HDSC 华大半导体 HUADA SEMICONDUCTOR

32 -bit microcontroller

FLASH operating instructions and precautions

Applicable

object series	product model series		product model series		product model
HC32L110	HC32L110C6UA	**HC32F00**	HC32F003C4UA	**HC32L13**	HC32L130E8PA
	HC32L110C6PA		HC32F003C4PA		HC32L130F8UA
	HC32L110C4UA		HC32F005C6UA		HC32L130J8TA
	HC32L110C4PA		HC32F005C6PA		HC32L136J8TA
	HC32L110B6PA		HC32F005D6UA		HC32L136K8TA
	HC32L110B4PA				
	HC32L110B6YA				
HC32F03	HC32F030E8PA	**HC32L07**	HC32L072PATA	**HC32F07**	HC32F072PATA
	HC32F030F8UA		HC32L072KATA		HC32F072KATA
	HC32F030F8TA		HC32L072JATA		HC32F072JATA
	HC32F030H8TA		HC32L073PATA		
	HC32F030J8TA		HC32L073KATA		
	HC32F030K8TA		HC32L073JATA		
HC32L17	HC32L176PATA	**HC32F17**	HC32F176PATA	**HC32L19**	HC32L196PCTA
	HC32L176MATA		HC32F176MATA		HC32L196MCTA
	HC32L176KATA		HC32F176KATA		HC32L196KCTA
	HC32L176JATA		HC32F176JATA		HC32L196JCTA
	HC32L170JATA		HC32F170JATA		HC32L190JCTA
	HC32L170FAUA		HC32F170FAUA		HC32L190FCUA
HC32F19	HC32F196PCTA				
	HC32F196MCTA				
	HC32F196KCTA				
	HC32F196JCTA				
	HC32F190JCTA				
	HC32F190FCUA				

content

1 Summary

This application note mainly introduces the FLASH security features, operation instructions and application methods of Huada Semiconductor MCU*.

Notice:

- This application note is a supplementary material for the application of Huada Semiconductor's MCU*, and cannot replace the user manual.

  Please refer to the user manual for the operation of the device and other related matters.

2 Introduction to FLASH

What is **FLASH?**

A type of flash memory device, flash memory is a non-volatile (Non-Volatile) memory, which can also be used without current supply.

Long enough to retain data, its storage characteristics are equivalent to hard disks, and this characteristic is what makes flash memory a variety of portable digital devices

the basis of storage media.

(Quoted from 'Baidu Encyclopedia', 'Interactive Encyclopedia', 'Wikipedia')

FLASH features?

Flash memory is non-volatile memory that can be erased, written and reprogrammed in blocks of memory cells called blocks. Any Flash device

write operations can only be performed in empty or erased cells, so in most cases, a write operation must be performed before

Erase is performed.

FLASH application?

FLASH is widely used in mobile storage, MP3 players, digital cameras, handheld computers and other emerging digital devices.

*

See the cover for supported models.

3 FLASH Application Notes

3.1 Introduction

Huada Semiconductor MCU* covers FLASH of 16/32/64/128/256/512K bytes (Byte) capacity according to different models

Memory, each FLASH is divided according to Sector, and the capacity of each Sector is 512 bytes. This memory supports erasing

Delete (Chip/Page Erase), Program, and Read operations. In addition, this module also supports the protection of FLASH memory erasing and writing, so as to

and write protection of the control registers.

3.2 Security Features

3.2.1 Operational source protection

Huada Semiconductor MCU* adopts a high-security hardware design for FLASH with a capacity of more than 32K, and has a FLASH operating source

Defense function: Only when the address of the FLASH operation function is located at 0~32K, the FLASH erasing and writing operation can be performed correctly.

0~32K of FLASH address has higher security, important functions must be placed in this area, such as important program entry, middle

Break entry function, high security algorithm module, UID, AES, true random number, RTC algorithm cooperation, form a high security authentication system

system.

3.2.2 Operational target protection

The entire 64K byte FLASH memory is divided into 128 pages, each 4 pages share an erase and write protection bit. When the page is protected

At this time, the erasing and writing operations on this page are invalid, and an alarm flag bit and an interrupt signal are generated. When any of the FLASH memory

When the page is protected, the full-chip erasing and writing of the FLASH is invalid, and an alarm flag and an interrupt signal are generated.

3.2.3 PC address erasing protection

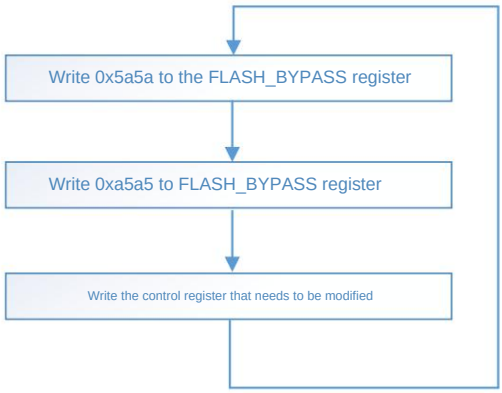
When the CPU runs the program in FLASH, if the current PC pointer falls within the page address range to be erased, then the

Erase and write operations are invalid and generate alarm flags and interrupt signals.

*
See the cover for supported models.

3.2.4 Register write protection

The controller of this module shields the ordinary write operation and must be modified by the write sequence method. The specific operation steps are shown in the following figure:



Notice:

- Write 0x5a5a, 0xa5a5, and write the target register. No write operations (write ROM, write ROM, RAM, REG), otherwise the value of the target register cannot be rewritten. If the rewriting fails, you need to redo these three steps do.

3.3 Function description

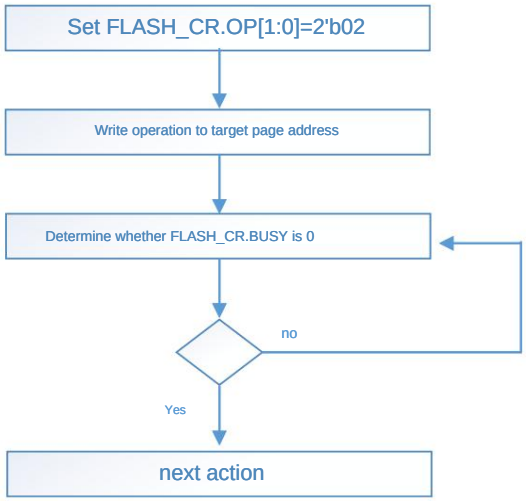
This section introduces the FLASH controller module function, workflow and programming method based on safety features.

This FLASH controller supports three bit-width read and write operations for eFLASH Byte (8bits), Half-word (16bits) and Word (32bits). Note that the address of Byte operation must be aligned by Byte, and the target address of Half-word operation must be aligned by Byte Half-word alignment (the lowest bit of the address is 1'b0), the address of the Word operation must be aligned in Word (the lowest two bits of the address are 2'b00). If the address of the read and write operation is not aligned according to the bit width, the operation is invalid, and the system will enter the Hard Fault Interrupt on error.

3.4 Introduction to Workflow

For detailed operation steps, please refer to the user manual of the corresponding series.

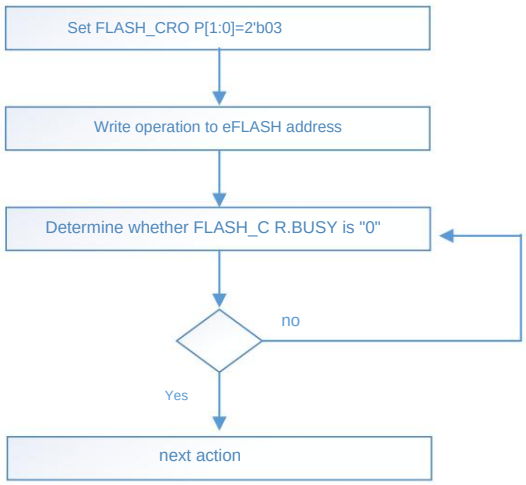
3.4.1 Sector Erase



Note: 1.

The controller ignores the lower 9 bits of the target address as long as the target address falls within the page. The write operation is used to trigger the page erase operation, and the written data will also be ignored by the controller. 2. If the current erase command is executed in eFLASH, the CPU will stop, and the hardware will automatically wait for the BUSY state of eFLASH to end. 3. If the current erase command is executed in RAM, the CPU will not be stopped. Before performing any operation on eFLASH, the software must judge whether the BUSY state of eFLASH is over

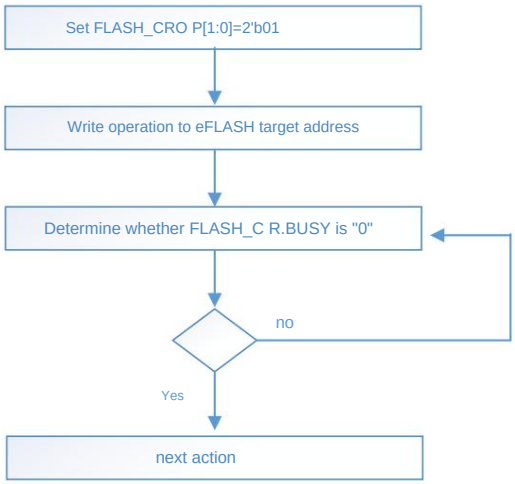
3.4.2 Chip Erase



Note: 1.

The controller ignores the lower 15 bits of the target address as long as the target address falls within the eFLASH address range. 2. The write operation is used to trigger the page erase operation, and the written data will also be ignored by the controller. 3. If the current erase command is executed in eFLASH, the CPU value will stop, and the hardware will automatically wait for the BUSY state of eFLASH to end. 4. If the current erase command is executed in RAM, the CPU value will not be stopped. Before performing any operation on eFLASH, the software must judge whether the BUSY state of eFLASH is over

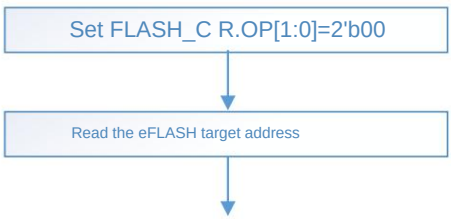
3.4.3 Write Operation



Note: 1.

If the current erase instruction is executed in eFLASH, the CPU value will stop, and the hardware will automatically wait for the BUSY state of eFLASH to end. 2. If the current erase instruction is executed in RAM, the CPU value will not be stopped. , before any operation on eFLASH , the software must judge whether the BUSY state of eFLASH is over

3.4.4 Read Operation



Note: 1.

The step of setting FLASH_C RO P[1:0] in the first step can actually be omitted, no matter what the value of FLASH_C RO P[1:0] is, the read operation can be performed

3.5 Programming method based on FLASH security features

In practical applications, for MCUs with a capacity greater than 32K, if you need to change the FLASH operation function and safety function function

etc. are placed in the 32K security area of FLASH, which can be realized in the following convenient ways.

illustrate:

- This example is for illustration purposes, the main example places the function "Flash_SectorErase()" at the address of the security area "0x400"

In practice, you can replace "example function" and "address" according to your own needs.

3.5.1 Programming method based on Keil MDK

In Keil MDK, the execution address mapping of security functions can be implemented simply as follows:

Add the following code to the declaration of the target function:

```
en_result_t Flash_SectorErase(uint32_t u32SectorAddr) __attribute__((section(".ARM.__at_0x400")));
```

3.5.2 IAR-based programming method

1. Add the following code to the target function definition:

```
en_result_t Flash_SectorErase(uint32_t u32SectorAddr) @".Flash_SectorErase"
```

2. Add the following code to the project "*.icf" file:

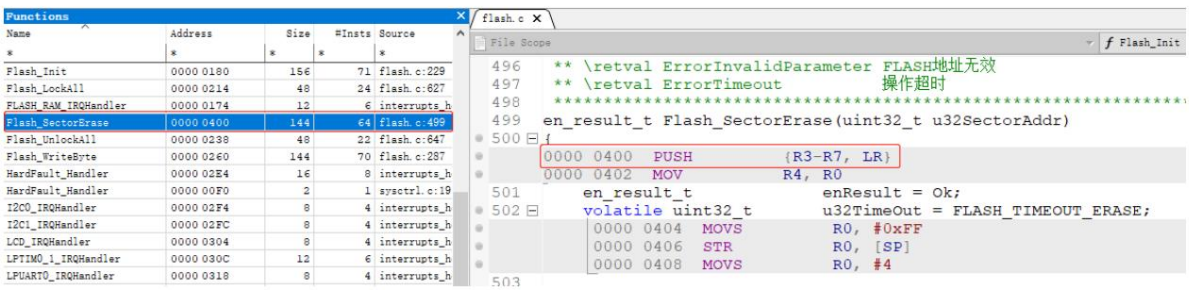
```
place at address mem:0x00000400 { readonly section .Flash_SectorErase};
```

3.5.3 Result View and Example

If you are interested in the specific result information produced by this method, you can pass the map file, debug file or during the debug run

Observe the execution address space of the code.

An example is as follows (it can be confirmed that the target function is indeed placed at the expected [0x400] address):



4 Summary

The above chapters mainly introduce the FLASH security features, workflow and FLASH-based security of Huada Semiconductor MCU*

The programming method of the characteristic, in the actual development, the user can modify or expand according to the above method to meet their own application.

*
See the cover for supported models.

5 Version Information & Contact Information

date	Version revision record
2019/8/19	Rev1.0 initial release

If you have any comments or suggestions in the process of purchasing and using, please feel free to contact us.

Email: mcu@hdsc.com.cn

Website: www.hdsc.com.cn

Mailing address: No. 39, Lane 572, Bibo Road, Zhangjiang Hi-Tech Park, Shanghai

Postcode: 201203