**# Title:**

RAS2 add ADDRESS\_TRANSLATION service.

**# Status:**

Draft

**# Document:**

ACPI Specification Version 6.5

**# License:**

SPDX-License-Identifier: CC-BY-4.0

**# Submitter:**

* Harb Abdulhamid, Ampere Computing
* Tyler Baicar, Ampere Computing
* Vanshidhar Konda, Ampere Computing
* Daniel Ferguson, Ampere Computing
* Rob Gough, Ampere Computing
* Leo Duran, AMD
* Jose Marinho, Arm
* TianoCore Community (<https://www.tianocore.org>)

# # Summary of the change

This ECR:

* Add language to clarify that firmware can publish RAS2 and RASF table, but OSPM should use only one
* Changes language about using the size of the RAS2 table to compute the number of PCC descriptors in the table
* adds the ADDRESS\_TRANSLATION feature to the RAS2 table Memory RAS Features.

# # Benefits of the change

Several RAS related features have vendor specific information either provided as input to the platform or from the platform to the OS. Adding vendor identification information to the RAS2 table will allow OS and application software to identify the vendor and take appropriate actions or interpret data according to the vendor defined manuals.

The ADDRESS\_TRANSLATION service enables OSPM to obtain:

* A logical address from a physical address. The logical address is required, for instance, for error injection when the error is injected on a system component with a local view of memory.
* A physical address from a logical address. The physical address is required, for instance, for injecting an error using EINJv2 targeting specific location on the memory device.

The physical address from logical address translation is like existing LA2PA\_TRANSLATION service. Using the ADDRESS\_TRANSLATION service instead allows software to have a consistent interface with the same parameters in both directions. The vendor specific information also allows the OSPM or platform to share more information related to a memory device for which the address is being requested.

# **# Impact of the change**

The ECR does not impact current implementations.

# # Detailed description of the change [normative updates]

Delta from ACPI 6.5

* Changes in **yellow**
* Insertions in **green**
* Removals in **~~red~~**
* References that need fixup in blue

5.2.21. ACPI RAS2 Feature Table (RAS2)

The RAS2 table provides interfaces for platform RAS features. RAS2 offers the same services as RASF, but is more

scalable than the latter. In particular, RAS2 supports independent RAS controls and capabilities for a given RAS

feature for multiple instances of the same component in a given system. Platform firmware can publish RAS2 and RASF table but OSPM should use only one.

*Table 5.79* ***RAS2 Table Format***

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Byte Length** | **Byte Offset** | **Description** |
| Header |  |  |  |
| Signature | 4 | 0 | Signature is set to ‘RAS2’ for RAS Feature 2 Table. |
| Length | 4 | 4 | Length in bytes for entire RAS2 table. ~~The length implies the number of PCC descriptors fields at the end of the table~~ |
| Revision | 1 | 8 | 1 |
| 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 utility that created the table |
| Creator Revision | 4 | 32 | Revision of utility that created the table |
| RAS2 Specific Entries | | | |
| Reserved | 2 | 36 | Reserved, should be zero. |
| Number of PCC descriptors | 2 | 38 | Number of PCC descriptors |
| RAS2 Platform Communication Channel (PCC) Descriptor List | N\*8 | 40 | List of PCC descriptors |

**The following table identifies the supported Memory RAS features.**

*Table 5.86****Platform RAS Feature Bitmap for Memory RAS***

|  |  |  |  |
| --- | --- | --- | --- |
| **Bit** | **RAS Feature** | **Feature Name** | **Description** |
| 0 | Hardware-based memory scrub feature | PATROL\_SCRUB | Indicates that the platform supports hardware-based memory scrubbing. OSPM must set this bit in the Set RAS Capabilities field to request memory scrubbing service. |
| 1 | Logical to Physical Address translation feature | LA2PA\_TRANSLATION | Indicates that the platform supports logical address to physical address translation service. OSPM must set this bit in the Set RAS Capabilities field to request address translation for a logical address. |
| 2 | Address translation feature | ADDRESS\_TRANSLATION | Indicates that the platform supports address translation service. OSPM must set this bit in the Set RAS Capabilities field to request an address translation. |
| 3~~2~~-127 | *Reserved* |  | *Reserved for future use* |

##### 5.2.21.2.3. Address Translation Service

The platform can use this feature to provide support for translation of physical addresses to logical addresses and vice versa.

The translation to logical addresses is required when the OSPM intends to inject an error on a component using the local view of memory of that component. The translation of logical address to physical address is required when OSPM only has the capability to inject an error using physical address but wants to target specific locations on a memory component.

*Table 5.zz****Parameter Block Structure for ADDRESS\_TRANSLATION***

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Byte Length** | **Byte Offset** | **Description** |
| Type  (FIXED OUTPUT) | 2 | 0 | 0x0002 – Address translation service  This field is set by Platform. RO for OSPM / Software. |
| Version  (FIXED OUTPUT) | 2 | 2 | Byte 0 - Minor Version  Byte 1 - Major Version  For this format of the parameter block, this field should be set to 0x0100.  This field is set by Platform. RO for OSPM / Software. |
| Length  (FIXED OUTPUT) | 2 | 4 | Length, in bytes of the entire parameter block structure.  This field is set by Platform. RO for OSPM / Software. This must be set to the maximum possible output of this parameter block. |
| Address Translation Command (INPUT) | 2 | 6 | 0x01 - GET\_PA2LA\_TRANSLATION  0x02 – GET\_LA2PA\_TRANSLATION  All other values are reserved. |
| Physical Address (INPUT/OUTPUT) | 8 | 8 | When OSPM uses the GET\_PA2LA\_TRANSLATION command it specifies the system physical address in this field for which it wants the local logical address, SMBIOS info or vendor specific info.  When OSPM uses the GET\_LA2PA\_TRANSLATION command the platform provides the system physical address in this field. |
| Status (OUTPUT) | 4 | 16 | The platform returns this value in response to ADDRESS\_TRANSLATION:  0x0000\_0000: Indicates that the translation succeeded.  0x0000\_0001: Indicates that the translation failed, the Logical Address (in response to GET\_PA2LA\_TRANSLATION command) or Physical Address (in response to GET\_LA2PA\_TRANSLATION command) returned by the platform may not be valid.  0x1000\_0000: Indicates that the translation command (GET\_LA2PA\_TRANSLATION or GET\_PA2LA\_TRANSLATION) is not supported by the platform.  0x2000\_0000: Indicates that the Logical Address Type is not supported when using the GET\_LA2PA\_TRANSLATION command  Other values are reserved for future use by this specification. |
| SMBIOS Locality Info  (INPUT/OUTPUT) | 2 | 20 | This field contains the SMBIOS handle for the Type 17 Memory Device Structure that represents the memory module.  This field can be optionally used to identify the memory component associated with the physical/logical address.  The platform returns the SMBIOS handle of the device associated with this physical address when using the GET\_PA2LA\_TRANSLATION command.  OSPM writes the SMBIOS handle of the device associated with the logical address when using the GET\_LA2PA\_TRANSLATION command.  If the value is 0xFFFF, platform and OSPM shall assume there is no SMBIOS locality information available.  In this case, the logical address should enable platform firmware to uniquely identify the memory component. |
| Reserved | 2 | 22 | Reserved. Must be zero. |
| Logical Address Type  (INPUT/OUTPUT) | 2 | 24 | This field identifies the type of encoding used for the Logical Address field.  0x0 – unused  0x1 – DDR4/DDR5  0xFF – Vendor defined  All other values are reserved. |
| Logical Address Length  (INPUT/OUTPUT) | 2 | 26 | Length of the Logical Address field in bytes. |
| Logical Address (INPUT/OUTPUT) | N | 28 | If there are multiple constituent components that fall within the Instance, this field can be used to point to the specific component to which the LA applies.  If the LA Type is 0x1, see Table 5.xx – DDR4/DDR5 Logical Address Structure  If the LA Type is 0xFF, see Table 5.yy – Vendor Defined Logical Address Structure. |

*Table 5.xx****DDR4/DDR5 Logical Address Structure***

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Byte Length** | **Byte Offset** | **Description** |
| Row | 4 | 0 | The row number of the memory location |
| Column | 4 | 4 | The column number of the memory location |
| Rank | 4 | 8 | The rank number of memory location |
| Bank | 2 | 12 | The bank number of the memory location.  Bit 7:0 – Bank Address  Bit 15:8 – Bank Group |
| Byte | 1 | 14 | The byte number of the memory location |
| Chip Identification | 1 | 15 | The Chip Identification |
| Node | 2 | 16 | In a multi-node system, this value identifies the node containing the memory component. |
| Card | 2 | 18 | The card number of the memory component. |
| Module | 2 | 20 | The module of the memory component (Node, Card, Module should provide the information necessary to identify the FRU being targeted). |

**Note:** The definition of the fields in Table 5.xx match the same field in the CPER Memory Error Section. Refer to UEFI Specification Appendix N – Common Platform Error Record for details.

*Table 5.yy****Vendor Defined Logical Address Structure***

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Byte Length** | **Byte Offset** | **Description** |
| Vendor ID | 4 | 0 | 4 letter ACPI ID of the vendor from the ACPI ID Registry. |
| Vendor Address Format Identifier | 4 | 4 | Vendor-specific value that maps to the vendor-specific Logical Address format. This may include a revision field. |
| Vendor defined Logical Address | N | 8 | Logical Address as defined by the Vendor  The length of field is the Logical Address Length field – 8 bytes. |