



32-bit microcontroller

HC32L110 / HC32F003 / HC32F005

series of FLASH

Huada MCU exchange group: 164973950

Suitable

series	Product number
HC32L110	HC32L110C6UA
	HC32L110C6PA
	HC32L110C4UA
	HC32L110C4PA
	HC32L110B6PA
	HC32L110B4PA
HC32F003	HC32F003C4UA
	HC32F003C4PA
HC32F005	HC32F005C6UA
	HC32F005C6PA
	HC32F005D6UA

content

1	Summary	3
2	Introduction to FLASH	3
3	FLASH.....	4
3.1	Introduction.....	4
3.2	Description.....	4
3.2.1	Register introduction.....	4
3.2.2	Workflow Introduction.....	5
4	Sample code	7
4.1	Code introduction.....	7
4.2	Code running.....	8
5	Summary	10
6	Version Information & Contact	11

Huada MCU exchange group: 164973950

1 Summary

This application note mainly introduces how to use HC32L110 / HC32F003 / HC32F005 series FLASH to erase, programming and reading.

2 Introduction to FLASH

What is **FLASH**?

A type of flash memory device, flash memory is a non-volatile (Non-Volatile) memory that 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 is non-volatile memory that can be erased, written and reprogrammed in blocks of memory cells called blocks, of any flash device

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

Erase.

FLASH application?

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

3 FLASH

3.1 Introduction

This device contains a 32kByte FLASH memory, which is divided into 64 Sectors, and the capacity of each Sector is 512Byte. This module supports erase, program and read operations of this memory. In addition, this module supports FLASH storage Erase and write protection of memory, and write protection of control registers.

3.2 Description

This section introduces the FLASH controller module of the HC32L110 / HC32F003 / HC32F005 series, including registers and working process.

This FLASH controller supports three eFLASH Byte (8bits), Half-word (16bits), Word (32bits)

Seed bit-width read and write operations. 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.

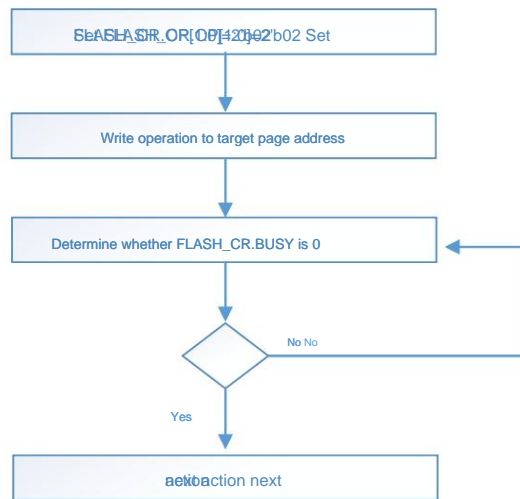
Huada MCU exchange group: 164973950

3.2.1 Register introduction

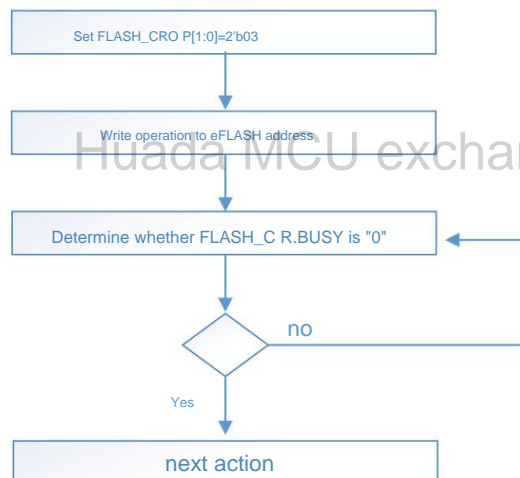
The operation of FLASH is mainly carried out through the following registers:

abbreviation	register name
FLASH_TNVS	Tnvs time parameter
FLASH_TPGS	Tpgs time parameter
FLASH_TPROG	Tprog time parameter
FLASH_TSERASE	Tserase time parameter
FLASH_TMERASE	Tmerase time parameter
FLASH_TPRCV	Tprcv time parameter
FLASH_TSRCV	Tsrcv time parameter
FLASH_TMRCV	Tmrcv time parameter
FLASH_CR	control register
FLASH_IFR	Interrupt Flag Register
FLASH_ICLR	Interrupt Flag Clear Register
FLASH_BYPASS	Bypass sequence register
FLASH_SLOCK	Sector erase and write protection register

3.2.2 Workflow Introduction

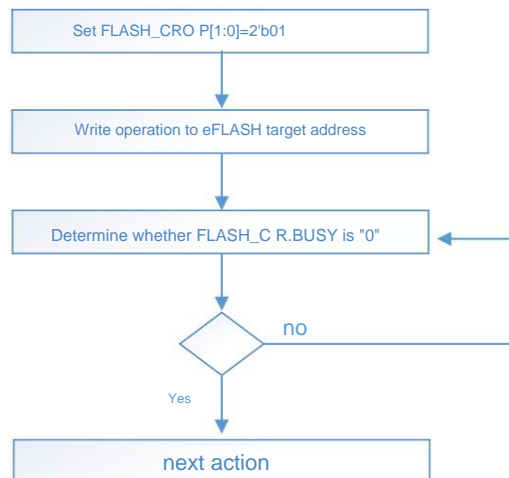
Sector Erase**Note:**

For the sector erase operation, 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.

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.

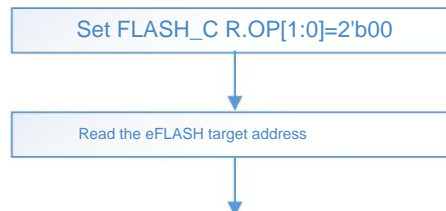
write operation



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

read operation



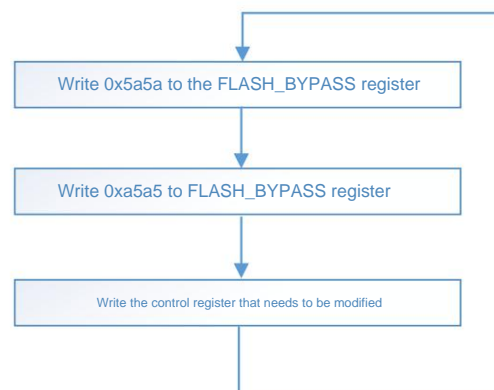
Note: 1.

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

Huada MCU exchange group: 164973950

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:

- No write operation can be inserted between the two operations of writing 0x5a5a and writing 0xa5a5, otherwise the Bypass sequence will be invalid.

The 0x5a5a-0xa5a5 sequence needs to be rewritten.

4 Sample code

4.1 Code introduction

Users can write their own code to learn and verify the module according to the above workflow, or directly through Huada Semiconductor

The sample code downloaded from the website to FLASH is directly encoded and verified using the API functions provided by the FLASH driver library.

application.

The following sections briefly describe the functionality of the various parts of the code:

1) FLASH data declaration and initialization:

```
//FLASH TEST DATA INIT
uint32_t      u32Addr = 0x7000;
uint8_t       u8Data = 0x5a;
uint16_t      u16Data = 0x5a5a;
uint32_t      u32Data = 0x5a5a5a5a;
```

2) FLASH initialization and Sector erasure coding:

```
Flash_Init(FlashInt, 0);

Flash_SectorErase(u32Addr);
```

3) FLASH programming and verification:

```
if (Ok == Flash_WriteByte(u32Addr, u8Data))
{
    if (*(volatile uint8_t*)u32Addr == u8Data) {

        enResult = Ok;

    }
    else
    {
        return enResult;
    }
}
else
{
    return enResult;
}
```

The erasing, programming and reading of FLASH can be completed once through the above code.

4.2 Code running

Users can download the FLASH sample code through the website of Huada Semiconductor, and run the relevant code with the evaluation board to learn

Use the FLASH module.

The following sections describe how to run the FLASH sample code on the evaluation board and observe the results:

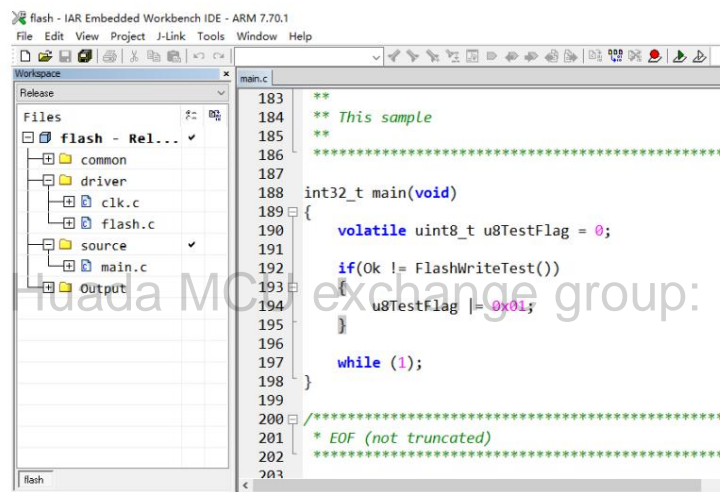
- Make sure to install the correct IAR (or Keil, here IAR is used as a sample description, the operation method is similar in Keil) tool (please


Completely download the corresponding installation package from Huada Semiconductor, and refer to the user manual for installation).


- Download the FLASH sample code from the Huada Semiconductor website.

- Download and run the sample code:

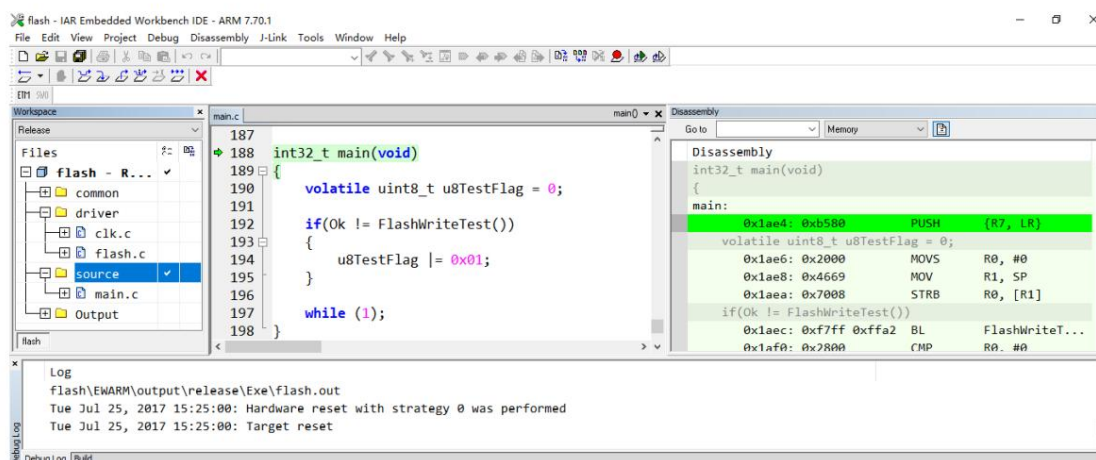
1) Open the FLASH project, and open 'main.c' as shown below:



2) Click  to link the entire project.

3) Click to  download the code to the evaluation board.

4) You can see a view similar to the following:



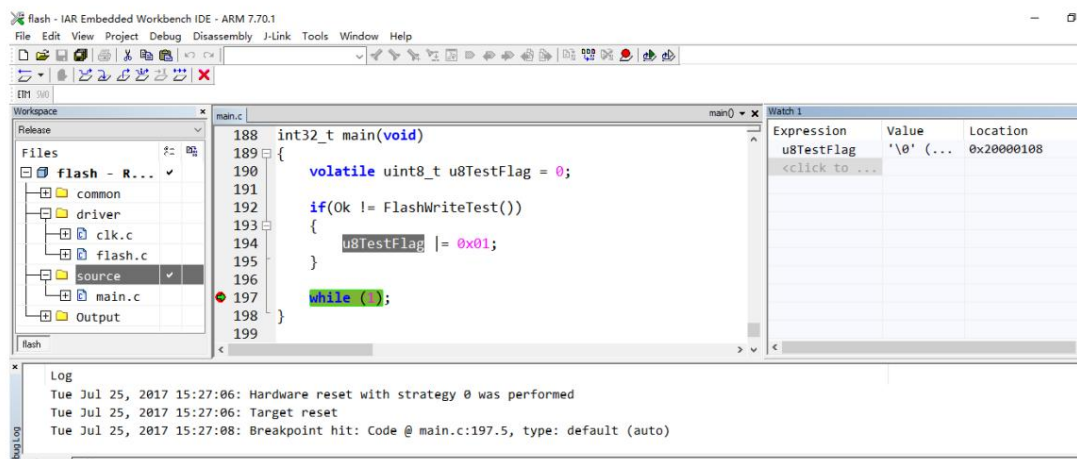
5) Set a breakpoint on the line after 'main(void)':

6) Click "View -> Watch -> Watch1" to open a 'watch1' window, and add the 'u8TestFlag' variable to observe its value.

7) Click Run. 

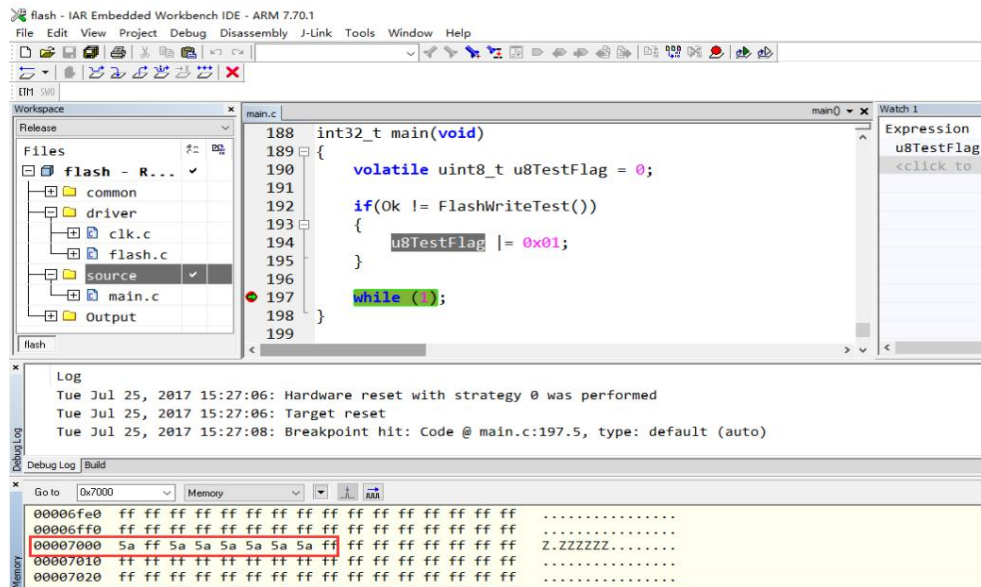
8) The code runs and stops at the breakpoint of 'main(void)', if 'u8TestFlag = 0', it means encoding and verification functions

Correct execution, as shown below.



9) You can also click "View -> Memory" to open a 'Memory' window to observe the data in the programming area, such as

The following figure.



10) You can close the project file after running.

11) Users can also modify the FLASH test data (data and length) and operation address (area) in the code to further

Learn the functions of the FLASH module.

Huada MCU exchange group: 164973950

5 Summary

The above chapters briefly introduced the FLASH of the HC32L110 / HC32F003 / HC32F005 series, and explained in detail

The registers and operation flow of the FLASH module demonstrate how to use the relevant sample code for erasing, programming and reading operations.

In actual development, users can configure and use the erase, program and read functions of FLASH according to their own needs.

Tel: 13840373805

6 Version Information & Contact Information

date	Version revision record
2018/6/4	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: ymcu@hdsc.com.cn

Website: www.hdsc.com.cn

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

Postcode: 201203

