RISC-V IO Mapping Table (RIMT) Specification

Version v1.0, 2025-03-31: Ratified

Table of Contents

Preamble

This document is in the Ratified state

No changes are allowed. Any desired or needed changes can be the subject of a follow-on new extension. Ratified extensions are never revised.

Copyright and license information

Contributors

This RISC-V specification has been contributed to directly or indirectly by:

- Aaron Durbin <adurbin@rivosinc.com>
- Andrew Jones <ajones@ventanamicro.com>
- Anup Patel <apatel@ventanamicro.com>
- Atish Kumar Patra <atishp@rivosinc.com>
- Heinrich Schuchardt <heinrich.schuchardt@canonical.com>
- Samuel Holland <samuel.holland@sifive.com>
- Sebastien Boeuf <seb@rivosinc.com>
- Sunil V L <sunilvl@ventanamicro.com>
- Tomasz Jeznach <tjeznach@rivosinc.com>
- Ved Shanbhogue <ved@rivosinc.com>

Changelog

- **Version v1.0**
  - Version update to indicate "Ratified" state.
- **Version v0.99**
  - Version update to indicate "Ratification Ready" state.
- **Version v0.92**
  - Update the copyright year to include 2025.
- **Version v0.91**
  - Addressed public review feedback.
  - Updated version and contributor list.
  - Ready for TSC sign-off.
- **Version 1.0.0-rc5**
  - Updated document state to Frozen.
- **Version 1.0.0-rc4**
  - Added source ID overlap restriction details.
  - Addressed other ARC feedback for RC3 version.
- **Version 1.0.0-rc3**
  - Addressed feedback from ARC review.
- **Version 1.0.0-rc2**
  - Draft for ARC review.
  - Addressed internal review feedback.
  - Used IEEE style bibliography.
  - Allowed HW ID to be valid for PCIe IOMMU as well.
- **Version 1.0.0-rc1**
  - Draft for internal review.
  - Added ID mapping examples.
  - Documentation template changes.
  - Addressed PRS TG feedback.
- **Version 0.0.1**
  - Initial draft for PRS TG review

Terms and Abbreviations

This specification uses the following terms and abbreviations:

Term	Meaning
ACPI	Advanced Configuration and Power Interface Specification
APLIC	Advanced Platform-Level Interrupt Controller
IOMMU	Input-Output Memory Management Unit
PLIC	Platform-Level Interrupt Controller
RCiEP	Root Complex Integrated End Point

Chapter 1. Introduction

The RISC-V IO Mapping Table (RIMT) provides information about the RISC-V IOMMU [1] and the relationship between the IO topology and the IOMMU in ACPI [2] based RISC-V platforms. The RIMT identifies which components are behind IOMMU and how they are connected together. RISC-V IOMMU can be implemented as either a PCIe device or a platform device.

Chapter 2. RISC-V IO Mapping Table (RIMT)

The Table 1 shows the structure of RIMT. Apart from the basic header, RIMT can contain several nodes. Each node represents a component, which can be an IOMMU, a PCIe root complex, or a platform device.

Table 1. RISC-V IO Mapping Table

Field	Byte Length	Byte Offset	Description
Signature	4	0	'RIMT' signature for the RISC-V IO Mapping Table.
Length	4	4	The length of the table, in bytes, of the entire RIMT.
Revision	1	8	1
Checksum	1	9	The entire table must sum to zero.
OEMID	6	10	OEM ID.
OEM Table ID	8	16	For the RIMT, the table ID is the manufacturer model ID.
OEM Revision	4	24	OEM revision of the RIMT for the supplied OEM Table ID.
Creator ID	4	28	The vendor ID of the utility that created the table.
Creator Revision	4	32	The revision of the utility that created the table.
Number of RIMT Nodes	4	36	Number of nodes in the RIMT nodes array.
Offset to RIMT Node Array	4	40	The offset from the start of this table to the first node in RIMT node array.
Reserved	4	44	Must be zero.
RIMT Node Array	-	48	List of RIMT nodes in the platform. Nodes listed may be one of the types listed in Table 2. This structure for node types is defined in the following sections.

2.1. RIMT node structure types

RIMT node structures can be broadly classified as two types: one is the actual IOMMU node structure and the other is the device node structure for devices bound to an IOMMU. The device node structure can be further classified as PCIe root complex and platform device structures bound to an IOMMU. For example, in a system with a single IOMMU, RIMT should have at least two nodes. One for the IOMMU itself and another for the devices behind this particular IOMMU. Table 2 lists possible types for those structures.

Table 2. RIMT Node Types

Value	Description
0	RISC-V IOMMU Node. See Table 3
1	PCIe Root Complex Node. See Table 5
2	Platform Device Node. See Table 7
3-255	Reserved

2.1.1. IOMMU Node

The IOMMU can be implemented as a platform device or as a PCIe device. The IOMMU node is the structure in RIMT used to report the configuration and capabilities of each IOMMU in the system.

Table 3. IOMMU Node

Field	Byte Length	Byte Offset	Description
Type	1	0	0 - IOMMU Node.
Revision	1	1	1
Length	2	2	The length of this structure.
Reserved	2	4	Must be zero.
ID	2	6	Unique ID of this node in the RIMT that can be used to locate it in the RIMT node array. It can be simply the array index in the RIMT node array.
Hardware ID	8	8	ACPI ID of the IOMMU when it is a platform device or PCIe ID (Vendor ID + Device ID) for the PCIe IOMMU device. This field adheres to the **_HID** format described by the ACPI [2] specification.
Base Address	8	16	Base address of the IOMMU registers. This field is valid for only an IOMMU that is a platform device. If IOMMU is a PCIe device, the base address of the IOMMU registers may be discovered from or programmed into the PCIe BAR of the IOMMU.
Flags	4	24	• Bit 0: IOMMU is a PCIe device ◦ 1: The IOMMU is implemented as a PCIe device. ◦ 0: The IOMMU is implemented as a platform device. • Bit 1: Proximity Domain valid ◦ 1: The Proximity Domain field has a valid value. ◦ 0: The Proximity Domain field does not have a valid value. • Bit [31-2]: Reserved, must be zero
Proximity Domain	4	28	The Proximity Domain to which this IOMMU belongs. This is valid only when the "Proximity Domain Valid" flag is set. For optimal IOMMU performance, the in-memory data structures used by the IOMMU may be located in memory from this proximity domain.
PCIe Segment number	2	32	If the IOMMU is implemented as a PCIe device (Bit 0 of Flags is 1), then this field holds the PCIe segment where this IOMMU is located.
PCIe B/D/F	2	34	If the IOMMU is implemented as a PCIe device (Bit 0 of Flags is 1), then this field provides the Bus/Device/Function of the IOMMU.

Field	Byte Length	Byte Offset	Description
Number of interrupt wires	2	36	An IOMMU may signal IOMMU initiated interrupts by using wires or as message signaled interrupts (MSI). When the IOMMU supports signaling interrupts by using wires, this field provides the number of interrupt wires. This field must be 0 if the IOMMU does not support wire-based interrupt generation.
Interrupt wire array offset	2	38	The offset from the start of this node entry to the first entry of the Interrupt Wire Array. This field is valid only if "Number of interrupt wires" is not 0.
List of interrupt wires.			
Interrupt wire array	8 * N	40	Array of Interrupt Wire Structures where N is the number of elements in the array. See Table 4.

Table 4. Interrupt Wire Structure

Field	Byte Length	Byte Offset	Description
Interrupt Number	4	0	Interrupt number. This should be a Global System Interrupt (GSI) number. These are wired interrupts with GSI numbers mapping to a particular PLIC or APLIC. The OSPM determines the mapping of the Global System Interrupts by determining how many interrupt inputs each PLIC or APLIC supports and by determining the global system interrupt base for each PLIC / APLIC.

Flags	4	4	• Bit 0: Interrupt Mode
			  ◦ 0: Edge Triggered.
			  ◦ 1: Level Triggered.
			• Bit 1: Interrupt Polarity
			  ◦ 0: Active Low.
			  ◦ 1: Active High.
			• Bit [31-2]: Reserved, must be zero

2.1.2. PCIe Root Complex Node

The PCIe root complex node is the logical PCIe root complex that can be used to represent an entire physical root complex, an RCiEP/set of RCiEPs, a standalone PCIe device, or the hierarchy following a PCIe host bridge.

Table 5. PCIe Root Complex Node

Field	Byte Length	Byte Offset	Description
Type	1	0	1 - PCIe Root Complex Node.
Revision	1	1	1
Length	2	2	The length of this structure.

Field	Byte Length	Byte Offset	Description
Reserved	2	4	Must be zero.
ID	2	6	Unique ID of this node in the RIMT that can be used to locate it in the RIMT node array. It can be simply the array index in the RIMT node array.
Flags	4	8	- Bit 0: ATS support - 0: ATS is not supported in this root complex. - 1: ATS supported in this root complex. - Bit 1: PRI support - 0: PRI is not supported in this root complex. - 1: PRI is supported in this root complex. - Bit [31-2]: Reserved, must be zero
Reserved	2	12	Must be zero.
PCIe Segment number	2	14	The PCIe segment number, as in MCFG and as returned by _SEG method in the ACPI namespace.
ID mapping array offset	2	16	The offset from the start of this node to the start of the ID mapping array.
Number of ID mappings	2	18	Number of elements in the ID mapping array.
List of ID mappings			
ID mapping array	20 * N	20	Array of ID mapping structures where N is the number of ID mapping structures. See Table 6.

The ID mapping structure provides information about how devices are connected to an IOMMU. The devices can be natively identified by a source ID, but the platform can use a remapped ID to identify transactions from the device to the IOMMU.

For PCIe devices, source ID is the 16-bit triplet of PCIe bus number (8-bit), device number (5-bit), and function number (3-bit) (collectively known as routing identifier or RID). A range of source IDs must map to a single IOMMU only. If there are multiple root complexes with the same PCIe segment number, then their source ID ranges must not overlap. For each ACPI device object of the root complex that belongs to the same PCIe segment, the firmware must include the Device Specific Method (_DSM), Function Index 5, for preserving boot configurations as defined by the PCI Firmware Specification [3]. The _DSM method must return zero to indicate that the Operating System must preserve PCIe resource assignments made by the firmware at boot time.

For platform devices, source ID is the implementation specific ID and managed by the device driver. Each ID mapping array entry provides a mapping from a range of source IDs to the corresponding device IDs that will be used at the input to the IOMMU. See Chapter 3 for an example of ID mapping structures.

Table 6. ID Mapping Structure

Field	Byte Length	Byte Offset	Description
Source ID Base	4	0	The base of a range of source IDs mapped by this entry to a range of device IDs that will be used at input to the IOMMU.
Number of IDs	4	4	Number of IDs in the range. The range must include the IDs of devices that may be enumerated later during OS boot (For example, SR-IOV Virtual Functions).
Destination Device ID Base	4	8	The base of the destination ID range as mapped by this entry. This is the **device_id** as defined by the RISC-V IOMMU specification [1]
Destination IOMMU Offset	4	12	The destination IOMMU that is associated with these IDs. This field is the offset of the RISC-V IOMMU node from the start of the RIMT table.
Flags	4	16	● Bit 0: ATS Required ● 0: ATS does not need to be enabled for the device to function. ● 1: ATS needs to be enabled for the device to function. ● Bit 1: PRI Required ● 0: PRI does not need to be enabled for the device to function. ● 1: PRI needs to be enabled for the device to function. ● Bit [31-2]: Reserved, must be zero

2.1.3. Platform Device Node

There may be non-PCIe platform devices that are enumerated by using Differentiated System Description Table(DSDT). These devices can have one or more source IDs in the mapping table, but they can have their own scheme to define the source IDs. Hence, those source IDs can be unique to only the ACPI platform device. The interpretation of those source IDs is expected to be managed by the platform device's device driver.

Table 7. Platform Device Node

Field	Byte Length	Byte Offset	Description
Type	1	0	2 - Platform Device Node.
Revision	1	1	1
Length	2	2	The length of this structure.
Reserved	2	4	Must be zero.
ID	2	6	Unique ID of this node in the RIMT that can be used to locate it in the RIMT node array. It can be simply the array index in the RIMT node array.

Field	**Byte Length**	**Byte Offset**	**Description**
ID mapping array offset	2	8	The offset from the start of this node to the start of the ID mapping array.
Number of ID mappings	2	10	Number of elements in the ID mapping array.
Device Object Name	M	12	Null terminated ASCII string. Full path to the device object in the ACPI namespace.
Padding	P	12 + M	Pad with zeros to align the ID mapping array at 4-byte offset.
List of ID mappings.			
ID Mapping Array	20 * N	12 + M + P	Array of ID mapping structures where N is the number of ID mapping structures. See Table 6.

Chapter 3. ID Mapping Examples

Table 8. PCIe device ID mapping example

Source ID Base	Number of IDs	Destination Device ID Base	Destination IOMMU Offset	Flags
0x0000	0x10	0x0	IOMMU0_OFFSET_IN_RIMT	0
0x0100	0x10	0x10	IOMMU0_OFFSET_IN_RIMT	0

Table 9. Platform device ID mapping example

Source ID Base	Number of IDs	Destination Device ID Base	Destination IOMMU Offset	Flags
0x0000	0x1	0x20	IOMMU0_OFFSET_IN_RIMT	0

Bibliography

[1] “RISC-V IOMMU Specification 1.0.0.” [Online]. Available: github.com/riscv-non-isa/riscv-iommu/releases/download/v1.0.0/riscv-iommu.pdf.

[2] “Advanced Configuration and Power Interface Specification 6.5.” [Online]. Available: uefi.org/specs/ACPI/6.5.

[3] “PCI Firmware Specification 3.3.” [Online]. Available: members.pcisig.com/wg/PCI-SIG/document/folder/862.