



1
2
3
4

Document Number: DSP0246

Date: 2009-12-11

Version: 1.0.1

5 **Platform Level Data Model (PLDM) for SMBIOS**
6 **Data Transfer Specification**

7 **Document Type: Specification**
8 **Document Status: DMTF Standard**
9 **Document Language: E**

10

11 Copyright notice

12 Copyright © 2008, 2009 Distributed Management Task Force, Inc. (DMTF). All rights reserved.

13 DMTF is a not-for-profit association of industry members dedicated to promoting enterprise and systems
14 management and interoperability. Members and non-members may reproduce DMTF specifications and
15 documents, provided that correct attribution is given. As DMTF specifications may be revised from time to
16 time, the particular version and release date should always be noted.

17 Implementation of certain elements of this standard or proposed standard may be subject to third party
18 patent rights, including provisional patent rights (herein "patent rights"). DMTF makes no representations
19 to users of the standard as to the existence of such rights, and is not responsible to recognize, disclose,
20 or identify any or all such third party patent right, owners or claimants, nor for any incomplete or
21 inaccurate identification or disclosure of such rights, owners or claimants. DMTF shall have no liability to
22 any party, in any manner or circumstance, under any legal theory whatsoever, for failure to recognize,
23 disclose, or identify any such third party patent rights, or for such party's reliance on the standard or
24 incorporation thereof in its product, protocols or testing procedures. DMTF shall have no liability to any
25 party implementing such standard, whether such implementation is foreseeable or not, nor to any patent
26 owner or claimant, and shall have no liability or responsibility for costs or losses incurred if a standard is
27 withdrawn or modified after publication, and shall be indemnified and held harmless by any party
28 implementing the standard from any and all claims of infringement by a patent owner for such
29 implementations.

30 For information about patents held by third-parties which have notified the DMTF that, in their opinion,
31 such patent may relate to or impact implementations of DMTF standards, visit
32 <http://www.dmtf.org/about/policies/disclosures.php>.

33

34

35

CONTENTS

36	Foreword	5
37	Introduction	6
38	1 Scope	7
39	2 Normative References.....	7
40	3 Terms and Definitions.....	7
41	4 Symbols and Abbreviated Terms.....	8
42	5 Conventions	8
43	6 SMBIOS Overview	8
44	7 PLDM for SMBIOS Data Transfer Overview	9
45	8 PLDM for SMBIOS Data Transfer	9
46	8.1 PLDM Representation of SMBIOS Structure Table.....	9
47	8.2 PLDM Commands for SMBIOS Data Transfer	11
48	8.3 PLDM for SMBIOS Data Transfer Version	16
49	9 PLDM for SMBIOS Data Transfer Examples	16
50	9.1 Multipart Transfers	16
51	9.2 SMBIOS Table Transfer from BIOS to MC Example.....	18
52	ANNEX A (Informative) Change Log.....	20
53		

54 Figures

55	Figure 1 – Multipart SMBIOS Structure Table Transfer Using the SetSMBIOSStructureTable Command 17
56	Figure 2 – Multipart SMBIOS Structure Table Transfer Using the GetSMBIOSStructureTable Command 18
57	Figure 3 – Example of SMBIOS Table Transfer Using the SetSMBIOSStructureTable Command 19

58 Tables

59	Table 1 – PLDM Representation of an SMBIOS Structure.....	10
60	Table 2 – PLDM Representation of SMBIOSStructureData	10
61	Table 3 – PLDM for SMBIOS Data Transfer Command Codes.....	11
62	Table 4 – GetSMBIOSStructureTableMetadata Command Format	11
63	Table 5 – SetSMBIOSStructureTableMetadata Command Format.....	12
64	Table 6 – GetSMBIOSStructureTable Command Format	13
65	Table 7 – SetSMBIOSStructureTable Command Format.....	14
66	Table 8 – GetSMBIOSStructureByType Command Format	14
67	Table 9 – GetSMBIOSStructureByHandle Command Format.....	15
68		

70

Foreword

71 The *Platform Level Data Model (PLDM) for SMBIOS Data Transfer Specification* (DSP0246) was
72 prepared by the Platform Management Components Intercommunications (PMCI) Working Group.

73 DMTF is a not-for-profit association of industry members dedicated to promoting enterprise and systems
74 management and interoperability.

75

Introduction

76 This specification describes Platform Level Data Model (PLDM) data structures and commands for
77 transferring SMBIOS data between the components of a platform management hardware subsystem.
78 This document specifies PLDM representations of SMBIOS structure table and SMBIOS structures, and a
79 set of commands for transferring SMBIOS structure table and SMBIOS structure data.

80

81 Platform Level Data Model (PLDM) for SMBIOS Data Transfer 82 Specification

83 1 Scope

84 DSP0134, *System Management BIOS (SMBIOS) Reference Specification*, defines BIOS extensions that
85 provide platform asset information such as BIOS version, processor speed/type, and memory capacity.
86 The SMBIOS structure table typically resides in the system memory and contains one or more SMBIOS
87 structures. This document describes Platform Level Data Model (PLDM) data structures and commands
88 for transferring SMBIOS data between the components of a platform management hardware subsystem.

89 This document meets the following objectives:

- 90 • Specifies PLDM representations of SMBIOS structure table and SMBIOS structures
- 91 • Specifies a set of commands for transferring SMBIOS structure table and SMBIOS structure
92 data

93 2 Normative References

94 The following referenced documents are indispensable for the application of this document. For dated
95 references, only the edition cited applies. For undated references, the latest edition of the referenced
96 document (including any amendments) applies.

97 DMTF DSP0134, *System Management BIOS (SMBIOS) Reference Specification 2.6*,
98 http://www.dmtf.org/standards/published_documents/DSP0134_2.6.pdf

99 DMTF DSP0240, *Platform Level Data Model (PLDM) Base Specification 1.0*,
100 http://www.dmtf.org/standards/published_documents/DSP0240_1.0.pdf

101 DMTF DSP0245, *Platform Level Data Model (PLDM) IDs and Codes 1.0*,
102 http://www.dmtf.org/standards/published_documents/DSP0245_1.0.pdf

103 ISO/IEC Directives, Part 2, *Rules for the structure and drafting of International Standards*,
104 <http://isotc.iso.org/livelink/livelink.exe?func=ll&objId=4230456&objAction=browse&sort=subtype>

105 OMG, *Unified Modeling Language (UML) from the Open Management Group (OMG)*, <http://www.uml.org/>

106 3 Terms and Definitions

107 Refer to [DSP0240](#) for terms and definitions that are used across the PLDM specifications. For the
108 purposes of this document, the following additional terms and definitions apply.

109 3.1

110 System Management BIOS 111 SMBIOS

112 BIOS extensions that provide platform asset information such as BIOS version, processor speed/type,
113 and memory capacity as specified in [DSP0134](#)

114 3.2**115 SMBIOS structure**

116 A SMBIOS structure provides information about a component within a platform. A SMBIOS structure has
117 a formatted section and an optional unformatted section. The formatted section of each structure begins
118 with a 4-byte header. Remaining data in the formatted section is determined by the structure type, as is
119 the overall length of the formatted section.

120 3.3**121 SMBIOS structure table**

122 the table that contains the SMBIOS structures

123 3.4**124 SMBIOS table**

125 See 3.3.

126 4 Symbols and Abbreviated Terms

127 Refer to [DSP0240](#) for symbols and abbreviated terms that are used across the PLDM specifications. For
128 the purposes of this document, the following additional symbols and abbreviated terms apply.

129 4.1**130 BIOS**

131 Basic Input Output System

132 4.2**133 SMBIOS**

134 System Management BIOS

135 5 Conventions

136 Refer to [DSP0240](#) for conventions, notations, and data types that are used in the PLDM specifications.

137 6 SMBIOS Overview

138 The *Platform Level Data Model (PLDM) for SMBIOS Data Transfer Specification* defines BIOS extensions
139 that provide platform asset information such as BIOS version, processor speed/type, and memory
140 capacity. The SMBIOS structure table resides in the system memory and contains one or more SMBIOS
141 structures. Each SMBIOS structure begins with a (type, length, and handle) header. The SMBIOS
142 structures are not ordered and searching for a specific structure requires parsing the SMBIOS structure
143 table.

144 The SMBIOS structure table data is important for the instrumentation because it is used in the following
145 ways:

- 146 • Platform asset information available in the SMBIOS structure table can be used to populate the
147 instances of CIM classes that provide physical asset and hardware inventory information to the
148 remote management console using the WBEM infrastructure.
- 149 • The information available in the SMBIOS structure table can be used for system health
150 monitoring.
- 151 • The event log information, if available in the SMBIOS structure table, can be used to access the
152 event log and perform system event monitoring.

153 7 PLDM for SMBIOS Data Transfer Overview

154 A Management Controller may wish to utilize the data in the SMBIOS structure table as a data source for
155 providing platform inventory information via a CIM-based external interface. Depending on the
156 implementation, additionally providing this information when the system is in low power states may
157 require maintaining multiple copies of SMBIOS structure table data within a platform and keeping the
158 copies consistent between the MC and the SMBIOS table information that is accessed by the system
159 software and BIOS. There is thus a need for a platform-level data model (PLDM) for SMBIOS data
160 transfer that can be used between the system firmware (BIOS) and a management controller, and
161 between management controllers. Following are the design characteristics for the PLDM for SMBIOS
162 data transfer:

- 163 • The PLDM defines commands to obtain the SMBIOS structure table metadata information, such
164 as versioning information, checksum information, table length, number of SMBIOS structures,
165 and maximum structure size.
- 166 • The PLDM preserves the SMBIOS structure format for the data transfer. By maintaining the
167 SMBIOS structure format at the PLDM level, the need to parse the SMBIOS structure data for
168 the PLDM data transfer is avoided.
- 169 • The SMBIOS structure table or SMBIOS structures can be large. The SMBIOS structure table
170 data or SMBIOS structure data may not fit in a single PLDM message. The PLDM defines
171 commands that allow the transfer of entire SMBIOS structure table or SMBIOS structures using
172 either a single request/response or multiple requests/responses.
- 173 • The PLDM supports both pull and push models for the SMBIOS structure table data transfer
174 and SMBIOS structure data transfer. In the push model, the SMBIOS structure table transfer is
175 initiated by the sender without being explicitly requested by the receiving entity. In the pull
176 model, the transfer of the SMBIOS structure table is requested by a receiving entity. The BIOS
177 initiating the transfer of its SMBIOS structure table to a management controller is an example of
178 the push model. A management controller sending read requests to BIOS telling it to provide
179 SMBIOS structure table data is an example of the pull model.
- 180 • The PLDM defines a data integrity check to protect the SMBIOS structure table data transfer
181 and SMBIOS structure data transfer.
- 182 • The PLDM defines commands to read SMBIOS structure data by type or by handle to enable
183 reading a subset of structures from the SMBIOS structure table.
- 184 • The PLDM does not define commands to update or write a subset of structures from the
185 SMBIOS structure table as it typically requires reading the entire table, followed by writing the
186 subset of structures, and updating the SMBIOS table integrity checksum that covers the entire
187 SMBIOS structure table.
- 188 • The PLDM does not define commands (read or write) to transfer partial SMBIOS structure or
189 elements of an SMBIOS structure.

190 8 PLDM for SMBIOS Data Transfer

191 This section defines the data structures and commands for SMBIOS data transfer.

192 8.1 PLDM Representation of SMBIOS Structure Table

193 In the PLDM messages for SMBIOS data transfers, an SMBIOS structure representation is the same as
194 described in the SMBIOS specification ([DSP0134](#)). Each SMBIOS structure has a formatted section and
195 an optional unformatted section as defined in [DSP0134](#). The formatted section begins with a 4-byte header:
196 Type (1 byte), Length (1 byte), and Handle (2 bytes). The unformatted section is used to pass variable
197 length structures (for example, text strings). Each SMBIOS structure is terminated by double null (0000h).

198 Table 1 shows the SMBIOS structure representation at the PLDM level.

199 **Table 1 – PLDM Representation of an SMBIOS Structure**

Byte	Type	Field
0	uint8	Type as defined in DSP0134
1	uint8	Length (L bytes) as defined in DSP0134
2:3	uint16	Handle as defined in DSP0134
4:L-1	–	The formatted area of the structure
Variable	–	Variable bytes of unformed area of the structure terminated by double null (0000h) as defined in DSP0134

200 The SMBIOS structure table data consists of multiple SMBIOS structures. When a set of one or more
 201 SMBIOS structures (up to the entire SMBIOS structure table data) is transferred using PLDM messages,
 202 the PLDM representation shown in Table 2 is used.

203 **Table 2 – PLDM Representation of SMBIOSStructureData**

Byte	Type	Field
Variable	–	SMBIOS structures (one or more) See Table 1 for the PLDM representation of an SMBIOS structure.
Variable	uint8[]	Pad 0 to 3 number of pad bytes. The value stored in each pad byte is 0x00. The transmitter can compute the number of pad bytes from the SMBIOSStructureData by using the following algorithm: Let L be the total number of bytes in the SMBIOSStructureData excluding the pad and the integrity checksum. if (L modulo 4 == 0) then NumPadBytes = 0; else NumPadBytes = 4 – L modulo 4; The receiver can compute the number of pad bytes from the SMBIOSStructureData by using the following algorithm. In the algorithm, the receiver parses SMBIOS structure data until the remaining bytes are less than 8. When it reaches that stage, the remaining bytes contain the pad bytes and four bytes of data integrity checksum. Let L be the total number of bytes in the SMBIOSStructureData including the pad and the integrity checksum. RemBytes = L; i = 0; while (RemBytes >= 8) { Process the i th SMBIOS structure in the SMBIOSStructureData; RemBytes = RemBytes - 4 – Total length of i th SMBIOS structure including the formatted and unformed areas; i = i+1; } NumPadBytes = RemBytes modulo 4;
	uint32	SMBIOSStructureDataIntegrityChecksum Integrity checksum on the SMBIOS structure data including the pad bytes (if any). It is calculated starting at the first byte of the PLDM representation of SMBIOSStructureData. For this specification, the CRC-32 algorithm with the polynomial $x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$ (same as the one used by IEEE 802.3) shall be used for the integrity checksum computation. The CRC computation involves processing a byte at a time with the least significant bit first.

204 **8.2 PLDM Commands for SMBIOS Data Transfer**

205 Table 3 defines the PLDM command codes defined in this section for the PLDM for SMBIOS data
 206 transfer.

207 **Table 3 – PLDM for SMBIOS Data Transfer Command Codes**

Command	Code Value	Requirement	Section
GetSMBIOSStructureTableMetadata	0x01	Mandatory	See 8.2.1.
SetSMBIOSStructureTableMetadata	0x02	Conditional ²	See 8.2.2.
GetSMBIOSStructureTable	0x03	Conditional ¹	See 8.2.3.
SetSMBIOSStructureTable	0x04	Conditional ¹	See 8.2.4.
GetSMBIOSStructureByType	0x05	Optional	See 8.2.5.
GetSMBIOSStructureByHandle	0x06	Optional	See 8.2.6.
¹ At least one of these two commands must be supported by the requester and the responder for a compliant PLDM for SMBIOS data transfer implementation. ² If an implementation is transferring SMBIOS structure table data using the SetSMBIOSStructureTable command, it shall support the SetSMBIOSStructureTableMetadata command.			

208 The requirements specified in Table 3 are relative to the services provided by the PLDM terminus.

209 The following sections define the PLDM commands for SMBIOS data transfer.

210 **8.2.1 GetSMBIOSStructureTableMetadata**

211 The GetSMBIOSStructureTableMetadata command, described in Table 4, is used to get the SMBIOS
 212 structure table metadata information that includes the SMBIOS major version, the SMBIOS minor version,
 213 the size of the largest SMBIOS structure, total length of the SMBIOS structure table, total number of
 214 SMBIOS structures, and the integrity checksum on the SMBIOS structure table data.

215 **Table 4 – GetSMBIOSStructureTableMetadata Command Format**

Byte	Type	Request Data
-	-	No Request Data
Byte	Type	Response Data
0	enum8	CompletionCode Possible values: { PLDM_BASE_CODES, NO_SMBIOS_STRUCTURE_TABLE_METADATA=0x83 }
1	uint8	SMBIOSMajorVersion The major version of the SMBIOS specification with which the SMBIOS structure table complies
2	uint8	SMBIOSMinorVersion The minor version of the SMBIOS specification with which the SMBIOS structure table complies

3:4	uint16	MaximumStructureSize Size of the largest SMBIOS structure, in bytes, including the structure's formatted area and unformed area
5:6	uint16	SMBIOSStructureTableLength Total length of the SMBIOS structure table, in bytes
7:8	uint16	NumberOfSMBIOSStructures Total number of SMBIOS structures present in the SMBIOS structure table
9:12	uint32	SMBIOSStructureTableIntegrityChecksum (CRC-32) Integrity checksum on the SMBIOS structure table data as shown in Table 2 excluding pad bytes. See Table 2 for more information about this integrity checksum.

216 8.2.2 SetSMBIOSStructureTableMetadata

217 The SetSMBIOSStructureTableMetadata command, described in Table 5, is used to set the SMBIOS
 218 structure table metadata information that includes the SMBIOS major version, the SMBIOS minor version,
 219 the size of the largest SMBIOS structure, total length of the SMBIOS structure table, total number of
 220 SMBIOS structures, and the integrity checksum on the SMBIOS structure table data.

221 **Table 5 – SetSMBIOSStructureTableMetadata Command Format**

Byte	Type	Request Data
0	uint8	SMBIOSMajorVersion The major version of the SMBIOS specification with which the SMBIOS structure table complies
1	uint8	SMBIOSMinorVersion The minor version of the SMBIOS specification with which the SMBIOS structure table complies
2:3	uint16	MaximumStructureSize Size of the largest SMBIOS structure, in bytes, including the structure's formatted area and unformed area
4:5	uint16	SMBIOSStructureTableLength Total length of the SMBIOS structure table, in bytes
6:7	uint16	NumberOfSMBIOSStructures Total number of SMBIOS structures present in the SMBIOS structure table
8:11	uint32	SMBIOSStructureTableIntegrityChecksum (CRC-32) Integrity checksum on the SMBIOS structure table data as shown in Table 2 excluding pad bytes. See Table 2 for more information about this integrity checksum.
Byte	Type	Response Data
0	enum8	CompletionCode Possible value: { PLDM_BASE_CODES}

222 8.2.3 GetSMBIOSStructureTable

223 The GetSMBIOSStructureTable command, described in Table 6, is used to get the SMBIOS structure
 224 table data. This command is defined to allow the SMBIOS structure table data to be transferred using a
 225 sequence of one or more command/response messages. When more than one command is used to
 226 transfer the SMBIOS structure table data, the response messages contain the non-overlapping
 227 contiguous portions of SMBIOS structure table data as defined in Table 2. By combining the portions of
 228 SMBIOS structure table data from the response messages, the entire SMBIOS structure table data can
 229 be reconstructed.

230

Table 6 – GetSMBIOSStructureTable Command Format

Byte	Type	Request Data
0:3	uint32	DataTransferHandle A handle that is used to identify an SMBIOS structure table data transfer. This handle is ignored by the responder when the TransferOperationFlag is set to GetFirstPart.
4	enum8	TransferOperationFlag The operation flag that indicates whether this is the start of the transfer Possible values: {GetNextPart=0x00, GetFirstPart=0x01}
Byte	Type	Response Data
0	enum8	CompletionCode Possible values: { PLDM_BASE_CODES, INVALID_DATA_TRANSFER_HANDLE=0x80, INVALID_TRANSFER_OPERATION_FLAG=0x81, SMBIOS_STRUCTURE_TABLE_UNAVAILABLE=0x85 }
1:4	uint32	NextDataTransferHandle A handle that is used to identify the next portion of the transfer
5	enum8	TransferFlag The transfer flag that indicates what part of the transfer this response represents Possible values: {Start=0x01, Middle=0x02, End=0x04, StartAndEnd = 0x05}
Variable	–	Portion of SMBIOSStructureData See Table 2 for the format.

231 8.2.4 SetSMBIOSStructureTable

232 The SetSMBIOSStructureTable command, described in Table 7, is used to push the SMBIOS structure
 233 table data. This command is defined to allow the SMBIOS structure table data to be transferred using a
 234 sequence of one or more command/response messages. When more than one command is used to
 235 transfer the SMBIOS structure table data, the request messages contain the non-overlapping contiguous
 236 portions of SMBIOS structure table data as defined in Table 2. By combining the portions of SMBIOS
 237 structure table data from the request messages, the entire SMBIOS structure table data can be
 238 reconstructed.

239

Table 7 – SetSMBIOSStructureTable Command Format

Byte	Type	Request Data
0:3	uint32	DataTransferHandle A handle that is used to identify SMBIOS structure table transfer. This handle is ignored by the responder when the TransferFlag is set to Start or StartAndEnd.
4	enum8	TransferFlag The transfer flag that indicates what part of the transfer this request represents Possible values: {Start=0x01, Middle=0x02, End=0x04, StartAndEnd = 0x05}
Variable	–	Portion of SMBIOSStructureData See Table 2 for the format.
Byte	Type	Response Data
0	enum8	CompletionCode Possible values: { PLDM_BASE_CODES, INVALID_DATA_TRANSFER_HANDLE=0x80, INVALID_TRANSFER_FLAG=0x82, INVALID_DATA_INTEGRITY_CHECK=0x84 }
1:4	uint32	NextDataTransferHandle A handle that is used to identify the next portion of the transfer

240 8.2.5 GetSMBIOSStructureByType

241 The GetSMBIOSStructureByType command, described in Table 8, is used to get the SMBIOS structures
242 of a specific type. This command is defined to allow the SMBIOS structure data to be transferred using a
243 sequence of one or more command/response messages. When more than one command is used to
244 transfer the SMBIOS structure data, the response messages contain the non-overlapping contiguous
245 portions of SMBIOS structure data as defined in Table 2. By combining the portions of SMBIOS structure
246 data from the response messages, the entire SMBIOS structure data can be reconstructed.

247

Table 8 – GetSMBIOSStructureByType Command Format

Byte	Type	Request Data
0:3	uint32	DataTransferHandle A handle that is used to identify SMBIOS structure data transfer. This handle is ignored by the responder when the TransferOperationFlag is set to GetFirstPart.
4	enum8	TransferOperationFlag The operation flag that indicates whether this is the start of the transfer Possible values: {GetNextPart=0x00, GetFirstPart=0x01}
5	uint8	Type Specifies the type of the SMBIOS structures

6:7	uint16	StructureInstanceID A handle that is used to identify an instance of an SMBIOS structure of the specified type Special values: 0xFFFF – All instances of the specified type
Byte	Type	Response Data
0	enum8	CompletionCode Possible values: { PLDM_BASE_CODES, INVALID_DATA_TRANSFER_HANDLE=0x80, INVALID_TRANSFER_OPERATION_FLAG=0x81, NO_SMBIOS_STRUCTURES=0x86, INVALID_SMBIOS_STRUCTURE_TYPE=0x87, INVALID_SMBIOS_STRUCTURE_INSTANCE_ID=0x89 }
1:4	uint32	NextDataTransferHandle A handle that is used to identify the next portion of the transfer
5	enum8	TransferFlag The transfer flag that indicates what part of the transfer this response represents Possible values: {Start=0x01, Middle=0x02, End=0x04, StartAndEnd = 0x05}
Variable	–	SMBIOSStructureData See Table 2 for the format.

248 **8.2.6 GetSMBIOSStructureByHandle**

249 The GetSMBIOSStructureByHandle command, described in Table 9, is used to get the SMBIOS structure
 250 by a specific handle. This command is defined to allow the SMBIOS structure data to be transferred by
 251 using a sequence of one or more command/response messages. When more than one command is used
 252 to transfer the SMBIOS structure data, the response messages contain the non-overlapping contiguous
 253 portions of SMBIOS structure data as defined in Table 2. By combining the portions of SMBIOS structure
 254 data from the response messages, the entire SMBIOS structure data can be constructed.

255 **Table 9 – GetSMBIOSStructureByHandle Command Format**

Byte	Type	Request Data
0:3	uint32	DataTransferHandle A handle that is used to identify SMBIOS structure data transfer. This handle is ignored by the responder when the TransferOperationFlag is set to GetFirstPart.
4	enum8	TransferOperationFlag The operation flag that indicates whether this is the start of the transfer Possible values: {GetNextPart=0x00, GetFirstPart=0x01}
5:6	uint16	Handle Specifies the handle of the SMBIOS structure

Byte	Type	Response Data
0	enum8	CompletionCode Possible values: { PLDM_BASE_CODES, INVALID_DATA_TRANSFER_HANDLE=0x80, INVALID_TRANSFER_OPERATION_FLAG=0x81, INVALID_SMBIOS_STRUCTURE_HANDLE=0x88 }
1:4	uint32	NextDataTransferHandle A handle that is used to identify the next portion of the transfer
5	enum8	TransferFlag The transfer flag that indicates what part of the transfer this response represents Possible values: {Start=0x01, Middle=0x02, End=0x04, StartAndEnd = 0x05}
Variable	–	SMBIOSStructureData See Table 2 for the format.

256 8.3 PLDM for SMBIOS Data Transfer Version

257 The version of this PLDM for SMBIOS data transfer specification shall be 1.0.1 (major version number 1,
 258 minor version number 0, update version number 1, and no alpha version).

259 For the GetPLDMVersion command described in [DSP0240](#), the version of this specification is reported
 260 using the encoding as: 0xF1F0F100.

261 9 PLDM for SMBIOS Data Transfer Examples

262 This section provides examples of PLDM communications using the PLDM commands defined in this
 263 specification.

264 9.1 Multipart Transfers

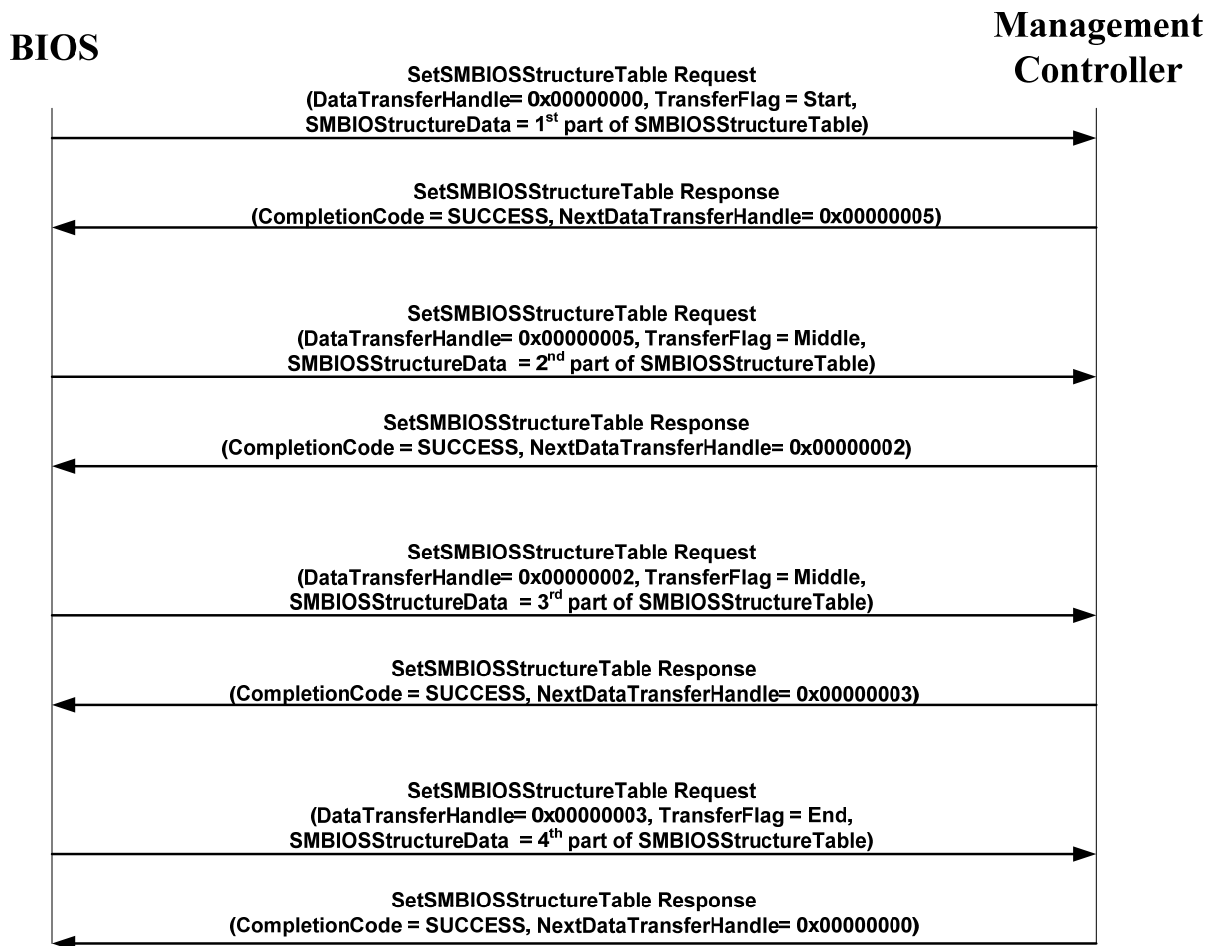
265 The commands defined in Section 8 for transferring SMBIOS structure table data or SMBIOS structure
 266 data support multipart transfers. The Get* and Set* commands use flags and data transfer handles to
 267 perform multipart transfers. A data transfer handle uniquely identifies the next part of the transfer. The
 268 data transfer handle values are implementation specific. For example, an implementation can use
 269 memory addresses or sequence numbers as data transfer handles. Following are some requirements for
 270 using TransferOperationFlag, TransferFlag, and DataTransferHandle for a given data transfer:

- 271 • For initiating a data transfer (or getting the first part of data) using a Get* command, the
 272 TransferOperationFlag shall be set to GetFirstPart in the request of the Get* command.
- 273 • For transferring a part other than the first part of data by using a Get* command, the
 274 TransferOperationFlag shall be set to GetNextPart and the DataTransferHandle shall be set to
 275 the NextDataTransferHandle that was obtained in the response of the previous Get* command
 276 for this data transfer.
- 277 • The TransferFlag specified in the request of a Set* command or the response of a Get*
 278 command has the following meanings:
 - 279 – Start, which is the first part of the data transfer

- 280 – Middle, which is neither the first nor the last part of the data transfer
- 281 – End, which is the last part of the data transfer
- 282 – StartAndEnd, which is the first and the last part of the data transfer
- 283 • The requester shall consider a data transfer complete when the TransferFlag in the response of
- 284 a Get* command is set to End or StartAndEnd.
- 285 • The responder shall consider a data transfer complete when the TransferFlag in the request of
- 286 a Set* command is set to End or StartAndEnd.

287 The following two examples show how the multipart transfers can be performed using the generic
 288 mechanism defined in the commands.

289 EXAMPLE 1: In this example, the MC maintains a copy of the SMBIOS structure table provided by the BIOS. The
 290 BIOS pushes a copy of its SMBIOS structure table to the MC by using the SetSMBIOSStructureTable command.
 291 Figure 1 shows the flow of the data transfer.



292

293 **Figure 1 – Multipart SMBIOS Structure Table Transfer Using the SetSMBIOSStructureTable**
 294 **Command**

295 EXAMPLE 2: In this example, the MC reads the SMBIOS structure table from the BIOS by using the
 296 GetSMBIOSStructureTable command. This example shows a pull model where the MC obtains a copy of the
 297 SMBIOS structure table from the BIOS. Figure 2 shows the flow of the data transfer.

Management Controller

BIOS

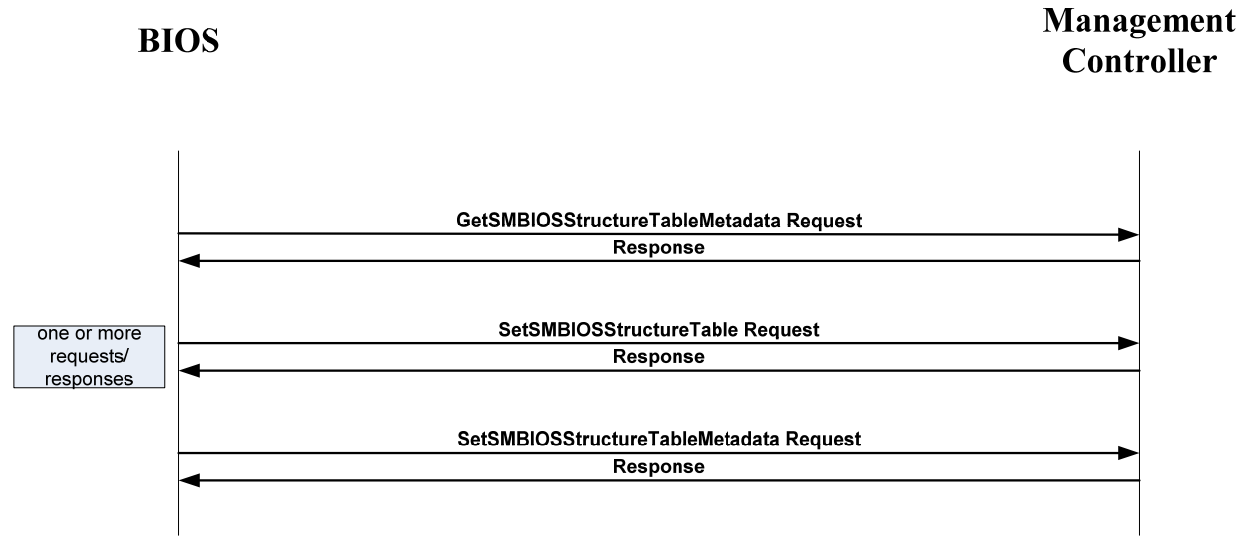


298

299 **Figure 2 – Multipart SMBIOS Structure Table Transfer Using the GetSMBIOSStructureTable**
 300 **Command**

301 9.2 SMBIOS Table Transfer from BIOS to MC Example

302 EXAMPLE: In this example, the BIOS sets the SMBIOS table on the MC. The BIOS first queries the SMBIOS
 303 table metadata by using the GetSMBIOSStructureTableMetadata command. The response from the MC to this
 304 command indicates that the MC does not have the latest SMBIOS structure table. Upon finding that the MC does not
 305 have the latest SMBIOS structure table, the BIOS transfers the latest SMBIOS structure table to the MC by using the
 306 SetSMBIOSStructureTable command. After transferring the latest SMBIOS structure table, the BIOS sets up the
 307 SMBIOS structure table metadata on the MC by using the SetSMBIOSStructureTableMetadata command. This
 308 example can be used in a push model where the MC is maintaining a copy of the SMBIOS structure table provided by
 309 the BIOS and the BIOS pushes to the MC a copy of the SMBIOS structure table by using SetSMBIOSStructureTable
 310 command. Figure 3 shows the data transfer.



311

312 **Figure 3 – Example of SMBIOS Table Transfer Using the SetSMBIOSStructureTable Command**

313

314

315
316
317
318
319

ANNEX A (Informative)

Change Log

Version	Date	Description
1.0.0	2009/4/23	DMTF Standard Release
1.0.1	2009/12/11	Erratum version to clarify that the integrity checksum in the metadata does not include pad bytes.

320