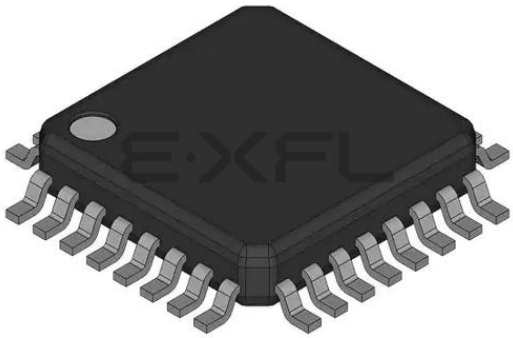


Welcome to E-XFL.COM

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	HC08
Core Size	8-Bit
Speed	8MHz
Connectivity	SCI, SPI
Peripherals	LVD, POR, PWM
Number of I/O	21
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 ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	32-LQFP
Supplier Device Package	32-LQFP (7x7)
Purchase URL	https://www.e-xfl.com/pro/item?MUrl=&PartUrl=mc908gr60acfje

Chapter 2
Memory

2.1 Introduction

The CPU08 can address 64 Kbytes of memory space. The memory map, shown in Figure 2-1, includes:

- 62,078 bytes of user FLASH memory
- 2048 bytes of random-access memory (RAM)
- 52 bytes of user-defined vectors

2.2 Unimplemented Memory Locations

Accessing an unimplemented location can cause an illegal address reset. In the memory map (Figure 2-1) and in register figures in this document, unimplemented locations are shaded.

2.3 Reserved Memory Locations

Accessing a reserved location can have unpredictable effects on microcontroller (MCU) operation. In the Figure 2-1 and in register figures in this document, reserved locations are marked with the word Reserved or with the letter R.

2.4 Input/Output (I/O) Section

Most of the control, status, and data registers are in the zero page area of $0000–$003F, or at $0440–$0461. Additional I/O registers have these addresses:

- $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
- $FF80; FLASH-1 block protect register, FL1BPR
- $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.

2.6.4 FLASH-1 Mass Erase Operation

Use this step-by-step procedure to erase the entire FLASH-1 memory:

1. Set both the ERASE bit and the MASS bit in the FLASH-1 control register (FL1CR).
2. Read the FLASH-1 block protect register (FL1BPR).

NOTE

Mass erase is disabled whenever any block is protected (FL1BPR does not equal $FF).

3. Write to any FLASH-1 address within the FLASH-1 array with any data.
4. Wait for a time, t_{NVS} (minimum 10 μs).
5. Set the HVEN bit.
6. Wait for a time, t_{MERASE} (minimum 4 ms).
7. Clear the ERASE and MASS bits.
8. Wait for a time, t_{NVHL} (minimum 100 μs).
9. Clear the HVEN bit.
10. Wait for a time, t_{RCV}, (typically 1 μs) after which the memory can be accessed in normal read mode.

NOTE

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.*

2.6.5 FLASH-1 Page Erase Operation

Use this step-by-step procedure to erase a page (128 bytes) of FLASH-1 memory:

1. Set the ERASE bit and clear the MASS bit in the FLASH-1 control register (FL1CR).
2. Read the FLASH-1 block protect register (FL1BPR).
3. Write any data to any FLASH-1 address within the address range of the page (128 byte block) to be erased.
4. Wait for time, t_{NVS} (minimum 10 μs).
5. Set the HVEN bit.
6. Wait for time, t_{ERASE} (minimum 1 ms or 4 ms).
7. Clear the ERASE bit.
8. Wait for time, t_{NVH} (minimum 5 μs).
9. Clear the HVEN bit.
10. Wait for a time, t_{RCV}, (typically 1 μs) after which the memory can be accessed in normal read mode.

NOTE

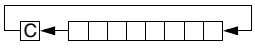
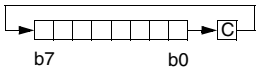
A. *Programming and erasing of FLASH locations can not be performed by code being executed from the same FLASH array.*

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.*

In applications that require more than 1000 program/erase cycles, use the 4 ms page erase specification to get improved long-term reliability. Any application can use this 4 ms page erase specification. However, in applications where a FLASH location will be erased and reprogrammed less than 1000 times, and speed is important, use the 1 ms page erase specification to get a shorter cycle time.

Table 7-1. Instruction Set Summary (Sheet 5 of 6)

Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Cycles
			V	H	I	N	Z	C				
PULA	Pull A from Stack	SP ← (SP + 1); Pull (A)	–	–	–	–	–	–	INH	86		2
PULH	Pull H from Stack	SP ← (SP + 1); Pull (H)	–	–	–	–	–	–	INH	8A		2
PULX	Pull X from Stack	SP ← (SP + 1); Pull (X)	–	–	–	–	–	–	INH	88		2
ROL *opr* ROLA ROLX ROL *opr*,X ROL ,X ROL *opr*,SP	Rotate Left through Carry		↕	–	–	↕	↕	↕	DIR INH INH IX1 IX SP1	39 49 59 69 79 9E69	dd ff ff	4 1 1 4 3 5
ROR *opr* RORA RORX ROR *opr*,X ROR ,X ROR *opr*,SP	Rotate Right through Carry		↕	–	–	↕	↕	↕	DIR INH INH IX1 IX SP1	36 46 56 66 76 9E66	dd ff ff	4 1 1 4 3 5
RSP	Reset Stack Pointer	SP ← $FF	–	–	–	–	–	–	INH	9C		1
RTI	Return from Interrupt	SP ← (SP) + 1; Pull (CCR) SP ← (SP) + 1; Pull (A) SP ← (SP) + 1; Pull (X) SP ← (SP) + 1; Pull (PCH) SP ← (SP) + 1; Pull (PCL)	↕	↕	↕	↕	↕	↕	INH	80		7
RTS	Return from Subroutine	SP ← SP + 1; Pull (PCH) SP ← SP + 1; Pull (PCL)	–	–	–	–	–	–	INH	81		4
SBC #*opr* SBC *opr* SBC *opr* SBC *opr*,X SBC *opr*,X SBC ,X SBC *opr*,SP SBC *opr*,SP	Subtract with Carry	A ← (A) – (M) – (C)	↕	–	–	↕	↕	↕	IMM DIR EXT IX2 IX1 IX SP1 SP2	A2 B2 C2 D2 E2 F2 9EE2 9ED2	ii dd hh ll ee ff ff ff ee ff	2 3 4 4 3 2 4 5
SEC	Set Carry Bit	C ← 1	–	–	–	–	–	1	INH	99		1
SEI	Set Interrupt Mask	I ← 1	–	–	1	–	–	–	INH	9B		2
STA *opr* STA *opr* STA *opr*,X STA *opr*,X STA ,X STA *opr*,SP STA *opr*,SP	Store A in M	M ← (A)	0	–	–	↕	↕	–	DIR EXT IX2 IX1 IX SP1 SP2	B7 C7 D7 E7 F7 9EE7 9ED7	dd hh ll ee ff ff ff ee ff	3 4 4 3 2 4 5
STHX *opr*	Store H:X in M	(M:M + 1) ← (H:X)	0	–	–	↕	↕	–	DIR	35	dd	4
STOP	Enable Interrupts, Stop Processing, Refer to MCU Documentation	I ← 0; Stop Processing	–	–	0	–	–	–	INH	8E		1
STX *opr* STX *opr* STX *opr*,X STX *opr*,X STX ,X STX *opr*,SP STX *opr*,SP	Store X in M	M ← (X)	0	–	–	↕	↕	–	DIR EXT IX2 IX1 IX SP1 SP2	BF CF DF EF FF 9EEF 9EDF	dd hh ll ee ff ff ff ee ff	3 4 4 3 2 4 5
SUB #*opr* SUB *opr* SUB *opr* SUB *opr*,X SUB *opr*,X SUB ,X SUB *opr*,SP SUB *opr*,SP	Subtract	A ← (A) – (M)	↕	–	–	↕	↕	↕	IMM DIR EXT IX2 IX1 IX SP1 SP2	A0 B0 C0 D0 E0 F0 9EE0 9ED0	ii dd hh ll ee ff ff ff ee ff	2 3 4 4 3 2 4 5

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.

10.3 Break Module (BRK)

10.3.1 Wait Mode

The break (BRK) module is active in wait mode. In the break routine, the user can subtract one from the return address on the stack if the SBSW bit in the break status register is set.

10.3.2 Stop Mode

The break module is inactive in stop mode. The STOP instruction does not affect break module register states.

12.3.2 Data Direction Register A

Data direction register A (DDRA) determines whether each port A pin is an input or an output. Writing a 1 to a DDRA bit enables the output buffer for the corresponding port A pin; a 0 disables the output buffer.

Address: $0004

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	DDRA7	DDRA6	DDRA5	DDRA4	DDRA3	DDRA2	DDRA1	DDRA0
Reset:	0	0	0	0	0	0	0	0

Figure 12-3. Data Direction Register A (DDRA)

DDRA7–DDRA0 — Data Direction Register A Bits

These read/write bits control port A data direction. Reset clears DDRA7–DDRA0, configuring all port A pins as inputs.

1 = Corresponding port A pin configured as output
0 = Corresponding port A pin configured as input

NOTE

Avoid glitches on port A pins by writing to the port A data register before changing data direction register A bits from 0 to 1.

Figure 12-4 shows the port A I/O logic.

When bit DDRAx is a 1, reading address $0000 reads the PTAx data latch. When bit DDRAx is a 0, reading address $0000 reads the voltage level on the pin. The data latch can always be written, regardless of the state of its data direction bit. Table 12-2 summarizes the operation of the port A pins.

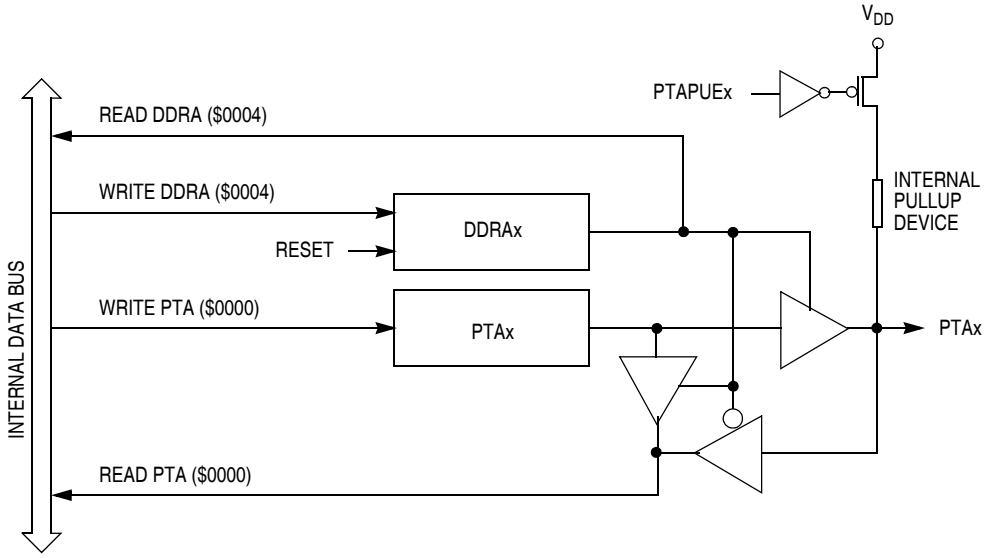


Figure 12-4. Port A I/O Circuit

12.4 Port B

Port B is an 8-bit special-function port that shares all eight of its pins with the analog-to-digital converter (ADC) module.

12.4.1 Port B Data Register

The port B data register (PTB) contains a data latch for each of the eight port pins.

Address:	$0001							
	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	PTB7	PTB6	PTB5	PTB4	PTB3	PTB2	PTB1	PTB0
Reset:	Unaffected by reset							
Alternate Function:	AD7	AD6	AD5	AD4	AD3	AD2	AD1	AD0

Figure 12-6. Port B Data Register (PTB)

PTB7–PTB0 — Port B Data Bits

These read/write bits are software-programmable. Data direction of each port B pin is under the control of the corresponding bit in data direction register B. Reset has no effect on port B data.

AD7–AD0 — Analog-to-Digital Input Bits

AD7–AD0 are pins used for the input channels to the analog-to-digital converter module. The channel select bits in the ADC status and control register define which port B pin will be used as an ADC input and overrides any control from the port I/O logic by forcing that pin as the input to the analog circuitry.

NOTE

Care must be taken when reading port B while applying analog voltages to AD7–AD0 pins. If the appropriate ADC channel is not enabled, excessive current drain may occur if analog voltages are applied to the PTBx/ADx pin, while PTB is read as a digital input during the CPU read cycle. Those ports not selected as analog input channels are considered digital I/O ports.

12.4.2 Data Direction Register B

Data direction register B (DDRB) determines whether each port B pin is an input or an output. Writing a 1 to a DDRB bit enables the output buffer for the corresponding port B pin; a 0 disables the output buffer.

Address:	$0005							
	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	DDRB7	DDRB6	DDRB5	DDRB4	DDRB3	DDRB2	DDRB1	DDRB0
Reset:	0	0	0	0	0	0	0	0

Figure 12-7. Data Direction Register B (DDRB)

DDRB7–DDRB0 — Data Direction Register B Bits

These read/write bits control port B data direction. Reset clears DDRB7–DDRB0, configuring all port B pins as inputs.

1 = Corresponding port B pin configured as output
0 = Corresponding port B pin configured as input

NOTE

With the WAKE bit clear, setting the RWU bit after the RxD pin has been idle will cause the receiver to wake up.

13.4.3.7 Receiver Interrupts

These sources can generate CPU interrupt requests from the ESCI receiver:

- ESCI receiver full (SCRF) — The SCRF bit in SCS1 indicates that the receive shift register has transferred a character to the SCDR. SCRF can generate a receiver CPU interrupt request. Setting the ESCI receive interrupt enable bit, SCRIE, in SCC2 enables the SCRF bit to generate receiver CPU interrupts.
- Idle input (IDLE) — The IDLE bit in SCS1 indicates that 10 or 11 consecutive 1s shifted in from the RxD pin. The idle line interrupt enable bit, ILIE, in SCC2 enables the IDLE bit to generate CPU interrupt requests.

13.4.3.8 Error Interrupts

These receiver error flags in SCS1 can generate CPU interrupt requests:

- Receiver overrun (OR) — The OR bit indicates that the receive shift register shifted in a new character before the previous character was read from the SCDR. The previous character remains in the SCDR, and the new character is lost. The overrun interrupt enable bit, ORIE, in SCC3 enables OR to generate ESCI error CPU interrupt requests.
- Noise flag (NF) — The NF bit is set when the ESCI detects noise on incoming data or break characters, including start, data, and stop bits. The noise error interrupt enable bit, NEIE, in SCC3 enables NF to generate ESCI error CPU interrupt requests.
- Framing error (FE) — The FE bit in SCS1 is set when a 0 occurs where the receiver expects a stop bit. The framing error interrupt enable bit, FEIE, in SCC3 enables FE to generate ESCI error CPU interrupt requests.
- Parity error (PE) — The PE bit in SCS1 is set when the ESCI detects a parity error in incoming data. The parity error interrupt enable bit, PEIE, in SCC3 enables PE to generate ESCI error CPU interrupt requests.

13.5 Low-Power Modes

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

13.5.1 Wait Mode

The ESCI module remains active in wait mode. Any enabled CPU interrupt request from the ESCI module can bring the MCU out of wait mode.

If ESCI module functions are not required during wait mode, reduce power consumption by disabling the module before executing the WAIT instruction.

13.5.2 Stop Mode

The ESCI module is inactive in stop mode. The STOP instruction does not affect ESCI register states. ESCI module operation resumes after the MCU exits stop mode.

Because the internal clock is inactive during stop mode, entering stop mode during an ESCI transmission or reception results in invalid data.

13.8.1 ESCI Control Register 1

ESCI control register 1 (SCC1):

- Enables loop mode operation
- Enables the ESCI
- Controls output polarity
- Controls character length
- Controls ESCI wakeup method
- Controls idle character detection
- Enables parity function
- Controls parity type

Address: $0013

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	LOOPS	ENSCI	TXINV	M	WAKE	ILTY	PEN	PTY
Reset:	0	0	0	0	0	0	0	0

Figure 13-10. ESCI Control Register 1 (SCC1)

LOOPS — Loop Mode Select Bit

This read/write bit enables loop mode operation. In loop mode the RxD pin is disconnected from the ESCI, and the transmitter output goes into the receiver input. Both the transmitter and the receiver must be enabled to use loop mode. Reset clears the LOOPS bit.

1 = Loop mode enabled
0 = Normal operation enabled

ENSCI — Enable ESCI Bit

This read/write bit enables the ESCI and the ESCI baud rate generator. Clearing ENSCI sets the SCTE and TC bits in ESCI status register 1 and disables transmitter interrupts. Reset clears the ENSCI bit.

1 = ESCI enabled
0 = ESCI disabled

TXINV — Transmit Inversion Bit

This read/write bit reverses the polarity of transmitted data. Reset clears the TXINV bit.

1 = Transmitter output inverted
0 = Transmitter output not inverted

NOTE

Setting the TXINV bit inverts all transmitted values including idle, break, start, and stop bits.

M — Mode (Character Length) Bit

This read/write bit determines whether ESCI characters are eight or nine bits long (See Table 13-5).The ninth bit can serve as a receiver wakeup signal or as a parity bit. Reset clears the M bit.

1 = 9-bit ESCI characters
0 = 8-bit ESCI characters

When the ESCI is receiving 8-bit characters, R8 is a copy of the eighth bit (bit 7). Reset has no effect on the R8 bit.

T8 — Transmitted Bit 8

When the ESCI is transmitting 9-bit characters, T8 is the read/write ninth bit (bit 8) of the transmitted character. T8 is loaded into the transmit shift register at the same time that the SCDR is loaded into the transmit shift register. Reset clears the T8 bit.

ORIE — Receiver Overrun Interrupt Enable Bit

This read/write bit enables ESCI error CPU interrupt requests generated by the receiver overrun bit, OR. Reset clears ORIE.

1 = ESCI error CPU interrupt requests from OR bit enabled
0 = ESCI error CPU interrupt requests from OR bit disabled

NEIE — Receiver Noise Error Interrupt Enable Bit

This read/write bit enables ESCI error CPU interrupt requests generated by the noise error bit, NE. Reset clears NEIE.

1 = ESCI error CPU interrupt requests from NE bit enabled
0 = ESCI error CPU interrupt requests from NE bit disabled

FEIE — Receiver Framing Error Interrupt Enable Bit

This read/write bit enables ESCI error CPU interrupt requests generated by the framing error bit, FE. Reset clears FEIE.

1 = ESCI error CPU interrupt requests from FE bit enabled
0 = ESCI error CPU interrupt requests from FE bit disabled

PEIE — Receiver Parity Error Interrupt Enable Bit

This read/write bit enables ESCI receiver CPU interrupt requests generated by the parity error bit, PE. Reset clears PEIE.

1 = ESCI error CPU interrupt requests from PE bit enabled
0 = ESCI error CPU interrupt requests from PE bit disabled

13.8.4 ESCI Status Register 1

ESCI status register 1 (SCS1) contains flags to signal these conditions:

- Transfer of SCDR data to transmit shift register complete
- Transmission complete
- Transfer of receive shift register data to SCDR complete
- Receiver input idle
- Receiver overrun
- Noisy data
- Framing error
- Parity error

Address: $0016

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	SCTE	TC	SCRF	IDLE	OR	NF	FE	PE
Write:								
Reset:	1	1	0	0	0	0	0	0

= Unimplemented

Figure 13-13. ESCI Status Register 1 (SCS1)

Table 13-11. ESCI Baud Rate Selection Examples

PDS[2:1:0]	PSSB[4:3:2:1:0]	SCP[1:0]	Prescaler Divisor (BPD)	SCR[2:1:0]	Baud Rate Divisor (BD)	Baud Rate (f_{Bus}= 4.9152 MHz)
0 0 0	X X X X X	0 0	1	0 0 0	1	76,800
1 1 1	0 0 0 0 0	0 0	1	0 0 0	1	9600
1 1 1	0 0 0 0 1	0 0	1	0 0 0	1	9562.65
1 1 1	0 0 0 1 0	0 0	1	0 0 0	1	9525.58
1 1 1	1 1 1 1 1	0 0	1	0 0 0	1	8563.07
0 0 0	X X X X X	0 0	1	0 0 1	2	38,400
0 0 0	X X X X X	0 0	1	0 1 0	4	19,200
0 0 0	X X X X X	0 0	1	0 1 1	8	9600
0 0 0	X X X X X	0 0	1	1 0 0	16	4800
0 0 0	X X X X X	0 0	1	1 0 1	32	2400
0 0 0	X X X X X	0 0	1	1 1 0	64	1200
0 0 0	X X X X X	0 0	1	1 1 1	128	600
0 0 0	X X X X X	0 1	3	0 0 0	1	25,600
0 0 0	X X X X X	0 1	3	0 0 1	2	12,800
0 0 0	X X X X X	0 1	3	0 1 0	4	6400
0 0 0	X X X X X	0 1	3	0 1 1	8	3200
0 0 0	X X X X X	0 1	3	1 0 0	16	1600
0 0 0	X X X X X	0 1	3	1 0 1	32	800
0 0 0	X X X X X	0 1	3	1 1 0	64	400
0 0 0	X X X X X	0 1	3	1 1 1	128	200
0 0 0	X X X X X	1 0	4	0 0 0	1	19,200
0 0 0	X X X X X	1 0	4	0 0 1	2	9600
0 0 0	X X X X X	1 0	4	0 1 0	4	4800
0 0 0	X X X X X	1 0	4	0 1 1	8	2400
0 0 0	X X X X X	1 0	4	1 0 0	16	1200
0 0 0	X X X X X	1 0	4	1 0 1	32	600
0 0 0	X X X X X	1 0	4	1 1 0	64	300
0 0 0	X X X X X	1 0	4	1 1 1	128	150
0 0 0	X X X X X	1 1	13	0 0 0	1	5908
0 0 0	X X X X X	1 1	13	0 0 1	2	2954
0 0 0	X X X X X	1 1	13	0 1 0	4	1477
0 0 0	X X X X X	1 1	13	0 1 1	8	739
0 0 0	X X X X X	1 1	13	1 0 0	16	369
0 0 0	X X X X X	1 1	13	1 0 1	32	185
0 0 0	X X X X X	1 1	13	1 1 0	64	92
0 0 0	X X X X X	1 1	13	1 1 1	128	46

AROVFL— Arbiter Counter Overflow Bit

This read-only bit indicates an arbiter counter overflow. Clear AROVFL by writing any value to SCIACTL. Writing 0s to AM1 and AM0 resets the counter keeps it in this idle state. Reset clears AROVFL.

1 = Arbiter counter overflow has occurred
0 = No arbiter counter overflow has occurred

ARD8— Arbiter Counter MSB

This read-only bit is the MSB of the 9-bit arbiter counter. Clear ARD8 by writing any value to SCIACTL. Reset clears ARD8.

13.9.2 ESCI Arbiter Data Register

Address: $000B

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	ARD7	ARD6	ARD5	ARD4	ARD3	ARD2	ARD1	ARD0
Write:								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 13-20. ESCI Arbiter Data Register (SCIADAT)

ARD7–ARD0 — Arbiter Least Significant Counter Bits

These read-only bits are the eight LSBs of the 9-bit arbiter counter. Clear ARD7–ARD0 by writing any value to SCIACTL. Writing 0s to AM1 and AM0 permanently resets the counter and keeps it in this idle state. Reset clears ARD7–ARD0.

13.9.3 Bit Time Measurement

Two bit time measurement modes, described here, are available according to the state of ACLK.

1. ACLK = 0 — The counter is clocked with the bus clock divided by four. The counter is started when a falling edge on the RxD pin is detected. The counter will be stopped on the next falling edge. ARUN is set while the counter is running, AFIN is set on the second falling edge on RxD (for instance, the counter is stopped). This mode is used to recover the received baud rate. See Figure 13-21.
2. ACLK = 1 — The counter is clocked with one half of the ESCI input clock generated by the ESCI prescaler. The counter is started when a 0 is detected on RxD (see Figure 13-22). A 0 on RxD on enabling the bit time measurement with ACLK = 1 leads to immediate start of the counter (see Figure 13-23). The counter will be stopped on the next rising edge of RxD. This mode is used to measure the length of a received break.

13.9.4 Arbitration Mode

If AM[1:0] is set to 10, the arbiter module operates in arbitration mode. On every rising edge of SCI_TxD (output of the ESCI module, internal chip signal), the counter is started. When the counter reaches $38 (ACLK = 0) or $08 (ACLK = 1), RxD is statically sensed. If in this case, RxD is sensed low (for example, another bus is driving the bus dominant) ALOST is set. As long as ALOST is set, the TxD pin is forced to 1, resulting in a seized transmission.

In a slave SPI (MSTR = 0), MODF generates an SPI receiver/error CPU interrupt request if the ERRIE bit is set. The MODF bit does not clear the SPE bit or reset the SPI in any way. Software can abort the SPI transmission by clearing the SPE bit of the slave.

NOTE

A high on the $\overline{SS}$ pin of a slave SPI puts the MISO pin in a high impedance state. Also, the slave SPI ignores all incoming SPSCK clocks, even if it was already in the middle of a transmission.

To clear the MODF flag, read the SPSCR with the MODF bit set and then write to the SPCR register. This entire clearing mechanism must occur with no MODF condition existing or else the flag is not cleared.

15.7 Interrupts

Four SPI status flags can be enabled to generate CPU interrupt requests. See Table 15-1.

Table 15-1. SPI Interrupts

Flag	Request
SPTE Transmitter empty	SPI transmitter CPU interrupt request (SPTIE = 1, SPE = 1)
SPRF Receiver full	SPI receiver CPU interrupt request (SPRIE = 1)
OVRF Overflow	SPI receiver/error interrupt request (ERRIE = 1)
MODF Mode fault	SPI receiver/error interrupt request (ERRIE = 1)

Reading the SPI status and control register with SPRF set and then reading the receive data register clears SPRF. The clearing mechanism for the SPTE flag is always just a write to the transmit data register.

The SPI transmitter interrupt enable bit (SPTIE) enables the SPTE flag to generate transmitter CPU interrupt requests, provided that the SPI is enabled (SPE = 1).

The SPI receiver interrupt enable bit (SPRIE) enables SPRF to generate receiver CPU interrupt requests, regardless of the state of SPE. See Figure 15-12.

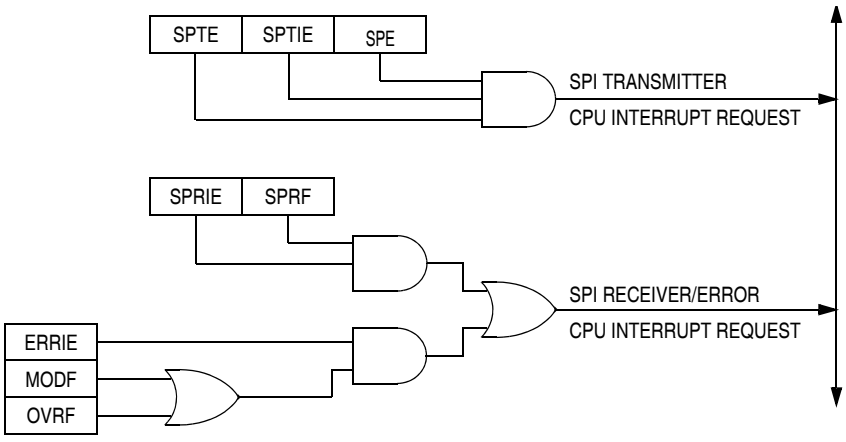


Figure 15-12. SPI Interrupt Request Generation

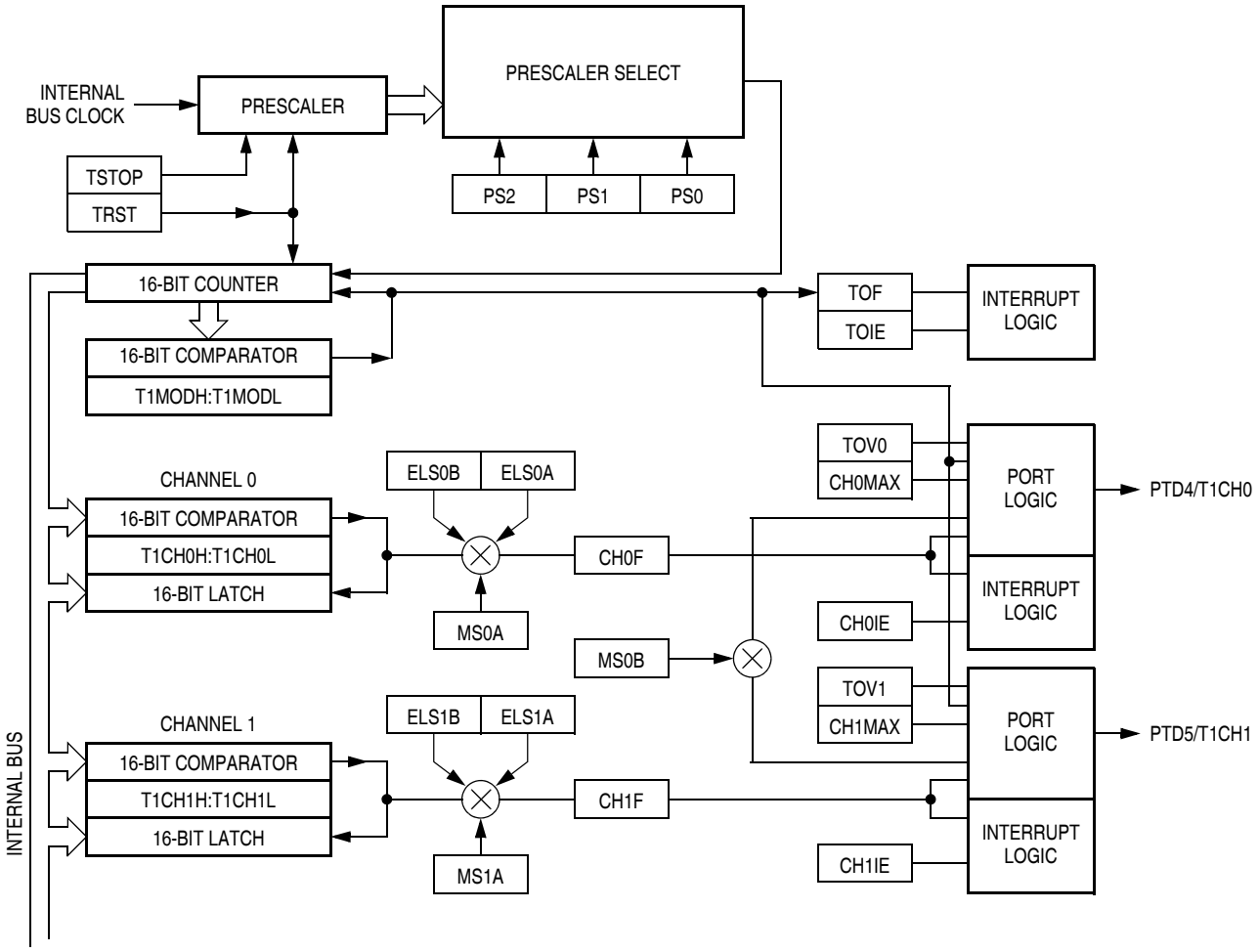


Figure 17-2. TIM1 Block Diagram

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0020	TIM1 Status and Control Register (T1SC) See page 234.	Read:	TOF	TOIE	TSTOP	0	0	PS2	PS1	PS0
		Write:	0			TRST				
		Reset:	0	0	1	0	0	0	0	0
$0021	TIM1 Counter Register High (T1CNTH) See page 235.	Read:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0022	TIM1 Counter Register Low (T1CNTL) See page 235.	Read:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		Write:								
		Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 17-3. TIM1 I/O Register Summary

17.8.3 TIM1 Counter Modulo Registers

The read/write TIM1 modulo registers contain the modulo value for the TIM1 counter. When the TIM1 counter reaches the modulo value, the overflow flag (TOF) becomes set, and the TIM1 counter resumes counting from $0000 at the next timer clock. Writing to the high byte (T1MODH) inhibits the TOF bit and overflow interrupts until the low byte (T1MODL) is written. Reset sets the TIM1 counter modulo registers.

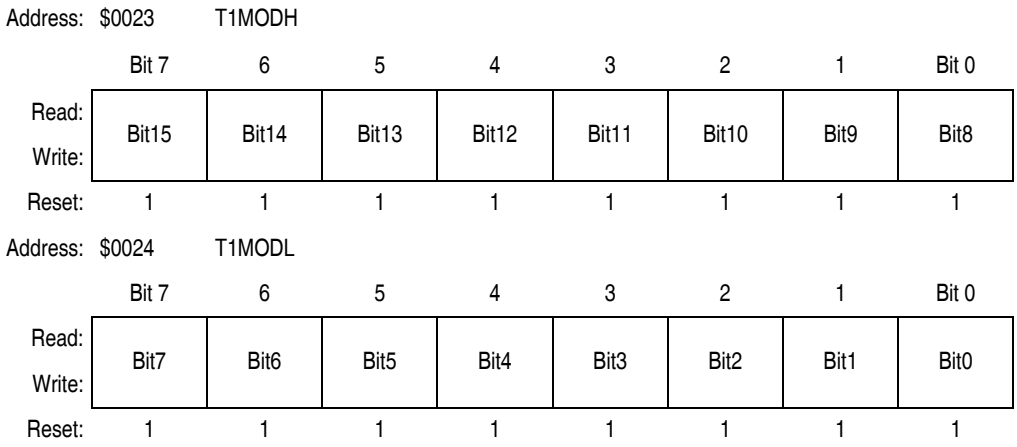


Figure 17-7. TIM1 Counter Modulo Registers (T1MODH:T1MODL)

NOTE

Reset the TIM1 counter before writing to the TIM1 counter modulo registers.

17.8.4 TIM1 Channel Status and Control Registers

Each of the TIM1 channel status and control registers does the following:

- Flags input captures and output compares
- Enables input capture and output compare interrupts
- Selects input capture, output compare, or PWM operation
- Selects high, low, or toggling output on output compare
- Selects rising edge, falling edge, or any edge as the active input capture trigger
- Selects output toggling on TIM1 overflow
- Selects 0% and 100% PWM duty cycle
- Selects buffered or unbuffered output compare/PWM operation

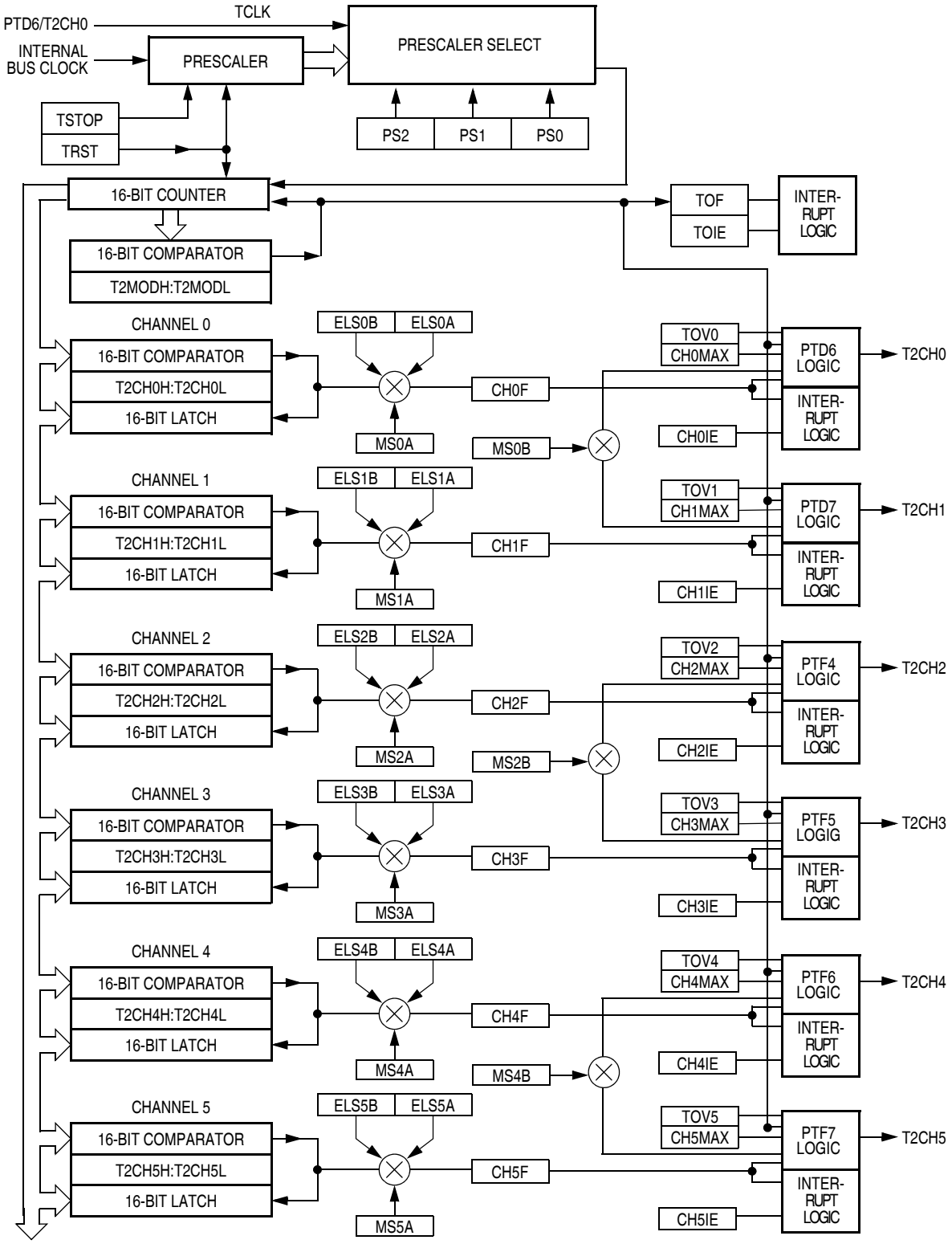


Figure 18-2. TIM2 Block Diagram

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0457	TIM2 Channel 2 Register High (T2CH2H) See page 258.	Read: Write:	Bit 15	14	13	12	11	10	9	Bit 8
		Reset:	Indeterminate after reset							
$0458	TIM2 Channel 2 Register Low (T2CH2L) See page 258.	Read: Write:	Bit 7	6	5	4	3	2	1	Bit 0
		Reset:	Indeterminate after reset							
$0459	TIM2 Channel 3 Status and Control Register (T2SC3) See page 255.	Read: Write:	CH3F 0	CH3IE	0	MS3A	ELS3B	ELS3A	TOV3	CH3MAX
		Reset:	0	0	0	0	0	0	0	0
$045A	TIM2 Channel 3 Register High (T2CH3H) See page 258.	Read: Write:	Bit 15	14	13	12	11	10	9	Bit 8
		Reset:	Indeterminate after reset							
$045B	TIM2 Channel 3 Register Low (T2CH3L) See page 258.	Read: Write:	Bit 7	6	5	4	3	2	1	Bit 0
		Reset:	Indeterminate after reset							
$045C	TIM2 Channel 4 Status and Control Register (T2SC4) See page 255.	Read: Write:	CH4F 0	CH4IE	MS4B	MS4A	ELS4B	ELS4A	TOV4	CH4MAX
		Reset:	0	0	0	0	0	0	0	0
$045D	TIM2 Channel 4 Register High (T2CH4H) See page 258.	Read: Write:	Bit 15	14	13	12	11	10	9	Bit 8
		Reset:	Indeterminate after reset							
$045E	TIM2 Channel 4 Register Low (T2CH4L) See page 258.	Read: Write:	Bit 7	6	5	4	3	2	1	Bit 0
		Reset:	Indeterminate after reset							
$045F	TIM2 Channel 5 Status and Control Register (T2SC5) See page 255.	Read: Write:	CH5F 0	CH5IE	0	MS5A	ELS5B	ELS5A	TOV5	CH5MAX
		Reset:	0	0	0	0	0	0	0	0
$0460	TIM2 Channel 5 Register High (T2CH5H) See page 258.	Read: Write:	Bit 15	14	13	12	11	10	9	Bit 8
		Reset:	Indeterminate after reset							
$0461	TIM2 Channel 5 Register Low (T2CH5L) See page 258.	Read: Write:	Bit 7	6	5	4	3	2	1	Bit 0
		Reset:	Indeterminate after reset							

= Unimplemented

Figure 18-3. TIM2 I/O Register Summary (Sheet 2 of 2)

18.3.1 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.

Address: $0034 T2CH1H

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
Reset:	Indeterminate after reset							

Address: $0035 T2CH1L

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Reset:	Indeterminate after reset							

Address: $0457 T2CH2H

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
Reset:	Indeterminate after reset							

Address: $0458 T2CH2L

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Reset:	Indeterminate after reset							

Address: $045A T2CH3H

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
Reset:	Indeterminate after reset							

Address: $045B T2CH3L

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Reset:	Indeterminate after reset							

Address: $045D T2CH4H

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
Reset:	Indeterminate after reset							

Address: $045E T2CH4L

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Reset:	Indeterminate after reset							

Figure 18-10. TIM2 Channel Registers (T2CH0H/L:T2CH5H/L) (Sheet 2 of 3)

Appendix B
MC68HC908GR32A

B.1 Introduction

The MC68HC908GR32A is a member of the low-cost, high-performance M68HC08 Family of 8-bit microcontroller units (MCUs). All MCUs in the family use the enhanced M68HC08 central processor unit (CPU08) and are available with a variety of modules, memory sizes and types, and package types.

The information contained in this document pertains to the MC68HC908GR32A with the exceptions shown in this appendix.

B.2 Block Diagram

See Figure B-1.

B.3 Memory

The MC68HC908GR32A can address 32 Kbytes of memory space. The memory map, shown in Figure B-2, includes:

- 32 Kbytes of user FLASH memory
- 1536 bytes of random-access memory (RAM)
- 52 bytes of user-defined vectors