AN14892

S32N55: Managing Isolation and XRDC Configuration via Domain Manager

Rev. 1.0 — 17 December 2025 | Application note

Document information

Information	Content
Keywords	S32N55, 5 nm S32N family of products, domain manager
Abstract	S32N55 is one of the initial chips in NXP's next-generation 5 nm S32N family of products. This application note is intended to help users understand the cohort concept and the steps to configure cohorts.

1 Overview

S32N55 is one of the initial chips in NXP's next-generation 5 nm S32N family of products. This chip is a real-time processor for multi-chip domain control applications and supports multi-application ASIL D control through advanced real-time processing, hardware isolation, and virtualization.

This application note is intended to help users understand the cohort concept and the steps to configure cohorts. The application note is only to be used as a reference, and for any conflicts in sections, users are advised to refer to the S32N55 RM[1] and FSS FW User Manual[2]. It is advised to go through the "Resource Management Overview" section of S32N55 RM[1], and “Cohort Definition”, “Booting a Cohort”, “Cohort States”, and “Cohort Status” sections of Boot Manager in the FSS FW User Manual[2].

The examples in this application note are developed based on the following software environment:

- FSS: S32N_FSS_FW_R21-11_1.8.1
- RTD: SW32N_RTD_R21-11_1.8.0_CD03
- GrayVIP 1.0.21.0 with 6 cohorts

2 Cohort concept and subsystem

A cohort is a computing environment composed of one or more UENVs that are deployed together to solve a common problem. In a cohort containing multiple UENVs, the UENVs are not required to have the same core architecture or lockstep mode.

Cohorts are defined by the chip's partition manager during the boot process. The figure below illustrates an example of how a chip's resources can be partitioned to support a specific application.

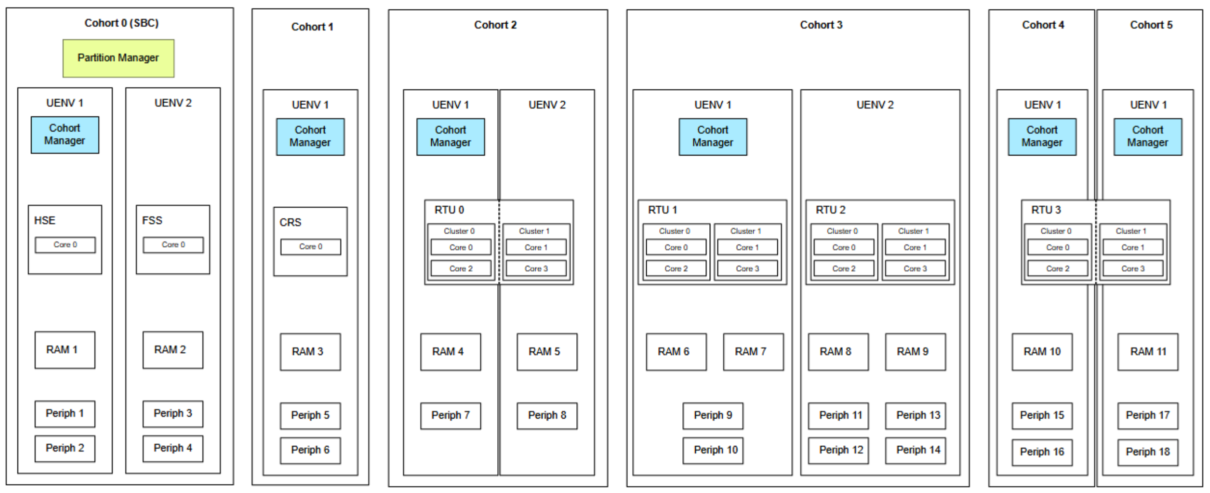


Figure 1. Application cohort diagram

2.1 Uniform environment

A Uniform Environment (UENV) is defined as a group of one or more cores that share the same architecture and lockstep mode, along with a set of associated resources such as memory regions and peripherals, which are accessible by those cores.

By definition, an RTU instance constitutes a single UENV, meaning it can be assigned to only one cohort. However, each RTU contains two clusters, and each cluster is also considered a UENV. As a result, the two clusters within an RTU can be assigned to different UENVs, even if those UENVs belong to different cohorts.

Note: *A UENV can only be included in at most one cohort.*

2.2 Subsystems

Several subsystems exist within the chip through hardware partitioning, such as FSS, CRS, RTU, and CIS. A subsystem is a hardware-defined concept, meaning it is determined during the design phase and cannot be modified by users. Memory and peripherals are assigned to a specific subsystem, indicating that these resources are tightly coupled to a particular bus, and their clock and power are managed by the same control instance. The diagram below illustrates the main subsystems in the S32N55 and how they are interconnected.

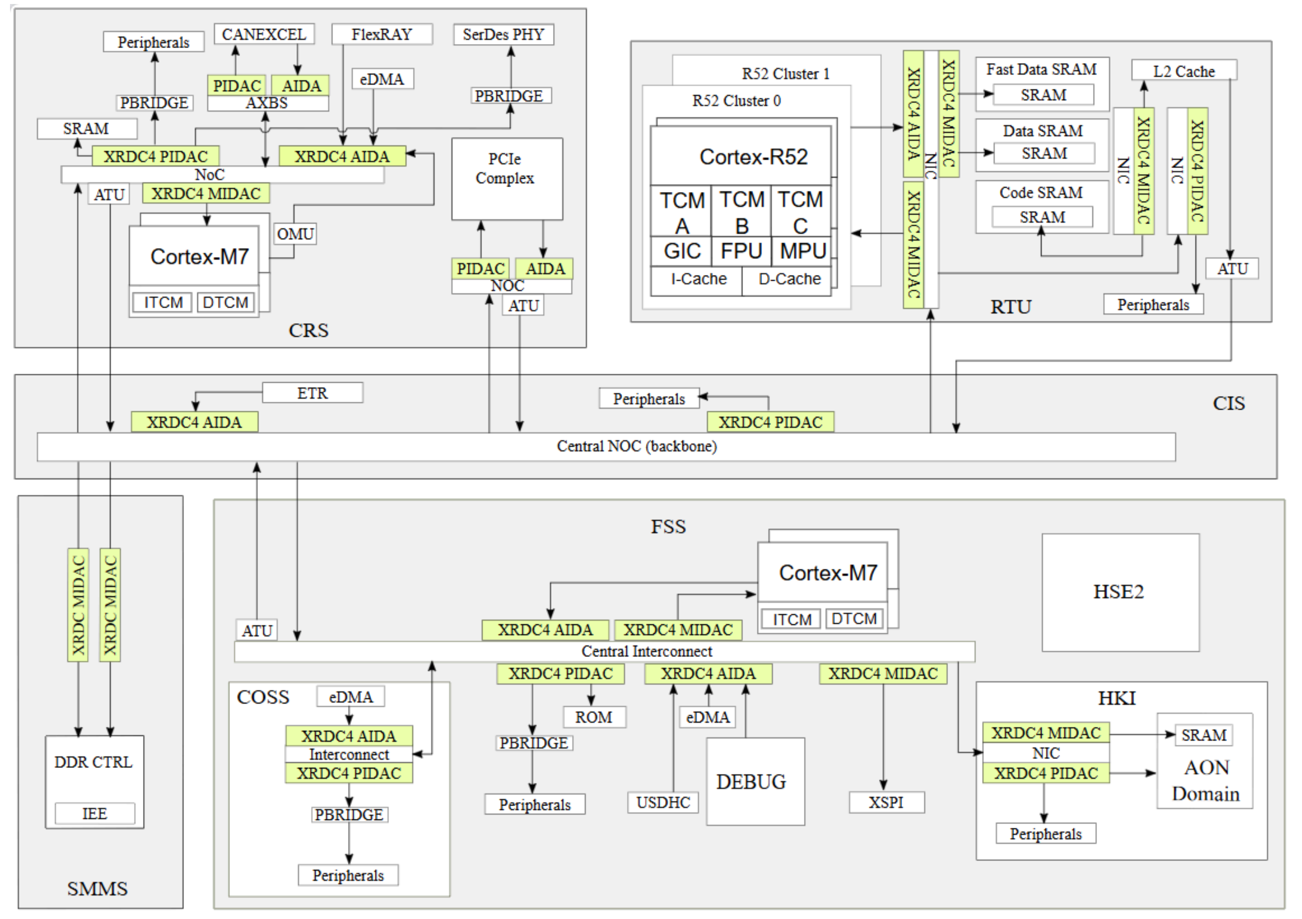


Figure 2. Subsystem diagram

2.3 Software components

Each cohort has its own Cohort Manager, allowing different vendors to independently develop or configure their respective managers. Within a cohort, each UENV possesses its own dedicated set of resources. Resource sharing among UENVs in the same cohort is managed by the software logic implemented in the Cohort Manager.

2.3.1 Partition manager

The Partition Manager is a chip-level function executed by software within the System Base Cohort (SBC). It plays a central role in defining and allocating system resources across all other cohorts. As the first software component to run during chip boot, the Partition Manager is responsible for resource allocation in the following aspects:

- Dividing system resources such as memory, peripherals, and processing units.
- Assigning access rights to each cohort.
- Managing XRDC4 Cohort IDs (CIDs) to enforce hardware-level access control.

The Partition Manager has full access to the configuration of all XRDC4 components, enabling it to completely define the resource boundaries of each cohort. This centralized control is especially effective in scenarios where:

- A cohort contains only one UENV.
- Multiple cohorts are owned by the same system developer who also controls the Partition Manager.

In the system architecture, the Partition Manager role is implemented by the FSS firmware, working in coordination with HSE2, to partition system resources into well-defined cohorts. The project Firmware_S32N55_FSS always assumes the role of PM. At the same time, it also takes on the role of CM for the SBC, which means all SBC configurations of XRDC can be performed within this project.

2.3.2 Cohort manager

A cohort manager is software that manages resource sharing within a single cohort. The cohort manager is loaded onto a core within a cohort by the partition manager. The cohort manager should do:

- Configures the XRDC4 components within a cohort. It assigns and manages DIDs, thereby allocating cohort peripherals to UENVs within the cohort.
- Loads, runs, verifies, and manages images for UENVs in the cohort.

2.3.3 Hypervisor

A hypervisor is an optional software component within a UENV (Unified Embedded Virtualization Environment) that is responsible for creating, scheduling, and isolating virtual machines (VMs).

To perform these tasks effectively, the hypervisor relies on several hardware mechanisms:

- Privilege level. Processor cores that support virtualization typically implement a dedicated privilege level specifically for hypervisor execution. In ARM architectures, these are defined as Exception Levels (ELx), where x denotes the level of privilege. For example, in the ARMv8-R architecture—such as the Cortex-R52 used in Real-Time Units (RTUs)—the hypervisor operates at EL2, which grants elevated access to system control resources required for managing virtual machines. This privilege level is not available on cores that lack hardware virtualization support, thereby restricting their ability to host hypervisors.
- Isolation mechanism(s). The ARMv8-R architecture introduces a second-stage Memory Protection Unit (MPU) to regulate core access to normal memory regions. However, device memory typically demands more granular access control than what the MPU can offer. To address this limitation, the Cortex-R52 associates a Virtual Machine Identifier (VMID) with memory transactions involving device memory, enabling VM-level differentiation. Furthermore, the eXtended Resource Domain Controller (XRDC4) functions as a hardware firewall, enforcing isolation between virtual machines executing on the same core. It also provides domain-based partitioning of core resources, ensuring isolation both within a single cohort and across multiple cohorts.

2.3.4 EL2 monitor

An EL2 monitor is a lightweight hypervisor implementation based on the Armv8-R architecture. It enforces a strict one-to-one mapping between each physical core and its corresponding virtual core as presented by the monitor. This design simplifies virtualization overhead and ensures deterministic behavior, which is critical in real-time embedded systems.

2.3.5 Virtual machine

A virtual machine is an abstract computing environment instantiated and managed by a hypervisor. The resources available to a VM may consist of virtualized components—emulated entirely in software—or physical hardware resources explicitly assigned to the VM.

The software running inside a VM can range from full-featured operating systems to bare-metal applications. From the perspective of the software within the VM, the system appears as a unified hardware platform, comprising both physical and virtual resources, seamlessly integrated by the hypervisor.

2.3.6 SW resources allocation

This section introduces the usage of the SW Resources allocation tab in the module Fss_Rem_Pm. These configurations are related to the clock gate(CG) and the software reset domain(SRD).

Each resource must be assigned to a single cohort as either an Owner or a User. A resource can have at most one Owner, and it will be released when its owner cohort starts up normally.

Note: *The chip will reset when accessing resources that have not been properly deasserted.*

2.4 Hardware components

2.4.1 AIDA

This XRDC is located outside the bus mastering interface of the chip IPs. It receives an identity from the bus master and translates it into an outbound AID, which is then used by the target XRDCs to determine whether access should be granted. The following diagram shows a simple process of accessing resources:

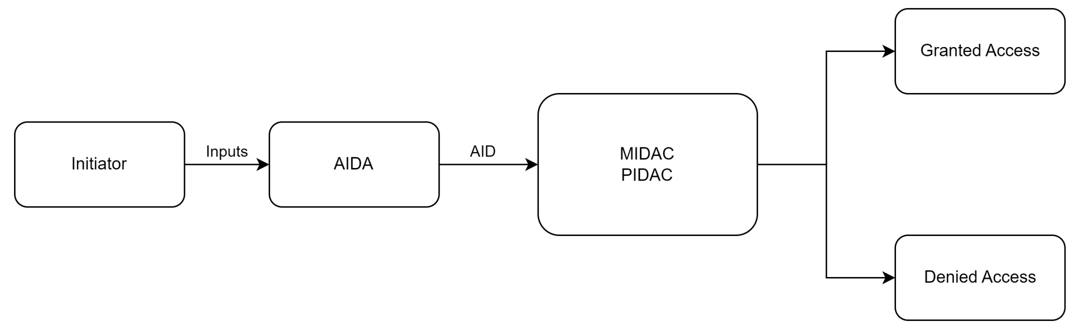


Figure 3. Resource accessing

The AID comprises two components: CID and DID, where CID denotes the Cohort ID and DID denotes the Domain ID. Certain AID combinations carry special meanings or behaviors, as described below:

- AID 0.0: Invalid as an access identifier.
- AID 0.1: Represents the partition manager within the system base cohort (CID = 0).
- AID 0.x: Located in the SBC, associated with FSS and HSE.
- AID x.1: Denotes the cohort manager for cohort x, where x ≠ 0.
- AID x.0: Valid and has no special purpose, where x ≠ 0.

There are two types of AIDA: AIDA_CR(Cortex-R) and AIDA_AD(Accelerators and DMA).

- AIDA_CR is optimized for processors that typically provide a VMID, NS, and PRIV attribute with each bus transaction. For every transaction issued by the manager, AIDA_CR generates an AID, which is a concatenation of the CID and DID. In addition to the AID, AIDA_CR also produces auxiliary bits, divided into

CAUX and DAUX. While the XRDC itself does not use these auxiliary bits, they may be useful to other system components.

- AIDA_AD, on the other hand, selects CID, DID, CAUX, and DAUX for each transaction based on a set of programmable entries. It uses a direct index selection method, where the index is derived from transaction attributes carried on the user sideband of the system bus. The specific attributes used as the index depend on the function of the manager agent. For example, an AIDA_AD assigned to a DMA manager might use the DMA channel number as the index selector. Therefore, the DMA channel number must be included in the transaction's user sideband. A compile-time parameter determines which bits of the user sideband are used as the index for AIDA_AD.

The inputs to RTU.AIDA_CR come from the VMID, NS, and PRIV attributes, which come from the ARM mechanism.

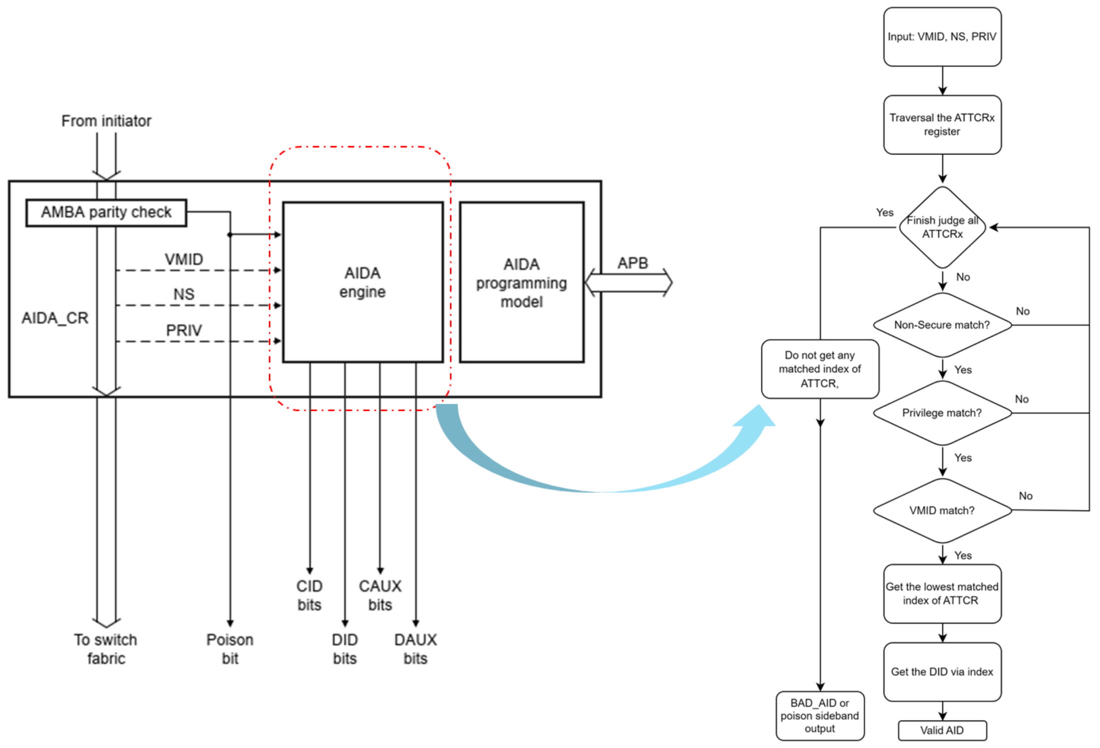


Figure 4. AIDA engine diagram

However, due to architectural limitations, the FSS core and CRS core do not support the aforementioned registers. As a result, they are driven in an alternative manner to allow optional usage and to align with the attributes of the incoming transactions:

Table 1. Inputs of FSS

Bit position	Description
[4:0]	Driven by MCM's Task ID(PID) register
[5]	Driven by CM7's ARMASTER field (which can be used to distinguish between CM7 transactions from the processor versus from its debug port)
[6]	Read: Reserved Write: Driven by a signal derived from the CM7's AWID field: • 1'b0: Transaction is not the result of an eviction

Table 1. Inputs of FSS*...continued*

Bit position	Description
	• 1’b1: Transaction is the result of an eviction
[7]	Driven by the FRB GPO[7] signal, which is configured by secure firmware, allowing it to optionally impact the selection of an output AID

The inputs for AIDA_AD differ from those of AIDA_CR. AIDA_AD uses a direct index selection method within the user sideband. For example, DMA modules typically use their channel number as the index, while NETC modules use their LDID.

More details can be found in the corresponding chapter of the Reference Manual. Below is a sample input for FSS.eDMA:

Table 2. Inputs of FSS.eDMA

Bit position	Description
[4:0]	Driven by FSS.eDMA channel number
[5]	Driven by the FRB GPO[8] signal, which is configured by secure firmware, allowing it to optionally impact the selection of an output AID
[6]	Strapped to 1’b0

The FSS.COSS.eDMA inputs:

Table 3. Inputs of FSS.COSS.eDMA

Bit position	Description
[0]	Driven by internal logic that is set by software, using each channel for both send and receive. This is unique for each FSS.COSS.eDMA AIDA AD instance.
[5:1]	Driven by FSS.COSS.eDMA(0-6) channel number
[6]	Reserved

The CRS.NETC inputs:

Table 4. Inputs of CRS.NETC

Function	Instance	Assigned LDID
EMDIO	0	2
Timer	0	1
Switch	0	3
ENETC	0	4
VSI	0	5
	1	6
	2	7
	3	8
	4	9
	5	10
	6	11

The following table lists the AIDA instances implemented on this chip:

Table 5. AIDA Instance

Subsystem	Category	Module instance	Master
FSS	AIDA_AD	XRDC_AIDA_AD1	eSTAM
		XRDC_AIDA_AD2	CSSI
		XRDC_AIDA_AD3	eDMA
		XRDC_AIDA_AD4	Usdhc
		XRDC_AIDA_AD5	APBIC_Debug
		XRDC_AIDA_AD7	COSS.eDMA0
		XRDC_AIDA_AD8	COSS.eDMA1
		XRDC_AIDA_AD9	COSS.eDMA2
		XRDC_AIDA_AD10	COSS.eDMA3
		XRDC_AIDA_AD11	COSS.eDMA4
		XRDC_AIDA_AD12	COSS.eDMA5
		XRDC_AIDA_AD13	COSS.eDMA6
	AIDA_CR	XRDC_AIDA_CR	FSS_CM7
CRS	AIDA_AD	-	AHB_CANXL_0
			AHB_CANXL_1
			AHB_CANXL_2
			AHB_CANXL_3
			AHB_FlexRay
			AHB_ACCEL_0
			AHB_ACCEL_1
			AXI_CAAM
			AXI_DMA_0
			AXI_DMA_1
			ACE_PCIe_Inbound
			ACE_NETC
	AIDA_CR		CRS_AXI_CM7
RTU.x	AIDA_CR	XRDC_AIDA_CR0	AXIM_Cluster0_Core0
		XRDC_AIDA_CR1	AXIM_Cluster1_Core0
		XRDC_AIDA_CR2	AXIM_Cluster0_Core1
		XRDC_AIDA_CR3	AXIM_Cluster1_Core1
		XRDC_AIDA_CR4	AXIF_Cluster0_Core0
		XRDC_AIDA_CR5	AXIF_Cluster1_Core0
		XRDC_AIDA_CR6	AXIF_Cluster0_Core1
		XRDC_AIDA_CR7	AXIF_Cluster1_Core1
		XRDC_AIDA_CR8	ETR
CIS	AIDA_AD		Debug ETR

2.4.2 PIDAC

PIDACs are positioned in front of peripheral targets. They receive the AID associated with each transaction and determine whether to allow or block access to the corresponding peripheral resource.

These components are part of the target-side XRDC4, which manages access control for peripherals and fixed-size regions of the address space.

Typically, PIDACs control access to groups of register banks—such as a set of peripherals sharing the same bus—but they can also manage access to memory regions, provided the fixed-size allocation is sufficient for all intended use cases.

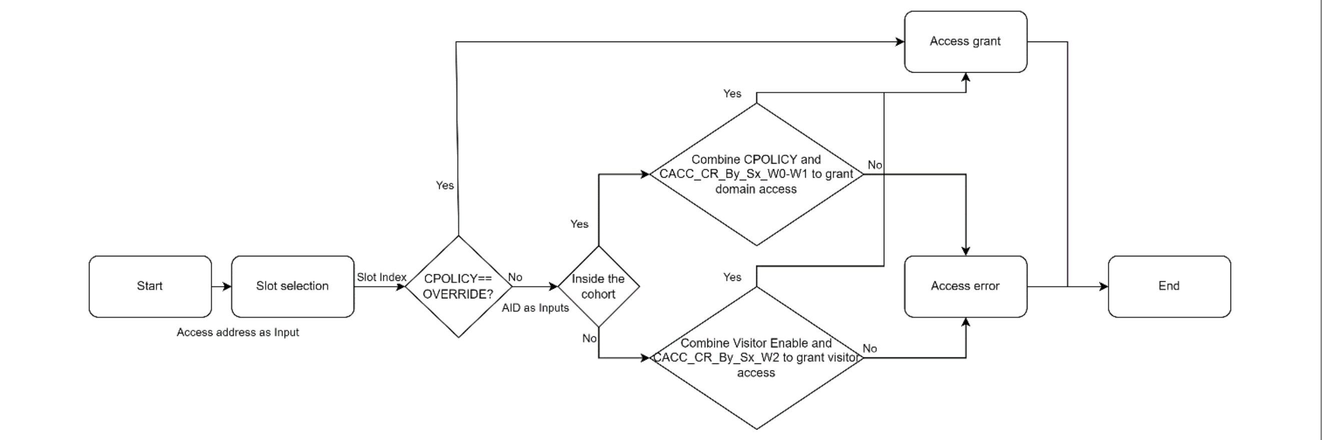


Figure 5. Access process of PIDAC

The cohort policy is an attribute of PIDAC and MIDAC that governs the initial gate of access control. The available options are:

- **NO_OVERRIDE**: Transactions are subject to the read and write permission vectors, which are configured via slot descriptor registers.
- **OVERRIDE**: Allows both read and write access from any entity.
- **READ_ONLY_OVERRIDE**: Read operations are subject to the read permission vector. Write operations are strictly prohibited, regardless of the write permission vector settings.

Here are some tips for the configuration of PIDAC:

- If a peripheral is accessible for both read and write operations by all requestors, the OCIDR.CPOLICY can simply be configured as OVERRIDE.
- When the PM writes a nonzero value to OCIDR.OCID, the corresponding CACC_CR_By_Sx_W0-1 is automatically asserted to all 1s by hardware. This indicates that all domains within the cohort are granted read and write access to this slot.
- If a slot needs to be accessed by a visitor, the OCIDR.NV register must be deasserted by the PM, and the CACC_CR_By_Sx_W2.VEN must be asserted by the CM.

Each PIDAC supports an Access Violation Alarm to help identify denied access attempts from initiators. This feature can be enabled via the AVEN register. When any access is denied, detailed violation information will be recorded. The table below lists the details, including:

Table 6. Details of Access Violation Alarm

Register	Bit	Description
AVEN	-	Access Violation Alarm Enable
AVADDR_HI	-	Access Violation Error Address High Range

Table 6. Details of Access Violation Alarm...*continued*

Register	Bit	Description
AVADDR_LO	-	Access Violation Error Address Low Range
AVATTR	0-AVSTAT	When reading 0b - Not captured 1b - Captured When writing 0b - No effect 1b - Clear the flag
	1-AVPRIV	0b - Violated unprivileged transaction 1b - Violated privileged transaction
	2-AVNS	0b - Violated secure transaction 1b - Violated nonsecure transaction
	3-AVOF	Access Violation Opcode Fetch 0b - Data access 1b - Opcode fetch
	5-AVMTX	1b - Mutex ownership denial
	11~6-AVDID	Access Violation Domain Identifier
	15~12-AVCID	Access Violation Cohort Identifier
	23~16-AVMST_ID	Access Violation Manager ID
	24-AVRW	Access Violation Read/Write 0b - Write transaction 1b - Read transaction
	29-AVI	Access Violation Inter Cohort

The following section outlines the PIDAC instances for all subsystems on the chip.

Table 7. PIDAC Instance

Subsystem	Instance	Module instance	Description
FSS	PIDAC0	XRDC_PIDAC_PERIPH_IPS1	FSS.Pbridge0
	PIDAC1	XRDC_PIDAC_PERIPH_IPS2	FSS.Pbridge1
	PIDAC2	XRDC_PIDAC_PERIPH_IPS3	FSS.Pbridge2
	PIDAC3	XRDC_PIDAC_AXI1	FSS.PRB
	PIDAC5	XRDC_PIDAC_AHB2	FSS.CSSI
	PIDAC6	XRDC_PIDAC_PERIPH_APB1	FSS.Dbg_APBIC
	PIDAC8	XRDC_PIDAC_AHB3	FSS.FTCM
	PIDAC9	XRDC_PIDAC_AHB1	FSS.SDB_NIC
	PIDAC11	XRDC_PIDAC_PERIPH_APB2	FSS.Pbridge2_APB
	PIDAC12	XRDC_PIDAC_PERIPH_IPS4	FSS.COSS.pbridge0
	PIDAC13	XRDC_PIDAC_PERIPH_IPS5	FSS.COSS.pbridge1
	PIDAC14	XRDC_PIDAC_PERIPH_IPS6	FSS.COSS.pbridge2
	PIDAC15	XRDC_PIDAC_PERIPH_IPS7	FSS.COSS.pbridge3

Table 7. PIDAC Instance*...continued*

Subsystem	Instance	Module instance	Description
	PIDAC16	XRDC_PIDAC_PERIPH_IPS9	FSS.HKI.Periph_IO
	PIDAC17	XRDC_PIDAC_PERIPH_IPS10	FSS.HKI.Periph.PM2
	PIDAC18	XRDC_PIDAC_PERIPH_IPS8	FSS.HKI.Periph_AON
	PIDAC19	XRDC_PIDAC_AHB4	FSS.HKI.AONSRAM
CRS	PIDAC0	XRDC_PIDAC_IPS_0	LIN, DMA0, FlexCAN
	PIDAC1	XRDC_PIDAC_IPS_1	LIN, DMA1, FlexCAN
	PIDAC2	XRDC_PIDAC_IPS_2	CANHUB, FlexRAY
	PIDAC3	XRDC_PIDAC_PCIe_APB	
	PIDAC4	XRDC_PIDAC_IPS_MAIN	SWT, vGPIO, MRU
	PIDAC5	XRDC_PIDAC_AHB_CANXL_0	
	PIDAC6	XRDC_PIDAC_AHB_CANXL_1	
	PIDAC7	XRDC_PIDAC_AHB_CANXL_2	
	PIDAC8	XRDC_PIDAC_AHB_CANXL_3	
	PIDAC9	XRDC_PIDAC_AES_ACCEL	
RTU	PIDAC0	PBRIDGE2_0	SWT, MRU, SRAMCTL_Cx
	PIDAC1	PBRIDGE2_1	L_VFCCU, OMU, MRU, SRAMCTL_Dx
	PIDAC2	PBRIDGE2_2	MRU
	PIDAC3	XRDC4_PIDAC_AHBrtuf	RTUF_NIC_D
	PIDAC4	XRDC4_PIDAC_AHBrtum	RTUM_NIC_D
	PIDAC5	XRDC4_PIDAC_AHBrtup	RTUP_NIC_B
SMMS	PIDAC0	AIPS_0	DDRC, CMU_FC, L_VFCCU
	PIDAC1	AIPS_1	DDR PHY
CIS	PIDAC0	SS Pbridge	IRQSTR, L_VFCCU

2.4.3 MIDAC

MIDACs are positioned in front of large memory spaces. Specifically, the NETC is protected by MIDAC. They receive the AID associated with each transaction and determine whether to allow or block access to the corresponding memory range.

Configuring the protection rules and defining the address regions managed by each XRDC4_MIDAC slot involves two key parts:

- As the PM role, enable memory regions for cohorts and assign slots to the corresponding cohorts.
- As the CM role, further configure the slot attributes and define access permissions for each slot.

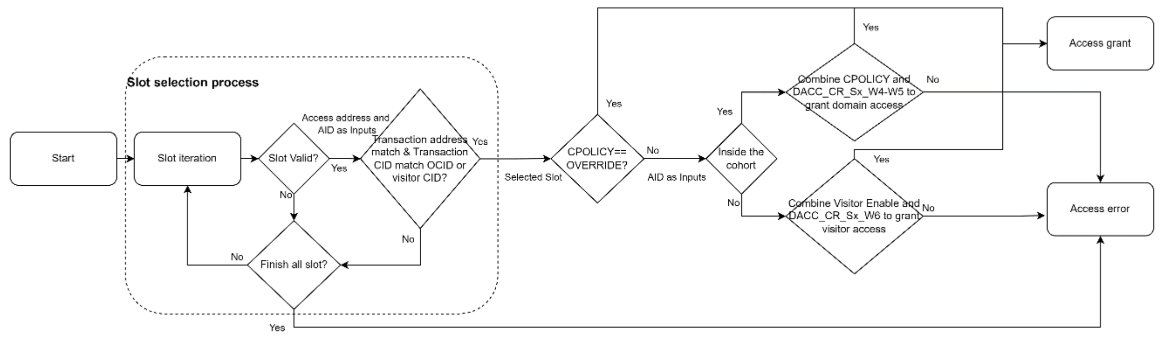

Start
Slot selection process
Slot iteration
Slot Valid?
Access address and AID as Inputs
Yes
No
Transaction address match & Transaction CID match OCID or visitor CID?
Yes
No
Finish all slot?
No
Yes
Selected Slot
CPOLICY== OVERRIDE?
Yes
No
AID as Inputs
Inside the cohort
Yes
No
Combine CPOLICY and DACC_CR_Sx_W4-W5 to grant domain access
Yes
No
Combine Visitor Enable and DACC_CR_Sx_W6 to grant visitor access
Yes
No
Access grant
Access error

Figure 6. Access process of MIDAC

Only the PM is permitted to update region-related registers, which include address ranges, cohort enable, cohort policy, and the no-visitor setting. The PM also manages the OCID register, which defines the owner cohort ID associated with each slot. Conversely, only the CM can update slot-related registers, which contain address ranges, region selection, domain access vectors, and visitor access permissions. However, certain slot registers are subject to automatic updates.

Each MIDAC instance supports monitoring access violations and modifying alarms, which can be enabled by setting the AVEN and MA_CR bits. The access violation monitor helps users identify detailed information about violations, including the address, AID, and the type of operation (read or write), please find more details in PIDAC chapter or refer to the RM.

The slot registers are automatically updated when the PM writes a nonzero value to the CIDR (Cohort ID Entry Register). Each CIDR register contains entries for four slots.

For slot x, the corresponding OCID field is stored in CIDRm[OCID_S_Nk], where m = x ÷ 4 and k = x mod 4. Here, x is the slot index, and m is the index of the CIDR register.

For example, if the PM writes 0x2 to CIDR5_N3, it means that slot 23 (since 23 = 4 × 5 + 3) belongs to cohort 2.

Based on the above example, the automatic updates are listed:

- DACC_CR_S23_W4-5 will assert all bits to provide full R/W to all domains within the owning cohort.
- DACC_CR_S23_W1.CHRT_REGION_SEL will update with cohort ID, indicating that the slot is associated with region index equal to CID(CID=2). That is why the PM defaults configure a region entry with CID.
- DACC_CR_S23_W0-3 will update the address range with the region address, which is stored in the region-related register with index CID (update with value in CHRT_R2_UPPER and CHRT_R2_LOWER).
- DACC_CR_S23_W7.SVLD will be set by hardware when the region is enabled for the corresponding cohort(through CHRT_Ry_UPPER_ADDRL.CnEn). In case CHRT_R2_UPPER_ADDRL.C2En bit is 1h, hardware will assert DACC_CR_S23_W7.SVLD==1.

The following section outlines the MIDAC instances for all subsystems on the chip.

Table 8. MIDAC Instance

Subsystem	Instance	Module instance	Description
FSS	MIDAC0	XRDC_MIDAC_AXI	FSS.SCB_NIC
	MIDAC1	XRDC_MIDAC_AHB	FSS.XSPI
CRS	MIDAC0	XRDC_MIDAC_AXI_PCIe_Outbound	Accel, PCIe
	MIDAC1	XRDC_MIDAC_AXI_MainMem_0	CRS.SRAM0
	MIDAC2	XRDC_MIDAC_AXI_MainMem_1	CRS.SRAM1
	MIDAC3	XRDC_MIDAC_AHB_M7	CRS.ITCM, CRS.DTCM

Table 8. MIDAC Instance*...continued*

Subsystem	Instance	Module instance	Description
	MIDAC4	XRDC_MIDAC_AXI_NETC	NETC, PCIe
RTU	MIDAC0	XRDC4_MIDAC_D0	DRAM0
	MIDAC1	XRDC4_MIDAC_D1	DRAM1
	MIDAC2	XRDC4_MIDAC_Dx	DRAM2
	MIDAC3	XRDC4_MIDAC_TCM0	Cluster 0 TCM
	MIDAC4	XRDC4_MIDAC_CODE	OMU
	MIDAC5	XRDC4_MIDAC_TCM1	Cluster 1 TCM
	MIDAC6	XRDC4_MIDAC0	CRAM0
	MIDAC7	XRDC4_MIDAC1	CRAM1
	MIDAC8	XRDC4_MIDAC2	CRAM2
	MIDAC9	XRDC4_MIDAC3	CRAM3
	MIDAC10	XRDC4_MIDAC4	CRAM4
	MIDAC11	XRDC4_MIDAC5	CRAM5
	MIDAC12	XRDC4_MIDAC6	CRAM6
	MIDAC13	XRDC4_MIDAC_LLC	LLC
SMMS	MIDAC0	XRDC4_MIDAC	DDRC access

3 How to assign the AID for an initiator

3.1 How to assign the AID of AIDA_CR

In this section, we aim to demonstrate how to assign an AID to a core initiator. As mentioned, there are three parts of AIDA_CR in this chip.

The process will be explained step by step, with examples to illustrate the configuration.

3.1.1 CRS

The CRS subsystem has only one instance of AIDA_CR. Each instance supports up to 16 domain entries, meaning that a core can be assigned to a maximum of 16 domains with distinct matching configurations simultaneously.

The inputs of the AIDA engine for CRS are mentioned in the previous chapter. For each transaction, the AIDA engine uses an attribute match index selection method. It evaluates the index via a match between the masked transaction NS, PRIV, and VMID with each entry of NS, PRIV, and VMID programmed in the AIDA_CR programming model.

This means that when a bit mask is deasserted, the corresponding bit will be excluded from the matching process. For example, if the NS_MASK of an entry is enabled (i.e., the NS bit is ignored), then the entry will match attributes regardless of the NS value — both secure and non-secure accesses will hit the entry.

AIDA_CR uses the lowest matched index to select the corresponding DID and DAUX entries. Also, it is responsible for generating a BAD_AID or poison sideband output for transactions that contain a parity error, originate from a faulted cohort, or fail to match any entry. For example, if both ATTCR1 and ATTCR5 result in a match, ATTCR1 takes precedence. In this case, index 1h is used to select the corresponding DID and DAUX values from the registers DIDCR1 and DAUXCR1.

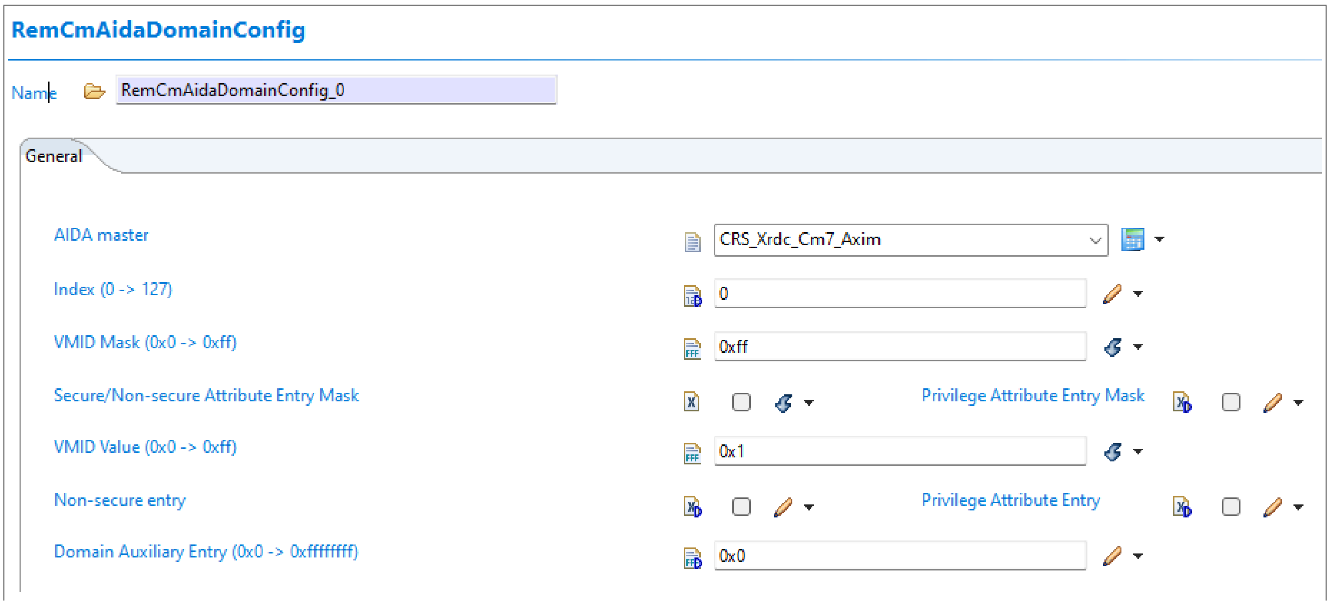
The following section outlines the configuration of CRS.AIDA_CR.

The goal is to:

- Add Domain 1 to enable access at the lowest hit level.
- Add Domain 2 to target transactions with VMID = 1.

Steps are listed:

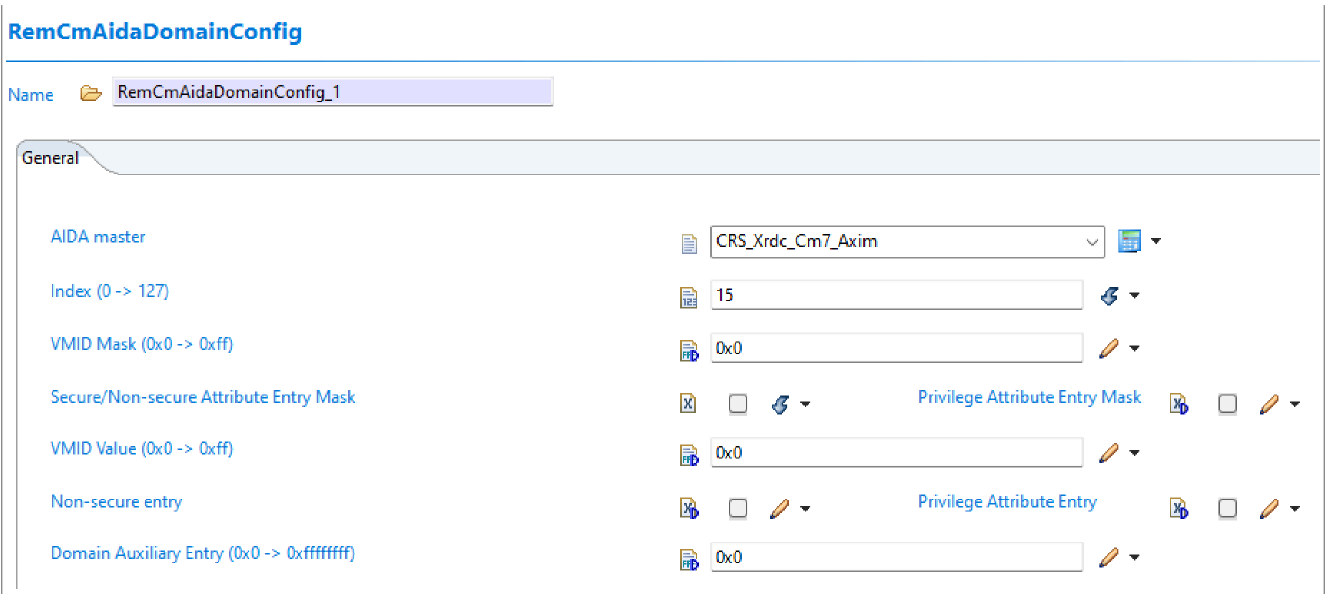
1. Ensure that CRS.AIDA_CR.XRDC_CM7_AXIM is assigned to Cohort 1 (CRS) in the FSS project.
2. Navigate to the Domain Config tab in the Rem_Cm module of the CRS project.
3. Add an entry for Domain 2.
4. Configure the match settings for Domain 2. Go to the RemCmAidaDomainConfig tab and add an entry with the following parameters:
   a. Index: 0
   b. VMID Mask: 0xFF
   c. Non-secure Mask: deasserted
   d. Privilege Mask: deasserted
   e. VMID Value: 0x1
5. Back to the Domain Config tab. Open the domain 1 entry.
6. Configure the match settings for Domain 3. Go to the RemCmAidaDomainConfig tab and add an entry with the following parameters:
   a. Index: 15
   b. VMID Mask: 0x0
   c. Non-secure Mask: deasserted
   d. Privilege Mask: deasserted


RemCmAidaDomainConfig

Name: RemCmAidaDomainConfig_0

General

AIDA master	CRS_Xrdc_Cm7_Axim
Index (0 -> 127)	0
VMID Mask (0x0 -> 0xff)	0xff
Secure/Non-secure Attribute Entry Mask	☐
Privilege Attribute Entry Mask	☐
VMID Value (0x0 -> 0xff)	0x1
Non-secure entry	☐
Privilege Attribute Entry	☐
Domain Auxiliary Entry (0x0 -> 0xffffffff)	0x0



RemCmAidaDomainConfig

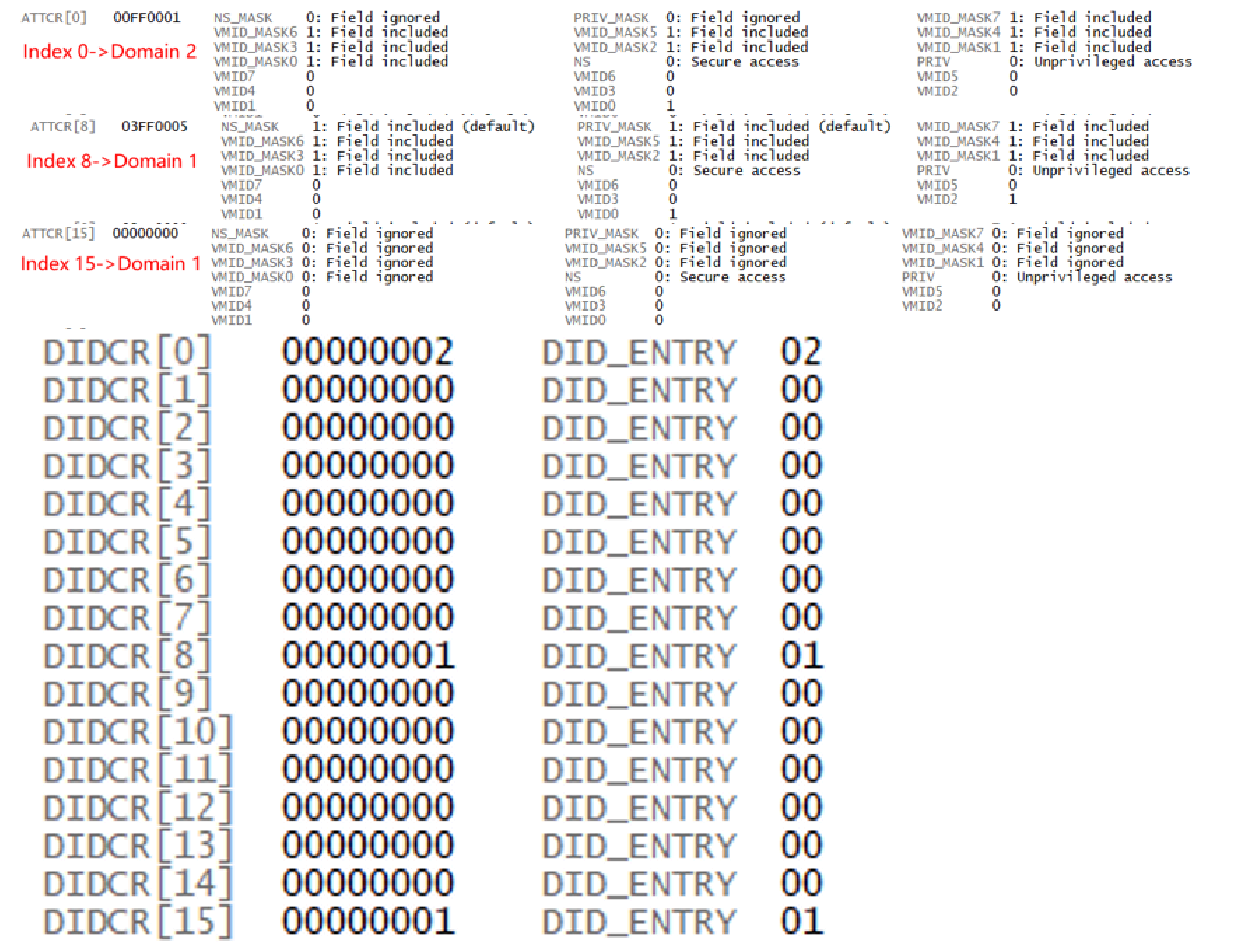
Name: RemCmAidaDomainConfig_1

General

AIDA master	CRS_Xrdc_Cm7_Axim
Index (0 -> 127)	15
VMID Mask (0x0 -> 0xff)	0x0
Secure/Non-secure Attribute Entry Mask	☐
Privilege Attribute Entry Mask	☐
VMID Value (0x0 -> 0xff)	0x0
Non-secure entry	☐
Privilege Attribute Entry	☐
Domain Auxiliary Entry (0x0 -> 0xffffffff)	0x0

Figure 7. Configuration for assigning the AID to CRS

Based on the configurations, a transaction will be assigned to a specific domain when its input attributes match a domain entry. If multiple entries match, the one with the lowest index takes precedence. Additionally, there is an entry with index 15. Even if all previous entries fail to match, this entry will still be hit. The transaction will be assigned DID 1 because this entry has all input masks enabled, allowing it to match any transaction.



```
ATTCR[0]    00FF0001   NS_MASK     0: Field ignored          PRIV_MASK   0: Field ignored            VMID_MASK7 1: Field included
Index 0->Domain 2      VMID_MASK6 1: Field included          VMID_MASK5 1: Field included            VMID_MASK4 1: Field included
                       VMID_MASK3 1: Field included          VMID_MASK2 1: Field included            VMID_MASK1 1: Field included
                       VMID_MASK0 1: Field included          NS          0: Secure access            PRIV       0: Unprivileged access
                       VMID7       0                         VMID6       0                           VMID5      0
                       VMID4       0                         VMID3       0                           VMID2      0
                       VMID1       0                         VMID0       1
ATTCR[8]    03FF0005   NS_MASK     1: Field included (default) PRIV_MASK 1: Field included (default) VMID_MASK7 1: Field included
Index 8->Domain 1      VMID_MASK6 1: Field included          VMID_MASK5 1: Field included            VMID_MASK4 1: Field included
                       VMID_MASK3 1: Field included          VMID_MASK2 1: Field included            VMID_MASK1 1: Field included
                       VMID_MASK0 1: Field included          NS          0: Secure access            PRIV       0: Unprivileged access
                       VMID7       0                         VMID6       0                           VMID5      0
                       VMID4       0                         VMID3       0                           VMID2      1
                       VMID1       0                         VMID0       1
ATTCR[15]   00000000   NS_MASK     0: Field ignored          PRIV_MASK   0: Field ignored            VMID_MASK7 0: Field ignored
Index 15->Domain 1     VMID_MASK6 0: Field ignored           VMID_MASK5 0: Field ignored             VMID_MASK4 0: Field ignored
                       VMID_MASK3 0: Field ignored           VMID_MASK2 0: Field ignored             VMID_MASK1 0: Field ignored
                       VMID_MASK0 0: Field ignored           NS          0: Secure access            PRIV       0: Unprivileged access
                       VMID7       0                         VMID6       0                           VMID5      0
                       VMID4       0                         VMID3       0                           VMID2      0
                       VMID1       0                         VMID0       0
DIDCR[0]    00000002   DID_ENTRY  02
DIDCR[1]    00000000   DID_ENTRY  00
DIDCR[2]    00000000   DID_ENTRY  00
DIDCR[3]    00000000   DID_ENTRY  00
DIDCR[4]    00000000   DID_ENTRY  00
DIDCR[5]    00000000   DID_ENTRY  00
DIDCR[6]    00000000   DID_ENTRY  00
DIDCR[7]    00000000   DID_ENTRY  00
DIDCR[8]    00000001   DID_ENTRY  01
DIDCR[9]    00000000   DID_ENTRY  00
DIDCR[10]   00000000   DID_ENTRY  00
DIDCR[11]   00000000   DID_ENTRY  00
DIDCR[12]   00000000   DID_ENTRY  00
DIDCR[13]   00000000   DID_ENTRY  00
DIDCR[14]   00000000   DID_ENTRY  00
DIDCR[15]   00000001   DID_ENTRY  01
```

Figure 8. Registers of CRS.AIDA_CR

As previously mentioned, the cohort manager uses AID x.1. Therefore, the XRDC can only be configured by an initiator with AID x.1. This requires that the XRDC configuration for the cohort manager include an entry with DID = 1 before the entire XRDC initialization process can be completed.

Note: *All entries for the Cohort Manager in the FSS project must be configured with DID = 1 to ensure successful XRDC configuration.*

3.1.2 RTU

Each RTU contains four R52 cores, which are distributed across two clusters. Each core includes two instances of AIDA_CR: AXIM and AXIF. According to ARM architecture restrictions, there are limitations on the VMID values of transactions.

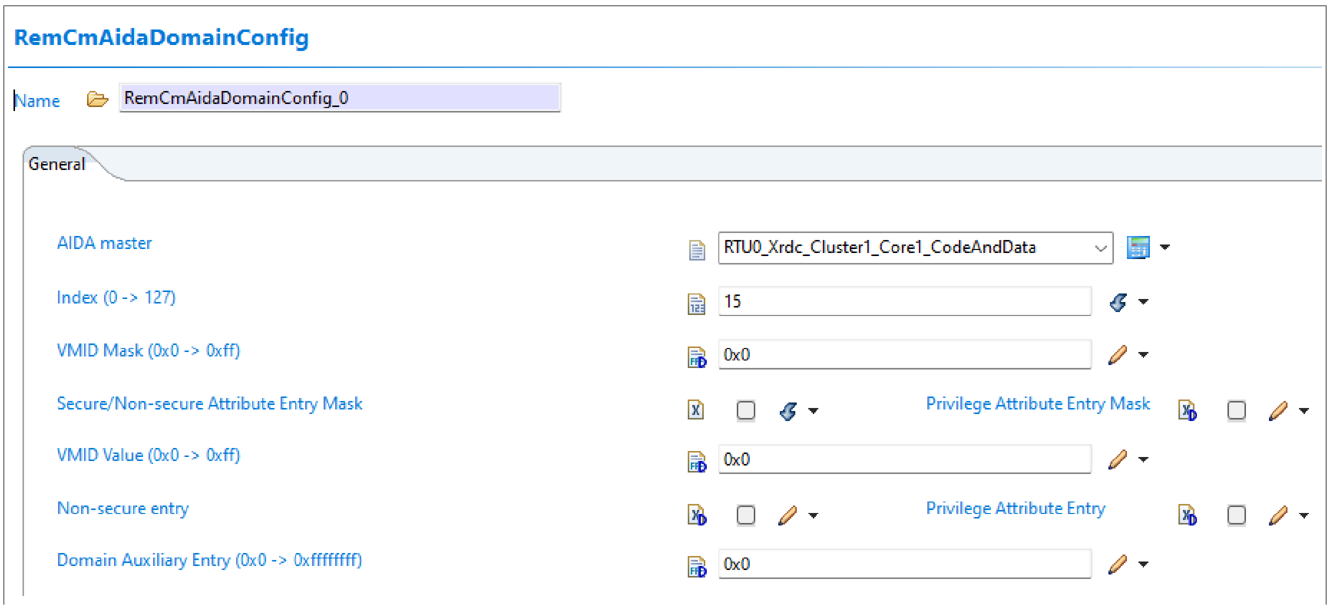
- For the AXIM interface, a valid VMID is only guaranteed for Device memory accesses. For Normal memory accesses, the VMID is set to 0.
- For the AXIF interface, the VMID is always set to 0.

This section focuses on the configuration of the AXIM instance.

The cohort manager of the RTU0_R52_3 AXIM (AIDA_CR3) instance is Cohort 2 (RTU0). Therefore, it allows RTU0_R52_0, which serves as the Cohort Manager of Cohort 2, to further configure RTU0.AIDA_CR3.

The AIDA_CR3 is expected to be assigned to Domain 1 and Domain 2 in this section. The steps are listed below:

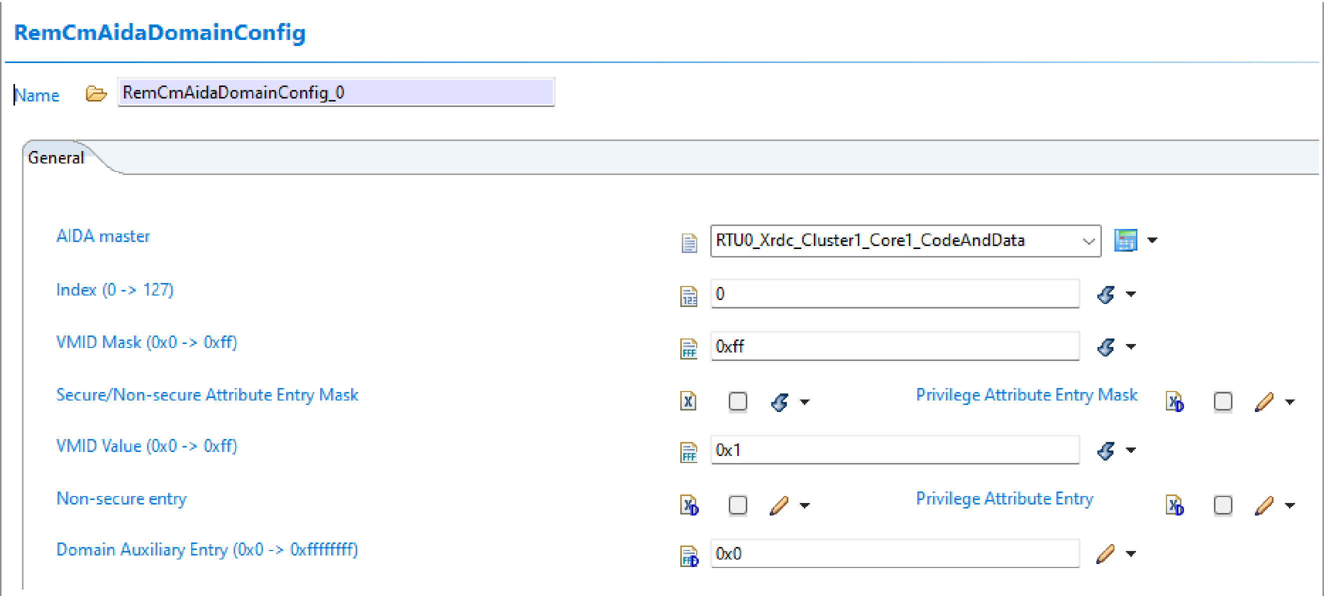
1. Navigate to the Domain Config tab under the module Rem_Cm in the project S32N5_RTU0_R52_0_VIP_App.
2. Assign AIDA_CR3 to Domain 1. Open the entry of XRDC4_DOMAIN1, add a new entry for RTU0_Xrdc_Cluster1_Core1_CodeAndData AIDA master under the RecmCmAidaDomainConfig tab. The parameters are listed:
   a. Index: 15
   b. VMID Mask: 0x0
   c. Non-Secure Mask: deassert
   d. Privilege Mask: deassert
3. The purpose of the previous entry is to assign AIDA_CR3 to Domain 1 when entries 0–14 are not matched. This ensures that the transaction is assigned to at least one domain, rather than being granted a BAD_AID.
4. Assign AIDA_CR3 to Domain 2. Navigate back to the Domain Config tab. Open the entry for XRDC4_DOMAIN2, and add a new entry for the RTU0_Xrdc_Cluster1_Core1_CodeAndData AIDA master with the following parameters:
   a. Index: 0
   b. VMID Mask: 0xFF
   c. Non-Secure Mask: deassert
   d. Privilege Mask: deassert
   e. VMID: 0x1
5. The purpose of the previous entry is to assign AIDA_CR3 to Domain 2 when the VMID is 0x1.


RemCmAidaDomainConfig

Name: RemCmAidaDomainConfig_0

General

AIDA master	RTU0_Xrdc_Cluster1_Core1_CodeAndData
Index (0 -> 127)	15
VMID Mask (0x0 -> 0xff)	0x0
Secure/Non-secure Attribute Entry Mask	☐
Privilege Attribute Entry Mask	☐
VMID Value (0x0 -> 0xff)	0x0
Non-secure entry	☐
Privilege Attribute Entry	☐
Domain Auxiliary Entry (0x0 -> 0xffffffff)	0x0



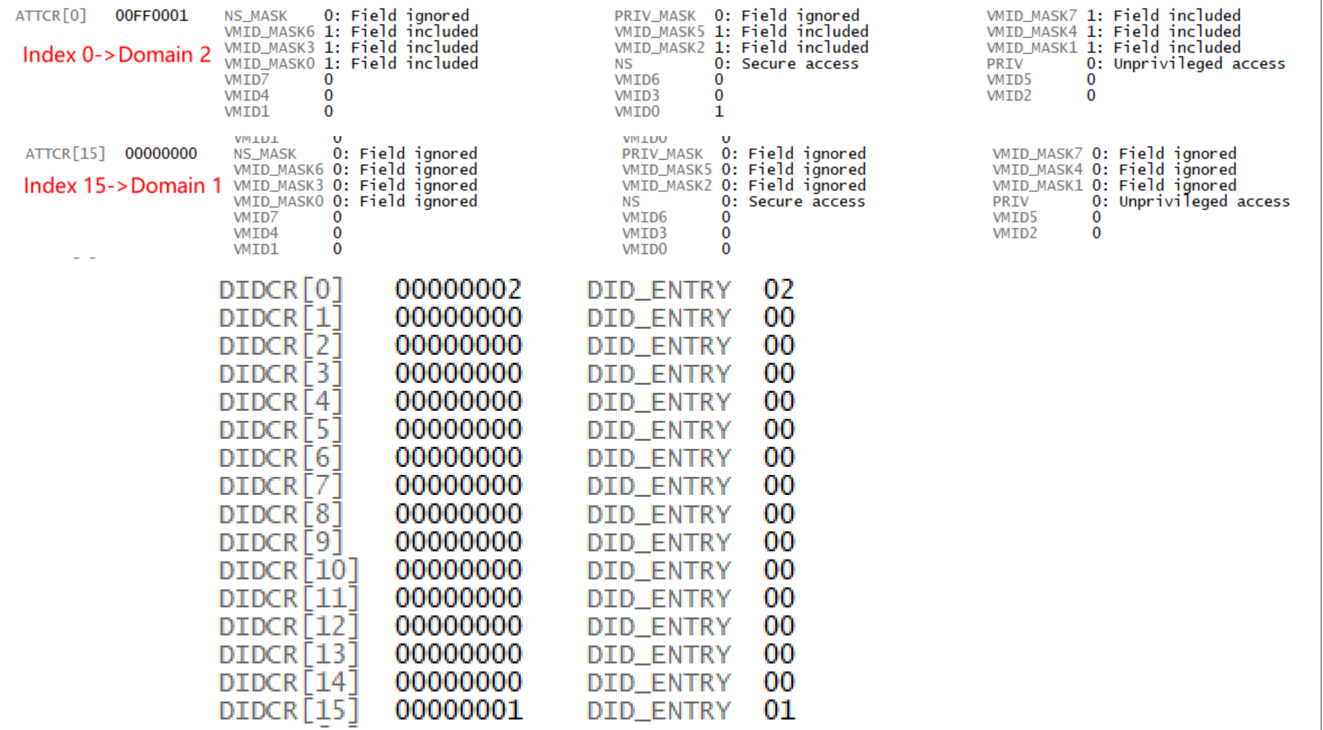
RemCmAidaDomainConfig

Name: RemCmAidaDomainConfig_0

General

AIDA master	RTU0_Xrdc_Cluster1_Core1_CodeAndData		
Index (0 -> 127)	0		
VMID Mask (0x0 -> 0xff)	0xff		
Secure/Non-secure Attribute Entry Mask	☐	Privilege Attribute Entry Mask	☐
VMID Value (0x0 -> 0xff)	0x1		
Non-secure entry	☐	Privilege Attribute Entry	☐
Domain Auxiliary Entry (0x0 -> 0xffffffff)	0x0		

Figure 9. Configuration for assigning the AID to RTU



```
ATTCR[0]    00FF0001   NS_MASK     0: Field ignored     PRIV_MASK   0: Field ignored     VMID_MASK7 1: Field included
                       VMID_MASK6  1: Field included    VMID_MASK5  1: Field included    VMID_MASK4 1: Field included
Index 0->Domain 2      VMID_MASK3  1: Field included    VMID_MASK2  1: Field included    VMID_MASK1 1: Field included
                       VMID_MASK0  1: Field included    NS          0: Secure access     PRIV       0: Unprivileged access
                       VMID7       0                    VMID6       0                    VMID5      0
                       VMID4       0                    VMID3       0                    VMID2      0
                       VMID1       0                    VMID0       1

ATTCR[15]   00000000   NS_MASK     0: Field ignored     PRIV_MASK   0: Field ignored     VMID_MASK7 0: Field ignored
                       VMID_MASK6  0: Field ignored     VMID_MASK5  0: Field ignored     VMID_MASK4 0: Field ignored
Index 15->Domain 1     VMID_MASK3  0: Field ignored     VMID_MASK2  0: Field ignored     VMID_MASK1 0: Field ignored
                       VMID_MASK0  0: Field ignored     NS          0: Secure access     PRIV       0: Unprivileged access
                       VMID7       0                    VMID6       0                    VMID5      0
                       VMID4       0                    VMID3       0                    VMID2      0
                       VMID1       0                    VMID0       0

DIDCR[0]    00000002   DID_ENTRY  02
DIDCR[1]    00000000   DID_ENTRY  00
DIDCR[2]    00000000   DID_ENTRY  00
DIDCR[3]    00000000   DID_ENTRY  00
DIDCR[4]    00000000   DID_ENTRY  00
DIDCR[5]    00000000   DID_ENTRY  00
DIDCR[6]    00000000   DID_ENTRY  00
DIDCR[7]    00000000   DID_ENTRY  00
DIDCR[8]    00000000   DID_ENTRY  00
DIDCR[9]    00000000   DID_ENTRY  00
DIDCR[10]   00000000   DID_ENTRY  00
DIDCR[11]   00000000   DID_ENTRY  00
DIDCR[12]   00000000   DID_ENTRY  00
DIDCR[13]   00000000   DID_ENTRY  00
DIDCR[14]   00000000   DID_ENTRY  00
DIDCR[15]   00000001   DID_ENTRY  01
```

Figure 10. Register of RTU0.AIDA_CR3

3.2 How to assign the AID of AIDA_AD

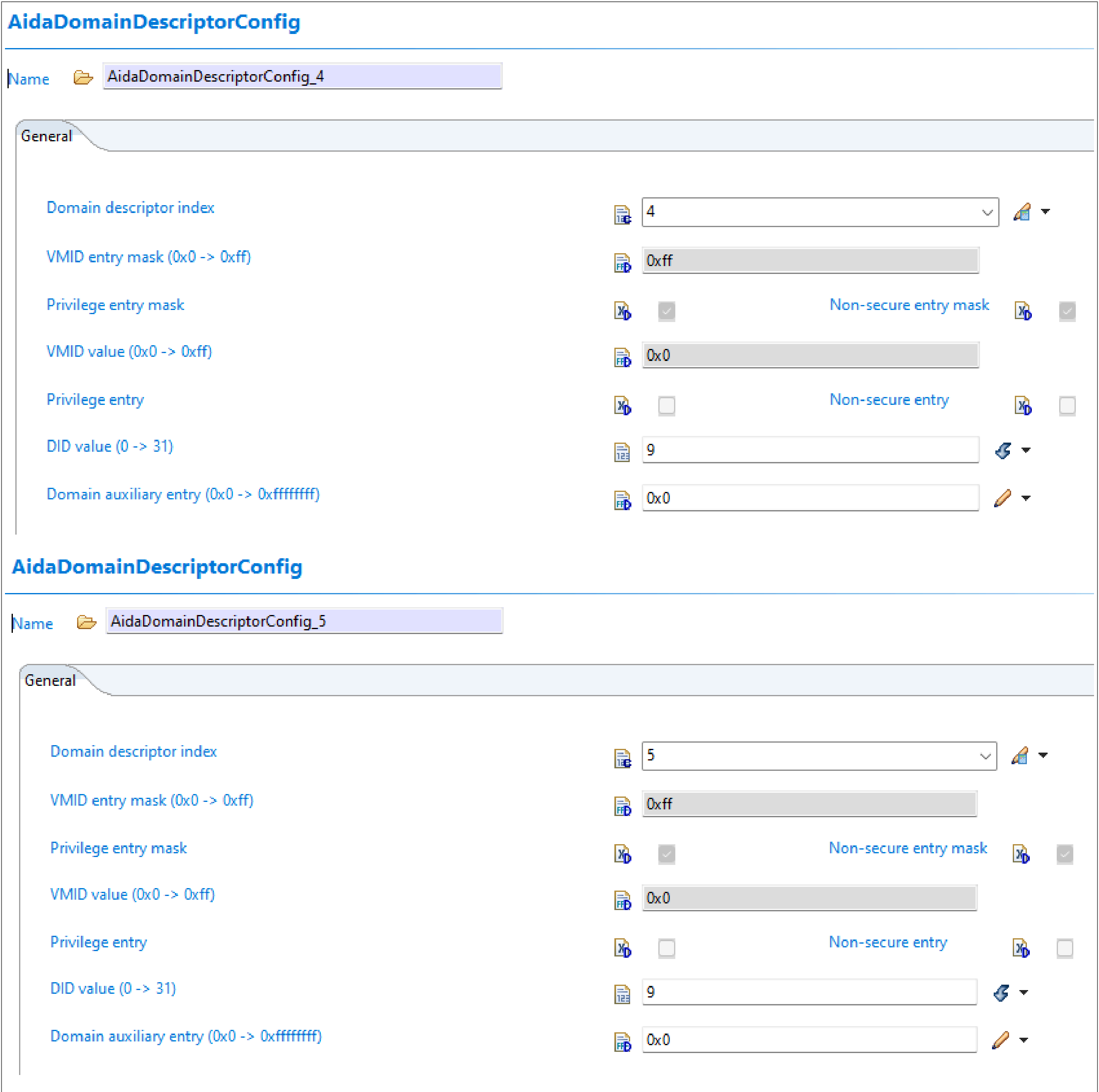
3.2.1 eDMA

This section aims to illustrate the domain assignment for FSS.COSS.eDMA.

As mentioned earlier, AIDA_AD uses a direct index selection method to hit the entry and retrieve the corresponding DID.

For example, FSS.XRDC_AIDA_AD7 (associated with FSS.COSS.eDMA0) supports up to 64 entries (indexed from 0 to 63) for both CIDR and DIDCR. This means each channel of FSS.COSS.eDMA0 is allocated two entries for CIDR and DIDCR, allowing each channel to be assigned to up to two different domains simultaneously under different scenarios.

There are two approaches to achieve this goal. One is to configure it directly within the FSS project. The other is to assign entry ownership to a specific cohort and configure its domain within the CM project. In the FSS project, the eDMA0.CH2 default be assigned to Domain 9.



AidaDomainDescriptorConfig

Name: AidaDomainDescriptorConfig_4

General

Domain descriptor index: 4
VMID entry mask (0x0 -> 0xff): 0xff
Privilege entry mask: ✓ Non-secure entry mask: ✓
VMID value (0x0 -> 0xff): 0x0
Privilege entry: ☐ Non-secure entry: ☐
DID value (0 -> 31): 9
Domain auxiliary entry (0x0 -> 0xffffffff): 0x0

AidaDomainDescriptorConfig

Name: AidaDomainDescriptorConfig_5

General

Domain descriptor index: 5
VMID entry mask (0x0 -> 0xff): 0xff
Privilege entry mask: ✓ Non-secure entry mask: ✓
VMID value (0x0 -> 0xff): 0x0
Privilege entry: ☐ Non-secure entry: ☐
DID value (0 -> 31): 9
Domain auxiliary entry (0x0 -> 0xffffffff): 0x0

Figure 11. Configuration of FSS.COSS.eDMA0 in FSS project

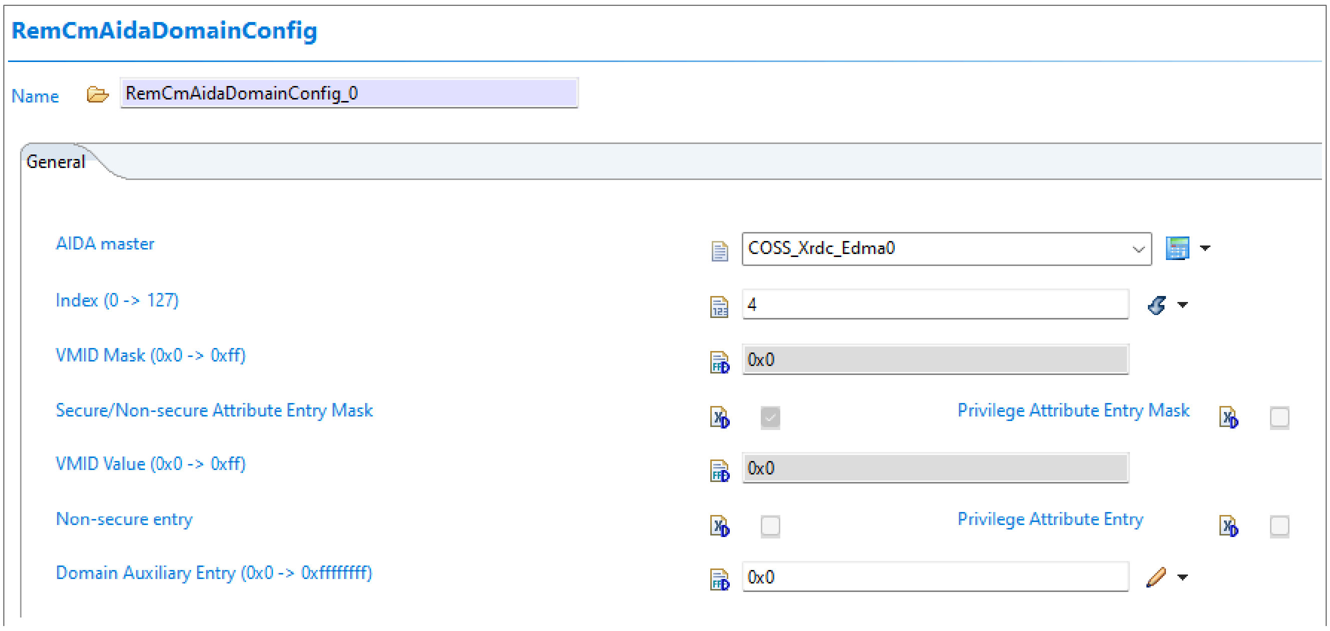
Based on the default configuration, FSS.COSS.eDMA0 is assigned to Cohort 2. The next step is to switch to Cohort 2 and further assign it to the appropriate domain.

The goal is to:

- Assign FSS.COSS.eDMA0.CH2 to Domain 5.
- Ensure FSS.COSS.eDMA0.CH2 grants access to CRS.SRAM1

Steps are listed:

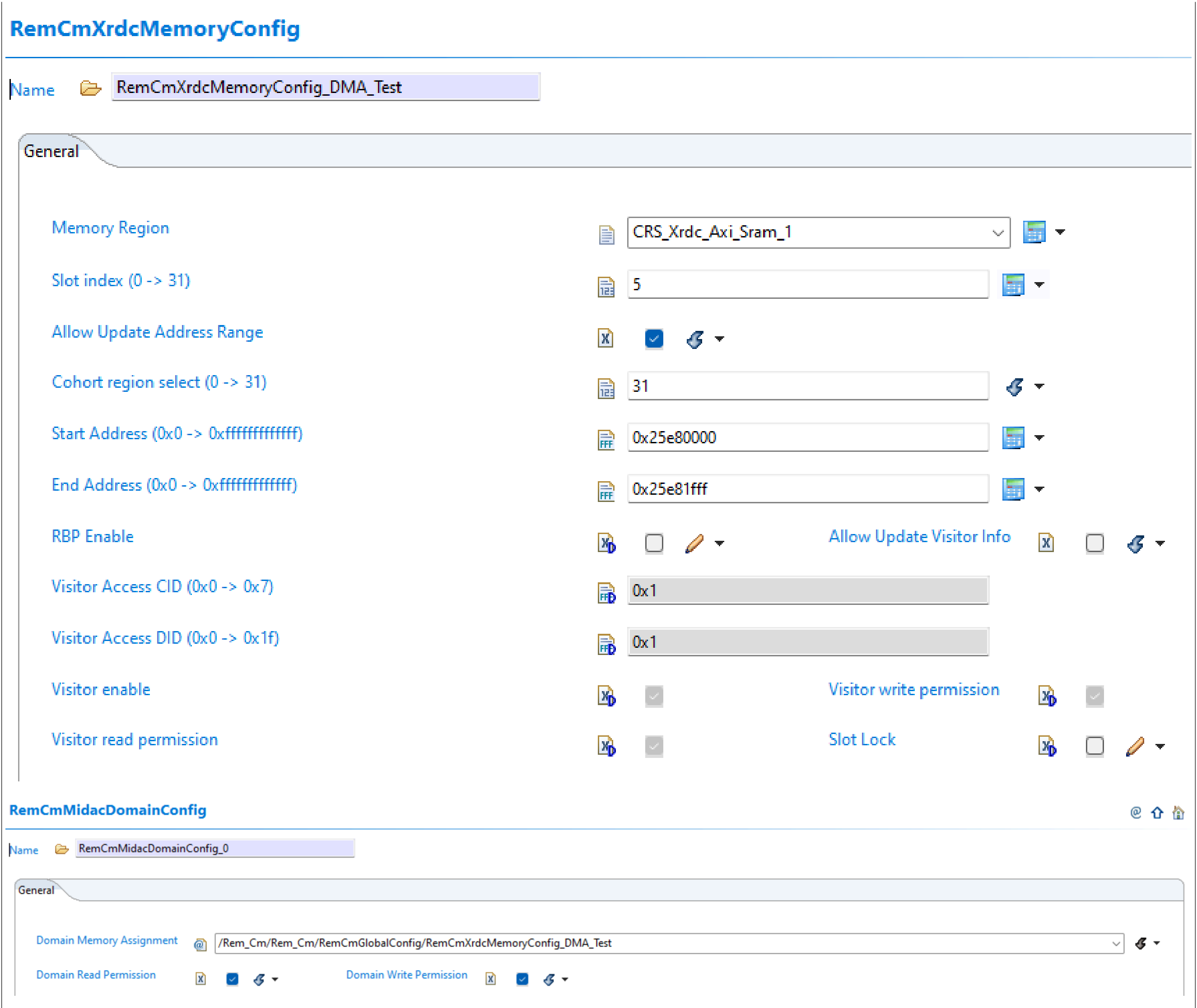
1. Navigate to the Domain Config tab under the module Rem_Cm in project S32N5_RTU0_R52_0_VIP_App.
2. Add an entry for Domain 5.
3. Configure the match settings for Domain 5. Go to the RemCmAidaDomainConfig tab and add an entry with the following parameters:
   a. AIDA master: COSS_XRDC_eDMA0
   b. Index: 4
4. Assign slot 5 of XRDC.CRS.SRAM1 to Cohort 2. Navigate to the tab Cohorts configuration. Add an entry with index 5 to Cohort 2, which is located at RTU0_AppCohort-> Subsystems configuration(Subsystem CRS)->MIDAC configuration(XRDC4_PIDAC2)->Slots configuration.
5. Configure the attributes of Slot 5 in XRDC.CRS.SRAM1. Return to the Rem_Cm module in the project S32N5_RTU0_R52_0_VIP_App, and navigate to the XRDC Memory Config tab. Add a new entry with the following parameters:
   a. Memory Region: CRS_XRDC_AXI_SRAM_1
   b. Slot Index: 5
   c. Allow Update Address Range: True
   d. Cohort Region select: 31
   e. Start Address: 0x25E8_0000
   f. End Address: 0x25E8_1FFF
6. Link the memory to Domain 5. Return to the configuration page of Domain 5, navigate to the RemCmMidacDomainConfig tab, and add an entry that links to the previously configured memory with R/W permissions.


RemCmAidaDomainConfig

Name: RemCmAidaDomainConfig_0

General

AIDA master	COSS_Xrdc_Edma0
Index (0 -> 127)	4
VMID Mask (0x0 -> 0xff)	0x0
Secure/Non-secure Attribute Entry Mask	✓
Privilege Attribute Entry Mask	☐
VMID Value (0x0 -> 0xff)	0x0
Non-secure entry	☐
Privilege Attribute Entry	☐
Domain Auxiliary Entry (0x0 -> 0xffffffff)	0x0



RemCmXrdcMemoryConfig

Name: RemCmXrdcMemoryConfig_DMA_Test

General

Memory Region	CRS_Xrdc_Axi_Sram_1
Slot index (0 -> 31)	5
Allow Update Address Range	✓
Cohort region select (0 -> 31)	31
Start Address (0x0 -> 0xffffffffffff)	0x25e80000
End Address (0x0 -> 0xffffffffffff)	0x25e81fff
RBP Enable	☐
Allow Update Visitor Info	☐
Visitor Access CID (0x0 -> 0x7)	0x1
Visitor Access DID (0x0 -> 0x1f)	0x1
Visitor enable	✓
Visitor write permission	✓
Visitor read permission	✓
Slot Lock	☐

RemCmMidacDomainConfig

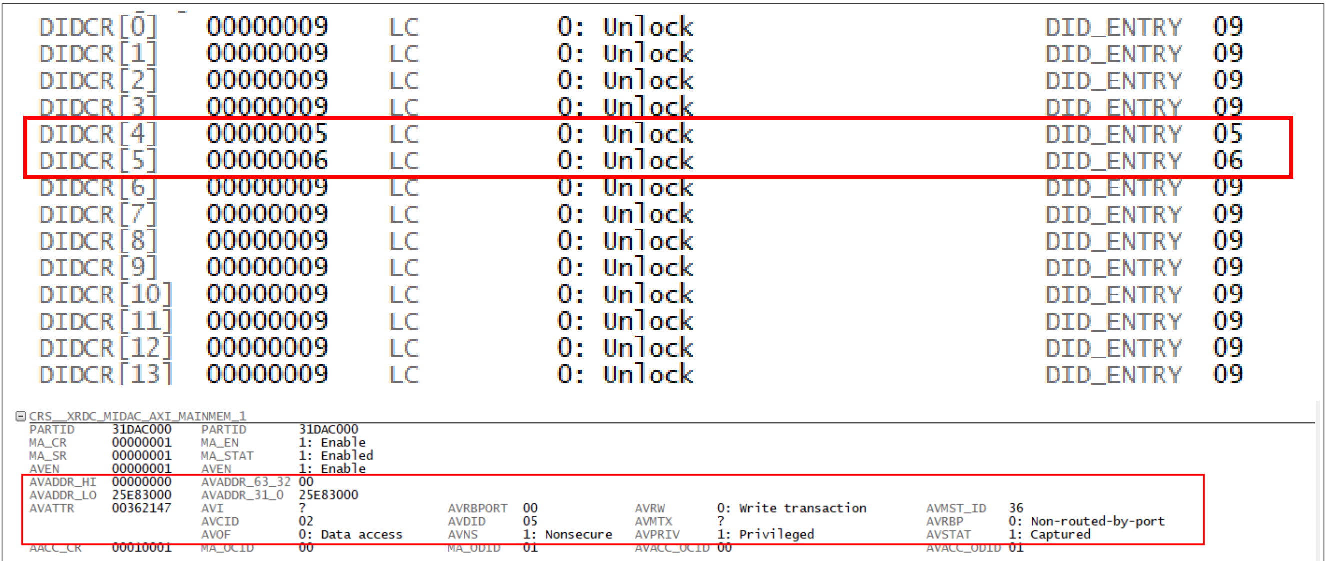
Name: RemCmMidacDomainConfig_0

General

Domain Memory Assignment	/Rem_Cm/Rem_Cm/RemCmGlobalConfig/RemCmXrdcMemoryConfig_DMA_Test
Domain Read Permission	✓
Domain Write Permission	✓

Figure 12. Configuration of FSS.COSS.eDMA0 in RTU0 project

Based on the previous configuration, FSS.COSS.eDMA0.CH2 is assigned to Domain 5, and Domain 5 is granted access to the address range 0x25E80_000~0x25E8_1FFF of CRS.SRAM1. For example, if the channel is triggered with a start address of 0x25E8_0000 and an end address of 0x25E8_1000, the transaction will complete successfully. However, if the channel is triggered with a start address of 0x25E8_0000 and an end address of 0x25E8_3000, the write operation will be denied. If the violation alarm is enabled, the violation attributes will be recorded accordingly.



```
DIDCR[0]   00000009   LC        0: Unlock          DID_ENTRY  09
DIDCR[1]   00000009   LC        0: Unlock          DID_ENTRY  09
DIDCR[2]   00000009   LC        0: Unlock          DID_ENTRY  09
DIDCR[3]   00000009   LC        0: Unlock          DID_ENTRY  09
DIDCR[4]   00000005   LC        0: Unlock          DID_ENTRY  05
DIDCR[5]   00000006   LC        0: Unlock          DID_ENTRY  06
DIDCR[6]   00000009   LC        0: Unlock          DID_ENTRY  09
DIDCR[7]   00000009   LC        0: Unlock          DID_ENTRY  09
DIDCR[8]   00000009   LC        0: Unlock          DID_ENTRY  09
DIDCR[9]   00000009   LC        0: Unlock          DID_ENTRY  09
DIDCR[10]  00000009   LC        0: Unlock          DID_ENTRY  09
DIDCR[11]  00000009   LC        0: Unlock          DID_ENTRY  09
DIDCR[12]  00000009   LC        0: Unlock          DID_ENTRY  09
DIDCR[13]  00000009   LC        0: Unlock          DID_ENTRY  09

CRS__XRDC_MIDAC_AXI_MAINMEM_1
 PARTID      31DAC000  PARTID        31DAC000
 MA_CR       00000001  MA_EN         1: Enable
 MA_SR       00000001  MA_STAT       1: Enabled
 AVEN        00000001  AVEN          1: Enable
 AVADDR_HI   00000000  AVADDR_63_32  00
 AVADDR_LO   25E83000  AVADDR_31_0   25E83000
 AVATTR      00362147  AVI           ?                AVRBPORT  00             AVRW        0: Write transaction   AVMST_ID   36
                       AVCID         02               AVDID     05             AVMTX       ?                      AVRBP      0: Non-routed-by-port
                       AVOF          0: Data access   AVNS      1: Nonsecure   AVPRIV      1: Privileged          AVSTAT     1: Captured
 AACC_CR     00010001  MA_OCID       00               MA_ODID   01             AVACC_OCID  00                     AVACC_ODID 01
```

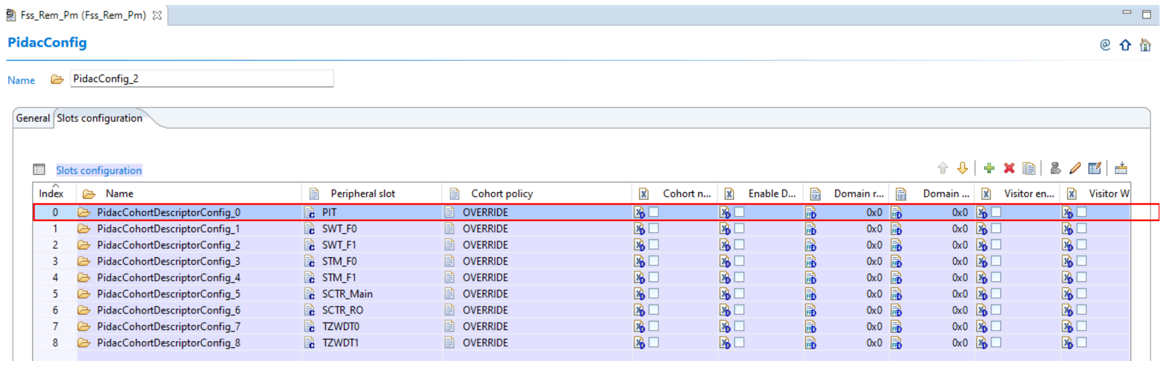
Figure 13. Registers of FSS.XRDC_AD.COSS.eDMA0

4 How to assign the peripheral

This section aims to detail how a peripheral can be shared across multiple cohorts or exclusively assigned to a single cohort. It will be divided into multiple subsections to cover different scenarios.

4.1 Share a peripheral among all cohorts

To share a peripheral among all cohorts, the only approach is to assign the cohort policy of the peripheral slot with the OVERRIDE attribute.



PidacConfig

Name: PidacConfig_2

Slots configuration

Index	Name	Peripheral slot	Cohort policy	Cohort n...	Enable D...	Domain r...	Domain ...	Visitor en...	Visitor W
0	PidacCohortDescriptorConfig_0	PIT	OVERRIDE	☐	☐	0x0	0x0	☐	☐
1	PidacCohortDescriptorConfig_1	SWT_F0	OVERRIDE	☐	☐	0x0	0x0	☐	☐
2	PidacCohortDescriptorConfig_2	SWT_F1	OVERRIDE	☐	☐	0x0	0x0	☐	☐
3	PidacCohortDescriptorConfig_3	STM_F0	OVERRIDE	☐	☐	0x0	0x0	☐	☐
4	PidacCohortDescriptorConfig_4	STM_F1	OVERRIDE	☐	☐	0x0	0x0	☐	☐
5	PidacCohortDescriptorConfig_5	SCTR_Main	OVERRIDE	☐	☐	0x0	0x0	☐	☐
6	PidacCohortDescriptorConfig_6	SCTR_RO	OVERRIDE	☐	☐	0x0	0x0	☐	☐
7	PidacCohortDescriptorConfig_7	TZWDT0	OVERRIDE	☐	☐	0x0	0x0	☐	☐
8	PidacCohortDescriptorConfig_8	TZWDT1	OVERRIDE	☐	☐	0x0	0x0	☐	☐

Figure 14. Configuration for sharing the peripheral among all cohorts

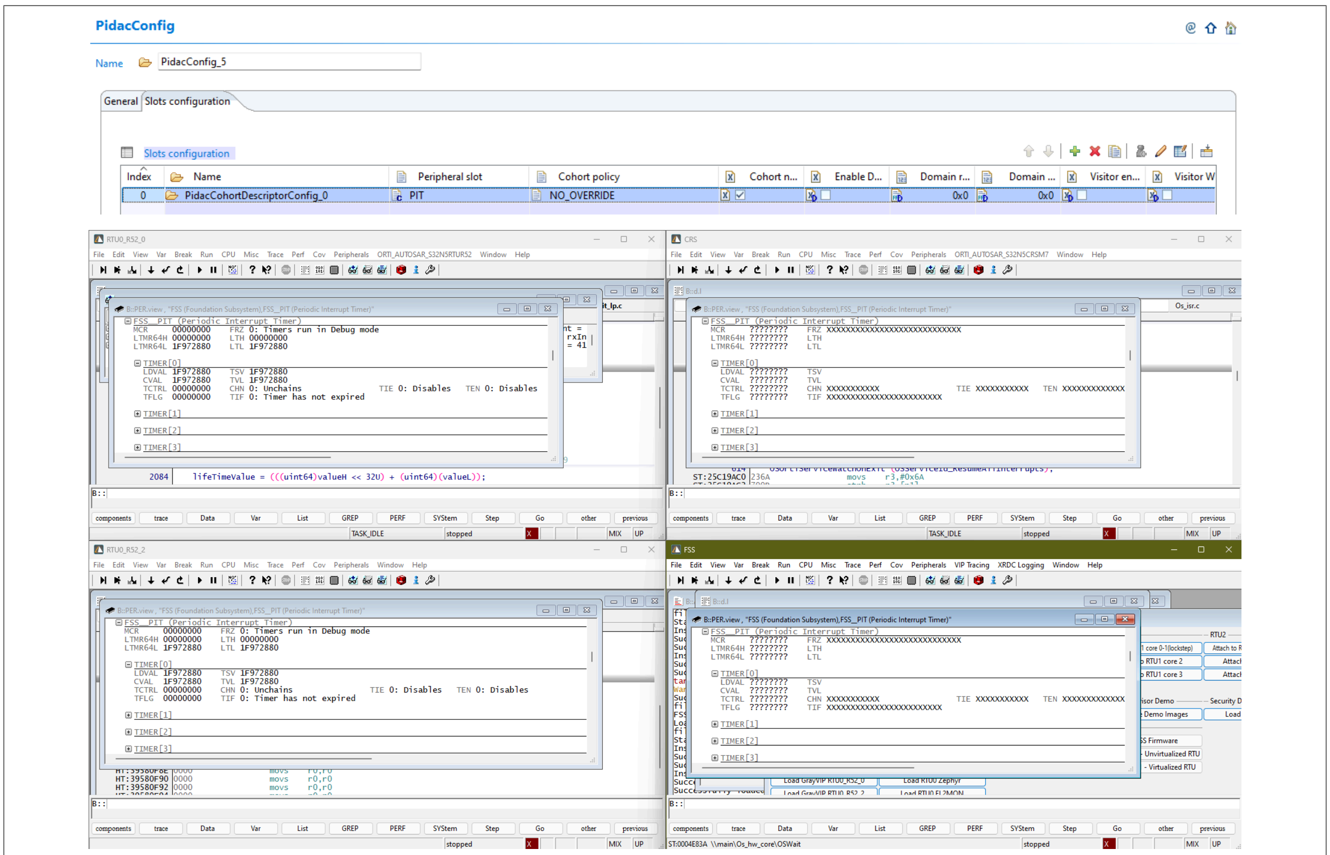
4.2 Share a peripheral inside one cohort

Based on GrayVIP's default 6-cohort configuration, try relocating FSS.PIT to Cohort 2 (RTU0) and enable intra-cohort sharing.

4.2.1 Share a peripheral slot to all domains

To assign a peripheral to a specific cohort and ensure it is inaccessible to other cohorts, follow these main steps:

1. Navigate to the Cohorts configuration tab under the Fss_Rem_Pm module in the project Firmware_S32N55_FSS.
2. The FSS.PIT default assigned to SBC. Remove the configuration of FSS.PIT, which is located at Subsystems configuration(Subsystem FSS)->PIDAC configuration(XRDC4_PIDAC2)->Slots configuration(PIT).
3. Add FSS.PIT slot to cohort 2, with the cohort policy is NO_OVERRIDE. Configuration path: Cohorts configuration(RTU0_AppCohort)->Subsystems configuration(Subsystem FSS)->PIDAC configuration(XRDC4_PIDAC2)->Slots configuration(PIT).



PidacConfig

Name: PidacConfig_5

General | Slots configuration

Slots configuration

Index	Name	Peripheral slot	Cohort policy	Cohort n...	Enable D...	Domain r...	Domain ...	Visitor en...	Visitor W
0	PidacCohortDescriptorConfig_0	PIT	NO_OVERRIDE	✓	☐	0x0	0x0	☐	☐

RTU0_R52_0 — B::PER.view , "FSS (Foundation Subsystem),FSS__PIT (Periodic Interrupt Timer)"

```
FSS__PIT (Periodic Interrupt Timer)
  MCR      00000000    FRZ 0: Timers run in Debug mode
  LTMR64H  00000000    LTH 00000000
  LTMR64L  1F972880    LTL 1F972880

  TIMER[0]
    LDVAL  1F972880    TSV 1F972880
    CVAL   1F972880    TVL 1F972880
    TCTRL  00000000    CHN 0: Unchains               TIE 0: Disables    TEN 0: Disables
    TFLG   00000000    TIF 0: Timer has not expired

  TIMER[1]
  TIMER[2]
  TIMER[3]

2084    lifeTimeValue = (((uint64)valueH << 32U) + (uint64)(valueL));
```

CRS — B::PER.view , "FSS (Foundation Subsystem),FSS__PIT (Periodic Interrupt Timer)"

```
FSS__PIT (Periodic Interrupt Timer)
  MCR      ????????    FRZ XXXXXXXXXXXXXXXXXXXXXXXXXXXX
  LTMR64H  ????????    LTH
  LTMR64L  ????????    LTL

  TIMER[0]
    LDVAL  ????????    TSV
    CVAL   ????????    TVL
    TCTRL  ????????    CHN XXXXXXXXXXX               TIE XXXXXXXXXXX    TEN XXXXXXXXXXXXX
    TFLG   ????????    TIF XXXXXXXXXXXXXXXXXXXXXXXXX

  TIMER[1]
  TIMER[2]
  TIMER[3]

ST:25C19AC0 236A          movs    r3,#0x6A
```

RTU0_R52_2 — B::PER.view , "FSS (Foundation Subsystem),FSS__PIT (Periodic Interrupt Timer)"

```
FSS__PIT (Periodic Interrupt Timer)
  MCR      00000000    FRZ 0: Timers run in Debug mode
  LTMR64H  00000000    LTH 00000000
  LTMR64L  1F972880    LTL 1F972880

  TIMER[0]
    LDVAL  1F972880    TSV 1F972880
    CVAL   1F972880    TVL 1F972880
    TCTRL  00000000    CHN 0: Unchains               TIE 0: Disables    TEN 0: Disables
    TFLG   00000000    TIF 0: Timer has not expired

  TIMER[1]
  TIMER[2]
  TIMER[3]

HT:39580F90 0000          movs    r0,r0
HT:39580F92 0000          movs    r0,r0
```

FSS — B::PER.view , "FSS (Foundation Subsystem),FSS__PIT (Periodic Interrupt Timer)"

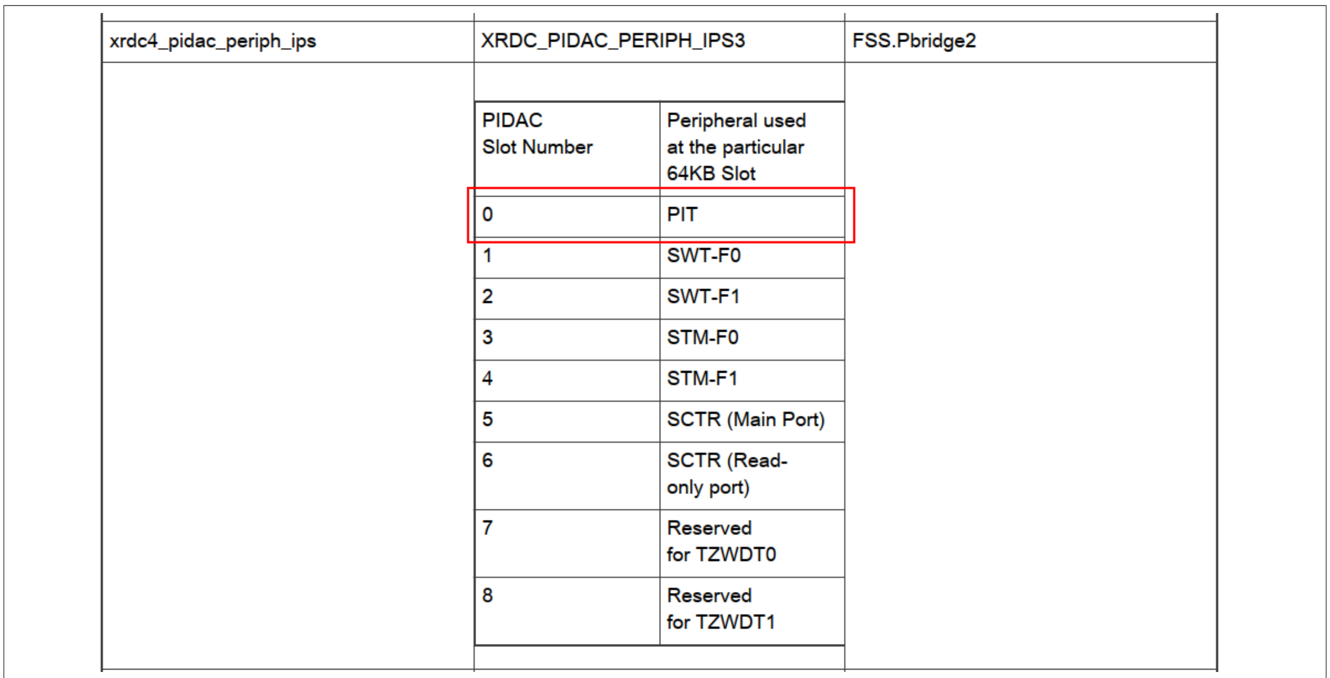
```
FSS__PIT (Periodic Interrupt Timer)
  MCR      ????????    FRZ XXXXXXXXXXXXXXXXXXXXXXXXXXXX
  LTMR64H  ????????    LTH
  LTMR64L  ????????    LTL

  TIMER[0]
    LDVAL  ????????    TSV
    CVAL   ????????    TVL
    TCTRL  ????????    CHN XXXXXXXXXXX               TIE XXXXXXXXXXX    TEN XXXXXXXXXXXXX
    TFLG   ????????    TIF XXXXXXXXXXXXXXXXXXXXXXXXX

  TIMER[1]
  TIMER[2]
  TIMER[3]
```

Figure 15. Configuration for sharing a peripheral to all domains inside one cohort

According to the description of FSS.PIDAC, the FSS.PIT peripheral is associated with XRDC_PIDAC_PERIPH_IPS3.Slot0.



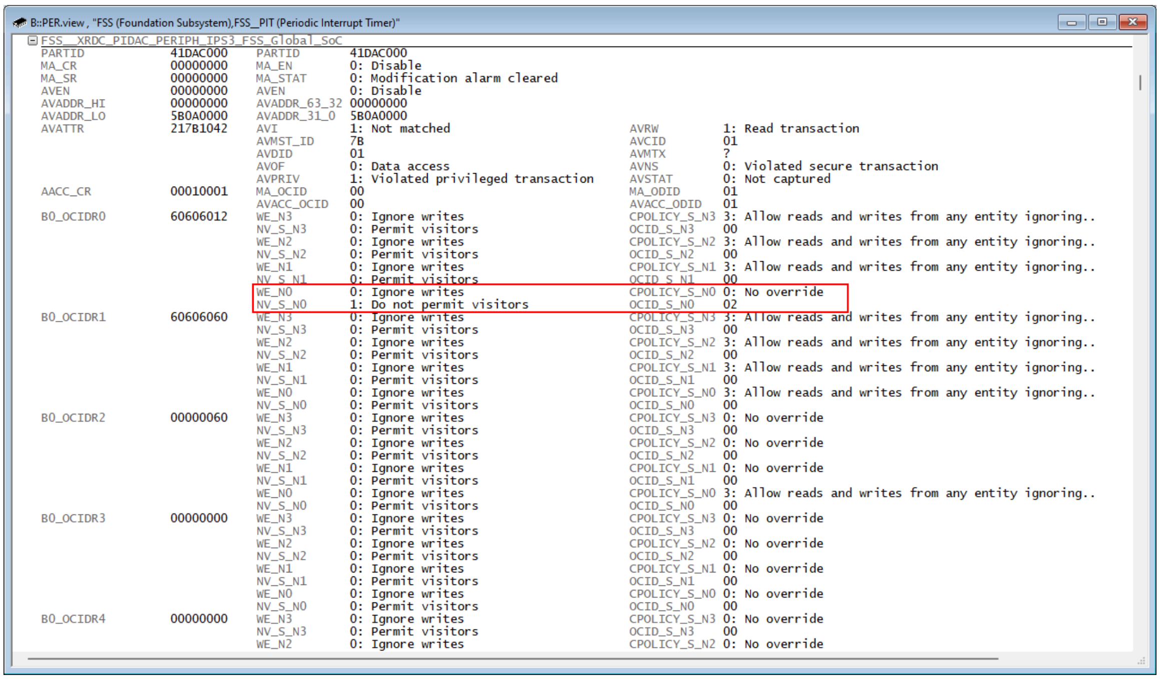
xrdc4_pidac_periph_ips	XRDC_PIDAC_PERIPH_IPS3	FSS.Pbridge2

PIDAC Slot Number	Peripheral used at the particular 64KB Slot
0	PIT
1	SWT-F0
2	SWT-F1
3	STM-F0
4	STM-F1
5	SCTR (Main Port)
6	SCTR (Read-only port)
7	Reserved for TZWDT0
8	Reserved for TZWDT1

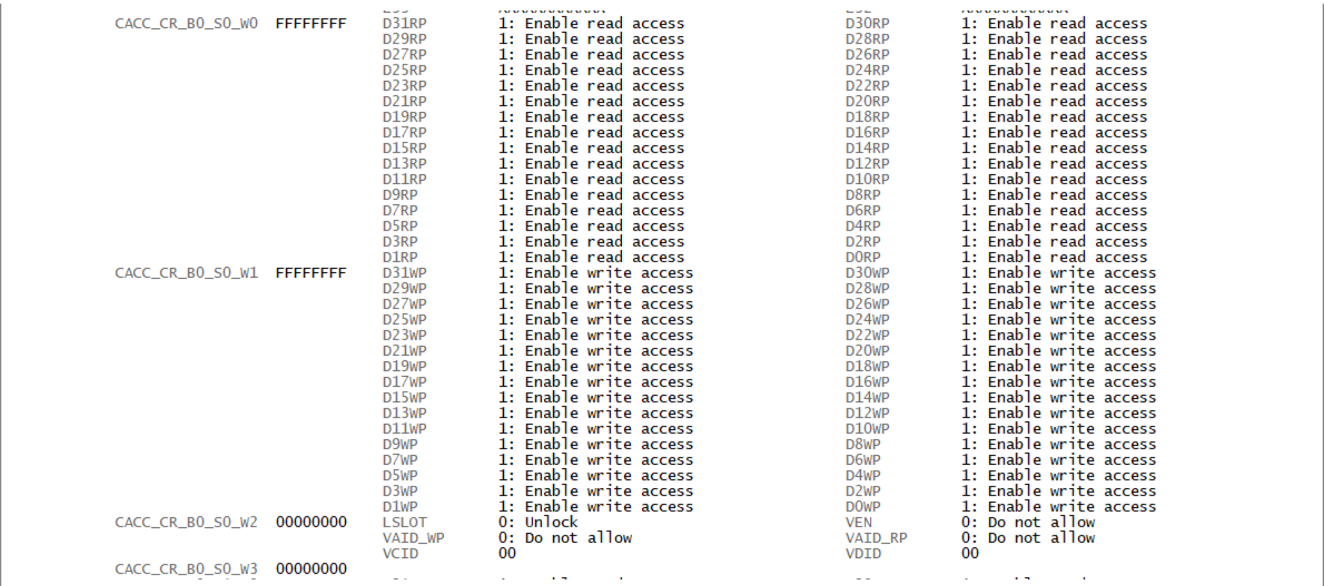
Figure 16. PIDAC instance of FSS.PIT

Therefore:

- Its Cohort ID Entry register is B0_OCIDR0.OCID_S_N0.
- Its slot-related registers (Cohort Access Control Word 0–3) are CAAC_CR_B0_S0_W0 to CAAC_CR_B0_S0_W3.



```
B::PER.view , "FSS (Foundation Subsystem),FSS__PIT (Periodic Interrupt Timer)"
⊟ FSS__XRDC_PIDAC_PERIPH_IPS3_FSS_Global_SoC
  PARTID       41DAC000  PARTID        41DAC000
  MA_CR        00000000  MA_EN         0: Disable
  MA_SR        00000000  MA_STAT       0: Modification alarm cleared
  AVEN         00000000  AVEN          0: Disable
  AVADDR_HI    00000000  AVADDR_63_32  00000000
  AVADDR_LO    5B0A0000  AVADDR_31_0   5B0A0000
  AVATTR       217B1042  AVI           1: Not matched                           AVRW         1: Read transaction
                         AVMST_ID      7B                                       AVCID        01
                         AVDID         01                                       AVMTX        ?
                         AVOF          0: Data access                           AVNS         0: Violated secure transaction
                         AVPRIV        1: Violated privileged transaction       AVSTAT       0: Not captured
  AACC_CR      00010001  MA_OCID       00                                       MA_ODID      01
                         AVACC_OCID    00                                       AVACC_ODID   01
  B0_OCIDR0    60606012  WE_N3         0: Ignore writes                         CPOLICY_S_N3 3: Allow reads and writes from any entity ignoring..
                         NV_S_N3       0: Permit visitors                       OCID_S_N3    00
                         WE_N2         0: Ignore writes                         CPOLICY_S_N2 3: Allow reads and writes from any entity ignoring..
                         NV_S_N2       0: Permit visitors                       OCID_S_N2    00
                         WE_N1         0: Ignore writes                         CPOLICY_S_N1 3: Allow reads and writes from any entity ignoring..
                         NV_S_N1       0: Permit visitors                       OCID_S_N1    00
                         WE_N0         0: Ignore writes                         CPOLICY_S_N0 0: No override
                         NV_S_N0       1: Do not permit visitors                OCID_S_N0    02
  B0_OCIDR1    60606060  WE_N3         0: Ignore writes                         CPOLICY_S_N3 3: Allow reads and writes from any entity ignoring..
                         NV_S_N3       0: Permit visitors                       OCID_S_N3    00
                         WE_N2         0: Ignore writes                         CPOLICY_S_N2 3: Allow reads and writes from any entity ignoring..
                         NV_S_N2       0: Permit visitors                       OCID_S_N2    00
                         WE_N1         0: Ignore writes                         CPOLICY_S_N1 3: Allow reads and writes from any entity ignoring..
                         NV_S_N1       0: Permit visitors                       OCID_S_N1    00
                         WE_N0         0: Ignore writes                         CPOLICY_S_N0 3: Allow reads and writes from any entity ignoring..
                         NV_S_N0       0: Permit visitors                       OCID_S_N0    00
  B0_OCIDR2    00000060  WE_N3         0: Ignore writes                         CPOLICY_S_N3 0: No override
                         NV_S_N3       0: Permit visitors                       OCID_S_N3    00
                         WE_N2         0: Ignore writes                         CPOLICY_S_N2 0: No override
                         NV_S_N2       0: Permit visitors                       OCID_S_N2    00
                         WE_N1         0: Ignore writes                         CPOLICY_S_N1 0: No override
                         NV_S_N1       0: Permit visitors                       OCID_S_N1    00
                         WE_N0         0: Ignore writes                         CPOLICY_S_N0 3: Allow reads and writes from any entity ignoring..
                         NV_S_N0       0: Permit visitors                       OCID_S_N0    00
  B0_OCIDR3    00000000  WE_N3         0: Ignore writes                         CPOLICY_S_N3 0: No override
                         NV_S_N3       0: Permit visitors                       OCID_S_N3    00
                         WE_N2         0: Ignore writes                         CPOLICY_S_N2 0: No override
                         NV_S_N2       0: Permit visitors                       OCID_S_N2    00
                         WE_N1         0: Ignore writes                         CPOLICY_S_N1 0: No override
                         NV_S_N1       0: Permit visitors                       OCID_S_N1    00
                         WE_N0         0: Ignore writes                         CPOLICY_S_N0 0: No override
                         NV_S_N0       0: Permit visitors                       OCID_S_N0    00
  B0_OCIDR4    00000000  WE_N3         0: Ignore writes                         CPOLICY_S_N3 0: No override
                         NV_S_N3       0: Permit visitors                       OCID_S_N3    00
                         WE_N2         0: Ignore writes                         CPOLICY_S_N2 0: No override
```



```
CACC_CR_B0_S0_W0   FFFFFFFF   D31RP      1: Enable read access     D30RP      1: Enable read access
                              D29RP      1: Enable read access     D28RP      1: Enable read access
                              D27RP      1: Enable read access     D26RP      1: Enable read access
                              D25RP      1: Enable read access     D24RP      1: Enable read access
                              D23RP      1: Enable read access     D22RP      1: Enable read access
                              D21RP      1: Enable read access     D20RP      1: Enable read access
                              D19RP      1: Enable read access     D18RP      1: Enable read access
                              D17RP      1: Enable read access     D16RP      1: Enable read access
                              D15RP      1: Enable read access     D14RP      1: Enable read access
                              D13RP      1: Enable read access     D12RP      1: Enable read access
                              D11RP      1: Enable read access     D10RP      1: Enable read access
                              D9RP       1: Enable read access     D8RP       1: Enable read access
                              D7RP       1: Enable read access     D6RP       1: Enable read access
                              D5RP       1: Enable read access     D4RP       1: Enable read access
                              D3RP       1: Enable read access     D2RP       1: Enable read access
                              D1RP       1: Enable read access     D0RP       1: Enable read access
CACC_CR_B0_S0_W1   FFFFFFFF   D31WP      1: Enable write access    D30WP      1: Enable write access
                              D29WP      1: Enable write access    D28WP      1: Enable write access
                              D27WP      1: Enable write access    D26WP      1: Enable write access
                              D25WP      1: Enable write access    D24WP      1: Enable write access
                              D23WP      1: Enable write access    D22WP      1: Enable write access
                              D21WP      1: Enable write access    D20WP      1: Enable write access
                              D19WP      1: Enable write access    D18WP      1: Enable write access
                              D17WP      1: Enable write access    D16WP      1: Enable write access
                              D15WP      1: Enable write access    D14WP      1: Enable write access
                              D13WP      1: Enable write access    D12WP      1: Enable write access
                              D11WP      1: Enable write access    D10WP      1: Enable write access
                              D9WP       1: Enable write access    D8WP       1: Enable write access
                              D7WP       1: Enable write access    D6WP       1: Enable write access
                              D5WP       1: Enable write access    D4WP       1: Enable write access
                              D3WP       1: Enable write access    D2WP       1: Enable write access
                              D1WP       1: Enable write access    D0WP       1: Enable write access
CACC_CR_B0_S0_W2   00000000   LSLOT      0: Unlock                 VEN        0: Do not allow
                              VAID_WP    0: Do not allow           VAID_RP    0: Do not allow
                              VCID       00                        VDID       00
CACC_CR_B0_S0_W3   00000000
```

Figure 17. Registers of PIDAC when assigning a peripheral to all domains in one cohort

From the debug tool's register view, the slot is associated with FSS.PIT belongs to Cohort 2, with the policy set to NO_OVERRIDE, and the "no visitor" attribute enabled—meaning it does not allow access from any external entities.

At the same time, this slot is accessible within the same cohort through CAAC_CR_B0_S0_W0 and CAAC_CR_B0_S0_W1:

- CAAC_CR_B0_S0_W0 enables read access for all domains.
- CAAC_CR_B0_S0_W1 enables write access for all domains.

The registers CAAC_CR_B0_S0_W0 and CAAC_CR_B0_S0_W1 are automatically asserted when a non-zero modification is made to B0_OCIDR0.OCID_S_N0.

4.2.2 Assign a peripheral slot to one or multiple domains

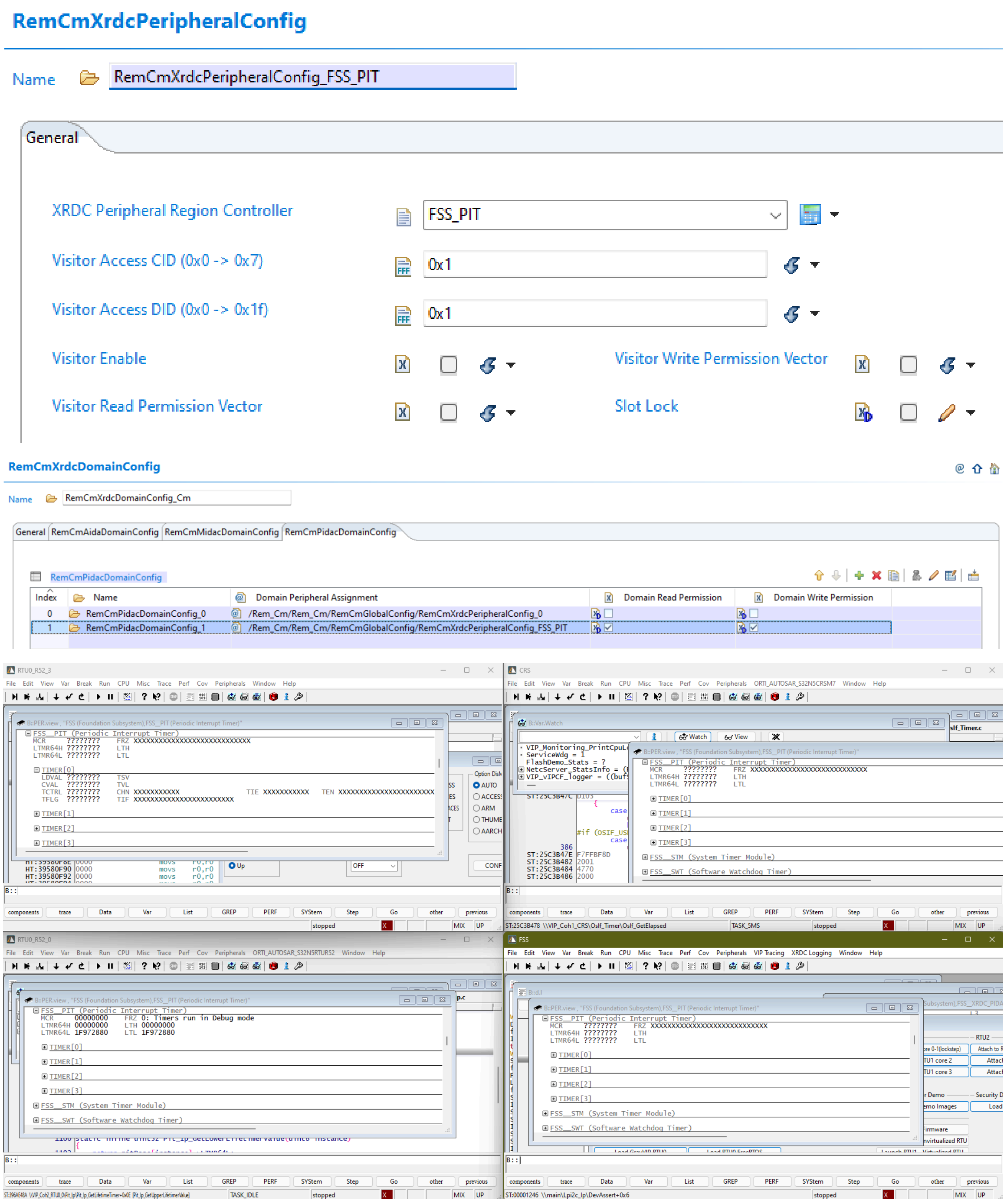
This section is based on the configuration of Fss_Rem_Pm in the previous section.

Here are the steps:

1. Navigate to the XRDC Peripheral Config tab under the Rem_Cm module in the project S32N5_RTU0_R52_0_VIP_App.
2. Add a new configuration named FSS_PIT as shown below. The visitor attribute will not take effect if OCIDR.NV is asserted, which is configured in the Fss_Rem_Pm module.
3. Navigate to the Domain Config tab.
4. Add RemCmPidacDomainConfig of FSS_PIT to Domain 1 and enable its read and write permissions.
   a. Open XRDC4_DOMAIN1 entry.
   b. Navigate to the RemCmPidacDomainConfig tab.
   c. Add an entry and link it to the Domain Peripheral Assignment of FSS_PIT. Enable both read and write permissions.

Note: *If a peripheral is never linked to any domain(without steps 3 and 4), it will deny all access attempts.*

Note: *If a peripheral requires access from multiple inner-cohort domains, you need to repeat Step 4 to configure each domain individually.*



RemCmXrdcPeripheralConfig

Name: RemCmXrdcPeripheralConfig_FSS_PIT

General

XRDC Peripheral Region Controller	FSS_PIT
Visitor Access CID (0x0 -> 0x7)	0x1
Visitor Access DID (0x0 -> 0x1f)	0x1
Visitor Enable	☐
Visitor Write Permission Vector	☐
Visitor Read Permission Vector	☐
Slot Lock	☐

RemCmXrdcDomainConfig

Name: RemCmXrdcDomainConfig_Cm

General | RemCmAidaDomainConfig | RemCmMidacDomainConfig | RemCmPidacDomainConfig

RemCmPidacDomainConfig

Index	Name	Domain Peripheral Assignment	Domain Read Permission	Domain Write Permission
0	RemCmPidacDomainConfig_0	/Rem_Cm/Rem_Cm/RemCmGlobalConfig/RemCmXrdcPeripheralConfig_0	☐	☐
1	RemCmPidacDomainConfig_1	/Rem_Cm/Rem_Cm/RemCmGlobalConfig/RemCmXrdcPeripheralConfig_FSS_PIT	✓	✓

RTU0_R52_3 — B::PER.view , "FSS (Foundation Subsystem),FSS__PIT (Periodic Interrupt Timer)"

```
FSS__PIT (Periodic Interrupt Timer)
  MCR     ????????    FRZ XXXXXXXXXXXXXXXXXXXXXXXXXXXX
  LTMR64H ????????    LTH
  LTMR64L ????????    LTL

  TIMER[0]
    LDVAL ????????    TSV
    CVAL  ????????    TVL
    TCTRL ????????    CHN XXXXXXXXXXX          TIE XXXXXXXXXXX    TEN XXXXXXXXXXXXXXXXXXXXXXX
    TFLG  ????????    TIF XXXXXXXXXXXXXXXXXXXXXXXX
  TIMER[1]
  TIMER[2]
  TIMER[3]
```

stopped

CRS — B::Var.Watch

```
• VIP_Monitoring_PrintCpuL
• ServiceWdg = 1
  FlashDemo_Stats = ?
⊞ NetcServer_StatsInfo = (
⊞ VIP_vIPCF_logger = ((buf
```

B::PER.view , "FSS (Foundation Subsystem),FSS__PIT (Periodic Interrupt Timer)"

```
FSS__PIT (Periodic Interrupt Timer)
  MCR     ????????    FRZ XXXXXXXXXXXXXXXXXXXXXXXXXXXX
  LTMR64H ????????    LTH
  LTMR64L ????????    LTL

  TIMER[0]
  TIMER[1]
  TIMER[2]
  TIMER[3]
FSS__STM (System Timer Module)
FSS__SWT (Software Watchdog Timer)
```

ST:25C3B478 \\VIP_Coh1_CRS\Oslf_Timer\Oslf_GetElapsed | TASK_5MS | stopped

RTU0_R52_0 — B::PER.view , "FSS (Foundation Subsystem),FSS__PIT (Periodic Interrupt Timer)"

```
FSS__PIT (Periodic Interrupt Timer)
  MCR     00000000    FRZ 0: Timers run in Debug mode
  LTMR64H 00000000    LTH 00000000
  LTMR64L 1F972880    LTL 1F972880

  TIMER[0]
  TIMER[1]
  TIMER[2]
  TIMER[3]
FSS__STM (System Timer Module)
FSS__SWT (Software Watchdog Timer)
```

ST:396AE48A \\VIP_Coh2_RTU0_0\Pit_Ip\Pit_Ip_GetLifetimeTimer+0x0E [Pit_Ip_GetUpperLifetimerValue] | TASK_IDLE | stopped

FSS — B::PER.view , "FSS (Foundation Subsystem),FSS__PIT (Periodic Interrupt Timer)"

```
FSS__PIT (Periodic Interrupt Timer)
  MCR     ????????    FRZ XXXXXXXXXXXXXXXXXXXXXXXXXXXX
  LTMR64H ????????    LTH
  LTMR64L ????????    LTL

  TIMER[0]
  TIMER[1]
  TIMER[2]
  TIMER[3]
FSS__STM (System Timer Module)
FSS__SWT (Software Watchdog Timer)
```

ST:00001246 \\main\Lpi2c_Ip\DevAssert+0x6 | stopped

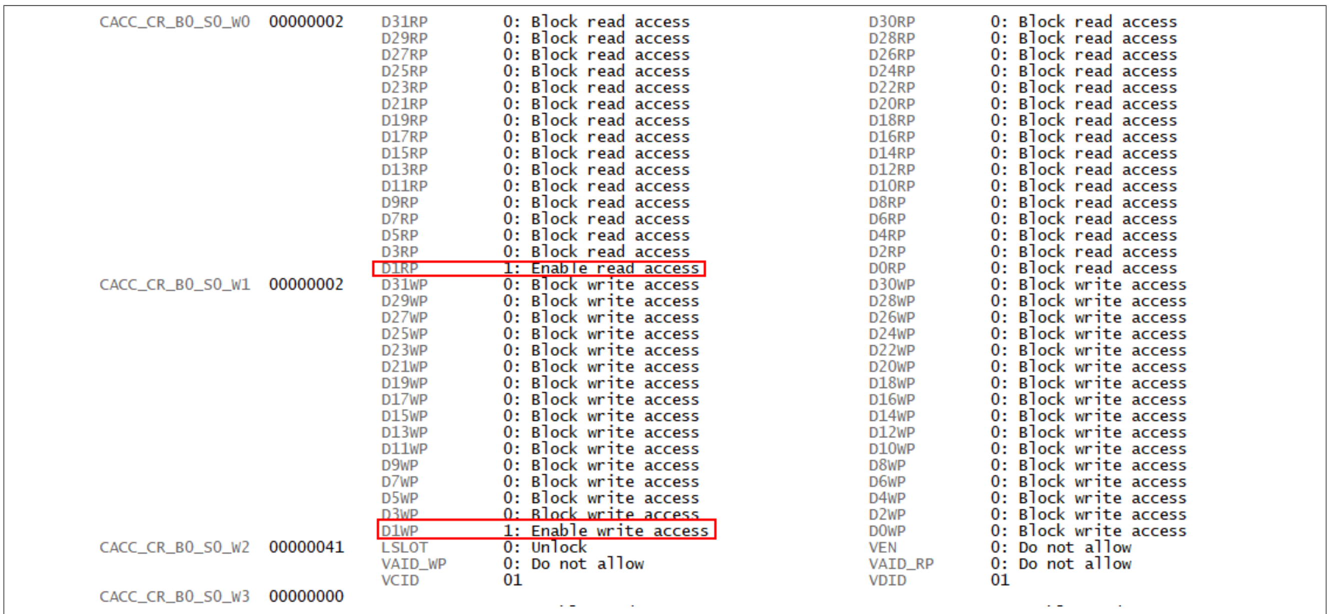
Figure 18. Configuration for sharing a peripheral to one domain

After completing these configurations and once RTU0_R52_0 finishes executing the Rem_Cm_Init function, the peripheral slot for FSS_PIT can only be accessed by AID 2.1.

Both RTU0_R52_0 and RTU0_R52_2 are associated with AID 2.1, so these two initiators are allowed to access FSS_PIT.

In contrast, RTU0_R52_3 is assigned to Domain 3. Therefore, its access to FSS_PIT is denied by XRDC.

The corresponding slot-related register values are shown below.



```
CACC_CR_B0_S0_W0  00000002  D31RP   0: Block read access    D30RP    0: Block read access
                            D29RP   0: Block read access    D28RP    0: Block read access
                            D27RP   0: Block read access    D26RP    0: Block read access
                            D25RP   0: Block read access    D24RP    0: Block read access
                            D23RP   0: Block read access    D22RP    0: Block read access
                            D21RP   0: Block read access    D20RP    0: Block read access
                            D19RP   0: Block read access    D18RP    0: Block read access
                            D17RP   0: Block read access    D16RP    0: Block read access
                            D15RP   0: Block read access    D14RP    0: Block read access
                            D13RP   0: Block read access    D12RP    0: Block read access
                            D11RP   0: Block read access    D10RP    0: Block read access
                            D9RP    0: Block read access    D8RP     0: Block read access
                            D7RP    0: Block read access    D6RP     0: Block read access
                            D5RP    0: Block read access    D4RP     0: Block read access
                            D3RP    0: Block read access    D2RP     0: Block read access
                            D1RP    1: Enable read access   D0RP     0: Block read access
CACC_CR_B0_S0_W1  00000002  D31WP   0: Block write access   D30WP    0: Block write access
                            D29WP   0: Block write access   D28WP    0: Block write access
                            D27WP   0: Block write access   D26WP    0: Block write access
                            D25WP   0: Block write access   D24WP    0: Block write access
                            D23WP   0: Block write access   D22WP    0: Block write access
                            D21WP   0: Block write access   D20WP    0: Block write access
                            D19WP   0: Block write access   D18WP    0: Block write access
                            D17WP   0: Block write access   D16WP    0: Block write access
                            D15WP   0: Block write access   D14WP    0: Block write access
                            D13WP   0: Block write access   D12WP    0: Block write access
                            D11WP   0: Block write access   D10WP    0: Block write access
                            D9WP    0: Block write access   D8WP     0: Block write access
                            D7WP    0: Block write access   D6WP     0: Block write access
                            D5WP    0: Block write access   D4WP     0: Block write access
                            D3WP    0: Block write access   D2WP     0: Block write access
                            D1WP    1: Enable write access  D0WP     0: Block write access
CACC_CR_B0_S0_W2  00000041  LSLOT   0: Unlock               VEN      0: Do not allow
                            VAID_WP 0: Do not allow         VAID_RP  0: Do not allow
                            VCID    01                      VDID     01
CACC_CR_B0_S0_W3  00000000
```

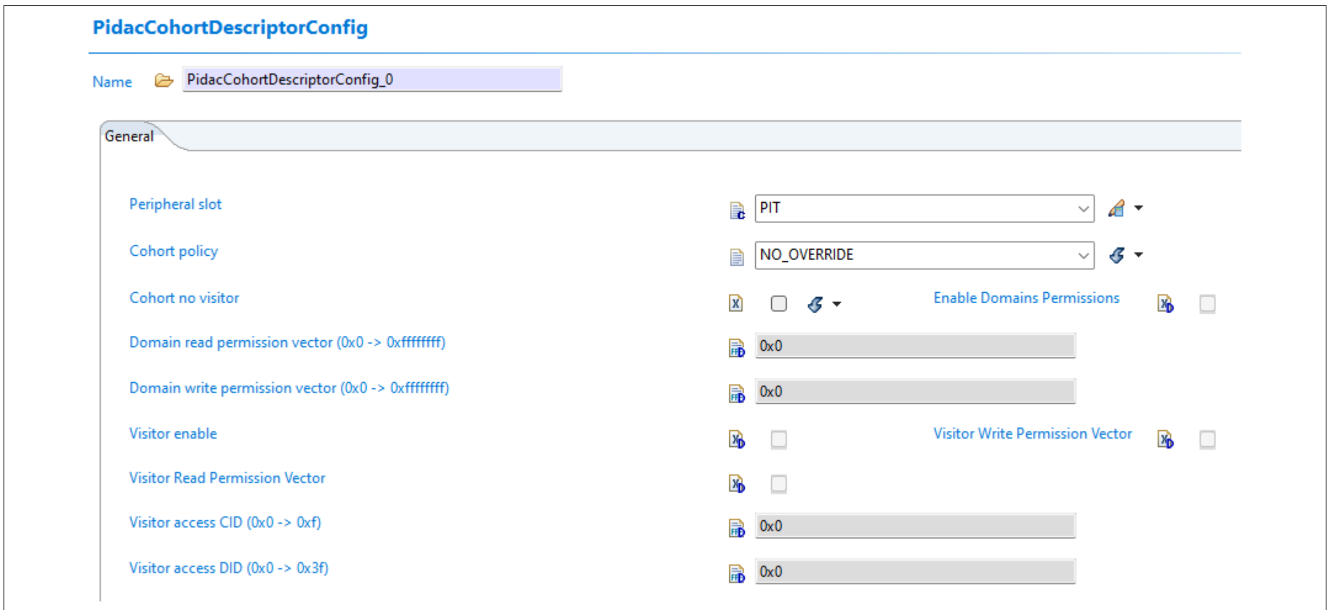
Figure 19. Registers of PIDAC when assigning a peripheral to one domain

4.3 Share a peripheral between two cohorts

For a peripheral slot, only one outer-cohort visitor is allowed. In other words, if the cohort policy of the peripheral slot is not set to OVERRIDE, the peripheral can only be accessed by two cohorts at most.

The steps for enabling a visitor are listed below:

1. Deassert the NV attribute of the peripheral slot in the Fss_Rem_Pm module.
2. Enable the VEN attribute in the Rem_Cm module.
3. Enable the VAID_WP and VAID_RP, depending on write and read access requirements.
4. Fill in the VCID and VDID fields to ensure that the corresponding initiator is granted access rights.

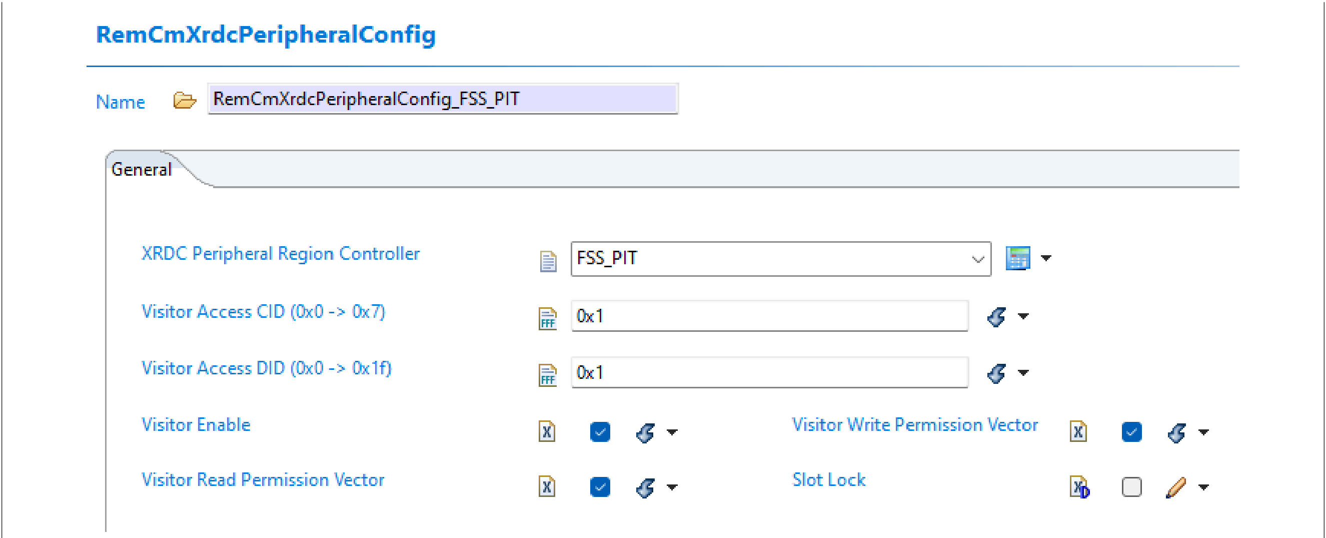


PidacCohortDescriptorConfig

Name: PidacCohortDescriptorConfig_0

General

Peripheral slot	PIT		
Cohort policy	NO_OVERRIDE		
Cohort no visitor	☐	Enable Domains Permissions	☐
Domain read permission vector (0x0 -> 0xffffffff)	0x0		
Domain write permission vector (0x0 -> 0xffffffff)	0x0		
Visitor enable	☐	Visitor Write Permission Vector	☐
Visitor Read Permission Vector	☐		
Visitor access CID (0x0 -> 0xf)	0x0		
Visitor access DID (0x0 -> 0x3f)	0x0		



RemCmXrdcPeripheralConfig

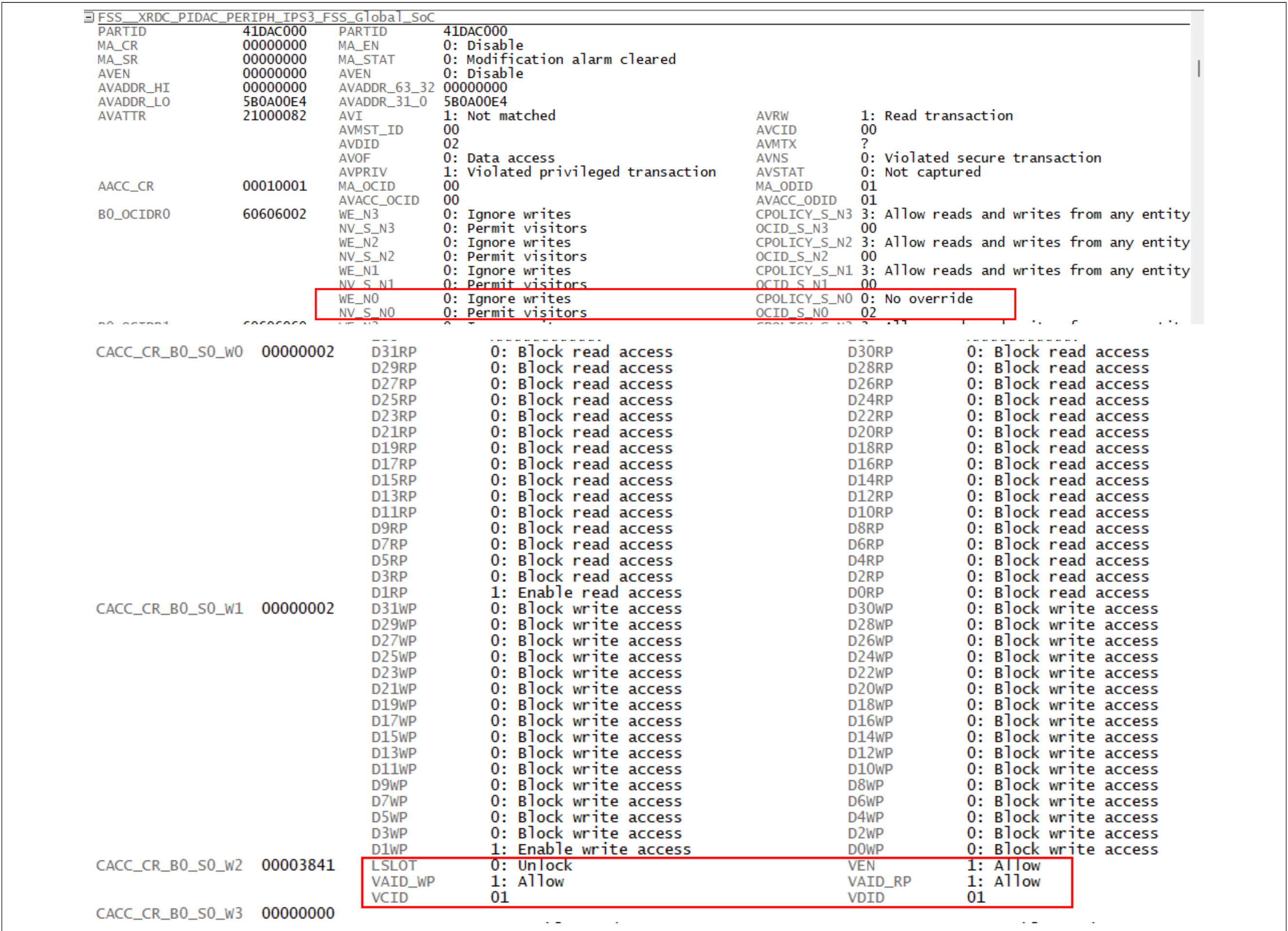
Name: RemCmXrdcPeripheralConfig_FSS_PIT

General

XRDC Peripheral Region Controller	FSS_PIT		
Visitor Access CID (0x0 -> 0x7)	0x1		
Visitor Access DID (0x0 -> 0x1f)	0x1		
Visitor Enable	✓	Visitor Write Permission Vector	✓
Visitor Read Permission Vector	✓	Slot Lock	☐

Figure 20. Configuration for sharing a peripheral via a visitor

In this example, the granted initiator is associated with AID 1.1, which means FSS_PIT can be accessed by domain 1 of cohort 1.



```
FSS__XRDC_PIDAC_PERIPH_IPS3_FSS_Global_SoC
PARTID       41DAC000   PARTID        41DAC000
MA_CR        00000000   MA_EN         0: Disable
MA_SR        00000000   MA_STAT       0: Modification alarm cleared
AVEN         00000000   AVEN          0: Disable
AVADDR_HI    00000000   AVADDR_63_32  00000000
AVADDR_LO    5B0A00E4   AVADDR_31_0   5B0A00E4
AVATTR       21000082   AVI           1: Not matched                     AVRW          1: Read transaction
                        AVMST_ID      00                                 AVCID         00
                        AVDID         02                                 AVMTX         ?
                        AVOF          0: Data access                     AVNS          0: Violated secure transaction
                        AVPRIV        1: Violated privileged transaction AVSTAT        0: Not captured
AACC_CR      00010001   MA_OCID       00                                 MA_ODID       01
                        AVACC_OCID    00                                 AVACC_ODID    01
B0_OCIDR0    60606002   WE_N3         0: Ignore writes                   CPOLICY_S_N3  3: Allow reads and writes from any entity
                        NV_S_N3       0: Permit visitors                 OCID_S_N3     00
                        WE_N2         0: Ignore writes                   CPOLICY_S_N2  3: Allow reads and writes from any entity
                        NV_S_N2       0: Permit visitors                 OCID_S_N2     00
                        WE_N1         0: Ignore writes                   CPOLICY_S_N1  3: Allow reads and writes from any entity
                        NV_S_N1       0: Permit visitors                 OCID_S_N1     00
                        WE_N0         0: Ignore writes                   CPOLICY_S_N0  0: No override
                        NV_S_N0       0: Permit visitors                 OCID_S_N0     02

CACC_CR_B0_S0_W0  00000002  D31RP   0: Block read access      D30RP   0: Block read access
                            D29RP   0: Block read access      D28RP   0: Block read access
                            D27RP   0: Block read access      D26RP   0: Block read access
                            D25RP   0: Block read access      D24RP   0: Block read access
                            D23RP   0: Block read access      D22RP   0: Block read access
                            D21RP   0: Block read access      D20RP   0: Block read access
                            D19RP   0: Block read access      D18RP   0: Block read access
                            D17RP   0: Block read access      D16RP   0: Block read access
                            D15RP   0: Block read access      D14RP   0: Block read access
                            D13RP   0: Block read access      D12RP   0: Block read access
                            D11RP   0: Block read access      D10RP   0: Block read access
                            D9RP    0: Block read access      D8RP    0: Block read access
                            D7RP    0: Block read access      D6RP    0: Block read access
                            D5RP    0: Block read access      D4RP    0: Block read access
                            D3RP    0: Block read access      D2RP    0: Block read access
                            D1RP    1: Enable read access     D0RP    0: Block read access
CACC_CR_B0_S0_W1  00000002  D31WP   0: Block write access     D30WP   0: Block write access
                            D29WP   0: Block write access     D28WP   0: Block write access
                            D27WP   0: Block write access     D26WP   0: Block write access
                            D25WP   0: Block write access     D24WP   0: Block write access
                            D23WP   0: Block write access     D22WP   0: Block write access
                            D21WP   0: Block write access     D20WP   0: Block write access
                            D19WP   0: Block write access     D18WP   0: Block write access
                            D17WP   0: Block write access     D16WP   0: Block write access
                            D15WP   0: Block write access     D14WP   0: Block write access
                            D13WP   0: Block write access     D12WP   0: Block write access
                            D11WP   0: Block write access     D10WP   0: Block write access
                            D9WP    0: Block write access     D8WP    0: Block write access
                            D7WP    0: Block write access     D6WP    0: Block write access
                            D5WP    0: Block write access     D4WP    0: Block write access
                            D3WP    0: Block write access     D2WP    0: Block write access
                            D1WP    1: Enable write access    D0WP    0: Block write access
CACC_CR_B0_S0_W2  00003841  LSLOT   0: Unlock                 VEN     1: Allow
                            VAID_WP 1: Allow                  VAID_RP 1: Allow
                            VCID    01                        VDID    01
CACC_CR_B0_S0_W3  00000000
```

Figure 21. Registers of PIDAC when assigning a peripheral between two cohorts

5 How to assign the memory

5.1 Share memory among all cohorts

There are two ways to share memory among all cohorts:

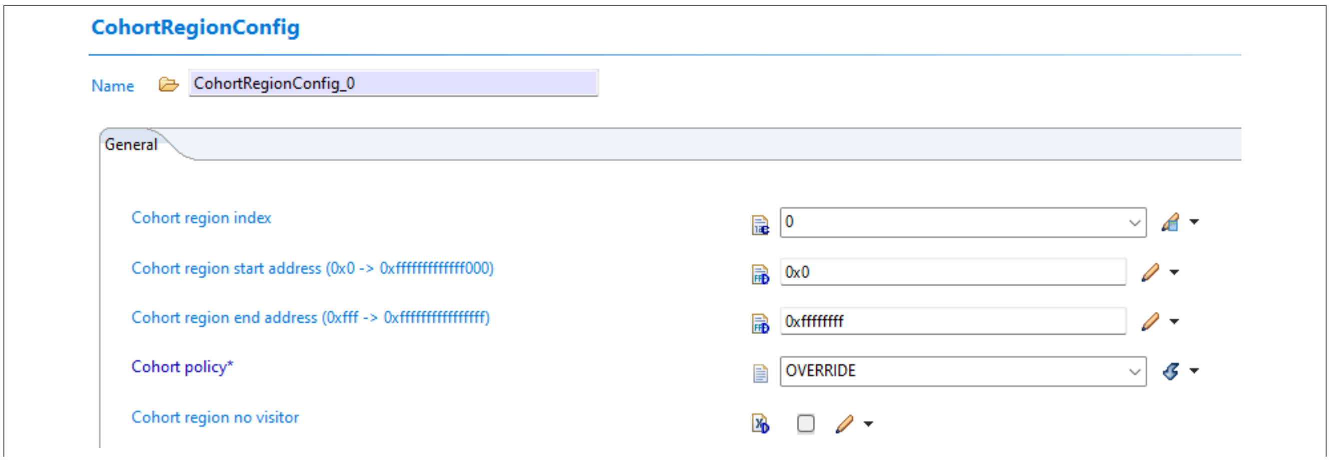
1. Ensure that all transactions match a slot that grants access to all cohorts.
2. Configure multiple slots to allow each cohort access individually.

In this section, the first method will be chosen to share memory across all cohorts. As previously mentioned, the slot with the lowest index—where the OCID or visitor CID matches the transaction CID and the address region bounds the transaction address—will be selected. Furthermore, if the Cohort Policy of the region associated with that slot is set to OVERRIDE, the slot will also be selected for all initiators. Therefore, it is preferable to choose the first memory slot that has the OVERRIDE attribute.

The goal is to: share RTU1.CRAM6 among all cohorts.

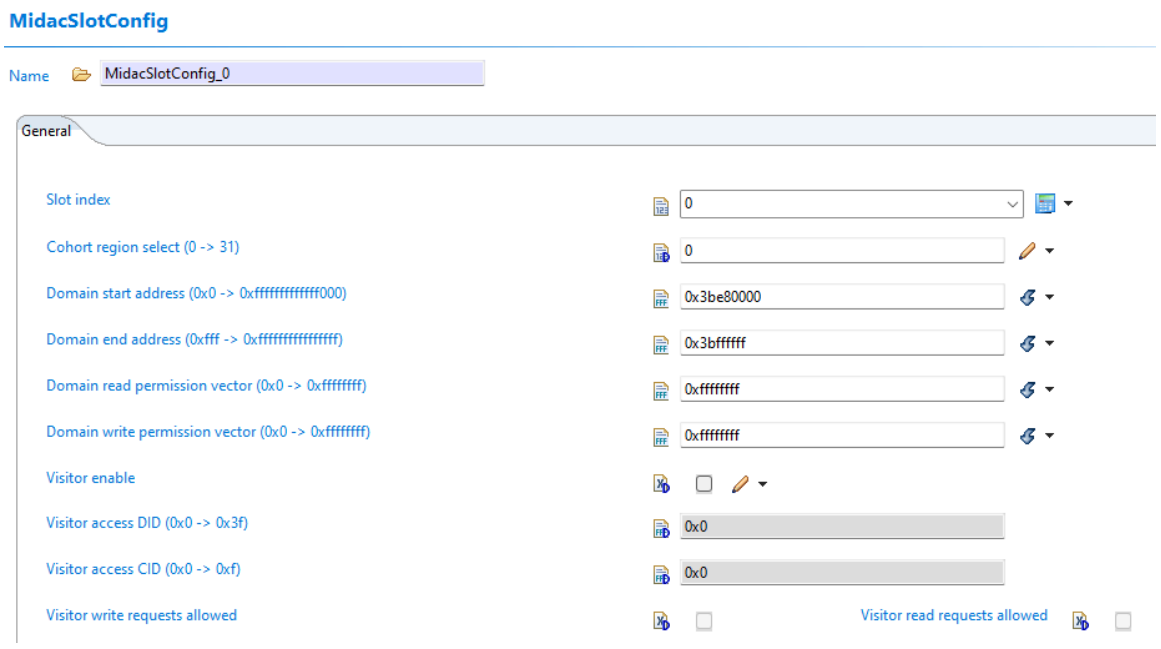
Steps are listed:

1. Navigate to the Cohorts configuration tab under the Fss_Rem_Pm module in the project Firmware_S32N55_FSS.
2. Adaptions for RTU1.MIDAC.CRAM6 in Cohort 0. Go to SBC_Cohort0->Subsystems configuration(RTU1)->MIDAC configuration(XRDC4_MIDAC12)->Cohort regions configuration. Add a new entry for the region with below configurations:
   a. Cohort region index: 0
   b. Cohort region start address: 0x0
   c. Cohort region end address: 0xFFFF_FFFF
   d. Cohort policy: OVERRIDE
3. Add a new entry for slot:
   a. Slot index: 0
   b. Cohort region select: 0
   c. Domain start address: 0x3BE8_0000
   d. Domain end address: 0x3BFF_FFFF
4. Remove the duplicate slot and region configuration for RTU1.MIDAC.CRAM6 from other cohort configuration pages.



CohortRegionConfig

Name	CohortRegionConfig_0
General	
Cohort region index	0
Cohort region start address (0x0 -> 0xfffffffffffff000)	0x0
Cohort region end address (0xfff -> 0xffffffffffffffff)	0xffffffff
Cohort policy*	OVERRIDE
Cohort region no visitor	☐


MidacSlotConfig

Name: MidacSlotConfig_0

General

Slot index: 0

Cohort region select (0 -> 31): 0

Domain start address (0x0 -> 0xfffffffffffff000): 0x3be80000

Domain end address (0xfff -> 0xffffffffffffffff): 0x3bffffff

Domain read permission vector (0x0 -> 0xffffffff): 0xffffffff

Domain write permission vector (0x0 -> 0xffffffff): 0xffffffff

Visitor enable: ☐

Visitor access DID (0x0 -> 0x3f): 0x0

Visitor access CID (0x0 -> 0xf): 0x0

Visitor write requests allowed: ☐

Visitor read requests allowed: ☐

Figure 22. Configuration for sharing memory among all cohorts

5.2 Share memory inside one cohort

This section demonstrates how to share memory within a single cohort. In the previous section, RTU1.CRAM6 was configured to be accessible by all initiators. First, the Cohort Policy for that configuration needs to be changed to NO_OVERRIDE. After this change, RTU1.CRAM6 will only be accessible by Cohort 0 and will follow a set of XRDC rules. Finally, Cohort 2 (RTU0) will be granted a series of permissions to access RTU1.CRAM6.

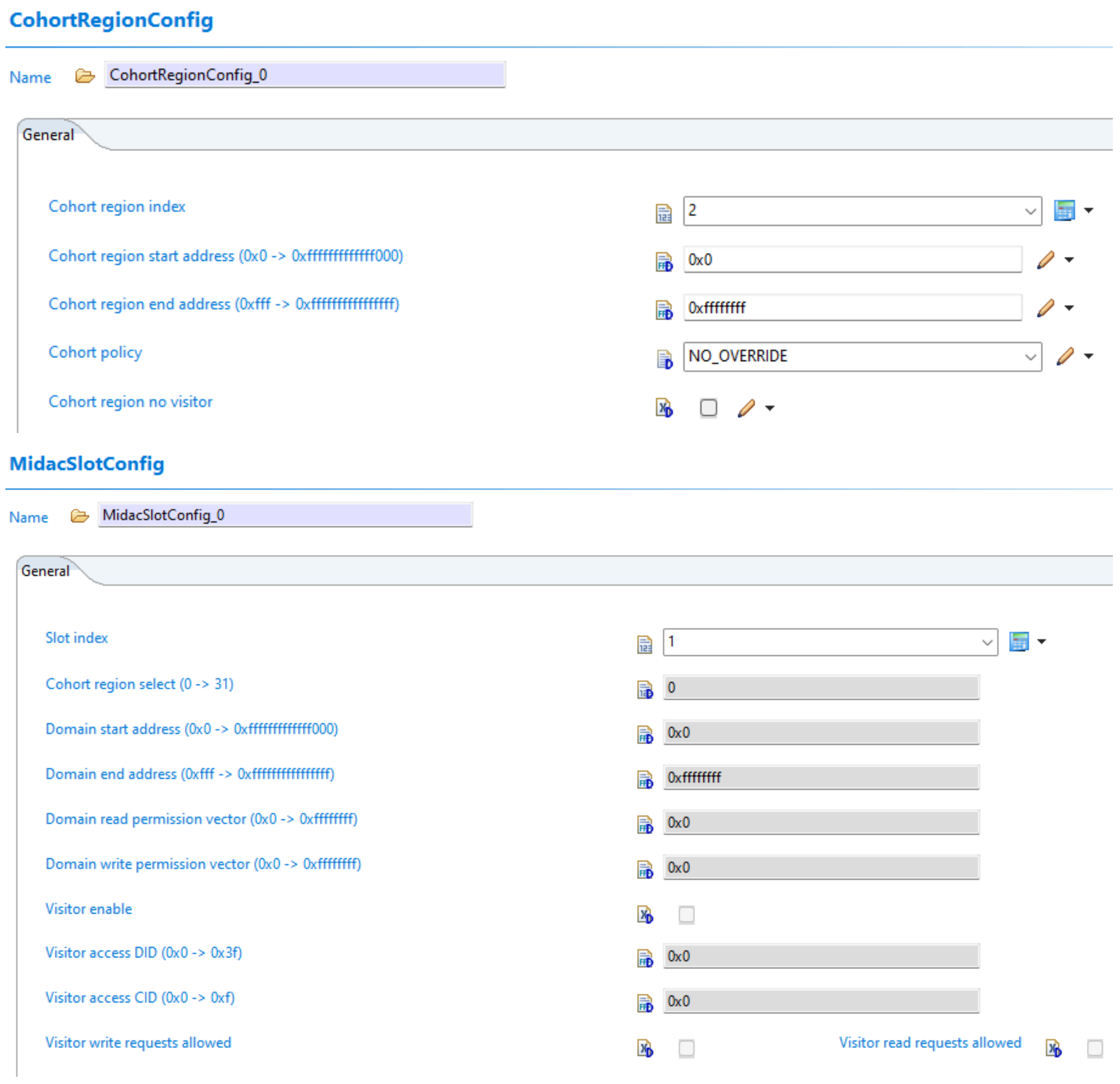
5.2.1 Share a memory among all domains

The objective is to grant Cohort 2 access to RTU1.CRAM6.

Steps are listed:

1. Navigate to the Cohorts configuration tab under the Fss_Rem_Pm module in the project Firmware_S32N55_FSS.
2. Change the Cohort Policy from OVERRIDE to NO_OVERRIDE in Cohort 0.
3. After the change, RTU1.CRAM6 can only be accessed by Cohort 0.
4. Add permission of RTU1.CRAM6 to Cohort 2. Go to Cohorts configuration(RTU0_AppCohort)->Subsystems configuration(RTU1)->MIDAC configuration(XRDC4_MIDAC12)->Cohort regions configuration. Add a new region entry:
   a. Cohort region index: 2(In this case, due to the automatic update mechanism, Region 2 is assigned to Cohort 2.)
   b. Cohort region start address: 0x0
   c. Cohort region end address: 0xFFFF_FFFF
   d. Cohort policy: NO_OVERRIDE
5. Add a new slot entry.
   a. Slot index: 1

Note: *The slot will automatically link to Region 2, as it belongs to Cohort 2. If Region 2 does not belong to Cohort 2, the slot becomes invalid.*



CohortRegionConfig

Name: CohortRegionConfig_0

General

Cohort region index	2
Cohort region start address (0x0 -> 0xfffffffffffff000)	0x0
Cohort region end address (0xfff -> 0xffffffffffffffff)	0xffffffff
Cohort policy	NO_OVERRIDE
Cohort region no visitor	☐

MidacSlotConfig

Name: MidacSlotConfig_0

General

Slot index	1
Cohort region select (0 -> 31)	0
Domain start address (0x0 -> 0xfffffffffffff000)	0x0
Domain end address (0xfff -> 0xffffffffffffffff)	0xffffffff
Domain read permission vector (0x0 -> 0xffffffff)	0x0
Domain write permission vector (0x0 -> 0xffffffff)	0x0
Visitor enable	☐
Visitor access DID (0x0 -> 0x3f)	0x0
Visitor access CID (0x0 -> 0xf)	0x0
Visitor write requests allowed	☐
Visitor read requests allowed	☐

Figure 23. Configuration for sharing memory inside one cohort in the Fss_Rem_Pm module

5.2.2 Assign a memory to one or multiple domains

Based on the above configurations, all domains in Cohort 2 have the read and write permissions for RTU1.CRAM6. Next, we will demonstrate how to control memory access for different domains, while also narrowing down the accessible memory regions.

The goal is to grant the AID 2.1 to access the memory, and memory range narrowing down with 0x3BF0_0000~0x3BFF_FFFF.

Steps are listed:

1. Navigate to the XRDC Memory Config tab under the Rem_Cm module in the project S32N5_RTU0_R52_0_VIP_App.
2. Add a new entry named with RemCmXrdcMemoryConfig_RTU1_CRAM6:
   a. Memory Region: RTU1_Xrdc_Axim_Sram6_Cram6
   b. Slot index: 1

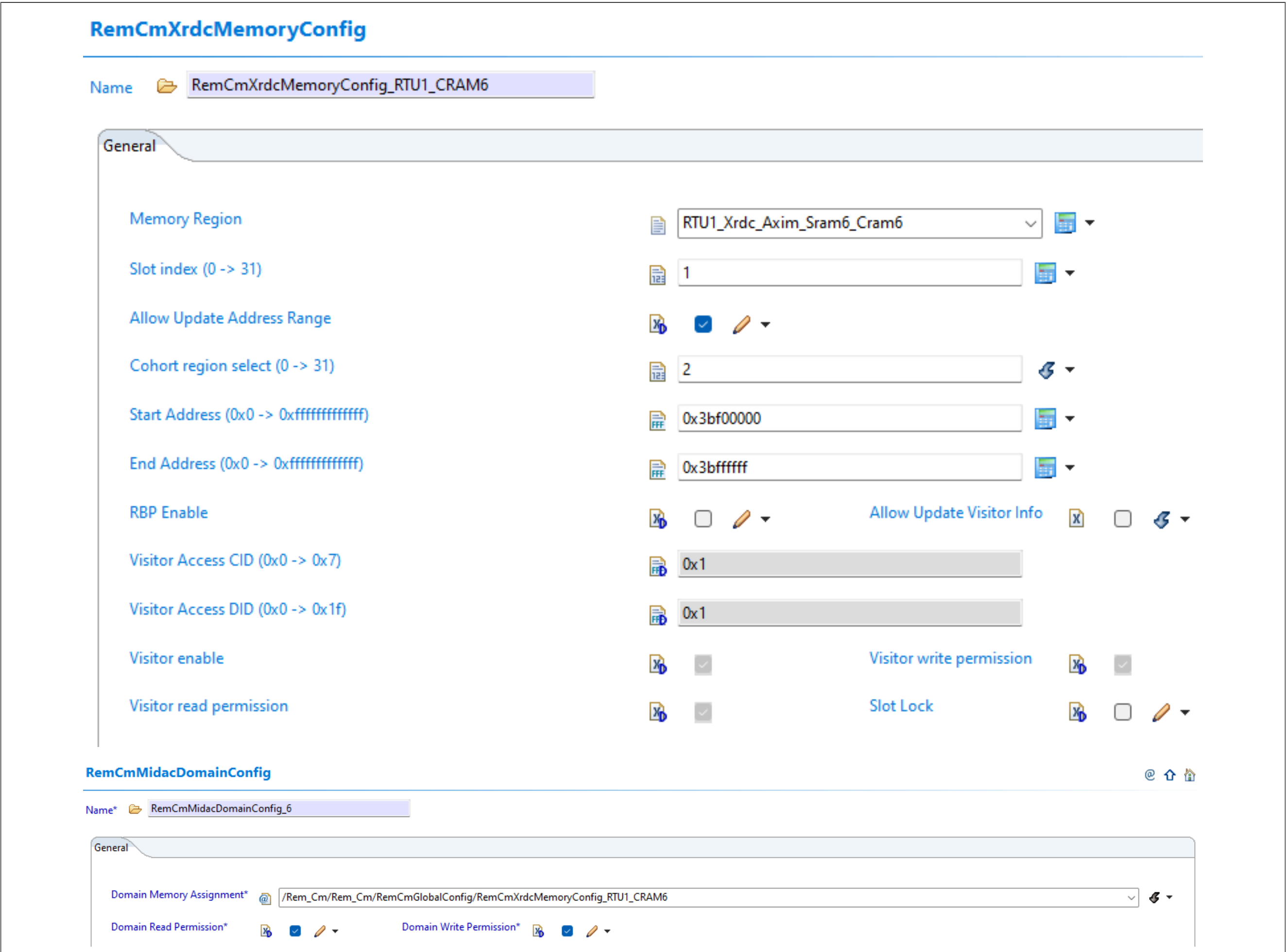
c. Allow Update Address Range: asserted
d. Cohort region select: 2
e. Start Address: 0x3BF0_0000
f. End Address: 0x3BFF_FFFF
g. RBP Enable: deasserted
h. Allow Update Visitor Info: deasserted

Note: *The selected region MUST be assigned to its own cohort, otherwise the slot will be excluded by deasserting SVLD.*

1. Navigate to the Domain Config tab.
2. Add RemCmMidacDomainConfig to Domain 1 and enable its read and write permissions.
   a. Open XRDC4_DOMAIN1 entry.
   b. Navigate to the RemCmMidacDomainConfig tab.
   c. Add an entry and link it to the Domain Memory Assignment of RTU1.CRAM6. Enable both read and write permissions

Note: *If a memory is never linked to any domain(without steps 3 and 4), it will deny all access attempts.*

Note: *If a memory requires access from multiple inner-cohort domains, you need to repeat Step 4 to configure each domain individually.*



RemCmXrdcMemoryConfig

Name: RemCmXrdcMemoryConfig_RTU1_CRAM6

General

Memory Region	RTU1_Xrdc_Axim_Sram6_Cram6
Slot index (0 -> 31)	1
Allow Update Address Range	✓
Cohort region select (0 -> 31)	2
Start Address (0x0 -> 0xffffffffffff)	0x3bf00000
End Address (0x0 -> 0xffffffffffff)	0x3bffffff
RBP Enable	☐
Allow Update Visitor Info	☐
Visitor Access CID (0x0 -> 0x7)	0x1
Visitor Access DID (0x0 -> 0x1f)	0x1
Visitor enable	✓
Visitor write permission	✓
Visitor read permission	✓
Slot Lock	☐

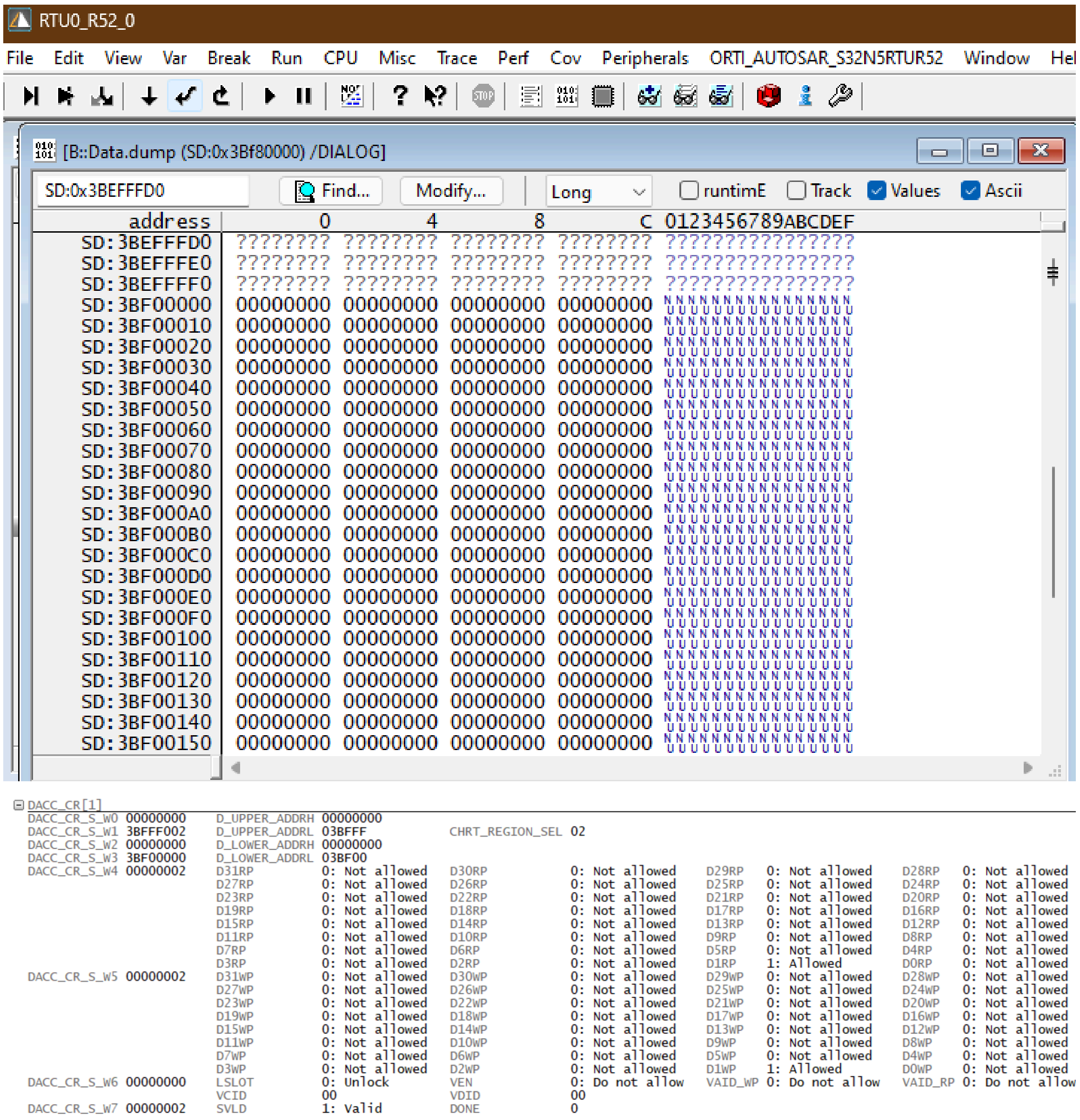
RemCmMidacDomainConfig

Name*: RemCmMidacDomainConfig_6

General

Domain Memory Assignment*	/Rem_Cm/Rem_Cm/RemCmGlobalConfig/RemCmXrdcMemoryConfig_RTU1_CRAM6
Domain Read Permission*	✓
Domain Write Permission*	✓

Figure 24. Configuration for sharing memory inside one cohort in the Rem_Cm module


RTU0_R52_0
File Edit View Var Break Run CPU Misc Trace Perf Cov Peripherals ORTI_AUTOSAR_S32N5RTUR52 Window Hel
[B::Data.dump (SD:0x3Bf80000) /DIALOG]
SD:0x3BEFFFD0
Find...
Modify...
Long
runtimE
Track
Values
Ascii

```
     address  0        4        8        C        0123456789ABCDEF
SD:3BEFFFD0   ???????? ???????? ???????? ???????? ????????????????
SD:3BEFFFE0   ???????? ???????? ???????? ???????? ????????????????
SD:3BEFFFF0   ???????? ???????? ???????? ???????? ????????????????
SD:3BF00000   00000000 00000000 00000000 00000000 ␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀
SD:3BF00010   00000000 00000000 00000000 00000000 ␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀
SD:3BF00020   00000000 00000000 00000000 00000000 ␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀
SD:3BF00030   00000000 00000000 00000000 00000000 ␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀
SD:3BF00040   00000000 00000000 00000000 00000000 ␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀
SD:3BF00050   00000000 00000000 00000000 00000000 ␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀
SD:3BF00060   00000000 00000000 00000000 00000000 ␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀
SD:3BF00070   00000000 00000000 00000000 00000000 ␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀
SD:3BF00080   00000000 00000000 00000000 00000000 ␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀
SD:3BF00090   00000000 00000000 00000000 00000000 ␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀
SD:3BF000A0   00000000 00000000 00000000 00000000 ␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀
SD:3BF000B0   00000000 00000000 00000000 00000000 ␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀
SD:3BF000C0   00000000 00000000 00000000 00000000 ␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀
SD:3BF000D0   00000000 00000000 00000000 00000000 ␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀
SD:3BF000E0   00000000 00000000 00000000 00000000 ␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀
SD:3BF000F0   00000000 00000000 00000000 00000000 ␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀
SD:3BF00100   00000000 00000000 00000000 00000000 ␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀
SD:3BF00110   00000000 00000000 00000000 00000000 ␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀
SD:3BF00120   00000000 00000000 00000000 00000000 ␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀
SD:3BF00130   00000000 00000000 00000000 00000000 ␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀
SD:3BF00140   00000000 00000000 00000000 00000000 ␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀
SD:3BF00150   00000000 00000000 00000000 00000000 ␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀
```

```
⊟ DACC_CR[1]
  DACC_CR_S_W0 00000000  D_UPPER_ADDRH 00000000
  DACC_CR_S_W1 3BFFF002  D_UPPER_ADDRL 03BFFF           CHRT_REGION_SEL 02
  DACC_CR_S_W2 00000000  D_LOWER_ADDRH 00000000
  DACC_CR_S_W3 3BF00000  D_LOWER_ADDRL 03BF00
  DACC_CR_S_W4 00000002  D31RP         0: Not allowed   D30RP           0: Not allowed   D29RP   0: Not allowed   D28RP   0: Not allowed
                         D27RP         0: Not allowed   D26RP           0: Not allowed   D25RP   0: Not allowed   D24RP   0: Not allowed
                         D23RP         0: Not allowed   D22RP           0: Not allowed   D21RP   0: Not allowed   D20RP   0: Not allowed
                         D19RP         0: Not allowed   D18RP           0: Not allowed   D17RP   0: Not allowed   D16RP   0: Not allowed
                         D15RP         0: Not allowed   D14RP           0: Not allowed   D13RP   0: Not allowed   D12RP   0: Not allowed
                         D11RP         0: Not allowed   D10RP           0: Not allowed   D9RP    0: Not allowed   D8RP    0: Not allowed
                         D7RP          0: Not allowed   D6RP            0: Not allowed   D5RP    0: Not allowed   D4RP    0: Not allowed
                         D3RP          0: Not allowed   D2RP            0: Not allowed   D1RP    1: Allowed       D0RP    0: Not allowed
  DACC_CR_S_W5 00000002  D31WP         0: Not allowed   D30WP           0: Not allowed   D29WP   0: Not allowed   D28WP   0: Not allowed
                         D27WP         0: Not allowed   D26WP           0: Not allowed   D25WP   0: Not allowed   D24WP   0: Not allowed
                         D23WP         0: Not allowed   D22WP           0: Not allowed   D21WP   0: Not allowed   D20WP   0: Not allowed
                         D19WP         0: Not allowed   D18WP           0: Not allowed   D17WP   0: Not allowed   D16WP   0: Not allowed
                         D15WP         0: Not allowed   D14WP           0: Not allowed   D13WP   0: Not allowed   D12WP   0: Not allowed
                         D11WP         0: Not allowed   D10WP           0: Not allowed   D9WP    0: Not allowed   D8WP    0: Not allowed
                         D7WP          0: Not allowed   D6WP            0: Not allowed   D5WP    0: Not allowed   D4WP    0: Not allowed
                         D3WP          0: Not allowed   D2WP            0: Not allowed   D1WP    1: Allowed       D0WP    0: Not allowed
  DACC_CR_S_W6 00000000  LSLOT         0: Unlock        VEN             0: Do not allow  VAID_WP 0: Do not allow  VAID_RP 0: Do not allow
                         VCID          00               VDID            00
  DACC_CR_S_W7 00000002  SVLD          1: Valid         DONE            0
```

Figure 25. Registers of MIDAC when assigning a memory to one domain

5.3 Share memory between multiple cohorts

PIDAC provides only one method for sharing a peripheral between two cohorts, which is through the visitor mechanism. In contrast, there are two approaches to sharing memory across multiple cohorts. One method is to assign different slot entries to each cohort, and the other is to use the visitor mechanism associated with the slot.

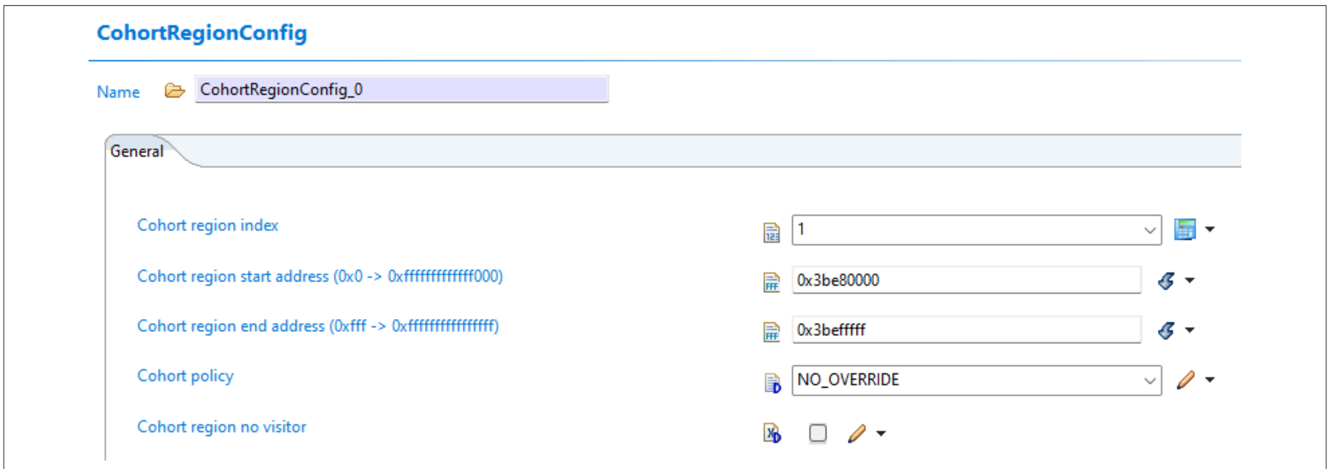
Based on the previous configuration, the CRS core cannot access RTU1.CRAM6. In this section, we will demonstrate how to grant the CRS core access to RTU1.CRAM6 using the two approaches mentioned above.

5.3.1 The method of the multiple slot entry

The goal is to grant CRS access to RTU1.CRAM6 with address range 0x3BE8_0000~0x3BEF_FFFF and 0x3BF0_0000~0x3BFF_FFFF.

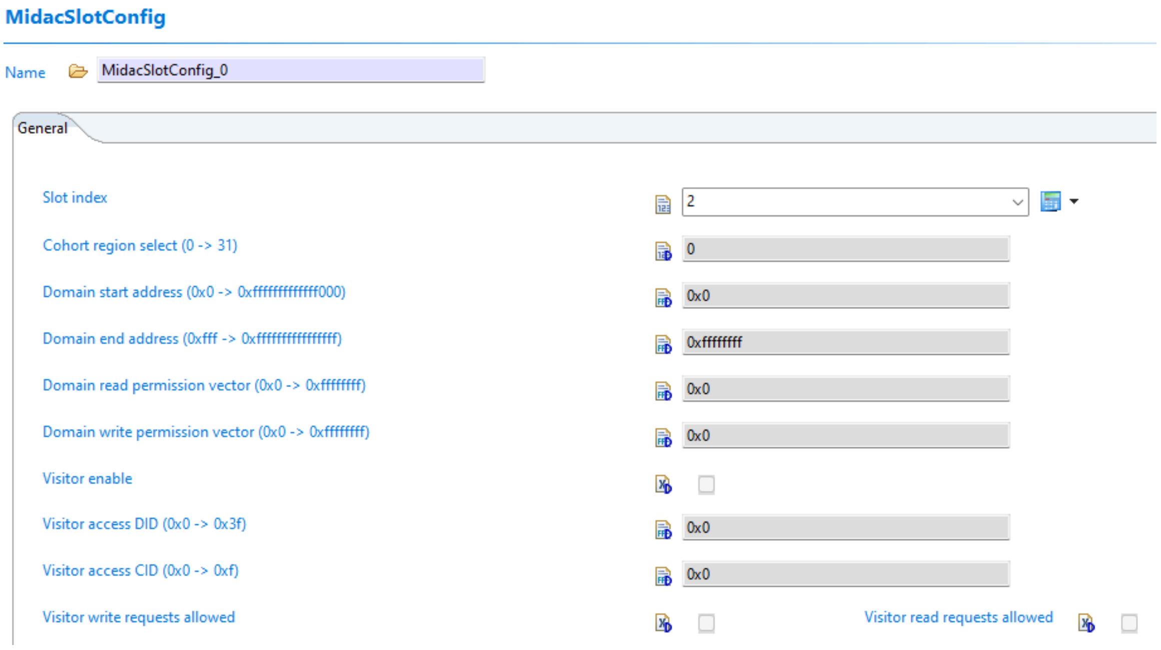
Steps are listed:

1. Navigate to the Cohorts configuration tab under the Fss_Rem_Pm module in the project Firmware_S32N55_FSS.
2. Add permission of RTU1.CRAM6 to Cohort 1. Go to Cohorts configuration(RTU0_AppCohort)-> Subsystems configuration(RTU1)->MIDAC configuration(XRDC4_MIDAC12)->Cohort regions configuration. Add a new region entry:
   a. Cohort region index: 1
   b. Cohort region start address: 0x3BE8_0000
   c. Cohort region end address: 0x3BEF_FFFF
   d. Cohort policy: NO_OVERRIDE
3. Add a new slot entry.
   a. Slot index: 2



Name: CohortRegionConfig_0

Cohort region index	1
Cohort region start address (0x0 -> 0xfffffffffffff000)	0x3be80000
Cohort region end address (0xfff -> 0xffffffffffffffff)	0x3beffff
Cohort policy	NO_OVERRIDE
Cohort region no visitor	☐



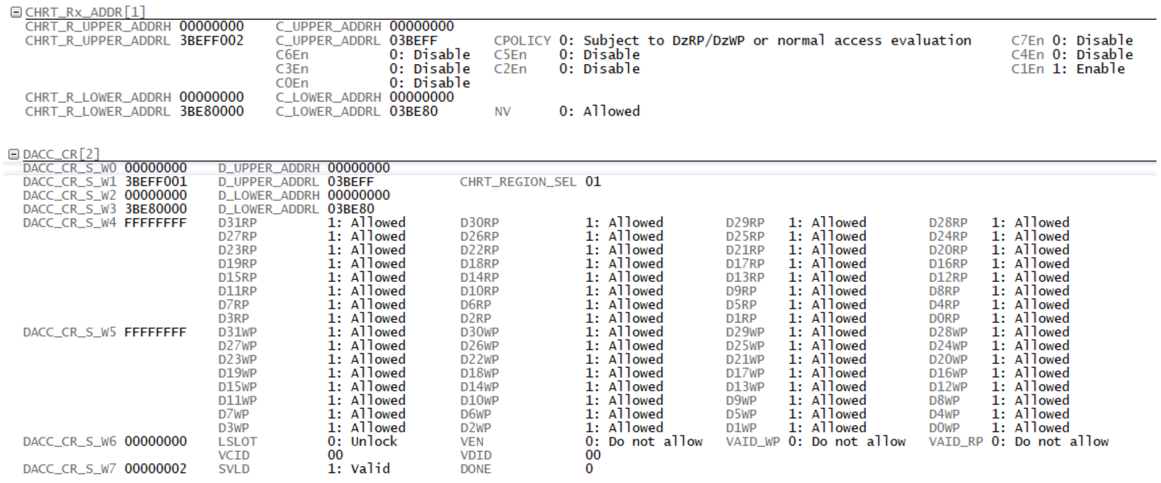
MidacSlotConfig

Name: MidacSlotConfig_0

General

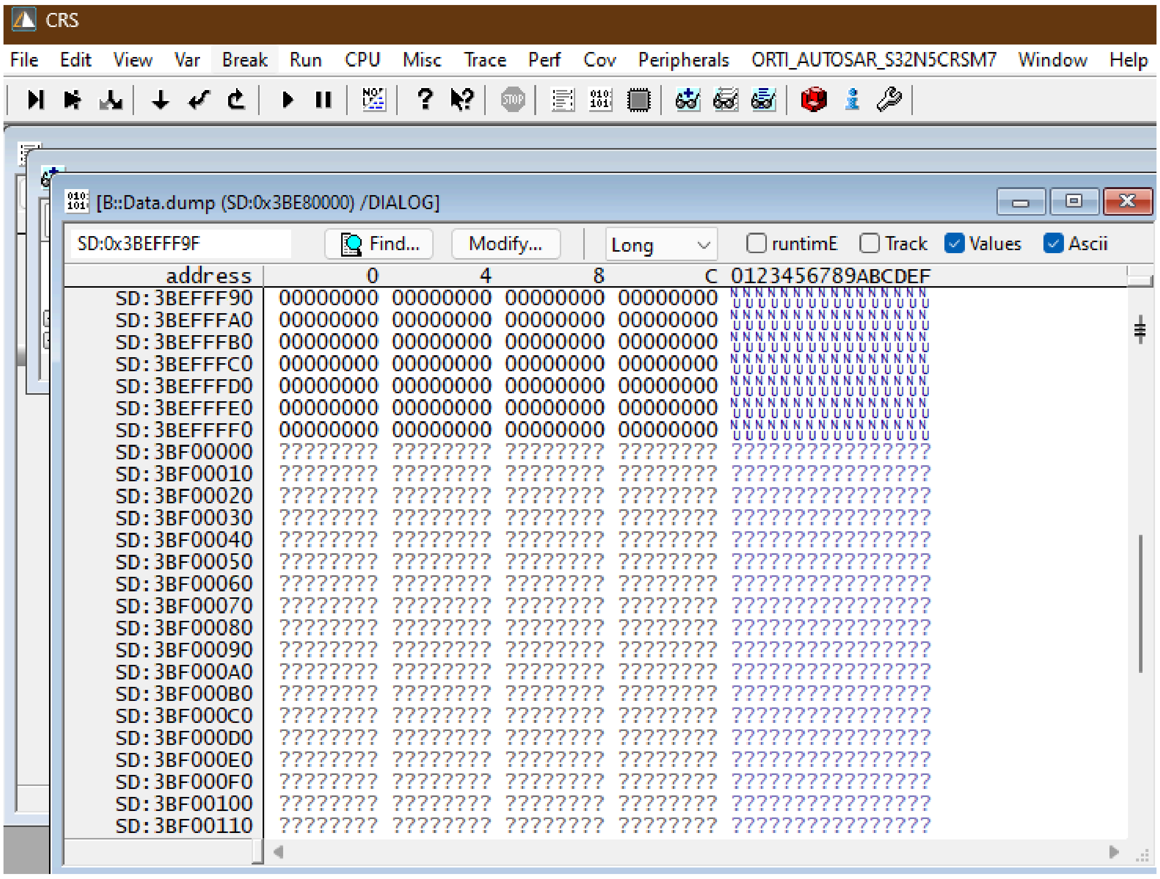
Slot index	2
Cohort region select (0 -> 31)	0
Domain start address (0x0 -> 0xfffffffffffff000)	0x0
Domain end address (0xfff -> 0xffffffffffffffff)	0xffffffff
Domain read permission vector (0x0 -> 0xffffffff)	0x0
Domain write permission vector (0x0 -> 0xffffffff)	0x0
Visitor enable	☐
Visitor access DID (0x0 -> 0x3f)	0x0
Visitor access CID (0x0 -> 0xf)	0x0
Visitor write requests allowed	☐
Visitor read requests allowed	☐

Figure 26. Configuration for sharing memory via the multiple slot entry



```
⊟ CHRT_Rx_ADDR[1]
  CHRT_R_UPPER_ADDRH 00000000   C_UPPER_ADDRH 00000000
  CHRT_R_UPPER_ADDRL 3BEFF002   C_UPPER_ADDRL 03BEFF     CPOLICY 0: Subject to DzRP/DzWP or normal access evaluation     C7En 0: Disable
                                C6En          0: Disable C5En    0: Disable                                             C4En 0: Disable
                                C3En          0: Disable C2En    0: Disable                                             C1En 1: Enable
                                C0En          0: Disable
  CHRT_R_LOWER_ADDRH 00000000   C_LOWER_ADDRH 00000000
  CHRT_R_LOWER_ADDRL 3BE80000   C_LOWER_ADDRL 03BE80     NV      0: Allowed

⊟ DACC_CR[2]
  DACC_CR_S_W0 00000000  D_UPPER_ADDRH 00000000
  DACC_CR_S_W1 3BEFF001  D_UPPER_ADDRL 03BEFF        CHRT_REGION_SEL 01
  DACC_CR_S_W2 00000000  D_LOWER_ADDRH 00000000
  DACC_CR_S_W3 3BE80000  D_LOWER_ADDRL 03BE80
  DACC_CR_S_W4 FFFFFFFF  D31RP         1: Allowed    D30RP           1: Allowed      D29RP    1: Allowed      D28RP    1: Allowed
                         D27RP         1: Allowed    D26RP           1: Allowed      D25RP    1: Allowed      D24RP    1: Allowed
                         D23RP         1: Allowed    D22RP           1: Allowed      D21RP    1: Allowed      D20RP    1: Allowed
                         D19RP         1: Allowed    D18RP           1: Allowed      D17RP    1: Allowed      D16RP    1: Allowed
                         D15RP         1: Allowed    D14RP           1: Allowed      D13RP    1: Allowed      D12RP    1: Allowed
                         D11RP         1: Allowed    D10RP           1: Allowed      D9RP     1: Allowed      D8RP     1: Allowed
                         D7RP          1: Allowed    D6RP            1: Allowed      D5RP     1: Allowed      D4RP     1: Allowed
                         D3RP          1: Allowed    D2RP            1: Allowed      D1RP     1: Allowed      D0RP     1: Allowed
  DACC_CR_S_W5 FFFFFFFF  D31WP         1: Allowed    D30WP           1: Allowed      D29WP    1: Allowed      D28WP    1: Allowed
                         D27WP         1: Allowed    D26WP           1: Allowed      D25WP    1: Allowed      D24WP    1: Allowed
                         D23WP         1: Allowed    D22WP           1: Allowed      D21WP    1: Allowed      D20WP    1: Allowed
                         D19WP         1: Allowed    D18WP           1: Allowed      D17WP    1: Allowed      D16WP    1: Allowed
                         D15WP         1: Allowed    D14WP           1: Allowed      D13WP    1: Allowed      D12WP    1: Allowed
                         D11WP         1: Allowed    D10WP           1: Allowed      D9WP     1: Allowed      D8WP     1: Allowed
                         D7WP          1: Allowed    D6WP            1: Allowed      D5WP     1: Allowed      D4WP     1: Allowed
                         D3WP          1: Allowed    D2WP            1: Allowed      D1WP     1: Allowed      D0WP     1: Allowed
  DACC_CR_S_W6 00000000  LSLOT         0: Unlock     VEN             0: Do not allow VAID_WP 0: Do not allow VAID_RP 0: Do not allow
                         VCID          00            VDID            00
  DACC_CR_S_W7 00000002  SVLD          1: Valid      DONE            0
```


CRS
File Edit View Var Break Run CPU Misc Trace Perf Cov Peripherals ORTI_AUTOSAR_S32N5CRSM7 Window Help
[B::Data.dump (SD:0x3BE80000) /DIALOG]
SD:0x3BEFFF9F
Find...
Modify...
Long
runtimE
Track
Values
Ascii

```
         address  0        4        8        C        0123456789ABCDEF
     SD:3BEFFF90  00000000 00000000 00000000 00000000
     SD:3BEFFFA0  00000000 00000000 00000000 00000000
     SD:3BEFFFB0  00000000 00000000 00000000 00000000
     SD:3BEFFFC0  00000000 00000000 00000000 00000000
     SD:3BEFFFD0  00000000 00000000 00000000 00000000
     SD:3BEFFFE0  00000000 00000000 00000000 00000000
     SD:3BEFFFF0  00000000 00000000 00000000 00000000
     SD:3BF00000  ???????? ???????? ???????? ???????? ????????????????
     SD:3BF00010  ???????? ???????? ???????? ???????? ????????????????
     SD:3BF00020  ???????? ???????? ???????? ???????? ????????????????
     SD:3BF00030  ???????? ???????? ???????? ???????? ????????????????
     SD:3BF00040  ???????? ???????? ???????? ???????? ????????????????
     SD:3BF00050  ???????? ???????? ???????? ???????? ????????????????
     SD:3BF00060  ???????? ???????? ???????? ???????? ????????????????
     SD:3BF00070  ???????? ???????? ???????? ???????? ????????????????
     SD:3BF00080  ???????? ???????? ???????? ???????? ????????????????
     SD:3BF00090  ???????? ???????? ???????? ???????? ????????????????
     SD:3BF000A0  ???????? ???????? ???????? ???????? ????????????????
     SD:3BF000B0  ???????? ???????? ???????? ???????? ????????????????
     SD:3BF000C0  ???????? ???????? ???????? ???????? ????????????????
     SD:3BF000D0  ???????? ???????? ???????? ???????? ????????????????
     SD:3BF000E0  ???????? ???????? ???????? ???????? ????????????????
     SD:3BF000F0  ???????? ???????? ???????? ???????? ????????????????
     SD:3BF00100  ???????? ???????? ???????? ???????? ????????????????
     SD:3BF00110  ???????? ???????? ???????? ???????? ????????????????
```

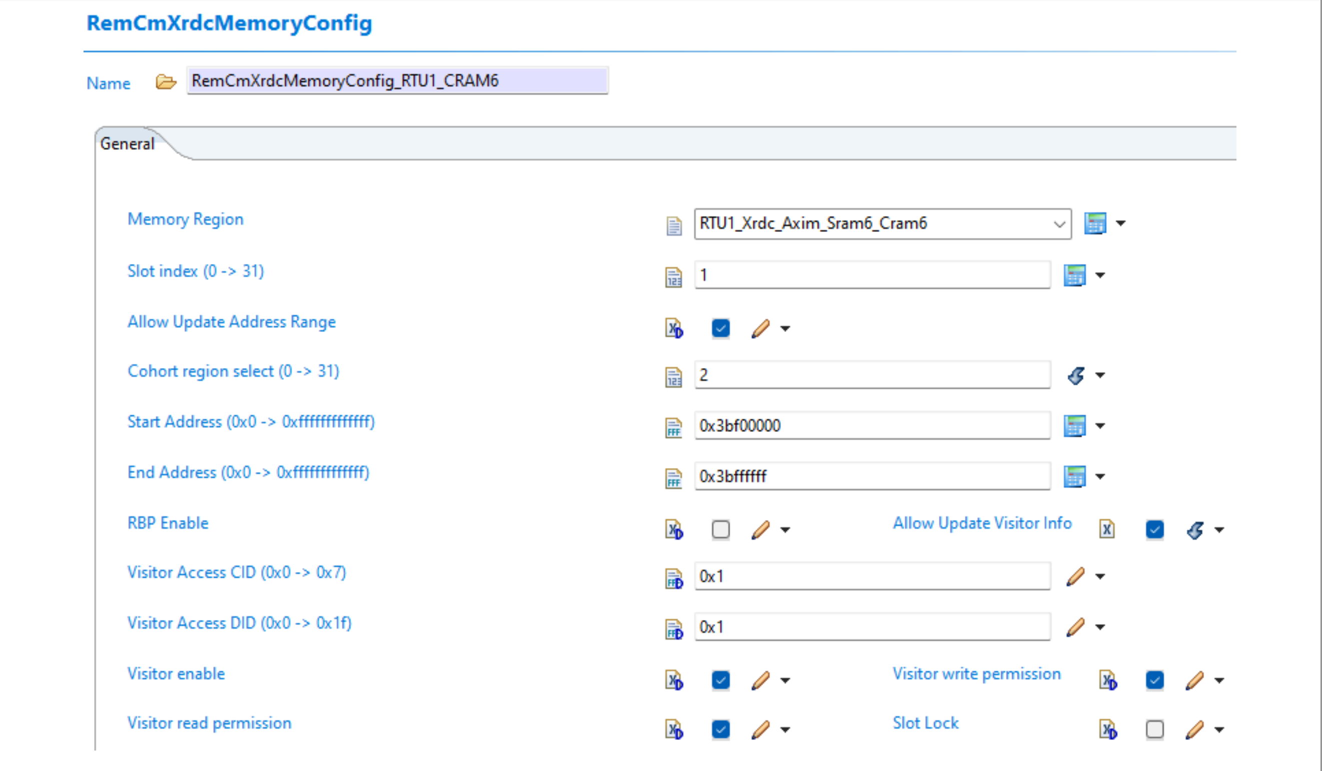
Figure 27. Registers of MIDAC for sharing memory via the multiple slot entry

5.3.2 The method of the visitor mechanism

After completing the configuration in the previous section, the CRS core can access RTU1.CRAM6 within the address range 0x3BE8_0000 to 0x3BEF_FFFF. Next, we will demonstrate how to share the address range 0x3BF0_0000 to 0x3BFF_FFFF using the visitor mechanism.

Steps are listed:

1. Navigate to the Fss_Rem_Pm module. Go to Cohorts configuration(RTU0_AppCohort)-> Subsystems configuration(RTU1)->MIDAC configuration(XRDC4_MIDAC12)->Cohort regions configuration(Cohort RegionConfig_0). Ensure that the "Cohort Region No Visitor" attribute is deasserted.
2. Navigate to the XRDC Memory Config tab under the Rem_Cm module in the project S32N5_RTU0_R52_0_VIP_App.
3. Modify the RemCmXrdcMemoryConfig_RTU1_CRAM6 entry:
   a. Allow Update Visitor Info: asserted
   b. Visitor Access CID: 0x1
   c. Visitor Access DID: 0x1
   d. Visitor enable: asserted
   e. Visitor write permission: asserted
   f. Visitor read permission: asserted



RemCmXrdcMemoryConfig

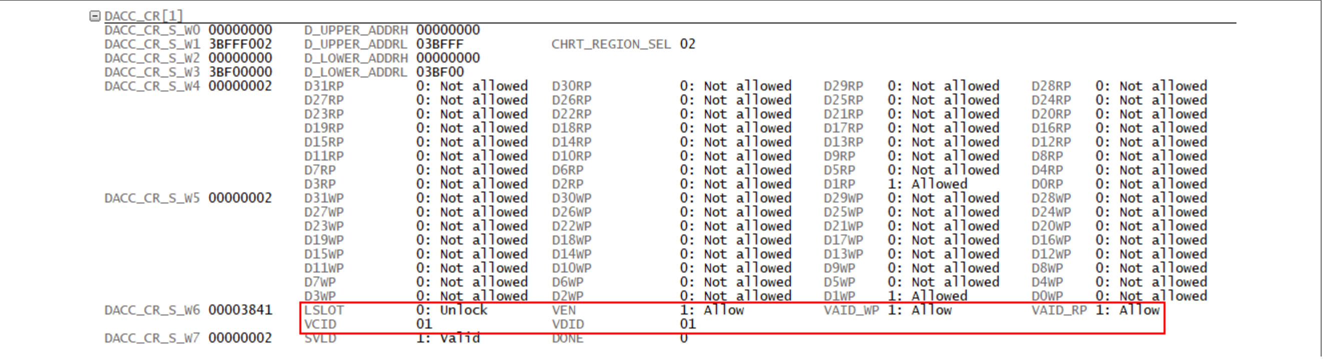
Name: RemCmXrdcMemoryConfig_RTU1_CRAM6

General

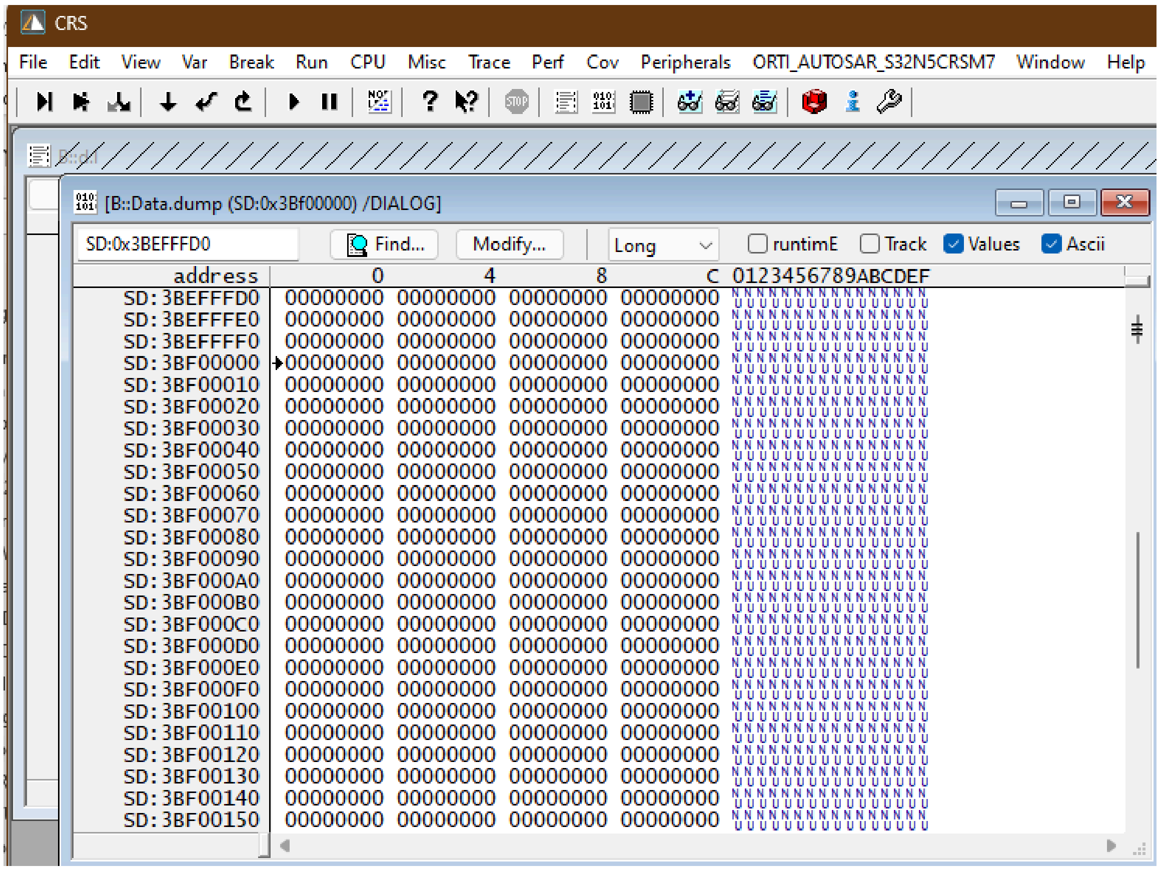
Memory Region	RTU1_Xrdc_Axim_Sram6_Cram6
Slot index (0 -> 31)	1
Allow Update Address Range	✓
Cohort region select (0 -> 31)	2
Start Address (0x0 -> 0xffffffffffff)	0x3bf00000
End Address (0x0 -> 0xffffffffffff)	0x3bffffff
RBP Enable	☐
Allow Update Visitor Info	✓
Visitor Access CID (0x0 -> 0x7)	0x1
Visitor Access DID (0x0 -> 0x1f)	0x1
Visitor enable	✓
Visitor write permission	✓
Visitor read permission	✓
Slot Lock	☐

Figure 28. Configuration for sharing memory via the visitor mechanism

When the CM of Cohort 2 completes the XRDC initialization by calling the Rem_Cm_Init() function, the CRS core is granted access to RTU1.CRAM6 within the address range 0x3BF0_0000 to 0x3BFF_FFFF via the visitor mechanism.



```
DACC_CR[1]
DACC_CR_S_W0 00000000   D_UPPER_ADDRH 00000000
DACC_CR_S_W1 3BFFF002   D_UPPER_ADDRL 03BFFF        CHRT_REGION_SEL 02
DACC_CR_S_W2 00000000   D_LOWER_ADDRH 00000000
DACC_CR_S_W3 3BF00000   D_LOWER_ADDRL 03BF00
DACC_CR_S_W4 00000002   D31RP         0: Not allowed  D30RP         0: Not allowed   D29RP    0: Not allowed   D28RP    0: Not allowed
                        D27RP         0: Not allowed  D26RP         0: Not allowed   D25RP    0: Not allowed   D24RP    0: Not allowed
                        D23RP         0: Not allowed  D22RP         0: Not allowed   D21RP    0: Not allowed   D20RP    0: Not allowed
                        D19RP         0: Not allowed  D18RP         0: Not allowed   D17RP    0: Not allowed   D16RP    0: Not allowed
                        D15RP         0: Not allowed  D14RP         0: Not allowed   D13RP    0: Not allowed   D12RP    0: Not allowed
                        D11RP         0: Not allowed  D10RP         0: Not allowed   D9RP     0: Not allowed   D8RP     0: Not allowed
                        D7RP          0: Not allowed  D6RP          0: Not allowed   D5RP     0: Not allowed   D4RP     0: Not allowed
                        D3RP          0: Not allowed  D2RP          0: Not allowed   D1RP     1: Allowed       D0RP     0: Not allowed
DACC_CR_S_W5 00000002   D31WP         0: Not allowed  D30WP         0: Not allowed   D29WP    0: Not allowed   D28WP    0: Not allowed
                        D27WP         0: Not allowed  D26WP         0: Not allowed   D25WP    0: Not allowed   D24WP    0: Not allowed
                        D23WP         0: Not allowed  D22WP         0: Not allowed   D21WP    0: Not allowed   D20WP    0: Not allowed
                        D19WP         0: Not allowed  D18WP         0: Not allowed   D17WP    0: Not allowed   D16WP    0: Not allowed
                        D15WP         0: Not allowed  D14WP         0: Not allowed   D13WP    0: Not allowed   D12WP    0: Not allowed
                        D11WP         0: Not allowed  D10WP         0: Not allowed   D9WP     0: Not allowed   D8WP     0: Not allowed
                        D7WP          0: Not allowed  D6WP          0: Not allowed   D5WP     0: Not allowed   D4WP     0: Not allowed
                        D3WP          0: Not allowed  D2WP          0: Not allowed   D1WP     1: Allowed       D0WP     0: Not allowed
DACC_CR_S_W6 00003841   LSLOT         0: Unlock       VEN           1: Allow         VAID_WP  1: Allow         VAID_RP  1: Allow
                        VCID          01              VDID          01
DACC_CR_S_W7 00000002   SVLD          1: Valid        DONE          0
```


CRS
File Edit View Var Break Run CPU Misc Trace Perf Cov Peripherals ORTI_AUTOSAR_S32N5CRSM7 Window Help
B::dd
Find...
Modify...

```
[B::Data.dump (SD:0x3Bf00000) /DIALOG]
SD:0x3BEFFFD0   Long   [ ] runtimE  [ ] Track  [✓] Values  [✓] Ascii
      address         0        4        8        C  0123456789ABCDEF
  SD:3BEFFFD0  00000000 00000000 00000000 00000000  ␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀
  SD:3BEFFFE0  00000000 00000000 00000000 00000000  ␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀
  SD:3BEFFFF0  00000000 00000000 00000000 00000000  ␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀
  SD:3BF00000 >00000000 00000000 00000000 00000000  ␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀
  SD:3BF00010  00000000 00000000 00000000 00000000  ␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀
  SD:3BF00020  00000000 00000000 00000000 00000000  ␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀
  SD:3BF00030  00000000 00000000 00000000 00000000  ␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀
  SD:3BF00040  00000000 00000000 00000000 00000000  ␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀
  SD:3BF00050  00000000 00000000 00000000 00000000  ␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀
  SD:3BF00060  00000000 00000000 00000000 00000000  ␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀
  SD:3BF00070  00000000 00000000 00000000 00000000  ␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀
  SD:3BF00080  00000000 00000000 00000000 00000000  ␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀
  SD:3BF00090  00000000 00000000 00000000 00000000  ␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀
  SD:3BF000A0  00000000 00000000 00000000 00000000  ␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀
  SD:3BF000B0  00000000 00000000 00000000 00000000  ␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀
  SD:3BF000C0  00000000 00000000 00000000 00000000  ␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀
  SD:3BF000D0  00000000 00000000 00000000 00000000  ␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀
  SD:3BF000E0  00000000 00000000 00000000 00000000  ␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀
  SD:3BF000F0  00000000 00000000 00000000 00000000  ␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀
  SD:3BF00100  00000000 00000000 00000000 00000000  ␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀
  SD:3BF00110  00000000 00000000 00000000 00000000  ␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀
  SD:3BF00120  00000000 00000000 00000000 00000000  ␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀
  SD:3BF00130  00000000 00000000 00000000 00000000  ␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀
  SD:3BF00140  00000000 00000000 00000000 00000000  ␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀
  SD:3BF00150  00000000 00000000 00000000 00000000  ␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀␀
```

Figure 29. Registers of MIDAC for sharing memory via the visitor mechanism

6 Acronyms and abbreviations

Table 9. Acronyms and abbreviations

Abbreviation	Explanation
UENV	Uniform Environment
PM	Partition Manager
CM	Cohort Manager
FSS	Foundation Subsystem
PaCo	Partition Contract
CoDef	Cohort Definition
PIDAC	Peripheral Identifier Access Control
MIDAC	Memory Identifier Access Control
AIDA	Access ID assignment
AID	Access Identifier
SBC	System Base Cohort

7 References

- S32N55 Reference Manual.pdf
- FSS_FW_UserManual.pdf

8 Note about the source code in the document

The example code shown in this document has the following copyright and BSD-3-Clause license:

9 Revision history

Table 10. Revision history

Document ID	Release date	Description
AN14892 v.1.0	17 Dec 2025	Initial release

Legal information

Tables

Figures

Contents

Date of release: 17 December 2025
Document identifier: AN14892