

# FTS512K4

## Block User Guide

### V01.06

Original Release Date: 08 FEB 2001  
Revised: 01 APR 2003

Motorola, Inc

Motorola reserves the right to make changes without further notice to any products herein to improve reliability, function or design. Motorola does not assume any liability arising out of the application or use of any product or circuit described herein; neither does it convey any license under its patent rights nor the rights of others. Motorola products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Motorola product could create a situation where personal injury or death may occur. Should Buyer purchase or use Motorola products for any such unintended or unauthorized application, Buyer shall indemnify and hold Motorola and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Motorola was negligent regarding the design or manufacture of the part.

# Revision History

Version Number	Revision Date	Effective Date	Author	Description of Changes
V01.00	30MAY01	30MAY01		Generated from generic HCS12 block guide(V02.00) Made formats SRS V2 compliant. Remove non-customer information. Reorder and restructure document. Add overview block diagram.
V01.01	19JUL01	19JUL01		Add document names. Hide names and variable definitions.
V01.02	30JAN02	30JAN02		Add description of WRALL bit. Add description of the Address and Data registers. Modify for use of 64Kx16 arrays.
v01.03	23MAR02	23MAR02		Modify FSEC register to include KEYEN[1:0]. Update security restrictions found in <b>4.5 Flash Security</b> : (i) \$0000 and \$FFFF keys are illegal. (ii) No back-to-back writes of keys allowed. (iii) Writing more than 4 keys in a sequence will not unsecure. (iv) Incorrect key sequence results in lock-up with exit by reset only.
V01.04	02AUG02			Modify document number. Fix <b>Table 3-2</b> entry for MCU Address Range \$C000-\$FFFF.
V01.05	02DEC02			Fix bit 15 entry in <b>Figure 3-10</b> .
V01.06	01APR03			Fix sector size in <b>Table 4-1</b> . Modify description of CBEIF and CCIF flags in <b>3.3.6 FSTAT — Flash Status Register</b> . Modify description of <b>3.3.5 FPROT — Flash Protection Register</b> to clarify mass erase restrictions.

# Table of Contents

## Section 1 Introduction

1.1	Overview	9
1.1.1	Glossary	9
1.2	Features	9
1.3	Modes of Operation	10
1.4	Block Diagram	11

## Section 2 External Signal Description

2.1	Overview	13
-----	----------	----

## Section 3 Memory Map and Registers

3.1	Overview	15
3.2	Modules Memory Map	15
3.3	Register Descriptions	21
3.3.1	FCLKDIV — Flash Clock Divider Register	21
3.3.2	FSEC — Flash Security Register	21
3.3.3	FTSTMOD — Flash Test Mode Register	23
3.3.4	FCNFG — Flash Configuration Register	23
3.3.5	FPROT — Flash Protection Register	24
3.3.6	FSTAT — Flash Status Register	26
3.3.7	FCMD — Flash Command Register	28
3.3.8	RESERVED1	28
3.3.9	FADDR — Flash Address Register	29
3.3.10	FDATA — Flash Data Register	29
3.3.11	RESERVED2	30
3.3.12	RESERVED3	30
3.3.13	RESERVED4	31
3.3.14	RESERVED5	31

## Section 4 Functional Description

4.1	Program and Erase Operation	33
4.1.1	Writing the FCLKDIV Register	33
4.1.2	Program and Erase Sequences in Normal Mode	36

- 4.1.3 Valid Flash Commands .....38
- 4.1.4 Illegal Flash Operations .....38
- 4.2 Wait Mode .....39
- 4.3 Stop Mode .....39
- 4.4 Background Debug Mode.....40
- 4.5 Flash Security.....40
- 4.5.1 Unsecuring via the Backdoor Key Access .....40

**Section 5 Resets**

- 5.1 General.....43

**Section 6 Interrupts**

- 6.1 General.....45
- 6.2 Description of Interrupt Operation .....45

# List of Figures

Figure 1-1	Module Block Diagram. . . . .	11
Figure 3-1	Flash Memory Map . . . . .	16
Figure 3-2	Flash Clock Divider Register (FCLKDIV). . . . .	21
Figure 3-3	Flash Security Register (FSEC). . . . .	21
Figure 3-4	Flash Test Mode Register (FTSTMOD). . . . .	23
Figure 3-5	Flash Configuration Register (FCNFG) . . . . .	23
Figure 3-6	Flash Protection Register (FPROT). . . . .	24
Figure 3-7	Flash Status Register (FSTAT) . . . . .	26
Figure 3-8	Flash Command Buffer and Register (FCMD). . . . .	28
Figure 3-9	RESERVED1. . . . .	28
Figure 3-10	Flash Address High Register (FADDRHI) . . . . .	29
Figure 3-11	Flash Address Low Register (FADDRLO) . . . . .	29
Figure 3-12	Flash Data High Register (FDATAHI) . . . . .	29
Figure 3-13	Flash Data Low Register (FDATALO) . . . . .	30
Figure 3-14	RESERVED2. . . . .	30
Figure 3-15	RESERVED3. . . . .	30
Figure 3-16	RESERVED4. . . . .	31
Figure 3-17	RESERVED5. . . . .	31
Figure 4-1	PRDIV8 and FDIV bits Determination Procedure . . . . .	35
Figure 4-2	Example Program Algorithm . . . . .	37
Figure 6-1	Flash Interrupt Implementation . . . . .	46



# List of Tables

Table 3-1 Flash Protection/Options Field . . . . .15

Table 3-2 Flash Memory Map Summary . . . . .17

Table 3-3 Flash Register Memory Map . . . . .20

Table 3-4 Flash KEYEN States . . . . .22

Table 3-5 Flash Security States . . . . .22

Table 3-6 Flash Register Bank Selects . . . . .24

Table 3-7 Loading of the Protection Register from Flash. . . . .25

Table 3-8 Flash Higher Address Range Protection . . . . .25

Table 3-9 Flash Lower Address Range Protection . . . . .26

Table 3-10 Flash Normal Mode Commands . . . . .28

Table 4-1 Valid Flash Commands . . . . .38

Table 6-1 Flash Interrupt Sources . . . . .45



# Section 1 Introduction

## 1.1 Overview

This document describes the FTS512K4 module which is a 512K byte Flash (Non-Volatile) memory. The Flash memory contains 4 blocks of 128K bytes with each block organized as 1024 rows of 128 bytes. The Flash block's erase sector size is 8 rows (1024 bytes).

The Flash memory may be read as either bytes, aligned words or misaligned words. Read access time is one bus cycle for byte and aligned word, and two bus cycles for misaligned words.

Program and erase functions are controlled by a command driven interface. Both sector erase and mass erase of an entire 128K byte Flash block are supported. An erased bit reads '1' and a programmed bit reads '0'. The high voltage required to program and erase is generated internally by on-chip charge pumps.

All Flash blocks can be programmed or erased at the same time. However, it is not possible to read from a Flash block while it is being erased or programmed.

The Flash memory is ideal for program and data storage for single-supply applications allowing for field reprogramming without requiring external programming voltage sources.

---

### WARNING

**A word must be erased before being programmed. Cumulative programming of bits within a word is not allowed.**

---

### 1.1.1 Glossary

#### ***Banked Register***

A register operating on one Flash block which shares the same register address as the equivalent registers for the other Flash blocks. The active register bank is selected by two bank-select bits in the unbanked register space.

#### ***Common Register***

A register which operates on all Flash blocks.

#### ***Command Sequence***

A three-step MCU instruction sequence to program, erase or erase-verify a Flash block.

## 1.2 Features

- 512K bytes of flash memory comprising four 128k byte blocks.
- Each block in the Flash module can be read, programmed or erased concurrently.

- Automated program and erase algorithm.
- Interrupts on Flash command completion and command buffer empty.
- Fast sector erase and word program operation.
- 2-stage command pipeline.
- Flexible protection scheme for protection against accidental program or erase.
- Single power supply program and erase.
- Security feature.

## 1.3 Modes of Operation

- Program and erase operation (please refer to **4.1** for details).

# 1.4 Block Diagram

Figure 1-1 shows a block diagram of the FTS512K4 module.

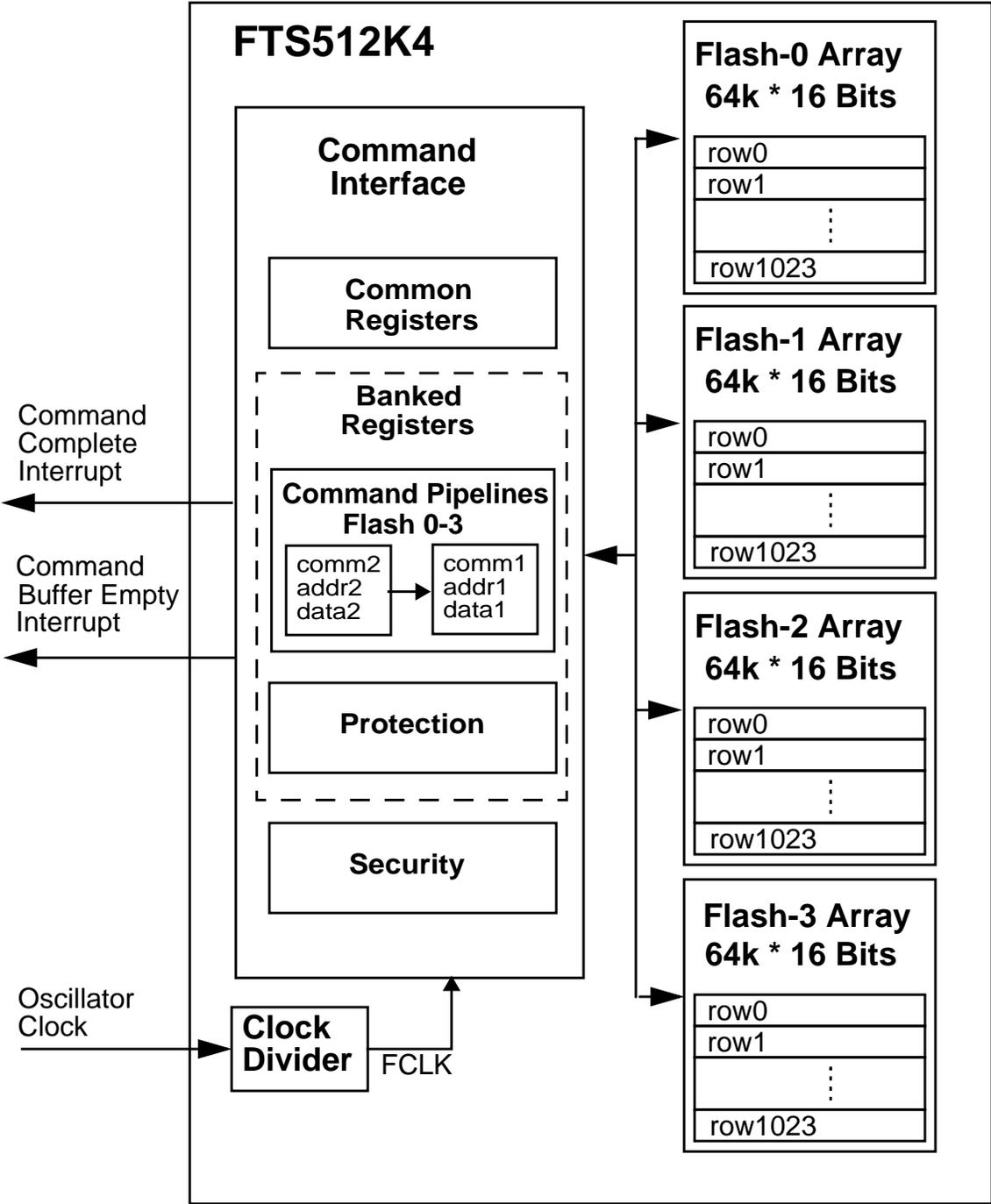


Figure 1-1 Module Block Diagram



## Section 2 External Signal Description

### 2.1 Overview

The FTS512K4 module contains no signals that connect off-chip.



## Section 3 Memory Map and Registers

### 3.1 Overview

This section describes the FTS512K4 memory map and registers

### 3.2 Modules Memory Map

**Figure 3-1** shows the FTS512K4 memory map. The HCS12 architecture places the Flash array addresses between \$4000 and \$FFFF, which corresponds to three 16K byte pages. The content of the HCS12 Core PPAGE register is used to map the logical middle page ranging from address \$8000 to \$BFFF to any physical 16K byte page in the physical memory.<sup>1</sup> Shown within the pages are a protection/options field, described in **Table 3-1**, and user defined Flash protected sectors, described in **Table 3-2**.

The FPOPEN bit in the FPROT register (see **3.3.5**) can globally protect the entirety of the corresponding Flash block. However, for all Flash blocks, two protected areas, one starting from the Flash block starting address (called lower) towards higher addresses and the other one growing downward from the Flash block end address (called higher) can be activated. For Flash block 0, the higher page is mainly targeted to hold the boot loader code since it covers the vector space.

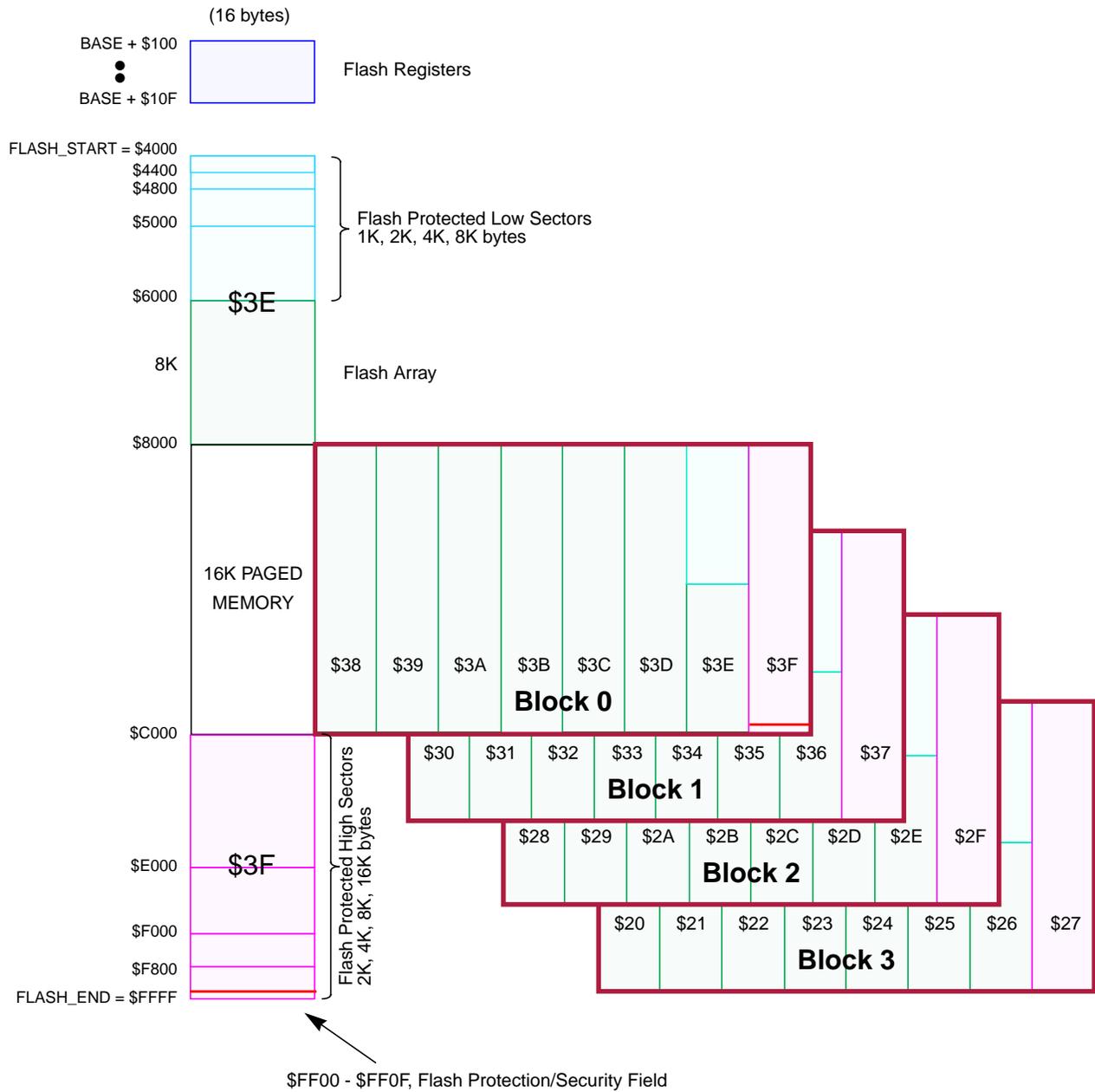
Security information that allows the MCU to prevent intrusive access to the Flash module is stored in the Flash Protection/Options field of Flash block 0 as described in **Table 3-1**.

**Table 3-1 Flash Protection/Options Field**

Array Address	Size (bytes)	Description
\$FF00 - \$FF07	8	Backdoor Comparison Keys
\$FF08 - \$FF09	2	Reserved
\$FF0A	1	Block 3 Flash Protection byte Refer to Section <b>3.3.5</b>
\$FF0B	1	Block 2 Flash Protection byte Refer to Section <b>3.3.5</b>
\$FF0C	1	Block 1 Flash Protection byte Refer to Section <b>3.3.5</b>
\$FF0D	1	Block 0 Flash Protection byte Refer to Section <b>3.3.5</b>
\$FF0E	1	Reserved
\$FF0F	1	Flash Options/Security byte Refer to Section <b>3.3.2</b>

NOTES:

1. By placing \$3F or \$3E in the PPAGE register, the bottom respectively top “fixed” 16Kbytes pages can be seen twice in the MCU memory map.



Note: \$20-\$3F correspond to the PPAGE register content

Figure 3-1 Flash Memory Map

**Table 3-2 Flash Memory Map Summary**

MCU Address Range	PPAGE	Protectable Low Range	Protectable High Range	Flash Block	Block Relative Address <sup>1</sup>
\$4000-\$7FFF	Unpaged (\$3E)	\$4000-\$43FF \$4000-\$47FF \$4000-\$4FFF \$4000-\$5FFF	N.A.	0	\$18000-\$1BFFF
\$8000-\$BFFF	\$20	N.A.	N.A.	3	\$00000-\$03FFF
	\$21	N.A.	N.A.		\$04000-\$07FFF
	\$22	N.A.	N.A.		\$08000-\$0BFFF
	\$23	N.A.	N.A.		\$0C000-\$0FFFF
	\$24	N.A.	N.A.		\$10000-\$13FFF
	\$25	N.A.	N.A.		\$14000-\$17FFF
	\$26	\$8000-\$83FF \$8000-\$87FF \$8000-\$8FFF \$8000-\$9FFF	N.A.		\$18000-\$1BFFF
	\$27	N.A.	\$B800-\$BFFF \$B000-\$BFFF \$A000-\$BFFF \$8000-\$BFFF		\$1C000-\$1FFFF
\$8000-\$BFFF	\$28	N.A.	N.A.	2	\$00000-\$03FFF
	\$29	N.A.	N.A.		\$04000-\$07FFF
	\$2A	N.A.	N.A.		\$08000-\$0BFFF
	\$2B	N.A.	N.A.		\$0C000-\$0FFFF
	\$2C	N.A.	N.A.		\$10000-\$13FFF
	\$2D	N.A.	N.A.		\$14000-\$17FFF
	\$2E	\$8000-\$83FF \$8000-\$87FF \$8000-\$8FFF \$8000-\$9FFF	N.A.		\$18000-\$1BFFF
	\$2F	N.A.	\$B800-\$BFFF \$B000-\$BFFF \$A000-\$BFFF \$8000-\$BFFF		\$1C000-\$1FFFF

**Table 3-2 Flash Memory Map Summary**

MCU Address Range	PPAGE	Protectable Low Range	Protectable High Range	Flash Block	Block Relative Address <sup>1</sup>
\$8000-\$BFFF	\$30	N.A.	N.A.	1	\$00000-\$03FFF
	\$31	N.A.	N.A.		\$04000-\$07FFF
	\$32	N.A.	N.A.		\$08000-\$0BFFF
	\$33	N.A.	N.A.		\$0C000-\$0FFFF
	\$34	N.A.	N.A.		\$10000-\$13FFF
	\$35	N.A.	N.A.		\$14000-\$17FFF
	\$36	\$8000-\$83FF \$8000-\$87FF \$8000-\$8FFF \$8000-\$9FFF	N.A.		\$18000-\$1BFFF
	\$37	N.A.	\$B800-\$BFFF \$B000-\$BFFF \$A000-\$BFFF \$8000-\$BFFF		\$1C000-\$1FFFF
\$8000-\$BFFF	\$38	N.A.	N.A.	0	\$00000-\$03FFF
	\$39	N.A.	N.A.		\$04000-\$07FFF
	\$3A	N.A.	N.A.		\$08000-\$0BFFF
	\$3B	N.A.	N.A.		\$0C000-\$0FFFF
	\$3C	N.A.	N.A.		\$10000-\$13FFF
	\$3D	N.A.	N.A.		\$14000-\$17FFF
	\$3E	\$8000-\$83FF \$8000-\$87FF \$8000-\$8FFF \$8000-\$9FFF	N.A.		\$18000-\$1BFFF
	\$3F	N.A.	\$B800-\$BFFF \$B000-\$BFFF \$A000-\$BFFF \$8000-\$BFFF		\$1C000-\$1FFFF
\$C000-\$FFFF	Unpaged (\$3F)	N.A.	\$F800-\$FFFF \$F000-\$FFFF \$E000-\$FFFF \$C000-\$FFFF	0	\$1C000-\$1FFFF

NOTES:

1. Inside each Flash block of size 128 Kbyte.

The Flash module also contains a set of 16 control and status registers located in address space BASE + \$100 to BASE + \$10F. In order to accommodate four Flash blocks with a minimum register address space, a set of registers (BASE+\$104 to BASE+\$10B) is duplicated in four banks. The active bank is selected by the BKSEL bits in the unbanked Flash Configuration Register (FCNFG). A summary of these registers is given in **Table 3-3**.

**Table 3-3 Flash Register Memory Map**

Address Offset	Use	Access
\$_00	Flash Clock Divider Register (FCLKDIV)	R/W
\$_01	Flash Security Register (FSEC)	R
\$_02	Flash Test Mode Register (FTSTMOD) <sup>1</sup>	R
\$_03	Flash Configuration Register (FCNFG)	R/W
\$_04	Flash Protection Register (FPROT)	R/W
\$_05	Flash Status Register (FSTAT)	R/W
\$_06	Flash Command Register (FCMD)	R/W
\$_07	RESERVED1 <sup>1</sup>	R
\$_08	Flash High Address Register (FADDRHI) <sup>1</sup>	R
\$_09	Flash Low Address Register (FADDRLO) <sup>1</sup>	R
\$_0A	Flash High Data Register (FDATAHI) <sup>1</sup>	R
\$_0B	Flash Low Data Register (FDATALO) <sup>1</sup>	R
\$_0C	RESERVED2 <sup>1</sup>	R
\$_0D	RESERVED3 <sup>1</sup>	R
\$_0E	RESERVED4 <sup>1</sup>	R
\$_0F	RESERVED5 <sup>1</sup>	R

NOTES:

1. Intended for factory test purposes only.

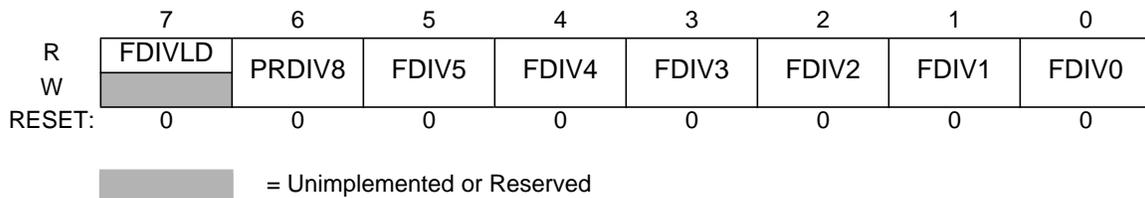
**NOTE:** *Register Address = Register Base Address + \$100 + Address Offset, where the Register Base Address is defined by the HCS12 Core INTRG register and the Address Offset is defined by the Flash module.*

### 3.3 Register Descriptions

#### 3.3.1 FCLKDIV — Flash Clock Divider Register

The unbanked FCLKDIV register is used to control timed events in program and erase algorithms.

Register address **BASE + \$100**



**Figure 3-2 Flash Clock Divider Register (FCLKDIV)**

All bits in the FCLKDIV register are readable, bits 6-0 are write once and bit 7 is not writable

**FDIVLD** — Clock Divider Loaded

- 1 = Register has been written to since the last reset
- 0 = Register has not been written

**PRDIV8** — Enable Prescaler by 8

- 1 = Enables a prescaler by 8, to divide the Flash module input oscillator clock before feeding into the CLKDIV divider
- 0 = The input oscillator clock is directly fed into the FCLKDIV divider

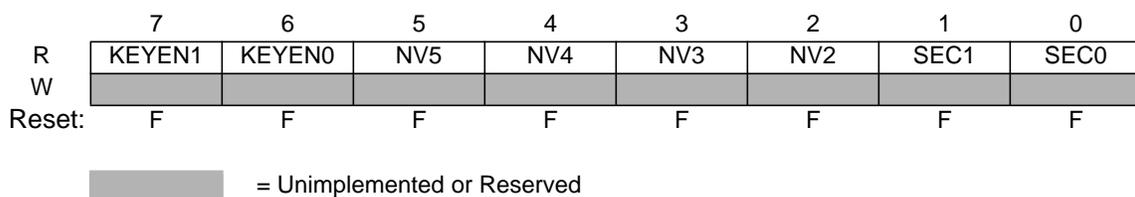
**FDIV[5:0]** — Clock Divider Bits

The combination of PRDIV8 and FDIV[5:0] effectively divides the Flash module input oscillator clock down to a frequency of 150kHz - 200kHz The maximum divide ratio is 512 Please refer to section 4.1.1 for more information

#### 3.3.2 FSEC — Flash Security Register

This unbanked FSEC register holds all bits associated with the security of the MCU.

Register address **BASE + \$101**



**Figure 3-3 Flash Security Register (FSEC)**

All bits in the FSEC register are readable but not writable.

The FSEC register is loaded from the Flash Protection/Options field byte at \$FF0F during the reset sequence, indicated by “F” in **Figure 3-3**

KEYEN[1:0]— Backdoor Key Security Enable Bits.

The KEYEN[1:0] bits define the enabling of the Backdoor Key Access to the Flash module as shown in **Table 3-4**

**Table 3-4 Flash KEYEN States**

KEYEN[1:0]	Description
00	Backdoor Key Access to Flash module DISABLED
01	Backdoor Key Access to Flash module DISABLED
10	Backdoor Key Access to Flash module <b>ENABLED</b>
11	Backdoor Key Access to Flash module DISABLED

NV[5:2] — Non-Volatile Flag Bits

These 4 bits are available to the user as non-volatile flags

SEC[1:0] — Flash Security Bits

The SEC[1:0] bits define the security state of the MCU as shown in **Table 3-5** If the Flash module is unsecured using the Backdoor Key Access, the SEC bits are forced to “10”.

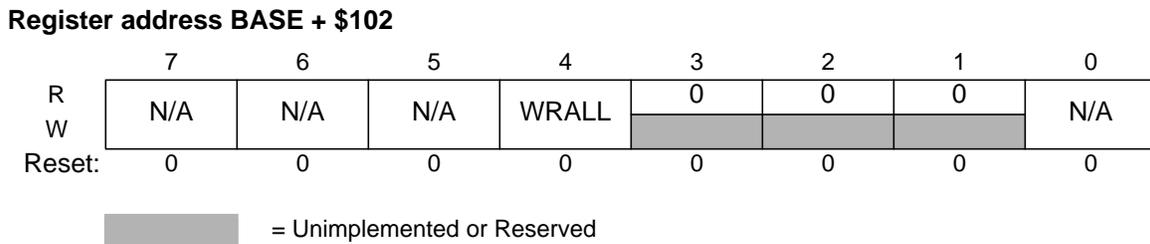
**Table 3-5 Flash Security States**

SEC[1:0]	Description
00	secured
01	secured
10	unsecured
11	secured

The security function in the Flash module is described in section **4.5**.

### 3.3.3 FTSTMOD — Flash Test Mode Register

The unbanked FTSTMOD register is used primarily to control the Flash Special modes.



**Figure 3-4 Flash Test Mode Register (FTSTMOD)**

In normal modes, all bits in the FTSTMOD register read zero and are not writable. The WRALL bit is writable only in special modes. The purpose of this bit is to launch a command on all blocks in parallel. This can be useful for mass erase and erase verify operations. All other bits in this register must be written to zero at all times.

WRALL — Write to all register banks.

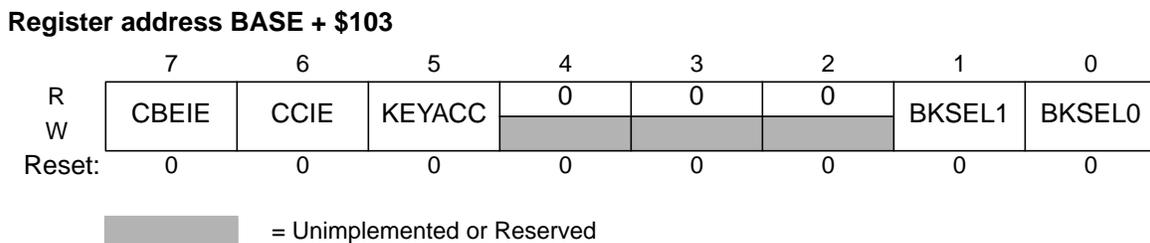
If this bit is set, all banked registers sharing the same address will be written simultaneously.

1 = Write to all register banks.

0 = Write only to the bank selected via BKSEL.

### 3.3.4 FCNFG — Flash Configuration Register

The unbanked FCNFG register enables the Flash interrupts, gates the security backdoor writes and selects the register bank to be operated on.



**Figure 3-5 Flash Configuration Register (FCNFG)**

CBEIE, CCIE, KEYACC, BKSEL1 and BKSEL0 are readable and writable. Bits 4-2 read zero and are not writable.

CBEIE — Command Buffer Empty Interrupt Enable.

The CBEIE bit enables the interrupts in case of an empty command buffer in the Flash module.

1 = An interrupt will be requested whenever the CBEIF flag, **Figure 3-7**, is set.

0 = Command Buffer Empty interrupts disabled.

CCIE — Command Complete Interrupt Enable.

The CCIE bit enables the interrupts in case of all commands being completed in the Flash module.

1 = An interrupt will be requested whenever the CCIF, **Figure 3-7**, flag is set.

0 = Command Complete interrupts disabled.

KEYACC — Enable Security Key Writing.

1 = Writes to Flash array are interpreted as keys to open the backdoor. Reads of the Flash array return invalid data.

0 = Flash module writes are interpreted as the start of a program or erase sequence.

BKSEL[1:0] — Register Bank Select.

These bits are used to select one of the four register banks. The register bank associated with Flash block 0 is the default out of reset. The bank selection is according to **Table 3-6**.

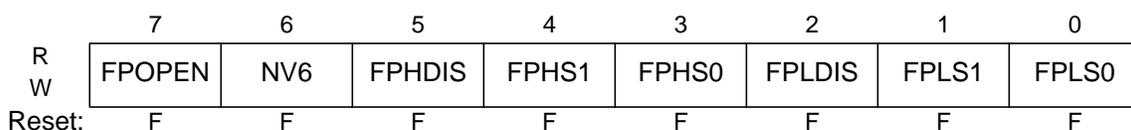
**Table 3-6 Flash Register Bank Selects**

BKSEL[1:0]	Selected Register Bank
00	Flash 0
01	Flash 1
10	Flash 2
11	Flash 3

### 3.3.5 FPROT — Flash Protection Register

The banked FPROT register defines which Flash sectors are protected against program or erase.

Register address **BASE + \$104**



 = Unimplemented or Reserved

**Figure 3-6 Flash Protection Register (FPROT)**

The FPROT register is readable in normal and special modes. Bit NV6 is not writable. FPOPEN, FPHDIS and FPLDIS bits in the FPROT register can only be written to the protected state (i.e. 0). FPLS[1:0] can be written anytime until bit FPLDIS is cleared. FPHS[1:0] bits can be written anytime until bit FPHDIS is

cleared. If the FPOPEN bit is cleared, then the state of the FPHDIS, FPHS[1:0], FPLDIS and FPLS[1:0] bits is irrelevant. The FPROT register is loaded from Flash block 0 during reset as shown in **Table 3-7**.

**Table 3-7 Loading of the Protection Register from Flash**

Flash Address	Protection byte for
\$FF0D	Flash 0
\$FF0C	Flash 1
\$FF0B	Flash 2
\$FF0A	Flash 3

To change the Flash protection that will be loaded on reset, the upper sector of Flash block 0 must be unprotected, then the Flash Protect/Security byte located as described in **Table 3-1** must be written.

A protected Flash sector is disabled by the bits FPHDIS and FPLDIS while the size of the protected sector is defined by FPHS[1:0] and FPLS[1:0] in the FPROT register.

Trying to alter any of the protected areas will result in a protect violation error and bit PVIOL will be set in the Flash Status Register (FSTAT). A mass erase of a whole Flash block is only possible when protection is fully disabled by setting the FPOPEN, FPLDIS, and FPHDIS bits. An attempt to mass erase a Flash block while protection is enabled in that block will set the PVIOL bit in the FSTAT register.

FPOPEN — Opens the Flash array for program or erase.

1 = The Flash sectors not protected are enabled for program or erase.

0 = The whole Flash array is protected. In this case the FPHDIS, FPHS[1:0], FPLDIS and FPLS[1:0] bits within the protection register are ignored.

FPHDIS — Flash Protection Higher address range Disable.

The FPHDIS bit determines whether there is a protected area in the higher space of the Flash block.

1 = Protection disabled.

0 = Protection enabled.

FPHS[1:0] — Flash Protection Higher Address Size.

The FPHS[1:0] bits determine the size of the protected sector. Refer to **Table 3-8**.

**Table 3-8 Flash Higher Address Range Protection**

FPHS[1:0]	Protected Address Range	Protected Size
00	see <b>Table 3-2</b>	2K bytes
01		4K
10		8K
11		16K

FPLDIS — Flash Protection Lower address range Disable.

The FPLDIS bit determines whether there is a protected sector in the lower space of the Flash block.

1 = Protection disabled.

0 = Protection enabled.

FPLS[1:0] — Flash Protection Lower Address Size.

The FPLS[1:0] bits determine the size of the protected sector. Refer to **Table 3-9**.

**Table 3-9 Flash Lower Address Range Protection**

FPLS[1:0]	Protected Address Range	Protected Size
00	see <b>Table 3-2</b>	1K Bytes
01		2K
10		4K
11		8K

NV6 — Non-Volatile Flag Bit.

The NV6 bit should remain in the erased state “1” for future enhancements.

### 3.3.6 FSTAT — Flash Status Register

The banked FSTAT register defines the Flash state machine command status and Flash array access, protection and erase verify status.

Register address **BASE + \$105**



**Figure 3-7 Flash Status Register (FSTAT)**

Register bits CBEIF, PVIOL and ACCERR are readable and writable, bits CCIF and BLANK are readable and not writable, bits 3, 1 and 0 read zero and are not writable.

CBEIF — Command Buffer Empty Interrupt Flag.

The CBEIF flag indicates that the address, data and command buffers are empty so that a new command sequence can be started. The CBEIF flag is cleared by writing a “1” to CBEIF. Writing a “0” to the CBEIF flag has no effect on CBEIF. Writing a “0” to CBEIF after writing an aligned word to the Flash address space but before CBEIF is cleared will abort a command sequence and cause the

ACCERR flag in the FSTAT register to be set. Writing a "0" to CBEIF outside of a command sequence will not set the ACCERR flag. The CBEIF flag is used together with the CBEIE bit in the FCNFG register to generate an interrupt request (see also **Figure 6-1**).

- 1 = Buffers are ready to accept a new command.
- 0 = Buffers are full.

#### CCIF — Command Complete Interrupt Flag.

The CCIF flag indicates that there are no more commands pending. The CCIF flag is cleared when CBEIF is clear and sets automatically upon completion of all active and pending commands. The CCIF flag does not set when an active commands completes and a pending command is fetched from the command buffer. Writing to the CCIF flag has no effect. The CCIF flag is used together with the CCIE bit in the FCNFG register to generate an interrupt request (see also **Figure 6-1**).

- 1 = All commands are completed.
- 0 = Command in progress.

#### PVIOL — Protection Violation.

The PVIOL flag indicates an attempt was made to program or erase an address in a protected Flash memory area. The PVIOL flag is cleared by writing a "1" to PVIOL. Writing a "0" to the PVIOL flag has no effect on PVIOL. While PVIOL is set in any of the FSTAT registers, it is not possible to launch another command in any of the Flash blocks.

- 1 = A protection violation has occurred.
- 0 = No failure.

#### ACCERR — Flash Access Error.

The ACCERR flag indicates an illegal access to the selected Flash block caused by either a violation of the command sequence, issuing an illegal command (illegal combination of the CMDDBx bits in the FCMD register) or the execution of a CPU STOP instruction while a command is executing (CCIF=0). The ACCERR flag is cleared by writing a "1" to ACCERR. Writing a "0" to the ACCERR flag has no effect on ACCERR. While ACCERR is set in any of the FSTAT registers, it is not possible to launch another command in any of the Flash blocks.

- 1 = Access error has occurred.
- 0 = No failure.

#### BLANK — Array has been verified as erased.

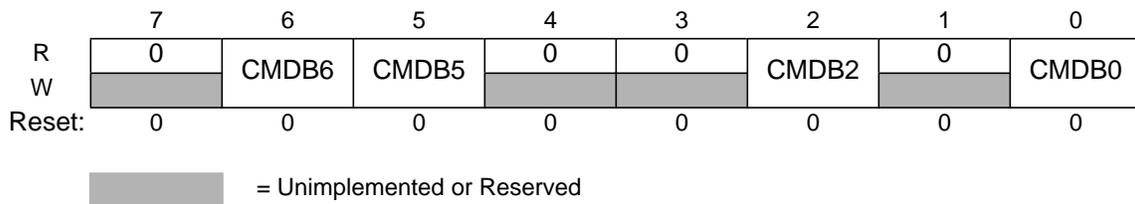
The BLANK flag indicates that an erase verify command has checked the Flash block and found it to be erased. The BLANK flag is cleared by hardware when CBEIF is cleared as part of a new valid command sequence. Writing to the BLANK flag has no effect on BLANK.

- 1 = Flash block verifies as erased.
- 0 = If an erase verify command has been requested, and the CCIF flag is set, then a zero in BLANK indicates the block is not erased.

### 3.3.7 FCMD — Flash Command Register

The banked FCMD register defines the Flash commands.

Register address **BASE + \$106**



**Figure 3-8 Flash Command Buffer and Register (FCMD)**

Bits 7, 4, 3 and 1 read zero and are not writable. Bits CMDB6, CMDB5, CMDB2 and CMDB0 are readable and writable during a command sequence.

CMDB — Valid normal mode commands are shown in **Table 3-10**. Any commands other than those mentioned in **Table 3-10** sets the ACCERR bit in the FSTAT register (**3.3.6**).

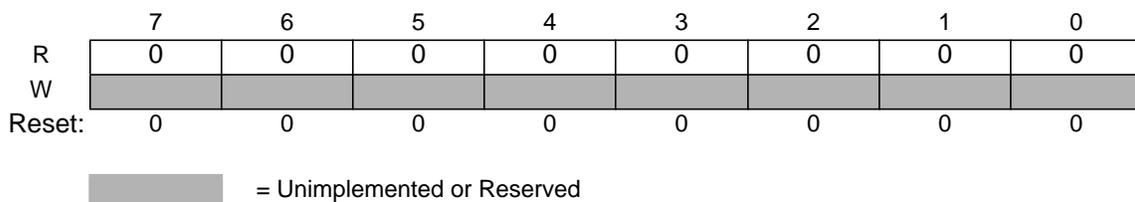
**Table 3-10 Flash Normal Mode Commands**

Command	Meaning
\$05	Erase Verify
\$20	Word Program
\$40	Sector Erase
\$41	Mass Erase

### 3.3.8 RESERVED1

This register is reserved for factory testing and is not accessible to the user.

Register address **BASE + \$107**



**Figure 3-9 RESERVED1**

All bits read zero and are not writable.

### 3.3.9 FADDR — Flash Address Register

FADDRHI and FADDRLO are the banked Flash address registers.

Register address Base + \$108

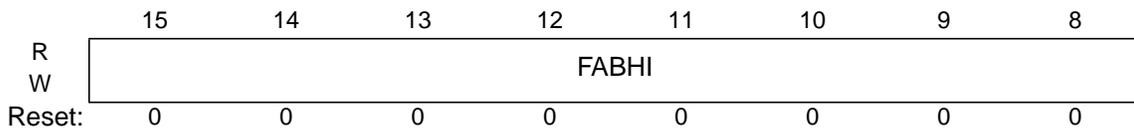


Figure 3-10 Flash Address High Register (FADDRHI)

Register address Base + \$109

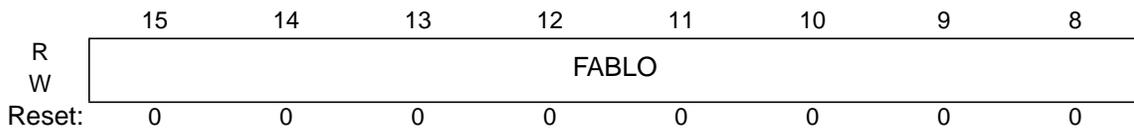


Figure 3-11 Flash Address Low Register (FADDRLO)

In normal modes, the FADDR (FADDRHI, FADDRLO) register reads zeros and is not writable.

The FADDRHI and FADDRLO registers can be written in special modes by writing to address BASE + \$108 and BASE + \$109 in the register space.

For sector erase, the MCU address bits AB[9:0] are ignored.

For mass erase, any address within the block is valid to start the command.

### 3.3.10 FDATA — Flash Data Register

FDATAHI and FDATALO are the banked Flash data registers.

Register address BASE + \$10A

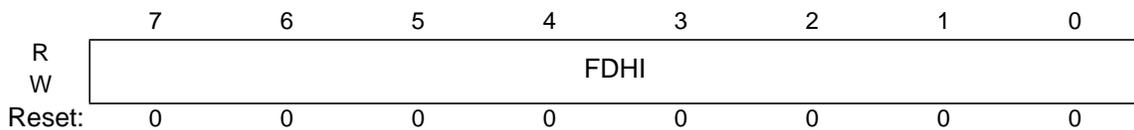
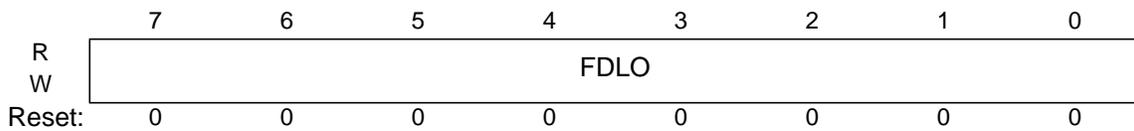


Figure 3-12 Flash Data High Register (FDATAHI)

Register address **BASE + \$10B**



**Figure 3-13 Flash Data Low Register (FDATALO)**

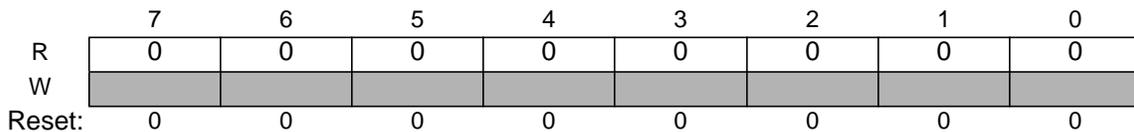
In normal modes, all FDATA bits read zero and are not writable.

In special modes, all FDATA bits are readable and writable when writing to an address within the Flash address range.

### 3.3.11 RESERVED2

This register is reserved for factory testing and is not accessible to the user.

Register address **BASE + \$10C**



= Unimplemented or Reserved

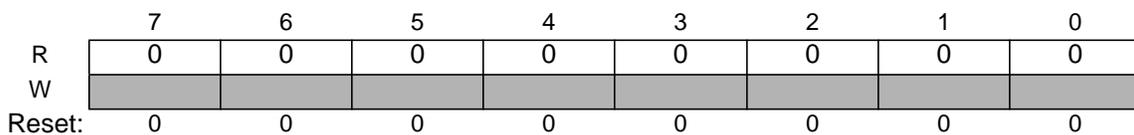
**Figure 3-14 RESERVED2**

All bits read zero and are not writable.

### 3.3.12 RESERVED3

This register is reserved for factory testing and is not accessible to the user.

Register address **BASE + \$10D**



= Unimplemented or Reserved

**Figure 3-15 RESERVED3**

All bits read zero and are not writable.

### 3.3.13 RESERVED4

This register is reserved for factory testing and is not accessible to the user.

Register address **BASE + \$10E**

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

**Figure 3-16 RESERVED4**

All bits read zero and are not writable.

### 3.3.14 RESERVED5

This register is reserved for factory testing and is not accessible to the user.

Register address **BASE + \$10F**

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

**Figure 3-17 RESERVED5**

All bits read zero and are not writable.



## Section 4 Functional Description

### 4.1 Program and Erase Operation

Write and read operations are both used for the program and erase algorithms described in this section. These algorithms are controlled by a state machine whose timebase FCLK is derived from the oscillator clock via a programmable divider. The command register as well as the associated address and data registers operate as a buffer and a register (2-stage FIFO) so that a new command along with the necessary data and address can be stored to the buffer while the previous command is still in progress. This pipelined operation allows a time optimization when programming more than one word on a specific row, as the high voltage generation can be kept ON in between two programming commands. The pipelined operation also allows a simplification of command launching. Buffer empty as well as command completion are signalled by flags in the Flash status register. Interrupts for the Flash will be generated if enabled.

The next four subsections describe:

- How to write the FCLKDIV register.
- The write sequences used to program, erase and erase-verify the Flash.
- Valid Flash commands.
- Errors resulting from illegal Flash operations.

#### 4.1.1 Writing the FCLKDIV Register

Prior to issuing any program or erase command, it is first necessary to write the FCLKDIV register to divide the oscillator down to within the 150kHz to 200kHz range. The program and erase timings are also a function of the bus clock, such that the FCLKDIV determination must take this information into account. If we define:

- FCLK as the clock of the Flash timing control block
- Tbus as the period of the bus clock
- INT(x) as taking the integer part of x (e.g. INT(4.323)=4),

then FCLKDIV register bits PRDIV8 and FDIV[5:0] are to be set as described in **Figure 4-1**.

For example, if the oscillator clock frequency is 4Mz and the bus clock is 25MHz, FCLKDIV bits FDIV[5:0] should be set to 20 (010100) and bit PRDIV8 set to 0. The resulting FCLK is then 190kHz. As a result, the Flash algorithm timings are increased over optimum target by:

$$(200 - 190)/200 \times 100 = 5\%$$

---

#### NOTE

Command execution time will increase proportionally with the period of FCLK.

---

---

**WARNING**

**Because of the impact of clock synchronization on the accuracy of the functional timings, programming or erasing the Flash cannot be performed if the bus clock runs at less than 1 MHz. Programming or erasing the Flash with an input clock < 150kHz should be avoided. Setting FCLKDIV to a value such that  $FCLK < 150kHz$  can destroy the Flash due to overstress. Setting FCLKDIV to a value such that  $(1/FCLK + T_{bus}) < 5\mu s$  can result in incomplete programming or erasure of the memory array cells.**

---

If the FCLKDIV register is written, the bit FDIVLD is set automatically. If this bit is zero, the register has not been written since the last reset. Program and erase commands will not be executed if this register has not been written to.

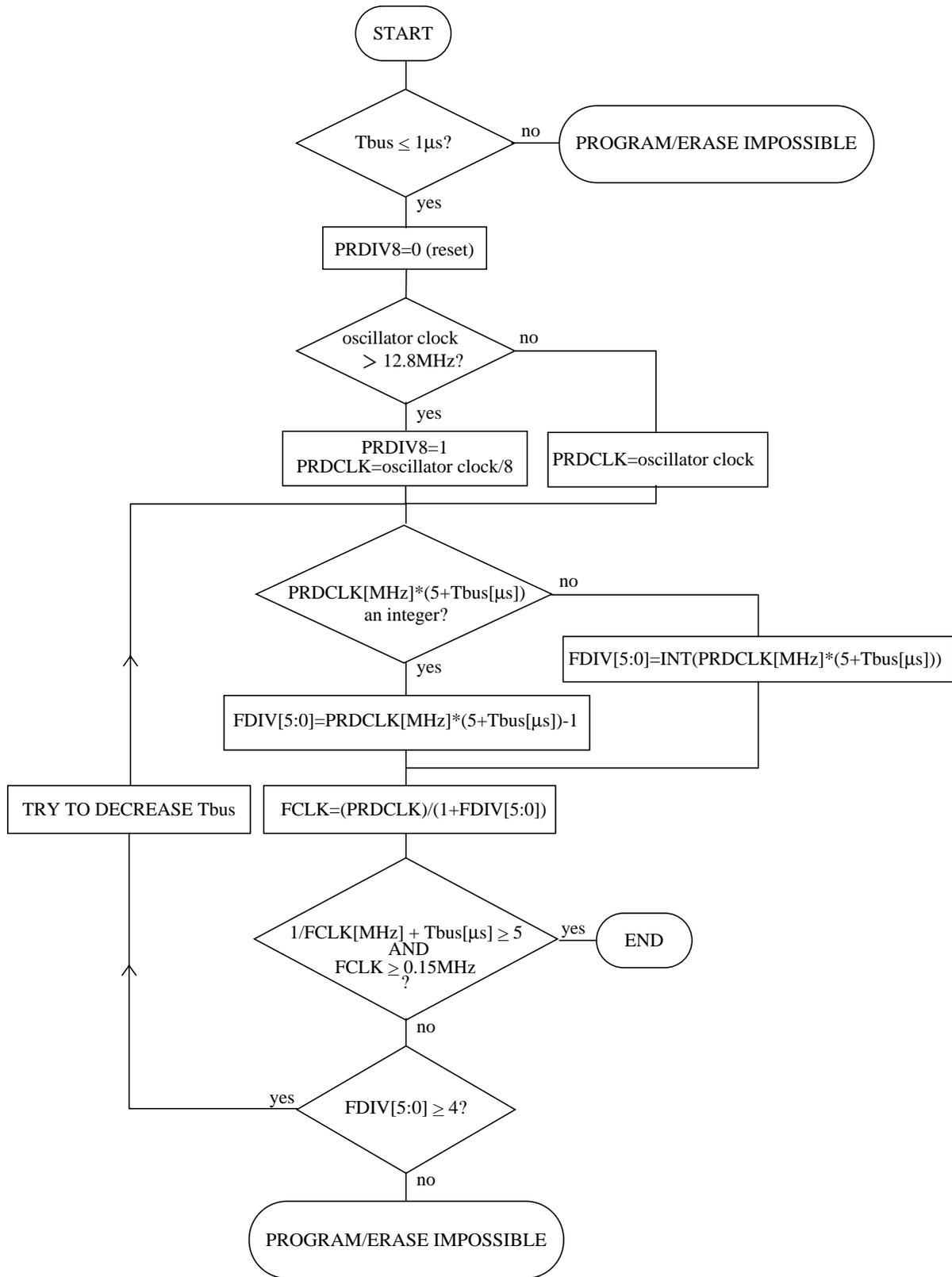


Figure 4-1 PRDIV8 and FDIV bits Determination Procedure

## 4.1.2 Program and Erase Sequences in Normal Mode

A Command State Machine is used to supervise the write sequencing for program and erase. The erase verify command follows the same flow. Before starting a command sequence, it is required that there are no pending access error or protection violations in any of the Flash blocks (the ACCERR and PVIOL flags should be cleared in the FSTAT registers). It is also required that the Flash FCNFG register and the HCS12 Core PPAGE register are set to select the Flash array address space to operate on. This initialization procedure is as follows:

1. Verify that the ACCERR and PVIOL flags in the FSTAT register are cleared in all banks. This requires a check of the FSTAT content for all conditions of the BKSEL bits in the FCNFG register.
2. Write to the BKSEL bit in the FCNFG register to select the bank of registers corresponding to the Flash block to be programmed or erased (see **Table 3-6**).
3. Write to the HCS12 Core PPAGE register (\$x030) to select one of the 16K byte pages to be programmed, if programming in the \$8000-\$BFFF address range. There is no need to set the PPAGE register when programming in the \$4000-\$7FFF or \$C000-\$FFFF address ranges.

After this optional initialization step, the CBEIF flag should be tested to ensure that the address, data and command buffers are empty. If the CBEIF flag is set, the program/erase command write sequence can be started. The following 3-step command write sequence must be strictly adhered to and no intermediate writes to the Flash module are permitted between the steps. However, the user is allowed to read any Flash register during a command write sequence. The command write sequence is as follows:

1. Write the aligned data word to be programmed to the valid Flash address space. The address and data will be stored in internal buffers. For program, all address bits are valid. For erase, the value of the data bytes is ignored. For mass erase, the address can be anywhere in the available address space of the block to be erased. For sector erase, the address bits[9:0] are ignored.
2. Write the program or erase command to the command buffer. These commands are listed in **Table 4-1**.
3. Clear the CBEIF flag by writing a “1” to it to launch the command. When the CBEIF flag is cleared, the CCIF flag is cleared by hardware indicating that the command was successfully launched. The CBEIF flag will be set again indicating the address, data and command buffers are ready for a new command sequence to begin.

The completion of the command is indicated by the setting of the CCIF flag. The CCIF flag only sets when all active and pending commands have been completed.

---

### NOTE

The Command State Machine will flag errors in program or erase write sequences by means of the ACCERR (access error) and PVIOL (protection violation) flags in the FSTAT register. An erroneous command write sequence will abort and set the appropriate flag. If set, the user must clear the ACCERR or PVIOL flags before commencing another command write sequence. By writing a “0” to the CBEIF flag, the command sequence can be aborted after the word write to the Flash address space or after writing a

command to the FCMD register and before the command is launched. Writing a “0” to the CBEIF flag in this way will set the ACCERR flag.

A summary of the program algorithm is shown in **Figure 4-2**. For the erase algorithm, the user writes either a mass or sector erase command to the FCMD register.

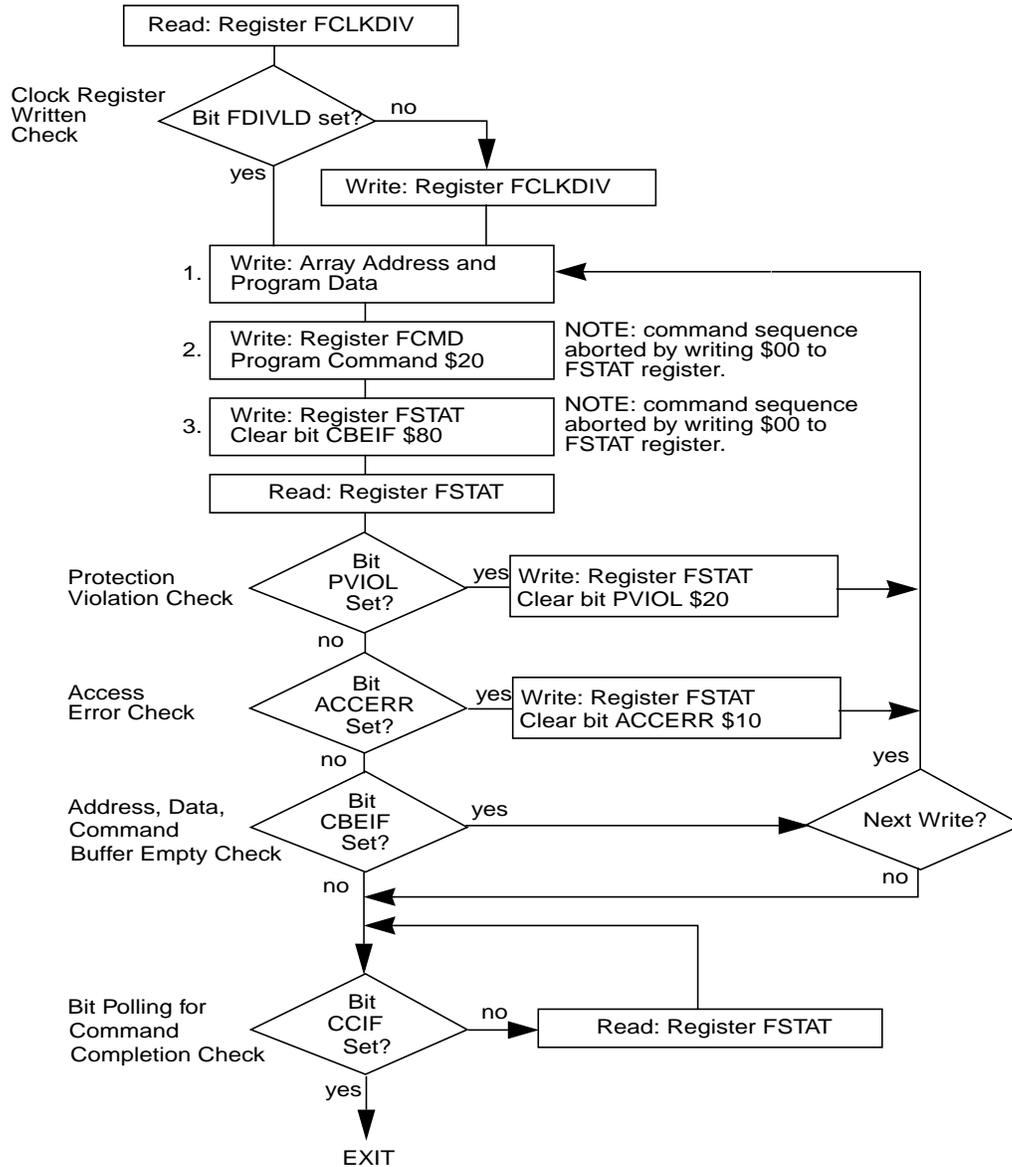


Figure 4-2 Example Program Algorithm

### 4.1.3 Valid Flash Commands

**Table 4-1** summarizes the valid Flash commands. Also shown are the effects of the commands on the Flash array.

**Table 4-1 Valid Flash Commands**

FCMD	Meaning	Function on Flash Array
\$05	Erase Verify	Verify all memory bytes of the Flash block are erased. If the block is erased, the BLANK bit will set in the FSTAT register upon command completion.
\$20	Program	Program a word (two bytes).
\$40	Sector Erase	Erase 512 words of Flash.
\$41	Mass Erase	Erase all of the Flash block. A mass erase of the full block is only possible when FPLDIS, FPHDIS and FPOPEN are set.

---

**WARNING**

**It is not permitted to program a Flash word without first erasing the sector in which that word resides.**

---

### 4.1.4 Illegal Flash Operations

The ACCERR flag will be set during the command write sequence if any of the following illegal operations are performed causing the command write sequence to immediately abort:

1. Writing to the Flash address space before initializing FCLKDIV.
2. Writing to the Flash address space in the range \$8000-\$BFFF when the HCS12 Core PPAGE register does not select a 16K byte page in the Flash block selected by the BKSEL bit in the FCNFG register.
3. Writing to the Flash address space \$4000-\$7FFF or \$C000-\$FFFF with the BKSEL bits in the FCNFG register not selecting Flash block 0.
4. Writing a misaligned word or a byte to the valid Flash address space.
5. Writing to the Flash address space while CBEIF is not set.
6. Writing a second word to the Flash address space before executing a program or erase command on the previously written word.
7. Writing to any Flash register other than FCMD after writing a word to the Flash address space.
8. Writing a second command to the FCMD register before executing the previously written command.
9. Writing an invalid command to the FCMD register.
10. Writing to any Flash register other than FSTAT (to clear CBEIF) after writing to the command

register (FCMD).

11. The part enters STOP mode and a program or erase command is in progress. The command is aborted and any pending command is killed.
12. When security is enabled, a command other than mass erase originating from a non-secure memory or from the Background Debug Mode is written to FCMD.
13. A “0” is written to the CBEIF bit in the FSTAT register.

The ACCERR flag will not be set if any Flash register is read during the command sequence.

If the Flash array is read during execution of an algorithm (i.e. CCIF bit in the FSTAT register is low), the read will return non-valid data and the ACCERR flag will not be set.

If an ACCERR flag is set in either of the FSTAT registers, the Command State Machine is locked. It is not possible to launch another command on any block until the ACCERR flag is cleared.

The PVIOL flag will be set during the command write sequence after the word write to the Flash address space if any of the following illegal operations are performed, causing the command sequence to immediately abort:

1. Writing a Flash address to program in a protected area of the Flash block.
2. Writing a Flash address to erase in a protected area of the Flash block.
3. Writing the mass erase command to FCMD while any protection is enabled. See Protection register description in **3.3.5**.

If a PVIOL flag is set in any of the FSTAT registers, the Command State Machine is locked. It is not possible to launch another command on any block until the PVIOL flag is cleared.

## 4.2 Wait Mode

When the MCU enters WAIT mode and if any command is active (CCIF=0), that command and any pending command will be completed.

The FTS512K4 module can recover the part from WAIT if the interrupts are enabled (see **Section 6**).

## 4.3 Stop Mode

If a command is active (CCIF = 0) when the MCU enters the STOP mode, the command will be aborted and the data being programmed or erased is lost. The high voltage circuitry to the Flash block will be switched off when entering STOP mode. CCIF and ACCERR flags will be set. If commands are active in multiple blocks when STOP occurs, then all the corresponding CCIF and ACCERR flags will be set. Upon exit from STOP, the CBEIF flag is set and any pending command will not be executed. All ACCERR flags must be cleared before returning to normal operation.

---

**WARNING**

As active commands are immediately aborted when the MCU enters STOP mode, it is strongly recommended that the user does not use the STOP command during program and erase execution.

---

## 4.4 Background Debug Mode

In Background Debug Mode (BDM), the FPROT registers are writable. If the MCU is unsecured, then all Flash commands listed in **Table 4-1** can be executed. If the MCU is secured and is in Special Single Chip mode, the only possible command to execute is mass erase.

## 4.5 Flash Security

The Flash module provides the necessary security information to the MCU. After each reset, the Flash module determines the security state of the MCU as defined in section **3.3.2**.

The contents of the Flash Protection/Options byte at \$FF0F in the Flash Protection/Options Field must be changed directly by programming \$FF0F when the device is unsecured and the higher address sector is unprotected. If the Flash Protection/Options byte is left in the secure state, any reset will cause the MCU to return to the secure operating mode.

### 4.5.1 Unsecuring via the Backdoor Key Access

The MCU may only be unsecured by using the Backdoor Key Access feature which requires knowledge of the contents of the Backdoor Keys (four 16-bit words programmed at addresses \$FF00 - \$FF07). If KEYEN[1:0]=10 and the KEYACC bit is set, a write to a Backdoor Key address in the Flash array triggers a comparison between the written data and the Backdoor Key data stored in the Flash array. If all four words of data are written to the correct addresses in the correct order and the data matches the Backdoor Keys stored in the Flash array, the MCU will be unsecured. The data must be written to the Backdoor Keys sequentially starting with \$FF00-1 and ending with \$FF06-7. \$0000 and \$FFFF keys are not permitted. When the KEYACC bit is set, reads of the Flash array will return invalid data.

The user code stored in the Flash array must have a method of receiving the Backdoor Key from an external stimulus. This external stimulus would typically be through one of the on-chip serial ports.

If KEYEN[1:0]=10 in the FSEC register, the MCU can be unsecured by the Backdoor Access Sequence described below:

1. Set the KEYACC bit in the Flash Configuration Register (FCNFG).
2. Write the correct four 16-bit words to Flash addresses \$FF00 - \$FF07 sequentially starting with \$FF00.
3. Clear the KEYACC bit.
4. If all four 16-bit words match the Backdoor Keys stored in Flash addresses \$FF00 - \$FF07, the MCU is unsecured and bits SEC[1:0] in the FSEC register are forced to the unsecure state of “10”.

The Backdoor Access Sequence is monitored by the internal Security State Machine. An illegal operation during the Backdoor Access Sequence will cause the Security State Machine to lock, leaving the MCU in the secured state. A reset of the MCU will cause the Security State Machine to exit the lock state and allow a new Backdoor Access Sequence to be attempted. The following illegal operations will lock the Security State Machine:

1. If any of the four 16-bit words does not match the backdoor keys programmed in the Flash array.
2. If the four 16-bit words are written in the wrong sequence.
3. If more than four 16-bit words are written.
4. If any of the four 16-bit words written are \$0000 or \$FFFF.
5. If the KEYACC bit does not remain set while the four 16-bit words are written.

After the Backdoor Access Sequence has been correctly matched, the MCU will be unsecured. The Flash security byte can be programmed to the unsecure state, if desired.

In the unsecure state, the user has full control of the contents of the four word Backdoor Key by programming it in bytes \$FF00 - \$FF07 of the Flash Protection/Options Field.

The security as defined in the Flash Security/Options byte (\$FF0F) is not changed by using the Backdoor Access Sequence to unsecure. The Backdoor Keys stored in addresses \$FF00 - \$FF07 are unaffected by the Backdoor Access Sequence. After the next reset sequence, the security state of the Flash module is determined by the Flash Security/Options byte (\$FF0F). The Backdoor Access Sequence has no effect on the program and erase protections defined in the Flash Protection Register (FPROT).

It is not possible to unsecure the MCU in Special Single Chip mode by the Backdoor Access Sequence via the Background Debug Mode.



## Section 5 Resets

### 5.1 General

If a reset occurs while any command is in progress that command will be immediately aborted. The state of the word being programmed or the sector / block being erased is not guaranteed.



## Section 6 Interrupts

### 6.1 General

The FTS512K4 module can generate an interrupt when all Flash commands are completed or the address, data and command buffers are empty.

**Table 6-1 Flash Interrupt Sources**

Interrupt Source	Interrupt Flag	Local Enable	Global (CCR) Mask
Flash Address, Data and Command Buffers empty	CBEIF (FSTAT from any Flash block)	CBEIE	I Bit
All Commands are completed on Flash	CCIF (FSTAT from any Flash block)	CCIE	I Bit

---

**NOTE**

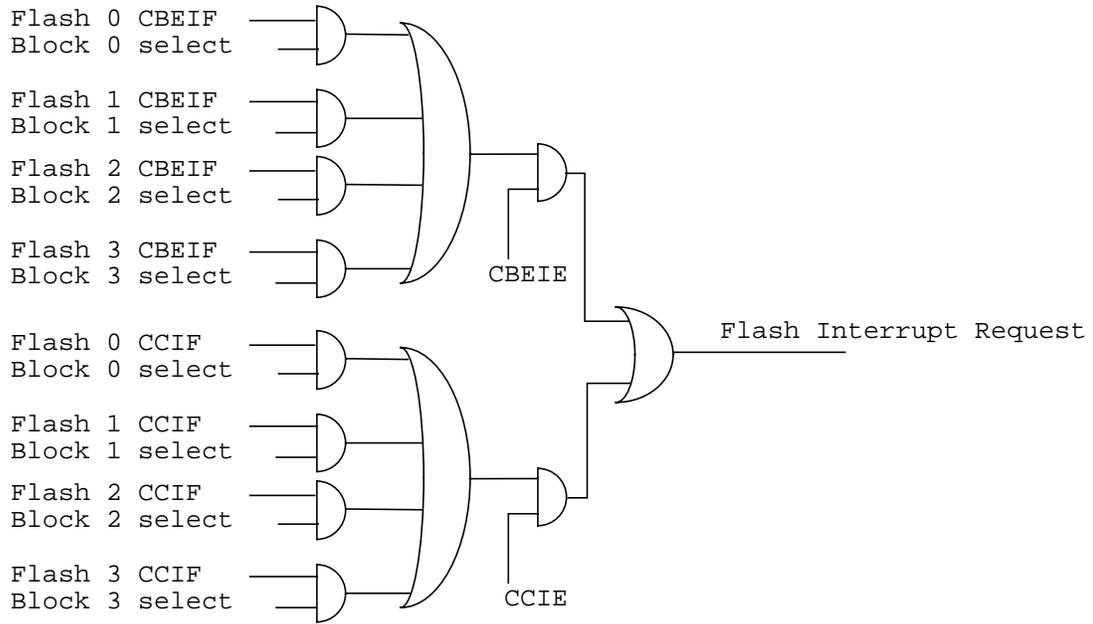
Vector addresses and their relative interrupt priority are determined at the MCU level

---

### 6.2 Description of Interrupt Operation

**Figure 6-1** shows the logic used for generating interrupt via the relevant block.

This system uses the CBEIF and CCIF flags in combination with the enable bits CBIE and CCIE in addition to the BKSEL bits) to discriminate for the interrupt generation. By taking account of the possible selected bank, the system is prevented from generating false interrupts when the command buffer is empty in an unselected bank.



**Figure 6-1 Flash Interrupt Implementation**

For a detailed description of the register bits, refer to the Flash Configuration register and Flash Status register sections (respectively **3.3.4** and **3.3.6**).

## Block Guide End Sheet

**FINAL PAGE OF  
48  
PAGES**