

## 2 Memory and bus architecture

### 2.1 System architecture

An AXI bus matrix, two AHB bus matrices and bus bridges allow interconnecting bus masters with bus slaves, as illustrated in [Table 3](#) and [Figure 1](#).

**Table 3. Bus-master-to-bus-slave interconnect**

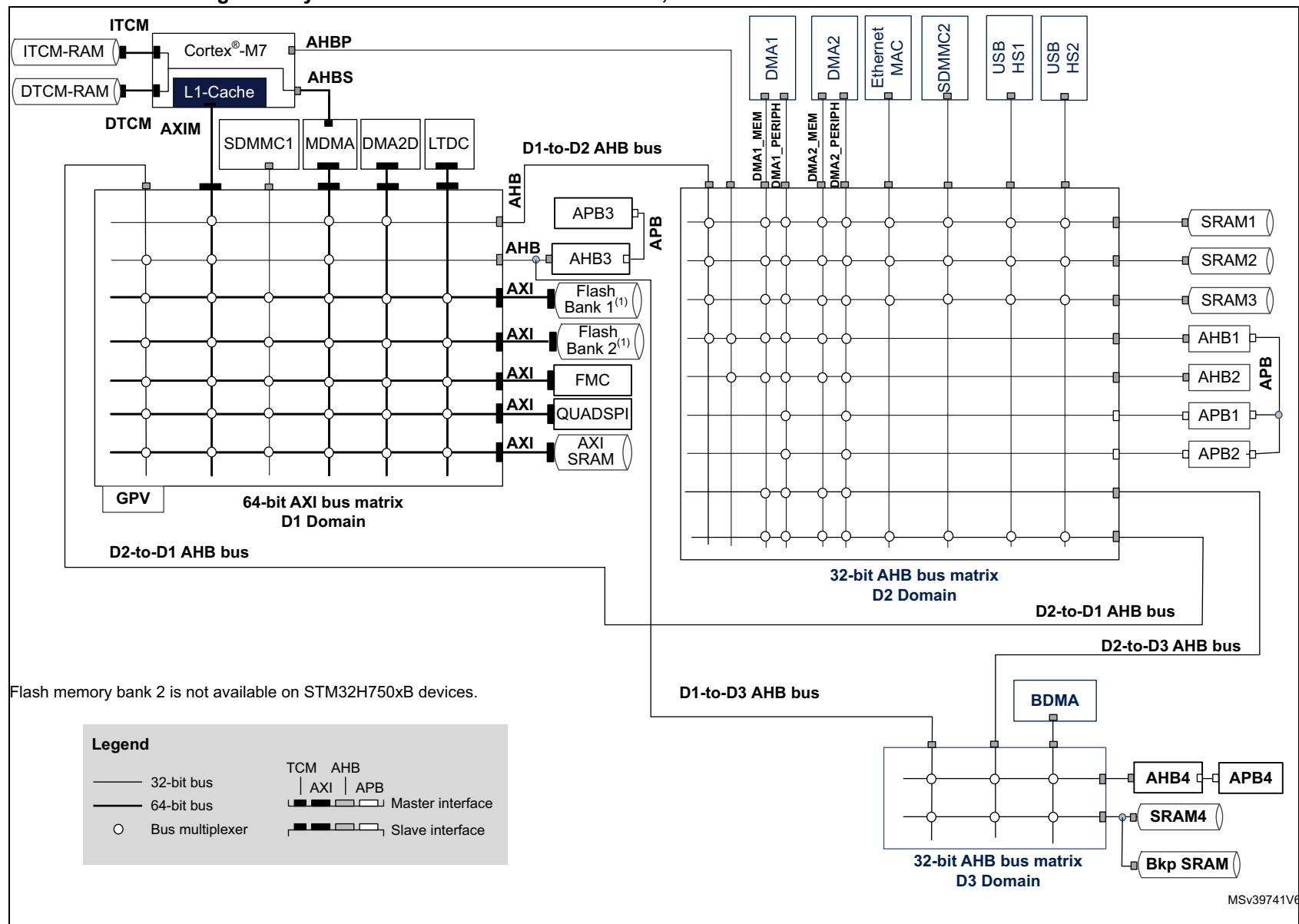
		Bus master / type <sup>(1)</sup>																	
		Cortex-M7 - AXIM	Cortex-M7 - AHBP	Cortex-M7 - ITCM	Cortex-M7 - DTCM	SDMMC1	MDMA - AXI	MDMA - AHBS	DMA2D	LTDC	DMA1 - MEM	DMA1 - PERIPH	DMA2 - MEM	DMA2 - PERIPH	Eth. MAC - AHB	SDMMC2 - AHB	USBHS1 - AHB	USBHS2 - AHB	BDMA - AHB
Bus slave / type <sup>(1)</sup>		Interconnect path and type <sup>(2)</sup>																	
<b>ITCM</b>	-	-	X	-	-	-	X	-	-	-	-	-	-	-	-	-	-	-	
<b>DTCM</b>	-	-	-	<b>X</b>	-	-	X	-	-	-	-	-	-	-	-	-	-	-	
AHB3 peripherals	X	-	-	-	-	-	X	-	-	X	X	X	X	X	X	X	X	X	
APB3 peripherals	X	-	-	-	-	-	X	-	-	X	X	X	X	X	X	X	X	X	
<b>Flash bank 1</b>	<b>X</b>	-	-	-	X	<b>X</b>	-	X	X	X	X	X	X	X	X	X	X	-	
<b>Flash bank 2<sup>(3)</sup></b>	X	-	-	-	X	<b>X</b>	-	X	X	X	X	X	X	X	X	X	X	-	
<b>AXI SRAM</b>	X	-	-	-	X	<b>X</b>	-	X	X	X	X	X	X	X	X	X	X	-	
<b>QUADSPI</b>	X	-	-	-	X	<b>X</b>	-	X	X	X	X	X	X	X	X	X	X	-	
<b>FMC</b>	<b>X</b>	-	-	-	X	<b>X</b>	-	X	X	X	X	X	X	X	X	X	X	-	
SRAM 1	X	-	-	-	-	X	-	X	-	X	X	X	X	X	X	X	X	-	
SRAM 2	X	-	-	-	-	X	-	X	-	X	X	X	X	X	X	X	X	-	
SRAM 3	X	-	-	-	-	X	-	X	-	X	X	X	X	X	X	X	X	-	
AHB1 peripherals	-	X	-	-	-	X	-	X	-	X	X	X	X	X	-	-	-	-	
APB1 peripherals	-	X	-	-	-	X	-	X	-	X	X	X	X	X	-	-	-	-	
AHB2 peripherals	-	X	-	-	-	-	-	-	-	X	X	X	X	-	-	-	-	-	
APB2 peripherals	-	X	-	-	-	X	-	X	-	X	X	X	X	-	-	-	-	-	
AHB4 peripherals	X	-	-	-	-	X	-	-	-	X	X	X	X	X	X	X	X	X	
APB4 peripherals	X	-	-	-	-	X	-	-	-	X	X	X	X	X	X	X	X	X	
SRAM4	X	-	-	-	-	X	-	-	-	X	X	X	X	X	X	X	X	X	
Backup RAM	X	-	-	-	-	X	-	-	-	X	X	X	X	X	X	X	X	X	

1. **Bold** font type denotes 64-bit bus, plain type denotes 32-bit bus.

2. “X” = access possible, “-” = access not possible, shading = access useful/usable.

3. Bank 2 is not available on STM32H750xB devices.

**Figure 1. System architecture for STM32H742xx, STM32H743/53xx and STM32H750xB devices**



## 2.1.1 Bus matrices

### AXI bus matrix in D1 domain

The D1 domain multi AXI bus matrix ensures and arbitrates concurrent accesses from multiple masters to multiple slaves. This allows efficient simultaneous operation of high-speed peripherals.

The arbitration uses a round-robin algorithm with QoS capability.

Refer to [Section 2.2: AXI interconnect matrix \(AXIM\)](#) for more information on AXI interconnect.

### AHB bus matrices in D2 and D3 domains

The AHB bus matrices in D2 and D3 domains ensure and arbitrate concurrent accesses from multiple masters to multiple slaves. This allows efficient simultaneous operation of high-speed peripherals.

The arbitration uses a round-robin algorithm.

## 2.1.2 TCM buses

The DTCM and ITCM (data and instruction tightly coupled RAMs) are connected through dedicated TCM buses directly to the Cortex-M7 core. The MDMA controller can access the DTCM and ITCM through AHBS, a specific CPU slave AHB. The ITCM is accessed by Cortex-M7 at CPU clock speed, with zero wait states.

## 2.1.3 Bus-to-bus bridges

To allow peripherals with different types of buses to communicate together, there is a number of bus-to-bus bridges in the system.

The AHB/APB bridges in D1 and D3 domains allow connecting peripherals on APB3 and APB4 to AHB3 and AHB4, respectively. The AHB/APB bridges in D2 domain allow peripherals on APB1 and APB2 to connect to AHB1. These AHB/APB bridges provide full synchronous interfacing, which allows the APB peripherals to operate with clocks independent of AHB that they connect to.

The AHB/APB bridges also allow APB1 and APB2 peripherals to connect to DMA1 and DMA2 peripheral buses, respectively, without transiting through AHB1.

The AHB/APB bridges convert 8-bit / 16-bit APB data to 32-bit AHB data, by replicating it to the three upper bytes / the upper half-word of the 32-bit word.

The AXI bus matrix incorporates AHB/AXI bus bridge functionality on its slave bus interfaces. The AXI/AHB bus bridges on its master interfaces marked as 32-bit in [Figure 1](#) are outside the matrix.

The Cortex-M7 CPU provides AHB/TCM-bus (ITCM and DTCM buses) translation from its AHBS slave AHB, allowing the MDMA controller to access the ITCM and DTCM.

## 2.1.4 Inter-domain buses

### D2-to-D1 AHB

This 32-bit bus connects the D2 domain to the AXI bus matrix in the D1 domain. It allows bus masters in the D2 domain to access resources (bus slaves) in the D1 domain and indirectly, via the D1-to-D3 AHB, in the D3 domain.

### D1-to-D2 AHB

This 32-bit bus connects the D1 domain to the D2 domain AHB bus matrix. It allows bus masters in the D1 domain to access resources (bus slaves) in the D2 domain.

### D1-to-D3 AHB

This 32-bit bus connects the D1 domain to the D3 domain AHB bus matrix. It allows bus masters in the D1 domain to access resources (bus slaves) in the D3 domain.

### D2-to-D3 AHB

This 32-bit bus connects the D2 domain to the D3 domain AHB bus matrix. It allows bus masters in the D2 domain to access resources (bus slaves) in the D3 domain.

## 2.1.5 CPU buses

### Cortex®-M7 AXIM bus

The Cortex®-M7 CPU uses the 64-bit AXIM bus to access all memories (excluding ITCM, and DTCM) and AHB3, AHB4, APB3 and APB4 peripherals (excluding AHB1, APB1 and APB2 peripherals).

The AXIM bus connects the CPU to the AXI bus matrix in the D1 domain.

### Cortex®-M7 ITCM bus

The Cortex®-M7 CPU uses the 64-bit ITCM bus for fetching instructions from and accessing data in the ITCM.

### Cortex®-M7 DTCM bus

The Cortex®-M7 CPU uses the 2x32-bit DTCM bus for accessing data in the DTCM. The 2x32-bit DTCM bus allows load/load and load/store instruction pairs to be dual-issued on the DTCM memory. It can also fetch instructions.

### Cortex®-M7 AHBS bus

The Cortex®-M7 CPU uses the 32-bit AHBS slave bus to allow the MDMA controller to access the ITCM and the DTCM.

### Cortex®-M7 AHBP bus

The Cortex®-M7 CPU uses the 32-bit AHBP bus for accessing AHB1, AHB2, APB1 and APB2 peripherals via the AHB bus matrix in the D2 domain.

## 2.1.6 Bus master peripherals

### SDMMC1

The SDMMC1 uses a 32-bit bus, connected to the AXI bus matrix, through which it can access internal AXI SRAM and Flash memories, and external memories through the Quad-SPI controller and the FMC.

### SDMMC2

The SDMMC2 uses a 32-bit bus, connected to the AHB bus matrix in D2 domain. Through the system bus matrices, it can access the internal AXI SRAM, SRAM1, SRAM2, SRAM3 and Flash memories, and external memories through the Quad-SPI controller and the FMC.

### MDMA controller

The MDMA controller has two bus masters: an AXI 64-bit bus, connected to the AXI bus matrix and an AHB 32-bit bus connected to the Cortex-M7 AHBS slave bus.

The MDMA is optimized for DMA data transfers between memories since it supports linked list transfers that allow performing a chained list of transfers without the need for CPU intervention. Through the system bus matrices and the Cortex-M7 AHBS slave bus, the MDMA can access all internal and external memories through the Quad-SPI controller and the FMC.

### DMA1 and DMA2 controllers

The DMA1 and DMA2 controllers have two 32-bit buses - memory bus and peripheral bus, connected to the AHB bus matrix in D2 domain.

The memory bus allows DMA data transfers between memories. Through the system bus matrices, the memory bus can access all internal memories except ITCM and DTCM, and external memories through the Quad-SPI controller and the FMC.

The peripheral bus allows DMA data transfers between two peripherals, between two memories or between a peripheral and a memory. Through the system bus matrices, the peripheral bus can access all internal memories except ITCM and DTCM, external memories through the Quad-SPI controller and the FMC, and all AHB and APB peripherals. A direct access to APB1 and APB2 is available, without passing through AHB1. Direct path to APB1 and APB2 bridges allows reducing the bandwidth usage on AHB1 bus by improving data treatment efficiency for APB and AHB peripherals.

### BDMA controller

The BDMA controller uses a 32-bit bus, connected to the AHB bus matrix in D3 domain, for DMA data transfers between two peripherals, between two memories or between a peripheral and a memory. BDMA transfers are limited to the D3 domain resources. It can access the internal SRAM4, backup RAM, and AHB4 and APB4 peripherals through the AHB bus matrix in the D3 domain.

### Chrom-Art Accelerator (DMA2D)

The DMA2D graphics accelerator uses a 64-bit bus, connected to the AXI bus matrix. Through the system bus matrices, internal AXI SRAM, SRAM1, SRAM2, SRAM3 and Flash memories, and external memories through the Quad-SPI controller and the FMC.

### LCD-TFT controller (LTDC)

The LCD-TFT display controller, LTDC, uses a 64-bit bus, connected to the AXI bus matrix, through which it can access internal AXI SRAM and Flash memories, and external memories through the Quad-SPI controller and the FMC.

### Ethernet MAC

The Ethernet MAC uses a 32-bit bus, connected to the AHB bus matrix in the D2 domain. Through the system bus matrices, it can access all internal memories except ITCM and DTCM, and external memories through the Quad-SPI controller and the FMC.

### USBHS1 and USBHS2 peripherals

The USBHS1 and USBHS2 peripherals use 32-bit buses, connected to the AHB bus matrix in the D2 domain. Through the system bus matrices, they can access all internal memories except ITCM and DTCM, and external memories through the Quad-SPI controller and the FMC.

## 2.1.7 Clocks to functional blocks

Upon reset, clocks to blocks such as peripherals and some memories are disabled (except for the SRAM, DTCM, ITCM and Flash memory). To operate a block with no clock upon reset, the software must first enable its clock through RCC\_AHBxENR or RCC\_APBxENR register, respectively.

## 2.2 AXI interconnect matrix (AXIM)

### 2.2.1 AXI introduction

The AXI (advanced extensible interface) interconnect is based on the Arm® CoreLink™ NIC-400 Network Interconnect. The interconnect has six initiator ports, or ASIBs (AMBA slave interface blocks), and seven target ports, or AMIBs (AMBA master interface blocks). The ASIBs are connected to the AMIBs via an AXI switch matrix.

Each ASIB is a slave on an AXI bus or AHB (advanced high-performance bus). Similarly, each AMIB is a master on an AXI or AHB bus. Where an ASIB or AMIB is connected to an AHB, it converts between the AHB and the AXI protocol.

The AXI interconnect includes a GPV (global programmer view) which contains registers for configuring certain parameters, such as the QoS (quality of service) level at each ASIB.

Any accesses to unallocated address space are handled by the default slave, which generates the return signals. This ensures that such transactions complete and do not block the issuing master and ASIB.

### 2.2.2 AXI interconnect main features

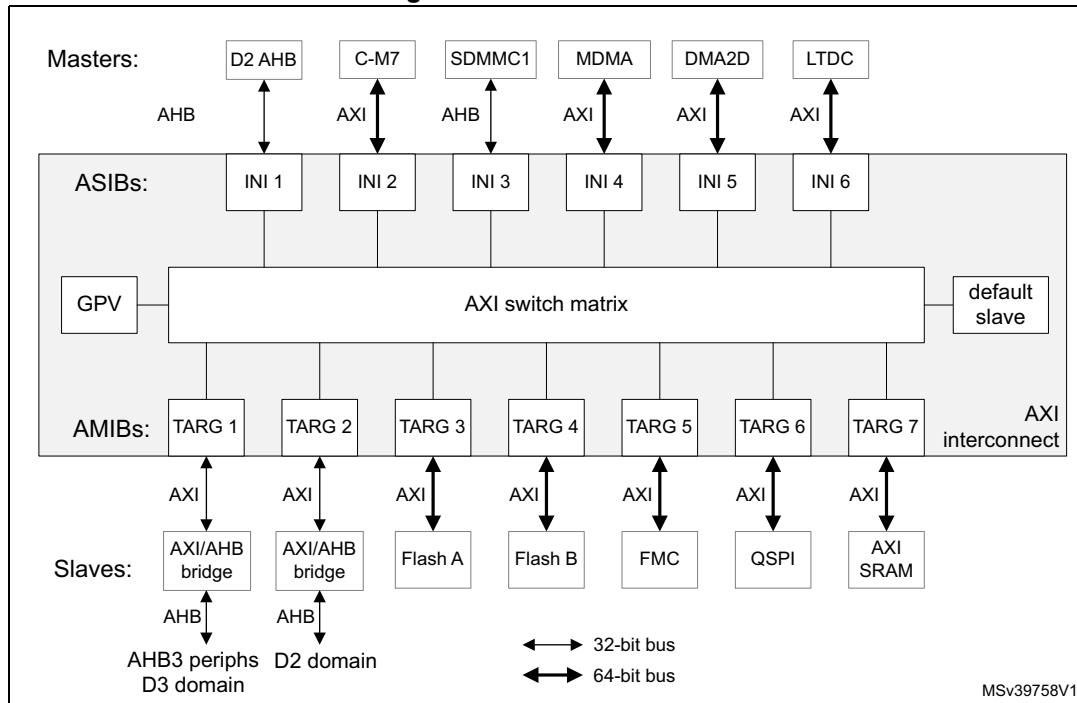
- 64-bit AXI bus switch matrix with six ASIBs and seven AMIBs, in D1 domain
- AHB/AXI bridge function built into the ASIBs
- concurrent connectivity of multiple ASIBs to multiple AMIBs
- programmable traffic priority management (QoS - quality of service)
- software-configurable via GPV

## 2.2.3 AXI interconnect functional description

### Block diagram

The AXI interconnect is shown in [Figure 2](#).

**Figure 2. AXI interconnect**



### ASIB configuration

[Table 4](#) summarizes the characteristics of the ASIBs.

**Table 4. ASIB configuration**

ASIB	Connected master	Protocol	Bus width	R/W issuing
INI 1	AHB from D2 domain	AHB-lite	32	1/4
INI 2	Cortex-M7	AXI4	64	7/32
INI 3	SDMMC1	AHB-lite	32	1/4
INI 4	MDMA	AXI4	64	4/1
INI 5	DMA2D	AXI4	64	2/1
INI 6	LTDC	AXI4	64	1/1

## AMIB configuration

*Table 5* summarizes the characteristics of the AMIBs.

**Table 5. AMIB configuration**

AMIB	Connected slave	Protocol	Bus width	R/W/Total acceptance
TARG 1	Peripheral 3 and D3 AHB	AXI4 <sup>(1)</sup>	32	1/1/1
TARG 2	D2 AHB	AXI4 <sup>(1)</sup>	32	1/1/1
TARG 3	Flash A	AXI4	64	3/2/5
TARG 4	Flash B	AXI4	64	3/2/5
TARG 5	FMC	AXI4	64	3/3/6
TARG 6	QUADSPI	AXI4	64	2/1/3
TARG 7	AXI SRAM	AXI3	64	2/2/2

- Conversion to AHB protocol is done via an AXI/AHB bridge sitting between AXI interconnect and the connected slave.

## Quality of service (QoS)

The AXI switch matrix uses a priority-based arbitration when two ASIB simultaneously attempt to access the same AMIB. Each ASIB has programmable read channel and write channel priorities, known as QoS, from 0 to 15, such that the higher the value, the higher the priority. The read channel QoS value is programmed in the [AXI interconnect - INI x read QoS register \(AXI\\_INIx\\_READ\\_QOS\)](#), and the write channel in the [AXI interconnect - INI x write QoS register \(AXI\\_INIx\\_WRITE\\_QOS\)](#). The default QoS value for all channels is 0 (lowest priority).

If two coincident transactions arrive at the same AMIB, the higher priority transaction passes before the lower priority. If the two transactions have the same QoS value, then a least-recently-used (LRU) priority scheme is adopted.

The QoS values should be programmed according to the latency requirements for the application. Setting a higher priority for an ASIB ensures a lower latency for transactions initiated by the associated bus master. This can be useful for real-time-constrained tasks, such as graphics processing (LTDC, DMA2D). Assigning a high priority to masters that can make many and frequent accesses to the same slave (such as the Cortex-M7 CPU) can block access to that slave by other lower-priority masters.

## Global programmer view (GPV)

The GPV contains configuration registers for the AXI interconnect (see [Section 2.2.4](#)). These registers are only accessible by the Cortex-M7 CPU.

## 2.2.4 AXI interconnect registers

### AXI interconnect - peripheral ID4 register (AXI\_PERIPH\_ID\_4)

Address offset: 0x1FD0

Reset value: 0x0000 0004

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.									
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	4KCOUNT[3:0]	JEP106CON[3:0]	r	r											

Bits 31:8 Reserved, must be kept at reset value.

Bits 7:4 **4KCOUNT[3:0]**: Register file size

0x0: N/A

Bits 3:0 **JEP106CON[3:0]**: JEP106 continuation code

0x4: Arm®

### AXI interconnect - peripheral ID0 register (AXI\_PERIPH\_ID\_0)

Address offset: 0x1FE0

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.									
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	PARTNUM[7:0]	r													

Bits 31:8 Reserved, must be kept at reset value.

Bits 7:0 **PARTNUM[7:0]**: Peripheral part number bits 0 to 7

0x00: Part number = 0x400

**AXI interconnect - peripheral ID1 register (AXI\_PERIPH\_ID\_1)**

Address offset: 0x1FE4

Reset value: 0x0000 00B4

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.									
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	JEP106ID[3:0]		PARTNUM[11:8]												
									r		r				

Bits 31:8 Reserved, must be kept at reset value.

Bits 7:4 **JEP106ID[3:0]**: JEP106 identity bits 0 to 3

0xB: Arm® JEDEC code

Bits 3:0 **PARTNUM[11:8]**: Peripheral part number bits 8 to 11

0x4: Part number = 0x400

**AXI interconnect - peripheral ID2 register (AXI\_PERIPH\_ID\_2)**

Address offset: 0x1FE8

Reset value: 0x0000 002B

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.									
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	REVISION[3:0]	JEDEC	JEP106ID[6:4]												
									r	r	r				

Bits 7:4 **REVISION[3:0]**: Peripheral revision number

0x2: r0p2

Bit 3 **JEDEC**: JEP106 code flag

0x1: JEDEC allocated code

Bits 2:0 **JEP106ID[6:4]**: JEP106 Identity bits 4 to 6

0x3: Arm® JEDEC code

**AXI interconnect - peripheral ID3 register (AXI\_PERIPH\_ID\_3)**

Address offset: 0x1FEC

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.								
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	REV_AND[3:0]				CUST_MOD_NUM[3:0]										
								r				r			

Bits 31:8 Reserved, must be kept at reset value.

Bits 7:4 **REV\_AND[3:0]**: Customer version

0: None

Bits 3:0 **CUST\_MOD\_NUM[3:0]**: Customer modification

0: None

**AXI interconnect - component ID0 register (AXI\_COMP\_ID\_0)**

Address offset: 0x1FF0

Reset value: 0x0000 000D

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.								
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	PREAMBLE[7:0]														
												r			

Bits 31:8 Reserved, must be kept at reset value.

Bits 7:0 **PREAMBLE[7:0]**: Preamble bits 0 to 7

0xD: Common ID value

**AXI interconnect - component ID1 register (AXI\_COMP\_ID\_1)**

Address offset: 0x1FF4

Reset value: 0x0000 00F0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
CLASS[3:0]								PREAMBLE[11:8]							
								r						r	

Bits 31:8 Reserved, must be kept at reset value.

Bits 7:4 **CLASS[3:0]**: Component class

0xF: Generic IP component class

Bits 3:0 **PREAMBLE[11:8]**: Preamble bits 8 to 11

0x0: Common ID value

**AXI interconnect - component ID2 register (AXI\_COMP\_ID\_2)**

Address offset: 0x1FF8

Reset value: 0x0000 0005

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
PREAMBLE[19:12]															
												r			

Bits 31:8 Reserved, must be kept at reset value.

Bits 7:0 **PREAMBLE[19:12]**: Preamble bits 12 to 19

0x05: Common ID value

**AXI interconnect - component ID3 register (AXI\_COMP\_ID\_3)**

Address offset: 0x1FFC

Reset value: 0x0000 00B1

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	PREAMBLE[27:20]														
															r

Bits 31:8 Reserved, must be kept at reset value.

Bits 7:0 **PREAMBLE[27:20]**: Preamble bits 20 to 27

0xB1: Common ID value

**AXI interconnect - TARG x bus matrix issuing functionality register (AXI\_TARGx\_FN\_MOD\_ISS\_BM)**

Address offset: 0x1008 + 0x1000 \* x, where x = 1 to 7

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	WRITE_ISS_OVERRIDE IDE														
															rw
Res.	READ_ISS_OVERRIDE IDE														
															rw

Bits 31:2 Reserved, must be kept at reset value.

Bit 1 **WRITE\_ISS\_OVERRIDE**: Switch matrix write issuing override for target

0: Normal issuing capability

1: Set switch matrix write issuing capability to 1

Bit 0 **READ\_ISS\_OVERRIDE**: Switch matrix read issuing override for target

0: Normal issuing capability

1: Set switch matrix read issuing capability to 1

### AXI interconnect - TARG x bus matrix functionality 2 register (AXI\_TARGx\_FN\_MOD2)

Address offset:  $0x1024 + 0x1000 * x$ , where  $x = 1, 2$  and  $7$

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	BYPASS_MERGE														
															rw

Bits 31:1 Reserved, must be kept at reset value.

Bit 0 **BYPASS\_MERGE**: Disable packing of beats to match the output data width. Unaligned transactions are not realigned to the input data word boundary.

- 0: Normal operation
- 1: Disable packing

### AXI interconnect - TARG x long burst functionality modification register (AXI\_TARGx\_FN\_MOD\_LB)

Address offset:  $0x102C + 0x1000 * x$ , where  $x = 1$  and  $2$

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	FN_MOD_LB														
															rw

Bits 31:1 Reserved, must be kept at reset value.

Bit 0 **FN\_MOD\_LB**: Controls burst breaking of long bursts

- 0: Long bursts can not be generated at the output of the ASIB
- 1: Long bursts can be generated at the output of the ASIB

### AXI interconnect - TARG x issuing functionality modification register (AXI\_TARGx\_FN\_MOD)

Address offset:  $0x1108 + 0x1000 * x$ , where  $x = 1, 2$  and  $7$

Reset value:  $0x0000\ 0000$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	WRITE_ISS_OVERRIDE	READ_ISS_OVERRIDE													
														rw	rw

Bits 31:2 Reserved, must be kept at reset value.

Bit 1 **WRITE\_ISS\_OVERRIDE**: Override AMIB write issuing capability

- 0: Normal issuing capability
- 1: Force issuing capability to 1

Bit 0 **READ\_ISS\_OVERRIDE**: Override AMIB read issuing capability

- 0: Normal issuing capability
- 1: Force issuing capability to 1

### AXI interconnect - INI x functionality modification 2 register (AXI\_INIx\_FN\_MOD2)

Address offset:  $0x41024 + 0x1000 * x$ , where  $x = 1$  and  $3$

Reset value:  $0x0000\ 0000$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	BYPASS_MERGE														
															rw

Bits 31:1 Reserved, must be kept at reset value.

Bit 0 **BYPASS\_MERGE**: Disables alteration of transactions by the up-sizer unless required by the protocol

- 0: Normal operation
- 1: Transactions pass through unaltered where allowed

### AXI interconnect - INI x AHB functionality modification register (AXI\_INIx\_FN\_MOD\_AHB)

Address offset:  $0x41028 + 0x1000 * x$ , where  $x = 1$  and  $3$

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	WR_IN_C_OVE_RRIDE	RD_IN_C_OVE_RRIDE													
														rw	rw

Bits 31:2 Reserved, must be kept at reset value.

Bit 1 **WR\_INC\_OVERRIDE**: Converts all AHB-Lite read transactions to a series of single beat AXI transactions.

- 0: Override disabled
- 1: Override enabled

Bit 0 **RD\_INC\_OVERRIDE**: Converts all AHB-Lite write transactions to a series of single beat AXI transactions, and each AHB-Lite write beat is acknowledged with the AXI buffered write response.

- 0: Override disabled
- 1: Override enabled

### AXI interconnect - INI x read QoS register (AXI\_INIx\_READ\_QOS)

Address offset:  $0x41100 + 0x1000 * x$ , where  $x = 1$  to  $76$

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	AR_QOS[3:0]														
														rw	

Bits 31:4 Reserved, must be kept at reset value.

Bits 3:0 **AR\_QOS[3:0]**: Read channel QoS setting

- 0x0: Lowest priority
- 0xF: Highest priority

**AXI interconnect - INI x write QoS register (AXI\_INIx\_WRITE\_QOS)**

Address offset: 0x41104 + 0x1000 \* x, where x = 1 to 76

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	AW_QOS[3:0]														
															rw

Bits 31:4 Reserved, must be kept at reset value.

Bits 3:0 **AW\_QOS[3:0]**: Write channel QoS setting

0x0: Lowest priority

0xF: Highest priority

**AXI interconnect - INI x issuing functionality modification register (AXI\_INIx\_FN\_MOD)**

Address offset: 0x41108 + 0x1000 \* x, where x = 1 to 76

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	WRITE_ISS_OVERRIDE READ_ISS_OVERRIDE IDE														
															rw rw

Bits 31:2 Reserved, must be kept at reset value.

Bit 1 **WRITE\_ISS\_OVERRIDE**: Override ASIB write issuing capability

0: Normal issuing capability

1: Force issuing capability to 1

Bit 0 **READ\_ISS\_OVERRIDE**: Override ASIB read issuing capability

0: Normal issuing capability

1: Force issuing capability to 1

## **2.2.5 AXI interconnect register map**

**Table 6. AXI interconnect register map and reset values**

**Table 6. AXI interconnect register map and reset values (continued)**

Offset	Register name	Reset value	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x1FF8	AXI_COMP_ID_2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.			
	Reset value	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.			
0x1FFC	AXI_COMP_ID_3	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.			
	Reset value	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.			
0x2000 - 0x2004	Reserved	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.			
0x2008	AXI_TARG1_FN_MOD_ISS_BM	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.			
	Reset value	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.			
0x200C - 0x2020	Reserved	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.			
0x2024	AXI_TARG1_FN_MOD2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.			
	Reset value	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.			
0x2028	Reserved	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.			
0x202C	AXI_TARG1_FN_MOD_LB	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.			
	Reset value	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.			
0x2030 - 0x2104	Reserved	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.			
0x2108	AXI_TARG1_FN_MOD	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.			
	Reset value	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.			
0x210C - 0x3004	Reserved	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.			

**Table 6. AXI interconnect register map and reset values (continued)**

Table 6. AXI interconnect register map and reset values (continued)

Offset	Register name	Reset value	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x5008	AXI_TARG4_ FN_MOD_ ISS_BM	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	
0x500C - 0x6004	Reserved	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.		
0x6008	AXI_TARG5_ FN_MOD_ ISS_BM	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.		
0x600C - 0x7004	Reserved	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.		
0x7008	AXI_TARG6_ FN_MOD_ ISS_BM	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.		
0x700C - 0x8004	Reserved	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.		
0x8008	AXI_TARG7_ FN_MOD_ ISS_BM	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.		
0x800C - 0x8020	Reserved	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.		
0x8024	AXI_TARG7_ FN_MOD2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.		
	Reset value	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	

**Table 6. AXI interconnect register map and reset values (continued)**

Offset	Register name	Reset value	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x8028 - 0x8104	Reserved	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	
0x8108	AXI_TARG7_FN_MOD	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res		
0x810C - 0x42020	Reserved	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res		
0x42024	AXI_INI1_FN_MOD2	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res		
0x42028	AXI_INI1_FN_MOD_AHB	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res		
0x4202C - 0x420FC	Reserved	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res		
0x42100	AXI_INI1_READ_QOS	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res		
0x42104	Reset value	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res		
0x42108	AXI_INI1_WRITE_QOS	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res		
0x4210C - 0x430FC	Reset value	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res		
0x42110	AXI_INI1_FN_MOD	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res		
0x42114	Reset value	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res		
0x42118	Reserved	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res		
0x4211C	AR_QOS [3:0]	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
0x42120	AW_QOS [3:0]	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
0x42124	WR_ISS_OVERRIDE	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res		
0x42128	RD_ISS_OVERRIDE	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res		
0x42132	WRITE_ISS_OVERRIDE	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res		
0x42136	READ_ISS_OVERRIDE	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res		

**Table 6. AXI interconnect register map and reset values (continued)**

Offset	Register name	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x43100	AXI_INI2_READ_QOS	Res.																															
	Reset value	Res.																															
0x43104	AXI_INI2_WRITE_QOS	Res.																															
	Reset value	Res.																															
0x43108	AXI_INI2_FN_MOD	Res.																															
	Reset value	Res.																															
0x4310C - 0x44020	Reserved	Res.																															
0x44024	AXI_INI3_FN_MOD2	Res.																															
	Reset value	Res.																															
0x44028	AXI_INI3_FN_MOD_AHB	Res.																															
	Reset value	Res.																															
0x4402C-0x440FC	Reserved	Res.																															
	AXI_INI3_READ_QOS	Res.																															
0x44100	Reset value	Res.																															
	AXI_INI3_WRITE_QOS	Res.																															
0x44104	Reset value	Res.																															
	AR_QOS [3:0]	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
0x44108	AW_QOS [3:0]	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
	WR_INC_OVERRIDE	Res.																															
0x44110	RD_INC_OVERRIDE	Res.																															
	BYPASS_MERGE	Res.																															
0x44114	WRITE_ISS_OVERRIDE	Res.																															
	READ_ISS_OVERRIDE	Res.																															

**Table 6. AXI interconnect register map and reset values (continued)**

Offset	Register name	Reset value	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x44108	AXI_INI3_FN_MOD		Res.	Res.																														
0x4410C-0x450FC	Reserved		Res.																															
0x45100	AXI_INI4_READ_QOS	Reset value																															AR_QOS [3:0]	
0x45104	AXI_INI4_WRITE_QOS	Reset value																														AW_QOS [3:0]		
0x45108	AXI_INI4_FN_MOD	Reset value																														AR_QOS [3:0]		
0x4510C-0x460FC	Reserved		Res.																															
0x46100	AXI_INI5_READ_QOS	Reset value																														AR_QOS [3:0]		
0x46104	AXI_INI5_WRITE_QOS	Reset value																														AW_QOS [3:0]		
0x46108	AXI_INI5_FN_MOD	Reset value																														AR_QOS [3:0]		
0x4610C-0x470FC	Reserved		Res.																															
0x47100	AXI_INI6_READ_QOS	Reset value																														AR_QOS [3:0]		
			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		

**Table 6. AXI interconnect register map and reset values (continued)**

Refer to [Section 2.3 on page 129](#) for the register boundary addresses.

## 2.3 Memory organization

### 2.3.1 Introduction

Program memory, data memory, registers and I/O ports are organized within the same linear 4-Gbyte address space.

The bytes are coded in memory in Little Endian format. The lowest numbered byte in a word is considered the word's least significant byte and the highest numbered byte the most significant.

The addressable memory space is divided into eight main blocks, of 512 Mbytes each.

### 2.3.2 Memory map and register boundary addresses

**Table 7. Memory map and default device memory area attributes**

Region	Boundary address	Arm® Cortex®-M7	Type	Attribute s	Execute never	
External Devices	0xD0000000 - 0xFFFFFFFF	FMC SDRAM Bank2 (or Reserved in case of FMC remap)	Device	-	Yes	
	0xCC000000 - 0xFFFFFFFF	FMC SDRAM Bank1 (or remap of FMC NOR/PSRAM/SRAM 4 Bank1)				
	0xC8000000 - 0xCBFFFFFF	FMC SDRAM Bank1 (or remap of FMC NOR/PSRAM/SRAM 3 Bank1)				
	0xC4000000 - 0xC7FFFFFF	FMC SDRAM Bank1 (or remap of FMC NOR/PSRAM/SRAM 2 Bank1)				
	0xC0000000 - 0xC3FFFFFF	FMC SDRAM Bank1 (or remap of FMC NOR/PSRAM/SRAM 1 Bank1)				
	0xA0000000 - 0xBFFFFFFF	Reserved				
External Memories	0x90000000 - 0x9FFFFFFF	QUADSPI	Normal	Write-through cache attribute	No	
	0x80000000 - 0x8FFFFFFF	FMC NAND Flash memory		Write-back, write allocate cache attribute		
	0x70000000 - 0x7FFFFFFF	Reserved (or remap of FMC SDRAM Bank2)				
	0x6C000000 - 0x6FFFFFFF	FMC NOR/PSRAM/SRAM 4 Bank1 (or remap of FMC SDRAM Bank1)				
	0x68000000 - 0x6BFFFFFF	FMC NOR/PSRAM/SRAM 3 Bank1 (or remap of FMC SDRAM Bank1)				
	0x64000000 - 0x67FFFFFF	FMC NOR/PSRAM/SRAM 2 Bank1 (or remap of FMC SDRAM Bank1)				
	0x60000000 - 0x63FFFFFF	FMC NOR/PSRAM/SRAM 1 Bank1 (or remap of FMC SDRAM Bank1)				
Peripherals	0x40000000 - 0x5FFFFFFF	Peripherals (refer to <a href="#">Table 8: Register boundary addresses</a> )	Device	-	Yes	

**Table 7. Memory map and default device memory area attributes**

Region	Boundary address	Arm® Cortex®-M7	Type	Attributes	Execute never
RAM	0x38801000 - 0x3FFFFFFF	Reserved	Normal	Write-back, write allocate cache attribute	No
	0x38800000 - 0x38800FFF	Backup SRAM			
	0x38010000 - 0x387FFFFF	Reserved			
	0x38000000 - 0x3800FFFF	SRAM4			
	0x30048000 - 0x37FFFFFF	Reserved			
	0x30040000 - 0x30047FFF	SRAM3			
	0x30020000 - 0x3003FFFF	SRAM2			
	0x30000000 - 0x3001FFFF	SRAM1			
	0x24080000 - 0x2FFFFFFF	Reserved			
	0x24000000 - 0x2407FFFF	AXI SRAM			
	0x20020000 - 0x23FFFFFF	Reserved			
	0x20000000 - 0x2001FFFF	DTCM			
Code	0x1FF20000 - 0x1FFFFFFF	Reserved	Normal	Write-through cache attribute	No
	Bank 1: 0x1FF00000 - 0x1FF1FFFF Bank 2: 0x1FF40000 - 0x1FF5FFFF	System memory			
	0x08200000 - 0x1FEFFFFF	Reserved			
	0x08100000 - 0x081FFFFF	Flash memory bank 2 <sup>(1)</sup>			
	0x08000000 - 0x080FFFFF	Flash memory bank 1 <sup>(2)</sup>			
	0x00010000 - 0x07FFFFFF	Reserved			
	0x00000000 - 0x0000FFFF	ITCM			

1. Flash memory bank 2 boundary is limited to 0x08100000 - 0x0817FFFF on STM32H742xG/STM32H743xG. It is reserved on STM32H750xB.
2. Flash memory bank 1 boundary is limited to 0x08000000 - 0x0807FFFF and 0x08000000 - 0x0801FFFF on STM32H742xG/STM32H743xG and STM32H750xB boundary, respectively.

All the memory map areas that are not allocated to on-chip memories and peripherals are considered “Reserved”. For the detailed mapping of available memory and register areas, refer to the following table.

The following table gives the boundary addresses of the peripherals available in the devices.

**Table 8. Register boundary addresses<sup>(1)</sup>**

Boundary address	Peripheral	Bus	Register map
0x58027000 - 0x580273FF	RAMECC3	AHB4 (D3)	<a href="#">Section 3.4: RAMECC registers</a>
0x58026400 - 0x580267FF	HSEM		<a href="#">Section 10.4: HSEM registers</a>
0x58026000 - 0x580263FF	ADC3		<a href="#">Section 25.7: ADC common registers</a>
0x58025800 - 0x58025BFF	DMAMUX2		<a href="#">Section 17.6: DMAMUX registers</a>
0x58025400 - 0x580257FF	BDMA		<a href="#">Section 16.6: BDMA registers</a>
0x58024C00 - 0x58024FFF	CRC		<a href="#">Section 21.4: CRC registers</a>
0x58024800 - 0x58024BFF	PWR		<a href="#">Section 6.8: PWR register description</a>
0x58024400 - 0x580247FF	RCC		<a href="#">Section 8.7: RCC register description</a>
0x58022800 - 0x58022BFF	GPIOK		<a href="#">Section 11.4: GPIO registers</a>
0x58022400 - 0x580227FF	GPIOJ		<a href="#">Section 11.4: GPIO registers</a>
0x58022000 - 0x580223FF	GPIOI		<a href="#">Section 11.4: GPIO registers</a>
0x58021C00 - 0x58021FFF	GPIOH		<a href="#">Section 11.4: GPIO registers</a>
0x58021800 - 0x58021BFF	GPIOG		<a href="#">Section 11.4: GPIO registers</a>
0x58021400 - 0x580217FF	GPIOF		<a href="#">Section 11.4: GPIO registers</a>
0x58021000 - 0x580213FF	GPIOE		<a href="#">Section 11.4: GPIO registers</a>
0x58020C00 - 0x58020FFF	GPIOD		<a href="#">Section 11.4: GPIO registers</a>
0x58020800 - 0x58020BFF	GPIOC		<a href="#">Section 11.4: GPIO registers</a>
0x58020400 - 0x580207FF	GPIOB		<a href="#">Section 11.4: GPIO registers</a>
0x58020000 - 0x580203FF	GPIOA		<a href="#">Section 11.4: GPIO registers</a>

**Table 8. Register boundary addresses<sup>(1)</sup> (continued)**

Boundary address	Peripheral	Bus	Register map
0x58005800 - 0x58006BFF	Reserved	APB4 (D3)	Reserved
0x58005400 - 0x580057FF	SAI4		<a href="#">Section 51.6: SAI registers</a>
0x58004C00 - 0x58004FFF	Reserved		Reserved
0x58004800 - 0x58004BFF	IWDG1		<a href="#">Section 45.4: IWDG registers</a>
0x58004000 - 0x580043FF	RTC & BKP registers		<a href="#">Section 46.6: RTC registers</a>
0x58003C00 - 0x58003FFF	VREF		<a href="#">Section 27.3: VREFBUF registers</a>
0x58003800 - 0x58003BFF	COMP1 - COMP2		<a href="#">Section 28.7: COMP registers</a>
0x58003000 - 0x580033FF	LPTIM5		<a href="#">Section 43.7: LPTIM registers</a>
0x58002C00 - 0x58002FFF	LPTIM4		<a href="#">Section 43.7: LPTIM registers</a>
0x58002800 - 0x58002BFF	LPTIM3		<a href="#">Section 43.7: LPTIM registers</a>
0x58002400 - 0x580027FF	LPTIM2		<a href="#">Section 43.7: LPTIM registers</a>
0x58001C00 - 0x58001FFF	I2C4		<a href="#">Section 47.7: I2C registers</a>
0x58001400 - 0x580017FF	SPI6		<a href="#">Section 50.11: SPI/I2S registers</a>
0x58000C00 - 0x58000FFF	LPUART1		<a href="#">Section 49.7: LPUART registers</a>
0x58000400 - 0x580007FF	SYSCFG		<a href="#">Section 12.3: SYSCFG registers</a>
0x58000000 - 0x580003FF	EXTI		<a href="#">Section 20.6: EXTI registers</a>
0x52009000 - 0x520093FF	RAMECC1	AHB3 (D1)	<a href="#">Section 3.4: RAMECC registers</a>
0x52008000 - 0x52008FFF	Delay Block SDMMC1		<a href="#">Section 24.4: DLYB registers</a>
0x52007000 - 0x52007FFF	SDMMC1		<a href="#">Section 55.10: SDMMC registers</a>
0x52006000 - 0x52006FFF	Delay Block QUADSPI		<a href="#">Section 24.4: DLYB registers</a>
0x52005000 - 0x52005FFF	QUADSPI control reg- isters		<a href="#">Section 23.5: QUADSPI registers</a>
0x52004000 - 0x52004FFF	FMC control registers		<a href="#">Section 22.7.6: NOR/PSRAM controller registers</a> , <a href="#">Section 22.8.7: NAND flash controller registers</a> , <a href="#">Section 22.9.5: SDRAM controller registers</a>
0x52003000 - 0x52003FFF	JPEG		<a href="#">Section 33.5: JPEG codec registers</a>
0x52002000 - 0x52002FFF	Flash interface regis- ters		<a href="#">Section 4.9: FLASH registers</a>
0x52001000 - 0x52001FFF	Chrom-Art (DMA2D)		<a href="#">Section 18.5: DMA2D registers</a>
0x52000000 - 0x52000FFF	MDMA		<a href="#">Section 14.5: MDMA registers</a>
0x51000000 - 0x510FFFFF	GPV		<a href="#">Section 2.2: AXI interconnect matrix (AXIM)</a>
0x50003000 - 0x50003FFF	WWDG1	APB3 (D1)	<a href="#">Section 44.5: WWDG registers</a>
0x50001000 - 0x50001FFF	LTDC		<a href="#">Section 32.7: LTDC registers</a>
0x50000000 - 0x50000FFF	Reserved		-
0x48023000 - 0x480233FF	RAMECC2		<a href="#">Section 3.4: RAMECC registers</a>

**Table 8. Register boundary addresses<sup>(1)</sup> (continued)**

Boundary address	Peripheral	Bus	Register map
0x48022800 - 0x48022BFF	Delay Block SDMMC2	AHB2 (D2)	<a href="#">Section 24.4: DLYB registers</a>
0x48022400 - 0x480227FF	SDMMC2		<a href="#">Section 55.10: SDMMC registers</a>
0x48021800 - 0x48021BFF	RNG		<a href="#">Section 34.7: RNG registers</a>
0x48021400 - 0x480217FF	HASH		<a href="#">Section 36.7: HASH registers</a>
0x48021000 - 0x480213FF	CRYPTO		<a href="#">Section 35.7: CRYP registers</a>
0x48020000 - 0x480203FF	DCMI		<a href="#">Section 31.5: DCMI registers</a>
0x40080000 - 0x400BFFFF	USB2 OTG_FS	AHB1 (D2)	<a href="#">Section 57.14: OTG_HS registers</a>
0x40040000 - 0x4007FFFF	USB1 OTG_HS		<a href="#">Section 57.14: OTG_HS registers</a>
0x40028000 - 0x400293FF	ETHERNET MAC		<a href="#">Section 58.11: Ethernet registers</a>
0x40024400 - 0x400247FF	Reserved		Reserved
0x40022000 - 0x400223FF	ADC1 - ADC2		<a href="#">Section 25.7: ADC common registers</a>
0x40020800 - 0x40020BFF	DMAMUX1		<a href="#">Section 17.6: DMAMUX registers</a>
0x40020400 - 0x400207FF	DMA2		<a href="#">Section 15.5: DMA registers</a>
0x40020000 - 0x400203FF	DMA1		<a href="#">Section 15.5: DMA registers</a>
0x40017400 - 0x400177FF	HRTIM	APB2 (D2)	<a href="#">Section 37.5: HRTIM registers</a>
0x40017000 - 0x400173FF	DFSDM1		<a href="#">Section 30.7: DFSDM channel y registers (y=0..7), Section 30.8: DFSDM filter x module registers (x=0..3)</a>
0x40016000 - 0x400163FF	SAI3		<a href="#">Section 51.6: SAI registers</a>
0x40015C00 - 0x40015FFF	SAI2		<a href="#">Section 51.6: SAI registers</a>
0x40015800 - 0x40015BFF	SAI1		<a href="#">Section 51.6: SAI registers</a>
0x40015000 - 0x400153FF	SPI5		<a href="#">Section 50.11: SPI/I2S registers</a>
0x40014800 - 0x40014BFF	TIM17		<a href="#">Section 41.6: TIM16/TIM17 registers</a>
0x40014400 - 0x400147FF	TIM16		<a href="#">Section 41.6: TIM16/TIM17 registers</a>
0x40014000 - 0x400143FF	TIM15		<a href="#">Section 41.5: TIM15 registers</a>
0x40013400 - 0x400137FF	SPI4		<a href="#">Section 50.11: SPI/I2S registers</a>
0x40013000 - 0x400133FF	SPI1 / I2S1		<a href="#">Section 50.11: SPI/I2S registers</a>
0x40011400 - 0x400117FF	USART6		<a href="#">Section 48.8: USART registers</a>
0x40011000 - 0x400113FF	USART1		<a href="#">Section 48.8: USART registers</a>
0x40010400 - 0x400107FF	TIM8		<a href="#">Section 38.4: TIM1/TIM8 registers</a>
0x40010000 - 0x400103FF	TIM1		<a href="#">Section 38.4: TIM1/TIM8 registers</a>

**Table 8. Register boundary addresses<sup>(1)</sup> (continued)**

Boundary address	Peripheral	Bus	Register map
0x4000AC00 - 0x4000D3FF	CAN Message RAM	APB1 (D2)	<a href="#">Section 56.5: FDCAN registers</a>
0x4000A800 - 0x4000ABFF	CAN CCU		<a href="#">Section 56.5: FDCAN registers</a>
0x4000A400 - 0x4000A7FF	FDCAN2		<a href="#">Section 56.5: FDCAN registers</a>
0x4000A000 - 0x4000A3FF	FDCAN1		<a href="#">Section 56.5: FDCAN registers</a>
0x40009400 - 0x400097FF	MDIOS		<a href="#">Section 54.4: MDIOS registers</a>
0x40009000 - 0x400093FF	OPAMP		<a href="#">Section 29.6: OPAMP registers</a>
0x40008800 - 0x40008BFF	SWPMI		<a href="#">Section 53.6: SWPMI registers</a>
0x40008400 - 0x400087FF	CRS		<a href="#">Section 9.8: CRS registers</a>
0x40007C00 - 0x40007FFF	UART8		<a href="#">Section 48.8: USART registers</a>
0x40007800 - 0x40007BFF	UART7		<a href="#">Section 48.8: USART registers</a>
0x40007400 - 0x400077FF	DAC1		<a href="#">Section 26.7: DAC registers</a>
0x40006C00 - 0x40006FFF	HDMI-CEC		<a href="#">Section 59.7: HDMI-CEC registers</a>
0x40005C00 - 0x40005FFF	I2C3		<a href="#">Section 47.7: I2C registers</a>
0x40005800 - 0x40005BFF	I2C2		<a href="#">Section 47.7: I2C registers</a>
0x40005400 - 0x400057FF	I2C1		<a href="#">Section 47.7: I2C registers</a>
0x40005000 - 0x400053FF	UART5		<a href="#">Section 48.8: USART registers</a>
0x40004C00 - 0x40004FFF	UART4		<a href="#">Section 48.8: USART registers</a>
0x40004800 - 0x40004BFF	USART3		<a href="#">Section 48.8: USART registers</a>
0x40004400 - 0x400047FF	USART2		<a href="#">Section 48.8: USART registers</a>
0x40004000 - 0x400043FF	SPDIFRX1		<a href="#">Section 52.5: SPDIFRX interface registers</a>
0x40003C00 - 0x40003FFF	SPI3 / I2S3		<a href="#">Section 50.11: SPI/I2S registers</a>
0x40003800 - 0x40003BFF	SPI2 / I2S2		<a href="#">Section 50.11: SPI/I2S registers</a>
0x40002C00 - 0x40002FFF	Reserved		Reserved
0x40002400 - 0x400027FF	LPTIM1		<a href="#">Section 43.7: LPTIM registers</a>
0x40002000 - 0x400023FF	TIM14		<a href="#">Section 39.4: TIM2/TIM3/TIM4/TIM5 registers</a>
0x40001C00 - 0x40001FFF	TIM13		<a href="#">Section 39.4: TIM2/TIM3/TIM4/TIM5 registers</a>
0x40001800 - 0x40001BFF	TIM12		<a href="#">Section 39.4: TIM2/TIM3/TIM4/TIM5 registers</a>
0x40001400 - 0x400017FF	TIM7		<a href="#">Section 42.4: TIM6/TIM7 registers</a>
0x40001000 - 0x400013FF	TIM6		<a href="#">Section 42.4: TIM6/TIM7 registers</a>
0x40000C00 - 0x40000FFF	TIM5		<a href="#">Section 39.4: TIM2/TIM3/TIM4/TIM5 registers</a>
0x40000800 - 0x40000BFF	TIM4		<a href="#">Section 39.4: TIM2/TIM3/TIM4/TIM5 registers</a>
0x40000400 - 0x400007FF	TIM3		<a href="#">Section 39.4: TIM2/TIM3/TIM4/TIM5 registers</a>
0x40000000 - 0x400003FF	TIM2		<a href="#">Section 39.4: TIM2/TIM3/TIM4/TIM5 registers</a>

1. Accessing a reserved area results in a bus error. Accessing undefined memory space in a peripheral returns zeros.

## 2.4 Embedded SRAM

The STM32H742xx, STM32H743/53xx and STM32H750xB devices feature:

- Up to 864 Kbytes of System SRAM
- 128 Kbytes of data TCM RAM
- 64 Kbytes of instruction TCM RAM
- 4 Kbytes of backup SRAM

The embedded system SRAM is divided into up to five blocks over the three power domains:

- D1 domain, AXI SRAM:
  - AXI SRAM is mapped at address 0x2400 0000 and accessible by all system masters except BDMA through D1 domain AXI bus matrix. AXI SRAM can be used for application data which are not allocated in DTCM RAM or reserved for graphic objects (such as frame buffers)
- D2 domain, AHB SRAM:
  - AHB SRAM1 is mapped at address 0x3000 0000 and accessible by all system masters except BDMA through D2 domain AHB matrix. AHB SRAM1 can be used as DMA buffers to store peripheral input/output data in D2 domain.
  - AHB SRAM2 is mapped at address 0x3002 0000 and accessible by all system masters except BDMA through D2 domain AHB matrix. AHB SRAM2 can be used as DMA buffers to store peripheral input/output data in D2 domain.
  - AHB SRAM3 is mapped at address 0x3004 0000 and accessible by all system masters except BDMA through D2 domain AHB matrix. AHB SRAM3 can be used as buffers to store peripheral input/output data for Ethernet and USB.
- D3 domain, AHB SRAM:
  - AHB SRAM4 is mapped at address 0x3800 0000 and accessible by most of system masters through D3 domain AHB matrix. AHB SRAM4 can be used as BDMA buffers to store peripheral input/output data in D3 domain. It can also be used to retain some application data when D1 and D2 domain enter DStandby mode.

The system AHB SRAM can be accessed as bytes, half-words (16-bit units) or words (32-bit units), while the system AXI SRAM can be accessed as bytes, half-words, words or double-words (64-bit units). These memories can be addressed at maximum system clock frequency without wait state.

The AHB masters can read/write-access an SRAM section concurrently with the Ethernet MAC or the USB OTG HS peripheral accessing another SRAM section. For example, the Ethernet MAC accesses the SRAM2 while the CPU accesses the SRAM1, concurrently.

The TCM SRAMs are dedicated to the Cortex<sup>®</sup>-M7:

- DTCM-RAM on TCM interface is mapped at the address 0x2000 0000 and accessible by Cortex<sup>®</sup>-M7, and by MDMA through AHBS slave bus of the Cortex<sup>®</sup>-M7 CPU. The DTCM-RAM can be used as read-write segment to host critical real-time data (such as stack and heap) for application running on Cortex<sup>®</sup>-M7 CPU.
- ITCM-RAM on TCM interface mapped at the address 0x0000 0000 and accessible by Cortex<sup>®</sup>-M7 and by MDMA through AHBS slave bus of the Cortex<sup>®</sup>-M7 CPU. The ITCM-RAM can be used to host code for time-critical routines (such as interrupt handlers) that requires deterministic execution.

The backup RAM is mapped at the address 0x3880 0000 and is accessible by most of the system masters through D3 domain's AHB matrix. With a battery connected to the V<sub>BAT</sub> pin, the backup SRAM can be used to retain data during low-power mode (Standby and V<sub>BAT</sub> mode).

### Error code correction (ECC)

SRAM data are protected by ECC:

- 7 ECC bits are added per 32-bit word.
- 8 ECC bits are added per 64-bit word for AXI-SRAM and ITCM-RAM.

The ECC mechanism is based on the SECDED algorithm. It supports single-error correction and double-error detection.

When an incomplete word is written to an internal SRAM and a reset occurs, the last incomplete word is not really written. This is due to the ECC behavior. To ensure that an incomplete word is written to SRAM, write an additional dummy incomplete word to the same RAM at a different address before issuing a reset.

## 2.5 Flash memory overview

The Flash memory interface manages CPU AXI accesses to the Flash memory. It implements the erase and program Flash memory operations and the read and write protection mechanisms.

The Flash memory is organized as follows:

- Two main memory block divided into sectors.
- An information block:
  - System memory from which the device boots in System memory boot mode
  - Option bytes to configure read and write protection, BOR level, watchdog software/hardware and reset when the device is in Standby or Stop mode.

Refer to [Section 4: Embedded flash memory \(FLASH\)](#) for more details.

## 2.6 Boot configuration

In the STM32H742, STM32H743/753 and STM32H750, two different boot areas can be selected through the BOOT pin and the boot base address programmed in the BOOT\_ADD0 and BOOT\_ADD1 option bytes as shown in the [Table 9](#).

**Table 9. Boot modes**

Boot mode selection		Boot area
BOOT	Boot address option bytes	
0	BOOT_ADD0[15:0]	Boot address defined by user option byte BOOT_ADD0[15:0] ST programmed value: Flash memory at 0x0800 0000
1	BOOT_ADD1[15:0]	Boot address defined by user option byte BOOT_ADD1[15:0] ST programmed value: System memory in bank 1 at 0x1FF0 0000

The values on the BOOT pin are latched on the 4th rising edge of SYSCLK after reset release. It is up to the user to set the BOOT pin after reset.

The BOOT pin is also re-sampled when the device exits the Standby mode. Consequently, they must be kept in the required Boot mode configuration when the device is in the Standby mode.

After startup delay, the selection of the boot area is done before releasing the processor reset.

The BOOT\_ADD0 and BOOT\_ADD1 address option bytes allows to program any boot memory address from 0x0000 0000 to 0x3FFF 0000 which includes:

- All Flash address space
- All RAM address space: ITCM, DTCM RAMs and SRAMs
- The TCM-RAM

The BOOT\_ADD0 / BOOT\_ADD1 option bytes can be modified after reset in order to boot from any other boot address after next reset.

If the programmed boot memory address is out of the memory mapped area or a reserved area, the default boot fetch address is programmed as follows:

- Boot address 0: FLASH at 0x0800 0000
- Boot address 1: ITCM-RAM at 0x0000 0000

When the Flash level 2 protection is enabled, only boot from Flash memory is available. If the boot address already programmed in the BOOT\_ADD0 / BOOT\_ADD1 option bytes is out of the memory range or belongs to the RAM address range, the default fetch will be forced from Flash memory at address 0x0800 0000 .

### Embedded bootloader

The embedded bootloader code is located in system memory. It is programmed by ST during production. It is used to reprogram the Flash memory using one of the following serial interfaces:

- USART1 on PA9/PA10 and PB14/PB15 pins, USART2 on PA3/PA2 pins, and USART3 on PB10/PB11 pins.
- I2C1 on PB6/PB9 pins, I2C2 on PF0/PF1 pins, and I2C3 on PA8/PC9 pins.
- USB OTG FS in Device mode (DFU) on PA11/PA12 pins
- SPI1 on PA4/PA5/PA6/PA7 pins, SPI2 on PI0/PI1/PI2/PI3 pins, SPI3 on PC10/PC11/PC12/PA15 pins, and SPI4 on PE11/PE12/PE13/PE14 pins.

For additional information, refer to the application note AN2606.

## 3 RAM ECC monitoring (RAMECC)

### 3.1 Introduction

The STM32H742, STM32H743/753 and STM32H750 devices feature a RAM ECC monitoring unit (RAMECC). It provides a mean for application software to verify ECC status and execute service routines when an error occurs.

*Note:* *For further information on STM32 RAMECC usage, refer to application note AN5342 "Error correction code (ECC) management for internal memories protection on STM32H7 Series" available from [www.st.com](http://www.st.com).*

### 3.2 RAMECC main features

SRAM data are protected by ECC. The ECC mechanism is based on the SECDED algorithm. It supports single- and double-error detection, as well as single-error correction:

- 7 ECC bits are added per 32-bit word.
- 8 ECC bits are added per 64-bit word for AXI-SRAM and ITCM-RAM.

RAM data word integrity is checked at each memory read access, or partial RAM word write operation. Two cycles are required to perform a partial RAM word write (read-modify-write).

The RAMECC monitoring unit includes the following features:

- RAM ECC monitoring per domain
- RAM failing address/data identification

### 3.3 RAMECC functional description

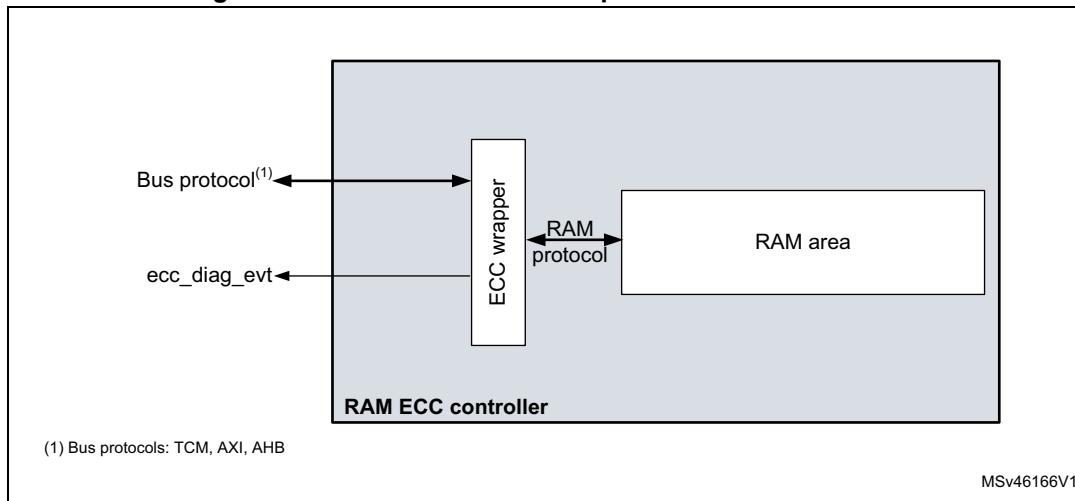
#### 3.3.1 RAMECC block diagram

An ECC controller is associated to each RAM area. It performs the following functions:

- ECC encoding: ECC code computation and storage.
- ECC decoding: RAM data word loading and ECC code decoding to detect errors
- Error detection: single- and double-error detection
- Error correction: single-error correction.

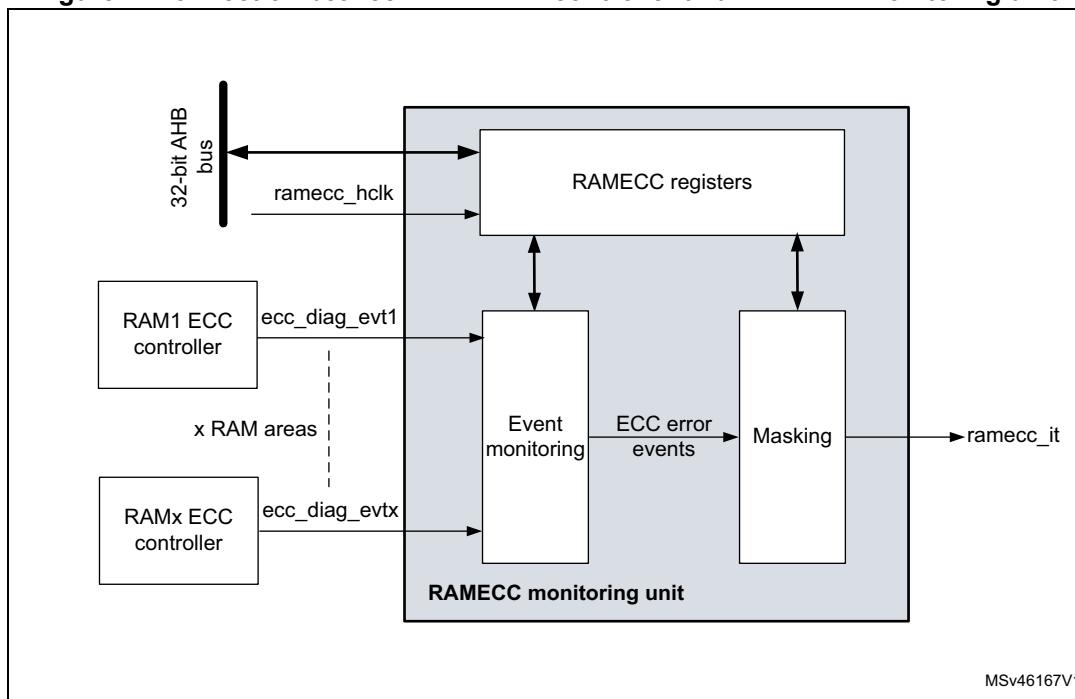
*Note:* *All the RAM ECC controllers are always enabled.*

*Figure 3* describes the implementation of RAM ECC controllers.

**Figure 3. RAM ECC controller implementation schematic**

A dedicated RAM ECC monitoring area is defined for each domain (see [Section 3.3.3: RAMECC monitor mapping](#)). The RAMECC allows the collection of ECC diagnostic events from each RAM ECC controller and provides a mean for the CPU to verify the ECC status.

[Figure 4](#) shows the connection schematic between the RAM ECC controller and the RAMECC monitoring unit.

**Figure 4. Connection between RAM ECC controller and RAMECC monitoring unit**

### 3.3.2 RAMECC internal signals

*Table 10* gives the list of the internal signals that control the RAMECC unit.

**Table 10. RAMECC internal input/output signals**

Internal signal name	Signal type	Description
ramecc_hclk	Input	AHB clock
ecc_diag_evtx	Input	ECC diagnostic event generated by RAMx ECC controller x
ramecc_it	Output	Interrupt generated by the RAMECC monitoring unit when an ECC error is detected.

### 3.3.3 RAMECC monitor mapping

STM32H742, STM32H743/753 and STM32H750 devices features three RAMECC monitoring units (one per domain). The inputs from the ECC controllers are mapped as described in *Table 11*. The RAM ECC event monitoring status and configuration registers are described in *Section 3.4: RAMECC registers*.

**Table 11. ECC controller mapping**

RAMECC units	Monitor number	SRAM ECC event monitoring status and configuration registers		Size in Kbytes	Address Offset
D1 domain RAMECC unit	1	AXI SRAM ECC monitoring unit		512	0x20
	2	ITCM-RAM ECC monitoring unit		64	0x40
	3	DTCM-RAM ECC monitoring unit	D0TCM	64	0x60
	4		D1TCM	64	0x80
	5	ETM RAM ECC monitoring unit		4	0xA0
D2 domain RAMECC unit	1	SRAM1 ECC monitoring unit	SRAM1_0	64	0x20
	2		SRAM1_1	64	0x40
	3	SRAM2 ECC monitoring unit	SRAM2_0	64	0x60
	4		SRAM2_1	64	0x80
	5	SRAM3 ECC monitoring unit		32	0xA0
D3 domain RAMECC unit	1	SRAM4 ECC monitoring unit		64	0x20
	2	Backup RAM ECC monitoring unit		4	0x40

## 3.4 RAMECC registers

RAMECC registers can be accessed only in 32-bit (word) mode. Byte and half-word formats are not allowed.

### 3.4.1 RAMECC interrupt enable register (RAMECC\_IER)

Address offset: 0x00

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.												
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	GECCDEBWIE	GECCDEIE	GECCSEIE	GIE											
												rw	rw	rw	rw

Bits 31:4 Reserved, must be kept at reset value.

Bit 3 **GECCDEBWIE**: Global ECC double error on byte write (BW) interrupt enable

When GECCDEBWIE bit is set to 1, an interrupt is generated when an ECC double detection error occurs during a byte write operation to RAM (incomplete word write).

0: no interrupt generated when an ECC double detection error occurs on byte write

1: interrupt generated if an ECC double detection error occurs on byte write

Bit 2 **GECCDEIE**: Global ECC double error interrupt enable

When GECCDEIE bit is set to 1, an interrupt is generated when an ECC double detection error occurs during a read operation from RAM.

0: no interrupt generated when an ECC double detection error occurs

1: interrupt generated if an ECC double detection error occurs

Bit 1 **GECCSEIE**: Global ECC single error interrupt enable

When GECCSEIE bit is set to 1, an interrupt is generated when an ECC single error occurs during a read operation from RAM.

0: no interrupt generated when an ECC single error occurs

1: interrupt generated when an ECC single error occurs

Bit 0 **GIE**: Global interrupt enable

When GIE bit is set to 1, an interrupt is generated when an enabled global ECC error (GECCDEBWIE, GECCDEIE or GECCSEIE) occurs.

0: no interrupt generated when an ECC error occurs

1: interrupt generated when an ECC error occurs

### 3.4.2 RAMECC monitor x configuration register (RAMECC\_MxCR)

Address offset: 0x20 \* x

Reset value: 0x0000 0000

x is the ECC monitoring unit number

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.										
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	ECCELEN	ECCDEBWIE	ECCDEIE	ECCSEIE	Res.	Res.									
										rw	rw	rw	rw		

Bits 31:6 Reserved, must be kept at reset value.

#### Bit 5 ECCELEN: ECC error latching enable

When ECCELEN bit is set to 1, if an ECC error occurs (both for single error correction or double detection) during a read operation, the context (address, data and ECC code) that generated the error are latched to their respective registers.

- 0: no error context preserved when an ECC error occurs
- 1: error context preserved when an ECC error occurs

#### Bit 4 ECCDEBWIE: ECC double error on byte write (BW) interrupt enable

When ECCDEBWIE bit is set to 1, monitor x generates an interrupt when an ECC double detection error occurs during a byte write operation to RAM.

- 0: no interrupt generated when an ECC double detection error occurs on byte write
- 1: interrupt generated if an ECC double detection error occurs on byte write

#### Bit 3 ECCDEIE: ECC double error interrupt enable

When ECCDEIE bit is set to 1, monitor x generates an interrupt when an ECC double detection error occurs during a read operation from RAM.

- 0: no interrupt generated when an ECC double detection error occurs
- 1: interrupt generated if an ECC double detection error occurs

#### Bit 2 ECCSEIE: ECC single error interrupt enable

When ECCSEIE bit is set to 1, monitor x generates an interrupt when an ECC single error occurs during a read operation from RAM.

- 0: no interrupt generated when an ECC single error occurs
- 1: interrupt generated when an ECC single error occurs

Bits 1:0 Reserved, must be kept at reset value.

### 3.4.3 RAMECC monitor x status register (RAMECC\_MxSR)

Address offset: 0x24 + 0x20 \* (x - 1), (x= ECC monitoring unit number)

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.													
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	DEBW DF	DED F	SEDCF												
													rc_w0	rc_w0	rc_w0

Bits 31: 3 Reserved, must be kept at reset value.

**Bit 2 DEBWDF:** ECC double error on byte write (BW) detected flag

This bit is set by hardware. It is cleared by software by writing a 0

0: no error detected

1: error detected

**Bit 1 DEDF:** ECC double error detected flag

This bit is set by hardware. It is cleared by software by writing a 0

0: no error detected

1: error detected

**Bit 0 SEDCF:** ECC single error detected and corrected flag

This bit is set by hardware. It is cleared by software by writing a 0

0: no error detected and corrected

1: error detected and corrected

### 3.4.4 RAMECC monitor x failing address register (RAMECC\_MxFAR)

Address offset:  $0x28 + 0x20 * (x-1)$ , (x= ECC monitoring unit number)

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FADD[31:16]															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FADD[15:0]															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Bits 31:0 **FADD[31:0]:** ECC error failing address

When an ECC error occurs the FADD bitfield contains the address that generated the ECC error.

### 3.4.5 RAMECC monitor x failing data low register (RAMECC\_MxFDRL)

Address offset:  $0x2C + 0x20 * (x-1)$ , (x= ECC monitoring unit number)

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FDATAL[31:16]															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FDATAL[15:0]															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Bits 31:0 **FDATAL[31:0]:** Failing data low

When an ECC error occurs the FDATAL bitfield contains the LSB part of the data that generated the error. For 32-bit word SRAM, this bitfield contains the full memory word that generated the error.

### 3.4.6 RAMECC monitor x failing data high register (RAMECC\_MxFDRH)

Address offset:  $0x30 + 0x20 * (x-1)$ , (x= ECC monitoring unit number)

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FDATAH[31:16]															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FDATAH[15:0]															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Bits 31:0 **FDATAH[31:0]:** Failing data high (64-bit memory)

When an ECC error occurs the FDATAH bitfield contains the MSB part of the data that generated the error.

Note: *This register is reserved in case of 32-bit word SRAM.*

### 3.4.7 RAMECC monitor x failing ECC error code register RAMECC\_MxFECR)

Address offset:  $0x34 + 0x20 * (x-1)$ , (x= ECC monitoring unit number)

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FEC[31:16]															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FEC[15:0]															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Bits 31:0 **FEC [31:0]:** Failing error code

When an ECC error occurs the FEC bitfield contains the ECC failing code that generated the error.

### **3.4.8 RAMECC register map**

**Table 12. RAMECC register map and reset values**

Refer to [Section 2.3 on page 129](#) for the register boundary addresses.