AN12564

Implement RWW on i.MX RT Series

Rev. 0 — Septeber 2019 Application Note

Contents

1 Introduction

Usually, some applications need to store the data to flash, and run code simultaneously that requires Read While Write(RWW). The i.MX RT has high-performance, supports rich peripherals, and can interface with many memory devices including Quad SPI flash, hyper flash, serial NAND, and parallel NAND flash. It also provides more options to meet customer requirements. This application note intends to introduce how to implement RWW requirement on i.MXRT series.

2 Overview

To implement RWW, the flash support feature is required in nature, but most flash do not have this feature. One way to implement RWW is based on common flash device, which is described in this document.

i.MX RT is a flash-less microcontroller, Quad SPI flash is a common selection as firmware storage, also some application may save data in this flash, but it only has one flash bank and interface.One limitation is it do not respond to any further command until flash idle, that means it must allocate the flash write function to other memory(except currently writing flash, generally allocate to internal SRAM), and disable interrupt to avoid any unexpected flash access until writing operation complete. As write operation spend more time in most flash devices, for example, it takes 0.4 mS to complete one page program in most flash part, and take more time on sector erase. This is unacceptable to disable interrupt lasting a long time, especially some interrupt require high-timely response, RWW is helpful in this condition.

There are two solutions to implement RWW on i.MX RT Series, one is to choose the flash supported RWW feature; the other is to use multiple flash to implement this function. Following chapters describe the RWW implementation in detail.

3 RWW implement

This chapter explains, how to implement RWW based on FlexSPI interface, which can interface with Quad SPI flash, octal flash and hyper flash, this is also a common usage on i.MX RT Series.

3.1 Multiple flash implementing RWW

i.MX RT supports up to 4 flash device on the same FlexSPI interface by different chip select signals. It also supports bus arbitration which is very helpful in RWW application. See Figure 1 for four flash device connection.

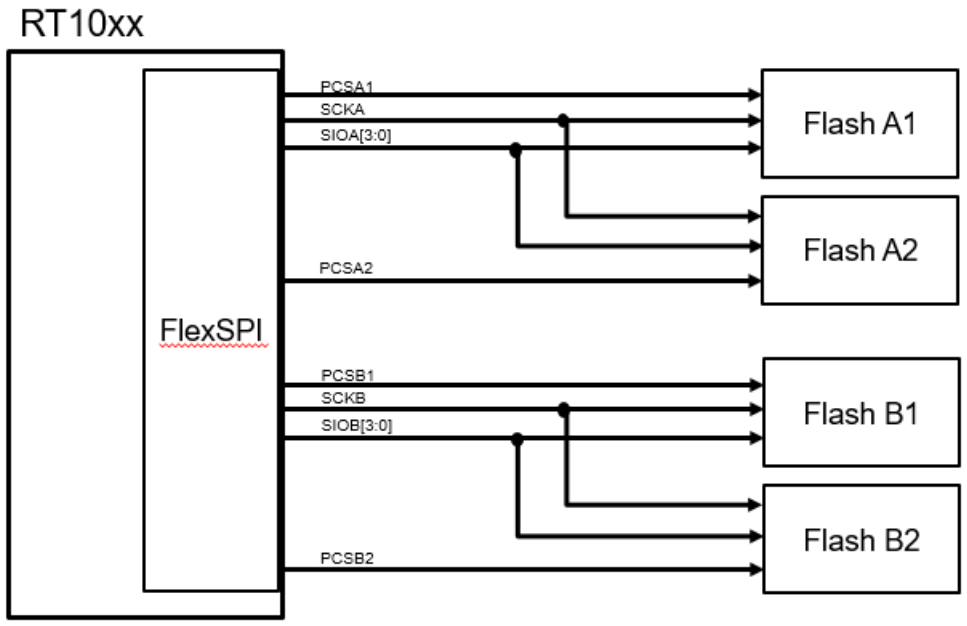

Figure 1. Flash connection diagram with four devices

Generally users can connect Flash A1 and Flash A2 to port A of FlexSPI to implement RWW function. OR, connect Flash A1 and Flash B1 to port A and port B respectively. Flash A1 is used to code storage and running, the other flash is for data storage. If connected to the same port, it must use the same flash device, and keep the same flash timings. The advantage of connecting is, the two ports may set the different timings for two flash devices by DLLACR or DLLBCR registers.

How to implement RWW by connecting two flash devices is described below.

3.1.1 Hardware design

Figure 2 shows how to connect two QSPI flash on the same port, and take different PCS for these two QSPI flash:

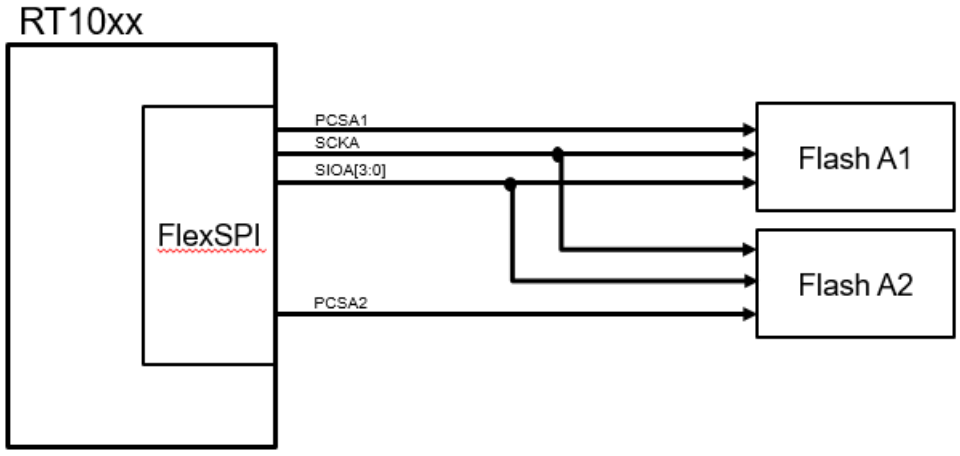

Figure 2. Example of connect the two QSPI flash

Figure 3shows the pins assignment:

QSPI Flash	IO functions	ALT	Pins assignment	Comments
Flash A1	flexspi.A_SS0_B	1	GPIO_SD_B1_06	no special requirement to FlexSPI_A_SS1 pin assignemnt, user can assign it by applications.
	flexspi.A_SCLK	1	GPIO_SD_B1_07	
	flexspi.A_DATA[0]	1	GPIO_SD_B1_08	
	flexspi.A_DATA[1]	1	GPIO_SD_B1_09	
	flexspi.A_DATA[2]	1	GPIO_SD_B1_10	
	flexspi.A_DATA[3]	1	GPIO_SD_B1_11	
Flash A2	flexspi.A_SCLK	1	GPIO_SD_B1_07	
	flexspi.A_DATA[0]	1	GPIO_SD_B1_08	
	flexspi.A_DATA[1]	1	GPIO_SD_B1_09	
	flexspi.A_DATA[2]	1	GPIO_SD_B1_10	
	flexspi.A_DATA[3]	1	GPIO_SD_B1_11	
	flexspi.A_SS1_B	6	GPIO_SD_B0_00	
		4	GPIO_SD_B1_04	

Figure 3. Two QSPI flash pins connections

The Flash A1 is a boot device, so it must connect to default boot pins, and Flash A2 chip select may remap to other pins according to application. User can change it freely.

3.1.2 Software design

ROM may configure FlexSPI supporting two QSPI flash by flash config block information. It can also modify the xip to change flash configure block. One example of modifying the xip file is to support two QSPI flash as below:

.sflashA2Size = 8u * 1024u * 1024u,

Add one code to define Flash A2 size in qspiflash_config structure, also may find it in attached example code(evkmimxrt1060_flexspi_nor_config.c), ROM can configure FlexSPI to enable A1 and A2 according to the flash config block saved in QSPI flash(A1).

NOTE

By default ROM configures GPIO_SD_B0_00 as SS1 for Flash A2. If it uses other pins for A2 chip select pin, first try to disable GPIO_SD_B0_00 pin, then configure correct pin MUX and pad for the chosen pin.

To enable RWW function, update LUT, and add API function for flash operation. The basic initialization flow is as Figure 4:

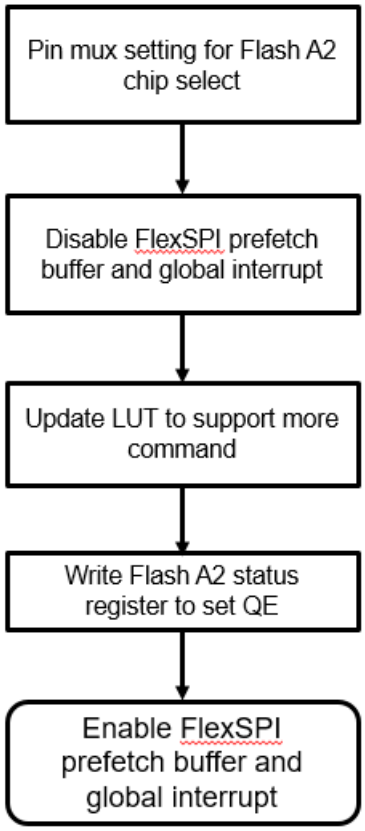

Figure 4. RWW Initialization flowchart

NOTE

Before updating LUT, it require to do software reset by setting SWRESET bit, and then allocate the code of updating LUT to internal SRAM, It avoids the collision issue.

The common code for flash erase API function is also available for RWW implement, it is available in SDK package. For flash programming, as it only has one FlexSPI interface, so if the CPU transfers the data, it generates the hard fault. To avoid this issue, one way is to use DMA for data transfer instead CPU.

Refer to attached example code and flowchart as below:

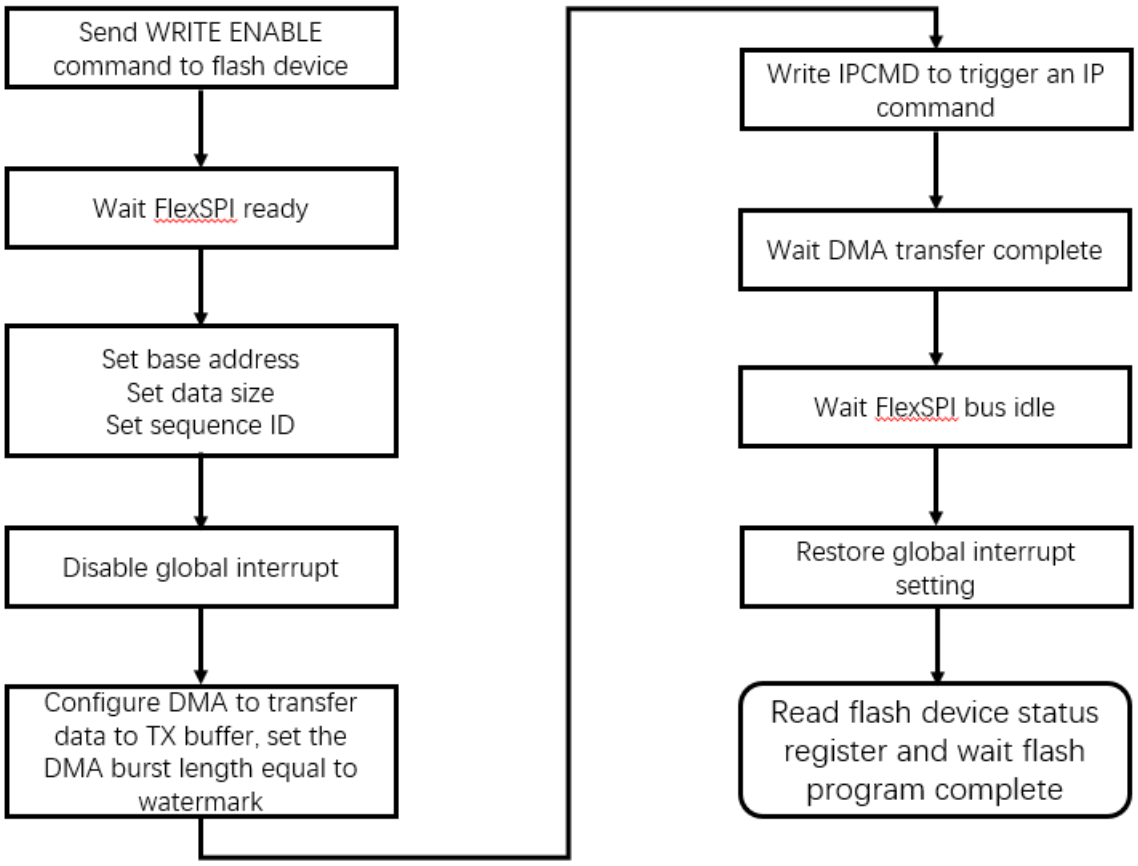

Figure 5. Flash program flowchart

When a new IP command is triggered, the AHB command is stalled by bus arbitration, after FlexSPI complete the IP command execution, and entry into idle status, it resume the suspended command. Take flash program command as example, when program command is send out, it stall others AHB commands, including prefetch operation, after it complete to send out program command and transferred data(one page size for page program command), then it resume previous suspended command, here stalled duration contains command and data transferring, below take one GPIO to measure this duration, it take 6 uS for QSPI flash working at the speed of 133 Mhz, and quad mode is enabled.

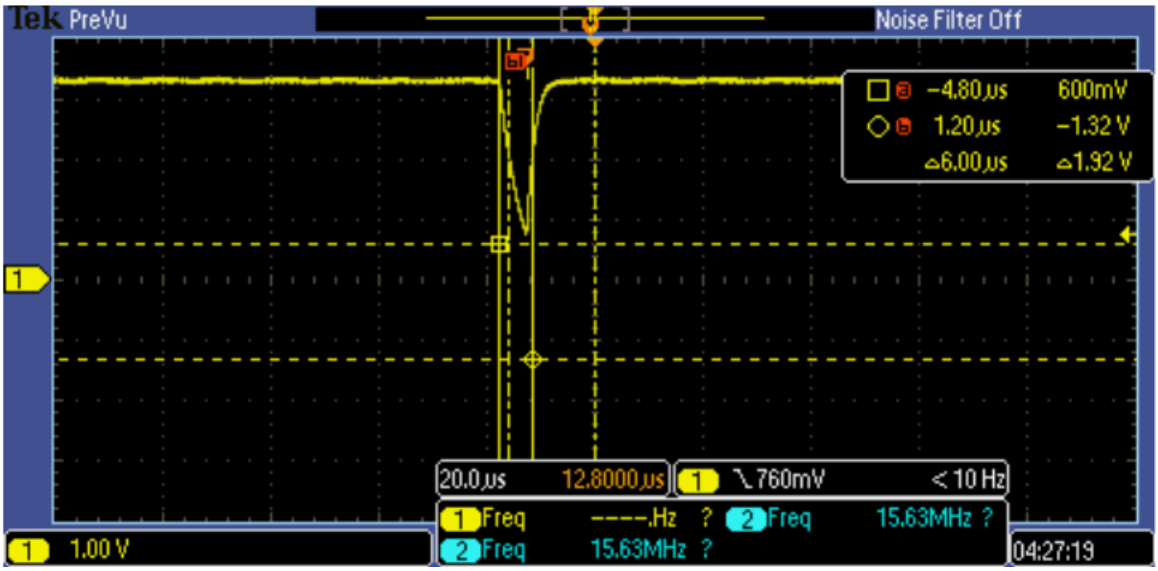

Figure 6. Measurement of one page data transferring

Cache can help to improve the performance, and meanwhile also can cover up some issue in some cases. To verify, if it can truely work well on RWW, the example code also do some test to disable cache and prefetch buffer. Check if it still can work well with above introduced solution.

A snippet code as below:

```
EDMA_StartTransfer(&g_EDMA_FlexSPITxHandle);
base->IPTXFCR |= FLEXSPI_IPTXFCR_TXDMAEN_MASK;
base->IPCMD |= FLEXSPI_IPCMD_TRG_MASK;
while(!(FLEXSPI_DMA->TCD[FLEXSPI_DMA_TX_CHANNEL].CSR & DMA_CSR_DONE_MASK));
```

Figure 7. Snippet code of flash programming

Test results as below with ICACHE and DCACHE disabled.

```
RWW test start ...
Disable DCACHE and ICACHE
start erase flash with address 0x60800000.

erase successed!
start program flash with address 0x60800000.

page program successed!

check flash address 0x60800000 contents.
FlexSPI read/write successfully!
```

Figure 8. RWW test result

3.2 Dedicated RWW Flash Device

Some vendors provide the dedicated flash supporting RWW function, for example, MX25UW12845G, which supports multi-bank to implement RWW function. Figure 9shows flash partition for this flash device.

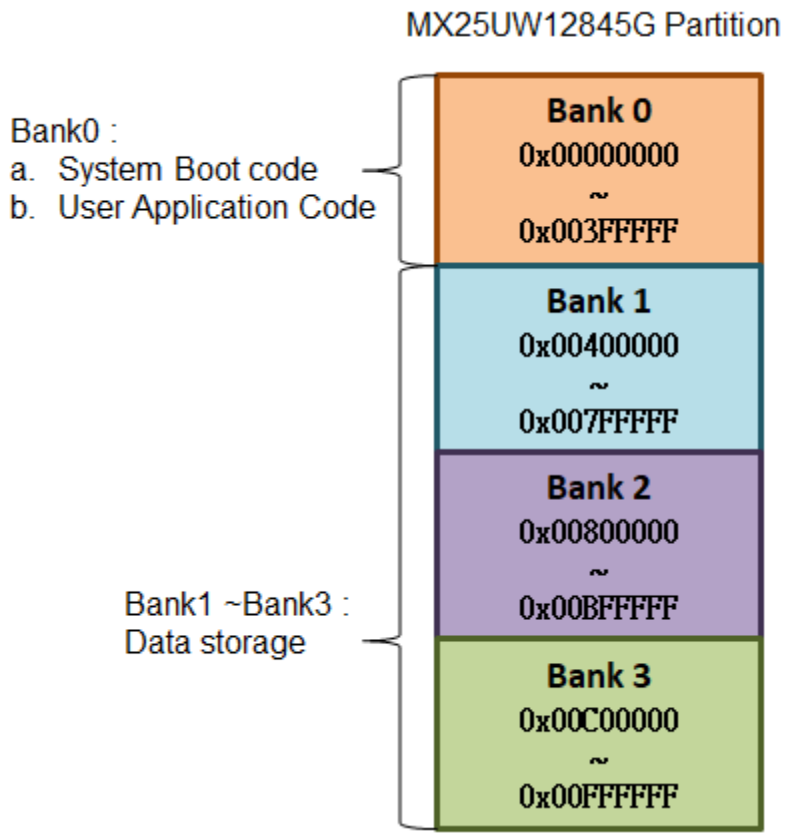

Figure 9. Example of RWW flash

Bank 0 can be used to store firmware, and Bank 1 to Bank 3 used to store data, Figure 9 illustrate the address range for different Banks.

There is no change for hardware connection, it is fully compatible with the common flash device, MX25UW12845G is octal flash, can enable FlexSPI working on combined mode to support octal flash, please refer to AN12107 for octal flash booting.

And for software for MX25UW12845G, please refer to chapter 3.1.2 for modification.

4 Conclusion

This application note introduces, how to implement the RWW based on i.MX RT series. It provides one way to take two QSPI flash to implement RWW. It is very useful for the customer as it is a low-cost solution to implement read while write in some applications.

5 Revision history

Table 1. Revision history

Revision number	Date	Substantive changes
0	09/2019	Initial release

How To Reach Us

Home Page:

nxp.com

Web Support:

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: nxp.com/SalesTermsandConditions.

While NXP has implemented advanced security features, all products may be subject to unidentified vulnerabilities. Customers are responsible for the design and operation of their applications and products to reduce the effect of these vulnerabilities on customer’s applications and products, and NXP accepts no liability for any vulnerability that is discovered. Customers should implement appropriate design and operating safeguards to minimize the risks associated with their applications and products.

For more information, please visit: http://www.nxp.com

For sales office addresses, please send an email to: salesaddresses@nxp.com

Date of release: Septeber 2019

Document identifier: AN12564

arm