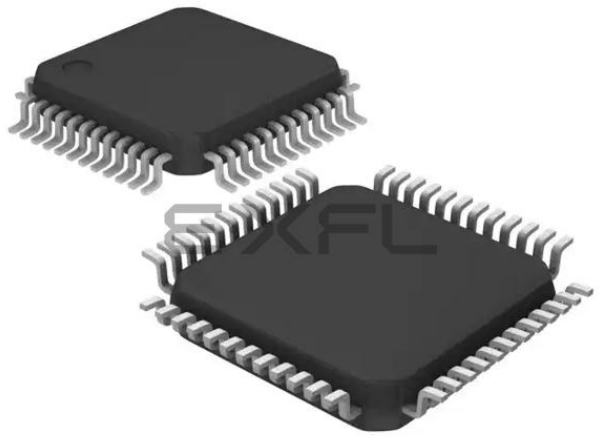


Welcome to E-XFL.COM

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	Obsolete
Core Processor	HC08
Core Size	8-Bit
Speed	8MHz
Connectivity	CANbus, SCI, SPI
Peripherals	LVD, POR, PWM
Number of I/O	37
Program Memory Size	60KB (60K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	2K x 8
Voltage - Supply (Vcc/Vdd)	3V ~ 5.5V
Data Converters	A/D 24x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 125°C (TA)
Mounting Type	Surface Mount
Package / Case	48-LQFP
Supplier Device Package	48-LQFP (7x7)
Purchase URL	https://www.e-xfl.com/product-detail/nxp-semiconductors/mc68hc908gz60mfa

Chapter 20
Development Support

Chapter 21
Electrical Specifications

Chapter 22
Ordering Information and Mechanical Specifications

Appendix A
MC68HC908GZ48

Appendix B
MC68HC908GZ32

- $FF81; FLASH-2 block protect register, FL2BPR
- $FF88; FLASH-1 control register, FL1CR

Data registers are shown in Figure 2-2. Table 2-1 is a list of vector locations.

$0000 ↓ $003F	I/O REGISTERS 64 BYTES
$0040 ↓ $043F	RAM-1 1024 BYTES
$0440 ↓ $0461	I/O REGISTERS 34 BYTES
$0462 ↓ $04FF	FLASH-2 158 BYTES
$0500 ↓ $057F	MSCAN CONTROL AND MESSAGE BUFFER 128 BYTES
$0580 ↓ $097F	RAM-2 1024 BYTES
$0980 ↓ $1B7F	FLASH-2 4608 BYTES
$1B80 ↓ $1DFF	RESERVED 640 BYTES
$1E00 ↓ $1E0F	MONITOR ROM 16 BYTES
$1E10 ↓ $1E1F	RESERVED 16 BYTES
$1E20 ↓ $7FFF	FLASH-2 25,056 BYTES
$8000 ↓ $FDFF	FLASH-1 32,256 BYTES

$FE00	SIM BREAK STATUS REGISTER (BSR)
$FE01	SIM RESET STATUS REGISTER (SRSR)
$FE02	RESERVED
$FE03	SIM BREAK FLAG CONTROL REGISTER (BFCR)
$FE04	INTERRUPT STATUS REGISTER 1 (INT1)
$FE05	INTERRUPT STATUS REGISTER 2 (INT2)
$FE06	INTERRUPT STATUS REGISTER 3 (INT3)
$FE07	INTERRUPT STATUS REGISTER 4 (INT4)
$FE08	FLASH-2 CONTROL REGISTER (FL2CR)
$FE09	BREAK ADDRESS REGISTER HIGH (BRKH)
$FE0A	BREAK ADDRESS REGISTER LOW (BRKL)
$FE0B	BREAK STATUS AND CONTROL REGISTER (BRKSCR)
$FE0C	LVI STATUS REGISTER (LVISR)
$FE0D	FLASH-2 TEST CONTROL REGISTER (FLTCR2)
$FE0E	FLASH-1 TEST CONTROL REGISTER (FLTCR1)
$FE0F	UNIMPLEMENTED
$FE10 ↓ $FE1F	UNIMPLEMENTED 16 BYTES RESERVED FOR COMPATIBILITY WITH MONITOR CODE FOR A-FAMILY PART
$FE20 ↓ $FF7F	MONITOR ROM 352 BYTES
$FF80	FLASH-1 BLOCK PROTECT REGISTER (FL1BPR)
$FF81	FLASH-2 BLOCK PROTECT REGISTER (FL2BPR)
$FF82 ↓ $FF87	RESERVED 6 BYTES
$FF88	FLASH-1 CONTROL REGISTER (FL1CR)
$FF89 ↓ $FFCB	RESERVED 67 BYTES
$FFCC ↓ $FFFF(1)	FLASH-1 VECTORS 52 BYTES

1. $FFF6–$FFFD used for eight security bytes

Figure 2-1. MC68HC908GZ60 Memory Map

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$FE03	Break Flag Control Register (BFCR) See page 238.	Read:	BCFE	R	R	R	R	R	R	R
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$FE04	Interrupt Status Register 1 (INT1) See page 231.	Read:	IF6	IF5	IF4	IF3	IF2	IF1	0	0
		Write:	R	R	R	R	R	R	R	R
		Reset:	0	0	0	0	0	0	0	0
$FE05	Interrupt Status Register 2 (INT2) See page 233.	Read:	IF14	IF13	IF12	IF11	IF10	IF9	IF8	IF7
		Write:	R	R	R	R	R	R	R	R
		Reset:	0	0	0	0	0	0	0	0
$FE06	Interrupt Status Register 3 (INT3) See page 233.	Read:	IF22	IF21	IF20	IF19	IF18	IF17	IF16	IF15
		Write:	R	R	R	R	R	R	R	R
		Reset:	0	0	0	0	0	0	0	0
$FE07	Interrupt Status Register 4 (INT4) See page 233.	Read:	0	0	0	0	0	0	IF24	IF23
		Write:	R	R	R	R	R	R	R	R
		Reset:	0	0	0	0	0	0	0	0
$FE08	FLASH-2 Control Register (FL2CR) See page 53.	Read:	0	0	0	0	HVEN	MASS	ERASE	PGM
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$FE09	Break Address Register High (BRKH) See page 303.	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$FE0A	Break Address Register Low (BRKL) See page 303.	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$FE0B	Break Status and Control Register (BRKSCR) See page 303.	Read:	BRKE	BRKA	0	0	0	0	0	0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$FE0C	LVI Status Register (LVISR) See page 133.	Read:	LVIOUT	0	0	0	0	0	0	0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$FE0D	FLASH-2 Test Control Register (FLTCR2)	Read:	R	R	R	R	R	R	R	R
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$FE0E	FLASH-1 Test Control Register (FLTCR1)	Read:	R	R	R	R	R	R	R	R
		Write:								
		Reset:	0	0	0	0	0	0	0	0

= Unimplemented R = Reserved U = Unaffected

Figure 2-2. Control, Status, and Data Registers (Sheet 8 of 9)

During the programming cycle, make sure that all addresses being written to fit within one of the ranges specified above. Attempts to program addresses in different row ranges in one programming cycle will fail.

Use this step-by-step procedure to program a row of FLASH-1 memory.

NOTE

Only bytes which are currently $FF may be programmed.

1. Set the PGM bit in the FLASH-1 control register (FL1CR). This configures the memory for program operation and enables the latching of address and data programming.
2. Read the FLASH-1 block protect register (FL1BPR).
3. Write to any FLASH-1 address within the row address range desired with any data.
4. Wait for time, t_{NVS} (minimum 10 μs).
5. Set the HVEN bit.
6. Wait for time, t_{PGS} (minimum 5 μs).
7. Write data byte to the FLASH-1 address to be programmed.
8. Wait for time, t_{PROG} (minimum 30 μs).
9. Repeat steps 7 and 8 until all the bytes within the row are programmed.
10. Clear the PGM bit.
11. Wait for time, t_{NVH} (minimum 5 μs)
12. Clear the HVEN bit.
13. Wait for a time, t_{RCV}, (typically 1 μs) after which the memory can be accessed in normal read mode.

The FLASH programming algorithm flowchart is shown in Figure 2-6.

NOTES

A. *Programming and erasing of FLASH locations can not be performed by code being executed from the same FLASH array.*

B. *While these operations must be performed in the order shown, other unrelated operations may occur between the steps. However, care must be taken to ensure that these operations do not access any address within the FLASH array memory space such as the COP control register (COPCTL) at $FFFF.*

C. *It is highly recommended that interrupts be disabled during program/erase operations.*

D. *Do not exceed t_{PROG} maximum or t_{HV} maximum. t_{HV} is defined as the cumulative high voltage programming time to the same row before next erase. t_{HV} must satisfy this condition:*

$$t_{NVS} + t_{NVH} + t_{PGS} + (t_{PROG} \times 64) \leq t_{HV} \text{ maximum}$$

E. *The time between each FLASH address change (step 7 to step 7), or the time between the last FLASH address programmed to clearing the PGM bit (step 7 to step 10) must not exceed the maximum programming time, t_{PROG} maximum.*

F. *Be cautious when programming the FLASH-1 array to ensure that non-FLASH locations are not used as the address that is written to when selecting either the desired row address range in step 3 of the algorithm or the byte to be programmed in step 7 of the algorithm.*

E. *The time between each FLASH address change (step 7 to step 7), or the time between the last FLASH address programmed to clearing the PGM bit (step 7 to step 10) must not exceed the maximum programming time, t_{PROG} maximum.*

F. *Be cautious when programming the FLASH-2 array to ensure that non-FLASH locations are not used as the address that is written to when selecting either the desired row address range in step 3 of the algorithm or the byte to be programmed in step 7 of the algorithm.*

2.7.7 Low-Power Modes

The WAIT and STOP instructions will place the MCU in low power-consumption standby modes.

2.7.7.1 Wait Mode

Putting the MCU into wait mode while the FLASH is in read mode does not affect the operation of the FLASH memory directly; however, no memory activity will take place since the CPU is inactive.

The WAIT instruction should not be executed while performing a program or erase operation on the FLASH. Wait mode will suspend any FLASH program/erase operations and leave the memory in a standby mode.

2.7.7.2 Stop Mode

Putting the MCU into stop mode while the FLASH is in read mode does not affect the operation of the FLASH memory directly; however, no memory activity will take place since the CPU is inactive.

The STOP instruction should not be executed while performing a program or erase operation on the FLASH. Stop mode will suspend any FLASH program/erase operations and leave the memory in a standby mode.

NOTE

Standby mode is the power saving mode of the FLASH module, in which all internal control signals to the FLASH are inactive and the current consumption of the FLASH is minimum.

4.3.9 CGM External Connections

In its typical configuration, the CGM requires external components. Five of these are for the crystal oscillator and two or four are for the PLL.

The crystal oscillator is normally connected in a Pierce oscillator configuration, as shown in Figure 4-2. Figure 4-2 shows only the logical representation of the internal components and may not represent actual circuitry. The oscillator configuration uses five components:

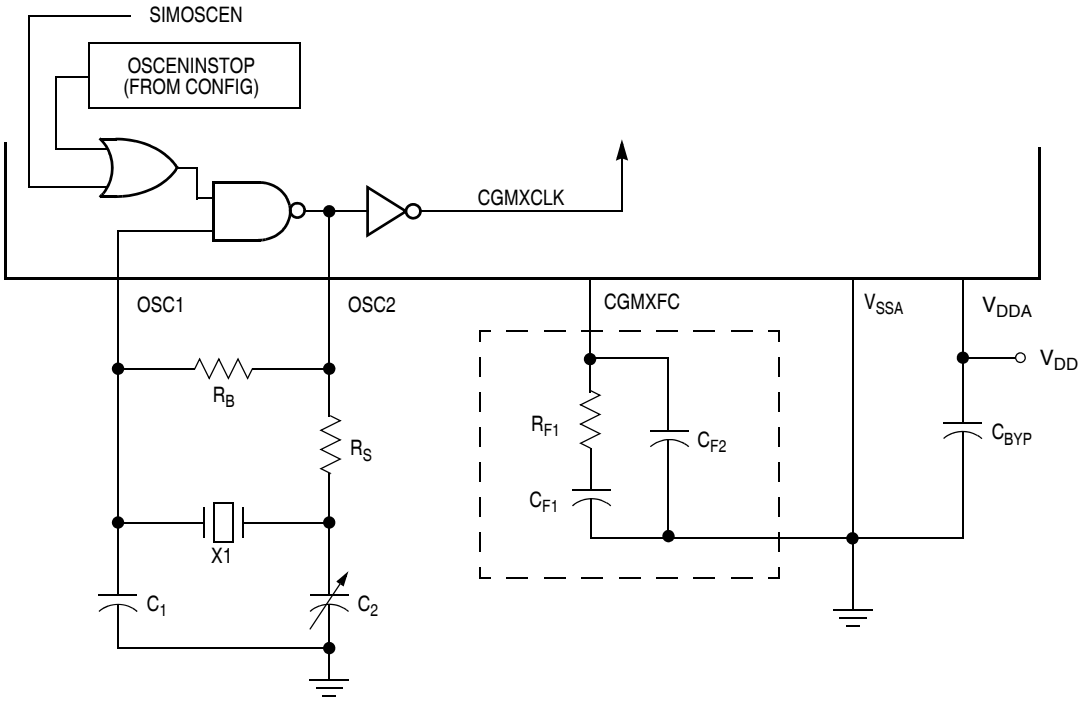
- Crystal, X_1
- Fixed capacitor, C_1
- Tuning capacitor, C_2 (can also be a fixed capacitor)
- Feedback resistor, R_B
- Series resistor, R_S

The series resistor (R_S) is included in the diagram to follow strict Pierce oscillator guidelines. Refer to the crystal manufacturer's data for more information regarding values for C1 and C2.

Figure 4-2 also shows the external components for the PLL:

- Bypass capacitor, C_{BYP}
- Filter network

Routing should be done with great care to minimize signal cross talk and noise.



Note: Filter network in box can be replaced with a single capacitor, but will degrade stability.

Figure 4-2. CGM External Connections

Chapter 10
Low-Power Modes

10.1 Introduction

The microcontroller (MCU) may enter two low-power modes: wait mode and stop mode. They are common to all HC08 MCUs and are entered through instruction execution. This section describes how each module acts in the low-power modes.

10.1.1 Wait Mode

The WAIT instruction puts the MCU in a low-power standby mode in which the central processor unit (CPU) clock is disabled but the bus clock continues to run. Power consumption can be further reduced by disabling the low-voltage inhibit (LVI) module through bits in the CONFIG1 register. See Chapter 5 Configuration Register (CONFIG).

10.1.2 Stop Mode

Stop mode is entered when a STOP instruction is executed. The CPU clock is disabled and the bus clock is disabled if the OSCENINSTOP bit in the CONFIG2 register is a 0. See Chapter 5 Configuration Register (CONFIG).

10.2 Analog-to-Digital Converter (ADC)

10.2.1 Wait Mode

The analog-to-digital converter (ADC) continues normal operation during wait mode. Any enabled CPU interrupt request from the ADC can bring the MCU out of wait mode. If the ADC is not required to bring the MCU out of wait mode, power down the ADC by setting ADCH4–ADCH0 bits in the ADC status and control register before executing the WAIT instruction.

10.2.2 Stop Mode

The ADC module is inactive after the execution of a STOP instruction. Any pending conversion is aborted. ADC conversions resume when the MCU exits stop mode after an external interrupt. Allow one conversion cycle to stabilize the analog circuitry.

12.3 External Pins

The MSCAN08 uses two external pins, one input (CAN_{RX}) and one output (CAN_{TX}). The CAN_{TX} output pin represents the logic level on the CAN: 0 is for a dominant state, and 1 is for a recessive state.

A typical CAN system with MSCAN08 is shown in Figure 12-2.

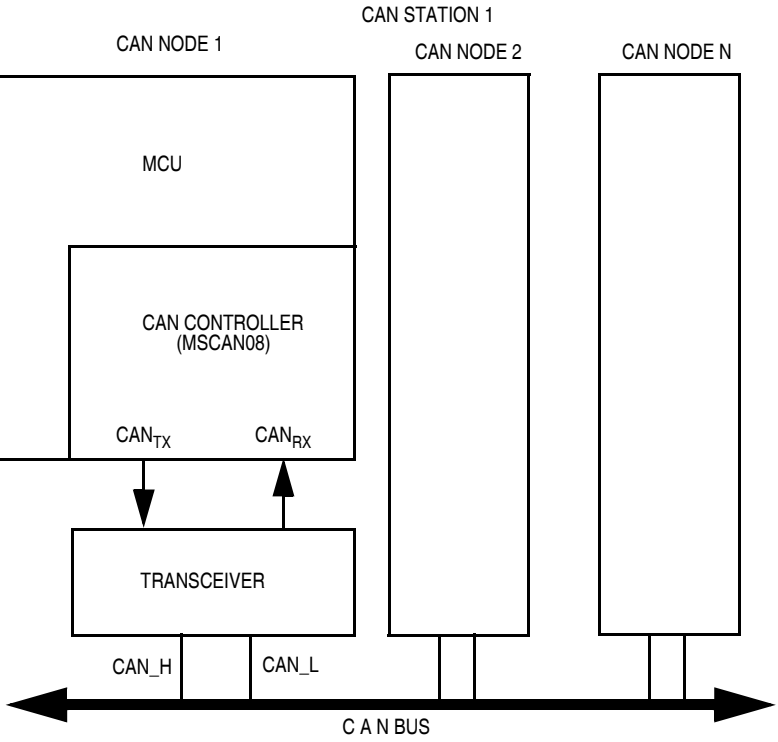


Figure 12-2. The CAN System

Each CAN station is connected physically to the CAN bus lines through a transceiver chip. The transceiver is capable of driving the large current needed for the CAN and has current protection against defected CAN or defected stations.

12.4 Message Storage

MSCAN08 facilitates a sophisticated message storage system which addresses the requirements of a broad range of network applications.

12.4.1 Background

Modern application layer software is built under two fundamental assumptions:

1. Any CAN node is able to send out a stream of scheduled messages without releasing the bus between two messages. Such nodes will arbitrate for the bus right after sending the previous message and will only release the bus in case of lost arbitration.
2. The internal message queue within any CAN node is organized as such that the highest priority message will be sent out first if more than one message is ready to be sent.

Above behavior cannot be achieved with a single transmit buffer. That buffer must be reloaded right after the previous message has been sent. This loading process lasts a definite amount of time and has to be

12.6 Interrupts

The MSCAN08 supports four interrupt vectors mapped onto eleven different interrupt sources, any of which can be individually masked. For details, see 12.13.5 MSCAN08 Receiver Flag Register (CRFLG) through 12.13.8 MSCAN08 Transmitter Control Register.

1. *Transmit Interrupt*: At least one of the three transmit buffers is empty (not scheduled) and can be loaded to schedule a message for transmission. The TXE flags of the empty message buffers are set.
2. *Receive Interrupt*: A message has been received successfully and loaded into the foreground receive buffer. This interrupt will be emitted immediately after receiving the EOF symbol. The RXF flag is set.
3. *Wakeup Interrupt*: An activity on the CAN bus occurred during MSCAN08 internal sleep mode or power-down mode (provided SLPAK = WUPIE = 1).
4. *Error Interrupt*: An overrun, error, or warning condition occurred. The receiver flag register (CRFLG) will indicate one of the following conditions:
   - *Overrun:* An overrun condition as described in 12.4.2 Receive Structures, has occurred.
   - *Receiver Warning*: The receive error counter has reached the CPU warning limit of 96.
   - *Transmitter Warning*: The transmit error counter has reached the CPU warning limit of 96.
   - *Receiver Error Passive*: The receive error counter has exceeded the error passive limit of 127 and MSCAN08 has gone to error passive state.
   - *Transmitter Error Passive*: The transmit error counter has exceeded the error passive limit of 127 and MSCAN08 has gone to error passive state.
   - *Bus Off*: The transmit error counter has exceeded 255 and MSCAN08 has gone to bus off state.

12.6.1 Interrupt Acknowledge

Interrupts are directly associated with one or more status flags in either the MSCAN08 receiver flag register (CRFLG) or the MSCAN08 transmitter flag register (CTFLG). Interrupts are pending as long as one of the corresponding flags is set. The flags in the above registers must be reset within the interrupt handler in order to handshake the interrupt. The flags are reset through writing a ‘1’ to the corresponding bit position. A flag cannot be cleared if the respective condition still prevails.

NOTE

Bit manipulation instructions (BSET) shall not be used to clear interrupt flags.

12.6.2 Interrupt Vectors

The MSCAN08 supports four interrupt vectors as shown in Table 12-1. The vector addresses and the relative interrupt priority are dependent on the chip integration and to be defined.

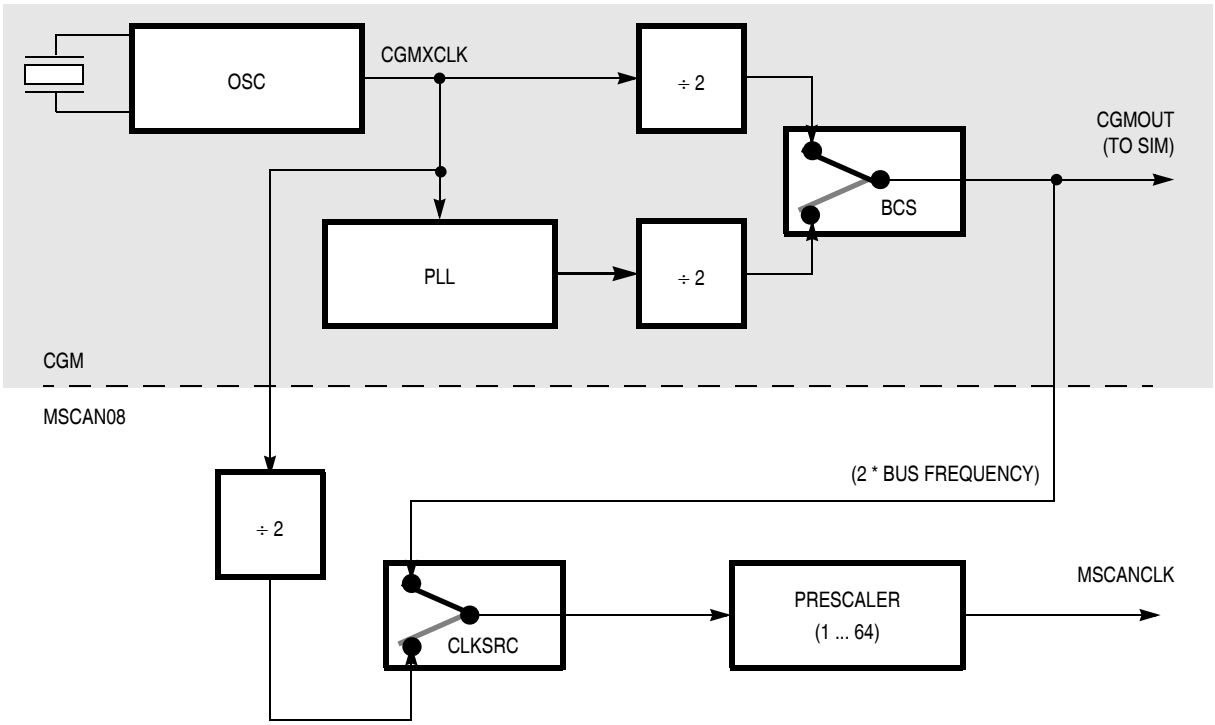


Figure 12-8. Clocking Scheme

A programmable prescaler is used to generate out of the MSCAN08 clock the time quanta (Tq) clock. A time quantum is the atomic unit of time handled by the MSCAN08.

$$f_{Tq} = \frac{f_{MSCANCLK}}{\text{Presc value}}$$

A bit time is subdivided into three segments[1] (see Figure 12-9):

- SYNC_SEG: This segment has a fixed length of one time quantum. Signal edges are expected to happen within this section.
- Time segment 1: This segment includes the PROP_SEG and the PHASE_SEG1 of the CAN standard. It can be programmed by setting the parameter TSEG1 to consist of 4 to 16 time quanta.
- Time segment 2: This segment represents PHASE_SEG2 of the CAN standard. It can be programmed by setting the TSEG2 parameter to be 2 to 8 time quanta long.

$$\text{Bit rate} = \frac{f_{Tq}}{\text{No. of time quanta}}$$

The synchronization jump width (SJW) can be programmed in a range of 1 to 4 time quanta by setting the SJW parameter.

1. For further explanation of the underlying concepts please refer to ISO/DIS 11 519-1, Section 10.3.

14.9 ESCI Arbiter

The ESCI module comprises an arbiter module designed to support software for communication tasks as bus arbitration, baud rate recovery and break time detection. The arbiter module consists of an 9-bit counter with 1-bit overflow and control logic. The CPU can control operation mode via the ESCI arbiter control register (SCIACTL).

14.9.1 ESCI Arbiter Control Register

Address: $000A

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	AM1	ALOST	AM0	ACLK	AFIN	ARUN	AROVFL	ARD8
Write:								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 14-19. ESCI Arbiter Control Register (SCIACTL)

AM1 and AM0 — Arbiter Mode Select Bits

These read/write bits select the mode of the arbiter module as shown in Table 14-12. Reset clears AM1 and AM0.

Table 14-12. ESCI Arbiter Selectable Modes

AM[1:0]	ESCI Arbiter Mode
0 0	Idle / counter reset
0 1	Bit time measurement
1 0	Bus arbitration
1 1	Reserved / do not use

ALOST — Arbitration Lost Flag

This read-only bit indicates loss of arbitration. Clear ALOST by writing a 0 to AM1. Reset clears ALOST.

ACLK — Arbiter Counter Clock Select Bit

This read/write bit selects the arbiter counter clock source. Reset clears ACLK.

1 = Arbiter counter is clocked with one half of the ESCI input clock generated by the ESCI prescaler
0 = Arbiter counter is clocked with the bus clock divided by four

NOTE

For ACLK = 1, the arbiter input clock is driven from the ESCI prescaler. The prescaler can be clocked by either the bus clock or CGMXCLK depending on the state of the SCIBDSRC bit in CONFIG2.

AFIN— Arbiter Bit Time Measurement Finish Flag

This read-only bit indicates bit time measurement has finished. Clear AFIN by writing any value to SCIACTL. Reset clears AFIN.

1 = Bit time measurement has finished
0 = Bit time measurement not yet finished

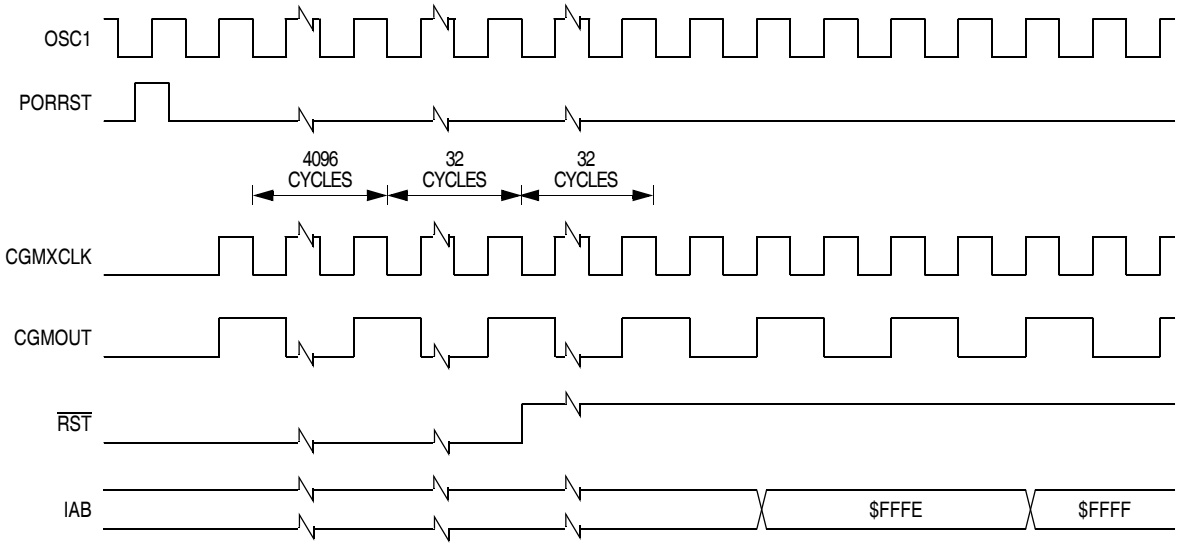


Figure 15-7. POR Recovery

15.3.2.3 Illegal Opcode Reset

The SIM decodes signals from the CPU to detect illegal instructions. An illegal instruction sets the ILOP bit in the SIM reset status register (SRSR) and causes a reset.

If the stop enable bit, STOP, in the CONFIG1 register is 0, the SIM treats the STOP instruction as an illegal opcode and causes an illegal opcode reset. The SIM actively pulls down the $\overline{\text{RST}}$ pin for all internal reset sources.

15.3.2.4 Illegal Address Reset

An opcode fetch from an unmapped address generates an illegal address reset. The SIM verifies that the CPU is fetching an opcode prior to asserting the ILAD bit in the SIM reset status register (SRSR) and resetting the MCU. A data fetch from an unmapped address does not generate a reset. The SIM actively pulls down the $\overline{\text{RST}}$ pin for all internal reset sources.

15.3.2.5 Low-Voltage Inhibit (LVI) Reset

The low-voltage inhibit module (LVI) asserts its output to the SIM when the V_{DD} voltage falls to the V_{TRIPF} voltage. The LVI bit in the SIM reset status register (SRSR) is set, and the external reset pin ($\overline{\text{RST}}$) is asserted if the LVIPWRD and LVIRSTD bits in the CONFIG1 register are 0. The $\overline{\text{RST}}$ pin will be held low while the SIM counter counts out 4096 + 32 CGMXCLK cycles after V_{DD} rises above V_{TRIPR}. Thirty-two CGMXCLK cycles later, the CPU is released from reset to allow the reset vector sequence to occur. The SIM actively pulls down the $\overline{\text{RST}}$ pin for all internal reset sources.

15.3.2.6 Monitor Mode Entry Module Reset (MODRST)

The monitor mode entry module reset (MODRST) asserts its output to the SIM when monitor mode is entered in the condition where the reset vectors are erased ($FF) (see 20.3.1.1 Normal Monitor Mode). When MODRST gets asserted, an internal reset occurs. The SIM actively pulls down the $\overline{\text{RST}}$ pin for all internal reset sources.

Table 15-3. Interrupt Sources

Source	Flag	Mask[1]	INT Register Flag	Priority[2]	Vector Address
Reset	None	None	None	0	$FFFE–$FFFF
SWI instruction	None	None	None	0	$FFFC–$FFFD
$\overline{\text{IRQ}}$ pin	IRQF	IMASK1	IF1	1	$FFFA–$FFFB
CGM change in lock	PLLF	PLLIE	IF2	2	$FFF8–$FFF9
TIM1 channel 0	CH0F	CH0IE	IF3	3	$FFF6–$FFF7
TIM1 channel 1	CH1F	CH1IE	IF4	4	$FFF4–$FFF5
TIM1 overflow	TOF	TOIE	IF5	5	$FFF2–$FFF3
TIM2 channel 0	CH0F	CH0IE	IF6	6	$FFF0–$FFF1
TIM2 channel 1	CH1F	CH1IE	IF7	7	$FFEE–$FFEF
TIM2 overflow	TOF	TOIE	IF8	8	$FFEC–$FFED
SPI receiver full	SPRF	SPRIE	IF9	9	$FFEA–$FFEB
SPI overflow	OVRF	ERRIE			
SPI mode fault	MODF	ERRIE			
SPI transmitter empty	SPTE	SPTIE	IF10	10	$FFE8–$FFE9
SCI receiver overrun	OR	ORIE	IF11	11	$FFE6–$FFE7
SCI noise flag	NF	NEIE			
SCI framing error	FE	FEIE			
SCI parity error	PE	PEIE			
SCI receiver full	SCRF	SCRIE	IF12	12	$FFE4–$FFE5
SCI input idle	IDLE	ILIE			
SCI transmitter empty	SCTE	SCTIE	IF13	13	$FFE2–$FFE3
SCI transmission complete	TC	TCIE			
Keyboard pin	KEYF	IMASKK	IF14	14	$FFE0–$FFE1
ADC conversion complete	COCO	AIEN	IF15	15	$FFDE–$FFDF
Timebase	TBIF	TBIE	IF16	16	$FFDC–$FFDD
MSCAN08 receiver wakeup	WUPIF	WUPIE	IF17	17	$FFDA–$FFDB
MSCAN08 error	RWRNIF TWRNIF RERIF TERRIF BOFFIF OVRIF	RWRNIE TWRNIE RERRIE TERRIE BOFFIE OVRIE	IF18	18	$FFD8–$FFD9
MSCAN08 receiver	RXF	RXFIE	IF19	19	$FFD6–$FFD7
MSCAN08 transmitter	TXE2 TXE1 TXE0	TXEIE2 TXEIE1 TXEIE0	IF20	20	$FFD4–$FFD5
TIM2 channel 2	CH2F	CH2IE	IF21	21	$FFD2–FFD3
TIM2 channel 3	CH3F	CH3IE	IF22	22	$FFD0–FFD1
TIM2 channel 4	CH4F	CH4IE	IF23	23	$FFCE–FFCF
TIM2 channel 5	CH5F	CH5IE	IF24	24	$FFCC–FFCD

1. The I bit in the condition code register is a global mask for all interrupt sources except the SWI instruction.
2. 0 = highest priority

17.7 Timebase Control Register

The timebase has one register, the timebase control register (TBCR), which is used to enable the timebase interrupts and set the rate.

Address: $001C

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	TBIF	TBR2	TBR1	TBR0	0	TBIE	TBON	R
Write:					TACK			
Reset:	0	0	0	0	0	0	0	0

= Unimplemented R = Reserved

Figure 17-2. Timebase Control Register (TBCR)

TBIF — Timebase Interrupt Flag

This read-only flag bit is set when the timebase counter has rolled over.

1 = Timebase interrupt pending
0 = Timebase interrupt not pending

TBR2–TBR0 — Timebase Divider Selection Bits

These read/write bits select the tap in the counter to be used for timebase interrupts as shown in Table 17-1.

NOTE

Do not change TBR2–TBR0 bits while the timebase is enabled (TBON = 1).

TACK— Timebase Acknowledge Bit

The TACK bit is a write-only bit and always reads as 0. Writing a 1 to this bit clears TBIF, the timebase interrupt flag bit. Writing a 0 to this bit has no effect.

1 = Clear timebase interrupt flag
0 = No effect

TBIE — Timebase Interrupt Enabled Bit

This read/write bit enables the timebase interrupt when the TBIF bit becomes set. Reset clears the TBIE bit.

1 = Timebase interrupt is enabled.
0 = Timebase interrupt is disabled.

TBON — Timebase Enabled Bit

This read/write bit enables the timebase. Timebase may be turned off to reduce power consumption when its function is not necessary. The counter can be initialized by clearing and then setting this bit. Reset clears the TBON bit.

1 = Timebase is enabled.
0 = Timebase is disabled and the counter initialized to 0s.

19.3.1 TIM2 Counter Prescaler

The TIM2 clock source can be one of the seven prescaler outputs or the TIM2 clock pin, T2CH0. The prescaler generates seven clock rates from the internal bus clock. The prescaler select bits, PS[2:0], in the TIM2 status and control register select the TIM2 clock source.

19.3.2 Input Capture

An input capture function has three basic parts: edge select logic, an input capture latch, and a 16-bit counter. Two 8-bit registers, which make up the 16-bit input capture register, are used to latch the value of the free-running counter after the corresponding input capture edge detector senses a defined transition. The polarity of the active edge is programmable. The level transition which triggers the counter transfer is defined by the corresponding input edge bits (ELSxB and ELSxA in T2SC0 through T2SC5 control registers with x referring to the active channel number). When an active edge occurs on the pin of an input capture channel, the TIM2 latches the contents of the TIM2 counter into the TIM2 channel registers, T2CHxH:T2CHxL. Input captures can generate TIM2 CPU interrupt requests. Software can determine that an input capture event has occurred by enabling input capture interrupts or by polling the status flag bit.

The free-running counter contents are transferred to the TIM2 channel registers (T2CHxH:T2CHxL) (see 19.8.5 TIM2 Channel Registers) on each proper signal transition regardless of whether the TIM2 channel flag (CH0F–CH5F in T2SC0–T2SC5 registers) is set or clear. When the status flag is set, a CPU interrupt is generated if enabled. The value of the count latched or “captured” is the time of the event. Because this value is stored in the input capture register when the actual event occurs, user software can respond to this event at a later time and determine the actual time of the event. However, this must be done prior to another input capture on the same pin; otherwise, the previous time value will be lost.

By recording the times for successive edges on an incoming signal, software can determine the period and/or pulse width of the signal. To measure a period, two successive edges of the same polarity are captured. To measure a pulse width, two alternate polarity edges are captured. Software should track the overflows at the 16-bit module counter to extend its range.

Another use for the input capture function is to establish a time reference. In this case, an input capture function is used in conjunction with an output compare function. For example, to activate an output signal a specified number of clock cycles after detecting an input event (edge), use the input capture function to record the time at which the edge occurred. A number corresponding to the desired delay is added to this captured value and stored to an output compare register (see 19.8.5 TIM2 Channel Registers). Because both input captures and output compares are referenced to the same 16-bit modulo counter, the delay can be controlled to the resolution of the counter independent of software latencies.

Reset does not affect the contents of the input capture channel (T2CHxH:T2CHxL) registers.

19.3.3 Output Compare

With the output compare function, the TIM2 can generate a periodic pulse with a programmable polarity, duration, and frequency. When the counter reaches the value in the registers of an output compare channel, the TIM2 can set, clear, or toggle the channel pin. Output compares can generate TIM2 CPU interrupt requests.

20.3.1.1 Normal Monitor Mode

If V_{TST} is applied to $\overline{IRQ}$ and PTB4 is low upon monitor mode entry, the bus frequency is a divide-by-two of the input clock. If PTB4 is high with V_{TST} applied to $\overline{IRQ}$ upon monitor mode entry, the bus frequency will be a divide-by-four of the input clock. Holding the PTB4 pin low when entering monitor mode causes a bypass of a divide-by-two stage at the oscillator *only if V_{TST} is applied to $\overline{IRQ}$*. In this event, the CGMOUT frequency is equal to the CGMXCLK frequency, and the OSC1 input directly generates internal bus clocks. In this case, the OSC1 signal must have a 50% duty cycle at maximum bus frequency.

When monitor mode was entered with V_{TST} on $\overline{IRQ}$, the computer operating properly (COP) is disabled as long as V_{TST} is applied to either $\overline{IRQ}$ or $\overline{RST}$.

This condition states that as long as V_{TST} is maintained on the $\overline{IRQ}$ pin after entering monitor mode, or if V_{TST} is applied to $\overline{RST}$ after the initial reset to get into monitor mode (when V_{TST} was applied to $\overline{IRQ}$), then the COP will be disabled. In the latter situation, after V_{TST} is applied to the $\overline{RST}$ pin, V_{TST} can be removed from the $\overline{IRQ}$ pin in the interest of freeing the $\overline{IRQ}$ for normal functionality in monitor mode.

20.3.1.2 Forced Monitor Mode

If entering monitor mode without high voltage on $\overline{IRQ}$, then all port B pin requirements and conditions, including the PTB4 frequency divisor selection, are not in effect. This is to reduce circuit requirements when performing in-circuit programming.

NOTE

If the reset vector is blank and monitor mode is entered, the chip will see an additional reset cycle after the initial power-on reset (POR). Once the reset vector has been programmed, the traditional method of applying a voltage, V_{TST}, to $\overline{IRQ}$ must be used to enter monitor mode.

An external oscillator of 8 MHz is required for a baud rate of 7200, as the internal bus frequency is automatically set to the external frequency divided by four.

When the forced monitor mode is entered the COP is always disabled regardless of the state of $\overline{IRQ}$ or $\overline{RST}$.

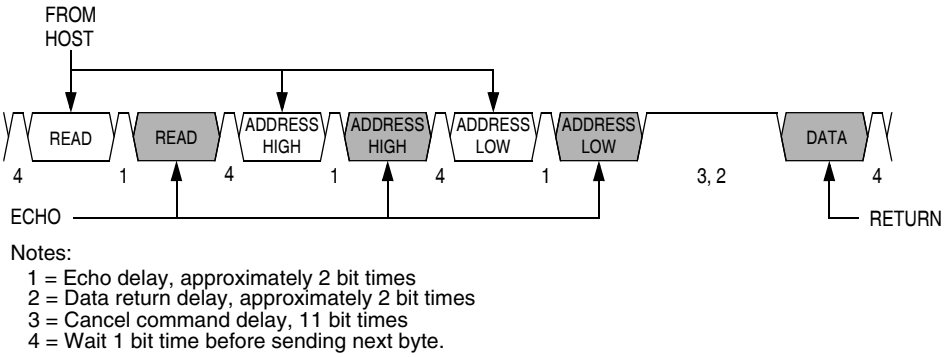
20.3.1.3 Monitor Vectors

In monitor mode, the MCU uses different vectors for reset, SWI (software interrupt), and break interrupt than those for user mode. The alternate vectors are in the $FE page instead of the $FF page and allow code execution from the internal monitor firmware instead of user code.

Table 20-2 summarizes the differences between user mode and monitor mode.

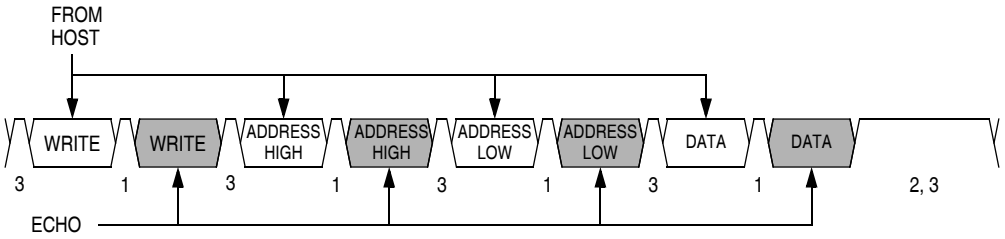
Table 20-2. Mode Differences

Modes	Functions					
	Reset Vector High	Reset Vector Low	Break Vector High	Break Vector Low	SWI Vector High	SWI Vector Low
User	$FFFE	$FFFF	$FFFC	$FFFD	$FFFC	$FFFD
Monitor	$FEFE	$FEFF	$FEFC	$FEFD	$FEFC	$FEFD



Notes:
1 = Echo delay, approximately 2 bit times
2 = Data return delay, approximately 2 bit times
3 = Cancel command delay, 11 bit times
4 = Wait 1 bit time before sending next byte.

Figure 20-14. Read Transaction



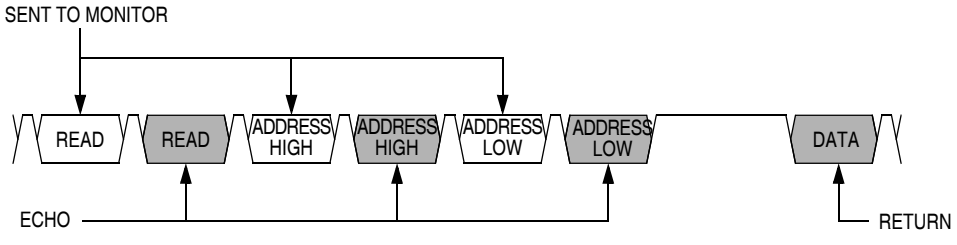
Notes:
1 = Echo delay, approximately 2 bit times
2 = Cancel command delay, 11 bit times
3 = Wait 1 bit time before sending next byte.

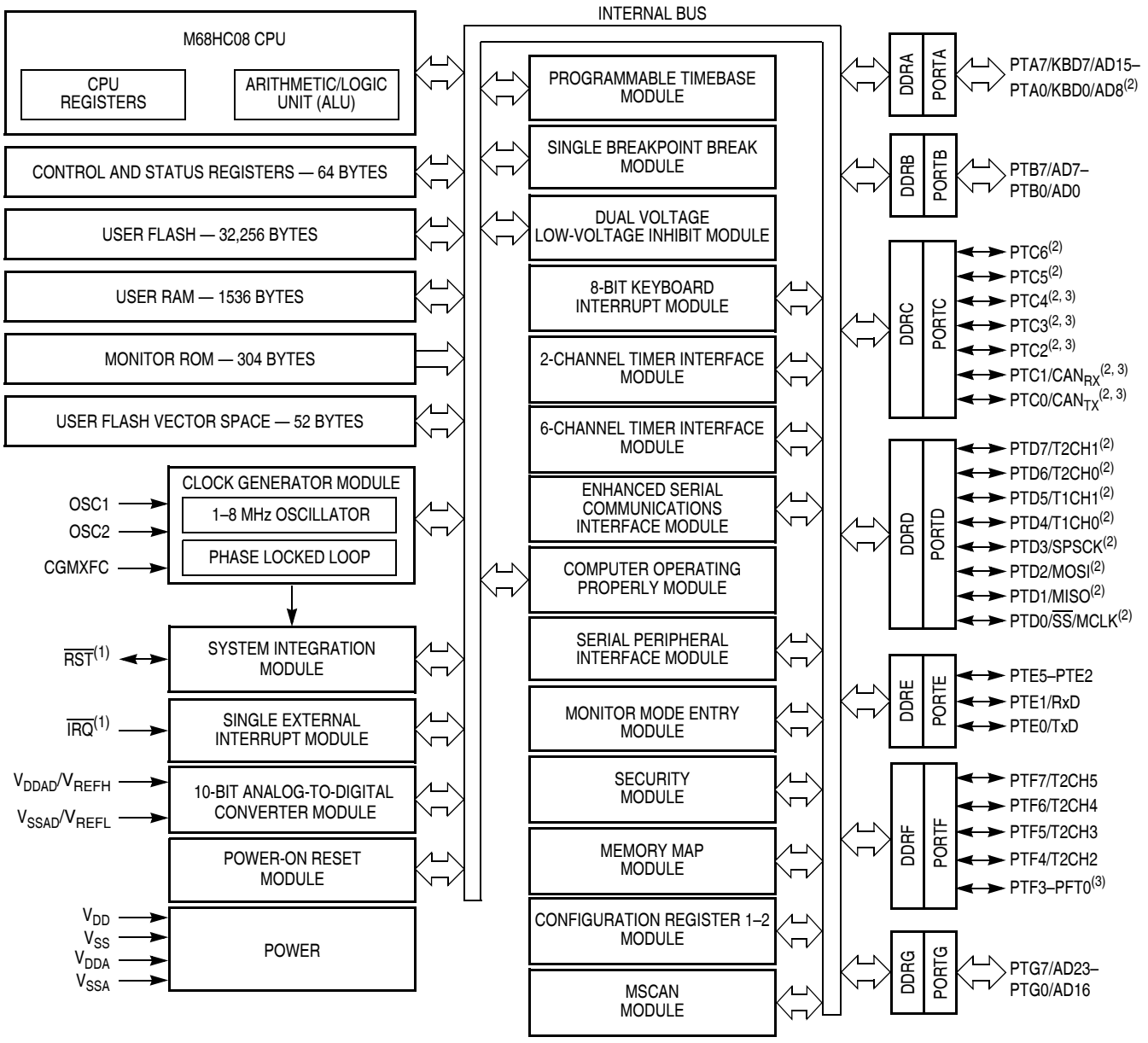
Figure 20-15. Write Transaction

A brief description of each monitor mode command is given in Table 20-3 through Table 20-8.

Table 20-3. READ (Read Memory) Command

Description	Read byte from memory
Operand	2-byte address in high-byte:low-byte order
Data Returned	Returns contents of specified address
Opcode	$4A
Command Sequence	





1. Pin contains integrated pullup device.
2. Ports are software configurable with pullup device if input port or pullup/pulldown device for keyboard input.
3. Higher current drive port pins

Figure B-1. MC68HC908GZ32 Block Diagram