



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	60
Program Memory Size	64KB (64K 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 ~ 125°C (TA)
Mounting Type	Surface Mount
Package / Case	80-QFP
Supplier Device Package	80-QFP (14x14)
Purchase URL	https://www.e-xfl.com/pro/item?MUrl=&PartUrl=mc9s12gc64mfue

2.3.2 Register Descriptions

Table 2-2 summarizes the effect on the various configuration bits — data direction (DDR), input/output level (I/O), reduced drive (RDR), pull enable (PE), pull select (PS), and interrupt enable (IE) for the ports. The configuration bit PS is used for two purposes:

1. Configure the sensitive interrupt edge (rising or falling), if interrupt is enabled.
2. Select either a pull-up or pull-down device if PE is active.

Table 2-2. Pin Configuration Summary

DDR	IO	RDR	PE	PS	IE ⁽¹⁾	Function	Pull Device	Interrupt
0	X	X	0	X	0	Input	Disabled	Disabled
0	X	X	1	0	0	Input	Pull up	Disabled
0	X	X	1	1	0	Input	Pull down	Disabled
0	X	X	0	0	1	Input	Disabled	Falling edge
0	X	X	0	1	1	Input	Disabled	Rising edge
0	X	X	1	0	1	Input	Pull up	Falling edge
0	X	X	1	1	1	Input	Pull down	rising edge
1	0	0	X	X	0	Output, full drive to 0	Disabled	Disabled
1	1	0	X	X	0	Output, full drive to 1	Disabled	Disabled
1	0	1	X	X	0	Output, reduced drive to 0	Disabled	Disabled
1	1	1	X	X	0	Output, reduced drive to 1	Disabled	Disabled
1	0	0	X	0	1	Output, full drive to 0	Disabled	Falling edge
1	1	0	X	1	1	Output, full drive to 1	Disabled	Rising edge
1	0	1	X	0	1	Output, reduced drive to 0	Disabled	Falling edge
1	1	1	X	1	1	Output, reduced drive to 1	Disabled	Rising edge

1. Applicable only on ports P and J.

NOTE

All bits of all registers in this module are completely synchronous to internal clocks during a register read.

These register locations are not used (reserved). All unused registers and bits in this block return logic 0s when read. Writes to these registers have no effect.

These registers are not in the on-chip map in special peripheral mode.

4.3.2.6 Port E Data Register (PORTE)

Module Base + 0x0008

Starting address location affected by INITRG register setting.

	7	6	5	4	3	2	1	0
R	Bit 7	6	5	4	3	2	Bit 1	Bit 0
W								
Reset	0	0	0	0	0	0	u	u
Alternate Pin Function	NOACC	MODB or IPIPE1 or CLKTO	MODA or IPIPE0	ECLK	$\overline{\text{LSTRB}}$ or $\overline{\text{TAGLO}}$	R/ $\overline{\text{W}}$	$\overline{\text{IRQ}}$	$\overline{\text{XIRQ}}$

= Unimplemented or Reserved
 u = Unaffected by reset

Figure 4-10. Port E Data Register (PORTE)

Read: Anytime when register is in the map

Write: Anytime when register is in the map

Port E is associated with external bus control signals and interrupt inputs. These include mode select (MODB/IPIPE1, MODA/IPIPE0), E clock, size ($\overline{\text{LSTRB}}$ / $\overline{\text{TAGLO}}$), read/write (R/ $\overline{\text{W}}$), $\overline{\text{IRQ}}$, and $\overline{\text{XIRQ}}$. When not used for one of these specific functions, port E pins 7:2 can be used as general-purpose I/O and pins 1:0 can be used as general-purpose input. The port E assignment register (PEAR) selects the function of each pin and DDRE determines whether each pin is an input or output when it is configured to be general-purpose I/O. DDRE also determines the source of data for a read of PORTE.

Some of these pins have software selectable pull resistors. $\overline{\text{IRQ}}$ and $\overline{\text{XIRQ}}$ can only be pulled up whereas the polarity of the PE7, PE4, PE3, and PE2 pull resistors are determined by chip integration. Please refer to the device overview chapter (Signal Property Summary) to determine the polarity of these resistors. A single control bit enables the pull devices for all of these pins when they are configured as inputs.

This register is not in the on-chip map in special peripheral mode or in expanded modes when the EME bit is set. Therefore, these accesses will be echoed externally.

NOTE

It is unwise to write PORTE and DDRE as a word access. If you are changing port E pins from being inputs to outputs, the data may have extra transitions during the write. It is best to initialize PORTE before enabling as outputs.

4.3.2.15 Port K Data Register (PORTK)

Module Base + 0x0032

Starting address location affected by INITRG register setting.

	7	6	5	4	3	2	1	0
R	Bit 7	6	5	4	3	2	1	Bit 0
W								
Reset	0	0	0	0	0	0	0	0
Alternate Pin Function	$\overline{\text{ECS}}$	$\overline{\text{XCS}}$	XAB19	XAB18	XAB17	XAB16	XAB15	XAB14

Figure 4-19. Port K Data Register (PORTK)

Read: Anytime

Write: Anytime

This port is associated with the internal memory expansion emulation pins. When the port is not enabled to emulate the internal memory expansion, the port pins are used as general-purpose I/O. When port K is operating as a general-purpose I/O port, DDRK determines the primary direction for each port K pin. A 1 causes the associated port pin to be an output and a 0 causes the associated pin to be a high-impedance input. The value in a DDR bit also affects the source of data for reads of the corresponding PORTK register. If the DDR bit is 0 (input) the buffered pin input is read. If the DDR bit is 1 (output) the output of the port data register is read.

This register is not in the map in peripheral or expanded modes while the EMK control bit in MODE register is set. Therefore, these accesses will be echoed externally.

When inputs, these pins can be selected to be high impedance or pulled up, based upon the state of the PUPKE bit in the PUCR register.

Table 4-13. PORTK Field Descriptions

Field	Description
7 Port K, Bit 7	Port K, Bit 7 — This bit is used as an emulation chip select signal for the emulation of the internal memory expansion, or as general-purpose I/O, depending upon the state of the EMK bit in the MODE register. While this bit is used as a chip select, the external bit will return to its de-asserted state (V_{DD}) for approximately 1/4 cycle just after the negative edge of ECLK, unless the external access is stretched and ECLK is free-running (ESTR bit in EBICTL = 0). See the MMC block description chapter for additional details on when this signal will be active.
6 Port K, Bit 6	Port K, Bit 6 — This bit is used as an external chip select signal for most external accesses that are not selected by $\overline{\text{ECS}}$ (see the MMC block description chapter for more details), depending upon the state of the EMK bit in the MODE register. While this bit is used as a chip select, the external pin will return to its de-asserted state (V_{DD}) for approximately 1/4 cycle just after the negative edge of ECLK, unless the external access is stretched and ECLK is free-running (ESTR bit in EBICTL = 0).
5:0 Port K, Bits 5:0	Port K, Bits 5:0 — These six bits are used to determine which FLASH/ROM or external memory array page is being accessed. They can be viewed as expanded addresses XAB19–XAB14 of the 20-bit address used to access up to 1M byte internal FLASH/ROM or external memory array. Alternatively, these bits can be used for general-purpose I/O depending upon the state of the EMK bit in the MODE register.

1. The DBG module is designed for backwards compatibility to existing BKP modules. Register and bit names have changed from the BKP module. This column shows the DBG register name, as well as the BKP register name for reference.
2. Comparator C can be used to enhance the BKP mode by providing a third breakpoint.

7.3.2.1 Debug Control Register 1 (DBGC1)

NOTE

All bits are used in DBG mode only.

Module Base + 0x0020

Starting address location affected by INITRG register setting.

	7	6	5	4	3	2	1	0
R	DBGEN	ARM	TRGSEL	BEGIN	DBGBRK	0	CAPMOD	
W								
Reset	0	0	0	0	0	0	0	0

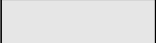
 = Unimplemented or Reserved

Figure 7-4. Debug Control Register (DBGC1)

NOTE

This register cannot be written if BKP mode is enabled (BKABEN in DBGC2 is set).

Table 7-3. DBGC1 Field Descriptions

Field	Description
7 DBGEN	DBG Mode Enable Bit — The DBGEN bit enables the DBG module for use in DBG mode. This bit cannot be set if the MCU is in secure mode. 0 DBG mode disabled 1 DBG mode enabled
6 ARM	Arm Bit — The ARM bit controls whether the debugger is comparing and storing data in the trace buffer. See Section 7.4.2.4, “Arming the DBG Module,” for more information. 0 Debugger unarmed 1 Debugger armed Note: This bit cannot be set if the DBGEN bit is not also being set at the same time. For example, a write of 01 to DBGEN[7:6] will be interpreted as a write of 00.
5 TRGSEL	Trigger Selection Bit — The TRGSEL bit controls the triggering condition for comparators A and B in DBG mode. It serves essentially the same function as the TAGAB bit in the DBGC2 register does in BKP mode. See Section 7.4.2.1.2, “Trigger Selection,” for more information. TRGSEL may also determine the type of breakpoint based on comparator A and B if enabled in DBG mode (DBGBRK = 1). Please refer to Section 7.4.3.1, “Breakpoint Based on Comparator A and B.” 0 Trigger on any compare address match 1 Trigger before opcode at compare address gets executed (tagged-type)
4 BEGIN	Begin/End Trigger Bit — The BEGIN bit controls whether the trigger begins or ends storing of data in the trace buffer. See Section 7.4.2.8.1, “Storing with Begin-Trigger,” and Section 7.4.2.8.2, “Storing with End-Trigger,” for more details. 0 Trigger at end of stored data 1 Trigger before storing data

9.3.2.2 CRG Reference Divider Register (REFDV)

The REFDV register provides a finer granularity for the PLL multiplier steps. The count in the reference divider divides OSCCLK frequency by REFDV + 1.

Module Base + 0x0001

	7	6	5	4	3	2	1	0
R	0	0	0	0	REFDV3	REFDV2	REFDV1	REFDV0
W								
Reset	0	0	0	0	0	0	0	0


 = Unimplemented or Reserved

Figure 9-5. CRG Reference Divider Register (REFDV)

Read: anytime

Write: anytime except when PLLSEL = 1

NOTE

Write to this register initializes the lock detector bit and the track detector bit.

9.3.2.3 Reserved Register (CTFLG)

This register is reserved for factory testing of the CRGV4 module and is not available in normal modes.

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

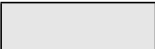
 = Unimplemented or Reserved

Figure 9-6. CRG Reserved Register (CTFLG)

Read: always reads 0x0000 in normal modes

Write: unimplemented in normal modes

NOTE

Writing to this register when in special mode can alter the CRGV4 functionality.

Table 10-1. CANCTL0 Register Field Descriptions (continued)

Field	Description
1 SLPRQ ⁽⁵⁾	<p>Sleep Mode Request — This bit requests the MSCAN to enter sleep mode, which is an internal power saving mode (see Section 10.4.5.4, “MSCAN Sleep Mode”). The sleep mode request is serviced when the CAN bus is idle, i.e., the module is not receiving a message and all transmit buffers are empty. The module indicates entry to sleep mode by setting SLPK = 1 (see Section 10.3.2.2, “MSCAN Control Register 1 (CANCTL1)”). SLPRQ cannot be set while the WUPE flag is set (see Section 10.3.2.5, “MSCAN Receiver Flag Register (CANRFLG)”). Sleep mode will be active until SLPRQ is cleared by the CPU or, depending on the setting of WUPE, the MSCAN detects activity on the CAN bus and clears SLPRQ itself.</p> <p>0 Running — The MSCAN functions normally</p> <p>1 Sleep mode request — The MSCAN enters sleep mode when CAN bus idle</p>
0 INITRQ ^{(6),(7)}	<p>Initialization Mode Request — When this bit is set by the CPU, the MSCAN skips to initialization mode (see Section 10.4.5.5, “MSCAN Initialization Mode”). Any ongoing transmission or reception is aborted and synchronization to the CAN bus is lost. The module indicates entry to initialization mode by setting INITAK = 1 (Section 10.3.2.2, “MSCAN Control Register 1 (CANCTL1)”).</p> <p>The following registers enter their hard reset state and restore their default values: CANCTL0⁽⁸⁾, CANRFLG⁽⁹⁾, CANRIER⁽¹⁰⁾, CANTFLG, CANTIER, CANTARQ, CANTAOK, and CANTBSEL.</p> <p>The registers CANCTL1, CANBTR0, CANBTR1, CANIDAC, CANIDAR0-7, and CANIDMR0-7 can only be written by the CPU when the MSCAN is in initialization mode (INITRQ = 1 and INITAK = 1). The values of the error counters are not affected by initialization mode.</p> <p>When this bit is cleared by the CPU, the MSCAN restarts and then tries to synchronize to the CAN bus. If the MSCAN is not in bus-off state, it synchronizes after 11 consecutive recessive bits on the CAN bus; if the MSCAN is in bus-off state, it continues to wait for 128 occurrences of 11 consecutive recessive bits.</p> <p>Writing to other bits in CANCTL0, CANRFLG, CANRIER, CANTFLG, or CANTIER must be done only after initialization mode is exited, which is INITRQ = 0 and INITAK = 0.</p> <p>0 Normal operation</p> <p>1 MSCAN in initialization mode</p>

1. The MSCAN must be in normal mode for this bit to become set.
2. See the Bosch CAN 2.0A/B specification for a detailed definition of transmitter and receiver states.
3. In order to protect from accidentally violating the CAN protocol, the TXCAN pin is immediately forced to a recessive state when the CPU enters wait (CSWAI = 1) or stop mode (see [Section 10.4.5.2, “Operation in Wait Mode”](#) and [Section 10.4.5.3, “Operation in Stop Mode”](#)).
4. The CPU has to make sure that the WUPE register and the WUPE wake-up interrupt enable register (see [Section 10.3.2.6, “MSCAN Receiver Interrupt Enable Register \(CANRIER\)”](#)) is enabled, if the recovery mechanism from stop or wait is required.
5. The CPU cannot clear SLPRQ before the MSCAN has entered sleep mode (SLPRQ = 1 and SLPK = 1).
6. The CPU cannot clear INITRQ before the MSCAN has entered initialization mode (INITRQ = 1 and INITAK = 1).
7. In order to protect from accidentally violating the CAN protocol, the TXCAN pin is immediately forced to a recessive state when the initialization mode is requested by the CPU. Thus, the recommended procedure is to bring the MSCAN into sleep mode (SLPRQ = 1 and SLPK = 1) before requesting initialization mode.
8. Not including WUPE, INITRQ, and SLPRQ.
9. TSTAT1 and TSTAT0 are not affected by initialization mode.
10. RSTAT1 and RSTAT0 are not affected by initialization mode.

10.3.2.2 MSCAN Control Register 1 (CANCTL1)

The CANCTL1 register provides various control bits and handshake status information of the MSCAN module as described below.

Module Base + 0x00X3

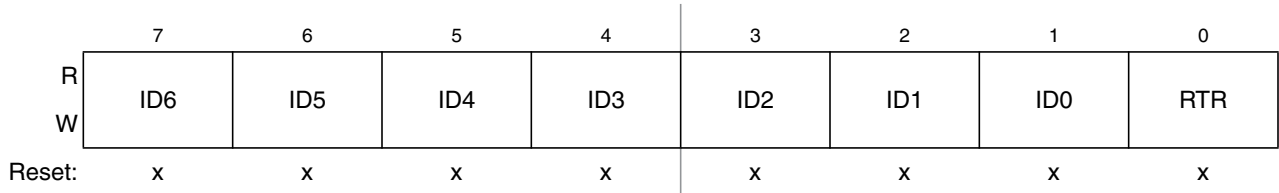


Figure 10-28. Identifier Register 3 (IDR3) — Extended Identifier Mapping

Table 10-27. IDR3 Register Field Descriptions — Extended

Field	Description
7:1 ID[6:0]	Extended Format Identifier — The identifiers consist of 29 bits (ID[28:0]) for the extended format. ID28 is the most significant bit and is transmitted first on the CAN bus during the arbitration procedure. The priority of an identifier is defined to be highest for the smallest binary number.
0 RTR	Remote Transmission Request — This flag reflects the status of the remote transmission request bit in the CAN frame. In the case of a receive buffer, it indicates the status of the received frame and supports the transmission of an answering frame in software. In the case of a transmit buffer, this flag defines the setting of the RTR bit to be sent. 0 Data frame 1 Remote frame

10.3.3.1.2 IDR0–IDR3 for Standard Identifier Mapping

Module Base + 0x00X0

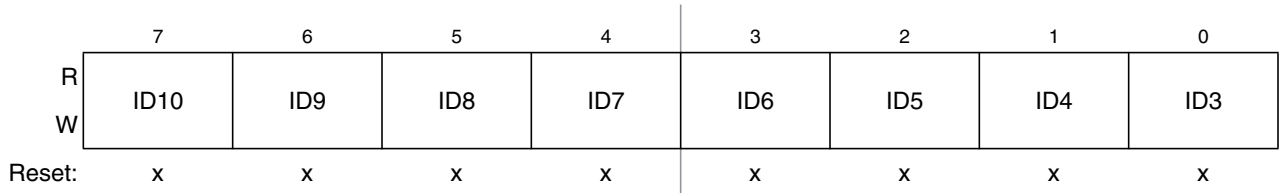


Figure 10-29. Identifier Register 0 — Standard Mapping

Table 10-28. IDR0 Register Field Descriptions — Standard

Field	Description
7:0 ID[10:3]	Standard Format Identifier — The identifiers consist of 11 bits (ID[10:0]) for the standard format. ID10 is the most significant bit and is transmitted first on the CAN bus during the arbitration procedure. The priority of an identifier is defined to be highest for the smallest binary number. See also ID bits in Table 10-29 .

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 10.3.2.2, “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 10.4.7.5, “Error Interrupt”](#)). The MSCAN remains able to transmit messages while the receiver FIFO 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.

10.4.3 Identifier Acceptance Filter

The MSCAN identifier acceptance registers (see [Section 10.3.2.12, “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 10.3.2.17, “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 10.3.2.12, “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 (see Bosch CAN 2.0A/B protocol specification):

- 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 messages¹. This mode implements two filters for a full length CAN 2.0B compliant extended identifier. [Figure 10-39](#) shows how the first 32-bit filter bank (CANIDAR0–CANIDAR3, CANIDMR0–CANIDMR3) produces a filter 0 hit. Similarly, the second filter bank (CANIDAR4–CANIDAR7, CANIDMR4–CANIDMR7) produces a filter 1 hit.
- Four identifier acceptance filters, each to be applied to

1. Although this mode can be used for standard identifiers, it is recommended to use the four or eight identifier acceptance filters for standard identifiers

- The channel is disabled

In this way, the output of the PWM will always be either the old waveform or the new waveform, not some variation in between. If the channel is not enabled, then writes to the period register will go directly to the latches as well as the buffer.

NOTE

Reads of this register return the most recent value written. Reads do not necessarily return the value of the currently active period due to the double buffering scheme.

Reference [Section 12.4.2.3, “PWM Period and Duty,”](#) for more information.

To calculate the output period, take the selected clock source period for the channel of interest (A, B, SA, or SB) and multiply it by the value in the period register for that channel:

- Left aligned output (CAEx = 0)
- PWMx period = channel clock period * PWMPERx center aligned output (CAEx = 1)
- PWMx period = channel clock period * (2 * PWMPERx)

For boundary case programming values, please refer to [Section 12.4.2.8, “PWM Boundary Cases.”](#)

Module Base + 0x0012

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

Figure 12-21. PWM Channel Period Registers (PWMPER0)

Module Base + 0x0013

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

Figure 12-22. PWM Channel Period Registers (PWMPER1)

Module Base + 0x0014

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

Figure 12-23. PWM Channel Period Registers (PWMPER2)

12.4.2.4 PWM Timer Counters

Each channel has a dedicated 8-bit up/down counter which runs at the rate of the selected clock source (reference [Figure 12-34](#) for the available clock sources and rates). The counter compares to two registers, a duty register and a period register as shown in [Figure 12-35](#). When the PWM counter matches the duty register the output flip-flop changes state causing the PWM waveform to also change state. A match between the PWM counter and the period register behaves differently depending on what output mode is selected as shown in [Figure 12-35](#) and described in [Section 12.4.2.5, “Left Aligned Outputs,”](#) and [Section 12.4.2.6, “Center Aligned Outputs.”](#)

Each channel counter can be read at anytime without affecting the count or the operation of the PWM channel.

Any value written to the counter causes the counter to reset to 0x0000, the counter direction to be set to up, the immediate load of both duty and period registers with values from the buffers, and the output to change according to the polarity bit. When the channel is disabled ($PWMEx = 0$), the counter stops. When a channel becomes enabled ($PWMEx = 1$), the associated PWM counter continues from the count in the $PWMCNTx$ register. This allows the waveform to resume when the channel is re-enabled. When the channel is disabled, writing 0 to the period register will cause the counter to reset on the next selected clock.

NOTE

If the user wants to start a new “clean” PWM waveform without any “history” from the old waveform, the user must write to channel counter ($PWMCNTx$) prior to enabling the PWM channel ($PWMEx = 1$).

Generally, writes to the counter are done prior to enabling a channel to start from a known state. However, writing a counter can also be done while the PWM channel is enabled (counting). The effect is similar to writing the counter when the channel is disabled except that the new period is started immediately with the output set according to the polarity bit.

NOTE

Writing to the counter while the channel is enabled can cause an irregular PWM cycle to occur.

The counter is cleared at the end of the effective period (see [Section 12.4.2.5, “Left Aligned Outputs,”](#) and [Section 12.4.2.6, “Center Aligned Outputs,”](#) for more details).

Table 12-11. PWM Timer Counter Conditions

Counter Clears (0x0000)	Counter Counts	Counter Stops
When $PWMCNTx$ register written to any value	When PWM channel is enabled ($PWMEx = 1$). Counts from last value in $PWMCNTx$.	When PWM channel is disabled ($PWMEx = 0$)
Effective period ends		

13.4.5 Single-Wire Operation

Normally, the SCI uses two pins for transmitting and receiving. In single-wire operation, the RXD pin is disconnected from the SCI. The SCI uses the TXD pin for both receiving and transmitting.

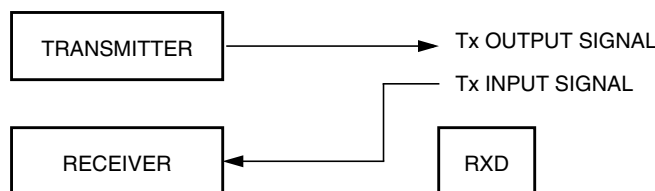


Figure 13-22. Single-Wire Operation (LOOPS = 1, RSRC = 1)

Enable single-wire operation by setting the LOOPS bit and the receiver source bit, RSRC, in SCI control register 1 (SCICR1). Setting the LOOPS bit disables the path from the **Rx Input** signal to the receiver. Setting the RSRC bit connects the receiver input to the output of the TXD pin driver. Both the transmitter and receiver must be enabled (TE = 1 and RE = 1). The TXDIR bit (SCISR2[1]) determines whether the TXD pin is going to be used as an input (TXDIR = 0) or an output (TXDIR = 1) in this mode of operation.

13.4.6 Loop Operation

In loop operation the transmitter output goes to the receiver input. The **Rx Input** signal is disconnected from the SCI

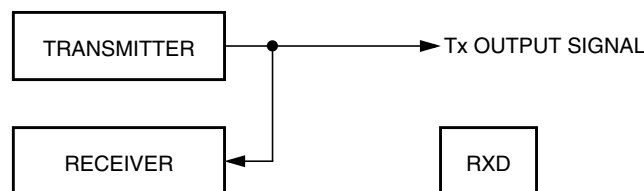


Figure 13-23. Loop Operation (LOOPS = 1, RSRC = 0)

Enable loop operation by setting the LOOPS bit and clearing the RSRC bit in SCI control register 1 (SCICR1). Setting the LOOPS bit disables the path from the **Rx Input** signal to the receiver. Clearing the RSRC bit connects the transmitter output to the receiver input. Both the transmitter and receiver must be enabled (TE = 1 and RE = 1).

13.5 Initialization Information

13.5.1 Reset Initialization

The reset state of each individual bit is listed in [Section 13.3, “Memory Map and Registers”](#) which details the registers and their bit fields. All special functions or modes which are initialized during or just following reset are described within this section.

14.3.2 Register Descriptions

This section consists of register descriptions in address order. Each description includes a standard register diagram with an associated figure number. Details of register bit and field function follow the register diagrams, in bit order.

Name		7	6	5	4	3	2	1	0
0x0000 SPICR1	R W	SPIE	SPE	SPTIE	MSTR	CPOL	CPHA	SSOE	LSBFE
0x0001 SPICR2	R W	0	0	0	MODFEN	BIDIROE	0	SPISWAI	SPC0
0x0002 SPIBR	R W	0	SPPR2	SPPR1	SPPR0	0	SPR2	SPR1	SPR0
0x0003 SPISR	R W	SPIF	0	SPTIEF	MODF	0	0	0	0
0x0004 Reserved	R W								
0x0005 SPIDR	R W	Bit 7	6	5	4	3	2	2	Bit 0
0x0006 Reserved	R W								
0x0007 Reserved	R W								

= Unimplemented or Reserved

Figure 14-2. SPI Register Summary

14.3.2.1 SPI Control Register 1 (SPICR1)

Module Base 0x0000

	7	6	5	4	3	2	1	0
R W	SPIE	SPE	SPTIE	MSTR	CPOL	CPHA	SSOE	LSBFE
Reset	0	0	0	0	0	1	0	0

Figure 14-3. SPI Control Register 1 (SPICR1)

Read: anytime

Write: anytime

15.3.2.10 Timer Interrupt Enable Register (TIE)

Module Base + 0x000C

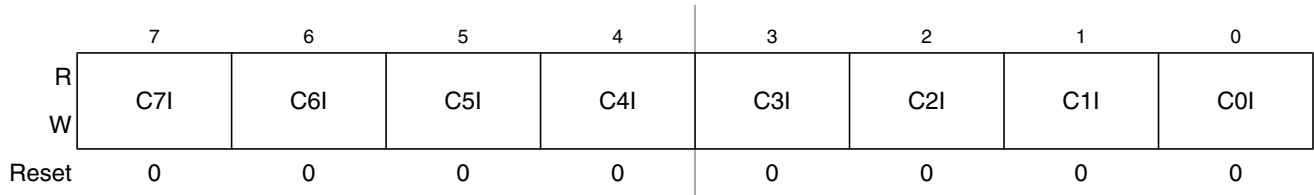


Figure 15-18. Timer Interrupt Enable Register (TIE)

Read: Anytime

Write: Anytime.

Table 15-14. TIE Field Descriptions

Field	Description
7:0 C7I:C0I	Input Capture/Output Compare “x” Interrupt Enable — 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

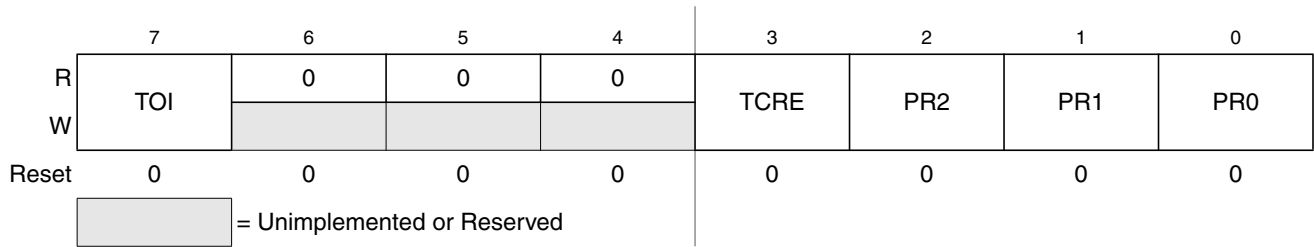


Figure 15-19. Timer System Control Register 2 (TSCR2)

Read: Anytime

Write: Anytime.

Table 19-2. Flash Array Memory Map Summary

MCU Address Range	PPAGE	Protectable Low Range	Protectable High Range	Array Relative Address ⁽¹⁾
0x0000–0x3FFF ⁽²⁾	Unpaged (0x3D)	N.A.	N.A.	0x14000–0x17FFF
0x4000–0x7FFF	Unpaged (0x3E)	0x4000–0x43FF 0x4000–0x47FF 0x4000–0x4FFF 0x4000–0x5FFF	N.A.	0x18000–0x1BFFF
0x8000–0xBFFF	0x38	N.A.	N.A.	0x00000–0x03FFF
	0x39	N.A.	N.A.	0x04000–0x07FFF
	0x3A	N.A.	N.A.	0x08000–0x0BFFF
	0x3B	N.A.	N.A.	0x0C000–0x0FFFF
	0x3C	N.A.	N.A.	0x10000–0x13FFF
	0x3D	N.A.	N.A.	0x14000–0x17FFF
	0x3E	0x8000–0x83FF 0x8000–0x87FF 0x8000–0x8FFF 0x8000–0x9FFF	N.A.	0x18000–0x1BFFF
	0x3F	N.A.	0xB800–0xBFFF 0xB000–0xBFFF 0xA000–0xBFFF 0x8000–0xBFFF	0x1C000–0x1FFFF
0xC000–0xFFFF	Unpaged (0x3F)	N.A.	0xF800–0xFFFF 0xF000–0xFFFF 0xE000–0xFFFF 0xC000–0xFFFF	0x1C000–0x1FFFF

1. Inside Flash block.

2. If allowed by MCU.

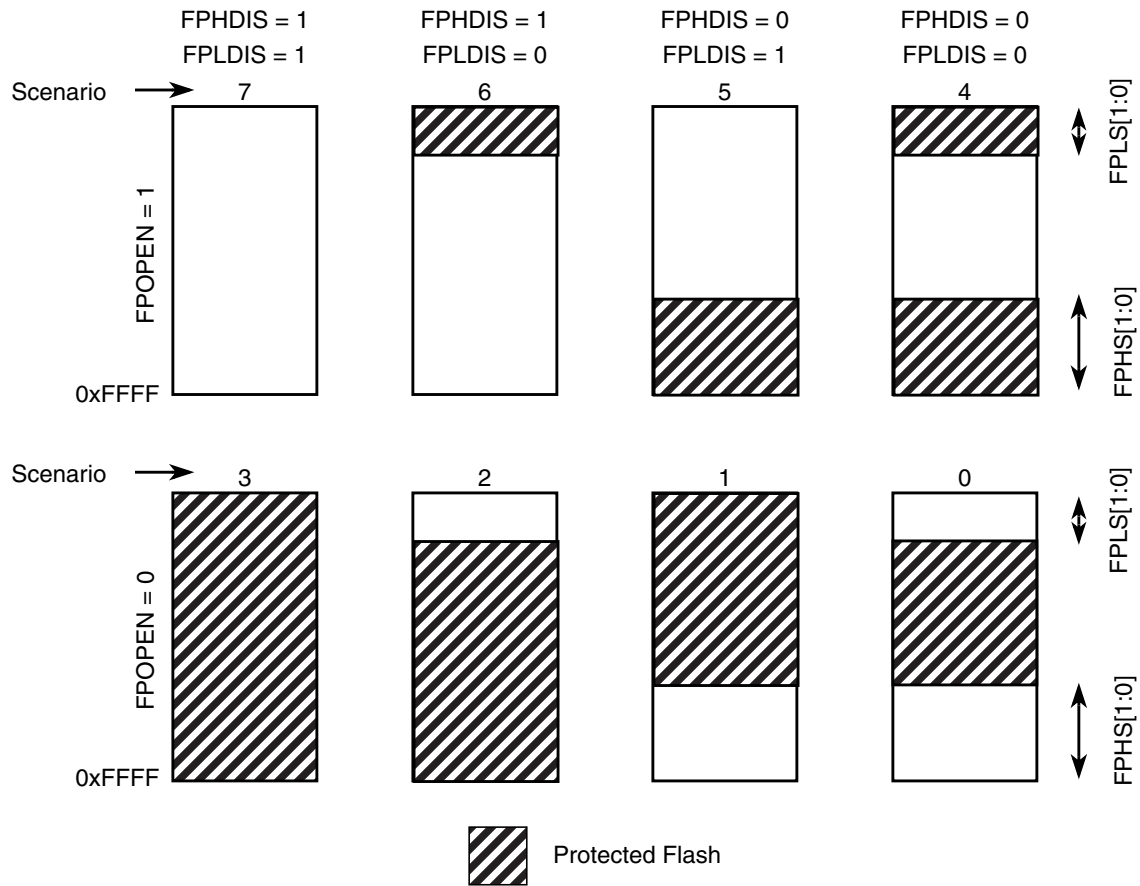


Figure 19-11. Flash Protection Scenarios

19.3.2.5.1 Flash Protection Restrictions

The general guideline is that protection can only be added, not removed. All valid transitions between Flash protection scenarios are specified in Table 19-13. Any attempt to write an invalid scenario to the FPROT register will be ignored and the FPROT register will remain unchanged. The contents of the FPROT register reflect the active protection scenario.

Table 19-13. Flash Protection Scenario Transitions

From Protection Scenario	To Protection Scenario ⁽¹⁾							
	0	1	2	3	4	5	6	7
0	X	X	X	X				
1		X		X				
2			X	X				
3				X				
4				X	X			
5			X	X	X	X		

Module Base + 0x0008

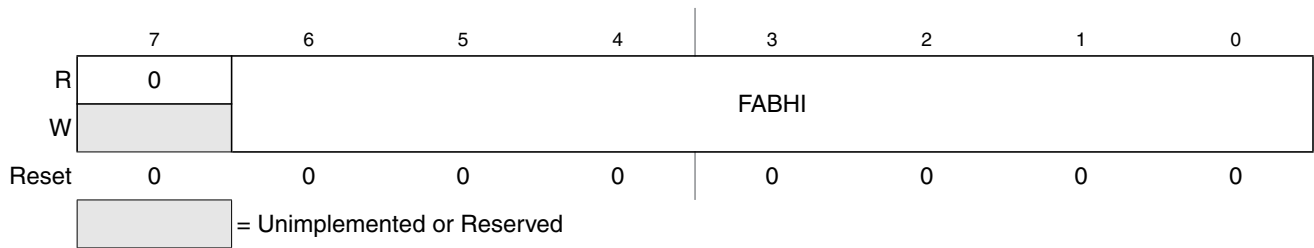


Figure 19-16. Flash Address High Register (FADDRHI)

Module Base + 0x0009

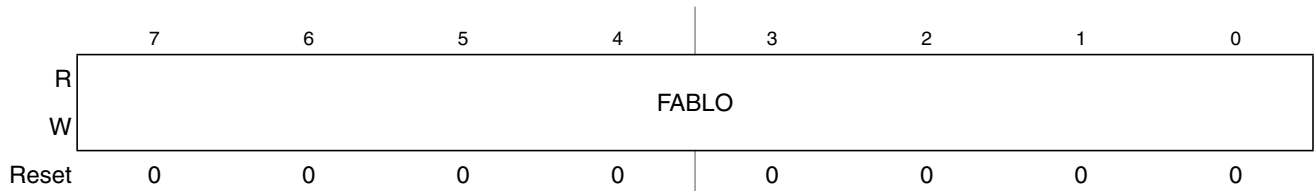


Figure 19-17. Flash Address Low Register (FADDRLO)

In normal modes, all FABHI and FABLO bits read 0 and are not writable. In special modes, the FABHI and FABLO bits are readable and writable. For sector erase, the MCU address bits [9:0] are ignored. For mass erase, any address within the Flash array is valid to start the command.

19.3.2.10 Flash Data Register (FDATA)

FDATAHI and FDATALO are the Flash data registers.

Module Base + 0x000A

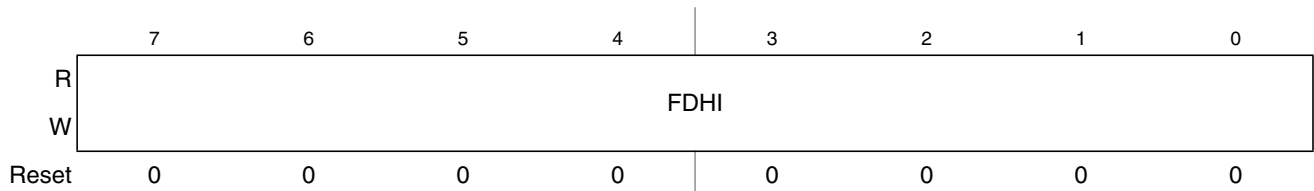


Figure 19-18. Flash Data High Register (FDATAHI)

Module Base + 0x000B

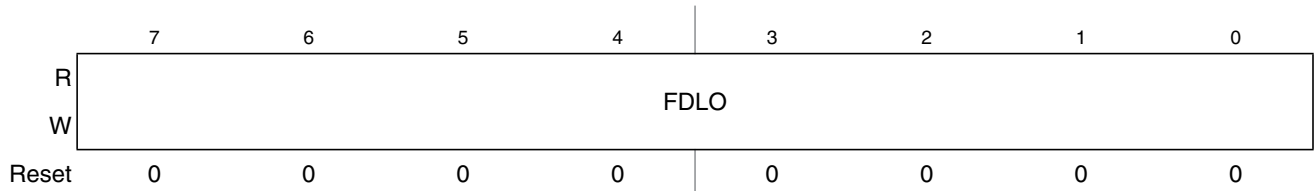


Figure 19-19. Flash Data Low Register (FDATALO)

In normal modes, all FDATAHI and FDATALO bits read 0 and are not writable. In special modes, all FDATAHI and FDATALO bits are readable and writable when writing to an address within the Flash address range.

19.4.1.3.4 Mass Erase Command

The mass erase operation will erase all addresses in a Flash array using an embedded algorithm.

An example flow to execute the mass erase operation is shown in [Figure 19-28](#). The mass erase command write sequence is as follows:

1. Write to a Flash array 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, 0x41, to the FCMD register.
3. Clear the CBEIF flag in the FSTAT register by writing a 1 to CBEIF to launch the mass erase command.

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

20.1.3 Modes of Operation

See [Section 20.4.2, “Operating Modes”](#) for a description of the Flash module operating modes. For program and erase operations, refer to [Section 20.4.1, “Flash Command Operations”](#).

20.1.4 Block Diagram

Figure 20-1Figure 20-2 shows a block diagram of the [FTS128K1FTS96K](#) module.

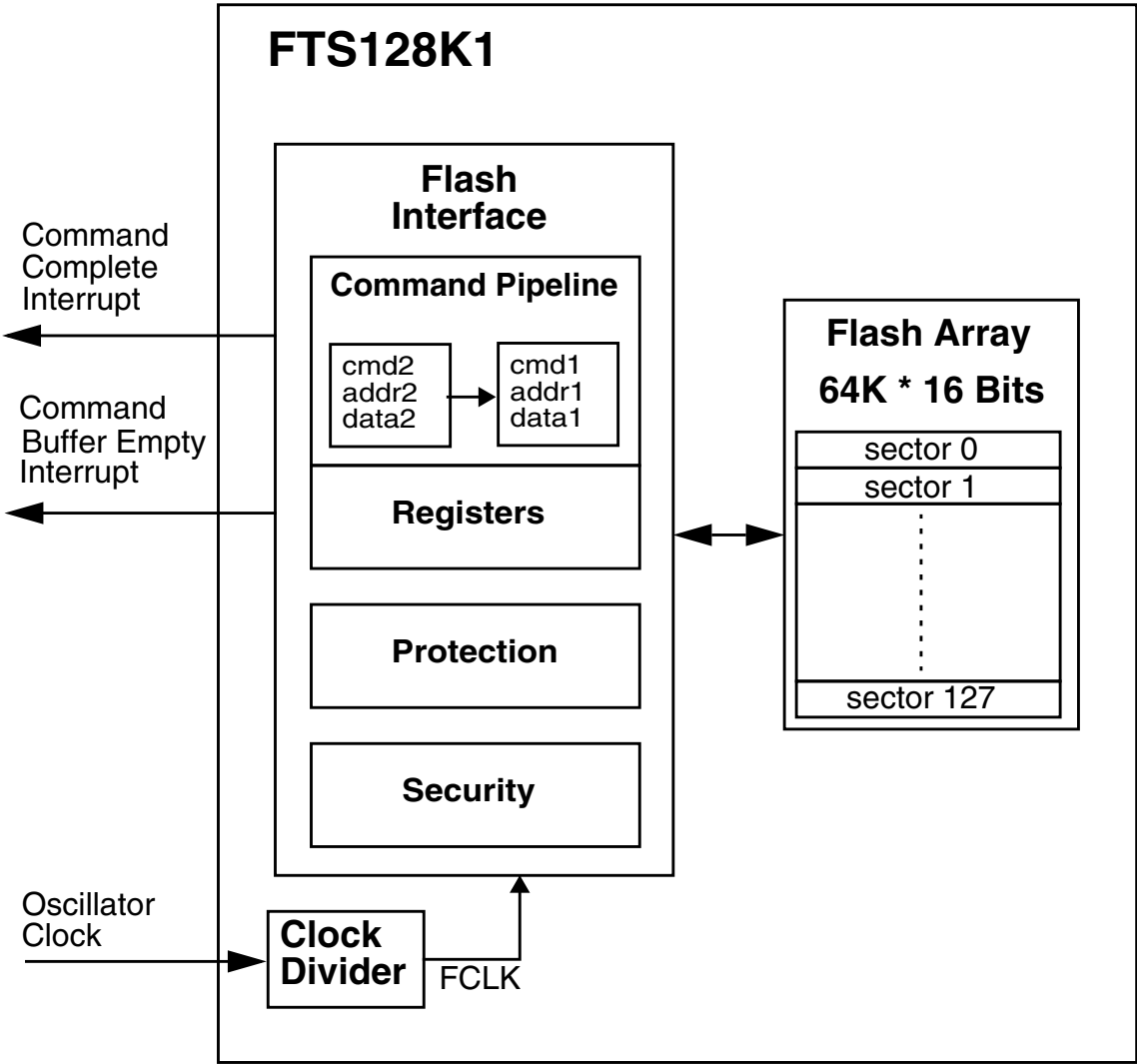


Figure 20-1. FTS128K1 Block Diagram

Table 21-13. FSTAT Field Descriptions

Field	Description
5 PVIOL	Protection Violation — The PVIOL flag indicates an attempt was made to program or erase an address in a protected Flash array memory area. 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. 0 No protection violation detected 1 Protection violation has occurred
4 ACCERR	Access Error — The ACCERR flag indicates an illegal access to the Flash array caused by either a violation of the command write sequence, issuing an illegal command (illegal combination of the CMDBx bits in the FCMD register) 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. 0 No access error detected 1 Access error has occurred
2 BLANK	Flash Array Has Been Verified as Erased — The BLANK flag indicates that an erase verify command has checked the Flash 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 write 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 the array is not erased 1 Flash array verifies as erased
1 FAIL	Flag Indicating a Failed Flash Operation — In special modes, the FAIL flag will set if the erase verify operation fails (Flash array verified as not erased). Writing a 0 to the FAIL flag has no effect on FAIL. The FAIL flag is cleared by writing a 1 to FAIL. While FAIL is set, it is not possible to launch another command. 0 Flash operation completed without error 1 Flash operation failed
0 DONE	Flag Indicating a Failed Operation is not Active — In special modes, the DONE flag will clear if a program, erase, or erase verify operation is active. 0 Flash operation is active 1 Flash operation is not active

21.3.2.7 Flash Command Register (FCMD)

The FCMD register defines the Flash commands.

Module Base + 0x0006

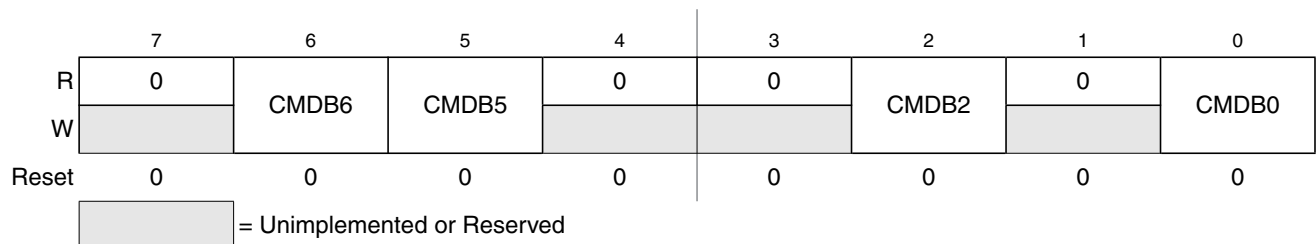


Figure 21-11. Flash Command Register (FCMD)

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

Appendix E Ordering Information

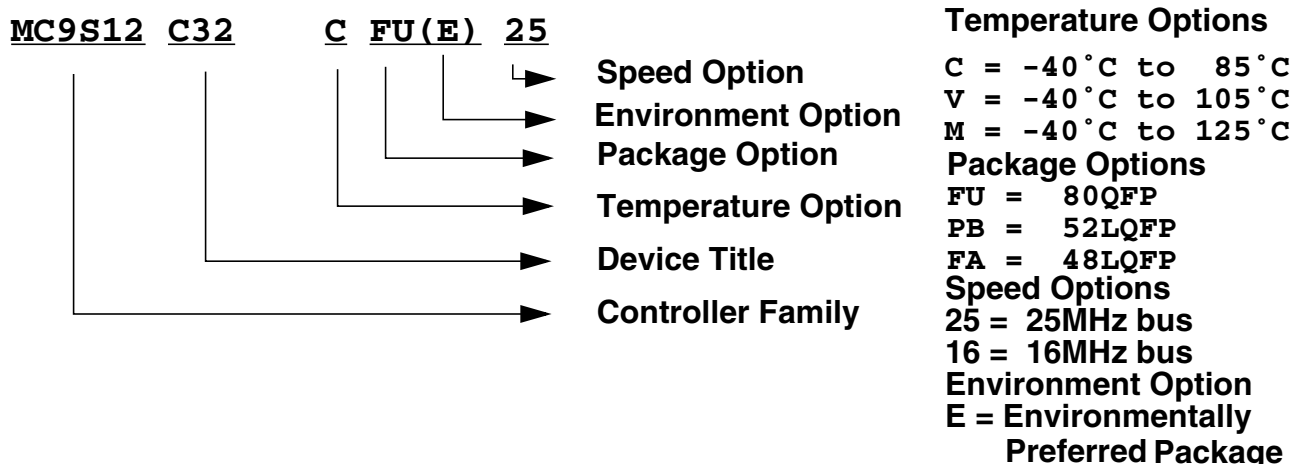


Figure E-1. Order Part number Coding

Table E-1. lists C-family part number coding based on package, speed and temperature and die options.

Table E-2. lists CG-family part number coding based on package, speed and temperature and die options.

Table E-1. MC9S12C-Family / MC9S12GC-Family Part Number Coding

Part Number	Mask ⁽¹⁾ set	Temp.	Package	Speed	Die Type	Flash	RAM	I/O ⁽²⁾ , (3)
MC9S12C128CFA	XL09S/0M66G	-40°C, 85°C	48LQFP	25MHz	C128 die	128K	4K	31
MC9S12C128CPB	XL09S/0M66G	-40°C, 85°C	52LQFP	25MHz	C128 die	128K	4K	35
MC9S12C128CFU	XL09S/0M66G	-40°C, 85°C	80QFP	25MHz	C128 die	128K	4K	60
MC9S12C128VFA	XL09S/0M66G	-40°C, 105°C	48LQFP	25MHz	C128 die	128K	4K	31
MC9S12C128VPB	XL09S/0M66G	-40°C, 105°C	52LQFP	25MHz	C128 die	128K	4K	35
MC9S12C128VFU	XL09S/0M66G	-40°C, 105°C	80QFP	25MHz	C128 die	128K	4K	60
MC9S12C128MFA	XL09S/0M66G	-40°C, 125°C	48LQFP	25MHz	C128 die	128K	4K	31
MC9S12C128MPB	XL09S/0M66G	-40°C, 125°C	52LQFP	25MHz	C128 die	128K	4K	35
MC9S12C128MFU	XL09S/0M66G	-40°C, 125°C	80QFP	25MHz	C128 die	128K	4K	60
MC9S12C96CFA	XL09S/0M66G	-40°C, 85°C	48LQFP	25MHz	C128 die	96K	4K	31
MC9S12C96CPB	XL09S/0M66G	-40°C, 85°C	52LQFP	25MHz	C128 die	96K	4K	35
MC9S12C96CFU	XL09S/0M66G	-40°C, 85°C	80QFP	25MHz	C128 die	96K	4K	60
MC9S12C96VFA	XL09S/0M66G	-40°C, 105°C	48LQFP	25MHz	C128 die	96K	4K	31
MC9S12C96VPB	XL09S/0M66G	-40°C, 105°C	52LQFP	25MHz	C128 die	96K	4K	35
MC9S12C96VFU	XL09S/0M66G	-40°C, 105°C	80QFP	25MHz	C128 die	96K	4K	60
MC9S12C96MFA	XL09S/0M66G	-40°C, 125°C	48LQFP	25MHz	C128 die	96K	4K	31
MC9S12C96MPB	XL09S/0M66G	-40°C, 125°C	52LQFP	25MHz	C128 die	96K	4K	35
MC9S12C96MFU	XL09S/0M66G	-40°C, 125°C	80QFP	25MHz	C128 die	96K	4K	60
MC9S12C64CFA	XL09S/0M66G	-40°C, 85°C	48LQFP	25MHz	C128 die	64K	4K	31
MC9S12C64CPB	XL09S/0M66G	-40°C, 85°C	52LQFP	25MHz	C128 die	64K	4K	35