Chapter 1
2 Kbyte EEPROM Module (S12EETS2KV1)

1.1 Introduction

This document describes the EETS2K module which is a 2 Kbyte EEPROM (nonvolatile) memory. The EETS2K block uses a small sector Flash memory to emulate EEPROM functionality. It is an array of electrically erasable and programmable, nonvolatile memory. The EEPROM memory is organized as 1024 rows of 2 bytes (1 word). The EEPROM memory's erase sector size is 2 rows or 2 words (4 bytes).

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

Program and erase functions are controlled by a command driven interface. Both sector erase and mass erase of the entire EEPROM memory are supported. An erased bit reads 1 and a programmed bit reads 0. The high voltage required to program and erase is generated internally by on-chip charge pumps.

It is not possible to read from the EEPROM memory while it is being erased or programmed.

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

CAUTION

An EEPROM word must be in the erased state 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 program, erase, or erase verify the EEPROM.

1.1.2 Features

- 2 Kbytes of EEPROM memory
- Minimum erase sector of 4 bytes
- Automated program and erase algorithms
- Interrupts on EEPROM command completion and command buffer empty
- Fast sector erase and word program operation
- 2-stage command pipeline
- Flexible protection scheme for protection against accidental program or erase
- Single power supply program and erase

1.1.3 Modes of Operation

Program and erase operation (please refer to Section 1.4.1 for details).

1.1.4 Block Diagram

Figure 1-1 shows a block diagram of the EETS2K module.

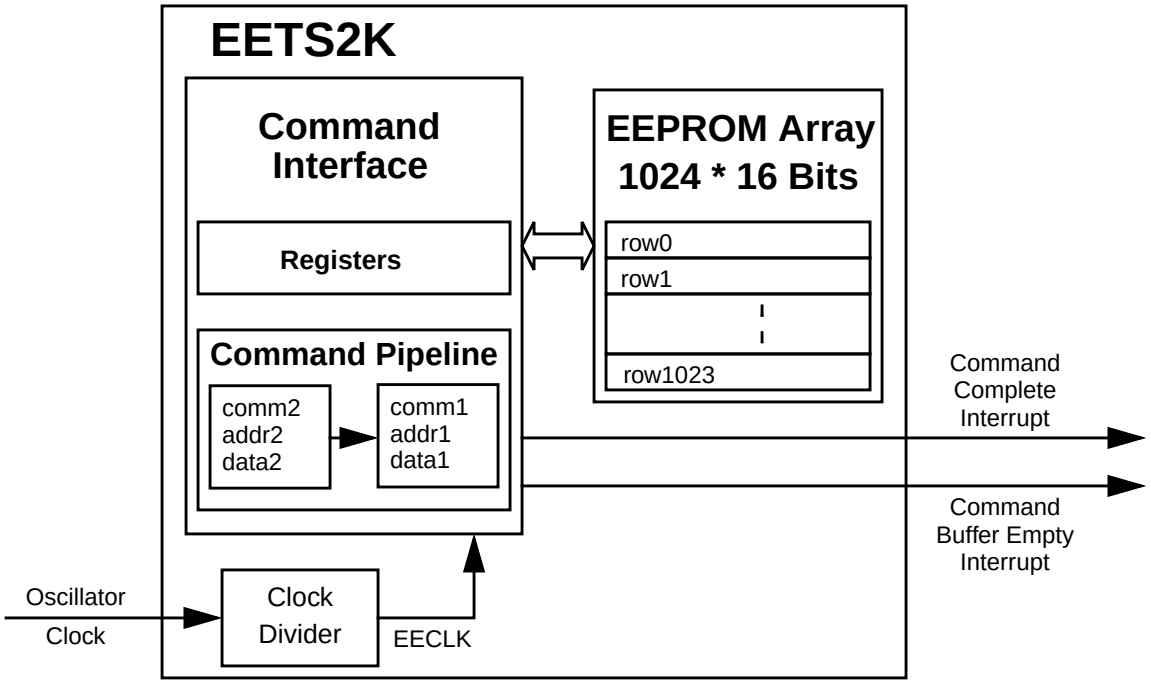

Figure 1-1. EETS2K Block Diagram

1.2 External Signal Description

The EETS2K module contains no signals that connect off chip.

1.3 Memory Map and Register Definition

This section describes the EETS2K memory map and registers.

1.3.1 Module Memory Map

Figure 1-2 shows the EETS2K memory map. Location of the EEPROM array in the MCU memory map is defined in the Device Overview chapter and is reflected in the INITEE register contents defined in the INT block description chapter. Shown within the EEPROM array are: a protection/reserved field and user-defined EEPROM protected sectors. The 16-byte protection/reserved field is located in the EEPROM array from address 0x07F0 to 0x07FF. A description of this protection/reserved field is given in Table 1-1.

Table 1-1. EEPROM Protection/Reserved Field

Address Offset	Size (Bytes)	Description
0x07F0 – 0x07FC	13	Reserved
0x07FD	1	EEPROM protection byte
0x07FE – 0x07FF	2	Reserved

The EEPROM module has hardware interlocks which protect data from accidental corruption. A protected sector is located at the higher address end of the EEPROM array, just below address 0x07FF. The protected sector in the EEPROM array can be sized from 64 bytes to 512 bytes. In addition, the EPOPEN bit in the EPROT register, described in Section 1.3.2.5, “EEPROM Protection Register (EPROT)”, can be set to globally protect the entire EEPROM array.

Chip security is defined at the MCU level.

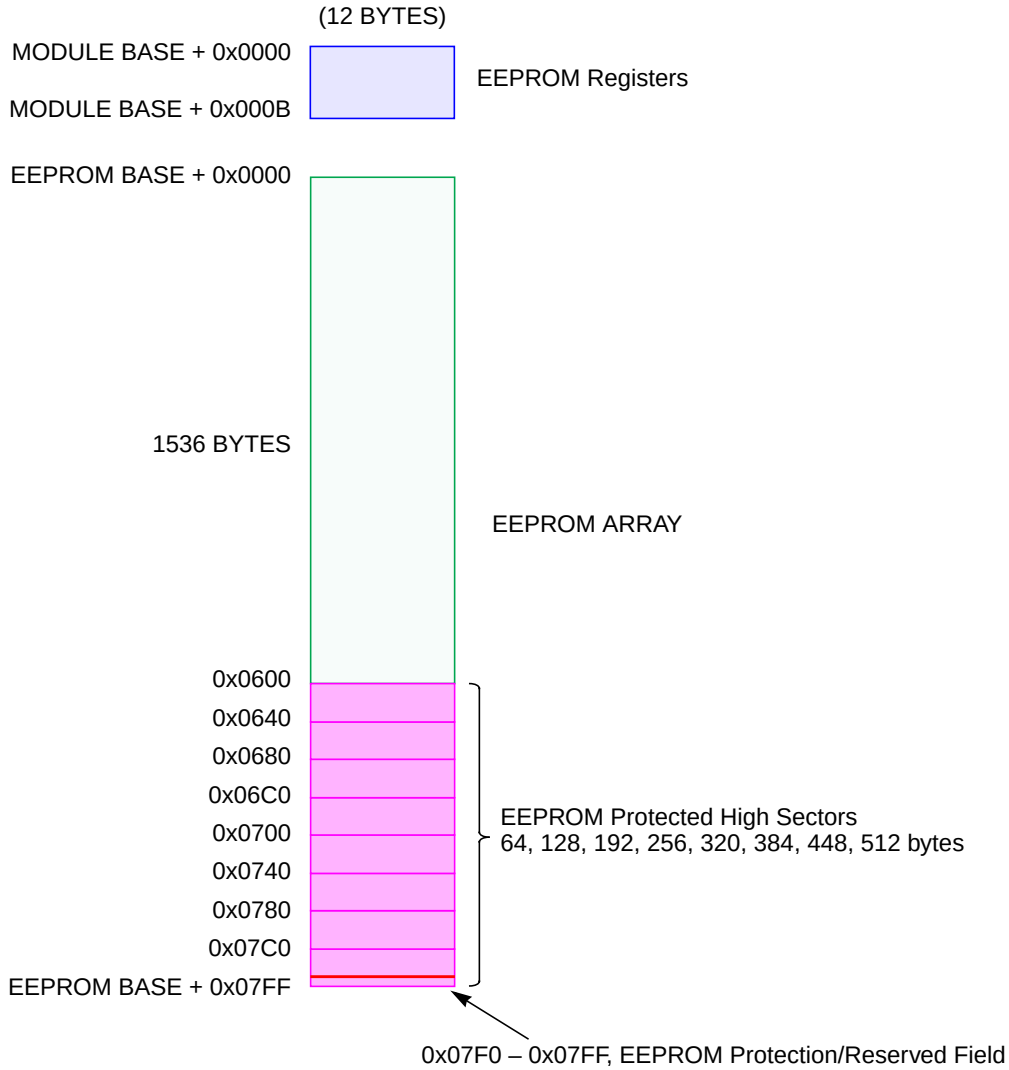

Figure 1-2. EEPROM Memory Map

The EEPROM module also contains a set of 12 control and status registers located in address space module base + 0x0000 to module base + 0x000B.

Table 1-2 gives an overview of all EETS2K registers.

Table 1-2. EEPROM Register Map

Module Base +	Register Name	Normal Mode Access
0x0000	EEPROM Clock Divider Register (ECLKDIV)	R/W
0x0001	RESERVED1[1]	R
0x0002	RESERVED2[1]	R
0x0003	EEPROM Configuration Register (ECNFG)	R/W
0x0004	EEPROM Protection Register (EPROT)	R/W
0x0005	EEPROM Status Register (ESTAT)	R/W
0x0006	EEPROM Command Register (ECMD)	R/W
0x0007	RESERVED3[1]	R
0x0008	EEPROM High Address Register (EADDRHI)	R/W
0x0009	EEPROM Low Address Register (EADDRLO)	R/W
0x000A	EEPROM High Data Register (EDATAHI)	R/W
0x000B	EEPROM Low Data Register (EDATALO)	R/W

1 Intended for factory test purposes only.

1.3.2 Register Descriptions

Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0000 ECLKDIV	R	EDIVLD	PRDIV8	EDIV5	EDIV4	EDIV3	EDIV2	EDIV1	EDIV0
	W								
0x0001 RESERVED1	R	0	0	0	0	0	0	0	0
	W								
0x0002 RESERVED2	R	0	0	0	0	0	0	0	0
	W								
0x0003 ECNFG	R	CBEIE	CCIE	0	0	0	0	0	0
	W								
0x0004 EPROT	R	EPOPEN	NV6	NV5	NV4	EPDIS	EP2	EP1	EPO
	W								
0x0005 ESTAT	R	CBEIF	CCIF	PVIOL	ACCERR	0	BLANK	0	0
	W								
0x0006 ECMD	R	0	CMDB6	CMDB5	0	0	CMDB2	0	CMDB0
	W								
0x0007 RESERVED3	R	0	0	0	0	0	0	0	0
	W								
0x0008 EADDRHI	R	0	0	0	0	0	0	EABHI	
	W								
0x0009 EADDRLO	R	EABLO							
	W								
0x000A EDATAHI	R	EDHI							
	W								
0x000B EDATALO	R	EDLO							
	W								

= Unimplemented or Reserved

Figure 1-3. EETS2K Register Summary

1.3.2.1 EEPROM Clock Divider Register (ECLKDIV)

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

Module Base + 0x0000

	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 1-4. EEPROM Clock Divider Register (ECLKDIV)

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

Table 1-3. ECLKDIV Field Descriptions

Field	Description
7 EDIVLD	**Clock Divider Loaded** 0 Register has not been written. 1 Register has been written to since the last reset.
6 PRDIV8	**Enable Prescaler by 8** 0 The oscillator clock is directly fed into the ECLKDIV divider. 1 The oscillator clock is divided by 8 before feeding into the clock divider.
5:0 EDIV[5:0]	**Clock Divider Bits** — The combination of PRDIV8 and EDIV[5:0] must divide the oscillator clock down to a frequency of 150 kHz – 200 kHz. The maximum divide ratio is 512. Please refer to Section 1.4.1.1, “Writing the ECLKDIV Register” for more information.

1.3.2.2 RESERVED1

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

Module Base + 0x0001

	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 1-5. RESERVED1

All bits read 0 and are not writable.

1.3.2.3 RESERVED2

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

Module Base + 0x0002

	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 1-6. RESERVED2

All bits read 0 and are not writable.

1.3.2.4 EEPROM Configuration Register (ECNFG)

The ECNFG register enables the EEPROM interrupts.

Module Base + 0x0003

	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 1-7. EEPROM Configuration Register (ECNFG)

CBEIE and CCIE bits are readable and writable while bits 5-0 read 0 and are not writable.

Table 1-4. ECNFG Field Descriptions

Field	Description
7 CBEIE	**Command Buffer Empty Interrupt Enable** — The CBEIE bit enables the interrupts in case of an empty command buffer in the EEPROM. 0 Command buffer empty interrupts disabled. 1 An interrupt will be requested whenever the CBEIF flag is set (see Section 1.3.2.6, “EEPROM Status Register (ESTAT)”).
6 CCIE	**Command Complete Interrupt Enable** — The CCIE bit enables the interrupts in case of all commands being completed in the EEPROM. 0 Command complete interrupts disabled. 1 An interrupt will be requested whenever the CCIF flag is set (see Section 1.3.2.6, “EEPROM Status Register (ESTAT)”).

1.3.2.5 EEPROM Protection Register (EPROT)

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

Module Base + 0x0004

	7	6	5	4	3	2	1	0
R	EPOPEN	NV6	NV5	NV4	EPDIS	EP2	EP1	EP0
W								
Reset	F	F	F	F	F	F	F	F

= Unimplemented or Reserved

Figure 1-8. EEPROM Protection Register (EPROT)

The EPROT register is loaded from EEPROM array address 0x07FD during reset, as indicated by the F in Figure 1-8.

All bits in the EPROT register are readable. Bits NV[6:4] are not writable. The EPOPEN and EPDIS bits in the EPROT register can only be written to the protected state (i.e., 0). The EP[2:0] bits can be written anytime until bit EPDIS is cleared. If the EPOPEN bit is cleared, then the state of the EPDIS and EP[2:0] bits is irrelevant.

To change the EEPROM protection that will be loaded on reset, the upper sector of EEPROM must first be unprotected, then the EEPROM protect byte located at address 0x07FD must be written to.

A protected EEPROM sector is disabled by the EPDIS bit while the size of the protected sector is defined by the EP bits in the EPROT register.

Trying to alter any of the protected areas will result in a protect violation error and PVIOL flag will be set in the ESTAT register. A mass erase of a whole EEPROM block is only possible when protection is fully disabled by setting the EPOPEN and EPDIS bits. An attempt to mass erase an EEPROM block while protection is enabled will set the PVIOL flag in the ESTAT register.

Table 1-5. EPROT Field Descriptions

Field	Description
7 EPOPEN	**Opens EEPROM for Program or Erase** 0 The whole EEPROM array is protected. In this case, the EPDIS and EP bits within the protection register are ignored. 1 The EEPROM sectors not protected are enabled for program or erase.
6:4 NV[6:4]	**Nonvolatile Flag Bits** — These three bits are available to the user as nonvolatile flags.
3 EPDIS	**EEPROM Protection Address Range Disable** — The EPDIS bit determines whether there is a protected area in the space of the EEPROM address map. 0 Protection enabled 1 Protection disabled
2:0 EP[2:0]	**EEPROM Protection Address Size** — The EP[2:0] bits determine the size of the protected sector. Refer to Table 1-6.

Table 1-6. EEPROM Address Range Protection

EP[2:0]	Protected Address Range	Protected Size
000	0x07C0-0x07FF	64 bytes
001	0x0780-0x07FF	128 bytes
010	0x0740-0x07FF	192 bytes
011	0x0700-0x07FF	256 bytes
100	0x06C0-0x07FF	320 bytes
101	0x0680-0x07FF	384 bytes
110	0x0640-0x07FF	448 bytes
111	0x0600-0x07FF	512 bytes

1.3.2.6 EEPROM Status Register (ESTAT)

The ESTAT register defines the EEPROM state machine command status and EEPROM array access, protection and erase verify status.

Module Base + 0x0005

	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 1-9. EEPROM Status Register (ESTAT - Normal Mode)

Module Base + 0x0005

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

= Unimplemented or Reserved

Figure 1-10. EEPROM Status Register (ESTAT - Special Mode)

CBEIF, PVIOL, and ACCERR bits are readable and writable, CCIF and BLANK bits are readable but not writable, remaining bits read 0 and are not writable in normal mode. FAIL is readable and writable in special mode. FAIL must be clear when starting a command write sequence. DONE is readable but not writable in special mode.

Table 1-7. ESTAT Field Descriptions

Field	Description
7 CBEIF	**Command Buffer Empty Interrupt Flag** — The CBEIF flag indicates that the address, data, and command buffers are empty so that a new command sequence can be started. The CBEIF flag is cleared by writing a 1 to CBEIF. Writing a 0 to the CBEIF flag has no effect on CBEIF. Writing a 0 to CBEIF after writing an aligned word to the EEPROM address space but before CBEIF is cleared will abort a command sequence and cause the ACCERR flag in the ESTAT register to be set. Writing a 0 to CBEIF outside of a command sequence will not set the ACCERR flag. The CBEIF flag is used together with the CBEIE bit in the ECNFG register to generate an interrupt request. 0 Buffers are full 1 Buffers are ready to accept a new command
6 CCIF	**Command Complete Interrupt Flag** — The CCIF flag indicates that there are no more commands pending. The CCIF flag is cleared when CBEIF is cleared and sets automatically upon completion of all active and pending commands. The CCIF flag does not set when an active command completes and a pending command is fetched from the command buffer. Writing to the CCIF flag has no effect. The CCIF flag is used together with the CCIE bit in the ECNFG register to generate an interrupt request. 0 Command in progress 1 All commands are completed
5 PVIOL	**Protection Violation** — The PVIOL flag indicates an attempt was made to program or erase an address in a protected EEPROM memory area (Section 1.4.1.4, “Illegal EEPROM Operations”). 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 another command in the EEPROM. 0 No failure 1 A protection violation has occurred
4 ACCERR	**EEPROM Access Error** — The ACCERR flag indicates an illegal access to the selected EEPROM array (Section 1.4.1.4, “Illegal EEPROM Operations). This can be either a violation of the command sequence, issuing an illegal command (illegal combination of the CMDBx bits in the ECMD register) or the execution of a CPU STOP instruction while a command is executing (CCIF = 0). The ACCERR flag is cleared by writing a 1 to ACCERR. Writing a 0 to the ACCERR flag has no effect on ACCERR. While ACCERR is set, it is not possible to launch another command in the EEPROM. 0 No failure 1 Access error has occurred
2 BLANK	**Array Has Been Verified as Erased** — The BLANK flag indicates that an erase verify command has checked the EEPROM array and found it to be erased. The BLANK flag is cleared by hardware when CBEIF is cleared as part of a new valid command sequence. Writing to the BLANK flag has no effect on BLANK. 0 If an erase verify command has been requested and the CCIF flag is set, then a 0 in BLANK indicates array is not erased 1 EEPROM array verifies as erased
1 FAIL	**Flag Indicating a Failed EEPROM Operation** — The FAIL flag will set if the erase verify operation fails (EEPROM block verified as not erased). The FAIL flag is cleared writing a 1 to FAIL. Writing a 0 to the FAIL flag has no effect on FAIL. 0 EEPROM operation completed without error 1 EEPROM operation failed
0 DONE	**Flag Indicating a Completed EEPROM Operation** 0 EEPROM operation is active (program, erase, erase verify) 1 EEPROM operation not active

1.3.2.7 EEPROM Command Register (ECMD)

The ECMD register defines the EEPROM commands.

Module Base + 0x0006

	7	6	5	4	3	2	1	0
R	0	CMDB6	CMDB5	0	0	CMDB2	0	CMDB0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 1-11. EEPROM Command Register (ECMD)

CMDB6, CMDB5, CMDB2, and CMDB0 bits are readable and writable during a command sequence while bits 7, 4, 3, and 1 read 0 and are not writable.

Table 1-8. ECMD Field Descriptions

Field	Description
6, 5, 2, 0 CMDB[6:5] CMDB2 CMDB0	**EEPROM Command** — Valid EEPROM commands are shown in Table 1-9. Any other command written than those mentioned in Table 1-9 sets the ACCERR bit in the ESTAT register.

Table 1-9. Valid EEPROM Command List

Command	Meaning
0x05	Erase verify
0x20	Word program
0x40	Sector erase
0x41	Mass erase
0x60	Sector modify

1.3.2.8 RESERVED3

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

Module Base + 0x0007

	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 1-12. RESERVED3

All bits read 0 and are not writable.

1.3.2.9 EEPROM Address Register (EADDR)

EADDRHI and EADDRLO are the EEPROM address registers.

Module Base + 0x0008

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

= Unimplemented or Reserved

Figure 1-13. EEPROM Address High Register (EADDRHI)

Module Base + 0x0009

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

Figure 1-14. EEPROM Address Low Register (EADDRLO)

In normal modes, all EADDRHI and EADDRLO bits read 0 and are not writable.

In special modes, all EADDRHI and EADDRLO bits are readable and writable except EADDRHI[7:2] which are not writable and always read 0.

For sector erase, the MCU address bits AB[1:0] are ignored.

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

1.3.2.10 EEPROM Data Register (EDATA)

EDATAHI and EDATALO are the EEPROM data registers.

Module Base + 0x000A

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

Figure 1-15. EEPROM Data High Register (EDATAHI)

Module Base + 0x000B

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

Figure 1-16. EEPROM Data Low Register (EDATALO)

In normal modes, all EDATAHI and EDATALO bits read 0 and are not writable.

In special modes, all EDATAHI and EDATALO bits are readable and writable.

1.4 Functional Description

1.4.1 Program and Erase Operation

Write and read operations are both used for the program and erase algorithms described in this subsection. These 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 new command along with the necessary data and address can be stored to the buffer while the previous command is remains in progress. The pipelined operation allows a simplification of command launching. Buffer empty as well as command completion are signalled by flags in the EEPROM status register. Interrupts for the EEPROM will be generated if enabled.

The next four subsections describe:

- How to write the ECLKDIV register.
- Command write sequences used to program, erase, and verify the EEPROM memory.
- Valid EEPROM commands.
- Errors resulting from illegal EEPROM operations.

1.4.1.1 Writing the ECLKDIV Register

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

- EECLK 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 1-17.

For example, if the oscillator clock is 950 kHz and the bus clock is 10 MHz, ECLKDIV bits EDIV[5:0] must be set to 4 (binary 000100) and bit PRDIV8 set to 0. The resulting EECLK is then 190 kHz. As a result, the EEPROM algorithm timings are increased over optimum target by:

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

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

CAUTION

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

If the ECLKDIV register is written, the bit EDIVLD is set automatically. If this bit is 0, the register has not been written since the last reset. EEPROM commands will not be executed if this register has not been written to.

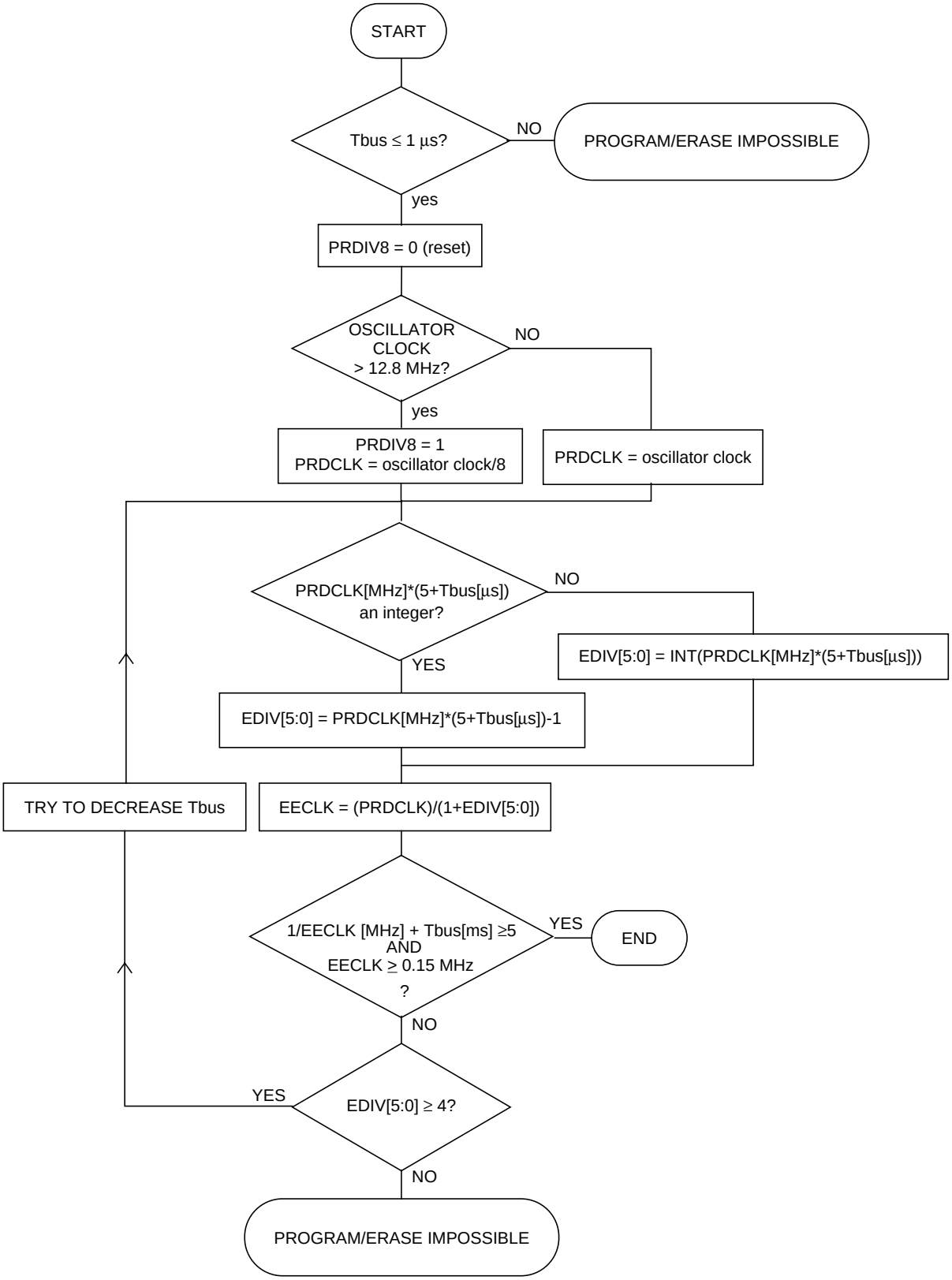

Figure 1-17. PRDIV8 and EDIV Bits Determination Procedure

1.4.1.2 Command Write Sequence

The EEPROM command controller is used to supervise the command write sequence to execute program, erase, mass erase, sector modify, and erase verify operations. Before starting a command write sequence, it is necessary to check that there is no pending access error or protection violation (the ACCERR and PVIOL flags must be cleared in the ESTAT register).

After this initial step, the CBEIF flag must be tested to ensure that the address, data and command buffers are empty. If so, the command sequence can be started. The following 3-step command write sequence must be strictly adhered to and no intermediate access to the EEPROM array is permitted between the 3 steps. It is possible to read any EEPROM register during a command sequence. The command write sequence is as follows:

1. Write an aligned word to be to a valid EEPROM array address. The address and data will be stored in internal buffers.
   - For program and sector modify, all address and data bits are valid.
   - For erase, the value of the data bytes are ignored.
   - For mass erase and erase verify, the address can be anywhere in the available address space of the array.
   - For sector erase, the address bits[1:0] are ignored.
2. Write a valid command, listed in Table 1-10, to the ECMD register.
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 by hardware indicating that the command was successfully launched. The CBEIF flag will be set again indicating the address, data, and command buffers are ready for a new command write sequence to begin.

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

The EEPROM command controller will flag errors in command write sequences by means of the ACCERR (access error) and PVIOL (protection violation) flags in the ESTAT register. An erroneous command write sequence will abort and set the appropriate flag. If set, the user must clear the ACCERR or PVIOL flags before commencing another command write sequence. By writing a 0 to the CBEIF flag the command sequence can be aborted after the word write to the EEPROM address space or after writing a command to the ECMD register and before the command is launched. Writing a 0 to the CBEIF flag in this way will set the ACCERR flag.

A summary of the launching of a program operation is shown in Figure 1-18. For other operations, the user writes the appropriate command to the ECMD register.

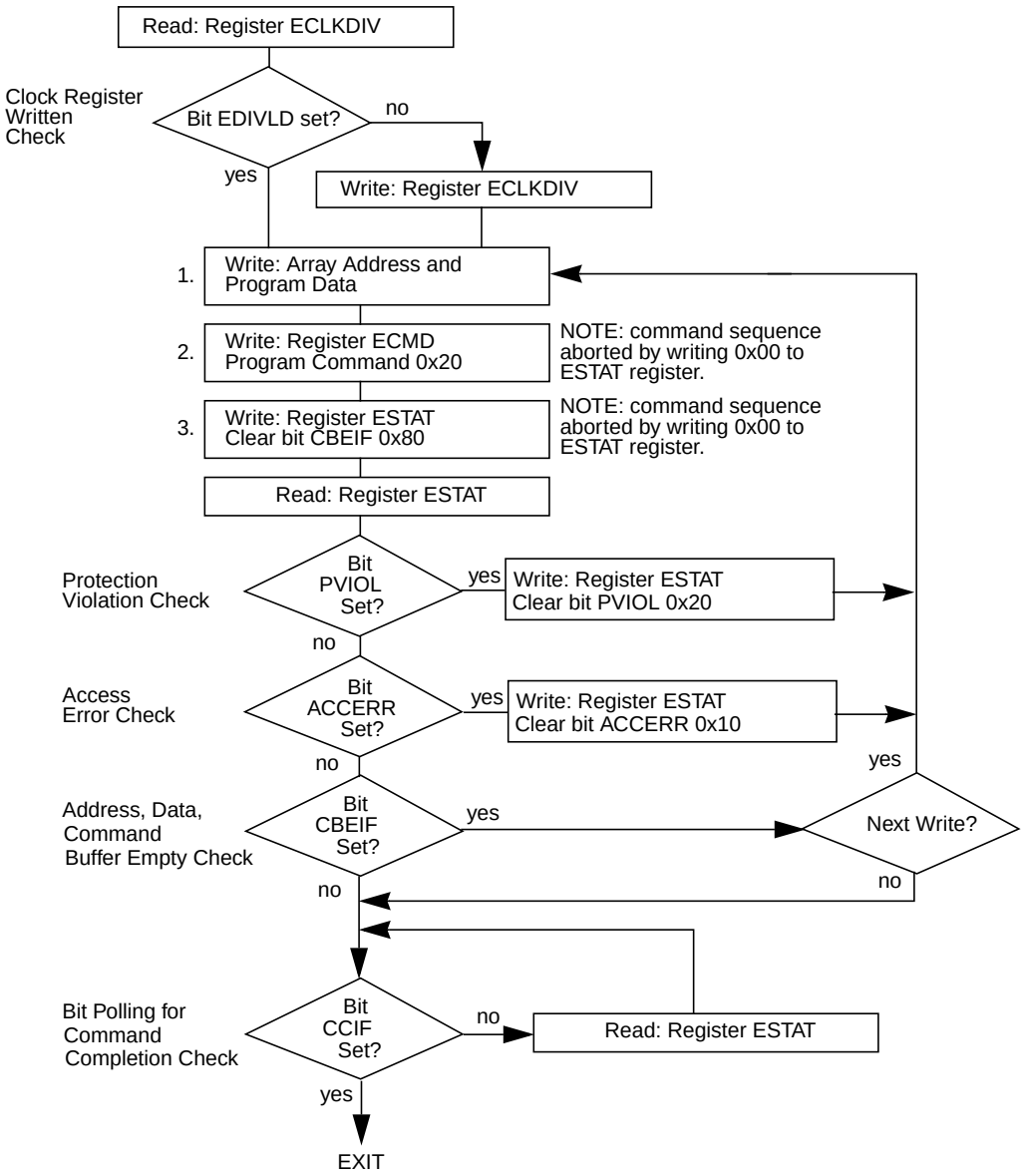

Figure 1-18. Example Program Command Flow

1.4.1.3 Valid EEPROM Commands

Table 1-10 summarizes the valid EEPROM commands. Also shown are the effects of the commands on the EEPROM array.

Table 1-10. Valid EEPROM Commands

ECMD	Meaning	Function on EEPROM Array
0x05	Erase Verify	Verify all memory bytes of the EEPROM array are erased. If the array is erased, the BLANK bit will set in the ESTAT register upon command completion.
0x20	Program	Program a word (two bytes).
0x40	Sector Erase	Erase two words (four bytes) of EEPROM array.
0x41	Mass Erase	Erase all of the EEPROM array. A mass erase of the full array is only possible when EPDIS and EPOPEN are set.
0x60	Sector Modify	Erase two words of EEPROM, re-program one word.

CAUTION

An EEPROM word must be in an erased state before being programmed. Cumulative programming of bits within a word is not allowed.

The sector modify command (0x60) is a two-step command which first erases a sector (2 words) of the EEPROM array and then re-programs one of the words in that sector. The EEPROM sector which is erased by the sector modify command is the sector containing the address of the aligned array write which starts the valid command sequence. That same address is re-programmed with the data which is written. By launching a sector modify command and then pipelining a program command it is possible to completely replace the contents of an EEPROM sector.

1.4.1.4 Illegal EEPROM Operations

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

1. Writing to the EEPROM address space before initializing ECLKDIV.
2. Writing a misaligned word or a byte to the valid EEPROM address space.
3. Writing to the EEPROM address space while CBEIF is not set.
4. Writing a second word to the EEPROM address space before executing a program or erase command on the previously written word.
5. Writing to any EEPROM register other than ECMD after writing a word to the EEPROM address space.
6. Writing a second command to the ECMD register before executing the previously written command.
7. Writing an invalid command to the ECMD register in normal mode.
8. Writing to any EEPROM register other than ESTAT (to clear CBEIF) after writing to the command register (ECMD).

9. The part enters stop mode and a program or erase command is in progress. The command is aborted and any pending command is killed.
10. A 0 is written to the CBEIF bit in the ESTAT register.

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

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

When an ACCERR flag is set in the ESTAT register, the command state machine is locked. It is not possible to launch another command until the ACCERR flag is cleared.

The PVIOL flag will be set during the command write sequence after the word write to the EEPROM address space and the command sequence will be aborted if any of the following illegal operations are performed.

1. Writing a EEPROM address to program in a protected area of the EEPROM.
2. Writing a EEPROM address to erase in a protected area of the EEPROM.
3. Writing the mass erase command to ECMD while any protection is enabled.

When the PVIOL flag is set in the ESTAT register the command state machine is locked. It is not possible to launch another command until the PVIOL flag is cleared.

1.5 Operating Modes

1.5.1 Wait Mode

If an EEPROM command is active (CCIF = 0) when the MCU enters wait mode, that command and any pending command will be completed.

The EETS2K module can recover the MCU from wait mode if the interrupts are enabled (see Section 1.7, “Interrupts”).

1.5.2 Stop Mode

If a command is active (CCIF = 0) when the MCU enters stop mode, the operation will be aborted and if the operation is program, erase, or sector modify, the 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 array will be switched off when entering stop mode. Upon exit from stop mode, the CBEIF flag is set and any pending command will not be launched. The ACCERR flag must be cleared before starting a new command write sequence.

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 instruction during program, erase, or sector modify operations.

1.5.3 Background Debug Mode

In background debug mode (BDM), the EPROT register is writable. If the chip is unsecured then all EEPROM commands listed in Table 1-10 can be executed. If the chip is secured in special single-chip mode, then the only possible command to execute is mass erase.

1.6 Resets

If a reset occurs while any EEPROM 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.

1.7 Interrupts

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

Table 1-11. EEPROM Interrupt Sources

Interrupt Source	Interrupt Flag	Local Enable	Global (CCR) Mask
EEPROM address, data and command buffers empty	CBEIF (ESTAT register)	CBEIE	I Bit
All commands are completed on EEPROM	CCIF (ESTAT register)	CCIE	I Bit

NOTE

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

For a detailed description of the register bits, refer to Section 1.3.2.4, “EEPROM Configuration Register (ECNFG)” and Section 1.3.2.6, “EEPROM Status Register (ESTAT)”.