1 SysMMU

This chapter includes the following sections:

- Overview
- Functional description
- Register description

1.1 Overview

System Memory Management Unit (SysMMU) translates the Virtual Address (VA) in the transactions initiated by master IPs in a system to the Physical Address (PA) that are accepted by slaves in System-on-Chips (SoCs).

Figure 1-1 illustrates that SysMMU must be placed between VA and PA domains in an AXI-compliant interconnect. SysMMU is targeted only to the masters without an embedded MMU such as multimedia masters.

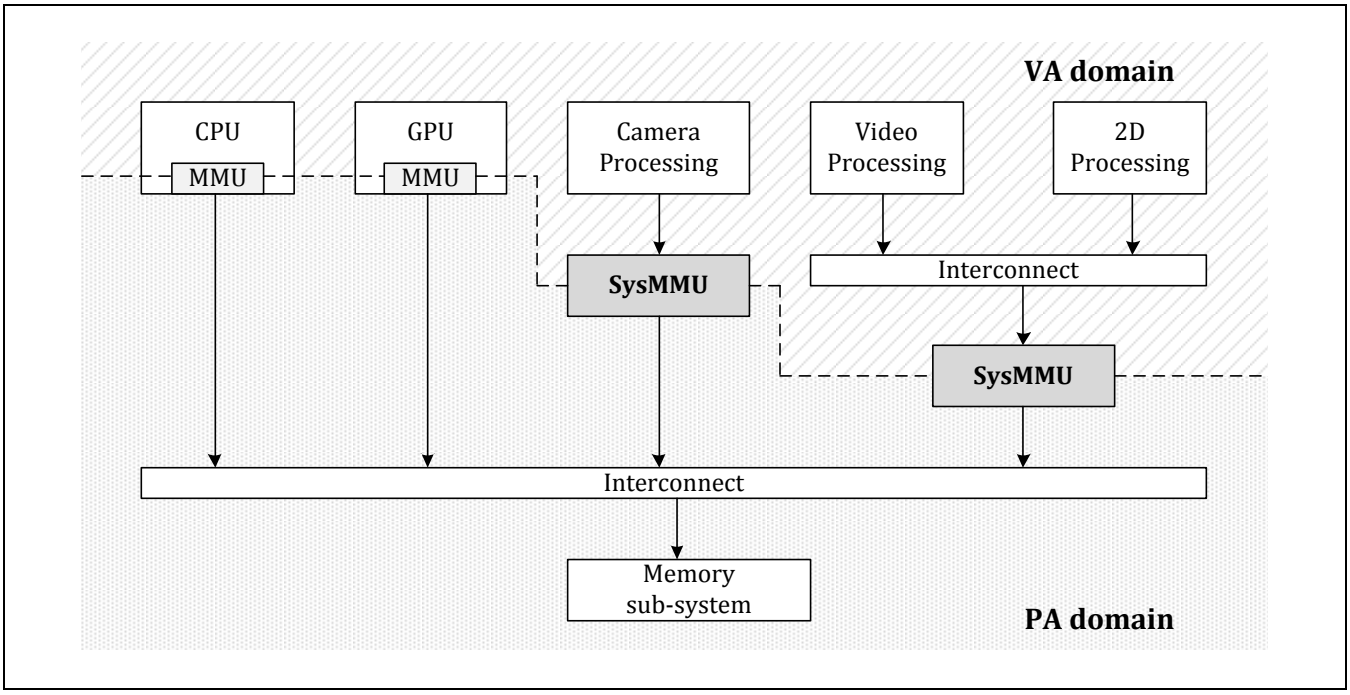

Figure 1-1 Placement of SysMMU in SoCs

Operating system (OS) allocates the memory resources to masters in pages for efficient memory management, where each page has 4 KB of memory fragment. Therefore, buffers used by masters consist of multiple distant pages. However, masters require contiguous memory view for efficient DMA implementation. SysMMU provides a virtual contiguous memory view irrespective of actual placement of allocated pages in the memory.

Page table consists of all the mappings between virtual memory view and physical memory view. Therefore, SysMMU must always refer the page table to translate the virtual addresses to physical addresses and is called Page Table Walk (PTW). However, PTW for every transaction takes relatively long time because it accesses the main memory in which the page table is stored.

To reduce the translation time, SysMMU uses Translation Look-ahead Buffer (TLB). TLB contains recently-read mapping information by PTW, called page descriptor, to enable temporal locality. So, if TLB already has the page descriptor for an address translation request, then the address translation is immediately performed without accessing the main memory for PTW.

Figure 1-2 illustrates the three major tasks performed by the SysMMU, namely Page Table Walk (PTW), address translation, and cache.

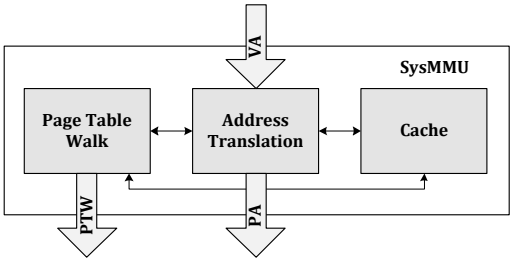

Figure 1-2 Tasks Performed in SysMMU

1.2 Functional Description

1.2.1 Address Translation

To use the virtual memory system, SysMMU provides function of the address translation from VA to PA to master IPs. SysMMU must refer the page table for address translation where page table consists of a list of page descriptors. Each page descriptor has mapping information of VA to PA.

Following are the specifications of SysMMU:

- Supports only 32/36-bit VA to 36-bit PA translation.
- Memory must contain the page table before performing the address translation in SysMMU.

1.2.1.1 Multiple Page Granularities

OS uses a list of memory fragments or page to manage the memory resources, where each page has 4 KB space. Therefore, each memory allocation must update the mapping information from virtual pages to physical pages in 4 KB granularity as illustrated in *Figure 1-3*.

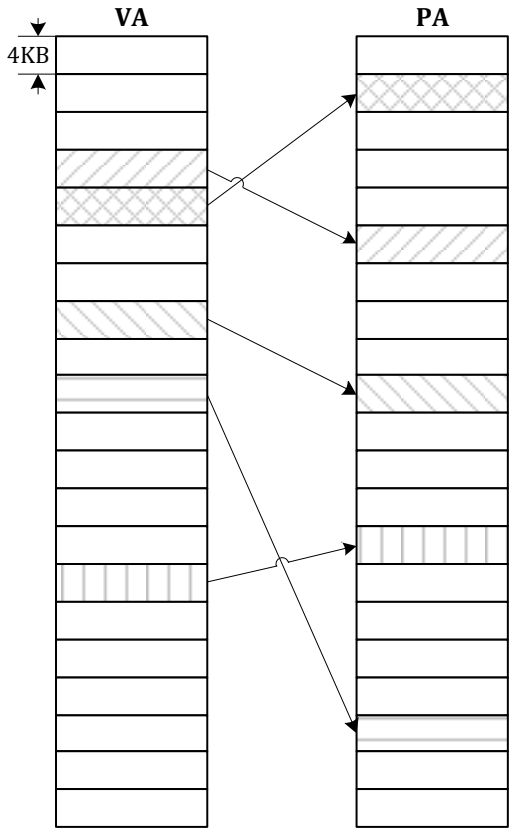

Figure 1-3 Mappings from VA to PA in 4 KB Page Granularity

If multiple mappings between virtual pages and physical pages have consecutive relationships, then a single description covers the mapping relationships. SysMMU supports multiple page granularities, namely 4 KB, 64 KB, 1 MB, 2 MB, and 16 MB. For example, mapping descriptions with consecutive relationship for sixteen 4 KB pages are described by one mapping description for the corresponding 64 KB page as illustrated in *Figure 1-4*. It also enhances the reachability of TLB, so that one cache line covers up to 16 MB memory region.

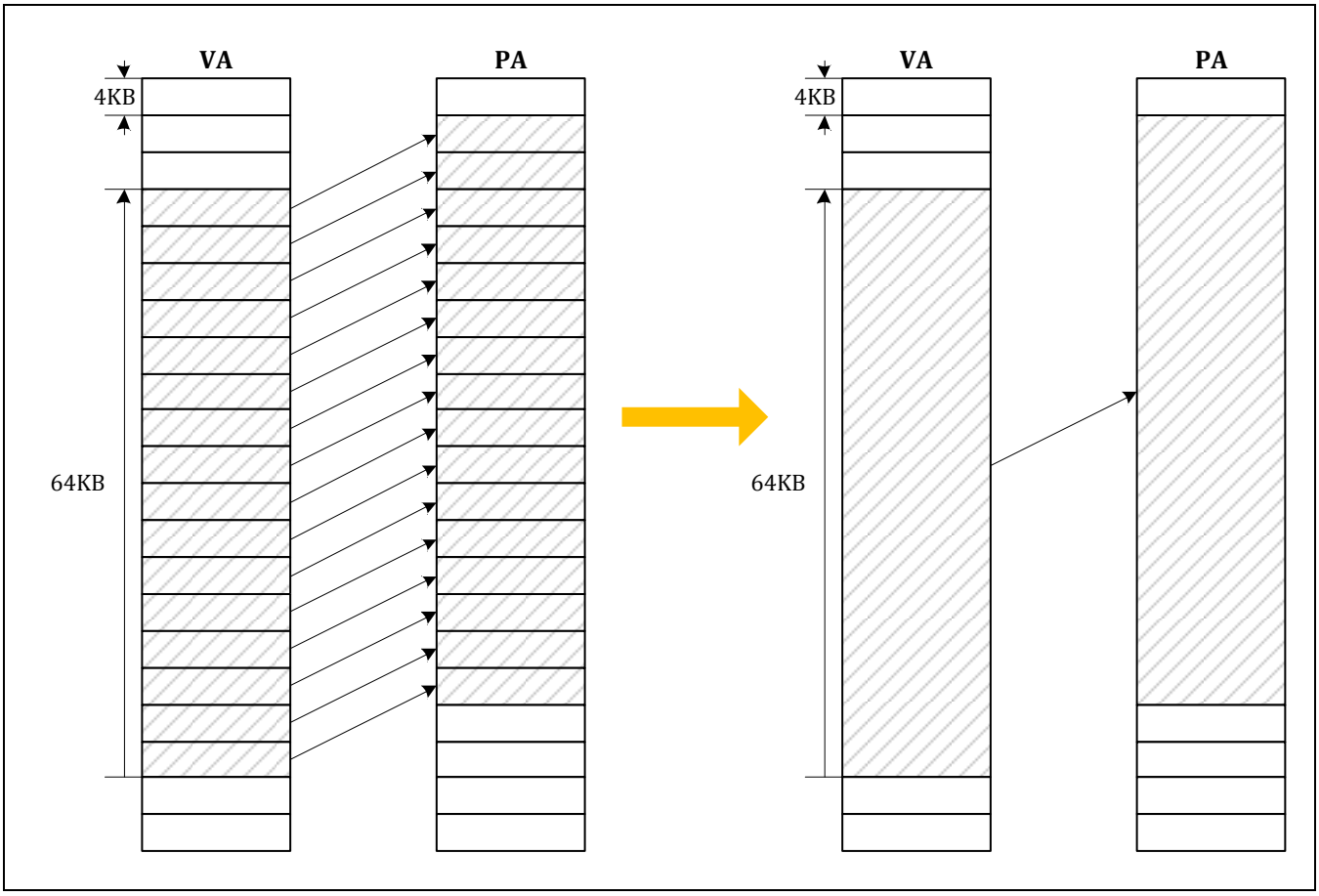

Figure 1-4 Mappings from VA to PA in 64 KB Page Granularity when Mappings for 16 4 KB Pages has Consecutive Relationship

1.2.1.2 Two-Level Page Tables

To cover 4 or 64 GB memory (according to VA width) by page descriptors mapping 4 KB pages, page table must have 2^{20} or 2^{24} page descriptors. If each page descriptor has 4 bytes, 4 or 64 MB size of memory is required to manage a 4 or 64 GB memory.

However, mappings for 4 KB do not always exist, because some memory chunks are mapped in larger page granularity or some memory chunks are not allocated. To utilize this characteristic, multi-level page tables are used to minimize the memory size for page tables.

Increasing the number of levels efficiently manages the size of page tables, but it increases the address translation time for smaller granule pages, because memory access is required to traverse all levels of page tables.

SysMMU supports two-level page tables to counter the issue. First-Level Page Table (FLPT) manages the whole memory space in 1 MB granularity. Therefore, each page descriptor in FLPT contains a mapping from 1 MB virtual memory space to physical memory space. Each page descriptor in FLPT also has a pointer to Second-Level Page Table (SLPT) to manage 1 MB memory space in fine-grain 4 KB pages. *Figure 1-5* illustrates the relationship between two-level page tables and physical memory.

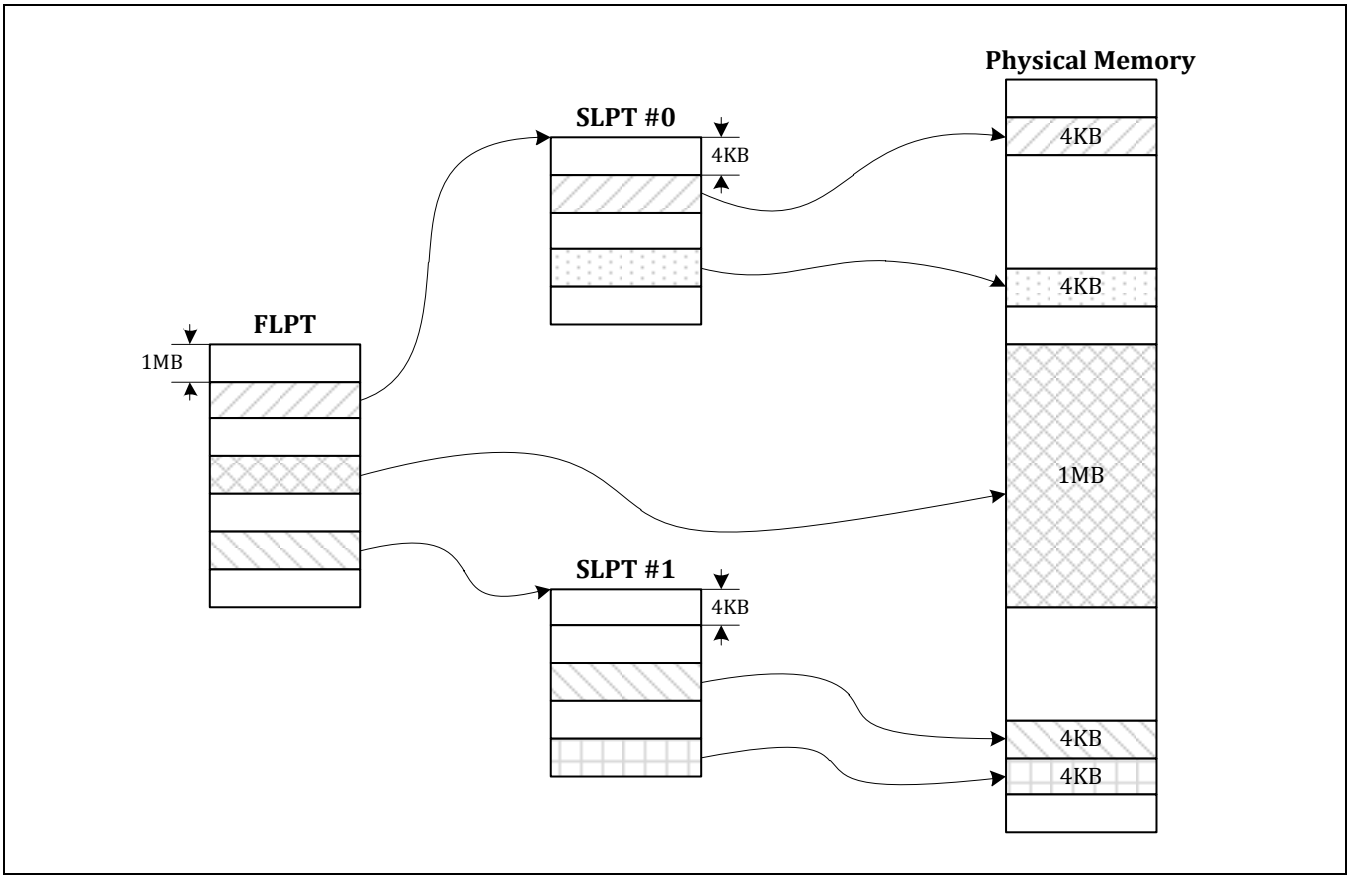

Figure 1-5 Relationship between Two-Level Page Tables and Physical Memory

1.2.1.3 First-Level Page Table Organization

First-Level Page Table (FLPT) is a list of page descriptors that are indexed by VA[***VA_WIDTH*** - 1:20], where ***VA_WIDTH*** can be 32 or 36.

Following are the specifications of FLPT:

- Each page descriptor has 4-byte or 32-bit width and must be stored in aligned.
- Each page descriptor represents a mapping for 1, 2, or 16 MB contiguous memory region.
- Base address of FLPT must be aligned in 16 KB for 32 bit VA or 256 KB for 36 bit VA.
- Two same FLPDs must be consecutively placed on the corresponding VPN location for 2 MB pages (*Figure 1-6*) and must be aligned by a multiple of 2.
- 16 same FLPDs must be placed on the corresponding VPN location for 16 MB pages (*Figure 1-6*) and must be aligned by a multiple of 16.
- A page descriptor in FLPT has a pointer to SLPT for page mappings of smaller granularity.

Figure 1-6 illustrates a sample FLPT organization.

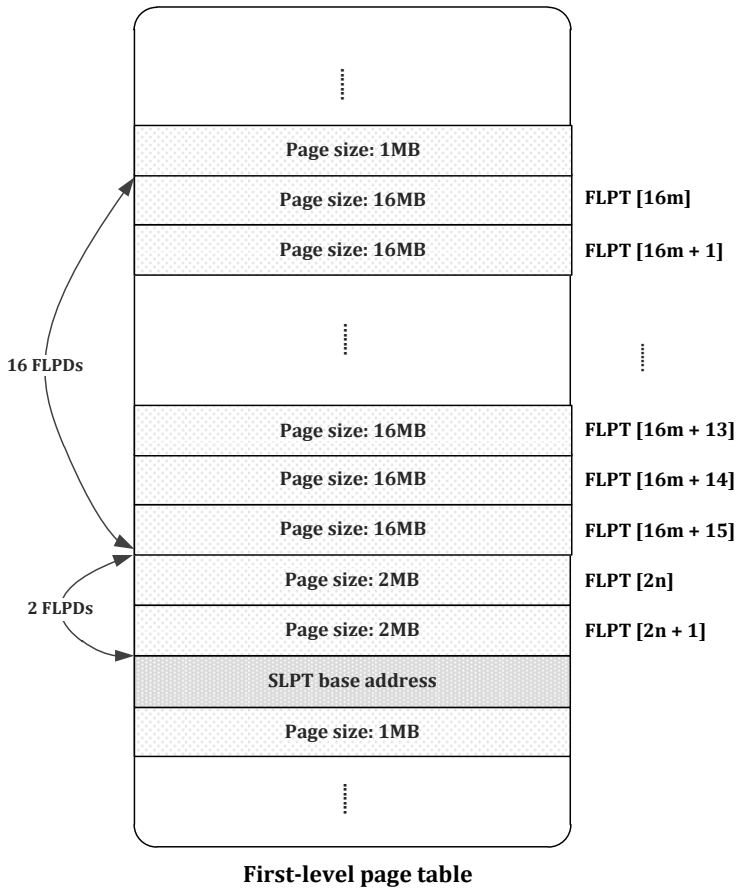

Figure 1-6 Sample FLPT Organization

Figure 1-7 illustrates the format of First-Level Page Descriptor (FLPD).

First-Level Page Descriptor (FLPD) Format

31:20	19:17	16	15:10	9:7	6	5	4	3	2	1	0	Page Size
Second-Level Page Table Base Address[35:10]						X		NS	X		1	4KB/64KB Page
PPN for 1MB, PA[35:20]			X	C	S	AP [1]	AP [0]	NS	0	1	0	1MB Page
PPN for 2MB, PA[35:21]		X		C	S	AP [1]	AP [0]	NS	1	0	0	2MB Page
PPN for 16MB, PA[35:24]	X			C	S	AP [1]	AP [0]	NS	1	1	0	16MB Page
X									0	0	0	Unmapped Page

Figure 1-7 FLPD Format

Legend	Description
FLPD[2:0]	Indicates the page size, whether 1 MB, 2 MB, or 16 MB, and the pointer of SLPT. For each page size, the corresponding bit-encoding is assigned as illustrated in Figure 1-7. Unmapped page indicates that the virtual page is not allocated to a physical page.
FLPD[3], Non-Secure (NS)	If NS bit is set, the corresponding page is allocated to a non-secure domain. Otherwise, the corresponding page is allocated to a secure domain.
FLPD[5:4], Access Permission (AP)	If AP[0] is set, the corresponding page has read access. If AP[1] is set, write access is allowed for the corresponding page. It is only valid for 1 MB/2 MB/16 MB pages.
FLPD[6], Shareablility (S)	Indicates that the access for the corresponding page must snoop the caches of CPU because the data in the corresponding page is cached in caches of CPU. It is only valid for 1 MB/2 MB/16 MB pages.
FLPD[9:7], Cache (C)	This value is propagated to AxCACHE of AXI transaction.
FLPD[31:6], Physical Page Number (PPN), or SLPT base address	Indicates the PPN for 1 MB/2 MB/16 MB pages. It also indicates SLPT base address to perform a second-level page table walk.

1.2.1.4 Second-Level Page Table Organization

Second-Level Page Table (SLPT) is a list of page descriptors that is indexed by VA[***VPN_WIDTH*** - 1:12], where ***VPN_WIDTH*** can be 20 or 24.

Following are the specifications of SLPT:

- Each page descriptor has 4 byte or 32-bit width and must be stored in aligned.
- Each page descriptor represents a mapping for 4 KB or 64 KB contiguous memory region.
- Base of SLPT must be 1-KB-aligned.
- 16 same SLPDs must be placed on the corresponding VPN location for 64 KB pages as illustrated in *Figure 1-8* and must be aligned by a multiple of 16.

Figure 1-8 illustrates an example of SLPT organization.

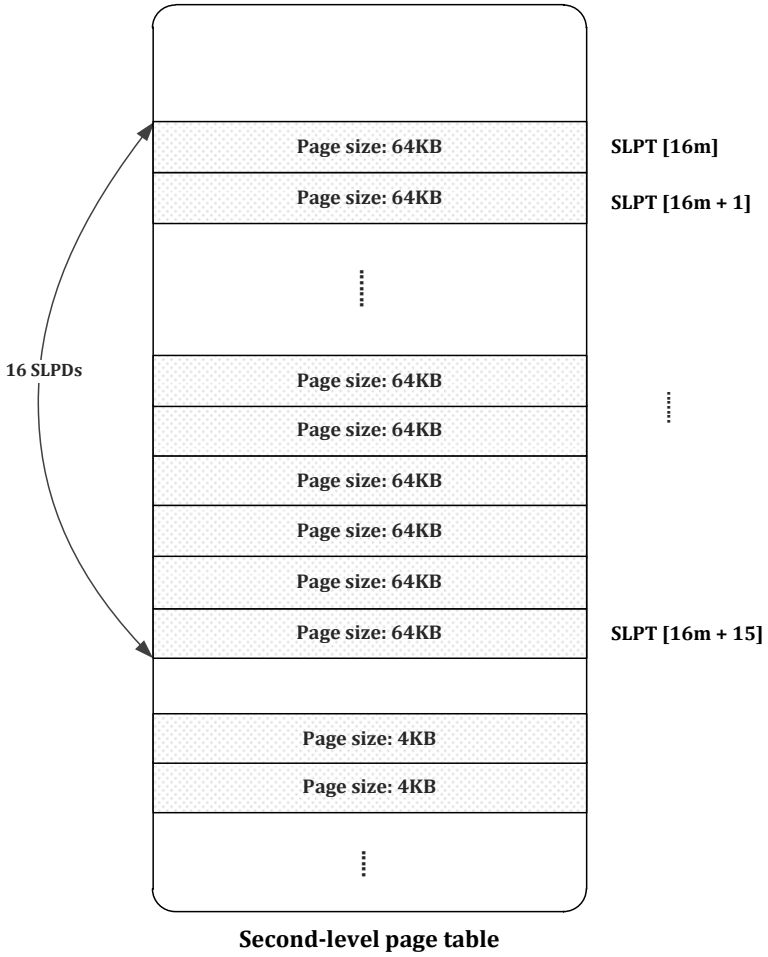

Figure 1-8 Example of SLPT Organization

Figure 1-9 illustrates the Second-Level Page Descriptor (SLPD) format.

Second-Level Page Descriptor (SLPD) Format

31:12	11:8	7:5	4	3	2	1	0	Page Size
PPN for 4KB, PA[35:12]		C	S	AP [1]	AP [0]	1	X	4KB Page
PPN for 64KB, PA[35:16]	X	C	S	AP [1]	AP [0]	0	1	64KB Page
X						0	0	Unmapped Page

Figure 1-9 SLPD Format

Legend	Description
SLPD[1:0]	Indicates the page size, that is, 4 KB and 64 KB. The corresponding bit-encoding is assigned as illustrated in Figure 1-9. Unmapped page indicates the virtual page is not allocated to a physical page.
SLPD[3:2], Access Permission (AP)	If AP[0] is set, read access is allowed for the corresponding page. If AP[1] is set, write access is allowed for the corresponding page
SLPD[4], Shareablility (S)	Indicates the access for the corresponding page must snoop the L2 caches of CPU because the data in the corresponding page is cached in L2 caches of CPU.
SLPD[7:5], Cache (C)	This value is propagated to AxCACHE of AXI transaction.
SLPD[31:8], PPN	Indicates the PPN for 4 KB and 64 KB pages.

1.2.1.5 Address Translation by FLPD

Address translation at the first-level is performed as the following steps:

- FLPD is read from FLPT by PTW
- If FLPD is a page descriptor for 1 MB, 2 MB, or 16 MB page, then FLPD calculates PA.
- If FLPD is a base address of SLPT, then the second-level address translation is performed.
- If FLPD is a page descriptor for unmapped page, then the fault must be generated.

To read FLPD from FLPT, first-level Page Table Walk (PTW) request must be generated. Use the following formula to calculate the physical address of the first-level PTW.

Physical address of first-level PTW (36-bit) = FLPT_BASE_ADDR[35:0] + (4 × VA [VA_WIDTH - 1:20])
FLPT_BASE_ADDR is the software programmed base address of FLPT

NOTE: If multiple page tables supporting is enabled, FLPT_BASE_ADDR differs depending on the request.

If FLPD is a page descriptor for 1 MB, 2 MB, or 16 MB page, use the following formulae to calculate PA of the request.

PA for 1 MB page (36-bit) = {FLPD[31:16], VA[19:0]}
PA for 2 MB page (36-bit) = {FLPD[31:17], VA[20:0]}
PA for 16 MB page (36-bit) = {FLPD[31:20], VA[23:0]}

Figure 1-10 illustrates the address translation process for a 1 MB page.

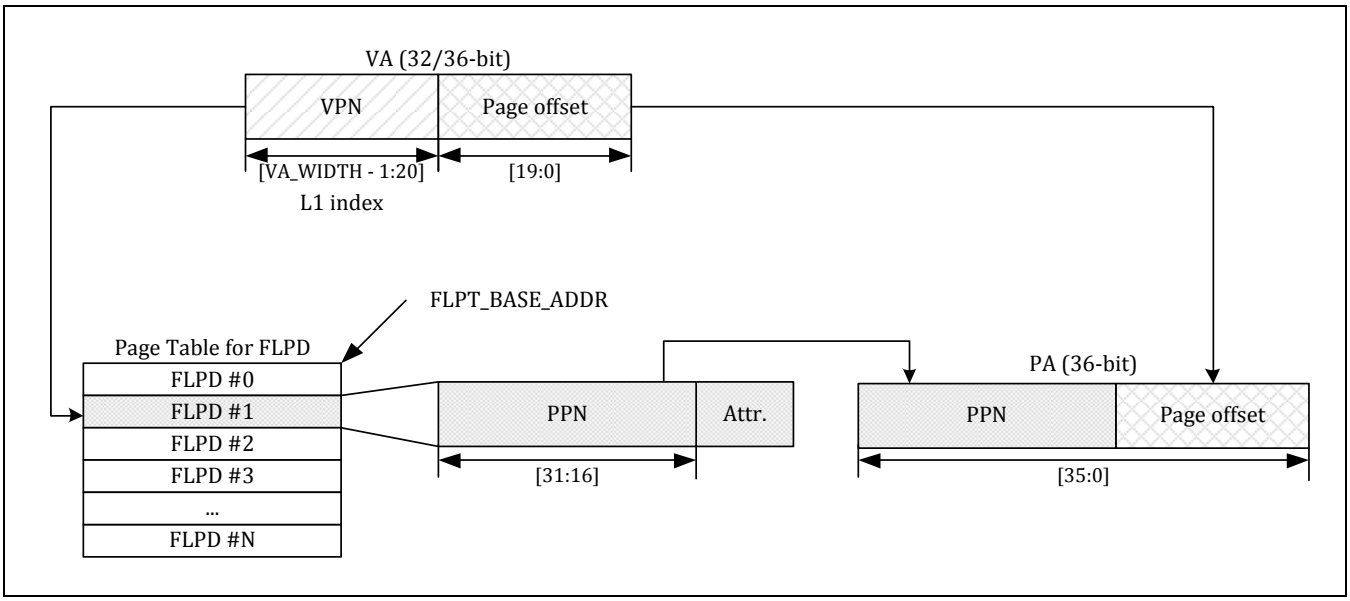

Figure 1-10 Translation Process for a 1 MB Page

1.2.1.6 Address Translation by SLPD

If FLPD indicates 4 KB/64 KB page size, SLPD must perform the address translation.

Following steps are performed for address translation at the second-level:

- SLPT reads SLPD
- SLPD calculates PA
- If SLPD is a page descriptor for an unmapped page, then the fault must be generated.

To read SLPD from SLPT, second-level Page Table Walk (PTW) request must be generated. Use the following formula to calculate the physical address of the second-level PTW.

Physical address of second-level PTW (36-bit) = {FLPD[31:6], VA[19:12], 2'b00};
FLPD[31:6] is the base address of SLPT

Use the following equation to calculate PA of the request for 4 KB and 64 KB pages.

PA for 4 KB page (36-bit) = {SLPD[31:8], VA[11:0]}
PA for 64 KB page (36-bit) = {SLPD[31:12], VA[15:0]}

Figure 1-11 illustrates the address translation process for a 4 KB page.

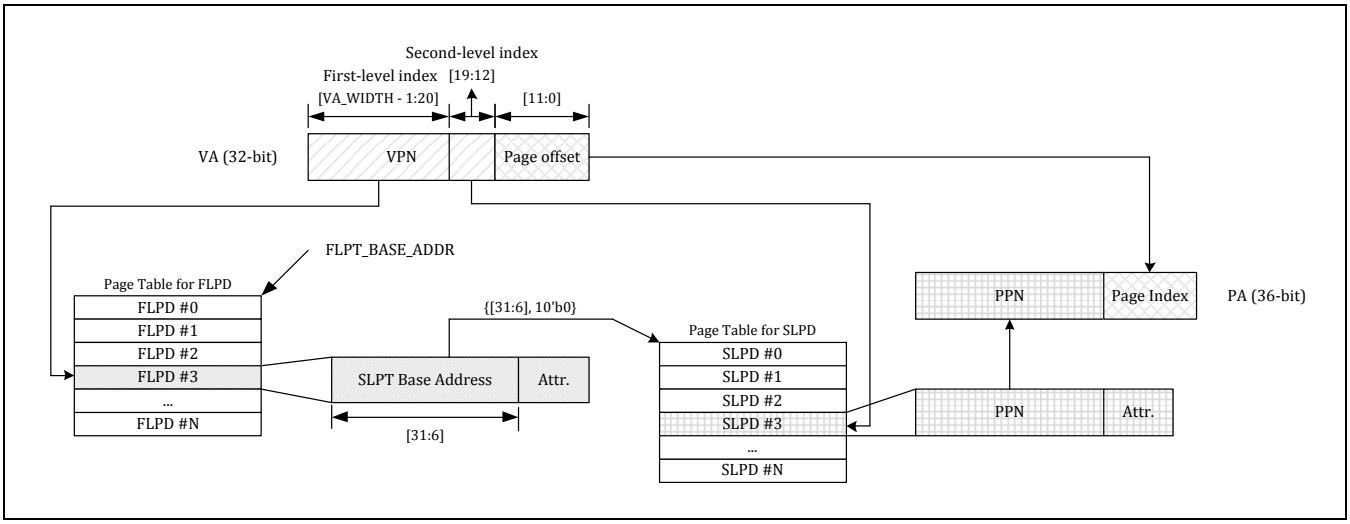

Figure 1-11 Translation Process for a 4 KB Page

1.2.1.7 Address Translation Processes

Figure 1-12 illustrates the address translation processes in SysMMU.

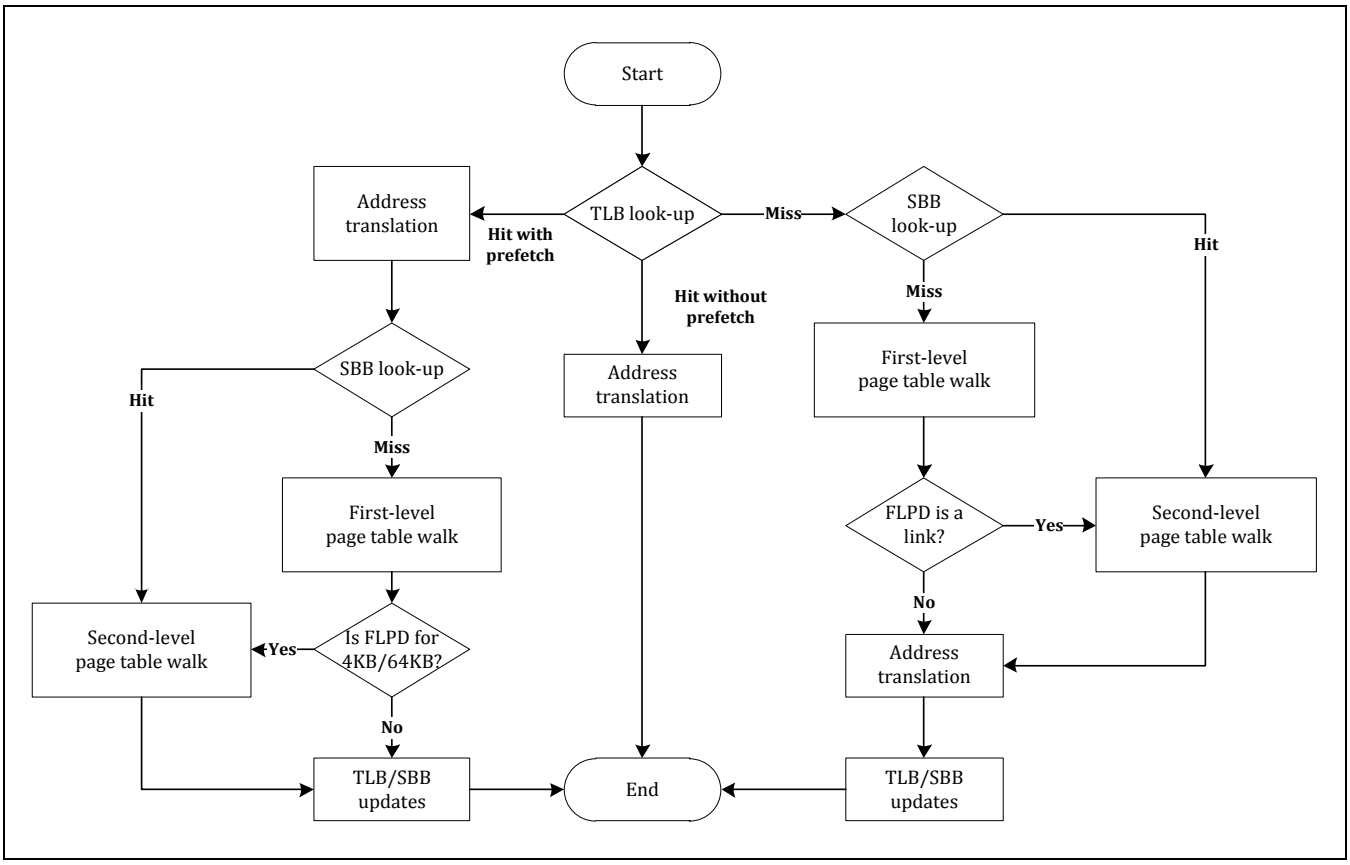

Figure 1-12 Address Translation Processes in SysMMU

When a request with VA is received, SysMMU performs looking-up Translation Look-aside Buffer (TLB), which is a cache containing recently-used FLPDs and SLPDs for address translation.

Looking-up TLB results in one of the following three responses:

- hit-without-pre-fetch
- hit-with-pre-fetch
- miss

Following steps are performed if TLB result is a hit-with-pre-fetch:

- Page descriptor in TLB performs the address translation
- Looking-up Second-level page table Base address Buffer (SBB) is performed for pre-fetch
- Depending on the SBB look-up result, first-level or second-level Page Table Walk (PTW) is performed
- The page descriptors fetched by PTW are cached in TLB and SBB

SBB is a cache containing FLPDs for 4 KB/64 KB pages. In other words, SBB has the recently-used base addresses of SLPT. If SSB look-up result is hit, only second-level page table walk must be performed.

If TLB look-up result is hit-without-pre-fetch, use the page descriptor in TLB to perform the address translation. If TLB look-up result is a miss, then the address translation is performed after the page table walks are completed, as described in the following steps:

- SBB Look-up is performed
- Depending on the result of SBB look-up, first-level or second-level page table walk is performed to get the required page descriptor for address translation
- Address translation is performed by page descriptors that are fetched by the page table walks
- The page descriptors fetched by PTW are cached in TLB and SBB

1.2.1.8 Hit-Under-Miss and Miss-Under-Miss

SysMMU supports Hit-under-Miss (HuM) and Miss-under-Miss (MuM) to reduce the impact of latency on subsequent requests due to TLB miss.

HuM enables the subsequent requests with TLB hit to be placed ahead of a request with TLB miss. Thus, the requests with TLB hit does not need to wait for completion of address translation of a request with TLB miss. Write data buffers take considerable silicon area and are required to support HuM for write requests, hence, HuM is supported for only read requests. Ordering requirement between transactions with the same ID should be maintained. That is, only reordering transactions by HuM occurs between the transactions with different ID.

MuM performs page table walks for multiple requests with TLB miss, so that initial miss penalty is not propagated to the subsequent requests with TLB miss. MuM is supported for both read/write requests.

To support HuM and MuM, SysMMU accepts multiple requests for address translation. Implementation determines the maximum capacity to store multiple requests and not the performance and area trade-off.

1.2.1.9 Access Permission

SysMMU supports a protection feature based on the access permission in the page descriptor. If a request does not have access permission for the page, the request is defined as an illegal request and an access permission fault is generated.

Run-time programmed option enables the access permission check. If it is disabled, access permission check is not performed, so that the request is translated even if the request does not have access permission for the page. Default value disables the access permission check.

1.2.1.10 Security Attribute

SysMMU supports a protection feature based on the security attribute in the page descriptor. For security attribute checking, apply the following rules:

- Non-secure request accesses the only non-secure physical page and cannot access the secure physical page resulting in security fault.
- Secure request accesses both non-secure and secure physical page. But, if secure request wants to access non-secure page table, the request with translated physical address must be changed to non-secure request.

Run-time programming enables the security attribute checking. If security attribute checking is disabled, then SysMMU ignores the NS field of page descriptor during the address translation. Default value of checking security attribute is disabled.

Security attribute of secure request trying to access a non-secure page must be changed to non-secure to avoid the following scenario.

Cache in CPUs differentiates the secure and non-secure cache lines. Let us presume that the coherency between CPU cache and memory is broken.

- Secure master generates a shareable transaction to access the non-secure region.
- SysMMU and cache coherency interconnect forward the secure transaction to CPU.
- CPU performs the cache look-up. When the cache has the dirty line and it is tagged with non-secure, the cache miss occurs.
- Transaction accesses the out-dated data in the main memory as a result of cache miss.

1.2.2 Page Table Walk

Address translation requires a page descriptor for the corresponding virtual address. Page descriptors are located in the page table, hence SysMMU must perform the read operation to receive the page descriptor, which is called Page Table Walk (PTW).

SysMMU uses two-level page tables and hence PTWs are classified into the following two types:

- First-level PTWs that receive FLPDs from FLPT
- Second-level PTWs that receive SLPDs from SLPT

Alternatively, following are the classifications of PTWs based on their purpose:

- PTWs are issued if TLB does not have the page descriptor for address translation, fetch
- PTWs are issued for future use, pre-fetch

1.2.2.1 Fetch Size

A single PTW enables multiple page descriptors to exploit spatial locality in TLB. Number of page descriptors that a single PTW fetches from the page table is called fetch size.

First-level PTW fetches only one FLPD because a single FLPD covers 1 MB to utilize the spatial locality. Second-level PTW fetches multiple SLPDs depending on the run-time programmable parameter.

Table 1-1 describes the specification of fetch size.

Table 1-1 Specification of Fetch Size

PTW Types	Fetch Size
First-level PTW	1 FLPD
Second-level PTW	Fetch size is run-time programmable • 4 SLPDs • 8 SLPDs • 16 SLPDs • 32 SLPDs • 64 SLPDs • 128 SLPDs

1.2.2.2 Number of Transactions and Burst Length

SysMMU supports less than 64 KB PTW for higher utilization of downstream system. Table 1-2 describes the number of transactions and burst length for a single PTW depending on the fetch size.

Table 1-2 Number of Transactions and Burst Length Depending on the Fetch Size

PTW Types	Fetch Size	Number of Transactions	Burst Length
First-level PTW	1 FLPD	1	1
Second-level PTW	4 SLPDs	1	1
	8 SLPDs	1	2
	16 SLPDs	1	4
	32 SLPDs	2	4
	64 SLPDs	4	4
	128 SLPDs	8	4

1.2.2.3 Physical Address of Transactions

Table 1-3 describes the physical addresses of transactions for PTW requests.

Table 1-3 Physical Address Derivation

PTW Types	Fetch Size	Transaction Number (0 – 3)	Physical Address of Transactions
First-level PTW	1 FLPD	0	{FLPT_BASE_ADDR[35:14], VA[31:22], 4'b00} - 32 bit VA {FLPT_BASE_ADDR[35:18], VA[35:22], 4'b00} - 36 bit VA
Second-level PTW	4 SLPDs	0	{FLPD[31:6], VA[19:14], 4'b00}
	8 SLPDs	0	{FLPD[31:6], VA[19:15], 5'b00}
	16 SLPDs	0	{FLPD[31:6], VA[19:16], 6'b00}
	32 SLPDs	0	{FLPD[31:6], VA[19:17], 1'b0, 6'b00}
		1	{FLPD[31:6], VA[19:17], 1'b1, 6'b00}
	64 SLPDs	0	{FLPD[31:6], VA[19:18], 2'b00, 6'b00}
		1	{FLPD[31:6], VA[19:18], 2'b01, 6'b00}
		2	{FLPD[31:6], VA[19:18], 2'b10, 6'b00}
		3	{FLPD[31:6], VA[19:18], 2'b11, 6'b00}
	128 SLPDs	0	{FLPD[31:6], VA[19], 3'b000, 6'b00}
		1	{FLPD[31:6], VA[19], 3'b001, 6'b00}
		2	{FLPD[31:6], VA[19], 3'b010, 6'b00}
		3	{FLPD[31:6], VA[19], 3'b011, 6'b00}
		4	{FLPD[31:6], VA[19], 3'b100, 6'b00}
		5	{FLPD[31:6], VA[19], 3'b101, 6'b00}
		6	{FLPD[31:6], VA[19], 3'b110, 6'b00}
		7	{FLPD[31:6], VA[19], 3'b111, 6'b00}

NOTE: Data-width of bus interconnect for PTW is 128-bit.

1.2.2.4 Transaction ID

PTW requests and master requests use the same physical channel. Hence, those requests must be separable by the least significant bit of transaction ID.

SysMMU issues multiple transactions for different PTW requests, where each PTW request uses different resource in SysMMU. SysMMU also issues multiple transactions for a single PTW request depending on the fetch size. Therefore, transactions for PTW are separable for matching the internal resource with each transaction using a transaction ID.

SysMMU must support unique-ID feature for PTW requests to eliminate the reorder buffers in the interconnect backbone.

To integrate the above requirement of transaction ID, SysMMU has the following policy to generate a transaction ID:

- Transaction ID for PTW: {PTW_ID, TR_NUM, 1'b1}
- Transaction ID for master request: {original transaction ID, 1'b0}
- Maximum width of the above transaction IDs determines the ID width.

PTW_ID identifies the PTW request that the transaction belongs to. TR_NUM is the transaction number when the fetch size is 32-SLPD, 64-SLPD, or 128-SLPD.

1.2.2.5 Transaction Specification for Other Fields

Table 1-4 describes the specification of other fields in AXI transaction for PTW.

Table 1-4 Specification of Other Fields in AXI Transaction for PTW

AXI Field	Value	Description
ARSIZE	0x4	Only 128 bits data width supported
ARBURST	0x1	It must be incremental-burst
ARLOCK	0x0	It must be normal access
ARPROT	{1'b0, AxPROT[1], AxPROT[0]} ARPROT[1:0] are inherited from the request of master	
ARCACHE	4'b0010 (default)	PTW_ARCACHE configures it. Default is normal, non-cacheable, and non-bufferable.
ARQOS	Inherited from AxQoS in master request when QOS_OVERRIDE is zero. QOS_VALUE when QOS_OVERRIDE is non-zero.	QOS_OVERRIDE field selects the mode for setting ARQOS.
ARUSER	Inherited from AxUSER in master request	
ARSHARABLE	Value of PT_SHARABLE	

1.2.2.6 Multiple Page Table Walks

SysMMU issues multiple PTW requests to support Miss-under-Miss (MuM). Design-time configuration determines the maximum outstanding PTWs. To prevent any blocking because of lack of maximum outstanding PTWs, it is recommended that the maximum outstanding PTWs are set to the value equal to the number of ATMs.

1.2.2.7 Pending Hit

For multiple requests with the same VPN, duplicated PTW requests are issued due to consecutive TLB misses. This takes unnecessary bandwidth of interconnect, so that SysMMU removes the duplicated PTW requests. Only, initial PTW request is issued and the subsequent requests are kept pending for the completion of PTWs. It is called pending hit for the following requests.

1.2.3 Fault

SysMMU must detect the illegal requests during an address translation called fault. A fault occurs due to either a software bug or intention of a programmer. SysMMU must detect the fault and use an interrupt to notify it to the software. Then, software resolves the fault and resumes the operation.

1.2.3.1 Fault Type

Table 1-5 describes fault types that must be detected in SysMMU.

Table 1-5 Fault Types

Fault type	Description	Option to detect
Page	Request has VA and that is not mapped to any PA	Always
PTW access	Transactions for PTW request that have illegal address, so that error responses are received	Always
Access permission	The request tries to access the physical page that is not allowed to access	Detects it only when ENABLE_ACCESS_PROT is enabled
Security	Non-secure request tries to access the secure page	Detects it only when ENABLE_SECURITY_PROT is enabled

SysMMU must handle multiple faults simultaneously. However, SysMMU can notify only one of those faults to the software.

1.2.3.2 Logging Fault Request

When fault occurs, ensure that the following requested information is logged and is accessible by software:

- Virtual address: Software checks the page table indexed by this virtual address.
- Transaction ID: Software detects the DMA or IP that causes the fault.
- Direction of transaction (read or write): Software detects the transaction type that creates the fault.

1.2.3.3 Fault Handling Mode

There are two fault handling mode; stall mode and termination mode.

- Stall mode: When a fault is detected, SysMMU stalls further address translation request and wait fault retry by software programming.
- Termination mode: When a fault is detected, SysMMU terminates fault transaction by responding error to master which caused the fault.

1.2.3.4 Fault Handling

1.2.3.4.1 Fault Handling in Stall Mode

When SysMMU detects a fault, following fault handling procedures are performed:

- SysMMU blocks the subsequent requests to enter a block-mode
- SysMMU logs the request
- SysMMU generates an interrupt to notify the fault to the software
- Responding to the interrupt, software uses the logged information to recover the fault state
- After resolving the fault, software clears the interrupt
- SysMMU retries address translation of the fault request

1.2.3.4.2 Fault Handling in Termination Mode

When SysMMU detects a fault, following fault handling procedures are performed:

- SysMMU blocks the subsequent requests to enter a block-mode
- SysMMU logs the request
- SysMMU generates an interrupt to notify the fault to the software
- SysMMU responds error to master which caused the fault. If the access type of fault request is write access, corresponding write data is consumed by SysMMU after then SysMMU responds error.
- Responding to the interrupt, software uses the logged information to recover the fault state
- After resolving the fault, software clears the interrupt

1.2.4 Translation Look-Aside Buffer

Translation Look-Aside Buffer (TLB) uses spatial and temporal locality to store the page descriptors to provide a translation service without memory access to page table.

TLB has a set-associative structure, whose shape is determined by way, set, and line numbers as illustrated in *Figure 1-13*. Each number is configurable at the design time based on the address pattern and target miss rate of TLB. The following lists provide the permissible value for each number.

- Number of way: positive integer value (1 – 32)
- Number of set: 1, 2, 4, 8, 16, 32, 64
- Number of line: 4 (4 denotes that each line consists of four entries for the consecutive page descriptors. Each entry stores a single page descriptor.)

Figure 1-13 illustrates the TLB based on set-associative cache structure.

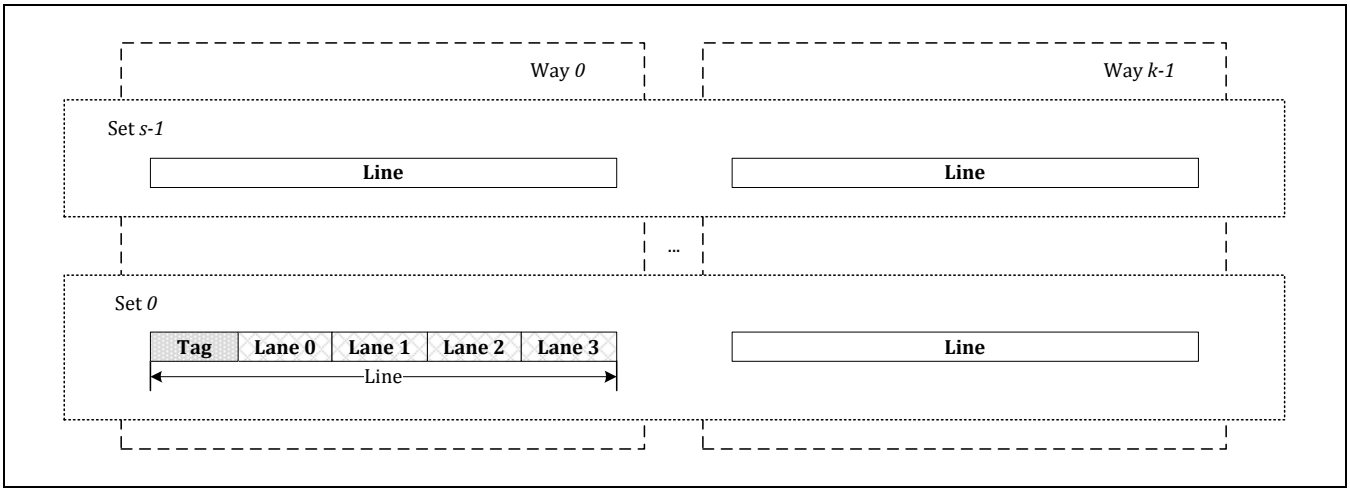

Figure 1-13 TLB Based on Set-Associative Cache Structure

SysMMU instantiates multiple TLBs. Each TLB has own configuration of number of set/way. It is due to optimize TLB capacitance according to address access pattern of traffic stream which is dedicated own TLB.

Figure 1-14 illustrates the multiple TLBs.

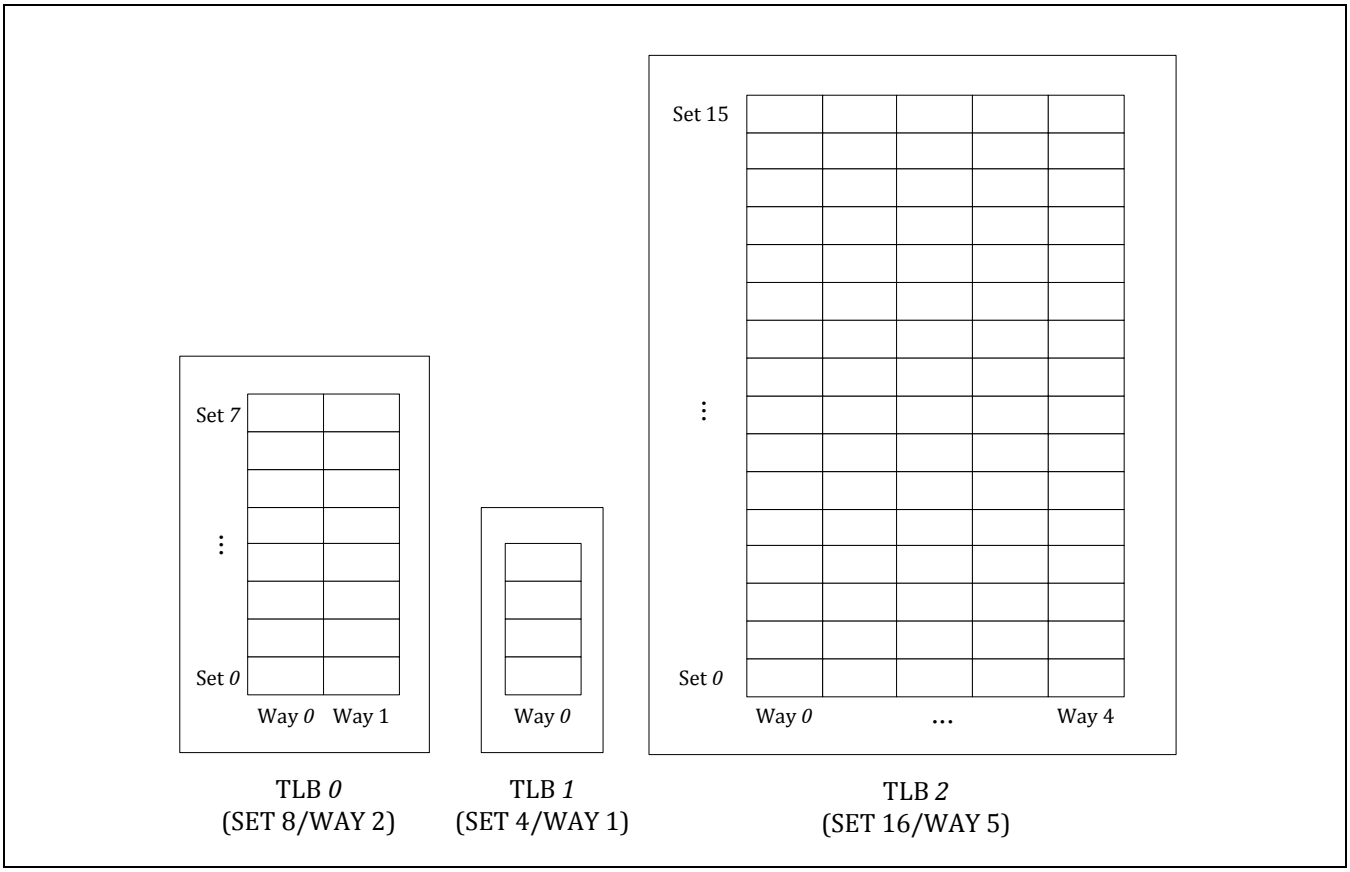

Figure 1-14 Multiple TLB structure

TLB also supports an additional logic for pre-fetch function. For more information, refer to the Section *1.2.5 Pre-Fetch*.

1.2.4.1 TLB Tag Specification

TLB has a set-associative structure. Therefore, each TLB line must have a tag to find the TLB line for a TLB look-up. *Table 1-6* describes the fields included in a TLB tag.

Table 1-6 Specification of TLB Tag

Name	Description
VPN	Represents the virtual page number. Each TLB line has four page descriptors and each page descriptor is used for translation of 4 KB region. Hence, lower 2-bit of VPN need not be cached in TLB line.
SC	Represents the security attribute of the page table that page descriptors in the TLB line belongs to.
PTID	Represents an identifier for page table that page descriptors in the TLB line belongs to.
PS	Represents the page size of page descriptors in the TLB line.
V	Represents the validity of each lane in the TLB line. Lane represents single page descriptor storage. Therefore, a TLB line has four lanes.

1.2.4.2 TLB Data Specification

Each TLB line has the data required for address translation. *Table 1-7* describes the fields included in the TLB data.

Table 1-7 Specification of TLB data

Name	Description
PPN	Represents the physical page number.
S	Represents the shareability attribute of the physical page. Data that belongs to the physical page is shared in multiple caches, so that cache coherency operation is required.
AP	Represents the access permission attribute of the physical page. Four types of access permission attributes must be supported, namely read-and-write, read-only, write-only, and not-allowed.
NS	Represents the security attribute of the physical page.

1.2.4.3 Consideration of Multiple Page Granularity

TLB line must consider multiple page granularities. *Figure 1-15* illustrates how TLB line stores page descriptors based on the page size.

Tag		Data			
PS	V	LANE 0	LANE 1	LANE 2	LANE 3
4KB	4'b1111	PPN 0	PPN 1	PPN 2	PPN 3
4KB	4'b1001	PPN 0	Unmapped	Unmapped	PPN 3
4KB	4'b0000	Unmapped	Unmapped	Unmapped	Unmapped
64KB	4'b1111	PPN 0	PPN 0	PPN 0	PPN 0
1MB	4'b1111	PPN 0	PPN 0	PPN 0	PPN 0
1MB	4'b0000	Unmapped	Unmapped	Unmapped	Unmapped
2MB	4'b1111	PPN 0	PPN 0	PPN 0	PPN 0
16MB	4'b1111	PPN 0	PPN 0	PPN 0	PPN 0

Figure 1-15 Caching Specification TLB Line Considering Multiple Page Granularity

1.2.4.4 TLB Dedication

TLB consists of multiple configurable sub TLBs to provide a certain traffic stream for exclusive purpose called TLB dedication. TLB that are shared with other traffic streams are called default TLB. *Figure 1-16* illustrates the structure of TLB dedication.

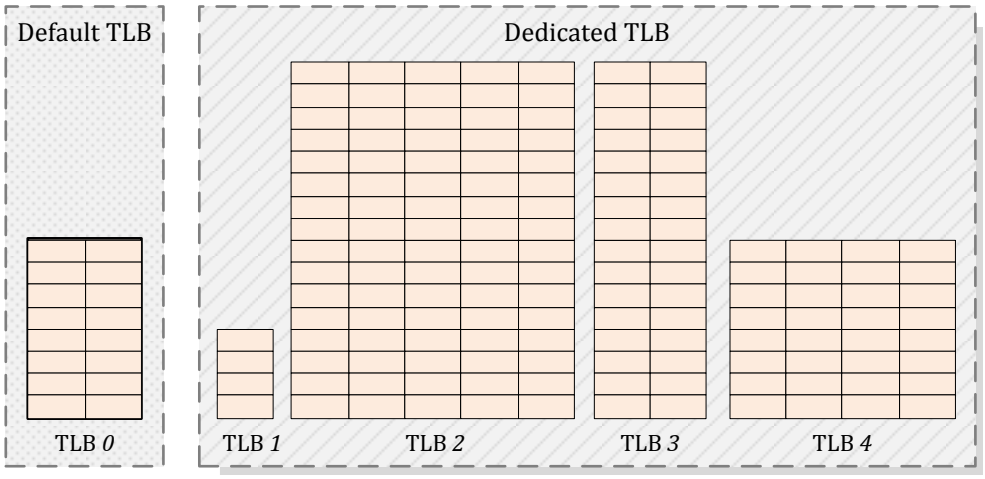

Figure 1-16 Structure of TLB dedication

For the TLB dedication, address range and ID are matched with TLB matching configuration which is controlled by SFRs. Address range is configured only for traffics in that address range to exclusively use the TLB. ID mask and value is configured only for traffics with the configured ID values to exclusively use the TLB. *Figure 1-16* describes the configurations for TLB dedication.

Table 1-8 Configurations for TLB Dedication

Configuration	Description
PORT_MASK	Represents that whether each port can be matched to the TLB
TARGET_CH	Represents the target channel to use this dedicated TLB, namely read-write-allocated, read-allocated, write-allocated, and not-allocated.
TARGET_ID_MASK	Represents a mask for ID matching to identify the traffic stream.
TARGET_ID_VALUE	Represents a value for ID matching to identify the traffic stream.
TARGET_SVA	Represents a start address to identify the traffic stream.
TARGET_EVA	Represents an end address to identify the traffic stream.

1.2.4.5 Replacement Policy - Bit-Based PLRU

When updating a TLB line, a victim TLB line must be selected. Multiple TLB lines are the candidates for replacement. In TLB, bit-based Pseudo Least Recently-Used (PLRU) is used for victim selection because of the following reasons:

- PLRU utilizes a temporal locality.
- Bit-based PLRU is efficiently implemented when the number of replacement candidates is an integer value.

Figure 1-17, Figure 1-18, Figure 1-19, Figure 1-20, and Figure 1-21 illustrate the examples of bit-based PLRU algorithm. Numbers in circles represent the state bit of ways, where the corresponding way is recently-used if it is set to 1. Numbers in rectangles represents the way number. Bit-based PLRU selects a way among the ways that has the state value of zero as a victim.

In the initial state, all of state bits must be initialized to 0. In this case, all the ways are the replacement candidates. In this case, PLRU selects the way with the lowest index as a victim.

At every TLB look-up or TLB update, the state bit for accessed way must be set to 1. This indicates that the way is a recently-used way.

If all the state bits become 1, then the only state bit of last accessed way is set to 1 and other states bits are set to 0.

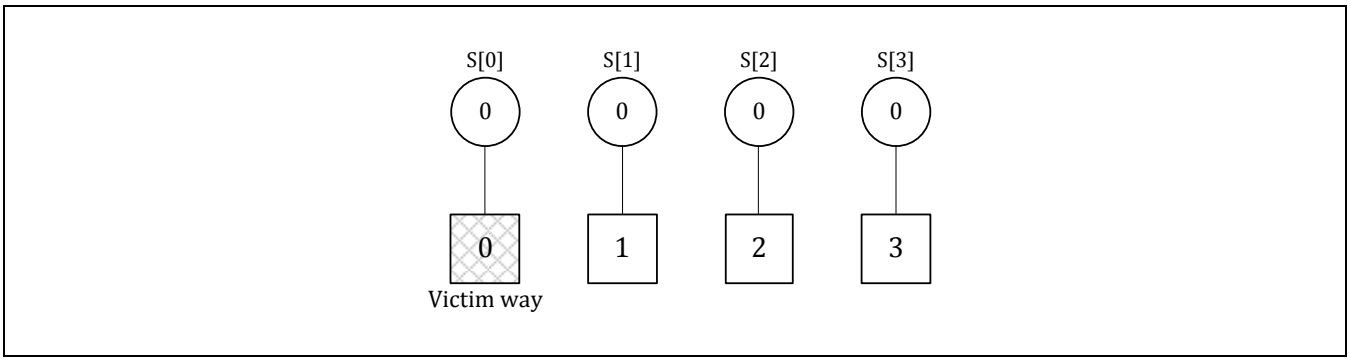

Figure 1-17 Initial State of Bit-Based PLRU

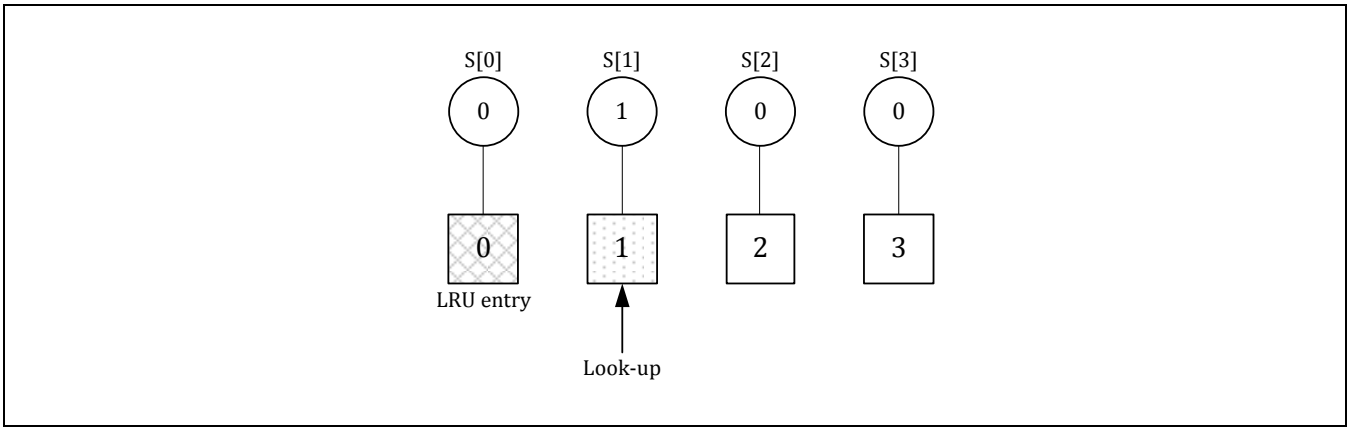

Figure 1-18 Next State after TLB Look-Up for Way 1

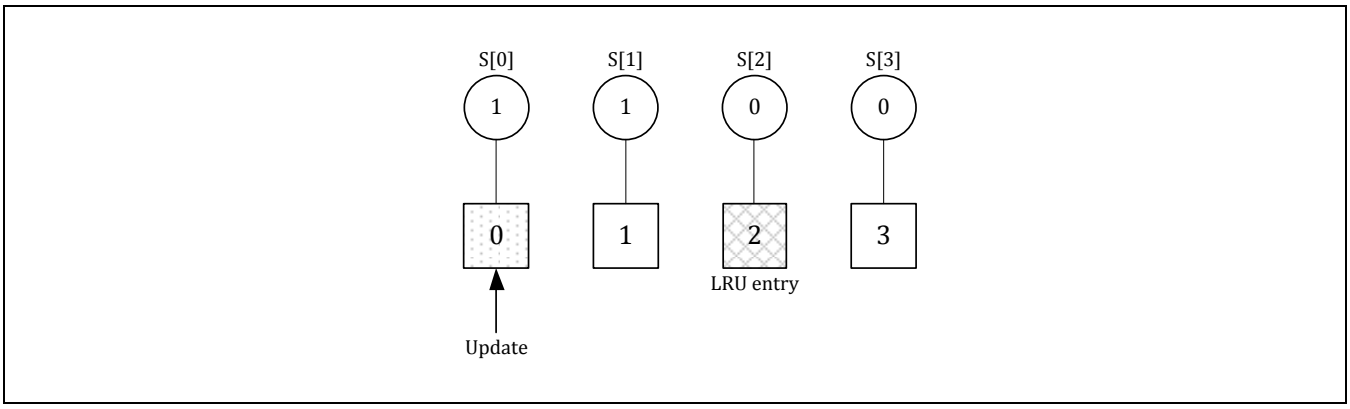

Figure 1-19 Next State after TLB Update for Way 0

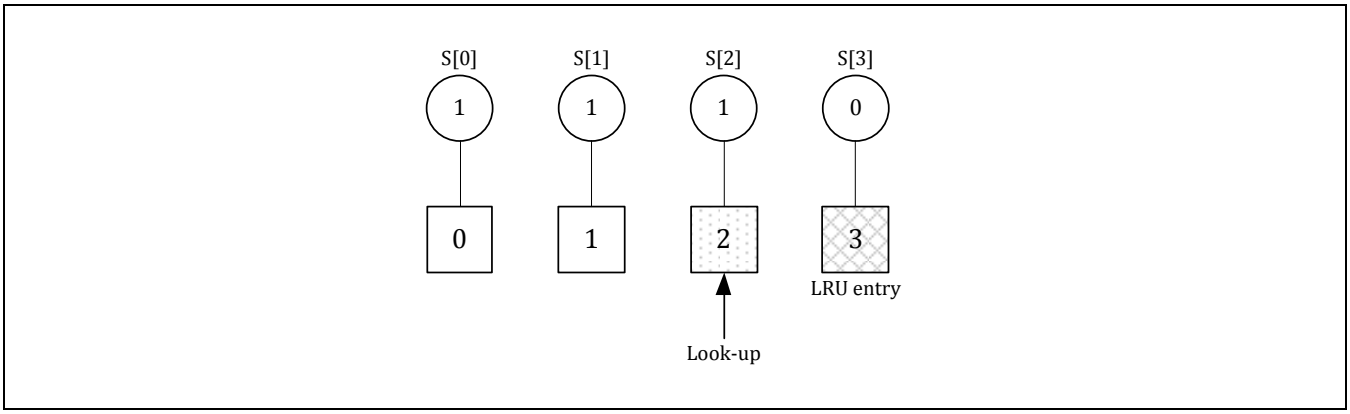

Figure 1-20 Next State after TLB look-Up for Way 2

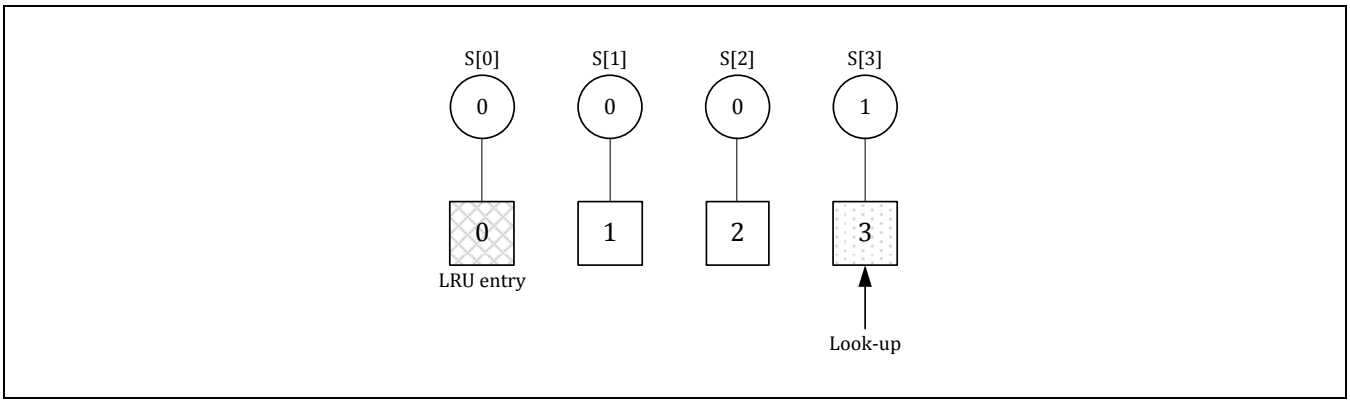

Figure 1-21 Next State after TLB Look-Up for Way 3

1.2.4.6 Security and Multi-Page Table Awareness

SC and PTID fields in each TLB tag support security and multi-page table. Even if VPN matches with the request, if SC and PTID fields are not matched with SC and PTID of the request, TLB must return TLB miss. Following factors generate the SC and PTID of the request:

- SC: Use ARPROT[1] or AXPROT[1] signal in AXI protocol
- PTID: Matching the ID and direction (read/write) of the request with each configuration derives the PTID. Note that the ID and direction (read/write) configurations for each page table are already provided.

1.2.4.7 Preventing the Duplicated Page Descriptors Storage with the Same VPN

Multiple requests for the same page leads to duplication for the pre-fetched page in TLB because pre-fetch request is issued whenever TLB returns a hit for the address translation request. This degrades TLB performance because pre-fetch requests replace the effective TLB entries.

To prevent this, SysMMU must check if ATI (address translation information) for the pre-fetched page is already cached in TLB. If ATI is already cached, do not issue the pre-fetch request.

There is still possibility to store duplicated page descriptors in TLB when consecutive address translation requests for the same page create TLB hit and the corresponding ATI to pre-fetch is not cached. In this case, multiple pre-fetch requests are issued for the same page that leads to duplication in TLB.

To prevent this, TLB must be updated such as that TLB contains the page descriptors for different VPNs.

1.2.5 Pre-Fetch

SysMMU supports pre-fetch to provide lower latency penalty for traffic streams with deterministic address patterns.

1.2.5.1 Pre-Fetch Configuration

Each TLB has its own pre-fetch configurations. *Table 1-9* describes the pre-fetch configurations.

Table 1-9 Pre-Fetch Configurations

Configuration	Description
PF_EN	Represents whether pre-fetch is used or not.
PF_DIR	Represents an address direction to pre-fetch.
FETCH_SIZE	Represents the size of page descriptors to fetch. It determines the number of page descriptors fetched by a single PTW for normal fetch and pre-fetch. Normal fetch refers to a PTW due to TLB miss.

1.2.5.2 Condition to Issue a Page Table Walk for Pre-Fetch

Issue a PTW for pre-fetch when the following conditions are satisfied:

- Pre-fetch configuration for the master request is enabled.
- TLB look-up result for the master request is TLB hit.
- TLB does not contain the page descriptors that the pre-fetch tries to fetch to prevent redundancy.

1.2.5.3 Prediction of Pre-Fetch Direction

Some masters such as VPP and MC-SCALER change their address patterns frame-by-frame during a hardware event without any software intervention. Even if software sets the appropriate pre-fetch direction configuration based on the address pattern, software cannot set it up on time for such masters. This increases the translation latency due to incorrect pre-fetching. Therefore, SysMMU provides a hardware predictor for pre-fetch direction.

Hardware predictor is based on bi-modal predictor, which consists of four states as illustrated in *Figure 1-22*. Predictor compares the recently-accessed address with the current address of master request. Based on the comparison result, the state machine is updated as illustrated in the *Figure 1-22*. Then, the state determines the pre-fetch direction. If the compared addresses are same, the state is not changed.

Figure 1-22 illustrates that if the address pattern is changed, then at least one miss-prediction happens. Even if two-state predictor removes such one miss-prediction, two-state predictor is not preferred when the address is increasing/decreasing for only one transaction. When this occurs in an image-rotation scenario in VPP or MSCL, use a four-state predictor. Masters issue multiple transactions during short period and it saves the penalty of one miss-predication. Hence, two different PTWs for pre-fetch are issued, first is issued by miss-prediction and the other is issued by correct prediction. To hide the latency penalty due to one miss-prediction, support the MUM by processing two PTWs in parallel.

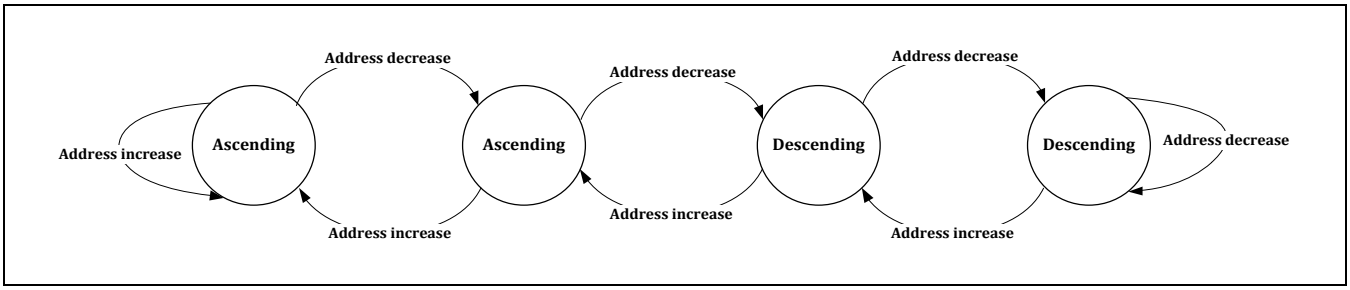

Figure 1-22 Four-State Hardware Predictor to Determine Pre-Fetch Direction

1.2.6 Second-Level Page Table Base Address Buffer

Second-level page table Base address Buffer (SBB) is a cache that stores FLPDs that contain the base address of Second-Level Page Table (SLPT). SBB eliminates the first-level page table walking for 4 KB and 64 KB pages to reduce the translation latency at TLB miss.

SBB has a fully-associative structure, where the number of lines determines its size. The number of lines is configurable at the design time, which is 8, 16, or 32. *Figure 1-23* illustrates the SBB based on the fully-associative cache structure.

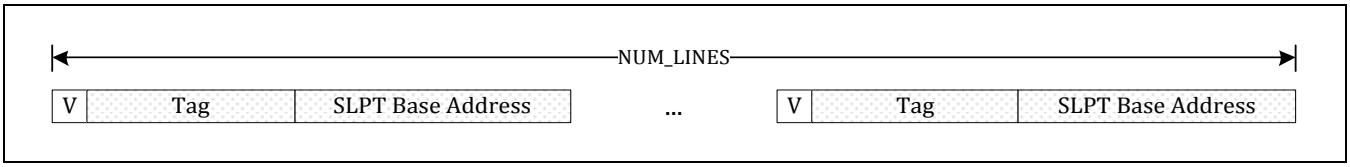

Figure 1-23 SBB Based on the Fully-Associative Cache Structure

1.2.6.1 SBB Tag Specification

Each SBB line must have a tag for a SBB look-up. *Table 1-10* describes the fields included in an SBB tag.

Table 1-10 Tag Specification in SBB

Name	Description
VPN	Represents the virtual page number. Each SBB line represents 1 MB page region, hence do not store the lower 10-bit in SBB line.
SC	Represents the security attribute of the page table that page descriptor in the SBB line belongs to.
PTID	Represents an identifier for page table that page descriptor in the SBB line belongs to.
V	Represents the validity of SBB line.

1.2.6.2 SBB Data Specification

Each SBB line has the data required for second-level page table walk. *Table 1-11* describes the fields included in the SBB data.

Table 1-11 Data Specification in SBB

Name	Description
SLPT_BASE_ADDR	Represents the base address of SLPT for 1 MB page region that VPN in the tag indicates.
NS	Represents the security attribute of 1 MB physical page.

1.2.6.3 Replacement Policy - Tree-Based PLRU

When you update an SBB line, use tree-based PLRU to select a victim TLB line for the following reasons:

PLRU utilizes a temporal locality.

Tree-based PLRU is implemented efficiently when the number of replacement candidates is a power of 2 like 2, 4, 6, 8, and so on.

Figure 1-24, *Figure 1-25*, and *Figure 1-26* illustrate tree-based PLRU algorithm, which consists of NUM_LINE - 1 state bits to construct a LRU tree. Each bit can be either 0 or 1. 0 indicates that LRU line is placed in the left-side set of lines. 1 indicates that LRU line is placed in the right-side set of lines.

In the initial state, initialize all the state bits to 0. Ensure that LRU line must be entry 0.

When any look-up or update occurs, ensure that state bits are updated. If a look-up accesses line 3, state bits placed in the path from root to line 3 are updated as LRU pointer indicates the other-side of entry from line 3. For example, S2_0 is set to 1 to LRU pointer indicates that the other-side set of line 3, that is, lines 4, 5, 6, or 7. S1_0 and S0_1 are also similarly updated. When a SBB update occurs, the state bits are also similarly updated.

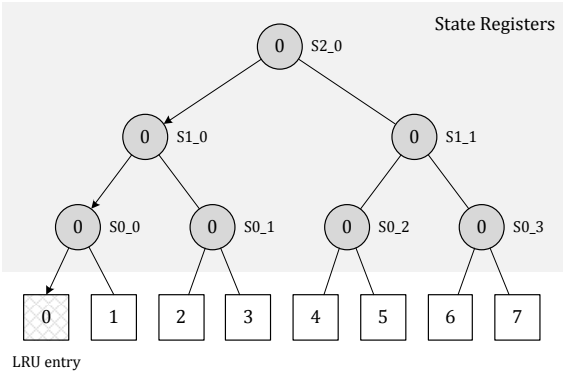

Figure 1-24 Initial PLRU Register State

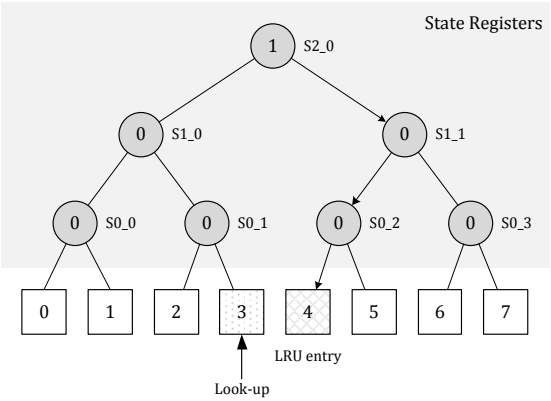

Figure 1-25 PLRU Register State When Look-Up Happens

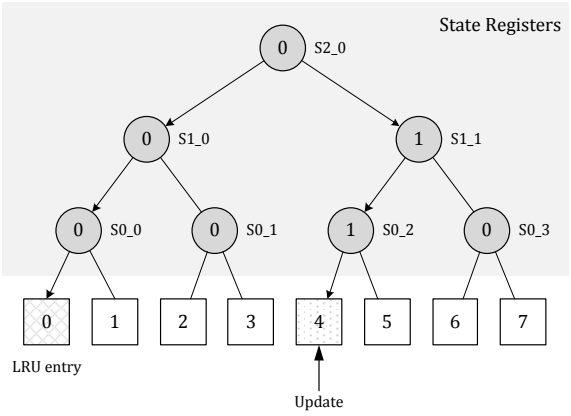

Figure 1-26 PLRU Register State When Update Happens

1.2.6.4 Security and Multi-Page Table Awareness

SC and PTID fields in each SBB tag support security and multi-page table. Even if VPN matches the request, if SC and PTID fields are not matched with SC and PTID of the request, SBB must return an SBB miss. Following factors generate the SC and PTID of the request:

- SC: Use ARPROT[1] or AXPROT[1] signal in AXI protocol.
- PTID: Matching the ID and direction (read/write) of the request with each configuration derives the PTID. Note that ID and direction (read/write) configurations for each page table are already provided.

1.2.6.5 Preventing the Duplicated Page Descriptors Storage with the Same VPN

SBB must check if the updated base address is already stored in SBB, because each SBB line has different base address for second-level page table.

1.2.7 Invalidation

TLB and SBB must be synchronized with the latest page table because virtual memory management software modifies the page descriptors based on the allocation or de-allocation of memory pages. For this synchronization, invalidation removes the lines that the software specifies to fetch the latest page descriptors for address translation.

SysMMU supports three types of invalidation commands, namely all-invalidation, range-invalidation, and VPN-invalidation. SysMMU supports multiple page tables, hence the invalidation commands must have a Page Table ID (PTID) parameter. All three invalidation commands are effective on the page descriptors of the same security domain. For example, secure OS cannot invalidate the lines that have the page descriptors of the non-secure page table. *Table 1-12* describes the invalidation commands.

Table 1-12 Invalidation Commands

Commands	Description	Parameters
All-invalidation	All the lines in TLB and SBB that is matched to a certain PTID or all the cache entries in the same security domain are invalidated. Following are the two types of all-invalidation: • All-invalidation for a specific page table: All the lines that have page descriptors of a specific page table are invalidated. • All-invalidation for all PTs: All the lines, irrespective of the page table, are invalidated. But, it must not invalidate the lines that have page descriptors of different security domain	• Page table ID • Security domain: Security attribute of programming interface determines it
Range-invalidation	TLB and SBB lines that have VPNs in a user-specified range are invalidated. PTID must be provided.	• Page table ID • Security domain • Range of virtual pages
VPN-invalidation	TLB and SBB lines that have the specified VPN are invalidated. PTID must be provided.	• Page table ID • Security domain • Virtual page number

Ensure that the following conditions are satisfied when invalidating TLB and SBB lines:

- There must be no outstanding transaction with the virtual address that is targeted during the invalidation.
- Software must appropriately perform invalidation whenever there is any modification in page tables.
- During invalidation, the page descriptors that are being fetched by PTW should not be updated to TLB and SBB. It is because the fetched page descriptors have old data before the modification.
- During invalidation, SysMMU does not block the incoming transactions.
- During invalidation, TLB and SBB look-ups requests are not serviced and are blocked until the invalidation completes.

1.2.8 Secure Page Table

SysMMU simultaneously provides both secure and non-secure page tables. It shares a single SysMMU with secure and non-secure traffics. It also enables virtual memory management in secure-domain. That is, it provides on-demand memory allocation for secure applications without any reserved or carved-out memory. *Figure 1-27* illustrates the two page tables, where each page table is used in an individual security domain. Secure and non-secure page tables address-translate the secure traffics and non-secure traffics respectively.

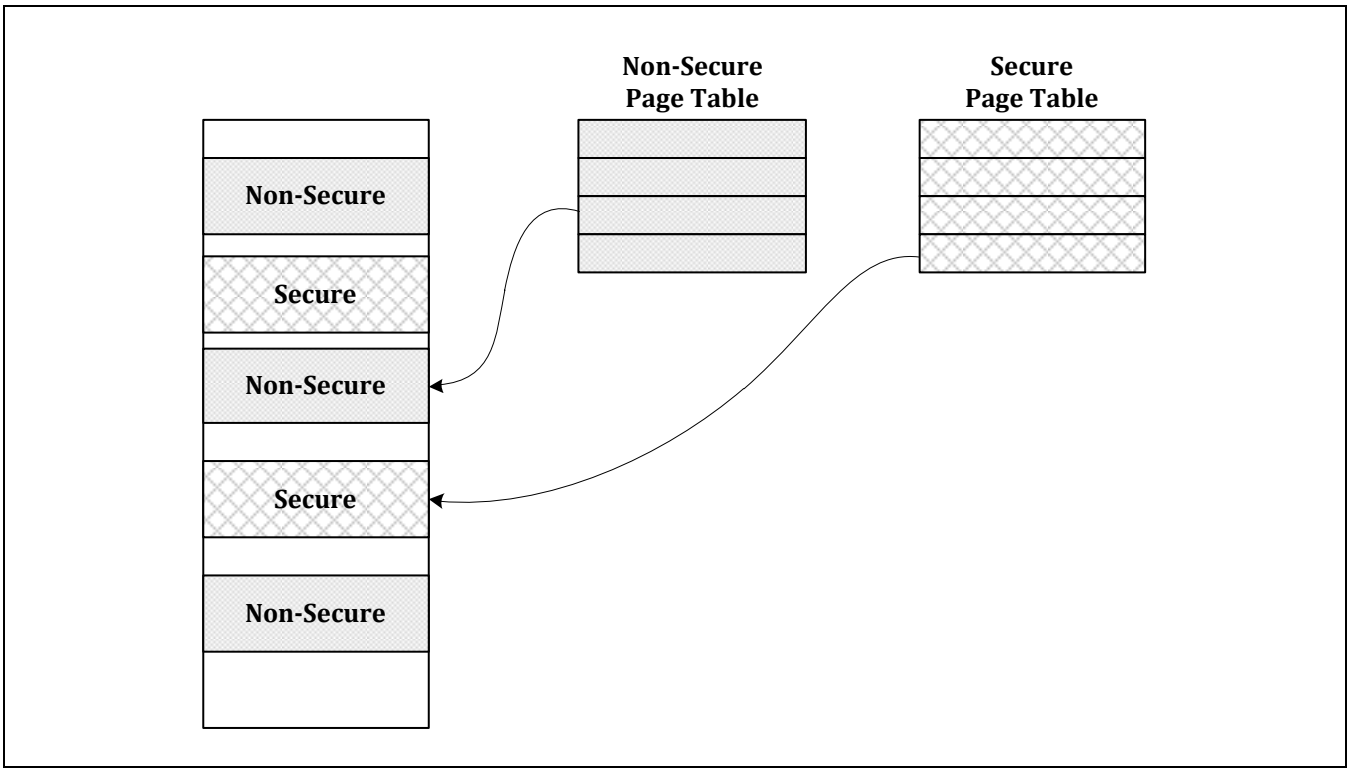

Figure 1-27 Secure and Non-Secure Page Tables

NOTE: Supporting secure page table design is configurable.

1.2.8.1 Secure Programming Interface

Secure page table must be set on SysMMU in a secure channel. That is, only secure programming interface must set FLPT_BASE_ADDR_S, the base address of FLPT that is used for secure transactions. Therefore, when the secure page table is supported, provide the additional secure programming interface. SysMMU must be independently controlled by the software in each security domain. *Figure 1-28* illustrates the programming interfaces that support both secure and non-secure page tables.

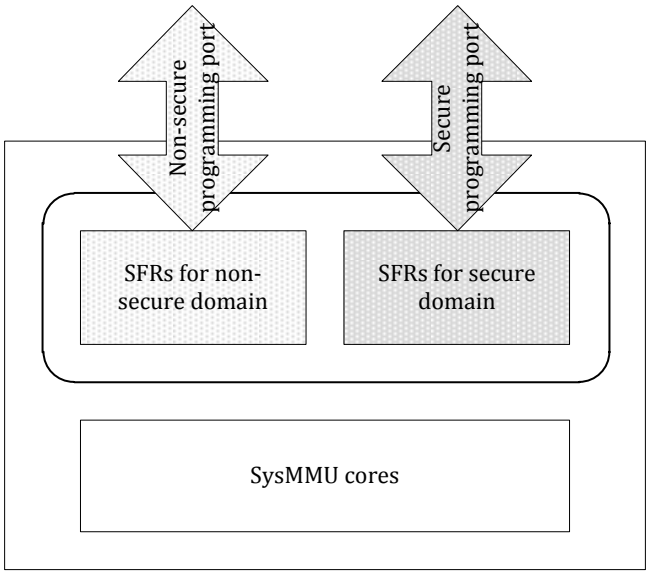

Figure 1-28 Programming Interfaces Supporting both Secure and Non-Secure Page Tables

1.2.8.2 Security Field in TLB and SBB Tags

Each line must provide the flag for security attribute of stored page descriptors, because the page descriptors in both security domains are stored in TLB and SBB. Therefore, TLB and SBB tags have an SC field to indicate whether the stored page descriptors belong to the secure domain or non-secure domain.

Ensure that look-up and invalidation in TLB and SBB consider the SC field as following entities:

- Even if TLB has page descriptor for the VPN of master request, if master security attribute is different for the corresponding SC field, TLB must return TLB miss.
- Invalidation commands from each programming interface must make TLB lines with the corresponding security attribute to be invalidated.

1.2.9 Non-secure Page Table

SysMMU supports non-secure page table for each non-secure virtual machine domain, respectively. Each non-secure transaction is address-translated by using own non-secure page table indexed by virtual machine domain identifier.

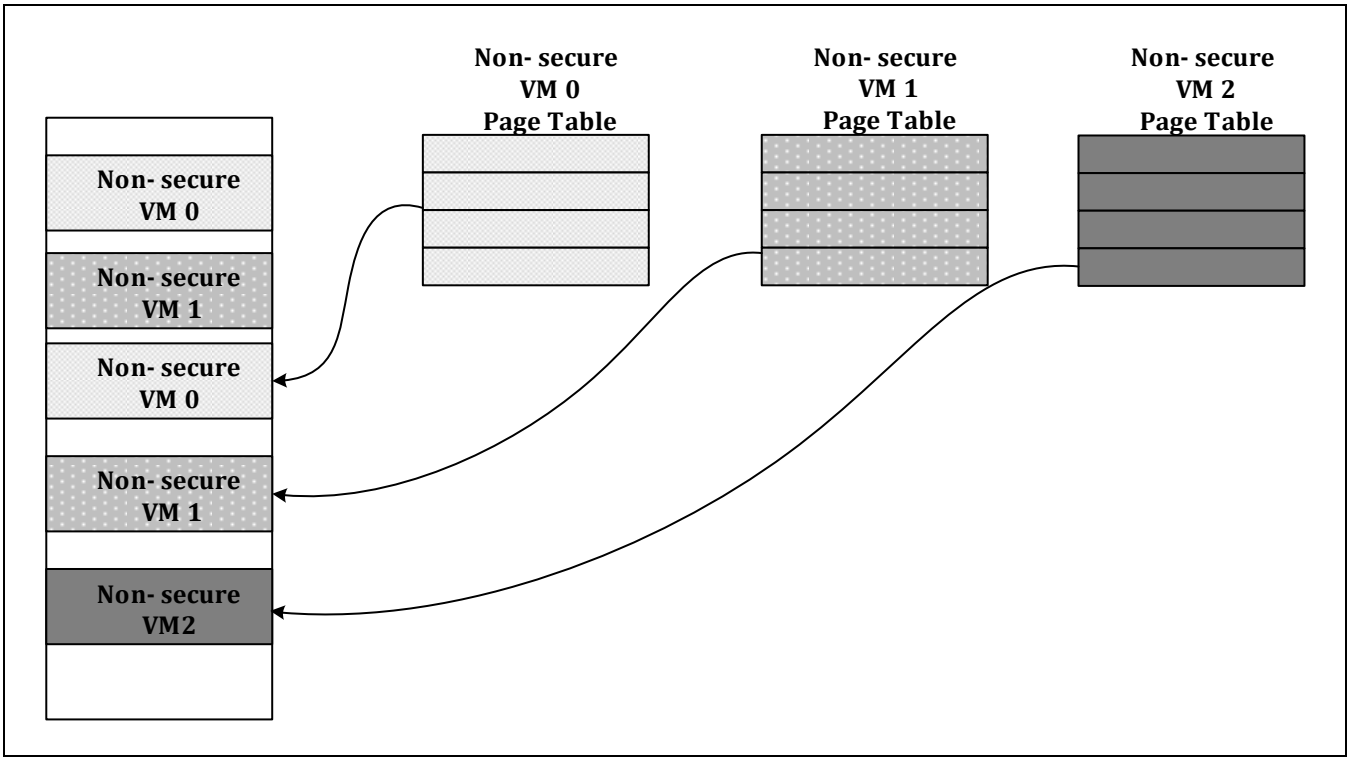

Figure 1-29 Non-secure Page Tables

1.2.10 Multiple Page Tables

SysMMU provides multiple page tables, which enables a certain traffic stream to use a different page table with another traffic stream. It shares a single SysMMU resource with different masters, where each master uses different page tables.

You can also use multiple page tables to provide the firewall function that controls the access to specific memory regions for some traffic streams. *Figure 1-30* illustrates an example to provide a firewall function by multiple page tables.

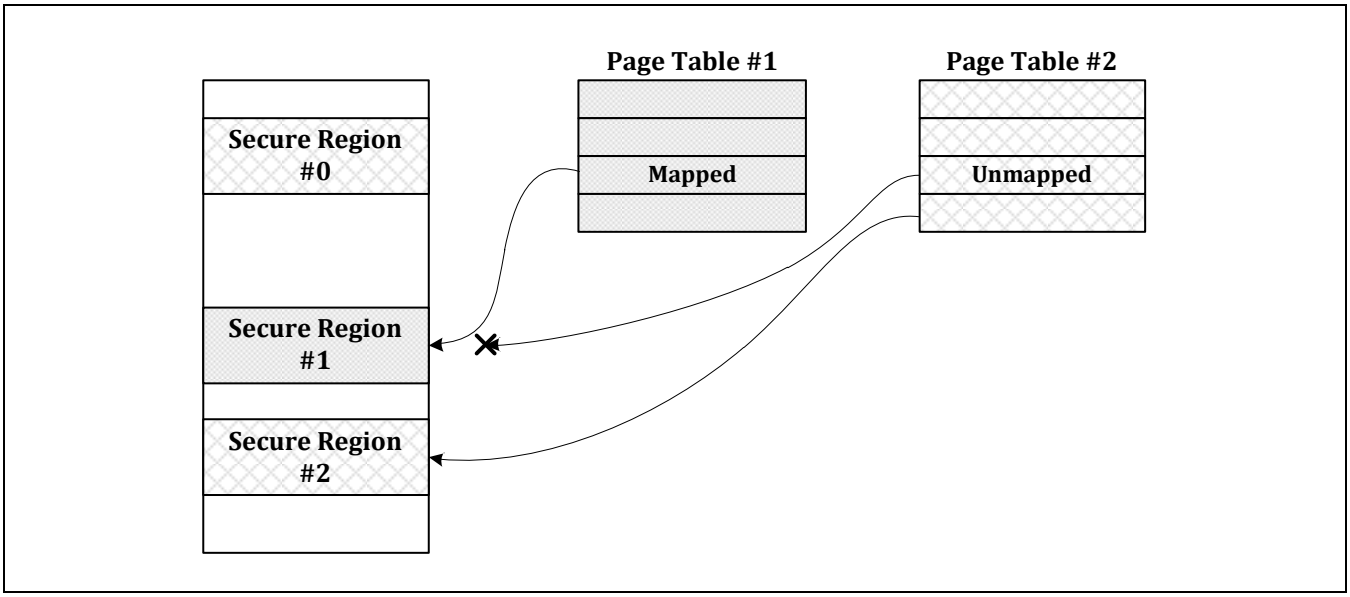

Figure 1-30 An Example of Firewall Function using Multiple Page Tables

In *Figure 1-30*, secure region 1 is only mapped onto page table 1. That is, only traffic streams using page table 1 accesses the secure region 1. Traffic streams that use page table 2 does not access secure region 1, the page fault happens when that traffics accesses the secure region 1.

NOTE: Supporting multiple page tables and the number of page tables design are configurable.

1.2.10.1 Configuration for Multiple Page Tables

For a SysMMU instance supporting multiple page tables, programming interface must configure the following information:

- Base address of page table
- Traffic attributes to identify the traffic streams that use the page table such as read/write and transaction ID mask/value
- Shareable attribute of page table: If page table is in shareable region, user signal of ARSHAREABLE must be generated as 0x1 for page table walk request to indicate that page table is located on shareable region
- Cache attribute of page table walk request
- Shareability generation (For more information, refer to Section *1.2.12.1 Shareability Generation*)
- Cacheable overriding (For more information, refer to Section *1.2.12.3 AxCACHE Overriding*)

The default page table is used for traffics that are not identified by the traffic attributes. Therefore, traffic attribute configuration is not required for the default page table.

When there is no traffic for the page table, ensure that the page table configuration is done.

1.2.10.2 Page Table ID Field in TLB and SBB Tags

Each line must provide the Page Table ID (PTID) field for page descriptors, because page descriptors for different page tables must be distinguishable in TLB and SBB. PTID field indicates the page table that the stored page descriptors belong to.

Look-up and invalidation in TLB and SBB should consider the PTID field as any of the following:

- Even if TLB has page descriptor for VPN of master request, if PTID of page table that master uses is different for the corresponding PTID field, TLB must return TLB miss.
- Invalidation commands from each programming interface must invalidate the TLB lines with the PTID specified by that programming interface.

1.2.11 SysMMU Control

1.2.11.1 Enabling and Disabling SysMMU

To provide virtual memory space with masters, enable SysMMU. When SysMMU is enabled, it performs address translation. If SysMMU is disabled, all incoming requests are forwarded to the memory-side without any address translation, so that the master must issue a memory request with the physical address. Following are the behaviors of SysMMU:

- When it is enabled, it performs address translation. Ensure that master requests to include the virtual address.
- When it is disabled, it forwards all the incoming requests to the memory-side without address translation. Master requests must have the physical address.

When SysMMU is disabled, SysMMU do not translate but it performs the zero-padding to address and ID filed. Address of master interface is forced by address of slave interface with zero-pad upper 4-bit of address field. ID of master interface is forced by ID of slave interface with zero-pad lower 1-bit of ID. Following are the description of address and ID extension:

- Address: Bit-width of address field in slave and master interfaces are 32-bit and 36-bit, respectively, hence upper 4-bit must be zero-padded.
- Transaction ID: Bit-width of ID field in the master interface is one-bit wider than the slave interface, hence ID value must be shifted by one at the master interface.

Ensure that although SysMMU is disabled, system clock is supplied.

When security-awareness is enabled, software in both security domains independently enables and disables the SysMMU. Therefore, minimize the enabling and disabling control in each security domain that affects SysMMU behavior in other security domain. Therefore, apply the following rules described in *Table 1-13* in security-aware SysMMU.

Table 1-13 Behaviors of Enabled and Disabled Controls in Each Security Domain

–	Disabled in Non-Secure Domain	Enabled in Non-Secure Domain
Disabled in secure domain	Non-secure traffics are not translated Secure traffics are not translated	Address translation is performed for non-secure traffics Secure traffics are not translated
Enabled in secure domain	Non-secure traffics are not translated Address translation is performed for non-secure traffics	Address translation is performed for non-secure traffics Address translation is performed for secure traffics

If the states for enabling and disabling are different in each security domain, transactions with same ID must have the same security attribute. For example, if two transactions have the same ID value, it is not allowed that one transaction is secure and the other transaction is non-secure.

SFR programming controls the enabling and disabling of SysMMU. Following are the default values:

- Non-secure domain: SysMMU disabled
- Secure domain: SysMMU enabled to enable the firewall function by default

1.2.11.2 Blocking and Unblocking

SysMMU blocks the incoming requests when a fault occurs or based on the intention of the user. If the incoming traffic is blocked due to fault and when the fault is resolved, SysMMU automatically unblock the traffics.

Software directly gives a command to block or unblock the incoming traffics. If SysMMU uses a software command to block the incoming traffics, it must be blocked until the software provides the unblock command.

When the security-awareness is enabled, software in both security domains blocks and unblocks the incoming traffics with their security attribute. For example, a non-secure domain provides the blocking command that blocks the non-secure traffics but does not affect the secure traffics.

After the traffic is blocked, the following traffics are not forwarded because of head-of-line blocking. Therefore, the following conditions are recommended.

- In a non-secure domain, blocking by software is not recommended.
- In a secure domain, blocking is required for firewall. But, it is recommended that the software unblocks the traffic before enabling the master to issue traffics that prevent head-of-line blocking.

If the software issues a block command through an SFR programming, some delay is required to enter the actual block state. Therefore, software must check if SysMMU reads the status to enter the block mode.

1.2.12 Run-Time Configuration

1.2.12.1 Shareability Generation

To indicate the shareability of accessible memory region, ensure to generate the AxSHAREABLE signal as a user signal at the master interface. S bit in the page descriptor denotes the shareability. Run-time programming also overrides AxSHARABLE, however overriding option is disabled by default.

Following conditions summarize the generation method for AxSHAREABLE:

- Overriding disabled (default): S bit in the page descriptor sets AxSHAREABLE
- Overriding enabled: Programmed value sets AxSHAREABLE

1.2.12.2 QoS Value Overriding

SysMMU supports a programmable feature for QoS value control of PTW request. QoS value is either inherited from the corresponding request that causes PTW or the configured value overrides it. By default, overriding option is disabled. Following conditions determine the QoS value of PTW request:

- Overriding disabled (default): QoS value of PTW request is inherited from the QoS value of request that creates it.
- Overriding enabled: Programmed value sets the QoS value of PTW request.

1.2.12.3 AxCACHE Overriding

SysMMU supports the programmable feature to override AxCACHE field based on the cache (C) field in page descriptor to utilize the system-level cache. *Table 1-14* describes the overriding rule for AxCACHE. For example, if a read request is referred to the page descriptor with C field of 3'b100, ARCACHE field is overridden as 4'b1011.

Table 1-14 AxCACHE Override Rules

C[2]	C[1]	C[0]	Description	ARCACHE	AWCACHE
0	0	0	Non-cacheable non-bufferable	4'b0010	4'b0010
0	1	1	Write-through no-allocate	4'b1010	4'b0110
1	0	0	Write-back no-allocate	4'b1011	4'b0111
1	1	0	Write-back read-allocate	4'b1111	4'b0111
1	0	1	Write-back write-allocate	4'b1011	4'b1111
1	1	1	Write-back read/write allocate	4'b1111	4'b1111
Others			Not override	Use AxCACHE of original request	

AxCACHE overriding is the run-time programming option. Default value is overriding-disabled, that is, original AxCACHE value is not modified in SysMMU. To override AxCACHE, configure SysMMU to enable AxCACHE overriding.

1.2.13 Low-Power Features

1.2.13.1 Architectural Clock Gating

SysMMU supports architectural clock gating to achieve high clock gating ratio. When the internal modules are in idle state, the clock supply to these modules can be turned off. The method to build up the tree of architectural clock gating is implementation-defined. However, apply the following rules to build a tree of architectural clock gating:

- You can use an external signal to turn off the architectural clock gating.
- Ensure that the architectural clock gating is turned off in test mode.
- When an external input is received, provide the clock with the module even if the corresponding module is not expected to receive it.

1.2.13.2 Q-Channel Interface

SysMMU supports Q-channel slave interface to control the clock supply from an external Clock Management Unit (CMU). Q-channel slave interface must interface with an external clock controller under the state machine as illustrated in *Figure 1-31*. For more information, refer to Low-Power Interface Specification - ARM Q-channel and P-channel interfaces.

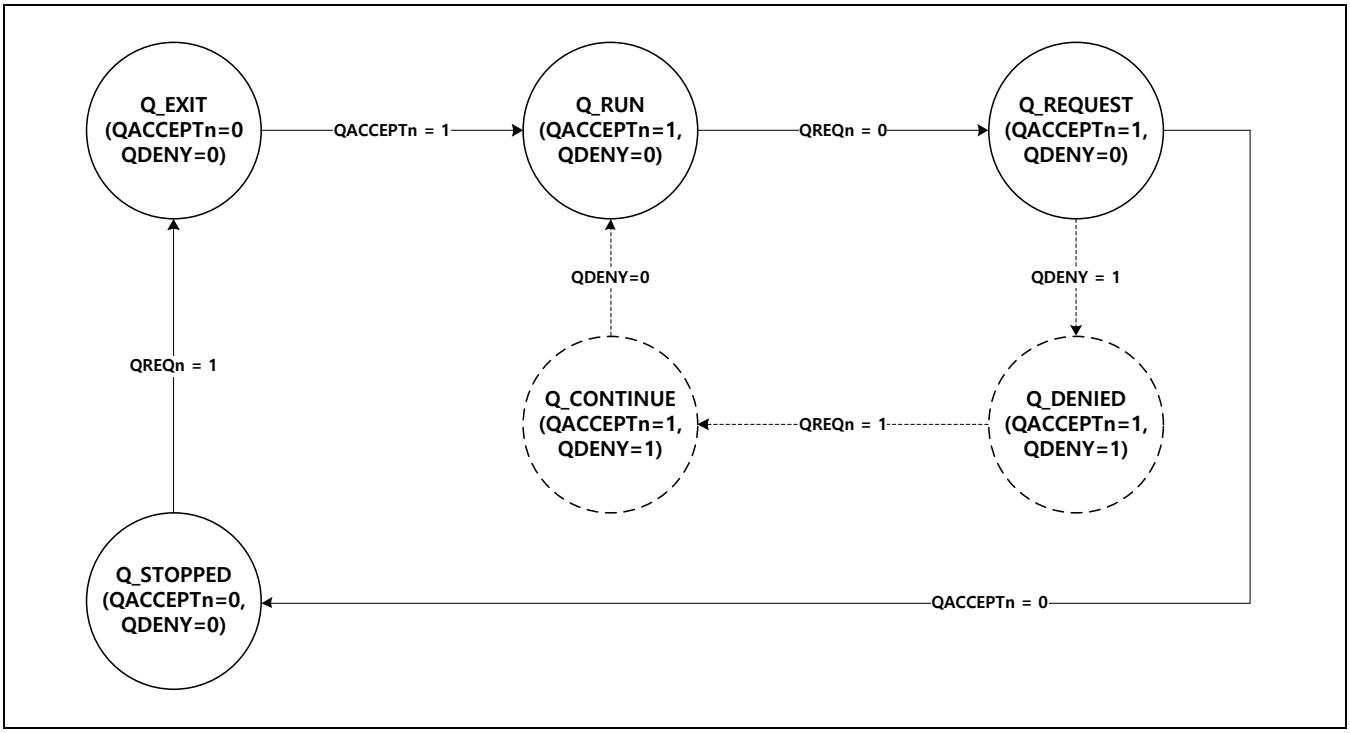

Figure 1-31 Q-Channel State

NOTE: Denial feature is not supported.

Following are the conditions to assert/de-assert the various Q-channel states:

- Assert QACTIVE: QACTIVE is a hint signal to request the clock-supply to an external clock controller. Asserting condition for QACTIVE must be of combinational logic because QACTIVE must be asserted even when a clock is not supplied to SysMMU.
- De-assert QACCEPTn: SysMMU must determine the condition to de-assert QACCEPTn to indicate permission to turn off the clock supply based on the request to turn off the clock supply from an external clock controller. If QREQn is de-asserted, QACCEPTn must be de-asserted in bounded time to prevent the system from hanging up.
- Assert QACCEPTn: SysMMU must determine the condition to assert QACCEPTn to perform initialization operation based on the request to turn on the clock supply from an external clock controller. SysMMU does not perform the initialization operation, hence QACCEPTn is asserted after 1 cycle. However, if implementation requires the initialization operation at Q_EXIT state, SysMMU designer must determine the condition to assert QACCEPTn.

1.2.14 Functional Safety

SysMMU has SRAM which contains address translation information. If there has defect on address translation information, it results to make wrong physical address and corrupts critical data section. Therefore, SRAM need to be protected and has error correction code (ECC).

1.3 Register Description

1.3.1 Base Address of Each Register Set

Table 1-15 describes all the SysMMU instances. Instances support security-awareness and virtual machine-awareness feature. Therefore, three register sets are existing in each SysMMU instance.

Table 1-15 Summary of Instances

Block	SysMMU Instances	Register Set Name (Security)	Offset (Base Address)
BLK_ACC	SYSMMU_ACC	REGISTERS0 (non-secure global)	0x1A9D_0000
		REGISTERS1 (secure)	0x1A9E_0000
		REGISTERS2 (non-secure virtual)	0x1A9F_0000
	SYSMMU_ISPPRE	REGISTERS0 (non-secure global)	0x1A8D_0000
		REGISTERS1 (secure)	0x1A8E_0000
		REGISTERS2 (non-secure virtual)	0x1A8F_0000
BLK_AUD	SYSMMU_AUD	REGISTERS0 (non-secure global)	0x19E0_0000
		REGISTERS1 (secure)	0x19E1_0000
		REGISTERS2 (non-secure virtual)	0x19E4_0000
BLK_DPUM	SYSMMU_DPU0	REGISTERS0 (non-secure global)	0x18C8_0000
		REGISTERS1 (secure)	0x18CC_0000
		REGISTERS2 (non-secure virtual)	0x18DC_0000
	SYSMMU_DPU1	REGISTERS0 (non-secure global)	0x18C9_0000
		REGISTERS1 (secure)	0x18CD_0000
		REGISTERS2 (non-secure virtual)	0x18DD_0000
	SYSMMU_DPU2	REGISTERS0 (non-secure global)	0x18CA_0000
		REGISTERS1 (secure)	0x18CE_0000
		REGISTERS2 (non-secure virtual)	0x18DE_0000
	SYSMMU_DPU3	REGISTERS0 (non-secure global)	0x18CB_0000
		REGISTERS1 (secure)	0x18CF_0000
		REGISTERS2 (non-secure virtual)	0x18DF_0000
BLK_DPUS0	SYSMMU_DPU0	REGISTERS0 (non-secure global)	0x1AA8_0000
		REGISTERS1 (secure)	0x1AAC_0000
		REGISTERS2 (non-secure virtual)	0x1ABC_0000
	SYSMMU_DPU1	REGISTERS0 (non-secure global)	0x1AA9_0000
		REGISTERS1 (secure)	0x1AAD_0000
		REGISTERS2 (non-secure virtual)	0x1ABD_0000
	SYSMMU_DPU2	REGISTERS0 (non-secure global)	0x1AAA_0000
		REGISTERS1 (secure)	0x1AAE_0000
		REGISTERS2 (non-secure virtual)	0x1ABE_0000

Block	SysMMU Instances	Register Set Name (Security)	Offset (Base Address)
	SYSMMU_DPU3	REGISTERS0 (non-secure global)	0x1AAB_0000
		REGISTERS1 (secure)	0x1AAF_0000
		REGISTERS2 (non-secure virtual)	0x1ABF_0000
BLK_DPUS1	SYSMMU_DPU0	REGISTERS0 (non-secure global)	0x1AC8_0000
		REGISTERS1 (secure)	0x1ACC_0000
		REGISTERS2 (non-secure virtual)	0x1ADC_0000
	SYSMMU_DPU1	REGISTERS0 (non-secure global)	0x1AC9_0000
		REGISTERS1 (secure)	0x1ACD_0000
		REGISTERS2 (non-secure virtual)	0x1ADD_0000
	SYSMMU_DPU2	REGISTERS0 (non-secure global)	0x1ACA_0000
		REGISTERS1 (secure)	0x1ACE_0000
		REGISTERS2 (non-secure virtual)	0x1ADE_0000
	SYSMMU_DPU3	REGISTERS0 (non-secure global)	0x1ACB_0000
		REGISTERS1 (secure)	0x1ACF_0000
		REGISTERS2 (non-secure virtual)	0x1ADF_0000
BLK_DPUS2	SYSMMU_DPU0	REGISTERS0 (non-secure global)	0x1AE8_0000
		REGISTERS1 (secure)	0x1AEC_0000
		REGISTERS2 (non-secure virtual)	0x1AFC_0000
	SYSMMU_DPU1	REGISTERS0 (non-secure global)	0x1AE9_0000
		REGISTERS1 (secure)	0x1AED_0000
		REGISTERS2 (non-secure virtual)	0x1AFD_0000
	SYSMMU_DPU2	REGISTERS0 (non-secure global)	0x1AEA_0000
		REGISTERS1 (secure)	0x1AEE_0000
		REGISTERS2 (non-secure virtual)	0x1AFE_0000
	SYSMMU_DPU3	REGISTERS0 (non-secure global)	0x1AEB_0000
		REGISTERS1 (secure)	0x1AEF_0000
		REGISTERS2 (non-secure virtual)	0x1AFF_0000
BLK_FSYS0	SYSMMU_PCIE_GEN3A_4L	REGISTERS0 (non-secure global)	0x1780_0000
		REGISTERS1 (secure)	0x1781_0000
		REGISTERS2 (non-secure virtual)	0x1782_0000
	SYSMMU_PCIE_GEN3B_4L	REGISTERS0 (non-secure global)	0x1783_0000
		REGISTERS1 (secure)	0x1784_0000
		REGISTERS2 (non-secure virtual)	0x1785_0000
	SYSMMU_PCIE_GEN3A_2L0	REGISTERS0 (non-secure global)	0x1786_0000
		REGISTERS1 (secure)	0x1787_0000
		REGISTERS2 (non-secure virtual)	0x1788_0000

Block	SysMMU Instances	Register Set Name (Security)	Offset (Base Address)
	SYSMMU_PCIE_GEN3B_2L0	REGISTERS0 (non-secure global)	0x1789_0000
		REGISTERS1 (secure)	0x178A_0000
		REGISTERS2 (non-secure virtual)	0x178B_0000
	SYSMMU_PCIE_GEN3A_2L1	REGISTERS0 (non-secure global)	0x178C_0000
		REGISTERS1 (secure)	0x178D_0000
		REGISTERS2 (non-secure virtual)	0x178E_0000
	SYSMMU_PCIE_GEN3B_2L1	REGISTERS0 (non-secure global)	0x178F_0000
		REGISTERS1 (secure)	0x1790_0000
		REGISTERS2 (non-secure virtual)	0x1791_0000
BLK_FSYS2	SYSMMU_ETHERNET0	REGISTERS0 (non-secure global)	0x17C8_0000
		REGISTERS1 (secure)	0x17C9_0000
		REGISTERS2 (non-secure virtual)	0x17CA_0000
	SYSMMU_ETHERNET1	REGISTERS0 (non-secure global)	0x17CB_0000
		REGISTERS1 (secure)	0x17CC_0000
		REGISTERS2 (non-secure virtual)	0x17CD_0000
BLK_G2D	SYSMMU_G2D0	REGISTERS0 (non-secure global)	0x19A6_0000
		REGISTERS1 (secure)	0x19A7_0000
		REGISTERS2 (non-secure virtual)	0x19A8_0000
	SYSMMU_G2D1	REGISTERS0 (non-secure global)	0x19A9_0000
		REGISTERS1 (secure)	0x19AA_0000
		REGISTERS2 (non-secure virtual)	0x19AB_0000
	SYSMMU_G2D2	REGISTERS0 (non-secure global)	0x19B9_0000
		REGISTERS1 (secure)	0x19BA_0000
		REGISTERS2 (non-secure virtual)	0x19BB_0000
BLK_ISPB0	SYSMMU_ISPB	REGISTERS0 (non-secure global)	0x1805_0000
		REGISTERS1 (secure)	0x1806_0000
		REGISTERS2 (non-secure virtual)	0x1807_0000
BLK_ISPB1	SYSMMU_ISPB	REGISTERS0 (non-secure global)	0x1825_0000
		REGISTERS1 (secure)	0x1826_0000
		REGISTERS2 (non-secure virtual)	0x1827_0000
BLK_ISPB2	SYSMMU_ISPB	REGISTERS0 (non-secure global)	0x1845_0000
		REGISTERS1 (secure)	0x1846_0000
		REGISTERS2 (non-secure virtual)	0x1847_0000
BLK_ISPB3	SYSMMU_ISPB	REGISTERS0 (non-secure global)	0x1865_0000
		REGISTERS1 (secure)	0x1866_0000
		REGISTERS2 (non-secure virtual)	0x1867_0000

Block	SysMMU Instances	Register Set Name (Security)	Offset (Base Address)
BLK_MFC	SYSMMU_MFC0	REGISTERS0 (non-secure global)	0x1988_0000
		REGISTERS1 (secure)	0x198A_0000
		REGISTERS2 (non-secure virtual)	0X1989_9999
	SYSMMU_MFC1	REGISTERS0 (non-secure global)	0x198B_0000
		REGISTERS1 (secure)	0x198D_0000
		REGISTERS2 (non-secure virtual)	0x198C_0000
BLK_NPU00	SYSMMU_NPU0	REGISTERS0 (non-secure global)	0x1D93_0000
		REGISTERS1 (secure)	0x1D94_0000
		REGISTERS2 (non-secure virtual)	0x1D98_0000
BLK_NPU01	SYSMMU_NPU0	REGISTERS0 (non-secure global)	0x1DD3_0000
		REGISTERS1 (secure)	0x1DD4_0000
		REGISTERS2 (non-secure virtual)	0x1DD8_0000
BLK_TAA0	SYSMMU_TAA	REGISTERS0 (non-secure global)	0x1A08_0000
		REGISTERS1 (secure)	0x1A09_0000
		REGISTERS2 (non-secure virtual)	0x1A0A_0000
BLK_TAA1	SYSMMU_TAA	REGISTERS0 (non-secure global)	0x1A28_0000
		REGISTERS1 (secure)	0x1A29_0000
		REGISTERS2 (non-secure virtual)	0x1A2A_0000
BLK_TAA2	SYSMMU_TAA	REGISTERS0 (non-secure global)	0x1A48_0000
		REGISTERS1 (secure)	0x1A49_0000
		REGISTERS2 (non-secure virtual)	0x1A4A_0000
BLK_TAA3	SYSMMU_TAA	REGISTERS0 (non-secure global)	0x1A68_0000
		REGISTERS1 (secure)	0x1A69_0000
		REGISTERS2 (non-secure virtual)	0x1A6A_0000

1.3.2 Register Map Summary

Following table describes all SFRs in SysMMU. Offset values of some SFRs are computational. They are described as formula with index n. Reset value of some SFRs differs based on the RTL configurations such as security domain TLB capacity, and are described as Implement.

1.3.2.1 Secure Context Register Summary

- Base Address: For more information on the base address of each block, refer *Table 1-15*.

Register	Offset	Description	Reset Value
MMU_CTRL	0x0	Control Register	Implement
MMU_CFG	0x4	Configuration Register	0x0002_0000
MMU_STATUS	0x8	Status Register	Implement
MMU_DEFAULT_FLPT_BASE	0xC	FLPT Base Address Register	0x0000_0000
MMU_ALL_INVALIDATION	0x10	Invalidation All Register	0x0000_0000
MMU_VPN_INVALIDATION	0x14	Invalidation VPN Register	0x0000_0000
MMU_RANGE_INVALIDATION	0x18	Invalidation Range Register	0x0000_0000
MMU_RANGE_INVALIDATION_START_VPN	0x20	Invalidation Range Start VPN Register	0x0000_0000
MMU_RANGE_INVALIDATION_END_VPN	0x24	Invalidation Range End VPN Register	0x0000_0000
MMU_VERSION	0x34	Version Register	Implement
MMU_INTERRUPT_STATUS	0x60	Interrupt Status Register	0x0000_0000
MMU_INTERRUPT_CLEAR	0x64	Interrupt Clear Register	0x0000_0000
MMU_FAULT_VA	0x70	Fault Virtual Address Register	0x0000_0000
MMU_FAULT_TRANS_INFO	0x78	Fault Transaction Information Register(1)	0x0000_0000
MMU_FAULT_TRANS_INFO2	0x7C	Fault Transaction Information Register(2)	0x0000_0000
MMU_CAPA_0	0x870	Capacity Register (0)	Implement
MMU_CAPA_1	0x874	Capacity Register (1)	Implement
MMU_PT_FLPT_BASE_n	0x1000 + n × 0x20	Page Table FLPT Base Address Register	0x0000_0000
MMU_PT_CFG_n	0x1000 + n × 0x20 + 0x4	Page Table Configuration Register	0x0000_0000
MMU_PT_MATCH_CFG_n	0x1000 + n × 0x20 + 0x8	Page Table Matching Configuration Register	0x0000_0000
MMU_PT_MATCH_ID_CFG_n	0x1000 + n × 0x20 + 0xC	Page Table ID Matching Configuration Register	0x0000_0000
MMU_TLB_READ	0x8000	TLB Read Request Register	0x0000_0000
MMU_TLB_VPN	0x8004	TLB Read Response Register (VPN)	0x0000_0000
MMU_TLB_PPN	0x8008	TLB Read Response Register (PPN)	0x0000_0000
MMU_TLB_ATTRIBUTE	0x800C	TLB Read Response Register (Attribute)	0x0000_0000

Register	Offset	Description	Reset Value
MMU_SBB_READ	0x8020	SBB Read Request Register	0x0000_0000
MMU_SBB_VPN	0x8024	SBB Read Response Register (VPN)	0x0000_0000
MMU_SBB_LINK	0x8028	SBB Read Response Register (LINK)	0x0000_0000
MMU_SBB_ATTRIBUTE	0x802C	SBB Read Response Register (Attribute)	0x0000_0000
MMU_EMULATION_PRELOAD	0x8040	Preloading emulation information Register	0x0000_0000
MMU_EMULATION_SHOT	0x8044	Emulation shot Register	0x0000_0000

1.3.2.2 Non-secure Global Register Summary

- Base Address: For more information on the base address of each block, refer *Table 1-15*.

Register	Offset	Description	Reset Value
MMU_CTRL	0x0	Control Register	Implement
MMU_CFG	0x4	Configuration Register	0x0002_0000
MMU_STATUS	0x8	Status Register	Implement
MMU_VERSION	0x34	Version Register	Implement
MMU_INTERRUPT_STATUS	0x60	Interrupt Status Register	0x0000_0000
MMU_INTERRUPT_CLEAR	0x64	Interrupt Clear Register	0x0000_0000
MMU_FAULT_VA	0x70	Fault Virtual Address Register	0x0000_0000
MMU_FAULT_TRANS_INFO	0x78	Fault Transaction Information Register(1)	0x0000_0000
MMU_FAULT_TRANS_INFO2	0x7C	Fault Transaction Information Register(2)	0x0000_0000
MMU_CAPA_0	0x870	Capacity Register (0)	Implement
MMU_CAPA_1	0x874	Capacity Register (1)	Implement
MMU_ECC_EN	0x880	ECC enable Register	0x0000_0000
MMU_TLB_INFO_n	0x2000 + n × 0x20	TLB Information Register	Implement
MMU_TLB_CFG_n	0x2000 + n × 0x20 + 0x4	TLB Configuration Register	0x0000_0000
MMU_TLB_MATCH_CFG_n	0x2000 + n × 0x20 + 0x8	TLB Matching Configuration Register	0x0000_0000
MMU_TLB_MATCH_SVA_n	0x2000 + n × 0x20 + 0xC	TLB Matching Start Virtual Address Register	0x0000_0000
MMU_TLB_MATCH_EVA_n	0x2000 + n × 0x20 + 0x10	TLB Matching End Virtual Address Register	0xFFFF_FFFF
MMU_TLB_MATCH_ID_n	0x2000 + n × 0x20 + 0x14	TLB Matching ID Register	0x0000_0000
MMU_TLB_READ	0x8000	TLB Read Request Register	0x0000_0000
MMU_TLB_VPN	0x8004	TLB Read Response Register (VPN)	0x0000_0000
MMU_TLB_PPN	0x8008	TLB Read Response Register (PPN)	0x0000_0000
MMU_TLB_ATTRIBUTE	0x800C	TLB Read Response Register (Attribute)	0x0000_0000

Register	Offset	Description	Reset Value
MMU_SBB_READ	0x8020	SBB Read Request Register	0x0000_0000
MMU_SBB_VPN	0x8024	SBB Read Response Register (VPN)	0x0000_0000
MMU_SBB_LINK	0x8028	SBB Read Response Register (LINK)	0x0000_0000
MMU_SBB_ATTRIBUTE	0x802C	SBB Read Response Register (Attribute)	0x0000_0000
MMU_EMULATION_PRELOAD	0x8040	Preloading emulation information Register	0x0000_0000
MMU_EMULATION_SHOT	0x8044	Emulation shot Register	0x0000_0000
PM_CFG	0x9000	Performance measurement configuration Register	0x0000_0000
PM_INIT	0x9004	Performance measurement counter initialization Register	0x0000_0000
PM_EVENTn	0x9010 + n × 0x4	Performance measurement event Register	0x0000_0000
PM_CNTn	0x9020 + n × 0x4	Performance measurement counter Register	0x0000_0000

1.3.2.3 Non-secure Virtual Context Register Summary

- Base Address: For more information on the base address of each block, refer *Table 1-15*.

Register	Offset	Description	Reset Value
MMU_CTRL	0x0	Control Register	Implement
MMU_CFG	0x4	Configuration Register	0x0002_0000
MMU_DEFAULT_FLPT_BASE	0xC	FLPT Base Address Register	0x0000_0000
MMU_ALL_INVALIDATION	0x10	Invalidation All Register	0x0000_0000
MMU_VPN_INVALIDATION	0x14	Invalidation VPN Register	0x0000_0000
MMU_RANGE_INVALIDATION	0x18	Invalidation Range Register	0x0000_0000
MMU_RANGE_INVALIDATION_START_VPN	0x20	Invalidation Range Start VPN Register	0x0000_0000
MMU_RANGE_INVALIDATION_END_VPN	0x24	Invalidation Range End VPN Register	0x0000_0000
MMU_VERSION	0x34	Version Register	Implement
MMU_CAPA_0	0x870	Capacity Register (0)	Implement
MMU_CAPA_1	0x874	Capacity Register (1)	Implement

1.3.3 Secure Context Register Set

1.3.3.1 MMU_CTRL

- Address = Base Address + 0x0000, Reset Value = 0x0000_0007

  MMU_CTRL is a control register of the basic operation. MMU_CTRL includes the operation mode and interrupt enable. Reset value varies based on the SFR set security attribute.

Name	Bit	Type	Description	Reset value
RSVD	[31:6]	R	Reserved	0x0
ERR_RESP_VALUE	[5]	RW	Error response value 0x0 = OKAY 0x1 = SLVERR	0x0
FAULT_MODE_OVERRIDE	[4]	RW	Fault mode override. If it is set 0x1, fault mode of all non-secure context is overridden by termination mode 0x0: not override 0x1: override by termination mode	0x0
FAULT_MODE	[3]	RW	Fault mode 0x0: termination mode 0x1: stall mode	0x0
INT_ENABLE	[2]	RW	Enables the interrupt generation 0x0 = Disable 0x1 = Enable	0x1
MMU_BLOCK	[1]	RW	Blocks the MMU requests 0x0 = Un-block requests 0x1 = Block requests	0x1
MMU_ENABLE	[0]	RW	Enables the SysMMU 0x0 = Disable SysMMU 0x1 = Enable SysMMU	0x1

1.3.3.2 MMU_CFG

- Address = Base Address + 0x0004, Reset Value = 0x0002_0000

  MMU_CFG is a configuration register to configure translation and page table walk operation.

Name	Bit	Type	Description	Reset value
CACHEABLE_OVERRIDE	[31]	RW	Indicates whether the cacheable field in page descriptor overrides the AxCACHE of normal request or not 0x0 = AxCACHE of normal request does not override 0x1 = AxCACHE of normal request is overridden by the cacheable field in page descriptor	0x0
RSVD	[30]	R	Reserved	0x0
SHAREABILITY	[29]	RW	Indicates the translated results of request is shareable or not This value is overrode to AxSHAREABLE_M0 when MMU_CFG.SHAREABILITY_OVERRID = 0x1. 0x0 = Non-shareable 0x1 = Shareable	0x0
SHAREABILITY_OVERRIDE	[28]	RW	Indicates whether the MMU_CFG.SHAREABILITY overrides AxSHAREABLE_M0 or not 0x0 = AxSHAREABLE_M0 of master requests is set by shareability information of the corresponding entry in the page table 0x1 = AxSHAREABLE_M0 of master requests is forced to MMU_CFG. SHAREBILITY	0x0
RSVD	[27:20]	R	Reserved	0x0
PTW_ARCACHE	[19:16]	RW	ARCACHE value of PTW request	0x2
RSVD	[15:13]	R	Reserved	0x0
PT_SHAREABLE	[12]	RW	Indicates the value of the ARSHAREABLE_M0 of PTW request 0x0 = Page tables are located in a non-sharable region. ARSHARABLE_M0 is 0x0 for PTW request. 0x1 = Page tables are located in a sharable region. ARSHARABLE_M0 is 0x1 for PTW request.	0x0
QOS_OVERRIDE	[11]	RW	Indicates whether the MMU_CFG.QOS_VALUE overrides ARQOS_M0 or not 0x0 = QoS value of PTW requests is inherited from the corresponding requests issued from master 0x1 = QoS value of PTW requests is forced to MMU_CFG. QOS_VALUE	0x0
QOS_VALUE	[10:7]	RW	QoS value of PTW requests It is only effective when QOS_OVERRIDE is 0x1.	0x0
RSVD	[6:4]	R	Reserved	0x0
SECURITY_PROT_ENABLE	[3]	RW	Enables the security protection It is effective only when SysMMU is enabled 0x0 = Disable security protection	0x0

Name	Bit	Type	Description	Reset value
			0x1 = Enable security protection	
ACCESS_PROT_ENABLE	[2]	RW	Enables the access protection checking It is effective only when SysMMU is enabled 0x0 = Disable checking access protection 0x1 = Enable checking access protection	0x0
RSVD	[1:0]	R	Reserved	0x0

1.3.3.3 MMU_STATUS

- Address = Base Address + 0x0008, Reset Value = 0x0000_0001

  MMU_STATUS is current status register. Reset value varies based on the SFR set security attribute.

Name	Bit	Type	Description	Reset value
RSVD	[31:5]	R	Reserved	0x0
ON_INVALIDATING	[4]	R	Indicates the status of invalidation 0x0 = There is no invalidation in SysMMU 0x1 = There is ongoing invalidation in SysMMU	0x0
AR_PEND	[3]	R	Indicates the read request on translation in SysMMU 0x0 = There is no read request on translation in SysMMU 0x1 = There is a read request on translation in SysMMU	0x0
AW_PEND	[2]	R	Indicates the write request on translation in SysMMU 0x0 = There is no write request on translation in SysMMU 0x1 = There is a write request on translation in SysMMU	0x0
PTW_ACTIVE	[1]	R	Indicates the active PTW request 0x0 = There is no active PTW request 0x1 = There is an active PTW request	0x0
BLOCKED	[0]	R	Indicates whether the AXI slave interface of SysMMU is blocked or not 0x0 = AXI slave interface of SysMMU is un-blocked 0x1 = AXI slave interface of SysMMU is blocked	0x1

1.3.3.4 MMU_DEFAULT_FLPT_BASE

- Address = Base Address + 0x000C, Reset Value = 0x0000_0000.

  MMU_DEFAULT_FLPT_BASE is first-level page table base address register that stores the base address of first-level page table.

Name	Bit	Type	Description	Reset value
RSVD	[31:24]	R	Reserved	0x0
FLPT_BASE_PPN	[23:0]	RW	PPN of first-level page table for default page table (default page table means page table 0)	0x0

1.3.3.5 MMU_ALL_INVALIDATION

- Address = Base Address + 0x0010, Reset Value = 0x0000_0000

  MMU_ALL_INVALIDATION is a register that invalidates all page descriptors in SysMMU.
  When MMU_ALL_INVALIDATION.INVALIDATE = 0x1, the only page table of ID that matches with MMU_ALL_INVALIDATION.PTID is invalidated. If MMU_ALL_INVALIDATION.INVALIDATE = 0x3, all page table regardless of PTID are invalidated.

Name	Bit	Type	Description	Reset value
RSVD	[31:8]	R	Reserved	0x0
PTID	[7:4]	W	It indicates the page table ID to perform all-invalidation. It is effective only when performing all-invalidation for a specific page table.	0x0
RSVD	[3:2]	R	Reserved	0x0
INVALIDATE	[1:0]	W	Indicates invalidation mode 0x0 = No operation 0x1 = All-invalidation for a specific page table 0x2 = No operation 0x3 = All-invalidation for all page table	0x0

1.3.3.6 MMU_VPN_INVALIDATION

- Address = Base Address + 0x0014, Reset Value = 0x0000_0000

  MMU_VPN_INVALIDATION is a register that invalidates the specific page descriptor in the page table of PTID that matches with MMU_VPN_INVALIDATION.PTID

Name	Bit	Type	Description	Reset value
VPN	[31:12]	W	Virtual page number for 4 KB page granularity (VPN[19:0])	0x0
VPN_UPPER (NOTE)	[11:8]	W	Upper 4bits of VPN	0x0
PTID	[7:4]	W	Page table ID	0x0
RSVD	[3:1]	R	Reserved	0x0
INVALIDATE	[0]	W	Indicates invalidation mode 0x0 = No operation 0x1 = VPN-invalidation for a specific page table	0x0

NOTE: It is only accessible when 36bits-VA translation is enabled, otherwise it is reserved.

1.3.3.7 MMU_RANGE_INVALIDATION

- Address = Base Address + 0x0018, Reset Value = 0x0000_0000

  MMU_RANGE_INVALIDATION is a register to invalidate all the page descriptor within ranges. Before setting MMU_RANGE_INVALIDATION, ensure that MMU_RANGE_INVALIDATION_START_VPN and MMU_RANGE_INVALIDATION_END_VPN are set.

Name	Bit	Type	Description	Reset value
RSVD	[31:8]	R	Reserved	0x0
PTID	[7:4]	W	Page table ID	0x0
RSVD	[3:1]	R	Reserved	0x0
INVALIDATE	[0]	W	Indicates invalidation mode 0x0 = No operation 0x1 = Range-invalidation for a specific page table	0x0

1.3.3.8 MMU_RANGE_INVALIDATION_START_VPN

- Address = Base Address + 0x0020, Reset Value = 0x0000_0000

  MMU_RANGE_INVALIDATION_START_VPN register represents the start VPN of range invalidation.

Name	Bit	Type	Description	Reset value
VPN	[31:12]	W	Start virtual page number for 4 KB page granularity of range-invalidation (VPN[19:0])	0x0
*VPN_UPPER	[11:8]	W	Upper 4bits of VPN	0x0
RSVD	[7:0]	R	Reserved	0x0

* It is only accessible when 36bits-VA translation is enabled, otherwise it is reserved.

1.3.3.9 MMU_RANGE_INVALIDATION_END_VPN

- Address = Base Address + 0x0024, Reset Value = 0x0000_0000

  MMU_RANGE_INVALIDATION_END_VPN register indicates the end VPN of range invalidation.

Name	Bit	Type	Description	Reset value
VPN	[31:12]	W	End virtual page number for 4 KB page granularity of range-invalidation (VPN[19:0])	0x0
*VPU_UPPPER	[11:8]	R	Upper 4bits of VPN	0x0
RSVD	[7:0]	R	Reserved	0x0

* It is only accessible when 36bits-VA translation is enabled, otherwise it is reserved.

1.3.3.10 MMU_VERSION

- Address = Base Address + 0x0034, Reset Value = 0x7800_0000

  MMU_VERSION contains the version information. The reset value varies with the minor architecture versions and RTL version.

Name	Bit	Type	Description	Reset value
MAJOR_ARCH_VER	[31:28]	R	Major architecture version	0x7
MINOR_ARCH_VER	[27:21]	R	Minor architecture version	0x4
REV_ARCH_VER	[20:17]	R	Revision architecture version	0x0
RSVD	[16:8]	R	Reserved	0x0
RTL_VER	[7:0]	R	RTL version	0x0

1.3.3.11 MMU_INTERRUPT_STATUS

- Address = Base Address + 0x0060, Reset Value = 0x0000_0000

  MMU_INTERRUPT_STATUS stores the current interrupt information.

Name	Bit	Type	Description	Reset value
RSVD	[31:5]	R	Reserved	0x0
SECURITY_FAULT	[4]	R	Security fault 0x0 = This fault is not detected 0x1 = This fault is detected	0x0
ACCESS_FAULT	[3]	R	Access permission fault 0x0 = This fault is not detected 0x1 = This fault is detected	0x0
RSVD	[2]	R	Reserved	0x0
PAGE_FAULT	[1]	R	Page fault 0x0 = This fault is not detected 0x1 = This fault is detected	0x0
PTW_ACCESS_FAULT	[0]	R	PTW access fault 0x0 = This fault is not detected 0x1 = This fault is detected	0x0

1.3.3.12 MMU_INTERRUPT_CLEAR

- Address = Base Address + 0x0064, Reset Value = 0x0000_0000

  MMU_INTERRUPT_CLEAR is a register to clear the interrupt. To clear the current interrupt, set the corresponding register field. If the interrupt is not generated, MMU_INTERRUPT_CLEAR has no effect.

Name	Bit	Type	Description	Reset value
RSVD	[31:5]	R	Reserved	0x0
SECURITY_FAULT	[4]	W	Security fault 0x0 = No operation 0x1 = Clear this fault status	0x0
ACCESS_FAULT	[3]	W	Access permission fault 0x0 = No operation 0x1 = Clear this fault status	0x0
RSVD	[2]	R	Reserved	0x0
PAGE_FAULT	[1]	W	Page fault 0x0 = No operation 0x1 = Clear this fault status	0x0
PTW_ACCESS_FAULT	[0]	W	PTW access fault 0x0 = No operation 0x1 = Clear this fault status	0x0

1.3.3.13 MMU_FAULT_VA

- Address = Base Address + 0x0070, Reset Value = 0x0000_0000
  MMU_FAULT_VA contains the virtual address that causes the fault.

Name	Bit	Type	Description	Reset value
VA	[31:0]	R	Virtual address of the request with fault (VA[31:0])	0x0

1.3.3.14 MMU_FAULT_TRANS_INFO

- Address = Base Address + 0x0078, Reset Value = 0x0000_0000
  MMU_FAULT_TRANS_INFO contains the transaction information that causes the fault.

Name	Bit	Type	Description	Reset value
*VA_UPPER	[31:28]	R	Upper 4bits of VA	0x0
PID	[27:24]	R	Port where the request come from	0x0
RSVD	[23:21]	R	Reserved	0x0
RW	[20]	R	It represents whether the request with fault is read request or write request 0 = Read request 1 = Write request	0x0
LEN	[19:16]	R	AxLEN of the request with fault	0x0
ID	[15:0]	R	AxID of the request with fault	0x0

* It is only accessible when 36bits-VA translation is enabled, otherwise it is reserved.

1.3.3.15 MMU_FAULT_TRANS_INFO2

- Address = Base Address + 0x007C, Reset Value = 0x0000_0000
  MMU_FAULT_TRANS_INFO contains the transaction information that causes the fault.

Name	Bit	Type	Description	Reset value
RSVD	[31:16]	R	Reserved	0x0
PTID	[15:0]	R	PTID of the request with fault	0x0

1.3.3.16 MMU_CAPA_0

- Address = Base Address + 0x0870, Reset Value is different from the configuration set.
  MMU_CAPA_0 register indicates the SysMMU capacity information.

Name	Bit	Type	Description	Reset value
MAX_PTW_MO	[31:26]	R	Maximum number of outstanding PTW	0x8 (NOTE)
MAX_AT_MO	[25:20]	R	Maximum number of outstanding address translation	0x8 (NOTE)
NUM_PAGE_TABLE	[19:16]	R	Number of page tables in SysMMU	0x1 (NOTE)
NUM_SBB_ENTRY	[15:12]	R	Indicates the number of SBB entries 0x0 = 1 entry 0x1 = 2 entries 0x2 = 4 entries 0x3 = 8 entries 0x4 = 16 entries 0x5 = 32 entries Others = Reserved	0x3 (NOTE)
CAPA_1_EXIST	[11]	R	Indicates existence of MMU_CAPA_1 0x0 = MMU_CAPA_1 SFR is not exist 0x1 = MMU_CAPA_1 SFR is exist	0x1 (NOTE)
RSVD	[10:0]	R	Reserved	0x0

NOTE: Each value of MMU_CAPA_0 field is different from the configuration set.

1.3.3.17 MMU_CAPA_1

- Address = Base Address + 0x0874, Reset Value is different from the configuration set.

  MMU_CAPA_1 register indicates the SysMMU capacity information.

Name	Bit	Type	Description	Reset value
TYPE	[31:28]	R	Indicates type of MMU_CAPA_1 0x0 = type0 (NUM_TLB, NUM_PORT) Others = Reserved	0x0
RSVD	[27:13]	R	Reserved	0x0
VCR_ENABLE	[14]	R	Virtual machine control register enable 0x0: disabled 0x1: enabled	0x0
STALL_MODE_SUPPORT	[13]	R	Indicates VA width 0x0: stall mode not supported 0x1: stall mode supported	0x1
VA_WIDTH	[12]	R	Indicates VA width 0x0: VA width is 32 bits 0x1: VA width is 36 bits	0x1(NOTE)
NUM_TLB	[11:4]	R	The number of TLBs in SysMMU	0x4 (NOTE)
NUM_PORT	[3:0]	R	The number of AXI ports of SysMMU	0x1 (NOTE)

NOTE: Each value of MMU_CAPA_1 field is different from the configuration set.

1.3.3.18 MMU_PT_FLPT_BASE_n

- Address = Base Address + 0x1000 + n × 0x20, n is from 0 to the number of page table.
  Reset Value = 0x0000_0000

  MMU_PT_FLPT_BASE_n is first-level page table base address register that stores the base address of first-level page table. Number of registers varies based on the RTL configuration option for the number of page table. Offset address varies based on the index of target page table to be configured.

Name	Bit	Type	Description	Reset value
RSVD	[31:24]	R	Reserved	0x0
FLPT_BASE_PPN	[23:0]	RW	PPN of first-level page table for PTID(n)	0x0

1.3.3.19 MMU_PT_CFG_n

- Address = Base Address + 0x1000 + n × 0x20 + 0x4, n is from 0 to the number of page table
  Reset Value = 0x0000_0000

  MMU_PT_CFG_n is a page table configuration register. Number of registers varies based on the RTL configuration option for the number of page table. Offset address varies based on the index of target page table to be configured.

Name	Bit	Type	Description	Reset value
CACHEABLE_OVERRIDE	[31]	RW	Indicates whether the cacheable field in page descriptor overrides the AxCACHE of normal request or not 0x0 = AxCACHE of normal request does not override 0x1 = AxCACHE of normal request is overridden by the cacheable field in page descriptor	0x0
RSVD	[30]	R	Reserved	0x0
SHAREABILITY	[29]	RW	Indicates the translated results of request is shareable or not This value is overrode to AxSHAREABLE_M0 when MMU_PT_CFG_n.SHAREABILITY_OVERRID = 0x1. 0x0 = Non-shareable 0x1 = Shareable	0x0
SHAREABILITY_OVERRIDE	[28]	RW	Indicates whether the MMU_PT_CFG_n.SHAREABILITY overrides AxSHAREABLE_M0 or not 0x0 = AxSHAREABLE_M0 of master requests is set by shareability information of the corresponding entry in the page table 0x1 = AxSHAREABLE_M0 of master requests is forced to SHAREBILITY	0x0
RSVD	[27:20]	R	Reserved	0x0
PTW_ARCACHE	[19:16]	RW	ARCACHE value of PTW request	4'b0010
RSVD	[15:13]	R	Reserved	0x0
PT_SHAREABLE	[12]	RW	Indicates the value of the ARSHAREABLE_M0 of PTW request 0x0 = Page tables are located in a non-sharable region. ARSHARABLE_M0 is 0x0 for PTW request. 0x1 = Page tables are located in a sharable region. ARSHARABLE_M0 is 0x1 for PTW request.	0x0
RSVD	[11:2]	R	Reserved	0x0
RW	[1:0]	RW	Indicates which direction (Read/Write) is permitted to the page table 0x0 = Both read/write requests are not matched to PTID(n) 0x1 = Only read requests are matched to PTID(n) 0x2 = Only write requests are matched to PTID(n) 0x3 = Both read/write requests are matched to PTID(n)	0x0

1.3.3.20 MMU_PT_MATCH_CFG_n

- Address = Base Address + 0x1000 + n × 0x20 + 0x8, n is from 1 to the number of page table
  Reset Value = 0x0000_0000

  MMU_PT_MATCH_CFG_n is page table matching configuration register.
  Number of registers varies based on the RTL configuration option for the number of page table. Offset address varies based on the index of target page table to be configured.

Name	Bit	Type	Description	Reset value
PORT_MASK	[31:16]	RW	It indicates whether each port can be matched to this page table. Each bit indicates each AXI port For example, the value of PORT_MASK[0] determines whether requests from port 0 can access page table n or not. In the same way, the value of PORT_MASK[1] determines whether requests from port 1 can access page table n or not.	0x0
RSVD	[15:2]	R	Reserved	0x0
RW	[1:0]	RW	Indicates which direction (Read/Write) is permitted to the page table 0x0 = Both read/write requests are not matched to PTID(n) 0x1 = Only read requests are matched to PTID(n) 0x2 = Only write requests are matched to PTID(n) 0x3 = Both read/write requests are matched to PTID(n)	0x0

1.3.3.21 MMU_PT_MATCH_ID_n

- Address = Base Address + 0x1000 + n × 0x20 + 0xC, n is from 1 to the number of page table
  Reset Value = 0x0000_0000

  MMU_PT_MATCH_ID_n is a page table ID configuration register.
  Number of registers varies based on the RTL configuration option for the number of page table. Offset address varies based on the index of target page table to be configured.

Name	Bit	Type	Description	Reset value
MASK	[31:16]	RW	Mask for matching the ID for PTID(n)	0x0
VALUE	[15:0]	RW	Value for matching the ID for PTID(n)	0x0

1.3.3.22 MMU_TLB_READ

- Address = Base Address + 0x8000, Reset Value = 0x0000_0000

MMU_TLB_READ makes TLB read command in SysMMU. TLB read command loads MMU_TLB_VPN, MMU_TLB_PPN, and MMU_TLB_ATTRIBUTE registers with corresponding contents of TLB read command.

Name	Bit	Type	Description	Reset value
RSVD	[31:24]	R	Reserved	0x0
TID	[23:20]	WO	TLB ID to read	0x0
RSVD	[19:18]	R	Reserved	0x0
SUBLINE	[17:16]	WO	Sub-line number to read	0x0
WAY	[15:8]	WO	Way number of sub-line to read	0x0
SET	[7:0]	WO	Set number of sub-line to read	0x0

1.3.3.23 MMU_TLB_VPN

- Address = Base Address + 0x8004, Reset Value = 0x0000_0000

MMU_TLB_VPN contains corresponding VPN result of MMU_TLB_READ.

Name	Bit	Type	Description	Reset value
RSVD	[31:29]	R	Reserved	0x0
VALID	[28]	RO	Indicates whether sub-line is valid. 0x0: Invalid 0x1: Valid	0x0
RSVD	[27:24]	R	Reserved	0x0
*VPN_UPPER	[23:20]	RO	Upper 4bits of VPN	0x0
VPN	[19:0]	RO	Virtual page number that is cached in the specified sub-line	0x0

* It is only accessible when 36bits-VA translation is enabled, otherwise it is reserved.

1.3.3.24 MMU_TLB_PPN

- Address = Base Address + 0x8008, Reset Value = 0x0000_0000

MMU_TLB_PPN contains corresponding PPN result of MMU_TLB_READ.

Name	Bit	Type	Description	Reset value
RSVD	[31:29]	R	Reserved	0x0
VALID	[28]	RO	Indicates whether sub-line is valid. 0x0: Invalid 0x1: Valid	0x0
RSVD	[27:22]	R	Reserved	0x0
PPN	[23:0]	RO	Physical page number that is cached in the specified sub-line	0x0

1.3.3.25 MMU_TLB_ATTRIBUTE

- Address = Base Address + 0x800C, Reset Value = 0x0000_0000

  MMU_TLB_ATTRIBUTE contains corresponding attributes result of MMU_TLB_READ.

Name	Bit	Type	Description	Reset value
RSVD	[31:29]	R	Reserved	0x0
VALID	[28]	RO	Indicates whether sub-line is valid. 0x0: Invalid 0x1: Valid	0x0
RSVD	[27:24]	R	Reserved	0x0
PTID	[23:20]	RO	Number of page table ID	0x0
RSVD	[19]	R	Reserved	0x0
C	[18:16]	RO	Indicates the cacheable field of page 0x0: Non-cacheable non-bufferable 0x2: Write-through no-allocate 0x4: Write-back no-allocate 0x6: Write-back read-allocate 0x5: Write-back write-allocate 0x7: Write-back read/write allocate Others: Not override	0x0
RSVD	[15:13]	R	Reserved	0x0
S	[12]	RO	Indicates whether this page is shareable of not 0x0: Non-shareable page 0x1: Shareable page	0x0
RSVD	[11]	R	Reserved	0x0
PS	[10:8]	RO	Indicates the page size of page 0x1: 4KB 0x2: 64KB 0x3: 1MB 0x4: 2MB 0x5: 16MB Others: Reserved	0x0
RSVD	[7:5]	R	Reserved	0x0
NS	[4]	RO	Indicates whether this page is non-secure or not 0x0: Secure page 0x1: Non-secure page	0x0
RSVD	[3:2]	R	Reserved	0x0
AP	[1:0]	RO	Indicates the access permission of the page 0x0: Access is not allowed 0x1: Read-only 0x2: Write-only 0x3: Read/write	0x0

1.3.3.26 MMU_SBB_READ

- Address = Base Address + 0x8020, Reset Value = 0x0000_0000

MMU_SBB_READ makes SBB read command in SysMMU. SBB read command loads MMU_SBB_VPN, MMU_SBB_LINK, and MMU_SBB_ATTRIBUTE registers with corresponding contents of SBB read command.

Name	Bit	Type	Description	Reset value
RSVD	[31:24]	R	Reserved	0x0
LINE	[4:0]	WO	Line number to read	0x0

1.3.3.27 MMU_SBB_VPN

- Address = Base Address + 0x8024, Reset Value = 0x0000_0000

MMU_SBB_VPN contains corresponding VPN result of MMU_SBB_READ.

Name	Bit	Type	Description	Reset value
RSVD	[31:29]	R	Reserved	0x0
VALID	[28]	RO	Indicates whether line is valid. 0x0: Invalid 0x1: Valid	0x0
RSVD	[27:24]	R	Reserved	0x0
*VPN_UPPER	[23:20]	RO	Upper 4bits of VPN	0x0
VPN	[19:0]	RO	Virtual page number that is cached in the specified line	0x0

* It is only accessible when 36bits-VA translation is enabled, otherwise it is reserved.

1.3.3.28 MMU_SBB_LINK

- Address = Base Address + 0x8028, Reset Value = 0x0000_0000

MMU_SBB_LINK contains corresponding LINK result of MMU_SBBB_READ.

Name	Bit	Type	Description	Reset value
RSVD	[31:29]	R	Reserved	0x0
VALID	[28]	RO	Indicates whether line is valid. 0x0: Invalid 0x1: Valid	0x0
RSVD	[27:26]	R	Reserved	0x0
LINK	[25:0]	RO	SLPT link information that is cached in the specified line.	0x0

1.3.3.29 MMU_SBB_ATTRIBUTE

- Address = Base Address + 0x802C, Reset Value = 0x0000_0000

  MMU_SBB_ATTRIBUTE contains corresponding attributes result of MMU_SBB_READ.

Name	Bit	Type	Description	Reset value
RSVD	[31:29]	R	Reserved	0x0
VALID	[28]	RO	Indicates whether line is valid. 0x0: Invalid 0x1: Valid	0x0
RSVD	[27:24]	R	Reserved	0x0
PTID	[23:20]	RO	Number of page table ID	0x0
RSVD	[19:5]	R	Reserved	0x0
NS	[4]	RO	Indicates whether this page is non-secure or not 0x0: Secure page 0x1: Non-secure page	0x0
RSVD	[3:0]	R	Reserved	0x0

1.3.3.30 MMU_EMULATION_PRELOAD

- Address = Base Address + 0x8040, Reset Value = 0x0000_0000

  MMU_EMULATION_PRELOAD

Field Name	Bit	TYPE	Description	Reset value
RSVD	[31:28]	RO	Reserved	0x0
*VPN_UPPER	[27:24]	WO	Upper 4bits of VPN	0x0
RSVD	[23:21]	RO	Reserved	0x0
RW	[20]	WO	It represents read or write request	0x0
VPN	[19:0]	WO	It represents target virtual page number for emulating address translation (VPN[19:0]).	0x0

1.3.3.31 MMU_EMULATION_SHOT

- Address = Base Address + 0x8044, Reset Value = 0x0000_0000

  MMU_EMULATION_SHOT

Field Name	Bit	TYPE	Description	Reset value
USER	[31:20]	WO	It represents user filed value to be emulated.	0x0
PID	[19:16]	WO	It represents AXI port information to be emulated.	0x0
ID	[15:0]	WO	It represents target virtual page number for emulating address translation.	0x0

1.3.4 Non-secure Global Register Set

1.3.4.1 MMU_CTRL

- Address = Base Address + 0x0000, Reset Value = 0x0000_0004

  MMU_CTRL is a control register of the basic operation. MMU_CTRL includes the operation mode and interrupt enable. Reset value varies based on the SFR set security attribute.

Name	Bit	Type	Description	Reset value
RSVD	[31:6]	R	Reserved	0x0
ERR_RESP_VALUE	[5]	RW	Error response value 0x0 = OKAY 0x1 = SLVERR	0x0
FAULT_MODE_OVERRIDE	[4]	RW	Fault mode override. If it is set 0x1, fault mode of all non-secure context is overridden by termination mode 0x0: not override 0x1: override by termination mode	0x0
RSVD	[3]	R	Reserved	0x0
INT_ENABLE	[2]	RW	Enables the interrupt generation 0x0 = Disable 0x1 = Enable	0x1
MMU_BLOCK	[1]	RW	Blocks the MMU requests 0x0 = Un-block requests 0x1 = Block requests	0x0
MMU_ENABLE	[0]	RW	Enables the SysMMU 0x0 = Disable SysMMU 0x1 = Enable SysMMU	0x0

1.3.4.2 MMU_CFG

- Address = Base Address + 0x0004, Reset Value = 0x0000_0000

  MMU_CFG is a configuration register to configure translation and page table walk operation.

Name	Bit	Type	Description	Reset value
RSVD	[31:12]	R	Reserved	0x0
QOS_OVERRIDE	[11]	RW	Indicates whether the MMU_CFG.QOS_VALUE overrides ARQOS_M0 or not 0x0 = QoS value of PTW requests is inherited from the corresponding requests issued from master 0x1 = QoS value of PTW requests is forced to MMU_CFG. QOS_VALUE	0x0
QOS_VALUE	[10:7]	RW	QoS value of PTW requests It is only effective when QOS_OVERRIDE is 0x1.	0x0
RSVD	[6:0]	R	Reserved	0x0

1.3.4.3 MMU_STATUS

- Address = Base Address + 0x0008, Reset Value = 0x0000_0000

  MMU_STATUS is current status register. Reset value varies based on the SFR set security attribute.

Name	Bit	Type	Description	Reset value
RSVD	[31:5]	R	Reserved	0x0
ON_INVALIDATING	[4]	R	Indicates the status of invalidation 0x0 = There is no invalidation in SysMMU 0x1 = There is ongoing invalidation in SysMMU	0x0
AR_PEND	[3]	R	Indicates the read request on translation in SysMMU 0x0 = There is no read request on translation in SysMMU 0x1 = There is a read request on translation in SysMMU	0x0
AW_PEND	[2]	R	Indicates the write request on translation in SysMMU 0x0 = There is no write request on translation in SysMMU 0x1 = There is a write request on translation in SysMMU	0x0
PTW_ACTIVE	[1]	R	Indicates the active PTW request 0x0 = There is no active PTW request 0x1 = There is an active PTW request	0x0
BLOCKED	[0]	R	Indicates whether the AXI slave interface of SysMMU is blocked or not 0x0 = AXI slave interface of SysMMU is un-blocked 0x1 = AXI slave interface of SysMMU is blocked	0x0

1.3.4.4 MMU_VERSION

- Address = Base Address + 0x0034, Reset Value = 0x7800_0000

  MMU_VERSION contains the version information. The reset value varies with the minor architecture versions and RTL version.

Name	Bit	Type	Description	Reset value
MAJOR_ARCH_VER	[31:28]	R	Major architecture version	0x7
MINOR_ARCH_VER	[27:21]	R	Minor architecture version	0x4
REV_ARCH_VER	[20:17]	R	Revision architecture version	0x0
RSVD	[16:8]	R	Reserved	0x0
RTL_VER	[7:0]	R	RTL version	0x0

1.3.4.5 MMU_INTERRUPT_STATUS

- Address = Base Address + 0x0060, Reset Value = 0x0000_0000

  MMU_INTERRUPT_STATUS stores the current interrupt information.

Name	Bit	Type	Description	Reset value

Name	Bit	Type	Description	Reset value
RSVD	[31:5]	R	Reserved	0x0
SECURITY_FAULT	[4]	R	Security fault 0x0 = This fault is not detected 0x1 = This fault is detected	0x0
ACCESS_FAULT	[3]	R	Access permission fault 0x0 = This fault is not detected 0x1 = This fault is detected	0x0
RSVD	[2]	R	Reserved	0x0
PAGE_FAULT	[1]	R	Page fault 0x0 = This fault is not detected 0x1 = This fault is detected	0x0
PTW_ACCESS_FAULT	[0]	R	PTW access fault 0x0 = This fault is not detected 0x1 = This fault is detected	0x0

1.3.4.6 MMU_INTERRUPT_CLEAR

- Address = Base Address + 0x0064, Reset Value = 0x0000_0000

  MMU_INTERRUPT_CLEAR is a register to clear the interrupt. To clear the current interrupt, set the corresponding register field. If the interrupt is not generated, MMU_INTERRUPT_CLEAR has no effect.

Name	Bit	Type	Description	Reset value
RSVD	[31:5]	R	Reserved	0x0
SECURITY_FAULT	[4]	W	Security fault 0x0 = No operation 0x1 = Clear this fault status	0x0
ACCESS_FAULT	[3]	W	Access permission fault 0x0 = No operation 0x1 = Clear this fault status	0x0
RSVD	[2]	R	Reserved	0x0
PAGE_FAULT	[1]	W	Page fault 0x0 = No operation 0x1 = Clear this fault status	0x0
PTW_ACCESS_FAULT	[0]	W	PTW access fault 0x0 = No operation 0x1 = Clear this fault status	0x0

1.3.4.7 MMU_FAULT_VA

- Address = Base Address + 0x0070, Reset Value = 0x0000_0000

  MMU_FAULT_VA contains the virtual address that causes the fault.

Name	Bit	Type	Description	Reset value
VA	[31:0]	R	Virtual address of the request with fault (VA[31:0])	0x0

1.3.4.8 MMU_FAULT_TRANS_INFO

- Address = Base Address + 0x0078, Reset Value = 0x0000_0000

  MMU_FAULT_TRANS_INFO contains the transaction information that causes the fault.

Name	Bit	Type	Description	Reset value
*VA_UPPER	[31:28]	R	Upper 4bits of VA	0x0
PID	[27:24]	R	Port where the request come from	0x0
RSVD	[23:21]	R	Reserved	0x0
RW	[20]	R	It represents whether the request with fault is read request or write request 0 = Read request 1 = Write request	0x0
LEN	[19:16]	R	AxLEN of the request with fault	0x0
ID	[15:0]	R	AxID of the request with fault	0x0

* It is only accessible when 36bits-VA translation is enabled, otherwise it is reserved.

1.3.4.9 MMU_FAULT_TRANS_INFO2

- Address = Base Address + 0x007C, Reset Value = 0x0000_0000

  MMU_FAULT_TRANS_INFO contains the transaction information that causes the fault.

Name	Bit	Type	Description	Reset value
RSVD	[31:16]	R	Reserved	0x0
PTID	[15:0]	R	PTID of the request with fault	0x0

1.3.4.10 MMU_CAPA_0

- Address = Base Address + 0x0870, Reset Value is different from the configuration set.

  MMU_CAPA_0 register indicates the SysMMU capacity information.

Name	Bit	Type	Description	Reset value
MAX_PTW_MO	[31:26]	R	Maximum number of outstanding PTW	0x8 (NOTE)
MAX_AT_MO	[25:20]	R	Maximum number of outstanding address translation	0x8 (NOTE)
NUM_PAGE_TABLE	[19:16]	R	Number of page tables in SysMMU	0x1 (NOTE)
NUM_SBB_ENTRY	[15:12]	R	Indicates the number of SBB entries 0x0 = 1 entry 0x1 = 2 entries 0x2 = 4 entries 0x3 = 8 entries	0x3 (NOTE)

			0x4 = 16 entries 0x5 = 32 entries Others = Reserved	
CAPA_1_EXIST	[11]	R	Indicates existence of MMU_CAPA_1 0x0 = MMU_CAPA_1 SFR is not exist 0x1 = MMU_CAPA_1 SFR is exist	0x1 (NOTE)
RSVD	[10:0]	R	Reserved	0x0

NOTE: Each value of MMU_CAPA_0 field is different from the configuration set.

1.3.4.11 MMU_CAPA_1

- Address = Base Address + 0x0874, Reset Value is different from the configuration set.
  MMU_CAPA_1 register indicates the SysMMU capacity information.

Name	Bit	Type	Description	Reset value
TYPE	[31:28]	R	Indicates type of MMU_CAPA_1 0x0 = type0 (NUM_TLB, NUM_PORT) Others = Reserved	0x0
RSVD	[27:13]	R	Reserved	0x0
VCR_ENABLE	[14]	R	Virtual machine control register enable 0x0: disabled 0x1: enabled	0x1
STALL_MODE_SUPPORT	[13]	R	Indicates VA width 0x0: stall mode not supported 0x1: stall mode supported	0x1
VA_WIDTH	[12]	R	Indicates VA width 0x0: VA width is 32 bits 0x1: VA width is 36 bits	0x1(NOTE)
NUM_TLB	[11:4]	R	The number of TLBs in SysMMU	0x4 (NOTE)
NUM_PORT	[3:0]	R	The number of AXI ports of SysMMU	0x1 (NOTE)

NOTE: Each value of MMU_CAPA_1 field is different from the configuration set.

1.3.4.12 MMU_ECC_EN

- Address = Base Address + 0x0880, Reset Value = 0x0000_0000
  MMU_ECC_EN is a control register of error code correction.

Name	Bit	Type	Description	Reset value
RSVD	[31:1]	R	Reserved	0x0
ECC_ENABLE	[0]	RW	Enables ECC(error correction code) 0x0 = Disable ECC 0x1 = Enable ECC	0x0

1.3.4.13 MMU_TLB_INFO_n

- Address = Base Address + 0x2000 + n × 0x20, n is from 0 to the number of TLB
  Reset Value is different from the configuration set

  MMU_TLB_INFO_n is a TLB information register. Number of registers varies based on the RTL configuration option for the number of TLB.

Name	Bit	Type	Description	Reset value
RSVD	[31:24]	R	Reserved	0x0
TLB_SET	[23:16]	R	The number of sets in TLB	0x0
RSVD	[15:8]	R	Reserved	0x0
TLB_WAY	[7:0]	R	The number of ways in TLB	0x0

1.3.4.14 MMU_TLB_CFG_n

- Address = Base Address + 0x2000 + n x 0x20 + 0x4, n is from 0 to the number of TLB
  Reset Value = 0x0000_0000

  MMU_TLB_CFG_n is a configuration register that configures the property of n-th TLB.

Name	Bit	Type	Description	Reset value
RSVD	[31:8]	R	Reserved	0x0
FETCH_SIZE	[7:5]	RW	Indicates fetch size of PTW 0x0 = BL1 (4 page descriptors) 0x1 = BL2 (8 page descriptors) 0x2 = BL4 (16 page descriptors) 0x3 = BL8 (32 page descriptors) 0x4 = BL16 (64 page descriptors) 0x5 = BL32 (128 page descriptors) Others: reserved	0x0
RSVD	[4]	R	Reserved	0x0
PREFETCH_DIR	[3:2]	RW	Indicates whether the pre-fetch address is calculated in ascending order or descending order 0x0 = Pre-fetch address is calculated in descending order 0x1 = Pre-fetch address is calculated in ascending order Others: Pre-fetch address is calculated by prediction	0x0
PREFETCH_ENABLE	[1]	RW	Enables the pre-fetch 0x0 = Disable pre-fetch 0x1 = Enable pre-fetch	0x0
RSVD	[0]	R	Reserved	0x0

1.3.4.15 MMU_TLB_MATCH_CFG_n

- Address = Base Address +0x2000 + n × 0x20 +0x8, n is from 1 to the number of TLB
  Reset Value = 0x0000_0000

  MMU_TLB_MATCH_CFG_n is a configuration register, which configures matching source of n-th TLB.

Name	Bit	Type	Description	Reset value
PORT_MASK	[31:16]	RW	It indicates whether each port can be matched to the TLB. Each bit indicates each AXI port For example, the value of PORT_MASK[0] determines whether requests from port 0 can be matched to TLB(n) or not. In the same way, the value of PORT_MASK[1] determines whether requests from port 1 can be matched TLB(n) or not.	0x0
RSVD	[15:10]	R	Reserved	0x0
TARGET_CH	[9:8]	RW	Indicates which direction (Read/Write) is matched to the TLB 0x0 = Both read/write requests are not matched to TLB(n) 0x1 = Only read requests are matched to TLB(n) 0x2 = Only write requests are matched to TLB(n) 0x3 = Both read/write requests are matched to TLB(n)	0x0
*EVA_UPPER	[7:4]	RW	Upper 4bits of EVA	0xF
*SVA_UPPER	[3:0]	RW	Upper 4bits of SVA	0x0

* It is only accessible when 36bits-VA translation is enabled, otherwise it is reserved.

1.3.4.16 MMU_TLB_MATCH_SVA_n

- Address = Base Address + 0x2000 + n × 0x20 + 0xC, n is from 1 to the number of TLB
  Reset Value = 0x0000_0000

  MMU_TLB_MATCH_SVA_n contains the start virtual address of range of n-th TLB. If the virtual address of transaction is in the range of MMU_TLB_MATCH_SVA_n and MMU_TLB_MATCH_EVA_n, the transaction is matched to n-th TLB.

Name	Bit	Type	Description	Reset value
VA	[31:0]	RW	Start virtual address (VA[31:0])	0x0

1.3.4.17 MMU_TLB_MATCH_EVA_n

- Address = Base Address + 0x2000 + n × 0x20 + 0x10, n is from 1 to the number of TLB
  Reset Value = 0xFFFF_FFFF

  MMU_TLB_MATCH_EVA_n contains the start virtual address of range of n-th TLB. If the virtual address of transaction is in the range of MMU_TLB_MATCH_SVA_n and MMU_TLB_MATCH_EVA_n, the transaction is matched to n-th TLB.

Name	Bit	Type	Description	Reset value
VA	[31:0]	RW	End virtual address (VA[31:0])	0xFFFF_FFFF

1.3.4.18 MMU_TLB_MATCH_ID_n

- Address = Base Address + 0x2000 + n × 0x20 + 0x14, n is from 1 to the number of TLB
  Reset Value = 0x0000_0000

  MMU_TLB_MATCH_ID contains ID value and mask of n-th TLB for TLB mathcing. If a transaction ID which is masked with MMU_TLB_MATCH_ID_n.MASK matches with MMU_TLB_MATCH_ID_n.VALUE, the transaction is in the dedicated n-th TLB.

Name	Bit	Type	Description	Reset value
MASK	[31:16]	RW	Mask	0x0
VALUE	[15:0]	RW	Value	0x0

1.3.4.19 MMU_TLB_READ

- Address = Base Address + 0x8000, Reset Value = 0x0000_0000

MMU_TLB_READ makes TLB read command in SysMMU. TLB read command loads MMU_TLB_VPN, MMU_TLB_PPN, and MMU_TLB_ATTRIBUTE registers with corresponding contents of TLB read command.

Name	Bit	Type	Description	Reset value
RSVD	[31:24]	R	Reserved	0x0
TID	[23:20]	WO	TLB ID to read	0x0
RSVD	[19:18]	R	Reserved	0x0
SUBLINE	[17:16]	WO	Sub-line number to read	0x0
WAY	[15:8]	WO	Way number of sub-line to read	0x0
SET	[7:0]	WO	Set number of sub-line to read	0x0

1.3.4.20 MMU_TLB_VPN

- Address = Base Address + 0x8004, Reset Value = 0x0000_0000
  MMU_TLB_VPN contains corresponding VPN result of MMU_TLB_READ.

Name	Bit	Type	Description	Reset value
RSVD	[31:29]	R	Reserved	0x0
VALID	[28]	RO	Indicates whether sub-line is valid. 0x0: Invalid 0x1: Valid	0x0
RSVD	[27:24]	R	Reserved	0x0
*VPN_UPPER	[23:20]	RO	Upper 4bits of VPN	0x0
VPN	[19:0]	RO	Virtual page number that is cached in the specified sub-line	0x0

* It is only accessible when 36bits-VA translation is enabled, otherwise it is reserved.

1.3.4.21 MMU_TLB_PPN

- Address = Base Address + 0x8008, Reset Value = 0x0000_0000
  MMU_TLB_PPN contains corresponding PPN result of MMU_TLB_READ.

Name	Bit	Type	Description	Reset value
RSVD	[31:29]	R	Reserved	0x0
VALID	[28]	RO	Indicates whether sub-line is valid. 0x0: Invalid 0x1: Valid	0x0
RSVD	[27:22]	R	Reserved	0x0
PPN	[23:0]	RO	Physical page number that is cached in the specified sub-line	0x0

1.3.4.22 MMU_TLB_ATTRIBUTE

- Address = Base Address + 0x800C, Reset Value = 0x0000_0000
  MMU_TLB_ATTRIBUTE contains corresponding attributes result of MMU_TLB_READ.

Name	Bit	Type	Description	Reset value
RSVD	[31:29]	R	Reserved	0x0
VALID	[28]	RO	Indicates whether sub-line is valid. 0x0: Invalid 0x1: Valid	0x0
RSVD	[27:24]	R	Reserved	0x0
PTID	[23:20]	RO	Number of page table ID	0x0
RSVD	[19]	R	Reserved	0x0
C	[18:16]	RO	Indicates the cacheable field of page 0x0: Non-cacheable non-bufferable 0x2: Write-through no-allocate 0x4: Write-back no-allocate 0x6: Write-back read-allocate 0x5: Write-back write-allocate 0x7: Write-back read/write allocate Others: Not override	0x0
RSVD	[15:13]	R	Reserved	0x0
S	[12]	RO	Indicates whether this page is shareable of not 0x0: Non-shareable page 0x1: Shareable page	0x0
RSVD	[11]	R	Reserved	0x0
PS	[10:8]	RO	Indicates the page size of page 0x1: 4KB 0x2: 64KB 0x3: 1MB 0x4: 2MB 0x5: 16MB Others: Reserved	0x0
RSVD	[7:5]	R	Reserved	0x0
NS	[4]	RO	Indicates whether this page is non-secure or not 0x0: Secure page 0x1: Non-secure page	0x0
RSVD	[3:2]	R	Reserved	0x0
AP	[1:0]	RO	Indicates the access permission of the page 0x0: Access is not allowed 0x1: Read-only 0x2: Write-only 0x3: Read/write	0x0

1.3.4.23 MMU_SBB_READ

- Address = Base Address + 0x8020, Reset Value = 0x0000_0000

MMU_SBB_READ makes SBB read command in SysMMU. SBB read command loads MMU_SBB_VPN, MMU_SBB_LINK, and MMU_SBB_ATTRIBUTE registers with corresponding contents of SBB read command.

Name	Bit	Type	Description	Reset value
RSVD	[31:24]	R	Reserved	0x0
LINE	[4:0]	WO	Line number to read	0x0

1.3.4.24 MMU_SBB_VPN

- Address = Base Address + 0x8024, Reset Value = 0x0000_0000

MMU_SBB_VPN contains corresponding VPN result of MMU_SBB_READ.

Name	Bit	Type	Description	Reset value
RSVD	[31:29]	R	Reserved	0x0
VALID	[28]	RO	Indicates whether line is valid. 0x0: Invalid 0x1: Valid	0x0
RSVD	[27:24]	R	Reserved	0x0
*VPN_UPPER	[23:20]	RO	Upper 4bits of VPN	0x0
VPN	[19:0]	RO	Virtual page number that is cached in the specified line	0x0

* It is only accessible when 36bits-VA translation is enabled, otherwise it is reserved.

1.3.4.25 MMU_SBB_LINK

- Address = Base Address + 0x8028, Reset Value = 0x0000_0000

MMU_SBB_LINK contains corresponding LINK result of MMU_SBBB_READ.

Name	Bit	Type	Description	Reset value
RSVD	[31:29]	R	Reserved	0x0
VALID	[28]	RO	Indicates whether line is valid. 0x0: Invalid 0x1: Valid	0x0
RSVD	[27:26]	R	Reserved	0x0
LINK	[25:0]	RO	SLPT link information that is cached in the specified line.	0x0

1.3.4.26 MMU_SBB_ATTRIBUTE

- Address = Base Address + 0x802C, Reset Value = 0x0000_0000

  MMU_SBB_ATTRIBUTE contains corresponding attributes result of MMU_SBB_READ.

Name	Bit	Type	Description	Reset value
RSVD	[31:29]	R	Reserved	0x0
VALID	[28]	RO	Indicates whether line is valid. 0x0: Invalid 0x1: Valid	0x0
RSVD	[27:24]	R	Reserved	0x0
PTID	[23:20]	RO	Number of page table ID	0x0
RSVD	[19:5]	R	Reserved	0x0
NS	[4]	RO	Indicates whether this page is non-secure or not 0x0: Secure page 0x1: Non-secure page	0x0
RSVD	[3:0]	R	Reserved	0x0

1.3.4.27 MMU_EMULATION_PRELOAD

- Address = Base Address + 0x8040, Reset Value = 0x0000_0000
- MMU_EMULATION_PRELOAD

Field Name	Bit	TYPE	Description	Reset value
RSVD	[31:28]	RO	Reserved	0x0
*VPN_UPPER	[27:24]	WO	Upper 4bits of VPN	0x0
RSVD	[23:21]	RO	Reserved	0x0
RW	[20]	WO	It represents read or write request	0x0
VPN	[19:0]	WO	It represents target virtual page number for emulating address translation (VPN[19:0]).	0x0

1.3.4.28 MMU_EMULATION_SHOT

- Address = Base Address + 0x8044, Reset Value = 0x0000_0000
- MMU_EMULATION_SHOT

Field Name	Bit	TYPE	Description	Reset value
USER	[31:20]	WO	It represents user filed value to be emulated.	0x0
PID	[19:16]	WO	It represents AXI port information to be emulated.	0x0
ID	[15:0]	WO	It represents target virtual page number for emulating address translation.	0x0

1.3.4.29 PM_CFG

- Base Address: Refer Table 1-15 which describes base address of each block.
- Address = Base Address + 0x9000
- PM_CFG is configuration register for performance measurement.

Field Name	Bit	TYPE	Description	Reset value
RSVD	[31:1]	RO	Reserved	0x0
CNT_EN	[0]	RW	0x0 : Count is disabled 0x1 : Counter is enabled	0x0

1.3.4.30 PM_INIT

- Base Address: Refer Table 1-15 which describes base address of each block.
- Address = Base Address + 0x9004
- PM_INIT is for initializing performance counters

Field Name	Bit	TYPE	Description	Reset value
RSVD	[31:1]	RO	Reserved	0x0
INIT	[0]	WO	0x0 : No operation 0x1 : Counter is initialized.	0x0

1.3.4.31 PM_EVENTn

- Base Address: Refer Table 1-15 which describes base address of each block.
- Address = Base Address + 0x9010 + 0x4*n (where n is 4)
- PM_EVENTn is configuration register for setting target event to be measured by the corresponding performance counter. 4 performance events can be measured, concurrently. You can specify TLB, AXI port and access type for more detail performance measurement.

Field Name	Bit	TYPE	Description	Reset

				value
RSVD	[31:20]	RO	Reserved	0x0
SPECIFIED_TID	[19]	RW	0x0 : Target TLB is not specified, all TLBs are target. 0x1 : Target TLB is specified, TLB is referred.	0x0
TID	[18:12]	RW	Target TLB to be measured.	0x0
SPECIFIED_PID	[11]	RW	0x0 : Target AXI port is not specified, all AXI ports are target. 0x1 : Target AXI port is specified, AXI_PORT is referred.	0x0
PID	[10:8]	RW	Target AXI port to be measured	0x0
RSVD	[7:6]	RO	Reserved	0x0
ACCESS_TYPE	[5:4]	RW	Target access type to be measured 0x1 : Read only 0x2 : Write only 0x3 : Read and Write Others : Reserved	0x3
EVENT	[3:0]	RW	0x0 : Total number of requests 0x1 : Total number of TLB miss 0x2 : Total number of SBB miss 0x3 : Total number of PTW read data Others : Reserved	0x0

1.3.4.32 PM_CNTn

- Base Address: Refer Table 1-15 which describes base address of each block.
- Address = Base Address + 0x9020 + 0x4*n (where n is 4)
- PM_CNTn is read-only SFRs which shows performance counting value. You can initialize this counter value as 0x0 by writing PM_INIT.

Field Name	Bit	TYPE	Description	Reset value
VALUE	[31:0]	RO	Counter value	0x0

1.3.5 Non-secure Virtual Context Register Set

1.3.5.1 MMU_CTRL

- Address = Base Address + 0x1000 * n + 0x0000 (n is from 0 to 7), Reset Value = 0x0000_0007

  MMU_CTRL is a control register of the basic operation. MMU_CTRL includes the operation mode and interrupt enable. Reset value varies based on the SFR set security attribute.

Name	Bit	Type	Description	Reset value
RSVD	[31:4]	R	Reserved	0x0
FAULT_MODE	[3]	RW	Fault mode 0x0: termination mode 0x1: stall mode	0x0
RSVD	[2:1]	R	Reserved	0x0
MMU_ENABLE	[0]	RW	Enables the SysMMU 0x0 = Disable SysMMU 0x1 = Enable SysMMU	0x1

1.3.5.2 MMU_CFG

- Address = Base Address + 0x1000 * n + 0x0004 (n is from 0 to 7), Reset Value = 0x0002_0000

  MMU_CFG is a configuration register to configure translation and page table walk operation.

Name	Bit	Type	Description	Reset value
CACHEABLE_OVERRIDE	[31]	RW	Indicates whether the cacheable field in page descriptor overrides the AxCACHE of normal request or not 0x0 = AxCACHE of normal request does not override 0x1 = AxCACHE of normal request is overridden by the cacheable field in page descriptor	0x0
RSVD	[30]	R	Reserved	0x0
SHAREABILITY	[29]	RW	Indicates the translated results of request is shareable or not This value is overrode to AxSHAREABLE_M0 when MMU_CFG.SHAREABILITY_OVERRID = 0x1. 0x0 = Non-shareable 0x1 = Shareable	0x0
SHAREABILITY_OVERRIDE	[28]	RW	Indicates whether the MMU_CFG.SHAREABILITY overrides AxSHAREABLE_M0 or not 0x0 = AxSHAREABLE_M0 of master requests is set by shareability information of the corresponding entry in the page table 0x1 = AxSHAREABLE_M0 of master requests is forced to MMU_CFG. SHAREBILITY	0x0
RSVD	[27:20]	R	Reserved	0x0
PTW_ARCACHE	[19:16]	RW	ARCACHE value of PTW request	0x2
RSVD	[15:13]	R	Reserved	0x0

Name	Bit	Type	Description	Reset value
PT_SHAREABLE	[12]	RW	Indicates the value of the ARSHAREABLE_M0 of PTW request 0x0 = Page tables are located in a non-sharable region. ARSHARABLE_M0 is 0x0 for PTW request. 0x1 = Page tables are located in a sharable region. ARSHARABLE_M0 is 0x1 for PTW request.	0x0
RSVD	[11:4]	R	Reserved	0x0
SECURITY_PROT_ENABLE	[3]	RW	Enables the security protection It is effective only when SysMMU is enabled 0x0 = Disable security protection 0x1 = Enable security protection	0x0
ACCESS_PROT_ENABLE	[2]	RW	Enables the access protection checking It is effective only when SysMMU is enabled 0x0 = Disable checking access protection 0x1 = Enable checking access protection	0x0
RSVD	[1:0]	R	Reserved	0x0

1.3.5.3 MMU_DEFAULT_FLPT_BASE

- Address = Base Address + 0x1000 * n + 0x000C (n is from 0 to 7), Reset Value = 0x0000_0000.

  MMU_DEFAULT_FLPT_BASE is first-level page table base address register that stores the base address of first-level page table.

Name	Bit	Type	Description	Reset value
RSVD	[31:24]	R	Reserved	0x0
FLPT_BASE_PPN	[23:0]	RW	PPN of first-level page table for default page table (default page table means page table 0)	0x0

1.3.5.4 MMU_ALL_INVALIDATION

- Address = Base Address + 0x1000 * n + 0x0010 (n is from 0 to 7), Reset Value = 0x0000_0000

  MMU_ALL_INVALIDATION is a register that invalidates all page descriptors in SysMMU.
  When MMU_ALL_INVALIDATION.INVALIDATE = 0x1, the only page table of ID that matches with MMU_ALL_INVALIDATION.PTID is invalidated. If MMU_ALL_INVALIDATION.INVALIDATE = 0x3, all page table regardless of PTID are invalidated.

Name	Bit	Type	Description	Reset value
RSVD	[31:2]	R	Reserved	0x0
INVALIDATE	[1:0]	W	Indicates invalidation mode 0x0 = No operation 0x1 = All-invalidation for a specific page table 0x2 = No operation 0x3 = All-invalidation for all page table	0x0

1.3.5.5 MMU_VPN_INVALIDATION

- Address = Base Address + 0x1000 * n + 0x0014 (n is from 0 to 7), Reset Value = 0x0000_0000

  MMU_VPN_INVALIDATION is a register that invalidates the specific page descriptor in the page table of PTID that matches with MMU_VPN_INVALIDATION.PTID

Name	Bit	Type	Description	Reset value
VPN	[31:12]	W	Virtual page number for 4 KB page granularity (VPN[19:0])	0x0
*VPN_UPPER	[11:8]	W	Upper 4bits of VPN	0x0
RSVD	[7:1]	R	Reserved	0x0
INVALIDATE	[0]	W	Indicates invalidation mode 0x0 = No operation 0x1 = VPN-invalidation for a specific page table	0x0

* It is only accessible when 36bits-VA translation is enabled, otherwise it is reserved.

1.3.5.6 MMU_RANGE_INVALIDATION

- Address = Base Address + 0x1000 * n + 0x0018 (n is from 0 to 7), Reset Value = 0x0000_0000

  MMU_RANGE_INVALIDATION is a register to invalidate all the page descriptor within ranges. Before setting MMU_RANGE_INVALIDATION, ensure that MMU_RANGE_INVALIDATION_START_VPN and MMU_RANGE_INVALIDATION_END_VPN are set.

Name	Bit	Type	Description	Reset value
RSVD	[31:1]	R	Reserved	0x0
INVALIDATE	[0]	W	Indicates invalidation mode 0x0 = No operation 0x1 = Range-invalidation for a specific page table	0x0

1.3.5.7 MMU_RANGE_INVALIDATION_START_VPN

- Address = Base Address + 0x1000 * n + 0x0020, Reset Value = 0x0000_0000

  MMU_RANGE_INVALIDATION_START_VPN register represents the start VPN of range invalidation.

Name	Bit	Type	Description	Reset value
VPN	[31:12]	W	Start virtual page number for 4 KB page granularity of range-invalidation (VPN[19:0])	0x0
*VPN_UPPER	[11:8]	W	Upper 4bits of VPN	0x0
RSVD	[7:0]	R	Reserved	0x0

* It is only accessible when 36bits-VA translation is enabled, otherwise it is reserved.

1.3.5.8 MMU_RANGE_INVALIDATION_END_VPN

- Address = Base Address + 0x1000 * n + 0x0024, Reset Value = 0x0000_0000

  MMU_RANGE_INVALIDATION_END_VPN register indicates the end VPN of range invalidation.

Name	Bit	Type	Description	Reset value
VPN	[31:12]	W	End virtual page number for 4 KB page granularity of range-invalidation (VPN[19:0])	0x0
*VPU_UPPPER	[11:8]	R	Upper 4bits of VPN	0x0
RSVD	[7:0]	R	Reserved	0x0

* It is only accessible when 36bits-VA translation is enabled, otherwise it is reserved.

1.3.5.9 MMU_VERSION

- Address = Base Address + 0x1000 * n + 0x0034, Reset Value = 0x7800_0000

  MMU_VERSION contains the version information. The reset value varies with the minor architecture versions and RTL version.

Name	Bit	Type	Description	Reset value
MAJOR_ARCH_VER	[31:28]	R	Major architecture version	0x7
MINOR_ARCH_VER	[27:21]	R	Minor architecture version	0x4
REV_ARCH_VER	[20:17]	R	Revision architecture version	0x0
RSVD	[16:8]	R	Reserved	0x0
RTL_VER	[7:0]	R	RTL version	0x0

1.3.5.10 MMU_CAPA_0

- Address = Base Address + 0x1000 * n + 0x0870, Reset Value is different from the configuration set.

  MMU_CAPA_0 register indicates the SysMMU capacity information.

Name	Bit	Type	Description	Reset value
MAX_PTW_MO	[31:26]	R	Maximum number of outstanding PTW	0x8 (NOTE)
MAX_AT_MO	[25:20]	R	Maximum number of outstanding address translation	0x8 (NOTE)
NUM_PAGE_TABLE	[19:16]	R	Number of page tables in SysMMU	0x1 (NOTE)
NUM_SBB_ENTRY	[15:12]	R	Indicates the number of SBB entries 0x0 = 1 entry 0x1 = 2 entries 0x2 = 4 entries 0x3 = 8 entries 0x4 = 16 entries 0x5 = 32 entries Others = Reserved	0x3 (NOTE)
CAPA_1_EXIST	[11]	R	Indicates existence of MMU_CAPA_1 0x0 = MMU_CAPA_1 SFR is not exist 0x1 = MMU_CAPA_1 SFR is exist	0x1 (NOTE)
RSVD	[10:0]	R	Reserved	0x0

NOTE: Each value of MMU_CAPA_0 field is different from the configuration set.

1.3.5.11 MMU_CAPA_1

- Address = Base Address + 0x1000 * n + 0x0874, Reset Value is different from the configuration set. MMU_CAPA_1 register indicates the SysMMU capacity information.

Name	Bit	Type	Description	Reset value
TYPE	[31:28]	R	Indicates type of MMU_CAPA_1 0x0 = type0 (NUM_TLB, NUM_PORT) Others = Reserved	0x0
RSVD	[27:13]	R	Reserved	0x0
VCR_ENABLE	[14]	R	Virtual machine control register enable 0x0: disabled 0x1: enabled	0x1
STALL_MODE_SUPPORT	[13]	R	Indicates VA width 0x0: stall mode not supported 0x1: stall mode supported	0x1
VA_WIDTH	[12]	R	Indicates VA width 0x0: VA width is 32 bits 0x1: VA width is 36 bits	0x1(NOTE)
NUM_TLB	[11:4]	R	The number of TLBs in SysMMU	0x4 (NOTE)
NUM_PORT	[3:0]	R	The number of AXI ports of SysMMU	0x1 (NOTE)

NOTE: Each value of MMU_CAPA_1 field is different from the configuration set.

NOTE: