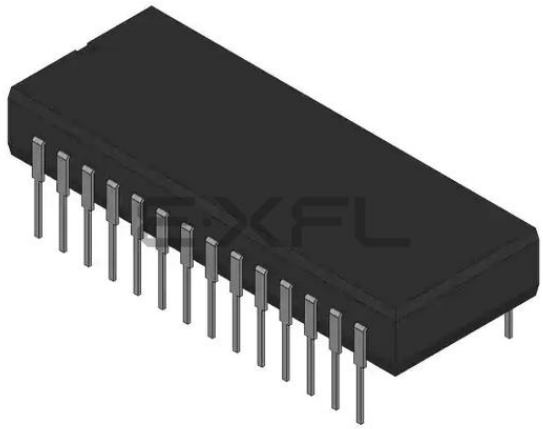


Welcome to E-XFL.COM

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	I²C, SCI
Peripherals	LED, LVD, POR, PWM
Number of I/O	23
Program Memory Size	16KB (16K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	512 x 8
Voltage - Supply (Vcc/Vdd)	2.7V ~ 5.5V
Data Converters	A/D 12x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Through Hole
Package / Case	28-DIP (0.600", 15.24mm)
Supplier Device Package	28-PDIP
Purchase URL	https://www.e-xfl.com/pro/item?MUrl=&PartUrl=mc908jl16cpe

- Master reset pin with internal pull-up and power-on reset
- $\overline{\text{IRQ}}$ with schmitt-trigger input and programmable pull up
- The MC68HC908JL16 is available in the following packages:
  - 28-pin plastic dual in-line package (PDIP)
  - 28-pin small outline integrated package (SOIC)
  - 32-pin shrink dual in-line package (SDIP)
  - 32-pin low-profile quad flat pack (LQFP)
- Specific features in 28-pin packages are:
  - 23 general-purpose I/Os only
  - 7 keyboard interrupt with internal pull up
  - 10 light-emitting diode (LED) drivers (sink)
  - 12-channel ADC
  - Timer I/O pins on TIM1 only

Features of the CPU08 include the following:

- Enhanced HC05 programming model
- Extensive loop control functions
- 16 addressing modes (eight more than the HC05)
- 16-bit index register and stack pointer
- Memory-to-memory data transfers
- Fast 8 $\times$ 8 multiply instruction
- Fast 16/8 divide instruction
- Binary-coded decimal (BCD) instructions
- Optimization for controller applications
- Efficient C language support

1.3 MCU Block Diagram

Figure 1-1 shows the structure of the MC68HC908JL16.

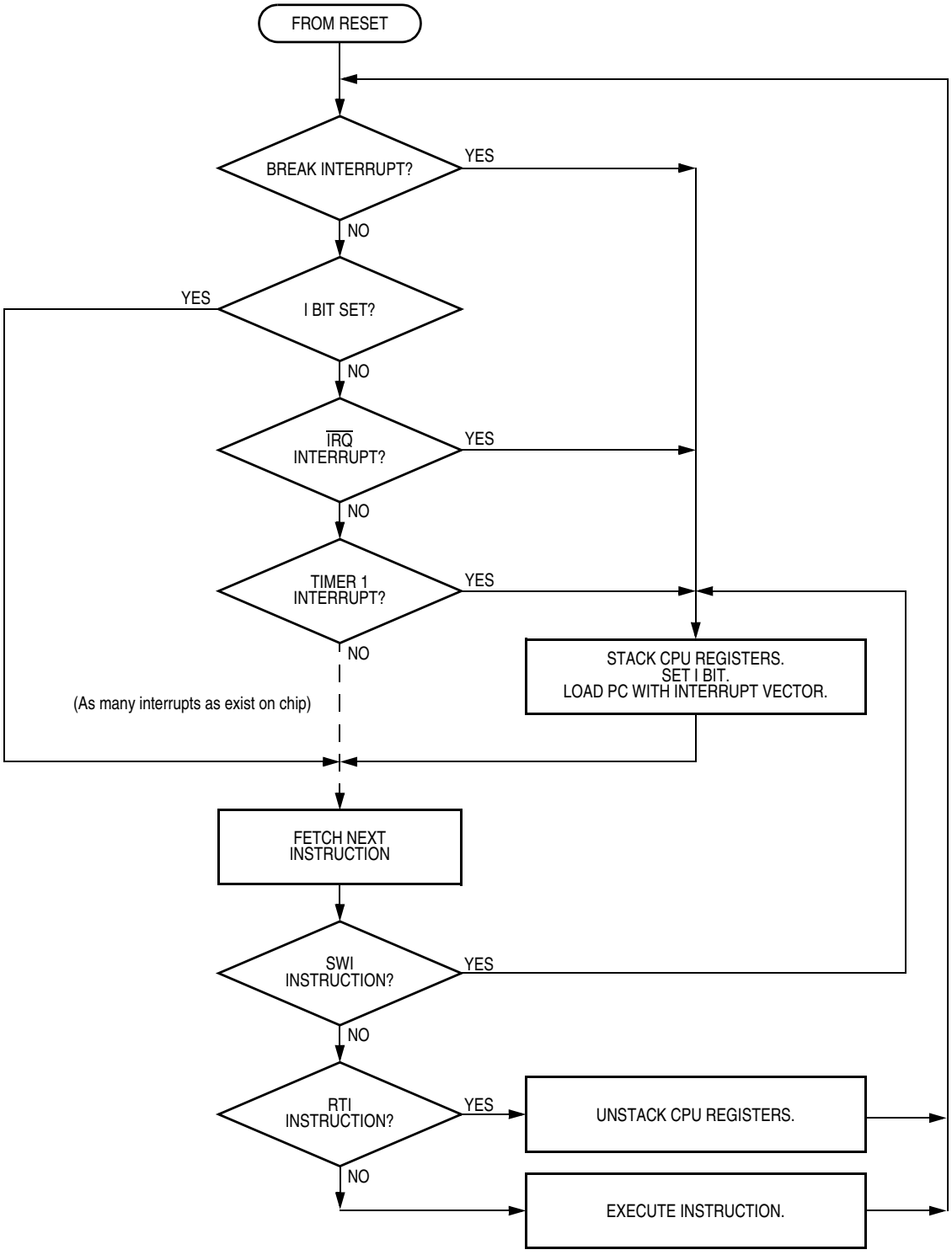


Figure 4-8. Interrupt Processing

Chapter 5
Oscillator (OSC)

5.1 Introduction

The oscillator module provides the reference clocks for the MCU system and bus. Two oscillators are running on the device:

Selectable oscillator — for bus clock

- Crystal oscillator (XTAL) — built-in oscillator that requires an external crystal or ceramic-resonator. This option also allows an external clock that can be driven directly into OSC1.
- RC oscillator (RC) — built-in oscillator that requires an external resistor-capacitor connection only.

The selected oscillator is used to drive the bus clock, the SIM, and other modules on the MCU. The oscillator type is selected by programming a bit FLASH memory. The RC and crystal oscillator cannot run concurrently; one is disabled while the other is selected; because the RC and XTAL circuits share the same OSC1 pin.

Non-selectable oscillator — for COP

- Internal oscillator — built-in RC oscillator that requires no external components.

This internal oscillator is used to drive the computer operating properly (COP) module and the SIM. The internal oscillator runs continuously after a POR or reset, and is always available.

5.2 Oscillator Selection

The oscillator type is selected by programming a bit in a FLASH memory location; the mask option register (MOR), at $FFD0. (See 3.5 Mask Option Register (MOR).)

NOTE

On the ROM device, the oscillator is selected by a ROM-mask layer at factory.

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0033	TIM2 Counter Modulo Register High (T2MODH)	Read: Write:	Bit 15	14	13	12	11	10	9	Bit 8
		Reset:	1	1	1	1	1	1	1	1
$0034	TIM2 Counter Modulo Register Low (T2MODL)	Read: Write:	Bit 7	6	5	4	3	2	1	Bit 0
		Reset:	1	1	1	1	1	1	1	1
$0035	TIM2 Channel 0 Status and Control Register (T2SC0)	Read: Write:	CH0F 0	CH0IE	MS0B	MS0A	ELS0B	ELS0A	TOV0	CH0MAX
		Reset:	0	0	0	0	0	0	0	0
$0036	TIM2 Channel 0 Register High (T2CH0H)	Read: Write:	Bit 15	14	13	12	11	10	9	Bit 8
		Reset:	Indeterminate after reset							
$0037	TIM2 Channel 0 Register Low (T2CH0L)	Read: Write:	Bit 7	6	5	4	3	2	1	Bit 0
		Reset:	Indeterminate after reset							
$0038	TIM2 Channel 1 Status and Control Register (T2SC1)	Read: Write:	CH1F 0	CH1IE	0 ▒	MS1A	ELS1B	ELS1A	TOV1	CH1MAX
		Reset:	0	0	0	0	0	0	0	0
$0039	TIM2 Channel 1 Register High (T2CH1H)	Read: Write:	Bit 15	14	13	12	11	10	9	Bit 8
		Reset:	Indeterminate after reset							
$003A	TIM2 Channel 1 Register Low (T2CH1L)	Read: Write:	Bit 7	6	5	4	3	2	1	Bit 0
		Reset:	Indeterminate after reset							

▒ = Unimplemented

Figure 6-2. TIM I/O Register Summary (Sheet 2 of 2)

6.4.1 TIM Counter Prescaler

The TIM1 clock source can be one of the seven prescaler outputs; TIM2 clock source can be one of the seven prescaler outputs or the TIM2 clock pin, T2CLK. The prescaler generates seven clock rates from the internal bus clock. The prescaler select bits, PS[2:0], in the TIM status and control register select the TIM clock source.

6.4.2 Input Capture

With the input capture function, the TIM can capture the time at which an external event occurs. When an active edge occurs on the pin of an input capture channel, the TIM latches the contents of the TIM counter into the TIM channel registers, TCHxH:TCHxL. The polarity of the active edge is programmable. Input captures can generate TIM CPU interrupt requests.

6.4.3 Output Compare

With the output compare function, the TIM 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 TIM can set, clear, or toggle the channel pin. Output compares can generate TIM CPU interrupt requests.

PS[2:0] — Prescaler Select Bits

These read/write bits select one of the seven prescaler outputs as the input to the TIM counter as Table 6-2 shows. Reset clears the PS[2:0] bits.

Table 6-2. Prescaler Selection

PS2	PS1	PS0	TIM Clock Source
0	0	0	Internal bus clock ÷ 1
0	0	1	Internal bus clock ÷ 2
0	1	0	Internal bus clock ÷ 4
0	1	1	Internal bus clock ÷ 8
1	0	0	Internal bus clock ÷ 16
1	0	1	Internal bus clock ÷ 32
1	1	0	Internal bus clock ÷ 64
1	1	1	T2CLK (for TIM2 only)

6.9.2 TIM Counter Registers

The two read-only TIM counter registers contain the high and low bytes of the value in the TIM counter. Reading the high byte (TCNTH) latches the contents of the low byte (TCNTL) into a buffer. Subsequent reads of TCNTH do not affect the latched TCNTL value until TCNTL is read. Reset clears the TIM counter registers. Setting the TIM reset bit (TRST) also clears the TIM counter registers.

NOTE

If you read TCNTH during a break interrupt, be sure to unlatch TCNTL by reading TCNTL before exiting the break interrupt. Otherwise, TCNTL retains the value latched during the break.

Address: T1CNTH, $0021 and T2CNTH, $0031

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

= Unimplemented

Figure 6-5. TIM Counter Registers High (TCNTH)

Address: T1CNTL, $0022 and T2CNTL, $0032

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

= Unimplemented

Figure 6-6. TIM Counter Registers Low (TCNTL)

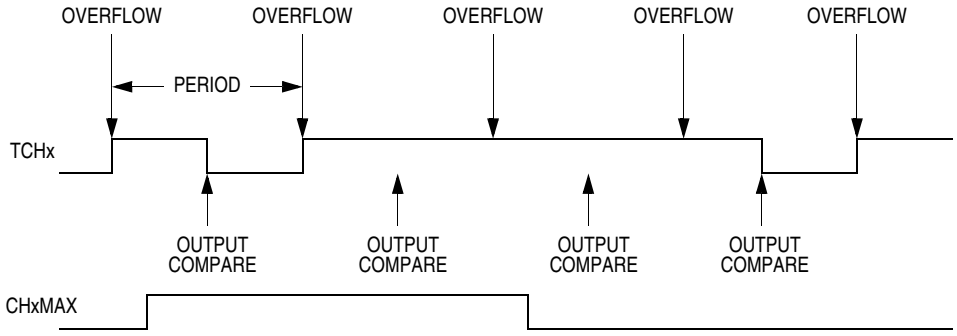


Figure 6-11. CHxMAX Latency

6.9.5 TIM Channel Registers

These read/write registers contain the captured TIM counter value of the input capture function or the output compare value of the output compare function. The state of the TIM channel registers after reset is unknown.

In input capture mode (MSxB:MSxA = 0:0), reading the high byte of the TIM channel x registers (TCHxH) inhibits input captures until the low byte (TCHxL) is read.

In output compare mode (MSxB:MSxA ≠ 0:0), writing to the high byte of the TIM channel x registers (TCHxH) inhibits output compares until the low byte (TCHxL) is written.

Address: T1CH0H, $0026 and T2CH0H, $0036

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

Figure 6-12. TIM Channel 0 Register High (TCH0H)

Address: T1CH0L, $0027 and T2CH0L $0037

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

Figure 6-13. TIM Channel 0 Register Low (TCH0L)

Address: T1CH1H, $0029 and T2CH1H, $0039

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

Figure 6-14. TIM Channel 1 Register High (TCH1H)

7.3 Pin Name Conventions

The generic names of the SCI I/O pins are:

- RxD (receive data)
- TxD (transmit data)

The SCI I/O (input/output) lines are dedicated pins for the SCI module. Table 7-1 shows the full names and the generic names of the SCI I/O pins.

The generic pin names appear in the text of this section.

Table 7-1. Pin Name Conventions

Generic Pin Names:	RxD	TxD
Full Pin Names:	PTD7/RxD/SDA(1)	PTD6/TxD/SCL(1)

1. Position of MMIIC module pins (SDA and SCL) is user selectable using CONFIG2 option bit. Refer to Chapter 3 Configuration and Mask Option Registers (CONFIG and MOR) for additional information. SDA/SCL have priority over the RxD/TxD when MMIIC is enabled and using PTD7/PTD6 for its pins. For more information on MMIIC, (see Chapter 8 Multi-Master IIC Interface (MMIIC)).

7.4 Functional Description

Figure 7-2 shows the structure of the SCI module. The SCI allows full-duplex, asynchronous, NRZ serial communication among the MCU and remote devices, including other MCUs. The transmitter and receiver of the SCI operate independently, although they use the same baud rate generator. During normal operation, the CPU monitors the status of the SCI, writes the data to be transmitted, and processes received data. The baud rate clock source for the SCI is the bus clock.

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0013	SCI Control Register 1 (SCC1)	Read: Write:	LOOPS	ENSCI	TXINV	M	WAKE	ILTY	PEN	PTY
		Reset:	0	0	0	0	0	0	0	0
$0014	SCI Control Register 2 (SCC2)	Read: Write:	SCTIE	TCIE	SCRIE	ILIE	TE	RE	RWU	SBK
		Reset:	0	0	0	0	0	0	0	0
$0015	SCI Control Register 3 (SCC3)	Read: Write:	R8	T8	DMARE	DMATE	ORIE	NEIE	FEIE	PEIE
		Reset:	U	U	0	0	0	0	0	0
$0016	SCI Status Register 1 (SCS1)	Read: Write:	SCTE	TC	SCRF	IDLE	OR	NF	FE	PE
		Reset:	1	1	0	0	0	0	0	0
$0017	SCI Status Register 2 (SCS2)	Read: Write:							BKF	RPF
		Reset:	0	0	0	0	0	0	0	0
$0018	SCI Data Register (SCDR)	Read:	R7	R6	R5	R4	R3	R2	R1	R0
		Write:	T7	T6	T5	T4	T3	T2	T1	T0
		Reset:	Unaffected by reset							
$0019	SCI Baud Rate Register (SCBR)	Read: Write:	0	0	SCP1	SCP0	R	SCR2	SCR1	SCR0
		Reset:	0	0	0	0	0	0	0	0

= Unimplemented R = Reserved U = Unaffected

Figure 7-1. SCI I/O Register Summary

Table 7-8. SCI Baud Rate Selection Examples

SCP1 and SCP0	Prescaler Divisor (PD)	SCR2, SCR1, and SCR0	Baud Rate Divisor (BD)	Baud Rate (BUS CLOCK= 4.9152 MHz)
00	1	000	1	76,800
00	1	001	2	38,400
00	1	010	4	19,200
00	1	011	8	9,600
00	1	100	16	4,800
00	1	101	32	2,400
00	1	110	64	1,200
00	1	111	128	600
01	3	000	1	25,600
01	3	001	2	12,800
01	3	010	4	6,400
01	3	011	8	3,200
01	3	100	16	1,600
01	3	101	32	800
01	3	110	64	400
01	3	111	128	200
10	4	000	1	19,200
10	4	001	2	9,600
10	4	010	4	4,800
10	4	011	8	2,400
10	4	100	16	1,200
10	4	101	32	600
10	4	110	64	300
10	4	111	128	150
11	13	000	1	5,908
11	13	001	2	2,954
11	13	010	4	1,477
11	13	011	8	739
11	13	100	16	369
11	13	101	32	185
11	13	110	64	92
11	13	111	128	46

The MMIIC module will respond to the calling address:

Bit 7	6	5	4	3	2	Bit 1
1	1	0	1	0	1	0

or the general calling address:

0	0	0	0	0	0	0

or the calling address:

0	0	0	1	1	0	0

NOTE

Bit 0 of the 8-bit calling address is the MMRW bit from the calling master.

8.8.2 Multi-Master IIC Control Register (MMCR)

Address: $0042

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	MMEN	MMIEN	0	0	MMTXAK	REPSEN	0	0
Write:								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 8-5. Multi-Master IIC Control Register (MMCR)

MMEN — Multi-Master IIC Enable

This bit is set to enable the Multi-master IIC module. When MMEN = 0, module is disabled and all flags will restore to its power-on default states. Reset clears this bit.

1 = MMIIC module enabled
0 = MMIIC module disabled

MMIEN — Multi-Master IIC Interrupt Enable

When this bit is set, the MMTXIF, MMRXIF, MMALIF, and MMNAKIF flags are enabled to generate an interrupt request to the CPU. When MMIEN is cleared, the these flags are prevented from generating an interrupt request. Reset clears this bit.

1 = MMTXIF, MMRXIF, MMALIF, and/or MMNAKIF bit set will generate interrupt request to CPU
0 = MMTXIF, MMRXIF, MMALIF, and/or MMNAKIF bit set will not generate interrupt request to CPU

MMTXAK — Transmit Acknowledge Enable

This bit is set to disable the MMIIC from sending out an acknowledge signal to the bus at the 9th clock bit after receiving 8 data bits. When MMTXAK is cleared, an acknowledge signal will be sent at the 9th clock bit. Reset clears this bit.

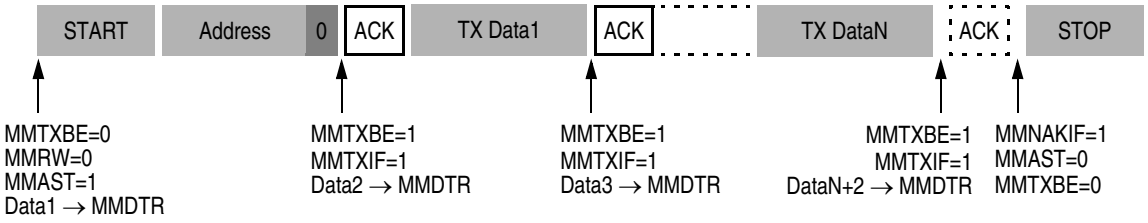
1 = MMIIC does not send acknowledge signals at 9th clock bit
0 = MMIIC sends acknowledge signal at 9th clock bit

REPSEN — Repeated Start Enable

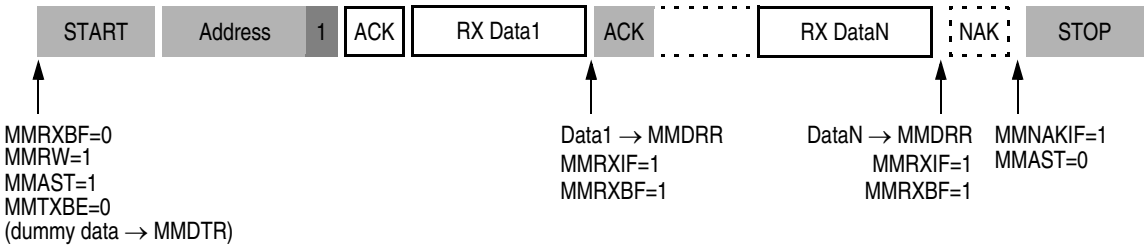
This bit is set to enable repeated START signal to be generated when in master mode transfer (MMAST = 1). The REPSEN bit is cleared by hardware after the completion of repeated START signal or when the MMAST bit is cleared. Reset clears this bit.

1 = Repeated START signal will be generated if MMAST bit is set
0 = No repeated START signal will be generated

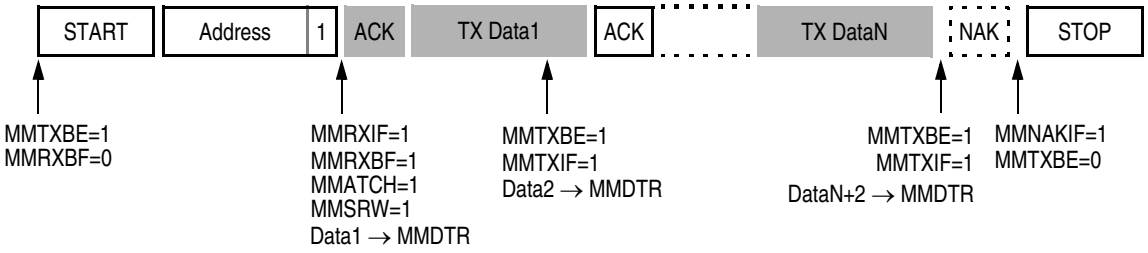
(a) Master Transmit Mode



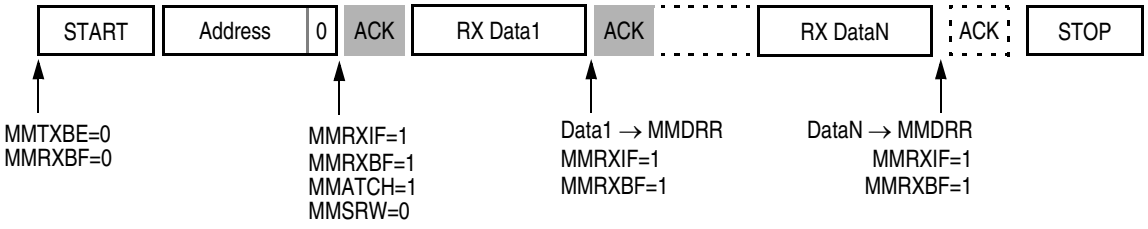
(b) Master Receive Mode



(c) Slave Transmit Mode



(d) Slave Receive Mode



Shaded data packets indicate transmissions by the MCU

Figure 8-10. Data Transfer Sequences for Master/Slave Transmit/Receive Modes

A blocking mechanism prevents a new result from overwriting previous data in ADRH and ADRL if the previous data is in the process of being read while in 10-bit mode (ADRH has been read but ADRL has not). In this case the data transfer is blocked, COCO is not set, and the new result is lost. When a data transfer is blocked, another conversion is initiated regardless of the state of ADCO (single or continuous conversions enabled). If single conversions are enabled, this could result in several discarded conversions and excess power consumption. To avoid this issue, the data registers must not be read after initiating a single conversion until the conversion completes.

9.3.3.3 Aborting Conversions

Any conversion in progress will be aborted when:

- A write to ADCSC occurs (the current conversion will be aborted and a new conversion will be initiated, if ADCH are not all 1s).
- A write to ADCLK occurs.
- The MCU is reset.
- The MCU enters stop mode with ACLK not enabled.

When a conversion is aborted, the contents of the data registers, ADRH and ADRL, are not altered but continue to be the values transferred after the completion of the last successful conversion. In the case that the conversion was aborted by a reset, ADRH and ADRL return to their reset states.

Upon reset or when a conversion is otherwise aborted, the ADC10 module will enter a low power, inactive state. In this state, all internal clocks and references are disabled. This state is entered asynchronously and immediately upon aborting of a conversion.

9.3.3.4 Total Conversion Time

The total conversion time depends on many factors such as sample time, bus frequency, whether ACLKEN is set, and synchronization time. The total conversion time is summarized in Table 9-1.

Table 9-1. Total Conversion Time versus Control Conditions

Conversion Mode	ACLKEN	Maximum Conversion Time
8-Bit Mode (short sample — ADLSMP = 0):		
Single or 1st continuous	0	18 ADCK + 3 bus clock
Single or 1st continuous	1	18 ADCK + 3 bus clock + 5 µs
Subsequent continuous ($f_{Bus} \geq f_{ADCK}$)	X	16 ADCK
8-Bit Mode (long sample — ADLSMP = 1):		
Single or 1st continuous	0	38 ADCK + 3 bus clock
Single or 1st continuous	1	38 ADCK + 3 bus clock + 5 µs
Subsequent continuous ($f_{Bus} \geq f_{ADCK}$)	X	36 ADCK
10-Bit Mode (short sample — ADLSMP = 0):		
Single or 1st continuous	0	21 ADCK + 3 bus clock
Single or 1st continuous	1	21 ADCK + 3 bus clock + 5 µs
Subsequent continuous ($f_{Bus} \geq f_{ADCK}$)	X	19 ADCK
10-Bit Mode (long sample — ADLSMP = 1):		
Single or 1st continuous	0	41 ADCK + 3 bus clock
Single or 1st continuous	1	41 ADCK + 3 bus clock + 5 µs
Subsequent continuous ($f_{Bus} \geq f_{ADCK}$)	X	39 ADCK

10.4.1 Port D Data Register (PTD)

The port D data register contains a data latch for each of the eight port D pins.

Address: $0003

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	PTD7	PTD6	PTD5	PTD4	PTD3	PTD2	PTD1	PTD0
Reset:	Unaffected by reset							
Additional Functions	LED (Sink)	LED (Sink)			LED (Sink)	LED (Sink)		
	25mA sink (Slow Edge)	25mA sink (Slow Edge)						
	pull-up	pull-up						
Alternative Functions:	RxD	TxD	T1CH1	T1CH0	ADC8	ADC9	ADC10	ADC11
Alternative Functions:	SDA	SCL						

= Unimplemented

Figure 10-10. Port D Data Register (PTD)

PTD[7:0] — Port D Data Bits

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

ADC11–ADC8 — ADC channels 11 to 8

ADC[11:8] are pins used for the input channels to the analog-to-digital converter module. The channel select bits, ADCH[4:0], in the ADC status and control register define which port pin will be used as an ADC input. See Chapter 9 Analog-to-Digital Converter (ADC).

NOTE

When a pin is to be used as an ADC channel, the user must make sure that any pin that is shared with another module is disabled and pin is configured as input port.

T1CH1, T1CH0 — Timer 1 Channel I/Os

The T1CH1 and T1CH0 pins are the TIM1 input capture/output compare pins. The edge/level select bits, ELSxB:ELSxA, determine whether the PTD4/T1CH0 and PTD5/T1CH1 pins are timer channel I/O pins or general-purpose I/O pins. See Chapter 6 Timer Interface Module (TIM).

TxD, RxD — SCI Data I/O Pins

The TxD and RxD pins are the transmit data output and receive data input for the SCI module. The enable SCI bit, ENSCI, in the SCI control register 1 enables the PTD6/TxD and PTD7/RxD pins as SCI TxD and RxD pins and overrides any control from the port I/O logic. See Chapter 7 Serial Communications Interface (SCI).

SDA and SCL — MMIIC Module Pins

The MMIIC pins can be configured to use PTD6 and PTD7 as IIC communication pins, see Chapter 8 Multi-Master IIC Interface (MMIIC). The position of MMIIC module pins is user selectable using CONFIG2 option bit, to allow PTD6/PTD7 to be MMIIC pins (see Figure 3-3. Configuration Register 2 (CONFIG2)).

10.4.2 Data Direction Register D (DDRD)

Data direction register D determines whether each port D pin is an input or an output. Writing a logic 1 to a DDRD bit enables the output buffer for the corresponding port D pin; a logic 0 disables the output buffer.

Address: $0007

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	DDRD7	DDRD6	DDRD5	DDRD4	DDRD3	DDRD2	DDRD1	DDRD0
Reset:	0	0	0	0	0	0	0	0

Figure 10-11. Data Direction Register D (DDRD)

DDRD[7:0] — Data Direction Register D Bits

These read/write bits control port D data direction. Reset clears DDRD[7:0], configuring all port D pins as inputs.

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

NOTE

Avoid glitches on port D pins by writing to the port D data register before changing data direction register D bits from 0 to 1. Figure 10-12 shows the port D I/O logic.

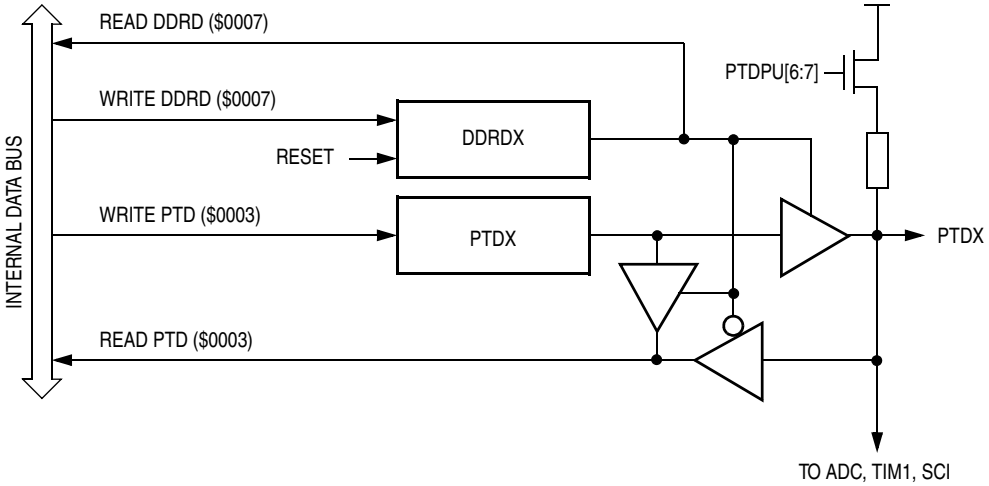


Figure 10-12. Port D I/O Circuit

When DDRDx is a logic 1, reading address $0003 reads the PTDx data latch. When DDRDx is a logic 0, reading address $0003 reads the voltage level on the pin. The data latch can always be written, regardless of the state of its data direction bit.

Chapter 15
Central Processor Unit (CPU)

15.1 Introduction

The M68HC08 CPU (central processor unit) is an enhanced and fully object-code-compatible version of the M68HC05 CPU. The *CPU08 Reference Manual* (document order number CPU08RM/AD) contains a description of the CPU instruction set, addressing modes, and architecture.

15.2 Features

Features of the CPU include:

- Object code fully upward-compatible with M68HC05 Family
- 16-bit stack pointer with stack manipulation instructions
- 16-bit index register with x-register manipulation instructions
- 8-MHz CPU internal bus frequency
- 64-Kbyte program/data memory space
- 16 addressing modes
- Memory-to-memory data moves without using accumulator
- Fast 8-bit by 8-bit multiply and 16-bit by 8-bit divide instructions
- Enhanced binary-coded decimal (BCD) data handling
- Modular architecture with expandable internal bus definition for extension of addressing range beyond 64 Kbytes
- Low-power stop and wait modes

15.3 CPU Registers

Figure 15-1 shows the five CPU registers. CPU registers are not part of the memory map.

15.7 Instruction Set Summary

Table 15-1 provides a summary of the M68HC08 instruction set.

Table 15-1. Instruction Set Summary (Sheet 1 of 6)

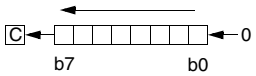
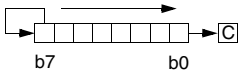
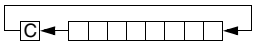
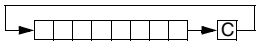
Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Cycles
			V	H	I	N	Z	C				
ADC #*opr* ADC *opr* ADC *opr* ADC *opr*,X ADC *opr*,X ADC ,X ADC *opr*,SP ADC *opr*,SP	Add with Carry	A ← (A) + (M) + (C)	↕	↕	–	↕	↕	↕	IMM DIR EXT IX2 IX1 IX SP1 SP2	A9 B9 C9 D9 E9 F9 9EE9 9ED9	ii dd hh ll ee ff ff ff ee ff	2 3 4 4 3 2 4 5
ADD #*opr* ADD *opr* ADD *opr* ADD *opr*,X ADD *opr*,X ADD ,X ADD *opr*,SP ADD *opr*,SP	Add without Carry	A ← (A) + (M)	↕	↕	–	↕	↕	↕	IMM DIR EXT IX2 IX1 IX SP1 SP2	AB BB CB DB EB FB 9EEB 9EDB	ii dd hh ll ee ff ff ff ee ff	2 3 4 4 3 2 4 5
AIS #*opr*	Add Immediate Value (Signed) to SP	SP ← (SP) + (16 « M)	–	–	–	–	–	–	IMM	A7	ii	2
AIX #*opr*	Add Immediate Value (Signed) to H:X	H:X ← (H:X) + (16 « M)	–	–	–	–	–	–	IMM	AF	ii	2
AND #*opr* AND *opr* AND *opr* AND *opr*,X AND *opr*,X AND ,X AND *opr*,SP AND *opr*,SP	Logical AND	A ← (A) & (M)	0	–	–	↕	↕	–	IMM DIR EXT IX2 IX1 IX SP1 SP2	A4 B4 C4 D4 E4 F4 9EE4 9ED4	ii dd hh ll ee ff ff ff ee ff	2 3 4 4 3 2 4 5
ASL *opr* ASLA ASLX ASL *opr*,X ASL ,X ASL *opr*,SP	Arithmetic Shift Left (Same as LSL)		↕	–	–	↕	↕	↕	DIR INH INH IX1 IX SP1	38 48 58 68 78 9E68	dd ff ff	4 1 1 4 3 5
ASR *opr* ASRA ASRX ASR *opr*,X ASR *opr*,X ASR *opr*,SP	Arithmetic Shift Right		↕	–	–	↕	↕	↕	DIR INH INH IX1 IX SP1	37 47 57 67 77 9E67	dd ff ff	4 1 1 4 3 5
BCC *rel*	Branch if Carry Bit Clear	PC ← (PC) + 2 + rel ? (C) = 0	–	–	–	–	–	–	REL	24	rr	3
BCLR *n*, *opr*	Clear Bit n in M	Mn ← 0	–	–	–	–	–	–	DIR (b0) DIR (b1) DIR (b2) DIR (b3) DIR (b4) DIR (b5) DIR (b6) DIR (b7)	11 13 15 17 19 1B 1D 1F	dd dd dd dd dd dd dd dd	4 4 4 4 4 4 4 4
BCS *rel*	Branch if Carry Bit Set (Same as BLO)	PC ← (PC) + 2 + *rel* ? (C) = 1	–	–	–	–	–	–	REL	25	rr	3
BEQ *rel*	Branch if Equal	PC ← (PC) + 2 + *rel* ? (Z) = 1	–	–	–	–	–	–	REL	27	rr	3
BGE *opr*	Branch if Greater Than or Equal To (Signed Operands)	PC ← (PC) + 2 + *rel* ? (N ⊕ V) = 0	–	–	–	–	–	–	REL	90	rr	3
BGT *opr*	Branch if Greater Than (Signed Operands)	PC ← (PC) + 2 + *rel* ? (Z) ǀ (N ⊕ V) = 0	–	–	–	–	–	–	REL	92	rr	3
BHCC *rel*	Branch if Half Carry Bit Clear	PC ← (PC) + 2 + *rel* ? (H) = 0	–	–	–	–	–	–	REL	28	rr	3
BHCS *rel*	Branch if Half Carry Bit Set	PC ← (PC) + 2 + *rel* ? (H) = 1	–	–	–	–	–	–	REL	29	rr	3
BHI *rel*	Branch if Higher	PC ← (PC) + 2 + *rel* ? (C) ǀ (Z) = 0	–	–	–	–	–	–	REL	22	rr	3

Table 15-1. 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

The control and data bytes are described below.

- **Bus speed** — This one byte indicates the operating bus speed of the MCU. The value of this byte should be the nearest integer of the bus speed (in MHz) times 4, and should not be set to less than 4 (i.e. minimum bus speed is 1 MHz).
- **Data size** — This one byte indicates the number of bytes in the data array that are to be manipulated. The maximum data array size is 128. Routines EE_WRITE and EE_READ are restricted to manipulate a data array between 2 to 15 bytes. Whereas routines ERARNGE and MON_ERARNGE do not manipulate a data array, thus, this data size byte has no meaning.
- **Start address** — These two bytes, high byte followed by low byte, indicate the start address of the FLASH memory to be manipulated.
- **Data array** — This data array contains data that are to be manipulated. Data in this array are programmed to FLASH memory by the programming routines: PRGRNGE, MON_PRGRNGE, EE_WRITE. For the read routines: LDRNGE, MON_LDRNGE, and EE_READ, data is read from FLASH and stored in this array.

16.3.9.1 PRGRNGE

PRGRNGE is used to program a range of FLASH locations with data loaded into the data array.

Table 16-11. PRGRNGE Routine

Routine Name	PRGRNGE
Routine Description	Program a range of locations
Calling Address	$FC06
Stack Used	11 bytes
Data Block Format	Bus speed (BUS_SPD) Data size (DATASIZE) Start address high (ADDRH) Start address (ADDRL) Data 1 (DATA1) : Data N (DATAN)

The start location of the FLASH to be programmed is specified by the address ADDRH:ADDRL and the number of bytes to be programmed is specified by DATASIZE. The maximum number of bytes that can be programmed in one routine call is 128 bytes (max. DATASIZE is 128).

ADDRH:ADDRL do not need to be at a page boundary, the routine handles any boundary misalignment during programming. User software must ensure that the selected range is first erase. User software is also responsible for verifying programmed locations.

The coding example below is to program 32 bytes of data starting at FLASH location $EF00, with a bus speed of 4.9152 MHz. The coding assumes the data block is already loaded in RAM, with the address pointer, FILE_PTR, pointing to the first byte of the data block.

```
                ORG     RAM
                :
FILE_PTR:
BUS_SPD         DS.B    1       ; Indicates 4x bus frequency
DATASIZE        DS.B    1       ; Data size to be programmed
START_ADDR      DS.W    1       ; FLASH start address
DATAARRAY       DS.B    32      ; Reserved data array
```

```
                ORG     RAM
                :
FILE_PTR:
BUS_SPD         DS.B    1       ; Indicates 4x bus frequency
DATASIZE        DS.B    1       ; Data size to be programmed
START_ADDR      DS.W    1       ; FLASH page start address
DATAARRAY       DS.B    15      ; Reserved data array

EE_WRITE        EQU     $FD3F
FLASH_START     EQU     $EF00

                ORG     FLASH
INITIALISATION:
        MOV     #20,BUS_SPD
        MOV     #15,DATASIZE
        LDHX    #FLASH_START
        STHX    START_ADDR
        RTS
MAIN:
        BSR     INITIALISATION
        :
        :
        LHDX    #FILE_PTR
        JSR     EE_WRITE
```

NOTE

The EE_WRITE routine is unable to check for incorrect data blocks, such as the FLASH page boundary address and data size. It is the responsibility of the user to ensure the starting address indicated in the data block is at the FLASH page boundary and the data size is 2 to 15. When the EE_WRITE routine detects a different data size from the size set up in the previous operation, the operation will not be executed. However in some situations, the routine cannot detect incorrect data size. The user must ensure that data size is same as the previous operation whenever this routine is executed.