.

The NVM subsystem may not be capable of maintaining state during D3cold power state, therefore it is possible for an asynchronous notification to be generated during the D3cold to D0initialized power state transition.

#### 5.3.4.4 Page Definitions

##### 5.3.4.4.1 Health Status Page

The Health Status Log Page describes the health of the NVM device in the 64 byte page shown in Figure 42.

**Figure 42: Health Status Log Page**

| Byte | Description |
|---|---|
| 00h | 1h |
| 01h | Page Parameters |
| 02h | Critical Error Status |
| 03h | Overall NVM Device Status |
| 0Fh:04h | Reserved |
| 1Fh:10h | Health Range Report 0 |
| 2Fh:20h | Health Range Report 1 |
| 3Fh:30h | Health Range Report 2 |

The Critical Error Status (as shown in Figure 43) reports why the PxCMD.NCE bit is set to ‘1’.

**Figure 43: Health Status - Critical Error Status**

| Bit | Description |
|---|---|
| 00h | When set to ‘1’, the NVM device is present, but communication could not be established |
| 01h | When set to ‘1’, the NVM device is present, but is unable to be initialized |
| 02h | When set to ‘1’, the NVM device should no longer be used. |
| 07h:03h | Reserved |

The Overall NVM Device Status (as shown in Figure 44) reports the current status of the NVM device.

**Figure 44: Health Status - Overall NVM Device Status**

| Bit | Description |
|---|---|
| 00h | When set to ‘1’ the NVM device is now read only. |
| 01h | When set to ‘1’ the NVM device should be backed up. |
| 07h:02h | Reserved |

The Health Range Reports (as shown in Figure 45) describe the Health Status for individual LBA range of the NVM device.

Note: A NVM device may have more Health Range Reports than may be accommodated in a single health status page. In this case the Health Status page indicates that there is more status to be retrieved by setting the Status Continued bit set to ‘1’. To retrieve the additional Health Range Reports host software issues another Get Status command with the ‘A’ bit in the Parameters field set to ‘1’.

**Figure 45: Health Status - Health Range Report**

| Health Range | Byte | Field |
|---|---|---|
| | | |
| Health Range 0 | 01:00 | Health Vitals |
| | 03:02 | Reserved |
| | 07:04 | Length in NVM pages |
| | 15:08 | Starting sector address |
| | | |
| Health Range 1 | 17:16 | Health Vitals |
| | 19:18 | Reserved |
| | 23:20 | Length in NVM pages |
| | 31:24 | Starting sector address |
| | | |
| Health Range 2 | 33:32 | Health Vitals |
| | 35:34 | Reserved |
| | 39:36 | Length in NVM pages |
| | 47:40 | Starting sector address |

The Health Vitals (as shown in Figure 46) report the current health status of the associated LBA range described in a given Health Range Report. If a Health Range is not valid, then the field “Length in NVM pages” for that Health Range shall report a length of 0h. Health Ranges shall not be sparsely populated; i.e. if a Health Range is not valid then subsequent Health Ranges shall also be marked as not valid.

**Figure 46: Health Status - Health Vitals**

| Bit | Description |
|---|---|
| 00h | When set to ‘1’ this range is now read only. |
| 01h | When set to ‘1’ this range should be backed up. |
| 02h | When set to ‘1’, this range’s data has been lost. |
| 03h | When set to ‘1’ this range should no longer be used. |
| 0Fh:04h | Reserved |

### 5.3.5 Identify

The Identify command returns a 512 byte data structure describing the NVM subsystem capabilities.

#### 5.3.5.1 Command Parameters

The command layout is described in Figure 47. The command parameters are described in Figure 48. The data structure returned by the Identify command is defined in Figure 49. The data structure specifies parameters in words. The least significant word of the parameter corresponds to the first word. See section 1.8 for more information on representation of byte, word and Dword values.

**Figure 47: Identify – Command Parameters**

| Field | 31 | 23 | 15 | 7 0 |
|---|---|---|---|---|
| ***Command*** | Timeout | | *Reserved* | Opcode: 12h |
| ***Address Low*** | *Reserved* | | | |
| ***Address High*** | *Reserved* | | | |
| ***Transfer Count*** | *Reserved* | | | |
| ***Parameters*** | *Reserved* | | | |
| ***Attributes*** | *Reserved* | | | |

**Figure 48: Identify – Parameter Description**

| Parameter | Field | Bits | Description |
|---|---|---|---|
| Opcode | Command | 07:00 | The opcode for Identify is 12h. |
| Timeout | Command | 31:16 | The relative time from command issue until the command should be completed by the NVM device. This field is in 10 millisecond units. The minimum value is 10 (corresponding to 100 milliseconds). |

**Figure 49: Identify – Data Structure**

| Word | O/M | Description |
|---|---|---|
| 0-9 | M | Serial Number (20 ASCII characters) |
| 10-13 | M | Firmware Revision (8 ASCII characters) |
| 14-33 | M | Model Number (40 ASCII characters) |
| 34-35 | O | Manufacture Code |
| 36 | M | Sector Size |
| 37 | M | NVM Page Size |
| 38 | M | Optimal Write Transfer Size |
| 39 | M | Metadata Size |
| 40 | M | Allocation Unit |
| 41-42 | M | Capacity |
| 43 | M | Capabilities<br>15 Asynchronous status notification supported<br>14 Partial page metadata operations supported<br>13-0 Reserved |
| 44 | M | Sector Alignment |
| 45 | M | Maximum Burst Speed |
| 46 | M | Recommended Minimum Number of Writes |
| 47 | M | Recommended Minimum Number of Reads |
| 48 | M | Dataset Management Support |
| 49-127 | M | Reserved |
| 128-255 | M | Vendor Specific |

O/M: O = Optional, M = Mandatory

#### 5.3.5.1.1 Word 0-9: Serial Number

Contains the serial number for the NVM device. See section 1.7 for ASCII string requirements.

#### 5.3.5.1.2 Word 10-13: Firmware Revision

Contains the firmware revision for the NVM subsystem. See section 1.7 for ASCII string requirements.

#### 5.3.5.1.3 Word 14-33: Model Number

Contains the model number for the NVM subsystem. See section 1.7 for ASCII string requirements.

#### 5.3.5.1.4 Word 34-35: Manufacture Code

Contains information about the date when the module was manufactured. This field is optional. If not supported, then the field shall be cleared to 00h.

**Bits 7-0:** Module revision ID (vendor specific format)

**Bits 15-8**: Day of module manufacture in binary coded decimal

**Bits 23-16**: Month of module manufacture in binary coded decimal

**Bits 31-24**: Year of module manufacture in binary coded decimal

#### 5.3.5.1.5 Word 36: Sector Size

The sector size is the smallest unit of data that may be read from or written to the NVM device. The value in this field is reported in terms of a power of two. For instance, a reported value of 14 corresponds to a sector size of 2^14 or 16384 bytes. A value smaller than 9 (i.e. 512 bytes) is not supported.

#### 5.3.5.1.6 Word 37: NVM Page Size

The NVM page size is the recommended minimum write granularity. This value in this field is reported in terms of sectors.

#### 5.3.5.1.7 Word 38: Optimal Write Transfer Size

The optimal write transfer size is the number of NVM pages that should be transferred as a unit to yield the highest throughput. Multiple units may be transferred as part of one Write command.

#### 5.3.5.1.8 Word 39: Metadata Size

The metadata size indicates the number of bytes of metadata that is provided per NVM page for host use. If this field is ‘0’, then metadata is not supported and host software shall not set the CH[z].DT[0] bit to ‘1’. When metadata is transferred as part of a Read or Write command, the amount of metadata transferred per NVM page shall be equal to the size reported in this field.

##### 5.3.5.1.8.1 Recommended Minimum Metadata Size (Informative)

If an NVM device supports metadata, then there are recommended minimum amounts of metadata to provide for some NVM page sizes. Adherence to the recommendations contained in this section has no effect on the compliance of an NVM device with this specification: it is valid for a compliant NVM device to report a metadata size of zero bytes, or less than the minimum size recommended in this section.

The recommendations for 4KB and 8KB NVM page sizes are listed below. No recommendations are provided for other NVM page sizes (e.g. 1KB and 2KB), however these NVM page sizes do benefit from the NVM subsystem providing metadata where possible.

| NVM Page Size | Recommended Minimum Metadata Size (if supported) |
|---|---|
| 4KB | 14 bytes |
| 8KB | 32 bytes |

#### 5.3.5.1.9 Word 40: Allocation Unit

The allocation unit is the recommended minimum number of NVM pages that should be allocated by host software for file or cluster allocations. File and/or cluster allocations should be done in allocation unit multiples.

#### 5.3.5.1.10 Word 41-42: Capacity

This field defines the total number of NVM pages present in the NVM device. This value shall be the total user storage capacity of the NVM device in bytes divided by the NVM page size in bytes.

#### 5.3.5.1.11 Word 43: Capabilities

The Capabilities field indicates the capabilities of the NVM subsystem.

**Bits 0-13:** Reserved

**Bit 14:** Partial page metadata operations supported. When set to one indicates that the NVM subsystem supports Write commands that are not a multiple of the NVM page size and may start at an arbitrary sector with metadata provided. When cleared to zero indicates that all Write commands with metadata shall be a multiple of the NVM page size and the starting sector shall be at the beginning of an NVM page.

**Bit 15:** Asynchronous status notification supported. When set to one indicates that the NVM subsystem is able to generate an asynchronous interrupt to inform the host that the status of the NVM subsystem should be checked using the Get Status command. When cleared to zero, the NVM subsystem does not generate asynchronous interrupts.

#### 5.3.5.1.12 Word 44: Sector Alignment

This field defines the physical offset in sectors within the NVM page where the first sector is placed.

For example, assume there are eight sectors per NVM page reported in the Identify data structure, see section 5.3.5.1.6. If the value in this field is 3h, then the first sector of the first aligned NVM page starts at sector address 3 and the first sector of the second aligned NVM page starts at sector address 11. Sector addresses 0-2 may be used, but would not be part of an aligned NVM page access.

#### 5.3.5.1.13 Word 45: Maximum Burst Speed

This field reports the maximum burst speed supported by the controller in combination with the attached NVM device. The value is reported in 10 MB/s units. For example, a value of 6 in this field corresponds to a maximum burst speed of 60 MB/s. The NVM device may not achieve this level of performance depending on the command sequence issued.

#### 5.3.5.1.14 Word 46: Recommended Minimum Number of Writes

This field reports the recommended minimum number of NVM page writes that should be issued without other intervening commands for good performance.

#### 5.3.5.1.15 Word 47: Recommended Minimum Number of Reads

This field reports the recommended minimum number of NVM page reads that should be issued without other intervening commands for good performance.

#### 5.3.5.1.16 Word 48: Dataset Management Support

The Dataset Management Support field indicates whether the NVM subsystem supports dataset management and the particular context attributes supported. If dataset management is supported, then the NVM subsystem shall support at least one set of context attributes.

**Bit 0:** If set to ‘1’, then the NVM subsystem supports the Dataset Management command and the Attributes field in Read and Write commands is supported. If cleared to ‘0’, then the NVM subsystem does not support the Dataset Management command and the Attributes field in Read and Write commands shall be cleared to 0h by host software.

**Bit 1:** If set to ‘1’, then the NVM subsystem supports the Read Frequency, Write Frequency, Read Latency, and Write Latency context attributes. If cleared to ‘0’, then the NVM subsystem does not support the Read Frequency, Write Frequency, Read Latency, and Write Latency context attributes.

**Bit 2:** If set to ‘1’, then the NVM subsystem supports the Deallocate and Write Prepare context attributes. If cleared to ‘0’, then the NVM subsystem does not support the Deallocate and Write Prepare context attributes.

**Bit 3:** If set to ‘1’, then the NVM subsystem supports the Atomic Read, Atomic Write, and Command Access Size context attributes. If cleared to ‘0’, then the NVM subsystem does not support the Atomic Read, Atomic Write, and Command Access Size context attributes. The Atomic Read and Atomic Write support applies to both global and per range attributes.

**Bits 4-15:** Reserved

#### 5.3.5.2 Command Status

This section describes the Command Specific Status field for the Identify command. Figure 50 describes the Command Specific Status when the command was successful (CH[z].CS.OCS.FE = ‘0’). Figure 51 describes the Command Specific Status when the command had a fatal error (CH[z].CS.OCS.FE = ‘1’).

**Figure 50: Identify – Command Specific Status, Success**

| Bits | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Command Specific Status | Reserved | | | | | | | |

**Figure 51: Identify – Command Specific Status, Fatal Error**

| Bits | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Command Specific Status | Reserved | | | | | RETR | R | UNC |

UNC: Shall be set to ‘1’ if the data structure could not be returned. This could be due to an NVM media error, a bus transfer error, or insufficient error correction capability.

RETR: Shall be set to ‘1’ if the NVM subsystem requests that host software retry the request.

### 5.3.6 Read

The Read command returns data and/or metadata to the host for the sectors indicated. Whether data and/or metadata is returned is indicated in the CH[z].DT field.

#### 5.3.6.1 Command Parameters

The command layout is described in Figure 52. The command parameters are described in Figure 53.

**Figure 52: Read – Command Parameters**

| Field | 31 | 23 | 15 | 7 | 0 |
|---|---|---|---|---|---|
| ***Command*** | Timeout | | *Reserved* W A T | Opcode: 10h | |
| ***Address Low*** | Starting Sector Address, lower 32-bits | | | | |
| ***Address High*** | Starting Sector Address, upper 32-bits | | | | |
| ***Transfer Count*** | *Reserved* | | Number of Sectors | | |
| ***Parameters*** | *Reserved* | | Avg WS | UW | UR PR |
| ***Attributes*** | Command Attributes | | | | |

**Figure 53: Read – Parameter Description**

| Parameter | Field | Bits | Description |
|---|---|---|---|
| Opcode | Command | 07:00 | The opcode for Read is 10h. |
| T | Command | 08 | If set to ‘1’, then this command is a retry (a previous Read of these sectors failed). If cleared to ‘0’, this command is not a retry. |
| A | Command | 09 | If set to ‘1’, then the Attributes are present and valid. This bit shall only be set to ‘1’ for NVM page aligned and NVM page granularity requests. |
| W | Command | 10 | If set to ‘1’, then the workload information specified in the Parameters field is valid. If cleared to ‘0’, then the workload information in the Parameters field is undefined. |
| Timeout | Command | 31:16 | The relative time from command issue until the command should be completed by the NVM device. This field is in 10 millisecond units. The minimum value is 10 (corresponding to 100 milliseconds). |
| Starting Sector Address | Address Low<br>Address High | 31:00<br>31:00 | The first sector to be read as part of the read transfer. |
| Number of Sectors | Transfer Count | 15:00 | The number of sectors to transfer. A value of 0h represents 0 sectors. The maximum transfer is 65535 sectors. |
| PR: Priority | Parameters | 01:00 | Indicates the priority of the request:<br>00b: Low priority<br>01b: Normal priority<br>10b: High priority<br>11b: Critical priority<br>The NVM subsystem may use this information to help determine the command to service next. |
| UR: Upcoming Reads | Parameters | 03:02 | Indicates the number of queued read requests that are yet to be issued by host software. These requests may be for any sector location.<br>00b: 0 requests<br>01b: 1 or more requests<br>10b-11b: Reserved |
| UW: Upcoming Writes | Parameters | 07:04 | Indicates the number of queued write requests that are yet to be issued by host software. These requests may be for any sector location.<br>00h: 0 requests<br>01h: 1 request<br>02h: 2 requests<br>03h: 3 to 5 requests<br>04h: 6 to 10 requests<br>05h: 11 to 20 requests<br>06h: Over 20 requests<br>07h – 0Fh: Reserved |
| Avg WS: Average Write Size | Parameters | 15:08 | This field indicates the average size in number of NVM pages for the queued write requests that are yet to be issued by host software. |
| Command Attributes | Attributes | 31:00 | Contains bits 31:00 of the Context Attributes defined in section 5.3.1.3. This field is only valid if A in the Command parameter is set to ‘1’. |

#### 5.3.6.2 Command Status

This section describes the Command Specific Status field for the Read command. Figure 54 describes the Command Specific Status when the command was successful (CH[z].CS.OCS.FE = ‘0’). Figure 55 describes the Command Specific Status when the command had a fatal error (CH[z].CS.OCS.FE = ‘1’).

**Figure 54: Read – Command Specific Status, Success**

| Bits | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Command Specific Status | Reserved | | | | RCVR | DEAL | ATTR | UNAL |

UNAL: Shall be set to '1' if the starting sector address was not NVM page aligned. This bit is informative.

ATTR: Shall be set to '1' if any of the following conditions hold: 1) conflicting attribute settings were provided, 2) attributes were provided for an unaligned NVM Page request, 3) attributes were provided in a granularity not equal to the NVM page size. When this bit is set to '1', the attribute settings provided may not have been accepted by the NVM device.

DEAL: Shall be set to '1' if any of the sectors are currently deallocated.

RCVR: Shall be set to '1' if the data returned has undergone significant recovery actions in order to return valid data. This could be due to the amount of correction applied to the data nearing the corrector's maximum capability.

**Figure 55: Read – Command Specific Status, Fatal Error**

| Bits | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Command Specific Status | Reserved | | | | OOR | RETR | R | UNC |

UNC: Shall be set to '1' if the data requested could not be returned. This could be due to an NVM media error, a bus transfer error, or insufficient error correction capability.

RETR: Shall be set to '1' if the NVM subsystem requests that host software retry the request.

OOR: Shall be set to '1' if any of the sectors requested are not in the user accessible address range provided in the Identify command.

### 5.3.7 Set Features

The Set Features command is used to set parameters for features that the NVM subsystem supports.

#### 5.3.7.1 Command Parameters

The command layout is described in Figure 56. The command parameters are described in Figure 57.

**Figure 56: Set Features – Command Parameters**

| Field | 31 | 23 | 15 | 7 0 |
|---|---|---|---|---|
| ***Command*** | Timeout | | *Reserved* | Opcode: 21h |
| ***Address Low*** | *Reserved* | | | Feature |
| ***Address High*** | *Reserved* | | | |
| ***Transfer Count*** | *Reserved* | | | # of Dwords |
| ***Parameters*** | *Reserved* | | | |
| ***Attributes*** | *Reserved* | | | |

Figure 57: Set Features – Parameter Description

| Parameter | Field | Bits | Description |
|---|---|---|---|
| Opcode | Command | 07:00 | The opcode for Set Features is 21h. |
| Timeout | Command | 31:16 | The relative time from command issue until the command should be completed by the NVM device. This field is in 10 millisecond units. The minimum value is 10 (corresponding to 100 milliseconds). |
| Feature | Address Low | 07:00 | The Feature to transfer settings for. |
| # of Dwords | Transfer Count | 07:00 | The number of Dwords to transfer. The maximum transfer is 255 Dwords. The number of Dwords to transfer is based on the Feature being set and its associated data buffer transfer size. |

#### 5.3.7.2 Command Status

This section describes the Command Specific Status field for the Set Features command. Figure 58 describes the Command Specific Status when the command was successful (CH[z].CS.OCS.FE = ‘0’). Figure 59 describes the Command Specific Status when the command had a fatal error (CH[z].CS.OCS.FE = ‘1’).

Figure 58: Set Features – Command Specific Status, Success

| Bits | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Command Specific Status | Reserved | | | | | | | |

Figure 59: Set Features – Command Specific Status, Fatal Error

| Bits | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Command Specific Status | Reserved | | | | | | PARM | ADDR |

ADDR: Shall be set to ‘1’ if the feature address specified is not supported by the NVM subsystem.

PARM: Shall be set to ‘1’ if the parameters provided are malformed.

#### 5.3.7.3 Feature Definitions

Figure 60 defines the features that may be supported in an implementation. These same features may be queried with Get Features.

Figure 60: Set Features – Feature Addresses

| Feature Address | O/M | Description |
|---|---|---|
| 00h | | Reserved |
| 01h | M | Linear Command Processing |
| 02h | M | Power Management |
| 03h | M | NVM Device Layout |
| 04h | M | Boot Cache |
| 05h – 7Fh | | Reserved |
| 80h – FFh | | Vendor specific |

O/M: O = Optional, M = Mandatory

#### 5.3.7.3.1 Linear Command Processing (01h)

The NVMHCI subsystem is allowed to process commands out of order. However, there may be times where host software needs the NVMHCI subsystem to process commands in order. To accomplish this, host software may request that the NVMHCI subsystem process requests in linear order (process from 0, 1, 2, … 31, and then wrap around to 0). Then if software issues commands in a linear order to the PxCI register, the commands will be completed in the order that they are issued. The controller begins checking for new commands starting from the last PxCI location that it executed from. Thus, if the last command executed was command slot 0, and the host issues command slots 1-4 at the same time then command slot 1 is executed next.

Note: There may be performance and/or NVM device lifetime degradations due to use of linear command processing. It is recommended that host software allow the NVMHCI subsystem to process commands out of order whenever possible.

Figure 61 defines the data structure constructed by host software in Set Features or by the NVM subsystem when Get Features is used. The size of the data structure is one Dword.

**Figure 61: Feature – Linear Command Processing data structure**

| Field | Bits | Description |
|---|---|---|
| Process Linearly | 00 | If set to ‘1’, then the NVMHCI subsystem shall process commands in linear order. If cleared to ‘0’ then the NVMHCI subsystem may process commands out of order. By default this value is cleared to ‘0’. |
| Reserved | 31:01 | Reserved |

#### 5.3.7.3.2 Power Management (02h)

Host software may indicate power management settings for the platform. Figure 62 defines the data structure constructed by host software in Set Features or by the NVM subsystem when Get Features is used. The size of the data structure is one Dword.

**Figure 62: Feature – Power Management data structure**

| Field | Bits | Description |
|---|---|---|
| Power Saving | 02:00 | This field is a sliding indication of the level of power saving requested. 00h is no power saving requested. 07h is maximum power savings requested. This is a sliding linear scale. The default value is vendor specific. |
| Performance | 05:03 | This field is a sliding indication of the level of performance requested. 00h is minimal performance requested. 07h is maximum performance requested. The default value is vendor specific. |
| AC/DC | 06 | If set to ‘1’ indicates that the host is running on AC power. If cleared to ‘0’ indicates that the host is running on DC power. This setting is required to be valid. This field is invalid (not used) when retrieved with Get Features. |
| Reserved | 31:07 | Reserved |

#### 5.3.7.3.3 NVM Device Layout (03h)

Host software will need to know how the NVM device has been previously laid out to avoid the loss of persistent data. There may be situations where the currently running host software is not the host software that setup the device initially (driver update, BIOS, etc). Under these conditions the currently running host software should adhere to the previous setup to allow users to continue to access their data. Setting the NVM Device Layout allows for the current host software to indicate how the device has been laid out to future/other host software. When host software allocates a portion of the NVM device for a particular use, it should create a corresponding Area entry in the NVM Device Layout structure.

Figure 63 defines how the NVM device layout is constructed by host software in Set Features or by the NVM subsystem when Get Features is used. A maximum of 16 areas is allowed. The host shall pack the areas; i.e. if there are n areas then they shall be provided in Area 0 through Area n-1. Invalid areas are marked as “Invalid Area Type” in the Area Type.

The Creator Signature should include a unique identifier that indicates the creator of that Area. A value of 0h for the Creator Signature indicates that the field is not used and should be ignored by software. The Creator Signature is unique to a particular company. It is recommended that companies review their selected Creator Signature with the NVMHCI Workgroup to ensure collisions are avoided.

The Area Layout defines the layout and structure of the Area Type. The Area Layout field is used only for “Cache Usage” Area Types. A value of 0h for the Area Layout indicates that the field is not used and should be ignored by software; this value should be used for Area Types other than “Cache Usage”. Area Layouts 0 – 191 are reserved for publicly defined layouts allocated by the NVMHCI Workgroup. Publicly defined layouts are published on the NVMHCI website. Area Layouts 192 – 255 are for vendor specific use.

**Figure 63: Feature – NVM Device Layout data structure**

| Area | Byte | Field |
|---|---|---|
| | | |
| Area 0 | 00 | Area Type |
| | 01 | Area Layout |
| | 07:02 | Length in NVM pages |
| | 15:08 | Starting sector address |
| | 19:16 | Creator Signature |
| | 31:20 | Reserved |
| | | |
| Area 1 | 32 | Area Type |
| | 33 | Area Layout |
| | 39:34 | Length in NVM pages |
| | 47:40 | Starting sector address |
| | 51:48 | Creator Signature |
| | 63:52 | Reserved |
| … | | |

Figure 64 defines the various types of areas that may be specified by the host. Host software should not modify an area if it does not understand the area type or if that area is vendor specific.

**Figure 64: Feature – Area Types**

| Value | Description |
|---|---|
| 00h | Persistent Data Usage |
| 01h | Cache Usage |
| 02-79h | Reserved |
| 80h-FEh | Vendor specific |
| FFh | Invalid Area Type |

#### 5.3.7.3.4 Boot Cache (04h)

The NVMHCI controller may discover and use any cache area within the NVMHCI controller whose contents it recognizes in order to help optimize operations to boot the system. However, there may be times when no cached areas should be used during boot.

Figure 65 defines the data structure constructed by host software in Set Features or by the NVM subsystem when Get Features is used. The size of the data structure is one Dword.

**Figure 65: Feature – Boot Cache data structure**

| Field | Bits | Description |
|---|---|---|
| Boot Cache Enable/Disable | 00 | If set to ‘1’, then the BIOS/OROM is allowed to read any NVMHCI “Cache Usage” area type for this boot session. If cleared to ‘0’, then the BIOS/OROM shall not access any NVMHCI “Cache Usage” area type for this boot session. |
| Reserved | 31:01 | Reserved |

### 5.3.7.4 Feature Persistence Across Device Power States

The feature parameters that may be specified with the Set Features command have specific persistence requirements with regards to power state transitions:

- Process Linearly: This setting shall be retained across D3hot->D0initialized device state transitions. The NVMHCI subsystem resets this feature to its default value upon D3cold->D0initialized state transitions.
- Power Management: These settings shall be retained across D3hot->D0initialized device state transitions. The NVMHCI subsystem resets this feature to its default value upon D3cold->D0initialized state transitions. Note that the host is responsible for specifying the correct AC/DC setting whenever there is a transition.
- NVM Device Layout: These settings shall persist across all Dx state transitions.
- Boot Cache: These settings shall persist across all Dx state transitions.

### 5.3.8 Write

The Write command writes data and/or metadata to the NVM device for the sectors indicated. Whether data and/or metadata is written is indicated in the CH[z].DT field.

If a Write command is not NVM page aligned and/or not in an NVM page size granularity (regardless of whether metadata is written or not), the controller shall merge in data previously written to other sectors within that NVM page. If metadata is written as part of the command, any previous metadata for the affected NVM page(s) is overwritten. If metadata is not written, then any previous metadata written for the NVM page is indeterminate. It is host software's responsibility to ensure metadata is preserved as needed for a particular NVM page.

### 5.3.8.1 Command Parameters

The command layout is described in Figure 66. The command parameters are described in Figure 67.

**Figure 66: Write – Command Parameters**

<table>
<tr><th>Field</th><th>31</th><th>23</th><th colspan="4">15</th><th colspan="2">7</th><th>0</th></tr>
<tr><td><b><i>Command</i></b></td><td colspan="2">Timeout</td><td><i>Reserved</i></td><td>W</td><td>A</td><td>T</td><td colspan="3">Opcode: 20h</td></tr>
<tr><td><b><i>Address Low</i></b></td><td colspan="9">Starting Sector Address, lower 32-bits</td></tr>
<tr><td><b><i>Address High</i></b></td><td colspan="9">Starting Sector Address, upper 32-bits</td></tr>
<tr><td><b><i>Transfer Count</i></b></td><td colspan="2"><i>Reserved</i></td><td colspan="7">Number of Sectors</td></tr>
<tr><td><b><i>Parameters</i></b></td><td colspan="2"><i>Reserved</i></td><td colspan="4">Avg WS</td><td>UW</td><td>UR</td><td>PR</td></tr>
<tr><td><b><i>Attributes</i></b></td><td colspan="9">Command Attributes</td></tr>
</table>

**Figure 67: Write – Parameter Description**

| Parameter | Field | Bits | Description |
|---|---|---|---|
| Opcode | Command | 07:00 | The opcode for Write is 20h. |
| T | Command | 08 | If set to '1', then this command is a retry (a previous Write of these sectors failed). If cleared to '0', this command is not a retry. |
| A | Command | 09 | If set to '1', then the Attributes are present and valid. This bit shall only be set to '1' for NVM page aligned and NVM page granularity requests. |
| W | Command | 10 | If set to '1', then the workload information specified in the Parameters field is valid. If cleared to '0', then the workload information in the Parameters field is undefined. |
| Timeout | Command | 31:16 | The relative time from command issue until the command should be completed by the NVM device. This field is in 10 millisecond units. The minimum value is 10 (corresponding to 100 milliseconds). |
| Starting Sector Address | Address Low<br>Address High | 31:00<br>31:00 | The first sector to be written as part of the write transfer. |
| Number of Sectors | Transfer Count | 15:00 | The number of sectors to transfer. A value of 0h represents 0 sectors. The maximum transfer is 65535 sectors. |
| PR: Priority | Parameters | 01:00 | Indicates the priority of the request:<br>00b: Low priority<br>01b: Normal priority<br>10b: High priority<br>11b: Critical priority<br>The NVM subsystem may use this information to help determine the command to service next. |
| UR: Upcoming Reads | Parameters | 03:02 | Indicates the number of queued read requests that are yet to be issued by host software. These requests may be for any sector location.<br>00b: 0 requests<br>01b: 1 or more requests<br>10b-11b: Reserved |
| UW: Upcoming Writes | Parameters | 07:04 | Indicates the number of queued write requests that are yet to be issued by host software. These requests may be for any sector location.<br>00h: 0 requests<br>01h: 1 request<br>02h: 2 requests<br>03h: 3 to 5 requests<br>04h: 6 to 10 requests<br>05h: 11 to 20 requests<br>06h: Over 20 requests<br>07h – 0Fh: Reserved |
| Avg WS: Average Write Size | Parameters | 15:08 | This field indicates the average size in number of NVM pages for the queued write requests that are yet to be issued by host software. |
| Command Attributes | Attributes | 31:00 | Contains bits 31:00 of the Context Attributes defined in section 5.3.1.3. This field is only valid if A in the Command parameter is set to '1'. |

#### 5.3.8.2 Command Status

This section describes the Command Specific Status field for the Write command. Figure 54 describes the Command Specific Status when the command was successful (CH[z].CS.OCS.FE = '0'). Figure 55 describes the Command Specific Status when the command had a fatal error (CH[z].CS.OCS.FE = '1').

If a Write command experiences a fatal error (CH[z].CS.OCS.FE = ‘1’), the data in the sectors and metadata to be written have indeterminate values. Host software is responsible for taking appropriate retry measures to place the affected data and metadata into a known good state.

**Figure 68: Write – Command Specific Status, Success**

| **Bits** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
|---|---|---|---|---|---|---|---|---|
| Command Specific Status | Reserved | | | | | | ATTR | UNAL |

UNAL: Shall be set to ‘1’ if the starting sector address was not NVM page aligned. This bit is informative.

ATTR: Shall be set to ‘1’ if any of the following conditions hold: 1) conflicting attribute settings were provided, 2) attributes were provided for an unaligned NVM Page request, 3) attributes were provided in a granularity not equal to the NVM page size. When this bit is set to ‘1’, the attribute settings provided may not have been accepted by the NVM device.

**Figure 69: Write – Command Specific Status, Fatal Error**

| **Bits** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
|---|---|---|---|---|---|---|---|---|
| Command Specific Status | Reserved | | | | OOR | RETR | R | NME |

NME: Shall be set to ‘1’ if the data could not be written to the non-volatile memory due to media errors.

RETR: Shall be set to ‘1’ if the NVM subsystem requests that host software retry the request.

OOR: Shall be set to ‘1’ if any of the sectors requested are not in the user accessible address range provided in the Identify command.

# 6 Data Transfer Operation

## 6.1 Introduction

Host software presents a list of commands to the controller for the NVMHCI port, which then processes it. For controllers that have a command list depth of one, this is a single step operation, and software only presents a single command. For controllers that support a command list depth greater than one command, multiple commands may be posted to the command list.

Host software posts new commands received by the OS to empty slots in the list, and sets the corresponding slot bit in the PxCI register. Host software is permitted to set multiple PxCI bits in one write operation. The controller continuously looks at the PxCI to determine if there are commands to transmit to the NVM subsystem.

The NVM subsystem processes the command list in a slot number linear fashion if the linear command processing feature in section 5.3.7.3.1 is enabled.

The NVM subsystem may process the command list out of order if the linear command processing feature in section 5.3.7.3.1 is disabled. The NVM subsystem shall ensure that starvation is avoided for all commands issued by host software. Host software shall not place commands in the list that may not be re-ordered arbitrarily. Data may not be committed to the NVM device in the order that commands are received.

## 6.2 System Software Rules (Normative)

### 6.2.1 Basic Steps when Building a Command

When host software builds a command for the controller to execute, it first finds an empty command slot by reading the PxCI for the port. An empty command slot has its respective bit cleared to '0' in the PxCI register. After a free slot (slot pFreeSlot), is found:

1. Host software builds a command header at PxCLB[CH(pFreeSlot)] with:
   a. PRDTL containing the number of entries in the PRD table
   b. CL set to the length of the command
   c. W (Write) bit set if data is going to the NVM device
   d. DT[0] is set to '1' if metadata is going to be transferred
   e. I (interrupt) bit set if software requests PxIS.CCS to be set on command completion
   f. PITO shall be filled in with the offset to the beginning of the PRD Index Table
   g. MO shall be filled in with the offset to the beginning of the Metadata Region, if DT[0] is set to '1'
   h. CTBA and CTBAU shall be filled with the Command Table Base Address/Upper
2. Host software shall fill in the Command Table (as pointed to by CTBA and CTBAU)
3. Host software shall set PxCI.CI(pFreeSlot) to indicate to the HBA that a command is active. Host software should only write new bits to set to '1'; the previous register content of PxCI should not be re-written in the register write.

### 6.2.2 Processing Completed Commands

Host software processes the interrupt generated by the NVM subsystem for command completion. In the interrupt service routine, host software checks IS.IPS (in AHCI) to determine if the NVMHCI port has an interrupt pending.

If the NVMHCI port has an interrupt pending:

1. Host software determines the cause of the interrupt by reading the PxIS register. It is possible for multiple bits to be set.
2. Host software clears appropriate bits in the PxIS register corresponding to the cause of the interrupt.
3. Host software clears the interrupt bit in IS.IPS corresponding to the port.
4. Host software reads the PxCI register, and compares the current value to the list of commands previously issued by host software that are still outstanding. Host software completes with success any outstanding command whose corresponding bit

has been cleared in the respective register. PxCI is a volatile register; software should only use its value to determine commands that have completed, not to determine which commands have previously been issued.

5. If there were errors, noted in the PxIS register, host software performs error recovery actions (see section 7.2.1).

### 6.2.3 Data Transfer

#### 6.2.3.1 Read (with metadata)

Host software builds a command as described in section 6.2.1. The command shall have a PRD table, a PRD Index Table, and a Metadata Region. This is a read operation with metadata, therefore CH(z).W (Write) shall be cleared to ‘0’ and CH(z).DT shall be set to 11b.

#### 6.2.3.2 Write (with metadata)

Host software builds a command as described in section 6.2.1. The command shall have a PRD table, a PRD Index Table, and a Metadata Region. This is a write operation with metadata, therefore CH(z).W (Write) shall be set to ‘1’ and CH(z).DT shall be set to 11b.

#### 6.2.3.3 Read (without metadata)

Host software builds a command as described in section 6.2.1. The command shall have a PRD table and a PRD Index Table. This is a read operation without metadata, therefore CH(z).W (Write) shall be cleared to ‘0’ and CH(z).DT shall be set to 10b.

#### 6.2.3.4 Write (without metadata)

Host software builds a command as described in section 6.2.1. The command shall have a PRD table and a PRD Index Table. This is a write operation without metadata, therefore CH(z).W (Write) shall be set to ‘1’ and CH(z).DT shall be set to 10b.

### 6.2.4 Software Examples (with PRD index fill out)

During command generation the host shall fill in the PRD Index Table to enable out of order data transfer to the NVM device by the NVM subsystem. This structure avoids hardware walking the PRD table entries themselves to determine where the data transfer begins for a particular NVM page. Below is an example of host software filling in the PRD Index Table.

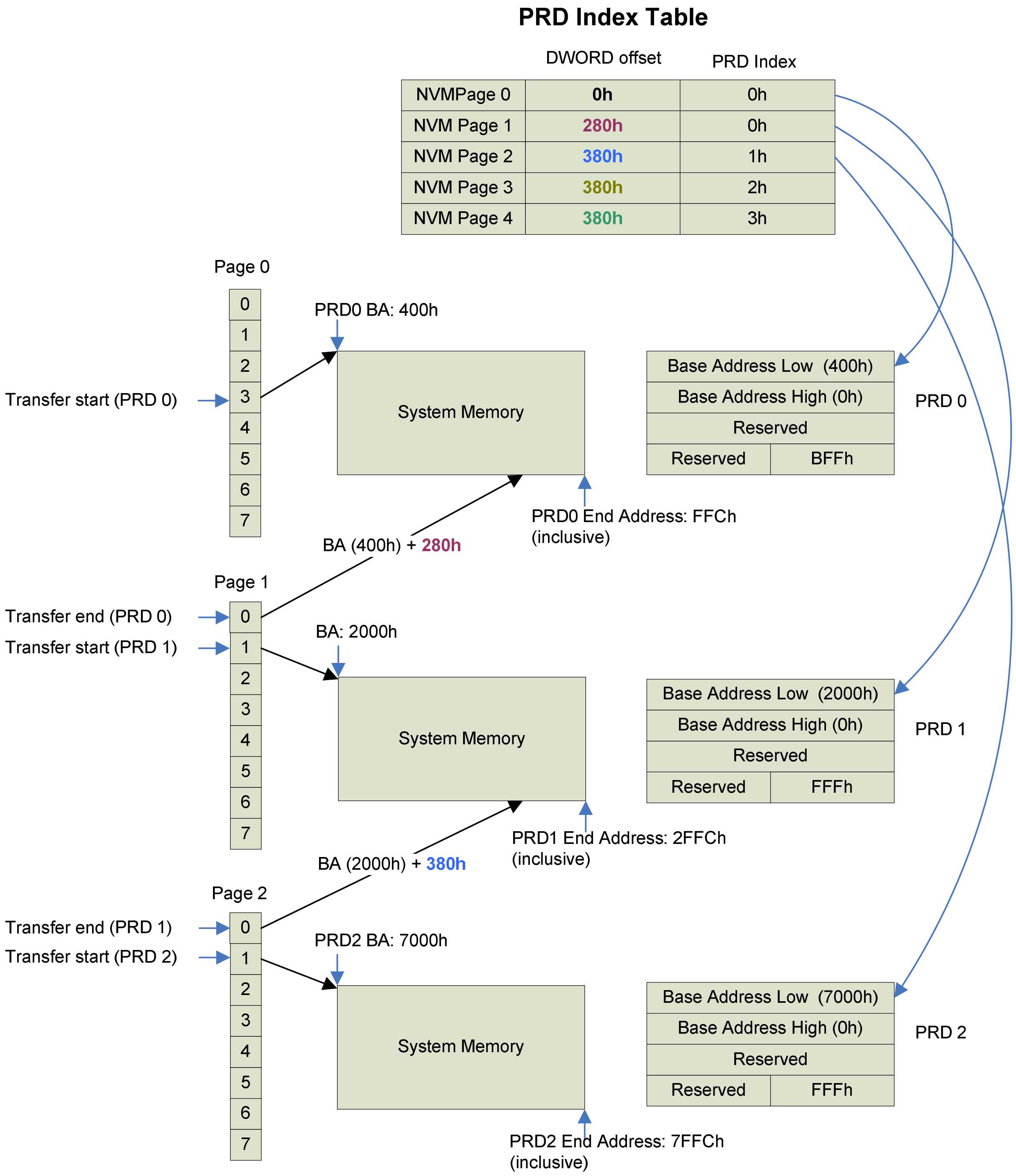
In the example, system software places a Read command for 16KB of data, starting at sector 3, in slot 2. The attributes of this command are:

- The Read includes a metadata transfer
- The NVM page size for the NVM device is 4KB
- The sector size is 512 bytes
- The Metadata Offset location in the Command Table is at offset 120h
- The size of the metadata per NVM page is 12 bytes
- The NVM subsystem has decided to transfer the data in the following order: 3rd NVM page, 1st NVM page, 2nd NVM page, 4th NVM page, and 5th NVM page

To accomplish this request, software builds a command in slot 2. The command shall have a PRD Table, an opcode of Read (10h), an Address Low of 3h, and a Transfer Count of 20h. CH(2).W (Write) shall be cleared to ‘0’, CH(2).I shall be set to ‘1’, and CH(2).CL shall be set to 5h. CH(2).DT shall be set to 11b to indicate that this is a data and metadata transfer. CH(2).MO shall be set to 120h.

Figure 70 and Figure 71 describe the PRD Index Table and example associated PRD entries. PRD entries describe the data buffer allocated by host software being distributed in five distinct physical system memory address ranges.

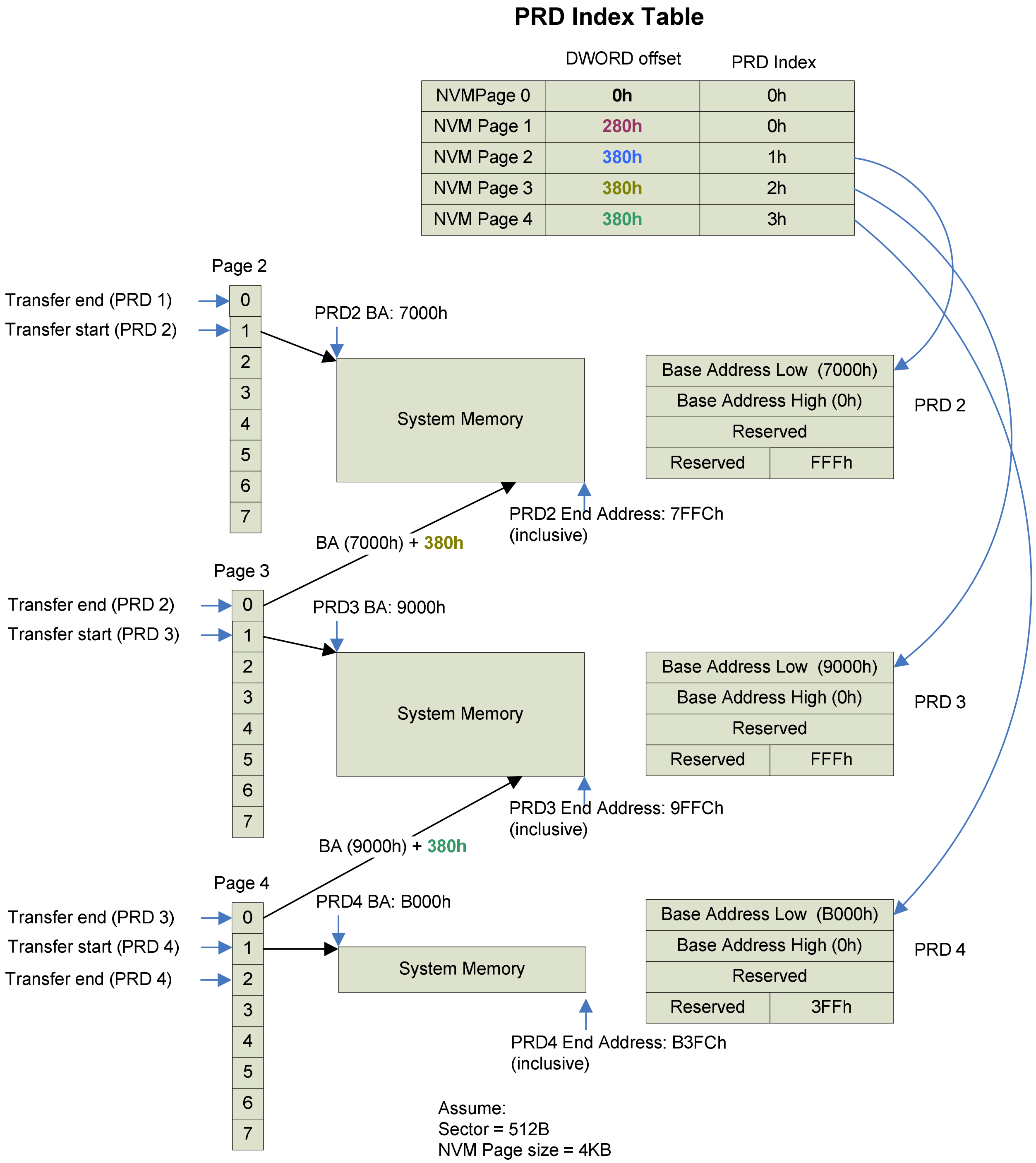
**Figure 70: PRD Index Table example**



PRD Index Table

| | DWORD offset | PRD Index |
|---|---|---|
| NVMPage 0 | **0h** | 0h |
| NVM Page 1 | **280h** | 0h |
| NVM Page 2 | **380h** | 1h |
| NVM Page 3 | **380h** | 2h |
| NVM Page 4 | **380h** | 3h |

**Figure 71: PRD Index Table example, continued**



## PRD Index Table

| | DWORD offset | PRD Index |
|---|---|---|
| NVMPage 0 | **0h** | 0h |
| NVM Page 1 | **280h** | 0h |
| NVM Page 2 | **380h** | 1h |
| NVM Page 3 | **380h** | 2h |
| NVM Page 4 | **380h** | 3h |

Assume:
Sector = 512B
NVM Page size = 4KB

As the controller traverses the PRD Index Table and reads data from the NVM device, it shall maintain ordering of the metadata corresponding to the NVM page being read. In this case the NVM subsystem has decided to transfer the data in the following order: 3rd NVM page, 1st NVM page, 2nd NVM page, 4th NVM Page, and 5th NVM page. Therefore, in this example, the controller writes the metadata to system memory as follows:

1. Metadata for NVM Page 2: (CTBA[2]/CTBAU[2]) + 120h + 18h
2. Metadata for NVM Page 0: (CTBA[2]/CTBAU[2]) + 120h + 0h
3. Metadata for NVM Page 1: (CTBA[2]/CTBAU[2]) + 120h + Ch
4. Metadata for NVM Page 3: (CTBA[2]/CTBAU[2]) + 120h + 24h
5. Metadata for NVM Page 4: (CTBA[2]/CTBAU[2]) + 120h + 30h

# 7 Error Reporting and Recovery

All errors occur within the NVMHCI port register space. There are no errors that apply to the entire host controller. There are several sources of errors that could occur during command execution. Examples of errors are:

- System Memory – Bad system memory pointers cause data fetches and stores to be lost.
- NVM Device – Data could not be stored or retrieved, etc.

## 7.1 Error Types

### 7.1.1 System Memory Errors

System memory errors such as target abort, master abort, and parity may cause the host to stop processing the currently executing command. These are serious errors that cannot be recovered from without software intervention (section 7.2.1).

A master/target abort error occurs when host software has given a pointer to the host controller that does not exist in system memory. When this occurs, the host controller aborts the command (if necessary) as described in section 7.2.1. When this is complete, the host controller sets PxIS.HBFS. If PxIE.HBFE is set, the host controller shall generate an interrupt.

A data error (such as CRC or parity in system memory), may or may not be transient. If such an error occurs, the host controller may stop execution and abort the command. When this is complete, the host controller sets PxIS.HBDS. If PxIE.HBDE is set, the host controller shall generate an interrupt.

### 7.1.2 Fatal NVM Device Errors

Fatal NVM device errors are when a command fails to complete successfully due to a fatal error condition that occurs when the NVM device is executing a command. When a fatal NVM device error occurs, the host controller shall set the PxIS.FES bit. If PxIE.FEE is set, the host controller shall generate an interrupt. The host controller shall clear the PxCI bit corresponding to the command and the CH[z].CS (Command Status) field shall indicate the reason for the NVM device error.

### 7.1.3 Status Errors

When a command completes (as indicated by PxIS.CCS), host software shall check CH[z].CS for command status. The host controller does not halt for non-fatal error conditions; software should evaluate CH[z].CS to enumerate fatal and non-fatal error conditions that may occur.

## 7.2 Error Recovery

### 7.2.1 Host Software Error Recovery

When an interrupt is generated due to an error condition, software will attempt to recover. Fatal errors (signified by the setting of PxIS.HBFS, PxIS.HBDS or PxIS.FES) will cause the host controller to halt operation. When the host controller is halted, it does not issue any new commands nor continue execution on any previously issued commands. To recover, host software needs to restart the port by clearing the PxCMD.ST bit to ‘0’ to acknowledge the error condition and then set PxCMD.ST to ‘1’ to start command execution.

For fatal errors, software shall determine which commands were not processed and either re-issue them or notify higher level software that the command failed. To detect an error that requires software recovery actions to be performed, software should check whether any of the following status bits are set on an interrupt: PxIS.HBFS, PxIS.HBDS, and PxIS.FES. If any of these bits are set, software should perform error recovery actions.

#### 7.2.1.1 Error Recovery Flow

The flow for host software to recover from an error when commands are issued is as follows:

- Reads PxCI to determine which commands are still outstanding
- Checks CH[z].CS for each command completed to determine if there was an error with that command, and if so the error condition

- Clears PxCMD.ST to ‘0’ to reset the PxCI register to 0h, waits for PxCMD.CR to clear to ‘0’ indicating that the host controller has halted
- Clears status bits in PxIS as appropriate
- Sets PxCMD.ST to ‘1’ to enable issuing new commands
- Optionally issue a command to gather information about the error, for example Get Status
- Host software then either completes the command that had the error and commands still outstanding with error to higher level software, or re-issues these commands to the host controller

# 8 Informative Appendix

## 8.1 Option ROM and EFI Information

For additional information on recommendations provided in this appendix, please refer to the following specifications:

- PCI Firmware 3.0 specification (http://www.pcisig.com)
- UEFI 2.1 specification (http://www.uefi.org)

### 8.1.1 EFI GUID

The NVMHCI revision 1.0 GUID shall be 06F1720F-6633-415E-92C5-9E6034DF6A74.

### 8.1.2 Version Information

Option ROM and EFI modules shall report their version information as a 16-bit value. The upper byte shall contain the Major Revision number and the lower byte shall contain the Minor Revision number. As an example, 0105h corresponds to a revision number of 1.05.

Option ROMs shall place the version number in the 16-bit CodeRevision field of the PCIRHeader for the ROM.

EFI Modules shall register their module in the EFI Configuration Table with the GUID listed in section 8.1.1. The associated data structure with the GUID shall contain the version information as the first 16 bits.

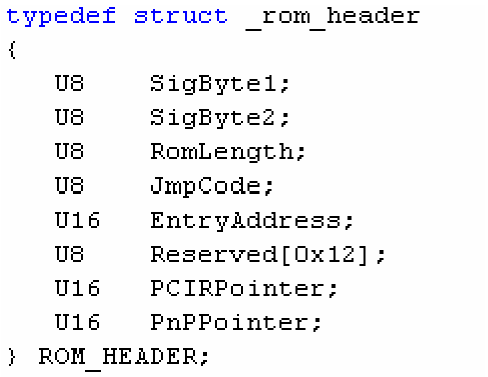
### 8.1.3 Option ROM Discovery

Option ROM (OROM) discovery is performed by walking physical memory from address 0xA000 to 0xF000 in 512 byte intervals and searching for the OROM Anchor String (0xAA55). The OROM Anchor String is the first two bytes of the ROM Header structure. If found then a valid OROM has been identified. The OROM Header structure shall point to the PCIR Header (via PCIRPointer), which contains the Base and Sub Class Codes of the device. If the NVMHCI Class Code is found then this is an NVMHCI Option ROM and the 16-bit CodeRevision shall report the version information.

If the class code is not for NVMHCI the physical address being searched should be updated by the ROMLength field in the OROM Header.

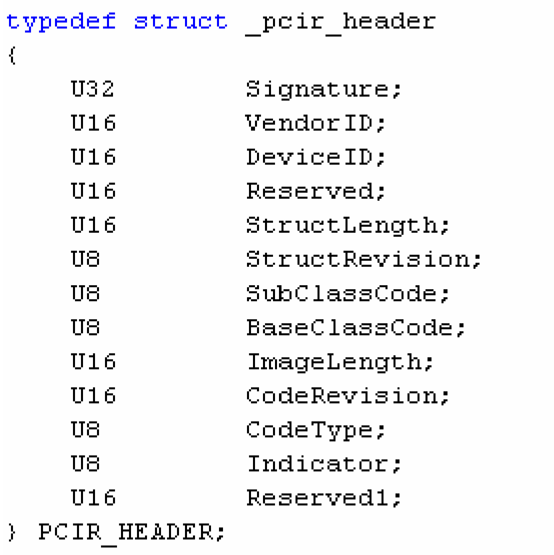
Figure 72 and Figure 73 show the ROM Header and PCIR Header data layout.

**Figure 72: ROM Header**



```
typedef struct _rom_header
{
   U8     SigByte1;
   U8     SigByte2;
   U8     RomLength;
   U8     JmpCode;
   U16    EntryAddress;
   U8     Reserved[0x12];
   U16    PCIRPointer;
   U16    PnPPointer;
} ROM_HEADER;
```

**Figure 73: PCIR Header**



```
typedef struct _pcir_header
{
    U32         Signature;
    U16         VendorID;
    U16         DeviceID;
    U16         Reserved;
    U16         StructLength;
    U8          StructRevision;
    U8          SubClassCode;
    U8          BaseClassCode;
    U16         ImageLength;
    U16         CodeRevision;
    U8          CodeType;
    U8          Indicator;
    U16         Reserved1;
} PCIR_HEADER;
```

### 8.1.4 EFI Module Discovery

EFI module discovery should be done using provided OS API calls to retrieve the EFI Configuration Table. The specific OS API calls are beyond the scope of this document.

Software should search the EFI Configuration Table for the NVMHCI GUID. If a match is found then an NVMHCI EFI module was or is currently loaded. The associated data structure may be read to obtain the version information.