NXP Semiconductors
Application Note

Document Number: AN12077
Rev. 3, 01/2021

Using the i.MX RT FlexRAM

1. Introduction

This document describes the flexible memory array available on the i.MX RT 4-digit crossover processors. 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 can also be power domains assigned to a dedicated FlexRAM bank or a group of banks, which can potentially reduce the power consumption in the low-power modes.

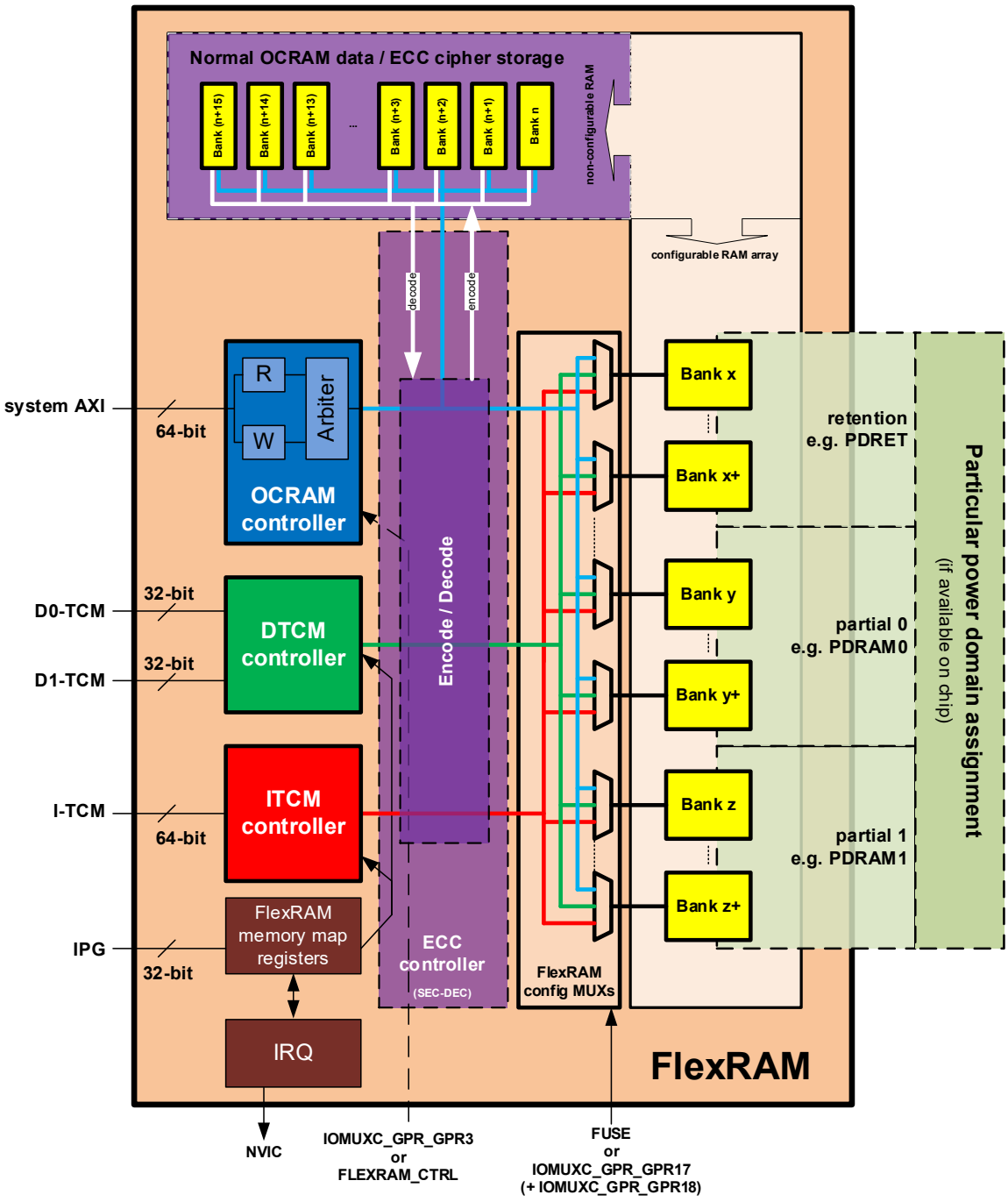

Figure 1. General block diagram of FlexRAM

NOTE

On some i.MX RT 4-digit devices, an additional OCRAM (which is not part of the FlexRAM) can be found. It is used to increase the total on-chip memory size. This kind of memory is not considered in this document as it is not included in the FlexRAM array.

NOTE

All the dashed blocks in Figure 1 are parts that are chip-specific. It means that they do not have to be available on all devices.

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 word-to-word 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. For example, on RT 1050, 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 performance degradation in the data access to the OCRAM in comparison to the xTCM memories. The L1 CACHE memory can help with that.

On RT10xx, the NIC clock ratio of the slave port versus the master port is fixed to 4 (for example, 528 MHz/132 MHz). On RT117x, the ratio depends on master clock and bus clock setting, for example in default it is 1GHz/240MHz

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 on RT 10xx devices. On RT117x devices, there is an additional IOMUXC_GPR_GPR18 register for configuration due to dual-core features implementation.

The selection between these two sources is done by the value of the FLEXRAM_BANK_CFG_SEL bit defined in the IOMUXC_GPR_GPR16 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 the fuses in the fusemap located at the 0x6D0 address in the [16-19]-bit position (fuses are called Default_FlexRAM_Part). Table 1 shows an example of the available configurations of the FlexRAM banks based on a corresponding device fuses setting. The blank device value is set to 0000, which represents the default FlexRAM configuration 0.

NOTE

The minimum configuration of OCRAM is 64 KB (see Table 1). This is required due to ROM code requires at least 64 KB of RAM for its execution. The minimum OCRAM requirements can be device dependent.

Table 1. Static FlexRAM configuration defined by fuses in RT1010

	FUSE FlexRAM Configuration Value	IOMUXC_GPR_GPR17 (FLEXRAM_BANK_CF) (binary)	Bank				OCRAM [kB]	DTCM [kB]	ITCM [kB]
	0x6D0 [19:16]		0	1	2	3			
0	0b0000	11100101	O	O	D	I	64	32	32
1	0b0001	11101001	O	D	D	I	32	64	32
2	0b0010	10100101	O	O	D	D	64	64	0
3	0b0011	10101001	O	D	D	D	32	96	0
4	0b0100	11111001	O	D	I	I	32	32	64
5	0b0101	01100101	O	O	D	O	96	32	0
6	0b0110	11111101	O	I	I	I	32	0	96
7	0b0111	11110101	O	O	I	I	64	0	64
8	0b1000	01110101	O	O	I	O	96	0	32
9	0b1111	01010101	O	O	O	O	128	0	0
	O - OCRAM, D - DTCM, I - ITCM								

Table 2. Static FlexRAM configuration defined by fuses in RT1020

	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			
0	0b0000	0101111110100101	O	O	D	D	I	I	O	O	128	64	64
1	0b0001	1111101010100101	O	O	D	D	D	D	I	I	64	128	64
2	0b0010	0101101010100101	O	O	D	D	D	D	O	O	128	128	0
3	0b0011	1110101010010101	O	O	O	D	D	D	D	I	96	128	32

4	0b0100	1111111110100101	O	O	D	D	I	I	I	I	64	64	128
5	0b0101	1010101010100101	O	O	D	D	D	D	D	D	64	192	0
6	0b0110	0101011110100101	O	O	D	D	I	O	O	O	160	64	32
7	0b0111	0101010110100101	O	O	D	D	O	O	O	O	192	64	0
8	0b1000	0101111101100101	O	O	D	O	I	I	O	O	160	32	64
9	0b1001	1111111101100101	O	O	D	O	I	I	I	I	96	32	128
10	0b1010	1111111111100101	O	O	D	I	I	I	I	I	64	32	160
11	0b1011	0101010101100101	O	O	D	O	O	O	O	O	224	32	0
12	0b1100	1111111101010101	O	O	O	O	I	I	I	I	128	0	128
13	0b1101	0101011101100101	O	O	D	O	I	O	O	O	192	32	32
14	0b1110	1111111111110101	O	O	I	I	I	I	I	I	64	0	192
15	0b1111	0101010101010101	O	O	O	O	O	O	O	O	256	0	0
			O - OCRAM, D - DTCM, I - ITCM										

Table 3. Static FlexRAM configuration defined by fuses in RT1050 / RT106x

	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	01010101010101011101010010101	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																		

Table 4. Static FlexRAM configuration defined by fuses on RT1170

FLEXRAM re-configurable part																							
	FUSE FlexRAM Configuration Value	IOMUXC_GPR_GPR17 + IOMUXC_GPR_GPR18 (FLEXRAM_BANK_CFG_LOW + FLEXRAM_BANK_CFG_HIGH) (binary)	Bank																		OCRAM [kB]	DTCM [kB]	ITCM [kB]
	0xC70 [5:0]		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15					
0	6'b000000	11111111101010101111111110101010	D	D	D	D	I	I	I	I	D	D	D	D	I	I	I	I	0	256	256
1	6'b000001	11111111101010101111101010101010	D	D	D	D	D	D	I	I	D	D	D	D	I	I	I	I	0	320	192
2	6'b000010	11111010101010101111101010101010	D	D	D	D	D	D	I	I	D	D	D	D	D	D	I	I	0	384	128
3	6'b000011	10101010101010101111101010101010	D	D	D	D	D	D	I	I	D	D	D	D	D	D	D	D	0	448	64
4	6'b000100	10101010101010101010101010101010	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	0	512	0
5	6'b000101	11111111111110101111111110101010	D	D	D	D	I	I	I	I	D	D	I	I	I	I	I	I	0	192	320
6	6'b000110	11111111111110101111111111111010	D	D	I	I	I	I	I	I	D	D	I	I	I	I	I	I	0	128	384
7	6'b000111	11111111111111111111111111111010	D	D	I	I	I	I	I	I	I	I	I	I	I	I	I	I	0	64	448
8	6'b001000	11111111111111111111111111111111	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	0	0	512
9	6'b001001	11111010101011111111101010100101	O	O	D	D	D	D	I	I	I	I	D	D	D	D	I	I	64	256	192
10	6'b001010	11111010101010101111101010100101	O	O	D	D	D	D	I	I	D	D	D	D	D	D	I	I	64	320	128
11	6'b001011	10101010101010101111101010100101	O	O	D	D	D	D	I	I	D	D	D	D	D	D	D	D	64	384	64
12	6'b001100	10101010101010101010101010100101	O	O	D	D	D	D	D	D	D	D	D	D	D	D	D	D	64	448	0
13	6'b001101	11111111101011111111101010100101	O	O	D	D	D	D	I	I	I	I	D	D	I	I	I	I	64	192	256
14	6'b001110	11111111111111111010111110100101	O	O	D	D	I	I	D	D	I	I	I	I	I	I	I	I	64	128	320
15	6'b001111	11111111111111111111111110100101	O	O	D	D	I	I	I	I	I	I	I	I	I	I	I	I	64	64	384
16	6'b010000	11111111111111111111111111110101	O	O	I	I	I	I	I	I	I	I	I	I	I	I	I	I	64	0	448
17	6'b010001	11111010111111111010101001010101	O	O	O	O	D	D	D	D	I	I	I	I	D	D	I	I	128	192	192
18	6'b010010	11111010101011111010101001010101	O	O	O	O	D	D	D	D	I	I	D	D	D	D	I	I	128	256	128
19	6'b010011	10101010101010101111101001010101	O	O	O	O	D	D	I	I	D	D	D	D	D	D	D	D	128	320	64
20	6'b010100	10101010101010101010101001010101	O	O	O	O	D	D	D	D	D	D	D	D	D	D	D	D	128	384	0
21	6'b010101	11111111101011111111101001010101	O	O	O	O	D	D	I	I	I	I	D	D	I	I	I	I	128	128	256
22	6'b010110	11111111111111111111101001010101	O	O	O	O	D	D	I	I	I	I	I	I	I	I	I	I	128	64	320
23	6'b010111	11111111111111111111111101010101	O	O	O	O	I	I	I	I	I	I	I	I	I	I	I	I	128	0	384
24	6'b011000	11111010101011111010010101010101	O	O	O	O	O	O	D	D	I	I	D	D	D	D	I	I	192	192	128
25	6'b011001	10101010101011111010010101010101	O	O	O	O	O	O	D	D	I	I	D	D	D	D	D	D	192	256	64
26	6'b011010	10101010101010101010010101010101	O	O	O	O	O	O	D	D	D	D	D	D	D	D	D	D	192	320	0
27	6'b011011	11111111101011111010010101010101	O	O	O	O	O	O	D	D	I	I	D	D	I	I	I	I	192	128	192
28	6'b011100	11111111111111111010010101010101	O	O	O	O	O	O	D	D	I	I	I	I	I	I	I	I	192	64	256
29	6'b011101	11111111111111111111010101010101	O	O	O	O	O	O	I	I	I	I	I	I	I	I	I	I	192	0	320
30	6'b011110	11111010111110100101010101010101	O	O	O	O	O	O	O	O	D	D	I	I	D	D	I	I	256	128	128
31	6'b011111	10101010111110100101010101010101	O	O	O	O	O	O	O	O	D	D	I	I	D	D	D	D	256	192	64
32	6'b100000	10101010101010100101010101010101	O	O	O	O	O	O	O	O	D	D	D	D	D	D	D	D	256	256	0
33	6'b100001	11111111111110100101010101010101	O	O	O	O	O	O	O	O	D	D	I	I	I	I	I	I	256	64	192
34	6'b100010	11111111111111110101010101010101	O	O	O	O	O	O	O	O	I	I	I	I	I	I	I	I	256	0	256
35	6'b100011	10101111101001010101010101010101	O	O	O	O	O	O	O	O	O	O	D	D	I	I	D	D	320	128	64
36	6'b100100	10101010101001010101010101010101	O	O	O	O	O	O	O	O	O	O	D	D	D	D	D	D	320	192	0
37	6'b100101	11111111101001010101010101010101	O	O	O	O	O	O	O	O	O	O	D	D	I	I	I	I	320	64	128
38	6'b100110	11111111111101010101010101010101	O	O	O	O	O	O	O	O	O	O	I	I	I	I	I	I	320	0	192
39	6'b100111	11111010010101010101010101010101	O	O	O	O	O	O	O	O	O	O	O	O	D	D	I	I	384	64	64
40	6'b101000	10101010010101010101010101010101	O	O	O	O	O	O	O	O	O	O	O	O	D	D	D	D	384	128	0
41	6'b101001	11111111010101010101010101010101	O	O	O	O	O	O	O	O	O	O	O	O	I	I	I	I	384	0	128
42	6'b101010	10100101010101010101010101010101	O	O	O	O	O	O	O	O	O	O	O	O	O	O	D	D	448	64	0
43	6'b101011	11110101010101010101010101010101	O	O	O	O	O	O	O	O	O	O	O	O	O	O	I	I	448	0	64
44	6'b101100	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

Due to fact that the RT1170 devices also support ECC, there is an additional bank array dedicated for storing the ECC cipher based on the FlexRAM configuration and ECC enable. This bank array contains additional 16 memory banks (8 kB each), so in total it is 128 kB. If the ECC is not enabled, it can be freely used to store normal data, as in the 128-kB OCRAM. If the ECC is enabled, an adequate portion of this array is dedicated to store the ECC cipher. For example, see row 9. If the ECC for both the OCRAM and TCM is disabled, the whole 128-kB memory space can be used to store data as in the OCRAM. If the TCM ECC is enabled, only 16 kBs can be used for the OCRAM data and the remaining 112 kBs are used to store the ECC cipher for the TCM (64 kBs for the DTC ECC cipher and 48 kBs for the ITCM ECC cipher). If the OCRAM ECC is enabled, 112 kBs of memory can be used for the free OCRAM data and the rest is used for the ECC cipher of the FlexRAM-configured OCRAM (64 kBs). If both the OCRAM ECC and the TCM ECC are enabled, then there is no free space for the OCRAM data and the whole memory space is dedicated for the ECC cipher proportional to the FlexRAM configuration (1:4).

Table 5. Static configuration of ECC cipher-storage-dedicated part of FlexRAM defined by fuses on RT1170

ECC dedicated OCRAM																								
	FUSE FlexRAM Configuration Value	Bank ECC																		ECC cipher partitioning [ECC **OCRAM**, ECC **TCM**] 0-disabled, 1-enabled					
																				DATA			ECC cipher		
																				[0, 0]	[0, 1]	[1, 0]	[1, 1]		
	0xC70 [5:0]	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31			OCRAM [kB]	OCRAM [kB]	OCRAM [kB]	OCRAM [kB]	DTCM [kB]	ITCM [kB]
0	6'b000000	D	D	D	D	I	I	I	I	D	D	D	D	I	I	I	I			128	0	0	0	64	64
1	6'b000001	D	D	D	D	D	D	I	I	D	D	D	D	I	I	I	I			128	0	0	0	80	48
2	6'b000010	D	D	D	D	D	D	I	I	D	D	D	D	D	D	I	I			128	0	0	0	96	32
3	6'b000011	D	D	D	D	D	D	I	I	D	D	D	D	D	D	D	D			128	0	0	0	112	16
4	6'b000100	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D			128	0	0	0	128	0
5	6'b000101	D	D	D	D	I	I	I	I	D	D	I	I	I	I	I	I			128	0	0	0	48	80
6	6'b000110	D	D	I	I	I	I	I	I	D	D	I	I	I	I	I	I			128	0	0	0	32	96
7	6'b000111	D	D	I	I	I	I	I	I	I	I	I	I	I	I	I	I			128	0	0	0	16	112
8	6'b001000	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I			128	0	0	0	0	128
9	6'b001001	O	O	D	D	D	D	I	I	I	I	D	D	D	D	I	I			128	16	112	16	64	48
10	6'b001010	O	O	D	D	D	D	I	I	D	D	D	D	D	D	I	I			128	16	112	16	80	32
11	6'b001011	O	O	D	D	D	D	I	I	D	D	D	D	D	D	D	D			128	16	112	16	96	16
12	6'b001100	O	O	D	D	D	D	D	D	D	D	D	D	D	D	D	D			128	16	112	16	112	0
13	6'b001101	O	O	D	D	D	D	I	I	I	I	D	D	I	I	I	I			128	16	112	16	48	64
14	6'b001110	O	O	D	D	I	I	D	D	I	I	I	I	I	I	I	I			128	16	112	16	32	80
15	6'b001111	O	O	D	D	I	I	I	I	I	I	I	I	I	I	I	I			128	16	112	16	16	96
16	6'b010000	O	O	I	I	I	I	I	I	I	I	I	I	I	I	I	I			128	16	112	16	0	112
17	6'b010001	O	O	O	O	D	D	D	D	I	I	I	I	D	D	I	I			128	32	96	32	48	48
18	6'b010010	O	O	O	O	D	D	D	D	I	I	D	D	D	D	I	I			128	32	96	32	64	32
19	6'b010011	O	O	O	O	D	D	I	I	D	D	D	D	D	D	D	D			128	32	96	32	80	16
20	6'b010100	O	O	O	O	D	D	D	D	D	D	D	D	D	D	D	D			128	32	96	32	96	0
21	6'b010101	O	O	O	O	D	D	I	I	I	I	D	D	I	I	I	I			128	32	96	32	32	64
22	6'b010110	O	O	O	O	D	D	I	I	I	I	I	I	I	I	I	I			128	32	96	32	16	80
23	6'b010111	O	O	O	O	I	I	I	I	I	I	I	I	I	I	I	I			128	32	96	32	0	96
24	6'b011000	O	O	O	O	O	O	D	D	I	I	D	D	D	D	I	I			128	48	80	48	48	32
25	6'b011001	O	O	O	O	O	O	D	D	I	I	D	D	D	D	D	D			128	48	80	48	64	16
26	6'b011010	O	O	O	O	O	O	D	D	D	D	D	D	D	D	D	D			128	48	80	48	80	0
27	6'b011011	O	O	O	O	O	O	D	D	I	I	D	D	I	I	I	I			128	48	80	48	32	48
28	6'b011100	O	O	O	O	O	O	D	D	I	I	I	I	I	I	I	I			128	48	80	48	16	64
29	6'b011101	O	O	O	O	O	O	I	I	I	I	I	I	I	I	I	I			128	48	80	48	0	80
30	6'b011110	O	O	O	O	O	O	O	O	D	D	I	I	D	D	I	I			128	64	64	64	32	32
31	6'b011111	O	O	O	O	O	O	O	O	D	D	I	I	D	D	D	D			128	64	64	64	48	16
32	6'b100000	O	O	O	O	O	O	O	O	D	D	D	D	D	D	D	D			128	64	64	64	64	0
33	6'b100001	O	O	O	O	O	O	O	O	D	D	I	I	I	I	I	I			128	64	64	64	16	48
34	6'b100010	O	O	O	O	O	O	O	O	I	I	I	I	I	I	I	I			128	64	64	64	0	64
35	6'b100011	O	O	O	O	O	O	O	O	O	O	D	D	I	I	D	D			128	80	48	80	32	16
36	6'b100100	O	O	O	O	O	O	O	O	O	O	D	D	D	D	D	D			128	80	48	80	48	0
37	6'b100101	O	O	O	O	O	O	O	O	O	O	D	D	I	I	I	I			128	80	48	80	16	32
38	6'b100110	O	O	O	O	O	O	O	O	O	O	I	I	I	I	I	I			128	80	48	80	0	48
39	6'b100111	O	O	O	O	O	O	O	O	O	O	O	O	D	D	I	I			128	96	32	96	16	16
40	6'b101000	O	O	O	O	O	O	O	O	O	O	O	O	D	D	D	D			128	96	32	96	32	0
41	6'b101001	O	O	O	O	O	O	O	O	O	O	O	O	I	I	I	I			128	96	32	96	0	32
42	6'b101010	O	O	O	O	O	O	O	O	O	O	O	O	O	O	D	D			128	112	16	112	16	0
43	6'b101011	O	O	O	O	O	O	O	O	O	O	O	O	O	O	I	I			128	112	16	112	0	16
44	6'b101100	O	O	O	O	O	O	O	O	O	O	O	O	O	O	O	O			128	128	0	128	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 (on RT1170, also IOMUXC_GPR_GPR18) register. This is possible when the FLEXRAM_BANK_CFG_SEL bit defined in the 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 and can be changed when the application runs.

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. 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 columns in Table 1, Table 2, Table 3, Table 4, and Table 5 show 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) before switching from the bank configuration defined by fuses, that means before the FLEXRAM_BANK_CFG_SEL bit is set to 1.

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. Consider at least 64 KB for the OCRAM configuration because the ROM code requires this portion of RAM for execution (stack/static data).

It is expected that:

- It is strongly recommended that the code that reconfigures the FlexRAM is executed from a different type of memory than the ITCM/DTCM/FlexRAM-dedicated OCRAM. For example, the code is executed from the FlexSPI serial NOR flash and the data are accessed from the SDRAM/non FlexRAM-dedicated OCRAM. Hence, it is expected that there is no access to the FlexRAM array during its reconfiguration. See Section 3.1.3, “Software implementation” for more details.
- It is expected that the code that reconfigures FlexRAM is executed before the first access to the stack, earlier in the application startup.
- Avoid enabling interrupts during FlexRAM reconfiguration. The enter/exit interrupt service routine requires an automatic push/pop from the stack, as well as a vector fetch from a dedicated

memory. This can potentially cause access to undefined memory space, which does not necessarily lead to problems during the reconfiguration, but later on in the application run.

- 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 6. 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

However, it still represents a continuous address space in the device memory map (see Table 5).

Table 7. 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 64 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 (see configurations 7, 10, 11 in Table 1). Avoid access to the empty RAM space configured by the corresponding FlexRAM configuration or use the recommended way, which defines the size of the TCM as a power-of-two number and reflect that by writing into the appropriate field in IOMUXC_GPR_GPR14->CM7_CFGxTCMSZ (it will update the CM7_xTCMCR accordingly).

If the requested ITCM/DTCM size is 0 Bytes, disable the corresponding TCM in IOMUXC_GPR_GPR16->INIT_xTCM_EN before configuring the size to 0 Bytes in IOMUXC_GPR_GPR14->CM7_CFGxTCMSZ.

If a dedicated RT device also supports the ECC, then there is an additional memory bank array dedicated for this feature. On RT1170, there is a 128-kB ECC-dedicated array. Each 32-kB bank within the configurable part of the FlexRAM (explained above) has its dedicated 8-kB ECC bank. For example, bank0 of the configurable FlexRAM array has its ECC cipher backup storage in bank16 of the ECC-dedicated FlexRAM (see Table 4 versus Table 5). This array can be freely used as a normal OCRAM data storage until the ECC function is enabled. If the ECC function is enabled, then a proportional size (1/4 on RT1170) of the configurable FlexRAM array has its portion in the ECC cipher storage. For example, a 64-kB OCRAM, configured via a configurable FlexRAM array, has its 16-kB ECC cipher storage if the OCRAM ECC is enabled. The proportional part of the configurable FlexRAM array, which does not have the ECC enabled, can be freely used for normal OCRAM data. For example, if there is 128 kBs of ITCM and 128kBs of DTCM and no ECC TCM is enabled, then there are 64 kBs freely available for normal OCRAM data.

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 (see FlexRAM config MUXs block in Figure 1) 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 (a 64-bit I-TCM bus or two 32-bit buses, D0-TCM and D1-TCM, with interleaved addressing word by word) 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 controller is connected to interconnect bus fabric (NIC) slave port via the 64-bit system AXI bus. It runs at a bus fabric clock frequency. Due to this fact, lower access time can be expected when compared to the access time of TCM interfaces. The OCRAM controller 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:

- on RT10xx:
  - OCRAM_CTRL[3:0] in general-purpose register IOMUX_GPR_GPR3
- on RT117x:

- OCRAM_WRADDR_PIPELINE_EN, OCRAM_WRDATA_PIPELINE_EN, OCRAM_RADDR_PIPELINE_EN, or OCRAM_RDATA_WAIT_EN in FLEXRAM_CTRL

After performing changes in the OCRAM control field, it is recommended to wait until the corresponded OCRAM status bit changes from 1 to 0 (1 means that the configuration is changed, but not applied yet):

- on RT10xx: OCRAM_STATUS[19:16]
- on RT117x: FLEXRAM_OCRAM_PIPELINE_STATUS

NOTE

It is expected that the code which is changing and then checking the corresponding OCRAM status bit cannot be executed from the OCRAM.

2.2.3. ECC memory controller

Except for the memory controllers which handle the conversion from the TCM/AXI interface signal to the memory array signal, some of the 4-digit RT devices (such as RT117x) also support the ECC. The ECC controller on RT117x includes support for the Single-bit Error Correction (SEC) and Dual-bit Error Detection (DED). It can be done for both TCM memories, as well as the OCRAM memory within the FlexRAM. Except for the 512-kB configurable memory array, there is a 128-kB memory array available for ECC purposes. If the ECC is enabled, it is dedicated for the ECC cipher storage by the ECC controller. Any bus masters' writes to this memory location are ignored and any reads from it return zeros. However, if the ECC is not required in the application, it can be disabled and this memory array can be freely used for normal OCRAM data storage (see Table 5 for more details).

The ECC within the FlexRAM can recognize single-bit or multi-bit errors (see Figure 2) within:

- The 64-bit data (located in the configurable FlexRAM array – OCRAM or ITCM), utilizing 8-bit ECC code (located in its adequate ECC FlexRAM array). This represents a ratio of 1:8. It means that the ECC requires only a 64-kB ECC cipher code footprint to back up 512 kBs. For example, if the whole FlexRAM is configured for the ITCM, then only 64 kBs are used by the ECC.
- The 32-bit data (located in the configurable FlexRAM array – D0TCM or D1TCM), utilizing the 8-bit ECC code (located in its adequate ECC FlexRAM array). This represents a ratio of 1:4. It means that the ECC requires a 128-kB ECC cipher code footprint to back up 512 kBs. For example, if the whole FlexRAM is configured for the DTCM, then all 128 kBs are used.

The ECC controller evaluates the ECC code on every access (read/write). Any write access is split into 64-bit/32-bit data written into the configurable 512-kB FlexRAM array and the 8-bit encoded ECC code is written into the non-configurable 128-kB FlexRAM array (ECC cipher). Any read access reads the 64-bit/32-bit data from the 512-kB configurable FlexRAM array and it reads the dedicated 8-bit ECC code. If the ECC check passes, it returns the data read on a dedicated bus. If the ECC check fails, it corrects the error (if it is a single bit) and initiates the ECC error detection mechanism based on the ECC configuration. For example, it can generate and handle single-bit or multi-bit interrupts where more information regarding the ECC error can be identified.

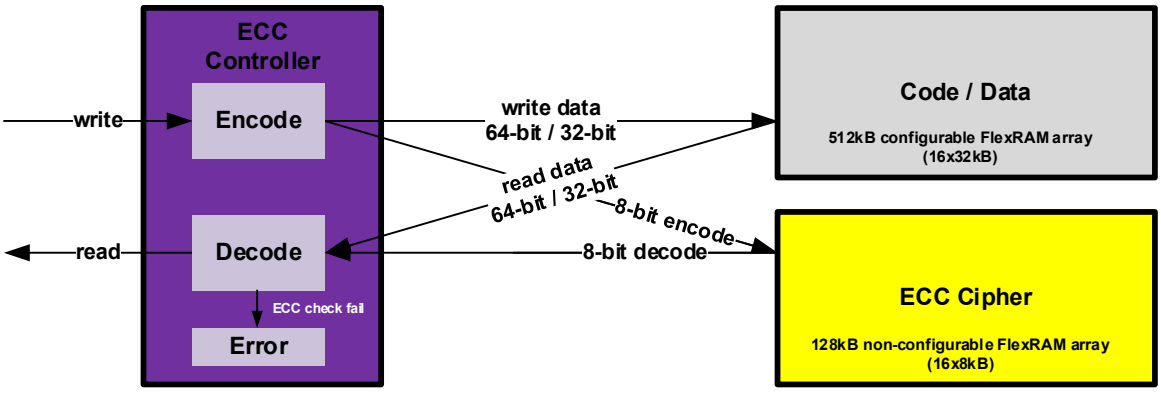

Figure 2. ECC memory controller functionality

2.2.3.1. ECC single-bit error correction

The RT117x devices support single-bit error correction. If the dedicated-memory (TCM/OCRAM) ECC is enabled, the single-bit ECC correction is available. Any single-bit error results in its correction. It can be also detected by software if the dedicated single-bit ECC error interrupt is enabled. For more details, see Section 2.5, "FlexRAM interrupt". If it is enabled, then it is possible to recognize the following:

- The adress offset on which the error happened (64-bit (OCRAM/ITCM) or 32-bit (D0TCM/D1TCM) aligned). It can be obtained by reading the xxxx_ECC_SINGLE_ERROR_ADDR registers.

NOTE

On D0TCM/D1TCM, consider their interleaved address meaning. For example, if the error happened on real address 0x20000020, the D0TCM address offset should be 0x00000010.

- Uncorrected data (64-bit (OCRAM/ITCM) or 32-bit (D0TCM/D1TCM) size). It can be obtained by reading the xxxx_ECC_SINGLE_ERROR_DATA registers.
- The bit position of a single-bit ECC error can be obtained as follows:
  - Using the look-up table and reading the 8-bit ECC syndrome field in the ECC error information registers – xxxx_ECC_SINGLE_ERROR_INFO.
- On the ITCM/DTCM (also available in the xxxx_ECC_SINGLE_ERROR_INFO register), you can obtain the following:
  - Privilege level of access.
  - Which master resource generates the access (instruction fetch/data access/debugger, and so on).
  - Size:
  - For ITCM, it is fixed to 64 bits.
  - For D0TCM/D1TCM, it is fixed to 32 bits.
  - Access type:
  - Read access.
  - Write access.

NOTE

The xxxx chain in the register description above represents OCRAM, ITCM, D0TCM, or D1TCM.

Table 8. ECC syndrome look-up tables for 64-bit (OCRAM/ITCM) and 32-bit (D0TCM/D1TCM) data

64-bit LUT (OCRAM / ITCM)				32-bit LUT (D0TCM / D1TCM)	
Syndrome value	Error bit	Syndrome value	Error bit	Syndrome value	Error bit
0xc1	0	0x46	32	0x61	0
0x43	1	0x91	33	0x51	1
0x9e	2	0x86	34	0x19	2
0x83	3	0x61	35	0x45	3
0x15	4	0x49	36	0x43	4
0x4c	5	0x98	37	0x31	5
0x4a	6	0x89	38	0x29	6
0x8c	7	0x68	39	0x13	7
0x31	8	0x32	40	0x62	8
0x1c	9	0x34	41	0x52	9
0xa2	10	0x07	42	0x4a	10
0xe0	11	0xc8	43	0x46	11
0x51	12	0x92	44	0x32	12
0x2c	13	0xa8	45	0x2a	13
0xc2	14	0xa7	46	0x23	14
0xd0	15	0x54	47	0x1a	15
0x19	16	0xa1	48	0x2c	16
0x1a	17	0xd9	49	0x64	17
0x26	18	0x25	50	0x26	18
0xea	19	0xf8	51	0x25	19
0x29	20	0x0e	52	0x34	20
0x94	21	0x0b	53	0x16	21
0x16	22	0x8a	54	0x15	22
0x64	23	0x2a	55	0x54	23
0x37	24	0x52	56	0x0b	24
0xa4	25	0x45	57	0x58	25
0x0d	26	0x13	58	0x1c	26
0xc4	27	0x85	59	0x4c	27
0x75	28	0x62	60	0x38	28
0x38	29	0x70	61	0x0e	29
0x4f	30	0x23	62	0x0d	30
0x58	31	0xb0	63	0x49	31

2.2.3.2. ECC multi-bit error detection

The RT117x devices support also multi-bit error detection. This process does not allow to correct the error, but it can detect an error in the data (more-than-one-bit error occurred within 64-bit data) before that data is evaluated by the application code. This helps to avoid a failure or an incorrect functionality of the application. If the dedicated-memory (TCM/OCRAM) ECC is enabled, then the multi-bit ECC detection is available. Any multi-bit error can be detected by software if the dedicated multi-bit ECC

error interrupt is enabled. for more details, see Section 2.5, “FlexRAM interrupt”. If it is enabled, then it is possible to recognize the following:

- The adress offset on which the multi-bit error happened (64-bit (OCRAM/ITCM) or 32-bit (D0TCM/D1TCM) aligned). It can be obtained by reading the xxxx_ECC_MULTI_ERROR_ADDR registers.
- The data (64-bit (OCRAM/ITCM) or 32-bit (D0TCM/D1TCM) size) in which the multi-bit error happened. It can be obtained by reading the xxxx_ECC_MULTI_ERROR_DATA registers.
- On the ITCM/DTCM (also available in the xxxx_ECC_MULTI_ERROR_INFO register), you can obtain the following:
  - Privilege level of access.
  - Which master resource generated the access (instruction fetch/data access/debugger, and so on).
  - Size:
  - For ITCM, it is fixed to 64 bits.
  - For D0TCM/D1TCM, it is fixed to 32 bits.
  - Access type:
  - Read access.
  - Write access.

NOTE

The information of the ECC syndrome does not matter when a multi-bit error happens.

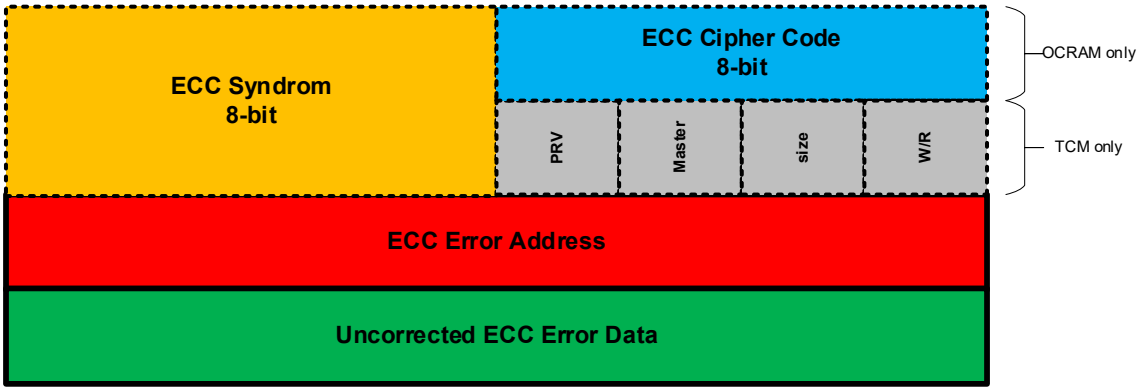

Figure 3. Information available on ECC error identification

NOTE

The RT10xx devices, such as RT101x, RT1015, RT102x, RT105x, and RT106x, do not include the ECC support.

Table 9. FlexRAM-module-related clocks and their gates

Device	Clock Name	Clock Source				peripheral interface / registers access clock			Clock gate
		Name	Derived source clock name	Default frequency	Maximum frequency	Name	Default frequency	Maximum frequency	
RT10xx	flexram_clk	module clock	core_clk	*396 MHz*	*600/528/400 MHz*	ipg_clk_root	*96 MHz*	*150 MHz*	CCM_CCGR3[CG9] - flexram_clk_en
	ocram_clk	ocram_exsc_aclk_exsc	axi_clk	*99 MHz*	*150/132/100 MHz*	ipg_clk_root	*96 MHz*	*150 MHz*	CCM_CCGR3[CG14]- ocram_clk_en
RT117x	flexram_clk	module clock	core_clk	*400 MHz*	*996 MHz*	ipg_clk_root	*200 MHz*	*240 MHz*	LPCG0 - clk_enable_flexram
	ocram_clk	ocram_exsc_aclk_exsc	axi_clk	*200 MHz*	*240 MHz*	ipg_clk_root	*200 MHz*	*240 MHz*	LPCG25 - clk_enable_ocram

Table 10. Dedicated RT device power domain bank assignment

RT device	Power domain bank assignment		
	PDRET	PDRAM0	PDRAM1
RT1060/RT1064	-	Bank0-Bank15	-
RT1050	Bank0	Bank1-Bank7	Bank8-Bank15
RT1020	Bank0-Bank7	-	-
RT1015	Bank0-Bank3	-	-
RT1010	Bank0-Bank3	-	-
RT1170	All banks share the same power domain.		

2.3. FlexRAM module-related clocks and clock gates

Table 11 summarizes all clocks related to the FlexRAM module:

Table 11. FlexRAM module related clocks and their gates

Device	Clock Name	Clock Source				peripheral interface / registers access clock			Clock gate
		Name	Derived source clock name	Default frequency	Maximum frequency	Name	Default frequency	Maximum frequency	
RT10xx	flexram_clk	module clock	core_clk	*396 MHz*	*600/528/400 MHz*	ipg_clk_root	*96 MHz*	*150 MHz*	CCM_CCGR3[CG9] - flexram_clk_en
	ocram_clk	ocram_exsc_aclk_exsc	axi_clk	*99 MHz*	*150/132/100 MHz*	ipg_clk_root	*96 MHz*	*150 MHz*	CCM_CCGR3[CG14]- ocram_clk_en
RT117x	flexram_clk	module clock	core_clk	*400 MHz*	*996 MHz*	ipg_clk_root	*200 MHz*	*240 MHz*	LPCG0 - clk_enable_flexram
	ocram_clk	ocram_exsc_aclk_exsc	axi_clk	*200 MHz*	*240 MHz*	ipg_clk_root	*200 MHz*	*240 MHz*	LPCG25 - clk_enable_ocram

The FlexRAM clock (flexram_clk) is the main clock the module is clocked by and is derived from the Arm core clock (core_clk). 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_clk) is derived from axi_clk and further divided accordingly. 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.

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, FlexSPI module, SEMC, and so on). It runs at a lower frequency than the Arm Cortex-M7 core frequency. The OCRAM runs at the same frequency as the interconnect bus fabric. The NIC master port versus the slave port clock ratio is chip-specific. On RT10xx devices, it is an integer number and it is fixed to 1:4. It means that if the Cortex-M7 clock is 600 MHz, then the slave port bus clock is limited to 150 MHz. On RT117x, it is an asynchronous conversion and its maximum frequency is from 996 MHz to 240 MHz.

2.4. FlexRAM power domains

The FlexRAM bank array can be partitioned to up to three different power sub-domains:

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

Table 7 shows the state of the dedicated power domain in different low-power modes.

Table 12. 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.

Table 8 summarizes the power domain bank assignment for the corresponding RT devices.

Table 13. Dedicated RT device power domain bank assignment

RT device	Power domain bank assignment		
	PDRET	PDRAM0	PDRAM1
RT1060/RT1064	-	Bank0-Bank15	-
RT1050	Bank0	Bank1-Bank7	Bank8-Bank15
RT1020	Bank0-Bank7	-	-
RT1010	Bank0-Bank3	-	-

NOTE

The RT117x devices utilize just one power domain for all banks of the FlexRAM. It uses an isolation power switch to switch the FlexRAM off.

2.4.1. PDRET power domain

This power domain is always on. It means that 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

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 FlexRAM banks assigned to this domain keep their content even when 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 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 critical data that are considered to be retained even in the suspend mode must be placed into the FlexRAM banks assigned to the PDRET power domain (if available).

2.5. FlexRAM interrupt

The FlexRAM module can generate interrupt requests based on two different events:

- Not allocated address access (address out of range).
- Magic address read/write access hit (not supported on all RT devices, see the corresponding reference manual for details).
- ECC error detection on read access.
- Partial write is launched and ECC is enabled.

The interrupt request signal is generated based on its configuration in the interrupt enable register (INT_SIG_EN). Each memory type (OCRAM/DTCM/ITCM) has its dedicated bit in this register for enabling the interrupt:

- xxxx_ERR_SIG_EN: when set, it enables the generation of the xxxx memory (OCRAM/DTCM/ITCM) out of the address-range IRQ.
- xxxx_MAM_SIG_EN: when set, it enables the generation of the xxxx memory

(OCRAM/DTCM/ITCM) magic address-access IRQ.

- xxxx_ERRS_INT_SIG_EN: when set, it enables the generation of the xxxx memory (OCRAM/DTCM/ITCM) single-ECC-error IRQ.
- xxxx_ERRM_INT_SIG_EN: when set, it enables the generation of the xxxx memory (OCRAM/DTCM/ITCM) multiple-ECC-error IRQ.
- xxxx_PARTIAL_WR_INT_SIG_EN: when set, it enables the generation of the xxxx memory (OCRAM/DTCM/ITCM) partial-write IRQ.

All these error signals are ORed (if enabled) and they generate one interrupt request to the NVIC with number 38 on the RT10xx devices and 50 on the RT117x devices.

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

- xxxx_ERR_STATUS: when set, it shows that the out-of-address-range IRQ happened on the dedicated xxxx memory (OCRAM/D0TCMD0TCM/D1TCM/ITCM).
- xxxx_MAM_STATUS: when set, it shows that the magic-address-access IRQ happened on the dedicated xxxx memory (OCRAM/D0TCMD0TCM/D1TCM/ITCM).
- xxxx_ECC_ERRS_STATUS: when set, it shows that the single-ECC-error IRQ happened on the dedicated xxxx memory (OCRAM/D0TCMD0TCM/D1TCM/ITCM).
- xxxx_ECC_ERRM_STATUS: when set, it shows that the multiple-ECC-error IRQ happened on the dedicated xxxx memory (OCRAM/D0TCMD0TCM/D1TCM/ITCM).
- xxxx_PARTIAL_WR_INT_S: when set, it shows that the partial-write IRQ happened on the dedicated xxxx memory (OCRAM/D0TCMD0TCM/D1TCM/ITCM).

Each status bit can be cleared by writing a log. 1 into it.

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

- xxxx_ERR_STAT_EN: when set, the out-of-address-range interrupt status is enabled on the dedicated xxxx memory (OCRAM/D0TCMD0TCM/D1TCM/ITCM).
- xxxx_MAM_STAT_EN: when set, the magic-address-access interrupt status is enabled on the dedicated xxxx memory (OCRAM/D0TCMD0TCM/D1TCM/ITCM).
- xxxx_ECC_ERRS_INT_EN: when set, the single-ECC-error interrupt status is enabled on the dedicated xxxx memory (OCRAM/D0TCMD0TCM/D1TCM/ITCM).
- xxxx_ECC_ERRM_INT_EN: when set, the multiple-ECC-error interrupt status is enabled on the dedicated xxxx memory (OCRAM/D0TCMD0TCM/D1TCM/ITCM).
- xxxx_PARTIAL_WR_INT_S_EN: when set, the partial-write interrupt status is enabled on the dedicated xxxx memory (OCRAM/D0TCMD0TCM/D1TCM/ITCM).

Some of iMX RT devices can also generate a FlexRAM dedicated interrupt when there is a specific software defined address access hit.

The address and access type are user-defined in the magic address registers. A dedicated memory type (ITCM/DTCM/OCRAM) magic address register contains two fields; one for the magic address definition and one for the selection of the access type (read or write):

- OCRAM_MAGIC_ADDR:
  - The OCRAM_MAGIC_ADDR field represents a 16-bit address within the OCRAM memory mapped address space.
  - OCRAM_WR_RD_SEL:
    - 0 – sets the interrupt generation for read access.
    - 1 – sets the interrupt generation for write access.
- DTCM_MAGIC_ADDR:
  - The DTCM _MAGIC_ADDR field represents a 16-bit address within the DTCM memory mapped address space.
  - DTCM _WR_RD_SEL:
    - 0 – sets the interrupt generation for read access.
    - 1 – sets the interrupt generation for write access.
- ITCM_MAGIC_ADDR:
  - The ITCM _MAGIC_ADDR field represents a 16-bit address within the ITCM memory mapped address space.
  - ITCM _WR_RD_SEL:
    - 0 – sets the interrupt generation for read access.
    - 1 – sets the interrupt generation for write access.

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 on iMX RT1050 devices

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 RT10xx devices do not have embedded flash (RT1064 includes serial SPI flash as SIP). 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) should be placed in the external memory. The data that is accessible only by the core (stack, static data, and so on) 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) should be placed in 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 14. Requirements

CODE	DATA
120 KB	889 KB

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 that contain the raw data from the image sensor (320x240 in the RGB 5-6-5 format).

NOTE

These buffers are considered critical data from the processing point of view. That kind of data are usually stored in the external SDRAM memory. There is a significantly faster access time to the OCRAM than to the SDRAM and the cost efficiency from the customer point of view is considered. It is recommended to fully utilize the on-chip memory (if possible).

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.

NOTE

The OCRAM is a cacheable memory. Care must be taken when both masters access the same memory region which is cacheable. It is recommended to disable the dedicated cache region in such case. If the cache is enabled, the software is responsible to ensure the synchronization of access for both masters.

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/knowledge/learn/programming/mastering-stack-and-heap-for-system-reliability), the stack size is estimated to be 4 KB (10 % addition included). It is recommended to place the stack into the DTCM because it is accessed only by the core.

3.1.2.3. Total memory footprint (i.MX RT1050)

Considering the above assumptions when using the i.MX RT1050 device, it can be summarized:

Table 15. 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 Table 11). According to the Arm TCM size configuration specification, the size of TCM can be the number of power of two (32 KB, 64 KB, 128 KB, 256 KB, or 512 KB).

Table 16. 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 considered to think about moving 15 KB of static data from the DTCM to the OCRAM (or ITCM). It depends on:

- OCRAM: the DMA channels loading the OCRAM controller write (CSI)/read (LCD) via the system AXI bus.
- ITCM: access to static data is not required in low-power modes.
- The effect on the overall performance.

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 5). If the

approach cannot be applied due to performance degradation, then the solution of moving data to the ITCM can be applied.

Table 12 shows the case when the application static data are moved from the DTCM memory to the OCRAM memory, considering no significant effect on performance.

Table 17. Application memory section requirements when application static data moved to OCRAM

	External memory	ITCM	External memory	DTCM	OCRAM	In TOTAL memory / banks
Memory size requirements	74	46	450	139-15	300+15	485
Number of FlexRAM banks required		2		4	10	16
Total size of memory		64		128	320	512

Both memory re-configurations (Table 12 and Table 13) fit into the i.MX RT1050 FlexRAM and all 16 banks are used.

Table 18. Application memory section requirements when application static data moved to ITCM

	External memory	ITCM	External memory	DTCM	OCRAM	In TOTAL memory / banks
Memory size requirements	74	46+15	450	139-15	300	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 the i.MX RT1050 FlexRAM power distribution (power sub-domains) 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 suspend 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.

Table 14 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 19. Application example of FlexRAM banks configuration

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

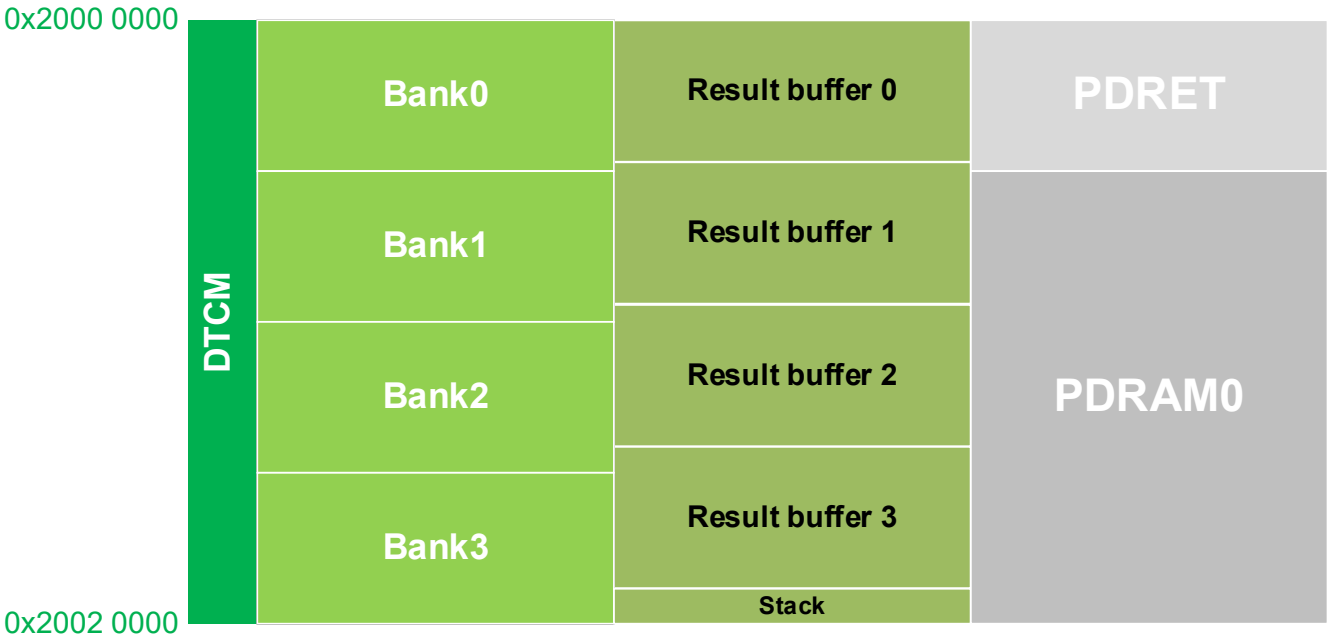

Figure 4. DTCM memory addresses, configuration, content, and appropriate power domain

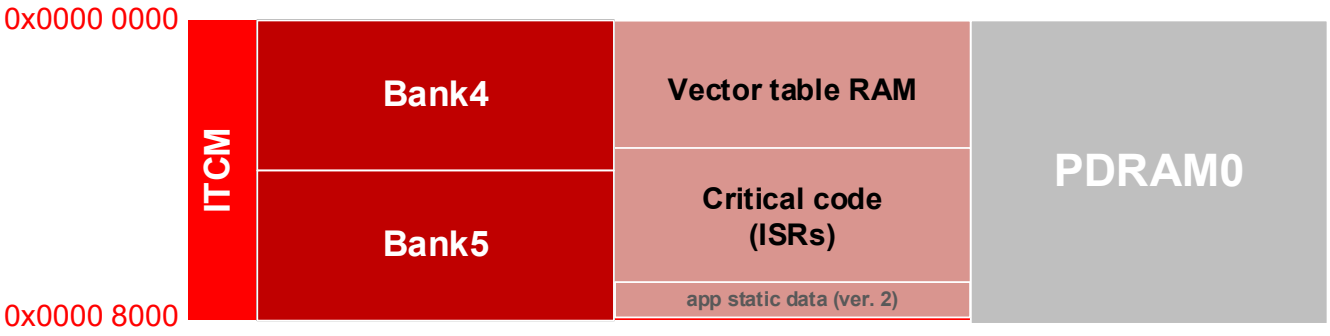

Figure 5. ITCM memory addresses, configuration, content, and appropriate power domain

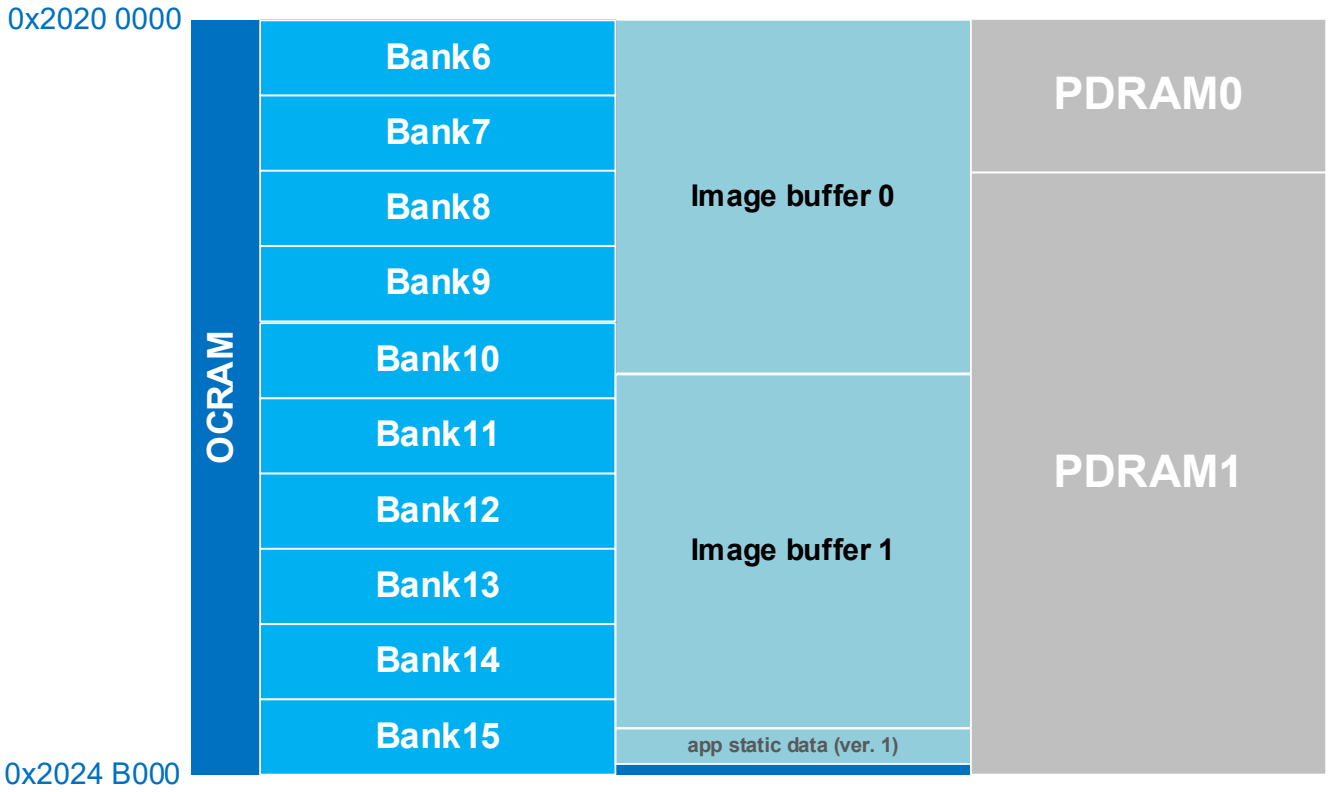

Figure 6. OCRAM memory addresses, configuration, content, and appropriate power domain

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.

The final FlexRAM configuration value will be 0x55555FAA:

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			
01010101010101010101111110101010	D	D	D	D	I	I	O	O	O	O	O	O	O	O	O	O	320	128	64

NOTE

Consider the release (no debug) target configuration of the application project when compiling.

3.1.3. Software implementation

The best place for FlexRAM configuration in the application software is the start-up code before the execution of static variable initialization, RW code initialization (if considered in TCM/OCRAM), vector table relocation, any stack access (PUSH/POP), and so on. The start-up code shall be executed from a different kind of memory; for example, external serial flash. During the FlexRAM configuration code execution, the ITCM/DTCM/OCRAM memory cannot be accessed and the interrupts shall be disabled (to avoid stacking/unstaking).

NOTE

It is possible to process the FlexRAM reconfiguration in the application run-time; for example, code located in the external serial flash (FlexSPI) or all data including the stack/heap in the external SDRAM (SEMC). Make sure to avoid data overlapping, access mismatch, and so on after the FlexRAM release. It is not recommended to reconfigure the FlexRAM memory using DCD (Device Configuration Data) due to a potential conflict with the ROM code memory allocation.

3.1.3.1. Considering Cortex-M7 TCM size limitations

The TCM-size Arm specification considers the size of TCM in a power-of-two number (in case of iMXRTxxxx FlexRAM, it is 0 k/32 k/64 k/128 k/256 k/512 k). The application use case mentioned above considers these limitations. In this case, the application software after the power-on reset event performs these steps:

1. Ensure that there is no access to any of the banks in the FlexRAM array during its reconfiguration code execution:
   - Execute the code from memory outside the FlexRAM array (OCRAM not included in FlexRAM, QSPI, SDRAM, and so on).
   - Disable the interrupts.
2. Configure the FlexRAM bank array according to application needs:
   - Use the FLEXRAM_BANK_CFG field in the IOMUXC_GPR_GPR17 register.
3. Switch from eFuse-defined FlexRAM configuration to user-defined FlexRAM configuration:
   - Use the FLEXRAM_BANK_CFG_SEL field in the IOMUXC_GPR_GPR16 register.
4. If there is a request to have 0 kB of any TCM memory, disable the corresponding TCM before setting the size to 0 kB. If not, omit this step:
   - Use the INT_xTCM_EN fields in the IOMUXC_GPR_GPR16 register to disable the corresponding xTCM memory before configuring it to 0 kB.
5. Configure the dedicated TCM memory size to the application-required size in a power-of-two number according to the Arm specification.

- Use the CM7_CFGxTCMSZ fields in the IOMUXC_GPR_GPR14 registers to configure the size to 0 k/32 k/64 k/128 k/256 k/512 k.

NOTE

It is possible to configure the TCM size into a lower size (4 k/8 k/16 k). The FlexRAM bank has a higher size (32 k). In such case, the FlexRAM is not utilized effectively because there is a part of available memory not accessible by the core. A bus fault is generated when accessing (read/write) unallocated memory space. The Cortex-M7 base register CM7_xTCMCR is updated according to this setting.

6. Run/Jump into the full memory-defined application code.

```
__iomux_gpr14_adr   EQU    0x400AC038
__iomux_gpr16_adr   EQU    0x400AC040
__iomux_gpr17_adr   EQU    0x400AC044
__flexram_bank_cfg  EQU    0x55555FAA        ; 320k OCRAM, 128k DTCM, 64k ITCM
__flexram_itcm_size EQU    0x7               ; 64kB
__flexram_dtcm_size EQU    0x8               ; 128k
__reconfig_init_sp          EQU    0x20001FFF        ; stack placed at beginning of DTCM (not at top)
                                                     ; size of stack 8k

Reset_Handler
      CPSID  I          ; Mask interrupts
      LDR    R0, =0xE000ED08
      LDR    R1, =__vector_table
      STR    R1, [R0]
      LDR    R2, [R1]
      MSR    MSP, R2
#ifdef FLEXRAM_CFG_ENABLE
; NOTE: Initial stack pointer (SP) presented in vector table must be set to correct value considering the following FlexRAM re-configuration
; NOTE: If the value presented in vector table is not considering correct stack it then the stack pointer (SP) must be re-configured here, e.g.:
      DSB
      ISB
      LDR    R0,=__reconfig_init_sp     ; load initial value of stack pointer into R0
      MSR    MSP, R0                    ; re-initialize stack pointer by new value
      LDR    R0,=__iomux_gpr17_adr      ; load IOMUXC_GPR17 register address to R0
      MOV32  R1,__flexram_bank_cfg      ; move FlexRAM configuration value to R1
      STR    R1,[R0]
      DSB
      ISB
      LDR    R0,=__iomux_gpr16_adr      ; load IOMUXC_GPR16 register address to R0
      LDR    R1,[R0]                    ; load IOMUXC_GPR16 register value to R1
      ORR    R1, R1, #4                 ; set corresponding FLEXRAM_BANK_CFG_SEL bit
      STR    R1,[R0]                    ; store the value to IOMUXC_GPR16 (user defined FlexRAM cfg enabled)
      DSB
      ISB
#ifdef FLEXRAM_ITCM_ZERO_SIZE
      LDR    R0,=__iomux_gpr16_adr      ; load IOMUXC_GPR16 register address to R0
      LDR    R1,[R0]                    ; load IOMUXC_GPR16 register value to R1
      AND    R1, R1, #0xFFFFFFFE        ; clear corresponding INIT_ITCM_EN bit
      STR    R1,[R0]                    ; store the value to IOMUXC_GPR16 (disable ITCM)
      DSB
      ISB
#endif
#ifdef FLEXRAM_DTCM_ZERO_SIZE
      LDR    R0,=__iomux_gpr16_adr      ; load IOMUXC_GPR16 register address to R0
      LDR    R1,[R0]                    ; load IOMUXC_GPR16 register value to R1
      AND    R1, R1, #0xFFFFFFFD        ; clear corresponding INIT_DTCM_EN bit
      STR    R1,[R0]                    ; store the value to IOMUXC_GPR16 (disable DTCM)
      DSB
      ISB
#endif
#ifdef FLEXRAM_XTCM_POWER_OF_TWO_SIZE
      LDR    R0,=__iomux_gpr14_adr      ; load IOMUXC_GPR14 register address to R0
      LDR    R1,[R0]                    ; load IOMUXC_GPR14 register value to R1
      MOVT   R1, #0x0000                ; clear upper halfword of IOMUXC_GPR14 register
      MOV    R2, #__flexram_itcm_size
      MOV    R3, #__flexram_dtcm_size
      LSL    R2, R2, #16
      LSL    R3, R3, #20
      ORR    R1, R2, R3
      STR    R1,[R0]                    ; store the value to IOMUXC_GPR14
      DSB
      ISB
#endif
#endif
      LDR    R0, =SystemInit
      BLX    R0
      CPSIE  I          ; Unmask interrupts
      LDR    R0, =__iar_program_start
      BX     R0
```

1.
2.
3.
4.
5.
6.

Figure 7. Example of the assembly code following previous rules

3.1.3.2. Ignoring Cortex-M7 TCM size limitations

This approach considers that the application software is aware of unallocated address space and avoids access to that space. The advantage of this approach is that the TCM memory does not need to consider the size in a power-of-two number.

Considerations:

- It is required to keep the default TCM size setting in the CM7_CFGxTCMSZ fields in the IOMUXC_GPR_GPR14 register (it considers the value of the whole FlexRAM size).
- The software must avoid accessing unallocated address space.

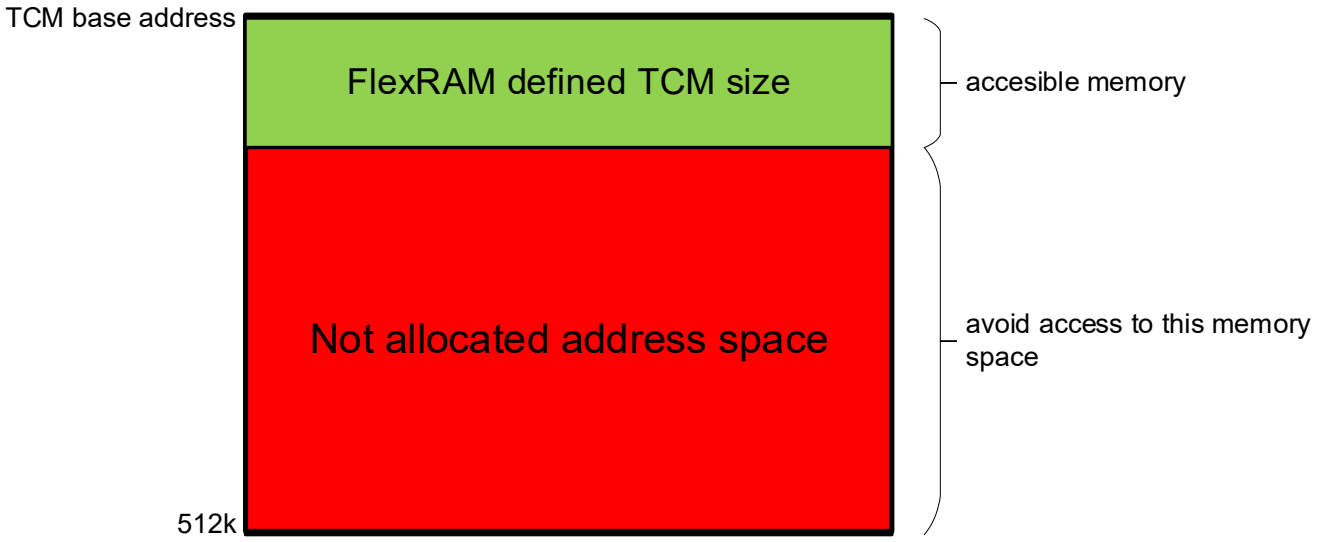

Figure 8. Example of accessible and non-accessible address space

The software implementation can be similar to the previous case, excluding point 5 and the power-of-two size consideration.

4. Revision history

Table summarizes the changes done to this document since the initial release.

Table 20. Revision history

Revision number	Date	Substantive changes
0	10/2017	Initial release
1	08/2018	Added Section 3.1.3, “Software implementation”. Added support for additional RT10xx devices.
2	09/2019	Added RT1010-based features.
3	01/2021	Added RT1170-based features.

How to Reach Us:

Home Page:
www.nxp.com

Web Support:
www.nxp.com/support

Document Number: AN12077
Rev. 3
01/2021