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	I²C, IRSCI, SCI, SPI
Peripherals	LED, LVD, POR, PWM
Number of I/O	32
Program Memory Size	8KB (8K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	1K x 8
Voltage - Supply (Vcc/Vdd)	4.5V ~ 5.5V
Data Converters	A/D 8x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	44-QFP
Supplier Device Package	44-QFP (10x10)
Purchase URL	https://www.e-xfl.com/pro/item?MUrl=&PartUrl=mc908ap8acfbe

Chapter 5
Oscillator (OSC)

Chapter 6
Clock Generator Module (CGM)

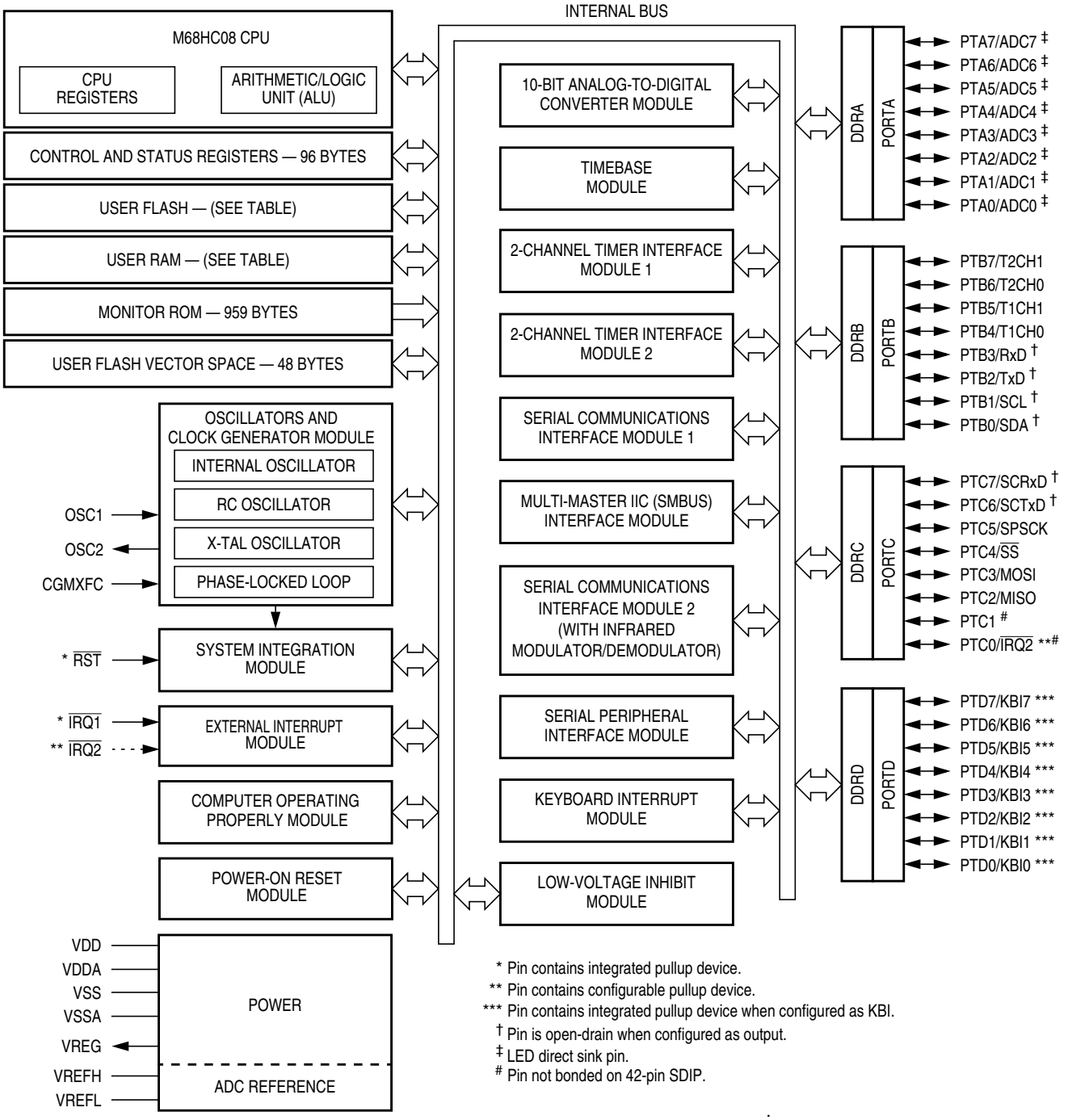
- Timebase module
- Serial communications interface module 1 (SCI)
- Serial communications interface module 2 (SCI) with infrared (IR) encoder/decoder
- Serial peripheral interface module (SPI)
- System management bus (SMBus), version 1.0/1.1 (multi-master IIC bus)
- 8-channel, 10-bit analog-to-digital converter (ADC)
- $\overline{\text{IRQ1}}$ external interrupt pin with integrated pullup
- $\overline{\text{IRQ2}}$ external interrupt pin with programmable pullup
- 8-bit keyboard wakeup port with integrated pullup
- 32 general-purpose input/output (I/O) pins:
  - 31 shared-function I/O pins
  - 8 LED drivers (sink)
  - 6×25mA open-drain I/O with pullup
- Low-power design (fully static with stop and wait modes)
- Master reset pin (with integrated pullup) and power-on reset
- System protection features
  - Optional computer operating properly (COP) reset, driven by internal RC oscillator
  - Low-voltage detection with optional reset or interrupt
  - Illegal opcode detection with reset
  - Illegal address detection with reset
- 48-pin low quad flat pack (LQFP), 44-pin quad flat pack (QFP), and 42-pin shrink dual-in-line package (SDIP)
- Specific features of the MC68HC908AP64A in 42-pin SDIP are:
  - 30 general-purpose I/Os only
  - External interrupt on $\overline{\text{IRQ1}}$ 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 8×8 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 MC68HC908AP64A.



DEVICE	USER RAM (bytes)	USER FLASH (bytes)
MC68HC908AP64A	2,048	62,368
MC68HC908AP32A	2,048	32,768
MC68HC908AP16A	1,024	16,384
MC68HC908AP8A	1,024	8,192

Figure 1-1. MC68HC908AP64A Block Diagram

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$000D	Unimplemented	Read: Write:								
		Reset:								
$000E	Unimplemented	Read: Write:								
		Reset:								
$000F	Unimplemented	Read: Write:								
		Reset:								
$0010	SPI Control Register (SPCR)	Read: Write:	SPRIE	R	SPMSTR	CPOL	CPHA	SPWOM	SPE	SPTIE
		Reset:	0	0	1	0	1	0	0	0
$0011	SPI Status and Control Register (SPSCR)	Read:	SPRF	ERRIE	OVRF	MODF	SPTE	MODFEN	SPR1	SPR0
		Write:								
		Reset:	0	0	0	0	1	0	0	0
$0012	SPI Data Register (SPDR)	Read:	R7	R6	R5	R4	R3	R2	R1	R0
		Write:	T7	T6	T5	T4	T3	T2	T1	T0
		Reset:	Unaffected by reset							
$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:	R8	T8	DMARE	DMATE	ORIE	NEIE	FEIE	PEIE
		Write:								
		Reset:	U	U	0	0	0	0	0	0
$0016	SCI Status Register 1 (SCS1)	Read:	SCTE	TC	SCRF	IDLE	OR	NF	FE	PE
		Write:								
		Reset:	1	1	0	0	0	0	0	0
$0017	SCI Status Register 2 (SCS2)	Read:	0	0	0	0	0	0	BKF	RPF
		Write:								
		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:	0	0	SCP1	SCP0	R	SCR2	SCR1	SCR0
		Write:								
		Reset:	0	0	0	0	0	0	0	0

U = Unaffected X = Indeterminate = Unimplemented R = Reserved

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

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0026	Timer 1 Channel 0 Register High (T1CH0H)	Read: Write:	Bit 15	14	13	12	11	10	9	Bit 8
		Reset:	Indeterminate after reset							
$0027	Timer 1 Channel 0 Register Low (T1CH0L)	Read: Write:	Bit 7	6	5	4	3	2	1	Bit 0
		Reset:	Indeterminate after reset							
$0028	Timer 1 Channel 1 Status and Control Register (T1SC1)	Read: Write:	CH1F 0	CH1IE	0	MS1A	ELS1B	ELS1A	TOV1	CH1MAX
		Reset:	0	0	0	0	0	0	0	0
$0029	Timer 1 Channel 1 Register High (T1CH1H)	Read: Write:	Bit 15	14	13	12	11	10	9	Bit 8
		Reset:	Indeterminate after reset							
$002A	Timer 1 Channel 1 Register Low (T1CH1L)	Read: Write:	Bit 7	6	5	4	3	2	1	Bit 0
		Reset:	Indeterminate after reset							
$002B	Timer 2 Status and Control Register (T2SC)	Read: Write:	TOF 0	TOIE	TSTOP	0 TRST	0	PS2	PS1	PS0
		Reset:	0	0	1	0	0	0	0	0
$002C	Timer 2 Counter Register High (T2CNTH)	Read: Write:	Bit 15	14	13	12	11	10	9	Bit 8
		Reset:	0	0	0	0	0	0	0	0
$002D	Timer 2 Counter Register Low (T2CNTL)	Read: Write:	Bit 7	6	5	4	3	2	1	Bit 0
		Reset:	0	0	0	0	0	0	0	0
$002E	Timer 2 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
$002F	Timer 2 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
$0030	Timer 2 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
$0031	Timer 2 Channel 0 Register High (T2CH0H)	Read: Write:	Bit 15	14	13	12	11	10	9	Bit 8
		Reset:	Indeterminate after reset							
$0032	Timer 2 Channel 0 Register Low (T2CH0L)	Read: Write:	Bit 7	6	5	4	3	2	1	Bit 0
		Reset:	Indeterminate after reset							

U = Unaffected X = Indeterminate = Unimplemented R = Reserved

Figure 2-2. Control, Status, and Data Registers (Sheet 4 of 9)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$004D	MMIIC Data Receive Register (MMDRR)	Read:	MMRD7	MMRD6	MMRD5	MMRD4	MMRD3	MMRD2	MMRD1	MMRD0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$004E	MMIIC CRC Data Register (MMCRDR)	Read:	MMCRCD7	MMCRCD6	MMCRCD5	MMCRCD4	MMCRCD3	MMCRCD2	MMCRCD1	MMCRCD0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$004F	MMIIC Frequency Divider Register (MMFDR)	Read:	0	0	0	0	0	MMBR2	MMBR1	MMBR0
		Write:								
		Reset:	0	0	0	0	0	1	0	0
$0050	Reserved	Read:	R	R	R	R	R	R	R	R
		Write:								
		Reset:								
$0051	Timebase Control Register (TBCR)	Read:	TBIF	TBR2	TBR1	TBR0	0	TBIE	TBON	R
		Write:					TACK			
		Reset:	0	0	0	0	0	0	0	0
$0052	Unimplemented	Read:								
		Write:								
		Reset:								
$0053	Unimplemented	Read:								
		Write:								
		Reset:								
$0054	Unimplemented	Read:								
		Write:								
		Reset:								
$0055	Unimplemented	Read:								
		Write:								
		Reset:								
$0056	Unimplemented	Read:								
		Write:								
		Reset:								
$0057	ADC Status and Control Register (ADSCR)	Read:	COCO	AIEN	ADCO	ADCH4	ADCH3	ADCH2	ADCH1	ADCH0
		Write:								
		Reset:	0	0	0	1	1	1	1	1
$0058	ADC Clock Control Register (ADICLK)	Read:	ADIV2	ADIV1	ADIV0	ADICLK	MODE1	MODE0	0	0
		Write:								R
		Reset:	0	0	0	0	0	0	0	0
$0059	ADC Data Register High 0 (ADRH0)	Read:	ADx	ADx	ADx	ADx	ADx	ADx	ADx	ADx
		Write:	R	R	R	R	R	R	R	R
		Reset:	0	0	0	0	0	0	0	0

U = Unaffected X = Indeterminate = Unimplemented R = Reserved

Figure 2-2. Control, Status, and Data Registers (Sheet 7 of 9)

4.6 CPU During Break Interrupts

If a break module is present on the MCU, the CPU starts a break interrupt by:

- Loading the instruction register with the SWI instruction
- Loading the program counter with $FFFC:$FFFD or with $FEFC:$FEFD in monitor mode

The break interrupt begins after completion of the CPU instruction in progress. If the break address register match occurs on the last cycle of a CPU instruction, the break interrupt begins immediately.

A return-from-interrupt instruction (RTI) in the break routine ends the break interrupt and returns the MCU to normal operation if the break interrupt has been deasserted.

4.7 Instruction Set Summary

Table 4-1 provides a summary of the M68HC08 instruction set.

4.8 Opcode Map

The opcode map is provided in Table 4-2.

Table 4-1. Instruction Set Summary

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

Table 4-1. Instruction Set Summary

Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Cycles
			V	H	I	N	Z	C				
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
CLR *opr* CLRA CLRX CLRH CLR *opr*,X CLR ,X CLR *opr*,SP	Clear	M ← $00 A ← $00 X ← $00 H ← $00 M ← $00 M ← $00 M ← $00	0	–	–	0	1	–	DIR INH INH INH IX1 IX SP1	3F 4F 5F 8C 6F 7F 9E6F	dd ff ff	3 1 1 1 3 2 4
CMP #*opr* CMP *opr* CMP *opr* CMP *opr*,X CMP *opr*,X CMP ,X CMP *opr*,SP CMP *opr*,SP	Compare A with M	(A) – (M)	↕	–	–	↕	↕	↕	IMM DIR EXT IX2 IX1 IX SP1 SP2	A1 B1 C1 D1 E1 F1 9EE1 9ED1	ii dd hh ll ee ff ff ff ee ff	2 3 4 4 3 2 4 5
COM *opr* COMA COMX COM *opr*,X COM ,X COM *opr*,SP	Complement (One's Complement)	M ← ($\overline{M}$) = $FF – (M) A ← ($\overline{A}$) = $FF – (M) X ← ($\overline{X}$) = $FF – (M) M ← ($\overline{M}$) = $FF – (M) M ← ($\overline{M}$) = $FF – (M) M ← ($\overline{M}$) = $FF – (M)	0	–	–	↕	↕	1	DIR INH INH IX1 IX SP1	33 43 53 63 73 9E63	dd ff ff	4 1 1 4 3 5
CPHX #*opr* CPHX *opr*	Compare H:X with M	(H:X) – (M:M + 1)	↕	–	–	↕	↕	↕	IMM DIR	65 75	ii ii+1 dd	3 4
CPX #*opr* CPX *opr* CPX *opr* CPX ,X CPX *opr*,X CPX *opr*,X CPX *opr*,SP CPX *opr*,SP	Compare X with M	(X) – (M)	↕	–	–	↕	↕	↕	IMM DIR EXT IX2 IX1 IX SP1 SP2	A3 B3 C3 D3 E3 F3 9EE3 9ED3	ii dd hh ll ee ff ff ff ee ff	2 3 4 4 3 2 4 5
DAA	Decimal Adjust A	$(A)_{10}$	U	–	–	↕	↕	↕	INH	72		2

register disables the PLL and clears the BCS bit in the PLL control register (PCTL). (See 6.3.8 Base Clock Selector Circuit and 6.3.7 Special Programming Exceptions.). Reset initializes the register to $40 for a default range multiply value of 64.

NOTE

The VCO range select bits have built-in protection such that they cannot be written when the PLL is on (PLLON = 1) and such that the VCO clock cannot be selected as the source of the base clock (BCS = 1) if the VCO range select bits are all clear.

The PLL VCO range select register must be programmed correctly. Incorrect programming can result in failure of the PLL to achieve lock.

6.5.5 PLL Reference Divider Select Register

The PLL reference divider select register (PMDS) contains the programming information for the modulo reference divider.

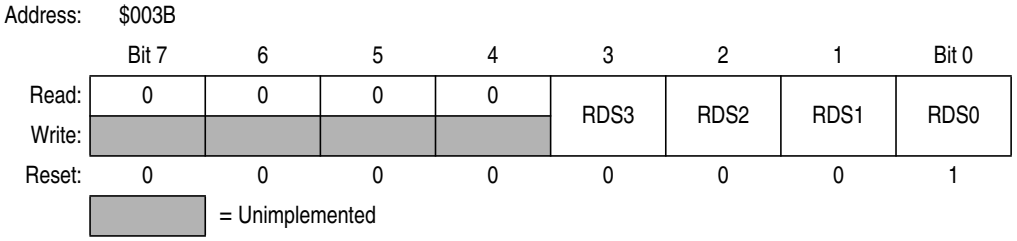


Figure 6-9. PLL Reference Divider Select Register (PMDS)

RDS[3:0] — Reference Divider Select Bits

These read/write bits control the modulo reference divider that selects the reference division factor, R. (See 6.3.3 PLL Circuits and 6.3.6 Programming the PLL.) RDS[3:0] cannot be written when the PLLON bit in the PCTL is set. A value of $00 in the reference divider select register configures the reference divider the same as a value of $01. (See 6.3.7 Special Programming Exceptions.) Reset initializes the register to $01 for a default divide value of 1.

NOTE

The reference divider select bits have built-in protection such that they cannot be written when the PLL is on (PLLON = 1).

NOTE

The default divide value of 1 is recommended for all applications.

6.6 Interrupts

When the AUTO bit is set in the PLL bandwidth control register (PBWC), the PLL can generate a CPU interrupt request every time the LOCK bit changes state. The PLLIE bit in the PLL control register (PCTL) enables CPU interrupts from the PLL. PLLF, the interrupt flag in the PCTL, becomes set whether interrupts are enabled or not. When the AUTO bit is clear, CPU interrupts from the PLL are disabled and PLLF reads as logic 0.

Software should read the LOCK bit after a PLL interrupt request to see if the request was due to an entry into lock or an exit from lock. When the PLL enters lock, the divided VCO clock, CGMPCLK, divided by two can be selected as the CGMOUT source by setting BCS in the PCTL. When the PLL exits lock, the

7.4.3 SIM Counter and Reset States

External reset has no effect on the SIM counter. (See 7.6.2 Stop Mode for details.) The SIM counter is free-running after all reset states. (See 7.3.2 Active Resets from Internal Sources for counter control and internal reset recovery sequences.)

7.5 Exception Control

Normal, sequential program execution can be changed in three different ways:

- Interrupts:
  - Maskable hardware CPU interrupts
  - Non-maskable software interrupt instruction (SWI)
- Reset
- Break interrupts

7.5.1 Interrupts

At the beginning of an interrupt, the CPU saves the CPU register contents on the stack and sets the interrupt mask (I bit) to prevent additional interrupts. At the end of an interrupt, the RTI instruction recovers the CPU register contents from the stack so that normal processing can resume. Figure 7-8 shows interrupt entry timing, and Figure 7-9 shows interrupt recovery timing.

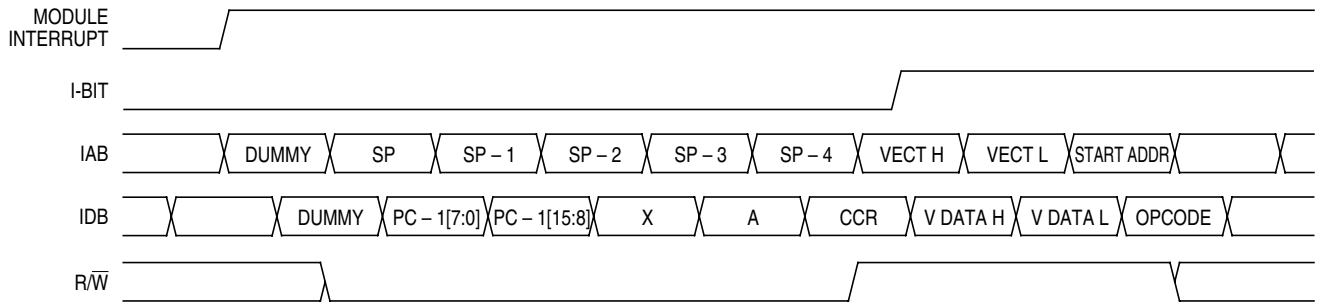


Figure 7-8. Interrupt Entry Timing

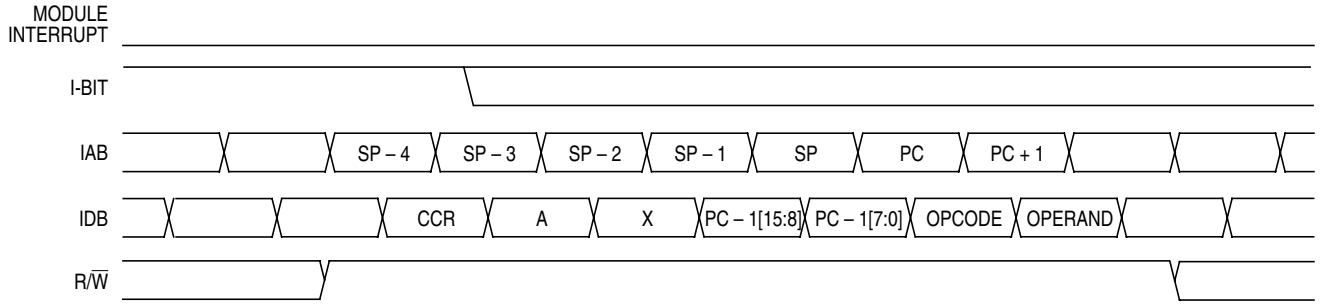


Figure 7-9. Interrupt Recovery Timing

Interrupts are latched, and arbitration is performed in the SIM at the start of interrupt processing. The arbitration result is a constant that the CPU uses to determine which vector to fetch. Once an interrupt is latched by the SIM, no other interrupt can take precedence, regardless of priority, until the latched interrupt is serviced (or the I bit is cleared).
(See Figure 7-10.)

Chapter 8
Monitor Mode (MON)

8.1 Introduction

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.

8.2 Features

Features of the monitor ROM include:

- Normal user-mode pin functionality
- One pin dedicated to serial communication between monitor ROM and host computer
- Standard mark/space non-return-to-zero (NRZ) communication with host computer
- Execution of code in RAM or FLASH
- FLASH memory security feature[1]
- 959 bytes monitor ROM code size ($FC00–$FDFF and $FE10–$FFCE)
- Monitor mode entry without high voltage, V_{TST}, if reset vector is blank ($FFFE and $FFFF contain $FF)
- Standard monitor mode entry if high voltage, V_{TST}, is applied to $\overline{IRQ1}$
- Resident routines for in-circuit programming

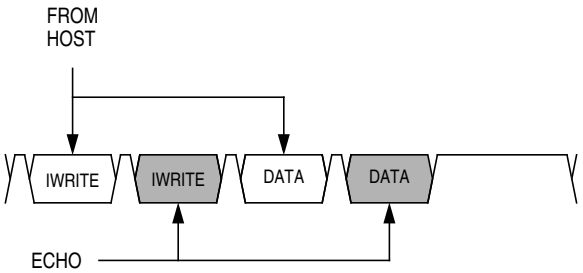
8.3 Functional Description

The monitor module receives and executes commands from a host computer. Figure 8-1 shows an example circuit 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.

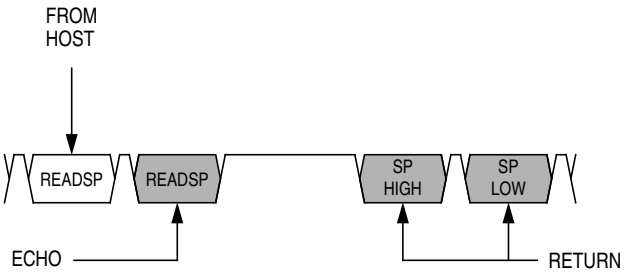
1. No security feature is absolutely secure. However, Freescale's strategy is to make reading or copying the FLASH difficult for unauthorized users.

Table 8-7. IWRITE (Indexed Write) Command

Description	Write to last address accessed + 1
Operand	Single data byte
Data Returned	None
Opcode	$19
Command Sequence	

A sequence of IREAD or IWRITE commands can access a block of memory sequentially over the full 64-Kbyte memory map.

Table 8-8. READSP (Read Stack Pointer) Command

Description	Reads stack pointer
Operand	None
Data Returned	Returns incremented stack pointer value (SP + 1) in high-byte:low-byte order
Opcode	$0C
Command Sequence	

TOF — TIM Overflow Flag Bit

This read/write flag is set when the TIM counter reaches the modulo value programmed in the TIM counter modulo registers. Clear TOF by reading the TIM status and control register when TOF is set and then writing a logic 0 to TOF. If another TIM overflow occurs before the clearing sequence is complete, then writing logic 0 to TOF has no effect. Therefore, a TOF interrupt request cannot be lost due to inadvertent clearing of TOF. Reset clears the TOF bit. Writing a logic 1 to TOF has no effect.

1 = TIM counter has reached modulo value
0 = TIM counter has not reached modulo value

TOIE — TIM Overflow Interrupt Enable Bit

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

1 = TIM overflow interrupts enabled
0 = TIM overflow interrupts disabled

TSTOP — TIM Stop Bit

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

1 = TIM counter stopped
0 = TIM counter active

NOTE

Do not set the TSTOP bit before entering wait mode if the TIM 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 the TSTOP bit is cleared.

TRST — TIM Reset Bit

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

1 = Prescaler and TIM counter cleared
0 = No effect

NOTE

Setting the TSTOP and TRST bits simultaneously stops the TIM 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 TIM counter as Table 9-2 shows. Reset clears the PS[2:0] bits.

10.5 Interrupts

The timebase module can interrupt the CPU on a regular basis with a rate defined by TBR[2:0]. When the timebase counter chain rolls over, the TBIF flag is set. If the TBIE bit is set, enabling the timebase interrupt, the counter chain overflow will generate a CPU interrupt request. The interrupt vector is defined in Table 2-1 . Vector Addresses.

Interrupts must be acknowledged by writing a logic 1 to the TACK bit.

10.6 Low-Power Modes

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

10.6.1 Wait Mode

The timebase module remains active after execution of the WAIT instruction. In wait mode, the timebase register is not accessible by the CPU.

If the timebase functions are not required during wait mode, reduce the power consumption by stopping the timebase before enabling the WAIT instruction.

10.6.2 Stop Mode

The timebase module may remain active after execution of the STOP instruction if the oscillator has been enabled to operate during stop mode through the stop mode oscillator enable bit (STOP_ICLKDIS, STOP_RCLKEN, or STOP_XCLKEN) for the selected oscillator in the CONFIG2 register. The timebase module can be used in this mode to generate a periodic walk-up from stop mode.

If the oscillator has not been enabled to operate in stop mode, the timebase module will not be active during stop mode. In stop mode the timebase register is not accessible by the CPU.

If the timebase functions are not required during stop mode, reduce the power consumption by stopping the timebase before enabling the STOP instruction.

12.5 SCI Functional Description

Figure 12-5 shows the structure of the SCI.

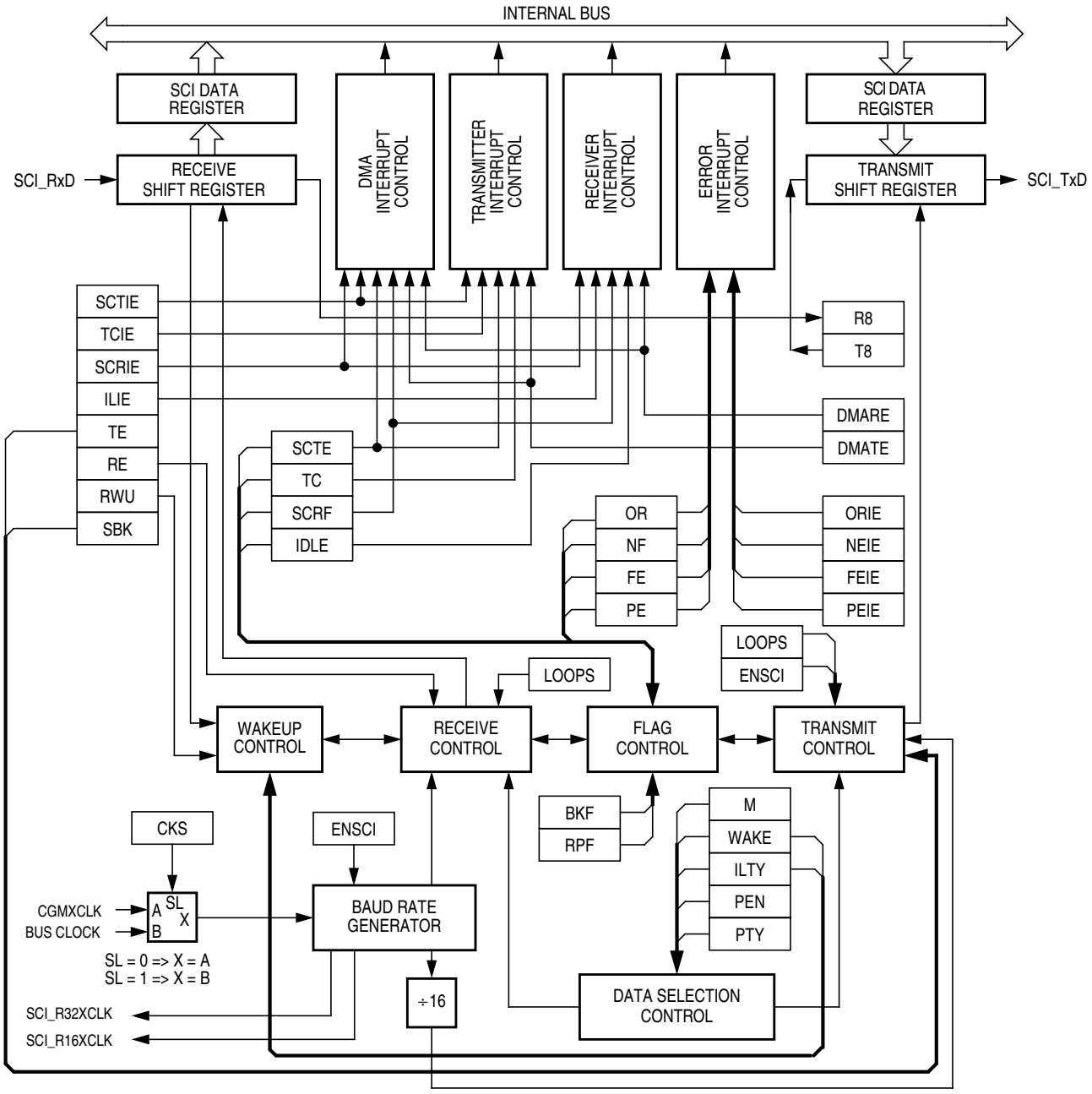


Figure 12-5. SCI Module Block Diagram

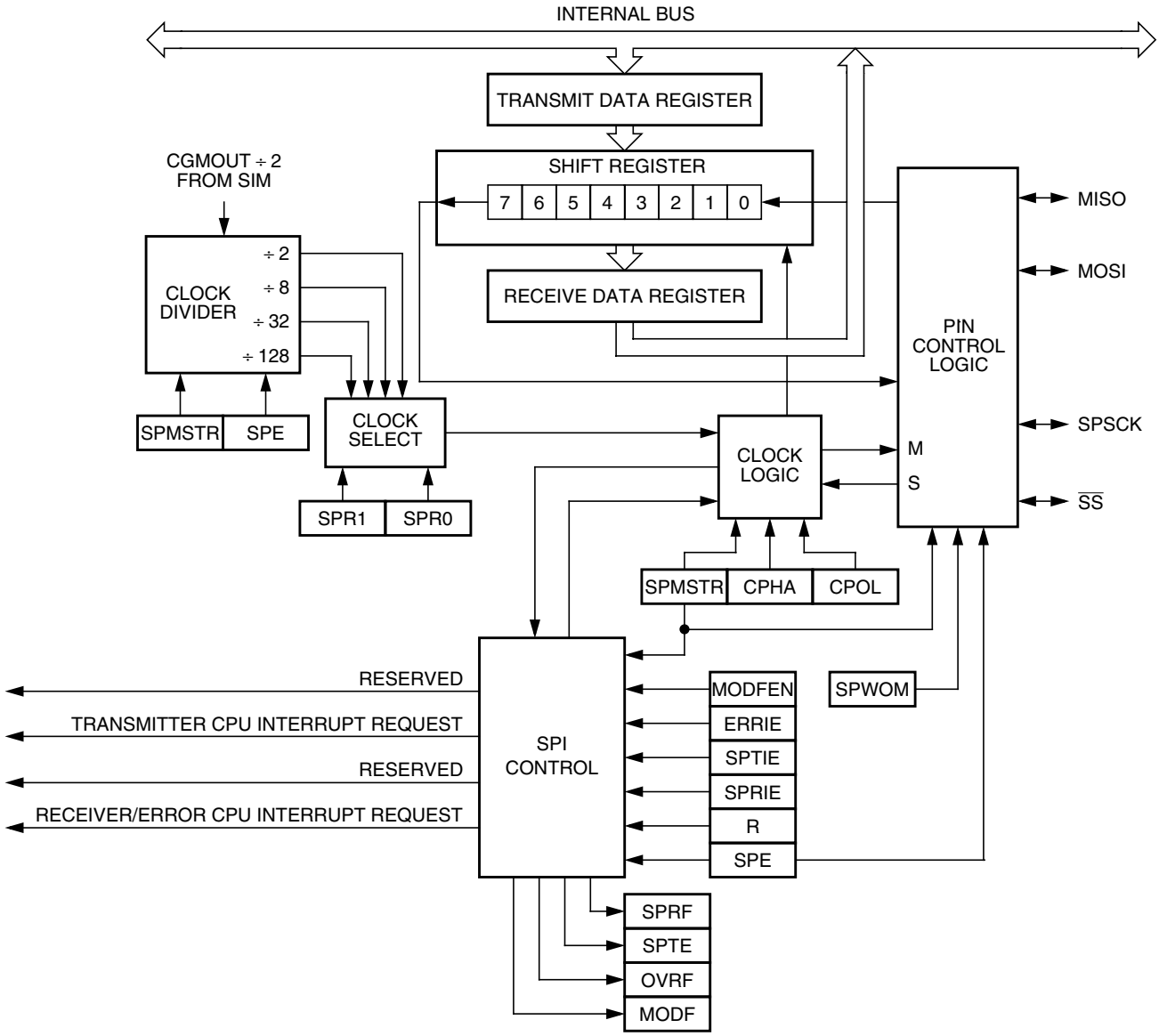


Figure 13-2. SPI Module Block Diagram

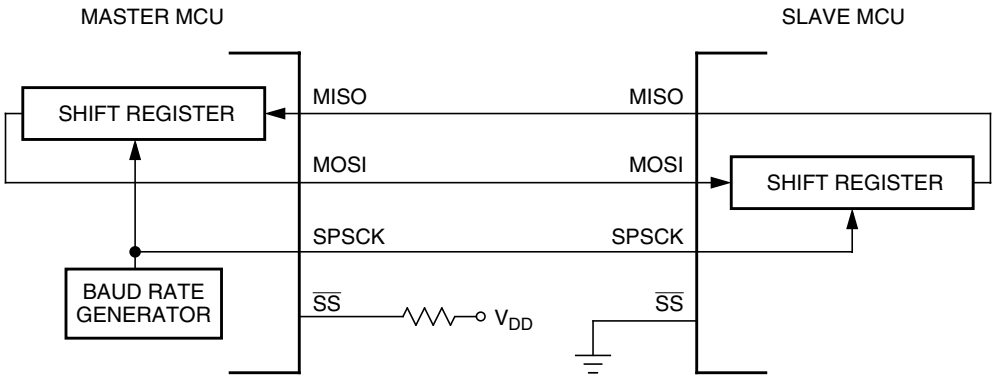
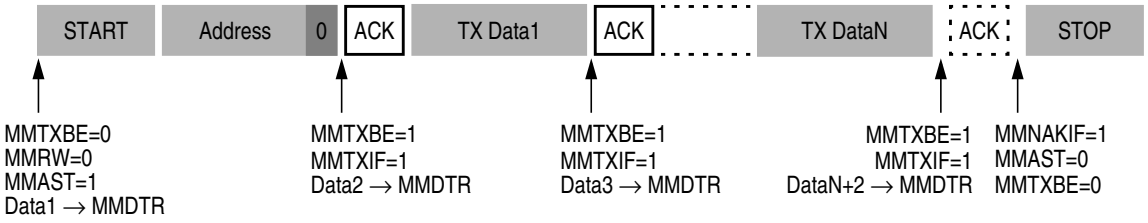


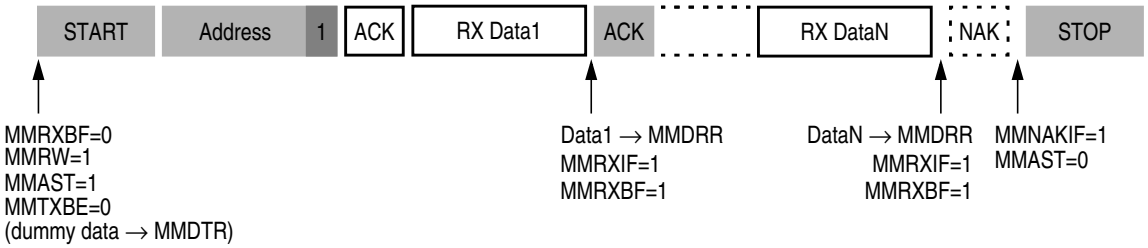
Figure 13-3. Full-Duplex Master-Slave Connections

14.7.1 Data Sequence

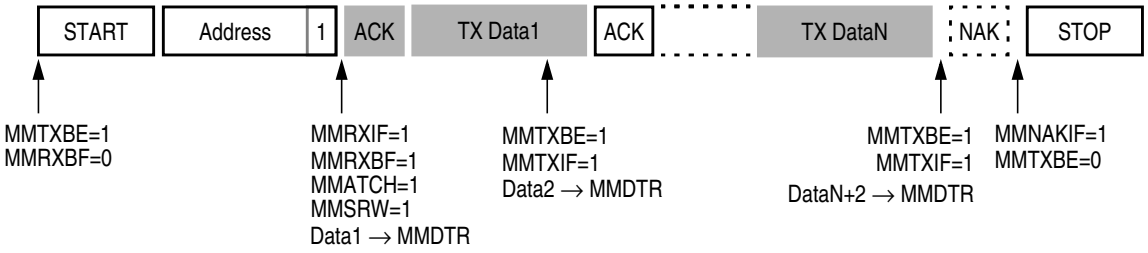
(a) Master Transmit Mode



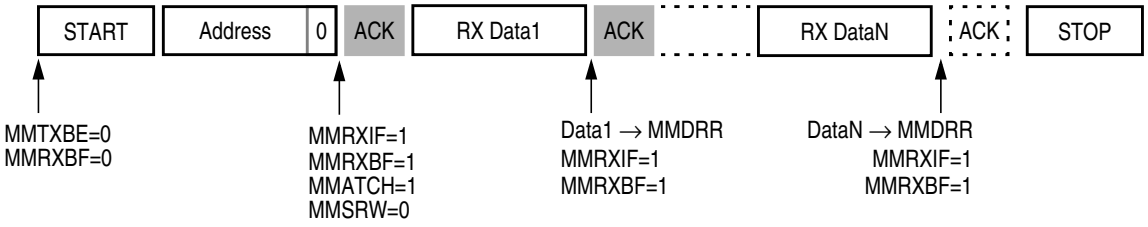
(b) Master Receive Mode



(c) Slave Transmit Mode



(d) Slave Receive Mode



Shaded data packets indicate transmissions by the MCU

Figure 14-12. Data Transfer Sequences for Master/Slave Transmit/Receive Modes

14.8 SMBus Protocols with PEC and without PEC

Following is a description of the various MMIIC bus protocols with and without a packet error code (PEC).

14.8.1 Quick Command

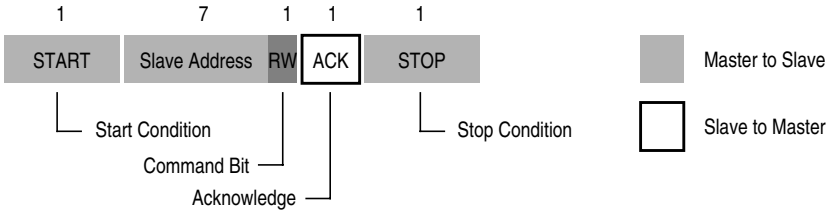


Figure 14-13. Quick Command

14.8.2 Send Byte

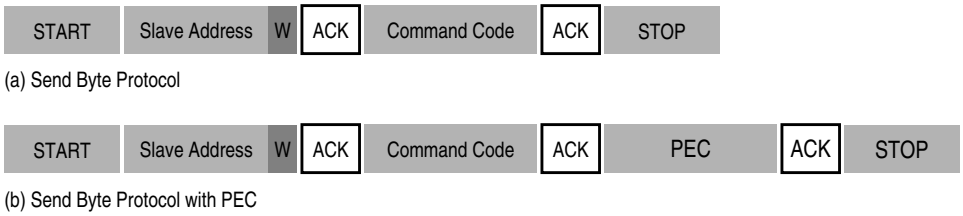


Figure 14-14. Send Byte

14.8.3 Receive Byte

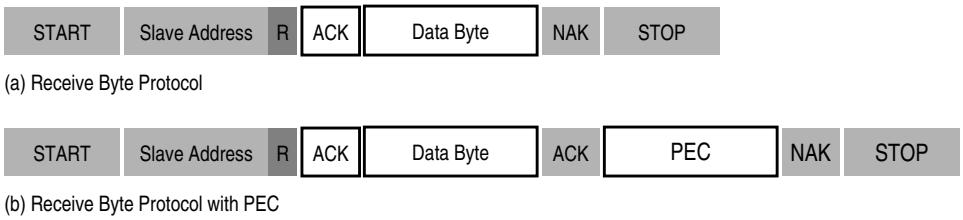


Figure 14-15. Receive Byte

SDA and SCL — Multi-Master IIC Data and Clock

The SDA and SCL pins are multi-master IIC data and clock pins. Setting the MMEN bit in the MMIIC control register 1 (MMCR1) configures the PTB0/SDA and PTB1/SCL pins for MMIIC function and overrides any control from the port I/O logic.

TxD and RxD — SCI Transmit and Receive Data

The TxD and RxD pins are SCI transmit and receive data pins. Setting the ENSCI bit in the SCI control register 1 (SCC1) configures the PTB2/TxD and PTB3/RxD pins for SCI function and overrides any control from the port I/O logic.

T1CH0 and T1CH1 — Timer 1 Channel I/O

The T1CH0 and T1CH1 pins are the TIM1 input capture/output compare pins. The edge/level select bits, ELSxB:ELSxA, determine whether the PTB4/T1CH0–PTB5/T1CH1 pins are timer channel I/O pins or general-purpose I/O pins.

T2CH0 and T2CH1 — Timer 2 Channel I/O

The T2CH0 and T2CH1 pins are the TIM2 input capture/output compare pins. The edge/level select bits, ELSxB:ELSxA, determine whether the PTB6/T2CH0–PTB7/T2CH1 pins are timer channel I/O pins or general-purpose I/O pins.

16.3.2 Data Direction Register B (DDRB)

Data direction register B determines whether each port B pin is an input or an output. Writing a logic 1 to a DDRB bit enables the output buffer for the corresponding port B pin; a logic 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 16-7. Data Direction Register B (DDRB)

DDRB[7:0] — Data Direction Register B Bits

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

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

NOTE

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

Figure 16-8 shows the port B I/O logic.

18.4 Functional Description

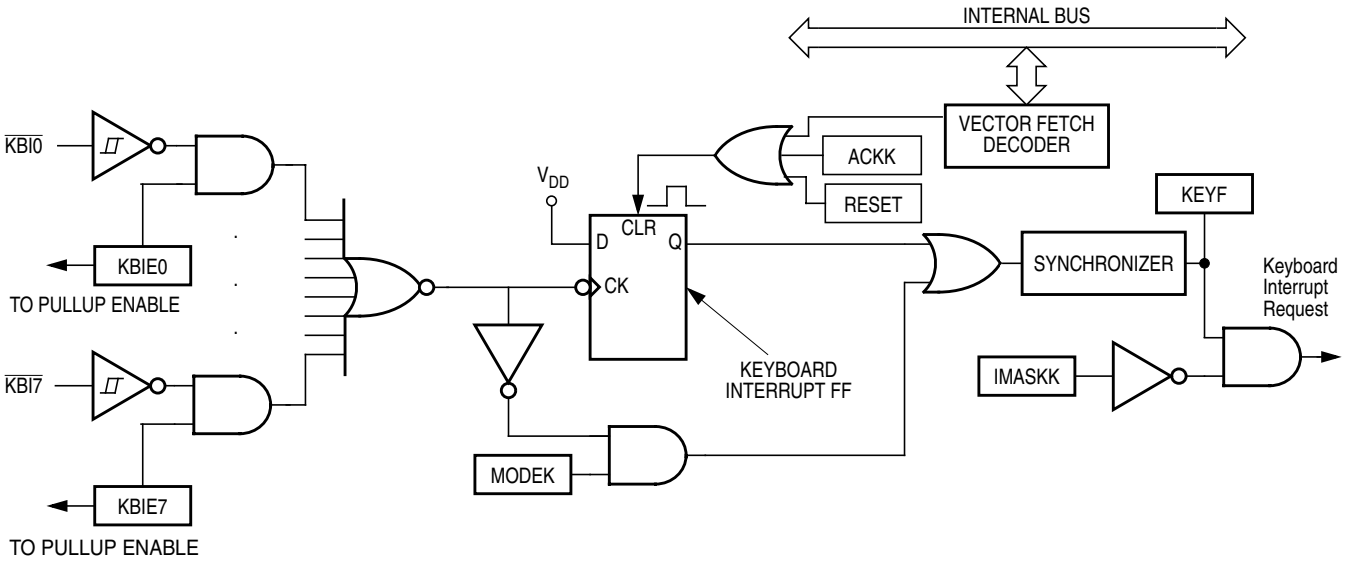


Figure 18-2. Keyboard Interrupt Block Diagram

Writing to the KBIE7–KBIE0 bits in the keyboard interrupt enable register independently enables or disables each port D pin as a keyboard interrupt pin. Enabling a keyboard interrupt pin in port D also enables its internal pull-up device. A logic 0 applied to an enabled keyboard interrupt pin latches a keyboard interrupt request.

A keyboard interrupt is latched when one or more keyboard pins goes low after all were high. The MODEK bit in the keyboard status and control register controls the triggering mode of the keyboard interrupt.

- If the keyboard interrupt is edge-sensitive only, a falling edge on a keyboard pin does not latch an interrupt request if another keyboard pin is already low. To prevent losing an interrupt request on one pin because another pin is still low, software can disable the latter pin while it is low.
- If the keyboard interrupt is falling edge- and low level-sensitive, an interrupt request is present as long as any keyboard pin is low.

If the MODEK bit is set, the keyboard interrupt pins are both falling edge- and low level-sensitive, and both of the following actions must occur to clear a keyboard interrupt request:

- Vector fetch or software clear — A vector fetch generates an interrupt acknowledge signal to clear the interrupt request. Software may generate the interrupt acknowledge signal by writing a logic 1 to the ACKK bit in the keyboard status and control register KBSCR. The ACKK bit is useful in applications that poll the keyboard interrupt pins and require software to clear the keyboard interrupt request. Writing to the ACKK bit prior to leaving an interrupt service routine can also prevent spurious interrupts due to noise. Setting ACKK does not affect subsequent transitions on the keyboard interrupt pins. A falling edge that occurs after writing to the ACKK bit latches another interrupt request. If the keyboard interrupt mask bit, IMASKK, is clear, the CPU loads the program counter with the vector address at locations $FFE0 and $FFE1.
- Return of all enabled keyboard interrupt pins to logic 1 — As long as any enabled keyboard interrupt pin is at logic 0, the keyboard interrupt remains set.

The vector fetch or software clear and the return of all enabled keyboard interrupt pins to logic 1 may occur in any order.