

Welcome to [E-XFL.COM](#)

### What is "[Embedded - Microcontrollers](#)"?

"[Embedded - Microcontrollers](#)" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

### Applications of "[Embedded - Microcontrollers](#)"

#### Details

Product Status	Active
Core Processor	HCS12
Core Size	16-Bit
Speed	25MHz
Connectivity	EBI/EMI, SCI, SPI
Peripherals	POR, PWM, WDT
Number of I/O	31
Program Memory Size	96KB (96K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	4K x 8
Voltage - Supply (Vcc/Vdd)	2.35V ~ 5.5V
Data Converters	A/D 8x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	48-LQFP
Supplier Device Package	48-LQFP (7x7)
Purchase URL	<a href="https://www.e-xfl.com/pro/item?MUrl=&amp;PartUrl=mc9s12gc96cfae">https://www.e-xfl.com/pro/item?MUrl=&amp;PartUrl=mc9s12gc96cfae</a>

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x0048	TCTL1	Read: Write:	OM7	OL7	OM6	OL6	OM5	OL5	OM4	OL4
0x0049	TCTL2	Read: Write:	OM3	OL3	OM2	OL2	OM1	OL1	OM0	OL0
0x004A	TCTL3	Read: Write:	EDG7B	EDG7A	EDG6B	EDG6A	EDG5B	EDG5A	EDG4B	EDG4A
0x004B	TCTL4	Read: Write:	EDG3B	EDG3A	EDG2B	EDG2A	EDG1B	EDG1A	EDG0B	EDG0A
0x004C	TIE	Read: Write:	C7I	C6I	C5I	C4I	C3I	C2I	C1I	C0I
0x004D	TSCR2	Read: Write:	TOI	0	0	0	TCRE	PR2	PR1	PR0
0x004E	TFLG1	Read: Write:	C7F	C6F	C5F	C4F	C3F	C2F	C1F	C0F
0x004F	TFLG2	Read: Write:	TOF	0	0	0	0	0	0	0
0x0050	TC0 (hi)	Read: Write:	Bit 15	14	13	12	11	10	9	Bit 8
0x0051	TC0 (lo)	Read: Write:	Bit 7	6	5	4	3	2	1	Bit 0
0x0052	TC1 (hi)	Read: Write:	Bit 15	14	13	12	11	10	9	Bit 8
0x0053	TC1 (lo)	Read: Write:	Bit 7	6	5	4	3	2	1	Bit 0
0x0054	TC2 (hi)	Read: Write:	Bit 15	14	13	12	11	10	9	Bit 8
0x0055	TC2 (lo)	Read: Write:	Bit 7	6	5	4	3	2	1	Bit 0
0x0056	TC3 (hi)	Read: Write:	Bit 15	14	13	12	11	10	9	Bit 8
0x0057	TC3 (lo)	Read: Write:	Bit 7	6	5	4	3	2	1	Bit 0
0x0058	TC4 (hi)	Read: Write:	Bit 15	14	13	12	11	10	9	Bit 8
0x0059	TC4 (lo)	Read: Write:	Bit 7	6	5	4	3	2	1	Bit 0
0x005A	TC5 (hi)	Read: Write:	Bit 15	14	13	12	11	10	9	Bit 8
0x005B	TC5 (lo)	Read: Write:	Bit 7	6	5	4	3	2	1	Bit 0
0x005C	TC6 (hi)	Read: Write:	Bit 15	14	13	12	11	10	9	Bit 8
0x005D	TC6 (lo)	Read: Write:	Bit 7	6	5	4	3	2	1	Bit 0
0x005E	TC7 (hi)	Read: Write:	Bit 15	14	13	12	11	10	9	Bit 8

## Chapter 4

# Multiplexed External Bus Interface (MEBIV3)

### 4.1 Introduction

This section describes the functionality of the multiplexed external bus interface (MEBI) sub-block of the S12 core platform. The functionality of the module is closely coupled with the S12 CPU and the memory map controller (MMC) sub-blocks.

Figure 4-1 is a block diagram of the MEBI. In Figure 4-1, the signals on the right hand side represent pins that are accessible externally. On some chips, these may not all be bonded out.

The MEBI sub-block of the core serves to provide access and/or visibility to internal core data manipulation operations including timing reference information at the external boundary of the core and/or system. Depending upon the system operating mode and the state of bits within the control registers of the MEBI, the internal 16-bit read and write data operations will be represented in 8-bit or 16-bit accesses externally. Using control information from other blocks within the system, the MEBI will determine the appropriate type of data access to be generated.

#### 4.1.1 Features

The block name includes these distinctive features:

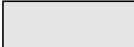
- External bus controller with four 8-bit ports A, B, E, and K
- Data and data direction registers for ports A, B, E, and K when used as general-purpose I/O
- Control register to enable/disable alternate functions on ports E and K
- Mode control register
- Control register to enable/disable pull resistors on ports A, B, E, and K
- Control register to enable/disable reduced output drive on ports A, B, E, and K
- Control register to configure external clock behavior
- Control register to configure  $\overline{\text{IRQ}}$  pin operation
- Logic to capture and synchronize external interrupt pin inputs

### 4.3.2.8 Port E Assignment Register (PEAR)

Module Base + 0x000A

Starting address location affected by INITRG register setting.

	7	6	5	4	3	2	1	0
R	NOACCE	0	PIPOE	NECLK	LSTRE	RDWE	0	0
W								
Reset								
Special Single Chip	0	0	0	0	0	0	0	0
Special Test	0	0	1	0	1	1	0	0
Peripheral	0	0	0	0	0	0	0	0
Emulation Expanded Narrow	1	0	1	0	1	1	0	0
Emulation Expanded Wide	1	0	1	0	1	1	0	0
Normal Single Chip	0	0	0	1	0	0	0	0
Normal Expanded Narrow	0	0	0	0	0	0	0	0
Normal Expanded Wide	0	0	0	0	0	0	0	0

 = Unimplemented or Reserved

**Figure 4-12. Port E Assignment Register (PEAR)**

Read: Anytime (provided this register is in the map).

Write: Each bit has specific write conditions. Please refer to the descriptions of each bit on the following pages.

Port E serves as general-purpose I/O or as system and bus control signals. The PEAR register is used to choose between the general-purpose I/O function and the alternate control functions. When an alternate control function is selected, the associated DDRE bits are overridden.

The reset condition of this register depends on the mode of operation because bus control signals are needed immediately after reset in some modes. In normal single-chip mode, no external bus control signals are needed so all of port E is configured for general-purpose I/O. In normal expanded modes, only the E clock is configured for its alternate bus control function and the other bits of port E are configured for general-purpose I/O. As the reset vector is located in external memory, the E clock is required for this access.  $R/\overline{W}$  is only needed by the system when there are external writable resources. If the normal expanded system needs any other bus control signals, PEAR would need to be written before any access that needed the additional signals. In special test and emulation modes, IPIPE1, IPIPE0, E,  $\overline{LSTRB}$ , and  $R/\overline{W}$  are configured out of reset as bus control signals.

This register is not in the on-chip memory map in expanded and special peripheral modes. Therefore, these accesses will be echoed externally.

Table 8-3. ATDCTL3 Field Descriptions (continued)

Field	Description
2 FIFO	<p><b>Result Register FIFO Mode</b> — If this bit is zero (non-FIFO mode), the A/D conversion results map into the result registers based on the conversion sequence; the result of the first conversion appears in the first result register, the second result in the second result register, and so on.</p> <p>If this bit is one (FIFO mode) the conversion counter is not reset at the beginning or ending of a conversion sequence; sequential conversion results are placed in consecutive result registers. In a continuously scanning conversion sequence, the result register counter will wrap around when it reaches the end of the result register file. The conversion counter value (CC2-0 in ATDSTAT0) can be used to determine where in the result register file, the current conversion result will be placed.</p> <p>Aborting a conversion or starting a new conversion by write to an ATDCTL register (ATDCTL5-0) clears the conversion counter even if FIFO=1. So the first result of a new conversion sequence, started by writing to ATDCTL5, will always be place in the first result register (ATDDDR0). Intended usage of FIFO mode is continuous conversion (SCAN=1) or triggered conversion (ETRIG=1).</p> <p>Which result registers hold valid data can be tracked using the conversion complete flags. Fast flag clear mode may or may not be useful in a particular application to track valid data.</p> <p>0 Conversion results are placed in the corresponding result register up to the selected sequence length.</p> <p>1 Conversion results are placed in consecutive result registers (wrap around at end).</p>
1–0 FRIZ[1:0]	<p><b>Background Debug Freeze Enable</b> — When debugging an application, it is useful in many cases to have the ATD pause when a breakpoint (Freeze Mode) is encountered. These 2 bits determine how the ATD will respond to a breakpoint as shown in Table 8-5. Leakage onto the storage node and comparator reference capacitors may compromise the accuracy of an immediately frozen conversion depending on the length of the freeze period.</p>

Table 8-4. Conversion Sequence Length Coding

S8C	S4C	S2C	S1C	Number of Conversions per Sequence
0	0	0	0	8
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
0	1	1	1	7
1	X	X	X	8

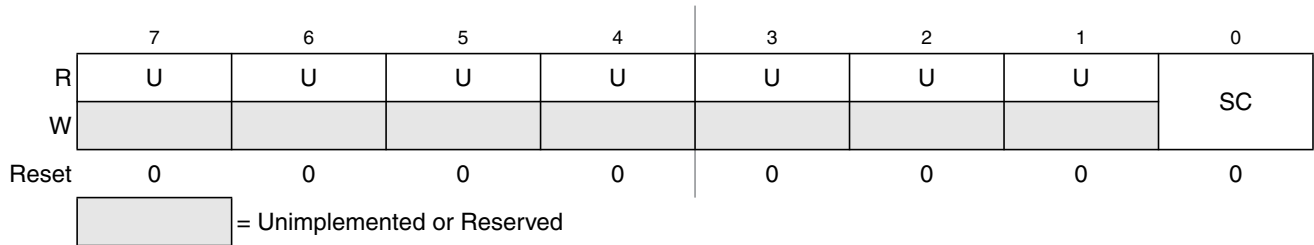
Table 8-5. ATD Behavior in Freeze Mode (Breakpoint)

FRZ1	FRZ0	Behavior in Freeze Mode
0	0	Continue conversion
0	1	Reserved
1	0	Finish current conversion, then freeze
1	1	Freeze Immediately

### 8.3.2.9 ATD Test Register 1 (ATDTEST1)

This register contains the SC bit used to enable special channel conversions.

Module Base + 0x0009



**Figure 8-11. ATD Test Register 1 (ATDTEST1)**

Read: Anytime, returns unpredictable values for Bit 7 and Bit 6

Write: Anytime

**Table 8-14. ATDTEST1 Field Descriptions**

Field	Description
0 SC	<b>Special Channel Conversion Bit</b> — If this bit is set, then special channel conversion can be selected using CC, CB, and CA of ATDCTL5. <a href="#">Table 8-15</a> lists the coding. 0 Special channel conversions disabled 1 Special channel conversions enabled <b>Note:</b> Always write remaining bits of ATDTEST1 (Bit7 to Bit1) zero when writing SC bit. Not doing so might result in unpredictable ATD behavior.

**Table 8-15. Special Channel Select Coding**

SC	CC	CB	CA	Analog Input Channel
1	0	X	X	Reserved
1	1	0	0	$V_{RH}$
1	1	0	1	$V_{RL}$
1	1	1	0	$(V_{RH}+V_{RL}) / 2$
1	1	1	1	Reserved

## 8.4 Functional Description

The ATD10B8C is structured in an analog and a digital sub-block.

### 8.4.1 Analog Sub-block

The analog sub-block contains all analog electronics required to perform a single conversion. Separate power supplies  $V_{DDA}$  and  $V_{SSA}$  allow to isolate noise of other MCU circuitry from the analog sub-block.

#### 8.4.1.1 Sample and Hold Machine

The sample and hold (S/H) machine accepts analog signals from the external surroundings and stores them as capacitor charge on a storage node.

The sample process uses a two stage approach. During the first stage, the sample amplifier is used to quickly charge the storage node. The second stage connects the input directly to the storage node to complete the sample for high accuracy.

When not sampling, the sample and hold machine disables its own clocks. The analog electronics still draw their quiescent current. The power down (ADPU) bit must be set to disable both the digital clocks and the analog power consumption.

The input analog signals are unipolar and must fall within the potential range of  $V_{SSA}$  to  $V_{DDA}$ .

#### 8.4.1.2 Analog Input Multiplexer

The analog input multiplexer connects one of the 8 external analog input channels to the sample and hold machine.

#### 8.4.1.3 Sample Buffer Amplifier

The sample amplifier is used to buffer the input analog signal so that the storage node can be quickly charged to the sample potential.

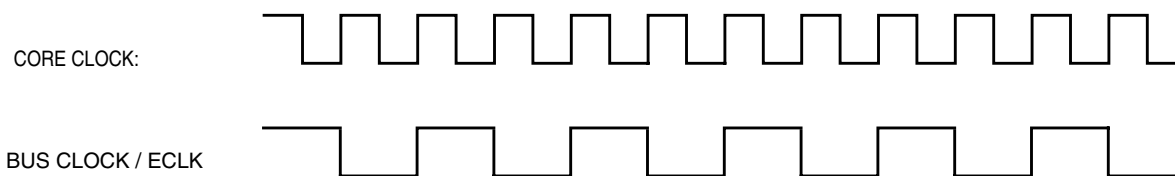
#### 8.4.1.4 Analog-to-Digital (A/D) Machine

The A/D machine performs analog-to-digital conversions. The resolution is program selectable at either 8 or 10 bits. The A/D machine uses a successive approximation architecture. It functions by comparing the stored analog sample potential with a series of digitally generated analog potentials. By following a binary search algorithm, the A/D machine locates the approximating potential that is nearest to the sampled potential.

When not converting the A/D machine disables its own clocks. The analog electronics still draws quiescent current. The power down (ADPU) bit must be set to disable both the digital clocks and the analog power consumption.

Only analog input signals within the potential range of  $V_{RL}$  to  $V_{RH}$  (A/D reference potentials) will result in a non-railed digital output codes.

of 4 OSCCLK plus 4 PLLCLK cycles to make the transition. During the transition, all clocks freeze and CPU activity ceases.



**Figure 9-18. Core Clock and Bus Clock Relationship**

## 9.4.3 Clock Monitor (CM)

If no OSCCLK edges are detected within a certain time, the clock monitor within the oscillator block generates a clock monitor fail event. The CRGV4 then asserts self-clock mode or generates a system reset depending on the state of SCME bit. If the clock monitor is disabled or the presence of clocks is detected no failure is indicated by the oscillator block. The clock monitor function is enabled/disabled by the CME control bit.

## 9.4.4 Clock Quality Checker

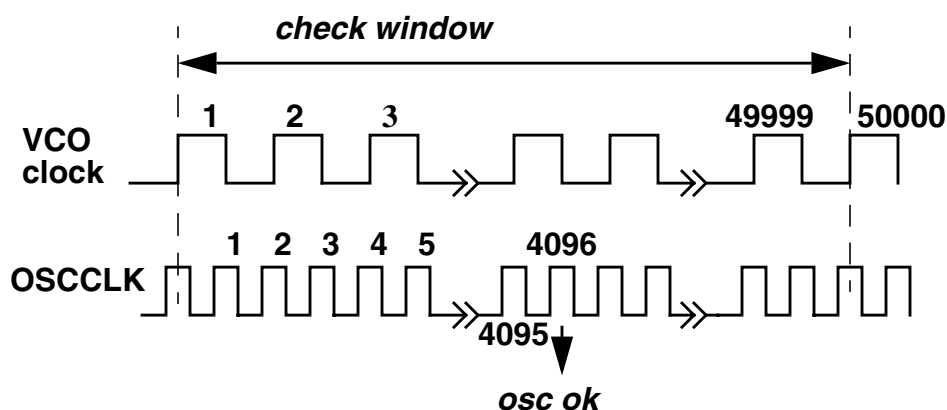
The clock monitor performs a coarse check on the incoming clock signal. The clock quality checker provides a more accurate check in addition to the clock monitor.

A clock quality check is triggered by any of the following events:

- Power-on reset (POR)
- Low voltage reset (LVR)
- Wake-up from full stop mode (exit full stop)
- Clock monitor fail indication (CM fail)

A time window of 50000 VCO clock cycles<sup>1</sup> is called *check window*.

A number greater equal than 4096 rising OSCCLK edges within a *check window* is called *osc ok*. Note that *osc ok* immediately terminates the current *check window*. See Figure 9-19 as an example.



**Figure 9-19. Check Window Example**

1. VCO clock cycles are generated by the PLL when running at minimum frequency  $f_{SCM}$ .



- a) the 14 most significant bits of the extended identifier plus the SRR and IDE bits of CAN 2.0B messages or
  - b) the 11 bits of the standard identifier, the RTR and IDE bits of CAN 2.0A/B messages.
- Figure 10-40 shows how the first 32-bit filter bank (CANIDAR0–CANIDA3, CANIDMR0–3CANIDMR) produces filter 0 and 1 hits. Similarly, the second filter bank (CANIDAR4–CANIDAR7, CANIDMR4–CANIDMR7) produces filter 2 and 3 hits.
- Eight identifier acceptance filters, each to be applied to the first 8 bits of the identifier. This mode implements eight independent filters for the first 8 bits of a CAN 2.0A/B compliant standard identifier or a CAN 2.0B compliant extended identifier. Figure 10-41 shows how the first 32-bit filter bank (CANIDAR0–CANIDAR3, CANIDMR0–CANIDMR3) produces filter 0 to 3 hits. Similarly, the second filter bank (CANIDAR4–CANIDAR7, CANIDMR4–CANIDMR7) produces filter 4 to 7 hits.
  - Closed filter. No CAN message is copied into the foreground buffer RxFG, and the RXF flag is never set.

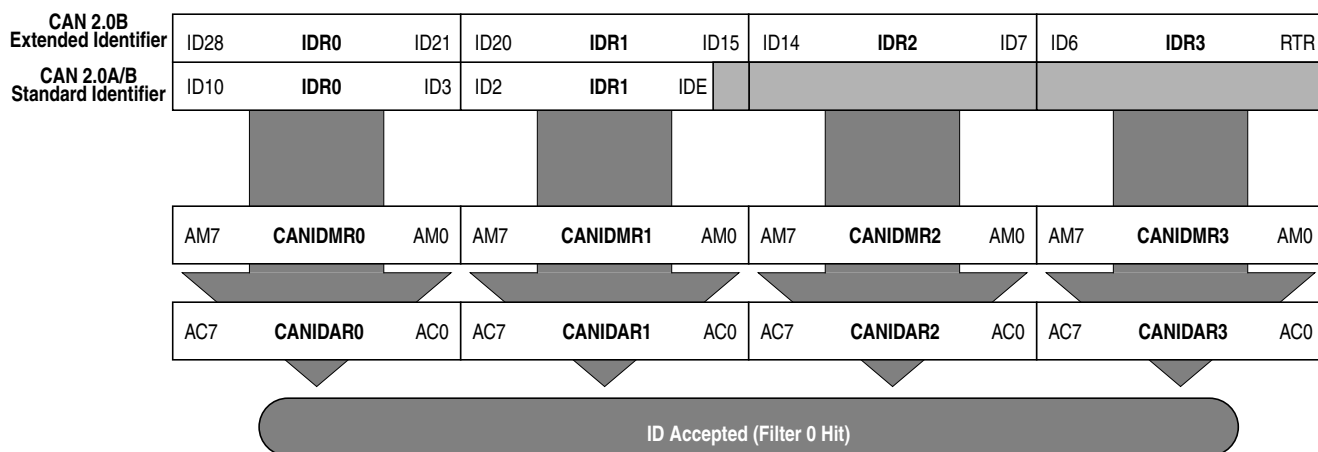


Figure 10-39. 32-bit Maskable Identifier Acceptance Filter

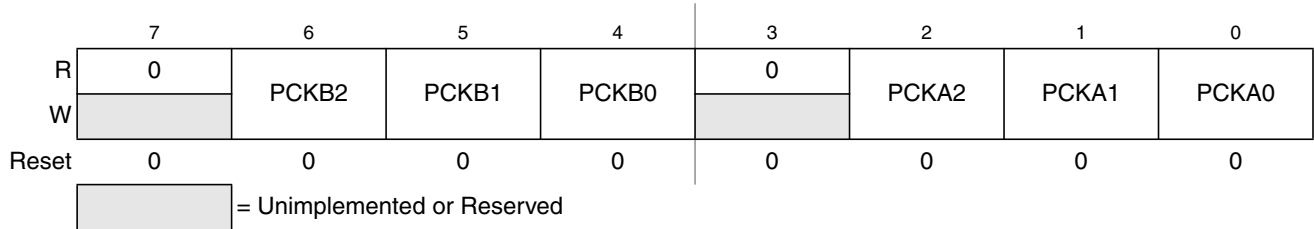
**Table 12-4. PWMCLK Field Descriptions**

Field	Description
5 PCLK5	<b>Pulse Width Channel 5 Clock Select</b> 0 Clock A is the clock source for PWM channel 5. 1 Clock SA is the clock source for PWM channel 5.
4 PCLK4	<b>Pulse Width Channel 4 Clock Select</b> 0 Clock A is the clock source for PWM channel 4. 1 Clock SA is the clock source for PWM channel 4.
3 PCLK3	<b>Pulse Width Channel 3 Clock Select</b> 0 Clock B is the clock source for PWM channel 3. 1 Clock SB is the clock source for PWM channel 3.
2 PCLK2	<b>Pulse Width Channel 2 Clock Select</b> 0 Clock B is the clock source for PWM channel 2. 1 Clock SB is the clock source for PWM channel 2.
1 PCLK1	<b>Pulse Width Channel 1 Clock Select</b> 0 Clock A is the clock source for PWM channel 1. 1 Clock SA is the clock source for PWM channel 1.
0 PCLK0	<b>Pulse Width Channel 0 Clock Select</b> 0 Clock A is the clock source for PWM channel 0. 1 Clock SA is the clock source for PWM channel 0.

### 12.3.2.4 PWM Prescale Clock Select Register (PWMPRCLK)

This register selects the prescale clock source for clocks A and B independently.

Module Base + 0x0003



**Figure 12-6. PWM Prescaler Clock Select Register (PWMPRCLK)**

Read: anytime

Write: anytime

#### NOTE

PCKB2–PCKB0 and PCKA2–PCKA0 register bits can be written anytime. If the clock prescale is changed while a PWM signal is being generated, a truncated or stretched pulse can occur during the transition.

**Table 13-11. Start Bit Verification**

RT3, RT5, and RT7 Samples	Start Bit Verification	Noise Flag
100	Yes	1
101	No	0
110	No	0
111	No	0

If start bit verification is not successful, the RT clock is reset and a new search for a start bit begins.

To determine the value of a data bit and to detect noise, recovery logic takes samples at RT8, RT9, and RT10. [Table 13-12](#) summarizes the results of the data bit samples.

**Table 13-12. Data Bit Recovery**

RT8, RT9, and RT10 Samples	Data Bit Determination	Noise Flag
000	0	0
001	0	1
010	0	1
011	1	1
100	0	1
101	1	1
110	1	1
111	1	0

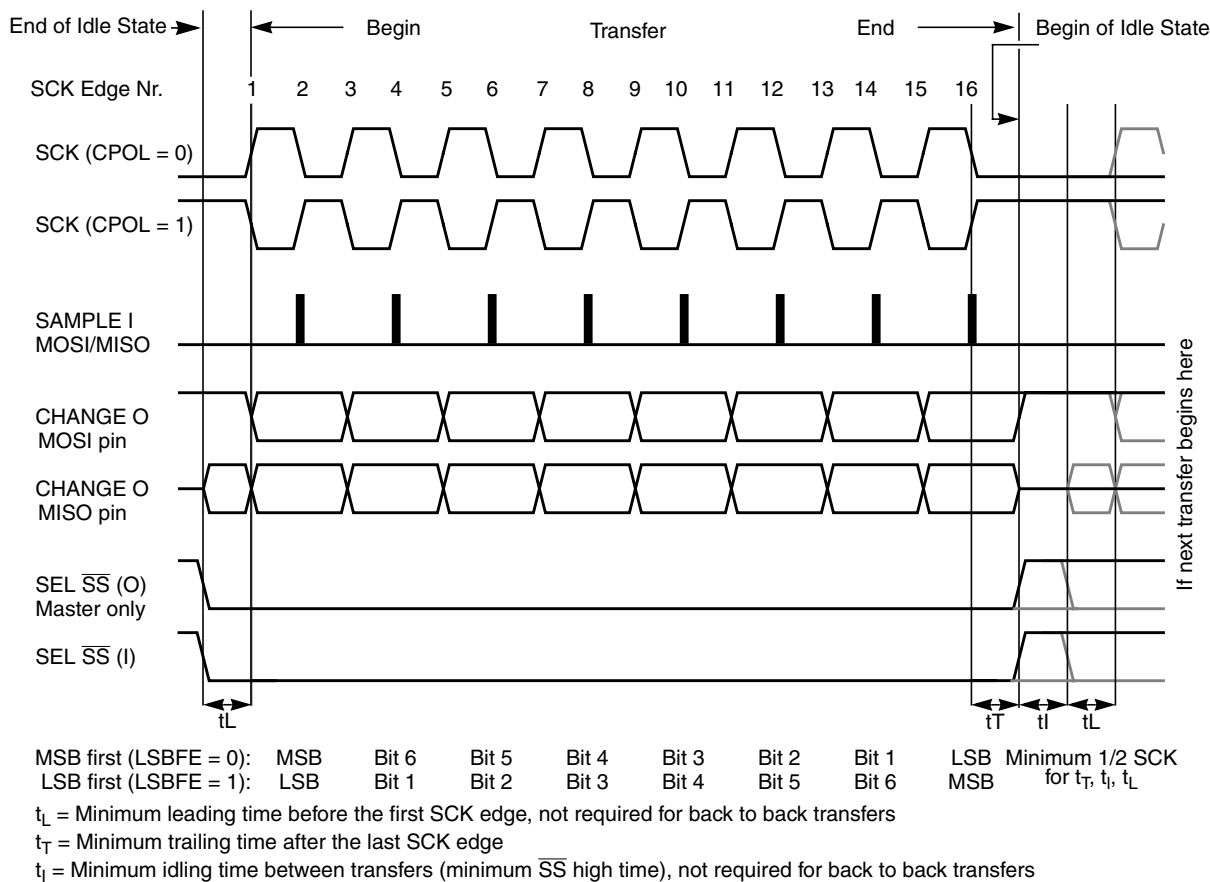
### NOTE

The RT8, RT9, and RT10 samples do not affect start bit verification. If any or all of the RT8, RT9, and RT10 start bit samples are logic 1s following a successful start bit verification, the noise flag (NF) is set and the receiver assumes that the bit is a start bit (logic 0).

To verify a stop bit and to detect noise, recovery logic takes samples at RT8, RT9, and RT10. [Table 13-13](#) summarizes the results of the stop bit samples.

**Table 13-13. Stop Bit Recovery**

RT8, RT9, and RT10 Samples	Framing Error Flag	Noise Flag
000	1	0
001	1	1
010	1	1
011	0	1
100	1	1
101	0	1
110	0	1
111	0	0



**Figure 14-10. SPI Clock Format 1 (CPHA = 1)**

#### 14.4.4 SPI Baud Rate Generation

Baud rate generation consists of a series of divider stages. Six bits in the SPI Baud Rate register (SPPR2, SPPR1, SPPR0, SPR2, SPR1, and SPR0) determine the divisor to the SPI module clock which results in the SPI baud rate.

The SPI clock rate is determined by the product of the value in the baud rate preselection bits (SPPR2–SPPR0) and the value in the baud rate selection bits (SPR2–SPR0). The module clock divisor equation is shown in [Figure 14-11](#)

When all bits are clear (the default condition), the SPI module clock is divided by 2. When the selection bits (SPR2–SPR0) are 001 and the preselection bits (SPPR2–SPPR0) are 000, the module clock divisor becomes 4. When the selection bits are 010, the module clock divisor becomes 8 etc.

When the preselection bits are 001, the divisor determined by the selection bits is multiplied by 2. When the preselection bits are 010, the divisor is multiplied by 3, etc. See [Table 14-7](#) for baud rate calculations for all bit conditions, based on a 25-MHz bus clock. The two sets of selects allows the clock to be divided by a non-power of two to achieve other baud rates such as divide by 6, divide by 10, etc.

### 15.3.2.10 Timer Interrupt Enable Register (TIE)

Module Base + 0x000C

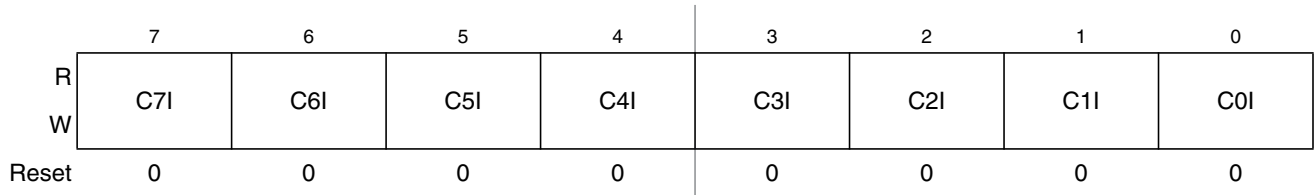


Figure 15-18. Timer Interrupt Enable Register (TIE)

Read: Anytime

Write: Anytime.

Table 15-14. TIE Field Descriptions

Field	Description
7:0 C7I:C0I	<b>Input Capture/Output Compare “x” Interrupt Enable</b> — The bits in TIE correspond bit-for-bit with the bits in the TFLG1 status register. If cleared, the corresponding flag is disabled from causing a hardware interrupt. If set, the corresponding flag is enabled to cause a interrupt.

### 15.3.2.11 Timer System Control Register 2 (TSCR2)

Module Base + 0x000D

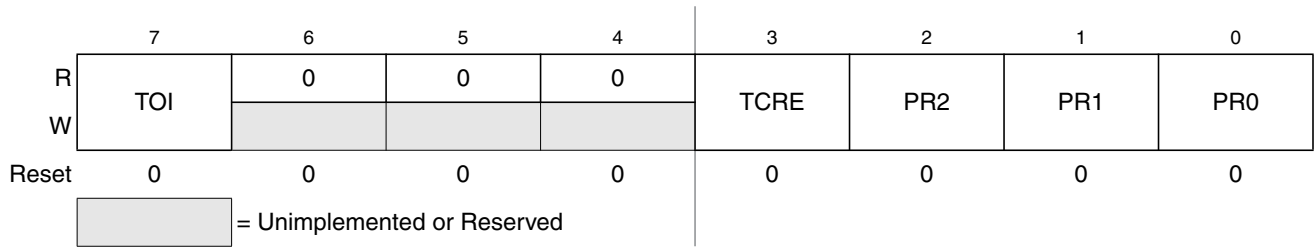


Figure 15-19. Timer System Control Register 2 (TSCR2)

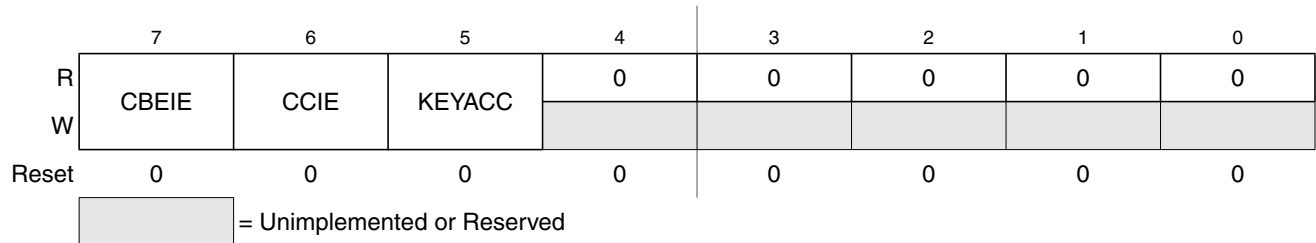
Read: Anytime

Write: Anytime.

### 17.3.2.4 Flash Configuration Register (FCNFG)

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

Module Base + 0x0003



**Figure 17-7. Flash Configuration Register (FCNFG)**

CBEIE, CCIE, and KEYACC are readable and writable while remaining bits read 0 and are not writable. KEYACC is only writable if the KEYEN bit in the FSEC register is set to the enabled state (see [Section 17.3.2.2](#)).

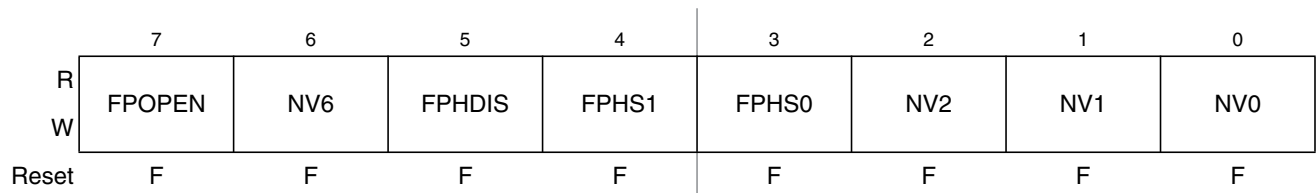
**Table 17-7. FCNFG Field Descriptions**

Field	Description
7 CBEIE	<b>Command Buffer Empty Interrupt Enable</b> — The CBEIE bit enables the interrupts in case of an empty command buffer in the Flash module. 0 Command Buffer Empty interrupts disabled 1 An interrupt will be requested whenever the CBEIF flag is set (see <a href="#">Section 17.3.2.6</a> )
6 CCIE	<b>Command Complete Interrupt Enable</b> — The CCIE bit enables the interrupts in case of all commands being completed in the Flash module. 0 Command Complete interrupts disabled 1 An interrupt will be requested whenever the CCIF flag is set (see <a href="#">Section 17.3.2.6</a> )
5 KEYACC	<b>Enable Security Key Writing.</b> 0 Flash writes are interpreted as the start of a command write sequence 1 Writes to the Flash array are interpreted as a backdoor key while reads of the Flash array return invalid data

### 17.3.2.5 Flash Protection Register (FPROT)

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

Module Base + 0x0004



**Figure 17-8. Flash Protection Register (FPROT)**

The FPROT register is readable in normal and special modes. FPOPEN can only be written from a 1 to a 0. FPHS[1:0] can be written anytime until FPHDIS is cleared. The FPROT register is loaded from Flash address 0xFF0D during the reset sequence, indicated by F in [Figure 17-8](#).

## 18.3 Memory Map and Registers

This section describes the [FTS32K](#) memory map and registers.

### 18.3.1 Module Memory Map

The [FTS32K](#) memory map is shown in [Figure 18-2](#). The HCS12 architecture places the Flash array addresses between [0x4000](#) and [0xFFFF](#), which corresponds to three 16 Kbyte pages. The content of the HCS12 Core PPAGE register is used to map the logical [middle](#) page ranging from address [0x8000](#) to [0xBFFF](#) to any physical 16K byte page in the Flash array memory.<sup>1</sup> The FPROT register (see [Section 18.3.2.5](#)) can be set to globally protect the entire Flash array. Three separate areas, one starting from the Flash array starting address (called lower) towards higher addresses, one growing downward from the Flash array end address (called higher), and the remaining addresses, can be activated for protection. The Flash array addresses covered by these protectable regions are shown in [Figure 18-2](#). The higher address area is mainly targeted to hold the boot loader code since it covers the vector space. The lower address area 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. Default protection settings as well as security information that allows the MCU to restrict access to the Flash module are stored in the Flash configuration field described in [Table 18-1](#).

**Table 18-1. Flash Configuration Field**

Flash Address	Size (bytes)	Description
0xFF00–0xFF07	8	Backdoor Key to unlock security
0xFF08–0xFF0C	5	Reserved
0xFF0D	1	Flash Protection byte Refer to <a href="#">Section 18.3.2.5</a> , “Flash Protection Register (FPROT)”
0xFF0E	1	Reserved
0xFF0F	1	Flash Security/Options byte Refer to <a href="#">Section 18.3.2.2</a> , “Flash Security Register (FSEC)”

1. By placing [0x3E/0x3F](#) in the HCS12 Core PPAGE register, the [bottom/top fixed](#) 16 Kbyte pages can be seen twice in the MCU memory map.

### 19.4.1.2 Command Write Sequence

The Flash command controller is used to supervise the command write sequence to execute program, erase, and erase verify algorithms.

Before starting a command write sequence, the ACCERR and PVIOL flags in the FSTAT register must be clear 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. The basic command write sequence is as follows:

1. Write to a valid address in the Flash array memory.
2. Write a valid command to the FCMD register.
3. Clear the CBEIF flag in the FSTAT register by writing a 1 to CBEIF to launch the command.

The address written in step 1 will be stored in the FADDR registers and the data will be stored in the FDATA registers. When the CBEIF flag is cleared in step 3, the CCIF flag is cleared by the Flash command controller indicating that the command was successfully launched. For all command write sequences, the CBEIF flag will set after the CCIF flag is cleared indicating that the address, data, and command buffers are ready for a new command write sequence to begin. A buffered command will wait for the active operation to be completed before being launched. Once a command is launched, the completion of the command operation is indicated by the setting of the CCIF flag in the FSTAT register. The CCIF flag will set upon completion of all active and buffered commands.



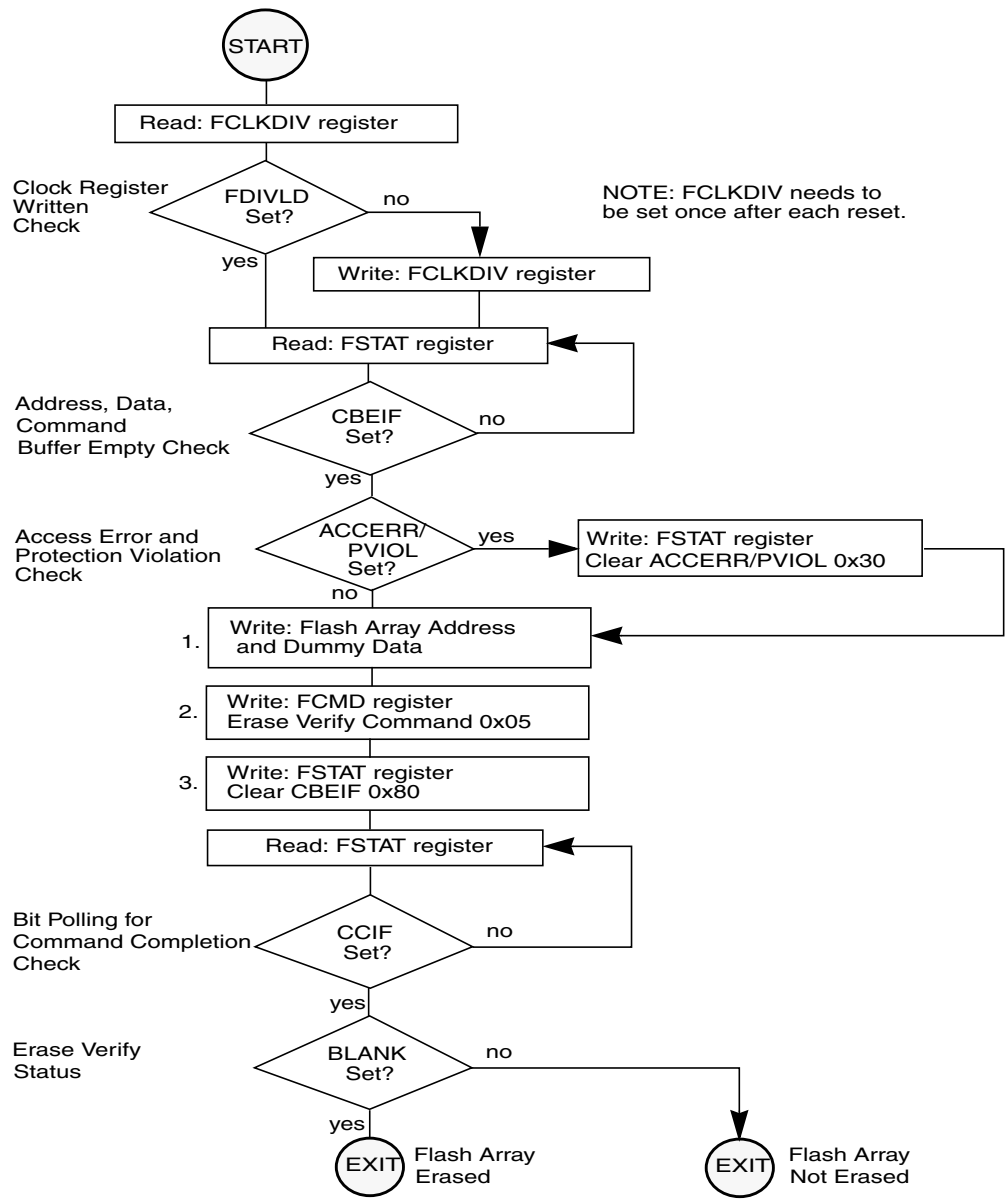


Figure 19-25. Example Erase Verify Command Flow

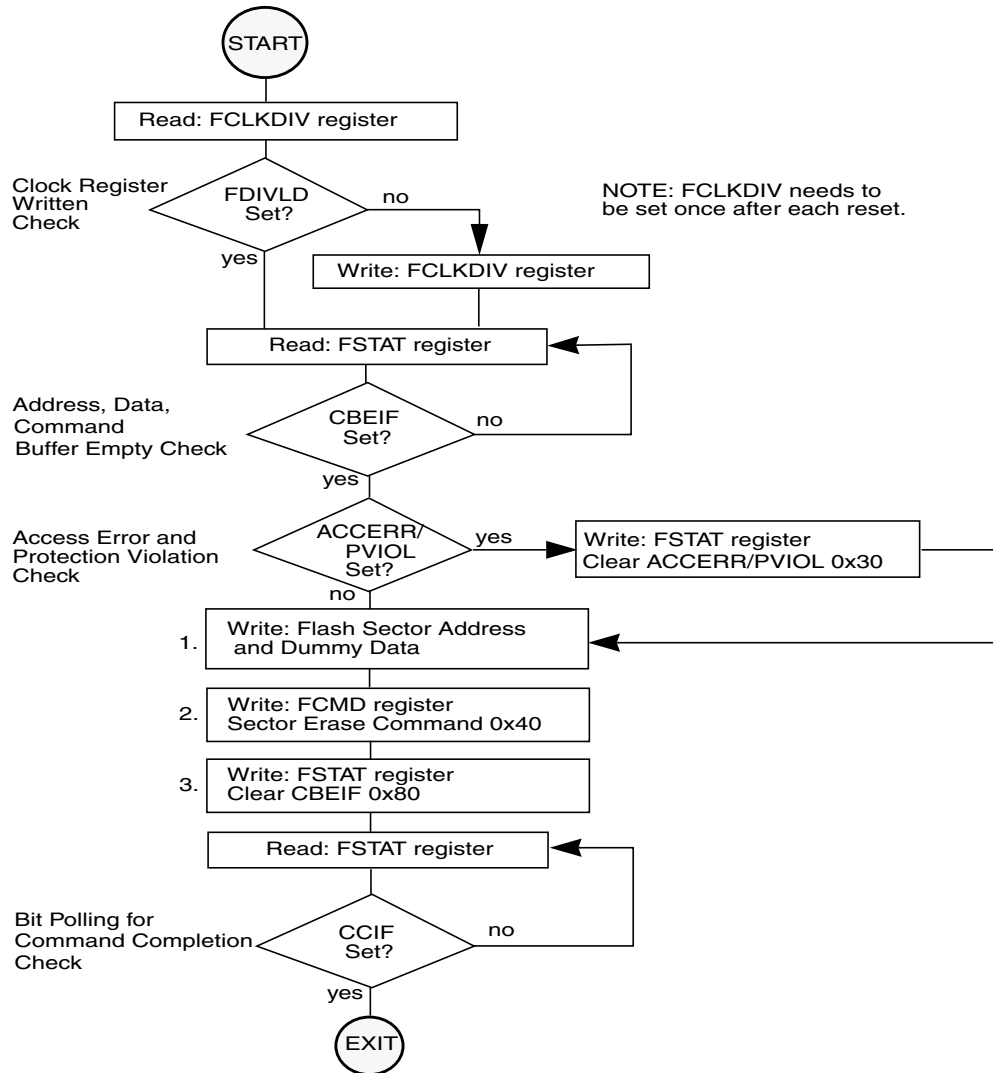
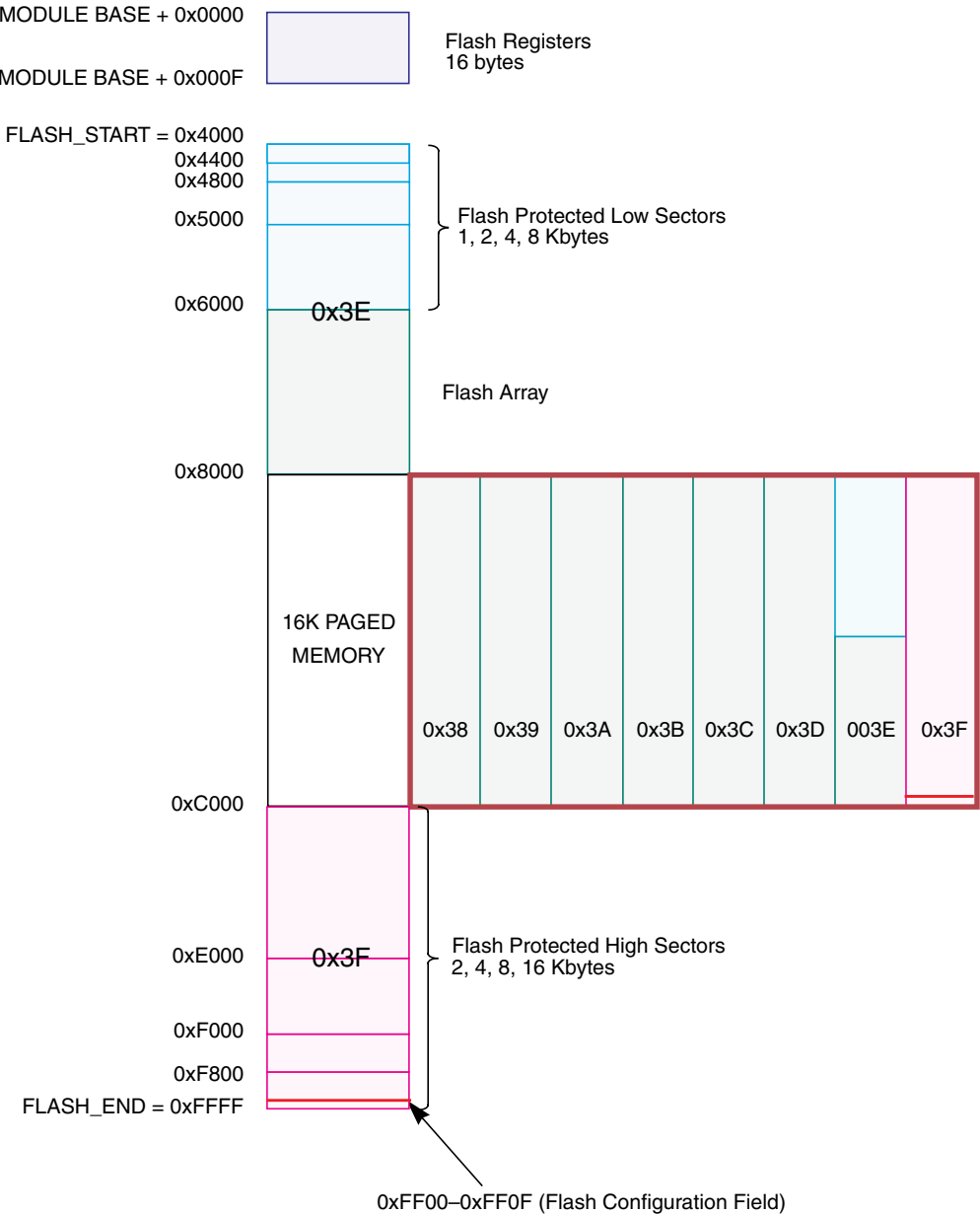


Figure 19-27. Example Sector Erase Command Flow



Note: 0x38-0x3F correspond to the PPAGE register content

Figure 21-2. Flash Memory Map

**Table A-5. Thermal Package Characteristics<sup>(1)</sup>**

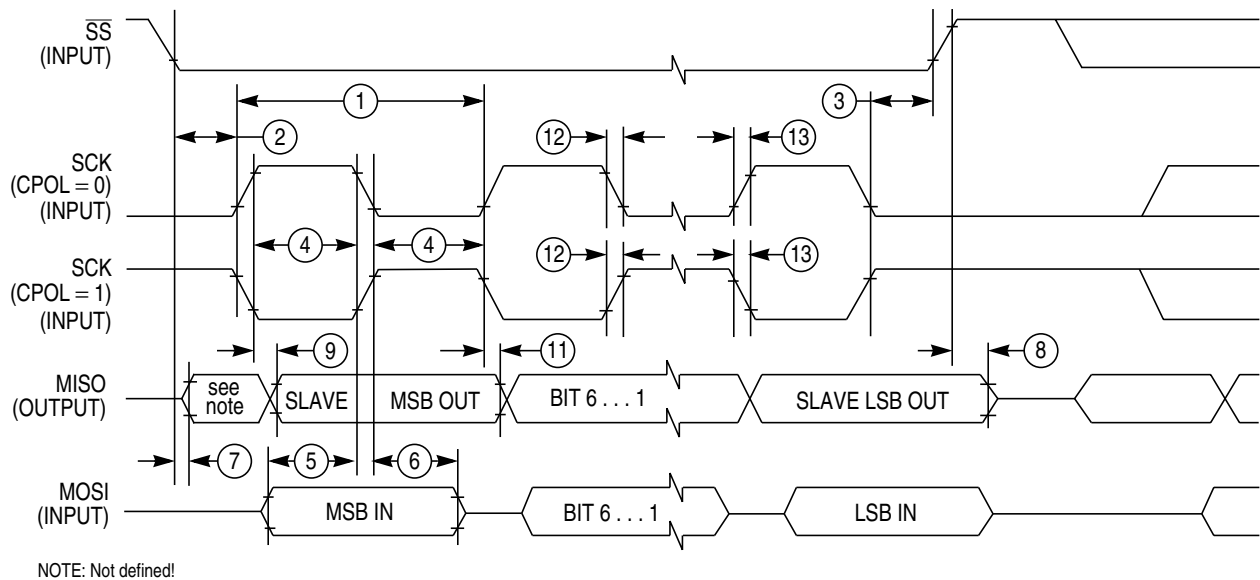
Num	C	Rating	Symbol	Min	Typ	Max	Unit
1	T	Thermal Resistance LQFP48, single layer PCB <sup>(2)</sup>	$\theta_{JA}$	—	—	69	°C/W
2	T	Thermal Resistance LQFP48, double sided PCB with 2 internal planes <sup>(3)</sup>	$\theta_{JA}$	—	—	53	°C/W
3	T	Junction to Board LQFP48	$\theta_{JB}$	—	—	30	°C/W
4	T	Junction to Case LQFP48	$\theta_{JC}$	—	—	20	°C/W
5	T	Junction to Package Top LQFP48	$\Psi_{JT}$	—	—	4	°C/W
6	T	Thermal Resistance LQFP52, single sided PCB	$\theta_{JA}$	—	—	65	°C/W
7	T	Thermal Resistance LQFP52, double sided PCB with 2 internal planes	$\theta_{JA}$	—	—	49	°C/W
8	T	Junction to Board LQFP52	$\theta_{JB}$	—	—	31	°C/W
9	T	Junction to Case LQFP52	$\theta_{JC}$	—	—	17	°C/W
10	T	Junction to Package Top LQFP52	$\Psi_{JT}$	—	—	3	°C/W
11	T	Thermal Resistance QFP 80, single sided PCB	$\theta_{JA}$	—	—	52	°C/W
12	T	Thermal Resistance QFP 80, double sided PCB with 2 internal planes	$\theta_{JA}$	—	—	42	°C/W
13	T	Junction to Board QFP80	$\theta_{JB}$	—	—	28	°C/W
14	T	Junction to Case QFP80	$\theta_{JC}$	—	—	18	°C/W
15	T	Junction to Package Top QFP80	$\Psi_{JT}$	—	—	4	°C/W

1. The values for thermal resistance are achieved by package simulations

2. PC Board according to EIA/JEDEC Standard 51-2

3. PC Board according to EIA/JEDEC Standard 51-7

In [Figure A-9](#) the timing diagram for slave mode with transmission format CPHA=1 is depicted.



**Figure A-9. SPI Slave Timing (CPHA=1)**

In [Table A-22](#) the timing characteristics for slave mode are listed.

**Table A-22. SPI Slave Mode Timing Characteristics**

Num	C	Characteristic	Symbol	Min	Typ	Max	Unit
1	D	SCK Frequency	$f_{sck}$	DC	—	1/4	$f_{bus}$
1	P	SCK Period	$t_{sck}$	4	—	$\infty$	$t_{bus}$
2	D	Enable Lead Time	$t_{lead}$	4	—	—	$t_{bus}$
3	D	Enable Lag Time	$t_{lag}$	4	—	—	$t_{bus}$
4	D	Clock (SCK) High or Low Time	$t_{wsck}$	4	—	—	$t_{bus}$
5	D	Data Setup Time (Inputs)	$t_{su}$	8	—	—	ns
6	D	Data Hold Time (Inputs)	$t_{hi}$	8	—	—	ns
7	D	Slave Access Time (time to data active)	$t_a$	—	—	20	ns
8	D	Slave MISO Disable Time	$t_{dis}$	—	—	22	ns
9	D	Data Valid after SCK Edge	$t_{vsck}$	—	—	$30 + t_{bus}^{(1)}$	ns
10	D	Data Valid after SS fall	$t_{vss}$	—	—	$30 + t_{bus}^{(1)}$	ns
11	D	Data Hold Time (Outputs)	$t_{ho}$	20	—	—	ns
12	D	Rise and Fall Time Inputs	$t_{rfi}$	—	—	8	ns
13	D	Rise and Fall Time Outputs	$t_{rfo}$	—	—	8	ns

1.  $t_{bus}$  added due to internal synchronization delay