DOCUMENT NUMBER
S12XEETX4KV2

EETX4K
Block Guide
V2.0

Original Release Date: 10 NOV 2003
Revised: 30 JUN 2004

Motorola, Inc.

Revision History

Version Number	Revision Date	Effective Date	Author	Description of Changes
01.00	10NOV03	10NOV03		Initial Version
02.00	07JUN04	30JUN04		
2.1	21 JUL 2004	22 JUL 2004		

Table of Contents

Section 1 Introduction

Section 2 External Signal Description

Section 3 Memory Map and Registers

Section 4 Functional Description

List of Figures

List of Tables

Section 1 Introduction

1.1 Overview

This document describes the EETX4K module which includes a 4K byte EEPROM (Non-Volatile) memory. The EEPROM 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.

The EEPROM memory is ideal for data storage for single-supply applications allowing for field reprogramming without requiring external voltage sources for program or erase. Program and erase functions are controlled by a command driven interface. The EEPROM module supports both block erase (all memory bytes) and sector erase (4 memory bytes). An erased bit reads ‘1’ and a programmed bit reads ‘0’. The high voltage required to program and erase the EEPROM memory is generated internally. It is not possible to read from the EEPROM block while it is being erased or programmed.

> ***NOTE:*** *An EEPROM word (2 bytes) must be erased before being programmed. Cumulative programming of bits within a word is not allowed.*

1.1.1 Glossary

Command Write Sequence

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

1.2 A subset of the MCU modes, defined by input signals “mmc_mode_ss”=”1”, or “mmc_mode_st”=”1”Features

- 4K bytes of EEPROM memory divided into 1024 sectors of 4 bytes.
- Automated program and erase algorithm.
- Interrupts on EEPROM command completion and command buffer empty.
- Fast sector erase and word program operation.
- 2-stage command pipeline.
- Sector erase abort feature for critical interrupt response.
- Flexible protection scheme to prevent accidental program or erase.
- Single power supply for all EEPROM operations including program and erase.

1.3 Modes of Operation

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

1.4 Block Diagram

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

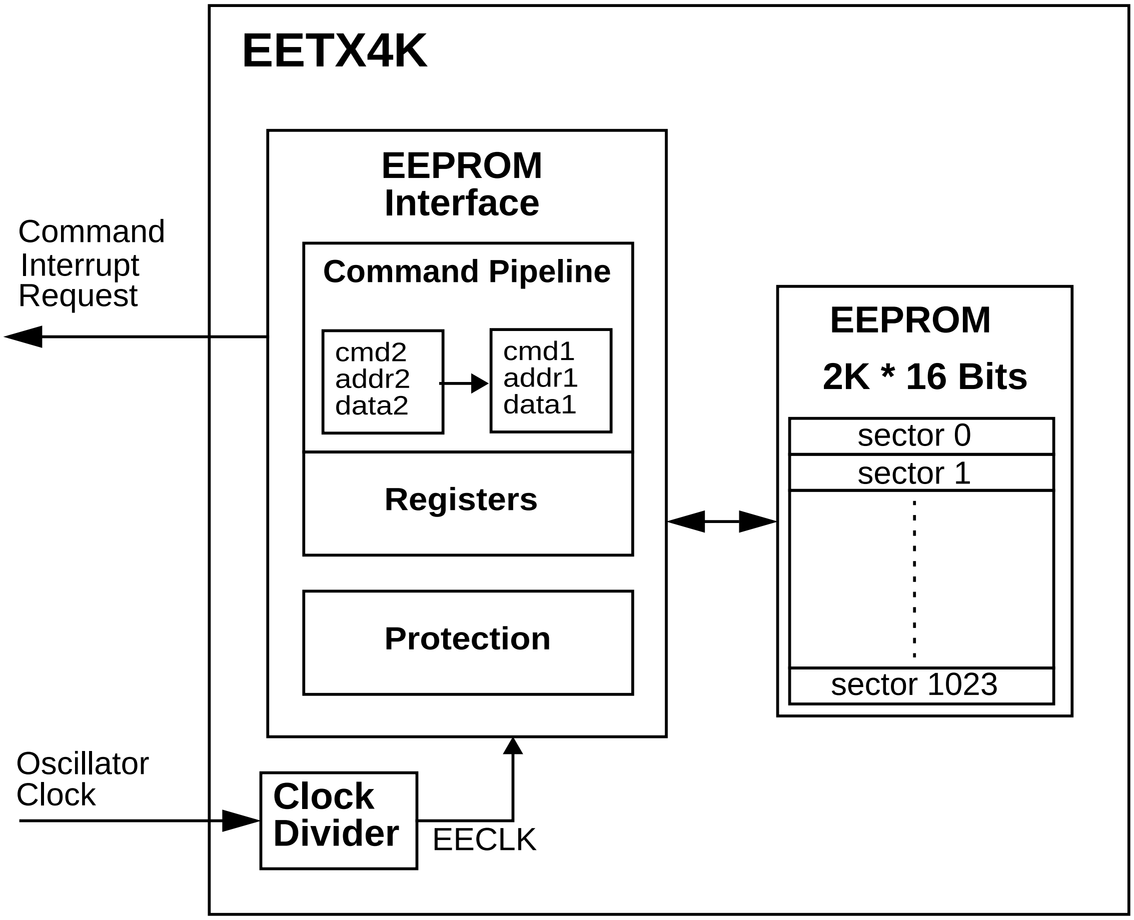

Figure 1-1 Module Block Diagram

Section 2 External Signal Description

2.1 Overview

The EEPROM 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 EEPROM module.

3.2 Module Memory Map

A linear EEPROM memory map is shown in **Figure 3-1**. The HCS12X architecture actually places the EEPROM memory addresses between logical addresses $0800 and $1000 with $0800 to $0BFF representing 1K byte of paged EEPROM memory and $0C00 to $0FFF representing 1K byte of fixed EEPROM memory. The EPROT register, described in section **3.3.5**, can be set to protect the upper region in the EEPROM memory from accidental program or erase. The EEPROM addresses covered by this protectable region are shown in the EEPROM memory map. The default protection setting is stored in the EEPROM configuration field as described in **Table 3-1**.

Table 3-1 EEPROM Configuration Field

EEPROM Memory Address Offset	Size (bytes)	Description
$_FFC	1	Reserved
$_FFD	1	EEPROM Protection byte Refer to Section **3.3.5 EPROT — EEPROM Protection Register**
$_FFE - $_FFF	2	Reserved

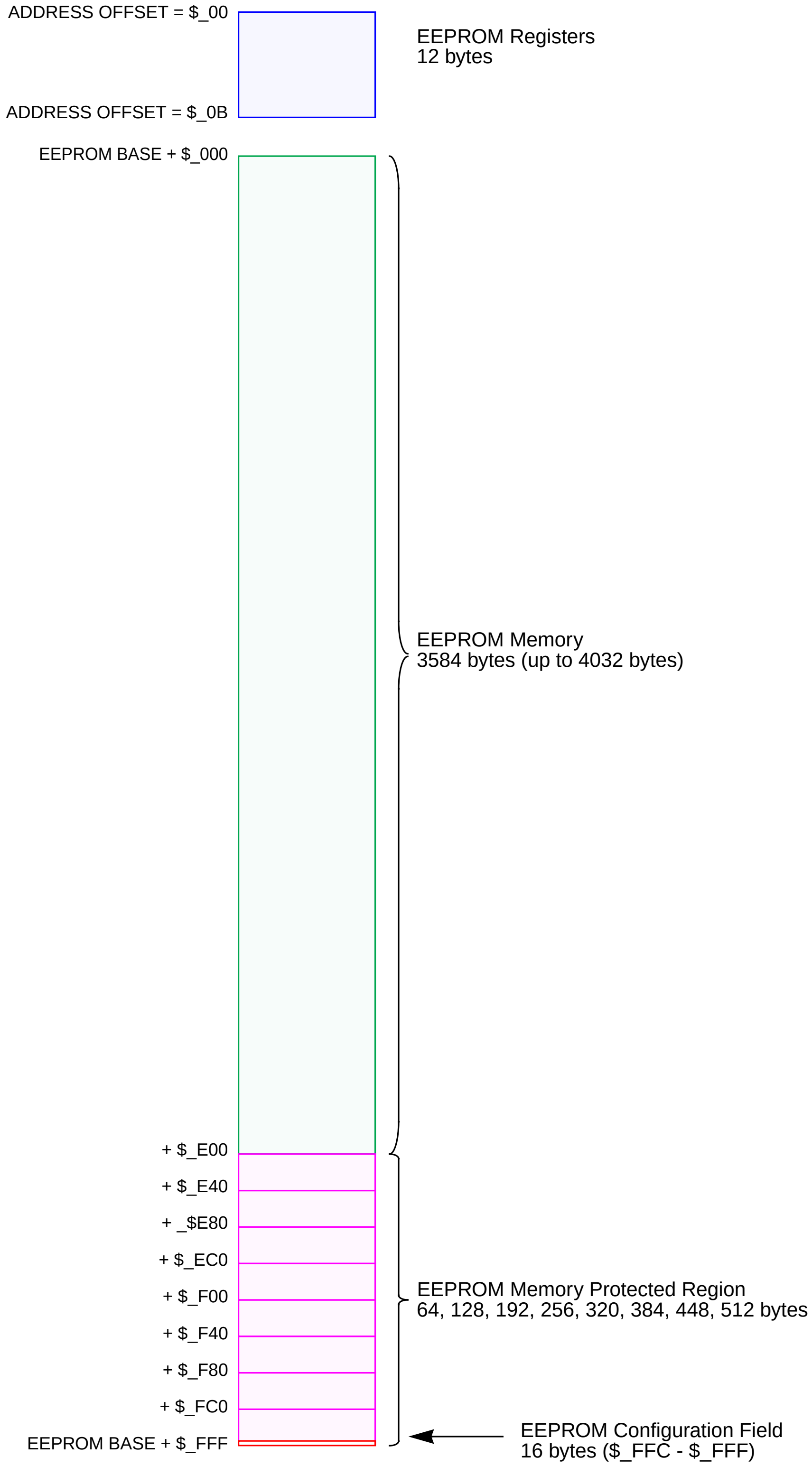

Figure 3-1 EEPROM Memory Map

The EEPROM module also contains a set of 12 control and status registers located between EEPROM register address offsets $_00 and $_0B. A summary of the EEPROM module registers is given in **Table 3-2** while their accessibility is detailed in section **3.3**.

Table 3-2 EEPROM Register Map

Address Offset	Register Name	Normal Mode Access
$_00	EEPROM Clock Divider Register (ECLKDIV)	R/W
$_01	RESERVED1[1]	R
$_02	RESERVED2[1]	R
$_03	EEPROM Configuration Register (ECNFG)	R/W
$_04	EEPROM Protection Register (EPROT)	R/W
$_05	EEPROM Status Register (ESTAT)	R/W
$_06	EEPROM Command Register (ECMD)	R/W
$_07	RESERVED3[1]	R
$_08	EEPROM High Address Register (EADDRHI)[1]	R
$_09	EEPROM Low Address Register (EADDRLO)[1]	R
$_0A	EEPROM High Data Register (EDATAHI)[1]	R
$_0B	EEPROM Low Data Register (EDATALO)[1]	R

NOTES:

1. Intended for factory test purposes only.

NOTE: *Register Address = EEPROM Control Register Base Address + Address Offset, where the EEPROM Control Register Base Address is defined at the MCU level and the Address Offset is defined at the EEPROM module level.*

3.3 Register Descriptions

3.3.1 ECLKDIV — EEPROM Clock Divider Register

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

Address Offset: $_00

	7	6	5	4	3	2	1	0
R	EDIVLD	PRDIV8	EDIV5	EDIV4	EDIV3	EDIV2	EDIV1	EDIV0
W	▨							
RESET:	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 3-2 EEPROM Clock Divider Register (ECLKDIV)

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

EDIVLD — Clock Divider Loaded.

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

PRDIV8 — Enable Prescalar by 8.

1 = Enables a prescalar by 8, to divide the oscillator clock before feeding into the clock divider.
0 = The oscillator clock is directly fed into the ECLKDIV divider.

EDIV[5:0] — Clock Divider Bits.

The combination of PRDIV8 and EDIV[5:0] effectively divides the EEPROM 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 RESERVED1

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

Address Offset: $_01

	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 3-3 RESERVED1

All bits read zero and are not writable.

3.3.3 RESERVED2

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

Address Offset: $_02

	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 3-4 RESERVED2

All bits read zero and are not writable.

3.3.4 ECNFG — EEPROM Configuration Register

The ECNFG register enables the EEPROM interrupts.

Address Offset: $_03

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

▨ = Unimplemented or Reserved

Figure 3-5 EEPROM Configuration Register (ECNFG)

CBEIE and CCIE bits are readable and writable while all remaining bits read zero and are not writable.

CBEIE — Command Buffer Empty Interrupt Enable.

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

3.3.5 EPROT — EEPROM Protection Register

The EPROT register defines which EEPROM sectors are protected against program or erase operations.

Address Offset: $_04

	7	6	5	4	3	2	1	0
R	EPOPEN	RNV6	RNV5	RNV4	EPDIS	EPS2	EPS1	EPS0
W		▨	▨	▨				
Reset:	F	F	F	F	F	F	F	F

▨ = Unimplemented or Reserved

Figure 3-6 EEPROM Protection Register (EPROT)

During the reset sequence, the EPROT register is **loaded** from the EEPROM Protection byte at address offset $_FFD (see **Table 3-1 EEPROM Configuration Field**).All bits in the EPROT register are readable and writable except for RNV[6:4] which are only readable. The EPOPEN and EPDIS bits can only be written to the protected state. The EPS bits can be written anytime until bit EPDIS is cleared. If the EPOPEN bit is cleared, the state of the EPDIS and EPS bits is irrelevant.

To change the EEPROM protection that will be loaded during the reset sequence, the EEPROM memory must be unprotected, then the EEPROM Protection byte must be reprogrammed. Trying to alter data in any protected area in the EEPROM memory will result in a protection violation error and the PVIOL flag will be set in the ESTAT register. The mass erase of an EEPROM block is possible only when protection is fully disabled by setting the EPOPEN and EPDIS bits.

EPOPEN — Opens the EEPROM for program or erase.
1 = The EEPROM sectors not protected are enabled for program or erase.
0 = The entire EEPROM memory is protected from program and erase.

RNV[6:4] — Reserved Non-Volatile Bits.

The RNV[6:4] bits should remain in the erased state “1” for future enhancements.

EPDIS — EEPROM Protection address range Disable.

The EPDIS bit determines whether there is a protected area in a specific region of the EEPROM memory ending with address offset $_FFF.
1 = Protection disabled.
0 = Protection enabled.

EPS[2:0] — EEPROM Protection Address Size.

The EPS[2:0] bits determine the size of the protected area as shown in**Table 3-3**. The EPS bits can only be written to while the EPDIS bit is set.

Table 3-3 EEPROM Protection Address Range

EPS[2:0]	Address Offset Range	Protected Size
000	$_FC0-$_FFF	64 bytes
001	$_F80-$_FFF	128 bytes
010	$_F40-$_FFF	192 bytes
011	$_F00-$_FFF	256 bytes
100	$_EC0-$_FFF	320 bytes
101	$_E80-$_FFF	384 bytes
110	$_E40-$_FFF	448 bytes
111	$_E00-$_FFF	512 bytes

3.3.6 ESTAT — EEPROM Status Register

The ESTAT register defines the operational status of the module.

Address Offset: $_05

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

▨ = Unimplemented or Reserved

Figure 3-7 EEPROM Status Register (ESTAT)

CBEIF, PVIOL and ACCERR 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 EEPROM 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 ECNFG register to generate an interrupt request (see **Figure 4-8**).

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 ECNFG register to generate an interrupt request (see **Figure 4-8**).

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 EEPROM memory 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 has occurred to the EEPROM memory caused by either a violation of the command write sequence (see section **4.1.2**), issuing an illegal EEPROM command (see **Table 3-4**), launching the sector erase abort command terminating a sector erase operation early (see section **4.1.3.5**) 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 an erase verify operation, any buffered command will not launch.

1 = Access error has occurred.
0 = No access error 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 EEPROM 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 = EEPROM block verified as erased.
0 = EEPROM block verified as not erased.

3.3.7 ECMD — EEPROM Command Register

The ECMD register is the EEPROM command register.

Address Offset: $_06

	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 3-8 EEPROM Command Register (ECMD)

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 EEPROM commands are shown in **Table 3-4**. Writing any command other than those listed in **Table 3-4** sets the ACCERR flag in the ESTAT register.

Table 3-4 Valid EEPROM Command List

CMDB[6:0]	Command
$05	Erase Verify
$20	Word Program
$40	Sector Erase
$41	Mass Erase
$47	Sector Erase Abort
$60	Sector Modify

3.3.8 RESERVED3

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

Address Offset: $_07

	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 3-9 RESERVED3

All bits read zero and are not writable.

3.3.9 EADDR — EEPROM Address Registers

The EADDRHI and EADDRLO registers are the EEPROM address registers.

Address Offset: $_08

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

▨ = Unimplemented or Reserved

Figure 3-10 EEPROM Address High Register (EADDRHI)

Address Offset: $_09

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

▨ = Unimplemented or Reserved

Figure 3-11 EEPROM Address Low Register (EADDRLO)

All EABHI and EABLO bits read zero and are not writable in normal modes.

All EABHI and EABLO bits are readable and writable in special modes.

The MCU address bit AB0 is not stored in the EADDR registers since the EEPROM block is not byte addressable.

3.3.10 EDATA — EEPROM Data Registers

The EDATAHI and EDATALO registers are the EEPROM data registers.

Address Offset: $_0A

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

▨ = Unimplemented or Reserved

Figure 3-12 EEPROM Data High Register (EDATAHI)

Address Offset: $_0B

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

▨ = Unimplemented or Reserved

Figure 3-13 EEPROM Data Low Register (EDATALO)

All EDHI and EDLO bits read zero and are not writable in normal modes.

All EDHI and EDLO bits are readable and writable in special modes.

Section 4 Functional Description

4.1 EEPROM Command Operations

Write and read operations are both used for the program, erase, erase verify, sector erase abort, and sector modify algorithms described in this section. The program, erase, and sector modify algorithms are controlled by a state machine whose timebase, EECLK, 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. Buffer empty as well as command completion are signalled by flags in the EEPROM status register with interrupts generated, if enabled.

The next sections describe:

1. How to write the ECLKDIV register.
2. Command write sequences to program, erase, erase verify, sector erase abort, and sector modify operations on the EEPROM memory.
3. Valid EEPROM commands.
4. Effects resulting from illegal EEPROM command write sequences or aborting EEPROM operations.

4.1.1 Writing the ECLKDIV Register

Prior to issuing any EEPROM command after a reset, the user is required to write the ECLKDIV 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 ECLKDIV determination must take this information into account.

If we define:

- ECLK as the clock of the EEPROM 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 ECLKDIV register bits PRDIV8 and EDIV[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, ECLKDIV bits EDIV[5:0] should be set to "4" (000100) and bit PRDIV8 set to "0". The resulting EECLK frequency is then 190kHz. As a result, the EEPROM program and erase algorithm timings are increased over the optimum target by:

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

If the oscillator clock frequency is 16MHz and the bus clock frequency is 40MHz, ECLKDIV bits EDIV[5:0] should be set to "50" (110010) and bit PRDIV8 set to "1". The resulting EECLK frequency is then 182kHz. In this case, the EEPROM program and erase algorithm timings are increased over the optimum target by:

$$(200 - 182) / 200 \times 100 = 9\%$$

NOTE: *"4"Program and erase command execution time will increase proportionally with the period of EECLK.*

NOTE: *Because of the impact of clock synchronization on the accuracy of the functional timings, programming or erasing the EEPROM memory cannot be performed if the bus clock runs at less than 1 MHz. Programming or erasing the EEPROM memory with EECLK < 150kHz should be avoided. Setting ECLKDIV to a value such that EECLK < 150kHz can destroy the EEPROM memory due to overstress. Setting ECLKDIV to a value such that (1/EECLK+Tbus) < 5µs can result in incomplete programming or erasure of the EEPROM memory cells.*

If the ECLKDIV register is written, the EDIVLD bit is set automatically. If the EDIVLD bit is zero, the ECLKDIV register has not been written since the last reset. If the ECLKDIV register has not been written to, the EEPROM command loaded during a command write sequence will not execute and the ACCERR flag in the ESTAT register will set.

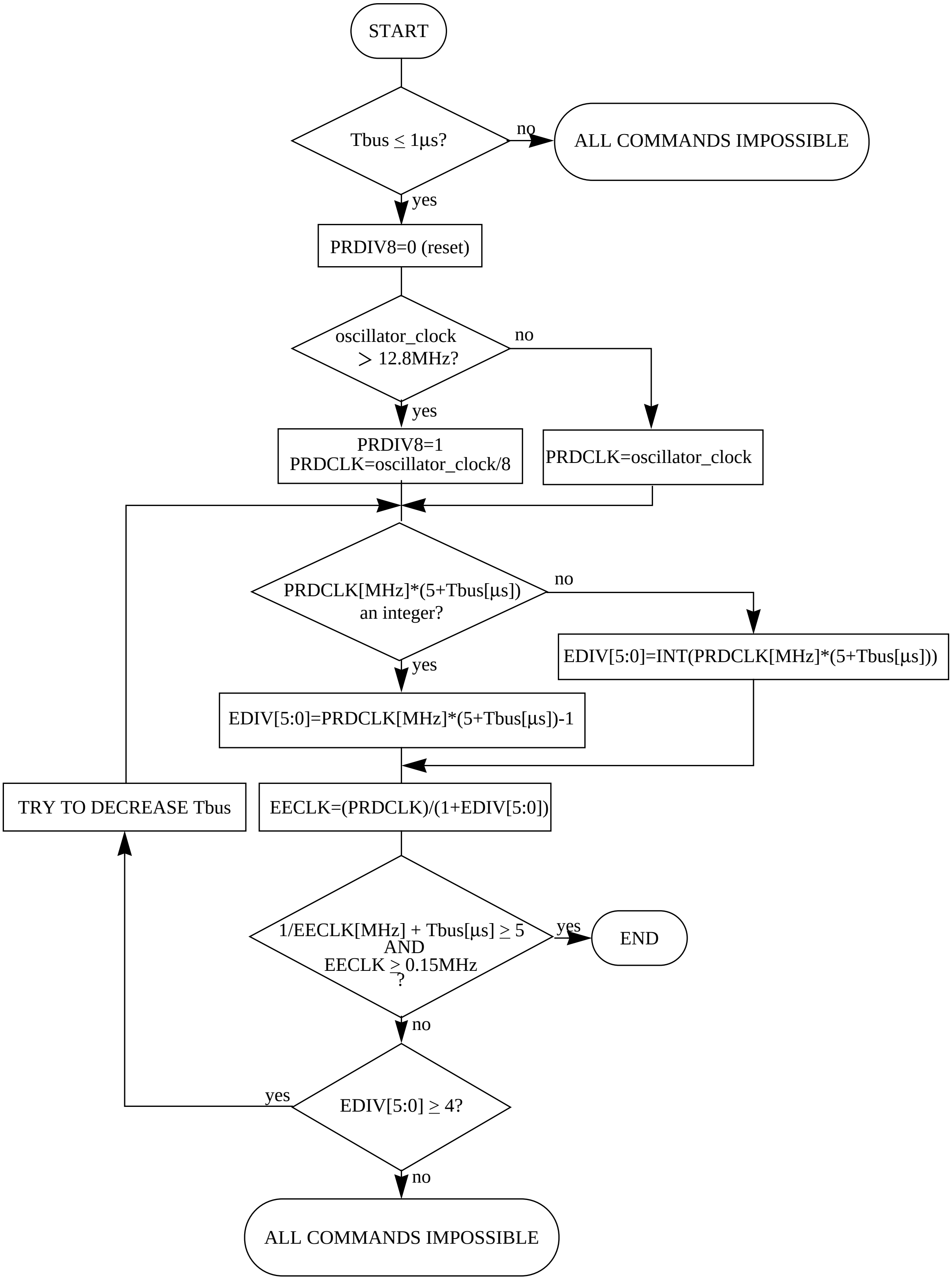
START
Tbus ≤ 1µs?
no
ALL COMMANDS IMPOSSIBLE
yes
PRDIV8=0 (reset)
oscillator_clock > 12.8MHz?
no
yes
PRDIV8=1
PRDCLK=oscillator_clock/8
PRDCLK=oscillator_clock
PRDCLK[MHz]*(5+Tbus[µs]) an integer?
no
EDIV[5:0]=INT(PRDCLK[MHz]*(5+Tbus[µs]))
yes
EDIV[5:0]=PRDCLK[MHz]*(5+Tbus[µs])-1
TRY TO DECREASE Tbus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

Figure 4-1 Determination Procedure for PRDIV8 and EDIV Bits

4.1.2 Command Write Sequence

The EEPROM command controller is used to supervise the command write sequence to execute program, erase, erase verify, sector erase abort, and sector modify algorithms.

Before starting a command write sequence, the ACCERR and PVIOL flags in the ESTAT 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 EEPROM module not permitted between the steps. However, EEPROM register and array reads are allowed during a command write sequence. The basic command write sequence is as follows:

1. Write to one address in the EEPROM memory.
2. Write a valid command to the ECMD register.
3. Clear the CBEIF flag in the ESTAT register by writing a “1” to CBEIF to launch the command.

The address written in step 1 will be stored in the EADDR registers and the data will be stored in the EDATA registers. If the CBEIF flag in the ESTAT register is clear when the first EEPROM array write occurs, the contents of the address and data buffers will be overwritten and the CBEIF flag will be set. When the CBEIF flag is cleared, the CCIF flag is cleared on the same bus cycle by the EEPROM command controller indicating that the command was successfully launched. For all command write sequences except 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 sector erase abort operations, the CBEIF flag will remain clear until the operation completes. 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 in the ESTAT register. The CCIF flag will set upon completion of all active and buffered commands .

A command write sequence can be aborted prior to clearing the CBEIF flag in the ESTAT register by writing a “0” to the CBEIF flag and will result in the ACCERR flag in the ESTAT register being set. The ACCERR flag in the ESTAT register must be cleared prior to starting a new command write sequence.

4.1.3 EEPROM Commands

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

Table 4-1 EEPROM Command Description

ECMDB	Command	Function on EEPROM Memory
$05	Erase Verify	Verify all memory bytes in the EEPROM block are erased. If the EEPROM block is erased, the BLANK flag in the ESTAT register will set upon command completion.

Table 4-1 EEPROM Command Description

ECMDB	Command	Function on EEPROM Memory
$20	Program	Program a word (two bytes) in the EEPROM block.
$40	Sector Erase	Erase all four memory bytes in a sector of the EEPROM block.
$41	Mass Erase	Erase all memory bytes in the EEPROM block. A mass erase of the full EEPROM block is only possible when EPOPEN and EPDIS bits in the EPROT 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 EEPROM sector should not be considered erased if the ACCERR flag is set upon command completion.
$60	Sector Modify	Erase all four memory bytes in a sector of the EEPROM block and reprogram the addressed word.

NOTE: *The user should not program an EEPROM word without first erasing the sector in which that word resides.*

4.1.3.1 Erase Verify Command

The erase verify operation will verify that the EEPROM memory is erased.

An example flow to execute the erase verify operation is shown in **Figure 4-2**. The erase verify command write sequence is as follows:

1. Write to an EEPROM address to start the command write sequence for the erase verify command. The address and data written will be ignored.
2. Write the erase verify command, $05, to the ECMD register.
3. Clear the CBEIF flag in the ESTAT register by writing a “1” to CBEIF to launch the erase verify command.

After launching the erase verify command, the CCIF flag in the ESTAT register will set after the operation has completed unless a new command write sequence has been buffered. The number of bus cycles required to execute the erase verify operation is equal to the number of words in the EEPROM memory plus 14 bus cycles as measured from the time the CBEIF flag is cleared until the CCIF flag is set. Upon completion of the erase verify operation, the BLANK flag in the ESTAT register will be set if all addresses in the EEPROM memory are verified to be erased. If any address in the EEPROM memory is not erased, the erase verify operation will terminate and the BLANK flag in the ESTAT register will remain clear.

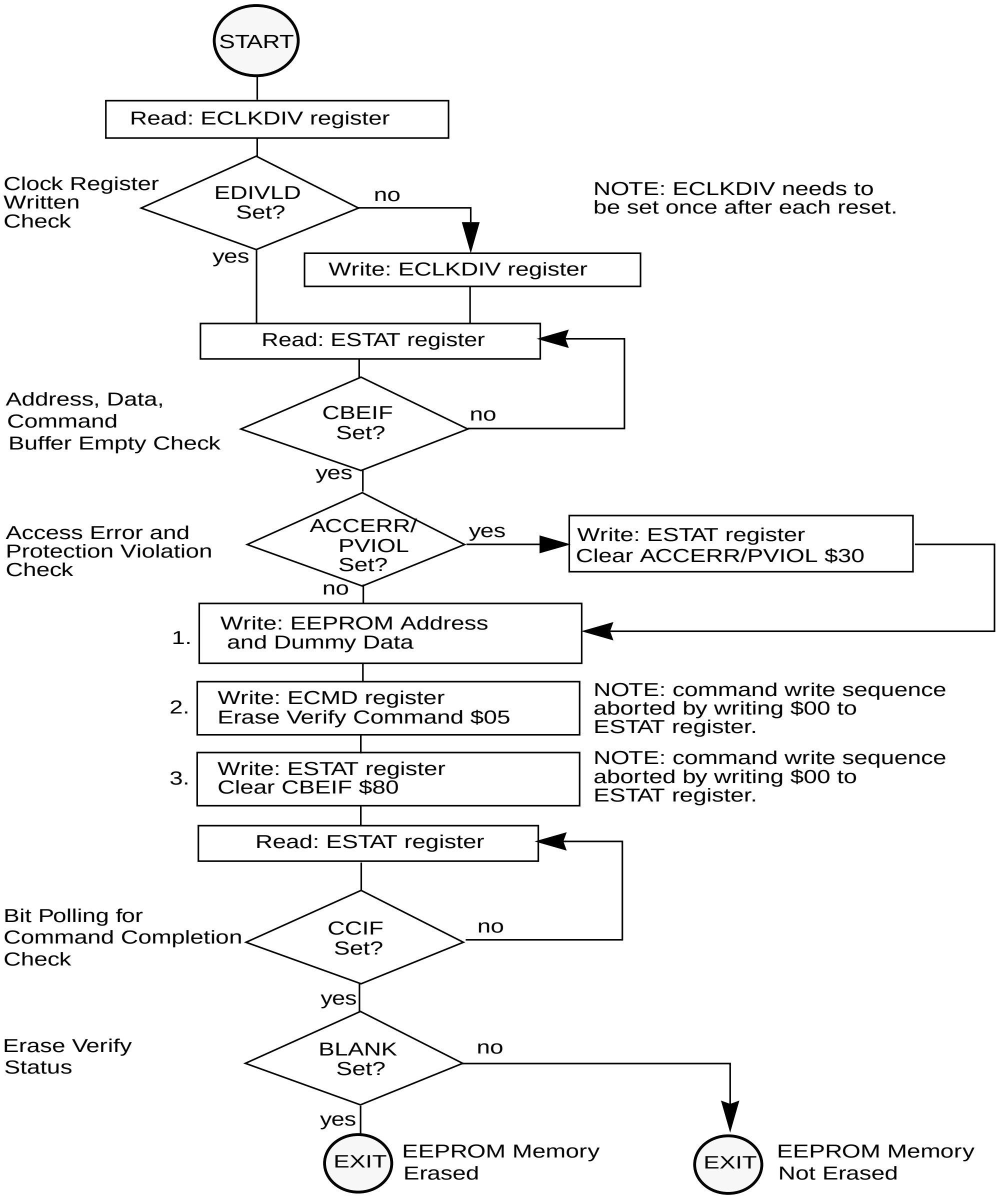
START
Read: ECLKDIV register
Clock Register Written Check
EDIVLD Set?
no
yes
NOTE: ECLKDIV needs to be set once after each reset.
Write: ECLKDIV register
Read: ESTAT register
Address, Data, Command Buffer Empty Check
CBEIF Set?
no
yes
Access Error and Protection Violation Check
ACCERR/ PVIOL Set?
yes
no
Write: ESTAT register Clear ACCERR/PVIOL $30
1.
Write: EEPROM Address and Dummy Data
2.
Write: ECMD register Erase Verify Command $05
NOTE: command write sequence aborted by writing $00 to ESTAT register.
3.
Write: ESTAT register Clear CBEIF $80
NOTE: command write sequence aborted by writing $00 to ESTAT register.
Read: ESTAT register
Bit Polling for Command Completion Check
CCIF Set?
no
yes
Erase Verify Status
BLANK Set?
no
yes
EXIT
EEPROM Memory Erased
EXIT
EEPROM Memory Not Erased

Figure 4-2 Example Erase Verify Command Flow

4.1.3.2 Program Command

The program operation will program a previously erased word in the EEPROM memory using an embedded algorithm.

An example flow to execute the program operation is shown in **Figure 4-3**. The program command write sequence is as follows:

1. Write to an EEPROM block address to start the command write sequence for the program command. The data written will be programmed to the address written.

2. Write the program command, $20, to the ECMD register.
3. Clear the CBEIF flag in the ESTAT register by writing a “1” to CBEIF to launch the program command.

If a word to be programmed is in a protected area of the EEPROM memory, the PVIOL flag in the ESTAT register will set and the program command will not launch. Once the program command has successfully launched, the CCIF flag in the ESTAT register will set after the program operation has completed unless a new command write sequence has been buffered.

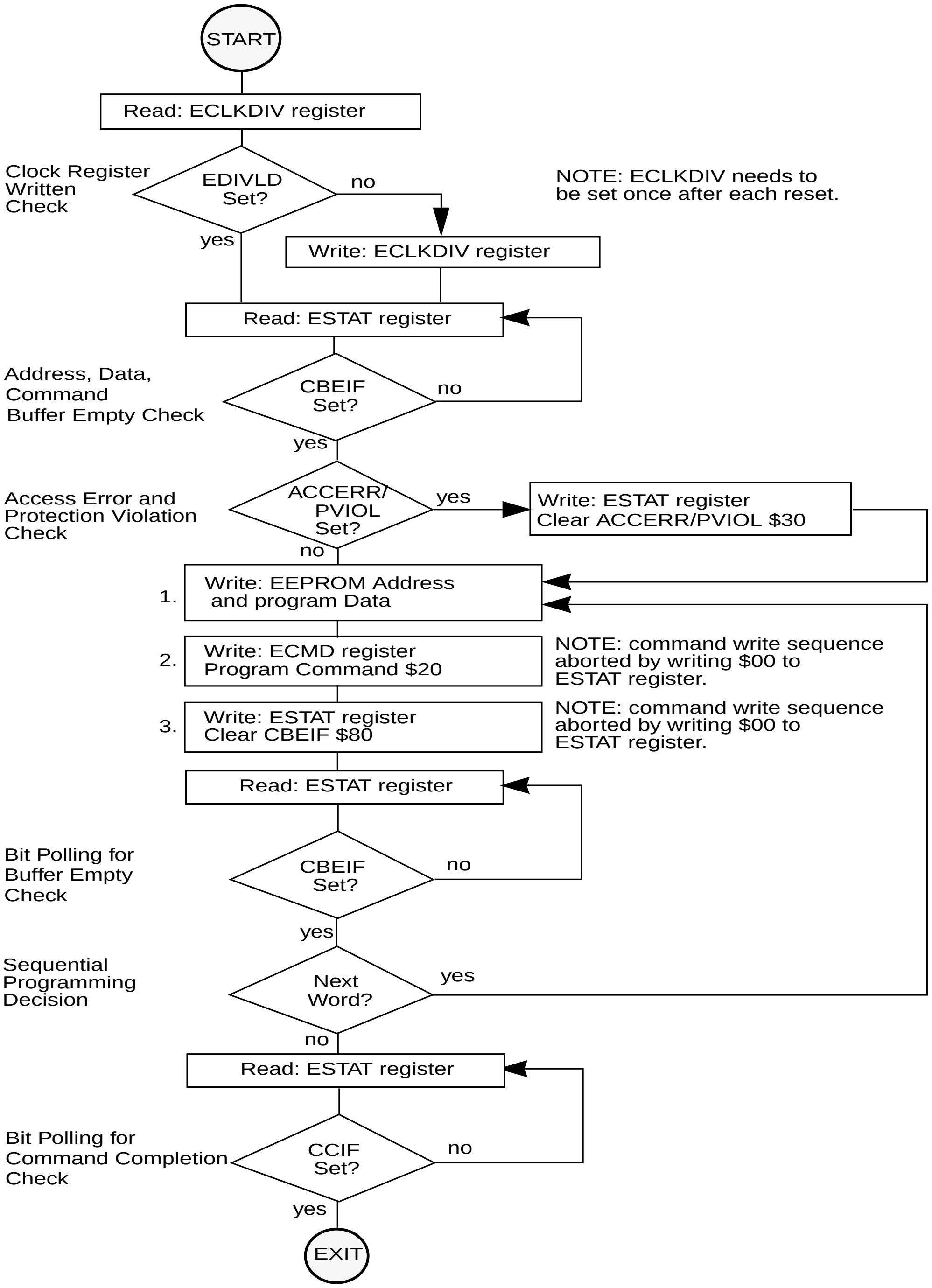
START
Read: ECLKDIV register
Clock Register Written Check
EDIVLD Set?
no
yes
NOTE: ECLKDIV needs to be set once after each reset.
Write: ECLKDIV register
Read: ESTAT register
Address, Data, Command Buffer Empty Check
CBEIF Set?
no
yes
Access Error and Protection Violation Check
ACCERR/ PVIOL Set?
yes
Write: ESTAT register Clear ACCERR/PVIOL $30
no
1.
Write: EEPROM Address and program Data
2.
Write: ECMD register Program Command $20
NOTE: command write sequence aborted by writing $00 to ESTAT register.
3.
Write: ESTAT register Clear CBEIF $80
NOTE: command write sequence aborted by writing $00 to ESTAT register.
Read: ESTAT register
Bit Polling for Buffer Empty Check
CBEIF Set?
no
yes
Sequential Programming Decision
Next Word?
yes
no
Read: ESTAT register
Bit Polling for Command Completion Check
CCIF Set?
no
yes
EXIT

Figure 4-3 Example Program Command Flow

4.1.3.3 Sector Erase Command

The sector erase operation will erase both words in a sector of EEPROM memory using an embedded algorithm.

An example flow to execute the sector erase operation is shown in **Figure 4-4**. The sector erase command write sequence is as follows:

1. Write to an EEPROM memory address to start the command write sequence for the sector erase command. The EEPROM address written determines the sector to be erased while global address bits [1:0] and the data written are ignored.
2. Write the sector erase command, $40, to the ECMD register.
3. Clear the CBEIF flag in the ESTAT register by writing a “1” to CBEIF to launch the sector erase command.

If an EEPROM sector to be erased is in a protected area of the EEPROM memory, the PVIOL flag in the ESTAT register will set and the sector erase command will not launch. Once the sector erase command has successfully launched, the CCIF flag in the ESTAT register will set after the sector erase operation has completed unless a new command write sequence has been buffered.

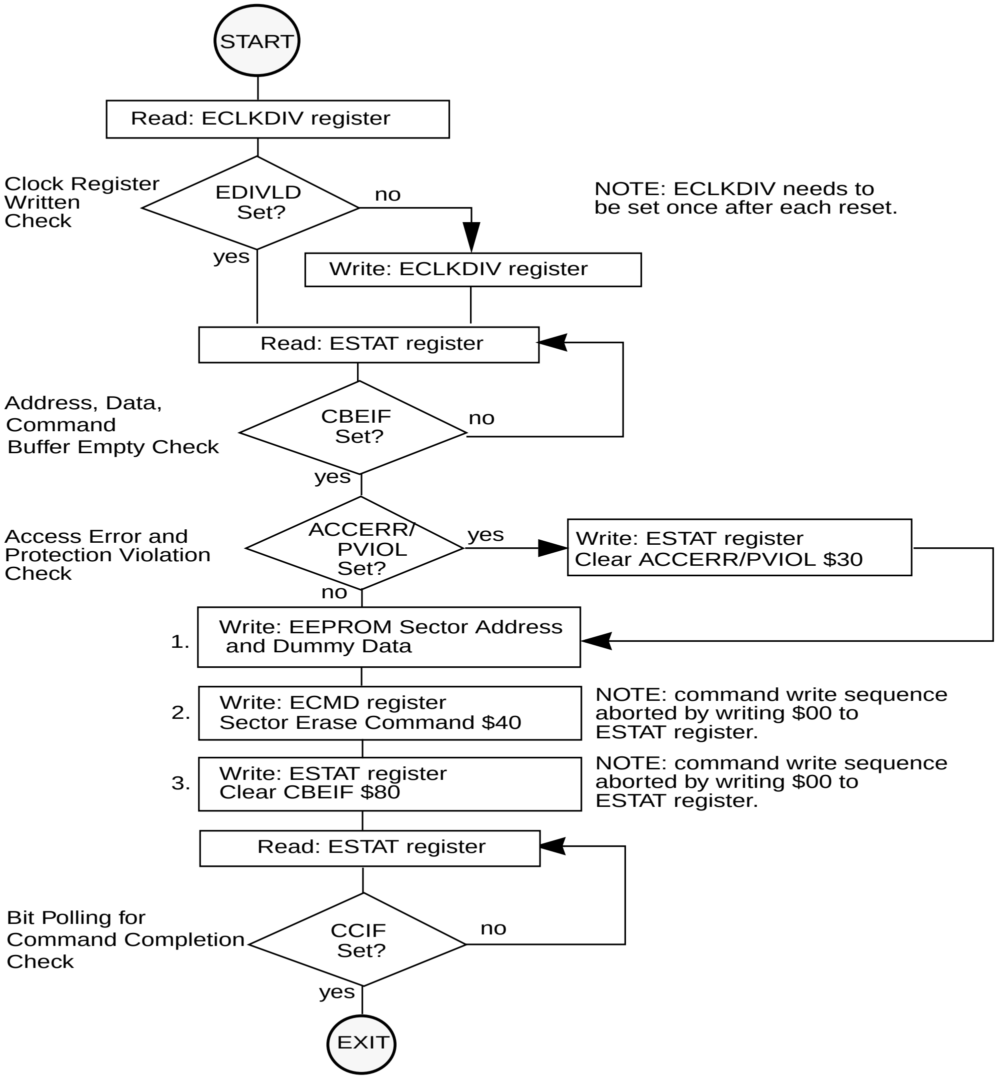
START
Read: ECLKDIV register
Clock Register Written Check
EDIVLD Set?
no
NOTE: ECLKDIV needs to be set once after each reset.
yes
Write: ECLKDIV register
Read: ESTAT register
Address, Data, Command Buffer Empty Check
CBEIF Set?
no
yes
Access Error and Protection Violation Check
ACCERR/ PVIOL Set?
yes
Write: ESTAT register Clear ACCERR/PVIOL $30
no
1.
Write: EEPROM Sector Address and Dummy Data
2.
Write: ECMD register Sector Erase Command $40
NOTE: command write sequence aborted by writing $00 to ESTAT register.
3.
Write: ESTAT register Clear CBEIF $80
NOTE: command write sequence aborted by writing $00 to ESTAT register.
Read: ESTAT register
Bit Polling for Command Completion Check
CCIF Set?
no
yes
EXIT

Figure 4-4 Example Sector Erase Command Flow

4.1.3.4 Mass Erase Command

The mass erase operation will erase all addresses in an EEPROM block using an embedded algorithm.

An example flow to execute the mass erase operation is shown in **Figure 4-5**. The mass erase command write sequence is as follows:

1. Write to an EEPROM memory address to start the command write sequence for the mass erase command. The address and data written will be ignored.
2. Write the mass erase command, $41, to the ECMD register.
3. Clear the CBEIF flag in the ESTAT register by writing a “1” to CBEIF to launch the mass erase command.

If the EEPROM memory to be erased contains any protected area, the PVIOL flag in the ESTAT register will set and the mass erase command will not launch. Once the mass erase command has successfully launched, the CCIF flag in the ESTAT register will set after the mass erase operation has completed unless a new command write sequence has been buffered.

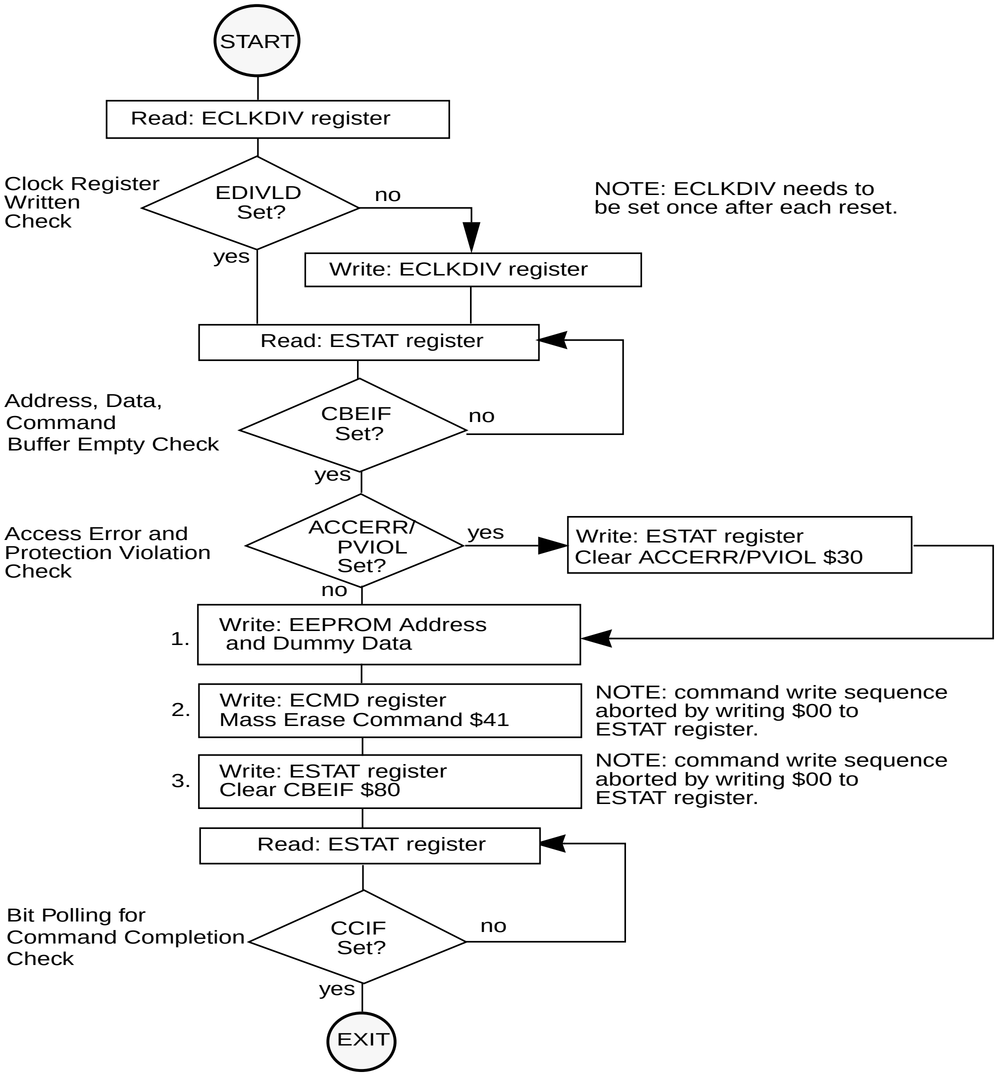
START
Read: ECLKDIV register
Clock Register Written Check
EDIVLD Set?
no
yes
NOTE: ECLKDIV needs to be set once after each reset.
Write: ECLKDIV register
Read: ESTAT register
Address, Data, Command Buffer Empty Check
CBEIF Set?
no
yes
Access Error and Protection Violation Check
ACCERR/ PVIOL Set?
yes
no
Write: ESTAT register Clear ACCERR/PVIOL $30
1.
Write: EEPROM Address and Dummy Data
2.
Write: ECMD register Mass Erase Command $41
NOTE: command write sequence aborted by writing $00 to ESTAT register.
3.
Write: ESTAT register Clear CBEIF $80
NOTE: command write sequence aborted by writing $00 to ESTAT register.
Read: ESTAT register
Bit Polling for Command Completion Check
CCIF Set?
no
yes
EXIT

Figure 4-5 Example Mass Erase Command Flow

4.1.3.5 Sector Erase Abort Command

The sector erase abort operation will terminate the active sector erase or sector modify operation so that other sectors in an EEPROM block are available for read and program operations without waiting for the sector erase or sector modify operation to complete.

An example flow to execute the sector erase abort operation is shown in **Figure 4-6**. The sector erase abort command write sequence is as follows:

1. Write to any EEPROM memory address to start the command write sequence for the sector erase abort command. The address and data written are ignored.
2. Write the sector erase abort command, $47, to the ECMD register.
3. Clear the CBEIF flag in the ESTAT register by writing a “1” to CBEIF to launch the sector erase abort command.

If the sector erase abort command is launched resulting in the early termination of an active sector erase or sector modify 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 EEPROM sector may not be fully erased and a new sector erase or sector modify 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 or sector modify operation completes normally, the ACCERR flag will not set upon completion of the operation as indicated by the CCIF flag being set. If the sector erase abort command is launched after the sector modify operation has completed the sector erase step, the program step will be allowed to complete. The maximum number of cycles required to abort a sector erase or sector modify operation is equal to four EECLK 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 ESTAT 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 ESTAT register will be set. A new command write sequence may be started after clearing the ACCERR flag, if set.*

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

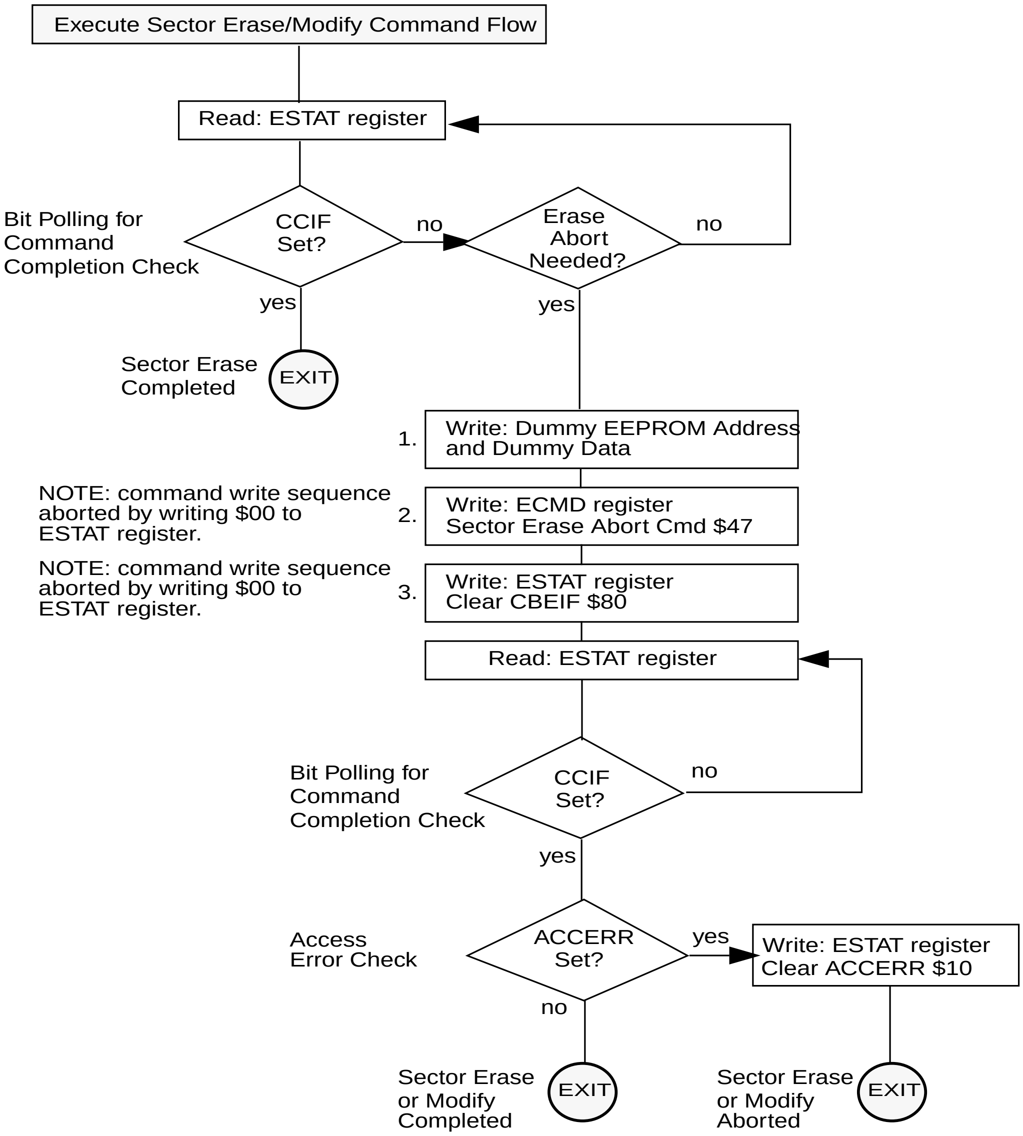
Execute Sector Erase/Modify Command Flow
Read: ESTAT register
Bit Polling for Command Completion Check
CCIF Set?
no
Erase Abort Needed?
no
yes
yes
Sector Erase Completed
EXIT
1.
Write: Dummy EEPROM Address and Dummy Data
NOTE: command write sequence aborted by writing $00 to ESTAT register.
2.
Write: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

Figure 4-6 Example Sector Erase Abort Command Flow

4.1.3.6 Sector Modify Command

The sector modify operation will erase both words in a sector of EEPROM memory followed by a reprogram of the addressed word using an embedded algorithm.

An example flow to execute the sector modify operation is shown in **Figure 4-7**. The sector modify command write sequence is as follows:

1. Write to an EEPROM memory address to start the command write sequence for the sector modify command. The EEPROM address written determines the sector to be erased and word to be reprogrammed while byte address bit 0 is ignored.

2. Write the sector modify command, $60, to the ECMD register.
3. Clear the CBEIF flag in the ESTAT register by writing a “1” to CBEIF to launch the sector erase command.

If an EEPROM sector to be modified is in a protected area of the EEPROM memory, the PVIOL flag in the ESTAT register will set and the sector modify command will not launch. Once the sector modify command has successfully launched, the CCIF flag in the ESTAT register will set after the sector modify operation has completed unless a new command write sequence has been buffered.

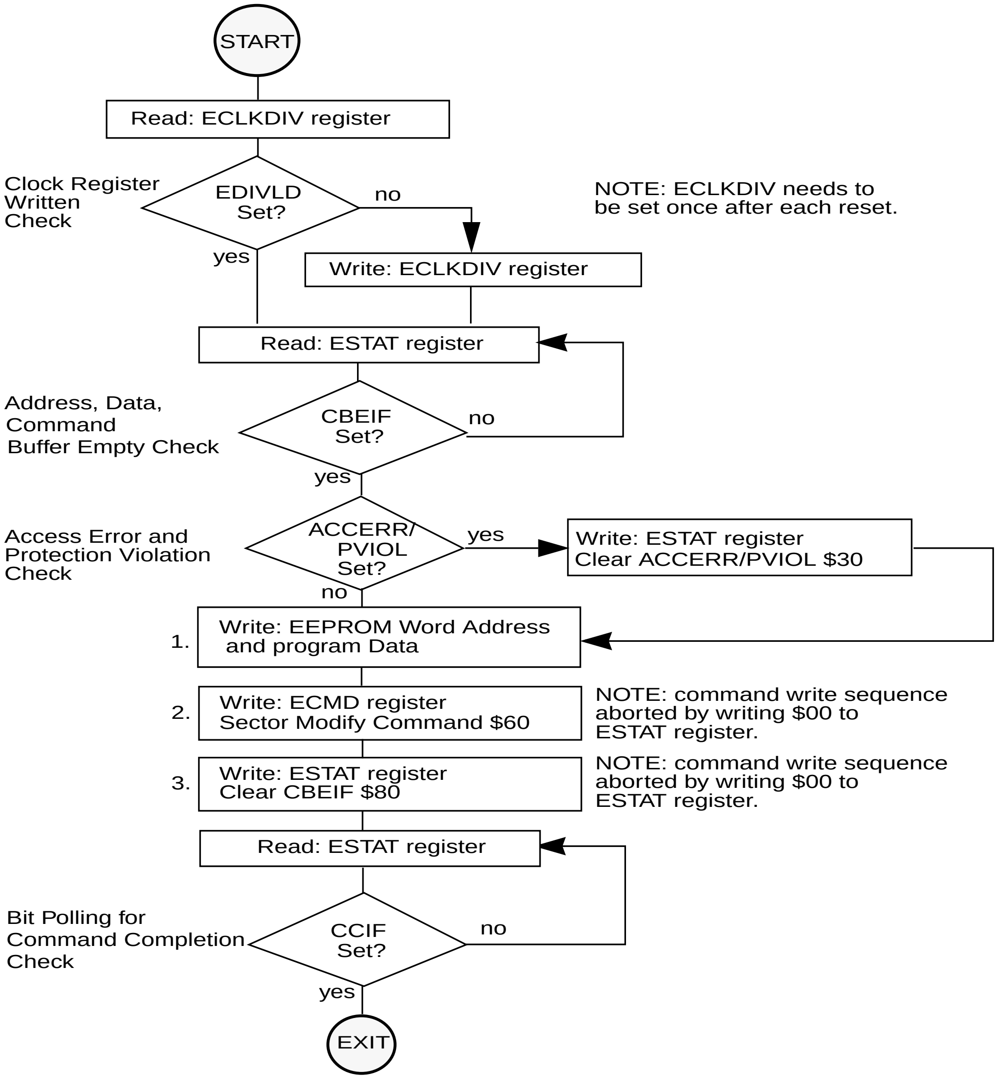
START
Read: ECLKDIV register
Clock Register Written Check
EDIVLD Set?
no
NOTE: ECLKDIV needs to be set once after each reset.
yes
Write: ECLKDIV register
Read: ESTAT register
Address, Data, Command Buffer Empty Check
CBEIF Set?
no
yes
Access Error and Protection Violation Check
ACCERR/ PVIOL Set?
yes
Write: ESTAT register Clear ACCERR/PVIOL $30
no
1.
Write: EEPROM Word Address and program Data
2.
Write: ECMD register Sector Modify Command $60
NOTE: command write sequence aborted by writing $00 to ESTAT register.
3.
Write: ESTAT register Clear CBEIF $80
NOTE: command write sequence aborted by writing $00 to ESTAT register.
Read: ESTAT register
Bit Polling for Command Completion Check
CCIF Set?
no
yes
EXIT

Figure 4-7 Example Sector Modify Command Flow

4.1.4 Illegal EEPROM 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 an EEPROM address before initializing the ECLKDIV register.
2. Writing a byte or misaligned word to a valid EEPROM address.
3. Starting a command write sequence while a sector erase abort operation is active.
4. Writing to any EEPROM register other than ECMD after writing to an EEPROM address.
5. Writing a second command to the ECMD register in the same command write sequence.
6. Writing an invalid command to the ECMD register.
7. Writing to an EEPROM address after writing to the ECMD register.
8. Writing to any EEPROM register other than ESTAT (to clear CBEIF) after writing to the ECMD register.
9. Writing a “0” to the CBEIF flag in the ESTAT register to abort a command write sequence.

The ACCERR flag will not be set if any EEPROM 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 or sector modify operation is active which results in the early termination of the sector erase or sector modify operation (see section **4.1.3.5**).
2. The MCU enters STOP mode and a command operation is in progress. The operation is aborted immediately and any pending command is purged (see section **4.3**).

If the EEPROM memory is read during execution of an algorithm (CCIF = 0), the read operation will return invalid data and the ACCERR flag will not be set.

If the ACCERR flag is set in the ESTAT 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 ECMD 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 EEPROM memory.
2. Writing the sector erase command if the address written in the command write sequence was in a protected area of the EEPROM memory.
3. Writing the mass erase command to the EEPROM memory while any EEPROM protection is enabled.
4. Writing the sector modify command if the address written in the command write sequence was in a protected area of the EEPROM memory.

If the PVIOL flag is set in the ESTAT 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 EEPROM module can recover the MCU from WAIT if the CBEIF and CCIF interrupts are enabled (see section **4.7**).

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, sector erase, mass erase, or sector modify, the EEPROM 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 EEPROM 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**).

NOTE: *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, sector erase, mass erase, or sector modify operations.*

4.4 Background Debug Mode

In background debug mode (BDM), the EPROT register is writable. If the MCU is unsecured, then all EEPROM commands listed in **Table 4-1** can be executed. If the MCU is secured and is in special single chip mode, the only command available to execute is mass erase.

4.5 EEPROM Module Security

The EEPROM module does not provide any security information to the MCU. After each reset, the security state of the MCU is a function of information provided by the Flash module (see the specific FTX Block Guide).

4.5.1 Unsecuring the MCU in Special Single Chip Mode via the BDM

Before the MCU can be unsecured in special single chip mode, the EEPROM memory must be erased using 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 EEPROM module, and execute a mass erase command write sequence to erase the EEPROM memory.

After the CCIF flag sets to indicate that the EEPROM mass operation has completed and assuming that the Flash memory has also been erased, reset the MCU into special single chip mode. The BDM secure

ROM will verify that the Flash and EEPROM memory are 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. Once the MCU is unsecured, BDM commands will be enabled and the Flash security byte may be programmed to the unsecure state.

4.6 Resets

4.6.1 EEPROM Reset Sequence

On each reset, the EEPROM module executes a reset sequence to hold CPU activity while loading the EPROT register from the EEPROM memory according to **Table 3-1**.

4.6.2 Reset While EEPROM Command Active

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

4.7 Interrupts

The EEPROM module can generate an interrupt when all EEPROM command operations have completed, when the EEPROM address, data and command buffers are empty.

Table 4-2 EEPROM Interrupt Sources

Interrupt Source	Interrupt Flag	Local Enable	Global (CCR) Mask
EEPROM Address, Data and Command Buffers empty	CBEIF (ESTAT register)	CBEIE (ECNFG register)	I-Bit
All EEPROM commands completed	CCIF (ESTAT register)	CCIE (ECNFG register)	I-Bit

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

4.7.1 Description of EEPROM Interrupt Operation

The logic used for generating interrupts is shown in **Figure 4-8**.

The EEPROM module uses the CBEIF and CCIF flags in combination with the CBIE and CCIE enable bits to generate the EEPROM command interrupt request.

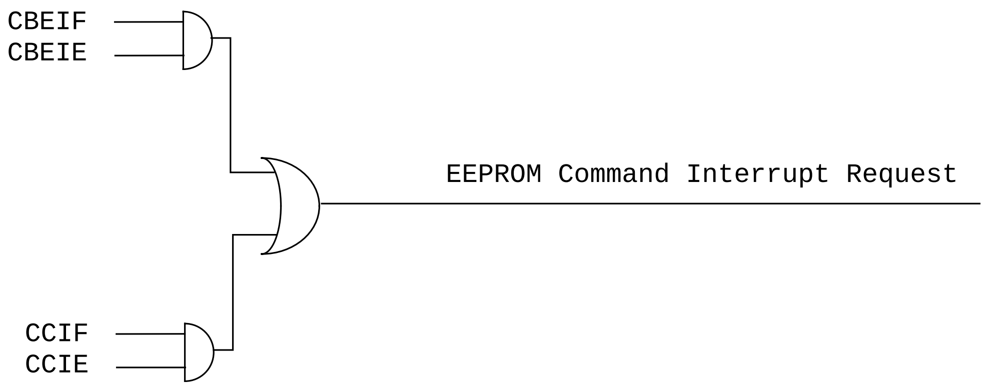

Figure 4-8 EEPROM Interrupt Implementation

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

Index

Block Guide End Sheet

FINAL PAGE OF
46
PAGES