Document Identifier: DSP2061

Date: 2023-06-13

Version: 1.0.0

PLDM Accelerator Modeling

Supersedes: None

Document Class: Informational

Document Status: Published

Document Language: en-US

CONTENTS

Figures

Tables

Foreword

The *PLDM Accelerator Modeling* (DSP2061) document was prepared by the Platform Management Communications Infrastructure (PMCI) Working Group of the DMTF.

DMTF is a not-for-profit association of industry members dedicated to promoting enterprise and systems management and interoperability. For information about the DMTF, see https://www.dmtf.org.

Acknowledgments

The DMTF acknowledges the following individuals for their contributions to this document:

Editors:

- Rama Rao Bisa – Dell Technologies
- Pavan Kumar Gavvala – Dell Technologies

Contributors:

- Bob Stevens – Dell Technologies
- Hemal Shah – Broadcom Inc.
- Patrick Caporale – Lenovo
- Yuval Itkin – Nvidia
- Eliel Louzoun – Intel Corporation
- Ryan Weldon – Groq
- Deepak Kodihalli – Nvidia
- Pierre-Philippe Stevens – Advanced Micro Devices
- Michael Garner – Meta

Introduction

This document describes a modeling scheme for an Accelerator card using PLDM for Platform Monitoring and Control DSP0248 semantics.

Document conventions

Typographical conventions

The following typographical conventions are used in this document:

- Document titles are marked in *italics*.
- Important terms that are used for the first time are marked in *italics*.
- Terms include a link to the term definition in the “Terms and definitions“ clause, enabling easy navigation to the term definition.
- ABNF rules are in `monospaced font`.

ABNF usage conventions

Format definitions in this document are specified using ABNF (see RFC5234), with the following deviations:

- Literal strings are to be interpreted as case-sensitive Unicode characters, as opposed to the definition in RFC5234 that interprets literal strings as case-insensitive US-ASCII characters.

Reserved and unassigned values

Unless otherwise specified, any reserved, unspecified, or unassigned values in enumerations or other numeric ranges are reserved for future definition by the DMTF.

Unless otherwise specified, numeric or bit fields that are designated as reserved shall be written as 0 (zero) and ignored when read.

Byte ordering

Unless otherwise specified, byte ordering of multibyte numeric fields or bit fields is “Big Endian” (that is, the lower byte offset holds the most-significant byte, and higher offsets hold less-significant bytes).

Other Conventions

See ANNEX A for other conventions

1 Scope

This document defines an example data model for implementing the systems management of accelerators using PLDM for Platform Monitoring and Control DSP0248 semantics. This document establishes a common framework that can provide implementation consistency between a system's Management Controller and accelerators and accelerator cards the system contains, focusing on FPGAs and GPUs and similar devices that offload processing from the host CPU. This data model is assumed to be extensible to a variety of physical implementations and should not be construed to be limited to the examples herein.

Accelerators and Accelerator card implementations may include ancillary features such as networking and storage that have management schemas defined in other data models and Specifications. The management of those features is outside the scope of this data model. The data model provided here focuses on the management of the accelerator features of the card, but composite sensors that return overall card status for example, may include metadata from those other functional areas. For instance, it may be appropriate to use either DSP2054 or DSP0222 for the management of networking features that may be included on the accelerator or card.

2 Normative references

The following referenced documents are indispensable for the application of this document. For dated or versioned references, only the edition cited (including any corrigenda or DMTF update versions) applies. For references without a date or version, the latest published edition of the referenced document (including any corrigenda or DMTF update versions) applies. Unless otherwise specified, for DMTF documents this means any document version that has minor or update version numbers that are later than those for the referenced document. The major version numbers must match the major version number given for the referenced document.

DMTF DSP0222, Network Controller Sideband Interface (NC-SI) Specification 1.1, https://www.dmtf.org/sites/default/files/standards/documents/DSP0222_1.1.0.pdf

DMTF DSP0236, *MCTP Base Specification 1.3*, https://www.dmtf.org/sites/default/files/standards/documents/DSP0236_1.3.0.pdf

DMTF DSP0240, *Platform Level Data Model (PLDM) Base Specification 1.1*, https://www.dmtf.org/sites/default/files/standards/documents/DSP0240_1.1.0.pdf

DMTF DSP0241, *Platform Level Data Model (PLDM) Over MCTP Binding Specification 1.0*, https://www.dmtf.org/sites/default/files/standards/documents/DSP0241_1.0.0.pdf

DMTF DSP0245, *Platform Level Data Model (PLDM) IDs and Codes Specification 1.3*, https://www.dmtf.org/sites/default/files/standards/documents/DSP0245_1.3.0.pdf

DMTF DSP0248, *Platform Level Data Model (PLDM) for Platform Monitoring and Control Specification 1.2*, https://www.dmtf.org/sites/default/files/standards/documents/DSP0248_1.2.0.pdf

DMTF DSP0249, *Platform Level Data Model (PLDM) State Sets Specification 1.1, https://www.dmtf.org/sites/default/files/standards/documents/DSP0249_1.1.0.pdf*

DMTF DSP0257, *Platform Level Data Model (PLDM) FRU Data Specification 1.0,* https://www.dmtf.org/sites/default/files/standards/documents/DSP0257_1.0.0.pdf

DMTF DSP0267, *Platform Level Data Model (PLDM) for Firmware Update Specification 1.1,* https://www.dmtf.org/sites/default/files/standards/documents/DSP0267_1.1.0.pdf

DMTF DSP2054, *Platform Level Data Model (PLDM) NIC Modeling Specification 1.0,* https://dmtf.org/sites/default/files/standards/documents/DSP2054_1.0.0.pdf

IETF RFC2781, *UTF-16, an encoding of ISO 10646*, February 2000, https://www.ietf.org/rfc/rfc2781.txt

IETF STD63, *UTF-8, a transformation format of ISO 10646* https://www.ietf.org/rfc/std/std63.txt

IETF RFC4122, *A Universally Unique Identifier (UUID) URN Namespace*, July 2005, https://www.ietf.org/rfc/rfc4122.txt

IETF RFC4646, *Tags for Identifying Languages*, September 2006, https://www.ietf.org/rfc/rfc4646.txt

ISO 8859-1, *Final Text of DIS 8859-1, 8-bit single-byte coded graphic character sets — Part 1: Latin alphabet No.1,* February 1998

ISO/IEC Directives, Part 2, *Rules for the structure and drafting of International Standards,* https://www.iso.org/sites/directives/current/part2/index.xhtml

IETF RFC5234, *ABNF: Augmented BNF for Syntax Specifications, January 2008*, https://tools.ietf.org/html/rfc5234

3 Terms and definitions

In this document, some terms have a specific meaning beyond the normal English meaning. Those terms are defined in this clause.

The terms “shall” (“required”), “shall not”, “should” (“recommended”), “should not” (“not recommended”), “may”, “need not” (“not required”), “can” and “cannot” in this document are to be interpreted as described in ISO/IEC Directives, Part 2, Clause 7. The terms in parentheses are alternatives for the preceding term, for use in exceptional cases when the preceding term cannot be used for linguistic reasons. Note that ISO/IEC Directives, Part 2, Clause 7 specifies additional alternatives. Occurrences of such additional alternatives shall be interpreted in their normal English meaning.

The terms “clause”, “subclause”, “paragraph”, and “annex” in this document are to be interpreted as described in ISO/IEC Directives, Part 2, Clause 6.

The terms “normative” and “informative” in this document are to be interpreted as described in ISO/IEC Directives, Part 2, Clause 3. In this document, clauses, subclauses, or annexes labeled “(informative)” do not contain normative content. Notes and examples are always informative elements.

Refer to DSP0240 for terms and definitions that are used across the PLDM specifications.

4 Symbols and abbreviated terms

Refer to DSP0240 and DSP0248 for symbols and abbreviated terms that are used across the PLDM specifications. For the purposes of this document, the following additional symbols and abbreviated terms apply.

4.1

PCB

Printed Circuit Board

4.2

FPGA

Field Programmable Gate Array

4.3

GPU

Graphics Processing Unit

5 PLDM Accelerator Modeling overview

5.1 General

This document describes a hierarchical modeling scheme for an Accelerator card using PLDM for Platform Monitoring and Control DSP0248 semantics. The model is scalable, allowing consistent modeling of Accelerator cards with different configuration options such as the number of Accelerators.

While PLDM for Platform Monitoring and Control DSP0248 is a published standard, using the model defined in this document simplifies interoperability by establishing a consistent schema.

The basic format that is used for sending PLDM messages is defined in DSP0240. The format that is used for carrying PLDM messages over a transport-layer protocol and medium is given in companion documents to the base specification. For example, DSP0241 defines how PLDM messages are formatted and sent using MCTP as the transport.

The model supports the following:

- Consistent modeling of an Accelerator card regardless of the specific configuration and resource count
- Accelerator card hardware structure description
- Reporting of configuration changes such as firmware update

5.2 Model elements

5.2.1 PLDM terminus

PLDM for Platform Monitoring and Control DSP0248 defines a single root for every device instance, referred to as PLDM Terminus and identified with a TID. The term “MC” is used to identify a PLDM terminus which communicates with an Accelerator card throughout this document.

When there are multiple Accelerators assembled on the same card, there may be a single Accelerator which reports all the sensors of all the elements on the Accelerator card to the MC. Alternatively, each Accelerator in the Accelerator card may present a separate PLDM terminus.

PLDM for Platform Monitoring and Control DSP0248 does not allow associating components reported via different PLDM termini since every database is relative to a given PLDM terminus. To overcome this constraint, the implementers can retrieve a globally unique ID (Board part number and serial number) from each TID and recognize these TIDs belonging to the same Accelerator card. The process to retrieve the globally unique ID (Board part number and serial number) from each TID is outside of this document.

All PLDM IDs specified by the model in this document shall be consistent across all TIDs on a given card. This avoids conflict from duplication of IDs in the combined model, generated by merging the TID-specific model elements reported as part of the overall model.

5.2.2 Accelerator card

In this model, the Accelerator card is the top-level element of the hierarchy containing one or more Accelerators on a PCB. An Accelerator card is a hardware and software solution that offloads certain processing from the host processor. The Accelerator card in this document refers to various form factors and is represented with PLDM Entity ID code 68 for Add-in card. The Accelerator card may contain sensors.

5.2.3 Accelerator

In this model, an Accelerator is the second level element of the hierarchy containing one or more sensors. An Accelerator is a hardware device with a main function of offloading certain processing from the host processor. An Accelerator may contain sensors such as health state, power-consumption, and temperature.

5.2.4 Memory

The term “memory” in this document covers the internal memory of the Accelerator, memory chips installed on the PCB, and the DIMMs. In this model, the memory is at the second level of the hierarchy. A Memory may contain sensors such as temperature, health state, and error statistics.

5.2.5 Inter-Accelerator card connection

The Accelerator cards may support communication with each other. Figure 1 depicts an Inter-Accelerator card connection, and it may not be the only communication interface between Accelerator cards.

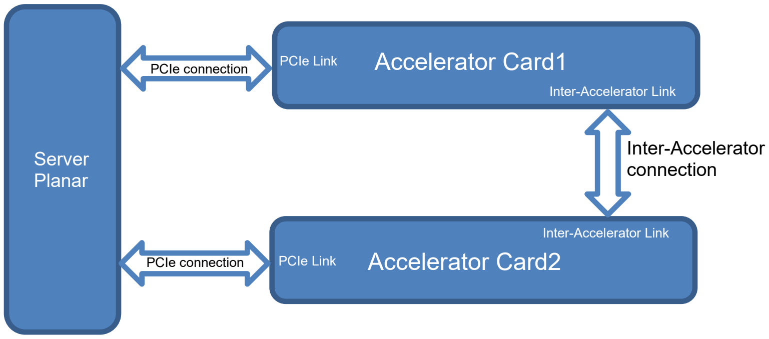


Figure 1 – Inter-Accelerator card connection

5.3 Model sensors

5.3.1 General

Attributes are reported by means of sensors. Numeric sensors are used to report specific measured attributes. State sensors report operational and/or health state. The default thresholds for all numeric sensors shall be set by the hardware vendor. The sensors can be associated with any entity such as the Accelerator card, Accelerator or Memory. The description of each sensor is applicable only for the implemented sensors and it is not mandatory to implement all the sensors described in this document. There may be auxiliary devices present on the accelerator card and each auxiliary device may present its own set of sensors.

Note: The Sensor Auxiliary Names PDR is recommended to provide the proper name of each sensor.

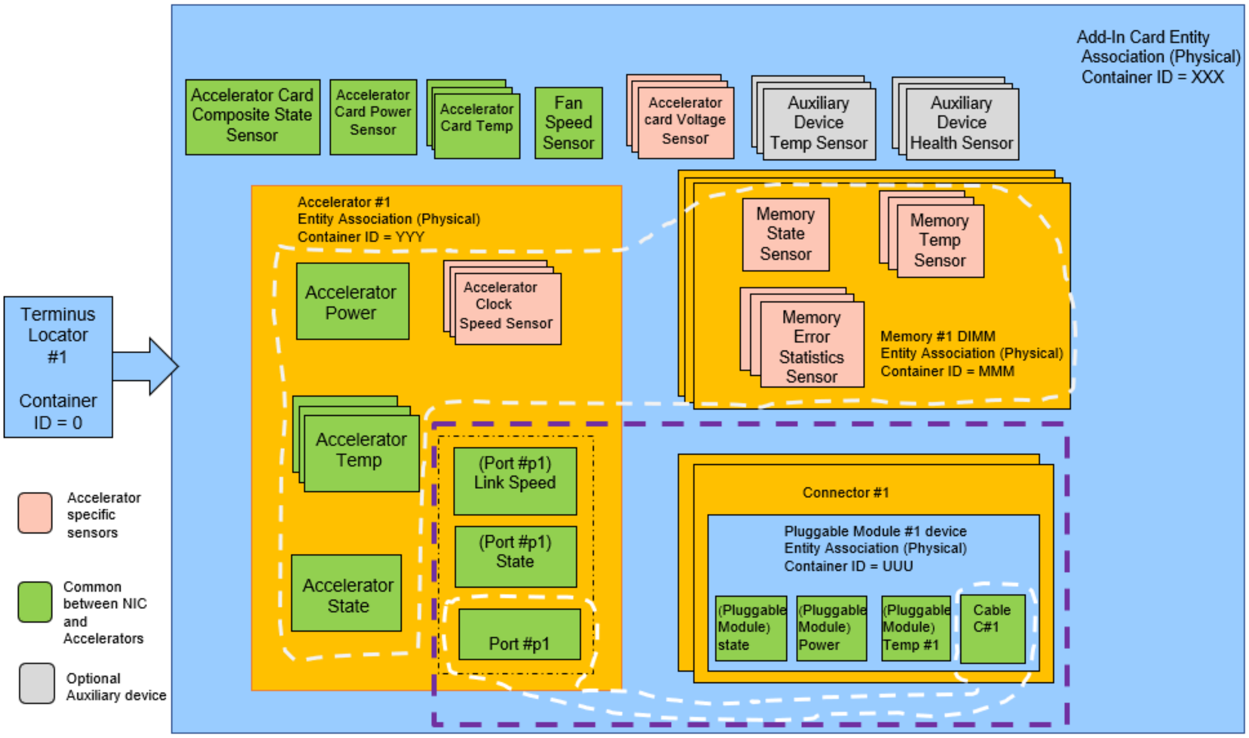

Add-In Card Entity Association (Physical) Container ID = XXX
Accelerator Card Composite State Sensor
Accelerator Card Power Sensor
Accelerator Card Temp
Fan Speed Sensor
Accelerator card Voltage Sensor
Auxiliary Device Temp Sensor
Auxiliary Device Health Sensor
Terminus Locator #1
Container ID = 0
Accelerator #1 Entity Association (Physical) Container ID = YYY
Accelerator Power
Accelerator Clock Speed Sensor
Accelerator Temp
Accelerator State
(Port #p1) Link Speed
(Port #p1) State
Port #p1
Memory State Sensor
Memory Temp Sensor
Memory Error Statistics Sensor
Memory #1 DIMM Entity Association (Physical) Container ID = MMM
Connector #1
Pluggable Module #1 device Entity Association (Physical) Container ID = UUU
(Pluggable Module) state
(Pluggable Module) Power
(Pluggable Module) Temp #1
Cable C#1
Accelerator specific sensors
Common between NIC and Accelerators
Optional Auxiliary device

Figure 2 – Accelerator card PLDM model diagram

5.3.2 Accelerator card temperature sensor

The temperature sensor on the Accelerator card reports the card's ambient temperature and is represented using a numeric sensor. There may be multiple temperature sensors installed on the Accelerator card.

5.3.3 Accelerator card power sensor

The power sensor on the Accelerator card reports the estimated or measured aggregate power consumption of the Accelerator card and is represented using a numeric sensor. An Accelerator card which cannot accurately report its real-time power consumption may report its estimated maximal power. When there are multiple Accelerators on the same Accelerator card, there may be no visibility by any Accelerator to the real-time information of the other Accelerators. For this reason, this sensor is only implemented when there is only one Accelerator on the Accelerator card, or when there is a hardware sensor which does allow measuring and reporting the total card power consumption or when the maximal estimated power is reported without being measured or when the accelerators can communicate with each other.

5.3.4 Accelerator card fan speed sensor

The fan speed sensor on the Accelerator card reports the speed of an active cooling fan and is represented using a numeric sensor. An Accelerator card may have multiple fans installed, each potentially with its own speed sensor.

5.3.5 Accelerator card voltage sensor

The voltage sensors on the Accelerator card report various voltages on the card and are represented using numeric sensors. There may be multiple voltage sensors installed on the card.

5.3.6 Accelerator card auxiliary device temperature sensor

The temperature sensor on the auxiliary device reports the ambient temperature of the auxiliary device and is represented using a numeric sensor. This document does not mandate having an auxiliary device temperature sensor.

5.3.7 Accelerator card auxiliary device health sensor

The health sensor on the auxiliary device reports the health state of the auxiliary device and is represented using a state sensor. This document does not mandate having an auxiliary device health sensor.

5.3.8 Accelerator card composite state sensor

The Accelerator card composite state sensor combines the Accelerator card thermal state sensor, the Memory operational fault state sensor, and the Accelerator card health state sensor. The Accelerator card health state is the aggregated health state of all the components on the card. The reported aggregated health state of the Accelerator card reflects the worst case of the reported health states for each of the elements monitored in the model. For example, if an Accelerator health state is non-critical and a memory heath state is critical, then the Accelerator card health state may be set to critical in the Accelerator card composite state sensor.

When there are multiple Accelerators, there may be no visibility by any Accelerator to the real-time information of other Accelerators. For this reason, this composite state sensor is only implemented when there is only a single Accelerator on the Accelerator card or when the Accelerator card has the needed visibility of all the components such as Accelerators and memory.

To determine the respective sensor states, the following steps shall be used: the accelerator card thermal state sensor shall also reflect the auxiliary device temperature and the accelerator card health state sensor shall also reflect the auxiliary device health state.

5.3.9 Accelerator temperature sensor

The temperature sensor of the Accelerator reflects the device temperature and is represented using a numeric sensor. This sensor is typically located in the thermally sensitive areas on the Accelerator.

5.3.10 Accelerator power sensor

The power sensor on the Accelerator reports the estimated or measured power consumption of the Accelerator and represented using a numeric sensor. An Accelerator which cannot accurately report its real-time power consumption may report its estimated maximal power.

5.3.11 Accelerator composite state sensor

The Accelerator composite state sensor combines the Accelerator Thermal trip state, Accelerator health state, Configuration valid state, Configuration change state, and Accelerator firmware version change state. The MC can use this sensor to identify issues with the Accelerator and to identify the specific maintenance operations that it needs to perform. These operations may include Accelerator reset, system-level shutdown for thermal protection, and other system-level maintenance.

Using the configuration change indication, the Accelerator notifies the MC to retrieve PDRs updated by the configuration change.

When a firmware update is detected, the composite state sensor can reflect this event to the MC, allowing the MC to take any action needed to respond to the update. Note that reading the new firmware version may be performed by the MC using protocols other than PLDM for Platform Monitoring and Control DSP0248, such as DSP0257 and/or DSP0267. Please note that firmware update only reflects the conclusion of the firmware programming operation; it is device-specific whether this detection additionally implies that new firmware is already active.

5.3.12 Accelerator clock speed sensor

The clock speed sensor of the Accelerator is used to read the clock speed and is represented using numeric sensors. An Accelerator may have multiple clock domains, each with its own clock speed sensor

5.3.13 Memory temperature sensor

The temperature sensors on the memory modules and internal memory report the memory temperatures and are represented using numeric sensors. There may be multiple memory temperature sensors installed on the internal memory, on the soldered memory, and on the DIMMs.

The memory that is soldered on the Accelerator card PCB may not have a temperature sensor on them. In this case, the implementations may choose to have a temperature sensor near the soldered memory chips calibrated to approximate the temperature of those memory devices.

5.3.14 Memory error statistics

The memory error statistics sensors report the memory error statistics (i.e., correctable errors and uncorrectable errors) and are represented using numeric sensors. Refer to the “sensorUnits enumeration” table in DSP0248.

5.3.15 Memory composite state sensor

The memory composite state sensor combines sensors such as memory health state sensor, memory cache state sensor, memory error state sensor, and memory redundant activity state sensor. The MC can use this sensor to identify issues with the memory and to identify the specific maintenance operations that it needs to perform. Refer to Table 11 (Memory-Related State Sets) of DSP0249 for all memory-related sensors and their states.

5.4 Hierarchy description of the Accelerator card model elements

5.4.1 General

PLDM Accelerator Modeling uses a hierarchical model. Refer to section 10 PLDM associations and section 11 Entity Association PDR of DSP0248 to understand physical and logical associations.

5.4.2 Physical entities association

Physical association is defined in DSP0248 as a method to associate components which are physically connected to each other. The model uses this concept to describe the following structures:

- Content of the Accelerator card PCB
- Content of the Accelerators
- Content of the Memory Modules

A hierarchy entity is defined using an entity association PDR identified with a unique ***containerID*** identifier parameter. The entity association PDR’s ***containerEntityContainerID*** references the PDR in which the entity is contained. This entity association PDR shall also contain the contained entities defined in DSP2054 for the elements shown inside the purple dotted line of Figure 2.

Figure 3 shows an example of how an Accelerator card entity association PDR references its container entity and contained entities:

Accelerator card Entity Association PDR

Container ID	100
Record Handle	1100

Container Entity		
Entity Type	68	Add-in card
Entity Instance Number	1	
Container Entity Container ID	0	System

Association Type	**Physical to Physical containment**

Contained Entity - Accelerator		
Entity Type	149	Accelerator
Entity Instance Number	1	
Container Entity Container ID	100	Accelerator card

Contained Entity - Memory		
Entity Type	66	Memory
Entity Instance Number	1	
Container Entity Container ID	100	Accelerator card

Figure 3 – Hierarchy description using containerEntityContainerID referencing the containedEntityContainerID

5.4.3 Logical entity association

The DSP0248 defines logical association as a method to associate components which collectively form a shared property yet are not physically part of the same component. This model uses logical association to describe the following structures:

Figure 4 shows logical association between an Accelerator and a memory module:

Channel #1 Entity Association PDR

Container ID	900
Record Handle	1180

Container Entity		
Entity Type	79	Processor/memory module (processor and memory together on a module)
Entity Instance Number	1	
Container Entity Container ID	100	Accelerator card

Association Type	**Logical containment**

Contained Entity - Accelerator		
Entity Type	149	Accelerator
Entity Instance Number	1	
Container Entity Container ID	100	Accelerator card

Contained Entity - Memory Module		
Entity Type	66	Memory module
Entity Instance Number	1	
Container Entity Container ID	100	Accelerator card

Figure 4 – Defining a logical association

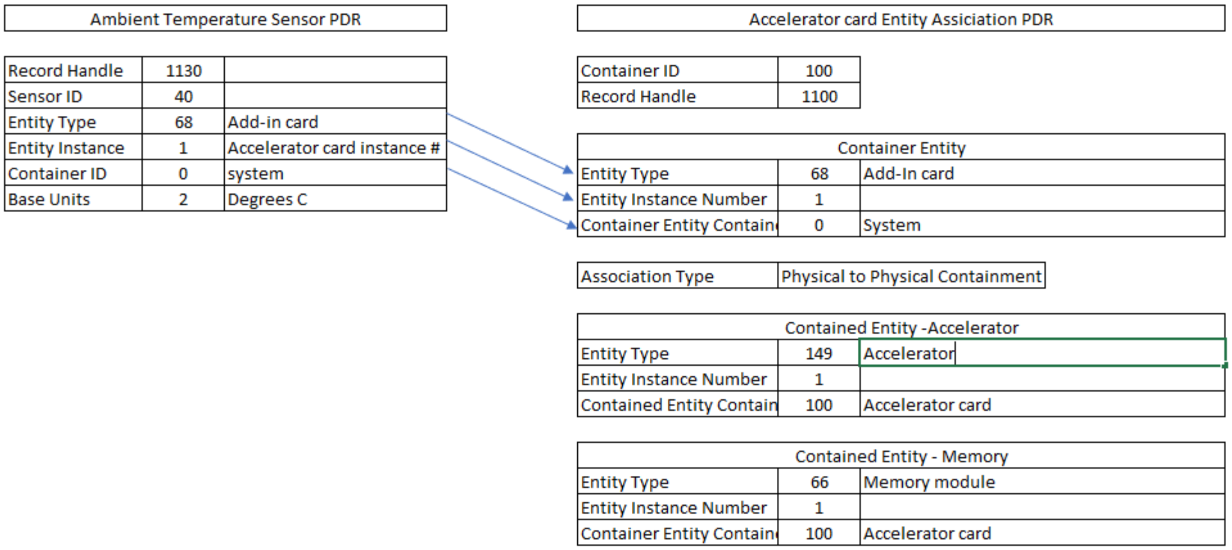
5.4.4 Sensor association

As per DSP0248, numeric and state sensors are not included inside entity association PDRs. They are instead associated to the measured entity by directly referencing the EntityContainerID, EntityType, and EntityInstanceNumber of the measured entity in an entity association PDR. A sensor is identified by a unique Sensor ID value.

5.4.4.1 Associating a sensor at the top level

When associating a sensor to the top-level entity which is the system the association uses the top-level ***containerEntityType containerEntityInstanceNumber***, and **containerEntityContainerID** parameters.

Figure 5 illustrates the association of a temperature sensor to the Accelerator card in the model.



Ambient Temperature Sensor PDR

Record Handle	1130	
Sensor ID	40	
Entity Type	68	Add-in card
Entity Instance	1	Accelerator card instance #
Container ID	0	system
Base Units	2	Degrees C

Accelerator card Entity Assiciation PDR

Container ID	100
Record Handle	1100

Container Entity		
Entity Type	68	Add-In card
Entity Instance Number	1	
Container Entity Contain	0	System

Association Type	Physical to Physical Containment

Contained Entity -Accelerator		
Entity Type	149	Accelerator
Entity Instance Number	1	
Contained Entity Contain	100	Accelerator card

Contained Entity - Memory		
Entity Type	66	Memory module
Entity Instance Number	1	
Container Entity Contain	100	Accelerator card

Figure 5 – Top-level sensor association

5.5 Element PLDM Type IDs

The model uses the following Type ID for each component in the model, selected from the available types defined in DSP0249. The following table lists the chosen Type IDs used in the model:

Table 1 – Type IDs used in the Accelerator card model

Component	Type ID
Accelerator card	68
Accelerator	149
Memory Module	66

5.6 Enumeration

5.6.1 General

PLDM for Platform Monitoring and Control DSP0248 uses enumerated IDs to define elements in the database. These IDs are labeled as:

- Container ID – unique for each container PDR in the model database
- Instance ID – unique for each entity type within a given hierarchy level
- Handle ID – unique ID for each PDR in the model database
- Sensor ID – unique for each sensor in the model database

The proposed model provides an example enumeration scheme for these IDs, allowing a reasonably scalable formulation. This model is only an example and implementations should not rely on these values.

5.6.2 Enumeration scheme

The model assumes some maximal limits to define the enumerated values. These limits are provided as an example and can be adjusted according to the specific Accelerator card requirements.

The example model enumeration is designed to support an Accelerator card that does not exceed the following limits:

Table 2 – Chosen enumeration limits in the model

Model Limit	Value
Max Accelerators	10
Max Memory Modules	10
Max board temperature sensors	10
Max temperature sensors per Accelerator	10

Note:

- If one of the above limits is insufficient for an Accelerator card, only the enumerated values will be affected and the model structure will not have to change.

Table 3 illustrates the enumeration scheme, calculated based on the above limits.

Table 3 – Example Enumeration Scheme with Type IDs

Item	Max count	Base Container	Max Container	Base Handle	Max Handle	Base Sensor ID	Max Sensor-ID	Base Instance	Max instance	Type-ID
Accelerator card	1	100		1100				1	1	68
Accelerator card Composite State Sensor	1			1101	1101	5	5	1	1	68
Accelerator card Power Sensor	1			1102	1102	6	6	1	1	68
Accelerator card Temperature sensors	10			1130	1139	20	29	1	10	68
Accelerator card fan speed sensor	10			1150	1159	40	49	1	10	68
Accelerator card Voltage sensor	10			1170	1179	80	89	1	10	68
Processor Memory Interface	10	900	909	1180	1189	90	99	1	10	68
Connectors	20	1040	1059	1190	1209	100	119	1	20	185
Memory module	10	1020	1029	1210	1219			1	10	66
Memory composite state sensor	1			1220	1220	120	120		1	66
Memory temperature sensor	20			1225	1244	125	144	1	20	66
Memory module correctable Errors	10			1255	1264	150	159		1	66
Memory module uncorrectable Errors	10			1275	1284	180	189		1	66
Accelerators	10	1000	1009	1295	1304			1	10	149
Accelerator power sensor	1			1310	1310	210	210		1	149
Accelerator State sensor	1			1315	1315	220	220		1	149
Accelerator temperature sensor	10			1325	1334	240	249	1	10	149
Accelerator clock speed sensor	10			1335	1344	260	269	1	10	149
Accelerators Ports	10			1345	1354	290	299	1	10	149
Accelerators Port State	10			1360	1369	320	329	1	10	149
Accelerators Link Speed	10			1380	1389	350	359	1	10	149
Auxiliary Device Temp Sensor	1			1395	1395	380	380		1	68
Auxiliary Device health sensor	1			1400	1400	395	395		1	68
Plugs	20	1070	1089	1410	1429	410	429	1	20	214
Plug Composite Sensor	1			1430	1430	450	450	1	1	214
Plug Power Sensor	20			1440	1459	470	489	1	20	214
Plug Temp Sensor	10			1470	1479	510	519	1	10	214
Cable	16							1	16	187
Communication Channel	100	800	899	1490	1589			1	100	79

Calculated	Model Constant	Model Sensors described in this doc	Common sensors for NIC and Accelerator	n/a

5.7 Model illustration

5.7.1 General

The Accelerator card PLDM model is a hierarchical model. The following subclauses describe the model for each of the hierarchy levels:

5.7.2 Accelerator Card

The Accelerator card top level may contain the PCB card, Accelerators, Memory modules, one or more thermal sensors, Accelerator card composite state sensor, Fan speed sensor, power sensor and voltage sensors. The PCB power consumption is represented with a power sensor. The Accelerator card operational state is represented by a composite state sensor. When there are multiple Accelerators on the same card, Accelerator card sensors are typically only reported by the first Accelerator. The Accelerator card is responsible for determining the order of accelerators in the card. Note that the top-level health state sensor of the composite state sensor may reflect the card level sensors and the health states of Accelerators.

Refer the purple dotted line in Figure 2 to the Network port link speed sensor, Network port link state sensor, Pluggable module temperature sensor, pluggable module power sensor and Pluggable module composite state sensor sections of DSP2054 specification for networking functionality.

5.7.3 Accelerator

The Accelerator hierarchy represents the active device (or one of multiple devices) that performs the Accelerator control interface. An Accelerator is represented as a collection of sensors.

5.7.4 Memory

The Memory hierarchy represents a memory device (or one of multiple devices). A Memory is represented as a collection of sensors.

5.8 Events

5.8.1 General

This model supports using PLDM events as a method to notify the MC upon changes in the sensor readings/states as described in DSP2048. The following example events can be used with the model and the implementation may choose to have more events.

5.8.2 Accelerator firmware version change

This event indicates to the MC that the firmware version of the Accelerator has changed. The MC may use the ***GetPDRRepositoryInfo*** command and check if the ***timestamp*** parameter value has changed since it last read the PDRs. The MC may update the whole PDR repository by re-reading all the PDRs. The value used for the ***timestamp*** can be a virtual time value initialized by the Accelerator at device initialization.

5.8.3 Health and state sensors events notifications

The sensors on the accelerator card may report a change in value, health, or state using a PLDM state or numeric sensor event. Providing such a notification can significantly shorten the response time, compared to waiting for the MC to poll the sensors, for an occurrence that requires the MC to take an action such as increasing the airflow from a cooling fan.

6 Model use example

6.1 General

The following example for modeling an Accelerator card using PLDM for Platform Monitoring and Control DSP0248 describes an Accelerator card with the following attributes:

- Accelerator Card
  - Temperature Sensor
  - State Sensor
  - Fan speed Sensor
  - Voltage Sensors
  - Power Sensor
  - Auxiliary Device Temperature Sensor
  - Auxiliary Device Health Sensor
- Accelerator
  - Temperature Sensor
  - Power Sensor
  - State Sensor
  - Clock speed Sensor
- Memory
  - Temperature Sensor
  - Memory State Sensor
  - Memory Error statistics Sensor

Figure 6 illustrates the model which is used in the example.

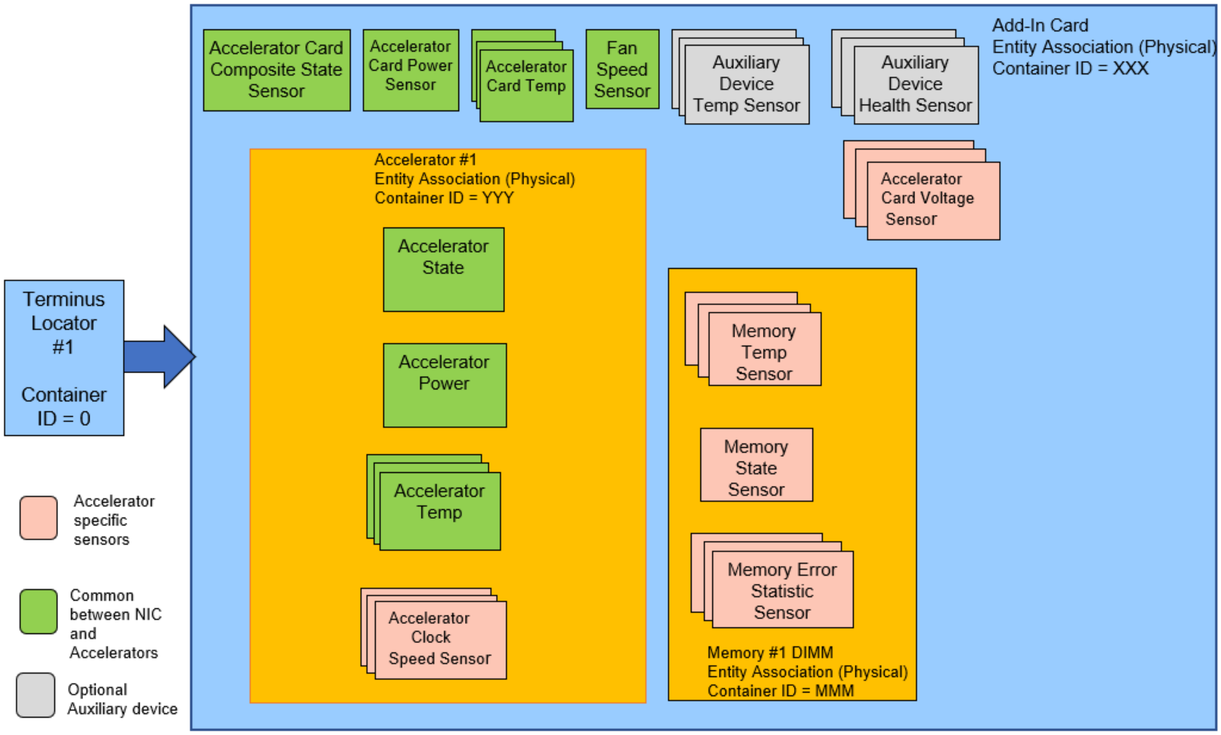

Accelerator Card Composite State Sensor
Accelerator Card Power Sensor
Accelerator Card Temp
Fan Speed Sensor
Auxiliary Device Temp Sensor
Auxiliary Device Health Sensor
Add-In Card Entity Association (Physical) Container ID = XXX
Accelerator #1 Entity Association (Physical) Container ID = YYY
Accelerator State
Accelerator Power
Accelerator Temp
Accelerator Clock Speed Sensor
Accelerator Card Voltage Sensor
Memory Temp Sensor
Memory State Sensor
Memory Error Statistic Sensor
Memory #1 DIMM Entity Association (Physical) Container ID = MMM
Terminus Locator #1
Container ID = 0
Accelerator specific sensors
Common between NIC and Accelerators
Optional Auxiliary device

Figure 6 – Example model diagram

6.2 Model hierarchy

The model PDRs identify the elements depicted in Figure 6. The hierarchies are illustrated in the following diagram. For simplicity, Figure 7 shows sensors of Accelerator and Memory Module.

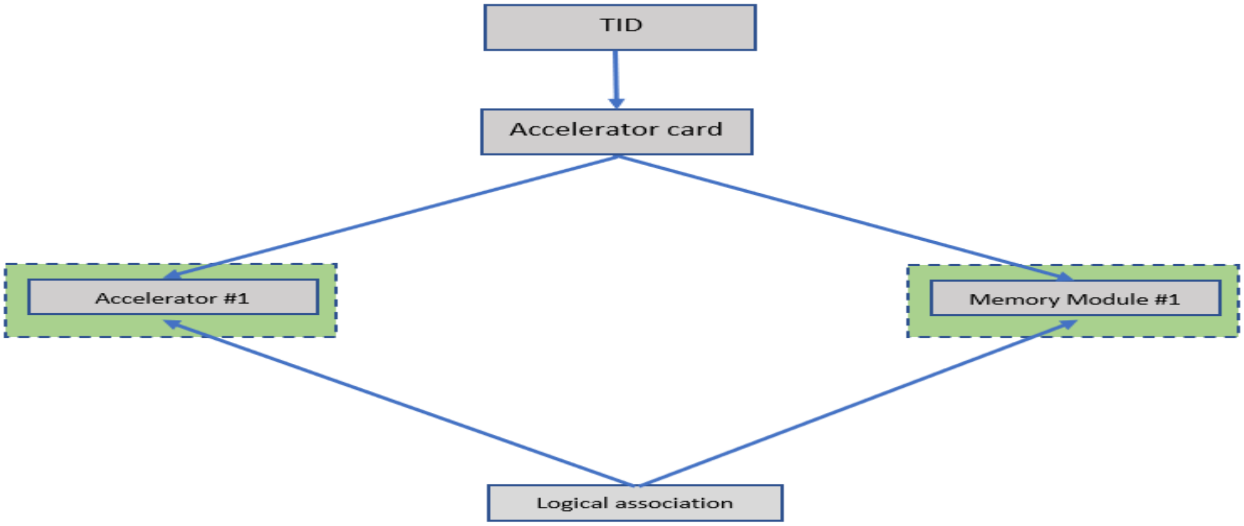


Figure 7 – Accelerator card model hierarchy

6.3 Top-level TID

The terminus ID is identified by the terminus locator PDR. The TID defines the top-level entry point to the PLDM model. Because there is only one Accelerator on the Accelerator card in this example, there is only one TID.

Table 4 – TID PDR

Field name	Value	Description
Container ID	0	System
TID		Assigned by MC
Record Handle	1100	Opaque number
Terminus Locator Size	1	Size of (EID) or size of (UID)
Terminus Locator Type	1	MCTP EID
EID	EID	MCTP assigned EID Value
UID	UID	Vendor provided UUID format value

The TID value is assigned to the terminus by the MC. When the transport layer is MCTP, the identification of the terminus is performed using the Endpoint ID (EID) value. When using PLDM over RBT, the terminus locator PDR shall use the UID (instead of EID). The UID value in the terminus locator PDR uses the device UUID value as the terminus UID. For more information regarding terminus locator PDR see DSP0248

6.4 Accelerator card

6.4.1 General

The top level of the model is the Accelerator card. The Accelerator card includes the physical elements which are an Accelerator (only one Accelerator in this example) and a memory module (only one memory module in this example).

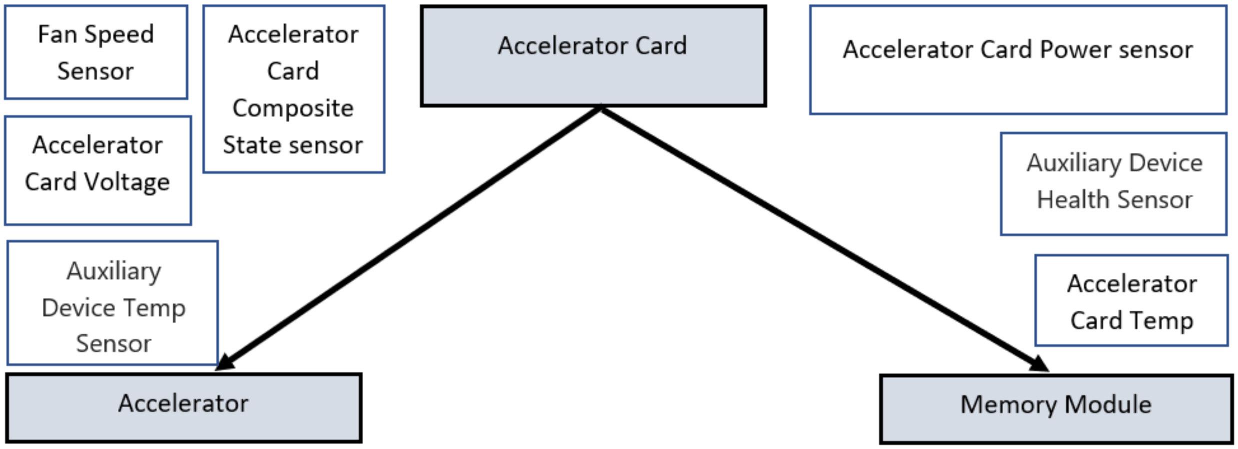

Fan Speed Sensor
Accelerator Card Composite State sensor
Accelerator Card
Accelerator Card Power sensor
Accelerator Card Voltage
Auxiliary Device Health Sensor
Auxiliary Device Temp Sensor
Accelerator Card Temp
Accelerator
Memory Module

Figure 8 – Accelerator card level elements

The sensors on the Accelerator card level are described using a reference to the measured entity, independent of the container that includes all the physical elements on the Accelerator card.

Accelerator card Entity Association PDR

Container ID	100
Record Handle	1100

Container Entity		
Entity Type	68	Add-In card
Entity Instance Number	1	
Container Entity Container ID	0	System

Association Type	Physical to Physical containment

Contained Entity – Accelerator		
Entity Type	149	Accelerator
Entity Instance Number	1	
Contained Entity Container ID	100	Accelerator card

Contained Entity – Memory		
Entity Type	66	Memory
Entity Instance Number	1	
Contained Entity Container ID	100	Accelerator card

Figure 9 – Accelerator card container PDR

Note that the Accelerator card container ID, 100, is referenced by the sensors not included in the entity association PDR. The enumeration model shown in

Table 3 includes the container ID for every hierarchy level.

6.4.2 Accelerator card power sensor

Field	Value	Description
Record Handle	1102	
Sensor ID	6	
Entity Type	68	Add-In card
Entity Instance	1	Accelerator card Instance #
Container ID	0	System
Base Unit	7	Watts
Unit Modifier	−1	0.1 watt resolution

Figure 10 – Accelerator card power sensor PDR

6.4.3 Accelerator card temperature sensor

Field	Value	Description
Record Handle	1130	
Sensor ID	20	
Entity Type	68	Add-In card
Entity Instance	1	Accelerator card Instance #
Container ID	0	System
Base Unit	2	Degrees Celsius
Unit Modifier	0	No need for scaling

Figure 11 – Ambient Temperature sensor PDR

6.4.4 Accelerator card fan speed sensor

Field	Value	Description
Record Handle	1150	
Sensor ID	40	
Entity Type	68	Add-In card
Entity Instance	1	Accelerator card Instance #
Container ID	0	System
Base Unit	19	RPM
Unit Modifier	0	No need for scaling

Figure 12 – Accelerator card fan speed sensor PDR

6.4.5 Accelerator card voltage sensor

Field	Value	Description
Record Handle	1170	
Sensor ID	80	
Entity Type	68	Add-In card
Entity Instance	1	Accelerator card Instance #
Container ID	0	System
Base Unit	5	Volts
Unit Modifier	−1	0.1 volt resolution

Figure 13 – Accelerator card voltage sensor PDR

6.4.6 Accelerator card auxiliary device temperature sensor

Field	Value	Description
Record Handle	1395	
Sensor ID	380	
Entity Type	68	Add-In card
Entity Instance	1	Accelerator card Instance #
Container ID	0	System
Base Unit	2	Degrees Celsius
Unit Modifier	0	No need for scaling

Figure 14 – Auxiliary device temperature sensor PDR

6.4.7 Accelerator card auxiliary device health sensor

Field	Value	Description
Record Handle	1400	
Sensor ID	395	
Entity Type	68	Add-In card
Entity Instance	1	Accelerator card Instance #
Container ID	0	System
Sensor Type	1	Health state
Possible States	Refer to Table 1 of DSP0249	

Figure 15 – Auxiliary device health sensor PDR

6.4.8 Accelerator card composite state sensor

Record Handle	1101	
Entity Type	68	Add-In card
Entity Instance Number	1	
Container Entity Container ID	0	System

Terminus Handle	0
Sensor ID	5
Composite Sensor Count	3

Sensor Type	1	Health state
Possible States	Refer to Table 1 of DSP0249	

Sensor Type	21	Thermal Trip
Possible States	Refer to Table 1 of DSP0249	

Sensor Type	10	Memory Operational Fault status
Possible States	Refer to Table 1 of DSP0249	

Figure 16 – Accelerator card composite state sensor PDR

6.5 Accelerator

6.5.1 General

The Accelerator is an active device and being a physical entity that doesn't include other entities, the Accelerator is not declared in its own PDR. It is instead declared in the Accelerator card container PDR. The Accelerator includes a set of device-level sensors. The following diagram illustrates the model sensors in the Accelerator:

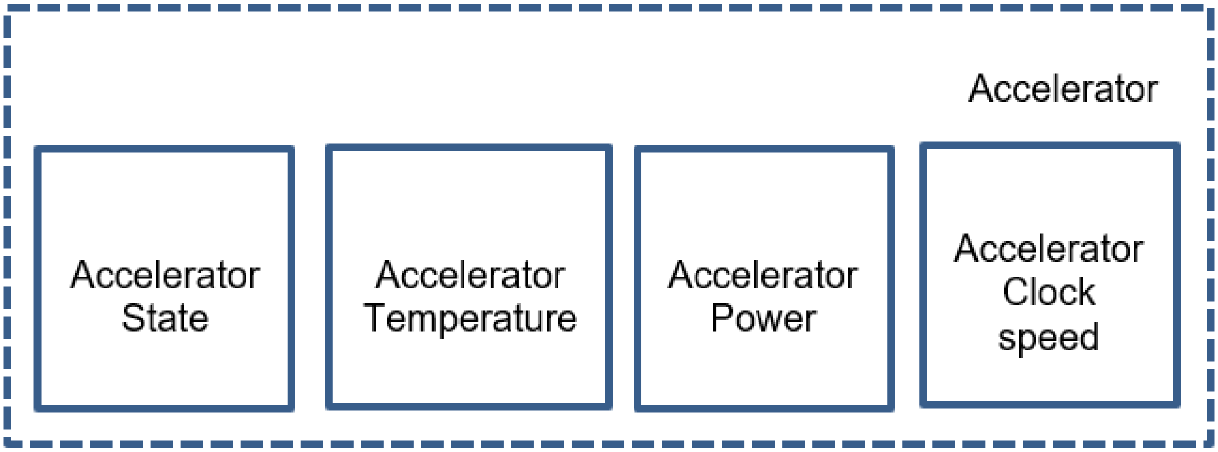


Figure 17 – Example model Accelerator

The Accelerator content is declared using an entity-association PDR that includes the hierarchical description of the Accelerator. The device-level sensors are declared with separate PDRs using direct references to the measured entities.

Container ID	1000
Record Handle	1295

Container Entity		
Entity Type	149	Accelerator
Entity Instance Number	1	
Container Entity Container ID	100	Accelerator card

Association Type	Physical to Physical containment

Figure 18 – Accelerator entity association PDR

6.5.2 Accelerator temperature sensor

Field	Value	Description
Record Handle	1325	
Sensor ID	240	
Entity Type	149	Accelerator
Entity Instance	1	Accelerator Instance #
Container ID	100	Accelerator card
Base Unit	2	Degrees Celsius

Figure 19 – Accelerator temperature sensor PDR

6.5.3 Accelerator power sensor

Field	Value	Description
Record Handle	1310	
Sensor ID	210	
Entity Type	149	Accelerator
Entity Instance	1	Accelerator Instance #
Container ID	100	Accelerator card
Base Unit	7	Watts
Unit Modifier	−1	0.1 watt resolution

Figure 20 – Accelerator power sensor PDR

6.5.4 Accelerator composite state sensor

Record Handle	1315	
Entity Type	149	Accelerator
Entity Instance Number	1	
Container Entity Container ID	100	Accelerator card

Terminus Handle	0
Sensor ID	220
Composite Sensor Count	5

Sensor Type	1	Health state
Possible States	Refer to Table 1 of DSP0249	

Sensor Type	21	Thermal Trip
Possible States	Refer to Table 1 of DSP0249	

Sensor Type	18	Firmware Version
Possible States	Refer to Table 1 of DSP0249	

Sensor Type	15	Configuration
Possible States	Refer to Table 1 of DSP0249	

Sensor Type	16	Configuration Change
Possible States	Refer to Table 1 of DSP0249	

Figure 21 – Accelerator composite state sensor PDR

6.5.5 Accelerator clock speed sensor

Field	Value	Description
Record Handle	1335	
Sensor ID	260	
Entity Type	149	Accelerator
Entity Instance	1	Accelerator Instance #
Container ID	100	Accelerator Card
Base Unit	20	Hertz
Unit Modifier	6	1 MHz resolution

Figure 22 – Accelerator card clock speed sensor PDR

6.6 Memory

6.6.1 General

The Memory is a physical entity in the model. The Memory is already declared within the Accelerator card container PDR. The Memory includes a set of device-level sensors. The Memory sensors cover all three types of memory i.e., DIMM, Internal memory and soldered memory chips. The following diagram illustrates the model sensors in the Memory:

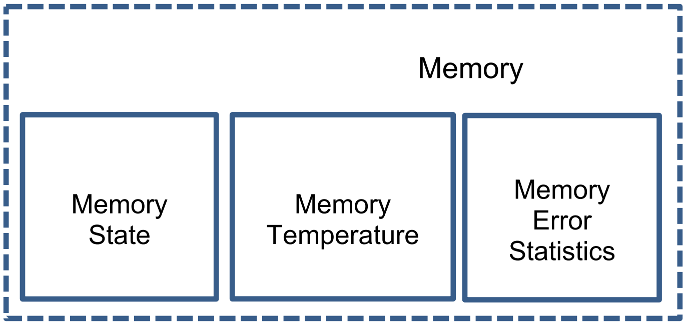


Figure 23 – Example Memory model

The Memory content is declared using an entity-association PDR that includes the hierarchical description of the Memory. The device-level sensors are declared with separate PDRs using direct references to the measured entities.

Container ID	1020
Record Handle	1210

Container Entity		
Entity Type	66	Memory
Entity Instance Number	1	
Container Entity Container ID	100	Accelerator card

Association Type	Physical to Physical containment

Figure 24 – Memory association PDR

6.6.2 Memory temperature sensor

Field	Value	Description
Record Handle	1225	
Sensor ID	125	
Entity Type	66	Memory
Entity Instance	1	Memory Instance #
Container ID	100	Accelerator card
Base Unit	2	Degrees C

Figure 25 – Memory temperature sensor PDR

6.6.3 Memory error statistics sensors

Field	Value	Description
Record Handle	1255	
Sensor ID	150	
Entity Type	66	Memory
Entity Instance	1	Memory instance #
Container ID	100	Accelerator card
Base Unit	80	Correctable Errors

Figure 26 – Memory correctable errors PDR

Field	Value	Description
Record Handle	1275	
Sensor ID	180	
Entity Type	66	Memory
Entity Instance	1	Memory Instance #
Container ID	100	Accelerator card
Base Unit	81	Uncorrectable Errors

Figure 27 – Memory uncorrectable errors PDR

6.6.4 Memory composite state sensor

Memory composite state sensor PDR		
Record Handle	1220	
Entity Type	66	Memory
Entity Instance Number	1	
Container Entity Container ID	100	Accelerator card

Terminus Handle	0
Sensor ID	120
Composite Sensor Count	4

Sensor Type	1	Health state
Possible States	Refer to Table 1 of DSP0249	

Sensor Type	320	Memory cache status
Possible States	Refer to Table 11 of DSP0249	

Sensor Type	321	Memory error status
Possible States	Refer to Table 11 of DSP0249	

Sensor Type	322	Redundant Memory activity status
Possible States	Refer to Table 11 of DSP0249	

Figure 28 – Memory composite state sensor PDR

ANNEX A
(informative)
Notation and conventions

A.1 Notations

Examples of notations used in this document are as follows:

- 2:N In field descriptions, this will typically be used to represent a range of byte offsets starting from byte two and continuing to and including byte N. The lowest offset is on the left; the highest is on the right.
- (6) Parentheses around a single number can be used in message field descriptions to indicate a byte field that may be present or absent.
- (3:6) Parentheses around a field consisting of a range of bytes indicates the entire range may be present or absent. The lowest offset is on the left; the highest is on the right.
- PCIe Underlined, blue text is typically used to indicate a reference to a document or specification called out in "Normative references" clause or to items hyperlinked within the document.
- rsvd This case-insensitive abbreviation is for "reserved."
- [4] Square brackets around a number are typically used to indicate a bit offset. Bit offsets are given as zero-based values (that is, the least significant bit [LSb] offset = 0).
- [7:5] This notation indicates a range of bit offsets. The most significant bit is on the left; the least significant bit is on the right.
- `1b` The lowercase "`b`" following a number consisting of `0`s and `1`s is used to indicate the number is being given in binary format.
- `0x12A` A leading "`0x`" is used to indicate a number given in hexadecimal format.

ANNEX B
(informative)

Change log

Version	Date	Description
1.0.0	5/25/2022	Initial draft
1.0.0	6/13/2023	Released for publication