DOCUMENT NUMBER
S12FTS256K2ECCV1

FTS256K2ECC
Block User Guide
V01.02

Original Release Date: 27 JUN 2003
Revised: 26 AUG 2003

Motorola, Inc

Revision History

Version Number	Revision Date	Effective Date	Author	Description of Changes
V01.00	27JUN03	27JUN03		Initial Version.
V01.01	01JUL03	07MAY03		1. Update Protection Address Range Tables. 2. Update Proection Scenarios FIgure.
V01.02	26AUG03	26AUG03		1. Reset sequence sets ACCERR if double bit fault detected. 2. ACCERR prevents command write sequences in all blocks.

Table of Contents

Section 1 Introduction

Section 2 External Signal Description

Section 3 Memory Map and Registers

Section 4 Functional Description

Section 5 Resets

Section 6 Interrupts

List of Figures

List of Tables

Section 1 Introduction

1.1 Overview

This document describes the FTS256K2ECC module which includes a 256K byte Flash (Non-Volatile) memory with built-in Error Code Correction (ECC).

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

Program and erase functions are controlled by a command driven interface. The Flash module supports both block erase and sector erase. An erased bit reads ‘1’ and a programmed bit reads ‘0’. The high voltage required to program and erase is generated internally.

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 voltage sources for program or erase.

The ECC logic is included in the Flash module with the program and erase operations automatically generating the ECC parity bits. The ECC logic implements a modified Hamming code capable of correcting single bit faults and detecting double bit faults in each word of the Flash memory.

WARNING

A Flash word must be erased before being programmed. Cumulative programming of bits within a Flash word is not allowed and will result in invalid data stored.

1.1.1 Glossary

Banked Register

A memory-mapped 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 bank-select bits in the unbanked register space.

Command Write Sequence

A three-step MCU instruction sequence to execute built-in algorithms (including program and erase) on the Flash memory.

Common Register

A memory-mapped register which operates on all Flash blocks.

1.2 Features

- 256K bytes of Flash memory comprised of two 128K byte blocks with each block divided into 128 sectors of 1024 bytes. Every word (two bytes) is accompanied by 6 ECC parity bits.
- Single bit fault correction per word during read operations.
- Automated program and erase algorithm with generation of ECC parity bits.
- Interrupts on Flash command completion, command buffer empty and double bit fault detection.
- Fast sector erase and word program operation.
- 2-stage command pipeline for faster multi-word program times.
- Sector erase abort feature for critical interrupt response.
- Flexible protection scheme to prevent accidental program or erase.
- Single power supply for all Flash operations including program and erase.
- Security feature to prevent unauthorized access to the Flash memory.
- Code integrity check using built-in data compression.

1.3 Modes of Operation

- Program, erase, erase verify, and data compress operations (please refer to section **4.1** for details).

1.4 Block Diagram

A block diagram of the Flash module is shown in **Figure 1-1**.

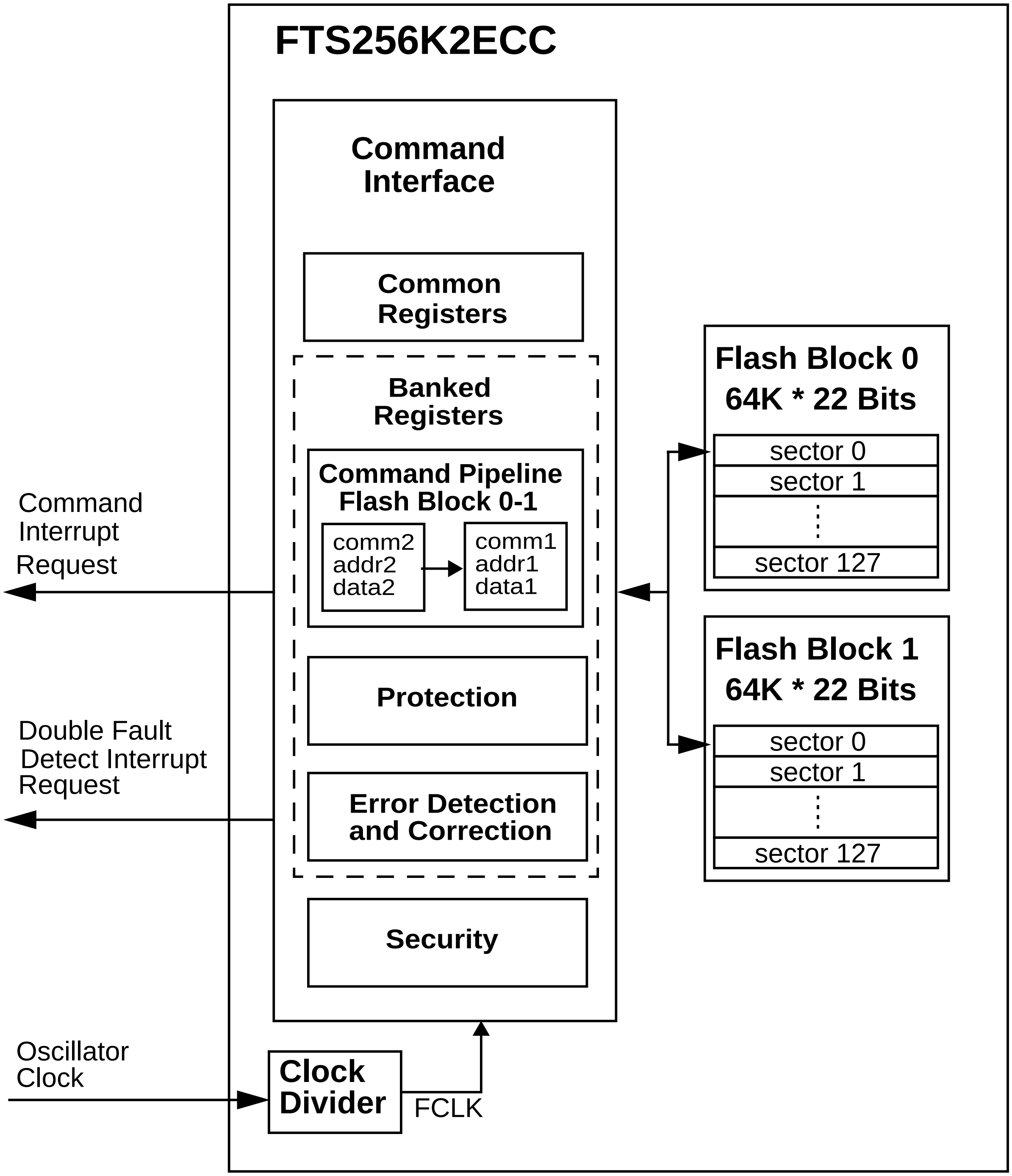


Figure 1-1 Module Block Diagram

Section 2 External Signal Description

2.1 Overview

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

Section 3 Memory Map and Registers

3.1 Overview

This section describes the memory map and registers for the Flash module.

3.2 Module Memory Map

The Flash memory map is shown in **Figure 3-1**. The HCS12 architecture places the Flash memory 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 Flash memory. By placing $3E or $3F in the HCS12 Core PPAGE register, the associated 16K byte pages appear twice in the MCU memory map.

The FPROT register, described in section **3.3.5**, can be set to globally protect a Flash block. However, three separate memory regions, one growing upward from the first address in the next-to-last page in the Flash block (called the lower region), one growing downward from the last address in the last page in the Flash block (called the higher region), and the remaining addresses in the Flash block, can be activated for protection. The Flash locations of these protectable regions are shown in **Table 3-2**. The higher address region of Flash block 0 is mainly targeted to hold the boot loader code since it covers the vector space. The lower address region of any Flash block can be used for EEPROM emulation in an MCU without an EEPROM module since it can be left unprotected while the remaining addresses are protected from program or erase.

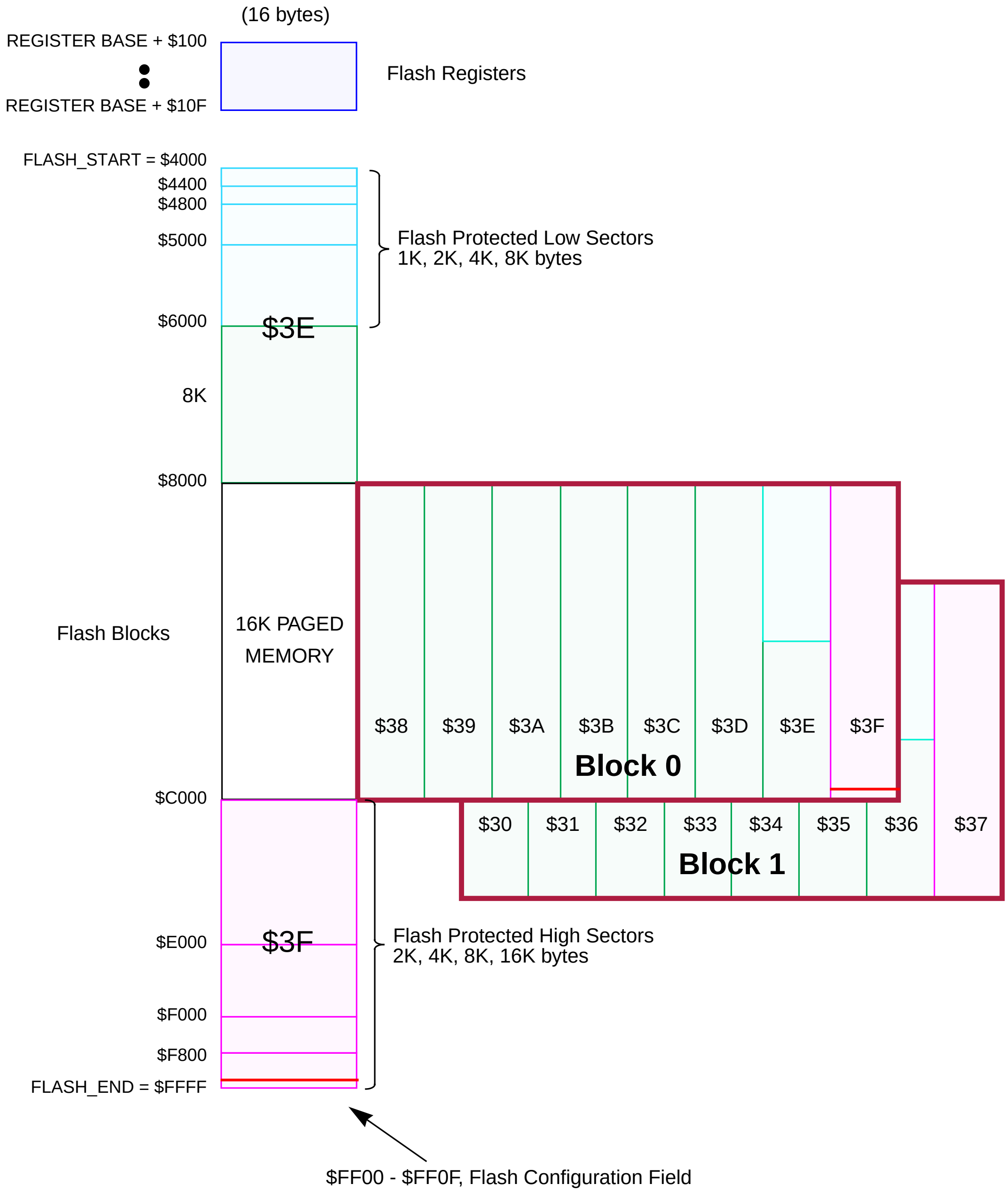
Security information that allows the MCU to restrict access to the Flash module is stored in the Flash configuration field found in Flash block 0, described in **Table 3-1**.

Table 3-1 Flash Configuration Field

Unpaged Flash Address	Paged Flash Address (PPAGE $3F)	Size (bytes)	Description
$FF00 - $FF07	$BF00-$BF07	8	Backdoor Comparison Key Refer to Section **4.5.1 Unsecuring the Flash Module via the Backdoor Key Access**
$FF08 - $FF0B	$BF08-$BF0B	4	Reserved
$FF0C	$BF0C	1	Block 1 Flash Protection byte Refer to Section **3.3.5 FPROT — Flash Protection Register**
$FF0D	$BF0D	1	Block 0 Flash Protection byte Refer to Section **3.3.5 FPROT — Flash Protection Register**

Table 3-1 Flash Configuration Field

Unpaged Flash Address	Paged Flash Address (PPAGE $3F)	Size (bytes)	Description
$FF0E	$BF0E	1	Flash Non-Volatile byte Refer to Section **3.3.8 FCTL — Flash Control Register**
$FF0F	$BF0F	1	Flash Security byte Refer to Section **3.3.2 FSEC — Flash Security Register**



Note: $30-$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 Lower Range	Protectable Higher Range	Flash Block	Block Relative Address[1]
$4000-$7FFF	Unpaged ($3E)	$4000-$43FF $4000-$47FF $4000-$4FFF $4000-$5FFF	N.A.	0	$18000-$1BFFF
$8000-$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

Table 3-2 Flash Memory Map Summary

MCU Address Range	PPAGE	Protectable Lower Range	Protectable Higher Range	Flash Block	Block Relative Address[1]
$C000-$FFFF	Unpaged ($3F)	N.A.	$F800-$FFFF $F000-$FFFF $E000-$FFFF $C000-$FFFF	0	$1C000-$1FFFF

NOTES:
1. Block Relative Address for each 128K byte Flash block consists of 17 address bits.

The Flash module also contains a set of 16 control and status registers located in address space REGISTER BASE + $100 to REGISTER BASE + $10F. In order to accommodate more than one Flash block with a minimum register address space, a set of registers (REGISTER BASE+$104 to REGISTER BASE+$10B) are repeated in all 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** while their accessibility is detailed in section **3.3**.

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

Table 3-3 Flash Register Map

Address Offset	Use	Normal Mode Access
$_00	Flash Clock Divider Register (FCLKDIV)	R/W
$_01	Flash Security Register (FSEC)	R
$_02	Flash Test Mode Register (FTSTMOD)	R/W
$_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	Flash Control Register (FCTL)	R
$_08	Flash High Address Register (FADDRHI)	R
$_09	Flash Low Address Register (FADDRLO)	R
$_0A	Flash High Data Register (FDATAHI)	R
$_0B	Flash Low Data Register (FDATALO)	R
$_0C	RESERVED1[1]	R
$_0D	RESERVED2[1]	R
$_0E	RESERVED3[1]	R
$_0F	RESERVED4[1]	R

NOTES:
1. Intended for factory test purposes only.

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 BASE + $100

	7	6	5	4	3	2	1	0
R	FDIVLD	PRDIV8	FDIV5	FDIV4	FDIV3	FDIV2	FDIV1	FDIV0
W	▨							
RESET:	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

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

The unbanked FSEC register holds all bits associated with the security of the MCU and Flash module.

REGISTER BASE + $101

	7	6	5	4	3	2	1	0
R	KEYEN		RNV5	RNV4	RNV3	RNV2	SEC	
W	▨	▨	▨	▨	▨	▨	▨	▨
Reset:	F	F	F	F	F	F	F	F

▨ = Unimplemented or Reserved

Figure 3-3 Flash Security Register (FSEC)

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

The FSEC register is loaded from the Flash Configuration Field at address $FF0F during the reset sequence, indicated by “F” in **Figure 3-3**. If the DFDIF flag in the FSTAT register is set while reading the security field location during the reset sequence, all bits in the FSEC register will be set to leave the module in a secured state with backdoor key access disabled.

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

The KEYEN[1:0] bits define the enabling of backdoor key access to the Flash module as shown in **Table 3-4**.

Table 3-4 Flash KEYEN States

KEYEN[1:0]	Status of Backdoor Key Access
00	DISABLED
01	DISABLED
10	**ENABLED**
11	DISABLED

RNV[5:2] — Reserved Non-Volatile Bits.

The RNV[5:2] bits should remain in the erased “1” state for future enhancements.

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 backdoor key access, the SEC bits are forced to “10”.

Table 3-5 Flash Security States

SEC[1:0]	Status of Security
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 to control Flash test features.

REGISTER BASE + $102

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

▨ = Unimplemented or Reserved

Figure 3-4 Flash Test Mode Register (FTSTMOD)

FDFD is readable and writable while all remaining bits read zero and are not writable. The WRALL bit is writable only in special mode to simplify mass erase and erase verify operations. All other bits in the FTSTMOD register must be written to zero.

WRALL — Write to all register banks.

If the WRALL bit is set, all banked registers sharing the same register address will be written simultaneously during a register write.
1 = Write to all register banks.
0 = Write only to the bank selected via BKSEL.

FDFD — Force Double Fault Detect.

The FDFD bit allows the user to simulate a double bit fault during Flash array read operations and check the associated interrupt routine. The FDFD bit is cleared by writing a “0” to FDFD.
1 = Any Flash array read operation will force the DFDIF flag in the FSTAT register to be set and an interrupt will be generated as long as the DFDIE interrupt enable in the FCNFG register is set.
0 = Flash array read operations will set the DFDIF flag in the FSTAT register only if a double bit fault is detected.

3.3.4 FCNFG — Flash Configuration Register

The unbanked FCNFG register enables the Flash interrupts and gates the security backdoor writes.

REGISTER BASE + $103

	7	6	5	4	3	2	1	0
R	CBEIE	CCIE	KEYACC	0	DFDIE	0	0	BKSEL
W				▨		▨	▨	
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 3-5 Flash Configuration Register (FCNFG)

CBEIE, CCIE, KEYACC, DFDIE and BKSEL bits are readable and writable while all remaining bits read zero and are not writable. KEYACC is only writable if KEYEN (see **3.3.2 FSEC — Flash Security Register**) is set to the enabled state.

CBEIE — Command Buffer Empty Interrupt Enable.

The CBEIE bit enables an interrupt in case of an empty command buffer in the Flash module.
1 = An interrupt will be requested whenever the CBEIF flag (see **3.3.6 FSTAT — Flash Status Register**) is set.
0 = Command Buffer Empty interrupt disabled.

CCIE — Command Complete Interrupt Enable.

The CCIE bit enables an interrupt in case all commands have been completed in the Flash module.
1 = An interrupt will be requested whenever the CCIF flag (see **3.3.6 FSTAT — Flash Status Register**) is set.
0 = Command Complete interrupt 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 writes are interpreted as the start of a command write sequence.

DFDIE — Double Fault Detect Interrupt Enable.

The DFDIE bit enables an interrupt in case a double bit fault is detected during a Flash block operation.
1 = An interrupt will be requested whenever the DFDIF flag (see **3.3.6 FSTAT — Flash Status Register**) is set.
0 = Double bit fault detect interrupt disabled.

BKSEL — Register Bank Select

The BKSEL bit selects the active register bank.
1 = Select register bank associated with Flash block 1.
0 = Select register bank associated with Flash block 0. The register bank associated with Flash block 0 is selected out of reset.

3.3.5 FPROT — Flash Protection Register

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

REGISTER BASE + $104

	7	6	5	4	3	2	1	0
R	FPOPEN	RNV6	FPHDIS	FPHS		FPLDIS	FPLS	
W		▨						
Reset:	F	F	F	F	F	F	F	F

▨ = Unimplemented or Reserved

Figure 3-6 Flash Protection Register (FPROT)

All bits in the FPROT register are readable and writable with restrictions (see **3.3.5.1 Flash Protection Restrictions**) except for RNV[6] which is only readable.

During reset, the banked FPROT registers are loaded from the Flash Configuration Field at the address shown in **Table 3-6**. To change the Flash protection that will be loaded during the reset sequence, the upper sector of the Flash memory must be unprotected, then the Flash Protect/Security byte located as described in **Table 3-1** must be reprogrammed. If the DFDIF flag in the FSTAT register is set while reading the protection field location during the reset sequence, the FPOPEN bit will be cleared and remaining bits in the FPROT register will be set to leave the Flash block fully protected.

Table 3-6 Reset Loading of FPROT

Flash Address	Protection Byte for
$FF0D	Flash Block 0
$FF0C	Flash Block 1

Trying to alter data in any of the protected areas in the Flash block will result in a protection violation error and the PVIOL flag will be set in the FSTAT register. A mass erase of the Flash block is not possible if any of the contained Flash sectors are protected.

FPOPEN — This bit determines the protection function for program or erase as shown in **Table 3-7**.

When FPOPEN is set, the FPHDIS and FPLDIS bits enable protection for the address range specified by the corresponding FPHS[1:0] and FPLS[1:0] bits.

When FPOPEN is clear, the FPHDIS and FPLDIS bits define unprotected address ranges as specified by the corresponding FPHS[1:0] and FPLS[1:0] bits. For an MCU without an EEPROM module, the FPOPEN clear state allows the main part of the Flash block to be protected while a small address range can remain unprotected for EEPROM emulation.

Table 3-7 Flash Protection Function

FPOPEN	FPHDIS	FPLDIS	Function[1]
1	1	1	No Protection
1	1	0	Protected Low Range
1	0	1	Protected High Range
1	0	0	Protected High and Low Ranges
0	1	1	Full Block Protected
0	1	0	Unprotected Low Range
0	0	1	Unprotected High Range
0	0	0	Unprotected High and Low Ranges

NOTES:

[1] For range sizes, refer to Flash Higher/Lower Protection Address Range tables.

RNV[6] — Reserved Non-Volatile Bit.

The RNV[6] bit should remain in the erased state “1” for future enhancements.

FPHDIS — Flash Protection Higher address range Disable.

The FPHDIS bit determines whether there is a protected/unprotected area in the higher address space of the Flash block.
1 = Protection/Unprotection disabled.
0 = Protection/Unprotection enabled.

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

The FPHS[1:0] bits determine the size of the protected/unprotected area as shown in **Table 3-8**. The FPHS[1:0] bits can only be written to while the FPHDIS bit is set.

Table 3-8 Flash Protection Higher Address Range

FPHS[1:0]	Unpaged Address Range	Paged Address Range	Protected Size
00	$F800-$FFFF	$37/$3F: $C800-$CFFF	2K bytes
01	$F000-$FFFF	$37/$3F: $C000-$CFFF	4K bytes
10	$E000-$FFFF	$37/$3F: $B000-$CFFF	8K bytes
11	$C000-$FFFF	$37/$3F: $8000-$CFFF	16K bytes

FPLDIS — Flash Protection Lower address range Disable.

The FPLDIS bit determines whether there is a protected/unprotected area in the lower address space of the Flash block.
1 = Protection/Unprotection disabled.
0 = Protection/Unprotection enabled.

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

The FPLS[1:0] bits determine the size of the protected/unprotected area as shown in **Table 3-9**. The FPLS[1:0] bits can only be written to while the FPLDIS bit is set.

Table 3-9 Flash Protection Lower Address Range

FPLS[1:0]	Unpaged Address Range	Paged Address Range	Protected Size
00	$4000-$43FF	$36/$3E: $8000-$83FF	1K bytes
01	$4000-$47FF	$36/$3E: $8000-$87FF	2K bytes
10	$4000-$4FFF	$36/$3E: $8000-$8FFF	4K bytes
11	$4000-$5FFF	$36/$3E: $8000-$9FFF	8K bytes

All possible Flash protection scenarios are illustrated in **Figure 3-7**. Although the protection scheme is loaded from the Flash array after reset, it can be changed by the user. This protection scheme can be used by applications requiring re-programming in single chip mode while providing as much protection as possible if re-programming is not required.

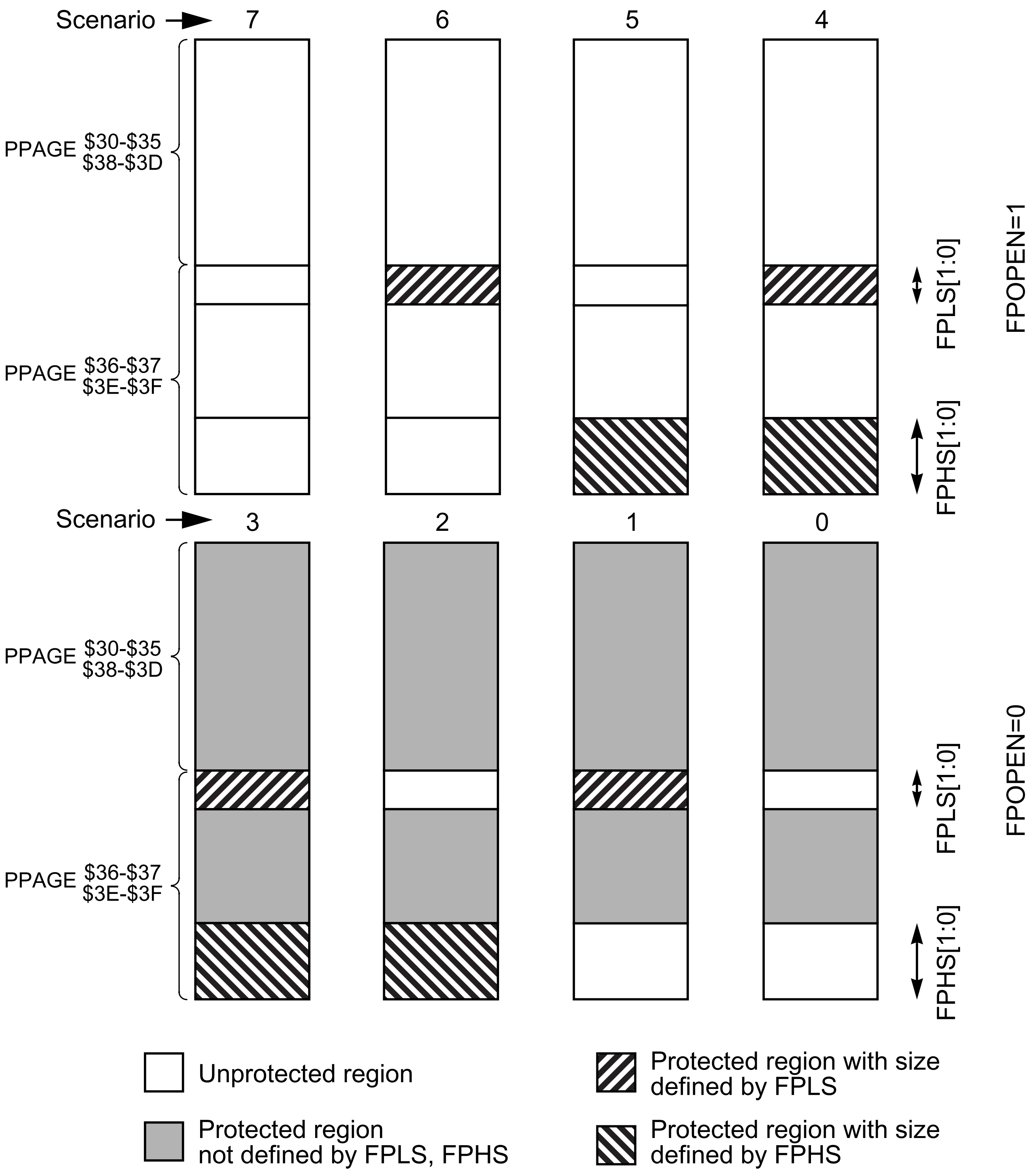


Figure 3-7 Flash Protection Scenarios

3.3.5.1 Flash Protection Restrictions

The general guideline is that Flash protection can only be added and not removed. **Table 3-10** specifies all valid transitions between Flash protection scenarios. Any attempt to write an invalid scenario to the FPROT register will be ignored and the FPROT register will remain unchanged. The contents of the FPROT register reflect the active protection scenario. See the FPHS and FPLS descriptions for additional restrictions.

Table 3-10 Flash Protection Scenario Transitions

From Protection Scenario	To Protection Scenario[1]							
	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**
0	X	X	X	X				
1		X		X				
2			X	X				
3				X				
4				X	X			
5			X	X	X	X		
6		X		X	X		X	
7	X	X	X	X	X	X	X	X

NOTES:
1. Allowed transitions marked with "X".

3.3.6 FSTAT — Flash Status Register

The banked FSTAT register defines the operational status of the module.

Register address BASE + $105

	7	6	5	4	3	2	1	0
R	CBEIF	CCIF	PVIOL	ACCERR	DFDIF	BLANK	0	0
W		▨				▨	▨	▨
Reset:	1	1	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 3-8 Flash Status Register (FSTAT)

CBEIF, PVIOL, ACCERR and DFDIF are readable and writable, CCIF and BLANK are readable and not writable, remaining bits 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 write 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 write sequence and cause the ACCERR flag to be set. Writing a “0” to CBEIF outside of a command write 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 **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 on CCIF. The CCIF flag is used together with the CCIE bit in the FCNFG register to generate an interrupt request (see **Figure 6-1**).

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

PVIOL — Protection Violation Flag.

The PVIOL flag indicates an attempt was made to program or erase an address in a protected area of the Flash block during a command write sequence. 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, it is not possible to launch a command or start a command write sequence.

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

ACCERR — Access Error Flag.

The ACCERR flag indicates an illegal access to the Flash array caused by either a violation of the command write sequence, issuing an illegal command (illegal combination of the CMDBx bits in the FCMD register), launching the sector erase abort command terminating a sector erase operation early, detection of a double fault 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, it is not possible to launch a command or start a command write sequence. If ACCERR is set by the detection of a double fault, an erase verify operation or a data compress operation, any buffered command will not launch. An ACCERR

1 = Access error has occurred.
0 = No access error detected.

DFDIF — Double Fault Detect Interrupt Flag.

The DFDIF flag indicates that one of the following Flash block operations has detected a double bit fault in the stored parity and data bits.

- Array Read.
- Erase Verify.
- Data Compress.

- Reset Sequence (reads of the protection and security fields stored in the Flash memory).

When the DFDIF flag is set during a Flash array read operation, the data read from the Flash module are the data bits read out of the Flash array without correction and should be considered invalid. When the DFDIF flag is set during a Flash array read, erase verify, data compress or reset sequence operation, the Flash block address containing the parity and data bits that caused the DFDIF flag to set will be stored in the FADDR register and the parity bits will be stored in the FDATA register. The DFDIF flag is cleared by writing a “1” to the ACCERR bit which is set when the DFDIF flag is set. Writing a “0” to the DFDIF flag has no effect on DFDIF. The DFDIF flag is used together with the DFDIE enable bit to generate an interrupt request (see **Figure 6-1**). While DFDIF is set, Flash array read operations are allowed. If DFDIF is not cleared and another double bit fault is detected, the FADDR and FDATA registers will maintain the contents from the fault that caused the DFDIF bit to set.

1 = Double bit fault detected.
0 = No double bit fault detected.

BLANK — Flag indicating the erase verify operation status.

When the CCIF flag is set after completion of an erase verify command, the BLANK flag indicates the result of the erase verify operation. The BLANK flag is cleared by the Flash module when CBEIF is cleared as part of a new valid command write sequence. Writing to the BLANK flag has no effect on BLANK.

1 = Flash block verified as erased.
0 = Flash block verified as not erased.

3.3.7 FCMD — Flash Command Register

The banked FCMD register is the Flash command register.

REGISTER BASE + $106

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

▨ = Unimplemented or Reserved

Figure 3-9 Flash Command Register (FCMD)

All CMDB bits are readable and writable during a command write sequence while bit 7 reads zero and is not writable.

CMDB[6:0] — Valid Flash commands are shown in **Table 3-11**. Writing any command other than those listed in **Table 3-11** sets the ACCERR flag in the FSTAT register.

Table 3-11 Valid Flash Command List

CMDB[6:0]	Command
$05	Erase Verify

Table 3-11 Valid Flash Command List

CMDB[6:0]	Command
$06	Data Compress
$20	Word Program
$40	Sector Erase
$41	Mass Erase
$47	Sector Erase Abort

3.3.8 FCTL — Flash Control Register

The banked FCTL register is the Flash control register.

REGISTER BASE + $107

	7	6	5	4	3	2	1	0
R	NV7	NV6	NV5	NV4	NV3	NV2	NV1	NV0
W	▨	▨	▨	▨	▨	▨	▨	▨
Reset:	F	F	F	F	F	F	F	F

▨ = Unimplemented or Reserved

Figure 3-10 Flash Control Register (FCTL)

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

The FCTL register is loaded from the Flash Configuration Field byte at $FF0E during the reset sequence, indicated by “F” in **Figure 3-10**.

NV[7:0] — Non-Volatile Bits.

The NV[7:0] bits are available as non-volatile bits.

3.3.9 FADDR — Flash Address Registers

The banked FADDRHI and FADDRLO registers are the Flash address registers.

REGISTER BASE + $108

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

▨ = Unimplemented or Reserved

Figure 3-11 Flash Address High Register (FADDRHI)

REGISETER BASE + $109

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

▨ = Unimplemented or Reserved

Figure 3-12 Flash Address Low Register (FADDRLO)

All FADDRHI and FADDRLO bits are readable but are not writable. After an array write as part of a command write sequence, the FADDR registers will contain the mapped MCU address written. If a double bit fault is detected, as indicated by the setting of the DFDIF bit in the FSTAT register, the faulty Flash block address is stored in the FADDR registers as a word address. The faulty Flash block address remains readable until the start of the next command write sequence. The mapping of the FADDR registers to the MCU address is shown in **Figure 3-13** and **Figure 3-14**.

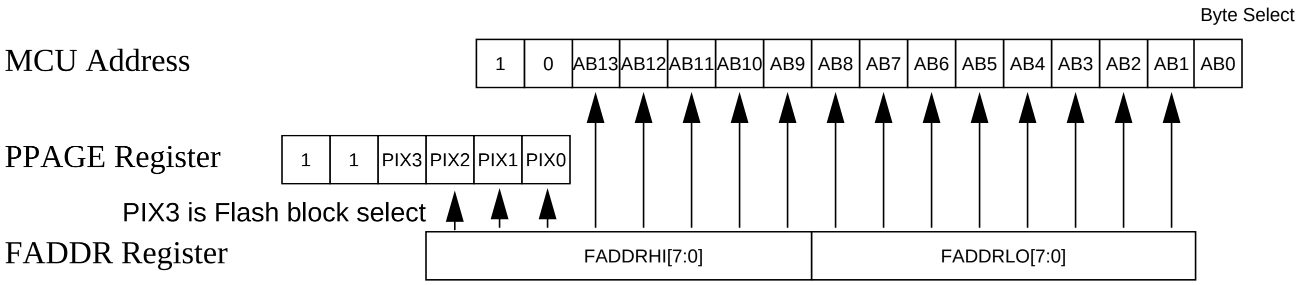


Figure 3-13 FADDR to MCU Address Mapping (Paged)

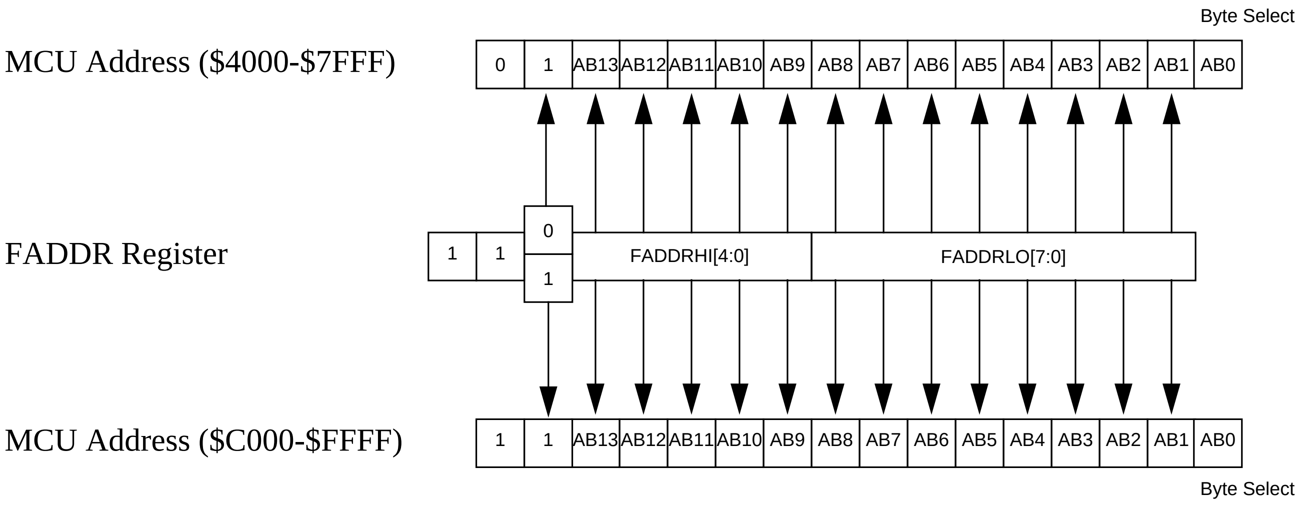


Figure 3-14 FADDR to MCU Address Mapping (Unpaged)

3.3.10 FDATA — Flash Data Registers

The banked FDATAHI and FDATALO registers are the Flash data registers.

REGISTER BASE + $10A

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

▨ = Unimplemented or Reserved

Figure 3-15 Flash Data High Register (FDATAHI)

REGISTER BASE + $10B

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

▨ = Unimplemented or Reserved

Figure 3-16 Flash Data Low Register (FDATALO)

All FDATAHI and FDATALO bits are readable but are not writable. After an array write as part of a command write sequence, the FDATA registers will contain the data written. At the completion of a data compress operation, the resulting 16-bit signature is stored in the FDATA registers. The data compression signature is readable in the FDATA registers until a new command write sequence is started or a double bit fault is detected in a Flash array read operation. If a double bit fault is detected during a Flash array read, erase verify or data compress operation, the parity bits stored in the Flash array at the failed location will be stored in the lower six bits of FDATALO. The faulty parity bits remain readable until the start of the next command write sequence.

3.3.11 RESERVED1

This register is reserved for factory testing and is not accessible.

REGISTER BASE + $10C

	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 RESERVED1

All bits read zero and are not writable.

3.3.12 RESERVED2

This register is reserved for factory testing and is not accessible.

REGISTER BASE + $10D

	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 3-18 RESERVED2

All bits read zero and are not writable.

3.3.13 RESERVED3

This register is reserved for factory testing and is not accessible.

REGISTER 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 3-19 RESERVED3

All bits read zero and are not writable.

3.3.14 RESERVED4

This register is reserved for factory testing and is not accessible.

REGISTER 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 3-20 RESERVED4

All bits read zero and are not writable.

Section 4 Functional Description

4.1 Program, Erase, Erase Verify and Data Compress Operations

Write and read operations are both used for the program, erase, erase verify and data compress algorithms described in this section. The program and erase algorithms are time 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 second command along with the necessary data and address can be stored to the buffer while the first command is still in progress. This pipelined operation allows a time optimization when programming more than one word on a specific row in the Flash block as the high voltage generation can be kept active 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 with interrupts generated, if enabled.

The next sections describe:

1. How to write the FCLKDIV register.
2. Command write sequences used to program, erase, and verify the Flash memory.
3. Valid Flash commands.
4. Effects resulting from illegal Flash command write sequences or aborting Flash operations.

4.1.1 Writing the FCLKDIV Register

Prior to issuing any program, erase, erase verify or data compress command, it is first necessary to write the FCLKDIV register to divide the oscillator clock down to within the 150kHz to 200kHz range. Since the program and erase timings are also a function of the bus clock, 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 950kHz and the bus clock frequency is 10MHz, FCLKDIV bits FDIV[5:0] should be set to 4 (000100) and bit PRDIV8 set to "0". The resulting FCLK frequency is then 190kHz. As a result, the Flash program and erase algorithm timings are increased over the optimum target by:

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

NOTE

Program and erase 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 memory cannot be performed if the bus clock runs at less than 1 MHz. Programming or erasing the Flash memory with FCLK < 150kHz should be avoided. Setting FCLKDIV to a value such that FCLK < 150kHz can destroy the Flash memory due to overstress. Setting FCLKDIV to a value such that (1/FCLK+Tbus) < 5µs can result in incomplete programming or erasure of the Flash memory cells.

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

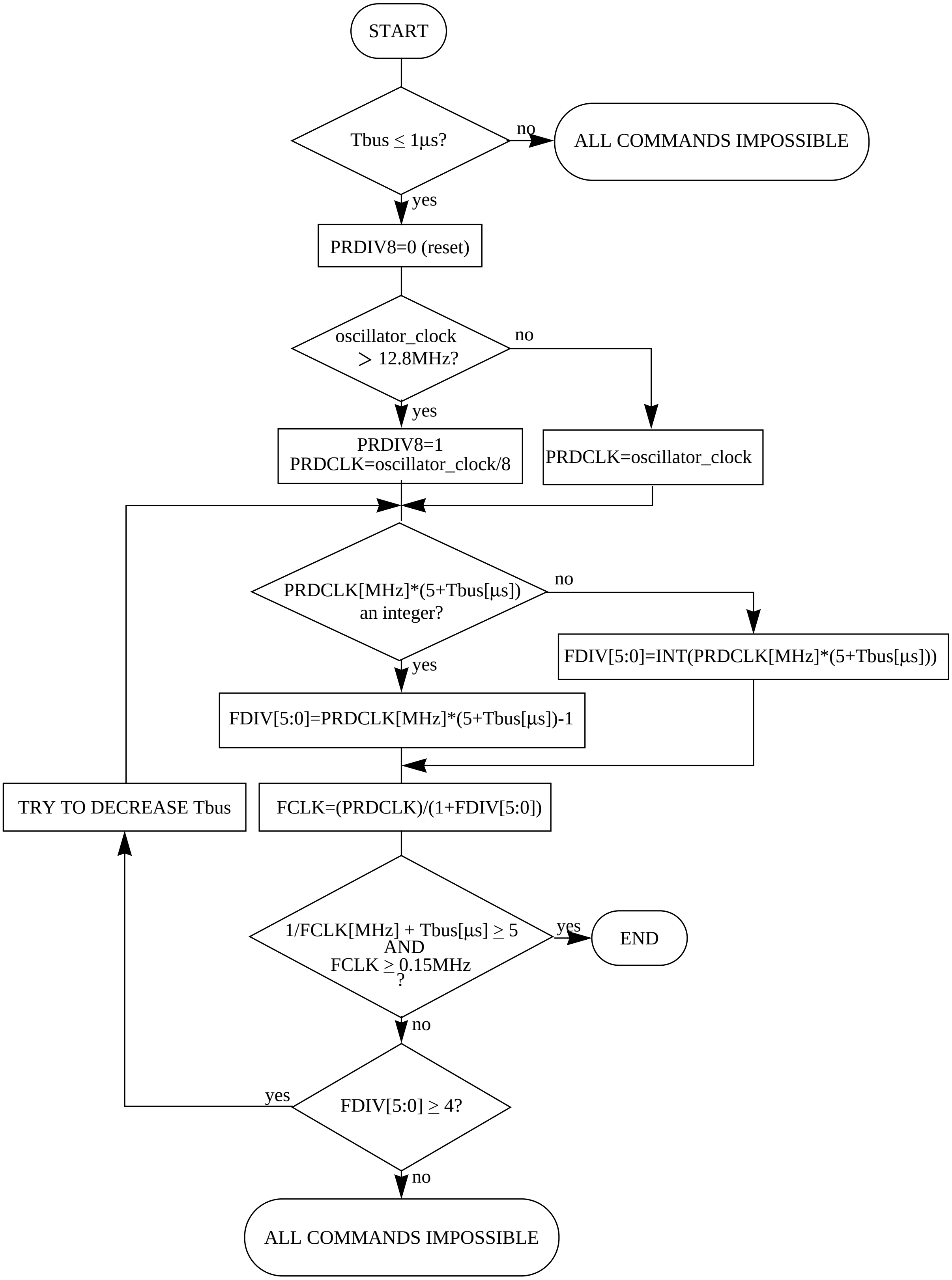

START
Tbus ≤ 1µs?
no
ALL COMMANDS IMPOSSIBLE
yes
PRDIV8=0 (reset)
oscillator_clock > 12.8MHz?
no
yes
PRDIV8=1
PRDCLK=oscillator_clock/8
PRDCLK=oscillator_clock
PRDCLK[MHz]*(5+Tbus[µs]) an integer?
no
yes
FDIV[5:0]=INT(PRDCLK[MHz]*(5+Tbus[µs]))
FDIV[5:0]=PRDCLK[MHz]*(5+Tbus[µs])-1
TRY TO DECREASE Tbus
FCLK=(PRDCLK)/(1+FDIV[5:0])
1/FCLK[MHz] + Tbus[µs] ≥ 5 AND FCLK ≥ 0.15MHz ?
yes
END
no
FDIV[5:0] ≥ 4?
yes
no
ALL COMMANDS IMPOSSIBLE

Figure 4-1 Determination Procedure for PRDIV8 and FDIV Bits

4.1.2 Command Write Sequence

The Command State Machine is used to supervise the command write sequence to execute program, erase, erase verify and data compress algorithms.

Before starting a command write sequence, the ACCERR and PVIOL flags in the FSTAT register must be clear (see section **3.3.6**) and the CBEIF flag should be tested to determine the state of the address, data and command buffers. If the CBEIF flag is set, indicating the buffers are empty, a new command write sequence can be started. If the CBEIF flag is clear, indicating the buffers are not available, a new command write sequence will overwrite the contents of the address, data and command buffers.

A command write sequence consists of three steps which must be strictly adhered to with writes to the Flash module not permitted between the steps. However, Flash register and array reads are allowed during a command write sequence. A command write sequence consists of the following steps:

1. Write an aligned data word to a valid Flash array address. The address and data will be stored in the address and data buffers, respectively. If the CBEIF flag is clear when the Flash array write occurs, the contents of the address and data buffers will be overwritten and the CBEIF flag will be set.
2. Write a valid command to the FCMD register.
    a. For the erase verify command (see section **4.1.3.1**), the contents of the data buffer are ignored and all address bits in the address buffer are ignored.
    b. For the data compress command (see section **4.1.3.2**), the contents of the data buffer represents the number of consecutive words to read for data compression and the contents of the address buffer represents the starting address.
    c. For the program command (see section **4.1.3.3**), the contents of the data buffer will be programmed to the address specified in the address buffer with all address bits valid.
    d. For the sector erase command (see section **4.1.3.4**), the contents of the data buffer are ignored and address bits [9:0] contained in the address buffer are ignored.
    e. For the mass erase command (see section **4.1.3.5**), the contents of the data buffer and address buffer are ignored.
    f. For the sector erase abort command (see section **4.1.3.6**), the contents of the data buffer and address buffer are ignored.
3. Clear the CBEIF flag by writing a “1” to CBEIF to launch the command. When the CBEIF flag is cleared, the CCIF flag is cleared on the same bus cycle by internal hardware indicating that the command was successfully launched. For all command write sequences except data compress and sector erase abort, the CBEIF flag will set four bus cycles after the CCIF flag is cleared indicating that the address, data and command buffers are ready for a new command write sequence to begin. For data compress and sector erase abort operations, the CBEIF flag will remain clear until the operation completes.

A command write sequence can be aborted prior to clearing the CBEIF flag by writing a “0” to the CBEIF flag and will result in the ACCERR flag being set.

Except for the sector erase abort command, a buffered command will wait for the active operation to be completed before being launched. The sector erase abort command is launched when the CBEIF flag is cleared as part of a sector erase abort command write sequence. Once a command is launched, the completion of the command operation is indicated by the setting of the CCIF flag. The CCIF flag only sets when all active and buffered commands have been completed.

4.1.3 Valid Flash Commands

Table 4-1 summarizes the valid Flash commands along with the effects of the commands on the Flash block.

Table 4-1 Valid Flash Command Description

FCMDB	Command	Function on Flash Memory
$05	Erase Verify	Verify all memory bytes in the Flash block are erased. If the Flash block is erased, the BLANK flag in the FSTAT register will set upon command completion.
$06	Data Compress	Compress data from a selected portion of the Flash block. The resulting signature is stored in the FDATA register.
$20	Program	Program a word (two bytes) in the Flash block.
$40	Sector Erase	Erase all memory bytes in a sector of the Flash block.
$41	Mass Erase	Erase all memory bytes in the Flash block. A mass erase of the full Flash block is only possible when FPLDIS, FPHDIS and FPOPEN bits in the FPROT register are set prior to launching the command.
$47	Sector Erase Abort	Abort the sector erase operation. The sector erase operation will terminate according to a set procedure. The Flash sector should not be considered erased if the ACCERR flag is set upon command completion.

WARNING

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

4.1.3.1 Erase Verify Command

The erase verify operation is used to confirm that a Flash block is erased. After launching the erase verify command, the CCIF flag in the FSTAT register will set after the operation has completed unless a second command has been buffered. The number of bus cycles required to execute the erase verify operation is equal to the number of addresses in the Flash block plus 12 bus cycles as measured from the time the CBEIF flag is cleared until the CCIF flag is set. The result of the erase verify operation is reflected in the state of the BLANK flag in the FSTAT register. If the BLANK flag is set in the FSTAT register, the Flash memory is erased.

If the ECC logic detects a double bit fault during the erase verify operation, the operation will terminate immediately and set the DFDIF and ACCERR flags in the FSTAT register. The faulty address will be stored in the FADDR registers and the ECC parity bits read at the faulty address will be stored in the FDATALO register. The CCIF flag will set after the DFDIF flag is set and the faulty information is stored in the FADDR and FDATALO registers.

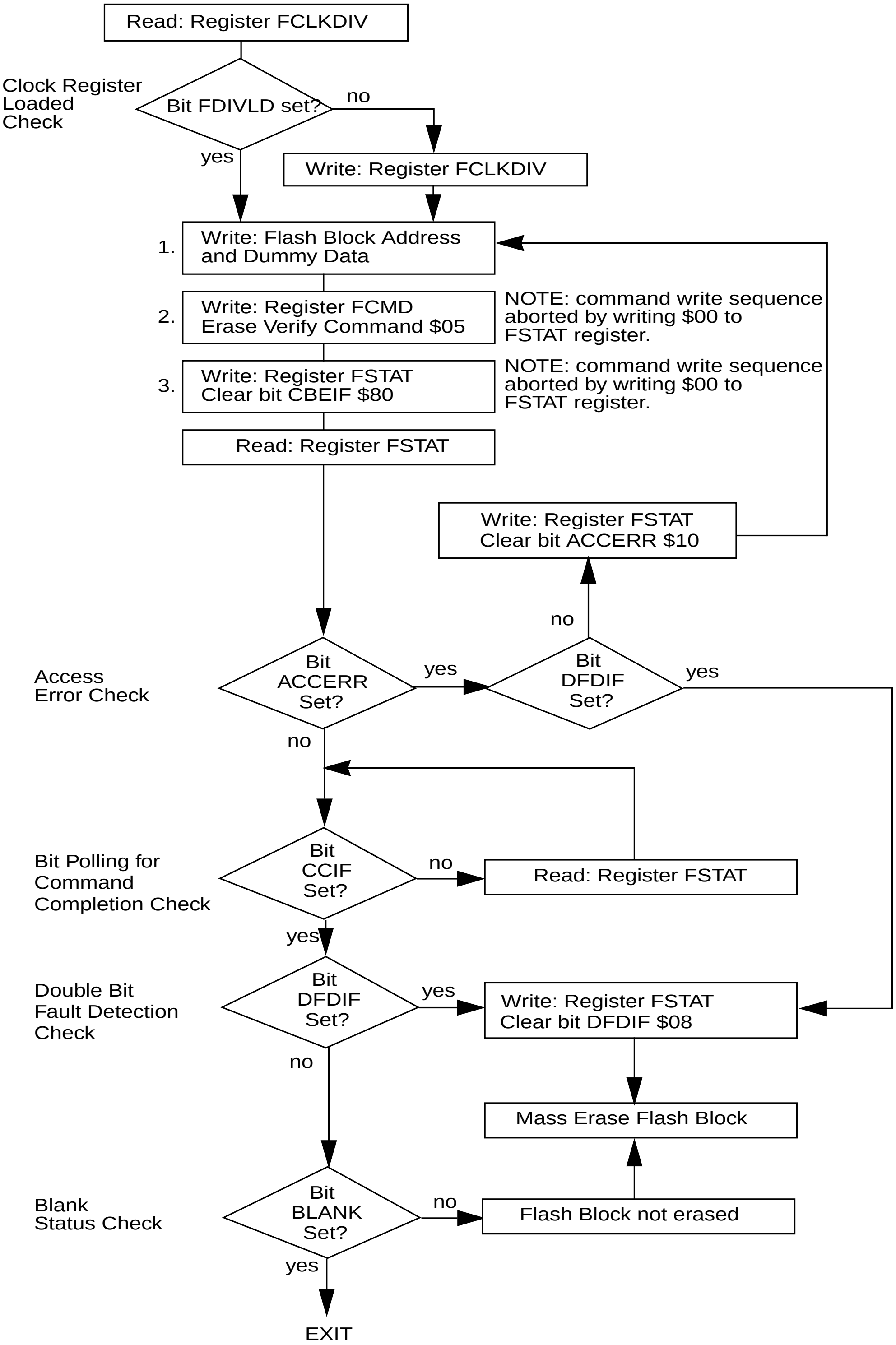

Read: Register FCLKDIV
Clock Register Loaded Check
Bit FDIVLD set?
no
yes
Write: Register FCLKDIV
1.
Write: Flash Block Address and Dummy Data
2.
Write: Register FCMD Erase Verify Command $05
NOTE: command write sequence aborted by writing $00 to FSTAT register.
3.
Write: Register FSTAT Clear bit CBEIF $80
NOTE: command write sequence aborted by writing $00 to FSTAT register.
Read: Register FSTAT
Write: Register FSTAT Clear bit ACCERR $10
no
Access Error Check
Bit ACCERR Set?
yes
Bit DFDIF Set?
yes
no
Bit Polling for Command Completion Check
Bit CCIF Set?
no
Read: Register FSTAT
yes
Double Bit Fault Detection Check
Bit DFDIF Set?
yes
Write: Register FSTAT Clear bit DFDIF $08
no
Mass Erase Flash Block
Blank Status Check
Bit BLANK Set?
no
Flash Block not erased
yes
EXIT

Figure 4-2 Example Erase Verify Command Flow

4.1.3.2 Data Compress Command

The data compress command is used to check Flash code integrity by compressing data from a selected portion of the Flash block into a signature analyzer. The starting address for the data compress operation is defined by the address written during the command write sequence. The number of consecutive word addresses compressed is defined by the data written during the command write sequence. If the data value written is $0000, 64K addresses or 128K bytes will be compressed. After launching the data compress command, the CCIF flag in the FSTAT register will set after the data compress operation has completed. The number of bus cycles required to execute the data compress operation is equal to two times the number of addresses read plus 20 bus cycles as measured from the time the CBEIF flag is cleared until the CCIF flag is set. Once the CCIF flag is set, the signature generated by the data compress operation is available in the FDATA register. The signature in the FDATA register can be compared to the expected signature to determine the integrity of the selected data stored in the Flash block. If the last address of the Flash block is reached during the data compress operation, data compression will continue with the starting address of the Flash block.

NOTE

Since the FDATA register (or data buffer) is written to as part of the data compress operation, a command write sequence is not allowed to be buffered behind a data compress command write sequence. The CBEIF flag will not set after launching the data compress command to indicate that a command should not be buffered behind it. If an attempt is made to start a new command write sequence with a data compress operation active, the ACCERR flag in the FSTAT register will be set. A new command write sequence should only be started after reading the signature stored in the FDATA register. A Flash array read that generates a double bit fault will overwrite the contents of the FDATA register.

In order to take corrective action, it is recommended that the data compress command be executed on a Flash sector or subset of a Flash sector. If the data compress operation on a Flash sector returns an invalid signature, the Flash sector should be erased using the sector erase command and then reprogrammed using the program command.

The data compress command can be used to verify that a sector or sequential set of sectors are erased.

If the ECC logic detects a double bit fault during the data compress operation, the operation will terminate immediately and set the DFDIF and ACCERR flags in the FSTAT register. The faulty address will be stored in the FADDR registers and the ECC parity bits read at the faulty address will be stored in the FDATALO register. The CCIF flag will set after the DFDIF flag is set and the faulty information is stored in the FADDR and FDATALO registers.

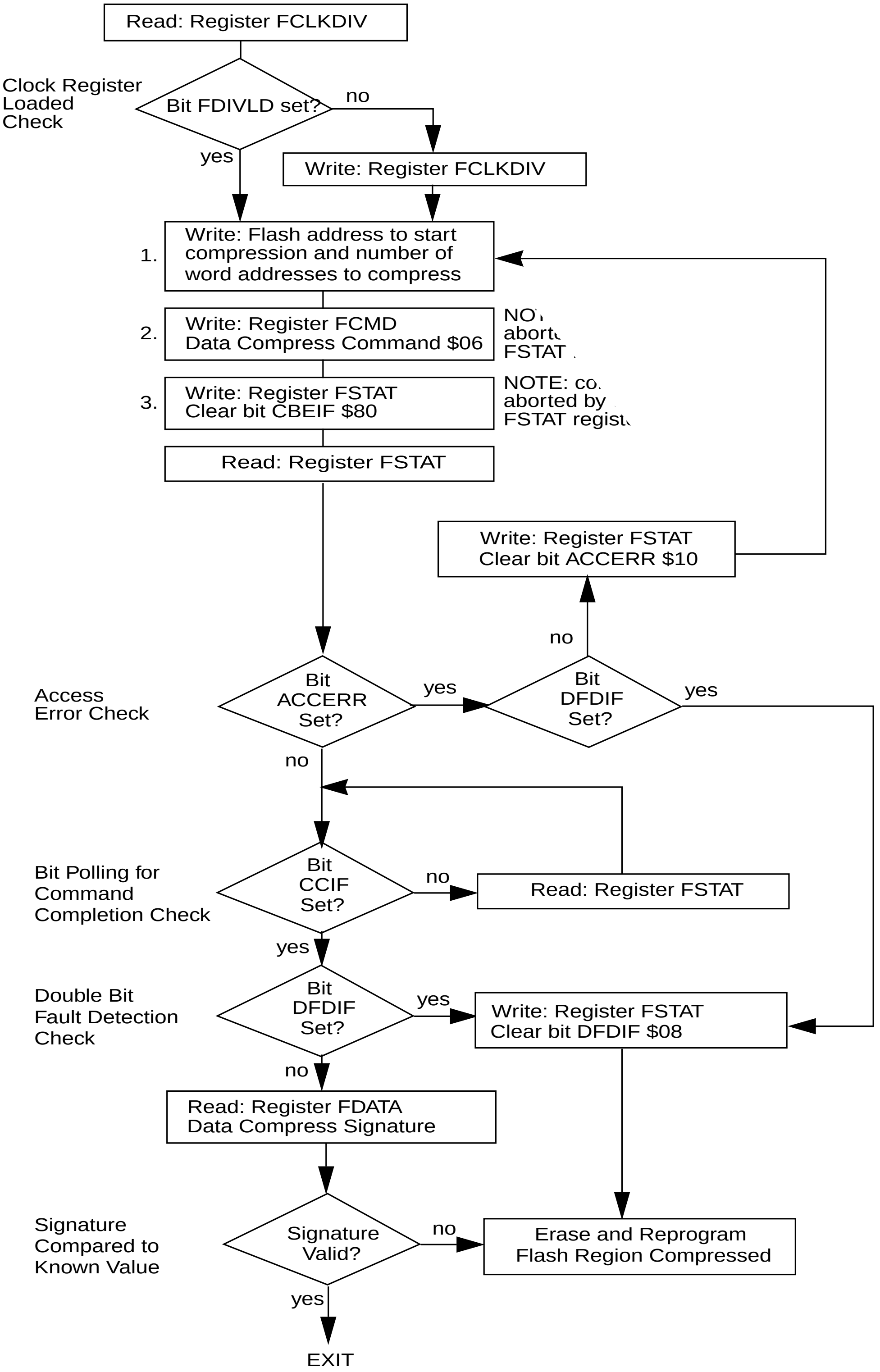

Read: Register FCLKDIV
Clock Register Loaded Check
Bit FDIVLD set?
no
yes
Write: Register FCLKDIV
1.
Write: Flash address to start compression and number of word addresses to compress
2.
Write: Register FCMD Data Compress Command $06
NOT abort FSTAT
3.
Write: Register FSTAT Clear bit CBEIF $80
NOTE: co aborted by FSTAT regist
Read: Register FSTAT
Write: Register FSTAT Clear bit ACCERR $10
no
Access Error Check
Bit ACCERR Set?
yes
Bit DFDIF Set?
yes
no
Bit Polling for Command Completion Check
Bit CCIF Set?
no
Read: Register FSTAT
yes
Double Bit Fault Detection Check
Bit DFDIF Set?
yes
Write: Register FSTAT Clear bit DFDIF $08
no
Read: Register FDATA Data Compress Signature
Signature Compared to Known Value
Signature Valid?
no
Erase and Reprogram Flash Region Compressed
yes
EXIT

Figure 4-3 Example Data Compress Command Flow

4.1.3.3 Program Command

The program command is used to program a previously erased word in the Flash memory using an embedded algorithm. If the word to be programmed is in a protected area of the Flash block, the PVIOL flag in the FSTAT register will set and the program command will not launch. Once the program command has successfully launched, the CCIF flag in the FSTAT register will set after the program operation has completed unless a second command has been buffered.

A summary of the launching of a program operation is shown in **Figure 4-4**.

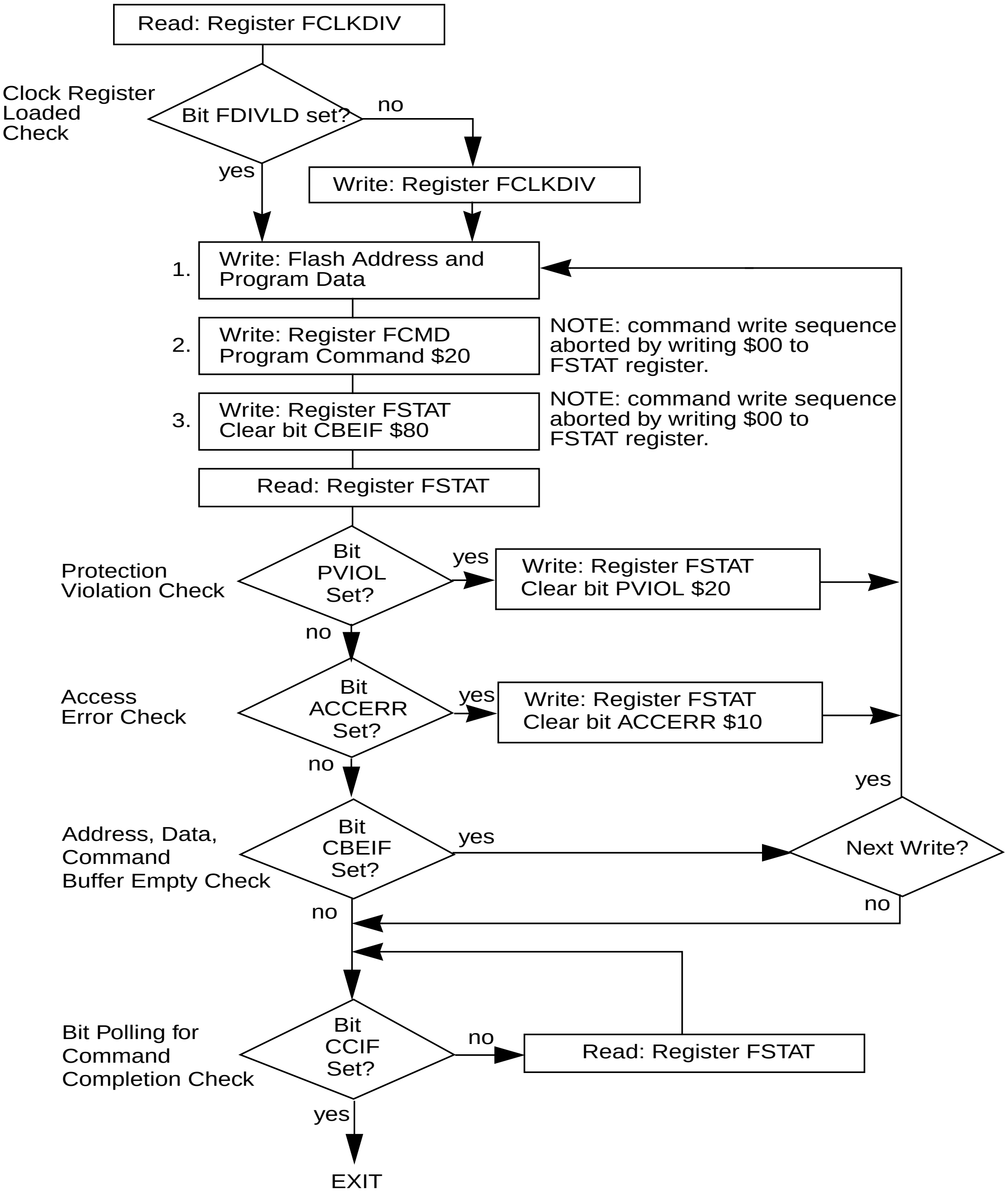

Read: Register FCLKDIV
Clock Register Loaded Check
Bit FDIVLD set?
no
yes
Write: Register FCLKDIV
1.
Write: Flash Address and Program Data
2.
Write: Register FCMD Program Command $20
NOTE: command write sequence aborted by writing $00 to FSTAT register.
3.
Write: Register FSTAT Clear bit CBEIF $80
NOTE: command write sequence aborted by writing $00 to FSTAT register.
Read: Register FSTAT
Protection Violation Check
Bit PVIOL Set?
yes
Write: Register FSTAT Clear bit PVIOL $20
no
Access Error Check
Bit ACCERR Set?
yes
Write: Register FSTAT Clear bit ACCERR $10
no
yes
Address, Data, Command Buffer Empty Check
Bit CBEIF Set?
yes
Next Write?
no
no
Bit Polling for Command Completion Check
Bit CCIF Set?
no
Read: Register FSTAT
yes
EXIT

Figure 4-4 Example Program Command Flow

4.1.3.4 Sector Erase Command

The sector erase command is used to erase the addressed sector in the Flash memory using an embedded algorithm. If the Flash sector to be erased is in a protected area of the Flash block, the PVIOL flag in the FSTAT register will set and the sector erase command will not launch. Once the sector erase command has successfully launched, the CCIF flag in the FSTAT register will set after the sector erase operation has completed unless a second command has been buffered.

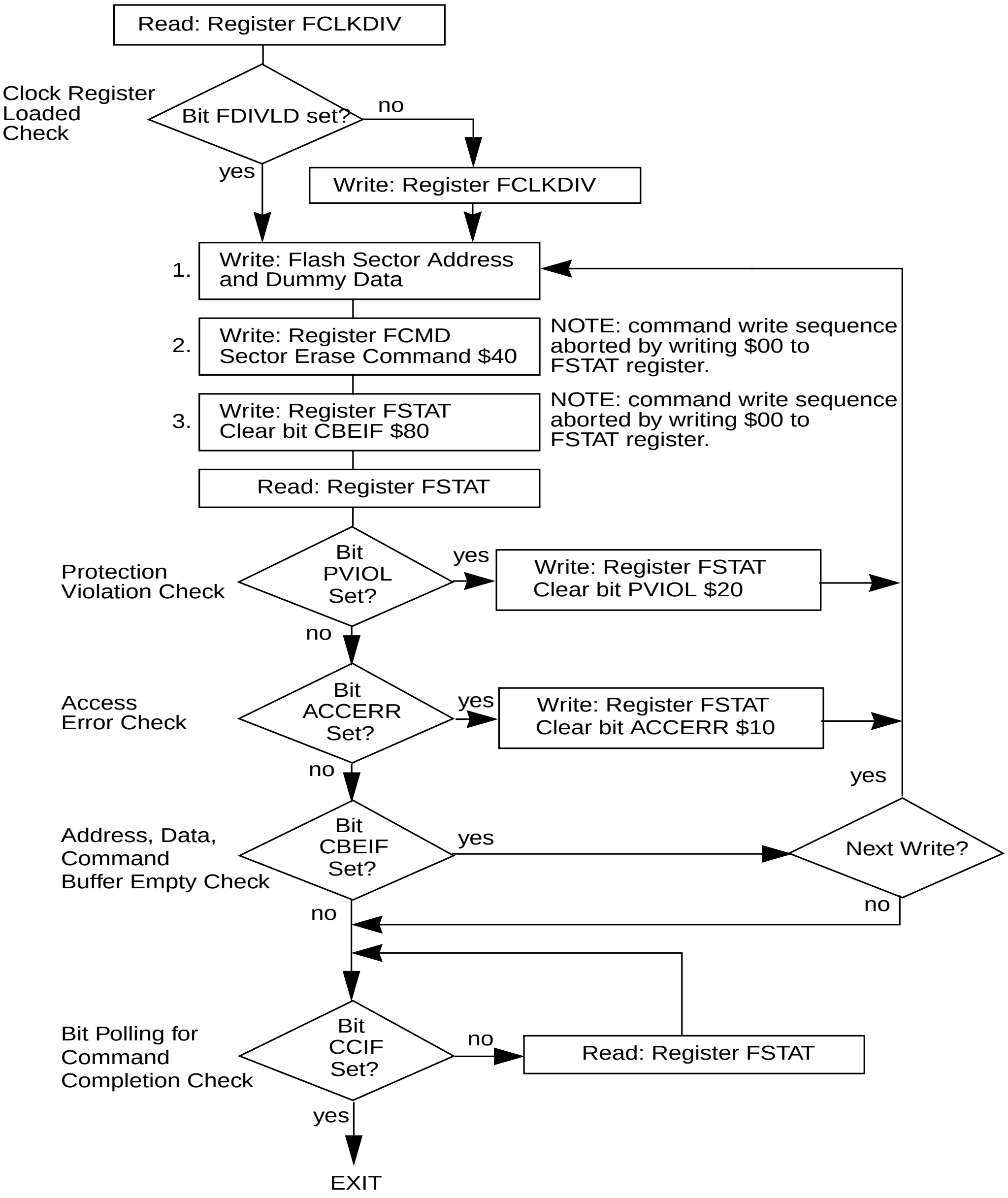

Read: Register FCLKDIV
Clock Register Loaded Check
Bit FDIVLD set?
no
yes
Write: Register FCLKDIV
1.
Write: Flash Sector Address and Dummy Data
2.
Write: Register FCMD Sector Erase Command $40
NOTE: command write sequence aborted by writing $00 to FSTAT register.
3.
Write: Register FSTAT Clear bit CBEIF $80
NOTE: command write sequence aborted by writing $00 to FSTAT register.
Read: Register FSTAT
Protection Violation Check
Bit PVIOL Set?
yes
Write: Register FSTAT Clear bit PVIOL $20
no
Access Error Check
Bit ACCERR Set?
yes
Write: Register FSTAT Clear bit ACCERR $10
no
yes
Address, Data, Command Buffer Empty Check
Bit CBEIF Set?
yes
Next Write?
no
no
Bit Polling for Command Completion Check
Bit CCIF Set?
no
Read: Register FSTAT
yes
EXIT

Figure 4-5 Example Sector Erase Command Flow

4.1.3.5 Mass Erase Command

The mass erase command is used to erase a Flash memory block using an embedded algorithm. If the Flash block to be erased contains any protected area, the PVIOL flag in the FSTAT register will set and the mass erase command will not launch. Once the mass erase command has successfully launched, the CCIF flag in the FSTAT register will set after the mass erase operation has completed unless a second command has been buffered.

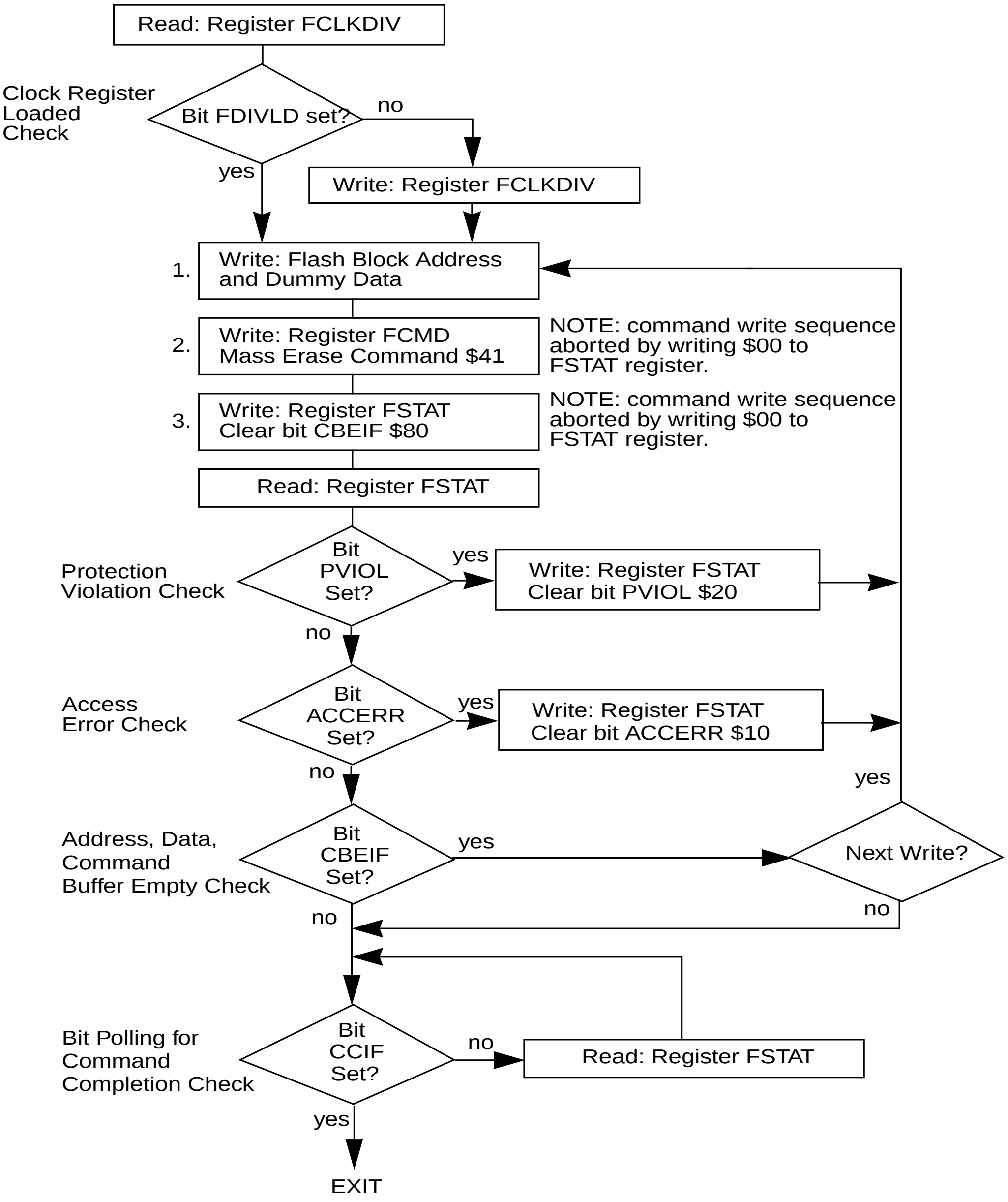

Read: Register FCLKDIV
Clock Register Loaded Check
Bit FDIVLD set?
no
yes
Write: Register FCLKDIV
1.
Write: Flash Block Address and Dummy Data
2.
Write: Register FCMD Mass Erase Command $41
NOTE: command write sequence aborted by writing $00 to FSTAT register.
3.
Write: Register FSTAT Clear bit CBEIF $80
NOTE: command write sequence aborted by writing $00 to FSTAT register.
Read: Register FSTAT
Protection Violation Check
Bit PVIOL Set?
yes
Write: Register FSTAT Clear bit PVIOL $20
no
Access Error Check
Bit ACCERR Set?
yes
Write: Register FSTAT Clear bit ACCERR $10
no
yes
Address, Data, Command Buffer Empty Check
Bit CBEIF Set?
yes
Next Write?
no
no
Bit Polling for Command Completion Check
Bit CCIF Set?
no
Read: Register FSTAT
yes
EXIT

Figure 4-6 Example Mass Erase Command Flow

4.1.3.6 Sector Erase Abort Command

The sector erase abort command is used to terminate the sector erase operation so that other sectors in the Flash block are available for read and program operations without waiting for the sector erase operation to complete. If the sector erase abort command is launched resulting in the early termination of an active sector erase operation, the ACCERR flag will set once the operation completes as indicated by the CCIF flag being set. The ACCERR flag sets to inform the user that the sector may not be fully erased and a new sector erase command must be launched before programming any location in that specific sector. If the sector erase abort command is launched but the active sector erase operation completes normally, the ACCERR flag will not set upon completion of the operation as indicated by the CCIF flag being set. Therefore, if the ACCERR flag is not set after the sector erase abort command has completed, the sector being erased when the abort command was launched is fully erased. The maximum number of cycles required to abort a sector erase operation is equal to four FCLK periods (see section **4.1.1**) plus five bus cycles as measured from the time the CBEIF flag is cleared until the CCIF flag is set.

NOTE

Since the ACCERR bit in the FSTAT register may be set at the completion of the sector erase abort operation, a command write sequence is not allowed to be buffered behind a sector erase abort command write sequence. The CBEIF flag will not set after launching the sector erase abort command to indicate that a command should not be buffered behind it. If an attempt is made to start a new command write sequence with a sector erase abort operation active, the ACCERR flag in the FSTAT register will be set. A new command write sequence may be started after clearing the ACCERR flag, if set.

WARNING

The sector erase abort command should be used sparingly since a sector erase operation that is aborted counts as a complete program/erase cycle.

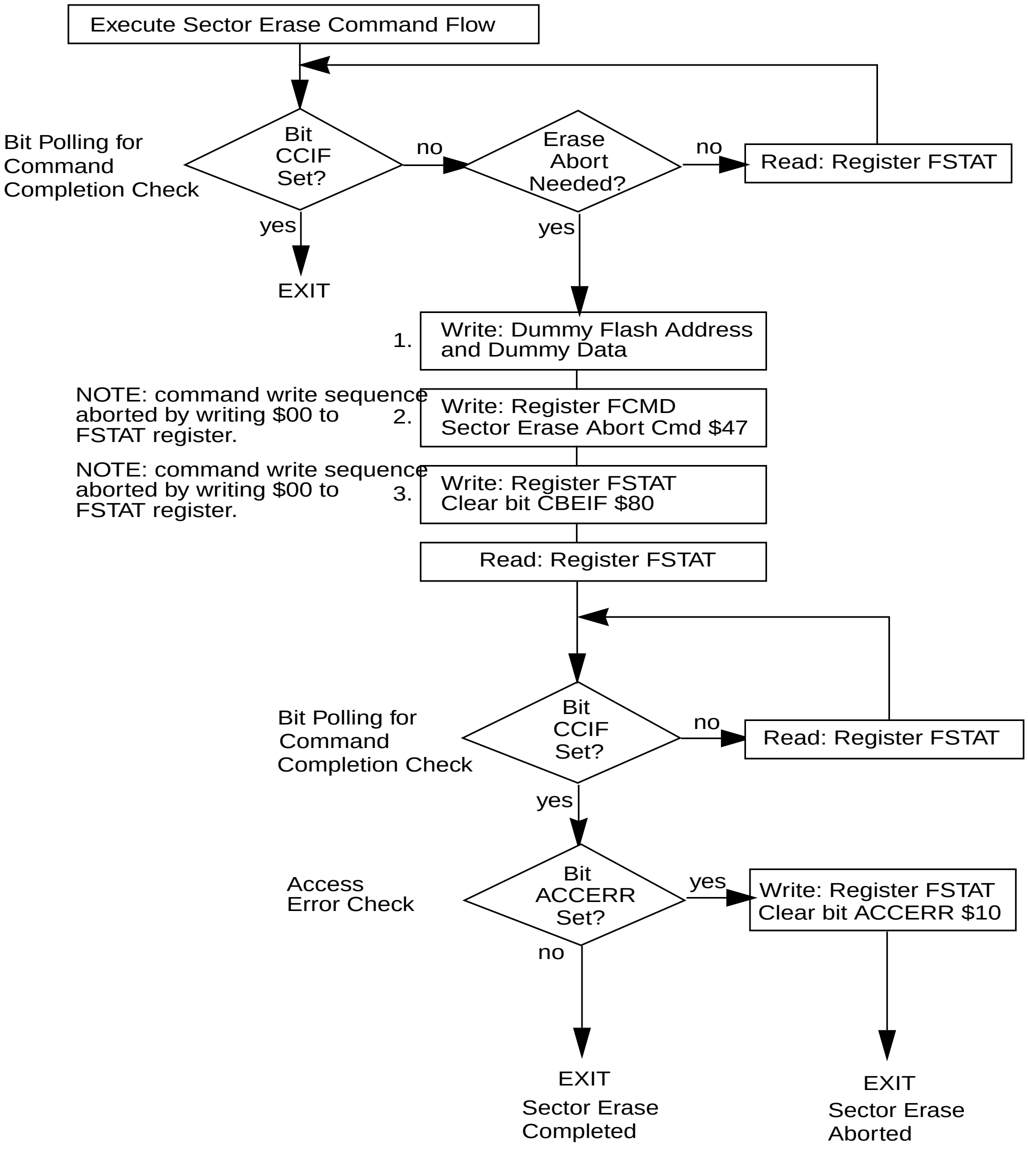

Execute Sector Erase Command Flow
Bit Polling for Command Completion Check
Bit CCIF Set?
no
Erase Abort Needed?
no
Read: Register FSTAT
yes
EXIT
yes
1.
Write: Dummy Flash Address and Dummy Data
NOTE: command write sequence aborted by writing $00 to FSTAT register.
2.
Write: Register FCMD Sector Erase Abort Cmd $47
NOTE: command write sequence aborted by writing $00 to FSTAT register.
3.
Write: Register FSTAT Clear bit CBEIF $80
Read: Register FSTAT
Bit Polling for Command Completion Check
Bit CCIF Set?
no
Read: Register FSTAT
yes
Access Error Check
Bit ACCERR Set?
yes
Write: Register FSTAT Clear bit ACCERR $10
no
EXIT
Sector Erase Completed
EXIT
Sector Erase Aborted

Figure 4-7 Example Sector Erase Abort Command Flow

4.1.4 Illegal Flash Operations

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

1. Writing to a Flash address before initializing the FCLKDIV register.
2. Writing to a Flash address in the range $8000-$BFFF when the 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 a Flash address in the range $4000-$7FFF or $C000-$FFFF with the BKSEL bit in the

FCNFG register not selecting Flash block 0.

4. Writing a byte or misaligned word to a valid Flash address.
5. Starting a command write sequence while a data compress operation is active.
6. Starting a command write sequence while a sector erase abort operation is active.
7. Writing a second word to a Flash address in the same command write sequence.
8. Writing to any Flash register other than FCMD after writing a word to a Flash address.
9. Writing a second command to the FCMD register in the same command write sequence.
10. Writing an invalid command to the FCMD register.
11. When security is enabled, writing a command other than mass erase to the FCMD register when the write originates from a non-secure memory location or from the Background Debug Mode.
12. Writing to any Flash register other than FSTAT (to clear CBEIF) after writing to the FCMD register.
13. Writing a “0” to the CBEIF flag in the FSTAT register to abort a command write sequence.

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

The ACCERR flag will also be set if any of the following events occur:

1. Launching the sector erase abort command while a sector erase operation is active which results in the early termination of the sector erase operation (see section **4.1.3.6**).
2. A double bit fault is detected in any of the following Flash operations:
    a. Array read.
    b. Erase Verify.
    c. Data Compress.
    d. Reset Sequence Array Read (Configuration Field).
3. The MCU enters STOP mode and a program or erase operation is in progress. The operation is aborted immediately and any pending command is purged (see section **4.3**).

If the Flash memory is read during execution of an algorithm (i.e. CCIF flag in the FSTAT register is low), the read operation will return invalid data and the ACCERR flag will not be set.

If the ACCERR flag is set in the FSTAT register, the user must clear the ACCERR flag before starting another command write sequence (see section **3.3.6**).

The PVIOL flag will be set after the command is written to the FCMD register during a command write sequence if any of the following illegal operations are attempted, causing the command write sequence to immediately abort:

1. Writing the program command if the address written in the command write sequence was in a protected area of the Flash memory.
2. Writing the sector erase command if the address written in the command write sequence was in a protected area of the Flash memory.

3. Writing the mass erase command while any Flash protection is enabled.

If the PVIOL flag is set in the FSTAT register, the user must clear the PVIOL flag before starting another command write sequence (see section **3.3.6**).

4.2 Wait Mode

If a command is active (CCIF=0) when the MCU enters the WAIT mode, the active command and any buffered command will be completed.

The Flash module can recover the MCU from WAIT if the CBEIF and CCIF 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 operation will be aborted and, if the operation is program or erase, the Flash array data being programmed or erased may be corrupted and the CCIF and ACCERR flags will be set. If active, the high voltage circuitry to the Flash memory will immediately be switched off when entering STOP mode. Upon exit from STOP, the CBEIF flag is set and any buffered command will not be launched. The ACCERR flag must be cleared before starting a command write sequence (see section **4.1.2**).

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 or erase operations.

4.4 Background Debug Mode

In background debug mode (BDM), the FPROT register is 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 commands available to execute are MASERS and LDPTMR.

4.5 Flash Module 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 security byte at $FF0F in the Flash Configuration Field must be changed directly by programming $FF0F when the MCU is unsecured and the higher address sector is unprotected. If the Flash security byte is left in a secured state, any reset will cause the MCU to initialize to a secure operating mode.

4.5.1 Unsecuring the Flash Module via the Backdoor Key Access

The MCU may 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 the KEYEN[1:0] bits are in the enabled state (see section **3.3.2**) and the KEYACC bit is set, a write to a backdoor key address in the Flash memory triggers a comparison between the written data and the backdoor key data stored in the Flash memory. 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 memory, 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 are not permitted as backdoor keys. While the KEYACC bit is set, reads of the Flash memory will return invalid data.

The user code stored in the Flash memory 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 the KEYEN[1:0] bits are in the enabled state (see section **3.3.2**), 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 the SEC[1:0] bits in the FSEC register are forced to the unsecure state of “10”.

The backdoor key access sequence is monitored by the internal security state machine. An illegal operation during the backdoor key 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 key access sequence to be attempted. The following operations during the backdoor key access sequence 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. Double bit faults detected while reading the backdoor keys from the Flash array are ignored.
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.
6. If any two of the four 16-bit words are written on successive MCU clock cycles.

After the backdoor keys have been correctly matched, the MCU will be unsecured. Once the MCU is 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 backdoor keys by programming addresses $FF00 - $FF07 in the Flash Configuration Field.

The security as defined in the Flash security byte ($FF0F) is not changed by using the backdoor key access sequence to unsecure. The backdoor keys stored in addresses $FF00 - $FF07 are unaffected by the backdoor key access sequence. After the next reset of the MCU, the security state of the Flash module is determined by the Flash security byte ($FF0F). The backdoor key access sequence has no effect on the program and erase protections defined in the Flash protection register.

It is not possible to unsecure the MCU in special single chip mode by using the backdoor key access sequence via the background debug mode (BDM).

4.5.2 Unsecuring the Flash Module in Special Single Chip Mode via the BDM

The MCU can be unsecured in special single chip mode by erasing the Flash module by the following method:

- Reset the MCU into special single chip mode, delay while the erase test is performed by the BDM secure ROM, send BDM commands to disable protection in the Flash module, and execute a mass erase command write sequence to erase the Flash memory.

After the CCIF flag sets to indicate that the mass operation has completed, reset the MCU into special single chip mode. The BDM secure ROM will verify that the Flash memory is erased and will assert the UNSEC bit in the BDM status register. This BDM action will cause the MCU to override the Flash security state and the MCU will be unsecured. All BDM commands will be enabled and the Flash security byte may be programmed to the unsecure state by the following method:

- Send BDM commands to execute a word program sequence to program the Flash security byte to the unsecured state and reset the MCU.

Section 5 Resets

5.1 General

If a reset occurs while any Flash 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.

5.2 Flash Reset Sequence

On each reset, the Flash module executes a reset sequence to hold CPU activity while loading the following registers from the Flash memory according to **Table 3-1**:

- FPROT - Flash Protection Register (see section **3.3.5**).

  If a double bit fault is detected during the read of the protection field as part of the reset sequence, the FPOPEN bit in the FPROT register will be clear and remaining bits will be set leaving the Flash block fully protected from program and erase.

- FCTL - Flash Control Register (see section **3.3.8**).

  If a double bit fault is detected during the read of the non-volatile byte as part of the reset sequence, all bits in the FCTL register will be set.

- FSEC - Flash Security Register (see section **3.3.2**).

  If a double bit fault is detected during the read of the security field as part of the reset sequence, all bits in the FSEC register will be set leaving the Flash module in a secure state with Backdoor Key Access disabled.

If a double bit fault is detected during array reads as part of the reset sequence, the ACCERR flag will set in the FSTAT register of all banks.

Section 6 Interrupts

6.1 General

The Flash module can generate an interrupt when all Flash command operations have completed, when the Flash address, data and command buffers are empty, or when a Flash array read or operation has detected a double bit fault.

Table 6-1 Flash Interrupt Sources

Interrupt Source	Interrupt Flag	Local Enable	Global (CCR) Mask
Flash Address, Data and Command Buffers empty	CBEIF (FSTAT register)	CBEIE (FCNFG register)	I-Bit
All Flash commands completed	CCIF (FSTAT register)	CCIE (FCNFG register)	I-Bit
Flash array read or verify operation detected a double bit fault	DFDIF (FSTAT register)	DFDIE (FCNFG register)	I-Bit

NOTE

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

6.2 Description of Flash Interrupt Operation

The logic used for generating interrupts is shown in **Figure 6-1**.

The Flash module uses the CBEIF and CCIF flags in combination with the CBIE and CCIE enable bits to generate the Flash command interrupt request. The Flash module uses the DFDIF flag in combination with the DFDIE enable bit to generate the Flash double fault detect interrupt request.

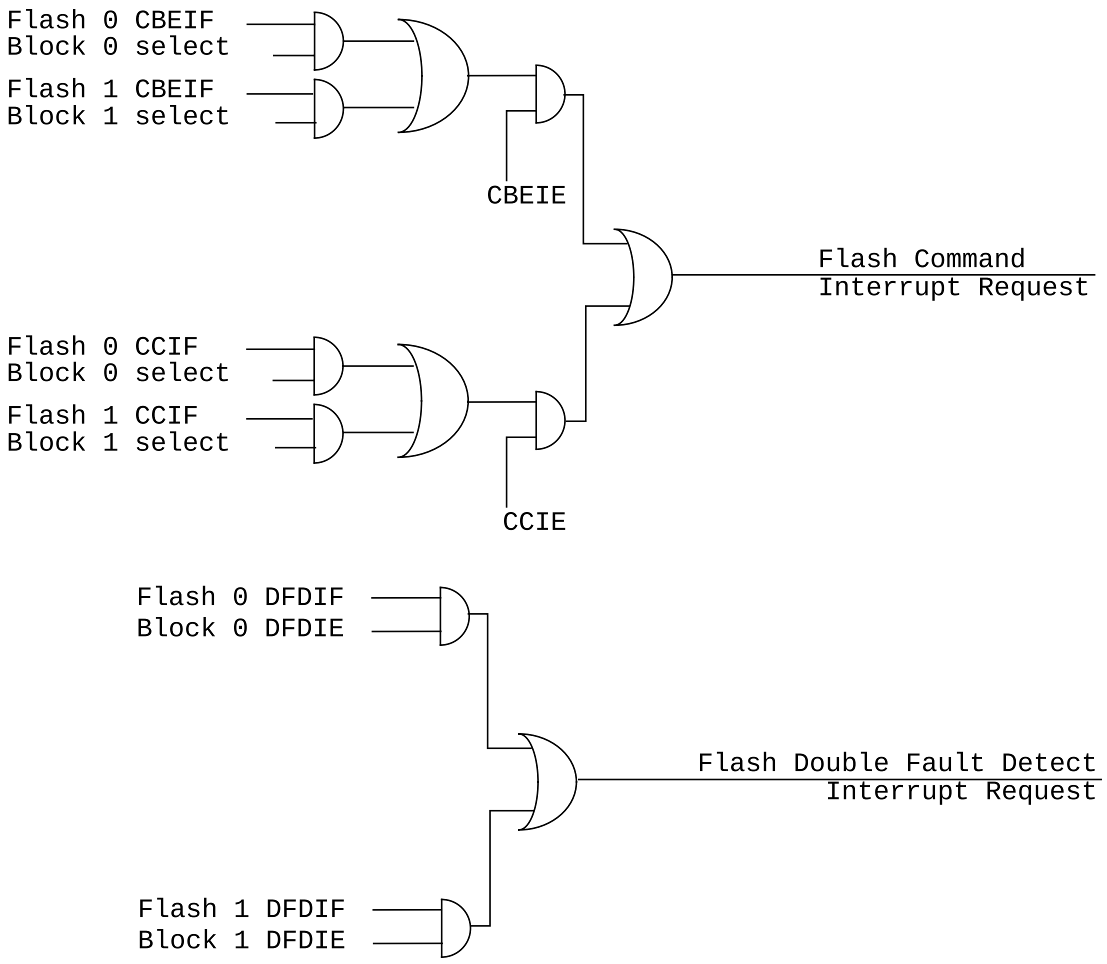


Figure 6-1 Flash Interrupt Implementation

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

Block User Guide End Sheet

FINAL PAGE OF
62
PAGES