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	HCS12X
Core Size	16-Bit
Speed	50MHz
Connectivity	CANbus, EBI/EMI, I²C, IrDA, SCI, SPI
Peripherals	LVD, POR, PWM, WDT
Number of I/O	91
Program Memory Size	512KB (512K x 8)
Program Memory Type	FLASH
EEPROM Size	4K x 8
RAM Size	32K x 8
Voltage - Supply (Vcc/Vdd)	1.72V ~ 5.5V
Data Converters	A/D 16x12b
Oscillator Type	External
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	112-LQFP
Supplier Device Package	112-LQFP (20x20)
Purchase URL	https://www.e-xfl.com/product-detail/nxp-semiconductors/mc9s12xeq512cal

Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0368 PTR	R	PTR7	PTR6	PTR5	PTR4	PTR3	PTR2	PTR1	PTR0
	W								
0x0369 PTIR	R	PTIR7	PTIR6	PTIR5	PTIR4	PTIR3	PTIR2	PTIR1	PTIR0
	W								
0x036A DDRR	R	DDRR7	DDRR6	DDRR5	DDRR4	DDRR3	DDRR2	DDRR1	DDRR0
	W								
0x036B RDRR	R	RDRR7	RDRR6	RDRR5	RDRR4	RDRR3	RDRR2	RDRR1	RDRR0
	W								
0x036C PERR	R	PERR7	PERR6	PERR5	PERR4	PERR3	PERR2	PERR1	PERR0
	W								
0x036D PPSR	R	PPSR7	PPSR6	PPSR5	PPSR4	PPSR3	PPSR2	PPSR1	PPSR0
	W								
0x036E Reserved	R	0	0	0	0	0	0	0	0
	W								
0x036F PTRRR	R	PTRRR7	PTRRR6	PTRRR5	PTRRR4	PTRRR3	PTRRR2	PTRRR1	PTRRR0
	W								
0x0370 PTL	R	PTL7	PTL6	PTL5	PTL4	PTL3	PTL2	PTL1	PTL0
	W								
0x0371 PTIL	R	PTIL7	PTIL6	PTIL5	PTIL4	PTIL3	PTIL2	PTIL1	PTIL0
	W								
0x0372 DDRL	R	DDRL7	DDRL6	DDRL5	DDRL4	DDRL3	DDRL2	DDRL1	DDRL0
	W								
0x0373 RDRL	R	RDRL7	RDRL6	RDRL5	RDRL4	RDRL3	RDRL2	RDRL1	RDRL0
	W								
0x0374 PERL	R	PERL7	PERL6	PERL5	PERL4	PERL3	PERL2	PERL1	PERL0
	W								
0x0375 PPSL	R	PPSL7	PPSL6	PPSL5	PPSL4	PPSL3	PPSL2	PPSL1	PPSL0
	W								
0x0376 WOML	R	WOML7	WOML6	WOML5	WOML4	WOML3	WOML2	WOML1	WOML0
	W								
0x0377 PTLRR	R	PTLRR7	PTLRR6	PTLRR5	PTLRR4	0	0	0	0
	W								

= Unimplemented or Reserved

- Each descriptor can be configured to allow one of four types of access privilege for the defined memory region
  - Bus master has full access (read, write and execute enabled)
  - Bus master can read and execute (write illegal)
  - Bus master can read and write (execution illegal)
  - Bus master can only read (write and execution illegal)
- Accesses to memory not covered by any protection descriptor will cause an access violation

4.1.4 Modes of Operation

The MPU module can be used in all MCU modes.

4.2 External Signal Description

The MPU module has no external signals.

4.3 Memory Map and Register Definition

This section provides a detailed description of address space and registers used by the MPU module.

6.4.1 S12X Exception Requests

The CPU handles both reset requests and interrupt requests. The XINT module contains registers to configure the priority level of each I bit maskable interrupt request which can be used to implement an interrupt priority scheme. This also includes the possibility to nest interrupt requests. A priority decoder is used to evaluate the priority of a pending interrupt request.

6.4.2 Interrupt Prioritization

After system reset all interrupt requests with a vector address lower than or equal to (vector base + 0x00F2) are enabled, are set up to be handled by the CPU and have a pre-configured priority level of 1. Exceptions to this rule are the non-maskable interrupt requests and the spurious interrupt vector request at (vector base + 0x0010) which cannot be disabled, are always handled by the CPU and have a fixed priority levels. A priority level of 0 effectively disables the associated I bit maskable interrupt request.

If more than one interrupt request is configured to the same interrupt priority level the interrupt request with the higher vector address wins the prioritization.

The following conditions must be met for an I bit maskable interrupt request to be processed.

1. The local interrupt enabled bit in the peripheral module must be set.
2. The setup in the configuration register associated with the interrupt request channel must meet the following conditions:
   a) The XGATE request enable bit must be 0 to have the CPU handle the interrupt request.
   b) The priority level must be set to non zero.
   c) The priority level must be greater than the current interrupt processing level in the condition code register (CCR) of the CPU (PRIOLVL[2:0] > IPL[2:0]).
3. The I bit in the condition code register (CCR) of the CPU must be cleared.
4. There is no access violation interrupt request pending.
5. There is no SYS, SWI, BDM, TRAP, or $\overline{\text{XIRQ}}$ request pending.

NOTE

All non I bit maskable interrupt requests always have higher priority than I bit maskable interrupt requests. If an I bit maskable interrupt request is interrupted by a non I bit maskable interrupt request, the currently active interrupt processing level (IPL) remains unaffected. It is possible to nest non I bit maskable interrupt requests, e.g., by nesting SWI or TRAP calls.

6.4.2.1 Interrupt Priority Stack

The current interrupt processing level (IPL) is stored in the condition code register (CCR) of the CPU. This way the current IPL is automatically pushed to the stack by the standard interrupt stacking procedure. The new IPL is copied to the CCR from the priority level of the highest priority active interrupt request channel which is configured to be handled by the CPU. The copying takes place when the interrupt vector is fetched. The previous IPL is automatically restored by executing the RTI instruction.

BITH — Bit Test Immediate 8 bit Constant (High Byte) — BITH

Operation

RD.H & IMM8 ⇒ NONE

Performs a bit wise logical AND between the high byte of register RD and an immediate 8 bit constant. Only the condition code flags get updated, but no result is written back.

CCR Effects

N	Z	V	C
Δ	Δ	0	—

N: Set if bit 15 of the result is set; cleared otherwise.
Z: Set if the 8 bit result is $00; cleared otherwise.
V: 0; cleared.
C: Not affected.

Code and CPU Cycles

Source Form	Address Mode	Machine Code							Cycles
BITH RD, #IMM8	IMM8	1	0	0	1	1	RD	IMM8	P

BPL Branch if Plus BPL

Operation

If N = 0, then PC + $0002 + (REL9 << 1) ⇒ PC

Tests the Sign flag and branches if N = 0.

CCR Effects

N	Z	V	C
—	—	—	—

N: Not affected.
Z: Not affected.
V: Not affected.
C: Not affected.

Code and CPU Cycles

Source Form	Address Mode	Machine Code								Cycles
BPL REL9	REL9	0	0	1	0	1	0	0	REL9	PP/P

LDH — Load Immediate 8 bit Constant (High Byte) — LDH

Operation

IMM8 ⇒ RD.H;

Loads an 8 bit immediate constant into the high byte of register RD. The low byte is not affected.

CCR Effects

N	Z	V	C
—	—	—	—

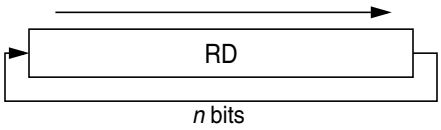
N: Not affected.
Z: Not affected.
V: Not affected.
C: Not affected.

Code and CPU Cycles

Source Form	Address Mode	Machine Code							Cycles
LDH RD, #IMM8	IMM8	1	1	1	1	1	RD	IMM8	P

ROR — Rotate Right — ROR

Operation



n = RS or IMM4

Rotates the bits in register RD *n* positions to the right. The upper *n* bits of the register RD are filled with the lower *n* bits. Two source forms are available. In the first form, the parameter *n* is contained in the instruction code as an immediate operand. In the second form, the parameter is contained in the lower bits of the source register RS[3:0]. All other bits in RS are ignored. If *n* is zero no shift will take place and the register RD will be unaffected; however, the condition code flags will be updated.

CCR Effects

N	Z	V	C
Δ	Δ	0	—

N: Set if bit 15 of the result is set; cleared otherwise.
Z: Set if the result is $0000; cleared otherwise.
V: 0; cleared.
C: Not affected.

Code and CPU Cycles

Source Form	Address Mode	Machine Code												Cycles
ROR RD, #IMM4	IMM4	0	0	0	0	1	RD	IMM4		1	1	1	1	P
ROR RD, RS	DYA	0	0	0	0	1	RD	RS	1	0	1	1	1	P

Module Base + 0x001F

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

Figure 14-35. Timer Input Capture/Output Compare Register 7 Low (TC7)

Read: Anytime

Write anytime for output compare function. Writes to these registers have no meaning or effect during input capture.

All bits reset to zero.

Depending on the TIOS bit for the corresponding channel, these registers are used to latch the value of the free-running counter when a defined transition is sensed by the corresponding input capture edge detector or to trigger an output action for output compare.

14.3.2.15 16-Bit Pulse Accumulator A Control Register (PACTL)

Module Base + 0x0020

	7	6	5	4	3	2	1	0
R	0	PAEN	PAMOD	PEDGE	CLK1	CLK0	PAOVI	PAI
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 14-36. 16-Bit Pulse Accumulator Control Register (PACTL)

Read: Anytime

Write: Anytime

All bits reset to zero.

Table 14-19. PACTL Field Descriptions

Field	Description
6 PAEN	**Pulse Accumulator A System Enable** — PAEN is independent from TEN. With timer disabled, the pulse accumulator can still function unless pulse accumulator is disabled. 0 16-Bit Pulse Accumulator A system disabled. 8-bit PAC3 and PAC2 can be enabled when their related enable bits in ICPAR are set. Pulse Accumulator Input Edge Flag (PAIF) function is disabled. 1 16-Bit Pulse Accumulator A system enabled. The two 8-bit pulse accumulators PAC3 and PAC2 are cascaded to form the PACA 16-bit pulse accumulator. When PACA in enabled, the PACN3 and PACN2 registers contents are respectively the high and low byte of the PACA. PA3EN and PA2EN control bits in ICPAR have no effect. Pulse Accumulator Input Edge Flag (PAIF) function is enabled. The PACA shares the input pin with IC7.
5 PAMOD	**Pulse Accumulator Mode** — This bit is active only when the Pulse Accumulator A is enabled (PAEN = 1). 0 Event counter mode 1 Gated time accumulation mode

16.4 Functional Description

16.4.1 General

This section provides a complete functional description of the MSCAN.

16.4.2 Message Storage

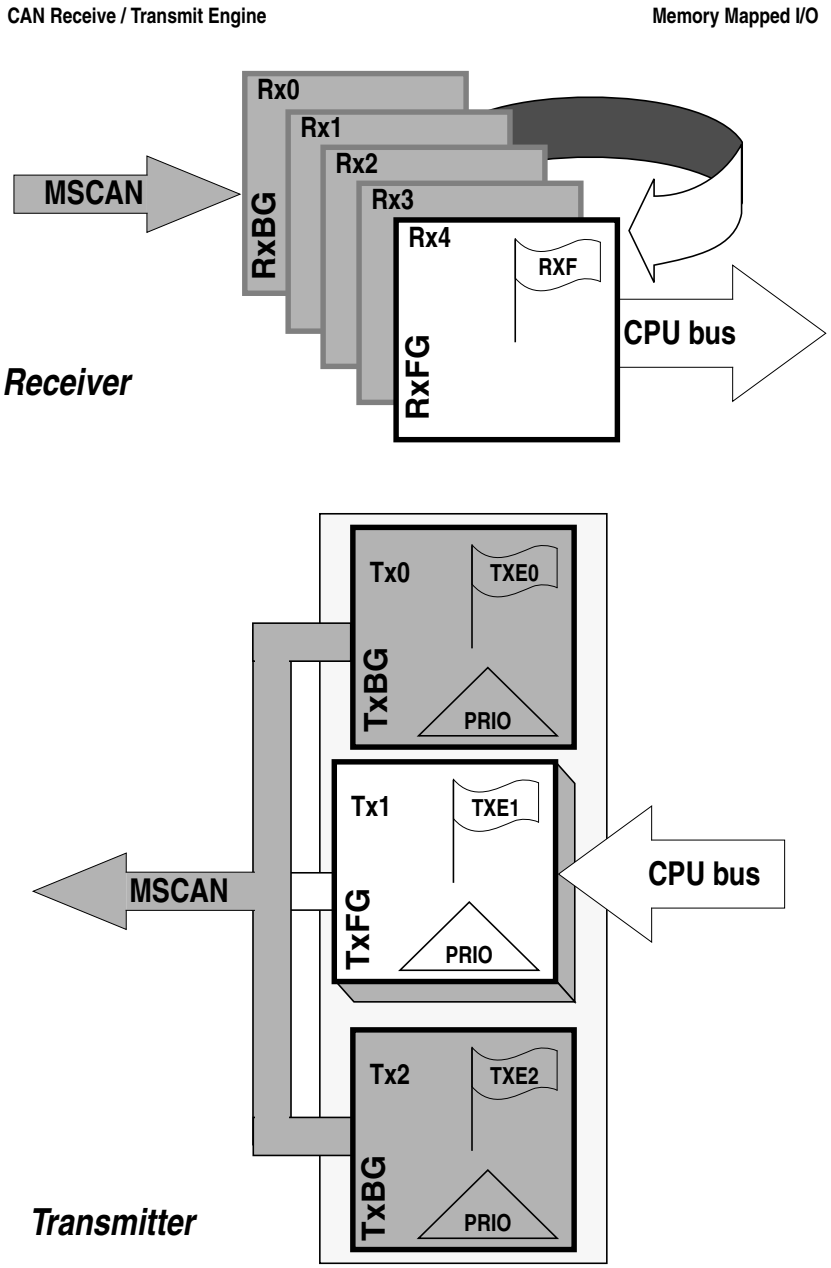


Figure 16-39. User Model for Message Buffer Organization

generates a receive interrupt[1] (see Section 16.4.7.3, "Receive Interrupt") to the CPU. The user's receive handler must read the received message from the RxFG and then reset the RXF flag to acknowledge the interrupt and to release the foreground buffer. A new message, which can follow immediately after the IFS field of the CAN frame, is received into the next available RxBG. If the MSCAN receives an invalid message in its RxBG (wrong identifier, transmission errors, etc.) the actual contents of the buffer will be over-written by the next message. The buffer will then not be shifted into the FIFO.

When the MSCAN module is transmitting, the MSCAN receives its own transmitted messages into the background receive buffer, RxBG, but does not shift it into the receiver FIFO, generate a receive interrupt, or acknowledge its own messages on the CAN bus. The exception to this rule is in loopback mode (see Section 16.3.2.2, "MSCAN Control Register 1 (CANCTL1)") where the MSCAN treats its own messages exactly like all other incoming messages. The MSCAN receives its own transmitted messages in the event that it loses arbitration. If arbitration is lost, the MSCAN must be prepared to become a receiver.

An overrun condition occurs when all receive message buffers in the FIFO are filled with correctly received messages with accepted identifiers and another message is correctly received from the CAN bus with an accepted identifier. The latter message is discarded and an error interrupt with overrun indication is generated if enabled (see Section 16.4.7.5, "Error Interrupt"). The MSCAN remains able to transmit messages while the receiver FIFO is being filled, but all incoming messages are discarded. As soon as a receive buffer in the FIFO is available again, new valid messages will be accepted.

16.4.3 Identifier Acceptance Filter

The MSCAN identifier acceptance registers (see Section 16.3.2.12, "MSCAN Identifier Acceptance Control Register (CANIDAC)") define the acceptable patterns of the standard or extended identifier (ID[10:0] or ID[28:0]). Any of these bits can be marked 'don't care' in the MSCAN identifier mask registers (see Section 16.3.2.18, "MSCAN Identifier Mask Registers (CANIDMR0–CANIDMR7)").

A filter hit is indicated to the application software by a set receive buffer full flag (RXF = 1) and three bits in the CANIDAC register (see Section 16.3.2.12, "MSCAN Identifier Acceptance Control Register (CANIDAC)"). These identifier hit flags (IDHIT[2:0]) clearly identify the filter section that caused the acceptance. They simplify the application software's task to identify the cause of the receiver interrupt. If more than one hit occurs (two or more filters match), the lower hit has priority.

A very flexible programmable generic identifier acceptance filter has been introduced to reduce the CPU interrupt loading. The filter is programmable to operate in four different modes:

- Two identifier acceptance filters, each to be applied to:
  - The full 29 bits of the extended identifier and to the following bits of the CAN 2.0B frame:
    - Remote transmission request (RTR)
    - Identifier extension (IDE)
    - Substitute remote request (SRR)
  - The 11 bits of the standard identifier plus the RTR and IDE bits of the CAN 2.0A/B messages. This mode implements two filters for a full length CAN 2.0B compliant extended identifier. Although this mode can be used for standard identifiers, it is recommended to use the four or eight identifier acceptance filters.

1. The receive interrupt occurs only if not masked. A polling scheme can be applied on RXF also.

21.3.2.3 SPI Baud Rate Register (SPIBR)

Module Base +0x0002

	7	6	5	4	3	2	1	0
R	0	SPPR2	SPPR1	SPPR0	0	SPR2	SPR1	SPR0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 21-5. SPI Baud Rate Register (SPIBR)

Read: Anytime

Write: Anytime; writes to the reserved bits have no effect

Table 21-6. SPIBR Field Descriptions

Field	Description
6–4 SPPR[2:0]	**SPI Baud Rate Preselection Bits** — These bits specify the SPI baud rates as shown in Table 21-7. In master mode, a change of these bits will abort a transmission in progress and force the SPI system into idle state.
2–0 SPR[2:0]	**SPI Baud Rate Selection Bits** — These bits specify the SPI baud rates as shown in Table 21-7. In master mode, a change of these bits will abort a transmission in progress and force the SPI system into idle state.

The baud rate divisor equation is as follows:

$$\text{BaudRateDivisor} = (\text{SPPR} + 1) \bullet 2^{(\text{SPR} + 1)} \qquad \textit{Eqn. 21-1}$$

The baud rate can be calculated with the following equation:

$$\text{Baud Rate} = \text{BusClock} / \text{BaudRateDivisor} \qquad \textit{Eqn. 21-2}$$

NOTE

For maximum allowed baud rates, please refer to the SPI Electrical Specification in the Electricals chapter of this data sheet.

Table 21-7. Example SPI Baud Rate Selection (25 MHz Bus Clock) (Sheet 1 of 3)

SPPR2	SPPR1	SPPR0	SPR2	SPR1	SPR0	Baud Rate Divisor	Baud Rate
0	0	0	0	0	0	2	12.5 Mbit/s
0	0	0	0	0	1	4	6.25 Mbit/s
0	0	0	0	1	0	8	3.125 Mbit/s
0	0	0	0	1	1	16	1.5625 Mbit/s
0	0	0	1	0	0	32	781.25 kbit/s
0	0	0	1	0	1	64	390.63 kbit/s
0	0	0	1	1	0	128	195.31 kbit/s
0	0	0	1	1	1	256	97.66 kbit/s
0	0	1	0	0	0	4	6.25 Mbit/s
0	0	1	0	0	1	8	3.125 Mbit/s
0	0	1	0	1	0	16	1.5625 Mbit/s
0	0	1	0	1	1	32	781.25 kbit/s

Table 24-45. Erase All Blocks Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x08	Not required

Upon clearing CCIF to launch the Erase All Blocks command, the Memory Controller will erase the entire Flash memory space and verify that it is erased. If the Memory Controller verifies that the entire Flash memory space was properly erased, security will be released. During the execution of this command (CCIF=0) the user must not write to any Flash module register. The CCIF flag will set after the Erase All Blocks operation has completed.

Table 24-46. Erase All Blocks Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 000 at command launch
		Set if command not available in current mode (see Table 24-30)
	FPVIOL	Set if any area of the P-Flash memory is protected
	MGSTAT1	Set if any errors have been encountered during the verify operation(1)
	MGSTAT0	Set if any non-correctable errors have been encountered during the verify operation[1]
FERSTAT	EPVIOLIF	Set if any area of the buffer RAM EEE partition is protected

1. As found in the memory map for FTM256K2.

24.4.2.8 Erase P-Flash Block Command

The Erase P-Flash Block operation will erase all addresses in a P-Flash block.

Table 24-47. Erase P-Flash Block Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x09	Global address [22:16] to identify P-Flash block
001	Global address [15:0] in P-Flash block to be erased	

Upon clearing CCIF to launch the Erase P-Flash Block command, the Memory Controller will erase the selected P-Flash block and verify that it is erased. The CCIF flag will set after the Erase P-Flash Block operation has completed.

Chapter 25
256 KByte Flash Module (S12XFTM256K2V1)

Table 25-1. Revision History

Revision Number	Revision Date	Sections Affected	Description of Changes
V01.08	14 Nov 2007	25.5.2/25-951 25.4.2/25-927 25.4.2.8/25-933	- Changed terminology from ‘word program’ to “Program P-Flash’ in the BDM unsecuring description, Section 25.5.2 - Added requirement that user not write any Flash module register during execution of commands ‘Erase All Blocks’, Section 25.4.2.8, and ‘Unsecure Flash’, Section 25.4.2.11 - Added statement that security is released upon successful completion of command ‘Erase All Blocks’, Section 25.4.2.8
V01.09	19 Dec 2007	25.4.2.5/25-930 25.4.2/25-927 25.3.1/25-896	- Corrected Error Handling table for Load Data Field command - Corrected Error Handling table for Full Partition D-Flash, Partition D-Flash, and EEPROM Emulation Query commands - Corrected P-Flash IFR Accessibility table
V01.10	25 Sep 2009	25.1/25-891 25.3.2.1/25-903 25.4.2.4/25-930 25.4.2.7/25-932 25.4.2.12/25-936 25.4.2.12/25-936 25.4.2.12/25-936 25.4.2.20/25-945 25.3.2/25-901 25.3.2.1/25-903 25.4.1.2/25-922 25.6/25-951	- Clarify single bit fault correction for P-Flash phrase - Expand FDIV vs OSCCLK Frequency table - Add statement concerning code runaway when executing Read Once command from Flash block containing associated fields - Add statement concerning code runaway when executing Program Once command from Flash block containing associated fields - Add statement concerning code runaway when executing Verify Backdoor Access Key command from Flash block containing associated fields - Relate Key 0 to associated Backdoor Comparison Key address - Change “power down reset” to “reset” - Add ACCERR condition for Disable EEPROM Emulation command The following changes were made to clarify module behavior related to Flash register access during reset sequence and while Flash commands are active: - Add caution concerning register writes while command is active - Writes to FCLKDIV are allowed during reset sequence while CCIF is clear - Add caution concerning register writes while command is active - Writes to FCCOBIX, FCCOBHI, FCCOBLO registers are ignored during reset sequence

25.1 Introduction

The FTM256K2 module implements the following:

- 256 Kbytes of P-Flash (Program Flash) memory, consisting of 2 physical Flash blocks, intended primarily for nonvolatile code storage

Offset Module Base + 0x000A

	7	6	5	4	3	2	1	0
R W	CCOB[15:8]							
Reset	0	0	0	0	0	0	0	0

Figure 27-16. Flash Common Command Object High Register (FCCOBHI)

Offset Module Base + 0x000B

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

Figure 27-17. Flash Common Command Object Low Register (FCCOBLO)

27.3.2.11.1 FCCOB - NVM Command Mode

NVM command mode uses the indexed FCCOB register to provide a command code and its relevant parameters to the Memory Controller. The user first sets up all required FCCOB fields and then initiates the command's execution by writing a 1 to the CCIF bit in the FSTAT register (a 1 written by the user clears the CCIF command completion flag to 0). When the user clears the CCIF bit in the FSTAT register all FCCOB parameter fields are locked and cannot be changed by the user until the command completes (as evidenced by the Memory Controller returning CCIF to 1). Some commands return information to the FCCOB register array.

The generic format for the FCCOB parameter fields in NVM command mode is shown in Table 27-26. The return values are available for reading after the CCIF flag in the FSTAT register has been returned to 1 by the Memory Controller. Writes to the unimplemented parameter fields (CCOBIX = 110 and CCOBIX = 111) are ignored with reads from these fields returning 0x0000.

Table 27-26 shows the generic Flash command format. The high byte of the first word in the CCOB array contains the command code, followed by the parameters for this specific Flash command. For details on the FCCOB settings required by each command, see the Flash command descriptions in Section 27.4.2.

Table 27-26. FCCOB - NVM Command Mode (Typical Usage)

CCOBIX[2:0]	Byte	FCCOB Parameter Fields (NVM Command Mode)
000	HI	FCMD[7:0] defining Flash command
	LO	0, Global address [22:16]
001	HI	Global address [15:8]
	LO	Global address [7:0]
010	HI	Data 0 [15:8]
	LO	Data 0 [7:0]

Table 27-28. FECCR Index=000 Bit Descriptions

Field	Description
15:8 PAR[7:0]	**ECC Parity Bits** — Contains the 8 parity bits from the 72 bit wide P-Flash data word or the 6 parity bits, allocated to PAR[5:0], from the 22 bit wide D-Flash word with PAR[7:6]=00.
7 XBUS01	**Bus Source Identifier** — The XBUS01 bit determines whether the ECC error was caused by a read access from the CPU or XGATE. 0 ECC Error happened on the CPU access 1 ECC Error happened on the XGATE access
6–0 GADDR[22:16]	**Global Address** — The GADDR[22:16] field contains the upper seven bits of the global address having caused the error.

The P-Flash word addressed by ECCRIX = 001 contains the lower 16 bits of the global address. The following four words addressed by ECCRIX = 010 to 101 contain the 64-bit wide data phrase. The four data words and the parity byte are the uncorrected data read from the P-Flash block.

The D-Flash word addressed by ECCRIX = 001 contains the lower 16 bits of the global address. The uncorrected 16-bit data word is addressed by ECCRIX = 010.

27.3.2.14 Flash Option Register (FOPT)

The FOPT register is the Flash option register.

Offset Module Base + 0x0010

	7	6	5	4	3	2	1	0
R	NV[7:0]							
W								
Reset	F	F	F	F	F	F	F	F

= Unimplemented or Reserved

Figure 27-22. Flash Option Register (FOPT)

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

During the reset sequence, the FOPT register is loaded from the Flash nonvolatile byte in the Flash configuration field at global address 0x7F_FF0E located in P-Flash memory (see Table 27-3) as indicated by reset condition F in Figure 27-22. If a double bit fault is detected while reading the P-Flash phrase containing the Flash nonvolatile byte during the reset sequence, all bits in the FOPT register will be set.

Table 27-29. FOPT Field Descriptions

Field	Description
7–0 NV[7:0]	**Nonvolatile Bits** — The NV[7:0] bits are available as nonvolatile bits. Refer to the device user guide for proper use of the NV bits.

27.3.2.15 Flash Reserved0 Register (FRSV0)

This Flash register is reserved for factory testing.

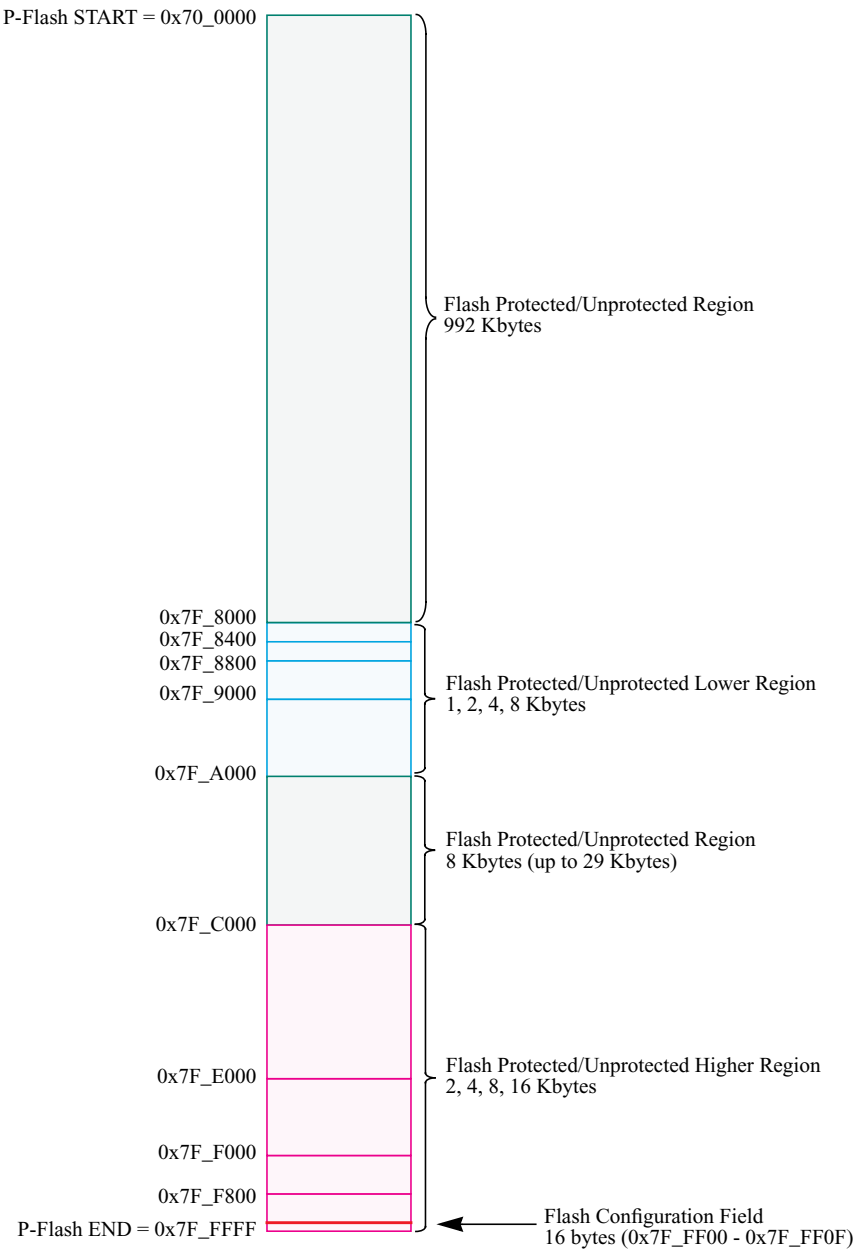


Figure 29-2. P-Flash Memory Map

Table 29-14. FECCRIX Field Descriptions

Field	Description
2-0 ECCRIX[2:0]	**ECC Error Register Index**— The ECCRIX bits are used to select which word of the FECCR register array is being read. See Section 29.3.2.13, “Flash ECC Error Results Register (FECCR),” for more details.

29.3.2.5 Flash Configuration Register (FCNFG)

The FCNFG register enables the Flash command complete interrupt and forces ECC faults on Flash array read access from the CPU or XGATE.

Offset Module Base + 0x0004

	7	6	5	4	3	2	1	0
R	CCIE	0	0	IGNSF	0	0	FDFD	FSFD
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 29-9. Flash Configuration Register (FCNFG)

CCIE, IGNSF, FDFD, and FSFD bits are readable and writable while remaining bits read 0 and are not writable.

Table 29-15. FCNFG Field Descriptions

Field	Description
7 CCIE	**Command Complete Interrupt Enable** — The CCIE bit controls interrupt generation when a Flash command has completed. 0 Command complete interrupt disabled 1 An interrupt will be requested whenever the CCIF flag in the FSTAT register is set (see Section 29.3.2.7)
4 IGNSF	**Ignore Single Bit Fault** — The IGNSF controls single bit fault reporting in the FERSTAT register (see Section 29.3.2.8). 0 All single bit faults detected during array reads are reported 1 Single bit faults detected during array reads are not reported and the single bit fault interrupt will not be generated

In Table A-28 the timing characteristics for master mode are listed.

Table A-28. SPI Master Mode Timing Characteristics

Num	C	Characteristic	Symbol	Min	Typ	Max	Unit
1	D	SCK frequency	f_{sck}	1/2048	—	1/2[1]	f_{bus}
1	D	SCK period	t_{sck}	2^1	—	2048	t_{bus}
2	D	Enable lead time	t_{lead}	—	1/2	—	t_{sck}
3	D	Enable lag time	t_{lag}	—	1/2	—	t_{sck}
4	D	Clock (SCK) high or low time	t_{wsck}	—	1/2	—	t_{sck}
5	D	Data setup time (inputs)	t_{su}	8	—	—	ns
6	D	Data hold time (inputs)	t_{hi}	8	—	—	ns
9	D	Data valid after SCK edge	t_{vsck}	—	—	15	ns
10	D	Data valid after $\overline{SS}$ fall (CPHA = 0)	t_{vss}	—	—	15	ns
11	D	Data hold time (outputs)	t_{ho}	0	—	—	ns
12	D	Rise and fall time inputs	t_{rfi}	—	—	8	ns
13	D	Rise and fall time outputs	t_{rfo}	—	—	8	ns

1. See **Figure A-9.**

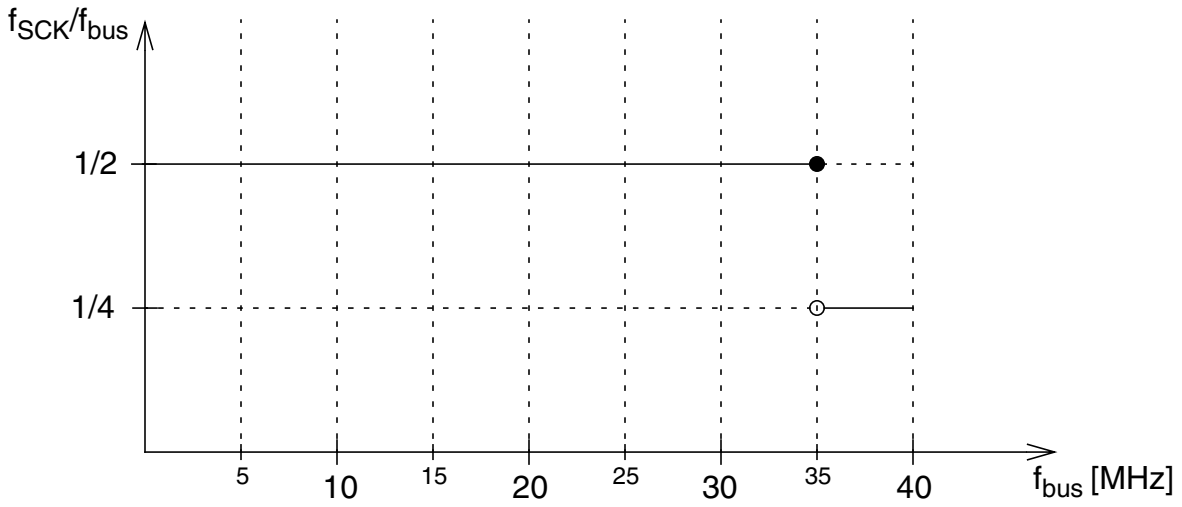


Figure A-9. Derating of maximum f_{SCK} to f_{bus} ratio in Master Mode

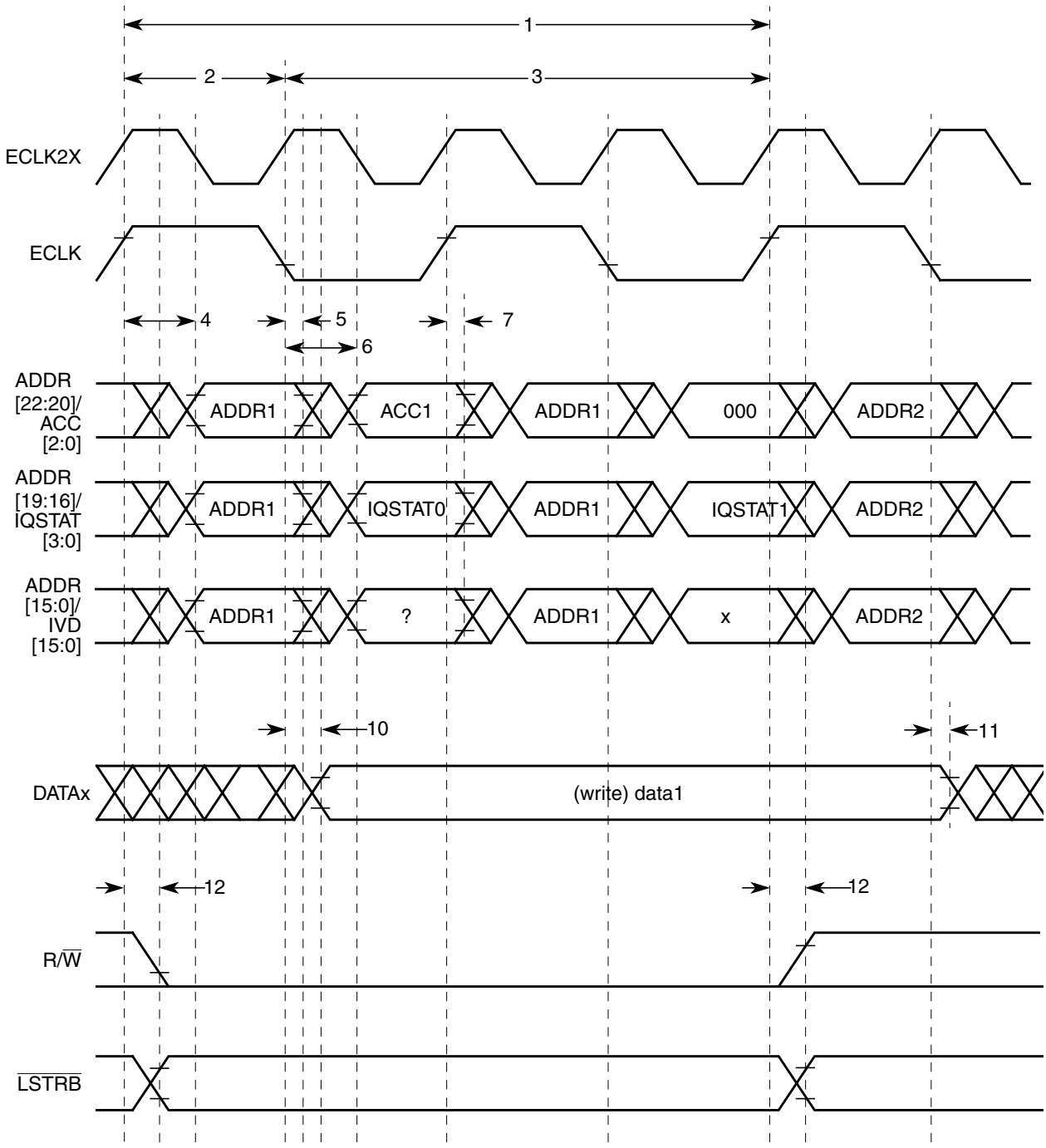


Figure A-17. Example 2b: Emulation Expanded Mode Ò Write with 1 Stretch Cycle

0x02C0–0x02EF Analog-to-Digital Converter 12-Bit 16-Channel (ATD0) Map (continued)

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x02ED	ATD0DR14L	R	Bit7	Bit6	0	0	0	0	0	0
		W								
0x02EE	ATD0DR15H	R	Bit15	14	13	12	11	10	9	Bit8
		W								
0x02EF	ATD0DR15L	R	Bit7	Bit6	0	0	0	0	0	0
		W								