**# Title:**

RAS2: Add Status field to Patrol Scrub parameter block.

**# Status:**

Draft

**# Document:**

ACPI Specification Version 6.5

**# License:**

SPDX-License-Identifier: CC-BY-4.0

**# Submitter:**

* Daniel Ferguson, Ampere Computing
* Vanshidhar Konda, Ampere Computing
* Bill Schwartz, Ampere Computing
* TianoCore Community (<https://www.tianocore.org>)

# # Summary of the change

This ECR:

* Updates the RAS2 Platform Communication Space to version 1.0
* Deprecates the RAS2 Set Capabilities Status with the note that the field is used to return status of PATROL\_SCRUB capability in prior versions
* Update PATROL\_SCRUB parameter block version to 1.0 to sync with the change to RAS2 Platform Communication Space
* Add STATUS field to the PATROL\_SCRUB parameter block

# # Benefits of the change

There are multiple RAS2 capabilities and OSPM can trigger action on multiple capabilities through a single PCC command. The “Set RAS Capabilities Status” field in the RAS2 Platform Communication space cannot indicate the Status of all the capabilities through this field. In earlier versions, only the status of the PATROL\_SCRUB capability would be reported through the field.

There are currently 3 RAS2 capabilities:

* PATROL\_SCRUB
* LA2PA\_TRANSLATION
* ADDRESS\_TRANSLATION

Both the LA2PA\_TRANSLATION and ADDRESS\_TRANSLATION parameter blocks have a Status field to indicate the status of those capabilities. PATROL\_SCRUB parameter block can also be updated to have Status field for OSPM to read. This provides a more consistent mechanism for OSPM to get status of each capability.

# **# 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

**5.2.21.1.3 RAS2 Platform Communication Channel**

The RAS2 platform communication channel format is defined below ([Table 5.83](file:///C:\Users\DanielFerguson\Downloads\pdf2docx\ACPI_Spec_6_5_Aug29.docx#page238)).

Table 5.83: RAS2 Platform Communication Channel Shared Memory Region

|  |  |  |  |
| --- | --- | --- | --- |
| Field | Byte Length | Byte Offset | Description |
| Signature | 4 | 0 | The PCC signature. The signature of a subspace is computed by a bitwise-or of the value 0x50434300 with the subspace ID. For example, subspace 3 has the signature 0x50434303. |
| Command | 2 | 4 | PCC command field; see PCC Command Codes used by RAS2. See Table 5.82 and Section 14. |
| Status | 2 | 6 | PCC status field. See Section 14. |
| Communication Space |  |  |  |
| - Version | 2 | 8 | Byte 0 - Minor Version  Byte 1 - Major Version  For this revision, this field must be set to 0x0100 |
| - RAS Features | 16 | 10 | Bitmap describing the platform RAS features as shown  in Table 5.84. The definition of the bits is system component specific. For example, Table 5.86 shows the bitmap definitions for Memory RAS features. The Platform populates this field to indicate which RAS features for the given feature type are supported for this system component instance. The OSPM uses this field for RAS feature discovery. |
| - Set RAS Capabilities | 16 | 26 | Bit Map of the RAS features for which the OSPM is  invoking the command. The Bit Map is described in  Section 5.2.21.1.4. OSPM sets the bit corresponding to a RAS capability to invoke a command on that capability. The bitmap implementation allows OSPM to invoke a command on each RAS feature supported by the platform at the same time. |
| - Number of RAS2 Parameter Blocks | 2 | 42 | The Number of parameter blocks will depend on how  many RAS Capabilities the Platform Supports. Typically, there will be one Parameter Block per RAS  Feature, using which that feature can be managed by  OSPM. |
| - Set RAS Capabilities Status  (DEPRECATED) | 4 | 44 | Must be Zero.  Prior to version 0x0100 this field would return the status of the PATROL\_SCRUB capability. This status is now reflected in the STATUS field of the PATROL\_SCRUB Parameter Block.  Status:  0000b = Success  0001b = Not Valid  0010b = Not Supported  0011b = Busy  0100b = Failed  0101b = Aborted  0110b = Invalid Data |
| - Parameter Blocks | Varies (N bytes) | 48 | Start of the parameter blocks, the structure of which  is shown in the Parameter Block Structure for PATROL\_SCRUB. These parameter blocks are used as  communication mailbox between the OSPM and the  platform, and there is 1 parameter block for each RAS  feature. NOTE: There can be only one parameter block  per type. |

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

##### 5.2.21.2.1. Hardware-based Memory Scrubbing

The platform can use this feature to expose controls and capabilities associated with hardware-based memory scrub engines. Modern scalable platforms have complex memory systems with a multitude of memory controllers that are in turn associated with NUMA domains. It is also common for RAS errors related to memory to be associated with NUMA domains, where the NUMA domain functions as a FRU identifier. This feature thus provides memory scrubbing at a NUMA domain granularity.

The following are supported:

* Independent memory scrubbing controls for each NUMA domain, identified using its proximity domain.
* Provision for background (patrol) scrubbing of the entire memory system, as well as on-demand scrubbing for a specific region of memory.

*Table 5.87***Parameter Block Structure for PATROL\_SCRUB**

|  |  |  |  |
| --- | --- | --- | --- |
| Field | Byte Length | Byte Offset | Description |
| Type  (FIXED OUTPUT) | 2 | 0 | 0x0000 – Hardware-based memory scrub RAS feature. |
| 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. |
| Length (FIXED OUTPUT) | 2 | 4 | Length, in bytes of the entire parameter block structure. |
| Patrol Scrub Command (INPUT) | 2 | 6 | 0x01 - GET\_PATROL\_PARAMETERS  0x02 - START\_PATROL\_SCRUBBER  0x03 - STOP\_PATROL\_SCRUBBER |
| Requested Address Range (INPUT) | 16 | 8 | OSPM Specifies the BASE (Bytes 7-0) and SIZE  (Bytes 15-8) of the address range to be patrol scrubbed.  If OSPM requests default scrubbing through Bit 0 of  the Configure patrol scrubbing field, then this field  must be ignored by the platform.  OSPM sets this parameter for the following commands:  GET\_PATROL\_PARAMETERS,  START\_PATROL\_SCRUBBER. |
| Actual Address Range (OUTPUT) | 16 | 24 | The platform returns this value in response to  GET\_PATROL\_PARAMETERS. The platform  calculates the nearest patrol scrub boundary address  from where it can start. This range should be a superset  of the Requested Address Range.  This field must be ignored by the OSPM if it is being  returned in response to a request to enable default  scrubbing through Bit 0 of the Configure patrol  scrubbing field.  BASE (Bytes 7-0) and SIZE (Bytes 15-8) of the  address. |
| Flags (OUPUT) | 4 | 40 | The platform returns this value in response to  GET\_PATROL\_PARAMETERS:  Bit [0]: Will be set if memory scrubber is already  running for address range specified in “Actual Address  Range”.  Bits [31:1]: Reserved, must be zero. |
| Scrub Parameters (OUTPUT) | 4 | 44 | The platform returns this value in response to  GET\_PATROL\_PARAMETERS:  Bits [7:0]: Current scrub rate that is in effect on the  memory region specified in “Actual Address Range”. If  OSPM requested background scrubbing, then this field  will reflect the current background patrol scrubbing  rate.  Bits [15:8]: Minimum scrub rate supported.  Bits [23:16]: Maximum scrub rate supported  Bits [31:24]: Reserved, must be zero. |
| Configure Scrub Parameters (INPUT) | 4 | 48 | The OSPM Sets this field as follows, for the  START\_PATROL\_SCRUBBER command:  Bit [0]: Request background patrol scrubbing.  Bits [7:1]: Reserved, must be zero.  Bits [15:8]: Requested scrub rate, must be in the range  (minimum scrub rate, maximum scrub rate).  Bits [31:16]: Reserved, must be zero. |
| Status (OUTPUT) | 4 | 52 | The platform sets this field when any problem is encountered during the execution of this feature.  Status:  0000b = Success  0001b = Not Valid  0010b = Not Supported  0011b = Busy  0100b = Failed  0101b = Aborted  0110b = Invalid Data |