7Bus and Storage

7.1 System Bus

The FM33LC0 line architecture consists of the following main components:

ÿ Two Masters

- Cortex-M0 core
- DMA controller

ÿ Five Slaves

- Internal Flash memory
- Internal SRAM memory
- GPIO controller module
- System module
- AHB-APB line swing bridge

The system line diagram of FM33LC0XX is as follows, including one AHB-Lite line and one APB line.

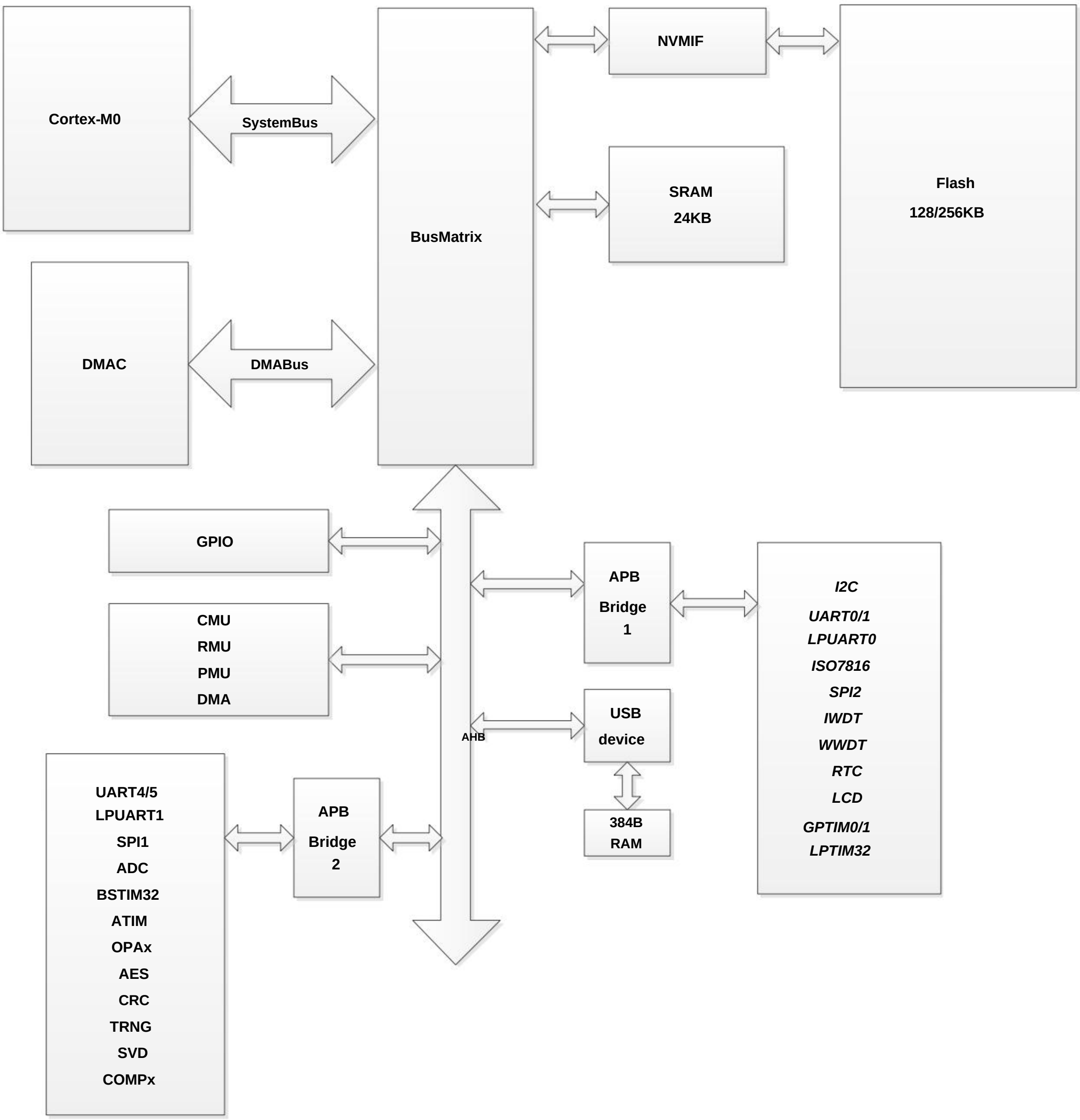


Figure **7-1** System bus diagram

7.2 Storage Space Allocation

7.2.1 Overview

The size of the Flash sector is 512 bytes, and every 16 sectors form an 8K-byte block.

Flash contains 4 information areas, 2 LDT areas, 1 redundant area, and 1 DCT area.

It is the chip factory reserved area, not used. Information is the configuration area, used to save the configuration information.

The option area is isolated from the main Flash area in terms of address.

When the chip boots from Flash, the address space allocation of FM33LC0XX is as follows (256KB Flash, 24KB RAM):

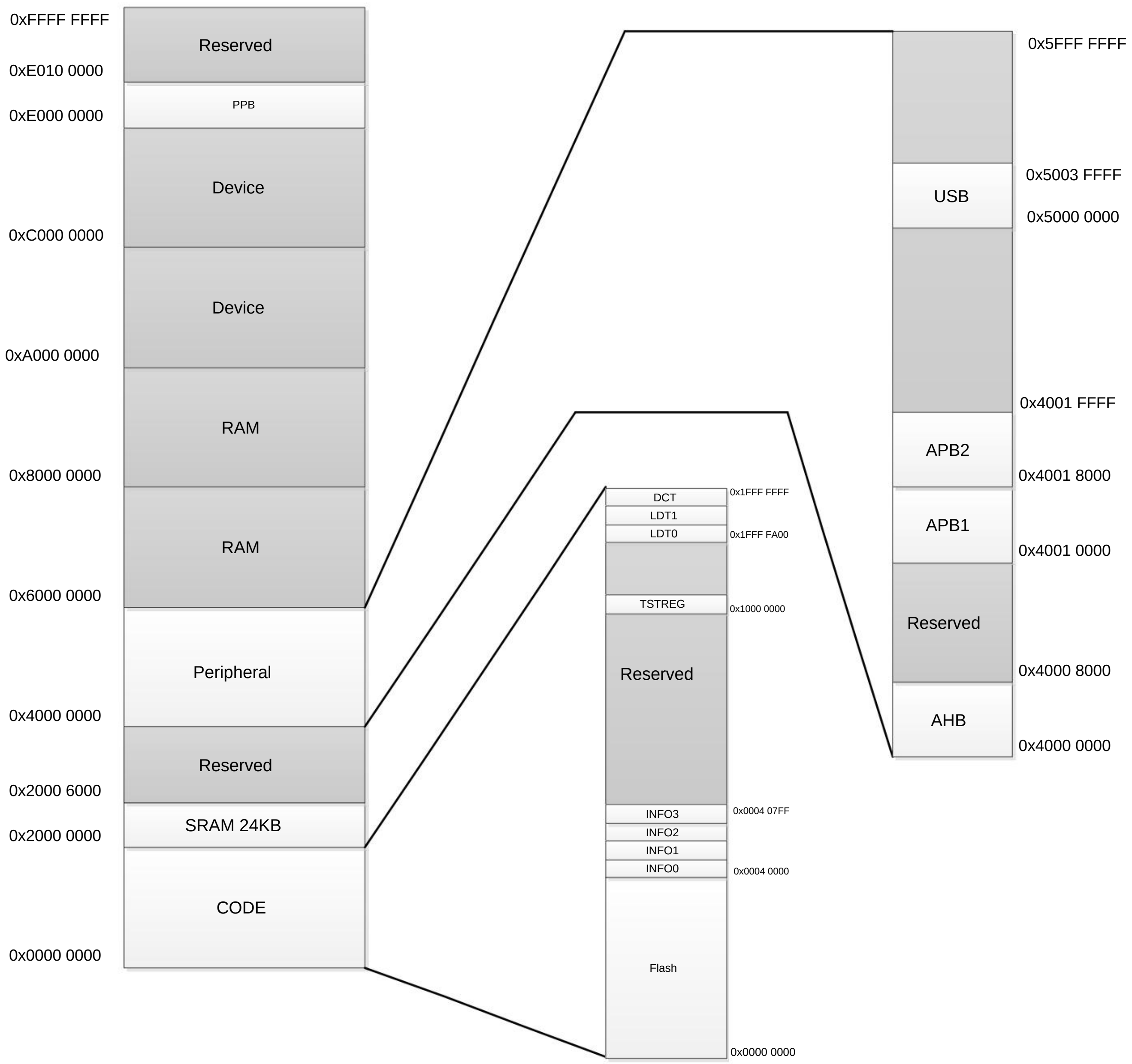


Figure **7-2FM33LC04x** bus address

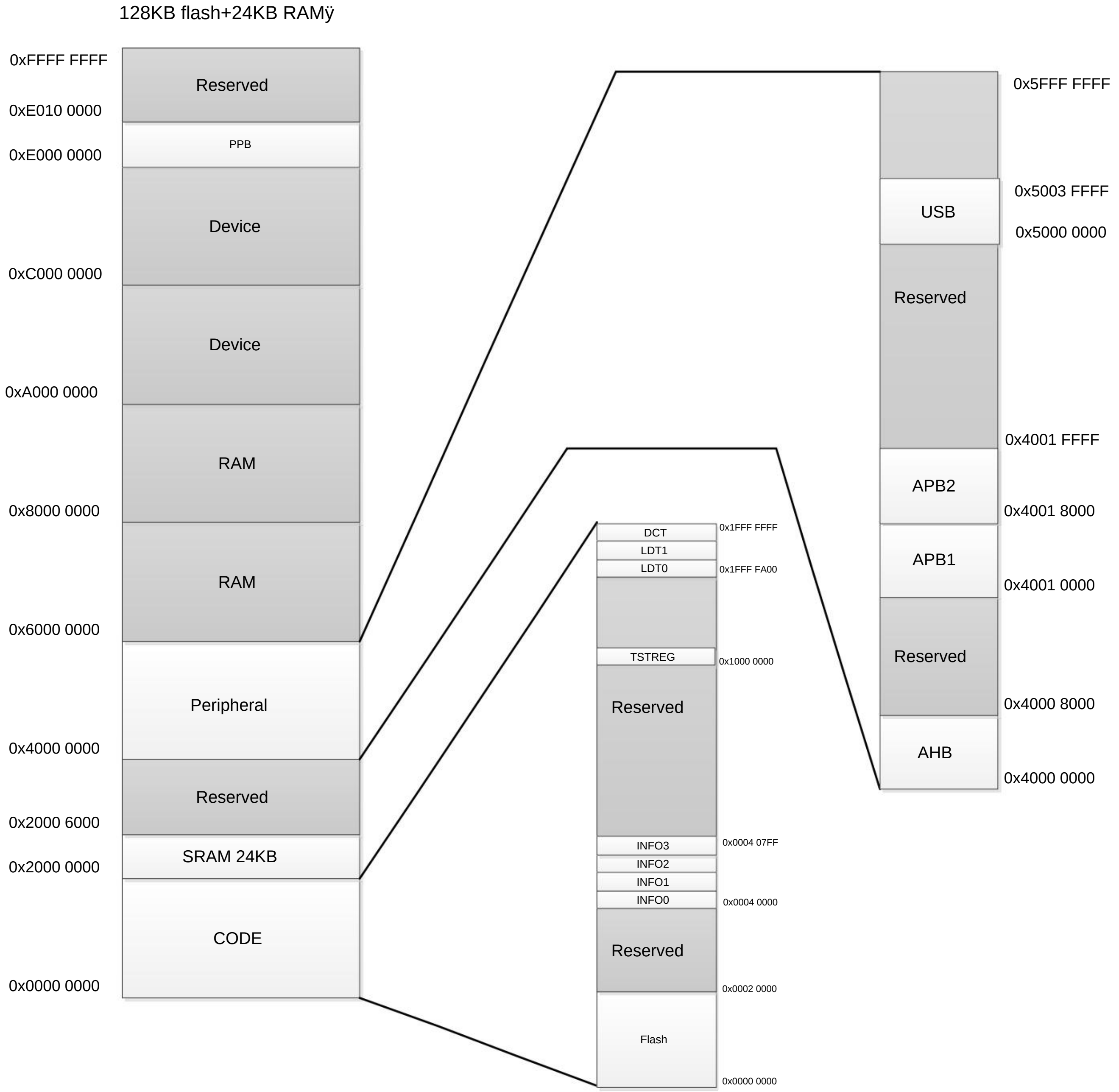


Figure **7-3FM33LC02x** bus address

7.2.2 Peripheral module register address allocation

The following table lists the address space allocation range of the peripheral modules. Each peripheral module occupies 1KB of address space.

Wire	Address	space	Peripherals
AHB	boundaries 0x4000_0C00~0x4000_0FFF	1KB	GPIO
	0x4000_0000~0x4000_03FF	1KB	SCU, PMU, CMU, RMU
	0x4000_0400~0x4000_07FF	1KB	DMA
	0x4000_1000~0x4000_13FF	1KB	NVMIF
	0x5000_0000~0x5001_FFFF	128KB	USB Controller
	0x5002_0000~0x5003_FFFF	128KB	USB packet RAM

	0x0000_0000~0x0003_FFFF	256KB	Flash main array
	0x1FFF_F000~0x1FFF_FFFF	4KB	Flash Option cell array
	0x2000_0000~0x2000_5FFF	24KB	SRAM
	0x4001_0000~0x4001_7FFF	32KB	APB1
	0x4001_8000~0x4001_FFFF	32KB	APB2
APB1	0x4001_0000~0x4001_03FF	1KB	ISO7816
	0x4001_0400~0x4001_07FF	1KB	LPUART0
	0x4001_0800~0x4001_0BFF	1KB	SPI2
	0x4001_0C00~0x4001_0FFF	1KB	LCD
	0x4001_1000~0x4001_13FF	1KB	RTC
	0x4001_1400~0x4001_17FF	1KB	IWDT
	0x4001_1800~0x4001_1BFF	1KB	WWDT
	0x4001_1C00~0x4001_1FFF	1KB	UART0
	0x4001_2000~0x4001_23FF	1KB	UART1
	0x4001_2400~0x4001_27FF	1KB	I2C
	0x4001_2800~0x4001_2BFF		Reserved
	0x4001_2C00~0x4001_2FFF	1KB	RAMBIST
	0x4001_3000~0x4001_33FF		Reserved
	0x4001_3400~0x4001_37FF	1KB	LPTIM32
	0x4001_3800~0x4001_3BFF	1KB	GTIMER0
	0x4001_3C00~0x4001_3FFF	1KB	GTIMER1
APB2	0x4001_8000~0x4001_83FF	1KB	CRC
	0x4001_8400~0x4001_87FF	1KB	LPUART1
	0x4001_8800~0x4001_8BFF		Reserved
	0x4001_8C00~0x4001_8FFF	1KB	SPI1
	0x4001_9000~0x4001_93FF		Reserved
	0x4001_9400~0x4001_97FF		Reserved
	0x4001_9800~0x4001_9BFF		Reserved
	0x4001_9C00~0x4001_9FFF		Reserved
	0x4001_A000~0x4001_A3FF	1KB	UART4
	0x4001_A400~0x4001_A7FF	1KB	UART5
	0x4001_A800~0x4001_ABFF	1KB	SVD,OPA,COMP
	0x4001_AC00~0x4001_AFFF	1KB	ADC
	0x4001_B000~0x4001_B3FF	1KB	ATIM
	0x4001_B400~0x4001_B7FF	1KB	BSTIM32
	0x4001_B800~0x4001_BBFF	1KB	AES
	0x4001_BC00~0x4001_BFFFTable	1KB	TRNG

7-1 Outline layout table

7.3 RAM

7.3.1 Overview

FM33LC0XX contains a 24KB RAM (6K*32), the address space range is 0x2000_0000~0x2000_5FFF, the soft

The software can access the SRAM by byte, half word, or word, and both the CPU and DMA can access the SRAM at the maximum system frequency.

The CPU can also fetch programs from SRAM, so in situations where the program rate is high, it can be

Import part of the code into SRAM to achieve the highest frequency and other rows.

7.4 Flash

7.4.1 Overview

The Flash capacity used by FM33LC0XX is 64K*32, that is, 256KB; the array organization grid includes page (512B) and sector (2KB).

The main array contains 512 areas, including page, zone and full.

7.4.2 Special Information Sector Description

In addition to the main array of Flash, there are several special areas available for use, as described below:

area	illustrate	use
LDT1	option area	option bytes (OPTBYTES)
IF	Information Area	4 pages with a total size of 2KB for use

7.4.2.1 **LDT1 page**

LDT1 is the user configuration information area and can be written in user mode (can only be written using SWD, that is, written by a programmer).

The line address of LDT1 is 0x1FFF_FC00~0x1FFF_FDFF; the internal physical address of Flash is (MCS=1, LDTEN[1:0]=10, ADD=0x0000~0x007F)

AHB addr	Bit[31:16]	Bit[15:0]	Description
0x1FFF_FC00	~OPTBYTES[15:0]	OPTBYTES[15:0]	Use the lower half word of the option byte
0x1FFF_FC04	~OPTBYTES[31:16]	OPTBYTES[31:16]	Use the upper half word of the option byte
0x1FFF_FC08	LOCK1		ACLOCK configuration word, control low 16 blocks
0x1FFF_FC0C	LOCK2		ACLOCK configuration word, 16 blocks high

Table **7-2LDT1** data content definition

OPTBYTES The option bytes are defined as follows:

Bitfield	Mnemonics	Functional Description	Factory default
31:24	BTSWPEN	Boot area enable 0x55: Enable the boot zone exchange function Others: No start zone	0xFF
23:20	IWDTSLP	Configure whether IWDT is allowed to stop counting in low power mode. 0xA: Allowed in Sleep/DeepSleep/RTCBKP mode Use to stop IWDT Others: Do not use IWDT to stop in any mode	0xF
19:16	DFLSEN	Data flash enable 0x5: Enable flash, main array up to 16KB	0xF

Bitfield Mnemonics		Function	Factory default
		address is defined as data flash Others: Flash is prohibited	
15:8	ACLKEN	Enabling with code 0x33: Prohibited ACLOCK Others: Enable ACLOCK	0x33
7:0	DBRDPEN	Debug port access protection enable 0xAA: Disable debug port protection Others: Enable debug port protection	0xAA

Table **7-3** User option byte definition

ÿNoteÿWhen leaving the factory, the SWD port can be used by software to write OPTBYTES; however, ACLKEN DBRDP Enabled, user must rewrite OPTBYTES after full flash via SWD.

The LOCK configuration byte is defined as follows:

Bitfield Mnemonics		Functional Description	Factory default
31:0	LOCK1	Block Lock word 1, every 2it for 8KB Block 11: Insurance 01, 10: Software read and write protection, only read 00: Software read/write protection, read only; SWD read/write protection LOCK1[1:0] for Block0 (8KB of Flash lowest address space), LOCK1[31:30] for Block15 (Flash address space 120~128KB), others like	0xFFFFFFFF
31:0	LOCK2	Block Lock word 2, each 2it for 8KB Block 11: Insurance 01, 10: Software read and write protection, only read 00: Software read/write protection, read only; SWD read/write protection LOCK2[1:0] for Block16 (Flash address space 128~136KB), LOCK2[31:30] for Block31 (Flash Address space 248~256KB), others like	0xFFFFFFFF

Table **7-4 LOCK** information definition

The relationship between the LOCK bit and the Flash permission lock address is shown in the following table:

Address	**LOCK bits**
0x0000_0000 ~ 0x0000_1FFF 0x0000_2000	LOCK1[1:0]
~ 0x0000_3FFF 0x0000_4000 ~ 0x0000_5FFF	LOCK1[3:2]
0x0000_6000 ~ 0x0000_7FFF 0x0000_8000	LOCK1[5:4]
~ 0x0000_9FFF 0x0000_A000 ~	LOCK1[7:6]
0x0000_BFFF 0x0000_C000 ~ 0x0000_DFFF	LOCK1[9:8]
0x0000_E000 ~ 0x0000_FFFF 0x0001_0000	LOCK1[11:10]
~ 0x0001_1FFF 0x0001_2000 ~ 0x0001_3FFF	LOCK1[13:12]
0x0001_4000 ~ 0x0001_5FFF	LOCK1[15:14]
	LOCK1[17:16]
	LOCK1[19:18]
	LOCK1[21:20]

Address	LOCK bits
0x0001_6000 ~ 0x0001_7FFF	LOCK1[23:22]
0x0001_8000 ~ 0x0001_9FFF	LOCK1[25:24]
0x0001_A000 ~ 0x0001_BFFF	LOCK1[27:26]
0x0001_C000 ~ 0x0001_DFFF	LOCK1[29:28]
0x0001_E000 ~ 0x0001_FFFF	LOCK1[31:30]
0x0002_0000 ~ 0x0002_1FFF	LOCK2[1:0]
0x0002_2000 ~ 0x0002_3FFF	LOCK2[3:2]
0x0002_4000 ~ 0x0002_5FFF	LOCK2[5:4]
0x0002_6000 ~ 0x0002_7FFF	LOCK2[7:6]
0x0002_8000 ~ 0x0002_9FFF	LOCK2[9:8]
0x0002_A000 ~ 0x0002_BFFF	LOCK2[11:10]
0x0002_C000 ~ 0x0002_DFFF	LOCK2[13:12]
0x0002_E000 ~ 0x0002_FFFF	LOCK2[15:14]
0x0003_0000 ~ 0x0003_1FFF	LOCK2[17:16]
0x0003_2000 ~ 0x0003_3FFF	LOCK2[19:18]
0x0003_4000 ~ 0x0003_5FFF	LOCK2[21:20]
0x0003_6000 ~ 0x0003_7FFF	LOCK2[23:22]
0x0003_8000 ~ 0x0003_9FFF	LOCK2[25:24]
0x0003_A000 ~ 0x0003_BFFF	LOCK2[27:26]
0x0003_C000 ~ 0x0003_DFFF	LOCK2[29:28]
0x0003_E000 ~ 0x0003_FFFF	LOCK2[31:30]

Table **7-5 LOCK** bit and **Flash** address correspondence table

7.4.2.2 **Information3 page**

Flash also contains 4 information areas, of which information3 is used to control the BootSwap function. Information3 line address

It is 0x0004_0600~0x0004_07FF;

When BOOTSWAPEN=0x55 in LDT1, BootSwap can be performed through the lowest address content in INFO3.

When it is 0x5454_ABAB, the chip will interleave the logical addresses of the two lowest 8KB spaces of the Flash (Note: ACLOCK

Only the logical address is processed without considering the actual physical address), thereby achieving the risk upgrade of the startup code.

The BootSwap diagram is as follows:

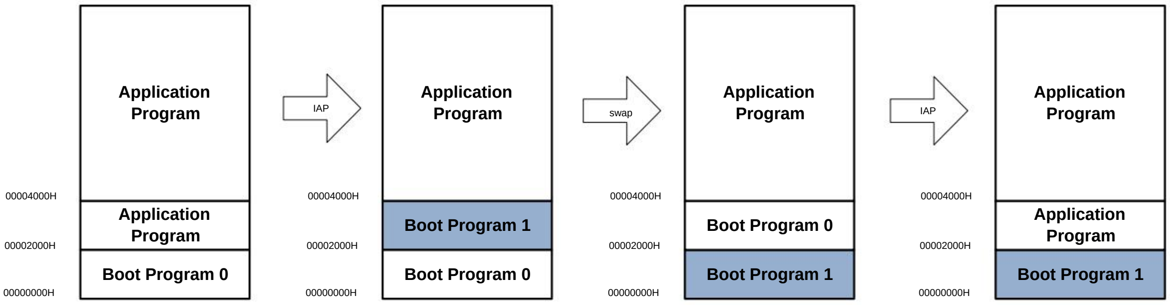


Figure **7-4 Bootswap** diagram

Its main purpose is to prevent unexpected events (power outages, frequent resets, etc.) when the system is updating the startup code.

The boot code has been deleted, which will cause the chip to not run normally after restarting.

After implementing the BootSwap function, assuming that the boot code occupies 8KB of space from 0000 to 1FFF, the boot code should be moved to

Write to address 2000~3FFF, and then enable BootSwap. There are several possibilities:

ÿ When the chip writes to the address 2000~3FFF, it will not restart because the original startup code is still there.

ÿ The chip is functionally written to boot program1, and then bootswap is enabled and soft reset is performed in parallel. After the chip restarts, the boot

program1

ÿ When the chip is writing boot program0, it will not respond to subsequent operations because boot program1 has already been written.

It is recommended to upgrade according to the following steps:

ÿ Update application program

ÿ If you need to upgrade the boot, first write the boot program into the second 8KB space

ÿ Configure INFO3 and enable BootSwap

ÿ Perform a soft reset and restart the boot program

ÿ Write the second 8KB space as the application program

The remapping of logical addresses to Flash physical addresses is done automatically by the chip. Both the program and the debugger access the Flash memory using logical addresses.

ask.

Note: ***IF3*** of ***BootSwap*** The function actually only uses the lowest one on this page ***word*** , the rest of the address space is read and written (specific

Function), software ***Debugger*** Can be written in any way.

7.4.2.3 Information1~2 pageÿDebugger only, lockableÿ

These two information areas are for user use. Only SWD can write them, and software can read them.

The line address is 0x0004_0200~0x0004_05FF, the low address is IF1, the high address is IF2, a total of 1KB.

The highest address byte of each of the two areas IF2~1 is the area lock mark. If SWD writes the highest address byte to 0x55, the chip

After the chip is reset, the front area is prohibited from being programmed, and after SWD is performed, it can be reprogrammed.

Regardless of whether there is a lock mark or not, these areas are readable by SWD and software.

7.4.2.4 **Information0** pageÿOTPÿ

IF0 is an OTP page. This page can only be programmed once in the user mode and cannot be erased or written after programming.

The address is 0x0004_0000~0x0004_01FF, a total of 512 bytes.

There is no restriction on reading the IF0 page.

7.4.3 **Flash** Programming

7.4.3.1 Overview

FM33LC0XX Flash programming method as follows:

ÿ In-system programming (ISP): Programming the chip by online emulation through the FMSH dedicated programmer, using the SWD port

ÿ In-use Programming (IAP): Chip self-programming is achieved through bootloader code, and any communication port can be defined, which can be used for

Implement online program upgrade

Flash must be cleared before programming. Flash can be cleared in three ways: full, area, and page.

7.4.3.2 **Flash** Erase Clock

The calibrated RCHF clock is used when writing to Flash, but the system clock can be any clock.

The frequencies are 8M, 16M and 24M.

The Flash write clock is independent of the CPU clock, so it will not affect program execution.

7.4.3.3 **Flash** Erase Method

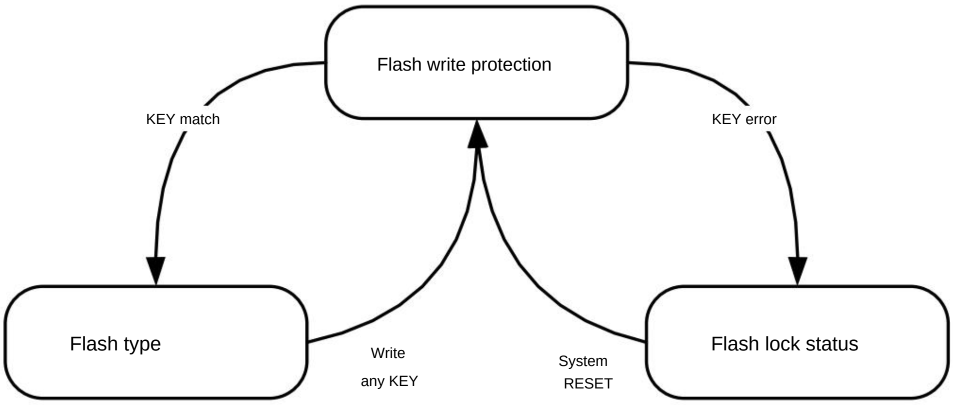
FM33LC0XX Flash deletion, as well as single programming and continuous programming.

Key verification is required before writing to Flash. If the writing order is wrong or the writing value is wrong, the Flash Key will be written before the verification is correct.

The Flash Key authentication error will result in the write failure.

Flash will not be reset until the next time. After the normal writing is completed, writing any value to the KEY register will return the state to the initial state.

Write a warranty. The warranty will be as shown below:



The software can confirm the current Key input status by querying FLSIF.KEYSTA. For details, refer to the register description.

7.4.3.4 **Matrix Erase**

The whole operation can only be started from the SWD port, and the software is prohibited from performing the whole operation. The whole operation only removes the main array and does not remove special

SWD can start the whole system in the manufacturer's commercial mode. The operation process is as follows:

ÿ The programmer configures the ERTYPE register to 10 via SWD

ÿ The programmer clears the PREQ register and sets the EREQ register via SWD

ÿ The programmer writes to Flash via SWD Full Key: 0x9696_9696 and 0x7D7D_7D7D

ÿ SWD writes to any address in Flash and requests 0x1234_ABCD

ÿ The chip starts all access to Flash and suspends any Master access to Flash

ÿ After the completion, the middle and full marks are set (the full mark means that the main array is completely divided, any Programming will clear this flag)

ÿ In the case of full mark, SWD can write LDT1 at will, otherwise writing LDT1 is prohibited and triggers an error

ÿ After the software confirms that the removal is complete, write any value to the FlashKEY register to rewrite the protection

Note: All works can only be removed ***Flash*** of ***main array*** , will not ring special message area

7.4.3.5 **Sector Erase**

Both SWD and user code can be used to perform partitioning. The operation flow is as follows:

ÿ Configure the ERTYPE register to 01

ÿ Clear the PREQ register and set the EREQ register

ÿ Write Flash block Key: 0x9696_9696 and 0x3C3C_3C3C

ÿ Write a request to delete any address in the Page to be deleted 0x1234_ABCD

ÿ The chip checks whether the target area belongs to the block locked by ACLOCK. If it is not locked, the target area is cleared. If locked, an error message is triggered

ÿ After the area is completed, the winning bid will be set

ÿ After the software confirms that the removal is complete, write any value to the FlashKEY register to rewrite the protection

7.4.3.6 Page Erase **Operation**

Both SWD and code can execute pages. The operation flow is as follows:

ÿ Configure the ERTYPE register to 00 11

ÿ Clear the PREQ register and set the EREQ register

ÿ Write Flash block Key: 0x9696_9696 and 0xEAEA_EAEA

ÿ Write a request to any address in the area to be deleted 0x1234_ABCD

ÿ The chip checks whether the target area belongs to the block locked by ACLOCK. If it is not locked, the target area is cleared. If locked, an error message is triggered ÿ After the area is completed, the winning bid will be set

ÿ After the software confirms that the removal is complete, write any value to the FlashKEY register to rewrite the protection

7.4.3.7 One-time programming

Single-shot programming is initiated by software, and the minimum programming unit is 32 bits. The operation process is as follows:

ÿ Clear the EREQ register and set the PREQ register

ÿ Clear the continuous programming enable register

ÿ Write Flash programming key: 0xA5A5_A5A5 and 0xF1F1_F1F1

ÿ Write to the Flash target address. If the target address is locked by ACLOCK, an error flag is triggered. If it is not locked,

Programming

ÿ After programming, set the winning bid

ÿ After the software confirms that programming is complete, write any value to the FlashKEY register to rewrite the protection

7.4.3.8 Continuous Programming

Continuous programming writes half-sector (256 bytes) to Flash at a time through the DMA memory channel.

Read from RAM address, Flash target programming address must be half-sector aligned, that is, Flash address lower 6

The bit is 0. When this method is used, the length of one-time programming is fixed, which is mainly used for fast mass writing.

During the continuous programming, DMA completely occupies the Flash line and suspends all CPU access to Flash.

The operation process is as follows:

ÿ Clear the EREQ register and set the PREQ register

ÿ Set the continuous programming enable register (DMA mode enable)

ÿ Write 256 bytes to RAM

ÿ Configure DMA memory channel, set transfer direction, read address and write address

ÿ Enable DMA memory channel

ÿ Write Flash programming key: 0xA5A5_A5A5 and 0xF1F1_F1F1

ÿ Software triggers DMA memory channel, DMA reads RAM 64 times continuously and programs Flash

ÿ The chip checks whether the programmed area is locked by ACLOCK. If locked, an error is triggered and DMA is notified to stop programming.

ÿ After 256 bytes are fully programmed, the Flash line is released.

ÿ After the software confirms that programming is complete, write any value to the FlashKEY register to rewrite the protection

Note: If the Flash is being written while the CPU is fetching from the Flash, the CPU fetch will be suspended until the write is completed.

If the CPU jumps to RAM to fetch the operation, Flash writing will not pause the CPU.

In the hope that the code can still respond in real time when it is executed in RAM, the vector table is remapped into RAM.

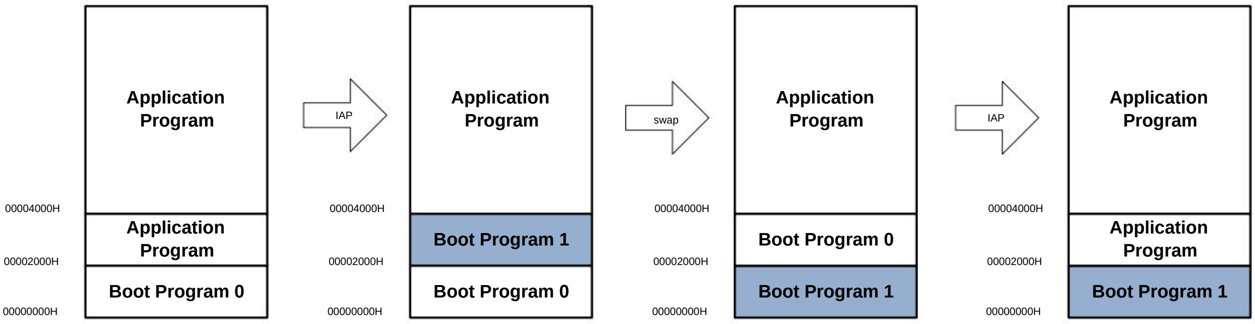
7.4.3.9 **Boot Area Swap (BootSwap)**

The main purpose of BootSwap is to prevent unexpected events (power outages, frequent resets, etc.) when the system is updating the startup code.

The original boot code has been deleted, which will cause the chip to not run normally after restart.

Page3 lowest address word implementation.

The BootSwap diagram is as follows:



After implementing the BootSwap function, assuming that the boot code occupies 8KB of space from 0000 to 1FFF, the boot code should be moved to

Write to address 2000~3FFF, and then enable BootSwap. There are several possibilities:

ÿ When the chip writes to the address 2000~3FFF, it will not restart because the original startup code is still there.

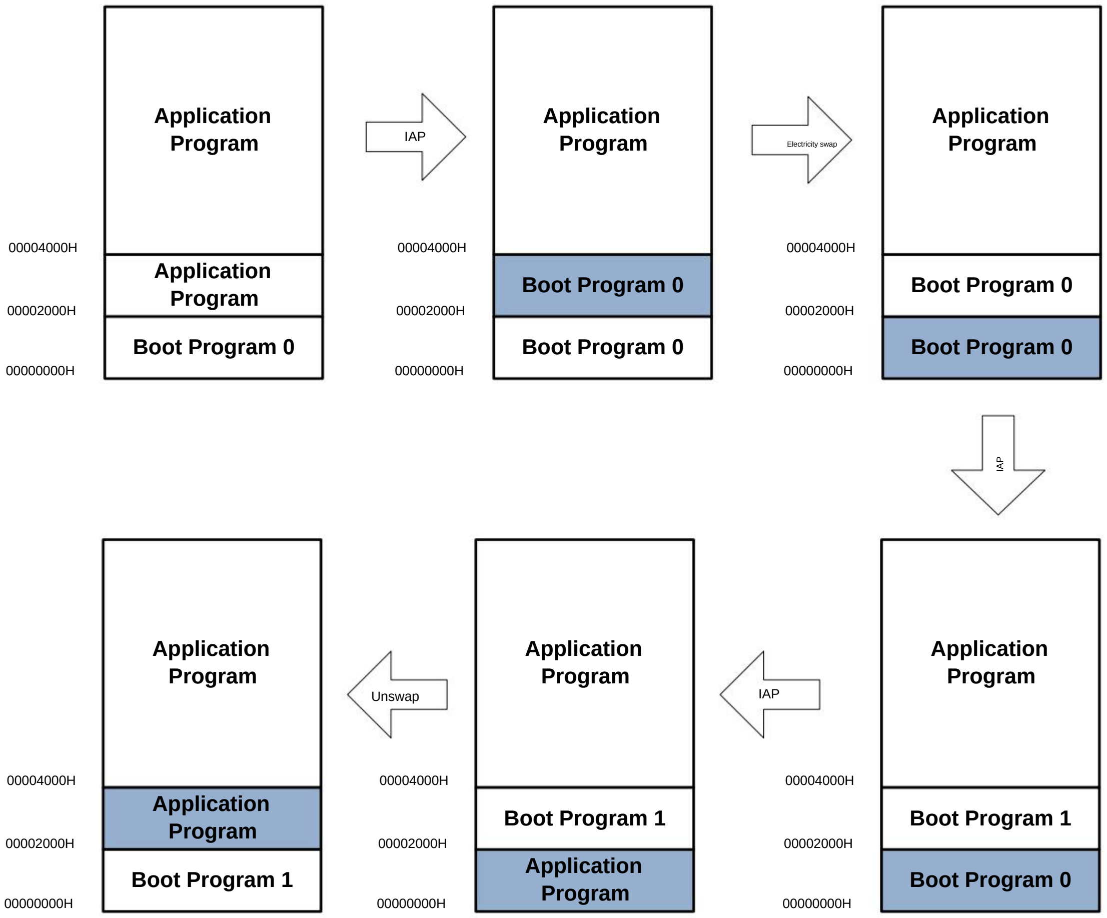
ÿ The chip is functionally written to boot program1, and then bootswap is enabled and soft reset is performed in parallel. After the chip restarts, the boot

program1

ÿ When the chip is writing boot program0, it will not respond to subsequent operations because boot program1 has already been written.

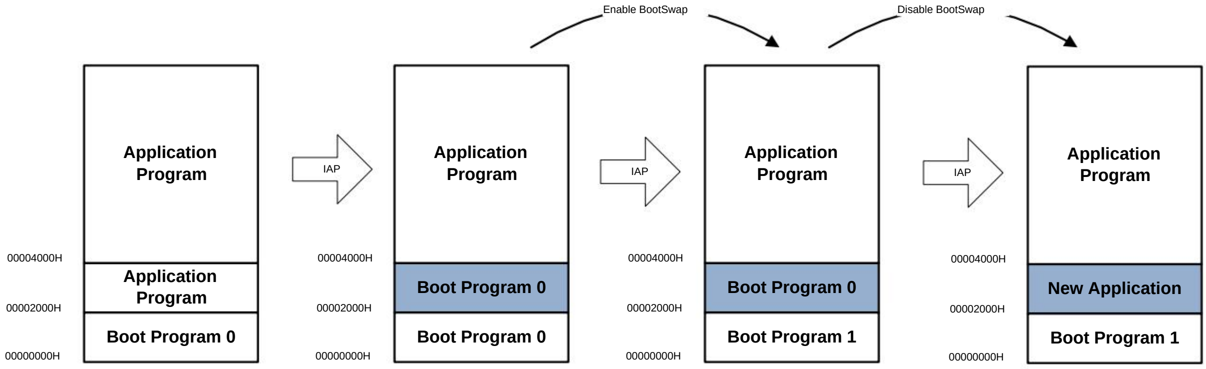
Another use of BootSwap is shown in the figure below. In order to ensure reliable boot code update, the 2nd 8KB physical space is used as the original

Backup of the Boot program. If a power failure occurs during programming, BootSwap is triggered:



If there is no power failure during the startup program update, you can perform a soft reset and a real swap.

Enable BootSwap before the Boot program is run, and disable BootSwap after the function is changed:



It is recommended to upgrade according to the following steps:

ÿ Update application program

ÿ If you need to upgrade the boot, first write the boot program into the second 8KB space

ÿ Configure information block and enable BootSwap

ÿ Perform a soft reset and restart the boot program ÿ Write the second 8KB space as the application program

The register mark (FLSIF.BTSF) is used to indicate whether the previous Boot area is the 1st 8KB physical address or the 2nd 8KB physical address.

Query the status before starting using the given code.

7.4.4 Data Flash

After enabling DFLSEN with configuration OPTBYTES, FM33LC0XX will use 16KB flash is used for

After enabling data flash, the flash capacity of different models is divided as follows:

model	Data flash size	Data flash address	Program flash size
FM33LC04x 16KB		0x0003_C000~0x0003_FFFF 240KB	
FM33LC02x 16KB		0x0001_C000~0x0001_FFFF 112KB	

After enabling data flash, the corresponding program flash space will be reduced by 16KB. Data flash is located at the flash logical address

The maximum space is 16KB.

Data flash and program flash have no difference in access rights and are also controlled by DBRDP and ACLOCK.

When the chip is fully operated, the data flash will not be cleared. When the data flash is disabled, the chip will clear the main array.

have.

Note: When enabling *data flash* In this case, the chip full time increases significantly. *256KB* Capacity model, full time from *8ms* Increase to *240ms* ,for *128KB* Capacity model, full time from *8ms* Increase to *112ms* ÿ

7.4.5 **Flash** Content Protection

Flash content protection is mainly used to protect the application code, application and application configuration information in Flash from being read and tampered with by unauthorized users.

ÿ

Flash protection includes two types: Debug port read protection (DBRDP-Debug Read Protection) and user code protection.

Flash lock is controlled by OPTBYTES in LDT1.

To make.

7.4.5.1 **Debug port protection (DBRDP)**

The main purpose of DBRDP is to prevent unauthorized third parties from accessing the chip Flash content through the debug port.

DBRDP is disabled by the DBRDPEN configuration word enabler in the LDT1 area (0xAA means DBRDP is disabled, the chip is factory-set

The default value is 0xAA). When DBRDP is enabled, it is not possible to read and write the Flash main array through the SWD port.

It is not possible to access the RAM through the SWD port.

Method to exit DBRDP: Clear the entire flash space through SWD. After the complete, SWD can write any

OPTBYTES disables DBRDP and then resets the chip; after the reset, the chip will be in debug mode.

7.4.5.2 **Application Code Protection (ACLOCK)**

The main purpose of ACLOCK is to prevent hacking code from reading and tampering with the application code in the Flash.

Function, you can set the CPU to only access certain areas of the Flash, not read-as-data, and not write.

ACLOCK works in blocks, that is, the granularity of Flash protection is 8KB, and each Flash contains 32 blocks.

Each block has a 2-bit LOCK signal. The factory default LOCK word is 0xFFFF_FFFF.

All bits are 11, which is the default lock state; when the LOCK bit is 01 10, this block prohibits the CPU from writing and reading

Read, can only read; when the LOCK bit is 00, the previous block prohibits CPU writing and reading, and prohibits SWD

Write and read. When the chip leaves the factory, the ACLOCK function is disabled in LDT0. You need to enable ACLOCK through the programmer and use

The code must be compiled in accordance with the ACLOCK configuration (for example, the literal pool cannot be compiled into a locked block).

ACLOCK features:

ÿ Security: Block allows CPU to access, read, and write, and does not restrict SWD access

ÿ Read and write protection: Block allows CPU to read, does not allow CPU and DMA to read and write, does not restrict SWD access

ÿ Software and SWD protection: Block allows CPU to read, does not allow CPU and DMA to read, write, does not allow SWD

Read, Write

The relationship between the LOCK bit and the Block access rights can be found in the following table:

LOCK bit software	read permission	Software	**SWD** read and write
11		permission	allow
01/10		permission	allow
00	prohibition prohibition	permission	prohibit

Table **7-6 LOCK** bit permission definition

The ACLOCK signal is loaded into the registers when the chip is reset. These registers can also be set by software, but cannot be written to 0 (i.e. can only be

Upgrade your insurance level).

When ACLOCK is not enabled, the LOCK register contents.

Note:*ACLOCK* The right restrictions are aimed at*Flash* each *Block* ,and *DBRDP* Independent of each other.*SWD* Orally speaking, *DBRDP*

has a higher priority than*ACLOCK* ,Right now *DBRDP* After being enabled, regardless of*ACLOCK* Whether to start, *SWD* All legal access *Flash* ÿ

Note: Prohibited use *ACLOCK* closure *1st block* Read permission, because *CPU* After reset, first *0* Address Read *MSP*

Needle, *ACLOCK* will result in *CPU* The method starts normally.

The method to exit ACLOCK is to clear the entire flash space through SWD. After that, SWD can write anything.

OPTBYTES disables ACLOCK and then resets the chip; after the reset, the chip will be in ACLOCK state.

7.4.5.3 **Flash** access permission description

Flash space access permission allocation:

Flash area DBRDP		**LOCK bits (per Block)[3]**	**Last byte in page**	**SWD**	**Application**
Main array	ON	00	x	-	R/E/W/F
		01	x	-	Block can only be taken
		10/11	x	-	Block can only be taken
	OFF	00	x	R/E/W	R/E/W/F
		01	x		Block can only be taken
		10/11	x	Access to Block Methods	Block can only be taken
LDT1	ON	x	x	R[2]	R
	OFF	x	x	R/E/W	R
IF3	x	x	x	R/E/W	R/E/W
IF2,1,0	x	x	55	R/E	R
			others	R/E/W	R

Table **7-7 Flash** access permission list

Note:

[1] R: Read, E: Erase, W: Write, F: Fetch

[2] After flash is completed, LDT1 can be written

[3] This assumes ACLOCKEN is set. Without ACLOCKEN, the LOCK bits have no effect.

7.5 Registers

offset address	name	symbol
FLS (module start address: 0x40001000)		
0x00000000	Flash read register ÿFlash Read Control Registerÿ Prefetch	FLS_RDCR
0x00000004	register (Flash Prefetch Control Register) Use the	FLS_PFCR
0x00000008	configuration word register ÿFlash Option Bytes Registerÿ	FLS_OPTBR
0x0000000C	ACLOCK Register 1 ÿFlash Application Code Lock Register1ÿ	FLS_ACLOCK1
0x00000010	ACLOCK Register 2 ÿFlash Application Code Lock Register2ÿ	FLS_ACLOCK2
0x00000014	Flash write register ÿFlash Erase/Program Control Registerÿ	FLS_EPCR
0x00000018	Flash Key Input Register ÿFlash Key Registerÿ	FLS_KEY
0x0000001C	Flash Enable Register ÿFlash Interrupt Enable Registerÿ	FLS_IER
0x00000020	Flash Token Register ÿFlash Interrupt Status Registerÿ	FLS_ISR

7.5.1 Flash Read Control Register (FLS_RDCR)

name	FLS_RDCR							
offset	0x00000000							
bit Bit31	name bit	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
	·							
	U-0							
permission	bit Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	·							
bit	U-0							
permission bit		Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
Bit15	·							
name	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	·						WAIT	
permission	bit name bit permission U-0						R/W-00	

Position No.	Mnemonics	Functional Description
31:2	·	RFU: Unimplemented, read as 0
1:0	WAIT	Flash Read Wait Cycles Config 00/11ÿ0 wait cycle 01ÿ1 wait cycle 10ÿ2 wait cycles When the CPU main frequency is less than or equal to 24MHz, there is no need to start wait; when the main frequency is greater than 24M When the frequency is less than 48Mhz, enable 1 wait, and when the frequency is greater than 48Mhz, enable 2 wait

7.5.2 Prefetch Control Register (FLS_PFCR)

name	**FLS_PFCR**							
offset	0x00000004							
bit Bit31	name bit	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
	·							
	U-0							
permission	bit Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	·							
bit	U-0							
permission bit		Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
Bit15	·							
name	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	·							PRFTEN
permission	bit name bit permission U-0							R/W-0

Position No.	Mnemonics	Functional Description
31:1	·	RFU: Unimplemented, read as 0
0	PRFTEN	To enable prefetch, write 1 when WAIT==00 (Prefetch Enable) 1: Enable Prefetch 0: Prohibit Prefetch

7.5.3 User Configuration Word Register (FLS_OPTBR)

name	FLS_OPTBR							
offset	0x00000008							
Bit31		Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
Position Name	IWDTSLP	·						
Bit permission	R-0	U-0						
Bit23 Bit	name Bit	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
	·					IF2LOCK	IF1LOCK	·
	U-0					R-0	R-0	U-0
permission	Bit15 Bit	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	·					DFLSEN	BTSEN	
Bit	U-0					R-0	R-01	
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	·				ACLOCK		DBRDPEN	
permission	Bit name Bit permission U-0				R-01		R-01	

Position No.	Mnemonics	Functional Description
31	IWDTSLP	Is it allowed to use the suspend counter in IWDT sleep mode? Enable) 1: Allows the user to suspend the IWDT counter in sleep mode. 0: Disable IWDT suspend in sleep mode
30:19 18	·	RFU: Unimplemented, read as 0
	IF2LOCK	Information2 Area Lock Flag (IF2 Lock Enable)

Position No.	Mnemonics	Functional Description
		0: Unlocked 1: Locked. After locked, software cannot write to this area.
17	IF1LOCK	Information1 Area Lock Flag (IF1 Lock Enable) 0: Unlocked 1: Locked. After locked, software cannot write to this area.
16:11	-	RFU: Unimplemented, read as 0
10	DFLSEN	DataFlash ÿÿ (DataFlash Enable) 0ÿ data flash 1: With data flash
9:8	BTSEN	BootSwap Enable 00/01/11: Disable BootSwap function 10: Allow BootSwap
7:4	-	RFU: Unimplemented, read as 0
3:2	ACLOCK	AppCode Lock Enable 00/01/11: ACLOCK is not enabled 10: ACLOCK enabled
1:0	DBRDPEN	Debug Port Read Protection Enable 00/01/11: DBRDP is not enabled 10: DBRDP enabled

7.5.4 **ACLOCK** Register **1 (FLS_ACLOCK1)**

name	**FLS_ACLOCK1**							
offset	0x0000000C							
Bit31 Bit	name bit	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
	LOCK1[31:24]							
	R/W-1111 1111							
permission	Bit23 Bit	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	LOCK1[23:16]							
bit	R/W-1111 1111							
permission	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
Bit	LOCK1[15:8]							
name	R/W-1111 1111							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	LOCK1[7:0]							
permission	Bit name bit permission	R/W-1111 1111						

Position No.	Mnemonics	Functional Description
31:0	LOCK1	The lower 32 bits of the ACLOCK configuration register are used to control Block 15 to Block 0. The code is used to read and write lock. Each block size is 8KB, and each block uses 2 bits for rights restriction. (Lock bits) 11: The previous block allows SWD and software reading and writing 01/10: The previous block allows SWD reading and writing, prohibits software reading and writing, and software can Pick 00: The previous block prohibits SWD reading and writing, prohibits software reading and writing, and software can read There are bits where the software can only write 0, not 1.

7.5.5 **ACLOCK** Register **2 (FLS_ACLOCK2)**

name	**FLS_ACLOCK2**							
offset	0x00000010							
bit Bit31	name bit	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
	LOCK2[31:24]							
	R/W-1111 1111							
permission	bit Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	LOCK2[23:16]							
bit	R/W-1111 1111							
permission bit		Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
Bit15	LOCK2[15:8]							
name	R/W-1111 1111							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	LOCK2[7:0]							
permission	bit name bit permission	R/W-1111 1111						

Position No.	Mnemonics	Functional Description
31:0	LOCK2	ACLOCK configuration register high 32 bits, used to make Block31~Block16 The code is used to read and write lock. Each block size is 8KB, and each block uses 2bit to restrict the right. (Lock Bits) 11: The previous block allows SWD and software reading and writing 01/10: The previous block allows SWD reading and writing, prohibits software reading and writing, and software can Pick 00: The previous block prohibits SWD reading and writing, prohibits software reading and writing, and software can read There are bits where the software can only write 0, not 1.

7.5.6 Flash erase control register (FLS_EPCR)

name	FLS_EPCR							
offset	0x00000014							
Bit31 Bit	name bit	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
	-							
	U-0							
permission	Bit23 Bit	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
bit	U-0							
permission Bit15		Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
Bit	-						ERTYPE	
name	U-0						R/W-00	
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	-						PREQ	EREQ
permission	Bit name bit permission U-0						R/W-0	R/W-0

Position No.	Mnemonics	Functional Description
31:10	-	RFU: Unimplemented, read as 0
9:8	ERTYPE	Flash Erase Type Configuration 00/11ÿPage Erase

Position No.	Mnemonics	Functional Description
		01ÿSector Erase 10ÿChip Erase (SWD only)
7:2	-	RFU: Unimplemented, read as 0
1	PREQ	Program Request Set by software, automatically cleared after hardware programming is completed
0	SWEAT	Erase Request Set by software, automatically cleared after hardware division

7.5.7 Flash Key Input Register (FLS_KEY)

name	FLS_KEY							
offset	0x00000018							
Bit31 Bit	name bit	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
	KEY[31:24]							
	W-0000 0000							
permission	Bit23 Bit	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	KEY[23:16]							
bit	W-0000 0000							
permissi	on Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
Bit	KEY[15:8]							
name	W-0000 0000							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	KEY[7:0]							
permission	Bit name bit permission	W-0000 0000						

Position No.	Mnemonics	Functional Description
31:0	KEY	Flash write key input register, software SWD must be written before starting Correctly write a legal KEY sequence to this address. (Flash Key)

7.5.8 Flash Interrupt Enable Register (FLS_IER)

name	FLS_IER							
offset	0x0000001C							
bit Bit31	name bit	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
	-							
	U-0							
permission	bit Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
bit	U-0							
permissi	on bit	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
Bit15	-				OTPIE AUTHIE KEYIE			CKIE
name	U-0				R/W-0	R/W-0	R/W-0	R/W-0
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	-						PRDIE ERDIE	
permission	bit name bit permission	U-0					R/W-0	R/W-0

Position No.	Mnemonics	Functional Description

Position No.	Mnemonics	Functional Description
31:12	-	RFU: Unimplemented, read as 0
11	OTPIE	OTP program error Interrupt Enable, 1 Yes (OTP program error Interrupt Enable)
10	AUTHIE	Flash read and write permission error enabled, 1 Yes (Flash Authentication Error Interrupt Enable)
9	KEYIE	Flash Key Error Interrupt Enable, 1 Yes (Flash Key Error Interrupt Enable)
8	CKIE	Write timing clock error enable, 1 has (Erase/Program Clock Error Interrupt Enable)
7:2	-	RFU: Unimplemented, read as 0
1	FORTY-FOURTH DAY	Program Done Interrupt Enable, 1 Yes (Program Done Interrupt Enable)
0	ERDIE	Write done interrupt enable, 1 Yes (Erase Done Interrupt Enable)

7.5.9 Flash Flag Register (FLS_ISR)

name	FLS_ISR							
offset	0x00000020							
Bit31 Bit name bit		Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
	-							
	U-0							
permission bit23 Bit		Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-				CASE			BTSF
bit	U-0				R-000			R-0
permission bit15		Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
Position Name	-				OTHER R	AUTHER R	KEYERR	CKERR
Bit permissions	U-0				R/W-0	R/W-0	R/W-0	R/W-0
Bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-						PRD	ERD
Bit permissions	U-0						R/W-0	R/W-0

Position No.	Mnemonics	Functional Description
31:20	-	RFU: Unimplemented, read as 0
19:17	CASE	Flash Write KEY Input Status (Flash Key Status) 000: Flash write protection, no KEY is entered 001: Fully unlocked 010: Zone unlock status 011: Programming unlock status 100: KEY error locked, need to be reset to unlock 101/110/111ÿRFU
16	BTSF	BootSwap Register (BootSwap) 0: The boot program area is the Flash physical address 0000H~1FFFH 1: The boot program area is the Flash physical address 2000H~3FFFH
15:12	-	RFU: Unimplemented, read as 0
11	OTPERR	OTP page programming permission error, hardware set, software write 1 to clear (OTP program Error Flag. Write 1 to clear) 1: Attempt to program an already programmed OTP byte 0: OTP programming error

Position No.	Mnemonics	Functional Description
10	AUTHERR	Flash read and write permission error, read LOCK block and set when writing LOCK block Bit, software writes 1 to clear. (Flash Authentication Error Flag, write 1 to clear) 1: Flash access permission error 0: No permission error occurred in Flash access
9	KEYERR	Flash Key Error Flag, write 1 to clear
8	CKERR	Write timing clock error, if RCHF is not enabled when NVMIF writes Flash, Then CKERR is triggered, and the software writes 1 to clear it. (Erase/Program Clock Error Flag, write 1 to clear)
7:2	-	RFU: Unimplemented, read as 0
1	PRD	Program Done, programming completed, hardware set, software write 1 clear (Program Done Flag,write 1 to clear)
0	ERD	Erase Done, write done flag, hardware set, software write 1 to clear (Erase Done Flag, write 1 to clear)