– DCMI –

Data Center Manageability Interface Specification v1.1, Revision 1.0

Addenda, Errata, and Clarifications

Addendum Document Revision 1
December 6, 2010

THIS SPECIFICATION IS PROVIDED "AS IS" WITH NO WARRANTIES WHATSOEVER, INCLUDING ANY WARRANTY OF MERCHANTABILITY, NONINFRINGEMENT, FITNESS FOR ANY PARTICULAR PURPOSE, OR ANY WARRANTY OTHERWISE ARISING OUT OF ANY PROPOSAL, SPECIFICATION OR SAMPLE. Intel disclaims all liability, including liability for infringement of any proprietary rights, relating to use of information in this specification. No license, express or implied, by estoppels or otherwise, to any intellectual property rights is granted herein, except that a license is hereby granted to copy and reproduce this specification for internal use only.

Intel retains the right to make changes to this document at any time, without notice. Intel make no warranty for the use of this document and assume no responsibility for any error which may appear in the document, nor does it make a commitment to update the information contained herein.

Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined." Intel reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them.

Table of Contents

Introduction

This document presents errata and clarifications applying to the *Data Center Manageability Interface Specification v1.1, Revision 1.0.* The section, table, and figure references are taken from the published version and revision of that specification unless otherwise noted. Where examples of text changes are given, text additions are identified with underlines and text deletions are identified with strike-through.

As of this writing the DCMI specifications are available at:
http://ww.intel.com/go/DCMI

Errata Numbers

Errata numbers in this document are relative to the published (non-markup) version of the specification. In some cases, errata numbers may be skipped in this document. This is intentional. The errata numbers are derived from numbers used to track errata and clarification requests among the DCMI stakeholders. Gaps in the errata number sequence can subsequently result due to requests that have been dropped or that are still under investigation.

Revision 1 Addenda, Errata, and Clarifications

E1: Typo & Clarification: Table 3-2, DCMI Compliant Sensor Definition

Note 1 in the table referenced an incorrect Entity ID code value. Also a clarification has been added to better cover the origin and usage of the Entity ID code values. This is corrected and clarified as follows:

Table 3-2, DCMI Compliant Sensor Definition

	Name/Description	Code	M/O
Sensor:	**Inlet Temperature**		
Associated Entity ID	Inlet Temperature	37h,40h[1]	M
Sensor Type Code:	Temperature	01h	M
Event / Reading Type Code:	Threshold	01h	M
Supported Thresholds / Events (Event Offsets)	Upper Non-critical - going high	07h	O
	Upper Critical - going high	09h	M
Sensor:	**CPU (Processor) Temperature**		
Associated Entity ID	CPU Temperature	03h,41h[2]	M
Sensor Type Code:	Temperature	01h	M
Event / Reading Type Code:	Threshold	01h	M
Supported Thresholds / Events (Event Offsets)	Upper Non-critical - going high	07h	O
	Upper Critical - going high	09h	M
Sensor:	**Baseboard Temperature**		
Associated Entity ID	Baseboard	07h,42h[3]	M
Sensor Type Code:	Temperature	01h	M
Event / Reading Type Code:	Threshold	01h	M
Supported Thresholds / Events (Event Offsets)	Upper Non-critical - going high	07h	O
	Upper Critical - going high	09h	M

[1] Either DCMI 1.0 defined Entity ID 40h or IPMI defined Entity ID ~~27h~~ 37h is acceptable. Note future revisions of the specification are expected to deprecate the DCMI 1.0 value in order to settle on a single Code value.

[2] Either DCMI 1.0 defined Entity ID 41h or IPMI defined Entity ID 03h is acceptable. Note future revisions of the specification are expected to deprecate the DCMI 1.0 value in order to settle on a single Code value.

[3] Either DCMI 1.0 defined Entity ID 42h or IPMI defined Entity ID 07h is acceptable. Note future revisions of the specification are expected to deprecate the DCMI 1.0 value in order to settle on a single Code value.

E2: Clarification: Section 5.4.1.2, RMCP+ Protocol for IPMI over LAN Access

The text was updated to emphasize that DCMI's primary communication is RMCP+ as defined in IPMI v2.0. While IPMI v2.0 also calls out for support for the older IPMI v1.5 RMCP Sessions. IPMI v1.5 RMCP Session support *is not* required for DCMI.

5.4.1.2. RMCP+ Protocol for IPMI over LAN Access

As described in [IPMI] Specification Section 13. Support for IPMI v1.5 RMCP sessions is optional for DCMI.

E3: Clarification: Table 6-1, Platform Command Definition, for Get and Send Message commands

The Get Message and Send Message command support for DCMI was initially called out as mandatory for implementations that required it for bridging to satellite controllers. Footnotes have been added to the table to clarify that the Get and Send Message support is otherwise optional, as follows:

Table 6-1, Platform Command Definition

	NetFn	**CMD**	**M/O[1]**	**Min Privilege Level**
Discovery Command				
Get DCMI Capabilities Info	DCGRP (2Ch)	01h	M	Session-less[4]
Chassis Commands				
Get Chassis Capabilities	Chassis (00h)	00h	M	User

...

Messaging Commands				
Get Message	App (06h)	33h	M[3]	System Interface[2]
Send Message	App (06h)	34h	M[3]	User[3]

...

Activate/Deactivate Power Limit	DCGRP (2Ch)	05h	O	Operator[5]

App = Application Network Function Code
S/E = Sensor/Event Network Function Code
DCGRP = DCMI Group Extension Network Function Code

[1] Mandatory or Optional Feature

[2] Command is only executable via the system interface.

[3] Mandatory if required to support message bridging to satellite controllers on IPMB. If supported for system software messaging, a~~A~~ User can use a Send Message command to deliver a message to system software, but Operator privilege is required to use it to access other channels.

[4] Command can be executed at any privilege level and is available before and after establishing a session.

[5] These commands are mandatory if DCMI power management is supported by the platform.

E4: Addendum: Reference Documents and corresponding cross references

The Reference Documents section was updated to add [RFC 2131] Dynamic Host Configuration Protocol, [RFC 2132] DHCP Options and BOOTP Vendor Extensions, and [DCMI-HI] DCMI Host Interface Specification, Revision 1.0. A corresponding cross reference to [RFC 2132] was added to section 5.5.2.1 as follows:

5.5.2.1. DHCP enabled management controllers

The management controllers shall publish a unique non-null Host Name or Fully-Qualified Domain Name (FQDN) using either DHCP option 12 or DHCP Option 81 (per [RFC 2132]) when sending DHCP Discover packets during DHCP negotiations. The maximum length of the identifier string is set to 64 bytes.

E5: Clarification: 5.3.1.2 LAN Interface Requirements

The text has been clarified regarding what a secondary LAN channel is. The following change is made to Section 5.3.1.2:

9. Platform may provide an additional LAN channel and designate it as the secondary LAN channel.

A corresponding clarification was made to the DCMI Capabilities parameters. See E12, Clarifications: Table 6-3, DCMI Capabilities Parameters.

E6: Clarification and Addendum: Sections 6.41, Asset Tag, 6.4.1.1, Get Asset Tag Command and 6.4.1.2 Set Asset Tag Command

The Set- and Get Asset Tag commands did not clearly specify error behaviors under implementations where IPMI FRU storage was being used to hold the Asset Tag information. Additionally, the Asset Tag specification for required the data to be stored in UTF-8 but did not define the mechanism to identify the character set encoding for the data when stored in IPMI FRU. (This issue arises because IPMI FRU does not intrinsically define a UTF-8 encoding for FRU data). Nor did the specification allow for the more common ASCII+Latin1 subset. Lastly, the specification was clarified to indicate that the maximum size of the Asset Tag should be 63 bytes, including Byte Order Mark if provided. This is due to the underlying limitation of the IPMI FRU data format, which limits Asset Tag data to 63 bytes. This has been addressed with the following additions and clarifications:

6.4.1 Asset Tag

…

~~The maximum size of the Asset Tag shall be 64 bytes.~~ The Asset Tag shall be encoded as UTF-8 with Byte Order Mark, or as the ASCII+Latin1 subset. The maximum size of the Asset Tag shall be 63 bytes, including Byte Order Mark, if provided. (The 63 byte limit is due to a restriction in underlying IPMI FRU data structures).

6.4.1.1 Get Asset Tag Command

This command enables ~~Provides ability for~~ management consoles or local software to ~~query~~ get the A~~a~~sset T~~t~~ag data.~~set by default or modified by the IPDC using Set Asset Tag Command.~~ UTF-8 encoding is identified when the first three bytes (offsets 0, 1, and 2) of the returned Asset Tag Data are set to the UTF-8 Byte Order Mark (BOM) pattern, EFh, BBh, BFh, respectively.

Table 6-4, Get Asset Tag Command

	byte	data field
Request Data	1	Group Extension Identification = DCh
	2	Offset to read
	3	Number of bytes to read (16 bytes maximum) Note: If the number of bytes to read starting from the given Offset to read exceeds the number remaining Asset Tag data bytes, the command will complete normally (completion code = 00h) but will only return the remaining bytes (Provided the Offset to read and bytes to read are within their correct ranges. See description of the Completion Code for this command.) . For example, if the Asset Tag Length is presently 20 bytes, submitting an Offset to read of 16 and a bytes to read of 16 will be accepted, but only the Asset Tag Data bytes at offsets 16-19 will be returned.
Response Data	1	Completion Code. Refer to Section C - DCMI Completion Codes. C9h shall be returned if offset >62, offset to read+bytes to read >63, or bytes to read >16. The following applies to implementations that keep the DCMI Asset Tag and IPMI FRU Asset Tag information synchronized: If the encoding indicated by the Type/Length byte in the IPMI FRU is not set to ASCII+Latin1 or the Language Code for the IPMI FRU Product Info Area is not set to English (0 or 25), the command shall return the requested data bytes, but shall also return a command-specific completion code based on the detected encoding type, as follows: 80h = Encoding type in FRU is binary / unspecified 81h = Encoding type in FRU is BCD Plus 8~~1~~2h = Encoding type in FRU is 6-bit ASCII Packed 8~~2~~3h = Encoding type in FRU is set to ASCII+Latin1 but language code is not set to English (indicating data is 2-byte UNICODE). The management controller does not check for, nor require, a BOM in the Asset Tag Data. Thus, Asset Tag data can be stored and retrieved as ASCII+Latin1 without receiving an error completion code.
	2	Group Extension Identification = DCh
	3	Total Asset Tag Length Note: The length shall be less than or equal to 64 bytes
	4 - N	Asset Tag Data (starting from offset to read)~~Data~~

6.4.1.2 Set Asset Tag Command

This command enables remote consoles or local software to set the Asset Tag data. UTF-8 encoding of the Asset Tag data is identified by setting the first three bytes (offsets 0, 1, and 2) of the Asset Tag Data to the UTF-8 Byte Order Mark (BOM) pattern, EFh, BBh, BFh, respectively. Otherwise, the data encoding shall be assumed to be the ASCII+Latin1 subset. Note that the management controller simply stores all eight bits of each of the given Asset Tag Data bytes. It does not check the encoding of the Asset Tag Data bytes, nor does it check for a BOM in the data.

Implementations that keep the Asset Tag in synch with the IPMI FRU data shall write the given characters to the Asset Tag field in the Product Info Area of the IPMI FRU Device and set the encoding of the corresponding Type/Length Byte field to ASCII+Latin1.

Table 6-5, Set Asset Tag Command

	byte	data field
Request Data	1	Group Extension Identification = DCh
	2	Offset to ~~Write~~write (0 to 62), The offset is relative to the first character of the Asset Tag data.
	3	Number of bytes to ~~Written~~ write (16 bytes maximum). Note: The command shall set the overall length of the Asset Tag (in bytes) to the value (offset to write + bytes to write). Any pre-existing Asset Tag bytes at offsets past that length are automatically deleted.
	4 - N	Asset Tag Data
Response Data	1	Completion Code. Refer to Section 8 – DCMI Completion Codes. C9h shall be returned if offset >62, offset+bytes to write >63, or bytes to write >16. A C9h completion code shall also be returned if an attempt is made to write to an offset that is more than one greater than the length of the presently stored Asset Tag data. Set operations for Asset Tags must be contiguous. For example, if the Asset Tag is presently seven bytes long an attempt to write starting at offset 10 will be rejected and a C9h completion code returned.
	2	Group Extension Identification = DCh
	3	Total Asset Tag Length ~~written~~. This is the length in bytes of the stored Asset Tag after the Set operation has completed. The Asset Tag length shall be set to the sum of the offset to write plus bytes to write. For example, if offset to write is 32 and bytes to write is 4, the Total Asset Tag Length returned will be 36.

E7: Clarification: Section 6.4.4., Management Controller Identifier String

The text was ambiguous regarding whether the maximum length included the NULL terminator or not. This is clarified as follows:

> The maximum length of the identifier string shall be 64 bytes ~~and~~ including a NULL ~~terminated~~terminator.

E8: Clarification: Section 5.3.1.1, System Interface Requirements

The text was updated to clarify that DCMI only specifies the System Interfaces for use by system management software. Definition of other interfaces, or KCS/DCMI Host Interface use for System Management Interrupt Handlers or proprietary manufacturing configuration utilities, is outside the scope of the specification.

E9: Addendum: Get DCMI Sensor Info Command

The command did not have an overview description. A description has been added as follows:

6.5.2 Get DCMI Sensor Info Command

This DCMI command returns information about a DCMI-specified sensor. A particular sensor is identified by the combination of its Entity ID and Entity Instance numbers, as listed in Table 6-8, DCMI Entity ID Extension.

E10: Clarification and Addendum: Table 6-11, Get Power Limit Command

The command neglected to list that an event should always be logged in order to provide a post-mortem record that the system was automatically powered down due to Power Limit exception action. Additionally, the definition of the Correction Time Limit parameter was ambiguous. This has been corrected as follows:

Table 6-11, Get Power Limit Command

	byte	data field
Request Data	1	Group Extension Identification = DCh
	2:3	Reserved for future use, use 0000h
Response Data	1	Completion Code. Refer to Section 8 - DCMI Completion Codes. 00h = Power Limit Active 80h = No Active Set Power Limit OEM can also provide Completion Codes.
	2	Group Extension Identification = DCh
	3:4	Reserved for future use
	5	Exception Actions, taken if the Power Limit is exceeded and cannot be controlled within the Correction Time Limit 01h Hard Power Off system and log event to SEL 02h – 10h OEM defined actions 11h Log event to SEL ~~Logging~~only 12h-FFh Reserved
	6:7	Power Limit Requested in Watts
	8:11	Correction Time Limit in milliseconds ~~Maximum time taken to limit the power, otherwise exception action will be taken as configured~~See description of corresponding parameter in Table 6-12, Set Power Limit Command.
	12:13	Reserved for future use
	14:15	Management application Statistics Sampling period in seconds

E11: Clarification and Addendum: Section 6.6.3 Set Power Limit and Table 6-12, Set Power Limit Command

The description of the Set Power Limit command was unclear regarding whether the requested Power Limit was setting a capping threshold or not. This has been clarified with additional text in the command description. Additional clarifications were made regarding when changes to the limit parameters take effect. Lastly, the Exception Actions were updated to line up with the clarifications to the Get Power Limit command.

6.6.3 Set Power Limit

The Set Power Limit command sets the power limit parameters on the system. The power limit defines a threshold which, if exceeded for a configurable amount of time, will trigger a system power off and/or event logging action. This enables the Power Limit to be used as a form of 'circuit breaker' for protecting data center power delivery from systems that have abnormal, prolonged power excursions outside their normal operating range.

It is recommended to do a Get Power Limit or check the Get Power Reading command before attempting to set and activate or re-activate the power limit. If the limit is already active, the Set Power Limit command may immediately change the limit that is in effect. However, software should always explicitly activate the limit using the Activate/Deactivate power limit command to ensure the setting takes effect.

It should be noted that in the current context, this command shall be used to set a static upper limit of system power usage and not used as a command interface for dynamic or frequently changing power limit. The power limit setting should be persistent across AC and DC cycles.

Table 6-12, Set Power Limit Command

	byte	data field
Request Data	1	Group Extension Identification = DCh
	2:4	Reserved for future use
	5	Exception Actions, taken if the Power Limit is exceeded and cannot be controlled within the Correction time limit 00h – No Action 01h – Hard Power Off system and log event to SEL 02h – 10h OEM defined actions 11h – Log event to SEL only ~~Logging~~ 12h-FFh Reserved
	6:7	Power Limit Requested in Watts
	8:11	Correction ~~time~~ Time L~~l~~imit in milliseconds Maximum time ~~taken to limit the power~~ after the platform power has reached the power limit~~, otherwise exception action will be taken as configured~~ before the Exception Action will be taken. The Exception Action shall be taken if the system power usage constantly exceeds the specified power limit for more than the Correction Time Limit interval. The Correction Time Limit timeout automatically restarts if the system power meets or drops below the Power Limit.
	12:13	Reserved for future use
	14:15	Management application Statistics Sampling period in seconds
Response Data	1	Completion Code. Refer to Section 8 - DCMI Completion Codes. =00h – Success =84h – Power Limit out of range =85h – Correction Time out of range =89h – Statistics Reporting Period out of range OEM can also provide Completion Codes.
	2	Group Extension Identification = DCh

E12: Clarifications: Table 6-3, DCMI Capabilities Parameters

Some changes were made to the wording of parameter 1 for consistency, to clarify the meaning of bit 0, and to clarify which bits are related to optional capabilities.

Table 6-3, DCMI Capabilities Parameters

Parameter	#	Parameter Data (non-volatile unless otherwise noted)
Supported DCMI Capabilities	1	This field returns the supported capabilities available in the server in conformance to DCMI specification for both Platform and Manageability access. All reserved bits shall be set to 0b byte 1 Reserved byte 2 ~~Optional~~ Platform capabilities All bits: 0b = Not present~~not Compliant with DCMI Specification~~ 1b = Available~~compliant with DCMI Specification~~ [7:1] Reserved [0] Power management / monitoring support This is defined as support for either power monitoring or power monitoring plus power limiting, per the DCMI Specification. byte 3 Manageability Access Capabilities All bits 0b = Not present 1b = Available [7:3] Reserved [2] Out-Of-Band Secondary (second) LAN Channel Available (optional). [1] ~~Out-Of-B~~ Serial TMODE Available (TMODE on serial port to management controller) (optional). [0] In-band System Interface Channel Available

...