Intel® Optane™ Persistent Memory

OS Provisioning Specification

February 2021

Revision 1.01

Document Number: 634430

Notice: This document contains information on products in the design phase of development. The information here is subject to change without notice. Do not finalize a design with this information.

Intel technologies features and benefits depend on system configuration and may require enabled hardware, software, or service activation. Learn more at intel.com, or from the OEM or retailer.

No computer system can be absolutely secure. Intel does not assume any liability for lost or stolen data or systems or any damages resulting from such losses.

You may not use or facilitate the use of this document in connection with any infringement or other legal analysis concerning Intel products described herein. You agree to grant Intel a non-exclusive, royalty-free license to any patent claim thereafter drafted which includes subject matter disclosed herein.

No license (express or implied, by estoppel or otherwise) to any intellectual property rights is granted by this document.

The products described may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

This document contains information on products, services and/or processes in development. All information provided here is subject to change without notice. Contact your Intel representative to obtain the latest Intel product specifications and roadmaps.

Intel disclaims all express and implied warranties, including without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement, as well as any warranty arising from course of performance, course of dealing, or usage in trade.

Copies of documents which have an order number and are referenced in this document may be obtained by calling 1-800-548-4725 or by visiting www.intel.com/design/literature.htm.

Intel, Intel Optane, and the Intel logo are trademarks of Intel Corporation or its subsidiaries.

*Other names and brands may be claimed as the property of others.

Contents

Figures

Tables

Revision History

Document Number	Revision Number	Description	Date
634430	1.01	• Updated *NVDIMM DSM Interface Specification* references to *Intel® Optane™ PMem DSM Interface Specification* • Added to the Introduction a brief description of Near Memory and Far Memory and their relationship to Memory Mode • Updated acronyms list	February 2021
634430	1.00	• Initial release	December 2020

1 Introduction

This document describes how to support provisioning Intel® Optane™ persistent memory (Intel® Optane™ PMem) 100 and 200 series for use in different operating modes using OS software.

PMem module capacity may be provisioned for use in Memory Mode (MM) or App Direct (AD). Provisioning a mix of the two modes is not supported by this documentation.

Provisioning PMem modules is a joint responsibility between the software and the platform Unified Extensible Firmware Interface (UEFI) Firmware (FW). The software issues a provisioning request by writing configuration metadata on the PMem modules. The platform UEFI FW reads the metadata from the PMem modules during the next boot and uses the information to map the DDR and PMem resources into the system address space. Note that in Memory Mode some portion of DDR in the system is consumed as a cache that fronts the PMem modules. This DDR memory as a cache is called Near Memory (NM) and the memory in the PMem modules that it fronts is called Far Memory (FM).

Full provisioning support for Intel® Optane™ PMem is provided by Intel management tooling, see https://github.com/intel/ipmctl. This document describes the interfaces and flows for OS native tool vendors to implement basic provisioning for Intel® Optane™ PMem. PMem tools provided by Intel are recommended to support complex configurations, such as mixed modes and to troubleshoot provisioning issues.

When provisioning PMem modules, the following general flow is used:

1. Request the module partition information using the pass-through _DSM.
2. Set the memory allocation goal using the pass-through _DSM.
3. Reboot.
4. Platform UEFI FW reads the information provided in the memory allocation goal (that is, a Configuration Input [CIN_]) and provides a response (that is, a Configuration Output [COUT]):
    a. The response can be verified by reading platform configuration data using the pass-through _DSM.
5. If the platform UEFI FW accepts the memory allocation goal as valid, the Current Configuration (CCUR) is updated to match the allocation request.

These steps are elaborated on in later sections of this document.

1.1 Acronyms

The following table includes the definition of acronyms used in this document.

Table 1-1. Acronyms

Acronym	Definition
2LM	Two-Level Memory
ACPI	Advanced Configuration and Power Interface
AD	App Direct
CCUR	Current Configuration
CIN_	Configuration Input
COUT	Configuration Output
DPA	Device Physical Address
DSM	Device Specific Method
FM	Far Memory
GUID	Globally Unique Identifier
iMC	Integrated Memory Controller
MM	Memory Mode
NM	Near Memory
OEM	Original Equipment Manufacturer
PCAT	Platform Configuration Attributes Table
PCD	Platform Configuration Data
PM	Persistent Memory
PMem	Intel® Optane™ persistent memory
SMM	System Management Mode
UEFI	Unified Extensible Firmware Interface

1.2 Reference Documents

The following table includes the documents referenced within this document.

Table 1-2. Reference Documents

Document	Location
ACPI Specification 6.3 or later	https://www.uefi.org
Intel® Optane™ PMem DSM Interface	https://pmem.io

2 Platform Configuration Attributes

The Platform Configuration Attributes Table (PCAT) is created by the platform firmware during boot and presented as one of the Advanced Configuration and Power Interface (ACPI) System Description Tables. If the platform supports PMem modules, the PCAT is provided to inform software what revision of Platform Configuration Data tables (described in the next section) to use for provisioning requests. Table 2-1 shows the format of the PCAT header, and the Revision field within it specifies which revision of Platform Configuration Data tables to use for provisioning requests.

Table 2-1. PCAT Header

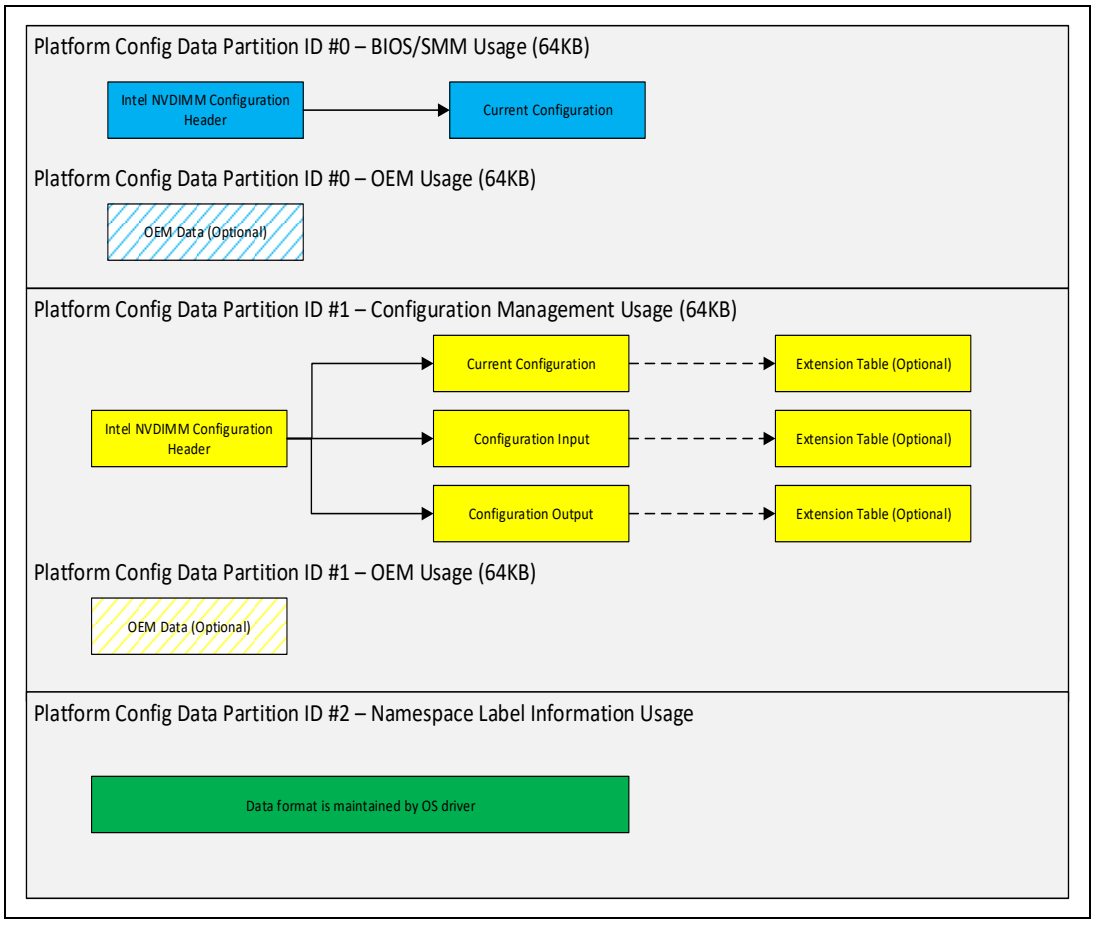
Field	Byte Length	Byte Offset	Description
Header			
Signature	4	0	"PCAT" is the signature for this table.
Length	4	4	Length in bytes for the entire table. The length implies the number of entry fields at the end of the table.
Revision	1	8	Bits[7:4]: Major revision Bits[3:0]: Minor revision **Note:** Supported revisions are 0.2 and 1.2.
Checksum	1	9	Entire table must sum to zero.
OEMID	6	10	OEM ID.
OEM Table ID	8	16	The table ID is the manufacturer model ID.
OEM Revision	4	24	OEM revision of table for the supplied OEM table ID.
Creator ID	4	28	Vendor ID of utility that created the table.
Creator Revision	4	32	Revision of utility that created the table.
Reserved	4	36	To make the structures 8-byte aligned.
Body			
Reserved	_	40	Reserved.

3 Platform Configuration Data

The module Platform Configuration Data (PCD) refers to a section of the PMem module that is used to store metadata. The metadata stored in the PCD is the architected interface between software and platform firmware to support PMem provisioning.

This section defines the format of the PCD. The PMem firmware supports commands to read and write three PCD partitions as shown in Figure 3-1. The details of the commands are defined in Section 4.

Figure 3-1. Platform Configuration Data



The first 64 KB of partition ID #0 is used by the platform firmware to store the current configuration information. This allows the platform firmware to restore the previous configuration if partition ID #1 is corrupted. The remaining 64 KB of partition ID #0 is reserved for OEM usage. Partition ID #0 should not be written to by software.

The first 64 KB of partition ID #1 is used by the software and the platform firmware to coordinate configuration change information. Partition ID #1 is

accessed by the software through the OS vendor-specific pass-through Device Specific Method (_DSM) function. The remaining 64 KB of partition ID #1 is reserved for OEM usage. This section for OEM usage must be preserved by the software when doing read or write operations.

The 128 KB for partition ID #2 is used to store the namespace label information through the OS vendor-specific pass-through _DSM function.

3.1 Configuration Header

The configuration header contains pointers to the current configuration, configuration input, and configuration output as shown in the following table. The configuration header starts at offset 0 of partition #0 and partition #1.

When reading a configuration header, it is important to read the revision field so that the data within the tables pointed to by the configuration header can be properly decoded using the appropriate revisions of the various tables (which are described in the following sections).

When writing a configuration header as part of a provisioning request, the revision field should match the revision field that was read from the PCAT. This ensures the platform firmware can understand the provisioning request since different revisions use slightly different tables. At this time, only two revisions (0.2 and 1.2) are supported and described in this document.

As an important side note, while every table within this document has a 0.2 description, the configuration header described up next is inconsistent and uses 0.1 instead of 0.2. As a result of this, in the case that 0.2 is read from the PCAT, the configuration header must use 0.1 while the other tables referenced within it use 0.2.

Table 3-1. Configuration Header (0.2 and 1.2)

Field	Byte Length	Byte Offset	Description
Header			
Signature	4	0	"DMHD" is the signature for this table.
Length	4	4	Length in bytes for the entire table.
Revision	1	8	Bits[7:4]: Major revision Bits[3:0]: Minor revision Supported revisions 0.1: Used with PCAT revision 0.2 1.2: Used with PCAT revision 1.2
Checksum	1	9	Entire table must sum to zero.
OEMID	6	10	OEM ID.
OEM Table ID	8	16	The table ID is the manufacturer model ID.

Field	Byte Length	Byte Offset	Description
OEM Revision	4	24	OEM revision of table for supplied OEM table ID.
Creator ID	4	28	Vendor ID of utility that created the table. Each tool should define their own unique ID. "INTL" is used by the Intel® Optane™ PMem Memory Command Line (IPMCTL) tool.
Creator Revision	4	32	Revision of the utility that created the table.
Body			
Current Configuration Data Size	4	36	Size of the data area allocated to the current configuration information in bytes.
Current Configuration Start Offset	4	40	Offset from the start of the partition that points to the current configuration information. This data is valid only if the size is non-zero.
Configuration Input Data Size	4	44	Size of the data area allocated to the configuration request area in bytes. Software could change the size.
Configuration Input Start Offset	4	48	Offset from the start of the partition that points to the configuration request data area. This data is valid only if the size is non-zero. Software could change the offset.
Configuration Output Data Size	4	52	Size of the data area allocated to the configuration response area in bytes.
Configuration Output Start Offset	4	56	Offset from the start of the partition that points to the configuration response data area. This data is valid only if the size is non-zero.

3.2 Current Configuration

The CCUR, as shown in the following table, is created by the platform firmware and updated on each PMem module during the memory configuration phase of the platform firmware.

When reading the CCUR table, it is important to read the revision field so that the data within the PCD table structures can be properly decoded using the appropriate revisions of the various tables (which are described in the following sections).

Table 3-2. Current Configuration (0.2 and 1.2)

Field	Byte Length	Byte Offset	Description
Header			
Signature	4	0	"CCUR" is the signature for this table.
Length	4	4	Length in bytes for entire table. The length implies the number of entry fields at the end of the table.
Revision	1	8	Bits[7:4]: Major revision

Field	Byte Length	Byte Offset	Description
			Bits[3:0]: Minor revision **Note:** Supported revisions are 0.2 and 1.2.
Checksum	1	9	Entire table must sum to zero.
OEMID	6	10	OEM ID.
OEM Table ID	8	16	The table ID is the manufacturer model ID.
OEM Revision	4	24	OEM revision of table for supplied OEM table ID.
Creator ID	4	28	Vendor ID of the utility that created the table. Each tool should define their own unique ID. "INTL" is used by the IPMCTL tool.
Creator Revision	4	32	Revision of the utility that created the table.
Body			
Configuration Status	2	36	Error Status: 0 – Undefined 1 – Success 2 – Reserved 3 – All the modules in the interleave set are not found. The volatile memory is mapped to the System Physical Address (SPA) if possible. Interleave set missing or no valid CCUR/CIN_ table present. 4 – The persistent memory is not mapped due to matching interleave set not found or invalid module location. The volatile memory is mapped to the SPA if possible. 5 – Reserved. 6 – New configuration input structures have errors, old configuration used. Refer to the configuration output structures for additional errors. 7 – New configuration input structures have errors. The volatile memory is mapped to the SPA if possible. Refer to the configuration output structures for additional errors. 8 – Configuration input checksum not valid. 9 – Configuration input data revision is not supported. 10 – PCD partition #0 current configuration checksum not valid. 11 - Module is not mapped to SPA due to a health issue or configuration change. 12 – Module is not mapped due to a population issue. 13 – Memory mapping failed because the Near Memory to Far Memory ratio (NM:FM) is not supported.

Field	Byte Length	Byte Offset	Description
			14 – Module is not mapped due to a violation of the CPU maximum memory limit. Other values reserved. 15 – Module persistent memory mapped, but volatile memory (if any) is not mapped due to a population issue. Other values reserved.
Reserved	2	38	
Volatile Memory Size Mapped into SPA	8	40	Total amount of volatile Two-Level Memory (2LM) size from the module mapped into the SPA.
Persistent Memory Size Mapped into SPA	8	48	Total amount of Persistent Memory (PM) size from the module mapped into the SPA.
PCD Table Structures[n]	Variable	56	A list of PCD table structures for this implementation. Only types 3 and 5 are used in this table.

3.3 Configuration Input

The CIN_, as shown in the following table, represents a configuration request created by the software. The platform firmware processes this table on the next system reboot and provides the response in the COUT table.

When reading the CIN_ table, it is important to read the revision field so that the data within the PCD table structures can be properly decoded using the appropriate revisions of the various tables (which are described in the following sections).

When writing the CIN_ table as part of a provisioning request, the revision field should match the revision field that was read from the PCAT. This ensures the platform firmware can understand the provisioning request since different revisions use slightly different tables. At this time, only two revisions (0.2 and 1.2) are supported and described in this document.

Table 3-3. Configuration Input (0.2 and 1.2)

Field	Byte Length	Byte Offset	Description
Header			
Signature	4	0	"CIN_" is the signature for this table.
Length	4	4	Length in bytes for the entire table. The length implies the number of entry fields at the end of the table.
Revision	1	8	Bits[7:4]: Major revision Bits[3:0]: Minor revision **Note:** Supported revisions are 0.2 and 1.2.

Field	Byte Length	Byte Offset	Description
Checksum	1	9	Entire table must sum to zero.
OEMID	6	10	OEM ID.
OEM Table ID	8	16	The table ID is the manufacturer model ID.
OEM Revision	4	24	OEM revision of table for supplied OEM table ID.
Creator ID	4	28	Vendor ID of the utility that created the table. Each tool should define their own unique ID. "INTL" is used by the IPMCTL tool.
Creator Revision	4	32	Revision of the utility that created the table.
Body			
Sequence Number	4	36	Request sequence number. The platform firmware copies this sequence number to the response structure once the platform firmware completes processing the request input. **Note:** To determine the sequence number to use in CIN_, the sequence number from the current CIN_ and COUT should be read. The higher of these two sequence numbers should be incremented by one, and that incremented value should be used as the sequence number for the new CIN_ request. In case that neither a current CIN_ or COUT exist or the maximum sequence number has been reached, then a sequence number of "1" should be used.
Reserved	8	40	
Variable Body			
PCD Table Structures[n]	Variable	48	A list of PCD table structures for this implementation. Only types 3, 4, and 5 are used in this table.

3.4 Configuration Output

The COUT, as shown in the following table, is created by the platform firmware in response to the software request input CIN_ table. The platform firmware copies the sequence number in the CIN_ table to the COUT table and provides the response for each request table type from the CIN_ table. Because of this behavior, a configuration request is known to have been processed by the platform firmware when the CIN_ sequence number matches the COUT sequence number. The success or failure of the processed configuration request can be determined based on the values contained in the validation status field of the COUT table and the configuration status field of the CCUR table.

When reading the COUT table, it is important to read the revision field so that the data within the PCD table structures can be properly decoded using the

appropriate revisions of the various tables (which are described in the following sections).

Table 3-4. Configuration Output (0.2 and 1.2)

Field	Byte Length	Byte Offset	Description
Header			
Signature	4	0	"COUT" is the signature for this table.
Length	4	4	Length in bytes for entire table. The length implies the number of entry fields at the end of the table.
Revision	1	8	Bits[7:4]: Major revision Bits[3:0]: Minor revision **Note:** Supported revisions are 0.2 and 1.2.
Checksum	1	9	Entire table must sum to zero.
OEMID	6	10	OEM ID.
OEM Table ID	8	16	The table ID is the manufacturer model ID.
OEM Revision	4	24	OEM revision of table for supplied OEM table ID.
Creator ID	4	28	Vendor ID of the utility that created the table. Each tool should define their own unique ID. "INTL" is used by the IPMCTL tool.
Creator Revision	4	32	Revision of the utility that created the table.
Body			
Sequence Number	4	36	Copy of the configuration input structure sequence number to indicate that the platform firmware has completed processing the input structure.
Validation Status	1	40	Platform firmware response after configuration input processing: 0 – Undefined. 1 – Configuration change applied successfully. 2 – Boot time processing complete, errors found. See Section 3.7 and Section 3.8 for additional error details. 6 – Goal not applied because the NM:FM ratio is not supported. 7 – Goal not applied due to a violation of the CPU maximum memory limit. 8 – Goal not applied due to a population issue. Other values reserved.
Reserved	7	41	
Variable Body			
PCD Table Structures[n]	Variable	48	A list of PCD table structures for this implementation. Only types 3, 4, and 5 are used in this table.

3.5 Platform Configuration Data Table Types

Table 3-5. PCD Structure Types (0.2 and 1.2)

Value	Description
0	Reserved
1	Reserved
2	Reserved
3	Configuration management attributes extension table
4	Partition size change table
5	Interleave Information Table
6-0xFFFF	Reserved

3.6 Configuration Management Attributes Extension Table (Type 3)

The Configuration Management Attributes Extension Table allows OEMs to define vendor-specific data to be used for management software extensions.

The table format allows the extension of configuration data information by adding Type 3 (Configuration Management Attributes Extension Table) structures to the following structures:

1. Current Configuration
2. Configuration Input
3. Configuration Output

The format of the management information extension table is shown in the following table.

Table 3-6. Configuration Management Attributes Extension Table (0.2 and 1.2)

Field	Byte Length	Byte Offset	Description
Type	2	0	3 - Configuration Management Attributes Extension Table.
Length	2	2	Length in bytes for the entire table.
Reserved	2	4	-
Vendor ID	2	6	Vendor ID of the generator of the Globally Unique Identifier (GUID) that maintains the format for the GUID data.
GUID	16	8	Vendor-specific GUID.
GUID Data	Variable	24	GUID data size must be 8-byte aligned. The GUID data contains the vendor-specific data.

3.7 Partition Size Change Table (Type 4)

The Partition Size Change Table, as shown in the following table, is used for changing the module partition size, and it is used in the configuration input and output structures.

Table 3-7. Partition Size Change Table (0.2 and 1.2)

Field	Byte Length	Byte Offset	Description
Header			
Type	2	0	4 - Partition Size Change Table.
Length	2	2	Length in bytes for the entire table.
Body			
Partition Size Change Status	4	4	Valid only on the configuration output structure: Byte [1-0]: Change status 0 – Undefined 1 – Success 2 – Reserved 3 – All the modules in the interleave set not found 4 – Matching interleave set not found (matching modules found) 5 – Total partition size defined in the interleave set exceeds the partition size input 6 – Module FW returned error 7 - Insufficient CPU resources available to map all the modules in the interleave set Byte [3-2]: Module FW error response code All other enumerations reserved.
Persistent Memory Partition Size	8	8	Size of persistent memory partition in bytes.

3.8 Interleave Information Table (Type 5)

The Interleave Information Table, as shown in the following tables, is used in the configuration input and output as well as the current configuration.

Note: There are different revisions of this table. The revision that should be used to decode this table corresponds to the specified revision in the structure (such as CCUR, CIN_, or COUT) in which the table is contained.

Table 3-8. Interleave Information Table (0.2)

Field	Byte Length	Byte Offset	Description
Header			
Type	2	0	5 – Module Interleave Information Table.

Field	Byte Length	Byte Offset	Description
Length	2	2	Length in bytes for the entire table.
Body			
Interleave Set Index	2	4	Logical index number for the interleave set. The value should be non-zero, and it should be the same for all the modules in the interleave set. It must also be unique across independent interleave sets.
Number of Modules in the Interleave Set	1 (n)	6	Total number of modules participating in this interleave set. The module identification information structure at the end of this structure is repeated for each module in the interleave set.
Interleave Memory Type	1	7	1 – Reserved 2 – App Direct mode All other enumerations are reserved.
Interleave Size	2	8	Byte [0]: Channel interleave size to be used for this interleave set. Byte [1]: Integrated Memory Controller (iMC) interleave size to be used for this interleave set. Can be hard-coded to 0x4040 for provisioning.
Interleave Ways	2	10	Byte[1-0]: Number of channel ways Bit[0]: 1-way Bit[1]: 2-way Bit[2]: 3-way Bit[3]: 4-way Bit[4]: 6-way Bit[5]: 8-way Bit[6]: 12-way Bit[7]: 16-way Bit[8]: 24-way Bit[15:9]: Reserved **Note:** The bit associated with the number of modules participating in the interleave set should be set.
Reserved	1	12	Reserved zero.
Interleave Change Status	1	13	Configuration input: Reserved zero Configuration output: 0 – Undefined 1 – Success

Field	Byte Length	Byte Offset	Description
			2 – Information not processed: indicates that the Interleave Information Table (Type 5) was not processed due to an error that is not described by a more specific error code. For example, if the interleave format is not valid, the interleave change status will be set to "2" (information not processed). 3 – All the modules in the interleave set not found. 4 – Matching interleave set not found (matching modules found). 5 – Insufficient CPU resources to map all the modules in the interleave set. This interleave set may not be mapped to system address space. 6 – Memory mapping failed due to not enough system physical address space available to map all the modules. 7 – Reserved. 8 – Partitioning request failed: indicates that the configuration request included a Partition Size Change Table (Type 4) and that the partitioning request failed. 9 – Matching modules found, but CIN_ is missing in a module in the interleave list. 10 – Channel interleave does not match between the iMCs being interleaved. 11 – Logical offset from the base of the partition or size in bytes contributed by this module is not properly aligned. Indicates that the partition offset or partition size is not a multiple of 1GiB. All other enumerations reserved.
Reserved	10	14	
Module Identification Information	n x 64	24	Module interleave set information. This field contains one entry of the Module Identification Information for Interleave Set table for each module in the interleave set. In other words, the value of "n" (that is, the number of module identification information entries) is determined by the number of modules in the interleave set as specified earlier in the table. **Note:** The order the Module Identification Information for Interleave Set tables are placed in this field is important and based on the module's location in the socket.

Field	Byte Length	Byte Offset	Description
			For most interleave sets, the order is determined as follows: Modules are first ordered by channel number (with lower channel numbers placed first), and modules with the same channel number are then ordered by iMC number (with lower iMC numbers placed first). Here is an example order for a four-way interleave set. Interleave set example (four-way): [Channel 0, iMC 0] [Channel 0, iMC 1] [Channel 1, iMC 0] [Channel 1, iMC 1] For six-way interleave sets (which can occur in sockets that have two iMCs, each with three channels), the order is determined as follows: Modules are first ordered by "(channel number + iMC number) modulus 2" (with lower modulus values placed first), and modules with the same modulus value are then ordered by channel number (with lower channel numbers placed first). Here is an example order for a six-way interleave set. Interleave set example (six-way): [Channel 0, iMC 0] [Channel 1, iMC 1] [Channel 2, iMC 0] [Channel 0, iMC 1] [Channel 1, iMC 0] [Channel 2, iMC 1]

Table 3-9. Interleave Information Table (1.2)

Field	Byte Length	Byte Offset	Description
Header			
Type	2	0	5 – Interleave Information Table.
Length	2	2	Length in bytes for the entire table.
Body			
Interleave Set Index	2	4	Logical index number for the interleave set. The value should be non-zero, and it should be the same for all the modules in the interleave set. It must also be unique across independent interleave sets.

Field	Byte Length	Byte Offset	Description
Number of Modules in the Interleave Set	1 (n)	6	Total number of modules participating in this interleave set. The module identification information structure at the end of this structure is repeated for each module in the interleave set.
Interleave Memory Type	1	7	1 – Reserved 2 – App Direct mode All other enumerations are reserved.
Interleave Size	2	8	Byte [0]: Channel interleave size to be used for this interleave set. Byte [1]: iMC interleave size to be used for this interleave set. Can be hard-coded to 0x4040 for provisioning.
Reserved	3	10	
Interleave Change Status	1	13	Configuration input: Reserved zero Configuration output: 0 – Undefined 1 – Success 2 – Information not processed: indicates that the Interleave Information Table (Type 5) was not processed due to an error that is not described by a more specific error code. For example, if the interleave format is not valid, the interleave change status will be set to “2” (information not processed). 3 – All the modules in the interleave set not found. 4 – Matching interleave set not found (matching modules found). 5 – Insufficient CPU resources to map all the modules in the interleave set. This interleave set may not be mapped to system address space. 6 – Memory mapping failed due to not enough system physical address space available to map all the modules. 7 – Reserved. 8 - Partitioning request failed: indicates that the configuration request included a Partition Size Change Table (Type 4) and that the partitioning request failed. 9 - Matching modules found, but CIN_ is missing in a module in the interleave list. 10 - Channel interleave does not match between the iMCs being interleaved.

Field	Byte Length	Byte Offset	Description
			11 - Logical offset from the base of the partition or size in bytes contributed by this module is not properly aligned. Indicates that the partition offset or partition size is not a multiple of 1GiB. 12 - Request is unsupported. All other enumerations reserved.
Reserved	10	14	
Module Identification Information	n x 64	24	Module interleave set information. **Note:** The order the module identification information for interleave set tables are placed in this field is important and based on the module's location in the socket. See this same field in the 0.2 revision of this table for details on this ordering.

3.8.1.1 Module Identification Information for Interleave Set

The Module Identification Information for Interleave Set Structure, as shown in the following tables, is used in the Interleave Information Table.

Note: There are different revisions of this table. The revision that should be used to decode this table corresponds to the specified revision in the structure (such as CCUR, CIN_, or COUT) in which the table is contained.

Table 3-10. Module Identification Information for Interleave Set (0.2)

Field	Byte Length	Byte Offset	Description
Body			
Module Unique Identifier	9	0	Unique, non-repeatable 9-byte module identifier obtained from data in the NVDIMM Control Region Structure from the NVDIMM Firmware Interface Table (NFIT) in the *ACPI Specification*. Example: Subsystem Vendor ID: 0x8980 Manufacturing location: 0xa1 Manufacturing date: 0x119 Serial number: 0xb5000000 Resulting unique identifier: 8089-a1-1901-000000b5
Reserved	23	9	
Partition Offset	8	32	Logical offset in bytes from the base of the PM partition. **Note:** This value should always be set to "0". Setting this value to anything other than "0" is beyond the scope of this document.

Field	Byte Length	Byte Offset	Description
Partition Size	8	40	Size in bytes contributed by this module for this interleave set. This size must be aligned to a 1-GiB size.

Table 3-11. Module Identification Information for Interleave Set (1.2)

Field	Byte Length	Byte Offset	Description
Body			
Module Unique Identifier	9	0	Unique, non-repeatable 9-byte module identifier obtained from data in the NVDIMM Control Region Structure from the NFIT table in the *ACPI Specification*. Example: Subsystem Vendor ID: 0x8980 Manufacturing location: 0xa1 Manufacturing date: 0x119 Serial number: 0xb5000000 Resulting unique identifier: 8089-a1-1901-000000b5
Module Location	8	9	Populated by software as part of the CIN_ request. Consists of the NFIT device handle from the NVDIMM Region Mapping Structure in the *ACPI Specification*. Used by the platform firmware to determine if the module location has changed, and if so, return an error in CCUR. Used by the software to provide guidance on how to correct issues related to module movement. Byte[0] - Socket ID Byte[1] - Die ID Byte[2] - iMC ID Byte[3] - Channel ID Byte[4] - Slot ID Byte[5-7] - Reserved Each ID is relative to the immediate parent. The mapping from the NFIT device handle to the previous bytes is as follows: Byte[0] = NFIT device handle Bits[19:12] (node and socket ID) Byte[1] = 0 Byte[2] = NFIT device handle Bits[11:8] (Mem Controller ID) Byte[3] = NFIT device handle Bits[7:4] (Mem Channel Num) Byte[4] = NFIT device handle Bits[3:0] (DIMM Num) Byte[5-7] = 0
Reserved	15	17	
Partition Offset	8	32	Logical offset in bytes from the base of the PM partition.

Field	Byte Length	Byte Offset	Description
			Note: This value should always be set to "0". Setting this value to anything other than "0" is beyond the scope of this document.
Partition Size	8	40	Size in bytes contributed by this module for this interleave set. This size must be aligned to a 1-GiB size.

4 Firmware Commands

The following Intel vendor-specific commands support reading and writing the PCD and partitioning the PMem module capacity between volatile and persistent regions.

Intel vendor-specific commands are sent to the PMem modules using the pass-through command _DSM (function index 9) as defined in the *Intel® Optane™ PMem DSM Interface Specification*.

The _DSM payloads for each command are described in this section.

4.1 Get Platform Configuration Data

This command retrieves the platform configuration data stored on a PMem module.

Note: The data from the module is received using a small payload interface that retrieves data in chunks of 128 bytes.

The three PCD regions are designated as follows:

Table 4-1. Configuration Data Partition IDs

Partition ID	Description
0	Platform firmware - Only writable via the platform firmware mailbox
1	Configuration data
2	Namespace label storage area

Table 4-2. Get Platform Configuration Data Input Format

Field	Byte Length	Byte Offset	Description
Header			
OpCode	4	0	0x0601
OpCode Parameters Data Length	4	4	6 bytes
Module Payload			
Partition ID	1	8	See Table 4-1. Hard-coded to "1" for provisioning.
Command Options	1	9	Command options: These are alternate options that can be passed to change the behavior of the command.

<table>
<tr><th>Field</th><th>Byte Length</th><th>Byte Offset</th><th>Description</th></tr>
<tr><td></td><td></td><td></td><td>
<table>
<tr><th>Bit</th><th>Description</th></tr>
<tr><td>0</td><td>Payload type: This designates which payload area that is requested to be used for the data transfer.
0 - Reserved
1 - Small payload (128-byte payload)</td></tr>
<tr><td>1</td><td>Retrieve option: Specifies the data that you want to retrieve.
0 - Partition data
1 - Partition size</td></tr>
<tr><td>7:2</td><td>Reserved</td></tr>
</table>
</td></tr>
<tr><td>Offset</td><td>4</td><td>10</td><td>Offset in bytes of the partition to start reading from (partition in question is relative to the platform configuration area and the partition selected in the partition ID byte).
Note: Data or offset must be on a 128-byte boundary.</td></tr>
</table>

Table 4-3. Get Platform Configuration Data Output Format – Partition Size

Field	Byte Length	Byte Offset	Description
Header			
Status	2	0	Defined in the *Intel® Optane™ PMem DSM Interface Specification*
Extended Status	2	2	Reserved
Output Data Length	4	4	4 bytes
Module Payload			
Partition Size	4	8	Size in bytes for the partition

Table 4-4. Get Platform Configuration Data Output Format – Partition Data

Field	Byte Length	Byte Offset	Description
Header			
Status	2	0	Defined in the *Intel® Optane™ PMem DSM Interface Specification*
Extended Status	2	2	Reserved

Field	Byte Length	Byte Offset	Description
Output Data Length	4	4	128 bytes
Module Payload			
Partition Data	128	8	The data from the partition

4.2 Set Platform Configuration Data

This command writes the platform configuration data to a PMem module. Data is sent to the module using a small payload interface that sends data in chunks of 128 bytes. However, because part of the module payload is consumed by data other than the raw platform configuration data (such as partition ID and offset), only 64 bytes of raw platform configuration data are sent with each command.

Table 4-5. Set Platform Configuration Data Input Format

Field	Byte Length	Byte Offset	Description
Header			
OpCode	4	0	0x0701
OpCode Parameters Data Length	4	4	128 bytes
Module Payload			
Partition ID	1	8	See Table 4-1. Hard-coded to "1" for provisioning.
Payload Type	1	9	Payload type: This designates which payload area that is requested to be used for the data transfer. • 0 – Reserved • 1 – Small payload (128-byte payload)
Offset	4	10	Offset in bytes of the selected partition to start writing to. **Note:** This value must be 64-byte aligned.
Reserved	58	14	
Data	64	72	The data that will be written to the partition.

Table 4-6. Set Platform Configuration Data Output Format

Field	Byte Length	Byte Offset	Description
Header			
Status	2	0	Defined in the *Intel® Optane™ PMem DSM Interface Specification*
Extended Status	2	2	Reserved
Output Data Length	4	4	0 bytes

4.3 Get Module Partition Information

This command retrieves information about how the PMem module capacity is partitioned between the volatile and persistent memories.

Table 4-7. Get Module Partition Information Input Format

Field	Byte Length	Byte Offset	Description
Header			
OpCode	4	0	0x0602
OpCode Parameters Data Length	4	4	0 bytes

Table 4-8. Get Module Partition Information Output Format

Field	Byte Length	Byte Offset	Description
Header			
Status	2	0	Defined in the *Intel® Optane™ PMem DSM Interface Specification*
Extended Status	2	2	Reserved
Output Data Length	4	4	36 bytes

Field	Byte Length	Byte Offset	Description
Module Payload			
Volatile (2LM) Capacity	4	8	Volatile (2LM) capacity: PMem volatile memory capacity (4-KiB units).
Reserved	4	12	
Volatile Start	8	16	Device Physical Address (DPA) start address of the 2LM region.
Persistent (PM) Capacity	4	24	PMem persistent memory capacity (4-KiB units). Special values: • 0x00000000 - no persistent capacity
Reserved	4	28	
PM Start	8	32	DPA start address of the PM region.
Raw Capacity (RC)	4	40	The maximum capacity, in 4-KiB units, of the PMem module that can be allocated for use in any of the supported modes (MM or AD).

5 Provisioning Examples

This section provides examples of how to provision PMem modules for use in different operation modes.

Use the general flow provided in Section 1 as reference. Steps 1 and 3 through 5 from the general flow remain constant. Step 2 changes depending on the PMem provisioning mode request, and for each provisioning mode an example is given here.

When providing a provisioning request, the configuration header that is used for the request contains offsets and sizes for CCUR, CIN_, and COUT. The CCUR offset and size point to the current active configuration (that is, the current CCUR block should be copied and used in the new provisioning request). The CIN_ offset and size point to the newly created provisioning request. The COUT offset and size can be cleared and set to "0".

In the following examples, if the configuration header size plus the CCUR and CIN_ size exceeds 64 bytes, the command would need to be adjusted to provide the data in 64-byte chunks. The offset would also need to be adjusted for each command sent. These examples use a default offset of "0".

Also, when setting the memory allocation goal, the pass-through _DSM for each PMem module in the platform must be called so that the metadata is written to every PMem module in the platform and is ready to be consumed by the platform firmware on reboot.

5.1 100% Memory Mode

When provisioning a module to be 100% memory mode, the set memory allocation goal step in the general provisioning flow would need the following data in the pass-through _DSM (See Section 4.2).

Table 5-1. 100% Memory Mode - Set Platform Configuration Data

Field	Byte Length	Byte Offset	Value
Header			
OpCode	4	0	0x0701
OpCode Parameters Data Length	4	4	128 bytes
Module Payload			
Partition ID	1	8	1

Field	Byte Length	Byte Offset	Value
Payload Type	1	9	1
Offset	4	10	0
Reserved	58	14	
Data	64	72	• Configuration header (See Section 3.1) • CCUR (See Section 3.2) – Current active configuration • COUT (See Section 3.4) – Cleared • CIN_ (See Section 3.3) – PCD Type 4 ○ Persistent memory partition size = 0

5.2 100% App Direct x1

When provisioning a module to be 100% App Direct x1 (that is, an interleave size of one), the set memory allocation goal step in the general provisioning flow would need the following data in the pass-through _DSM (See 4.2).

Table 5-2. 100% App Direct x1 - Set Platform Configuration Data

Field	Byte Length	Byte Offset	Value
Header			
OpCode	4	0	0x0701
OpCode Parameters Data Length	4	4	128 bytes
Module Payload			
Partition ID	1	8	1
Payload Type	1	9	1
Offset	4	10	0
Reserved	58	14	
Data	64	72	• Configuration header (See Section 3.1) • CCUR (See Section 3.2) – Current active configuration • COUT (See Section 3.4) – Cleared • CIN_ (See Section 3.3)

Field	Byte Length	Byte Offset	Value

			– PCD Type 4
			  ○ Persistent memory partition size = RC (obtained from the get module partition information firmware command)
			– PCD Type 5
			  ○ Interleave set index = 1
			  ○ Number of modules in the interleave set = 1
			  ○ Interleave memory type = 2 (AD)
			  ○ Interleave Size = 0x4040
			  ○ Module identification information (one entry)

5.3 100% App Direct Interleaved

When provisioning a module to be 100% App Direct interleaved (in this example with two modules in the interleave set), the set memory allocation goal step in the general provisioning flow would need the following data in the pass-through _DSM (See Section 4.2). When interleaving PMem modules, an interleave cannot span across CPU sockets or nodes. In other words, all interleaves must be contained within a single socket or node.

Note that to locate all the PMem modules in the platform, the NFIT table (as defined in the *ACPI Specification*) can be parsed to find all the NVDIMM Region Mapping Structures, and there is one of these structures per PMem module in the platform. In order to determine which socket or node each PMem module is connected to, the NFIT device handle contained in the associated NVDIMM Region Mapping Structure can be decoded (as defined in the *ACPI Specification*).

Table 5-3. 100% App Direct Interleaved - Set Platform Configuration Data

Field	Byte Length	Byte Offset	Value
Header			
OpCode	4	0	0x0701
OpCode Parameters Data Length	4	4	128 bytes
Module Payload			
Partition ID	1	8	1
Payload Type	1	9	1

Field	Byte Length	Byte Offset	Value
Offset	4	10	0
Reserved	58	14	
Data	64	72	• Configuration header (See Section 3.1) • CCUR (See Section 3.2) – Current active configuration • COUT (See Section 3.4) – Cleared • CIN_ (See Section 3.3) – PCD Type 4 ○ Persistent memory partition size = RC (obtained from the get module partition information firmware command) – PCD Type 5 ○ Interleave set index = 1 ○ Number of modules in the interleave set = 2 ○ Interleave memory type = 2 (AD) ○ Interleave size = 0x4040 ○ Module identification information (two entries)