Renesas RA8 Series

Getting Started with RA8 Memory Architecture, Configurations and Topologies

Introduction

This application note is intended to provide details on how to best utilize memory resources of the RA8 family of devices. As MCUs move into higher complexities, the on-chip memory resources may not be sufficient to support the advanced requirements, so using off-chip memories, becomes a necessity needed as well to fully realize advanced applications using MCUs. RA8 devices have exhaustive on-chip memory in addition to supporting off-chip memory interfaces.

This application note includes:

- Description of the internal Flash and SRAM memory architectures included on RA8 devices.
- Description of the external memory interfaces, including how to interface to OSPI and the SDRAM/Memory bus.
- Instructions on how to connect memory devices to the OSPI and select each using chip select (CS).
- Instructions on how to configure and use the standby SRAM, including the significance of the standby SRAM in an application.
- Instructions on how to configure and use the Tightly Coupled Memory (TCM). Steps in the document to enable the user to choose the TCM.
- Description of ECC and Parity memory on RA devices and its configuration on RA8 devices.

Required Resources

- Flexible Software Package (FSP) v5.1.0

Target devices

- RA8M1, RA8D1

Contents

1. RA8 Architecture 2
1.1 RA8 Bus Architecture 3
1.2 Data Path and RA8 System Bus 6

2. RA8 Memory 7
2.1 RA8 External Memory 9
2.1.1 SDRAM 9
2.1.2 Octal SPI Memory 11
2.1.3 CS Addressable Memory Space 13
2.2 RA8 External Memory Bus Interface 16
2.3 How to Interface External Memory to RA8 20
2.4 RA8 Internal Memory 22
2.4.1 Internal Flash Memory 23
2.4.2 On Chip ROM 28
2.4.3 Option Setting Memory 28
2.5 Option Setting Memory Registers 30

2.5.1 SRAM 33
2.5.2 SSRAM 34
2.5.3 Tightly coupled Memory for Instruction (ITCM) 36
2.5.4 Tightly Coupled Memory for Data (DTCM) 36
2.5.5 Cache 37

3. Memory Consideration while Designing a System 40
3.1 Internal Memory Selection in Designing a System using RA8 MCU 40
3.2 External Memory Selection in Designing a System using RA8 MCU 41

4. Low Power Mode and RA8 Memory 42
4.1 Operating State of Memory Modules in Low Power Mode 43
4.2 Data Retention by using SSRAM 43

5. Configuring the RA8 Memory using FSP Configurator 44
5.1 Modules Available in FSP to Support the RA8 Memory 44
5.2 Configuration Parameters for the RA8 Memory Selection 45

6. How to Interpret the Electrical Characteristic of RA8 MCU 46
6.1 How to Read the Electrical Characteristics Specified in the MCU User Manual 47

7. Example Projects and Reference Applications and Application Notes 54
7.1 Reference Example for OSPI 54
7.2 Reference Example for SSRAM 54
7.3 Reference Examples for SDRAM 55
7.4 Reference Example for D-cache, ITCM, DTCM 55

8. Reference: External Octal Memory Suppliers 55

9. References 56

Revision History 58

1. RA8 Architecture

An example block diagram of the RA8 MCU, based on the Cortex®-M85 processor architecture shown here has the interface between the processor core, caches, external memory interfaces, internal memory and memory-mapped registers using the AMBA (**A**dvanced **M**icrocontroller **B**us **A**rchitecture) interface.

The Cortex®-M85 processor implements AMBA® 5 AXI (Advanced eXtensible Interface) compliant manager AXI (M-AXI) interface for on-chip or off-chip memory and devices. It also provides external interfaces that comply with the AMBA® 5 AHB (Advanced High-performance Bus) protocol. Additionally, the Cortex®-M85 processor implements interfaces for CoreSight™ and other debug components and optional PMC-100 controller for on-line MBIST using the AMBA® 4 APB(Advanced Peripheral Bus) protocol (this is the same as APB protocol version 2.0) and the ATBv1.1 part of the AMBA® 4 ATB(Advanced Trace Bus) protocol.

The bus system includes :

- A master AXI (M-AXI) interface that can be used for on-chip or off-chip memory and devices.
- A single interface to an Instruction Tightly Coupled Memory (ITCM) and to Data Tightly Coupled Memories (DTCMs)

- A Slave AHB (S-AHB) interface for system access to the DMA
- An L1 Instruction Cache (I-Cache) and L1 Data Cache (D-Cache)
- Debug EPPB (External Private Peripheral Bus), for connecting to CoreSight™ debug and trace components, TPIU (Trace Port Interface Unit), ETB (Embedded Trace Buffer), MCU ROM Table and other external debug peripherals.
- Core EPPB, for connecting to EWIC (External Wakeup Interrupt Controller)
- A peripheral AHB (P-AHB) interface for access to external peripherals.

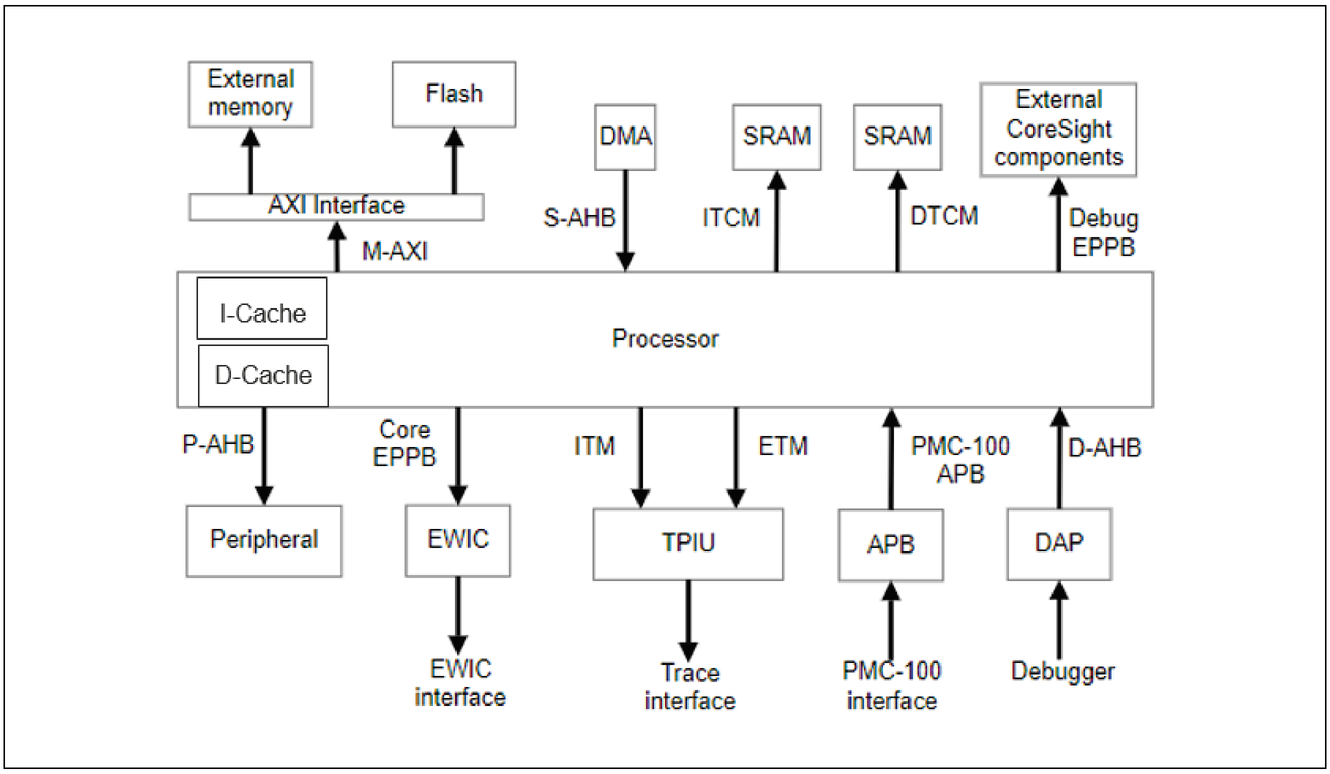


Figure 1. RA8 Architecture Overview

1.1 RA8 Bus Architecture

The RA8 MCU bus allows the microcontroller's CPU to interact with memory, peripherals, and other external devices efficiently. It enables the CPU to read instructions from memory, write data to memory, communicate with input/output devices, and perform various other coordinated operations required by the MCU.

The AMBA bus protocol in RA8, which includes AXI, AHB, and APB, provides a flexible and scalable interconnect framework for communicating high speed devices and low speed devices. It allows different IP components to communicate efficiently and enables the integration of various peripherals and subsystems into a single SoC kind of a design.

In RA8 MCU, a system bus refers to an interface that allows different components within the MCU to exchange data and instructions. It acts as a shared medium through which various parts of the MCU can communicate and coordinate their activities.

A “bus master” device has control over the bus and can initiate data transfers. It is responsible for initiating read and write operations on the bus and controlling the flow of data. The bus master device typically requests data from or sends data to other devices connected to the bus. Whereas a bus slave is a device that responds to requests from the bus master. It does not have direct control over the bus and can only respond to commands or data requests initiated by the bus master. Bus slaves are typically devices such as memory modules, input/output (I/O) devices, or other peripheral devices connected to the bus.

In Figure 2 below, RA8 MCU Bus architecture is shown. Inside the RA8 MCU you can find different sets of Master and Slave buses interfaced to the different memories and peripheral devices.

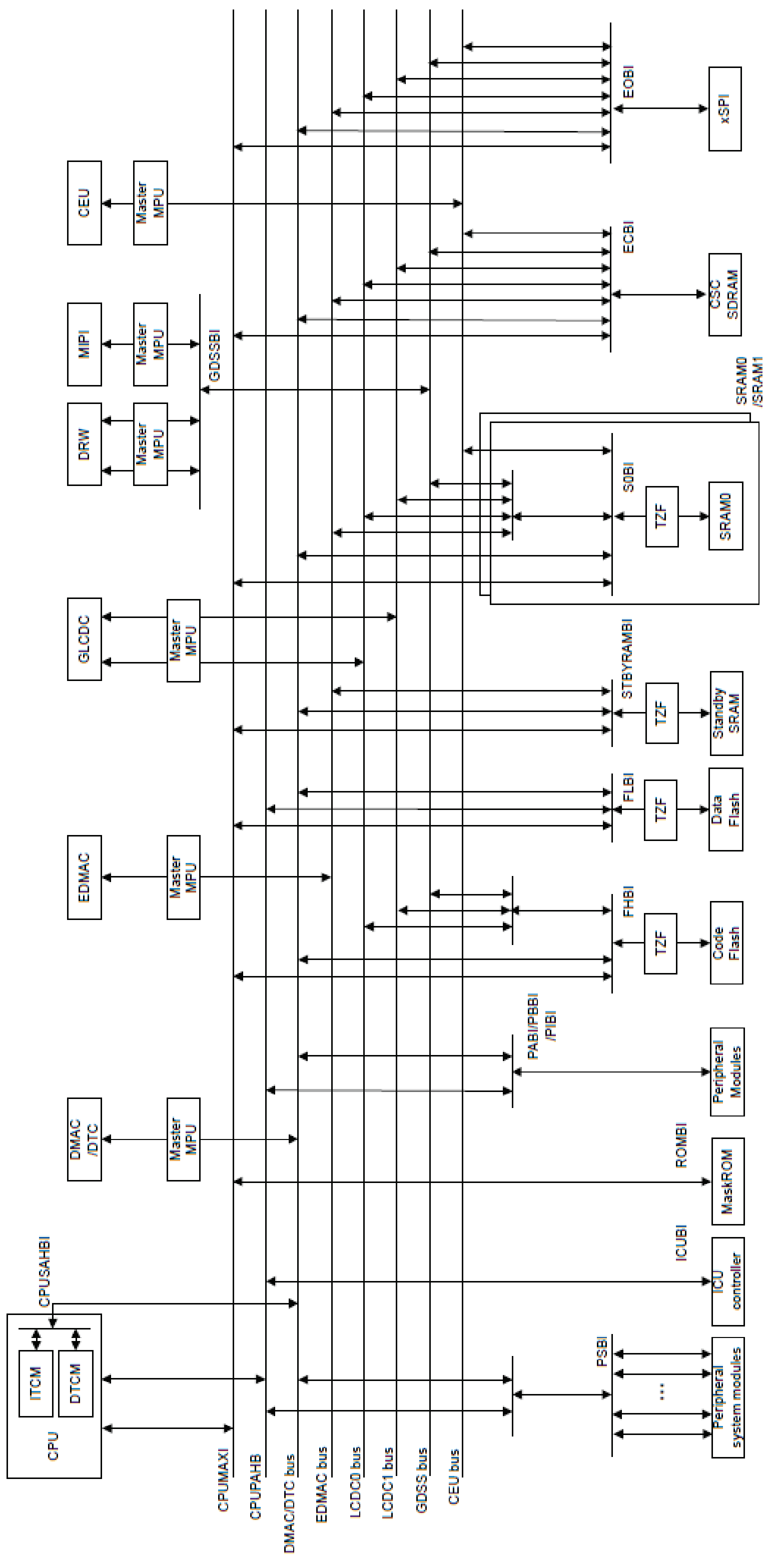


Figure 2. RA8 System Bus Connection Architecture

Table 1. RA8 Bus Masters with Sync Clock

Classification	Bus master/ slave name	Sync clock	Description
Bus masters	CPUMAXIBI (Arm® Cortex®-M85)	CPUCLK	Connected to the CPU Master-AXI (M-AXI) Interface
	CPUPAHBI (Arm® Cortex®-M85)	CPUCLK	Connected to the CPU Peripheral AHB (P-AHB) Interface
	DMAC/DTCBI	ICLK	Connected to the DMAC/DTC Interface
	EDMACBI	PCLKA	Connected to the Ether DMAC Interface
	CEUBI	PCLKA	Connected to the CEU Interface
	GLCDC0BI	PCLKA	Connected to the GLCDC0 Interface
	GLCDC1BI	PCLKA	Connected to the GLCDC1 Interface
	DRW0BI	PCLKA	Connected to the DRW0 Interface
	DRW1BI	PCLKA	Connected to the DRW1 Interface
	MIPIBI	PCLKA	Connected to the MIPI-DSI Interface

Table 2. RA8 Bus Slaves with Sync Clock

Classification	Bus master/ slave name	Sync clock	Description
Bus slaves	FHBI	ICLK	Connected to Code Flash memory and Configration area
	FLBI	FCLK	Connected to Data Flash memory and FACI
	CPUSAHBI	CPUCLK	Connected to CPU S-AHB
	S0BI	ICLK	Connected to SRAM0
	S1BI	ICLK	Connected to SRAM1
	STBYSBI	ICLK	Connected to Standby SRAM
	ECBI	BCLK	Connected to the external devices (External Memory Interface)
	EOBI	PCLKA	Connected to the OSPI (External Memory Interface)
	PBBI	PCLKB	Connected to peripheral modules synchronizes with PCLKB
	PABI	PCLKA	Connected to peripheral modules synchronizes with PCLKA
	PIBI	ICLK	Connected to peripheral modules synchronizes with ICLK
	ICUBI	ICLK	Connected to ICU controller
	PSBI	ICLK	Connected to peripheral system modules (MPU, CSC/SDRAM, SRAM, Debug component, Flash controller, Bus controller, common ICU controller, DMAC/DTC, CPU controller and security attribution controller)
		PCLKB	Connected to peripheral system module. (system controller)
		DCLK	Connected to peripheral system module. (debug controller)

List of bus masters and bus slaves on the RA8 MCUs and the associated sync clocks are listed in the Table 1 and Table 2. For detailed information on the System bus master and bus slaves, please refer to the "Bus" section of the *RA8 MCU Hardware User's Manual (HW UM)*

1.2 Data Path and RA8 System Bus

In RA8 MCU, the data bus connects various hardware components, such as the CPU, memory, input/output (I/O) devices, and other peripheral devices.

From Figure 2, you can see that the CPU has direct access to CPUMAXIBI, CPUPAHBI, DMAC/DTCBI. CPU uses CPUMAXIBI for accessing the MaskROM, Code Flash, Data Flash, SRAM, SSRAM, SDRAM and OSPI. CPU uses CPUPAHBI for accessing the Peripheral System modules via PSBI, Peripheral modules via PABI, PBBI, PIBI. Similarly CPU accesses the DMAC/DTCBI bus via CPUAHBI bus interface. On the other side for DMAC/DTCBI bus all the Peripherals and Memory interfaces are connected.

In a multi master bus system, the data flow may need concurrent access rather than sequential. Doing so allows maximizing the usage of available bus bandwidth. For instance, data transfer from a peripheral device (using PSBI Slave Bus) to CPU (using CPUPAHB Master Bus) and to SRAM using SRAM bus interface (SOBI Slave Bus) can happen in parallel, while the data transfer from SDRAM to GLCDC using LCD0/1 bus can happen without interruption or intervention of the CPU.

In addition, by involving the DMAC/DTCBI instead of CPU (CPUPAHB), the data transfer from the peripheral device (using PSBI Slave Bus) to SRAM using SRAM bus interface (SOBI Slave Bus) can be achieved, and thereby, CPU intervention can be minimized to a great extent.

In some cases, if the same master bus needs to be used by the two different slave buses, the master bus can only handle one transfer at a time, and as such, a method is needed to determine which device gets to use the bus. In the multiple bus system, “arbitration” is the process through which the system uses the shared buses, where multiple devices or components compete for access to the bus to communicate with other devices or the central processing unit (CPU). It is a mechanism for resolving conflicts when two or more devices attempt to use the bus simultaneously.

RA8 supports two different arbitration methods, Round Robin and Fixed Priority. By default, the Fixed Priority method is selected (BUSMABT and BUSSABTn is 0). If the user wants to configure to the Round Robin method, it can be done by configuring the BUSMABT: Bus Master Arbitration Control Register. For more details, refer the “Buses” section the RA8 UM.

Table 3 has the System Bus access path matrix, which shows the Master to slave access and slave to Master access. When a master can access the slave, it is indicated by (**T**) and when the master cannot be accessed, it is marked with (**F**).

Table 3. RA8 System Bus Access Path Matrix

		Master							
	Name	**CPUMAXIBI**	**CPUPAHBI**	**DMAC/DTCBI**	**EDMACBI**	**GLCDC0BI**	**GLCDC1BI**	**GDSSBI**[*1]	**CEUBI**
Slave	PSBI	F	T	T	F	F	F	F	F
	ICUBI	F	T	F	F	F	F	F	F
	CPUSAHBI	F	F	T	F	F	F	F	F
	PIBI	F	T	T	F	F	F	F	F
	PABI	F	T	T	F	F	F	F	F
	PBBI	F	T	T	F	F	F	F	F
	FHBI	T	F	T	F	T	T	T	F
	FLBI	T	T	T	F	F	F	F	F
	STBYRAMBI	T	F	T	T	F	F	F	F
	S0BI	T	F	T	T	T	T	T	T
	S1BI	T	F	T	T	T	T	T	T
	ECBI	T	F	T	T	T	T	T	T
	EOBI	T	F	T	T	T	T	T	T

For instance, CPU Master AXI Bus Interface (CPUMAXIBI) has access to slave bus SRAM 0/1 Bus Interface, and does not have access to the Peripheral System Bus Interface (PSBI), whereas the PSBI can be accessed by CPUPAHBI and DMAC/DTCBI.

2. RA8 Memory

RA8 MCU has addressing capacity of 2^32-bits, making a 4 GB addressable space ranging from 0x0000_0000 to 0XFFFF_FFFF. RA8 supports a variety of memories both internal and external to the MCU.

The different types of internal and external memory are shown in Table 4 below.

Table 4. RA8M1 Memory Example

Internal Memory	
Code flash memory	Up to Maximum 2 MB of code flash memory.
Data flash memory	12 KB of data flash memory.
Option-setting memory	The option-setting memory determines the state of the MCU after a reset.
SRAM	896 KB (384 KB SRAM0+ 512 KB SRAM1) of on-chip high-speed SRAM with either parity bit or Error Correction Code (ECC). SRAM0 is ECC. SRAM1 is Parity check.
Standby SRAM	1 KB On-chip SRAM that can retain data in Deep Software Standby mode 1.
ITCM and DTCM	128 KB (64 KB + 64 KB) of Tightly coupled Instruction and Data Memory.
ROM	On-chip immutable ROM contains First Stage Bootloader (FSBL)
I/D Cache	16 KB of Instruction Cache with ECC and 16 KB of Data Cache with ECC.
External Memory Support (2 slots of 256 MB for (OSPI or QSPI))	
OSPI Flash	256 MB *2 external memory address space
OSPI RAM	256 MB *2 external memory address space
QSPI Flash	256 MB *2 external memory address space
SDRAM	128 MB external memory address space (SDRAM)
CS Area (Chip Select addressable Area)	8 X 16 MB addressable memory regions for additional SRAM, and other memory mapped connections.

All these memories are not in a contiguous area. There are reserved areas for internal/future use. The memory layout can be mainly divided into 4 categories. Internal code memory (Internal Flash), Internal data memory (SRAM), External Bus interface, and Peripheral Registers. Another important feature with RA8 MCUs is that the memory regions are aliased to Secure and Non secure areas.

For instance, the regions from 0x0000_0000 to 0x0FFF_FFFF are aliased to 0x1000_0000 to 0x1FFF_FFFF for Secure and Non secure access. Figure 3 shows the RA8 Memory Map in detail.

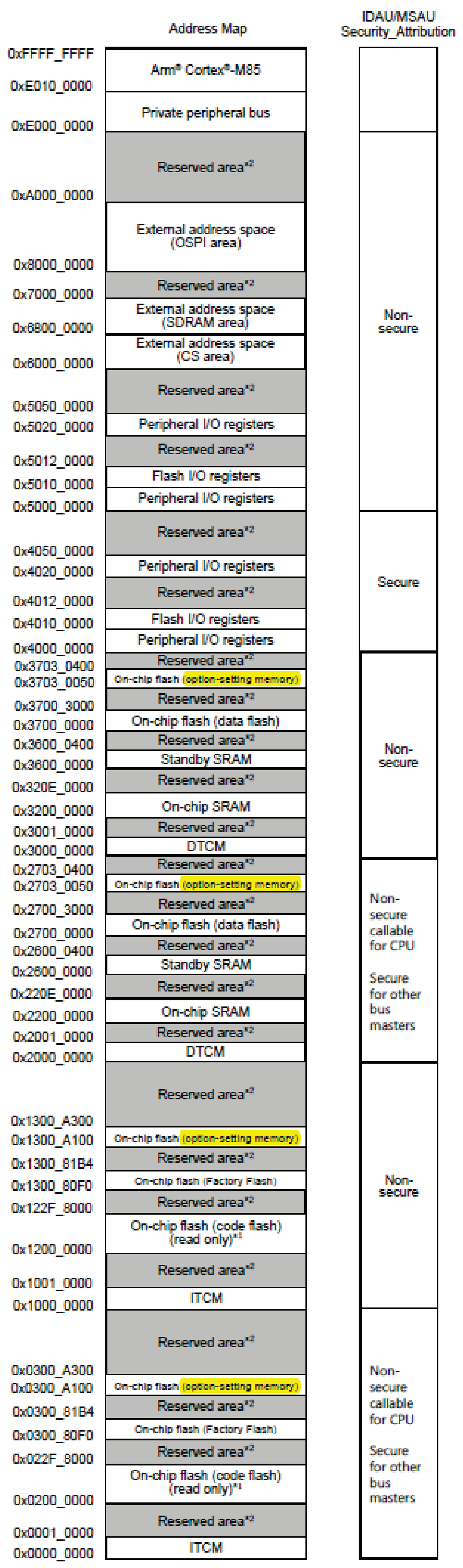


Figure 3. RA8 Memory Map

2.1 RA8 External Memory

When the internal memory capacity (for volatile and non-volatile data) proves insufficient, RA8 supports additional data and code memory via the external memory interface for QSPI flash, OSPI flash, OSPI RAM and SDRAM. Additionally, RA8 also supports 128 MB (16 *8) external CS area for memory mapped devices. All the external memory accesses are in the non-secure region.

In RA8, 128 MB SDRAM area (0x6800_0000 – 0x68FF_FFFF) is exclusively dedicated for the SDRAM memory interface, whereas the 512 MB address space (0x80000000 – 0x9FFF_FFFF) is dedicated for OSPI flash, OSPI RAM or QSPI Flash. There are two slots of 256 MB and can be used for OSPI flash and OSPI RAM, or two OSPI flash, or QSPI. The CS area is at the address space 0x6000_0000 – 0x67FF_FFFF.

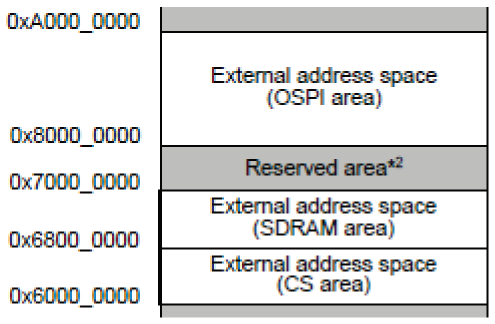

0xA000_0000
External address space (OSPI area)
0x8000_0000
Reserved area*2
0x7000_0000
External address space (SDRAM area)
0x6800_0000
External address space (CS area)
0x6000_0000

Figure 4. RA8 External Memory interface Address Map

The OSPI area is accessed via EOBI, whereas the CS area and SDRAM area are accessed via (ECBI).

Feature	Functional description
External buses	• CS area (ECBI): Connected to the external devices (external memory interface) • SDRAM area (ECBI): Connected to the SDRAM (external memory interface) • OSPI area (EOBI): Connected to the OSPI (external device interface)

Figure 5. RA8 External Memory Bus Interface

2.1.1 SDRAM

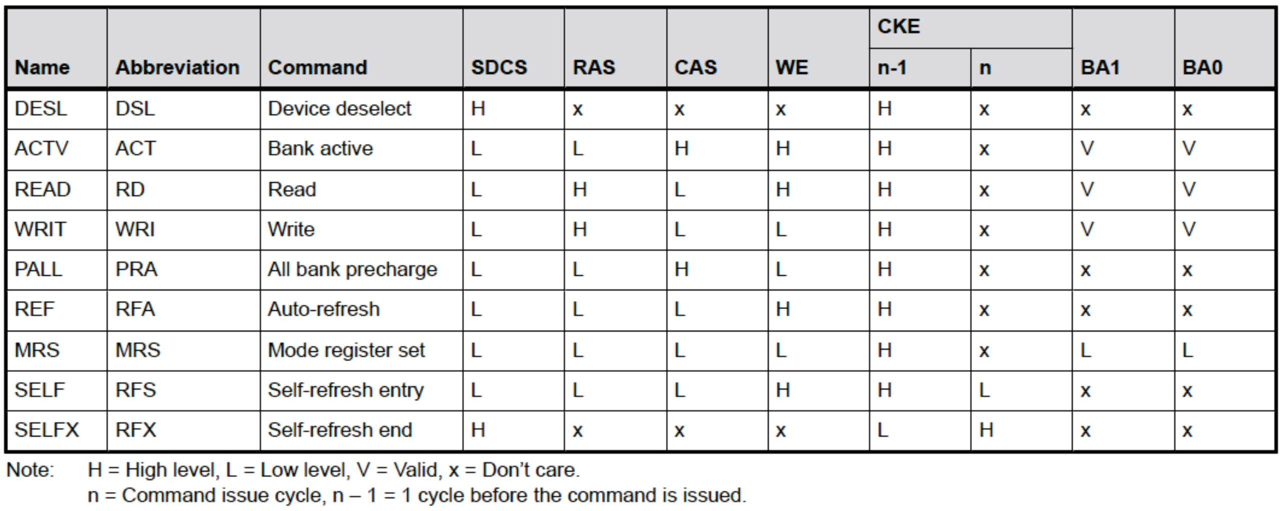
In RA8 MCU, the SDRAM area is placed at an address range from 0x6800_0000 – 0x6FFF_FFFF. RA8 MCU's SDRAM Controller (SDRAMC) extends the memory capabilities of the chip by providing the interface to an external 8-bit, 16-bit, or 32-bit SDRAM device.

The SDRAM controller supports a CAS (Column Access Strobe) latency of 1, 2, or 3 cycles, thus optimizing the read access depending on the frequency. Self-refresh, power down and Deep Power Down mode features can be used to minimize the power consumption of the SDRAM device. Selectable self-refresh and auto-refresh are provided and can multiplex the output of row address and column address (8, 9, 10, or 11 bits). SDRAMC operates in synchronization with the SDRAM clock (SDCLK).

Note 1. BCLK and SDCLK must operate at the same frequency when the SDRAM is in use.

SDRAM access can be enabled or disabled using the SDC Control Register (SDCCR). The SDRAM bus width can also be set using BSIZE[1:0] in SDCCR. The refresh operation is available even when the operation of the SDRAM address space is disabled, as long as self-refresh or auto-refresh is enabled. This feature is particularly helpful when developing strategies for the application to reduce power consumption by transferring application execution into SRDAM.
By default, the Endian mode of the SDRAM is Little endian, the mode can be configured using the EMODE of SDCMOD Register. SDRAM access mode can set for continuous access using BE in SDAMOD: SDRAM Access Mode Register.

To control the SDRAM, the SDRAMC issues a command for each bus cycle. Commands are defined by a combination of the SDCS, RAS, CAS, WE, CKE, and other signals.

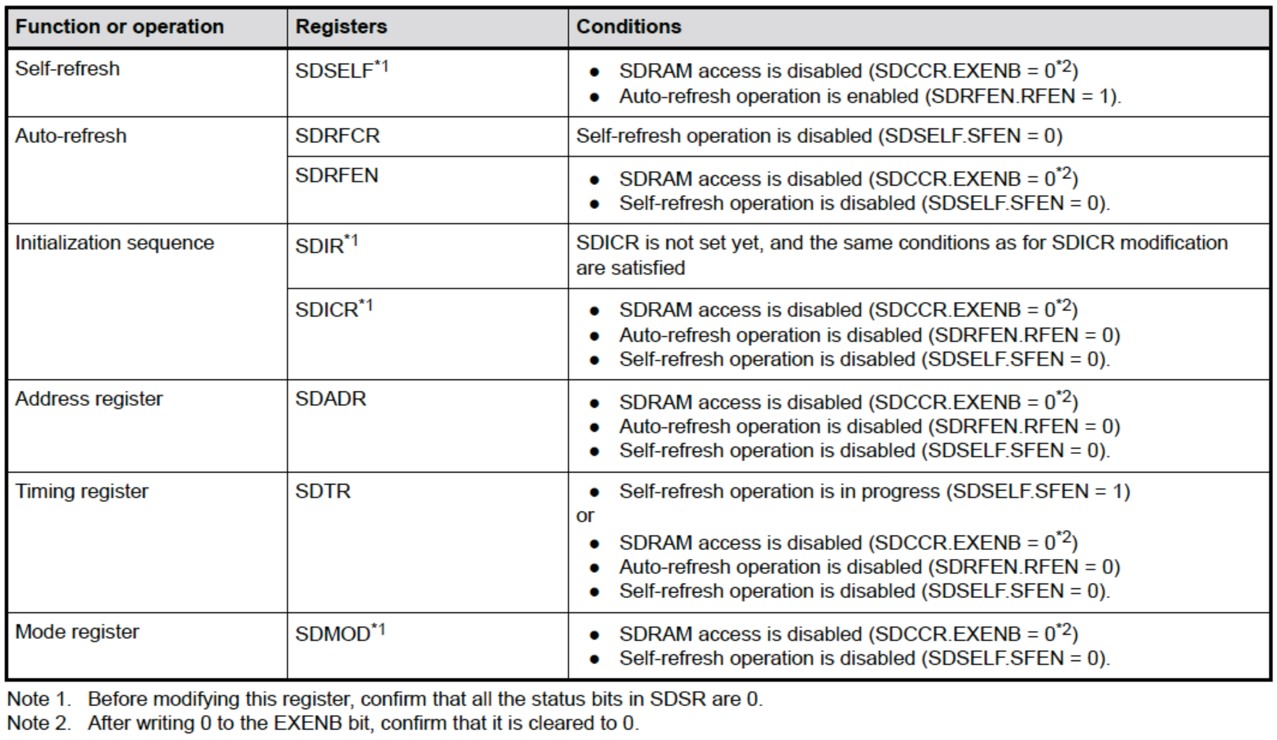


Name	Abbreviation	Command	SDCS	RAS	CAS	WE	CKE		BA1	BA0
							n-1	n		
DESL	DSL	Device deselect	H	x	x	x	H	x	x	x
ACTV	ACT	Bank active	L	L	H	H	H	x	V	V
READ	RD	Read	L	H	L	H	H	x	V	V
WRIT	WRI	Write	L	H	L	L	H	x	V	V
PALL	PRA	All bank precharge	L	L	H	L	H	x	x	x
REF	RFA	Auto-refresh	L	L	L	H	H	x	x	x
MRS	MRS	Mode register set	L	L	L	L	H	x	L	L
SELF	RFS	Self-refresh entry	L	L	L	H	H	L	x	x
SELFX	RFX	Self-refresh end	H	x	x	x	L	H	x	x

Note: H = High level, L = Low level, V = Valid, x = Don't care.
n = Command issue cycle, n – 1 = 1 cycle before the command is issued.

Figure 6. RA8 External Memory- SDRAM Commands

Conditions for Setting the SDRAMC Registers: The SDRAMC registers must only be modified when all the conditions are satisfied as shown in Figure 7.



Function or operation	Registers	Conditions
Self-refresh	SDSELF[*1]	• SDRAM access is disabled (SDCCR.EXENB = 0[*2]) • Auto-refresh operation is enabled (SDRFEN.RFEN = 1).
Auto-refresh	SDRFCR	Self-refresh operation is disabled (SDSELF.SFEN = 0)
	SDRFEN	• SDRAM access is disabled (SDCCR.EXENB = 0[*2]) • Self-refresh operation is disabled (SDSELF.SFEN = 0).
Initialization sequence	SDIR[*1]	SDICR is not set yet, and the same conditions as for SDICR modification are satisfied
	SDICR[*1]	• SDRAM access is disabled (SDCCR.EXENB = 0[*2]) • Auto-refresh operation is disabled (SDRFEN.RFEN = 0) • Self-refresh operation is disabled (SDSELF.SFEN = 0).
Address register	SDADR	• SDRAM access is disabled (SDCCR.EXENB = 0[*2]) • Auto-refresh operation is disabled (SDRFEN.RFEN = 0) • Self-refresh operation is disabled (SDSELF.SFEN = 0).
Timing register	SDTR	• Self-refresh operation is in progress (SDSELF.SFEN = 1) or • SDRAM access is disabled (SDCCR.EXENB = 0[*2]) • Auto-refresh operation is disabled (SDRFEN.RFEN = 0) • Self-refresh operation is disabled (SDSELF.SFEN = 0).
Mode register	SDMOD[*1]	• SDRAM access is disabled (SDCCR.EXENB = 0[*2]) • Self-refresh operation is disabled (SDSELF.SFEN = 0).

Note 1. Before modifying this register, confirm that all the status bits in SDSR are 0.
Note 2. After writing 0 to the EXENB bit, confirm that it is cleared to 0.

Figure 7. RA8 External Memory- Conditions for SDRAM Register Modification

Initialization Sequencer: The SDRAMC has a sequencer to issue SDRAM initialization commands. After a reset, the initialization sequencer must be activated without fail. Operation is not guaranteed if the SDRAM is not initialized.
The SDRAM initialization sequencer issues an all-bank pre-charge command followed by auto-refresh commands n times, where n = 1 to 15. The SDRAM initialization sequence timing can be set using the SDRAM Initialization Register (SDIR).
The SDRAM initialization sequence can be activated using the SDRAM Initialization Sequence Control Register (SDICR). These registers must be set only when the conditions listed in Figure 7 are satisfied.

SDRAMC Setting Procedure.

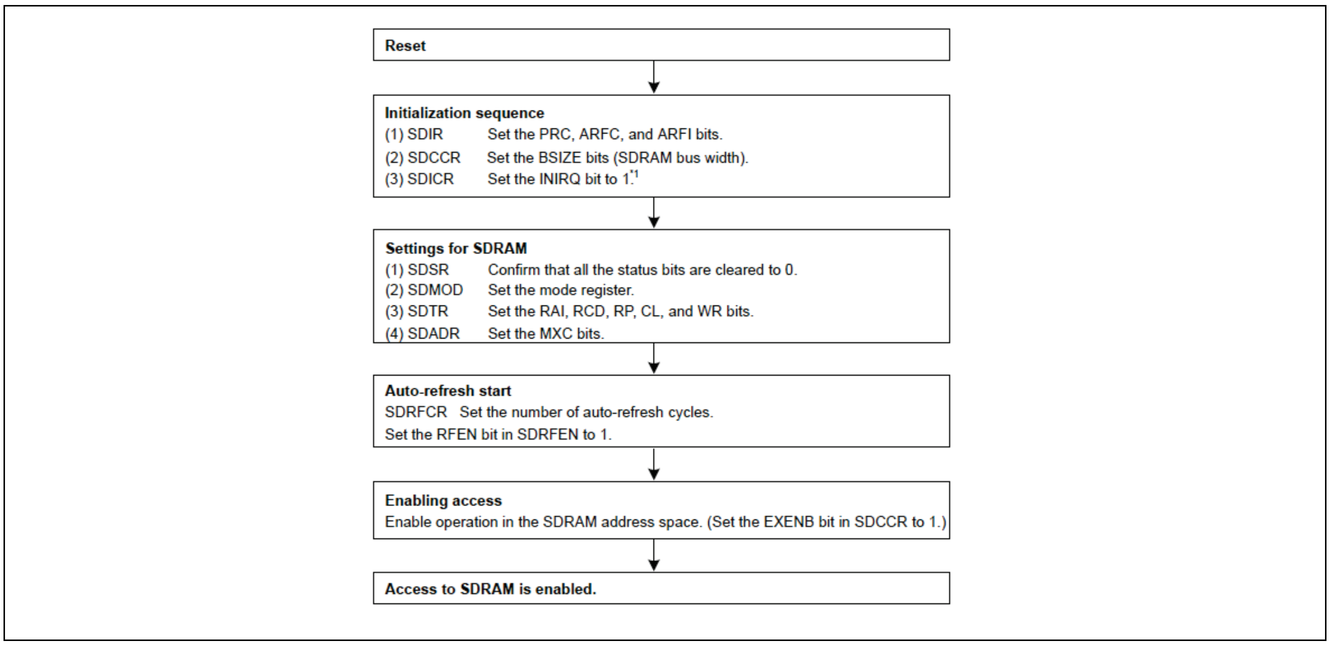

Reset
Initialization sequence
(1) SDIR Set the PRC, ARFC, and ARFI bits.
(2) SDCCR Set the BSIZE bits (SDRAM bus width).
(3) SDICR Set the INIRQ bit to 1.*1
Settings for SDRAM
(1) SDSR Confirm that all the status bits are cleared to 0.
(2) SDMOD Set the mode register.
(3) SDTR Set the RAI, RCD, RP, CL, and WR bits.
(4) SDADR Set the MXC bits.
Auto-refresh start
SDRFCR Set the number of auto-refresh cycles.
Set the RFEN bit in SDRFEN to 1.
Enabling access
Enable operation in the SDRAM address space. (Set the EXENB bit in SDCCR to 1.)
Access to SDRAM is enabled.

Figure 8. RA8 External Memory SDRAMC Setting Procedure

The Renesas FSP provides C header files in the CMSIS data structure that map all of the external bus control registers. The Renesas FSP provides an example to initialize SDRAM memory controller using direct register access for boards which interface the MCU and an SDRAM. Look for the *bsp_sdram_init* function in the file `ra > board > board_name > board_sdram.c.`

2.1.2 Octal SPI Memory

In RA8 MCU, the OSPI area is mapped to the address range from (0x8000_0000 – 0x9FFF_FFFF). The OSPI memory interface is compliant with the xSPI protocol. The OSPI is compliant with JEDEC standard JESD251 (profile 1.0 and 2.0), JESD251-1 and JESD252.

The OSPI interface can support two slave devices. But only one memory device can be accessed at a time. Data can be transferred at up to 200Mbytes/sec. A dedicated transfer target is provided (Channel 1: GLCDC Bus master. Channel 2 Other Bus Master), suitable for Graphical applications.

Decryption on the Fly (DOTF) is supported with memory mapped reads. The data can be encrypted using a pre-stored known key or a run-time generated key.

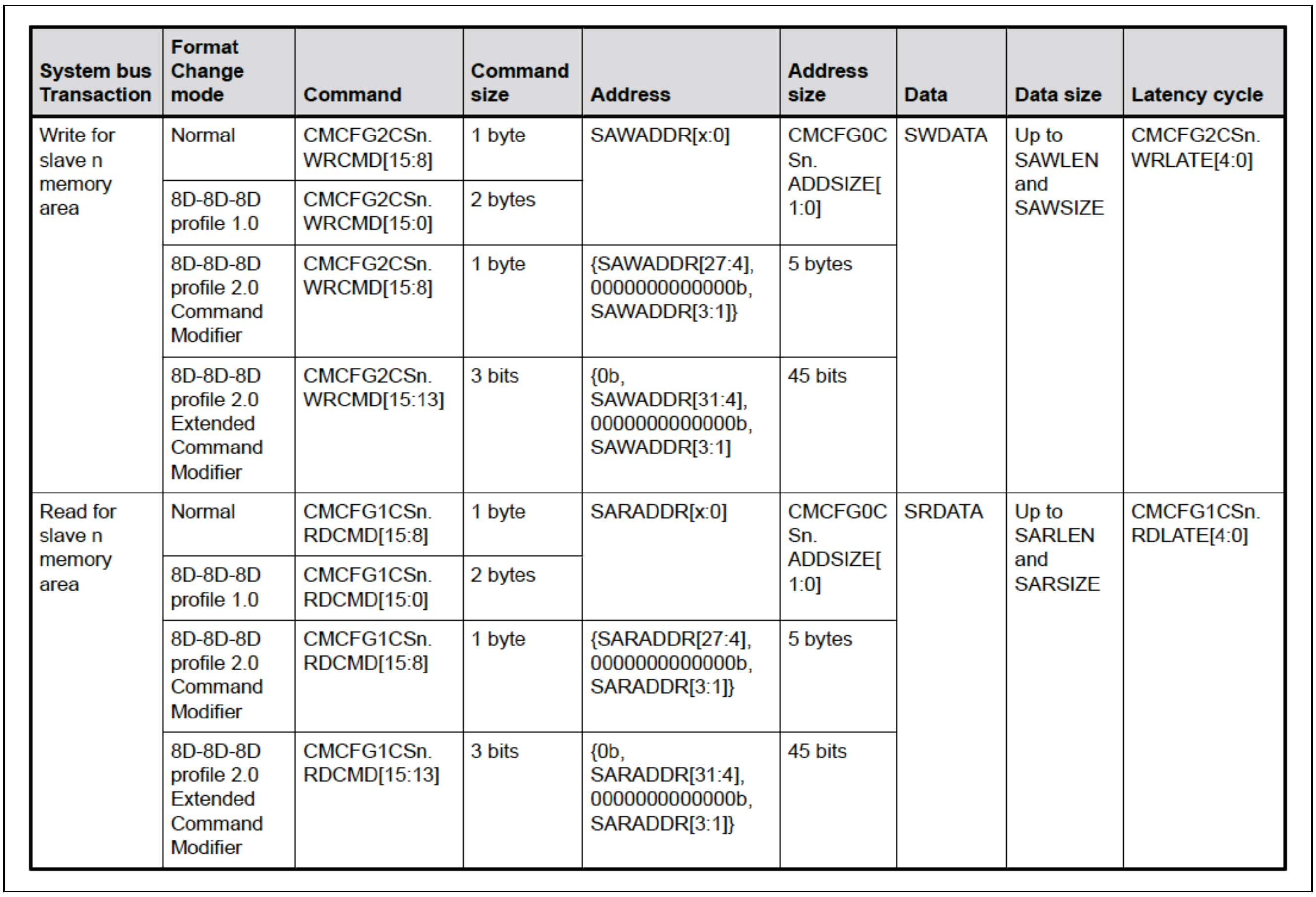
The OSPI device can be erased and programmed using the OSPI APIs from the OSPI module support as well the J-link driver that is integrated with the IDE support.

The following operating modes are supported:

- Protocol modes:
  - 1/4/8pin with SDR/DDR (1S-1S-1S, 4S-4D-4D, 8D-8D-8D)
  - 2/4pin with SDR (1S-2S-2S, 2S-2S-2S, 1S-4S-4S, 4S-4S-4S)
- Configurable address length
- Configurable initial access latency cycle
- Execute in place (XIP) mode

xSPI master interface has functions to issue the transaction for external memory with xSPI slave interfaces. It allows writing to registers in external memory or reading from them. This xSPI master has two modes to issue the transaction. One is a manual-command mode where software dynamically configures all fields of xSPI frame and starts the transaction by software request. The other is a memory-mapping mode in which the xSPI master automatically converts "system bus" for pre-configured memory area into xSPI transaction. It enables access from system bus to external memory area outside of chip via xSPI bus.
In memory mapping mode operation, the payload of address and data field are delivered from system bus signals. The information of command field and size are delivered from the configured register bits. When using FSP OSPI driver after `R_OSPI_Open` API is executed successfully, access to the OSPI data area will be performed in a memory mapped manner. Note that each memory mapped region has an associated CS

(channel selection on the OSPI FSP stack), and it is important to use the correct channel when calling the R_OSPI_Open. **Figure 9** shows the register bits for memory mapping.



System bus Transaction	Format Change mode	Command	Command size	Address	Address size	Data	Data size	Latency cycle
Write for slave n memory area	Normal	CMCFG2CSn. WRCMD[15:8]	1 byte	SAWADDR[x:0]	CMCFG0CSn. ADDSIZE[1:0]	SWDATA	Up to SAWLEN and SAWSIZE	CMCFG2CSn. WRLATE[4:0]
	8D-8D-8D profile 1.0	CMCFG2CSn. WRCMD[15:0]	2 bytes					
	8D-8D-8D profile 2.0 Command Modifier	CMCFG2CSn. WRCMD[15:8]	1 byte	{SAWADDR[27:4], 0000000000000b, SAWADDR[3:1]}	5 bytes			
	8D-8D-8D profile 2.0 Extended Command Modifier	CMCFG2CSn. WRCMD[15:13]	3 bits	{0b, SAWADDR[31:4], 0000000000000b, SAWADDR[3:1]	45 bits			
Read for slave n memory area	Normal	CMCFG1CSn. RDCMD[15:8]	1 byte	SARADDR[x:0]	CMCFG0CSn. ADDSIZE[1:0]	SRDATA	Up to SARLEN and SARSIZE	CMCFG1CSn. RDLATE[4:0]
	8D-8D-8D profile 1.0	CMCFG1CSn. RDCMD[15:0]	2 bytes					
	8D-8D-8D profile 2.0 Command Modifier	CMCFG1CSn. RDCMD[15:8]	1 byte	{SARADDR[27:4], 0000000000000b, SARADDR[3:1]}	5 bytes			
	8D-8D-8D profile 2.0 Extended Command Modifier	CMCFG1CSn. RDCMD[15:13]	3 bits	{0b, SARADDR[31:4], 0000000000000b, SARADDR[3:1]}	45 bits			

Figure 9. OSPI Memory-mapping Configuration for Memory Area Access (n = 0, 1) Specifications

The OSPI interface provides for DOTF functionality when configured in memory map mode. This provides a strong layer of protection when storing information in the external SPI devices. The DOTF functionality supports encrypted data and code storage on the xSPI device. The data can be encrypted using a pre-stored known key or a run-time generated key. A dedicated AES engine supports transparent OSPI operation for data read and code execution. For the details on the operational flow of this feature, user can reference application note R11AN0773 which is planned to be published in Q1 2024.

2.1.3 CS Addressable Memory Space

The CS area of 8, 16MB slots of Address spaces are available for memory and memory mapped peripherals as well. The address space for the CS0 – CS7 ranges from (0x60000000 – 0x67FFFFFF).

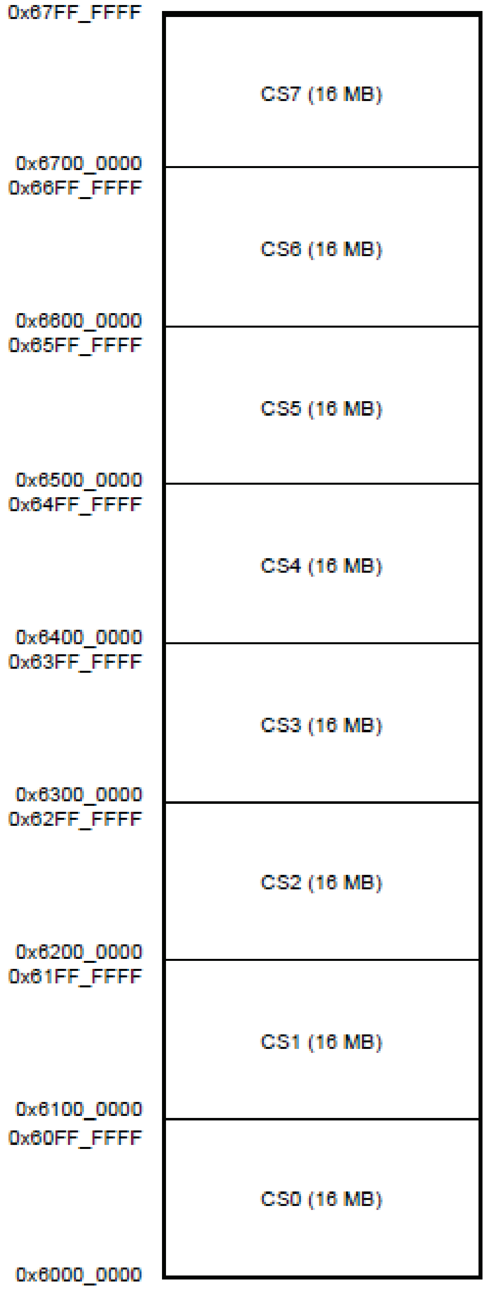


Figure 10. RA8 CS Area Memory Map

The use case for the external CS area is to provide additional memory capacity beyond the internal memory of a microcontroller or microprocessor. This external memory space is useful in various use cases where more memory is needed than the built-in memory can provide. Additional code storage such as Flash NAND or NOR flash storage, and data storage such as additional high speed RAM storage are examples of simple use cases.

In multi-processing systems, the CS area can be used to add a FGPA or SOC implementation which acts as a sub system for the MCU. Here the MCU can be interfaced to the sub systems. For example, the MCU can be connected to the FPGA subsystem, and the MCU can upgrade the FPGA image. The MCU can have a dual port memory through which the processed data from the FPGA can be accessed by MCU for application level processing.

The RA8 MCU's CS area can operate in a Separate Bus mode and Address Data Multiplexed Bus mode.

Separate Bus mode:
In Separate Bus mode, the CS area controller (CSC) operates in synchronization with the external bus clock, BCLK. Operation cycles, such as wait cycles, specified in the CSC register, are counted on BCLK.

Normal Access: When the PRENB and PWENB bits in CSnMOD registers are set to 0 to disable page read and page write access, all bus accesses take the form of normal read and write operations. Even when these bits are set to 1 to enable page read and page write access, a bus access other than the page access takes the form of normal read and write operations.

Page Access: When the PRENB and PWENB bits in CSnMOD are set to 1 to enable page read and page write access, the bus access for page access operations becomes page read and write. Page access can only occur when two or more rounds of external bus access are required for a single transfer request from the bus master. However, normal access is made when split accesses are not aligned or access extends across the 32-bit boundary.

Address/Data Multiplexed Bus mode:
When the address/data Multiplexed I/O Interface Select bit (MPXEN) in CSnCR is set to 1, addresses and data can be multiplexing input/output to/from the D15 to D00 pins in the corresponding area. Using this function enables direct connection of this MCU to peripherals of the MCU requiring address/data multiplexing. When 8-bit width is selected with the BSIZE[1:0] bits in CSnCR, D7 to D00 are multiplexed with A07 to A00. When 16-bit width is selected, D15 to D00 are multiplexed with A15 to A00. In the address/data multiplexed I/O space, accesses are controlled with the ALE, RD, WRn, and BCn signals.
Byte strobe mode or single-write strobe mode is selectable in the same way as for a separate bus. However, with regard to the BCn signals within the address cycle, the byte-control signal is output for the data being read or written.
During the address/data multiplexed I/O space access, after the number of wait cycles specified by the Address Cycle Wait Select bits (AWAIT[1:0]) in CSnWCR2 is inserted in the address output cycle, data access is performed.

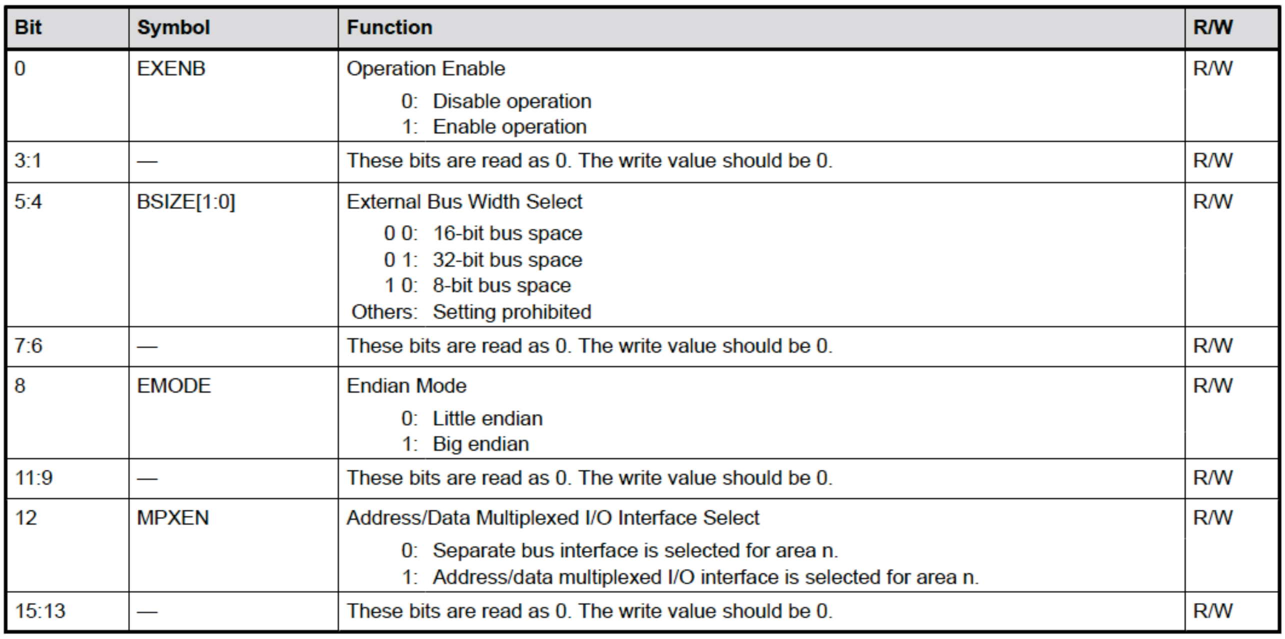
CS Area provides External Wait Function. Wait cycles can be extended by the WAIT signal beyond the length of the normal access cycle wait specified in the CSRWAIT[4:0] and CSWWAIT[4:0] bits in CSnWCR1, and the page access cycle wait specified in the CSPRWAIT[2:0] and CSPWWAIT[2:0] bits in CSnWCR1. When external wait is enabled (EWENB = 1 in CSnMOD), wait cycles are inserted while the WAIT signal is held low. When external wait is disabled (EWENB = 0 in CSnMOD), the WAIT signal has no effect. All wait cycles specified in CSnWCR1 are inserted independently of the WAIT signal. Configuring the WAIT signal enables the MCU to meet the specifications of the externally interfaced device.

With the Insertion of Recovery Cycles, recovery cycles can be inserted between consecutive rounds of external bus access by setting the Recovery Cycle Insertion Enable bit in CSRECEN to 1.

CS Area provides Write Buffer Function (External Bus). In write access, the main bus is released by writing data to the write buffer before the access is complete. This allows the next round of bus access to start. However, if the next access is to an external address space or to a register of the external bus controller, it is suspended until the external bus operations already in progress are complete.

Note: For detailed external bus read write timing diagram, refer to the HW UM of the respective MCUs.

CS Area control register is used for configuring the operation with different external bus widths and different endianness modes while choosing Separate or Multiplexed Address/Data Bus. CSnCR : CSn Control Register (n = 0 to 7).



Bit	Symbol	Function	R/W
0	EXENB	Operation Enable 0: Disable operation 1: Enable operation	R/W
3:1	—	These bits are read as 0. The write value should be 0.	R/W
5:4	BSIZE[1:0]	External Bus Width Select 0 0: 16-bit bus space 0 1: 32-bit bus space 1 0: 8-bit bus space Others: Setting prohibited	R/W
7:6	—	These bits are read as 0. The write value should be 0.	R/W
8	EMODE	Endian Mode 0: Little endian 1: Big endian	R/W
11:9	—	These bits are read as 0. The write value should be 0.	R/W
12	MPXEN	Address/Data Multiplexed I/O Interface Select 0: Separate bus interface is selected for area n. 1: Address/data multiplexed I/O interface is selected for area n.	R/W
15:13	—	These bits are read as 0. The write value should be 0.	R/W

Figure 11. RA8 External Memory- CS Area Control Register Modification

EXENB bit (Operation Enable)
The EXENB bit enables operation of the associated CS area. On MCU reset, operation is enabled by (EXENB = 1) only for area 0. Operation in other areas is disabled (EXENB = 0). Attempts to access disabled areas have no effect. When the CSC and SDRAMC are in use at the same time, EBCLK and SDCLK must operate at the same frequency.

BSIZE[1:0] bits (External Bus Width Select)
The BSIZE[1:0] bits specify the data bus width for the associated area.
EMODE bit (Endian Mode)
The EMODE bit specifies the endianness for the associated area. The Arm® Cortex®-M85 core is fixed at little-endian order, so instruction code can only be allocated to external spaces with little-endian specified. If an area is specified as big endian, no instruction code can be allocated to it. Only CPU, DMAC and DTC can access to big-endian area. Memory type of big-endian area must be Device-Memory.

MPXEN bit (Address/Data Multiplexed I/O Interface Select)
The MPXEN bit specifies separate bus interface or address/data multiplexed I/O interface of each area.

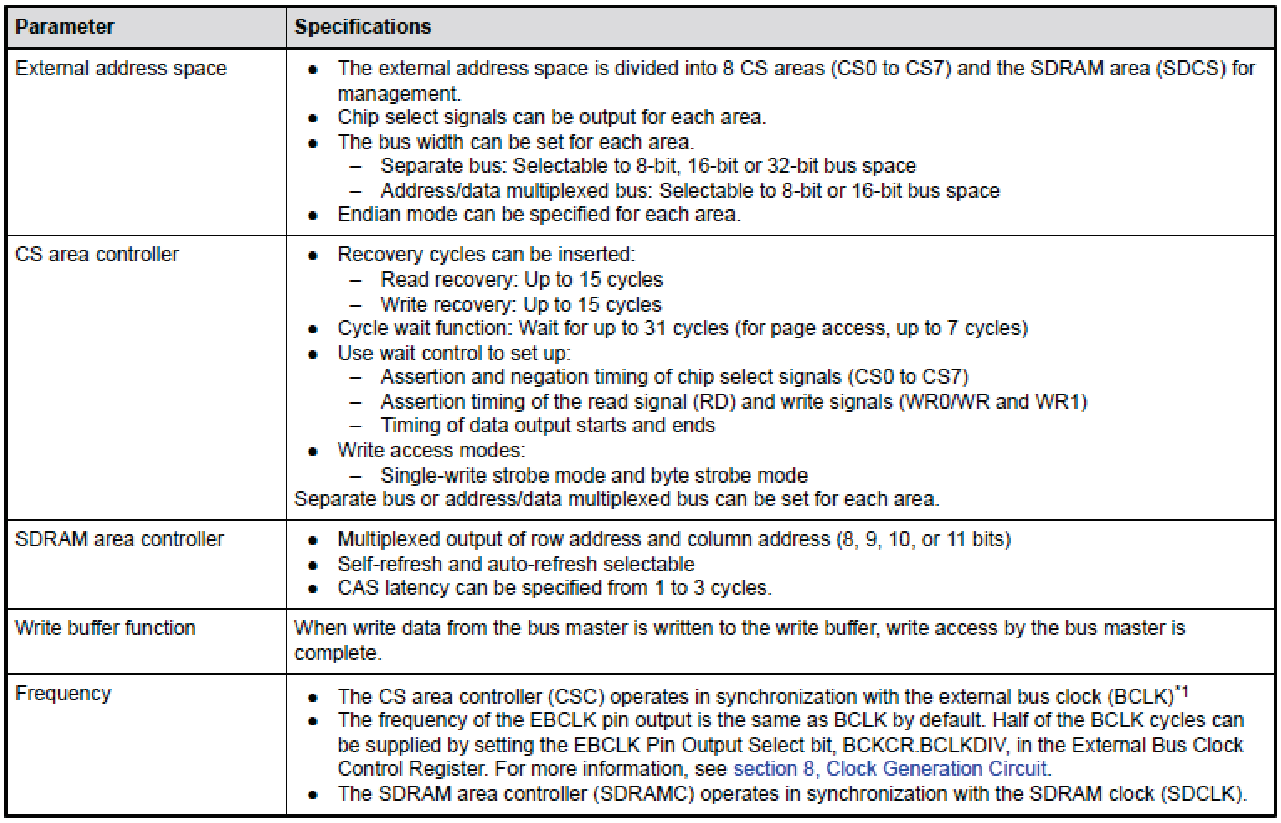
The Data alignment control for the CS area for 32 bit, 16 bit, 8 bit for little endian and big endian are shown in detail in the section ("Endian and Data Alignment" of the HW UM).

Note: BCLK (external bus clock) : 120 MHz (max.) (The CSC (CS area controller) operates in synchronization with the BCLK).

BCLK pin output: The frequency is the same as the default of BCLK. Half of BCLK can be supplied by setting the EBCLK pin and the output select bit (BCKCR.BCLKDIV) in the External Bus Clock Control register. For details, see section, Clock Generation Circuit.

2.2 RA8 External Memory Bus Interface

The External Memory Bus interface is to facilitate the transfer of data between the processor and external memory devices, such as RAM (Random Access Memory) or non-volatile memory like Flash memory. The External Memory Bus interface typically includes control signals, address lines, and data lines to establish communication and transfer data between the processor and the memory devices.

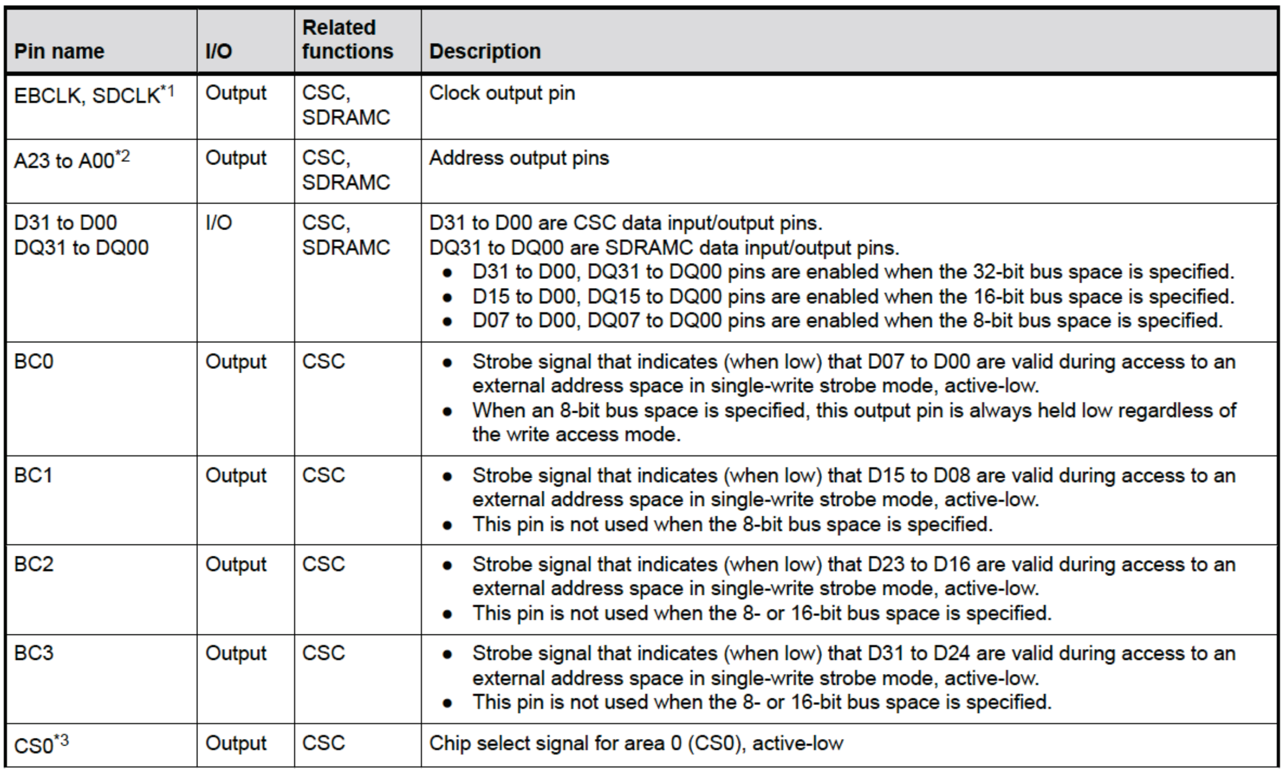


Parameter	Specifications
External address space	• The external address space is divided into 8 CS areas (CS0 to CS7) and the SDRAM area (SDCS) for management. • Chip select signals can be output for each area. • The bus width can be set for each area. – Separate bus: Selectable to 8-bit, 16-bit or 32-bit bus space – Address/data multiplexed bus: Selectable to 8-bit or 16-bit bus space • Endian mode can be specified for each area.
CS area controller	• Recovery cycles can be inserted: – Read recovery: Up to 15 cycles – Write recovery: Up to 15 cycles • Cycle wait function: Wait for up to 31 cycles (for page access, up to 7 cycles) • Use wait control to set up: – Assertion and negation timing of chip select signals (CS0 to CS7) – Assertion timing of the read signal (RD) and write signals (WR0/WR and WR1) – Timing of data output starts and ends • Write access modes: – Single-write strobe mode and byte strobe mode Separate bus or address/data multiplexed bus can be set for each area.
SDRAM area controller	• Multiplexed output of row address and column address (8, 9, 10, or 11 bits) • Self-refresh and auto-refresh selectable • CAS latency can be specified from 1 to 3 cycles.
Write buffer function	When write data from the bus master is written to the write buffer, write access by the bus master is complete.
Frequency	• The CS area controller (CSC) operates in synchronization with the external bus clock (BCLK)*1 • The frequency of the EBCLK pin output is the same as BCLK by default. Half of the BCLK cycles can be supplied by setting the EBCLK Pin Output Select bit, BCKCR.BCLKDIV, in the External Bus Clock Control Register. For more information, see section 8, Clock Generation Circuit. • The SDRAM area controller (SDRAMC) operates in synchronization with the SDRAM clock (SDCLK).

Figure 12. RA8 External Memory Bus Interface

The external bus controller arbitrates requests for bus access on the external address space from the CPU M-AXI bus, DMAC/DTC bus, EDMAC (Ether) bus, GLCDC0 bus, GLCDC1 bus, DRW0 bus, DRW1 bus, MIPI bus and CEU bus. For more details see section "Arbitration" for the priority and arbitration method of each master to the “External Bus” section of the UM of the RA8 MCU.

The physical signals required for RA8 to interface to the external memory are listed below in Figure 13. Refer to section “External Bus” in the UM for more details and specific to the RA8 MCU.



Pin name	I/O	Related functions	Description
EBCLK, SDCLK[*1]	Output	CSC, SDRAMC	Clock output pin
A23 to A00[*2]	Output	CSC, SDRAMC	Address output pins
D31 to D00 DQ31 to DQ00	I/O	CSC, SDRAMC	D31 to D00 are CSC data input/output pins. DQ31 to DQ00 are SDRAMC data input/output pins. ● D31 to D00, DQ31 to DQ00 pins are enabled when the 32-bit bus space is specified. ● D15 to D00, DQ15 to DQ00 pins are enabled when the 16-bit bus space is specified. ● D07 to D00, DQ07 to DQ00 pins are enabled when the 8-bit bus space is specified.
BC0	Output	CSC	● Strobe signal that indicates (when low) that D07 to D00 are valid during access to an external address space in single-write strobe mode, active-low. ● When an 8-bit bus space is specified, this output pin is always held low regardless of the write access mode.
BC1	Output	CSC	● Strobe signal that indicates (when low) that D15 to D08 are valid during access to an external address space in single-write strobe mode, active-low. ● This pin is not used when the 8-bit bus space is specified.
BC2	Output	CSC	● Strobe signal that indicates (when low) that D23 to D16 are valid during access to an external address space in single-write strobe mode, active-low. ● This pin is not used when the 8- or 16-bit bus space is specified.
BC3	Output	CSC	● Strobe signal that indicates (when low) that D31 to D24 are valid during access to an external address space in single-write strobe mode, active-low. ● This pin is not used when the 8- or 16-bit bus space is specified.
CS0[*3]	Output	CSC	Chip select signal for area 0 (CS0), active-low

Figure 13. RA8 External Memory Bus Pins and its Functions

Pin name	I/O	Related functions	Description
CS1[*3]	Output	CSC	Chip select signal for area 1 (CS1), active-low
CS2[*3]	Output	CSC	Chip select signal for area 2 (CS2), active-low
CS3[*3]	Output	CSC	Chip select signal for area 3 (CS3), active-low
CS4	Output	CSC	Chip select signal for area 4 (CS4), active-low
CS5	Output	CSC	Chip select signal for area 5 (CS5), active-low
CS6	Output	CSC	Chip select signal for area 6 (CS6), active-low
CS7	Output	CSC	Chip select signal for area 7 (CS7), active-low
RD	Output	CSC	Strobe signal that indicates that a read from an external address space (CS0 to CS7) is in progress, active-low.
WR0/WR[*4]	Output	CSC	• WR0 signal is a strobe signal that indicates that a write to an external address space is in progress in byte strobe mode, and D07 to D00 are valid, active-low. • WR signal is a strobe signal that indicates that a write to an external address space is in progress in single-write strobe mode, active-low. • When an 8-bit bus space is specified, this output pin is held low during a write access regardless of the write access mode.
WR1	Output	CSC	• Strobe signal that indicates that D15 to D08 are valid during a write to an external address space in byte strobe mode, active-low. • This signal is invalid in single-write strobe mode. • This pin is not used when the 8-bit bus space is specified.
WR2	Output	CSC	• Strobe signal that indicates that D23 to D16 are valid during a write to an external address space in byte strobe mode, active-low. • This signal is invalid in single-write strobe mode. • This pin is not used when the 8- or 16-bit bus space is specified.
WR3	Output	CSC	• Strobe signal that indicates that D31 to D24 are valid during a write to an external address space in byte strobe mode, active-low. • This signal is invalid in single-write strobe mode. • This pin is not used when the 8- or 16-bit bus space is specified.
ALE	Output	CSC	Address latch signal when address/data multiplexed bus is selected.
WAIT	Input	CSC	Wait request signal used when accessing the external address space (CS0 to CS7), active-low
CKE	Output	SDRAMC	Clock enable signal
SDCS	Output	SDRAMC	Chip select signal, active-low
RAS	Output	SDRAMC	Row address strobe signal, active-low
CAS	Output	SDRAMC	Column address strobe signal, active-low
WE	Output	SDRAMC	Write enable signal, active-low
DQM0	Output	SDRAMC	I/O data mask enable signal for DQ07 to DQ00
DQM1	Output	SDRAMC	I/O data mask enable signal for DQ15 to DQ08
DQM2	Output	SDRAMC	I/O data mask enable signal for DQ23 to DQ16
DQM3	Output	SDRAMC	I/O data mask enable signal for DQ31 to DQ24

Figure 14. RA8 External Memory Bus Pins and its Functions

Note 1: The EBCLK and the SDCLK pin functions are shared by the CS area controller (CSC) and the SDRAM area controller (SDRAMC). When using the CSC and the SDRAMC simultaneously, the SDCLK pin function is valid.

Note 2. The A23 to A00 pin functions are shared by the CSC and the SDRAMC.
When using the CSC only: The A00 and BC0 pin functions share the same pin, and either becomes valid according to the area, with the function being A00 in byte strobe mode and BC0 in single-write strobe mode. Setting the 8-bit external bus width is prohibited in single-write strobe mode.
When using the SDRAMC only: The A15 to A00 pin functions are valid. The A00 and DQM1 pin functions share the same pin, and either pin function becomes valid according to the external bus width. When selecting 8-bit bus width, the pin function is A00. When selecting 16-bit bus width, the pin function is DQM1. When using the CSC and the SDRAMC simultaneously: The A23 to A16 pin

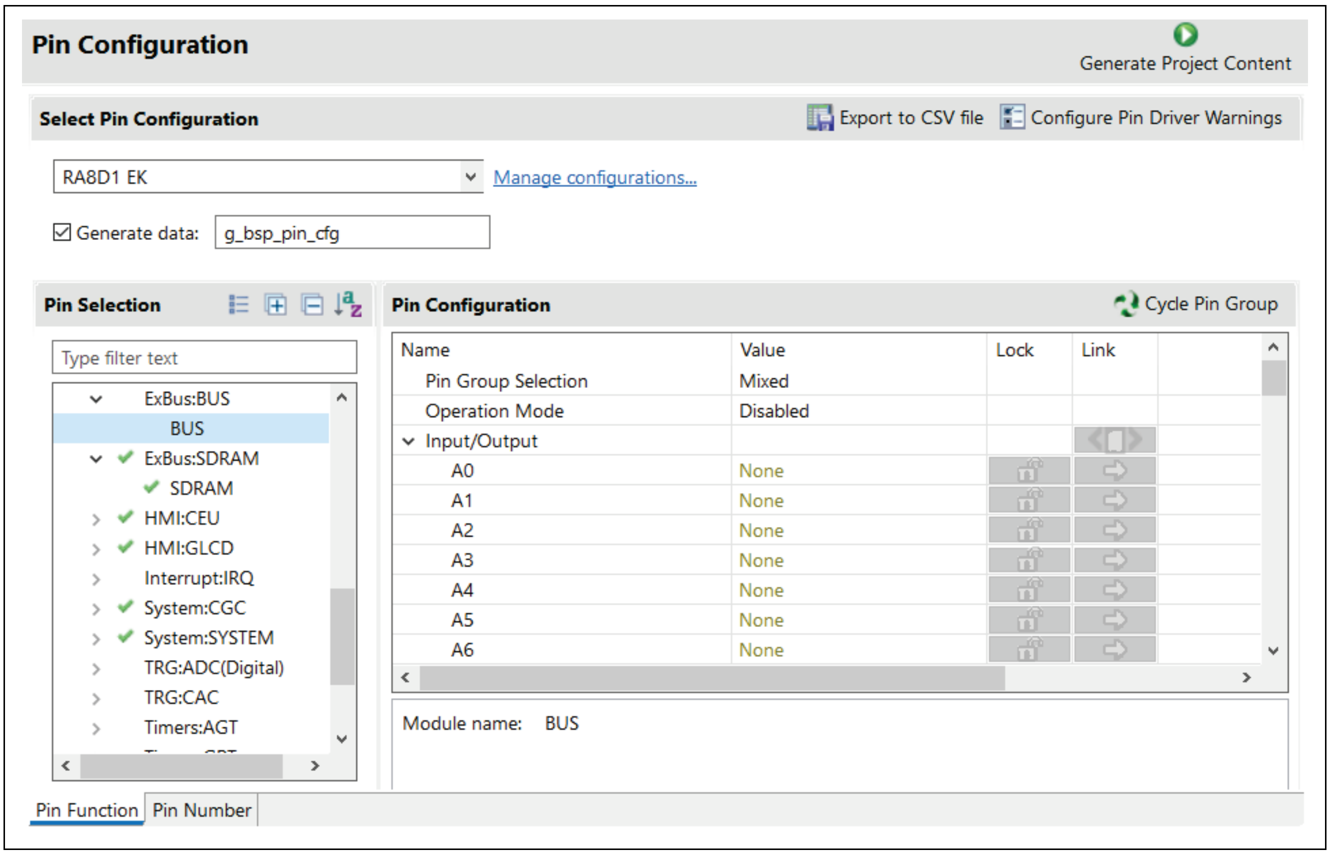
functions are valid for CSC. The A15 to A00 pin functions are shared by the CSC and the SDRAMC. In the SDRAMC functions, the A00 and the DQM1 pin functions work as described. In the CSC functions, the A00 and the BC0 pin function works as described.

Note 3: The CS0 to CS3 (CSC) and SDRAMC pin functions share the same pin. When using the CSC and the SDRAMC simultaneously, the CS0 to CS3 pin functions are invalid.

Note 4. The WR0 signal and WR signal are identical. The WR0 signal is referred to as WR in single-write strobe mode.

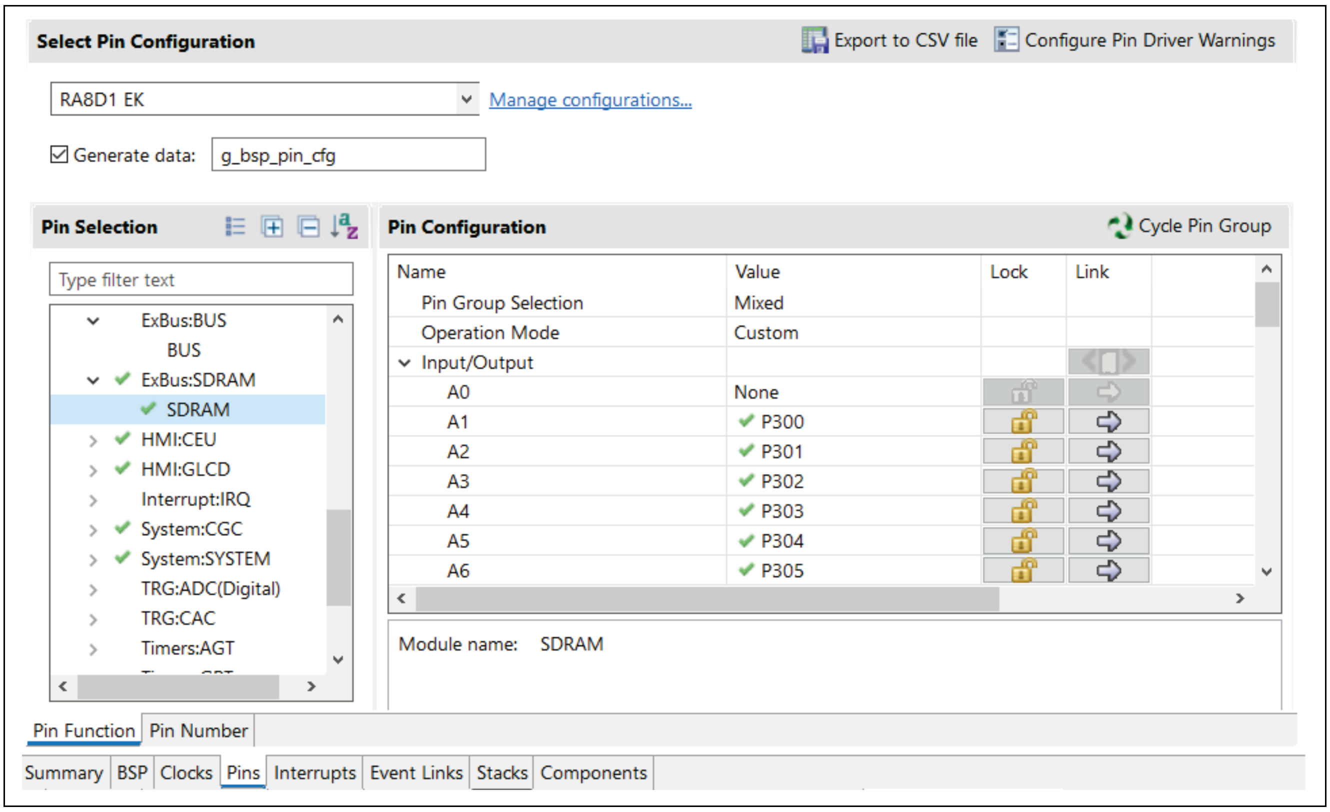
The external bus has a data alignment function to control which byte of the data bus (D31 to D24, D23 to D16, D15 to D08, D07 to D00) is used when accessing the external address space (the CS and SDRAM areas). Alignment is based on the bus specifications of the area to be accessed (8-bit, 16-bit, or 32-bit bus space), the data size, and the endian order.

FSP Configurator View
From the FSP perspective, the same signals can also be seen on the configurator **Pins** tab > **Peripherals** > **ExBus:Bus** > **Bus**.



Name	Value	Lock	Link
Pin Group Selection	Mixed		
Operation Mode	Disabled		
Input/Output			
A0	None		
A1	None		
A2	None		
A3	None		
A4	None		
A5	None		
A6	None		

Figure 15. RA8 External Memory Bus Pins Using FSP Configurator



Select Pin Configuration Export to CSV file Configure Pin Driver Warnings

RA8D1 EK Manage configurations...

☑ Generate data: g_bsp_pin_cfg

Pin Selection

Type filter text

- ExBus:BUS
  - BUS
- ✓ ExBus:SDRAM
  - ✓ SDRAM
- ✓ HMI:CEU
- ✓ HMI:GLCD
- Interrupt:IRQ
- ✓ System:CGC
- ✓ System:SYSTEM
- TRG:ADC(Digital)
- TRG:CAC
- Timers:AGT

Pin Configuration Cycle Pin Group

Name	Value	Lock	Link
Pin Group Selection	Mixed		
Operation Mode	Custom		
Input/Output			
A0	None		
A1	✓ P300		
A2	✓ P301		
A3	✓ P302		
A4	✓ P303		
A5	✓ P304		
A6	✓ P305		

Module name: SDRAM

Pin Function | Pin Number

Summary | BSP | Clocks | Pins | Interrupts | Event Links | Stacks | Components

Figure 16. RA8 External Memory Bus Pins and its Functions

2.3 How to Interface External Memory to RA8

The typical external memories which can be interfaced to the RA8 MCU are SDRAM, OSPI Flash and QSPI flash. RA8 provides additional 128 MB (16 MB X 8) CS area using which the additional RAM can be added to the system using parallel interface for faster access of data in a system where it goes beyond 1 MB.

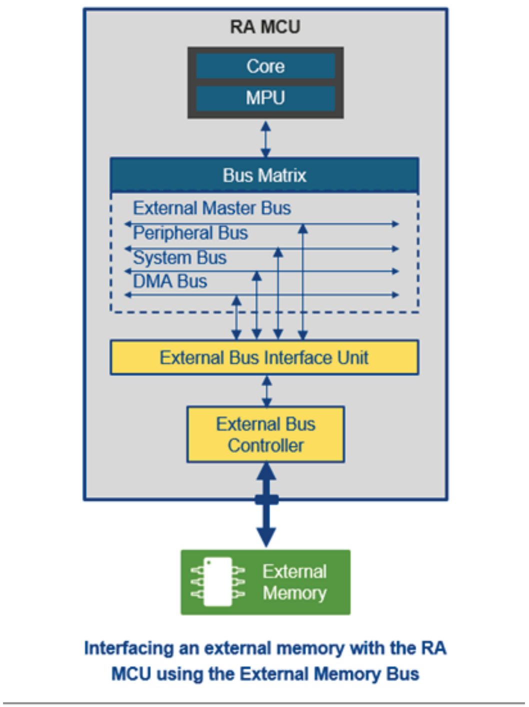


Figure 17. RA8 External Memory Interface Overview

The RA8 has a dedicated SDRAM controller and OSPI bus interface unit with bus controller through which the external SDRAM, OSPI RAM, OSPI Flash or QSPI flash memory can be interfaced.

For interfacing the OSPI Flash to RA8 MCU, it uses the External Octal Bus interface (EOBI) which is a serial interface. The Figure 18 shows the high-level block diagram of the external bus interface to the OSPI external memory.

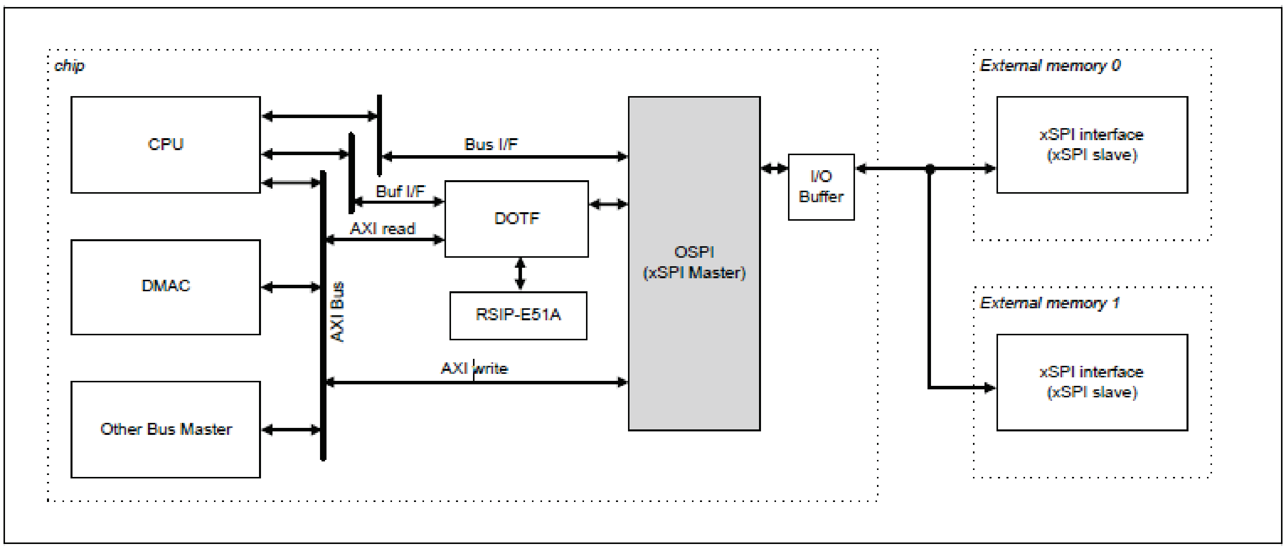


Figure 18. RA8 External Memory - OSPI Interface Overview

Figure 19 shows the pin name and functions which are used for connecting the MCU to the OSPI memory.

Pin name	I/O	Function
OM_SCLK	Output	Clock Positive
OM_SCLKN	Output	Clock Negative
OM_CS0	Output	Chip Select for slave0
OM_CS1	Output	Chip Select for slave1
OM_DQS	I/O	Read Data Strobe / Write Data Mask
OM_SIO0	I/O	Data 0 input/output
OM_SIO1	I/O	Data 1 input/output
OM_SIO2	I/O	Data 2 input/output
OM_SIO3	I/O	Data 3 input/output
OM_SIO4	I/O	Data 4 input/output
OM_SIO5	I/O	Data 5 input/output
OM_SIO6	I/O	Data 6 input/output
OM_SIO7	I/O	Data 7 input/output
OM_RESET	Output	Master reset status for slave0,1
OM_RSTO1	Input	Slave reset status for slave1
OM_ECSINT1	Input	Interrupt for slave1 / Error Correction Status for slave1
OM_WP1	Output	Write Protect for slave1

Figure 19. RA8 External Memory OSPI Bus Pins and Functions

The sample snapshot of the OSPI interface to EK-RA8M1 is shown in Figure 20. For more detailed example on interfacing the OSPI flash to RA8 MCU, please refer the schematics of the EK-RA8M1 evaluation kit.

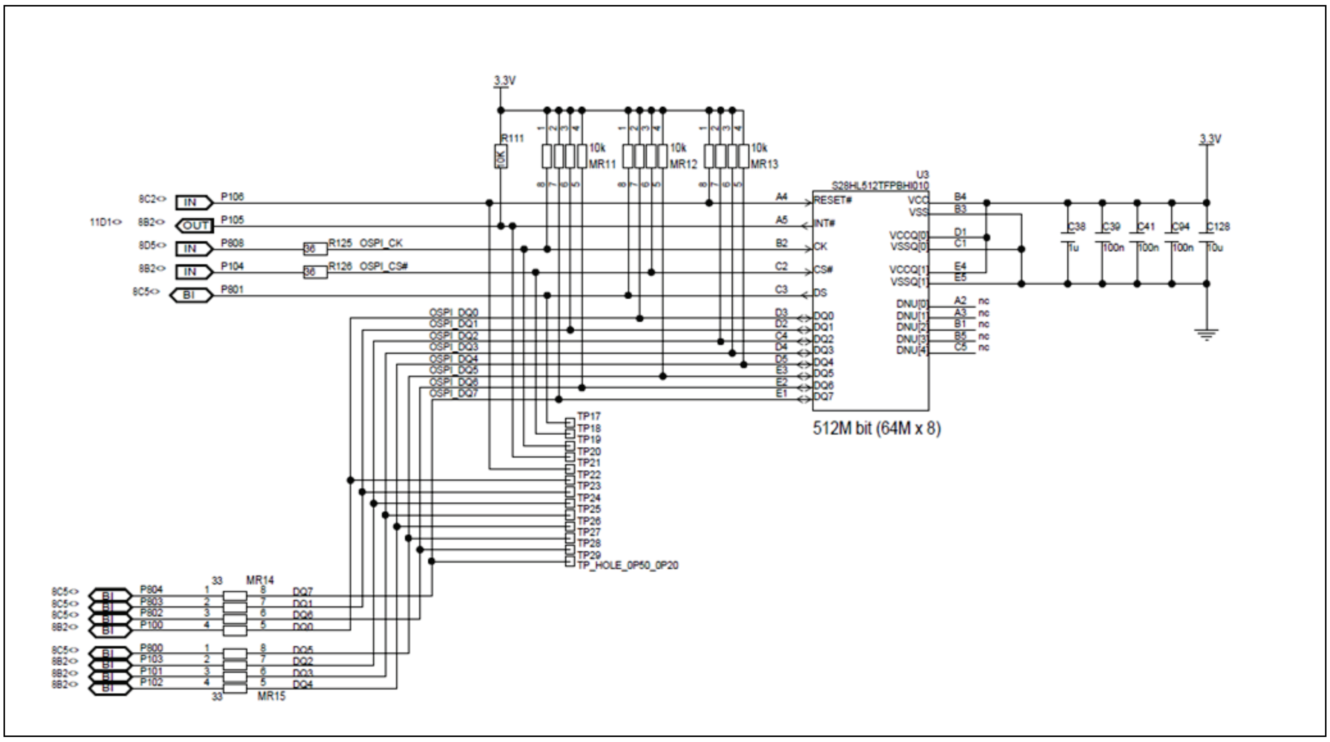


Figure 20. RA8 External Memory OSPI Interface Example

For interfacing the SDRAM, RA8 MCU uses the External memory CSC and SDRAM Bus Interface (ECBI), a parallel bus interface, which means you are required to connect a set of data, address, and control lines between microcontroller and SDRAM chip. The specific pins and connection required will depend on the Data bit mode you are trying to operate the SDRAM. RA8 MCU supports 8,16, and 32 bit modes. Refer to the UM of the MCU for pin assignments and connection details.

2.4 RA8 Internal Memory

The RA8 MCU provides an on-chip, high-density SRAM module with either parity-bit checking or Error Correction Code (ECC). SRAM0 is ECC. SRAM1 is Parity check. Also 128 KB of Tightly coupled memory for Instruction and Data (ITCM and DTCM) is available. For Low Power Mode operation, 1KB of standby SRAM is available. The MCU also provides L1 Instruction cache (16 KB) and Data Cache (16 KB) as part of the core. Caches are used to store frequently accessed data, which can help improve the performance and reduce the load on the underlying systems. However, introducing caches into the system may degrade deterministic behavior of operation.

For program storage, the MCU provides flash memory. A code memory of 2 MB is available for the code flash. Data flash memory of 12 KB is available for non-volatile storage of data. Table 5 provides the internal memory type and its size available on the RA8M1 MCU.

Table 5. RA8M1 Memory

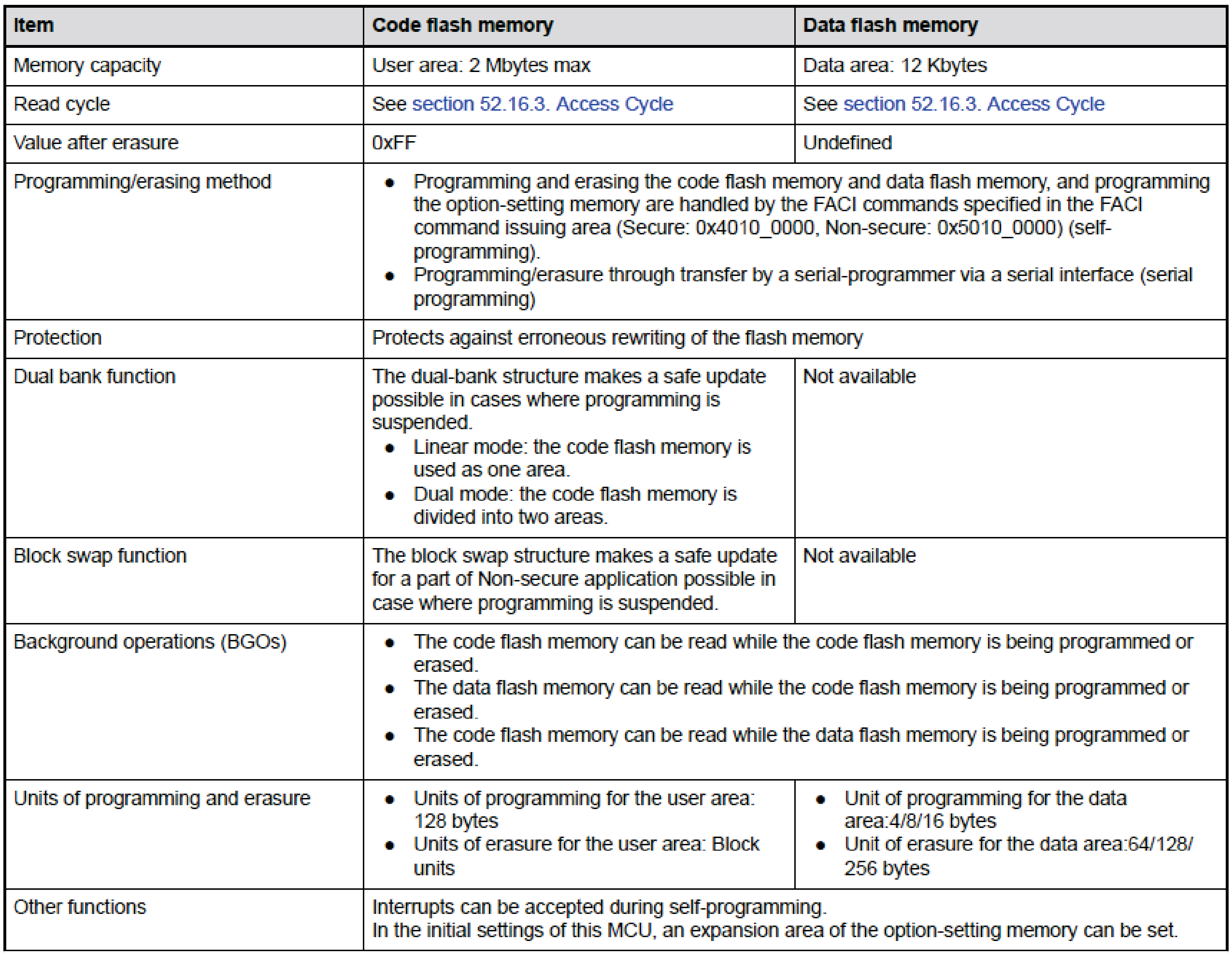
Internal Memory Type	Size	Description
Code flash memory	2MB	A Maximum of 2 MB of code flash memory for Program storage.
Data flash memory	12KB	Data flash memory for non-volatile storage of data.
Option-setting memory		The option-setting memory determines the state of the MCU after a reset.
SRAM	896KB	On-chip high-speed SRAM with either parity bit or Error Correction Code (ECC). SRAM0 (384KB) is ECC. SRAM1(512KB) is Parity check.
ITCM	64KB	Tightly coupled Memory for Instruction with ECC (8KB x 8 block)
DTCM	64KB	Tightly coupled Memory for Data with ECC (8KB x 8 block)
Standby SRAM	1KB	On-chip SRAM that can retain data in Deep Software Standby mode 1.
ROM		On-chip immutable ROM contains First Stage Bootloader (FSBL)
Instruction Cache	16KB	Instruction Cache of 16KB with ECC
Data Cache	16KB	Data Cache of 16KB with ECC

2.4.1 Internal Flash Memory

Internal flash memory can be mainly divided into Code flash and Data flash sections. From Figure 3, you can see the code flash is aliased at an address of (0x0200_0000 - 0x022F_8000) for Secure Access and (0x1200_0000- 0x122F_8000) for non-secure access. Similarly, the data flash is placed at an address of (0x2700_0000 - 0x2700_3000) for secure access and (0x3700_0000 - 0x3700_3000) for non-secure access. In addition, on-chip flash (Factory Flash) is placed at an address of (0x0300_80F0 - 0x0300_81B4) for secure access and (0x1300_80F0 - 0x1300_81B4) for non-secure access. On-chip flash (option-setting memory) is placed at an address of (0x0300_A100 - 0x0300_A300, 0x2703_0050 - 0x2703_0400) for secure access and (0x1300_A100 - 0x1300_A300, 0x3703_0050- 0x3703_0400) for non-secure access.

The Flash Control Unit (FCU) controls programming and erasure of the flash memory. The Flash Application Command Interface (FACI) controls the FCU in accordance with the specified FACI commands. The FCLK must be at least 4 MHz to perform programming and erasing on internal Code flash and data flash.

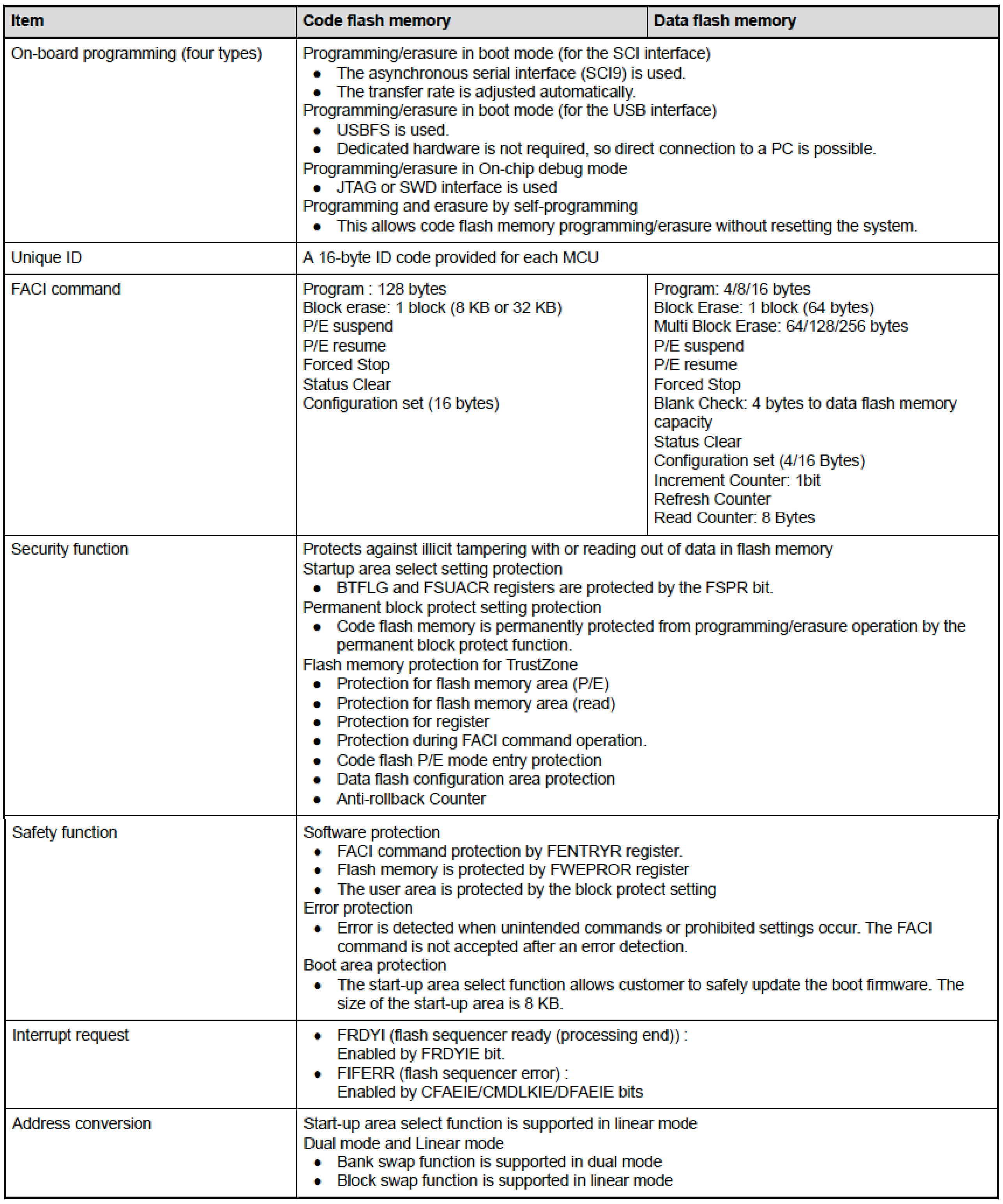
Figure 21 shows example specifications of code flash memory and data flash memory.



Item	Code flash memory	Data flash memory
Memory capacity	User area: 2 Mbytes max	Data area: 12 Kbytes
Read cycle	See section 52.16.3. Access Cycle	See section 52.16.3. Access Cycle
Value after erasure	0xFF	Undefined
Programming/erasing method	• Programming and erasing the code flash memory and data flash memory, and programming the option-setting memory are handled by the FACI commands specified in the FACI command issuing area (Secure: 0x4010_0000, Non-secure: 0x5010_0000) (self-programming). • Programming/erasure through transfer by a serial-programmer via a serial interface (serial programming)	
Protection	Protects against erroneous rewriting of the flash memory	
Dual bank function	The dual-bank structure makes a safe update possible in cases where programming is suspended. • Linear mode: the code flash memory is used as one area. • Dual mode: the code flash memory is divided into two areas.	Not available
Block swap function	The block swap structure makes a safe update for a part of Non-secure application possible in case where programming is suspended.	Not available
Background operations (BGOs)	• The code flash memory can be read while the code flash memory is being programmed or erased. • The data flash memory can be read while the code flash memory is being programmed or erased. • The code flash memory can be read while the data flash memory is being programmed or erased.	
Units of programming and erasure	• Units of programming for the user area: 128 bytes • Units of erasure for the user area: Block units	• Unit of programming for the data area:4/8/16 bytes • Unit of erasure for the data area:64/128/ 256 bytes
Other functions	Interrupts can be accepted during self-programming. In the initial settings of this MCU, an expansion area of the option-setting memory can be set.	

Figure 21. RA8 Internal Flash Memory

The code flash is designed to store user application code and constant data. The data flash is designed to store information that may be updated from time to time such as configuration parameters, user settings, or logged data. The units of programming and erasure in the data flash area are much smaller than that of the code flash (4 bytes for data flash versus 128 bytes for code flash). Both the data flash and code flash areas can be programmed or erased by application code that is, through self-programming. This enables field firmware updates without needing to connect an external programming tool. Renesas FSP provides HAL layer drivers for modifying both code flash memory and data flash memory.



Item	Code flash memory	Data flash memory
On-board programming (four types)	Programming/erasure in boot mode (for the SCI interface) • The asynchronous serial interface (SCI9) is used. • The transfer rate is adjusted automatically. Programming/erasure in boot mode (for the USB interface) • USBFS is used. • Dedicated hardware is not required, so direct connection to a PC is possible. Programming/erasure in On-chip debug mode • JTAG or SWD interface is used Programming and erasure by self-programming • This allows code flash memory programming/erasure without resetting the system.	
Unique ID	A 16-byte ID code provided for each MCU	
FACI command	Program : 128 bytes Block erase: 1 block (8 KB or 32 KB) P/E suspend P/E resume Forced Stop Status Clear Configuration set (16 bytes)	Program: 4/8/16 bytes Block Erase: 1 block (64 bytes) Multi Block Erase: 64/128/256 bytes P/E suspend P/E resume Forced Stop Blank Check: 4 bytes to data flash memory capacity Status Clear Configuration set (4/16 Bytes) Increment Counter: 1bit Refresh Counter Read Counter: 8 Bytes
Security function	Protects against illicit tampering with or reading out of data in flash memory Startup area select setting protection • BTFLG and FSUACR registers are protected by the FSPR bit. Permanent block protect setting protection • Code flash memory is permanently protected from programming/erasure operation by the permanent block protect function. Flash memory protection for TrustZone • Protection for flash memory area (P/E) • Protection for flash memory area (read) • Protection for register • Protection during FACI command operation. • Code flash P/E mode entry protection • Data flash configuration area protection • Anti-rollback Counter	
Safety function	Software protection • FACI command protection by FENTRYR register. • Flash memory is protected by FWEPROR register • The user area is protected by the block protect setting Error protection • Error is detected when unintended commands or prohibited settings occur. The FACI command is not accepted after an error detection. Boot area protection • The start-up area select function allows customer to safely update the boot firmware. The size of the start-up area is 8 KB.	
Interrupt request	• FRDYI (flash sequencer ready (processing end)) : Enabled by FRDYIE bit. • FIFERR (flash sequencer error) : Enabled by CFAEIE/CMDLKIE/DFAEIE bits	
Address conversion	Start-up area select function is supported in linear mode Dual mode and Linear mode • Bank swap function is supported in dual mode • Block swap function is supported in linear mode	

Figure 22. RA8 Internal Flash Memory Features and Functions

2.4.1.1 Code Flash Memory

Code Flash Memory in RA8 MCU is a non-volatile memory used for firmware that the MCU runs. This memory contains the instructions that the MCU executes when powered on. The process of programming or writing to code flash memory involves serial based programming tool such as Renesas Flash programmer or J-Link flash programmer.

RA8 code flash memory is divided into different regions, such factory flash regions, First Stage Bootloader, option settings memory.

Figure 23 shows the memory map of code flash memory in linear mode. Figure 24 shows the memory map of code flash memory in dual mode. RA8 MCU can use the code flash memory as two bank areas by using the dual bank function. This dual-bank structure allows a safe update of a program while a user program is running. The user area of the code flash memory in this MCU is divided into 8 KB and 32 KB blocks, which serve as the units of erasure.

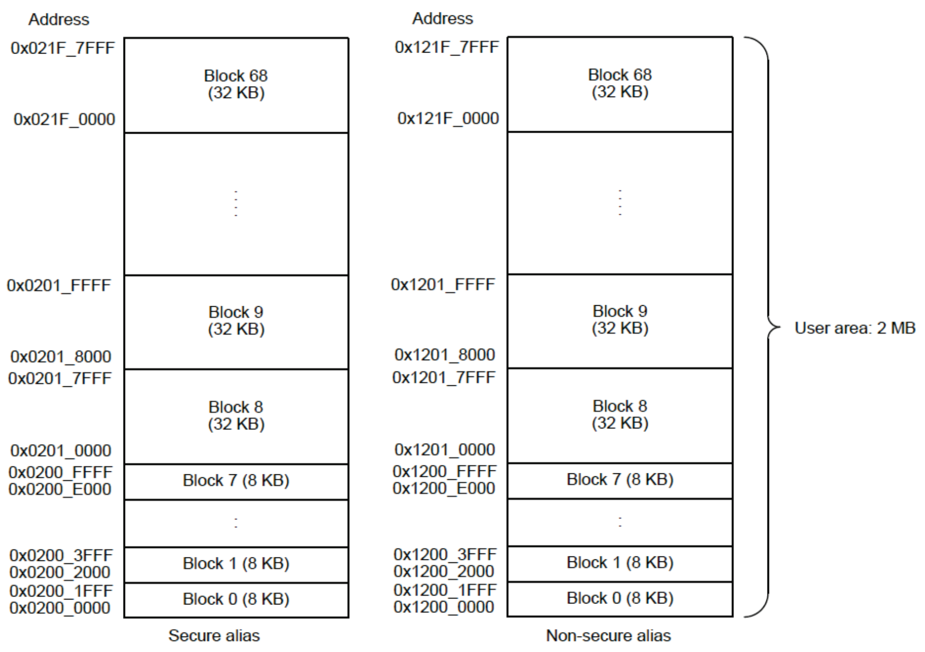


Figure 23. RA8 Map of the Code Flash Memory in Linear Mode

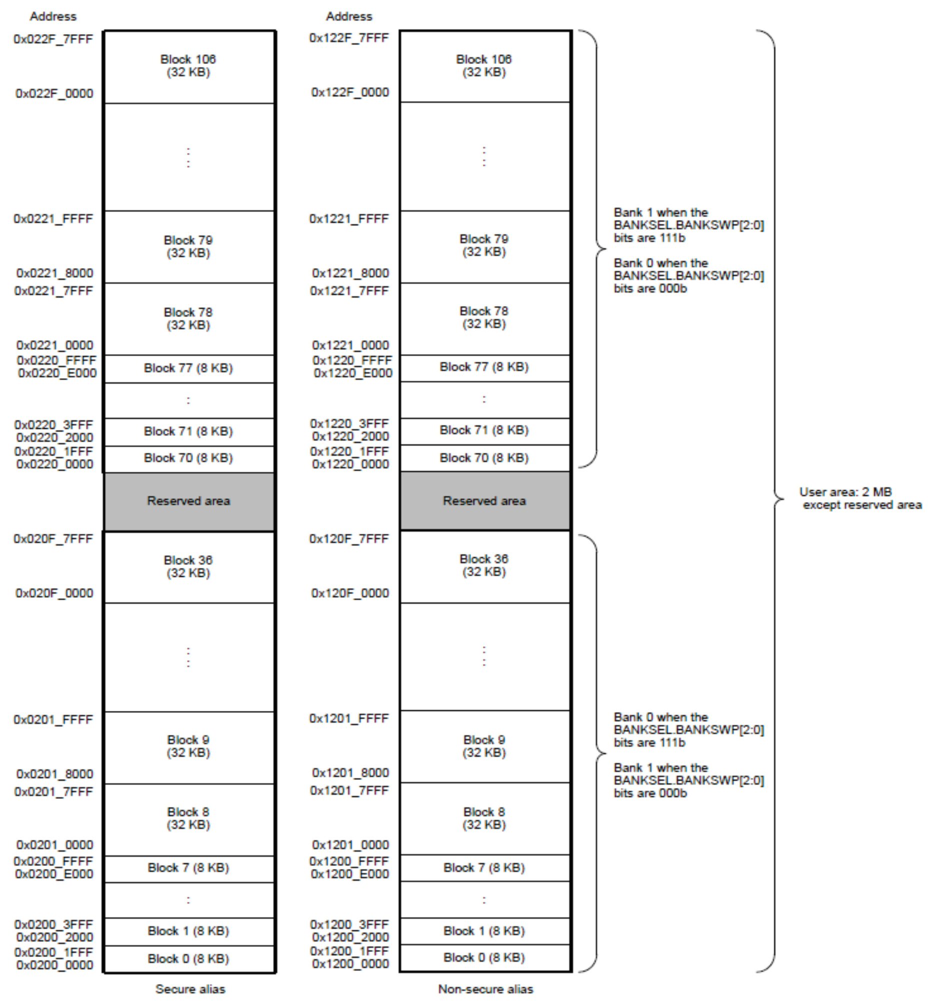


Figure 24. RA8 Map of the Code Flash Memory in Dual Mode

2.4.1.2 Data Flash Memory

Data flash memory in a RA8 MCU is a type of non-volatile memory designed for storing data that needs to be retained even when the MCU loses power. It is distinct from the code flash memory that stores the MCU's firmware or software code. Unlike many other non-volatile memory technologies like EEPROM, data flash memory in an MCU often supports random access. This means you can read and write data at specific memory addresses, similar to how you would interact with RAM.

Data flash memory in MCUs is designed for relatively fast read and write operations compared to external storage devices like SD cards. This makes it suitable for quickly accessing and modifying data used by the MCU during its operation.

RA8 data flash memory is limited in size to 12 KB compared to external storage options, like SD cards or NAND flash memory. To extend the lifespan of the data flash memory, FSP provides wear-leveling algorithms. These algorithms ensure that write and erase cycles are distributed evenly across the memory cells, which is especially important for flash memory technologies that have a limited number of write-erase cycles.

The data area of the data flash memory in this MCU is divided into 64 byte blocks, with each being a unit for erasure. Figure 25 shows the mapping of the data flash memory.

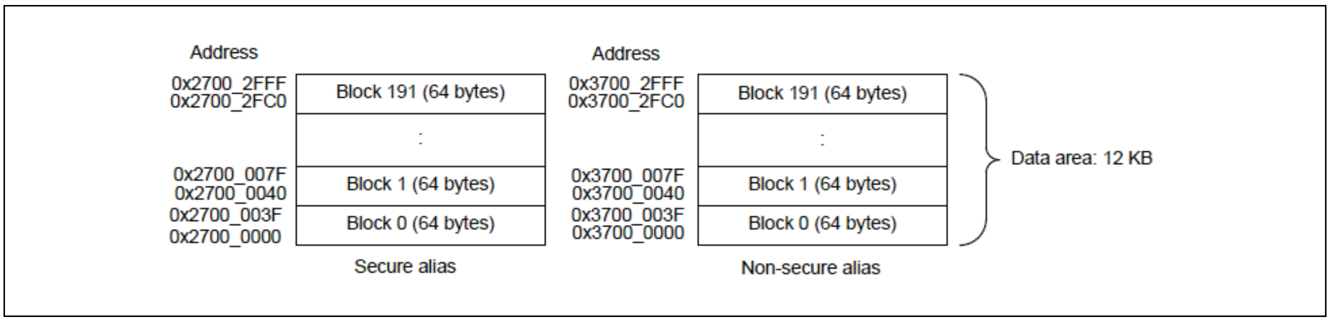


Figure 25. RA8 Map of the Data Flash Memory

Common uses for data flash memory in MCUs include:

- Creating file systems to store the system files, user files, and so forth.
- Storing configuration parameters and settings for the MCU or connected devices.
- Logging data such as sensor readings, event timestamps, or system status information.
- Saving user preferences, credentials and profiles in applications where customization is needed.
- Retaining calibration data for sensors and measurement instruments.
- Storing encryption keys, security-related data, or device-specific information.

2.4.2 On Chip ROM

RA8's On-chip immutable read-only memory (ROM) contains the First Stage Bootloader (FSBL) which is a bootloader that is permanently embedded in read-only memory, and it cannot be altered or modified. The primary purpose of such a bootloader is to provide the initial bootstrapping process for the MCU while ensuring its security and integrity of the first application (typically, an OEM developed bootloader) executed. FSBL is programmed into the ROM during the manufacturing process. This cannot be altered without physically altering the hardware, making it immune to external tampering or unauthorized modifications. FSBL enhances the security of a device by preventing unauthorized or malicious code from being loaded during the boot process. This ensures that only trusted and verified code can be executed.

The primary function of the FSBL is to initialize the hardware and load the second-stage bootloader or main firmware from a secure and trusted source, such as a secure storage medium or a secure boot mechanism. In RA8 MCU, the FSBL plays a critical role in implementing secure boot procedures. It verifies the authenticity and integrity of the second-stage bootloader or firmware before allowing it to execute. This helps protect against firmware-level attacks and unauthorized code execution. The FBSL can also facilitate firmware recovery in case of a system failure or corruption of the main firmware. It can be designed to have a minimal and stable codebase for restoring the system to a known good state.

The operation of the FSBL is managed by the Option-Setting Registers. The registers used during development phase can be configured using the BSP tab property setting. The FSBL when enabled can verify the integrity and authenticity of an OEM bootloader (OEM_BL) or a normal application (with no bootloader capability) starting at the application executable memory. The OEM_BL or a normal application is verified when it is initially programmed as well as prior to execution. For RA8 MCUs, user can use the FSP MCUboot module to establish the OEM_BL or use their own custom bootloader.

For specifications on secure boot operations, user can reference the *RA8 User's Manual: Hardware "section 43.7 Secure Boot"*. For examples on how to use the Secure Boot feature, user can reference the application project *R11AN0774 (OEM_BL and Application Design Using the RA8 First Stage Bootloader)*.

2.4.3 Option Setting Memory

The option-setting memory determines the state of the MCU after a reset. It is allocated to the configuration setting area and the program flash area of the flash memory. The available methods of setting are different for the two areas. Option setting memory may differ in size and layout for Cortex-M85 based devices.

The registers are detailed in the "*Option Setting Memory" chapter in the Hardware User's Manual*.

Option-setting memory registers have a discontinuous address space layout in the code flash memory. Sometimes the registers may be located in a portion of the flash memory which is near reserved areas in the internal flash. It is possible that some customers may inadvertently store data in the registers of option-setting memory or write into the reserved area in internal flash. This may result in improper functionality. It is advised that the user check to ensure that no unwanted data is written to these locations prior to programming the internal flash by reviewing the map file or s-record files generated by the compiler upon creation of the binary. For instance, settings in the flash option registers can enable the Independent Watchdog Timer (IWDT) immediately after reset. If data stored in program ROM inadvertently overlaps the option setting memory register, it is possible to turn on the IWDT on without realizing it. This may cause the debugger to have communication problems with the board. Additionally, security attribution of code flash option-setting memory can impact which value is applied at runtime, so user must confirm that required values are programmed into the option-setting memory.

The image below shows the option setting memory, which consists of the option function select registers on RA8M1, a Cortex-M85 device.

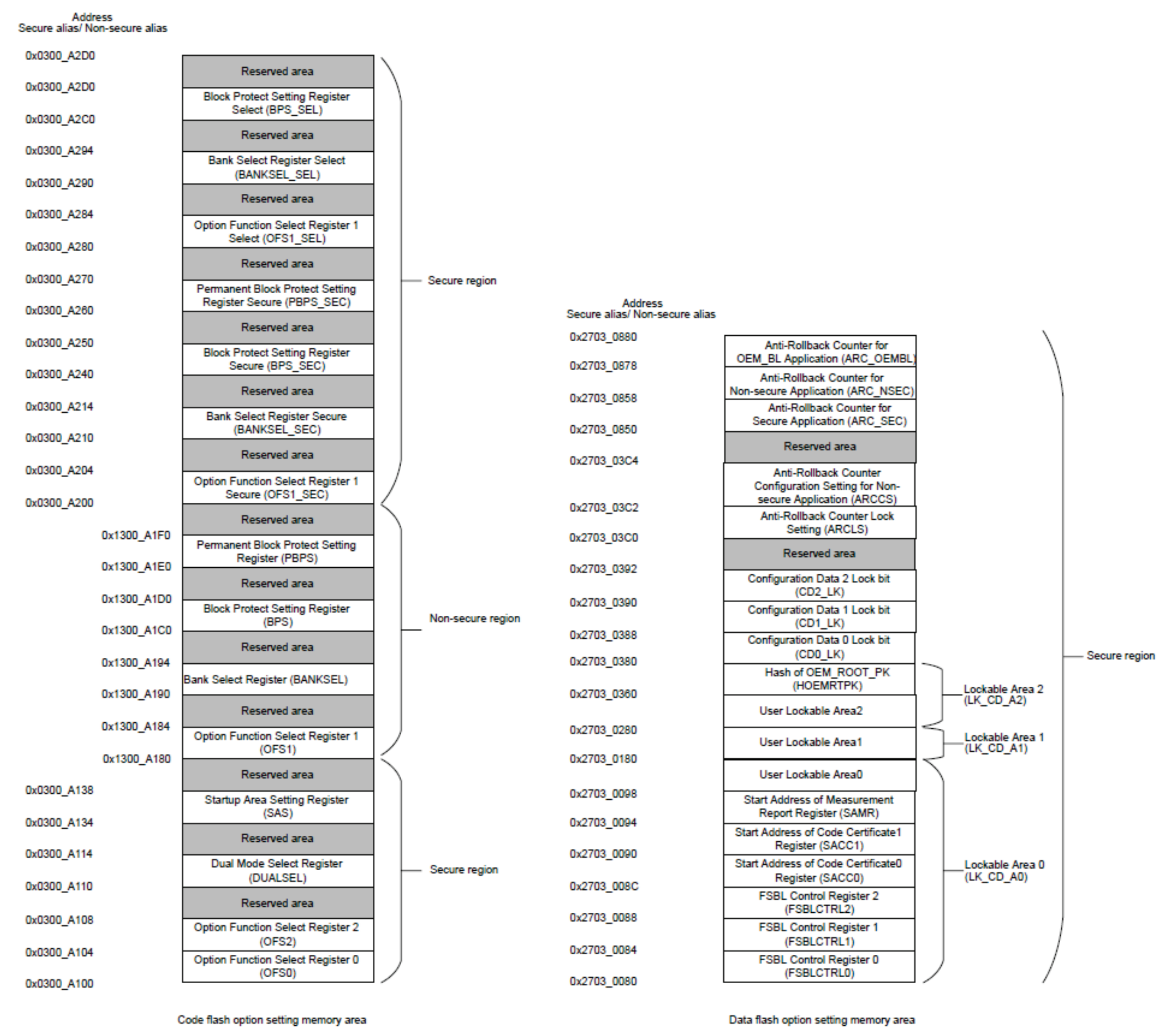

Address
Secure alias/ Non-secure alias
0x0300_A2D0
0x0300_A2D0
0x0300_A2C0
0x0300_A294
0x0300_A290
0x0300_A284
0x0300_A280
0x0300_A270
0x0300_A260
0x0300_A250
0x0300_A240
0x0300_A214
0x0300_A210
0x0300_A204
0x0300_A200
0x1300_A1F0
0x1300_A1E0
0x1300_A1D0
0x1300_A1C0
0x1300_A194
0x1300_A190
0x1300_A184
0x1300_A180
0x0300_A138
0x0300_A134
0x0300_A114
0x0300_A110
0x0300_A108
0x0300_A104
0x0300_A100
Reserved area
Block Protect Setting Register Select (BPS_SEL)
Reserved area
Bank Select Register Select (BANKSEL_SEL)
Reserved area
Option Function Select Register 1 Select (OFS1_SEL)
Reserved area
Permanent Block Protect Setting Register Secure (PBPS_SEC)
Reserved area
Block Protect Setting Register Secure (BPS_SEC)
Reserved area
Bank Select Register Secure (BANKSEL_SEC)
Reserved area
Option Function Select Register 1 Secure (OFS1_SEC)
Reserved area
Permanent Block Protect Setting Register (PBPS)
Reserved area
Block Protect Setting Register (BPS)
Reserved area
Bank Select Register (BANKSEL)
Reserved area
Option Function Select Register 1 (OFS1)
Reserved area
Startup Area Setting Register (SAS)
Reserved area
Dual Mode Select Register (DUALSEL)
Reserved area
Option Function Select Register 2 (OFS2)
Option Function Select Register 0 (OFS0)
Secure region
Non-secure region
Secure region
Code flash option setting memory area
Address
Secure alias/ Non-secure alias
0x2703_0880
0x2703_0878
0x2703_0858
0x2703_0850
0x2703_03C4
0x2703_03C2
0x2703_03C0
0x2703_0392
0x2703_0390
0x2703_0388
0x2703_0380
0x2703_0360
0x2703_0280
0x2703_0180
0x2703_0098
0x2703_0094
0x2703_0090
0x2703_008C
0x2703_0088
0x2703_0084
0x2703_0080
Anti-Rollback Counter for OEM_BL Application (ARC_OEMBL)
Anti-Rollback Counter for Non-secure Application (ARC_NSEC)
Anti-Rollback Counter for Secure Application (ARC_SEC)
Reserved area
Anti-Rollback Counter Configuration Setting for Non-secure Application (ARCCS)
Anti-Rollback Counter Lock Setting (ARCLS)
Reserved area
Configuration Data 2 Lock bit (CD2_LK)
Configuration Data 1 Lock bit (CD1_LK)
Configuration Data 0 Lock bit (CD0_LK)
Hash of OEM_ROOT_PK (HOEMRTPK)
User Lockable Area2
User Lockable Area1
User Lockable Area0
Start Address of Measurement Report Register (SAMR)
Start Address of Code Certificate1 Register (SACC1)
Start Address of Code Certificate0 Register (SACC0)
FSBL Control Register 2 (FSBLCTRL2)
FSBL Control Register 1 (FSBLCTRL1)
FSBL Control Register 0 (FSBLCTRL0)
Lockable Area 2 (LK_CD_A2)
Lockable Area 1 (LK_CD_A1)
Lockable Area 0 (LK_CD_A0)
Secure region
Data flash option setting memory area

Figure 26. RA8 Option Settings Memory Map

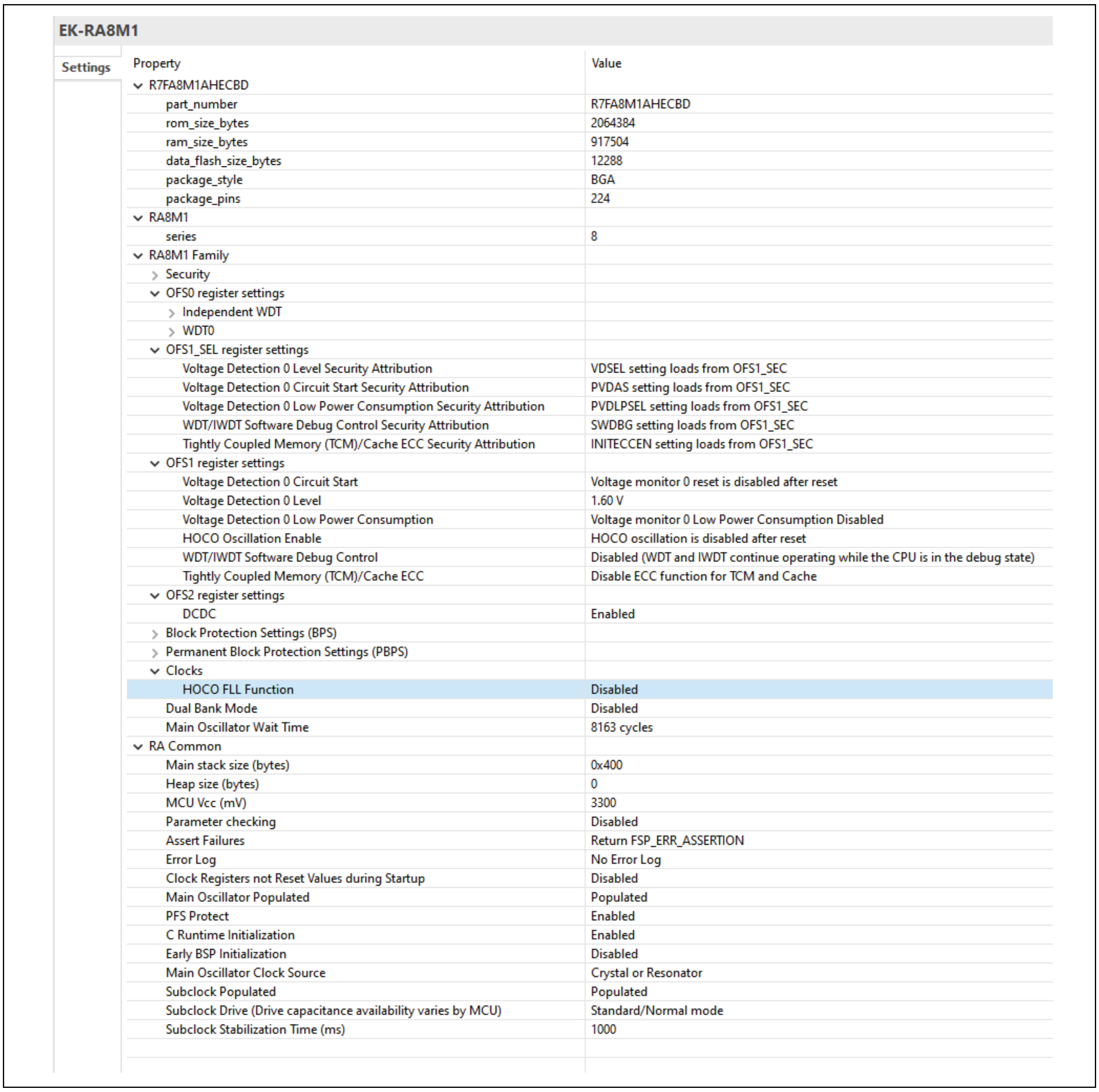
2.5 Option Setting Memory Registers

Following is a summary of the option setting memory registers. Make sure that they are configured properly before first programming the MCU for startup. To check the configuration, review the map file, and output file (using hex or s-record format) to confirm the values programmed into the option setting memory registers. See Figure 28 on memory usage view.

Table 6. RA8 Memory

MCU Sub-system Control Settings	
OFS0 register	Independent Watchdog Timer (IWDT) auto start
	IWDT timeout, frequency, windowing, interrupt type, and low power mode behavior
	Watchdog Timer (WDT) auto start
	WDT timeout, frequency, windowing, interrupt type, and low power mode behavior
OFS1 register	PVD0 startup settings after reset
	HOCO startup settings after reset
	Software Debug Control
	Set ECC function of TCM and CACHE
OFS2 register	DCDC enable at reset
Code Flash Memory Setting	
SAS register	Startup area selection
DUALSEL register	Code Flash memory mode upon reset
BANKSEL registers	Controls swapping of banks at the next reset
BPS, PBPS registers	Turns off erasing and programming capability of selected blocks of code flash memory. Note that when PBPS are set to disable the flash erase and programming, it can never be reverted.
Firmware Verification and Update Control Registers	
FSBLCTRLx registers	Controls operation of the First Stage Boot Loader which verifies the integrity and authenticity of an OEM boot loader (called OEM_BL) starting at the beginning of application executable memory.
SACCx register	Identifies Locations in code flash memory where x.509 certificates are stored for validating the application when FSBL is operating in Secure Boot mode.
	SACC0/1 is selected by the FSBL based on the MCU Start Area Selection and the Bank mode. User can reference the “Secure Boot” section in the Hardware User’s Manual to understand this selection process.
SAMR register	FSBL stores the measurement report to the SRAM address specified by the SAMR register
HOEMRTPK register	This register can be programmed only by the MCU boot firmware. It is programmed after a code image is validated. This register contains a processed Hash value.
CFGDxLOCK register	Specify write protection for corresponding Lockable Configuration Data Areas in Data Flash. Note that when the protection is enabled, it can never be reverted.
ARCLS register	Controls Anti Rollback Counter Lock functionality. Note that if this lock functionality is enabled, it can never be reverted. This register can be set using the Renesas Flash Programming (RFP) tool.
ARCCS register	Configures the Anti Rollback Counter operation for the non-secure application. Note that when this configuration is disabled, the “Increment Counter” or Read Counter” command cannot be issued anymore. This register can be set using the Renesas Flash Programming (RFP) tool.
ARC_SECn registers	Anti-Rollback Counter for Secure Application
ARC_NSECn registers	Anti-Rollback Counter for Non-Secure Application
ARC_OEMBLn registers	Anti-Rollback Counter for OEMBL

Renesas FSP Configurator supports setting of option memory in BSP settings, as shown in the following Figure 27 for RA8M1 MCU.



EK-RA8M1

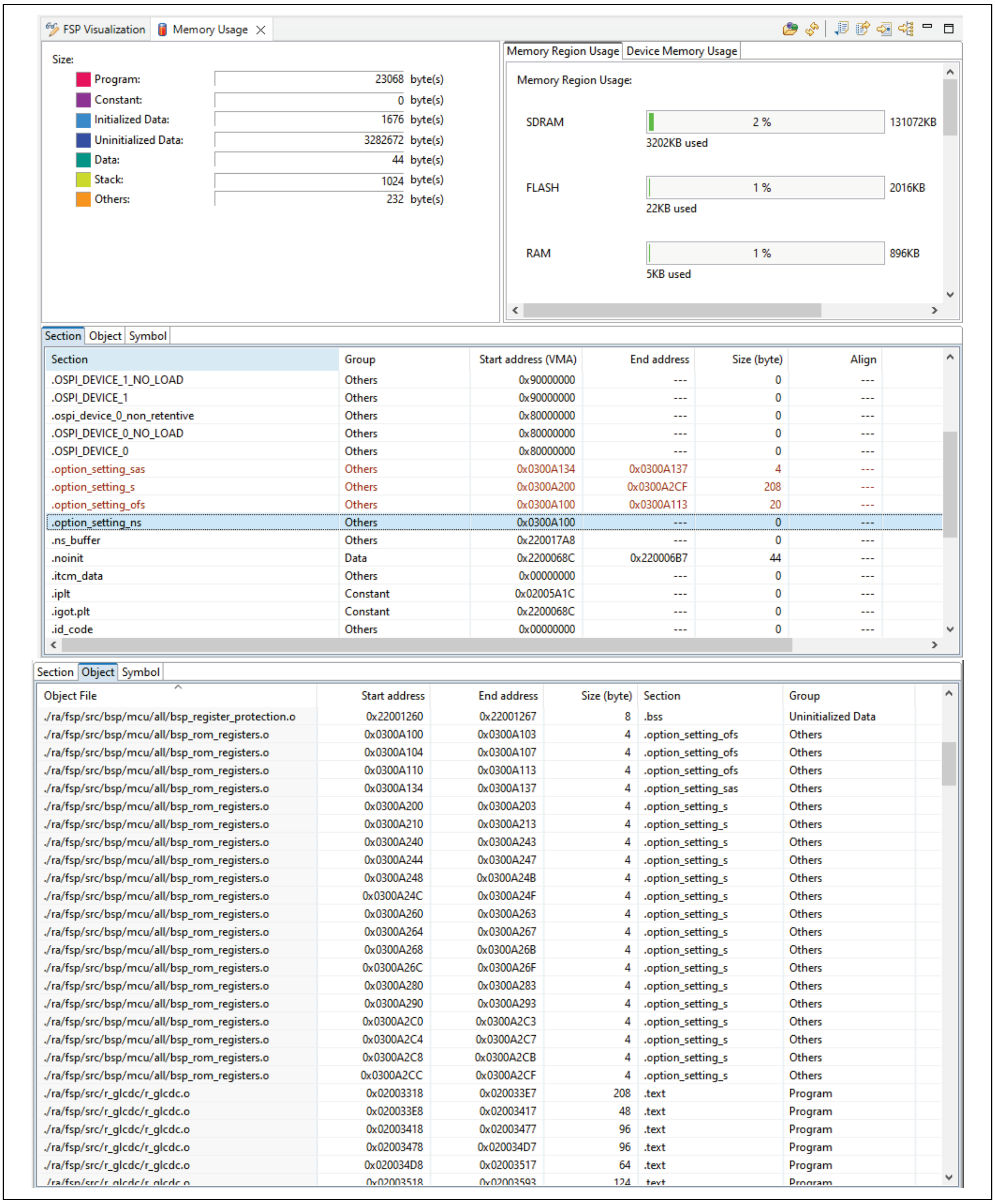
Settings

Property	Value
R7FA8M1AHECBD	
part_number	R7FA8M1AHECBD
rom_size_bytes	2064384
ram_size_bytes	917504
data_flash_size_bytes	12288
package_style	BGA
package_pins	224
RA8M1	
series	8
RA8M1 Family	
Security	
OFS0 register settings	
Independent WDT	
WDT0	
OFS1_SEL register settings	
Voltage Detection 0 Level Security Attribution	VDSEL setting loads from OFS1_SEC
Voltage Detection 0 Circuit Start Security Attribution	PVDAS setting loads from OFS1_SEC
Voltage Detection 0 Low Power Consumption Security Attribution	PVDLPSEL setting loads from OFS1_SEC
WDT/IWDT Software Debug Control Security Attribution	SWDBG setting loads from OFS1_SEC
Tightly Coupled Memory (TCM)/Cache ECC Security Attribution	INITECCEN setting loads from OFS1_SEC
OFS1 register settings	
Voltage Detection 0 Circuit Start	Voltage monitor 0 reset is disabled after reset
Voltage Detection 0 Level	1.60 V
Voltage Detection 0 Low Power Consumption	Voltage monitor 0 Low Power Consumption Disabled
HOCO Oscillation Enable	HOCO oscillation is disabled after reset
WDT/IWDT Software Debug Control	Disabled (WDT and IWDT continue operating while the CPU is in the debug state)
Tightly Coupled Memory (TCM)/Cache ECC	Disable ECC function for TCM and Cache
OFS2 register settings	
DCDC	Enabled
Block Protection Settings (BPS)	
Permanent Block Protection Settings (PBPS)	
Clocks	
HOCO FLL Function	Disabled
Dual Bank Mode	Disabled
Main Oscillator Wait Time	8163 cycles
RA Common	
Main stack size (bytes)	0x400
Heap size (bytes)	0
MCU Vcc (mV)	3300
Parameter checking	Disabled
Assert Failures	Return FSP_ERR_ASSERTION
Error Log	No Error Log
Clock Registers not Reset Values during Startup	Disabled
Main Oscillator Populated	Populated
PFS Protect	Enabled
C Runtime Initialization	Enabled
Early BSP Initialization	Disabled
Main Oscillator Clock Source	Crystal or Resonator
Subclock Populated	Populated
Subclock Drive (Drive capacitance availability varies by MCU)	Standard/Normal mode
Subclock Stabilization Time (ms)	1000

Figure 27. Option Memory Settings in FSP Configuration for RA8M1 MCU

You can also use the `objdump` command to check the settings of the option setting memory. Following is an example for dumping the option setting registers:

```
arm-none-eabi-objdump.exe -s --start-address=0x0300A100 --stop-
address=0x0300A300 elf_from_gcc.elf
```



Size:

Program:	23068	byte(s)
Constant:	0	byte(s)
Initialized Data:	1676	byte(s)
Uninitialized Data:	3282672	byte(s)
Data:	44	byte(s)
Stack:	1024	byte(s)
Others:	232	byte(s)

Memory Region Usage:

SDRAM	2 %	3202KB used	131072KB
FLASH	1 %	22KB used	2016KB
RAM	1 %	5KB used	896KB

Section	Group	Start address (VMA)	End address	Size (byte)	Align
.OSPI_DEVICE_1_NO_LOAD	Others	0x90000000	---	0	---
.OSPI_DEVICE_1	Others	0x90000000	---	0	---
.ospi_device_0_non_retentive	Others	0x80000000	---	0	---
.OSPI_DEVICE_0_NO_LOAD	Others	0x80000000	---	0	---
.OSPI_DEVICE_0	Others	0x80000000	---	0	---
.option_setting_sas	Others	0x0300A134	0x0300A137	4	---
.option_setting_s	Others	0x0300A200	0x0300A2CF	208	---
.option_setting_ofs	Others	0x0300A100	0x0300A113	20	---
.option_setting_ns	Others	0x0300A100	---	0	---
.ns_buffer	Others	0x220017A8	---	0	---
.noinit	Data	0x2200068C	0x220006B7	44	---
.itcm_data	Others	0x00000000	---	0	---
.iplt	Constant	0x02005A1C	---	0	---
.igot.plt	Constant	0x2200068C	---	0	---
.id_code	Others	0x00000000	---	0	---

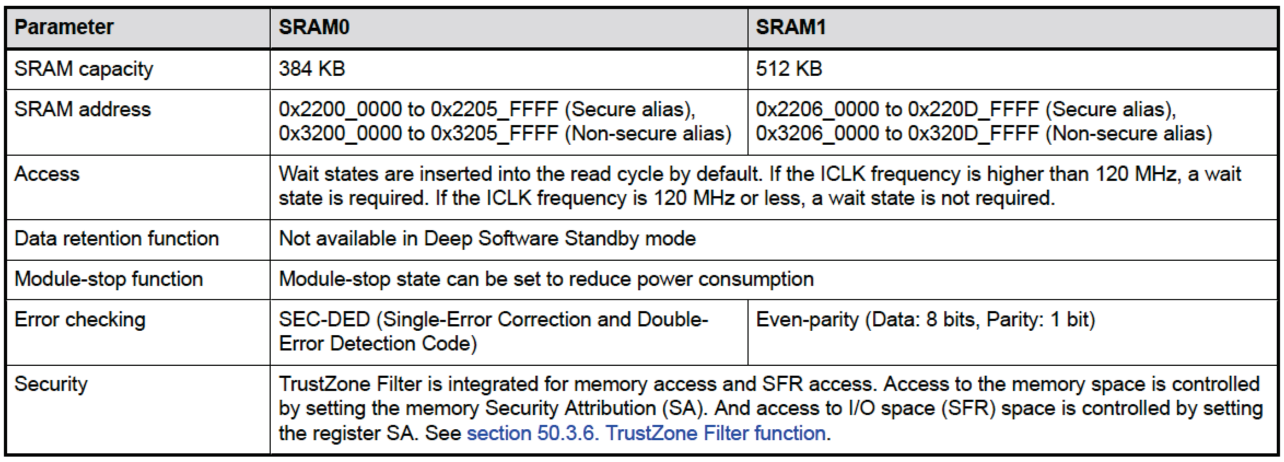
Object File	Start address	End address	Size (byte)	Section	Group
./ra/fsp/src/bsp/mcu/all/bsp_register_protection.o	0x22001260	0x22001267	8	.bss	Uninitialized Data
./ra/fsp/src/bsp/mcu/all/bsp_rom_registers.o	0x0300A100	0x0300A103	4	.option_setting_ofs	Others
./ra/fsp/src/bsp/mcu/all/bsp_rom_registers.o	0x0300A104	0x0300A107	4	.option_setting_ofs	Others
./ra/fsp/src/bsp/mcu/all/bsp_rom_registers.o	0x0300A110	0x0300A113	4	.option_setting_ofs	Others
./ra/fsp/src/bsp/mcu/all/bsp_rom_registers.o	0x0300A134	0x0300A137	4	.option_setting_sas	Others
./ra/fsp/src/bsp/mcu/all/bsp_rom_registers.o	0x0300A200	0x0300A203	4	.option_setting_s	Others
./ra/fsp/src/bsp/mcu/all/bsp_rom_registers.o	0x0300A210	0x0300A213	4	.option_setting_s	Others
./ra/fsp/src/bsp/mcu/all/bsp_rom_registers.o	0x0300A240	0x0300A243	4	.option_setting_s	Others
./ra/fsp/src/bsp/mcu/all/bsp_rom_registers.o	0x0300A244	0x0300A247	4	.option_setting_s	Others
./ra/fsp/src/bsp/mcu/all/bsp_rom_registers.o	0x0300A248	0x0300A24B	4	.option_setting_s	Others
./ra/fsp/src/bsp/mcu/all/bsp_rom_registers.o	0x0300A24C	0x0300A24F	4	.option_setting_s	Others
./ra/fsp/src/bsp/mcu/all/bsp_rom_registers.o	0x0300A260	0x0300A263	4	.option_setting_s	Others
./ra/fsp/src/bsp/mcu/all/bsp_rom_registers.o	0x0300A264	0x0300A267	4	.option_setting_s	Others
./ra/fsp/src/bsp/mcu/all/bsp_rom_registers.o	0x0300A268	0x0300A26B	4	.option_setting_s	Others
./ra/fsp/src/bsp/mcu/all/bsp_rom_registers.o	0x0300A26C	0x0300A26F	4	.option_setting_s	Others
./ra/fsp/src/bsp/mcu/all/bsp_rom_registers.o	0x0300A280	0x0300A283	4	.option_setting_s	Others
./ra/fsp/src/bsp/mcu/all/bsp_rom_registers.o	0x0300A290	0x0300A293	4	.option_setting_s	Others
./ra/fsp/src/bsp/mcu/all/bsp_rom_registers.o	0x0300A2C0	0x0300A2C3	4	.option_setting_s	Others
./ra/fsp/src/bsp/mcu/all/bsp_rom_registers.o	0x0300A2C4	0x0300A2C7	4	.option_setting_s	Others
./ra/fsp/src/bsp/mcu/all/bsp_rom_registers.o	0x0300A2C8	0x0300A2CB	4	.option_setting_s	Others
./ra/fsp/src/bsp/mcu/all/bsp_rom_registers.o	0x0300A2CC	0x0300A2CF	4	.option_setting_s	Others
./ra/fsp/src/r_glcdc/r_glcdc.o	0x02003318	0x020033E7	208	.text	Program
./ra/fsp/src/r_glcdc/r_glcdc.o	0x020033E8	0x02003417	48	.text	Program
./ra/fsp/src/r_glcdc/r_glcdc.o	0x02003418	0x02003477	96	.text	Program
./ra/fsp/src/r_glcdc/r_glcdc.o	0x02003478	0x020034D7	96	.text	Program
./ra/fsp/src/r_glcdc/r_glcdc.o	0x020034D8	0x02003517	64	.text	Program
./ra/fsp/src/r_glcdc/r_glcdc.o	0x02003518	0x02003593	124	.text	Program

Figure 28. Using the Memory Usage View to Identify Option Setting Address Locations Programmed

2.5.1 SRAM

RA8 contains on-chip 1 MB SRAM (128 KB of TCM RAM, 896 KB of user SRAM). On-chip high-speed SRAM supports error checking with either parity bit or Error Correction Code (ECC). SRAM0 is ECC. SRAM1 is parity check. It is accessible to bus masters such as CPUMAXI, DMAC/DTC bus, EDMAC bus, LCDC0/1 bus GDSS bus, CEU bus.
RA8 MCU supports power saving by reducing the power consumption by setting Module Stop Control Register A (MSTPCRA) to stop the supply of clock signal to SRAM. Data is retained in the Low Power mode (Software Standby mode). Data is not retained in the Deep Software Standby mode.

SRAM Specification



Parameter	SRAM0	SRAM1
SRAM capacity	384 KB	512 KB
SRAM address	0x2200_0000 to 0x2205_FFFF (Secure alias), 0x3200_0000 to 0x3205_FFFF (Non-secure alias)	0x2206_0000 to 0x220D_FFFF (Secure alias), 0x3206_0000 to 0x320D_FFFF (Non-secure alias)
Access	Wait states are inserted into the read cycle by default. If the ICLK frequency is higher than 120 MHz, a wait state is required. If the ICLK frequency is 120 MHz or less, a wait state is not required.	
Data retention function	Not available in Deep Software Standby mode	
Module-stop function	Module-stop state can be set to reduce power consumption	
Error checking	SEC-DED (Single-Error Correction and Double-Error Detection Code)	Even-parity (Data: 8 bits, Parity: 1 bit)
Security	TrustZone Filter is integrated for memory access and SFR access. Access to the memory space is controlled by setting the memory Security Attribution (SA). And access to I/O space (SFR) space is controlled by setting the register SA. See section 50.3.6. TrustZone Filter function.	

Figure 29. RA8 Internal SRAM Memory

ECC with SRAM: ECC (Error-Correcting Code) memory is a technology used to detect and correct single-bit errors in computer memory (RAM). ECC memory is typically used in mission-critical applications where data integrity is of utmost importance.

There are some drawbacks of using ECC memory. ECC memory typically has higher latency compared to non-ECC memory. This is because ECC memory requires additional time to check and potentially correct errors in data. ECC memory modules are more expensive than non-ECC memory. This can be a significant drawback, especially for cost-sensitive applications like consumer products, where data integrity is not as critical. ECC memory requires additional hardware and logic to implement error checking and correction. This overhead can result in slightly lower memory bandwidth and increased power consumption, although the impact is generally minimal.

Correction of ECC Errors: Enabling and disabling of ECC error correction can be selected by setting ECCMOD[1:0] bits of the SRAMCR0 register. In the initial state, ECC error correction is disabled. The ECC check type is SEC-DED (Single-Error Correction and Double-Error Detection Code).

When ECC function is enabled, 8-bit check bits are appended to 64-bit data for writing. For reading, 72-bit (data: 64 bits, check bits: 8 bits) data is read out from the SRAM (ECC area) .

When ECC function and error checking are enabled, error correction is done if a 1-bit error occurs and the ERR00 bit in the SRAMESR register is set to 1 if the E1STSEN bit in the SRAMCR0 register is 1. If a 2-bit error occurs, error detection is done and the ERR01 bit in the SRAMESR register is set to 1, though error correction is not performed.

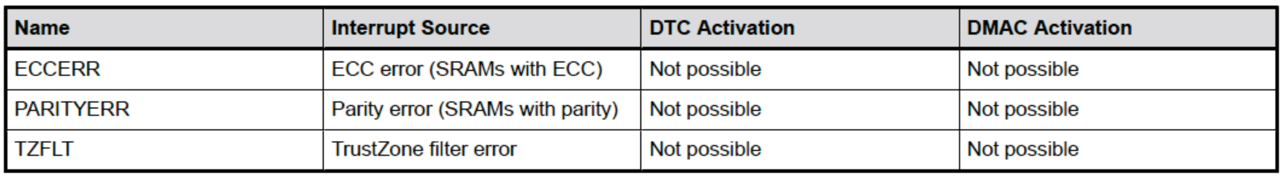
When ECC function is enabled and the error checking is disabled, error correction is done if a 1-bit error occurs but ERR00 bit in the SRAMESR register is not updated although E1STSEN bit in the SRAMCR0 register is 1. If a 2-bit error occurs, this error is detected but the ERR01 bit in the SRAMESR register is not updated, and error correction is not performed.

When ECC function is disabled, neither error correction nor error detection is done when 1-bit or 2-bit errors occur.

Parity Error Calculation Function : The IEC60730 standard requires the checking of SRAM data. When data is written, a parity bit is added to every 8-bit data in the SRAM, and when data is read, the parity is checked. When a parity error occurs, a parity-error notification is generated. This function can also be used to trigger a reset.

The parity-error notification can be specified as a non-maskable interrupt or a reset in the OAD bit of the SRAMCR1 register. When the OAD bit is set to 1, a parity error is output to the reset function. When the OAD bit is set to 0, a parity error is output to the ICU as a non-maskable interrupt.

SRAM Interrupt Source : The SRAM interrupt source includes an ECC error, Parity error and TrustZone® filter error. ECC error and Parity error can choose non-maskable interrupt or reset by SRAMCR0.OAD or SRAMCR1.OAD bit. The SRAM interrupt occurs when one error status in the SRAMESR register is set to 1, the SRAM interrupt continues to occur until the SRAMESR register flag is cleared. When common memory errors occur (NMISR.CMST=1 or RSTSR1.CMSR=1), read SRAMESR and check SRAM interrupt source.



Name	Interrupt Source	DTC Activation	DMAC Activation
ECCERR	ECC error (SRAMs with ECC)	Not possible	Not possible
PARITYERR	Parity error (SRAMs with parity)	Not possible	Not possible
TZFLT	TrustZone filter error	Not possible	Not possible

Figure 30. RA8 Internal SRAM Memory ECC and Parity error

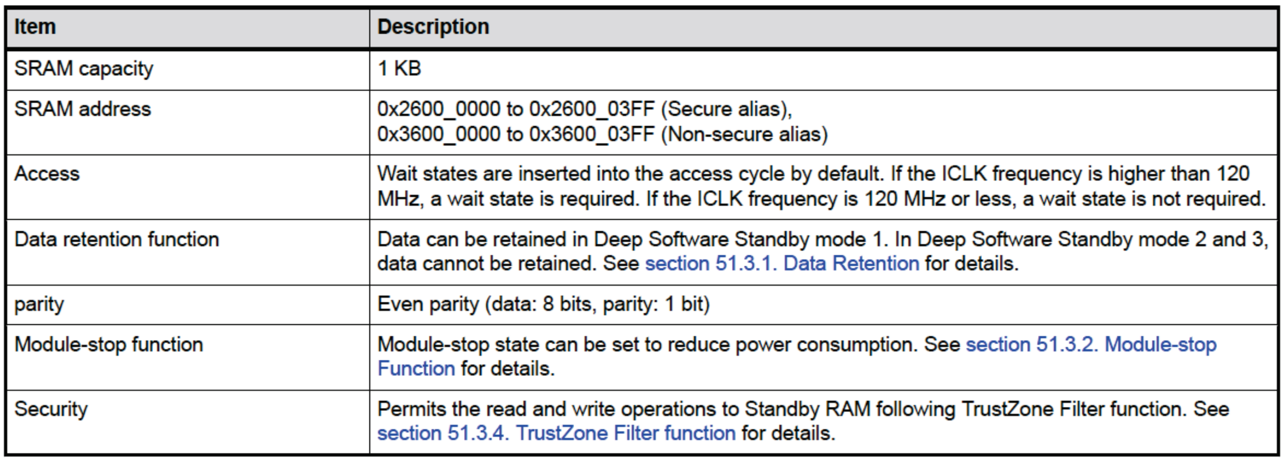
SRAM Wait State: When ICLK frequencies are higher than 120 MHz, do not set 0x00 in wait enable bit for the SRAMWTSC register, in order to insert a wait cycle. When the wait is not inserted, the operation is not guaranteed. Depending on the operating frequency of ICLK, the WAIT setting for SRAM access has the following conditions.

- 240 MHz ≥ ICLK > 120 MHz = 1 wait
- 120 MHz ≥ ICLK = No-wait

2.5.2 SSRAM

RA8 contains 1 KB of on-chip high speed standby SRAM, which supports Error checking with a parity bit. It is accessible to BUS masters such as CPUMAXI, DMAC/DTC bus, EDMAC bus.
RA8 MCU supports the power saving option to reduce the power consumption, by setting module stop control register A (MSTPCRA) to stop supply of the clock signal to SRAM. Data is retained in the Low power mode (Software standby mode) and in the Deep software Standby mode 1.

Standby SRAM Specification



Item	Description
SRAM capacity	1 KB
SRAM address	0x2600_0000 to 0x2600_03FF (Secure alias), 0x3600_0000 to 0x3600_03FF (Non-secure alias)
Access	Wait states are inserted into the access cycle by default. If the ICLK frequency is higher than 120 MHz, a wait state is required. If the ICLK frequency is 120 MHz or less, a wait state is not required.
Data retention function	Data can be retained in Deep Software Standby mode 1. In Deep Software Standby mode 2 and 3, data cannot be retained. See section 51.3.1. Data Retention for details.
parity	Even parity (data: 8 bits, parity: 1 bit)
Module-stop function	Module-stop state can be set to reduce power consumption. See section 51.3.2. Module-stop Function for details.
Security	Permits the read and write operations to Standby RAM following TrustZone Filter function. See section 51.3.4. TrustZone Filter function for details.

Figure 31. RA8 Internal Standby SRAM Memory

Module-stop Function: Power consumption can be reduced by setting the module stop control register A (MSTPCRA) to stop supply of the clock signal to SRAM. If the Standby SRAM bit in MSTPCRA is set to 1, supply of the clock signal to the Standby SRAM is stopped. The Standby SRAM is thus placed in the module-stop state by stopping supply of the clock signals.

The Standby SRAM is not accessible if it is in the module-stop state. A transition to the module-stop state should not be made while access to the standby SRAM is in progress.

Access to the Standby SRAM in the module-stop state is prohibited. If access is attempted, correct operation is not guaranteed.

Parity Calculation Function: The IEC60730 standard requires the checking of STBRAMCR data. When data is written, a parity bit is added to every 8-bit data in the Standby SRAM which has 32-bit data width, and when data is read, the parity is checked. When a parity error occurs, a parity error notification is generated. This function can also be used to trigger a reset.

The parity-error notification can be specified as a non-maskable interrupt or a reset in the OAD bit of the STBRAMCR register. When the OAD bit is set to 1, a parity error is output to the reset function. When the OAD bit is set to 0, a parity error is output to the ICU as a non-maskable interrupt.

SSRAM Interrupt Source: The Standby SRAM interrupt source includes a Parity error and TrustZone® filter error. Parity error can choose non-maskable interrupt or reset by STBRAMCR.OAD bit.

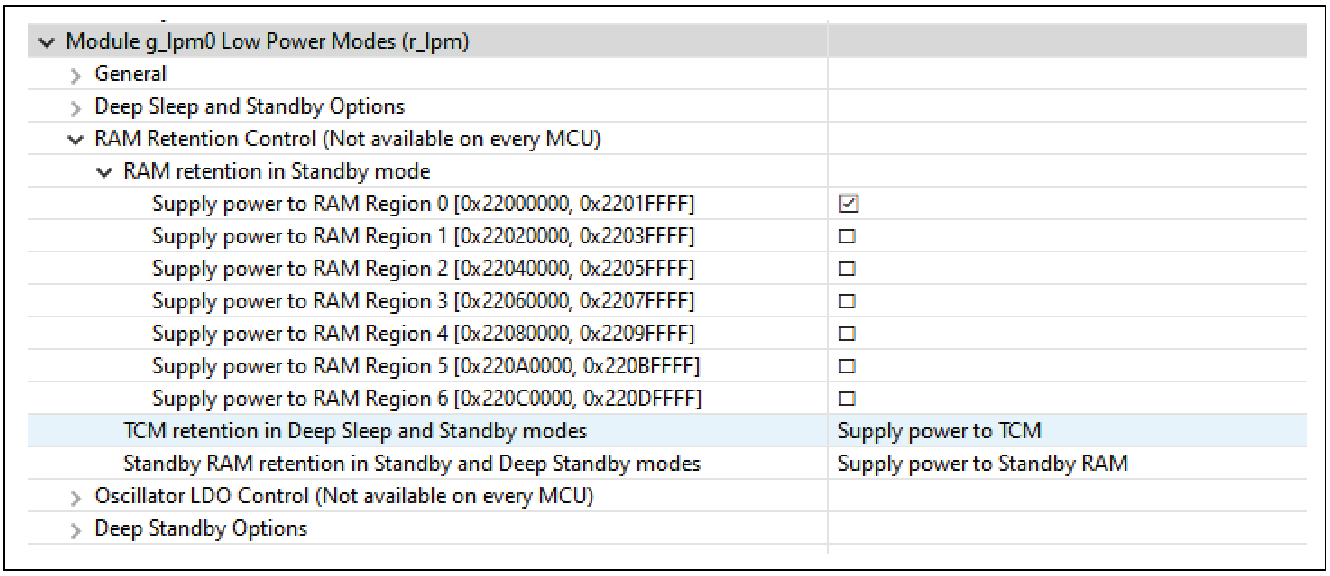
The Standby SRAM interrupt occurs when the SRAMESR.ERRS is set to 1. The Standby SRAM interrupt continues to occur until the SRAMESR.ERRS is cleared.

When Common memory errors occur (NMISR.CMST=1 or RSTSR1.CMSR=1), please read SRAMESR and check SRAM interrupt source. When the access from debugger, no error flag is set, reset and non-maskable interrupt are maskable.

SSRAM Wait State : When ICLK frequencies are higher than 120 MHz, do not set 0x00 in wait enable bit for the SRAMWTSC register in order to insert a wait cycle. When the wait is not inserted, the operation is not guaranteed. Depending on the operating frequency of ICLK, the WAIT setting for SSRAM access has the following conditions.

- 240 MHz ≥ ICLK > 120 MHz = 1 wait
- 120 MHz ≥ ICLK = No-wait

The LPM (Low Power Mode) driver in Renesas FSP provides an option to cut or keep power to different areas in Standby SRAM as shown in the following figure. The LPM driver's APIs still needs to be invoked to apply the selected settings to the MCU registers.



⌄ Module g_lpm0 Low Power Modes (r_lpm)	
› General	
› Deep Sleep and Standby Options	
⌄ RAM Retention Control (Not available on every MCU)	
⌄ RAM retention in Standby mode	
Supply power to RAM Region 0 [0x22000000, 0x2201FFFF]	☑
Supply power to RAM Region 1 [0x22020000, 0x2203FFFF]	☐
Supply power to RAM Region 2 [0x22040000, 0x2205FFFF]	☐
Supply power to RAM Region 3 [0x22060000, 0x2207FFFF]	☐
Supply power to RAM Region 4 [0x22080000, 0x2209FFFF]	☐
Supply power to RAM Region 5 [0x220A0000, 0x220BFFFF]	☐
Supply power to RAM Region 6 [0x220C0000, 0x220DFFFF]	☐
TCM retention in Deep Sleep and Standby modes	Supply power to TCM
Standby RAM retention in Standby and Deep Standby modes	Supply power to Standby RAM
› Oscillator LDO Control (Not available on every MCU)	
› Deep Standby Options	

Figure 32. Enable/Disable Power Supply to Standby SRAM Using Renesas FSP Configurator

RA8 microcontrollers have different power modes, each of which is a compromise between power consumption and wakeup time. Deep Software Standby mode features the lowest power consumption. In this mode, the CPU and peripherals are stopped, clocks are switched off, SRAM and register content are lost. The only peripherals that keep running are the RTC with its 32.768 KHz oscillator, TCM and the Standby SRAM. The microcontroller exits standby mode after a physical reset or an interrupt from the RTC clock. In either case, the CPU has lost all of its states, all the variables are gone and code execution restarts from the beginning.

There are two areas of memory that do not lose data during standby mode: FLASH, TCM (with TCM Power Enabled) and standby SRAM. Standby SRAM is the simplest to use, it consists of 1 Kbytes of memory that remains powered during standby. You just write to it like normal RAM. The only problem it has is that if power is lost (while changing the batteries for example) the contents are lost.

2.5.3 Tightly coupled Memory for Instruction (ITCM)

Tightly Coupled Memory for instruction (ITCM) is a specialized memory that is closely integrated with the RA8 MCU's core processor. ITCM is designed to provide fast and deterministic access to critical program instructions. It is physically located very close to the CPU core, this proximity reduces memory access latency, allowing instructions to be fetched and executed quickly.

ITCM is used to store program instructions that need to be fetched and executed with low latency and minimal contention for access. It is often employed to store critical parts of the program, such as interrupt service routines (ISRs), time-critical control loops, or frequently used functions.

In RA8 MCU, ITCM is 64 KB with ECC (8 KB × 8 block). ECC can be enabled（determined by OFS1.INITECCEN） ITCMCR.EN = 1. It is enabled by default. ITCM is separate from the main Flash memory. Separation helps avoid contention for access to Flash memory, which may be slower and have longer access times.

ITCM is placed at an address of (0x0000_0000 - 0x0000_FFFF) for secure access- and (0x1000_0000-0x1000_FFFF) non-secure access.

For MCU based RTOS systems, the micro kernel can be placed at the ITCM for improving the speed of the system. Motor control algorithms can be placed at the ITCM for improved(i.e. deterministic) execution.

ITCM instructions do not pass through the Cache and do not impose any restriction on wait states. This allows for direct and deterministic access to instructions from this tightly coupled memory, improving execution speed for critical or time-sensitive code. Accessing instructions from ITCM typically incurs fewer or zero wait states compared to accessing instructions from slower memory regions like Flash memory. This reduction in wait states enhances real-time performance and responsiveness of the MCU.

FSP Supplied ITCM initialization happens in `ra\fsp\src\bsp\cmsis\Device\RENESAS\Source\system.c`, inside the **`SystemInit().`**

```
/* Initialize ITCM RAM from ROM image. */
#if BSP_FEATURE_BSP_HAS_ITCM
   bsp_init_itcm();
#endif
```

The linker script has a section .itcm_data for the ITCM, users can place the data in the ITCM using the sample code in your application

```
#define ITCM_CODE       __attribute__((section(".itcm_data"), long_call))

ITCM_CODE void itcmfn() {

      // Place your code logic here
}
```

Users can refer to the *High Performance with RA8 MCU using CM85 core with Helium™* application note and application project which showcases the performance improvement using the ITCM.

2.5.4 Tightly Coupled Memory for Data (DTCM)

Tightly Coupled Memory for Data (DTCM) is another specialized memory on RA8 MCUs, closely integrated with the MCU's core processor, similar to Instruction Tightly Coupled Memory (ITCM). DTCM, however, is primarily used for storing critical data that requires fast and deterministic access.

In RA8 64KB of DTCM provides fast and deterministic access to critical data variables or buffers used by the program. It is used for storing data that requires low-latency access, real-time processing, or data that needs to be accessed frequently.

DTCM, like ITCM, is located physically close to the CPU core, separate from the main system RAM, reducing memory access latency and ensuring that data can be read or written quickly. Access to DTCM is deterministic, ensuring consistent and predictable memory access times. This is crucial for real-time applications where timing is critical.

The 64KB limited capacity of DTCM means that developers must carefully select which data variables or buffers are placed in it. Not all program data can be accommodated in DTCM.

DTCM is commonly used in real-time applications, such as motor control, digital signal processing, and control systems. Critical sensor data, intermediate computation results, or time-sensitive communication buffers can benefit from DTCM storage.

In RA8 MCU DTCM is 64 KB with ECC (8 KB × 8 block). ECC is be enabled（determined by OFS1.INITECCEN）DTCMCR.EN = 1. DTCM is separate from the main RAM memory. Separation helps avoid contention for access to RAM memory, which may be slower and have longer access times.

DTCM is placed at an address of (0x2000_0000 - 0x2000_FFFF) for secure access and (0x3000_0000-0x3000_FFFF) non-secure access. DTCM is connected to the CPUSAHBI and DMAC/DTC bus.

In Software Standby mode the Data of TCM data is retained. Whereas in the Deep software standby mode it is not retained.

DTCM data does not pass through the Cache and does not impose any restriction on wait states. This allows for direct and deterministic access to data from this tightly coupled memory, improving faster access to critical or time-sensitive data. Accessing Data from DTCM typically incurs fewer or zero wait states compared to accessing data from external memories. This reduction in wait states enhances real-time performance by fast and deterministic access.

FSP Supplied ITCM initialization happens in `ra\fsp\src\bsp\cmsis\Device\RENESAS\Source\system.c`, inside the **`SystemInit().`**

```
/* Initialize DTCM RAM from ROM image and zero initialize DTCM RAM BSS section..*/
#if BSP_FEATURE_BSP_HAS_DTCM
   bsp_init_dtcm();
#endif
```

The linker script has a section .dtcm_data for the DTCM, users can place the data in the DTCM RAM using the sample code in your application.

```
__attribute__((section(".dtcm_data"))) uint32_t my_array[64];
```

Users can refer to the *High Performance with RA8 MCU using CM85 core with Helium*™ application note and application project which showcases the performance improvement using the DTCM.

2.5.5 Cache

Cache is a high-speed, volatile memory that stores frequently accessed data to reduce the time it takes to access that data from the slower main memory or storage. The primary purpose of cache is to improve the overall performance of the system by providing faster access to frequently used data.

Usage of Caches: MCU Caches (Data and Instruction Caches) are used to store frequently accessed data and program instructions, respectively. These caches help reduce the time it takes for the CPU to fetch data and instructions from main memory, improving the overall execution speed of programs.

Drawbacks of Caches: Caches introduce the challenge of maintaining data coherence. When data is updated in one place (for example, main memory), the cached copies in different locations (for example, CPU caches) must be updated to reflect the changes. Failure to maintain coherence can lead to data inconsistencies.

Caches can become polluted with data that is rarely or never used, displacing more relevant data. This can reduce cache efficiency. Cache thrashing occurs when cache lines are frequently replaced, leading to a high cache miss rate. This can happen when programs access data in a non-local pattern, making it challenging for the cache to predict what data to store.

Caches add complexity to the system, both in terms of hardware and software. Managing cache hierarchies and cache coherence requires careful design and can increase the risk of bugs.

In summary, caches are valuable tools for improving system performance, but they come with trade-offs and challenges that need to be carefully managed. Proper cache management and design are essential to maximize their benefits while minimizing potential drawbacks.

RA8 MCU provides 3 types of caches - Flash Cache, Data Cache, and Instruction Cache to improve the processing speed and reduce latency.

2.5.5.1 Flash Cache

The Flash Cache (FCACHE) speeds up read access from bus master to the flash memory. The FCACHE includes:

- Flash Cache1 (FCACHE1), for CPU instruction fetches
- Flash Cache 2 (FCACHE2), for CPU operand access and access from EDMAC
- Flash Prefetch Buffer (FLPF), for the prefetch access in CPU instruction fetches

Table 7 and Table 8 show the overview of the Flash cache and Prefetch buffer.

Table 7. RA8 Flash Cache Overview

	Flash Cache 1	**Flash Cache 2**
Cache Target Region	Secure alias : 0x0200_0000 - 0x021F_7FFF Non-secure alias : 0x1200_0000 - 0x121F_7FFF	Secure alias : 0x0200_0000 - 0x021F_7FFF Non-secure alias : 0x1200_0000 - 0x121F_7FFF
Target Bus Master	CPU instruction Fetch	CPU Operand Access and Access from other than CPU
Capacity	256 Bytes	16 Bytes
Associativity	8WAY set associative. 128 bits/entry (128 bit aligned data), 2 entries/way	Full Associative 128 bits/entry (128 bit aligned data), 1 entry
Access Cycle	Cache Hit : 0 wait Cache Miss : wait number of Flash Wait Cycle Register	Cache Hit : 0 wait Cache Miss : wait number of Flash Wait Cycle Register

Table 8. RA8 Flash Prefetch Buffer Overview

	Prefetch Buffer
Cache Target Region	Secure alias : 0x0200_0000 - 0x021F_7FFF Non-secure alias : 0x1200_0000 - 0x121F_7FFF
Request Address	Next address of previous CPU Instruction
Capacity	32 Bytes
Associativity	128 bits/entry (128 bit aligned data), 2 entries
Access Cycle	Cache Hit : 0 wait Cache Miss : wait number of Flash Wait Cycle Register

2.5.5.2 Data Cache

RA8 MCU provides 16 KB of Data L1 Cache with ECC. L1 cache on the CPU side is used to store frequently accessed data to reduce the time it takes for the CPU to retrieve information from the main memory (RAM). This helps in accelerating program execution and data processing. The Cortex®-M85 processor L1 Data cache has the following features.

- It is a four-way set-associative cache.
- It has a cache line size of 32 bytes.
- It supports the following inner memory attributes and allocation hints for Non-shareable memory:
  - Write-Back and Write-Through Cacheable.
  - Read-Allocate and No Read-Allocate.
  - Write-Allocate and No Write-Allocate.
  - Transient and Non-transient. Clean cache lines that are associated with Transient memory are prioritized for eviction over lines that are associated with Non-transient memory.

Allocation into the L1 data cache depends on inner memory attributes only.

- The outer and inner memory attributes are exported on the Manager AXI (M-AXI) interface to support further system-level caching.

- The shareability attribute forces the region to be treated as non-cacheable, regardless of the inner memory attributes. This enables maintaining coherency at the system-level.

2.5.5.3 Instruction Cache

RA8 MCU provides 16KB of L1 Instruction Cache with ECC. L1 cache on CPU side is used to store frequently accessed instructions to reduce the time it takes for the CPU to retrieve information from the slave memory device used for retrieving instructions(Flash). This helps in accelerating program execution and data processing. The Cortex®-M85 processor L1 Data cache has the following features.

- It is a two-way set-associative cache.
- It has a cache line size of 32 bytes.
- It does not allow writes to be performed, except for allocations.
- It only supports read-allocate for inner cacheable memory. Write-Allocate, Write-Back, Write-Through, and Transient attribute hints are ignored. Allocation into the L1 data cache depends on inner memory attributes only.
- Outer and inner memory attributes are exported on the Manager-AXI (M-AXI) interface to support further system-level caching.
- The shareability attribute is ignored for instruction side accesses.
- The inner cache ability attributes are always respected.

Debug accesses from the Debug AHB (D-AHB) subordinate interface on the processor cannot read information from the instruction cache.

Software or a debugger must use the direct cache access registers to read the contents of RAM arrays. The instruction cache is logically organized into two sets of RAM arrays. The dimensions of these RAM arrays vary with the cache size and the inclusion of Error Correcting Code (ECC) logic.

2.5.5.4 How to Enable /Disable Cache Memory

FSP provides a code in the file `RA8/ra/arm/CMSIS_5/CMSIS/Core/Include/cachel1_armv7.h` for enabling and disabling the Cache. So with FSP supplied code, this can be enable or disabled at the application level. I-Cache is enabled by FSP by default in `ra\fsp\src\bsp\cmsis\Device\RENESAS\Source\system.c`, inside the **`SystemInit().`**

```
/**
  \brief   Enable I-Cache
  \details Turns on I-Cache
  */
__STATIC_FORCEINLINE void SCB_EnableICache (void)

/**
  \brief   Disable I-Cache
  \details Turns off I-Cache
  */
__STATIC_FORCEINLINE void SCB_DisableICache (void)


/**
  \brief   Enable D-Cache
  \details Turns on D-Cache
  */
__STATIC_FORCEINLINE void SCB_EnableDCache (void)


/**
  \brief   Disable D-Cache
  \details Turns off D-Cache
  */
__STATIC_FORCEINLINE void SCB_DisableDCache (void)
```

2.5.5.5 Cache Memory and Performance Impact on the System

Using 16 KB of L1 Instruction Cache and 16 KB of L1 Data Cache memory present in the RA8 MCU. It can significantly impact system performance by reducing memory latency, increasing data access speed, and improving overall system throughput. However, deterministic operation is reduced. In general though, the worst case performance of the application can be analyzed by turning off each cache and benchmarking performance.

These caches can be used for both internal and external memory. Only the Manager-AXI (M-AXI) interface accesses can be cached. TCM and Peripheral AHB (P-AHB) interface transactions or accesses cannot be cached.

Users can refer to the *High Performance with RA8 MCU using CM85 core with Helium™* " application note and application project which showcases the performance improvement using the D-cache.

3. Memory Consideration while Designing a System

MCU based system design with limited memory resources involves designing a system that utilizes both internal and external types of memory to optimize performance and storage capacity. Internal memory is also known as primary memory, while external memory refers to secondary memory outside the MCU such as SDRAM and QSPI/OSPI flash used for larger storage. Selecting the right memory resources directly impacts the performance, cost, and overall functionality of the product.

Some key factors to consider when choosing memory for MCU-based products are memory type, memory size, speed and access time, limited write cycles, power consumption, memory with ECC, electrical characteristics, security features, compatibility, regulatory compliance and so forth.

3.1 Internal Memory Selection in Designing a System using RA8 MCU

Internal memory is typically faster than external memory due to its proximity to the CPU. This results in quicker read and write operations, enhancing system performance. Internal memory is readily available to the MCU upon power-up, eliminating the need for time consuming initialization routines needed for external memory initialization.

Selecting the appropriate internal memory in designing a system, is a crucial decision that can significantly impact on the performance, cost, security, design, and functionality of your embedded system.

Internal memory does not require additional components or PCB traces, reducing the overall board space and component count in the design and thereby reducing the size and cost.

Some MCUs offer memory protection mechanisms that restrict access to internal memory regions, enhancing security and preventing unintended data corruption.

The system developed using the internal memory will have a small memory footprint and the unused external memory bus pins can be used for other uses such as GPIO. Smaller package MCUs can be used if the external memory is not required for the system. Internal memories consume lesser power compared to external memory. So in a system where power saving needs utmost care, it is suggested to utilize low-power modes for battery-powered or energy-efficient applications.

Conversely, internal memory size is finite and often limited, which may not be sufficient for applications with extensive data storage or large firmware. Once chosen, the internal memory size and type are fixed and cannot be easily expanded or modified without changing the MCU. Flash memory, often used for program storage, has a finite number of write/erase cycles. Frequent firmware updates can wear out the Flash memory, requiring careful management to ensure longevity.

Ultimately, the decision to use internal memory depends on the specific requirements of your embedded application, such as data storage needs, performance demands, power constraints, and cost considerations. Many systems use a combination of both internal and external memory to strike a balance between these factors and achieve an optimal solution for their application.

Given the size of up to 2 MB the program flash, 12 KB of Data flash and the 1MB of SRAM, Small to medium sized applications which do not involve image/graphics processing can be easily developed using the internal memories alone.

Table 9. RA8 Internal Memory

Internal Memory Type	Size	Example Usage Description, Features
Code flash memory	2 MB	To store the bootloader code, program code (firmware), read-only data.
Data flash memory	12 KB	For non-volatile storage of data, such as configuration, user logs, event logs, user credentials and so forth. A flash file system can be created for storing the data in form files.
SRAM	896 KB	SRAM memory is used for storing data, variables, and buffers required for runtime operations. It allows the MCU to read, write, and manipulate data during program execution. Part of the internal memory is also used for stack.
ITCM	64 KB	Tightly coupled Memory for Instruction ITCM which is linked directly to the CPU core. ITCMs support critical routines, such as interrupt handling routines or real-time tasks, ensuring the highest level of responsiveness with no cache latency.
DTCM	64 KB	Tightly coupled Memory for Instruction DTCM which is linked directly to the CPU core. For faster, predictable and deterministic data access. Critical buffers can be placed in the DTCM. Systems with rapid context switching can leverage DTCM for storing and managing the context switching data efficiently.
Standby SRAM	1 KB	On-chip Standby SRAM that can retain data in Deep Software Standby mode. Very useful in power saving applications such as battery powered systems.
Instruction Cache	16 KB	To improve the execution speed and efficiency of the MCU by storing frequently accessed program instructions in a small, high-speed memory close to the CPU core.
Data Cache	16 KB	Faster Data Access Reduce latency. Support write-back or write-through policies, allowing the MCU to efficiently manage data writes to external memory while maintaining data coherency.
Flash Cache	304 Bytes	The FCACHE (Flash Cache) speeds up read access from the bus master to the flash memory. Flash Cache 1 (FCACHE1) of size 256 Bytes, used for CPU instruction fetch. Flash Cache 2 (FCACHE2) of size 16 Bytes, used for CPU Operand Access and Access from other than CPU(EDMAC). Prefetch Buffer (FLPF) of size 32 Bytes, used for the prefetch access in CPU instruction fetches.

3.2 External Memory Selection in Designing a System using RA8 MCU

Depending on the application, internal memory or a combination of internal and external memory will be required for an efficient system. For systems which do not involve large sets of data or code, it can be managed with the internal memory alone. For applications which involve large code size and data to store or process, the internal code flash and the SRAM may not be sufficient.

For instance, when developing image processing or graphical applications using the graphical accelerator modules, the limited internal SRAM will only fit the frame buffer size for the smaller resolution LCD. In the case of the higher resolution LCD, external SDRAM is the alternate option. For higher resolution LCDs such as XGA (1024 x 768) or greater, the frame buffer can be placed in the SDRAM, helping achieve cost and performance targets.

For larger Graphical applications, the image binaries data can also be stored in the QSPI/OSPI flash instead of the Code flash of the MCU. This will provide additional storage when the image data are large. For improved security, the data on the OSPI or QSPI can be stored using encryption and decryption on the fly (DOTF) can be used while reading the data from the OSPI/QSPI.

While choosing the External memory such as SDRAM, OSPI/QSPI flash, the latency needs to be accounted as well. The performance can also be improved using the DMA bus, I-Cache and D-Cache usage.

Table 10. RA8 External Memory Selection Options

Internal Memory Type	Size	Usage Description, Features
OSPI flash memory interface*	256 MB *2	EOBI OSPI memory interface can support up to 2 slots of 256 MB devices which can be used for OSPI flash
OSPI RAM memory interface*	256 MB *2	EOBI OSPI memory interface can support up to 2 slots of 256 MB devices which can be used for OSPI RAM
QSPI flash memory interface*	256 MB *2	EOBI OSPI memory interface can support up to 2 slots of 256MB devices which can be used for QSPI flash
SDRAM memory	128 MB	
CS area	8 * 16 MB	8, 16 MB slots of address spaces are available for memory and memory mapped peripherals as well. The address space for the CS0 – CS7 ranges from (0x60000000 – 0x67FFFFFF).

Note: *There are only 2 slots available here and the memory can be either OSPI flash or OSPI RAM or QSPI flash when physically mounted.

4. Low Power Mode and RA8 Memory

Many of the embedded systems that MCUs control, such as implantable medical devices, networked sensors, and smart meters need to operate unattended for several years without the need for battery replacement.

In many applications such as thermostats, keyless entry, security systems, and so forth, the processor spends most of its time in a idle mode (staying idle such as forever loop, wakes up on event).

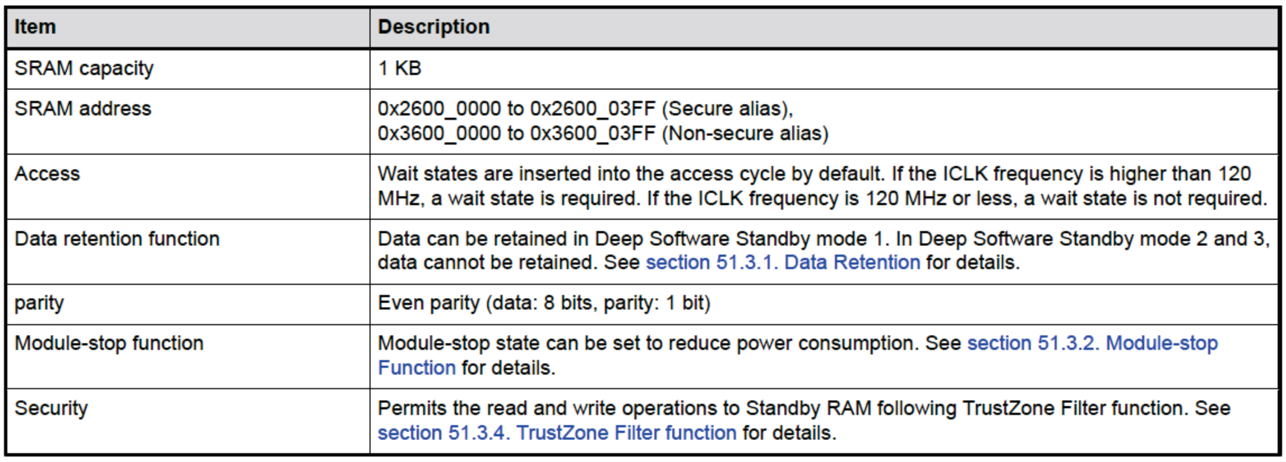
RA8 MCUs provide different types of lower power modes for saving power. In a Software Standby mode MCU core is halted, peripherals are disabled, and clock sources are turned off. However, the MCU stays powered on, which means that state information (consisting of the MCU registers and the contents of on-chip SRAM) is preserved during sleep. Although waking up from sleep is very fast, it is (as expected) not the lowest-power sleep mode possible.

Another type of sleep mode is Deep Software Standby mode, in which the entire MCU, including the on-chip SRAM, is powered down. While this results in the lowest power consumption possible during sleep, it does not preserve the SRAM state.

Therefore, the contents of the SRAM need to be saved (that is, checkpointed) to nonvolatile storage such as the on-chip flash of the MCU before entering this mode. When the MCU wakes up next, the saved state is restored from the flash to the SRAM and the MCU resumes execution. Unfortunately, due to the high write/erase time and power consumption of flash, the energy overhead is more for saving and restoring state.

For this reason MCUs including RA8 provide a specialized SRAM (Standby SRAM - SSRAM) area for LPM mode, where the data is retained during the Deep Software Standby mode and the MCU can wake up and use this data and context of the running code stored in the standby SRAM to start running from where it has left before going into the Deep Sleep mode. If data required to resume operation from low power mode cannot be fit into SSRAM, then user should consider storing the data in SDRAM with self-refresh or part of the data flash.

Standby SRAM Specification - RA8 MCU



Item	Description
SRAM capacity	1 KB
SRAM address	0x2600_0000 to 0x2600_03FF (Secure alias), 0x3600_0000 to 0x3600_03FF (Non-secure alias)
Access	Wait states are inserted into the access cycle by default. If the ICLK frequency is higher than 120 MHz, a wait state is required. If the ICLK frequency is 120 MHz or less, a wait state is not required.
Data retention function	Data can be retained in Deep Software Standby mode 1. In Deep Software Standby mode 2 and 3, data cannot be retained. See section 51.3.1. Data Retention for details.
parity	Even parity (data: 8 bits, parity: 1 bit)
Module-stop function	Module-stop state can be set to reduce power consumption. See section 51.3.2. Module-stop Function for details.
Security	Permits the read and write operations to Standby RAM following TrustZone Filter function. See section 51.3.4. TrustZone Filter function for details.

Figure 33. RA8 Internal Standby SRAM Memory Specification

Apart from the SSRAM, RA8 also provides LPM mode that is supported on the other memory modules as well. RA8 Memory modules and their operating states for Software Standby Mode and Deep software Standby mode are listed below.

4.1 Operating State of Memory Modules in Low Power Mode

Table 11. State of RA8 Memory Modules in Low Power Modes

Module	Software Standby Mode (SSTBY)	Deep Software Standby Mode (DSTBY)		
	SSTBY	DSTBY1	DSTBY2	DSTBY3
External Bus (EBCLK)	Stop (Retained)	Stop (Retained)		
CPU	Stop (Retained)	Stop (Undefined)		
TCM (SRAM)	Stop (Retained)*15	Stop (Undefined)		
User SRAM	Stop (Retained)*13	Stop (Undefined)		
Standby SRAM	Stop (Retained)*14	Stop (Retained)*14	Stop (Undefined)	
Backup register	Stop (Retained)	Stop (Retained)		
Flash memory	Stop (Retained)	Stop (Retained)		
Memory Protection Unit (MPU)	Stop (Retained)	Stop (Undefined)		
DMA Controller (DMAC)	Stop (Retained)	Stop (Undefined)		
Data Transfer Controller (DTC)	Stop (Retained)	Stop (Undefined)		

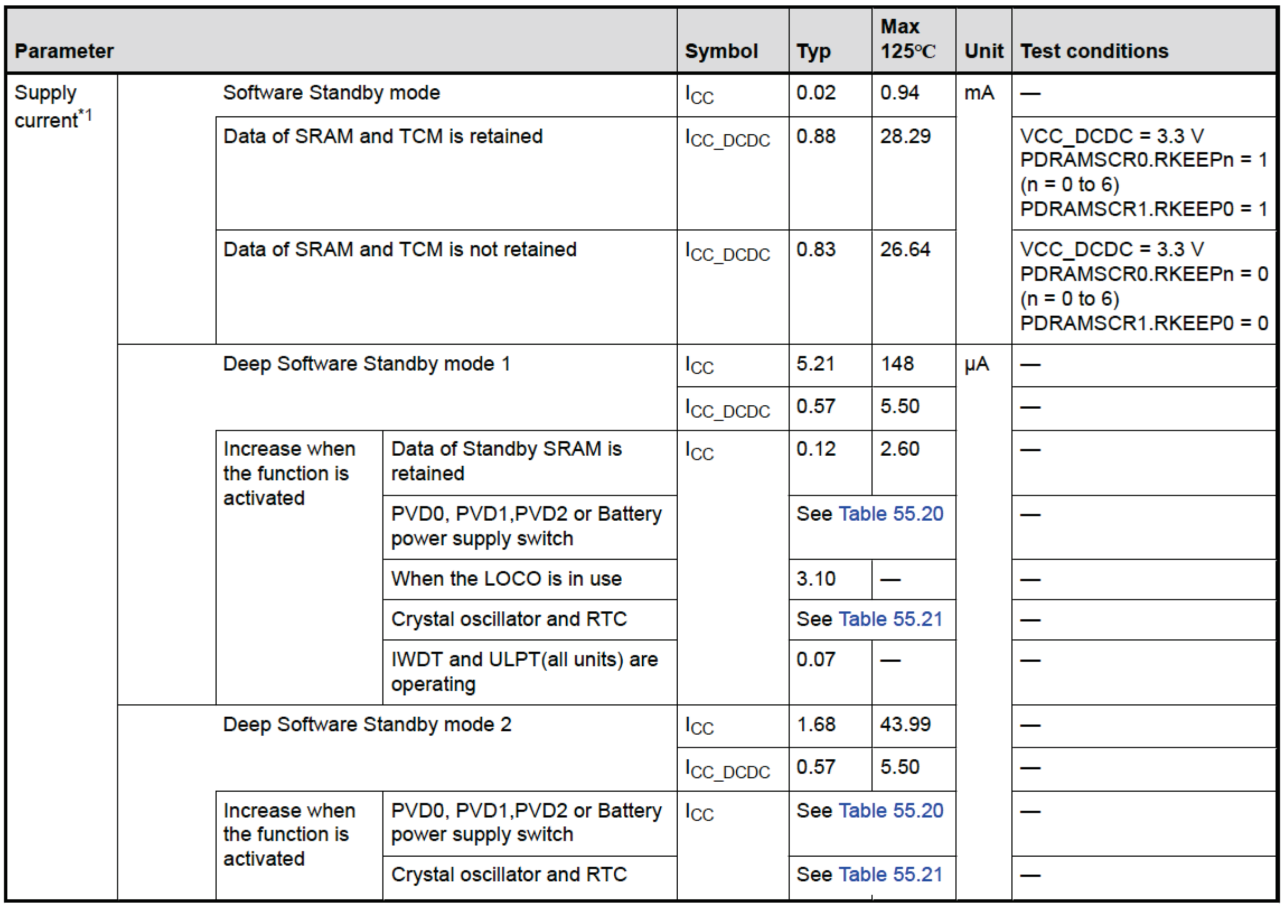
Note 13. If PDRAMSCR0.RKEEPn bit is set to 0, the contents of the target User SRAM is not retained.
Note 14. If DPSBYCR.SRKEEP bit is set to 0, the contents of Standby RAM is not retained.
Note 15. If PDRAMSCR1.RKEEPn bit is set to 0, the contents of the target TCM is not retained.

4.2 Data Retention by using SSRAM

The power supply to the Standby SRAM in Deep Software Standby mode is enabled by the DPSBYCR.SRKEEP bit. If the DPSBYCR.SRKEEP bit is set to 1b, data in the Standby SRAM is retained in Deep Software Standby mode 1. For more details on the Standby SRAM and Data retention, the Users can refer to the "Standby SRAM" section and "Low Power Modes" support in the RA8 UM.
The working example project demonstrating the Standby SRAM and the date retention is shown in the LPM modes for the RA8 MCU example projects.

Standby Current (DC-DC Mode)

The standby current for different LPM modes are given below. For more details on the power consumption for different modes, refer to the electrical characteristics section of the UM.



Parameter				Symbol	Typ	Max 125°C	Unit	Test conditions
Supply current^{*1}	Software Standby mode			I_{CC}	0.02	0.94	mA	—
		Data of SRAM and TCM is retained		I_{CC_DCDC}	0.88	28.29		VCC_DCDC = 3.3 V PDRAMSCR0.RKEEPn = 1 (n = 0 to 6) PDRAMSCR1.RKEEP0 = 1
		Data of SRAM and TCM is not retained		I_{CC_DCDC}	0.83	26.64		VCC_DCDC = 3.3 V PDRAMSCR0.RKEEPn = 0 (n = 0 to 6) PDRAMSCR1.RKEEP0 = 0
	Deep Software Standby mode 1			I_{CC}	5.21	148	μA	—
				I_{CC_DCDC}	0.57	5.50		—
		Increase when the function is activated	Data of Standby SRAM is retained	I_{CC}	0.12	2.60		—
			PVD0, PVD1,PVD2 or Battery power supply switch		See Table 55.20			—
			When the LOCO is in use		3.10	—		—
			Crystal oscillator and RTC		See Table 55.21			—
			IWDT and ULPT(all units) are operating		0.07	—		—
	Deep Software Standby mode 2			I_{CC}	1.68	43.99		—
				I_{CC_DCDC}	0.57	5.50		—
		Increase when the function is activated	PVD0, PVD1,PVD2 or Battery power supply switch	I_{CC}	See Table 55.20			—
			Crystal oscillator and RTC		See Table 55.21			—

Figure 34. RA8 Standby Current for Memory with LPM Modes

5. Configuring the RA8 Memory using FSP Configurator

FSP provides a configurator support for configuring the external memory such as OSPI and SDRAM. Similarly, the BSP tab also provides the configuration related to SRAM accessibility settings, standby SRAM protection, option setting memory registers and first stage bootloader specific settings.

5.1 Modules Available in FSP to Support the RA8 Memory

Users can configure OSPI and SDRAM memories using **New Stack** > **Storage** and the related driver selection. The detailed configurations can be made using the properties tab of the driver module.

On the **BSP** tab, SRAM accessibility for SRAM0/1 and standby SRAM protection for both secure and non-secure state can be set. In addition, bus accessibility and flash bank select accessibility can be set to both secure and non-secure.

OFS1 register settings also provides options for enabling and disabling the ECC functions for TCM and cache, flash block protection settings for different blocks.

5.2 Configuration Parameters for the RA8 Memory Selection

OSPI Configuration

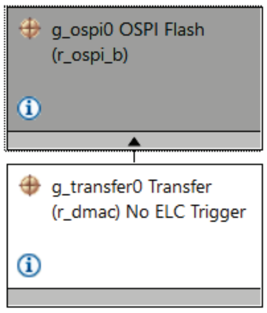
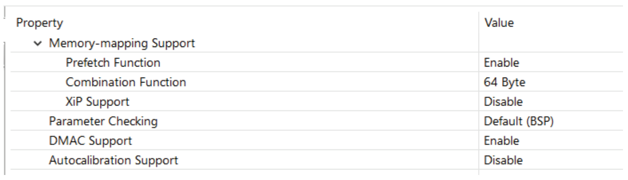

g_ospi0 OSPI Flash (r_ospi_b)
g_transfer0 Transfer (r_dmac) No ELC Trigger

Figure 35. OSPI Memory Configuration using Renesas FSP Configurator

In the case of OSPI memory configuration, you can see the configurator provides many different options and features of the OSPI to be set. Some of the configurable setting for memory mapping are enabling the prefetch function, Enable Data Combination function on memory mapped writes, XiP support, DMAC support, auto calibration support and so forth.



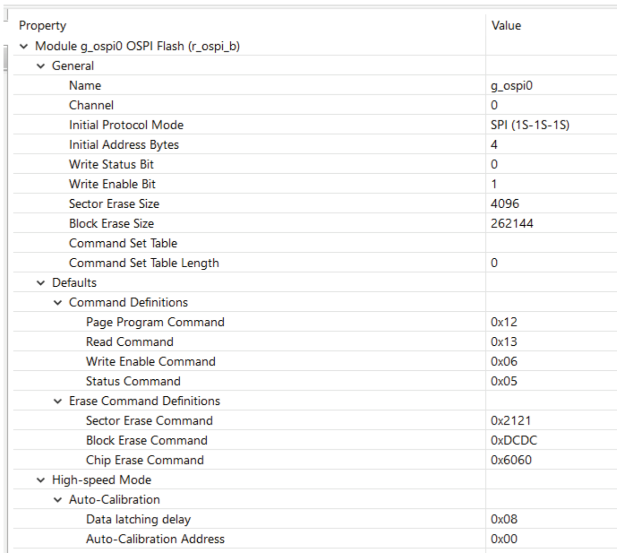
Property	Value
Memory-mapping Support	
Prefetch Function	Enable
Combination Function	64 Byte
XiP Support	Disable
Parameter Checking	Default (BSP)
DMAC Support	Enable
Autocalibration Support	Disable

Figure 36. OSPI Memory Configuration using Renesas FSP Configurator Properties Tab

FSP also provides channel selection, different SPI protocol modes for the OSPI, setting the sector and block size for erase. Command definitions such as for page program, read, write enable and status are also available for configuration.

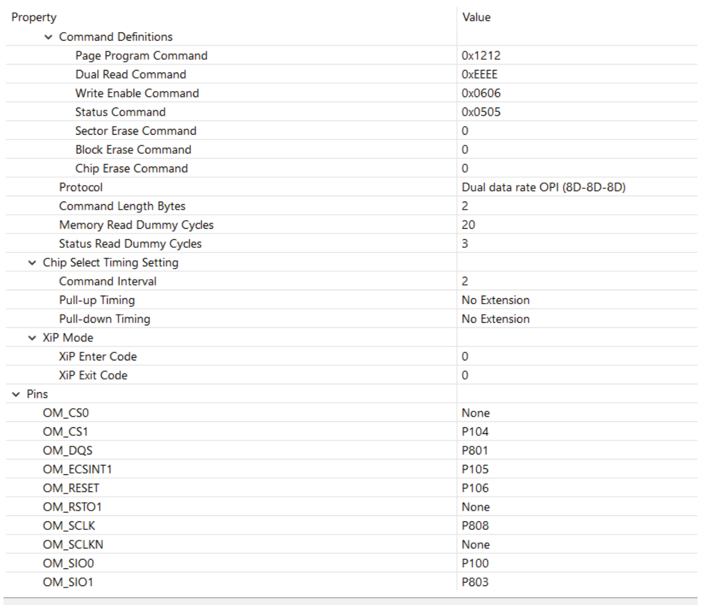
For erasing the OSPI flash, sector erase, block erase, chip erase commands can also be set using the FSP configurator.

Additionally, timing configuration, memory read dummy cycles, data latching delay, XiP mode can also be configured. The sample snapshots of the OSPI configuration using the FSP configurator are shown in Figure 37 and Figure 38.



Property	Value
Module g_ospi0 OSPI Flash (r_ospi_b)	
General	
Name	g_ospi0
Channel	0
Initial Protocol Mode	SPI (1S-1S-1S)
Initial Address Bytes	4
Write Status Bit	0
Write Enable Bit	1
Sector Erase Size	4096
Block Erase Size	262144
Command Set Table	
Command Set Table Length	0
Defaults	
Command Definitions	
Page Program Command	0x12
Read Command	0x13
Write Enable Command	0x06
Status Command	0x05
Erase Command Definitions	
Sector Erase Command	0x2121
Block Erase Command	0xDCDC
Chip Erase Command	0x6060
High-speed Mode	
Auto-Calibration	
Data latching delay	0x08
Auto-Calibration Address	0x00

Figure 37. OSPI Memory Configuration using Renesas FSP Configurator Properties Tab



Property	Value
Command Definitions	
Page Program Command	0x1212
Dual Read Command	0xEEEE
Write Enable Command	0x0606
Status Command	0x0505
Sector Erase Command	0
Block Erase Command	0
Chip Erase Command	0
Protocol	Dual data rate OPI (8D-8D-8D)
Command Length Bytes	2
Memory Read Dummy Cycles	20
Status Read Dummy Cycles	3
Chip Select Timing Setting	
Command Interval	2
Pull-up Timing	No Extension
Pull-down Timing	No Extension
XiP Mode	
XiP Enter Code	0
XiP Exit Code	0
Pins	
OM_CS0	None
OM_CS1	P104
OM_DQS	P801
OM_ECSINT1	P105
OM_RESET	P106
OM_RSTO1	None
OM_SCLK	P808
OM_SCLKN	None
OM_SIO0	P100
OM_SIO1	P803

Figure 38. OSPI Memory Configuration using Renesas FSP Configurator Properties Tab

6. How to Interpret the Electrical Characteristic of RA8 MCU

The electrical characteristics for RA8 MCU and its peripherals are crucial considerations when designing embedded systems. These characteristics include voltage levels, current requirements, timing specifications, and operating temperature and so forth. These electrical characteristics can impact system performance, reliability, and compatibility. The HW UM contains a detailed table for the electrical characteristics. In this section, some common electrical characteristics used for MCU and its peripherals are explained.

Voltage Levels: Supply Voltage (**Vcc/Vdd**): The voltage level required to power the RA8 and its peripherals. It is usually specified in volts (V) and should be within the MCUs acceptable range.

I/O Voltage Levels: VIH, VIL. VOH, VIL are the voltage levels at which the MCU's input and output pins operate. RA8 MCU supports different I/O voltage levels such as 3.3V or 5V. It is important to ensure compatibility with external devices.

Current Requirements: Supply Current (**Icc/Idd**): The current consumed by the MCU and its peripherals during operation. This includes both static (quiescent) and dynamic (active) current.

I/O Pin Current: The maximum current that can be sourced or sunk by each I/O pin. This specification is crucial for driving external devices or interfacing with other components.

Output Drive Strength: Output Current Drive: The maximum current that an I/O pin can source or sink. Higher drive strength is needed for driving loads with higher impedance.

Timing Specifications:

Setup and Hold Times: Timing requirements for data setup and hold times, particularly relevant for synchronous communication interfaces like SPI/QSPI/OSPI and I2C.

Access Time: The time it takes for the memory to respond to a read or write operation. Access time can affect the MCU's overall performance, especially in real-time applications.

Power Management: Low-Power Modes: Information about different low-power modes and their associated current consumption.

IO Schmitt Trigger Inputs: Specifications for Schmitt trigger inputs, which are commonly used for noisy input signals.

Erase Time: The time it takes to erase a block or sector of Flash memory. Erase times can vary depending on the MCU and the memory technology used.

Operating Temperature Range: The range of temperatures within which the MCU and its peripherals are guaranteed to operate reliably.

6.1 How to Read the Electrical Characteristics Specified in the MCU User Manual

RA8 MCU User Manual has a section for the electrical characteristics comprised of many categories. In this document we have given a brief description of the electrical characteristics and how to read and infer from the data provided in the UM. Following are some of the categories:

- Absolute Maximum Ratings
- DC Characteristics
  - Tj/Ta definition
  - I/O VIH, VIL
  - I/O IOH, IOL
  - I/O VOH, VOL, and other characteristics
  - Operating and Standby Current
  - VCC Rise and Fall Gradient and Ripple Frequency
  - Thermal Characteristics
- AC Characteristics
  - Frequency
  - Clock timing
  - Reset timing
  - Wakeup timing
  - Bus timing
  - OSPI timing and others
- Peripheral characteristics (for peripherals such as USB, MIP ADC, flash memory) and so on.

Absolute Maximum Ratings: Specifications in the electrical characteristics of a MCU that define the maximum limits of various electrical and environmental parameters beyond which the MCU may be damaged or may not operate correctly. For RA8, the absolute maximum ratings are given in the Table 12. Here, you can see that for a typical VCC of 3.3 V for the MCU, the VCC value can go to a maximum of 4.0 V. Applying voltages above 4.0V can damage the device. Similarly, the operating and storage temperature can be in the range of -55 to 125 °C.

When interfacing with external components, it is a good idea to ensure that the specifications of the chosen external components match the specifications of the MCU and vice versa.

Table 12. RA8 Absolute Maximum Ratings

Parameter	**Symbol**	**Value**	**Unit**
Power supply voltage	VCC, VCC2, VCC_DCDC, VCC_USB*2	–0.3 to +4.0	V
External power supply voltage	VCL	–0.3 to +1.6	V
VBATT power supply voltage	VBATT	–0.3 to +4.0	V
Input voltage (except for 5 V-tolerant ports*1)	Vin	–0.3 to VCC + 0.3 or –0.3 to VCC2 + 0.3	V
Reference power supply voltage	VREFH/VREFH0	–0.3 to AVCC0 + 0.3	V
Analog power supply voltage	AVCC0	–0.3 to +4.0	V
Analog input voltage	VAN	–0.3 to AVCC0 + 0.3	V
Operating junction temperature*3 *4	Topj	–40 to +125	°C
Storage temperature	Tstg	–55 to +125	°C

Even though the MCU can take values up to absolute maximum ratings for some peripherals, in order to perform reliably, it is also highly recommended to operate the MCU within the recommended operating voltages levels. Table 13 shows the recommended operating voltage ranges for the ETHERC/IIC/USB/SDRAM.

Table 13. RA8 Recommended Operating Rating

Parameter	Symbol		Min	Typ	Max	Unit
Power supply voltages	VCC, VCC_DCDC	Other than the following	1.68	—	3.60	V
		When ETHERC/IIC Fast-mode + is used	2.70	—	3.60	V
		When USB/SDRAM is used	3.00	—	3.60	V
	VCC2		1.65	—	3.60	V
	VCL	When external VDD is used*2	1.20	—	1.25	V
		When DCDC is used (High-speed mode)	—	1.21	—	V
		When DCDC is used (Low-speed mode or Software Standby mode)	—	1.18	—	V
	VSS, VSS_DCDC		—	0	—	V

DC Characteristics

Tj/Ta Definition The permissible operating junction temperature (also referred to as **Tj** or **Tj(max)**) is an important parameter in the DC characteristics of an MCU. It defines the maximum temperature at which the internal semiconductor components of the MCU can operate safely and reliably. Exceeding this temperature can lead to thermal stress, reduced performance, and potential damage to the MCU.

Table 14. RA8 Tj/Ta Ratings

Parameter	Symbol	Typ	Max	Unit	Test conditions
Permissible operating junction temperature	T_j	—	125	°C	High-speed mode Low-speed mode

Note: Make sure that $T_j = T_a + \theta_{ja} \times$ total power consumption (W), where total power consumption = $(VCC - V_{OH}) \times \Sigma I_{OH} + V_{OL} \times \Sigma I_{OL} + (I_{CC}max + I_{CC\_DCDC}max) \times VCC$.

Note: Minimum Ambient Temperature(Ta) is -40℃

I/O VIH, VIL : VIH (Voltage Input High) and **VIL** (Voltage Input Low) refer to the voltage levels that define the logical high and logical low states for a digital input signal. These voltage levels are crucial for ensuring reliable communication between digital devices.

Table 15. I/O VIH, VIL Ratings

Parameter		VCC/VCC2/ AVCC0	Symbol	Min	Typ	Max	Unit
Peripheral function pins	OSPI (except OM_RSTO1 and OM_ECSINT1)	2.70 V or above	V_{IH}	VCC2 × 0.8	—	—	V
			V_{IL}	—	—	VCC2 × 0.2	
		1.65 V or above	V_{IH}	VCC2 × 0.7	—	VCC2 + 0.3	
			V_{IL}	VSS - 0.3	—	VCC2 × 0.3	

From Table 15 you can see that in the case of OSPI Peripheral for **VIH**, the minimum voltage to recognize it as high is 0.8 times VCC and for **VIL** it is 0.2 times VCC.

I/O IOH, IOL : IOH (Output High Current) and **IOL** (Output Low Current) are electrical characteristics that specify the maximum current a digital output pin can source (**IOH**) or sink (**IOL**) while maintaining proper voltage levels.

Table 16. I/O IOH, IOL Ratings

Parameter			VCC/ VCC2/ AVCC0	Symbol	Min	Typ	Max	Unit
Permissible output current (average value per pin)	Ports P000 to P011, P014, P015, P201	—	—	I_{OH}	—	—	−2.0	mA
				I_{OL}	—	—	2.0	mA
	Other output pins[*5]	Low drive[*1]	—	I_{OH}	—	—	−2.0	mA
				I_{OL}	—	—	2.0	mA
		Middle drive[*2]	—	I_{OH}	—	—	−4.0	mA
				I_{OL}	—	—	4.0	mA
		High drive[*3]	—	I_{OH}	—	—	−16	mA
				I_{OL}	—	—	16.0	mA

From Table 16, you can see for the ports which are configured as output, when drive capacity is configured as low drive, **IOH** is -2.0 mA and **IOL** is 2.0 mA and for ports configured as middle drive, **IOH** is -4.0 mA and **IOL** is 4.0 mA. For ports configured as high drive, **IOH** is -16.0 mA and **IOL** is 16.0 mA.

Caution: To protect the reliability of the MCU, the output current values should not exceed the values mentioned in the I/O IOH, IOL table. The average output current indicates the average value of current measured during 100 µs.

RA8 MCU UM provides details on the permissible output current for various ports and pins. For specific details on the ports and other peripheral pins, refer to the RA8 UM.

I/O VOH, VOL : VOH (Output High Voltage) and **VOL** (Output Low Voltage) are electrical characteristics that specify the voltage levels of a digital output signal when it is in the high (**VOH**) or low (**VOL**) logic state.

In the case of SD peripheral as an example for **VOL**, minimum voltage to recognize it as logic low when VCC is 2.70 V or higher, it is 0.125 times the VCC for (IOL = 3.0mA) and for VOH it is 0.75 times the VCC for (IOL = -2.0mA). In the case of VCC between 1.70 V to 1.95 V, VOH is 1.4 V (with IOL = -2.0 mA) and it is 0.45 V for (IOL = 2.0 mA).

Table 17. I/O VOH, VOL ratings

Parameter		VCC/VCC2/ AVCC0	Symbol	Min	Typ	Max	Unit	Test conditions
Output voltage	IIC	2.70 V or above	V_{OL}	—	—	0.4	V	I_{OL} = 3.0 mA
			V_{OL}	—	—	0.6		I_{OL} = 6.0 mA
		1.68 V or above	V_{OL}	—	—	VCC × 0.2		I_{OL} = 3.0 mA
			V_{OL}	—	—	0.6		I_{OL} = 6.0 mA
	SD	2.70 V or above	V_{OH}	VCC × 0.75	—	—		I_{OH} = −2.0 mA
			V_{OL}	—	—	VCC × 0.125		I_{OL} = 3.0 mA
			V_{OH}	VCC2 × 0.75	—	—		I_{OH} = −2.0 mA
			V_{OL}	—	—	VCC2 × 0.125		I_{OL} = 3.0 mA
		1.70 V to 1.95 V	V_{OH}	1.4	—	—		I_{OH} = −2.0 mA
			V_{OL}	—	—	0.45		I_{OL} = 2.0 mA

Operating and Standby Current

Operating current is the current drawn by the device while it is in normal operation, executing tasks, processing data, and actively communicating with other components or devices. The operating current can vary depending on factors such as clock speed, workload, input/output activity, and other operational parameters.

Standby current, also known as quiescent current or sleep current, is the current consumed by a device or component when it is in a low-power or idle state. In this state, the device is not actively performing its primary functions but is still operational and ready to be awakened or respond to an external event.

Balancing the need for high performance during active operation with minimal power consumption during standby or idle periods is a fundamental aspect of energy-efficient system design. The RA8 UM gives the details of the Operating and Standby current for different modes.

In the following table, you can see the operating current for different clock frequencies with VCC_DCDC greater than 2.5 V and lesser than 2.5 V. Higher the frequency of operation, more the current consumption. From Table 18, you can get the approximate current consumption for your system and design. For calculating this current consumption there are other parameters to take into considerations. For more details and specific current consumption, users are required to refer to the MCU UM.

Table 18. Current of High-speed Mode, Maximum Condition (DCDC Mode)

Parameter			Symbol	Typ	Max		Unit	Test conditions
					105℃	125℃		
Supply current [*1*2]	—		I_{CC}	—	5.97	6.11	mA	
	CPUCLK = 480 MHz	VCC_DCDC ≧ 2.5 V	I_{CC_DCDC}[*5]	—	303	—	mA	VCC_DCDC = 3.3 V CPUCLK = 480 MHz, ICLK = 240 MHz, PCLKA = 120 MHz, PCLKB = 60 MHz, PCLKC = 60 MHz, PCLKD = 120 MHz, PCLKE = 240 MHz, FCLK = 60 MHz, BCLK = 120 MHz
			I_{DD}[*3]	—	624	—		
		VCC_DCDC < 2.5 V	I_{CC_DCDC}[*5]	—	320	—		VCC_DCDC = 1.8 V
			I_{DD}	—	400[*4]	—		
	CPUCLK = 400 MHz	VCC_DCDC ≧ 2.5 V	I_{CC_DCDC}[*5]	—	267	307	mA	VCC_DCDC = 3.3 V CPUCLK = 400 MHz, ICLK = 200 MHz, PCLKA = 100 MHz, PCLKB = 50 MHz, PCLKC = 50 MHz, PCLKD = 100 MHz, PCLKE = 100 MHz, FCLK = 50 MHz, BCLK = 100 MHz
			I_{DD}[*3]	—	550	632		
		VCC_DCDC < 2.5 V	I_{CC_DCDC}[*5]	—	320	320		VCC_DCDC = 1.8 V
			I_{DD}	—	400[*4]	400[*4]		
	CPUCLK = 360 MHz	VCC_DCDC ≧ 2.5 V	I_{CC_DCDC}[*5]	—	257	297	mA	VCC_DCDC = 3.3 V CPUCLK = 360 MHz, ICLK = 120 MHz, PCLKA = 120 MHz, PCLKB = 60 MHz, PCLKC = 60 MHz, PCLKD = 120 MHz, PCLKE = 120 MHz, FCLK = 60 MHz, BCLK = 120 MHz
			I_{DD}[*3]	—	530	612		
		VCC_DCDC < 2.5 V	I_{CC_DCDC}[*5]	—	320	320		VCC_DCDC = 1.8 V
			I_{DD}	—	400[*4]	400[*4]		
	CPUCLK = 240 MHz	VCC_DCDC ≧ 2.5 V	I_{CC_DCDC}[*5]	—	224	264	mA	VCC_DCDC = 3.3 V CPUCLK = 240 MHz, ICLK = 240 MHz, PCLKA = 120 MHz, PCLKB = 60 MHz, PCLKC = 60 MHz, PCLKD = 120 MHz, PCLKE = 120 MHz, FCLK = 60 MHz, BCLK = 120 MHz
			I_{DD}[*3]	—	460	544		
		VCC_DCDC < 2.5 V	I_{CC_DCDC}[*5]	—	320	320		VCC_DCDC = 1.8 V
			I_{DD}	—	400[*4]	400[*4]		

Bus Timing: The timing specifications associated with the communication and data transfer between peripheral devices or components on a bus.

Table 19 shows the bus timings for the various parameters with different voltage level conditions.

Table 19. Bus Timings with Different Conditions

Condition 1: When using the CS area controller (CSC)	
	VCC = VCC_DCDC = VCC_USB = VBATT = 1.68 V to 3.6 V, VCC2 = 1.65 V to 3.6 V
	BCLK = 8 to 120 MHz, EBCLK = 8 to 60 MHz (When VCC = VCC_USB = VBATT = 2.70 to 3.6 V)
	BCLK = 8 to 60 MHz, EBCLK = 8 to 30 MHz (When VCC = VCC_USB = VBATT = 1.68 to 3.6 V)
	Output load conditions: VOH = VCC × 0.5, VOL = VCC × 0.5, C = 30 pF
	EBCLK: High drive output is selected in the port drive capability bit in the PmnPFS register.
	Others: Middle drive output is selected in the port drive capability bit in the PmnPFS register.

Condition 2: When using the SDRAM area controller (SDRAMC)	
	BCLK = SDCLK = 8 to 120 MHz
	VCC = VCC2 = VCC_DCDC = VCC_USB = VBATT = 3.0 to 3.6 V
	Output load conditions: VOH = VCC × 0.5, VOL = VCC × 0.5, C = 15 pF
	SDCLK: High-speed high-drive output is selected in the port drive capability bit in the PmnPFS register.
	Others: High-drive output is selected in the port drive capability bit in the PmnPFS register.

Condition 3: When using the SDRAM area controller (SDRAMC) and CS area controller (CSC) simultaneously	
	BCLK = SDCLK = 8 to 60 MHz
	VCC = VCC2 = VCC_DCDC = VCC_USB = VBATT = 3.0 to 3.6 V
	Output load conditions: VOH = VCC × 0.5, VOL = VCC × 0.5, C = 15 pF
	EBCLK/SDCLK: High drive output is selected in the port drive capability bit in the PmnPFS register.
	Others: Middle drive output is selected in the port drive capability bit in the PmnPFS register.

The typical bus timings and their acceptable ranges are given in Table 20 and Table 21. This timing data is important when choosing the external peripherals such as OSPI flash or SDRAM. While designing external memory interface, it is highly recommended to refer to the electrical characteristics of the memory devices and to check that the timing parameters are in the range specified in the UM for the proper working of the system.

Table 20. Bus Timings

Parameter		Symbol	Min	Max	Unit	Test conditions
Address delay	2.70V or above	t_{AD}	1.0	12.5	ns	Figure 60.28 to Figure 60.34
	1.68V or above		1.0	12.5	ns	
Byte control delay	2.70V or above	t_{BCD}	1.0	12.5	ns	
	1.68V or above		1.0	12.5	ns	
CS delay	2.70V or above	t_{CSD}	1.0	12.5	ns	
	1.68V or above		1.0	12.5	ns	
ALE delay time	2.70V or above	t_{ALED}	1.0	12.5	ns	
	1.68V or above		1.0	12.5	ns	
RD delay	2.70V or above	t_{RSD}	1.0	12.5	ns	
	1.68V or above		1.0	12.5	ns	
Read data setup time	2.70V or above	t_{RDS}	12.5	—	ns	
	1.68V or above		20.5	—	ns	
Read data hold time	2.70V or above	t_{RDH}	0	—	ns	
	1.68V or above		0	—	ns	
WR/WRn delay	2.70V or above	t_{WRD}	1.0	12.5	ns	
	1.68V or above		1.0	12.5	ns	
Write data delay	2.70V or above	t_{WDD}	1.0	12.5	ns	
	1.68V or above		1.0	12.5	ns	
Write data hold time	2.70V or above	t_{WDH}	1.0	—	ns	
	1.68V or above		1.0	—	ns	
WAIT setup time	2.70V or above	t_{WTS}	12.5	—	ns	
	1.68V or above		20.5	—	ns	
WAIT hold time	2.70V or above	t_{WTH}	0	—	ns	
	1.68V or above		0	—	ns	

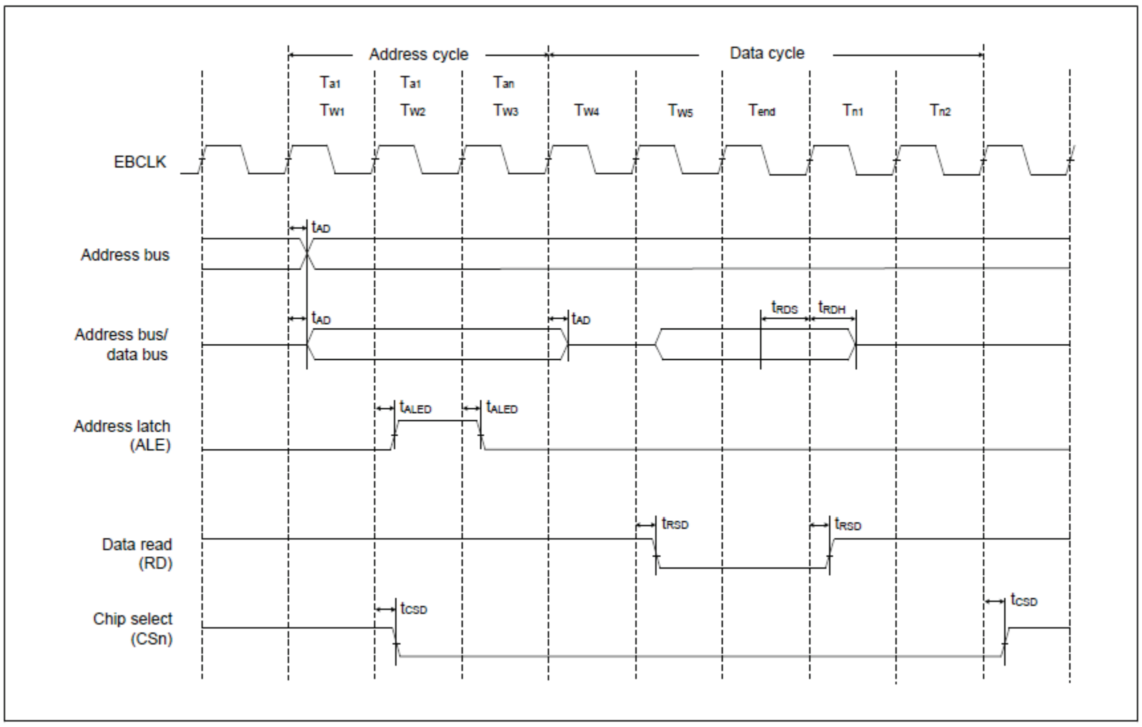


Figure 39. Address/data Multiplexed Bus Read Access Timing

Table 21. Bus Timings

Parameter		Symbol	Min	Max	Unit	Test conditions
Address delay 2 (SDRAM)	Condition 2	t_{AD2}	0.8	6.8	ns	Figure 60.35 to Figure 60.41
	Condition 3		0.8	10.8		
CS delay 2 (SDRAM)	Condition 2	t_{CSD2}	0.8	6.8	ns	
	Condition 3		0.8	10.8		
DQM delay (SDRAM)	Condition 2	t_{DQMD}	0.8	6.8	ns	
	Condition 3		0.8	10.8		
CKE delay (SDRAM)	Condition 2	t_{CKED}	0.8	6.8	ns	
	Condition 3		0.8	10.8		
Read data setup time 2 (SDRAM)	Condition 2	t_{RDS2}	2.9	—	ns	
	Condition 3		6.9	—		
Read data hold time 2 (SDRAM)	Condition 2	t_{RDH2}	1.5	—	ns	
	Condition 3		1.5	—		
Write data delay 2 (SDRAM)	Condition 2	t_{WDD2}	—	6.8	ns	
	Condition 3		—	10.8		
Write data hold time 2 (SDRAM)	Condition 2	t_{WDH2}	0.8	—	ns	
	Condition 3		0.8	—		
WE delay (SDRAM)	Condition 2	t_{WED}	0.8	6.8	ns	
	Condition 3		0.8	10.8		
RAS delay (SDRAM)	Condition 2	t_{RASD}	0.8	6.8	ns	
	Condition 3		0.8	10.8		
CAS delay (SDRAM)	Condition 2	t_{CASD}	0.8	6.8	ns	
	Condition 3		0.8	10.8		

SDRAM single read timing diagram is shown as an example where the minimum and maximum delay for different parameters such as address delay, chip select delay, write data delay, and are listed. This is just a guide in selecting the SDRAM devices. When choosing the SDRAM, the device electrical characteristics will also have min and max range of timing details for different parameters. For proper operation they need to be within the range for proper data read write and other operations.

For instance, for CS delay2 (SDRAM) t_{csd2}, the minimum delay is 0.8 ns and maximum delay is 6.0 ns, which means that the SDRAM device must also be in accordance with the limit specified. So, while choosing the SDRAM for the design, the electrical characteristics of the device UM also need to considered for all the matching characteristics.

In Figure 40, if the SDRAM clock is 120 MHZ (8.3 ns), the t_{csd2} is within the range of 0.8ns to 6.8 ns. Similar other parameters and the timings can be measured and verified before selecting the device.

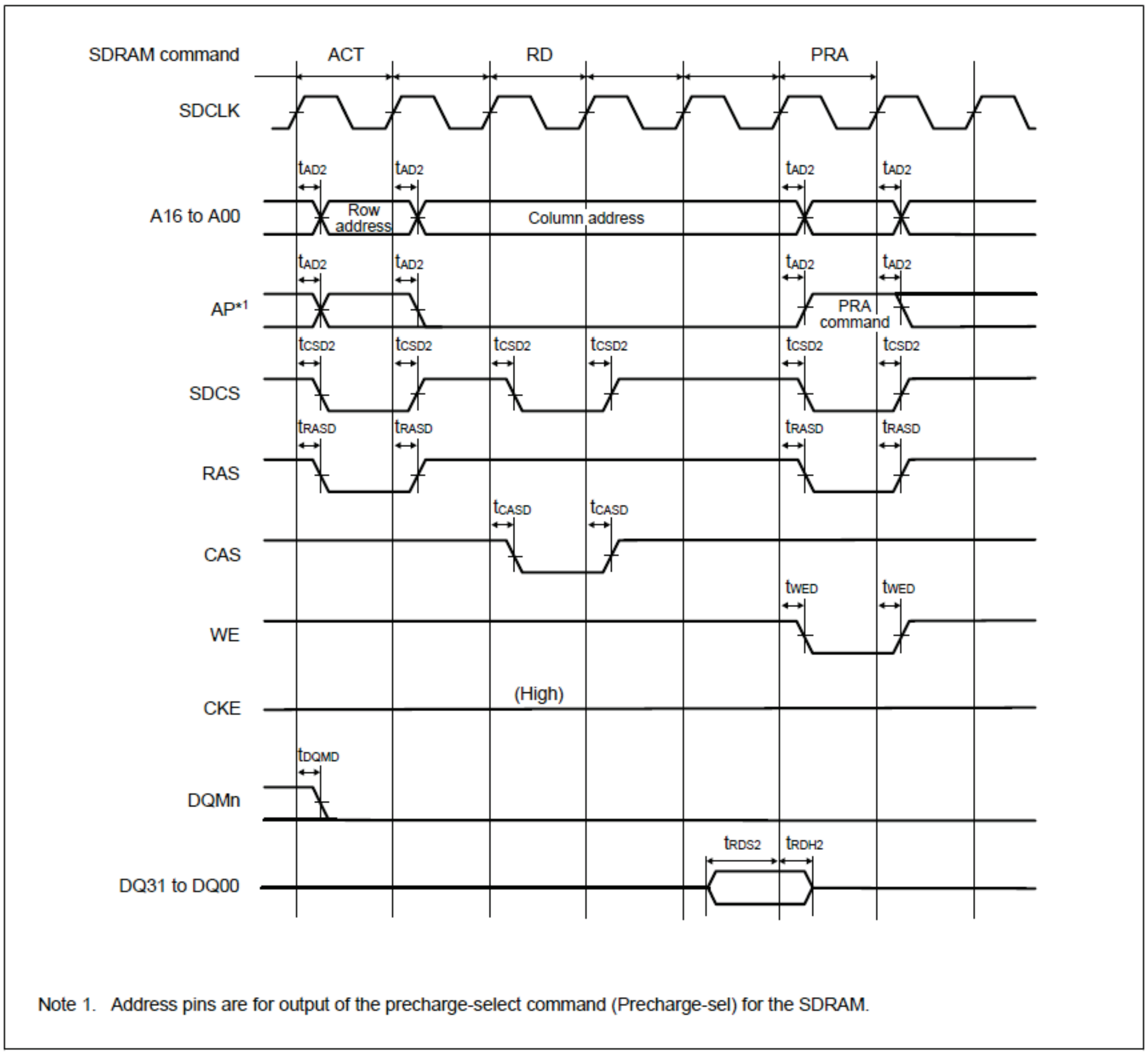


Figure 40. SDRAM Single Read Timing

7. Example Projects and Reference Applications and Application Notes

In this section, the various application projects and example projects, which serve as a reference for the usage of different RA8 memories are given. The users are required to download and run the application by referring to the supporting application notes.

7.1 Reference Example for OSPI

For understanding the operation of the external OSPI interface, refer to the xSPI Example Project using EK-RA8M1. In this Example project, the FSP OSPI driver is used show the operation of the OSPI module. The project uses the onboard OSPI device to demonstrate the 1S-1S-1S and 8D-8D-8D modes. The example project also gives the FSP configuration for the OSPI module.

The Example Project can be found at the location https://github.com/renesas/ra-fsp-examples/tree/master/example_projects/ek_ra8m1 and https://github.com/renesas/ra-fsp-examples/tree/master/example_projects/ek_ra8d1

7.2 Reference Example for SSRAM

The LPM Example project for RA8 will have an option to store the data under SSRAM and retrieve it after the Deep software standby mode 1. It can be found at the location https://github.com/renesas/ra-fsp-examples/tree/master/example_projects/ek_ra8m1 and https://github.com/renesas/ra-fsp-examples/tree/master/example_projects/ek_ra8d1

7.3 Reference Examples for SDRAM

The SDRAM Example project for RA8 is work in progress. Users can refer to the example projects for SDRAM usage under the following names:

- GUIX Hello World
- Thermostat Application using SEGGER emWin.

Note: The above projects are not published at the time of this publication. Alternatively, users can also locate them at renesas.com, under RA8 MCU page.

7.4 Reference Example for D-cache, ITCM, DTCM

The application project that is part of the *High Performance with RA8 MCU using CM85 core with Helium™* contains code demonstrating the usage of D-cache. It also contains a code demonstrating the usage of ITCM and DTCM, thereby showing improved performance.

8. Reference: External Octal Memory Suppliers

RA8 Series MCUs support xSPI compliant Octal SPI (OSPI) interface with support for Execute-In-Place and Decryption-on-the-fly (DOTF). The OSPI interface is compliant with JEDEC standard JESD251 (Profile 1.0 and 2.0), JESD251-1 and JESD252. RA8 Series MCUs are compatible with all flash devices that are JESD xSPI standard compliant and support both Octal Flash & RAM as well as HyperFlash and HyperRAM that are complaint with Infineon's HyperBus Specification.

Here are some of the external memory devices that the RA8x1 MCUs have been tested against

	Supplier	**Part Number**	**Compatible with RA8M1 (Y/N/In progress)**
Octal / Hyper RAM	JSC	JSC28SSU8AGDY	Y
	AP Memory	APS6408L-3OC	In progress
	ISSI	IS66WVO8M8DALL/BLL	In progress
	Cypress	S27KL0641	Y
Octal / Hyper Flash	Infineon	S28HS512TGABHI01	Y
	ISSI	IS25LX032/64/128	Y
	Macronix	MX25LM51245G	Y
	Macronix	MX25UW51245G	Y
	Cypress	S26KL512S	Y
	Micron (XccelaFlash)	MT25QL128ABA MT35XL512ABA	Y
	Cypress	S25FS512S	Y
	Macronix	MX25R1635F	Y

Note: RA8M1 is compatible with all flash devices that are JESD xSPI standard compliant . RA8 xSPI guarantees operation with JESD251 (xSPI for Non Volatile Memory) compliant memory.

9. References

- Renesas FSP User's Manual: https://renesas.github.io/fsp
- https://www.renesas.com/us/en/document/apn/flash-memory-programming
- Renesas RA MCU Datasheets: See http://renesas.com/ra and select the relevant MCU
- RA8 Example Projects on Renesas RA GitHub: https://github.com/renesas/ra-fsp-examples
- RA8 Quick Design Guide (r01an7087eu0100)
- High Performance with RA8 MCU using CM85 core with Helium™ (r01an7127eu0100)
- RA8M1 HW User's Manual (R01UH0994EJ0100 Rev.1.00)
- RA8D1 HW User's Manual (R01UH0995EJ0100 Rev.1.00)
- EK-RA8M1 Board Design Schematics.
- EK-RA8D1 Board Design Schematics.
- AMBA® AXI and ACE Protocol Specification
- Arm® AMBA® 5 AHB Protocol Specification
- AMBA® APB Protocol Version 2.0 Specification
- AMBA® ATB Protocol Specification

Website and Support

Visit the following vanity URLs to learn about key elements of the RA family, download components and related documentation, and get support.

RA Product Information	renesas.com/ra
RA Product Support Forum	renesas.com/ra/forum
RA Flexible Software Package	renesas.com/FSP
Renesas Support	renesas.com/support

Revision History

Rev.	Date	Description	
		Page	Summary
1.00	Oct.26. 23	-	Initial version
1.01	Nov.06.23	-	Minor correction to header
1.02	Dec.11.23	-	Added Memory Vendor List and Support for RA8D1