NXP Semiconductors
Application Note

Document Number: AN12077
Rev. 0, 10/2017

Using the i.MX RT FlexRAM

1. Introduction

This document describes the flexible memory array available on the i.MX RT MCUs. The first part of the document summarizes all features of the FlexRAM memory, including:

- Configuration of the bank array.
- Memory type size definition.
- Available memory controllers.
- Power domains and clocks.
- Interrupt request generation.

The second part of this document demonstrates the FlexRAM configuration usage on a specific application use case. It shows the things to consider in the application to fully utilize the FlexRAM memory in the i.MX RT1050 MCU. It focuses on the application memory capability from the performance point of view in a normal application runtime, as well as the low power feature implementation.

Contents

2. FlexRAM memory

FlexRAM is a highly configurable and flexible RAM memory array. This memory array contains memory banks which can be independently configured to be accessed by different type of interfaces, such as I-TCM (Instruction-Tightly Coupled Memory), D-TCM (Data- Tightly Coupled Memory), or AXI (system). The memory bank can act as an ITCM, DTCM, or OCRAM memory. There are also three different power domains assigned to the dedicated bank (PDRET) or group of banks (PDRAM0, PDRAM1) which reduces the power consumption in the low-power modes.

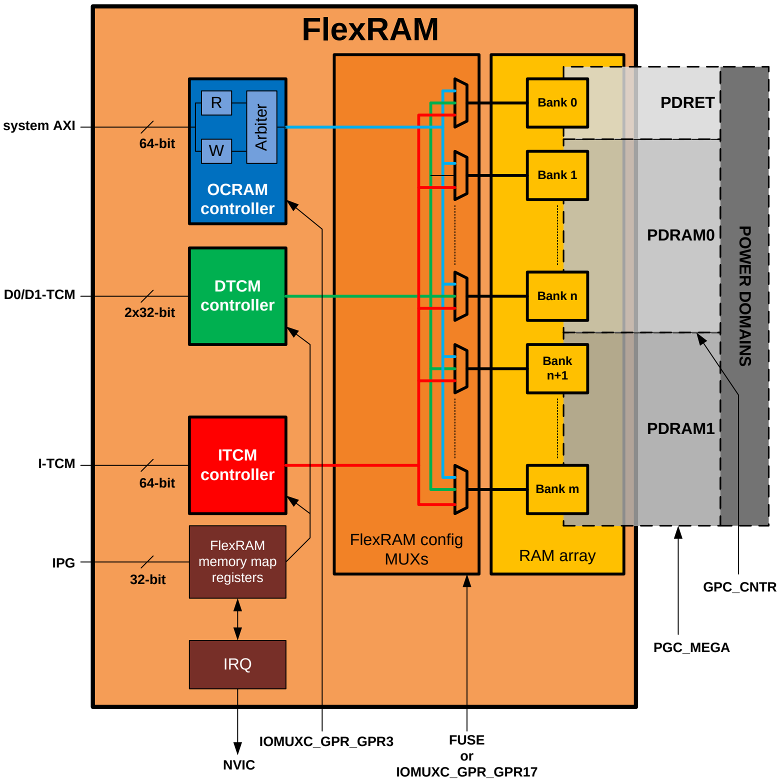


Figure 1. FlexRAM block diagram

2.1. FlexRAM configuration

FlexRAM is a configurable memory RAM array which contains a number of banks.

2.1.1. FlexRAM memory bank configuration

Each bank in the FlexRAM array can be configured to act as:

- I-TCM (Instruction Tightly-Coupled Memory) accessed by the 64-bit I-TCM interface.
- D-TCM (Data Tightly-Coupled Memory) accessed by two 32-bit (D0 and D1) TCM interfaces in the interleaved fashion.
- OC RAM (On-Chip RAM memory) accessed by the 64-bit system AXI bus.

NOTE

All TCM interfaces run at the same frequency as the Arm® Cortex®-M7 core and are synchronous to each other.

The OCRAM controller is connected through the 64-bit AXI bus to one slave port of the interconnect bus fabric (NIC). This slave port frequency is limited to ¼ of the core frequency. For example, if the Arm Cortex-M7 core runs at 528 MHz, then the AXI bus connected to the OCRAM controller is limited to 132 MHz. Expect a performance degradation in the data access to the OCRAM. Of course, the L1 CACHE memory available on the 64-bit system AXI bus master may help with that.

There are two sources to select the configuration of the FlexRAM banks:

- FUSE FlexRAM configuration value (default).
- FLEXRAM_BANK_CFG field value defined in the IOMUXC_GPR_GPR17 register.

The selection between these two sources is done by the value of the FLEXRAM_BANK_CFG_SEL bit defined in the IOMUXC_GPR_PGR16 register. It is set to 0 by default and uses the fuse value for the FlexRAM configuration.

2.1.1.1. Static configuration

The FUSE FlexRAM bank configuration value represents the static configuration of the FlexRAM banks because it cannot be changed after the device boots. The FUSE FlexRAM configuration value uses four fuses in the fusemap (MISC_CONF_LOCK) location which is located at the 0x6D0 address in the [16-19]-bit position. Table 1 shows 16 possible configurations of the FlexRAM banks based on a 4-bit fuse value. The default value is set to 0000, which represents the FlexRAM configuration 0 (see Table 1 for iMX RT1050).

Table 1. 16 possible configurations of FlexRAM banks using fuse value on i.MX RT1050

	FUSE FlexRAM Configuration Value	IOMUXC_GPR_GPR17 (FLEXRAM_BANK_CFG) (binary)	Bank																OCRAM [kB]	DTCM [kB]	ITCM [kB]
	0x6D0 [19:16]		**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**			
0	0b0000	01010101101011111111101001010101	*O*	*O*	*O*	*O*	*D*	*D*	*I*	*I*	*I*	*I*	*D*	*D*	*O*	*O*	*O*	*O*	256	128	128
1	0b0001	01010101010110101111101001010101	*O*	*O*	*O*	*O*	*D*	*D*	*I*	*I*	*D*	*D*	*O*	*O*	*O*	*O*	*O*	*O*	320	128	64
2	0b0010	01011010111111111111111110100101	*O*	*O*	*D*	*D*	*I*	*I*	*I*	*I*	*I*	*I*	*I*	*I*	*D*	*D*	*O*	*O*	128	128	256
3	0b0011	01010101010101011110101010010101	*O*	*O*	*O*	*D*	*D*	*D*	*D*	*I*	*O*	*O*	*O*	*O*	*O*	*O*	*O*	*O*	352	128	32
4	0b0100	01010101010111111111101001010101	*O*	*O*	*O*	*O*	*D*	*D*	*I*	*I*	*I*	*I*	*O*	*O*	*O*	*O*	*O*	*O*	320	64	128
5	0b0101	01010101010101011111101001010101	*O*	*O*	*O*	*O*	*D*	*D*	*I*	*I*	*O*	*O*	*O*	*O*	*O*	*O*	*O*	*O*	384	64	64
6	0b0110	01010101111111111111111110100101	*O*	*O*	*D*	*D*	*I*	*I*	*I*	*I*	*I*	*I*	*I*	*I*	*O*	*O*	*O*	*O*	192	64	256
7	0b0111	11111111111111111111111111110101	*O*	*O*	*I*	*I*	*I*	*I*	*I*	*I*	*I*	*I*	*I*	*I*	*I*	*I*	*I*	*I*	64	0	448
8	0b1000	01011010101011111111101010100101	*O*	*O*	*D*	*D*	*D*	*D*	*I*	*I*	*I*	*I*	*D*	*D*	*D*	*D*	*O*	*O*	128	256	128
9	0b1001	01010101101010101111101010100101	*O*	*O*	*D*	*D*	*D*	*D*	*I*	*I*	*D*	*D*	*D*	*D*	*O*	*O*	*O*	*O*	192	256	64
10	0b1010	10101010111111111111111110100101	*O*	*O*	*D*	*D*	*I*	*I*	*I*	*I*	*I*	*I*	*I*	*I*	*D*	*D*	*D*	*D*	64	192	256
11	0b1011	10101010101010101010101010100101	*O*	*O*	*D*	*D*	*D*	*D*	*D*	*D*	*D*	*D*	*D*	*D*	*D*	*D*	*D*	*D*	64	448	0
12	0b1100	01010101010101011111111101010101	*O*	*O*	*O*	*O*	*I*	*I*	*I*	*I*	*O*	*O*	*O*	*O*	*O*	*O*	*O*	*O*	384	0	128
13	0b1101	01010101010101010101011110010101	*O*	*O*	*O*	*D*	*I*	*O*	*O*	*O*	*O*	*O*	*O*	*O*	*O*	*O*	*O*	*O*	448	32	32
14	0b1110	01010101010111111111111111110101	*O*	*O*	*I*	*I*	*I*	*I*	*I*	*I*	*I*	*I*	*O*	*O*	*O*	*O*	*O*	*O*	256	0	256
15	0b1111	01010101010101010101010101010101	*O*	*O*	*O*	*O*	*O*	*O*	*O*	*O*	*O*	*O*	*O*	*O*	*O*	*O*	*O*	*O*	512	0	0
			O - OCRAM, D - DTCM, I - ITCM																		

2.1.1.2. Runtime configuration

The FlexRAM banks can be also configured at runtime by changing the value of the FLEXRAM_BANK_CFG field defined in the IOMUXC_GPR_GPR17 register. This is possible when the FLEXRAM_BANK_CFG_SEL bit defined the in IOMUXC_GPR_GPR16 register is set to 1. In such case, the configuration of the FlexRAM banks is directly dependent on the value in the FLEXRAM_BANK_CFG field defined in the IOMUXC_GPR_GPR17 register and can be changed at runtime.

The FlexRAM bank configuration value (FLEXRAM_BANK_CFG) is a 32-bit value. This 32-bit value includes the configuration values for each FlexRAM bank (16 banks in total). Each FlexRAM bank uses two bits for the configuration:

- 00b—bank is not used.
- 01b—bank is configured for OCRAM.
- 10b—bank is configured for DTCM.
- 11b—bank is configured for ITCM.

The third column in Table 1 shows the possible FLEXRAM_BANK_CFG field configuration values with respect to the static configuration done by the eFUSEs.

NOTE

It is recommended to write the appropriate FlexRAM bank configuration value (FLEXRAM_BANK_CFG) into the IOMUXC_GPR_GPR17 register before switching from the bank configuration defined by fuses.

Configuring the memory in such way may potentially affect the application safety (stack overflow, invalid instruction execution, out of range access, and so on). Consider the code/data boundary locations properly to avoid application crashes. Also, the memory banks that are being re-configured are changing their access interfaces and still contain the same data.

It is expected that:

- The data/code in the re-configured banks are not required for the application program flow anymore.
- The re-configured banks should be re-filled with the initializer data/code before access (no stack/heap is expected there).
- The addressable memory spaces have changed (avoid potential application pointers to access that space).

2.1.2. Definition of the memory type size

Each memory bank in the FlexRAM array has the same memory size which can be calculated as:

bank size = (Total Flex RAM size) / (number of banks in FlexRAM array)

For example, the iMXRT1050 MCU has:

- A total FlexRAM size of 512 kB.
- 16 banks in the FlexRAM array (block0-block15).
- A bank size of 512 kB / 16 = 32 kB.

The dedicated memory type (ITCM/DTCM/OCRAM) size can be easily calculated as:

Memory type size = (number of banks configured to dedicated memory type) * (bank size)

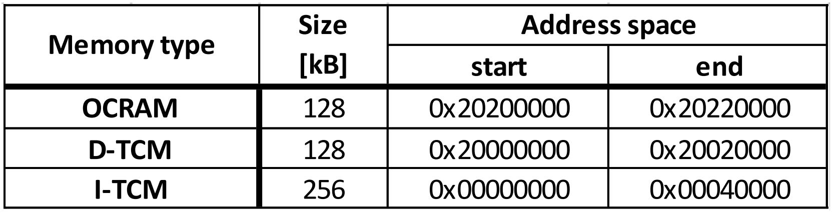
The memory type size depends strictly on the application needs and, in total, can vary (based on the configuration) from 0 B to the total FlexRAM size. For example, on iMXRT1050, it can range from 0 to 512 kB. The specified memory type (ITCM/DTCM/OCRAM) does not have to be configured strictly in a continuous block of FlexRAM banks.

Table 2. Example of non-continuous block of DTCM/OCRAM banks' configuration (i.MX RT1050)

IOMUXC_GPR_GPR17 (FLEXRAM_BANK_CFG)	**Bank**																**OCRAM**	**D-TCM**	**I-TCM**
	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**			
01011010111111111111111110100101	*O*	*O*	*D*	*D*	*I*	*I*	*I*	*I*	*I*	*I*	*I*	*I*	*D*	*D*	*O*	*O*	128	128	256

This still represents a continuous address space in the device memory map (see Table 3).

Table 3. Example of memory type address space



Memory type	Size [kB]	Address space	
		start	end
OCRAM	128	0x20200000	0x20220000
D-TCM	128	0x20000000	0x20020000
I-TCM	256	0x00000000	0x00040000

This feature enables you to configure the memory type and its size in accordance with the application needs from the power consumption aspect (see next part for more details).

NOTE

> The OCRAM cannot be configured to 0 kB due to the boot ROM code requirements. The 32 kB of OCRAM represents the minimum system requirement. The ITCM or DTCM can be configured to 0 kB (see also the possible static configuration shown in Table 1).
>
> The Arm Cortex-M7 specifications require the size of ITCM/DTCM to be a power-of-two number, which can conflict with the FlexRAM configuration capability. Avoid access to the empty RAM space configured by the corresponding FlexRAM configuration.

2.2. FlexRAM memory controllers

FlexRAM includes memory controllers responsible for converting the AXI (OCRAM) or TCM (I-TCM, D-TCM) interface signals into the RAM array interface signals. There are multiplexers between each memory controller and each RAM array bank that are responsible for a proper connection of the memory controller and its RAM bank based on the FlexRAM configuration. These memory controllers also control the access to the memory.

2.2.1. TCM memories controllers

The TCM controller converts the TCM (64-bit I-TCM bus or 2x32-bit D-TCM buses) interface signals into an RAM array signal. The TCM interfaces are synchronized with the Cortex-M7 core and run at the same frequency. The TCM controller can also control the access to the RAM bank and affects the memory data access time (fetches the instructions on the I-TCM or accesses the data on the D-TCM). You may choose between two modes for both the read and write accesses:

- Fast access mode (default)—the access is expected to be done in one cycle.
- Wait access mode—the access is expected to be done in two cycles.

This can be done by enabling/disabling the appropriate access bit in the TCM control register (TCM_CTRL) in the FlexRAM memory map:

- TCM_RWAIT_EN:
  - 0—fast mode selected.
  - 1—wait mode selected.

- TCM_WWAIT_EN:
  - 0—fast mode selected.
  - 1—wait mode selected.

The TCM controllers also include the dynamic clock gate control to reduce the power consumption when nothing is accessed.

2.2.2. OCRAM memory controller

The OCRAM memory acts as a slave port module connected to the 64-bit system AXI bus. It contains the OCRAM controller which handles the FlexRAM banks according to its configuration. The OCRAM controller converts the AXI interface signal to the RAM array signal. The read and write transactions are handled by two independent read and write control modules. The controller also contains an arbiter which takes control when two simultaneous requests come from both the read and write modules. The arbiter works in the round-robin scheme. The simultaneous read and write transactions are possible when targeted to different memory banks. When targeting the same bank, the read access gets a higher priority.

The OCRAM controller includes also features to avoid timing issues when the memory runs at a higher frequency. It supports adding the wait-states/pipeline into the memory access:

- Read/write address pipeline:
  - When enabled, this feature delays the reading/writing of an address from the AXI master by one cycle before it is accepted by the memory.
- Write data pipeline:
  - When enabled, this feature delays the writing of data from the AXI master by one cycle before it is accepted by the memory.
- Read data wait-state:
  - When enabled, this feature takes two cycles for each read access.

All previously mentioned features can be enabled/disabled by the corresponding OCRAM control bit (OCRAM_CTL[3:0]) in the general-purpose register **(**IOMUXC_GPR_GPR3).

After performing any changes in the OCRAM control field (OCRAM_CTL[3:0]) in the IOMUXC_GPR_GPR3 register, it is recommended to wait until the corresponded OCRAM status bit (OCRAM_STATUS[19:16]) changes from 1 to 0 (1 means that the configuration is changed, but not applied yet).

NOTE

> The 64-bit AXI bus on the OCRAM controller slave port runs at one quarter of the Cortex-M7 core frequency. The interconnect bus fabric (NIC) runs at the same frequency. For example, if the Cortex-M7 core runs at 528 MHz, the OCRAM interface runs at 132 MHz.
>
> It is expected that the small piece of code that changes checks that the bit is not accessing the OCRAM.

2.3. FlexRAM module-related clocks and clock gates

The table below summarizes all clock related to FlexRAM module:

Table 4. FlexRAM module

Clock Name	Clock Source			peripheral interface / registers access clock			Clock gate
	Name	Default frequency	Maximum frequency	Name	Default frequency	Maximum frequency	
flexram_clk	module clock	*6 MHz*	*600 MHz*	ipg_clk_root	*3 MHz*	*150 MHz*	CCM_CCGR3[CG9] - flexram_clk_en
ocram_clk	ocram_exsc_aclk_exsc	*3 MHz*	*150 MHz*	ipg_clk_root	*3 MHz*	*150 MHz*	CCM_CCGR3[CG14]- ocram_clk_en

The FlexRAM clock (flexram_clk) is the main clock the module is clocked by and is derived from the Arm core clock. The TCMs are fed by the same source clock. The on-chip RAM controller is fully synchronized to the system AXI interface and the clock for the OCRAM (ocram_exsc_aclk_exsc) is derived from it and further divided by 4. The last are the peripheral registers related to the FlexRAM module that are accessed by the peripheral bus (IPG interface). This part is clocked by the peripheral bus clock (ipg_clk_root).

NOTE

The interconnect bus fabric (NIC) is a bus matrix which interconnects the bus masters (like ARM AXIM, DMA, USB, ENET, uSDHC) with the bus slaves (OCRAM controller, ARM AHBS, FlexSPI module, SEMC, and so on). It runs at a quarter of the Arm Cortex-M7 core frequency. The OCRAM runs at the same frequency as the interconnect bus fabric.

2.4. FlexRAM power domains

The FlexRAM bank array is partitioned into three different power sub-domains:

- PDRET domain.
- PDRAM0 domain.
- PDRAM1 domain.

Table 5 shows the state of the dedicated power domain in different low-power modes.

Table 5. FlexRAM array power domains state in different low-power modes

	System IDLE	Low Power IDLE	SUSPEND	SNVS
ARM core	**WFI**	**WFI**	**power down**	**OFF**
FlexRAM (PDRET)	*ON*	*ON*	*ON*	*OFF*
FlexRAM (PDRAM0)	*ON*	*ON*	*ON/OFF*	*OFF*
FlexRAM (PDRAM1)	*ON/OFF*	*ON/OFF*	*Power Down*	*OFF*

The main purpose to split the FlexRAM banks into different power domains is to save power when running in different power modes.

2.4.1. PDRET power domain

This power domain is connected to FlexRAM bank 0, which represents 32 kB of memory space. This domain is always on (it is powered on down to the suspend mode). The only exception when the PDRET domain is powered off is the SNVS, which is a completely independent low-power domain usually supplied by a separate power supply (LiION battery).

2.4.2. PDRAM0 power domain

This power domain is connected to FlexRAM banks 1-7, which represents 224 kB of memory space. When the Arm core is powered off, this domain can be either powered on for data retention or powered off together with the core.

This feature is controllable via the FlexRAM PDRAM0 power gate enable (PDRAM0_PGE) bit in the general power controller interface control (GPC_CNTR) register. When this bit is set (default), the power domain for banks 1-7 keeps its content even the Arm core is powered down. When it is cleared, the PDRAM0 power domain is powered off when the Arm core is powered down.

2.4.3. PDRAM1 power domain

This power domain is connected to FlexRAM banks 8-15, which represents 256 kB of memory space.

This domain’s power gating is controlled via the power gate control mega (PGC_MEGA) registers:

- Power sequence timing control:
    - Power-down sequence time (PGC_MEGA_PDNSCR):
        - Between the power-down request and asserting the isolation defined by the number of IPG cycles in the ISO value.
        - Between asserting the isolation and the negation of the power-toggle signal defined by the number of IPG cycles in the SW2ISO value.
    - Power-up sequence time (PGC_MEGA_PUPSCR):
        - Between the power-up request and asserting the power-toggle signal defined by the number of IPG cycles in the ISO value.
        - Between asserting the power-toggle signal and the negation of isolation defined by the number of IPG cycles in the SW2ISO value.
- Power control (PGC_MEGA_CTRL):
    - Controls whether the power-down request signal switches the power for this domain on or off.

NOTE

The data are considered to be placed into the FlexRAM bank 0, which can keep the data retained in the suspend mode when no code execution is possible.

2.5. FlexRAM interrupt

The FlexRAM can generate an interrupt request if the address is out of range (or not allocated), based on its configuration. It means that the interrupt request is generated when the address out-of-range event occurs on the ITCM, DTCM, or OCRAM memories. This is configurable via the FlexRAM interrupt generation register (INT_SIG_EN). Each memory type has its dedicated bit in this register:

- OCRAM_ERR_SIG_EN—when set, it enables the generation of the OCRAM address out-of-range interrupt signal.
- DTCM_ERR_SIG_EN—when set, it enables the generation of the ITCM address out-of-range interrupt signal.
- ITCM_ERR_SIG_EN—when set, it enables the generation of the DTCM address out-of-range interrupt signal.

When enabled, all these interrupt error signals share the same interrupt line (are ORed), dedicated to the interrupt request number 38 in the NVIC.

Because there is just one interrupt vector for all three events, it is required to identify the dedicated source in the interrupt service routine by reading the interrupt status register (INT_STATUS):

- OCRAM_ERR_STATUS—when set, it signals that the OCRAM address out-of-range interrupt signal is generated.
- DTCM_ERR_STATUS—when set, it signals that the DTCM address out-of-range interrupt signal is generated.
- ITCM_ERR_STATUS—when set, it signals that the ITCM address out-of-range interrupt signal is generated.

The assertion of the status bits is conditioned by enabling the dedicated memory type bit in the interrupt status enable register (INT_STAT_EN):

- OCRAM_ERR_STAT_EN—when set, it signals that the OCRAM address out-of-range interrupt signal is generated.
- DTCM_ERR_STATUS_EN—when set, it signals that the DTCM address out-of-range interrupt signal is generated.
- ITCM_ERR_STATUS_EN—when set, it signals that the ITCM address out-of-range interrupt signal is generated.

3. Using FlexRAM features in the application

The FlexRAM memory implemented on the i.MX RT devices provides a big advantage when compared to devices with single on-chip memories. Depending on the application memory footprint (code/constant data/static data/stack, and so on), it is possible to reconfigure the on-chip memory and make the device more suitable for the application. This can be very useful for different applications built on the same hardware platform (for example). This approach is very attractive nowadays because it saves the development time/cost/production time and rapidly reduces the time to market.

The next sub-sections describe the flexibility of the FlexRAM from the application point of view.

It focuses on one specific configuration technique from the time perspective. It demonstrates the combination of the static and dynamic configurations of the FlexRAM memory. It cannot be called a static configuration because it does not use the fuse default configuration, which happens immediately after the device boot. This approach cannot be used due to the unavailable configuration in the fuse table (see Table 1) required by the application use case. It is not truly dynamic because it does not change the configuration during application run-time. It configures the FlexRAM memory in the application startup (just before calling the static data and read-write code initialization) and it keeps the start-up setting after that. It is possible to make the FlexRAM configuration decision based on the memory footprint requirements which can be identified during the application building time.

3.1. FlexRAM configuration demonstration

The application use case of the FlexRAM configuration described here represents the case when the definition of a memory section size is more precisely identified according to the application code compiler/linker outputs. The size of the ITCM/DTCM/OCRAM memory depends on how much code/constant data/static data/stack memory the application requires. It is similar to the static configuration described at the beginning of this document.

3.1.1. External memory versus FlexRAM memory access consideration

The i.MX RT devices do not have flash. An unlimited size of code/data can be loaded into the external memory. It is limited only by the external RAM region (1 GB) defined by Arm (0x6000 0000 – 0x9FFF FFFF). On the i.MX RT devices, the external memory is accessible by the FlexSPI/SEMC interfaces (and other). These interfaces are not fast enough to execute the code/access the data without a penalty in the wait states when running at a maximum frequency (considering a case when the core runs at 600 MHz and the FlexSPI at a maximum of 166 MHz (DDR mode) or SEMC at 166 MHz and the L1 I/D cache cannot cover all the code/data). On the other hand, the execution of code/the access to data from the TCMs can be considered as a single cycle (see the exceptions in the above sections).

It is also important to consider what code/data can be placed into the external memory before the FlexRAM configuration. The most critical code to execute as fast as possible (without the wait states to fully utilize the super-scalar pipeline nature of Cortex-M7) must be placed into the ITCM (Harvard bus architecture nature is preferred in this consideration). The less important data (accessed sporadically) must be placed into the external memory. The data that is accessible only by the core (or stack) must be placed into the DTCM. The Cortex-M7 core also supports direct access to the TCM through the AHBS interface. The DMA master can still access the TCM through the AHBS. However, the access is not as fast as the access by the core. It shall be used when the core is sleeping/powered down.

The data accessed by more than one bus master (for example; the core and DMA) must be placed into the OCRAM, especially when it is accessed by the DMA in the low-power modes.

Consider all above-mentioned features in the application memory footprint before the FlexRAM configuration.

3.1.2. FlexRAM configuration

The FlexRAM configuration must take the linker-defined memory section sizes into account. The definition of section sizes can be adjusted according to the application needs during the application development. The FlexRAM configuration may reflect the linker sections definition.

Here are some application examples:

- Sensing the image from a camera (using the CSI module) with a resolution of 320x240 in the 5-6-5 RGB mode.
- Storing two raw-data images in the data buffers using the DMA module and displaying them dynamically using the DMA on the LCD module.
- Processing an image (only software-based) with the results of an image stored in a 30-kB data buffer.
- The image processing must be done as fast as possible.
- Store the last four images' processing data results (4 x 30-kb data buffers).
- The last processed image data results must be available in the memory after waking up from the stop mode.
- There are three additional data buffers in the same raw image data format which are displayed sporadically (application idle state).

Assumption: during the project development, the application software requires:

Table 6. Requirements

CODE	DATA
120kB	889kB

3.1.2.1. Code memory footprint

The interrupt vector table and a couple of critical interrupt service routines must be placed in the ITCM (a 64-bit single-cycle access memory can pre-fetch 64-bit, 4 x 16-bit, or 2 x 32-bit instructions) to speed up its execution time. The interrupt vectors and corresponding interrupt service routines take 46 kB of memory.

3.1.2.2. Static data memory footprint

There are two 150-kB data buffers of that contain the raw data from the image sensor (320x240 in the RGB 5-6-5 format). These buffers are filled by the DMA channel and triggered by the camera sensor interface module (in a ping-pong fashion). They are also read by the core and processed. It is convenient to place these data buffers into the OCRAM which is accessed by the 64-bit system AXI bus.

The additional four data buffers are 30 kB in size and contain the resulting data of the image processing done by the core. These buffers are accessed only by the core and it is recommended to place these buffers into the DTCM memory that is accessed directly by the core data request. The remaining application static data (initialized, non-initialized, or initialized to zero) is handled only by the core and 15 kB in size. The best location for such data is the DTCM.

There are also three 150-kB constant data buffers (for example, static LCD images) which are displayed sporadically, based on the stand-by mode calling in the application. These buffers can be stored in an external type of memory.

Using the stack usage analyses (for example, www.iar.com/support/resources/articles/mastering-stack-and-heap-for-system-reliability), the stack size is estimated to be 4 kB (10 % addition included). The stack shall be placed into the DTCM because it is accessed only by the core.

3.1.2.3. Total memory footprint

Considering the above assumptions, it can be summarized:

Table 1. Application memory section requirements

	CODE		DATA			
	RO	RW	RO	RW	RW	RW
			constants	static (CPU only)	stack	static (all bus masters)
Memory size requirements	*74*	*46*	*150*	*30*	*4*	*150*
			150	*30*		*150*
			150	*30*		
				30		
				15		
In total	**74**	**46**	**450**	**135**	**4**	**300**

The total RAM memory requirement (in this case) is 483 kB. It fits into the i.MX RT1050 device with a 512-kB memory. However, when it is re-calculated into the 32-kB bank size, it is 544 kB, which means one more bank is required (see in table below).

Table 2. Application memory section assignments based on previous table

	External memory	ITCM	External memory	DTCM	OCRAM	In TOTAL memory / banks
Memory size requirements	74	46	450	139	300	485
Number of FlexRAM banks required		2		5	10	17
Total size of memory		64		160	320	544

In this case, it is desirable to think about moving 15 kB of static data from the DTCM to the OCRAM. It depends on:

- The DMA channels loading the OCRAM controller write (CSI)/read (LCD) modules via the system AXI bus.
- What is the request to access the application static data from the performance point of view.

If the approach of moving data from the DTCM to the OCRAM does not significantly affect the overall performance, then it can be done and fits into the FlexRAM memory configuration (see Table 3). If the approach cannot be applied due to performance degradation, then an additional solution must be found; for example, reducing the code executed from I-TCM or reducing the number of image processing result data buffers.

Table 3 shows the case when the application static data are moved from the DTCM memory to the OCRAM memory, considering no affect to performance.

Table 3. Application memory section requirements when application static data moved to OCRAM

	CODE		DATA			
	RO	RW	RO	RW	RW	RW
			constants	static (CPU only)	stack	static (all bus masters)
Memory size requirements	*74*	*46*	*150*	*30*	*4*	*150*
			150	*30*		*150*
			150	*30*		*15*
				30		
In total	**74**	**46**	**450**	**120**	**4**	**315**

This memory configuration fits into the RT1050 FlexRAM and all 16 banks are utilized.

Table 4. Application memory section assignments based on previous table

	External memory	ITCM	External memory	DTCM	OCRAM	In TOTAL memory / banks
Memory size requirements	74	46	450	124	315	485
Number of FlexRAM banks required		2		4	10	16
Total size of memory		64		128	320	512

The number of banks for each memory type configuration are known. However, it is still not clear what bank number uses what configuration. This depends on the application needs from the low-power view, because there are three different power domains used for the corresponding bank/bank groups. The features of each FlexRAM power domain can be utilized in the low-power modes of application. In this case, the application requires to retain the data of the last processed imaged data buffer. The size of this buffer (30 kB) fits into the size of one bank (32 kB). The FlexRAM bank 0 is in the PDRET power domain and keeps the data content down to the stop mode. The FlexRAM bank 0 can be used to store the last image processed result data buffer content. The remaining memory is used only in the normal run power mode.

Tab. 10 shows detailed configuration of FlexRAM bank for this example. The figures 2-4 shows also the starting addresses and the content of the individual memory configuration.

Table 5. Application example of FlexRAM banks configuration

FlexRAM bank	Configuration	Size	Power domain
Bank0	DTCM	128kB	PDRET
Bank1	DTCM		PDRAM0
Bank2	DTCM		
Bank3	DTCM		
Bank4	ITCM	64kB	
Bank5	ITCM		
Bank6	OCRAM	320kB	
Bank7	OCRAM		
Bank8	OCRAM		PDRAM1
Bank9	OCRAM		
Bank10	OCRAM		
Bank11	OCRAM		
Bank12	OCRAM		
Bank13	OCRAM		
Bank14	OCRAM		
Bank15	OCRAM		

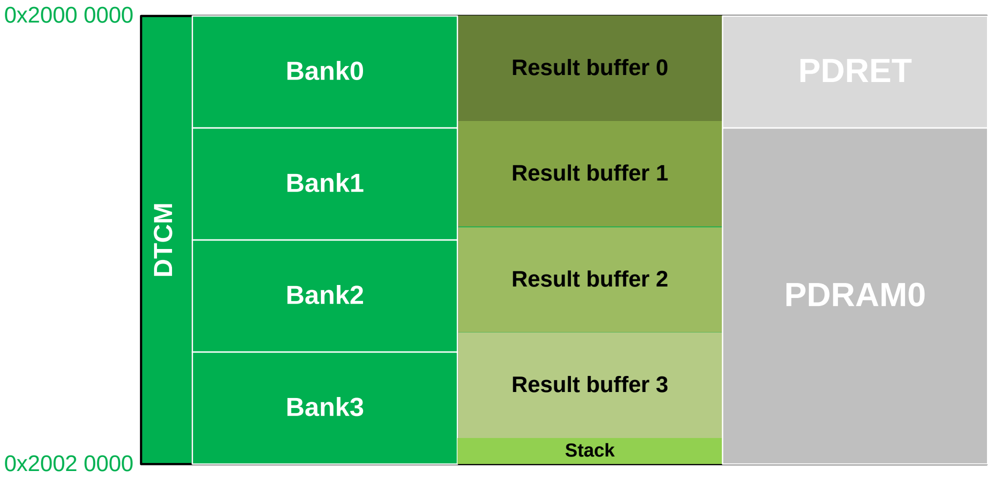


Figure 2. DTCM memory addresses, configuration, content, and appropriate power domain

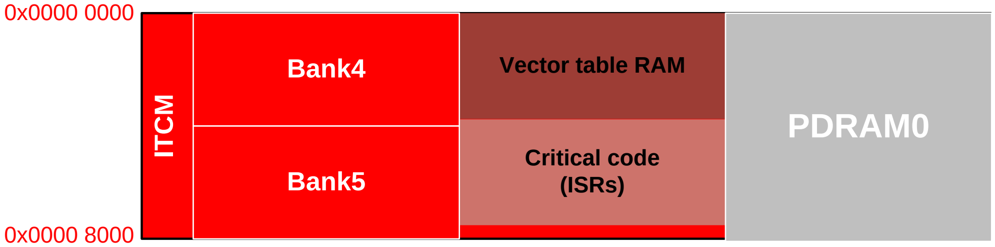


Figure 3. ITCM memory addresses, configuration, content, and appropriate power domain

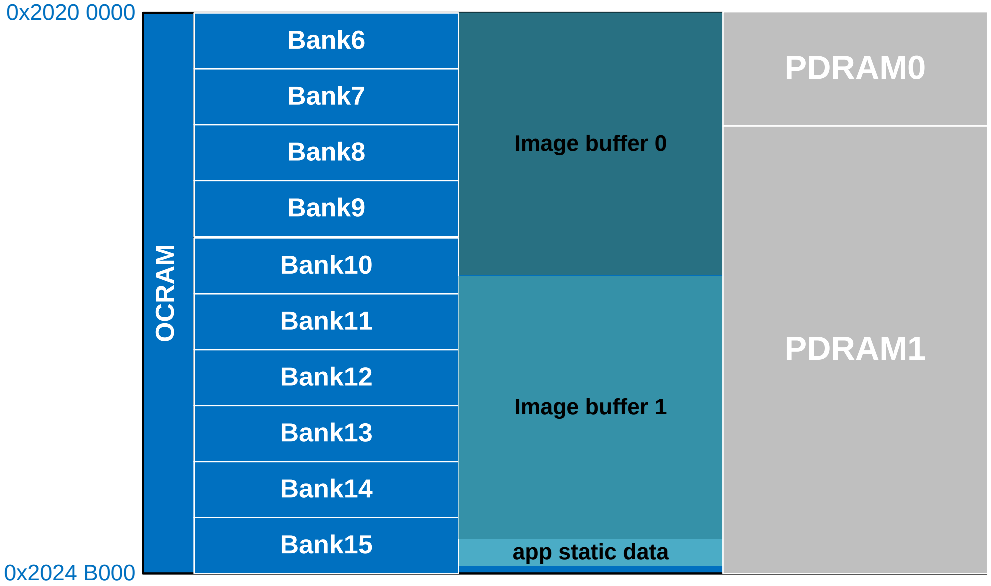


Figure 4. OCRAM memory addresses, configuration, content, and appropriate power domain

As shown in Table 1, there is no valid FlexRAM configuration available in the fuse default setting aligned with the configuration required by this use case. The static configuration cannot be utilized here. However, it is possible to configure the FlexRAM by the run-time configuration approach defined in Section 2.1, “FlexRAM configuration”. This configuration must be done before the start-up code calls the static data and r/w code initialization.

NOTE

Consider releasing the configuration of the application project when compiling. Consider placing the application static data into different FlexRAM banks because simultaneous 32-bit accesses for read and write are possible on the OCRAM controller when accessing different banks.

How to Reach Us:

Home Page:
www.nxp.com

Web Support:
www.nxp.com/support

Information in this document is provided solely to enable system and software implementers to use NXP products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits based on the information in this document. NXP reserves the right to make changes without further notice to any products herein.

NXP makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does NXP assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. “Typical” parameters that may be provided in NXP data sheets and/or specifications can and do vary in different applications, and actual performance may vary over time. All operating parameters, including “typicals,” must be validated for each customer application by customer’s technical experts. NXP does not convey any license under its patent rights nor the rights of others. NXP sells products pursuant to standard terms and conditions of sale, which can be found at the following address: www.nxp.com/SalesTermsandConditions.