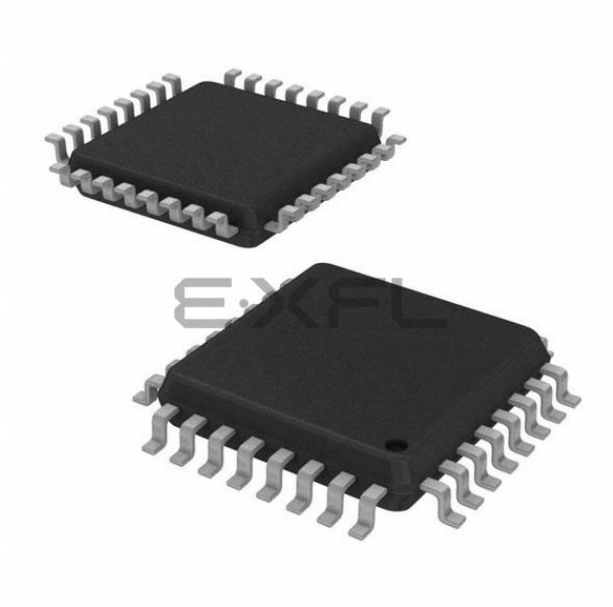


Welcome to E-XFL.COM

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	LINbus, SCI, SPI
Peripherals	POR, PWM
Number of I/O	24
Program Memory Size	16KB (16K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	512 x 8
Voltage - Supply (Vcc/Vdd)	4.5V ~ 5.5V
Data Converters	A/D 8x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 125°C (TA)
Mounting Type	Surface Mount
Package / Case	32-LQFP
Supplier Device Package	32-LQFP (7x7)
Purchase URL	https://www.e-xfl.com/product-detail/nxp-semiconductors/mc68hc908ey16mfa

1.5.6 Port A I/O Pins (PTA6/$\overline{SS}$, PTA5/SPSCK, PTA4/$\overline{KBD4}$–PTA0/$\overline{KBD0}$)

Port A input/output (I/O) pins (PTA6/$\overline{SS}$, PTA5/SPSCK, PTA4/$\overline{KBD4}$, PTA3/$\overline{KBD3}$, PTA2/$\overline{KBD2}$, PTA1/$\overline{KBD1}$, and PTA0/$\overline{KBD0}$) are special-function, bidirectional I/O port pins. PTA5 and PTA6 are shared with the serial peripheral interface (SPI). PTA4-PTA0 can be programmed to serve as keyboard interrupt pins.

See Chapter 12 Input/Output (I/O) Ports (PORTS) and Chapter 9 External Interrupt (IRQ).

1.5.7 Port B I/O Pins (PTB7/AD7/TBCH1, PTB6/AD6/TBCH0, PTB5/AD5–PTB0/AD0)

PTB7/AD7/TBCH1, PTB6/AD6/TBCH0, and PTB5/AD5–PTB0/AD0 are special-function, bidirectional I/O port pins that can also be used for ADC inputs. PTB7/AD7/TBCH1 and PTB6/AD6/TBCH0 are special function bidirectional I/O port pins that can also be used for timer interface pins.

See and Chapter 3 Analog-to-Digital Converter (ADC) Module and Chapter 17 Timer Interface A (TIMA) Module.

1.5.8 Port C I/O Pins (PTC4/OSC1, PTC3/OSC2, PTC2/MCLK, PTC1/MOSI, PTC0/MISO)

PTC4/OSC1–PTC0/MISO are special-function, bidirectional I/O port pins. See Chapter 12 Input/Output (I/O) Ports (PORTS). PTC3/OSC2 and PTC4/OSC1 are shared with the on-chip oscillator circuit through configuration options. See Chapter 8 Internal Clock Generator (ICG) Module.

When applications require:

- PTC3/OSC2 can be programmed to be OSC2
- PTC4/OSC1 can be programmed to be OSC1

PTC2/MCLK is software selectable to be MCLK, or bus clock out. PTC1/MOSI can be programmed to be the MOSI signal for the SPI. PTC0/MISO can be programmed to be the MISO signal for the SPI.

1.5.9 Port D I/O Pins (PTD1/TACH1–PTD0/TACH0)

PTD1/TACH1–PTD0/TACH0 are special-function, bidirectional I/O port pins that can also be programmed to be timer pins.

See Chapter 17 Timer Interface A (TIMA) Module and Chapter 12 Input/Output (I/O) Ports (PORTS).

1.5.10 Port E I/O Pins (PTE1/RxD–PTE0/TxD)

PTE1/RxD–PTE0/TxD are special-function, bidirectional I/O port pins that can also be programmed to be enhanced serial communication interface (ESCI) pins.

See Chapter 13 Enhanced Serial Communications Interface (ESCI) Module and Chapter 12 Input/Output (I/O) Ports (PORTS).

NOTE

Any unused inputs and I/O ports should be tied to an appropriate logic level (either V_{DD} or V_{SS}). Although the I/O ports of the MC68HC908EY16 do not require termination, termination is recommended to reduce the possibility of electro-static discharge damage.

Chapter 2
Memory

2.1 Introduction

The M68HC08 central processor unit (CPU08) can address 64 Kbytes of memory space. The memory map, shown in Figure 2-1, includes:

- 16 Kbytes of FLASH memory, 15, 872 bytes of user space
- 512 bytes of random-access memory (RAM)
- 36 bytes of user-defined vectors
- 310 bytes of monitor routines in read-only memory (ROM)
- 1024 bytes of integrated FLASH burn-in routines in ROM

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 unit (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. Additional I/O registers have these addresses:

- $FE00; SIM break status register, SBSR
- $FE01; SIM reset status register, SRSR
- $FE03; SIM break flag control register, SBFCR
- $FE08; FLASH control register, FLCR
- $FE09; break address register high, BRKH
- $FE0A; break address register low, BRKL
- $FE0B; break status and control register, BRKSCR
- $FE0C; LVI status register, LVISR
- $FF7E; FLASH block protect register, FLBPR
- $FF80; ICG trim value (optional), ICGT

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

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$000C	Reserved		R	R	R	R	R	R	R	R
$000D	SPI Control Register (SPCR) See page 184.	Read:	SPRIE	R	SPMSTR	CPOL	CPHA	SPWOM	SPE	SPTIE
		Write:								
		Reset:	0	0	1	0	1	0	0	0
$000E	SPI Status and Control Register (SPSCR) See page 186.	Read:	SPRF	ERRIE	OVRF	MODF	SPTE	MODFEN	SPR1	SPR0
		Write:								
		Reset:	0	0	0	0	1	0	0	0
$000F	SPI Data Register (SPDR) See page 188.	Read:	R7	R6	R5	R4	R3	R2	R1	R0
		Write:	T7	T6	T5	T4	T3	T2	T1	T0
		Reset:	Indeterminate after reset							
$0010	ESCI Control Register 1 (SCC1) See page 138.	Read:	LOOPS	ENSCI	TXINV	M	WAKE	ILTY	PEN	PTY
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0011	ESCI Control Register 2 (SCC2) See page 140.	Read:	SCTIE	TCIE	SCRIE	ILIE	TE	RE	RWU	SBK
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0012	ESCI Control Register 3 (SCC3) See page 142.	Read:	R8	T8	R	R	ORIE	NEIE	FEIE	PEIE
		Write:								
		Reset:	U	0	0	0	0	0	0	0
$0013	ESCI Status Register 1 (SCS1) See page 143.	Read:	SCTE	TC	SCRF	IDLE	OR	NF	FE	PE
		Write:								
		Reset:	1	1	0	0	0	0	0	0
$0014	ESCI Status Register 2 (SCS2) See page 145.	Read:	0	0	0	0	0	0	BKF	RPF
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0015	ESCI Data Register (SCDR) See page 146.	Read:	R7	R6	R5	R4	R3	R2	R1	R0
		Write:	T7	T6	T5	T4	T3	T2	T1	T0
		Reset:	Unaffected by reset							
$0016	ESCI Baud Rate Register (SCBR) See page 146.	Read:	R	LINR	SCP1	SCP0	R	SCR2	SCR1	SCR0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0017	ESCI Prescale Register (SCPSC) See page 147.	Read:	PDS2	PDS1	PDS0	PSSB4	PSSB3	PSSB2	PSSB1	PSSB0
		Write:								
		Reset:	0	0	0	0	0	0	0	0

= Unimplemented R = Reserved U = Unaffected

Figure 2-2. Control, Status, and Data Registers (Sheet 2 of 7)

2.6.4 FLASH Program/Read Operation

Programming of the FLASH memory is done on a row basis. A row consists of 32 consecutive bytes starting from addresses $XX00, $XX20, $XX40, $XX60, $XX80, $XXA0, $XXC0, and $XXE0. Use this step-by-step procedure to program a row of FLASH memory (Figure 2-4 is a flowchart representation).

NOTE

To avoid program disturbs, the row must be erased before any byte on that row is programmed.

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

This program sequence is repeated throughout the memory until all data is programmed.

NOTE

Programming and erasing of FLASH locations cannot be performed by code being executed from the FLASH memory. While these operations must be performed in the order shown, other unrelated operations may occur between the steps. Do not exceed t_{PROG} maximum.

1. The time between each FLASH address change, or the time between the last FLASH address programmed to clearing the PGM bit, must not exceed the maximum programming time, t_{PROG} maximum.

3.6.6.1 V_{REFH} and V_{REFL}

Both ac and dc current are drawn through the V_{REFH} and V_{REFL} loop. The AC current is in the form of current spikes required to supply charge to the capacitor array at each successive approximation step. The current flows through the internal resistor string. The best external component to meet both these current demands is a capacitor in the 0.01 µF to 1 µF range with good high frequency characteristics. This capacitor is connected between V_{REFH} and V_{REFL} and must be placed as close as possible to the package pins. Resistance in the path is not recommended because the dc current will cause a voltage drop which could result in conversion errors.

3.6.6.2 ANx

Empirical data shows that capacitors from the analog inputs to V_{REFL} improve ADC performance. 0.01-µF and 0.1-µF capacitors with good high-frequency characteristics are sufficient. These capacitors must be placed as close as possible to the package pins.

3.6.6.3 Grounding

In cases where separate power supplies are used for analog and digital power, the ground connection between these supplies should be at the V_{SSA} pin. This should be the only ground connection between these supplies if possible. The V_{SSA} pin makes a good single point ground location. Connect the V_{REFL} pin to the same potential as V_{SSA} at the single point ground location.

3.7 I/O Registers

These I/O registers control and monitor operation of the ADC:

- ADC status and control register, ADSCR
- ADC data registers, ADRH and ARDL
- ADC clock register, ADCLK

3.7.1 ADC Status and Control Register

This section describes the function of the ADC status and control register (ADSCR). Writing ADSCR aborts the current conversion and initiates a new conversion.

Address: $003C

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	COCO	AIEN	ADCO	ADCH4	ADCH3	ADCH2	ADCH1	ADCH0
Reset:	0	0	0	1	1	1	1	1

Figure 3-4. ADC Status and Control Register (ADSCR)

COCO — Conversions Complete Bit

When AIEN bit is 0, the COCO is a read-only bit which is set each time a conversion is completed except in the continuous conversion mode where it is set after the first conversion. This bit is cleared whenever the ADC status and control register is written or whenever the ADC data register is read.

If AIEN bit is 1, the COCO is a read/write bit. Reset clears this bit.

1 = Conversion completed (AIEN = 0)

0 = Conversion not completed (AIEN = 0)/CPU interrupt (AIEN = 1)

Chapter 5
Configuration Registers (CONFIG1 and CONFIG2)

5.1 Introduction

This section describes the configuration registers, CONFIG1 and CONFIG2. The configuration registers control these options:

- Stop mode recovery time, 32 CGMXCLK cycles or 4096 CGMXCLK cycles
- Computer operating properly (COP) timeout period, 262,128 or 8176 CGMXCLK cycles
- STOP instruction
- Computer operating properly (COP) module
- Low-voltage inhibit (LVI) module control and voltage trip point selection
- Enable/disable the oscillator (OSC) during stop mode
- External clock/crystal source control
- Enhanced SCI clock source selection

5.2 Functional Description

The configuration registers are used in the initialization of various options and can be written once after each reset. All of the configuration register bits are cleared during reset. Since the various options affect the operation of the microcontroller unit (MCU), it is recommended that these registers be written immediately after reset. The configuration registers are located at $001E and $001F. For compatibility, a write to a read-only memory (ROM) version of the MCU at this location will have no effect. The configuration register may be read at anytime.

NOTE

The CONFIG module is known as an MOR (mask option register) on a ROM device. On a ROM device, the options are fixed at the time of device fabrication and are neither writable nor changeable by the user.

On a FLASH device, the CONFIG registers are special registers containing one-time writable latches after each reset. Upon a reset, the CONFIG registers default to predetermined settings as shown in Figure 5-1 and Figure 5-2.

Address: $001E

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	R	ESCIBDSRC	EXTXTALEN	EXTSLOW	EXTCLKEN	TMBCLKSEL	OSCENINSTOP	SSBPUENB
Reset:	0	0	0	0	0	0	0	1

R = Reserved

Figure 5-1. Configuration Register 2 (CONFIG2)

Table 7-1. Instruction Set Summary (Sheet 2 of 6)

Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Cycles
			V	H	I	N	Z	C				
BHS *rel*	Branch if Higher or Same (Same as BCC)	PC ← (PC) + 2 + *rel* ? (C) = 0	–	–	–	–	–	–	REL	24	rr	3
BIH *rel*	Branch if $\overline{\text{IRQ}}$ Pin High	PC ← (PC) + 2 + *rel* ? $\overline{\text{IRQ}}$ = 1	–	–	–	–	–	–	REL	2F	rr	3
BIL *rel*	Branch if $\overline{\text{IRQ}}$ Pin Low	PC ← (PC) + 2 + *rel* ? $\overline{\text{IRQ}}$ = 0	–	–	–	–	–	–	REL	2E	rr	3
BIT #*opr* BIT *opr* BIT *opr* BIT *opr*,X BIT *opr*,X BIT ,X BIT *opr*,SP BIT *opr*,SP	Bit Test	(A) & (M)	0	–	–	↕	↕	–	IMM DIR EXT IX2 IX1 IX SP1 SP2	A5 B5 C5 D5 E5 F5 9EE5 9ED5	ii dd hh ll ee ff ff ff ee ff	2 3 4 4 3 2 4 5
BLE *opr*	Branch if Less Than or Equal To (Signed Operands)	PC ← (PC) + 2 + *rel* ? (Z) ∣ (N ⊕ V) = 1	–	–	–	–	–	–	REL	93	rr	3
BLO *rel*	Branch if Lower (Same as BCS)	PC ← (PC) + 2 + *rel* ? (C) = 1	–	–	–	–	–	–	REL	25	rr	3
BLS *rel*	Branch if Lower or Same	PC ← (PC) + 2 + *rel* ? (C) ∣ (Z) = 1	–	–	–	–	–	–	REL	23	rr	3
BLT *opr*	Branch if Less Than (Signed Operands)	PC ← (PC) + 2 + *rel* ? (N ⊕ V) =1	–	–	–	–	–	–	REL	91	rr	3
BMC *rel*	Branch if Interrupt Mask Clear	PC ← (PC) + 2 + *rel* ? (I) = 0	–	–	–	–	–	–	REL	2C	rr	3
BMI *rel*	Branch if Minus	PC ← (PC) + 2 + *rel* ? (N) = 1	–	–	–	–	–	–	REL	2B	rr	3
BMS *rel*	Branch if Interrupt Mask Set	PC ← (PC) + 2 + *rel* ? (I) = 1	–	–	–	–	–	–	REL	2D	rr	3
BNE *rel*	Branch if Not Equal	PC ← (PC) + 2 + *rel* ? (Z) = 0	–	–	–	–	–	–	REL	26	rr	3
BPL *rel*	Branch if Plus	PC ← (PC) + 2 + *rel* ? (N) = 0	–	–	–	–	–	–	REL	2A	rr	3
BRA *rel*	Branch Always	PC ← (PC) + 2 + *rel*	–	–	–	–	–	–	REL	20	rr	3
BRCLR *n*,*opr*,*rel*	Branch if Bit *n* in M Clear	PC ← (PC) + 3 + *rel* ? (Mn) = 0	–	–	–	–	–	↕	DIR (b0) DIR (b1) DIR (b2) DIR (b3) DIR (b4) DIR (b5) DIR (b6) DIR (b7)	01 03 05 07 09 0B 0D 0F	dd rr dd rr dd rr dd rr dd rr dd rr dd rr dd rr	5 5 5 5 5 5 5 5
BRN *rel*	Branch Never	PC ← (PC) + 2	–	–	–	–	–	–	REL	21	rr	3
BRSET *n*,*opr*,*rel*	Branch if Bit *n* in M Set	PC ← (PC) + 3 + *rel* ? (Mn) = 1	–	–	–	–	–	↕	DIR (b0) DIR (b1) DIR (b2) DIR (b3) DIR (b4) DIR (b5) DIR (b6) DIR (b7)	00 02 04 06 08 0A 0C 0E	dd rr dd rr dd rr dd rr dd rr dd rr dd rr dd rr	5 5 5 5 5 5 5 5
BSET *n*,*opr*	Set Bit *n* in M	Mn ← 1	–	–	–	–	–	–	DIR (b0) DIR (b1) DIR (b2) DIR (b3) DIR (b4) DIR (b5) DIR (b6) DIR (b7)	10 12 14 16 18 1A 1C 1E	dd dd dd dd dd dd dd dd	4 4 4 4 4 4 4 4
BSR *rel*	Branch to Subroutine	PC ← (PC) + 2; push (PCL) SP ← (SP) – 1; push (PCH) SP ← (SP) – 1 PC ← (PC) + *rel*	–	–	–	–	–	–	REL	AD	rr	4
CBEQ *opr*,*rel* CBEQA #*opr*,*rel* CBEQX #*opr*,*rel* CBEQ *opr*,X+,*rel* CBEQ X+,*rel* CBEQ *opr*,SP,*rel*	Compare and Branch if Equal	PC ← (PC) + 3 + rel ? (A) – (M) = $00 PC ← (PC) + 3 + rel ? (A) – (M) = $00 PC ← (PC) + 3 + rel ? (X) – (M) = $00 PC ← (PC) + 3 + rel ? (A) – (M) = $00 PC ← (PC) + 2 + rel ? (A) – (M) = $00 PC ← (PC) + 4 + rel ? (A) – (M) = $00	–	–	–	–	–	–	DIR IMM IMM IX1+ IX+ SP1	31 41 51 61 71 9E61	dd rr ii rr ii rr ff rr rr ff rr	5 4 4 5 4 6
CLC	Clear Carry Bit	C ← 0	–	–	–	–	–	0	INH	98		1
CLI	Clear Interrupt Mask	I ← 0	–	–	0	–	–	–	INH	9A		2

reference with comparators, whose outputs are fed to the digital loop filter. The dependence of these outputs on the capacitor size, current reference, and voltage reference causes up to ±25 percent error in f_{NOM}.

8.3.2.4 Digital Loop Filter

The digital loop filter (DLF) uses the outputs of the frequency comparator to adjust the internal clock (ICLK) clock period. The DLF generates the DCO divider control bits (DDIV[3:0]) and the DCO stage control bits (DSTG[7:0]), which are fed to the DCO. The DLF first concatenates the DDIV and DSTG registers (DDIV[3:0]:DSTG[7:0]) and then adds or subtracts a value dependent on the relative error in the low-frequency base clock's period, as shown in Table 8-1. In some extreme error conditions, such as operating at a V_{DD} level which is out of specification, the DLF may attempt to use a value above the maximum ($9FF) or below the minimum ($000). In both cases, the value for DDIV will be between $A and $F. In this range, the DDIV value will be interpreted the same as $9 (the slowest condition). Recovering from this condition requires subtracting (increasing frequency) in the normal fashion until the value is again below $9FF. (If the desired value is $9xx, the value may settle at $Axx through $Fxx. This is an acceptable operating condition.) If the error is less than ±5 percent, the internal clock generator's filter stable indicator (FICGS) is set, indicating relative frequency accuracy to the clock monitor.

Table 8-1. Correction Sizes from DLF to DCO

Frequency Error of IBASE Compared to f_{NOM}	DDVI[3:0]:DSTG[7:0] Correction	Current to New DDIV[3:0]:DSTG[7:0](1)		Relative Correction in DCO	
IBASE < 0.85 f_{NOM}	–32 (–$020)	Minimum	$xFF to $xDF	–2/31	–6.45%
		Maximum	$x20 to $x00	–2/19	–10.5%
0.85 f_{NOM} < IBASE IBASE < 0.95 f_{NOM}	–8 (–$008)	Minimum	$xFF to $xF7	–0.5/31	–1.61%
		Maximum	$x08 to $x00	–0.5/17.5	–2.86%
0.95 f_{NOM} < IBASE IBASE < f_{NOM}	–1 (–$001)	Minimum	$xFF to $xFE	–0.0625/31	–0.202%
		Maximum	$x01 to $x00	–0.0625/17.0625	–0.366%
f_{NOM} < IBASE IBASE < 1.05 f_{NOM}	+1 (+$001)	Minimum	$xFE to $xFF	+0.0625/30.9375	+0.202%
		Maximum	$x00 to $x01	+0.0625/17	+0.368%
1.05 f_{NOM} < IBASE IBASE < 1.15 f_{NOM}	+8 (+$008)	Minimum	$xF7 to $xFF	+0.5/30.5	+1.64%
		Maximum	$x00 to $x08	+0.5/17	+2.94%
1.15 f_{NOM} < IBASE	+32 (+$020)	Minimum	$xDF to $xFF	+2/29	+6.90%
		Maximum	$x00 to $x20	+2/17	+11.8%

1. x = Maximum error is independent of value in DDIV[3:0]. DDIV increments or decrements when an addition to DSTG[7:0] carries or borrows.

8.3.3 External Clock Generator

The ICG also provides for an external oscillator or external clock source, if desired. The external clock generator, shown in Figure 8-4, contains an external oscillator amplifier and an external clock input path.

8.3.3.2 External Clock Input Path

The external clock input path is the means by which the microcontroller uses an external clock source. The input to the path is the PTC4/OSC1 pin and the output is the external clock (ECLK). The path, which contains input buffering, is enabled when the external clock generator enable signal (ECGEN) is set. When not enabled, the PTC4/OSC1 pin reverts to its port function.

8.3.4 Clock Monitor Circuit

The ICG contains a clock monitor circuit which, when enabled, will continuously monitor both the external clock (ECLK) and the internal clock (ICLK) to determine if either clock source has been corrupted. The clock monitor circuit, shown in Figure 8-5, contains these blocks:

- Clock monitor reference generator
- Internal clock activity detector
- External clock activity detector

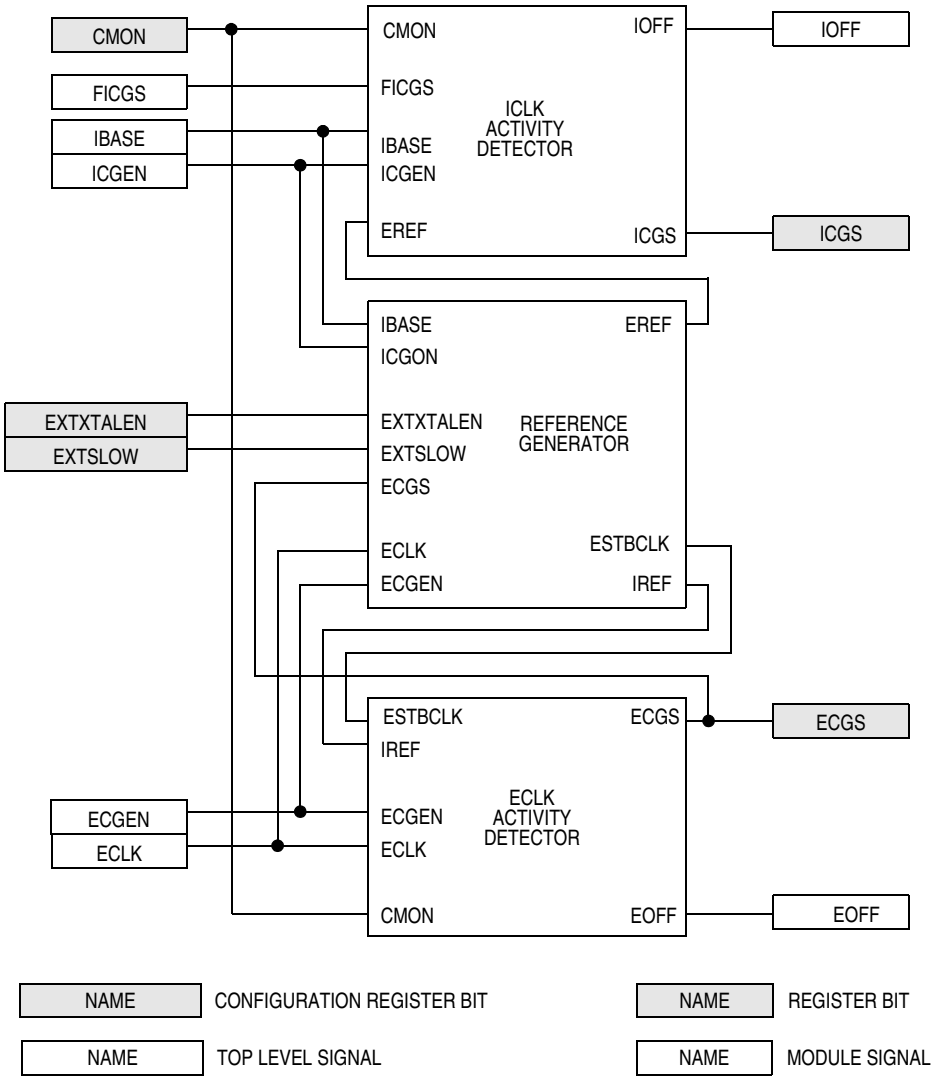


Figure 8-5. Clock Monitor Block Diagram

ICGON — Internal Clock Generator On Bit

This read/write bit enables the internal clock generator. ICGON can be cleared when the CS bit has been set and the CMON bit has been clear for at least one bus cycle. ICGON is forced set when the CMON bit is set, the CS bit is clear, or during reset.

1 = Internal clock generator enabled
0 = Internal clock generator disabled

ICGS — Internal Clock Generator Stable Bit

This read-only bit indicates when the internal clock generator has determined that the internal clock (ICLK) is within about 5 percent of the desired value. This bit is forced clear when the clock monitor determines the ICLK is inactive, when ICGON is clear, when the ICG multiplier register (ICGMR) is written, when the ICG TRIM register (ICGTR) is written, during stop mode with OSCENINSTOP low, or during reset.

1 = Internal clock is within 5 percent of the desired value.
0 = Internal clock may not be within 5 percent of the desired value.

ECGON — External Clock Generator On Bit

This read/write bit enables the external clock generator. ECGON can be cleared when the CS and CMON bits have been clear for at least one bus cycle. ECGON is forced set when the CMON bit or the CS bit is set. ECGON is forced clear during reset.

1 = External clock generator enabled
0 = External clock generator disabled

ECGS — External Clock Generator Stable Bit

This read-only bit indicates when at least 4096 external clock (ECLK) cycles have elapsed since the external clock generator was enabled. This is not an assurance of the stability of ECLK but is meant to provide a startup delay. This bit is forced clear when the clock monitor determines ECLK is inactive, when ECGON is clear, during stop mode with OSCENINSTOP low, or during reset.

1 = 4096 ECLK cycles have elapsed since ECGON was set.
0 = External clock is unstable, inactive, or disabled.

8.7.2 ICG Multiplier Register

Address: $0037

	Bit 7	6	5	4	3	2	1	Bit 0
Read:		N6	N5	N4	N3	N2	N1	N0
Write:								
Reset:	0	0	0	1	0	1	0	1

= Unimplemented

Figure 8-12. ICG Multiplier Register (ICGMR)

N6:N0 — ICG Multiplier Factor Bits

These read/write bits change the multiplier used by the internal clock generator. The internal clock (ICLK) will be:

(307.2 kHz ± 25 percent) * N

A value of $00 in this register is interpreted the same as a value of $01. This register cannot be written when the CMON bit is set. Reset sets this factor to $15 (decimal 21) for default frequency of 6.45 MHz ± 25 percent (1.613 MHz ± 25 percent bus).

10.7.2 Keyboard Interrupt Enable Register

The keyboard interrupt enable register enables or disables each port A pin to operate as a keyboard interrupt pin.

Address: $001B

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	KBIE4	KBIE3	KBIE2	KBIE1	KBIE0
Write:								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 10-4. Keyboard Interrupt Enable Register (KBIER)

KBIE4–KBIE0 — Keyboard Interrupt Enable Bits

Each of these read/write bits enables the corresponding keyboard interrupt pin to latch interrupt requests. Reset clears the keyboard interrupt enable register.

1 = KBDx pin enabled as keyboard interrupt pin
0 = KBDx pin not enabled as keyboard interrupt pin

Chapter 12
Input/Output (I/O) Ports (PORTS)

12.1 Introduction

Twenty-four bidirectional input/output (I/O) pins form five parallel ports. All I/O pins are programmable as inputs or outputs.

NOTE

Connect any unused I/O pins to an appropriate logic level, either V_{DD} or V_{SS}. Although the I/O ports do not require termination for proper operation, termination reduces excess current consumption and the possibility of electrostatic damage.

12.2 Port A

Port A is a 7-bit general-purpose bidirectional I/O port that shares pin functions with the serial peripheral interface (SPI) and keyboard (KBD) modules.

12.2.1 Port A Data Register

The port A data register contains a data latch for each of the seven port A pins.

Address:	$0000							
	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	PTA6	PTA5	PTA4	PTA3	PTA2	PTA1	PTA0
Write:								
Reset:	Unaffected by reset							
Alternative Function:		$\overline{SS}$	SPSCK	$\overline{KBD4}$	$\overline{KBD3}$	$\overline{KBD2}$	$\overline{KBD1}$	$\overline{KBD0}$

= Unimplemented

Figure 12-1. Port A Data Register (PTA)

PTA[6:0] — Port A Data Bits

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

12.2.2 Data Direction Register A

Data direction register A 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.

14.5.3 Break Interrupts

The break module can stop normal program flow at a software-programmable break point by asserting its break interrupt output. See 19.2 Break Module (BRK). The SIM puts the CPU into the break state by forcing it to the SWI vector location. Refer to the break interrupt subsection of each module to see how each module is affected by the break state.

14.5.4 Status Flag Protection in Break Mode

The SIM controls whether status flags contained in other modules can be cleared during break mode. The user can select whether flags are protected from being cleared by properly initializing the break clear flag enable bit (BCFE) in the SIM break flag control register (SBFCR).

Protecting flags in break mode ensures that set flags will not be cleared while in break mode. This protection allows registers to be freely read and written during break mode without losing status flag information.

Setting the BCFE bit enables the clearing mechanisms. Once cleared in break mode, a flag remains cleared even when break mode is exited. Status flags with a two-step clearing mechanism — for example, a read of one register followed by the read or write of another — are protected, even when the first step is accomplished prior to entering break mode. Upon leaving break mode, execution of the second step will clear the flag as normal.

14.6 Low-Power Modes

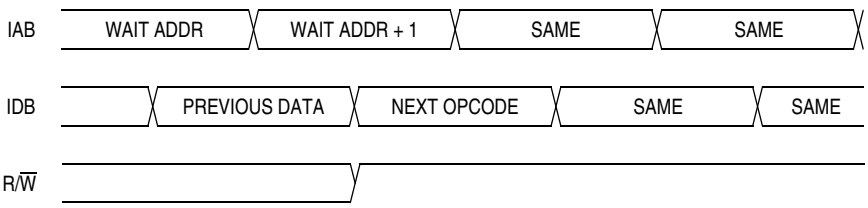
Executing the WAIT or STOP instruction puts the MCU in a low power- consumption mode for standby situations. The SIM holds the CPU in a non-clocked state. Both STOP and WAIT clear the interrupt mask (I) in the condition code register, allowing interrupts to occur. Low-power modes are exited via an interrupt or reset.

14.6.1 Wait Mode

In wait mode, the CPU clocks are inactive while one set of peripheral clocks continues to run. Figure 14-11 shows the timing for wait mode entry.

A module that is active during wait mode can wake up the CPU with an interrupt if the interrupt is enabled. Stacking for the interrupt begins one cycle after the WAIT instruction during which the interrupt occurred. Refer to the wait mode subsection of each module to see if the module is active or inactive in wait mode. Some modules can be programmed to be active in wait mode.

Wait mode can also be exited by a reset. If the COP disable bit, COPD, in the configuration register is 0, then the computer operating properly module (COP) is enabled and remains active in wait mode.



Note: Previous data can be operand data or the WAIT opcode, depending on the last instruction.

Figure 14-11. Wait Mode Entry Timing

Figure 14-12 and Figure 14-13 show the timing for WAIT recovery.

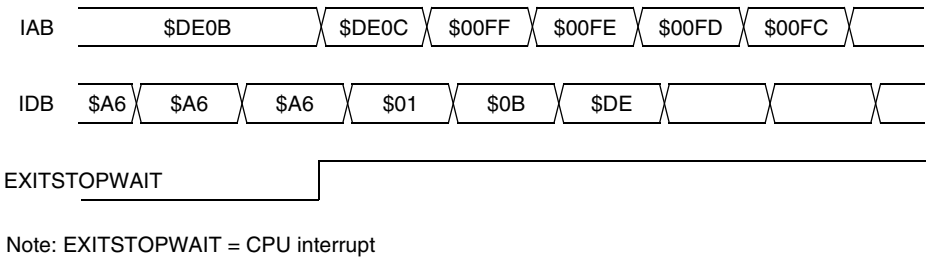


Figure 14-12. Wait Recovery from Interrupt

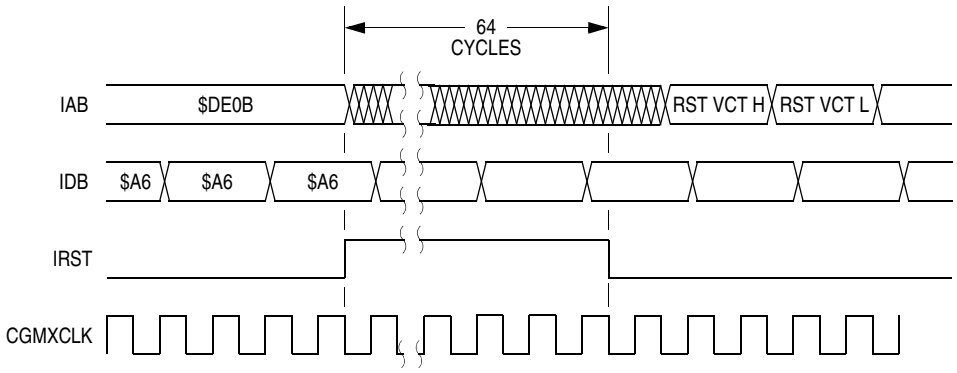


Figure 14-13. Wait Recovery from Internal Reset

14.6.2 Stop Mode

In stop mode, the SIM counter is held in reset and the CPU and peripheral clocks are held inactive. If the STOPOSCEN bit in the configuration register is not enabled, the SIM also disables the internal clock generator module outputs (CGMOUT and CGMXCLK).

The CPU and peripheral clocks do not become active until after the stop delay timeout. Stop mode is exited via an interrupt request from a module that is still active in stop mode or from a system reset.

An interrupt request from a module that is still active in stop mode can cause an exit from stop mode. Stop recovery time is selectable using the SSREC bit in the configuration register. If SSREC is set, stop recovery is reduced from the normal delay of 4096 CGMXCLK cycles down to 32. Stacking for interrupts begins after the selected stop recovery time has elapsed.

When stop mode is exited due to a reset condition, the SIM forces a long stop recovery time of 4096 CGMXCLK cycles.

NOTE

Short stop recovery is ideal for applications using canned oscillators that do not require long startup times for stop mode. External crystal applications should use the full stop recovery time by clearing the SSREC bit.

The SIM counter is held in reset from the execution of the STOP instruction until the beginning of stop recovery. It is then used to time the recovery period. Figure 14-14 shows stop mode entry timing.

SPR1 and SPR0 — SPI Baud Rate Select Bits

In master mode, these read/write bits select one of four baud rates as shown in Table 15-5. SPR1 and SPR0 have no effect in slave mode. Reset clears SPR1 and SPR0.

Table 15-5. SPI Master Baud Rate Selection

SPR1:SPR0	Baud Rate Divisor (BD)
00	2
01	8
10	32
11	128

Use this formula to calculate the SPI baud rate:

$$\text{Baud rate} = \frac{\text{CGMOUT}}{2 \times \text{BD}}$$

where:

CGMOUT = base clock output of the internal clock generator module (ICG), see Chapter 8 Internal Clock Generator (ICG) Module.

BD = baud rate divisor

15.13.3 SPI Data Register

The SPI data register is the read/write buffer for the receive data register and the transmit data register. Writing to the SPI data register writes data into the transmit data register. Reading the SPI data register reads data from the receive data register. The transmit data and receive data registers are separate buffers that can contain different values. See Figure 15-2

Address: $000F

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	R7	R6	R5	R4	R3	R2	R1	R0
Write:	T7	T6	T5	T4	T3	T2	T1	T0
Reset:	Indeterminate after Reset							

Figure 15-14. SPI Data Register (SPDR)

R7–R0/T7–T0 — Receive/Transmit Data Bits

NOTE

Do not use read-modify-write instructions on the SPI data register since the buffer read is not the same as the buffer written.

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 17.8.5 TIMA 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 register (TACHxH–TACHxL).

17.3.3 Output Compare

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

17.3.3.1 Unbuffered Output Compare

Any output compare channel can generate unbuffered output compare pulses as described in 17.3.3 Output Compare. The pulses are unbuffered because changing the output compare value requires writing the new value over the old value currently in the TIMA channel registers.

An unsynchronized write to the TIMA channel registers to change an output compare value could cause incorrect operation for up to two counter overflow periods. For example, writing a new value before the counter reaches the old value but after the counter reaches the new value prevents any compare during that counter overflow period. Also, using a TIMA overflow interrupt routine to write a new, smaller output compare value may cause the compare to be missed. The TIMA may pass the new value before it is written.

Use these methods to synchronize unbuffered changes in the output compare value on channel x:

- When changing to a smaller value, enable channel x output compare interrupts and write the new value in the output compare interrupt routine. The output compare interrupt occurs at the end of the current output compare pulse. The interrupt routine has until the end of the counter overflow period to write the new value.
- When changing to a larger output compare value, enable TIMA overflow interrupts and write the new value in the TIMA overflow interrupt routine. The TIMA overflow interrupt occurs at the end of the current counter overflow period. Writing a larger value in an output compare interrupt routine (at the end of the current pulse) could cause two output compares to occur in the same counter overflow period.

TOIE — TIMA Overflow Interrupt Enable Bit

This read/write bit enables TIMA overflow interrupts when the TOF bit becomes set. Reset clears the TOIE bit.

1 = TIMA overflow interrupts enabled
0 = TIMA overflow interrupts disabled

TSTOP — TIMA Stop Bit

This read/write bit stops the TIMA counter. Counting resumes when TSTOP is cleared. Reset sets the TSTOP bit, stopping the TIMA counter until software clears the TSTOP bit.

1 = TIMA counter stopped
0 = TIMA counter active

NOTE

Do not set the TSTOP bit before entering wait mode if the TIMA is required to exit wait mode. Also, when the TSTOP bit is set and the timer is configured for input capture operation, input captures are inhibited until TSTOP is cleared.

When using TSTOP to stop the timer counter, check for any timer flags being set. If a timer flag is set, it must be cleared by clearing TSTOP, then clearing the flag, then setting TSTOP again.

TRST — TIMA Reset Bit

Setting this write-only bit resets the TIMA counter and the TIMA prescaler. Setting TRST has no effect on any other registers. Counting resumes from $0000. TRST is cleared automatically after the TIMA counter is reset and always reads as 0. Reset clears the TRST bit.

1 = Prescaler and TIMA counter cleared
0 = No effect

NOTE

Setting the TSTOP and TRST bits simultaneously stops the TIMA counter at a value of $0000.

PS[2:0] — Prescaler Select Bits

These read/write bits select one of the seven prescaler outputs as the input to the TIMA counter as Table 17-1 shows. Reset clears the PS[2:0] bits.

Table 17-1. Prescaler Selection

PS[2:0]	TIMA 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	Unused

17.8.2 TIMA Counter Registers

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

NOTE

If TACNTH is read during a break interrupt, be sure to unlatch TACNTL by reading TACNTL before exiting the break interrupt. Otherwise, TACNTL retains the value latched during the break.

Register Name and Address: TACNTH — $0021

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

Register Name and Address: TACNTL — $0022

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

R = Reserved

Figure 17-5. TIMA Counter Registers (TACNTH and TACNTL)

17.8.3 TIMA Counter Modulo Registers

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

Register Name and Address: TAMODH — $0023

	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:	1	1	1	1	1	1	1	1

Register Name and Address: TAMODL — $0024

	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:	1	1	1	1	1	1	1	1

Figure 17-6. TIMA Counter Modulo Registers (TMODH and TMODL)

NOTE

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

19.3 Monitor Module (MON)

The monitor module allows debugging and programming of the microcontroller unit (MCU) through a single-wire interface with a host computer. Monitor mode entry can be achieved without use of the higher test voltage, V_{TST}, as long as vector addresses $FFFE and $FFFF are blank, thus reducing the hardware requirements for in-circuit programming.

Features of the monitor module include:

- Normal user-mode pin functionality
- One pin dedicated to serial communication between MCU and host computer
- Standard non-return-to-zero (NRZ) communication with host computer
- Standard communication baud rate (9600 @ 2.4576-MHz bus frequency)
- Execution of code in random-access memory (RAM) or FLASH
- FLASH memory security feature(1)
- FLASH memory programming interface
- Monitor mode entry without high voltage, V_{TST}, if reset vector is blank ($FFFE and $FFFF contain $FF)
- Normal monitor mode entry if V_{TST} is applied to $\overline{IRQ}$

19.3.1 Functional Description

Figure 19-8 shows a simplified diagram of the monitor mode.

The monitor module receives and executes commands from a host computer.
Figure 19-9 and Figure 19-11 show example circuits used to enter monitor mode and communicate with a host computer via a standard RS-232 interface.

Simple monitor commands can access any memory address. In monitor mode, the MCU can execute code downloaded into RAM by a host computer while most MCU pins retain normal operating mode functions. All communication between the host computer and the MCU is through the PTA0 pin. A level-shifting and multiplexing interface is required between PTA0 and the host computer. PTA0 is used in a wired-OR configuration and requires a pullup resistor.

Table 19-1 shows the pin conditions for entering monitor mode. As specified in the table, monitor mode may be entered after a power-on reset (POR) and will allow communication at 9600 baud provided one of the following sets of conditions is met:

- If $FFFE and $FFFF do not contain $FF (programmed state):
  - The external clock is 9.8304 MHz (9600 baud)
  - IRQ = V_{TST}
- If $FFFE and $FFFF contain $FF (erased state):
  - The external clock is 9.8304 MHz (9600 baud)
  - $\overline{IRQ}$ = V_{DD} (this can be implemented through the internal $\overline{IRQ}$ pullup)
- If $FFFE and $FFFF contain $FF (erased state):
  - The ICG clock is nominal 1.6 MHz (nominal 6300 baud)
  - $\overline{IRQ}$ = V_{SS}

1. No security feature is absolutely secure. However, Freescale's strategy is to make reading or copying the FLASH difficult for unauthorized users.

break interrupt — A software interrupt caused by the appearance on the internal address bus of the same value that is written in the break address registers.

bus — A set of wires that transfers logic signals.

bus clock — The bus clock is derived from the CGMOUT output from the CGM. The bus clock frequency, f_{op}, is equal to the frequency of the oscillator output, CGMXCLK, divided by four.

byte — A set of eight bits.

C — The carry/borrow bit in the condition code register. The CPU08 sets the carry/borrow bit when an addition operation produces a carry out of bit 7 of the accumulator or when a subtraction operation requires a borrow. Some logical operations and data manipulation instructions also clear or set the carry/borrow bit (as in bit test and branch instructions and shifts and rotates).

CCR — See “condition code register.”

central processor unit (CPU) — The primary functioning unit of any computer system. The CPU controls the execution of instructions.

CGM — See “clock generator module (CGM).”

clear — To change a bit from 1 to 0; the opposite of set.

clock — A square wave signal used to synchronize events in a computer.

clock generator module (CGM) — A module in the M68HC08 Family. The CGM generates a base clock signal from which the system clocks are derived. The CGM may include a crystal oscillator circuit and or phase-locked loop (PLL) circuit.

comparator — A device that compares the magnitude of two inputs. A digital comparator defines the equality or relative differences between two binary numbers.

computer operating properly module (COP) — A counter module in the M68HC08 Family that resets the MCU if allowed to overflow.

condition code register (CCR) — An 8-bit register in the CPU08 that contains the interrupt mask bit and five bits that indicate the results of the instruction just executed.

control bit — One bit of a register manipulated by software to control the operation of the module.

control unit — One of two major units of the CPU. The control unit contains logic functions that synchronize the machine and direct various operations. The control unit decodes instructions and generates the internal control signals that perform the requested operations. The outputs of the control unit drive the execution unit, which contains the arithmetic logic unit (ALU), CPU registers, and bus interface.

COP — See "computer operating properly module (COP)."

counter clock — The input clock to the TIM counter. This clock is the output of the TIM prescaler.

CPU — See “central processor unit (CPU).”

CPU08 — The central processor unit of the M68HC08 Family.