HC05

MC68HC05E0

TECHNICAL
DATA

freescale™
semiconductor

MC68HC05E0

High-density complementary metal oxide semiconductor (HCMOS) microcontroller unit

All Trade Marks recognized. This document contains information on new products. Specifications and information herein are subject to change without notice.

Conventions

Where abbreviations are used in the text, an explanation can be found in the glossary, at the back of this manual. Register and bit mnemonics are defined in the paragraphs describing them.

An overbar is used to designate an active-low signal, eg: $\overline{\text{RESET}}$.

Unless otherwise stated, shaded cells in a register diagram indicate that the bit is either unused or reserved; ‘u’ is used to indicate an undefined state (on reset).

TABLE OF CONTENTS

4
RESETS, INTERRUPTS AND LOW POWER MODES

5
MEMORY AND ADDRESSING

6
PARALLEL INPUT/OUTPUT PORTS

7
TIMERS

8
SERIAL INTERFACE

9
ELECTRICAL SPECIFICATIONS

LIST OF FIGURES

THIS PAGE LEFT BLANK INTENTIONALLY

LIST OF TABLES

THIS PAGE LEFT BLANK INTENTIONALLY

1
INTRODUCTION

1.1 General

The MC68HC05E0 is a high-performance fully-expandable ROM-less member of the M68HC05 Family of microcomputers. The M68HC05 CPU core has been enhanced with two powerful, independently controlled timer subsystems, and a serial interface (SI) which can operate in either SPI (Serial Peripheral Interface) or I^2C-bus compatible mode. An external 16-bit address/8-bit data expansion bus and chip-select logic are provided to allow access to external ROM, RAM and I/O. The MC68HC05E0, with 480 bytes of on-chip RAM and 36 I/O port lines, is available in a 68-pin PLCC package.

1.2 Features

- Industry-standard M68HC05 core and instruction set
- 64 kbyte address range
- 16-bit address/8-bit data expansion bus to interface to external memory and peripherals
- Address decoder provides select logic for internal and external areas of the memory map
- 480 bytes of on-chip RAM.
- 4 MHz bus frequency
- Programmable system timing control
- 36 I/O lines (four 8-bit bidirectional ports, one 4-bit bidirectional port)
- LED drive capability on 8 I/O pins (Port A)
- 6-bit timer with 8-bit prescaler
- 14-bit timer with 8-bit scaler
- Programmable Real Time Interrupt

- SPI/I^{2}C-bus interface
- Control signals (for emulation purposes)
- Temperature range: 0 - 70 ^{o}C

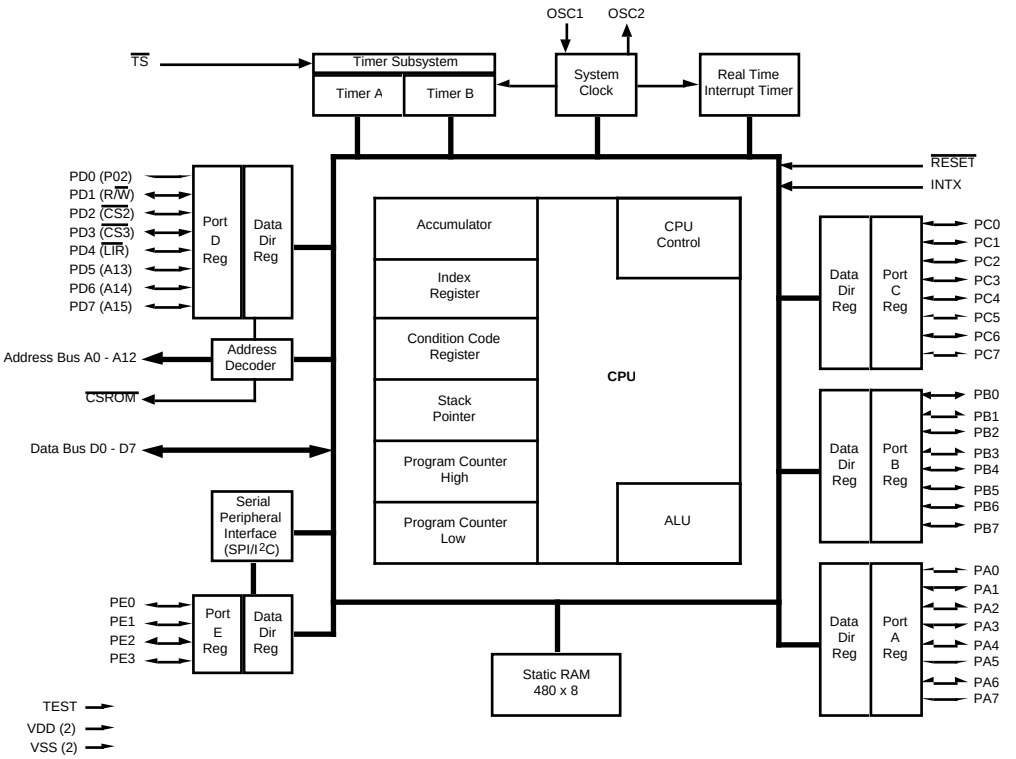


Figure 1-1 Functional Block Diagram

2
FUNCTIONAL PIN DESCRIPTION

All signal inputs on the MC68HC05E0 except $\overline{\text{RESET}}$ and OSC1 are TTL compatible.

2.1 VDD and VSS

Power is supplied to the microcontroller via two VDD and two VSS pins. VDD is the positive supply and VSS is ground.

2.2 OSC1/OSC2

These pins provide control input for an on-chip clock oscillator circuit. A crystal, ceramic resonator or external clock signal connected to these pins provides the oscillator clock. The oscillator frequency is divided by 2 to provide the internal bus frequency.

2.2.1 Crystal

The circuit shown in Figure 2-1(b) is recommended when using a crystal. The internal oscillator is designed to interface with an AT-cut parallel-resonant quartz crystal resonator in the frequency range specified for f_{OSC} (refer to Section 9.5). Use of an external CMOS oscillator is recommended when crystals outside the specified ranges are to be used. The crystal and components should be mounted as close as possible to the input pins to minimise output distortion and startup stabilization time.

2.2.2 Ceramic Resonator

A ceramic resonator may be used in place of the crystal in cost-sensitive applications. The circuit in Figure 2-1(b) is recommended when using a ceramic resonator. Figure 2-1(a) lists the

recommended capacitance and feedback resistance values. The manufacturer of the particular ceramic resonator being considered should be consulted for specific information.

2.2.3 External Clock

An external clock should be applied to the OSC1 input with the OSC2 pin not connected, as shown in Figure 2-1(d). The t_{OXOV} or t_{ILCH} specifications do not apply when using an external clock input. The equivalent specification of the external clock source should be used in lieu of t_{OXOV} or t_{ILCH}.

2.3 $\overline{\text{RESET}}$

This active low input-only pin is used to reset the MCU. Applying a logic zero to this pin forces the device to a known start-up state. An external RC-circuit can be connected to this pin to generate a power-on reset (POR). In this case, the time constant must be chosen high enough (minimum 100 ms) to allow the oscillator circuit to stabilise. This input has an internal pull-up resistor and an internal Schmitt trigger to improve noise immunity.

2.4 Port A (PA0 - PA7)

Port A is comprised of eight bidirectional pins (PA0 to PA7). The direction and state of each pin is software programmable, and each pin can drive one LED load. All pins are configured as inputs during power-on or reset.

2.5 Port B (PB0 - PB7)

Port B is comprised of eight bidirectional pins (PB0 to PB7). The direction and state of each pin is software programmable. All pins are configured as inputs during power-on or reset.

2.6 Port C (PC0 - PC7)

Port C is comprised of eight bidirectional pins (PC0 to PC7). The direction and state of each pin is software programmable. In addition, each input can be configured to support a wake-up function and trigger a processor interrupt. All pins are configured as inputs during power-on or reset.

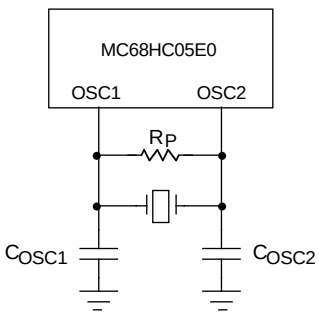
Crystal

	2 MHz	4 MHz	Units
R_{SMAX}	400	75	Ω
C_0	5	7	pF
C_1	0.008	0.012	µF
C_{OSC1}	15 - 40	15 - 30	pF
C_{OSC2}	15 - 30	15 - 25	pF
R_P	10	10	MΩ
Q	30	40	K

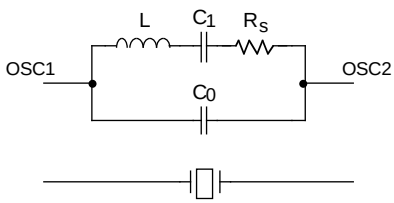
Ceramic Resonator

	2 - 4 MHz	Units
R_S (typical)	10	Ω
C_0	40	pF
C_1	4.3	pF
C_{OSC1}	30	pF
C_{OSC2}	30	pF
R_P	1 - 10	MΩ
Q	1250	–

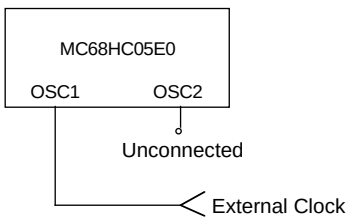
(a) Crystal/Ceramic Resonator Parameters



(b) Crystal/Ceramic Resonator Oscillator Connections



(c) Equivalent Crystal Circuit



(d) External Clock Source Connections

Figure 2-1 Oscillator Connections

2.7 Port D (PD0 - PD7)

Port D is comprised of eight bidirectional pins (PD0 to PD7). The direction and state of each pin is software programmable. Alternatively, Port D can be configured to provide address and control lines for interfacing to external memory. During power-on or reset all port pins except PD0 are defined as port inputs: PD0 is defined as an output and generates the internal bus timing signal P02.

2.8 Port E (PE0 - PE3)

Port E is comprised of four bidirectional pins (PE0 to PE3). The direction and state of each pin is software programmable. In addition, Port E can be configured to support the SPI and I^2C-bus functions. All pins are configured as inputs during power-on or reset.

2.9 $\overline{\text{CSROM}}$

This active low output signal is used as a chip select for external ROM. If the XROM bit in the Timer Control Register ($000C) is set to 1, this pin outputs a logic zero when an address in the range $3000 to $FFFF is present on the address bus. $\overline{\text{CSROM}}$ is normally gated with P02 and is only active during the “high” phase of P02. However, writing a logic zero to the XROM bit in the Timer Control Register ($000C) forces $\overline{\text{CSROM}}$ to remain permanently low throughout the full memory map ($0000-$FFFF). Clearing the XROM bit also has the effect of making all data bus lines input only. This feature is intended for use in a two chip system (MC68HC05E0 and ROM/EPROM) and helps minimise the amount of RFI generated by rapid switching of the bus and $\overline{\text{CSROM}}$ lines.

Note: Although $\overline{\text{CSROM}}$ is permanently low throughout the full memory map, data on the data bus will be ignored between addresses $0000-$01FF. This space is reserved for internal memory in the MC68HC05E0. (The XROM bit does not affect the internal memory map and Read and Write instructions can be executed as normal in this area.)

XROM	CSROM Pin	Data Bus
1	gated P02 ($3000-$FFFF)	Input/Output
0	always 0 ($0000-$FFFF)	Input only

2.10 INTX

INTX is an input pin for external interrupt sources. The interrupt type (edge or level sensitive), value (high of low), as well as interrupt masking can be selected via the Interrupt Control Register.

2.11 $\overline{TS}$

The active low $\overline{TS}$ input pin allows the internal timer functions Timer A and Timer B to be halted. This feature is particularly useful in the emulation environment.

2.12 TEST

The TEST input is only required for factory testing of internal functions. It must not be used during normal operation and must always be connected to VSS.

2.13 Expanded Address Bus (A0 – A12)

Address lines A0 to A12 are always available at these output-only pins and are directly controlled by the processor core. They are intended for accessing external memory. Additional external memory can be addressed by using these lines in conjunction with optional address lines A13 – A15, available on Port D.

2.14 External Data Bus (D0 – D7)

Data signals D0 to D7 are bidirectional signals and permit direct access to the internal data bus. The data bus lines can be configured as input only lines by clearing the XROM bit in the Timer Control Register ($000C); this feature is included to help minimise RFI.

Note: The address bus lines (A0 – A12) and the data bus lines (D0 – D7) have been designed to have a maximum driving current of 2mA, in order to minimise HF disturbance. The input and output signal levels are CMOS and TTL compatible.

THIS PAGE LEFT BLANK INTENTIONALLY

3
CPU CORE AND INSTRUCTION SET

This section provides a description of the CPU core registers, the instruction set and the addressing modes of the MC68HC05E0.

3.1 Registers

The MCU contains five registers, as shown in the programming model of Figure 3-1. The interrupt stacking order is shown in Figure 3-2.

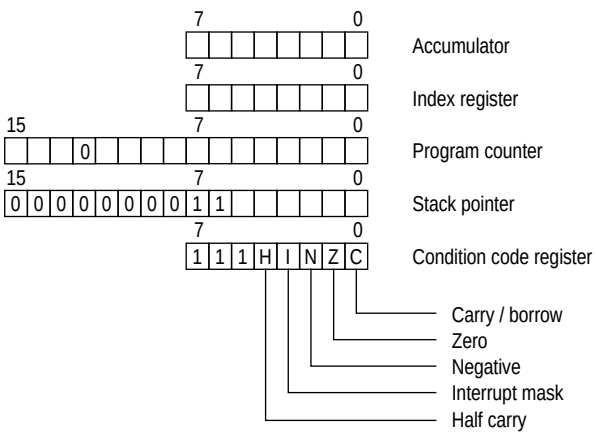


Figure 3-1 Programming model

3.1.1 Accumulator (A)

The accumulator is a general purpose 8-bit register used to hold operands and results of arithmetic calculations or data manipulations.

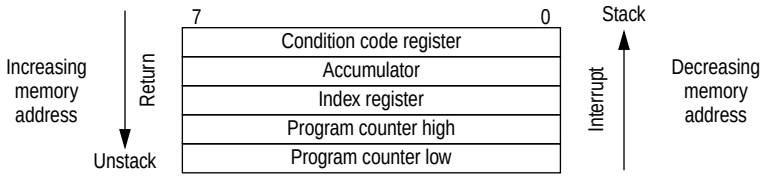


Figure 3-2 Stacking order

3.1.2 Index register (X)

The index register is an 8-bit register, which can contain the indexed addressing value used to create an effective address. The index register may also be used as a temporary storage area.

3.1.3 Program counter (PC)

The program counter is a 16-bit register, which contains the address of the next byte to be fetched. Although the M68HC05 CPU core can address 64K bytes of memory, the actual address range of the MC68HC05E0 is limited to 4K bytes. The four most significant bits of the program counter are therefore not used and are permanently set to zero.

3.1.4 Stack pointer (SP)

The stack pointer is a 16-bit register, which contains the address of the next free location on the stack. During an MCU reset or the reset stack pointer (RSP) instruction, the stack pointer is set to location $00FF. The stack pointer is then decremented as data is pushed onto the stack and incremented as data is pulled from the stack.

When accessing memory, the ten most significant bits are permanently set to 0000000011. These ten bits are appended to the six least significant register bits to produce an address within the range of $00C0 to $00FF. Subroutines and interrupts may use up to 64 (decimal) locations. If 64 locations are exceeded, the stack pointer wraps around and overwrites the previously stored information. A subroutine call occupies two locations on the stack; an interrupt uses five locations.

3.1.5 Condition code register (CCR)

The CCR is a 5-bit register in which four bits are used to indicate the results of the instruction just executed, and the fifth bit indicates whether interrupts are masked. These bits can be individually tested by a program, and specific actions can be taken as a result of their state. Each bit is explained in the following paragraphs.

Half carry (H)

This bit is set during ADD and ADC operations to indicate that a carry occurred between bits 3 and 4.

Interrupt (I)

When this bit is set, all maskable interrupts are masked. If an interrupt occurs while this bit is set, the interrupt is latched and remains pending until the interrupt bit is cleared.

Negative (N)

When set, this bit indicates that the result of the last arithmetic, logical, or data manipulation was negative.

Zero (Z)

When set, this bit indicates that the result of the last arithmetic, logical, or data manipulation was zero.

Carry/borrow (C)

When set, this bit indicates that a carry or borrow out of the arithmetic logical unit (ALU) occurred during the last arithmetic operation. This bit is also affected during bit test and branch instructions and during shifts and rotates.

3.2 Instruction set

The MCU has a set of 62 basic instructions. They can be grouped into five different types as follows:

- Register/memory
- Read/modify/write
- Branch
- Bit manipulation
- Control

The following paragraphs briefly explain each type. All the instructions within a given type are presented in individual tables.

This MCU uses all the instructions available in the M146805 CMOS family plus one more: the unsigned multiply (MUL) instruction. This instruction allows unsigned multiplication of the contents of the accumulator (A) and the index register (X). The high-order product is then stored in the index register and the low-order product is stored in the accumulator. A detailed definition of the MUL instruction is shown in Table 3-1.

3.2.1 Register/memory Instructions

Most of these instructions use two operands. The first operand is either the accumulator or the index register. The second operand is obtained from memory using one of the addressing modes. The jump unconditional (JMP) and jump to subroutine (JSR) instructions have no register operand. Refer to Table 3-2 for a complete list of register/memory instructions.

3.2.2 Branch instructions

These instructions cause the program to branch if a particular condition is met; otherwise, no operation is performed. Branch instructions are two-byte instructions. Refer to Table 3-3.

3.2.3 Bit manipulation instructions

The MCU can set or clear any writable bit that resides in the first 256 bytes of the memory space (page 0). All port data and data direction registers, timer and serial interface registers, control/status registers and a portion of the on-chip RAM reside in page 0. An additional feature allows the software to test and branch on the state of any bit within these locations. The bit set, bit clear, bit test and branch functions are all implemented with single instructions. For the test and branch instructions, the value of the bit tested is also placed in the carry bit of the condition code register. Refer to Table 3-4.

3.2.4 Read/modify/write instructions

These instructions read a memory location or a register, modify or test its contents, and write the modified value back to memory or to the register. The test for negative or zero (TST) instruction is an exception to this sequence of reading, modifying and writing, since it does not modify the value. Refer to Table 3-5 for a complete list of read/modify/write instructions.

3.2.5 Control instructions

These instructions are register reference instructions and are used to control processor operation during program execution. Refer to Table 3-6 for a complete list of control instructions.

3.2.6 Tables

Tables for all the instruction types listed above follow. In addition there is a complete alphabetical listing of all the instructions (see Table 3-7), and an opcode map for the instruction set of the M68HC05 MCU family (see Table 3-8).

Table 3-1 MUL instruction

Operation	**X:A ← X*A**			
Description	Multiplies the eight bits in the index register by the eight bits in the accumulator and places the 16-bit result in the concatenated accumulator and index register.			
Condition codes	H : Cleared I : Not affected N : Not affected Z : Not affected C : Cleared			
Source	MUL			
Form	Addressing mode	Cycles	Bytes	Opcode
	Inherent	11	1	$42

Table 3-2 Register/memory instructions

		Addressing modes																	
		Immediate			Direct			Extended			Indexed (no offset)			Indexed (8-bit offset)			Indexed (16-bit offset)		
Function	**Mnemonic**	**Opcode**	**# Bytes**	**# Cycles**	**Opcode**	**# Bytes**	**# Cycles**	**Opcode**	**# Bytes**	**# Cycles**	**Opcode**	**# Bytes**	**# Cycles**	**Opcode**	**# Bytes**	**# Cycles**	**Opcode**	**# Bytes**	**# Cycles**
Load A from memory	LDA	A6	2	2	B6	2	3	C6	3	4	F6	1	3	E6	2	4	D6	3	5
Load X from memory	LDX	AE	2	2	BE	2	3	CE	3	4	FE	1	3	EE	2	4	DE	3	5
Store A in memory	STA				B7	2	4	C7	3	5	F7	1	4	E7	2	5	D7	3	6
Store X in memory	STX				BF	2	4	CF	3	5	FF	1	4	EF	2	5	DF	3	6
Add memory to A	ADD	AB	2	2	BB	2	3	CB	3	4	FB	1	3	EB	2	4	DB	3	5
Add memory and carry to A	ADC	A9	2	2	B9	2	3	C9	3	4	F9	1	3	E9	2	4	D9	3	5
Subtract memory	SUB	A0	2	2	B0	2	3	C0	3	4	F0	1	3	E0	2	4	D0	3	5
Subtract memory from A with borrow	SBC	A2	2	2	B2	2	3	C2	3	4	F2	1	3	E2	2	4	D2	3	5
AND memory with A	AND	A4	2	2	B4	2	3	C4	3	4	F4	1	3	E4	2	4	D4	3	5
OR memory with A	ORA	AA	2	2	BA	2	3	CA	3	4	FA	1	3	EA	2	4	DA	3	5
Exclusive OR memory with A	EOR	A8	2	2	B8	2	3	C8	3	4	F8	1	3	E8	2	4	D8	3	5
Arithmetic compare A with memory	CMP	A1	2	2	B1	2	3	C1	3	4	F1	1	3	E1	2	4	D1	3	5
Arithmetic compare X with memory	CPX	A3	2	2	B3	2	3	C3	3	4	F3	1	3	E3	2	4	D3	3	5
Bit test memory with A (logical compare)	BIT	A5	2	2	B5	2	3	C5	3	4	F5	1	3	E5	2	4	D5	3	5
Jump unconditional	JMP				BC	2	2	CC	3	3	FC	1	2	EC	2	3	DC	3	4
Jump to subroutine	JSR				BD	2	5	CD	3	6	FD	1	5	ED	2	6	DD	3	7

Table 3-3 Branch instructions

		Relative addressing mode		
Function	**Mnemonic**	**Opcode**	**# Bytes**	**# Cycles**
Branch always	BRA	20	2	3
Branch never	BRN	21	2	3
Branch if higher	BHI	22	2	3
Branch if lower or same	BLS	23	2	3
Branch if carry clear	BCC	24	2	3
(Branch if higher or same)	(BHS)	24	2	3
Branch if carry set	BCS	25	2	3
(Branch if lower)	(BLO)	25	2	3
Branch if not equal	BNE	26	2	3
Branch if equal	BEQ	27	2	3
Branch if half carry clear	BHCC	28	2	3
Branch if half carry set	BHCS	29	2	3
Branch if plus	BPL	2A	2	3
Branch if minus	BMI	2B	2	3
Branch if interrupt mask bit is clear	BMC	2C	2	3
Branch if interrupt mask bit is set	BMS	2D	2	3
Branch if interrupt line is low	BIL	2E	2	3
Branch if interrupt line is high	BIH	2F	2	3
Branch to subroutine	BSR	AD	2	6

Table 3-4 Bit manipulation instructions

		Addressing modes					
		Bit set/clear			Bit test and branch		
Function	**Mnemonic**	**Opcode**	**# Bytes**	**# Cycles**	**Opcode**	**# Bytes**	**# Cycles**
Branch if bit n is set	BRSET n (n=0–7)				2•n	3	5
Branch if bit n is clear	BRCLR n (n=0–7)				01+2•n	3	5
Set bit n	BSET n (n=0–7)	10+2•n	2	5			
Clear bit n	BCLR n (n=0–7)	11+2•n	2	5			

Table 3-5 Read/modify/write instructions

		Addressing modes														
		Inherent (A)			Inherent (X)			Direct			Indexed (no offset)			Indexed (8-bit offset)		
Function	Mnemonic	Opcode	# Bytes	# Cycles	Opcode	# Bytes	# Cycles	Opcode	# Bytes	# Cycles	Opcode	# Bytes	# Cycles	Opcode	# Bytes	# Cycles
Increment	INC	4C	1	3	5C	1	3	3C	2	5	7C	1	5	6C	2	6
Decrement	DEC	4A	1	3	5A	1	3	3A	2	5	7A	1	5	6A	2	6
Clear	CLR	4F	1	3	5F	1	3	3F	2	5	7F	1	5	6F	2	6
Complement	COM	43	1	3	53	1	3	33	2	5	73	1	5	63	2	6
Negate (two's complement)	NEG	40	1	3	50	1	3	30	2	5	70	1	5	60	2	6
Rotate left through carry	ROL	49	1	3	59	1	3	39	2	5	79	1	5	69	2	6
Rotate right through carry	ROR	46	1	3	56	1	3	36	2	5	76	1	5	66	2	6
Logical shift left	LSL	48	1	3	58	1	3	38	2	5	78	1	5	68	2	6
Logical shift right	LSR	44	1	3	54	1	3	34	2	5	74	1	5	64	2	6
Arithmetic shift right	ASR	47	1	3	57	1	3	37	2	5	77	1	5	67	2	6
Test for negative or zero	TST	4D	1	3	5D	1	3	3D	2	4	7D	1	4	6D	2	5
Multiply	MUL	42	1	11												

Table 3-6 Control instructions

		Inherent addressing mode		
Function	Mnemonic	Opcode	# Bytes	# Cycles
Transfer A to X	TAX	97	1	2
Transfer X to A	TXA	9F	1	2
Set carry bit	SEC	99	1	2
Clear carry bit	CLC	98	1	2
Set interrupt mask bit	SEI	9B	1	2
Clear interrupt mask bit	CLI	9A	1	2
Software interrupt	SWI	83	1	10
Return from subroutine	RTS	81	1	6
Return from interrupt	RTI	80	1	9
Reset stack pointer	RSP	9C	1	2
No-operation	NOP	9D	1	2
Stop	STOP	8E	1	2
Wait	WAIT	8F	1	2

Table 3-7 Instruction set

Mnemonic	Addressing modes										Condition codes				
	INH	IMM	DIR	EXT	REL	IX	IX1	IX2	BSC	BTB	H	I	N	Z	C
ADC											◊	•	◊	◊	◊
ADD											◊	•	◊	◊	◊
AND											•	•	◊	◊	•
ASL											•	•	◊	◊	◊
ASR											•	•	◊	◊	◊
BCC											•	•	•	•	•
BCLR											•	•	•	•	•
BCS											•	•	•	•	•
BEQ											•	•	•	•	•
BHCC											•	•	•	•	•
BHCS											•	•	•	•	•
BHI											•	•	•	•	•
BHS											•	•	•	•	•
BIH											•	•	•	•	•
BIL											•	•	•	•	•
BIT											•	•	◊	◊	•
BLO											•	•	•	•	•
BLS											•	•	•	•	•
BMC											•	•	•	•	•
BMI											•	•	•	•	•
BMS											•	•	•	•	•
BNE											•	•	•	•	•
BPL											•	•	•	•	•
BRA											•	•	•	•	•
BRN											•	•	•	•	•
BRCLR											•	•	•	•	◊
BRSET											•	•	•	•	◊
BSET											•	•	•	•	•
BSR											•	•	•	•	•
CLC											•	•	•	•	0
CLI											•	0	•	•	•
CLR											•	•	0	1	•
CMP											•	•	◊	◊	◊

Address mode abbreviations

BSC	Bit set/clear	IMM	Immediate
BTB	Bit test & branch	IX	Indexed (no offset)
DIR	Direct	IX1	Indexed, 1 byte offset
EXT	Extended	IX2	Indexed, 2 byte offset
INH	Inherent	REL	Relative

Not implemented

Condition code symbols

H	Half carry (from bit 3)	◊	Tested and set if true, cleared otherwise
I	Interrupt mask	•	Not affected
N	Negate (sign bit)	?	Load CCR from stack
Z	Zero	0	Cleared
C	Carry/borrow	1	Set

Table 3-7 Instruction set (Continued)

Mnemonic	Addressing modes										Condition codes				
	INH	IMM	DIR	EXT	REL	IX	IX1	IX2	BSC	BTB	H	I	N	Z	C
COM											•	•	◊	◊	1
CPX											•	•	◊	◊	◊
DEC											•	•	◊	◊	•
EOR											•	•	◊	◊	•
INC											•	•	◊	◊	•
JMP											•	•	•	•	•
JSR											•	•	•	•	•
LDA											•	•	◊	◊	•
LDX											•	•	◊	◊	•
LSL											•	•	◊	◊	◊
LSR											•	•	0	◊	◊
MUL											0	•	•	•	0
NEG											•	•	◊	◊	◊
NOP											•	•	•	•	•
ORA											•	•	◊	◊	•
ROL											•	•	◊	◊	◊
ROR											•	•	◊	◊	◊
RSP											•	•	•	•	•
RTI											?	?	?	?	?
RTS											•	•	•	•	•
SBC											•	•	◊	◊	◊
SEC											•	•	•	•	1
SEI											•	1	•	•	•
STA											•	•	◊	◊	•
STOP											•	0	•	•	•
STX											•	•	◊	◊	•
SUB											•	•	◊	◊	◊
SWI											•	1	•	•	•
TAX											•	•	•	•	•
TST											•	•	◊	◊	•
TXA											•	•	•	•	•
WAIT											•	0	•	•	•

Address mode abbreviations

BSC	Bit set/clear	IMM	Immediate
BTB	Bit test & branch	IX	Indexed (no offset)
DIR	Direct	IX1	Indexed, 1 byte offset
EXT	Extended	IX2	Indexed, 2 byte offset
INH	Inherent	REL	Relative
	Not implemented		

Condition code symbols

H	Half carry (from bit 3)	◊	Tested and set if true, cleared otherwise
I	Interrupt mask	•	Not affected
N	Negate (sign bit)	?	Load CCR from stack
Z	Zero	0	Cleared
C	Carry/borrow	1	Set

Table 3-8 M68HC05 opcode map

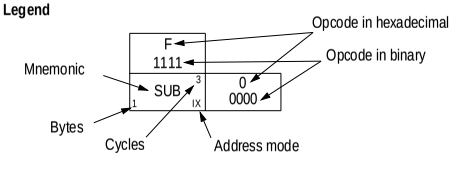
	Bit manipulation		Branch	Read/modify/write					Control		Register/memory						
	BTB	BSC	REL	DIR	INH	INH	IX1	IX	INH	INH	IMM	DIR	EXT	IX2	IX1	IX	
High / Low	0 0000	1 0001	2 0010	3 0011	4 0100	5 0101	6 0110	7 0111	8 1000	9 1001	A 1010	B 1011	C 1100	D 1101	E 1110	F 1111	High / Low
0 0000	BRSET0 5 3 BTB	BSET0 5 2 BSC	BRA 3 2 REL	NEG 5 2 DIR	NEGA 3 1 INH	NEGX 3 1 INH	NEG 6 2 IX1	NEG 5 1 IX	RTI 9 1 INH		SUB 2 2 IMM	SUB 3 2 DIR	SUB 4 3 EXT	SUB 5 3 IX2	SUB 4 2 IX1	SUB 3 1 IX	0 0000
1 0001	BRCLR0 5 3 BTB	BCLR0 5 2 BSC	BRN 3 2 REL						RTS 6 1 INH		CMP 2 2 IMM	CMP 3 2 DIR	CMP 4 3 EXT	CMP 5 3 IX2	CMP 4 2 IX1	CMP 3 1 IX	1 0001
2 0010	BRSET1 5 3 BTB	BSET1 5 2 BSC	BHI 3 2 REL		MUL 11 1 INH						SBC 2 2 IMM	SBC 3 2 DIR	SBC 4 3 EXT	SBC 5 3 IX2	SBC 4 2 IX1	SBC 3 1 IX	2 0010
3 0011	BRCLR1 5 3 BTB	BCLR1 5 2 BSC	BLS 3 2 REL	COM 5 2 DIR	COMA 3 1 INH	COMX 3 1 INH	COM 6 2 IX1	COM 5 1 IX	SWI 10 1 INH		CPX 2 2 IMM	CPX 3 2 DIR	CPX 4 3 EXT	CPX 5 3 IX2	CPX 4 2 IX1	CPX 3 1 IX	3 0011
4 0100	BRSET2 5 3 BTB	BSET2 5 2 BSC	BCC 3 2 REL	LSR 5 2 DIR	LSRA 3 1 INH	LSRX 3 1 INH	LSR 6 2 IX1	LSR 5 1 IX			AND 2 2 IMM	AND 3 2 DIR	AND 4 3 EXT	AND 5 3 IX2	AND 4 2 IX1	AND 3 1 IX	4 0100
5 0101	BRCLR2 5 3 BTB	BCLR2 5 2 BSC	BCS 3 2 REL								BIT 2 2 IMM	BIT 3 2 DIR	BIT 4 3 EXT	BIT 5 3 IX2	BIT 4 2 IX1	BIT 3 1 IX	5 0101
6 0110	BRSET3 5 3 BTB	BSET3 5 2 BSC	BNE 3 2 REL	ROR 5 2 DIR	RORA 3 1 INH	RORX 3 1 INH	ROR 6 2 IX1	ROR 5 1 IX			LDA 2 2 IMM	LDA 3 2 DIR	LDA 4 3 EXT	LDA 5 3 IX2	LDA 4 2 IX1	LDA 3 1 IX	6 0110
7 0111	BRCLR3 5 3 BTB	BCLR3 5 2 BSC	BEQ 3 2 REL	ASR 5 2 DIR	ASRA 3 1 INH	ASRX 3 1 INH	ASR 6 2 IX1	ASR 5 1 IX		TAX 2 1 INH		STA 4 2 DIR	STA 5 3 EXT	STA 6 3 IX2	STA 5 2 IX1	STA 4 1 IX	7 0111
8 1000	BRSET4 5 3 BTB	BSET4 5 2 BSC	BHCC 3 2 REL	LSL 5 2 DIR	LSLA 3 1 INH	LSLX 3 1 INH	LSL 6 2 IX1	LSL 5 1 IX		CLC 2 1 INH	EOR 2 2 IMM	EOR 3 2 DIR	EOR 4 3 EXT	EOR 5 3 IX2	EOR 4 2 IX1	EOR 3 1 IX	8 1000
9 1001	BRCLR4 5 3 BTB	BCLR4 5 2 BSC	BHCS 3 2 REL	ROL 5 2 DIR	ROLA 3 1 INH	ROLX 3 1 INH	ROL 6 2 IX1	ROL 5 1 IX		SEC 2 1 INH	ADC 2 2 IMM	ADC 3 2 DIR	ADC 4 3 EXT	ADC 5 3 IX2	ADC 4 2 IX1	ADC 3 1 IX	9 1001
A 1010	BRSET5 5 3 BTB	BSET5 5 2 BSC	BPL 3 2 REL	DEC 5 2 DIR	DECA 3 1 INH	DECX 3 1 INH	DEC 6 2 IX1	DEC 5 1 IX		CLI 2 1 INH	ORA 2 2 IMM	ORA 3 2 DIR	ORA 4 3 EXT	ORA 5 3 IX2	ORA 4 2 IX1	ORA 3 1 IX	A 1010
B 1011	BRCLR5 5 3 BTB	BCLR5 5 2 BSC	BMI 3 2 REL							SEI 2 1 INH	ADD 2 2 IMM	ADD 3 2 DIR	ADD 4 3 EXT	ADD 5 3 IX2	ADD 4 2 IX1	ADD 3 1 IX	B 1011
C 1100	BRSET6 5 3 BTB	BSET6 5 2 BSC	BMC 3 2 REL	INC 5 2 DIR	INCA 3 1 INH	INCX 3 1 INH	INC 6 2 IX1	INC 5 1 IX		RSP 2 1 INH		JMP 2 2 DIR	JMP 3 3 EXT	JMP 4 3 IX2	JMP 3 2 IX1	JMP 2 1 IX	C 1100
D 1101	BRCLR6 5 3 BTB	BCLR6 5 2 BSC	BMS 3 2 REL	TST 4 2 DIR	TSTA 3 1 INH	TSTX 3 1 INH	TST 5 2 IX1	TST 4 1 IX		NOP 2 1 INH	BSR 6 2 REL	JSR 5 2 DIR	JSR 6 3 EXT	JSR 7 3 IX2	JSR 6 2 IX1	JSR 5 1 IX	D 1101
E 1110	BRSET7 5 3 BTB	BSET7 5 2 BSC	BIL 3 2 REL						STOP 2 1 INH		LDX 2 2 IMM	LDX 3 2 DIR	LDX 4 3 EXT	LDX 5 3 IX2	LDX 4 2 IX1	LDX 3 1 IX	E 1110
F 1111	BRCLR7 5 3 BTB	BCLR7 5 2 BSC	BIH 3 2 REL	CLR 5 2 DIR	CLRA 3 1 INH	CLRX 3 1 INH	CLR 6 2 IX1	CLR 5 1 IX	WAIT 2 1 INH	TXA 2 1 INH		STX 4 2 DIR	STX 5 3 EXT	STX 6 3 IX2	STX 5 2 IX1	STX 4 1 IX	F 1111

Abbreviations for address modes and registers

BSC	Bit set/clear	IX	Indexed (no offset)
BTB	Bit test and branch	IX1	Indexed, 1 byte (8-bit) offset
DIR	Direct	IX2	Indexed, 2 byte (16-bit) offset
EXT	Extended	REL	Relative
INH	Inherent	A	Accumulator
IMM	Immediate	X	Index register

Not implemented

Legend



3.3 Addressing modes

Ten different addressing modes provide programmers with the flexibility to optimize their code for all situations. The various indexed addressing modes make it possible to locate data tables, code conversion tables and scaling tables anywhere in the memory space. Short indexed accesses are single byte instructions; the longest instructions (three bytes) enable access to tables throughout memory. Short absolute (direct) and long absolute (extended) addressing are also included. One or two byte direct addressing instructions access all data bytes in most applications. Extended addressing permits jump instructions to reach all memory locations.

The term ‘effective address’ (EA) is used in describing the various addressing modes. The effective address is defined as the address from which the argument for an instruction is fetched or stored. The ten addressing modes of the processor are described below. Parentheses are used to indicate ‘contents of’ the location or register referred to. For example, (PC) indicates the contents of the location pointed to by the PC (program counter). An arrow indicates ‘is replaced by’ and a colon indicates concatenation of two bytes. For additional details and graphical illustrations, refer to the ***M6805 HMOS/M146805 CMOS Family Microcomputer/ Microprocessor User's Manual*** or to the ***M68HC05 Applications Guide***.

3.3.1 Inherent

In the inherent addressing mode, all the information necessary to execute the instruction is contained in the opcode. Operations specifying only the index register or accumulator, as well as the control instruction, with no other arguments are included in this mode. These instructions are one byte long.

3.3.2 Immediate

In the immediate addressing mode, the operand is contained in the byte immediately following the opcode. The immediate addressing mode is used to access constants that do not change during program execution (e.g. a constant used to initialize a loop counter).

$$EA = PC+1;\ PC \leftarrow PC+2$$

3.3.3 Direct

In the direct addressing mode, the effective address of the argument is contained in a single byte following the opcode byte. Direct addressing allows the user to directly address the lowest 256 bytes in memory with a single two-byte instruction.

$$EA = (PC+1);\ PC \leftarrow PC+2$$
$$\text{Address bus high} \leftarrow 0;\ \text{Address bus low} \leftarrow (PC+1)$$

3.3.4 Extended

In the extended addressing mode, the effective address of the argument is contained in the two bytes following the opcode byte. Instructions with extended addressing mode are capable of referencing arguments anywhere in memory with a single three-byte instruction. When using the Motorola assembler, the user need not specify whether an instruction uses direct or extended addressing. The assembler automatically selects the short form of the instruction.

EA = (PC+1):(PC+2); PC ← PC+3
Address bus high ← (PC+1); Address bus low ← (PC+2)

3.3.5 Indexed, no offset

In the indexed, no offset addressing mode, the effective address of the argument is contained in the 8-bit index register. This addressing mode can access the first 256 memory locations. These instructions are only one byte long. This mode is often used to move a pointer through a table or to hold the address of a frequently referenced RAM or I/O location.

EA = X; PC ← PC+1
Address bus high ← 0; Address bus low ← X

3.3.6 Indexed, 8-bit offset

In the indexed, 8-bit offset addressing mode, the effective address is the sum of the contents of the unsigned 8-bit index register and the unsigned byte following the opcode. Therefore the operand can be located anywhere within the lowest 511 memory locations. This addressing mode is useful for selecting the mth element in an n element table.

EA = X+(PC+1); PC ← PC+2
Address bus high ← K; Address bus low ← X+(PC+1)
where K = the carry from the addition of X and (PC+1)

3.3.7 Indexed, 16-bit offset

In the indexed, 16-bit offset addressing mode, the effective address is the sum of the contents of the unsigned 8-bit index register and the two unsigned bytes following the opcode. This address mode can be used in a manner similar to indexed, 8-bit offset except that this three-byte instruction allows tables to be anywhere in memory. As with direct and extended addressing, the Motorola assembler determines the shortest form of indexed addressing.

EA = X+[(PC+1):(PC+2)]; PC ← PC+3
Address bus high ← (PC+1)+K; Address bus low ← X+(PC+2)
where K = the carry from the addition of X and (PC+2)

3.3.8 Relative

The relative addressing mode is only used in branch instructions. In relative addressing, the contents of the 8-bit signed byte (the offset) following the opcode are added to the PC if, and only if, the branch conditions are true. Otherwise, control proceeds to the next instruction. The span of relative addressing is from –126 to +129 from the opcode address. The programmer need not calculate the offset when using the Motorola assembler, since it calculates the proper offset and checks to see that it is within the span of the branch.

EA = PC+2+(PC+1); PC ← EA if branch taken;
otherwise EA = PC ← PC+2

3.3.9 Bit set/clear

In the bit set/clear addressing mode, the bit to be set or cleared is part of the opcode. The byte following the opcode specifies the address of the byte in which the specified bit is to be set or cleared. Any read/write bit in the first 256 locations of memory, including I/O, can be selectively set or cleared with a single two-byte instruction.

EA = (PC+1); PC ← PC+2
Address bus high ← 0; Address bus low ← (PC+1)

3.3.10 Bit test and branch

The bit test and branch addressing mode is a combination of direct addressing and relative addressing. The bit to be tested and its condition (set or clear) is included in the opcode. The address of the byte to be tested is in the single byte immediately following the opcode byte (EA1). The signed relative 8-bit offset in the third byte (EA2) is added to the PC if the specified bit is set or cleared in the specified memory location. This single three-byte instruction allows the program to branch based on the condition of any readable bit in the first 256 locations of memory. The span of branch is from –125 to +130 from the opcode address. The state of the tested bit is also transferred to the carry bit of the condition code register.

EA1 = (PC+1); PC ← PC+2
Address bus high ← 0; Address bus low ← (PC+1)
EA2 = PC+3+(PC+2); PC ← EA2 if branch taken;
otherwise PC ← PC+3

THIS PAGE LEFT BLANK INTENTIONALLY

4

RESETS, INTERRUPTS AND LOW POWER MODES

4.1 Resets

The MCU can be reset by the initial power-on reset function or by an active low level applied to the $\overline{\text{RESET}}$ pin (see Figure 4-1).

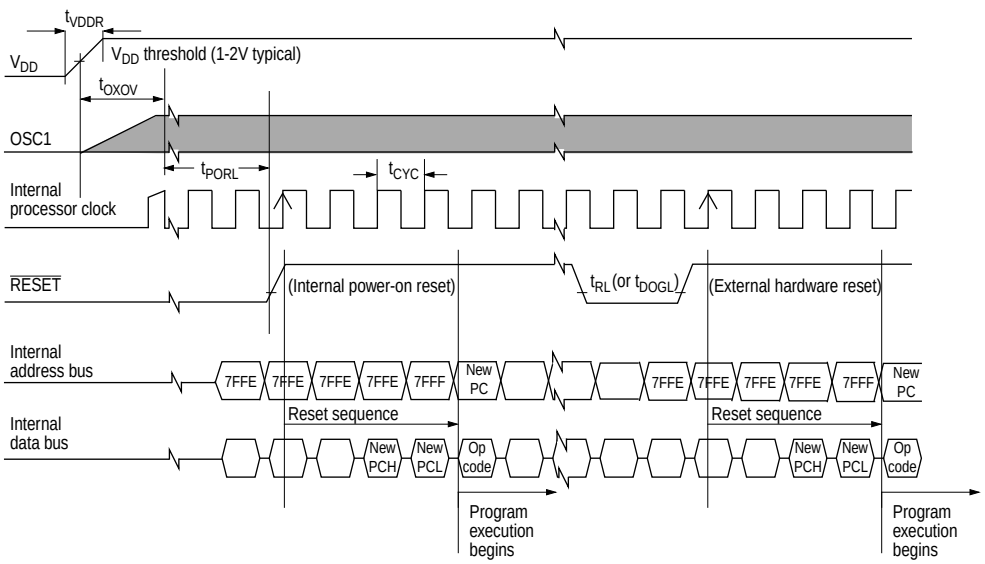


Figure 4-1 Power-on Reset and $\overline{\text{RESET}}$

4.1.1 Power-on Reset

A power-on reset occurs when a positive transition is detected on VDD. The power-on reset function is strictly for power turn-on conditions and should not be used to detect drops in the power supply voltage. The power-on circuitry provides for a 4064 t_{CYC} delay from the time that the oscillator becomes active. If the external $\overline{\text{RESET}}$ pin is low at the end of the 4064 cycle time-out, the processor remains in the reset state until $\overline{\text{RESET}}$ goes high. The user must ensure that the voltage on VDD has risen to a point where the MCU can operate properly by the time the 4064 cycles have elapsed. If there is doubt, the external $\overline{\text{RESET}}$ pin should remain low until the voltage on VDD has reached the specified minimum operating voltage.

4.1.2 $\overline{\text{RESET}}$ Pin

When the oscillator is running in a stable state, the MCU is reset when a logic zero is applied to the $\overline{\text{RESET}}$ input for a minimum period of one and one-half machine cycles (t_{CYC}). This pin contains an internal Schmitt Trigger as part of its input to improve noise immunity.

4.2 Interrupts

The MCU can be interrupted by seven different sources: six maskable hardware interrupts and one non-maskable software interrupt:

- External signal on the INTX pin,
- Real Time Interrupt,
- Port C Wake-up,
- Timer A,
- Timer B,
- Serial Interface,
- Software Interrupt instruction (SWI).

Interrupts cause the processor to save the register contents on the stack and to set the interrupt mask (I bit) to prevent additional interrupts. The RTI instruction (ReTurn from Interrupt) causes the register contents to be recovered from the stack and normal processing to resume.

Unlike reset, hardware interrupts do not cause the current instruction execution to be halted, but are considered pending until the current instruction is complete.

Note: The current instruction is the one already fetched and being operated on.

When the current instruction is complete, the processor checks all pending hardware interrupts. If interrupts are not masked (CCR I bit clear), the processor proceeds with interrupt processing; otherwise, the next instruction is fetched and executed.

Internal circuitry generates a processor interrupt signal IRQB if an interrupt occurs via either the external interrupt pin INTX, the wake-up function on Port C, or a real time interrupt (see Figure 4-2). The interrupting source can be determined by testing the state of the INTFX, WAKF and RTIF flag bits.

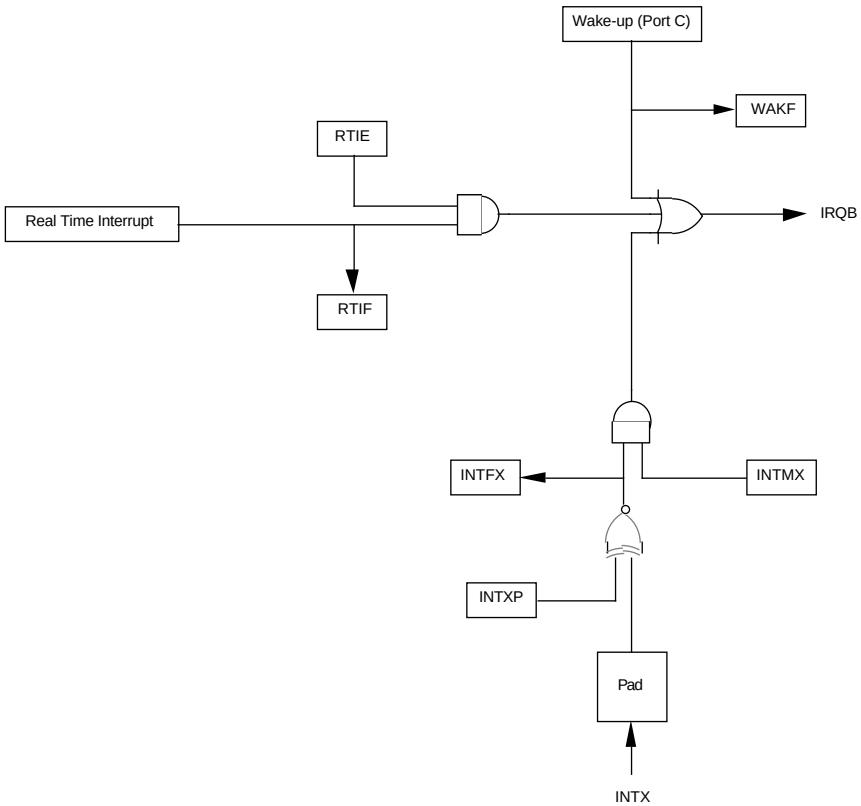


Figure 4-2 Internal Processor Interrupt Signal IRQB

Note: Acknowledging an interrupt signal too early could have an effect on the BIH and BIL command functions.

Table 4-1 shows the relative priority of all the possible interrupt sources. Internally, the External Interrupt, Real Time Interrupt, and Wake-up signals are “ORed” to generate the CPU interrupt signal IRQB. The source responsible for generating the IRQB interrupt can be determined by examining the interrupt flag bits associated with these sources.

Note: The external interrupt INTX can be disabled by setting the External Interrupt Mask bit (INTMX) in the Interrupt Control Register ($000E).

Table 4-1 Interrupt Priorities

Source	Vector address	Priority
Real time interrupt	$FFFA, $FFFB	highest
External interrupt (INTX)		
Wake-up (Port C)		
Timer A	$FFF8, $FFF9	
Timer B	$FFF6, $FFF7	
Serial interface	$FFF4, $FFF5	lowest

For example, if both an external interrupt and a Timer A interrupt are pending at the end of an instruction execution, the external interrupt is serviced first.

The software interrupt SWI is executed in the same way as any other instruction, regardless of the state of the I-bit.

4.2.1 Hardware Controlled Interrupt Sequence

The following three functions (RESET, STOP, and WAIT) are not in the strictest sense interrupts. However, they are acted upon in a similar manner. Flowcharts for hardware interrupts are shown in Figure 4-3, and for STOP and WAIT in Figure 4-4.

RESET: A reset condition causes the program to vector to its starting address which is specified by the contents of memory locations $FFFE (MSB) and $FFFF (LSB). The I bit in the condition code register is also set.

STOP: The STOP instruction causes the oscillator to be turned off and the processor to “sleep” until an external interrupt (INTX) (if enabled), a wake-up interrupt (if enabled) or reset occurs.

WAIT: The WAIT instruction causes all processor clocks to stop, but leaves the timer clock running. This “rest” state of the processor can be cleared by reset, an external interrupt (INTX) (if enabled), a wake-up interrupt or a Timer/SI interrupt. There are no special wait vectors for these individual interrupts.

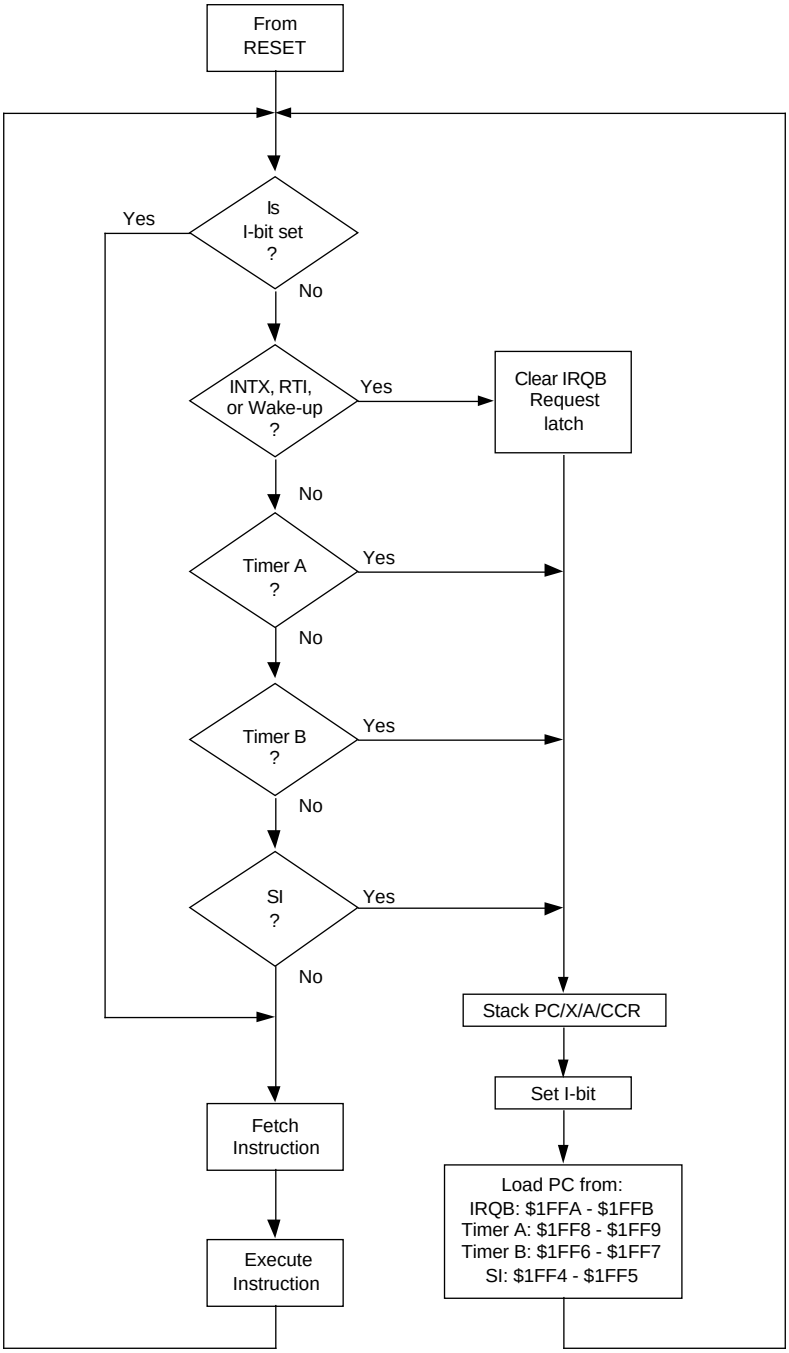


Figure 4-3 Hardware Interrupt Flow Chart

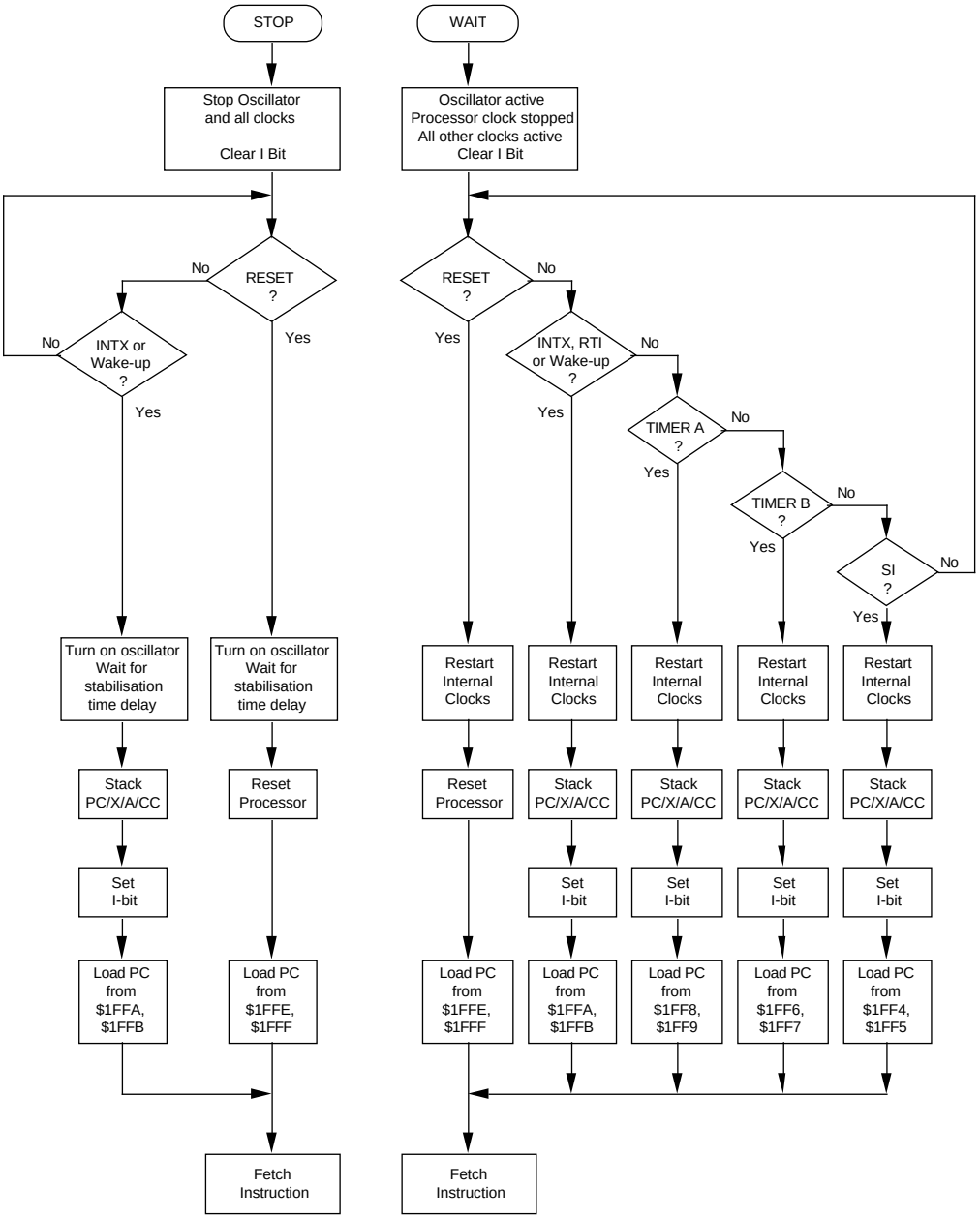


Figure 4-4 STOP/WAIT Flow Chart

4.2.2 Non-Maskable Software interrupt (SWI)

The software interrupt SWI is an executable instruction and a non-maskable interrupt: it is executed regardless of the state of the I bit in the CCR. If the I bit is zero (interrupts enabled), SWI is executed after interrupts which were pending when the SWI was fetched, but before interrupts generated after the SWI was fetched. The SWI interrupt service routine address is specified by the contents of memory locations $FFFC and $FFFD.

4.2.3 Maskable Hardware Interrupts

If the interrupt mask bit (I bit) of the CCR is set, all maskable interrupts (internal and external) are disabled. Clearing the I bit enables interrupts.

Note: The internal interrupt latch is cleared in the first part of the interrupt service routine; therefore, one external interrupt pulse could be latched and serviced as soon as the I bit is cleared.

4.2.3.1 External Interrupt (INTX)

The interrupt request is latched immediately following the selected edge on the INTX pin. It is then synchronized internally and serviced by the routine that has its start address contained in memory locations $FFFA and $FFFB. The External Interrupt Mask bit (INTMX) in the Interrupt Control Register ($000E) allows this interrupt to be masked from the processor.

4.2.3.2 Real Time Interrupt

The interrupt request is latched when the Real Time Interrupt Timer times out. It is then synchronized internally and serviced by the routine that has its start address contained in memory locations $FFFA and $FFFB. The Real Time Interrupt Enable bit (RTIE) in the RTI Control Register ($0018) allows this interrupt to be masked from the processor.

4.2.3.3 Port C Wake-up

The interrupt request is latched when a defined Wake-up signal appears on one of the pins of Port C. It is then synchronized internally and serviced by the routine that has its start address contained in memory locations $FFFA and $FFFB. The Wake-up Enable bits (WEn) in the Wake-up Enable Register ($0013) allow these interrupts to be masked from the processor.

4.2.3.4 Timer A Interrupt

The interrupt request is latched when Timer A times out. It is then synchronized internally and serviced by the routine that has its address contained in memory locations $FFF8 and $FFF9. The Timer A Interrupt Mask bit (INTMA) in the Interrupt Control Register ($000E) allows this interrupt to be masked from the processor.

4.2.3.5 Timer B Interrupt

The interrupt request is latched when Timer B times out. It is then synchronized internally and serviced by the routine that has its start address contained in memory locations $FFF6 and $FFF7. The Timer B Interrupt Mask bit (INTMB) in the Interrupt Control Register ($000E) allows this interrupt to be masked from the processor.

4.2.3.6 SI Interrupt

The interrupt request is latched immediately after each data transfer. It is then synchronized internally and serviced by the routine that has its start address contained in memory locations $FFF4 and $FFF5. The SI Enable bit (SIE) in the Port E/SI Mode Register ($0010) allows this interrupt to be enabled or disabled (see Section 8).

4.2.4 Interrupt Control

The Interrupt Control Register ($000E) allows masking and provides acknowledgement of the interrupt signals:

	Address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	State on reset
Interrupt control register	$000E	WAKF	INTXP	INTFB	INTFA	INTFX	INTMB	INTMA	INTMX	0000 0001

INTMX, INTMA, INTMB

These three interrupt mask bits permit INTX (external interrupt), Timer A and Timer B interrupt signals to be masked from the processor:

1 (set) – Interrupts not masked

0 (clear) – Interrupts masked

After a hardware reset, only external interrupts are permitted.

INTFX, INTFA, INTFB

These three bits (Interrupt Flag and Acknowledge) indicate the occurrence of interrupt signals from INTX (external interrupt), Timer A and Timer B. Timers A and B generate interrupt signals when their final values are reached:

1 (set) – Interrupt has occurred

0 (clear) – Interrupt has not occurred

Writing a “0” to one of these bits acknowledges the interrupt and resets the interrupt signal.

INTXP

The INTXP bit (Program External Interrupt) selects which signal edge or level on the INTX input generates an external interrupt:

1 (set) – Interrupt on rising edge or logic high level

0 (clear) – Interrupt on falling edge or logic low level

This bit only influences the effect of a signal coming from an external source (INTX) but not the function of the IRQB signal to the microprocessor.

WAKF

The WAKF bit (Wake-up Flag) indicates that an interrupt was generated by the Wake-up function on Port C:

1 (set) – Interrupt caused by wake-up function

0 (clear) – Interrupt not caused by wake-up function

Writing a “0” to one of these bits acknowledges the interrupt and resets the interrupt signal.

Additional control of external interrupts is achieved via the PITX bit in the Timer Control Register ($000C):

	Address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	State on reset
Timer control register	$000C	SC1	SC0	PITX	PITW	1	XHOM	CEIB	CEIA	uuuu 11uu

PITX

The PITX bit (Program Interrupt Type eXternal) provides additional control of external interrupts by selecting either edge or level sensitive triggering.

1 (set) – Level-triggered interrupt selected

0 (clear) – Edge-triggered only interrupt selected

Note: PITW (Program Interrupt Type Wake-up function), bit 4 of the Timer Control Register ($000C) provides additional control of wake-up interrupts on Port C by selecting either edge or level sensitive triggering.

4.3 Low Power Modes

4.3.1 STOP

The STOP instruction places the MCU in its lowest power consumption mode. The processor can only be started again by an external interrupt (on the INTX pin) (if enabled), a wake-up interrupt (if enabled) or reset. The oscillator is stopped, all CPU and timer functions are stopped, and an oscillator stabilisation delay of 4064 cycles is required to start the processor again.

Note: The RTI Timer restarts immediately after exiting the STOP state.

During the STOP mode, the RTI interrupt flag and interrupt enable bits are cleared by internal hardware to remove any pending interrupt requests. The RTI Timer prescaler is also cleared. The I bit in the CCR is cleared to enable external interrupts or wake-up function interrupts. All other bits and registers, and memory remain unaltered. All input/output lines remain unchanged.

Note: Pending interrupts from Timer A, Timer B, and SI are not cleared by the stop instruction. The external interrupt INTX can be disabled by the External Interrupt Mask bit (INTMX) in the Interrupt Control Register ($000E).

4.3.2 WAIT

The WAIT instruction places the MCU in a low-power consumption mode, but the WAIT mode consumes more power than the STOP mode. All CPU action is suspended, but the timers remain active. An interrupt from the timer can cause the MCU to exit the WAIT mode. During the WAIT mode, the I bit in the CCR is cleared to enable interrupts. All other registers, memory and input/output lines remain in their previous state.

5
MEMORY AND ADDRESSING

5.1 Memory Map

As shown in Figure 5-1, the MC68HC05E0 is capable of addressing a full 65536 bytes of memory and I/O using a non-multiplexed bus. The address space is divided into internal memory space and external memory space as shown in the memory map. The internal memory space is located within the first 512 bytes of memory (pages 0 and 1) and contains the I/O port data and data registers, all the timer, Serial Interface and wake-up control and data registers, and 480 bytes of RAM. Program writes to on-chip locations are repeated on the external bus enabling off-chip memory to duplicate the contents of on-chip memory. Program reads from on-chip locations also appear on the external bus, but the CPU accepts data only from the addressed on-chip locations, and ignores data appearing on the input bus.

5.2 RAM

480 bytes of on-chip static RAM are located from $0020 to $01FF. The processor stack starts at $00FF and is limited to 64 bytes ($00C0 to $00FF). When the stack overflows it wraps round from $00C0 to $00FF, overwriting any existing data.

Note: Using the stack area for data storage or as temporary work locations requires care to prevent it from being overwritten due to stacking from an interrupt or subroutine call.

External RAM can be located from $0200 to $1FFF and accessed via the external data and address buses. PD2 (Port D, bit 2) can be configured to output a chip select signal ($\overline{CS2}$) which can be used to select the external RAM, when an address in this range is present on the address bus.

5.3 ROM

External ROM can be located from $0200 to $FFFF and accessed via the external data and address buses. (Address lines A0 to A12 are always available; PD5 to PD7 (Port D bits 5 to 7) can be configured as address lines A13 to A15 to provide addressing above $8000.) Chip select signal $\overline{\text{CSROM}}$ can be used to select the external ROM, when an address in the range $3000-$FFFF is present on the address bus.

Note: Writing a logic zero to bit 2 (XROM) of the Timer Control Register ($000C) causes the $\overline{\text{CSROM}}$ output to remain permanently low throughout the full memory map ($0000-$FFFF). Clearing XROM also sets the external data bus lines to INPUT only (See Section 2.9).

5

5.4 Registers

Internal registers associated with the on-board hardware functions are located from $0000 to $001F. All internal registers and their contents are shown in Table 5-1.

External I/O (on peripheral devices) can be located from $2000 to $2FFF and accessed via the external data and address buses. PD3 (Port D, bit 3) can be configured to output a chip select signal ($\overline{\text{CS3}}$) which can be used to select the external I/O, when an address in this range is present on the address bus.

5.5 Vectors

All vectors for reset, hardware interrupts and software interrupt are located at the top of the memory map, from $FFF4 to $FFFF, as shown in Figure 3-1. Each vector location consists of two bytes containing the start address (in ROM) of the reset or interrupt routine (see Table 5-2).

5.6 Address Decoding and System Expansion

The address decoder partitions the entire memory space into RAM, ROM, register and interrupt vector areas. This allows access to external blocks of memory and peripherals as well as to the on-chip RAM and all the registers supporting the on-chip hardware functions. Address bus and data bus lines (A0 to A12 and D0 to D7, respectively) and chip select signal $\overline{\text{CSROM}}$ are always available. In addition, Port D can be configured (via the Port D Mode register) to provide the following address, control and chip select signals:-

Table 5-1 Vector Addresses for Interrupts and Reset

Register	Flag name	Interrupt source	CPU interrupt	Vector address
N/A	N/A	Reset	RESETS	$FFFE – $FFFF
CCR	I-bit	Software	SWI	$FFFC – $FFFD
Interrupt control	INTFX	External (INTX)	IRQB	$FFFA – $FFFB
RTI control	RTIF	Real time interrupt		
Interrupt control	WAKF	Wakje-up		
Interrupt control	INTFA	Timer A	INT2	$FFF8 – $FFF9
Interrupt control	INTFB	Timer B	INT1	$FFF6 – $FFF7
SI S	IRQ/ACK	SPI/I^{2}C	INT0	$FFF4 – $FFF5

address lines: A13 – PD5
A14 – PD6
A15 – PD7

control signals: P02 – PD0
R/$\overline{W}$ – PD1
$\overline{LIR}$ – PD4

chip select signals: $\overline{CS2}$ – PD2
$\overline{CS3}$ – PD3

The MC68HC05E0 directly provides gated chip-select and read/write signals. A complete system comprising MPU, program EPROM and RAM (optional) can thus be built without any additional circuitry. An example of this basic system is shown in Figure 5-2. The R/$\overline{W}$ signal and the chip-select signal for additional RAM (and additional I/O) is optionally available using Port D pins. $\overline{CSROM}$ is intended to select the program ROM/EPROM and is valid in the address range $3000 to $FFFF. $\overline{CS2}$ and $\overline{CS3}$ select address ranges $0020 to $1FFF and $2000 to $2FFF respectively.

Larger, more versatile systems with more external memory and/or peripherals can be designed with the inclusion of additional chip select logic. An example of an expanded system using an MC74HC138 is shown in Figure 5-3. The gating of the chip selects with P02 (using the CS1 pin on the MC74HC138) means that the output enable pins of the EPROM(s) and RAM(s) can be grounded. If the memory's chip select is not qualified by the clock, then its output enable pin should be driven by $\overline{P02}$ in order to avoid bus contention during the low period of P02. This type of expanded system would be useful, for example, in a development environment where several EPROM and/or RAM chips are required. Care should be taken to ensure that the loading specification of the buses is not exceeded.

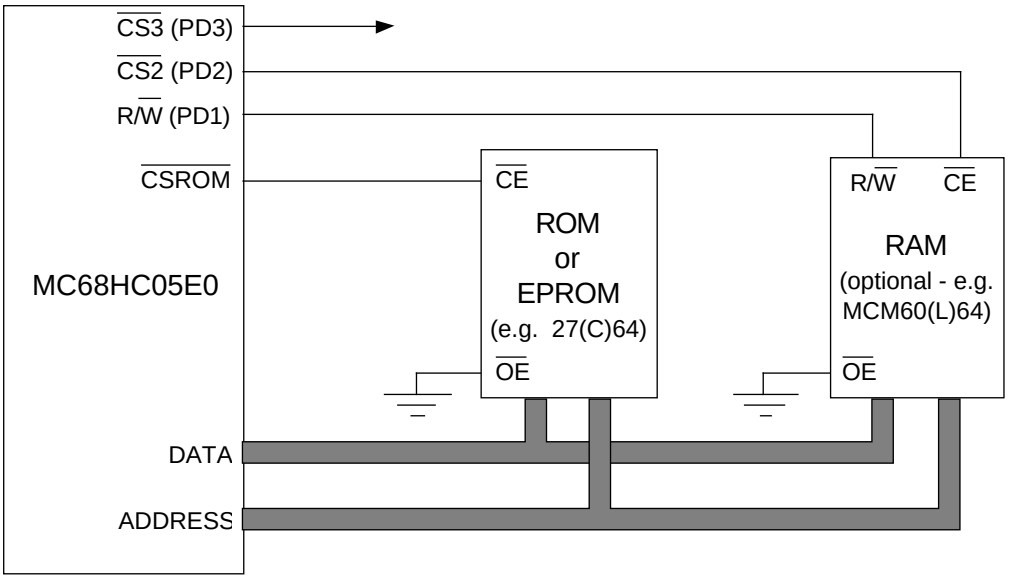


Figure 5-1 Minimum System with External Memory

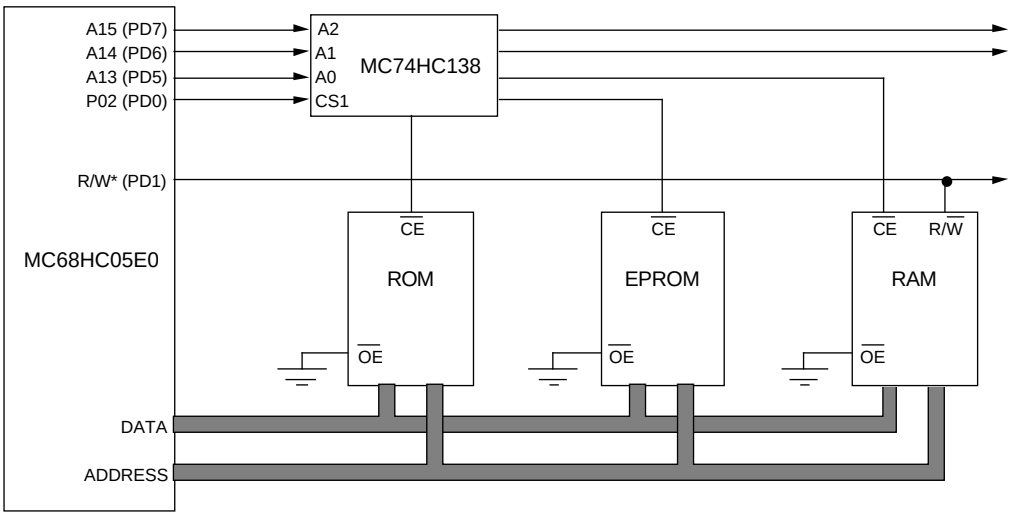


Figure 5-2 More Complex Expanded System

Table 5-2 Register outline

Register name	Address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	State on Reset
Port A data register	$0000	AD7	AD6	AD5	AD4	AD3	AD2	AD1	AD0	0000 0000
Port B data register	$0001	BD7	BD6	BD5	BD4	BD3	BD2	BD1	BD0	0000 0000
Port C data register	$0002	CD7	CD6	CD5	CD4	CD3	CD2	CD1	CD0	0000 0000
Port D data register	$0003	DD7	DD6	DD5	DD4	DD3	DD2	DD1	DD0	0000 0000
Port E data register	$0004	1	1	1	1	ED3	ED2	ED1	ED0	1111 0000
Port A data direction register	$0005	AR7	AR6	AR5	AR4	AR3	AR2	AR1	AR0	0000 0000
Port B data direction register	$0006	BR7	BR6	BR5	BR4	BR3	BR2	BR1	BR0	0000 0000
Port C data direction register	$0007	CR7	CR6	CR5	CR4	CR3	CR2	CR1	CR0	0000 0000
Port D data direction register	$0008	DR7	DR6	DR5	DR4	DR3	DR2	DR1	DR0	0000 0000
Port E data direction register	$0009	1	1	1	1	ER3	ER2	ER1	ER0	1111 0000
Timer A prescaler register	$000A	TA7	TA6	TA5	TA4	TA3	TA2	TA1	TA0	0000 0000
Timer B scaler register	$000B	TB7	TB6	TB5	TB4	TB3	TB2	TB1	TB0	0000 0000
Timer control register	$000C	SC1	SC0	PITX	PITW	1	XROM	CETB	CETA	0000 1100
	$000D									
Interrupt control register	$000E	WAKF	INTXP	INTFB	INTFA	INTFX	INTMB	INTMA	INTMX	0000 0001
SI data register	$000F	SD7	SD6	SD5	SD4	SD3	SD2	SD1	SD0	0000 0000
PortE/SI mode register	$0010	SIE	CPHA	CPOL	WL	BD1	BD0	PS1	PS0	0000 0000
SI S register	$0011	BB	ACK	TFF/ DOIT	IRQ/ ACK	NMA	SP	ST	R/WB	0000 0000
PortD mode register	$0012	EA15	EA14	EA13	DM4	DM3	DM2	DM1	DM0	0000 0001
Wake-up enable register	$0013	WE7	WE6	WE5	WE74	WE3	WE2	WE1	WE0	0000 0000
Wake-up phase register	$0014	WP7	WP6	WP5	WP74	WP3	WP2	WP1	WP0	0000 0000
	$0015									
	$0016									
	$0017									
RTI control register	$0018		RTIF		RTIE			RT1	RT0	0000 0011
RTI counter register	$0019	TC7	TC6	TC5	TC4	TC3	TC2	TC1	TC0	0000 0000

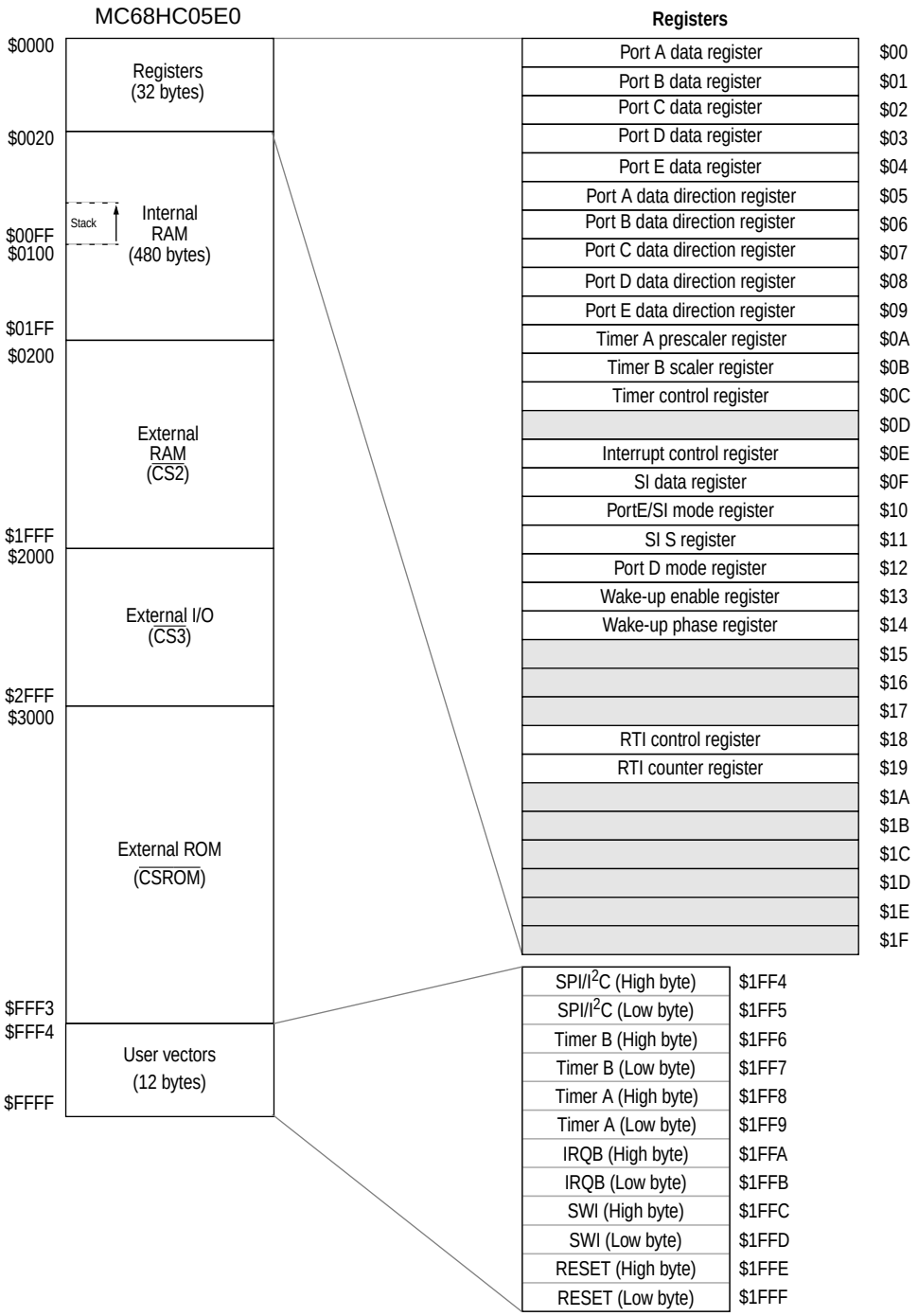


Figure 5-3 Memory map of the MC68HC05E0

6
PARALLEL INPUT/OUTPUT PORTS

6.1 Bidirectional Ports

Ports A, B, C, and D are bidirectional 8-bit ports and Port E is a 4-bit bidirectional port and may be configured as inputs or outputs under software control. The Port E lines are associated with bits 0–3 in the Port E data and data direction registers. The direction of any port line is determined by the state of the corresponding bit in the data direction register (DDR). Any port line is configured as an output if its corresponding DDR bit set to a logic one. A pin is configured as an input if its DDR bit is cleared to a logic zero. At power-on or reset, all DDRs are cleared to $00 (with the exception of DDRE which is set to $F0), which configures all I/O lines as inputs. (Note that, on reset, bit 0 of the Port D Mode register is set to one, thereby forcing Port D bit 0 to output the P02 clock signal.) All data and data direction registers can be written to and read by the CPU. Refer to Figure 6-1.

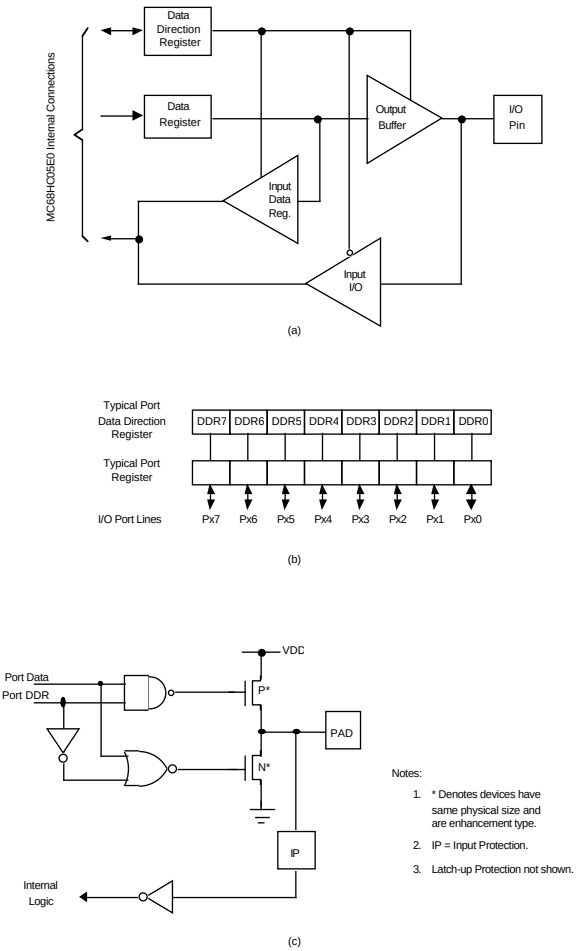


Figure 6-1 Bidirectional I/O Port Structure

6.2 Port C Wake-up Function

Each bit on Port C can be individually configured to operate in a wake-up mode which can interrupt the processor when an appropriate signal edge or level is applied to the input pin (see Figure 6-2). This is controlled by the corresponding Wake-up Enable bits (WE0 to WE7) in the Wake-up Enable Register ($0013) and Wake-up Phase bits (WP0 to WP7) in the wake-up Phase Register ($0014).

	Address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	State on reset
Wake-up enable register	$0013	WE7	WE6	WE5	WE4	WE3	WE2	WE1	WE0	0000 0000

WEx bit

1 (set) – Wake-up Interrupt enabled

0 (clear) – Wake-up Interrupt disabled

	Address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	State on reset
Wake-up phase register	$0014	WP7	WP6	WP5	WP4	WP3	WP2	WP1	WP0	0000 0000

WPx bit:

1 (set) – Wake-up on rising edge or logic high level

0 (clear) – Wake-up on falling edge or logic low level

PITW (program interrupt type wake-up function), bit 4 of the Timer Control Register ($000C), provides additional control of wake-up interrupts on Port C by selecting either edge or level sensitive triggering.

	Address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	State on reset
Timer control register	$000C	SC1	SC0	PITX	PITW	1	XROM	CETB	CETA	0000 1100

PITW

1 (set) – Level-triggered interrupt selected

0 (clear) – Edge-triggered only interrupt selected

If port C is used as an output port, the wake-up function is disabled. Therefore, a wake-up function can only be used in conjunction with Port C being an input (see also Timer Control Register).

The wake-up function uses the INTX vector ($FFFA, $FFFB). In the INTX interrupt routine, the user should test the WAKF flag (bit 7 in the Interrupt Control Register ($000E) to see if a wake-up interrupt is pending. The wake-up interrupt routine should also clear the WAKF flag before returning to the main program flow.

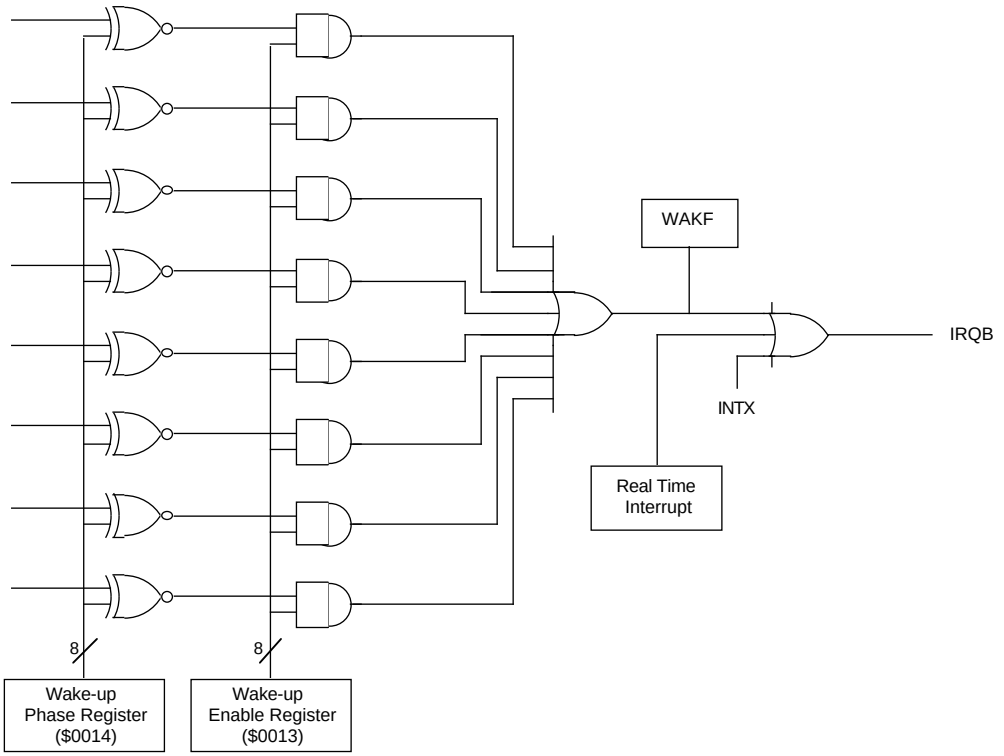


Figure 6-2 Port C Wake-up Function

6.3 Port D Alternate Functions

All Port D lines can be reconfigured to provide the signals needed to interface with external memory (see Table 6-1).

R/$\overline{W}$ is the Read/Write signal which is active low only during the high phase of P02 and stays high during the low phase of P02.

The chip select signals $\overline{CS2}$ and $\overline{CS3}$ are active low and are only active during the high phase of P02.

$\overline{LIR}$ is an output signal which goes low only during the first P02 clock cycle of each instruction, and remains low for the duration of that cycle. It is intended for use during debugging and emulation.

Table 6-1 Port D Alternate Functions

As bidirectional I/O	Alternate function
PD0	P02 — Bus frequency clock signal
PD1	R/$\overline{W}$ — Read/write signal (readwhen high, write when low)
PD2	$\overline{CS2}$ — Chip select signal (active low) for external RAM
PD3	$\overline{CS3}$ — Chip select signal (active low) for external I/O
PD4	$\overline{LIR}$ — Load instruction register (active low)
PD5	A13 — Address line
PD6	A14 — Address line
PD7	A15 — Address line

Pins PD5, PD6 and PD7 have internal pull-up resistors connected to VDD.

Configuration of Port D is done via the Port D Mode Register ($0012).

	Address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	State on reset
Port D mode register	$0012	EA15	EA14	EA13	DM4	DM3	DM2	DM1	DM0	0000 0001

If a bit is set in the mode register, the corresponding port bit is used only as an output irrespective of the value in the data direction register (see Table 6-2).

Figure 6-3 shows the internal structure of Port D.

Table 6-2 Port D Mode Table

Bit	Value	Function
DM0	0 1	port function (bidirectional) P02 signal as output signal
DM1	0 1	port function (bidirectional) RI$\overline{W}$ signal as output signal
DM2	0 1	port function (bidirectional) $\overline{CS2}$ (external RAM) signal as output signal
DM3	0 1	port function (bidirectional) $\overline{CS3}$ (external I/O) signal as output signal
DM4	0 1	port function (bidirectional) $\overline{LIR}$ signal as output signal
EA15	0 1	port function (bidirectional) A15 signal as output signal
EA14	0 1	port function (bidirectional) A14 signal as output signal
EA13	0 1	port function (bidirectional) A 13 signal as output signal

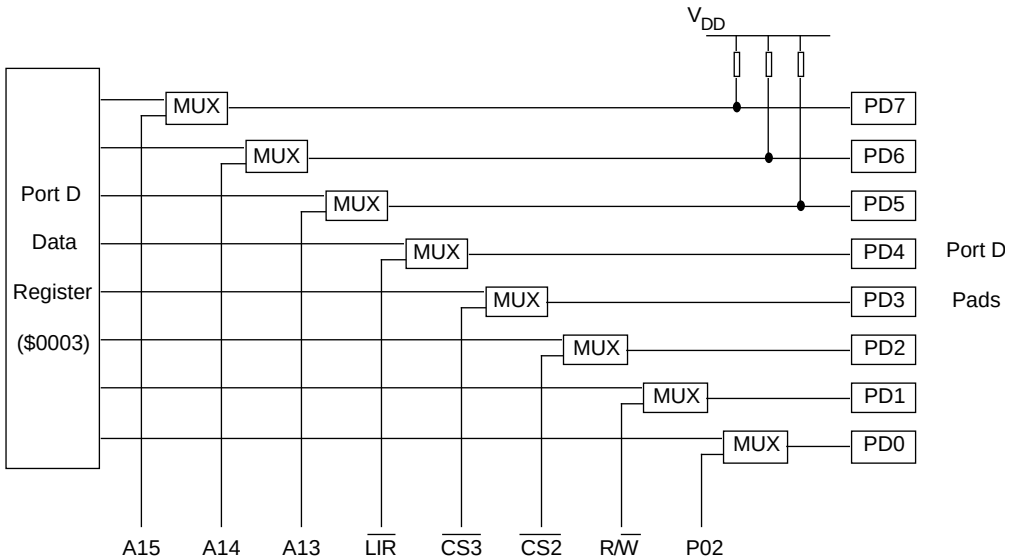


Figure 6-3 Port D Structure

6.4 Serial Interface Support Functions on Port E

Port E also supports the on-board Serial Interface (SPI and I^2C functions) (see Figure 6-4). Selection of these functions is controlled via the PS0 and PS1 bits in the Port E/SI Mode Register ($0010).

Table 6-3 Port E/SI Mode Selection

PS1	PS0	Function	PE0	PE1	PE2
0	0	PortE as parallel port	I/O	I/O	I/O
0	1	I^2C bus protocol	data	clock	I/O
1	0	SPI function	data	clock	I/O
1	1	SPI function	data	I/O	clock

See Section 8 for detailed information on the Serial Interface.

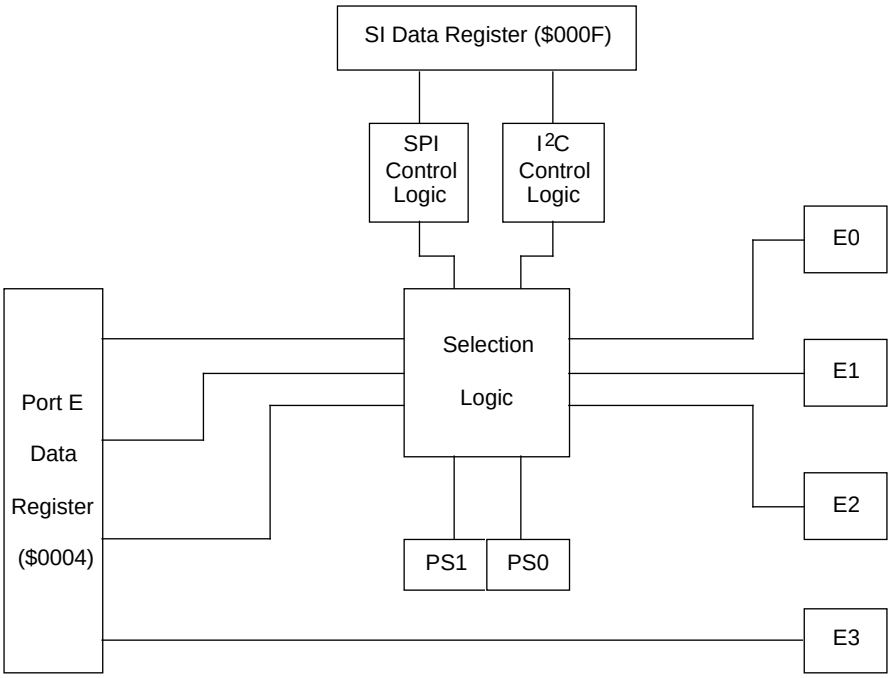


Figure 6-4 Port E Structure

6.5 Other Port Considerations

All ports are latched with the bus timing signal P02 (especially for inputs).

All output ports can emulate open-drain outputs. This is achieved by writing a zero to the relevant output port latch. By toggling the corresponding data direction bit, the port pin will either be an output zero or tri-state (an input). Refer to Figure 6-5.

When using a port pin as an open-drain output, certain precautions must be taken in the user software. If a read-modify-write instruction is used on a port where the open-drain is assigned and the pin at this time is programmed as an input, it will read it as a “one”. The read-modify-write instruction will then write this “one” into the output data latch on the next cycle. This would cause the open-drain pin not to output a “zero” when desired.

Note: “Open-drain” outputs should not be pulled above V_{DD}.

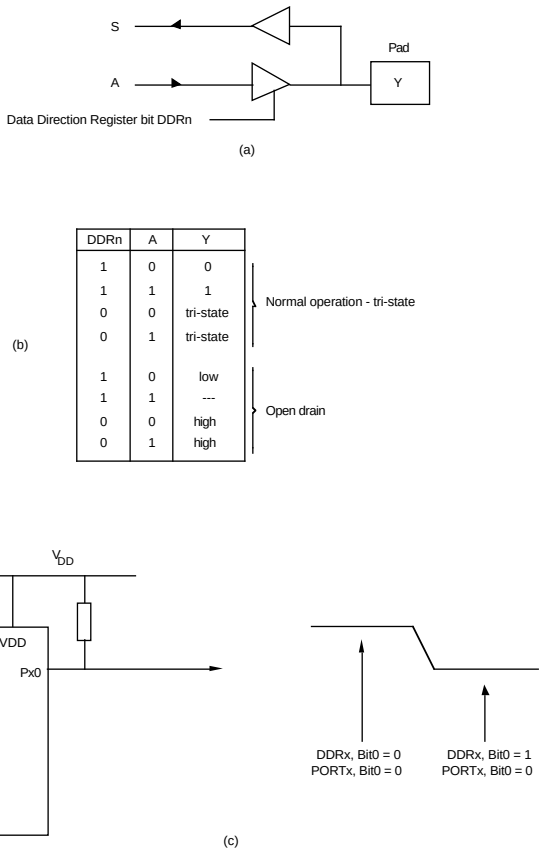


Figure 6-5 Port Logic Levels

7
TIMERS

The MC68HC05E0 contains three timers: Timer A, Timer B and the Real Time Interrupt Timer. Timers A and B are driven by the sub-system clock, which also drives the Serial Interface sub-system. The Real Time Interrupt timer is driven from the bus clock (P02).

Timer A and Timer B can generate an independent interrupt signal to the processor. Timer A has a higher interrupt priority than timer B. The Real Time Interrupt shares its interrupt vector with the External Interrupt (INTX) and the Port C Wake-up function.

7.1 Sub-system Clock Control

The sub-system clock frequency f_{sub} is generated from the bus clock frequency (P02) by a selectable prescaler (see Figure 7-1).

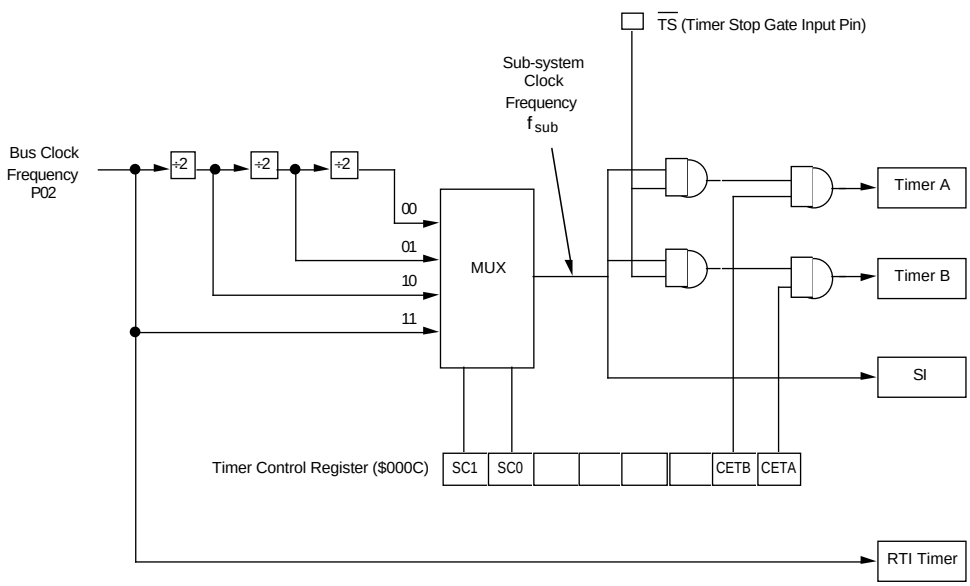


Figure 7-1 Sub-system Clock Generation

The maximum bus clock frequency (P02) is 4 MHz. One of four different sub-system clock frequencies can be selected via bits 6 and 7 (SC0 and SC1) of the Timer Control Register ($000C) (see Table 7-1).

Table 7-1 Sub-system Clock Frequency Selection

SC1	SC0	Sub-system clock frequency (f_{SUB})
0	0	P02÷8
0	1	P02÷4
1	0	P02÷2
1	1	P02÷1

A Timer Stop signal on the active low $\overline{TS}$ pin allows Timer A and Timer B to be halted. $\overline{TS}$ can be used as a gate input to the two timers, thereby allowing pulse width measurement to be carried out on the input signal. Halting the timers is also useful during emulation of the device and during software debug. The SPI/I^2C functions are not interrupted by the $\overline{TS}$ signal.

7.2 Timer A

Timer A consists of an 8-bit prescaler driving a 6-bit free-running counter (see Figure 7-2).

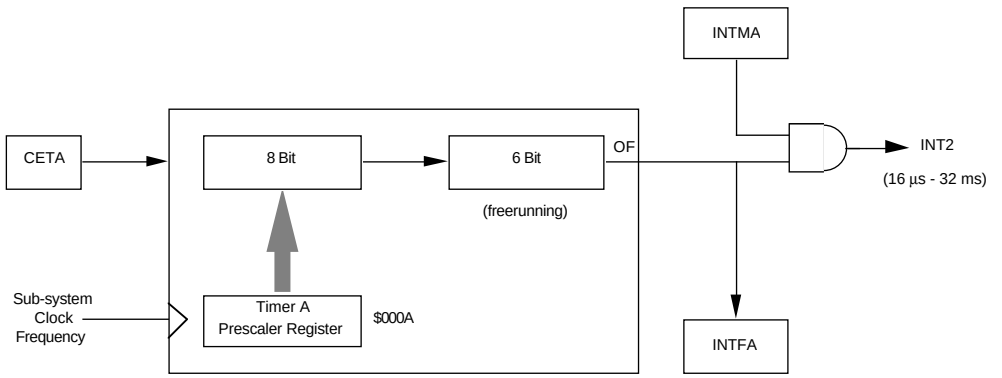


Figure 7-2 Timer A Structure

The Timer A Prescaler register ($000A) is an 8-bit wide data register which is used for setting the prescaler. It is a read/write register and can be loaded with any value between $00 and $FF. Writing N to this register causes the prescaler to divide by N+1. Writing to this register resets the entire timer.

With the bus clock running at 4 MHz, Timer A has a range of 16 µs to 32.8 ms.

7.3 Timer B

Timer B consists of a 14-bit free-running counter driving an 8-bit scaler (see Figure 7-3).

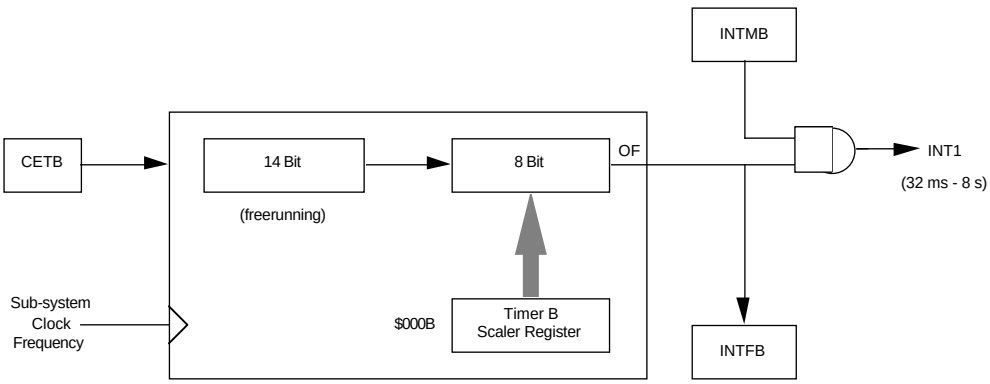


Figure 7-3 Timer B Structure

The Timer B Scaler Register ($000B) is an 8-bit wide data register which is used for setting the scaler. It is a read/write register and can be loaded with any value between $00 and $FF. Writing N to this register causes the scaler to divide by N+1. Writing to this register resets the entire timer.

With the bus clock running at 4 MHz, this timer has a range of 32.8 ms to 8.4 s.

7.4 Control Registers for Timer A and Timer B

Timer A and Timer B are enabled/disabled via the CETA (Count Enable Timer A) and CETB (Count Enable Timer B) control bits in the Timer Control Register ($000C).

	Address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	State on reset
Timer control register	$000C	SC1	SC0	PITX	PITW	1	XROM	CETB	CETA	0000 1100

CETA, CETB

1 (set) – Counter enabled

0 (clear) – Counter disabled

Both timers generate a signal on overflow which can be used to interrupt the processor. These signals can be masked by the INTMA and INTMB bits in the Interrupt Control Register ($000E). Timer Interrupt Flag and Acknowledge bits INTFA and INTFB indicate when a timer interrupt has occurred and can be written to reset the timer interrupt logic.

	Address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	State on reset
Interrupt control register	$000C	WAKF	INTXP	INTFB	INTFA	INTFX	INTMB	INTMA	INTMX	0000 0001

INTMA, INTMB

1 (set) – Interrupts not masked

0 (clear) – Interrupts masked

INTFA, INTF

1 (set) – Interrupt has occurred

0 (clear) – Interrupt has not occurred

Writing a “0” to INTFA or INTFB acknowledges the interrupt and resets the interrupt signal for that timer.

7.5 Real Time Interrupt Timer

The Real Time Interrupt circuit consists of a three stage divider and a 1 of 4 selector. The clock signal which drives the RTI circuit is P02/2^{14} (= P02/16384) with three additional divider stages giving a maximum interrupt period of 32.8 milliseconds at a bus frequency of 4 MHz (See Figure 4-1).

The 8-bit RTI Timer Counter Register ($0019) is a read only register which contains the current value of the 8-bit ripple counter at the beginning of the timer chain. Reset clears the counter. During WAIT mode the CPU clock halts but the timer remains active. If the RTIE bit is set, a timer interrupt will cause the processor to exit the WAIT mode. The timer is cleared when entering STOP mode.

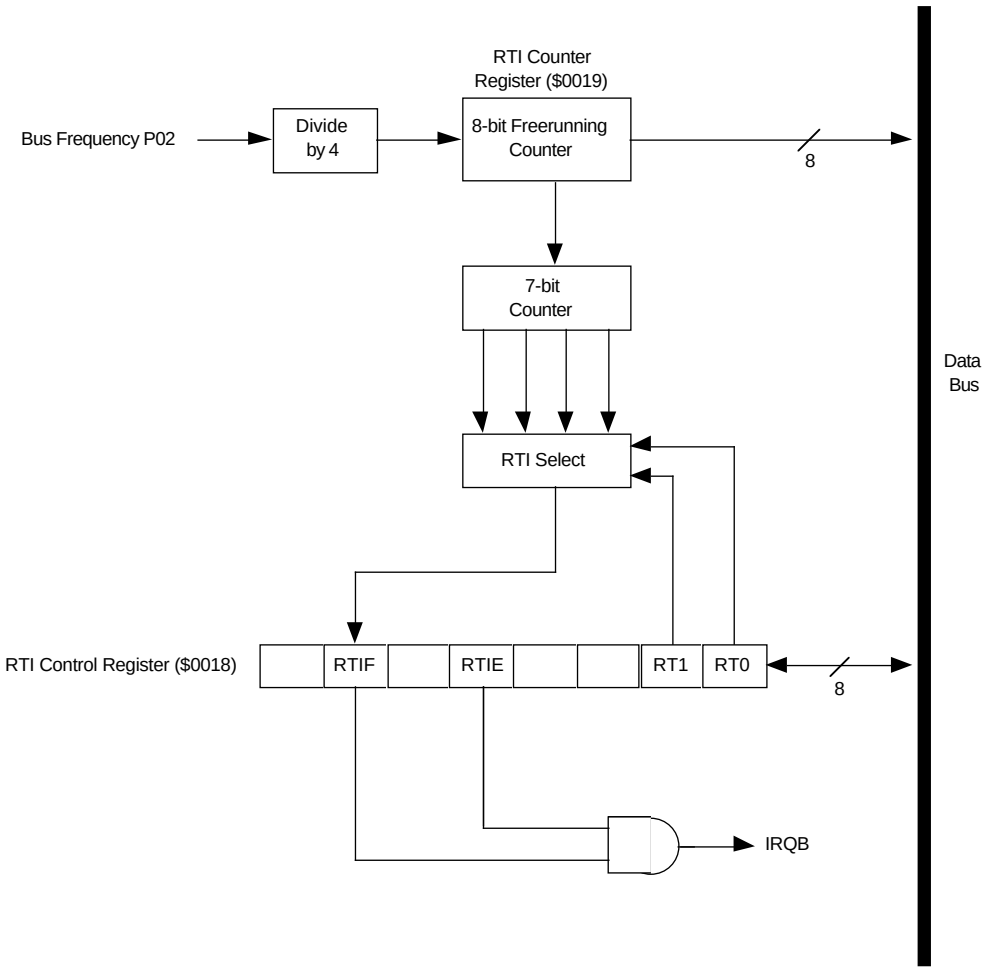


Figure 7-4 Real Time Interrupt Timer Structure

The RTI system is controlled via the RTI Control Register ($0018).

	Address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	State on reset
RTI control register	$0018		RTIF		RTIE			RT1	RT0	0000 0011

RTIF is a clearable, read-only status bit and is set when the output of the chosen selector stage (1 of 4) goes active. A CPU interrupt request will be generated (IRQB) if the Real Time Interrupt is enabled. Clearing the RTIF bit is done by writing a “zero” to it. Writing a “one” to RTIF has no effect. Reset clears RTIF.

RTIF

1 (set) – RTI function has timed out and generated interrupt signal.

0 (clear) – RTI function has not timed out.

RTIE is a read/write control bit which allows the Real Time Interrupt to generate a CPU interrupt request ($\overline{IRQ}$). Reset clears RTIE.

RTIE

1 (set) – RTI request enabled

0 (clear) – RTI request disabled

RT1, RT0 select one of four taps from the Real Time Interrupt Select logic. On reset, these bits are set to “one” which selects the lowest periodic interrupt rate, and gives the maximum time in which to modify the bits to select a different time-out period. Care should be taken when altering RT1 and RT0 if a timeout is imminent or uncertain; if a tap is selected during a cycle in which the counter is switching, an interrupt request could be missed, or an additional one generated.

Table 7-2 Real Time Interrupt Rates (bus frequency = 4MHz)

RT1	RT0	RTI time-out period (ms)
0	0	4.1
0	1	8.2
1	0	16.4
1	1	32.8

8
SERIAL INTERFACE

8.1 General

The serial interface (SI) on the MC68HC05E0 supports synchronous data transfer over one bidirectional data line and a clock line. The interface can be configured to operate as a serial peripheral interface (SPI) or as an I^2C-bus interface. The microprocessor always has control of the clock signal, and is therefore always “bus master”. The clock and data lines are multiplexed through port E. Port E and the serial interface lines are controlled by two serial port select bits (PS1, PS0) in the Port E/SI Mode Register ($0010.

	Address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	State on reset
Port E/SI mode register	$0010	SIE	CPHA	CPOL	WL	BD‘1	BD0	PS1	PS0	0000 0000

Table 8-1 PS0, PS1: Serial Port Function Selection Bits

PS1	PS0	Function	PE0	PE1	PE2
0	0	Port Eas parallel port	I/O	I/O	I/O
0	1	I^2C bus protocol	data	clock	I/O
1	0	SPI function	data	clock	I/O
1	1	SPI function	data	I/O	clock

Note that bit 3 of port E is always available as a port bit.

When PS0 and PS1 are set to select a serial function, the data register and direction register bits for the pins used as data and clock do not influence the state of these pins.

The sequence of events for starting a send/receive sequence in either SPI or I^{2}C-bus mode is as follows:

1) Write to SI Mode register ($0010). The state of bits 0 and 1 select the data and clock lines, and the protocol (see Table 8-1).
2) If sending data from the MCU to a peripheral, write the data to be sent to the SI Data register ($000F).
3) Start the send/receive sequence by writing a “1” to the DOIT bit (bit 5 in the Serial Interface Control/Status Register, the SI S Register, $0011). The SI S Register also contains control bits to determine the data direction, and whether STOP/START bits are to be added to the data (I^{2}C-bus only).

The transmission rate in SPI and I^{2}C-bus mode is determined by BD0 and BD1 (bits 2 and 3 in the SI Mode Register) which control the division ratio applied to the subsystem clock (see Figure 8-1). The division ratios are listed in Table 8-2. Note that the maximum SI transfer rate = P02 ÷ 6 (i.e. when SC0, SC1, BD0 and BD1 are all set to 1).

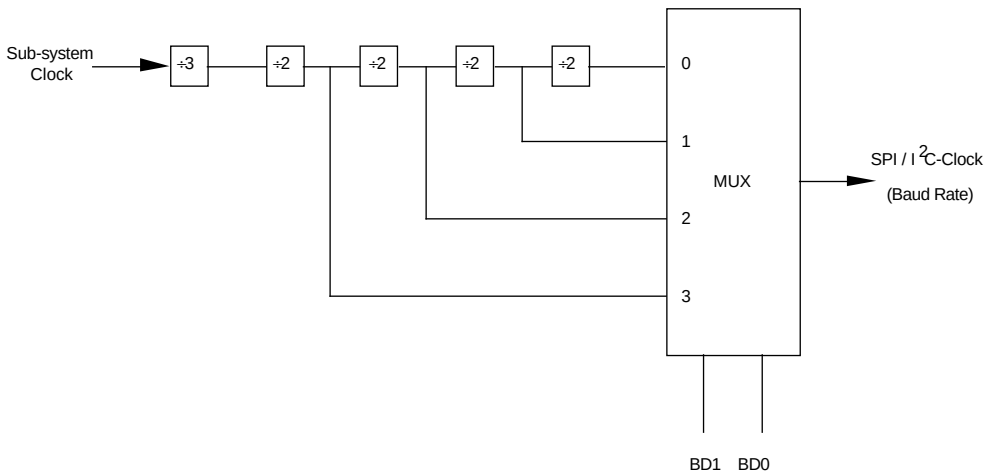


Figure 8-1 Serial Port Baud Rate Generation

Table 8-2 BD0, BD1: Serial Port Transfer Rate Selection

BD1	BD0	SI transfer rate
0	0	$f_{SUB} \div 48$
0	1	$f_{SUB} \div 24$
1	0	$f_{SUB} \div 12$
1	1	$f_{SUB} \div 6$

On completion of the transmit/receive sequence internal logic sets the IRQACK flag (bit 4 in the SI S Register) high, and an interrupt will be generated if the SIE bit (bit 7 in the SI Mode Register) was set high. In both SPI and I^2C-bus mode, when the serial interface is in send mode, the data on the external data line is sampled 3 times per data bit. The majority value of the sample (the value found by at least 2 out of 3 of the samples) is compared with the expected state of data. If they are not the same, the TFF (Transmission failure) flag (bit 5 of the SI S Register) is set. This indicates that, due perhaps to noise or a hard fault such as a short circuit on the data line, the transmitted data and the actual data on the line do not match. TFF is a read-only signal. TFF is set at the end of an SPI/I^2C-bus sequence.

8.2 SPI Configuration

The SPI serial interface allows data to be sent or received by the MCU over a single, bidirectional data line. An accompanying clock signal is also generated by the MCU.

Four different combinations of clock phase/polarity can be generated under the control of CPOL and CPHA (bits 5 and 6 respectively in the SI Mode Register, $0010). (Refer to Figure 8-2)

The clock and data pins selected by PS0 and PS1 are open-drain. External 4k7 pull-up resistors should be used. The external data line is hi-Z (pulled high by the external resistor) when the SPI is idle. The idle state of the clock line depends on the setting of CPOL/CPHA. Refer to Section 9 for more detailed information on SPI timing.

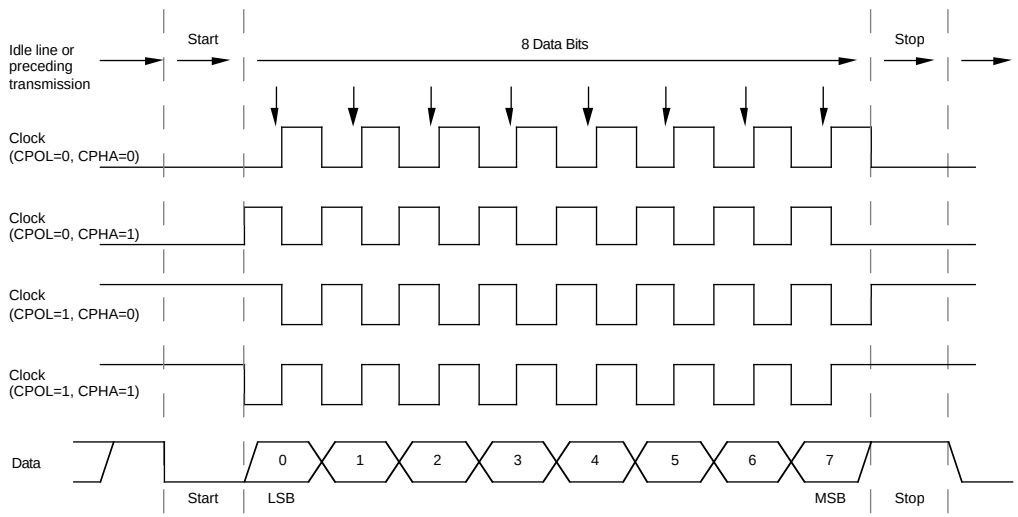


Figure 8-2 SPI Data/Clock Relationship

A START bit is generated on the data line before a data byte is transmitted. No START bit is generated for receive sequences. The START bit is a low state applied to the data line while the clock stays in its idle state. The START bit lasts for 1 SPI clock period (determined by BD0 and BD1 in the SI Mode Register).

The SPI bus allows either 6-bit or 8-bit words to be sent or received. WL (bit 4 in SI Mode Register) determines the word length (Low = 8-bit, High = 6-bit). When a 6-bit word is to be sent, it should be written to the 6 most significant bits of the SI Data Register. After transmission, the location of the bits in the SI Data Register will have changed. The 6-bit word now resides in bits 0 to 5, and the upper 2 bits are the remaining (unused) bits. This is because the SI Data Register functions as a shift register during transmission, and the most significant bit is shifted round to the least significant bit every time a new data bit is sent. Similarly, when receiving a 6-bit word, it will be located in the lower 6 bits of SI Data Register at the end of the receive sequence.

The direction of data along the data line is determined by R/WB (bit 0 in the SI S Register). R/WB = 0 sends data to a peripheral, R/WB = 1 receives data from a peripheral.

The Serial Peripheral Interface is configured via the Port E/SI Mode Register ($0010).

	Address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	State on reset
Port E/SI mode register	$0010	SIE	CPHA	CPOL	WL	BD‘1	BD0	PS1	PS0	0000 0000

SIE —Serial Interrupt Enable Bit

This bit masks the interrupt signal generated at the end of each SPI transfer.

1 (set) – Interrupts not masked

0 (clear) – Interrupts masked

CPOL, CPHA — Clock Polarity and Clock Phase Bits

These two bits provide four possible clock polarity/phase relationships for recognition of valid data, as shown in Figure 8-2. The MC68HC05E0 SPI is always the master.

The interrupt signal is generated after the last clock pulse. The START bit/STOP bit is automatically generated (only when sending). A receive sequence only clocks in the data bits sent from the slave.

WL — Word Length Bit.

This bit determines whether 6-bit or 8-bit data words are to be sent/received.

1 (set) – 6-bit word length

0 (clear) – 8-bit word length

Note: In the I^2C-bus mode only 8-bit words can be transferred.

BD1, BD0 — Baud Rate Select Bits (see Table 8-2).

PS1, PS0 — Port Select Bits

These bits should be 10 or 11 (binary) to select an SPI function (see Table 8-1).

The Serial Peripheral Interface is controlled via the SI S Register ($0011).

	Address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	State on reset
Serial interface S register	$0011	BB	ACK	TFF/ DOIT	IRQ/ ACK	NMA	SP	ST	R/WB	0000 0000

TFF/DOIT — Transmission Failure Flag/“Do it” Bit

Writing this bit high starts an SPI bus transaction. Reading this bit at the end of an SPI transmit sequence shows the state of the TFF flag. If high, this indicates a mismatch between data in the SI Data Register and the data sent on the serial data line. Writing a zero to this bit clears and resets the Transmission Failure Flag circuitry.

		Read	Write
1 (set)	–	Transmission error	Start Send or Receive
0 (clear)	–	No transmission error	Clear and reset TFF

IRQACK — Interrupt Acknowledge Flag Bit

This bit is the serial interface interrupt flag, raised at the end of every SPI/I^2C-bus transaction. Writing a zero to IRQACK clears the interrupt flag, resets the interrupt circuitry and permits new data to be written to the data register or newly received data to overwrite the old data in the reception register (meaning the data register has already been read).

1 (set) – SPI transaction completed

0 (clear) – No SPI transaction has occurred

R/WB — Read/Write Bit

This bit selects the direction of the data transfer during SPI or I^2C-bus operation.

1 (set) – Read (from peripheral)

0 (clear) – Write (to peripheral)

8.3 I²C-Bus Configuration

The I²C-bus protocol was specified by Philips. It consists of a bidirectional data line (SDA) and a bidirectional clock line (SCL). Each driver connected to the SDA and SCL lines should be open-drain. An off-chip pull-up resistor (4k7) pulls either line to a high state if the line is not being held low by an active peripheral. SDA and SCL pulled high is the I²C idle state.

The original I²C-bus specification allows for different devices connected to an SDA and SCL line to be receivers, transmitters, masters (generating the SCL clock) or slaves. In the MC68HC05E0 implementation, the micro is always master.

A variety of devices exist which can communicate over the I²C-bus, from complex microcontrollers (e.g. Philips 68070) to real-time clock chips, LCD drivers and “dumb” serial EEPROMs. The I²C-bus implementation on the MC68HC05E0 is intended for use in simpler bus systems, rather than complex multi-master systems. For this reason the following features of the full I²C-bus protocol are NOT supported:

- The passing of bus mastery from one peripheral to another. (The MC68HC05E0 is always bus master).
- Clock synchronization when two bus masters simultaneously drive SCL.
- Detection of data collision (2 devices sending data simultaneously).
- Address register and comparator to allow selection of which peripheral is to be addressed. (No I²C-bus address register is available on the MC68HC05E0)

The I²C-bus specification calls for a pause of at least 4.7 µs between the end of one transmission sequence and the start of another. This is not done in hardware by the MC68HC05E0 but, as at least three instructions are needed to load/read, set-up and start the I²C-bus circuit, the pause will always be greater than 5 µs when using a 4 MHz crystal. If an 8 MHz crystal is used, software must provide a delay greater than 4.7 µs. The user has control over whether START and STOP bits are generated. Also, the I²C-bus can be programmed such that the micro does or does not provide an acknowledge signal to peripherals.

The I²C-bus circuitry monitors the bus continuously to check that no other I²C-bus peripheral is currently using the bus.

Figure 8-3 shows the timing relationship between the clock (SCL) and data (SDA) in I²C-bus mode. Refer to Section 9 for more detailed timing information.

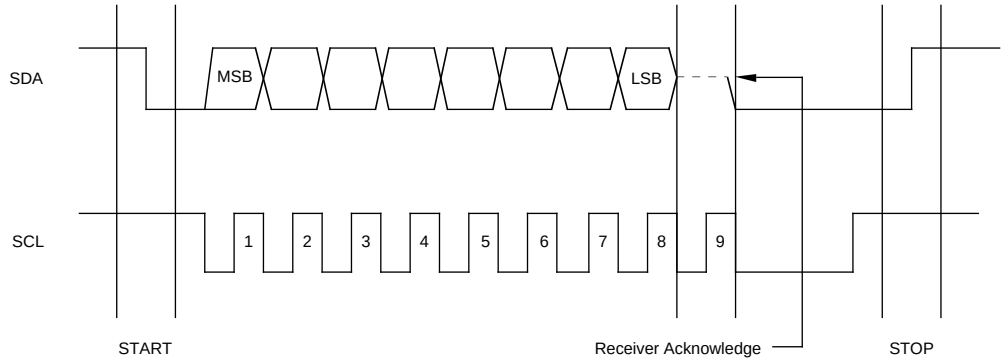


Figure 8-3 I2C-bus Data/Clock Relationship

A typical I2C-bus transaction starts with the MC68HC05E0 generating a START condition (SDA pulled low while SCL high). The MC68HC05E0 will then send a stream of 8 bits. The SCL line is pulsed to provide a clock waveform while data is sent. The first 7 bits of the byte specify the address of a slave peripheral, and the 8th bit specifies the direction of data between the slave and the micro - a “0” indicates write to slave, a “1” indicates read from slave. A 9th clock pulse is generated during which the addressed slave can indicate acknowledgement to the micro. This is done by the micro releasing the data line and the slave pulling the data line low. Successive bursts of 9 clock pulses are generated by the micro to synchronize transfer of data, and the receiver has the possibility to acknowledge on the 9th pulse. Data transfer is terminated by the micro generating a STOP condition (SDA goes high while SCL high).

The I2C-bus Interface is configured via the SI Mode Register ($0010)

	Address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	State on reset
Port E/SI mode register	$0010	SIE	CPHA	CPOL	WL	BD‘1	BD0	PS1	PS0	0000 0000

SIE — Serial Interrupt Enable Bit

This bit masks the interrupt signal generated at the end of each I2C transfer.

1 (set) – Interrupts not masked

0 (clear) – Interrupts masked

BD1, BD0 — Baud Rate Select Bits (see Table 8-2).

PS1, PS0 — Port Select Bits

These bits should be 01 (binary) to select an I^2C-bus function (see Table 8-1).

The I^2C-bus Interface is controlled via the SI S Register ($0011).

	Address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	State on reset
Serial interface S register	$0011	BB	ACK	TFF/ DOIT	IRQ/ ACK	NMA	SP	ST	R/WB	0000 0000

BB — Bus Busy Bit

BB is a read-only bit which indicates if the I^2C-bus is busy or idle. BB is set if a START condition is seen on the data line. This implies that another I^2C-bus master is using the bus, and that if the MC68HC05E0 tries to start an I^2C-bus sequence a data/clock collision will occur. BB is cleared when the corresponding STOP condition, implying that the bus is again idle, is detected on the bus. This flag generates no interrupt.

1 (set) – Busy

0 (clear) – Idle

ACK — Acknowledge Bit

ACK is a read-only bit which goes high if an acknowledge signal is given by a peripheral (via the 9th clock bit) after the MC68HC05E0 has transmitted a byte of data to it. ACK is updated at the end of each receive sequence.

1 (set) – Acknowledgement received from peripheral

0 (clear) – Acknowledgement not received from peripheral

TFF/DOIT — Transmission Failure Flag/“Do it” Bit

Writing this bit high starts an I^2C-bus transaction. Reading this bit at the end of an I^2C-bus transmit sequence shows the state of the TFF flag. If high, this indicates a mismatch between data in the SI Data Register and the data sent on the serial data line. Writing a zero to this bit clears and resets the Transmission Failure Flag circuitry.

	Read	Write
1 (set) –	Transmission error	Start Send or Receive
0 (clear) –	No transmission error	Clear and reset TFF

IRQACK — Interrupt Acknowledge Flag Bit

This bit is the serial interface interrupt flag, raised at the end of every SPI/I^2C-bus transaction. Writing a zero to IRQACK clears the interrupt flag, resets the interrupt circuitry and permits new data to be written to the data register or newly received data to overwrite the old data in the reception register (meaning the data register has already been read).

1 (set) – I^2C-bus transaction completed

0 (clear) – No I^2C-bus transaction has occurred

NMA — No Master Acknowledge Bit

When set high NMA prevents the MC68HC05E0 from giving an acknowledge signal to a peripheral after receiving a byte of data. An acknowledge is made if NMA is low.

1 (set) – Prevent acknowledgement after data receive

0 (clear) – Permit acknowledgement after data receive

SP — STOP Bit

If high, an I^2C-bus STOP condition is generated on the SDA line at the end of the next bus transfer. If low, no STOP bit is produced.

1 (set) – Generate STOP condition after data transfer

0 (clear) – Do not generate STOP condition after data transfer

ST — START Bit

If high, an I^2C-bus START condition is generated on the SDA line at the beginning of the next bus transfer. If low, no START bit is produced.

1 (set) – Generate START condition before data transfer

0 (clear) – Do not generate START condition before data transfer

R/WB — Read/Write Bit

This bit selects the direction of the data transfer during SPI or I^2C-bus operation.

1 (set) – Read (from peripheral)

0 (clear) – Write (to peripheral)

8.4 Transmission Error Detection

During a transmission sequence (i.e. only when sending data), the serial interface circuitry checks for transmission errors caused by external influences (e.g. picture tube arcing). This transmission error detection circuitry samples the state of the SDA output pin during transmission, using a multiple of the transmission frequency, and compares the sampled value with an expected value in an additional shadow register. The sampled values are filtered for every required value bit (i.e. a majority decision is taken). The result of the comparison is shown in the Transmission Failure Flag, TFF (bit 5 of the SI S Register, $0011) for each word transmitted. (See Figure 8-4.)

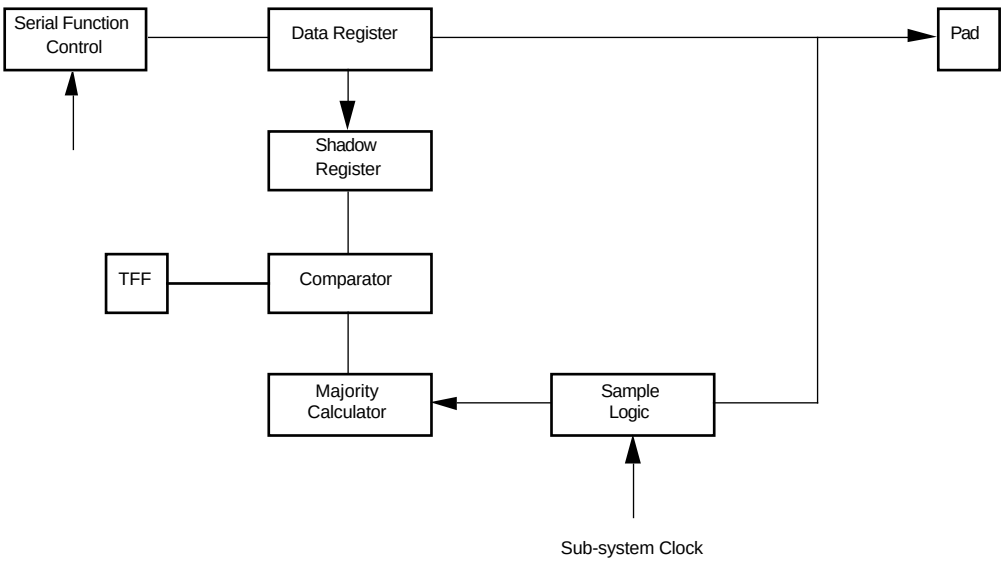


Figure 8-4 Serial Interface Transmission Error Detection

9
ELECTRICAL SPECIFICATIONS

9.1 Introduction

This section contains the electrical specifications and associated timing information for the MC68HC05E0.

9.2 Maximum Ratings

Table 9-1 Maximum ratings

Rating	Symbol	Value	Unit
Supply Voltage	V_{DD}	– 0.3 to 7.0	V
Input Voltage	V_{in}	V_{SS} – 0.3 to V_{DD} + 0.3	V
Operating Temperature Range	T_A	T_L to T_H 0 to 70	°C
Storage Temperature Range	T_{stg}	– 65 to 150	°C
Current Drain per Pin * Excluding V_{DD} and V_{SS}	I_D	25	mA

* One pin at a time, observing maximum power dissipation limits.

This device contains protective circuitry against damage due to high static voltages or electrical fields; however, it is advised that normal precautions be taken to avoid application of any voltages higher than maximum-rated voltages to this high-impedance circuit. Reliability of operation is enhanced if unused inputs are tied to an appropriate logic voltage level (eg. either GND or V_{DD}).

9.3 Thermal Characteristics and Power Considerations

The average chip junction temperature, T_J, in degrees Celsius can be obtained from the following equation:

$$T_J = T_A + (P_D \bullet \theta_{JA})$$

Where:

T_A	=	Ambient Temperature (^OC)
θ_{JA}	=	Package Thermal Resistance, Junction-to-Ambient ($^OC/W$)
P_D	=	$P_{INT} + P_{I/O}$
P_{INT}	=	$I_{DD} \times V_{DD}$ (W) = Internal Chip Power
$P_{I/O}$	=	Power Dissipation on Input and Output Pins (User Determined)

Note: For most applications $P_{I/O} < P_{INT}$ and can be neglected.

An approximate relationship between P_D and T_J (if $P_{I/O}$ is neglected) is:

$$P_D = K \div (T_J + 273^OC)$$

Solving the equations P_D and T_J for K gives:

$$K = P_D \bullet (T_A + 273^OC) + \theta_{JA} \bullet P_D^2$$

Where K is a constant pertaining to a particular part. K can be determined by measuring P_D (at equilibrium) for known T_A. Using this value of K the values of P_D and T_J can be obtained by solving the above equations for any value of T_A. The package thermal characteristics are shown in Table 9-2.

Table 9-2 Thermal Characteristics

Characteristics	Symbol	Value	Unit
Thermal Resistance Plastic 68-Pin Quad Pack (PLCC)	θ_{JA}	50	°C/W

Pins	R1	R2	C
A12 – A0, PD7 – PD5, $\overline{CSROM}$, R/$\overline{W}$, $\overline{CS2}$, $\overline{CS3}$	3.26 kΩ	2.38 kΩ	50 pF

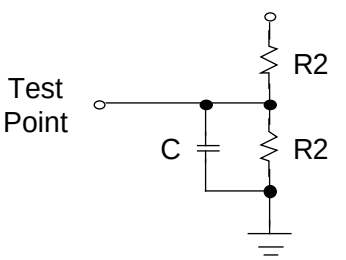


Figure 9-1 Equivalent Test Load

9.4 DC Electrical Characteristics

(V_{DD} = 5.0Vdc ± 10%, V_{SS} = 0 Vdc, T_A = 0 to 70°C)

Table 9-3 DC Electrical Characteristics (5V)

Characteristic	Symbol	Min	Typ	Max	Unit
OutputVoltage					
I_{Load} =10 µA	V_{OL}	–	–	0.1	V
I_{Load} = –10 µA	V_{OH}	V_{DD}–0.1	–	–	V
Output High Voltage (I_{Load} = 0.8 mA)	V_{OH}	V_{DD}–0.8	–	–	V
Output Low Voltage (I_{Load} = 1.6 mA)	V_{OL}	–	–	0.4	V
Input High Voltage	V_{IH}	0.7×V_{DD}	–	–	V
Input Low Voltage	V_{IL}	V_{SS}	–	0.2×V_{DD}	V
Supply Current (f_{op} = 4 MHz, f_{osc} = 8 MHz)	I_{DD}				
Run		–	15	–	mA
Wait		–	12	–	mA
Stop		–	2	–	µA
High–Z Leakage Current (All input pins except $\overline{RESET}$, PD5, PD6, PD7)	I_{IL}	–	–	±10	µA
Output High Current (V_{OH} = 2.4 V)	I_{OH}				
Port A		–	–	10	mA
Port B		–	–	5	mA
Output Low Current	I_{OL}				
Port A (V_{OL} = 0.5 V)		–	–	8	mA
Port A (V_{OL} = 1.0 V)		–	–	24	mA
Port B (V_{OL} = 0.5 V)		–	–	4	mA
Port B (V_{OL} = 1.0 V)		–	–	12	mA
Input Current	I_{in}				
$\overline{RESET}$, PD5, PD6, PD7		–70	–	–	µA
INTX, OSC1		–	–	±1	
Capacitance					
Ports (as input or output)	C_{out}	–	–	12	pF
$\overline{RESET}$, INTX	C_{in}	–	–	8	pF

Typical values at mid point of voltage range, 25°C only.

Wait I_{DD}: Only timer system active. If Serial Interface active add 10% to current drain.

Run I_{DD}, Wait I_{DD}: Measured using external square clock source (Fosc = 8 MHz), all inputs 0.2V from rail, no DC loads, maximum load on outputs 50pF (OSC2 load 20pF).

Wait, Stop I_{DD}: All ports configured as inputs, V_{il} = 0.2V and

V_{ih} = V_{DD} - 0.2V.

Stop I_{DD} measured with OSC1 = V_{SS}. Wait I_{DD} is affected linearly by the OSC2 capacitance.

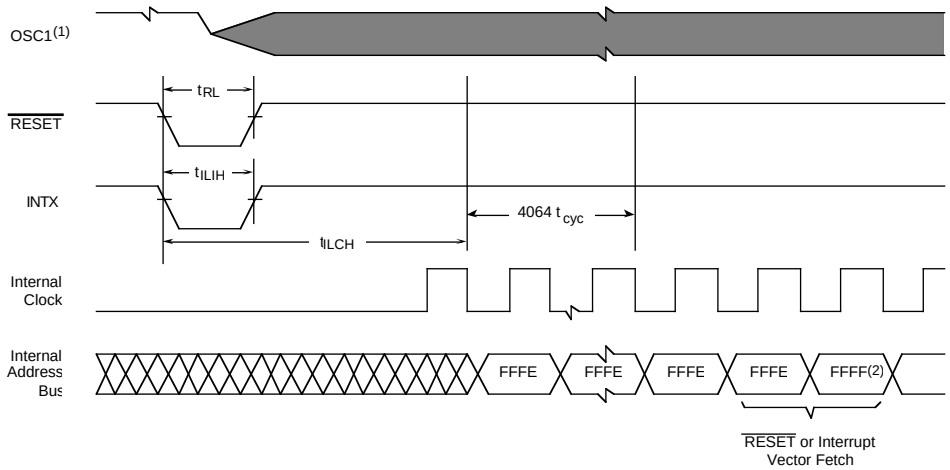
9.5 AC Electrical Characteristics

(V_{DD} = 5.0V dc ± 10%, V_{SS} = 0 Vdc, T_A= 0 to 70^{O}C)

Table 9-4 AC Electrical Characteristics (5V)

Characteristic	Symbol	Min	Max	Unit
Frequency of Operation				
Crystal	f_{osc}	–	8.0	MHz
External Clock	f_{osc}	dc	8.0	MHz
Internal Operating Frequency				
Crystal ($f_{osc} \div 2$)	P02	–	4.0	MHz
External Clock ($f_{osc} \div 2$)	P02	dc	4.0	MHz
Cycle Time	t_{cyc}	250	–	ns
Crystal Oscillator Startup Time	t_{OXOV}	–	100	ms
STOP Recovery Startup Time (Crystal Oscillator)	t_{ILCH}	–	100	ms
$\overline{\text{RESET}}$ Pulse Width	t_{RL}	1.5	–	t_{cyc}
Interrupt Pulse Width Low (Edge-Triggered)	t_{ILIH}	125	–	ns
Interrupt Pulse Period	t_{ILIL}	*	–	t_{cyc}
OSC1 Pulse Width	t_{OH}, t_{OL}	55	–	ns

* The minimum period t_{ILIL} should not lbe less than the number of cycle times it takes to execute the interrupt service routine plus 21 t_{cyc}.



NOTES:
1. Represents the internal gating of the OSC1 pin.
2. $\overline{\text{RESET}}$ vector address shown for timing example.

Figure 9-2 Stop Recovery Timing Diagram

Table 9-5 Expanded Bus Timing

Num	Characteristic	Symbol	Min	Typ	Max	Unit
1	Cycle Time	t_{CYC}	250	–	DC	ns
2	Clock Transition	t_R, t_F	–	10	25	ns
3	Read/Write Hold	t_{RWH}	-20	–	0	ns
4	Address Hold	t_{AH}	10	20	–	ns
5	Read/$\overline{\text{Write}}$ Delay	t_{RWD}	–	–	0	ns
6	Address Delay	t_{AD}	–	40	75	ns
7	Data Set-up (MPU Read)	t_{DSR}	40	–	–	ns
8	Data Hold (MPU Read)	t_{DHR}	-20	–	–	ns
9	Data Delay (MPU Write)	t_{DDW}	–	30	50	ns
10	Data Hold (MPU Write)	t_{DHW}	10	–	–	ns
11	Chip Select Hold	t_{CSH}	-20	–	0	ns
12	Chip Select Delay	t_{CSD}	–	–	0	ns

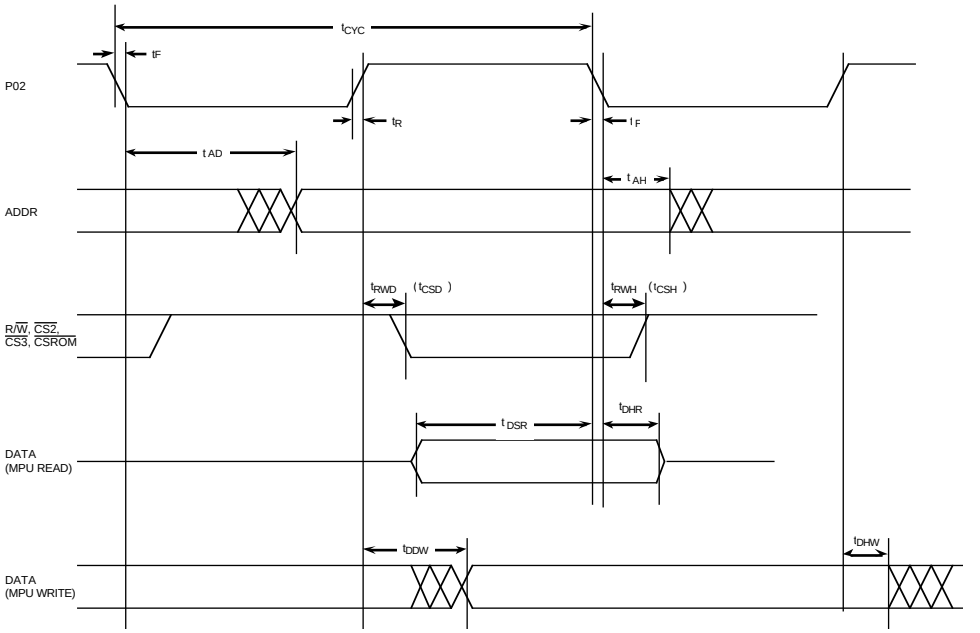
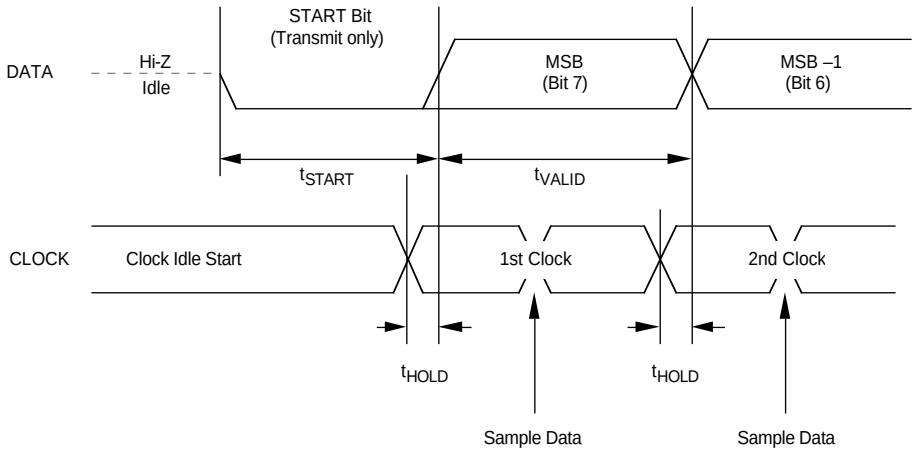


Figure 9-3 Expanded Bus Timing Diagram

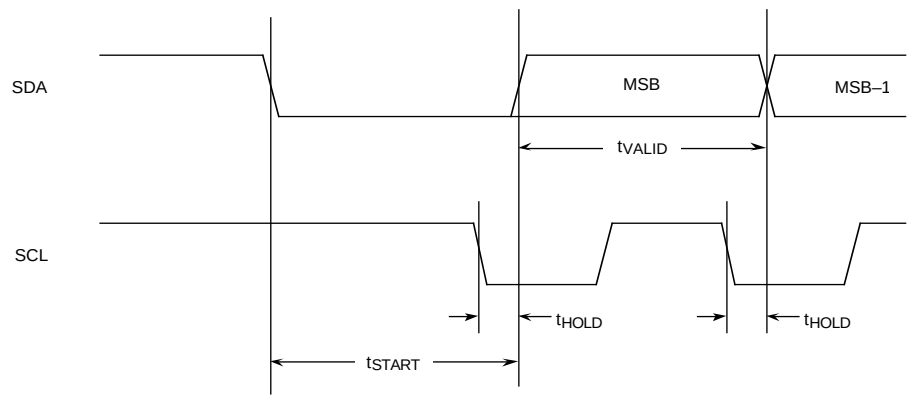
9.6 Serial Interface Timing



t_{HOLD} = 1 Bus Cycle Time + t_{pd} ($t_{pd} \approx 0 \rightarrow 30$ ns)

$t_{START} = t_{VALID}$ = 1/SPI Transfer Frequency (SPI Transfer Frequency determined by BD0/BD1)

Figure 9-4 SPI Timing Diagram



t_{HOLD} = 1 Bus Cycle Time + t_{pd} ($t_{pd} \approx 0 \rightarrow 30$ ns)

$t_{START} = t_{VALID}$ = 1 / I^2C Transfer Frequency (I^2C Transfer Frequency determined by BD0/BD1)

Figure 9-5 I^2C-bus Timing Diagram

THIS PAGE LEFT BLANK INTENTIONALLY

10
MECHANICAL DATA

10.1 68-pin PLCC Package

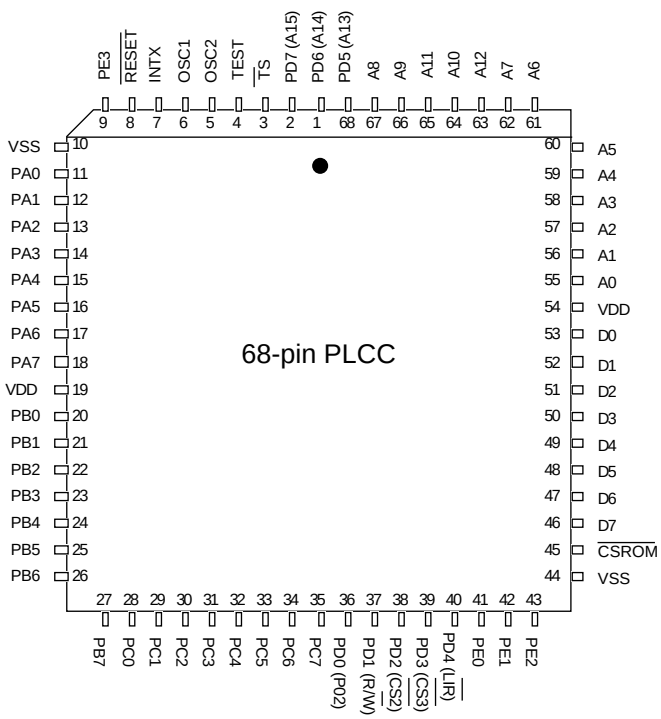


Figure 10-1 Pinout for 68-pin PLCC (Plastic Leadless Chip Carrier)

THIS PAGE LEFT BLANK INTENTIONALLY

11
ORDERING INFORMATION

11.1 EPROMs

An 8 kbyte EPROM programmed with the customer's software (positive logic for address and data) should be submitted for pattern generation. All unused bytes should be programmed to zeroes.

The EPROM should be clearly labelled, placed in a conductive IC carrier and securely packed.

11.2 Verification media

All original pattern media (EPROMs) are filed for contractual purposes and are not returned. A computer listing of the ROM code will be generated and returned along with a listing verification form. The listing should be thoroughly checked and the verification form completed, signed and returned to Motorola. The signed verification form constitutes the contractual agreement for creation of the custom mask. If desired, Motorola will program blank EPROMs (supplied by the customer) from the data file used to create the custom mask, to aid in the verification process.

11.3 ROM Verification Units (RVUs)

Ten MCUs containing the customer's ROM pattern will be sent for program verification. These units will have been made using the custom mask but are for the purpose of ROM verification only. For expediency, they are usually unmarked and are tested only at room temperature (25^{o}C) and at 5 volts. These RVUs are included in the mask charge and are not production parts. They are neither backed or guaranteed by Motorola Quality Assurance. At the customer's request, an RVU wafer can be supplied. This will be shipped against the customer’s initial production order.

THIS PAGE LEFT BLANK INTENTIONALLY

GLOSSARY

This section contains abbreviations and specialist words used in this data sheet and throughout the industry. Further information on many of the terms may be gleaned from Motorola's ***M68HC11 Reference Manual, M68HC11RM/AD***, or from a variety of standard electronics text books.

$xxxx	The digits following the '$' are in hexadecimal format.
%xxxx	The digits following the '%' are in binary format.
A/D, **ADC**	Analog-to-digital (converter).
Bootstrap mode	In this mode the device automatically loads its internal memory from an external source on reset and then allows this program to be executed.
Byte	Eight bits.
CAN	Controller area network.
CCR	Condition codes register; an integral part of the CPU.
CERQUAD	A ceramic package type, principally used for EPROM and high temperature devices.
Clear	'0' — the logic zero state; the opposite of 'set'.
CMOS	Complementary metal oxide semiconductor. A semiconductor technology chosen for its low power consumption and good noise immunity.
COP	Computer operating properly. *aka* 'watchdog'. This circuit is used to detect device runaway and provide a means for restoring correct operation.
CPU	Central processing unit.
D/A, DAC	Digital-to-analog (converter).
EEPROM	Electrically erasable programmable read only memory. *aka* 'EEROM'.
EPROM	Erasable programmable read only memory. This type of memory requires exposure to ultra-violet wavelengths in order to erase previous data. *aka* 'PROM'.
ESD	Electrostatic discharge.

Expanded mode	In this mode the internal address and data bus lines are connected to external pins. This enables the device to be used in much more complex systems, where there is a need for external memory for example.
EVS	Evaluation system. One of the range of platforms provided by Motorola for evaluation and emulation of their devices.
HCMOS	High-density complementary metal oxide semiconductor. A semiconductor technology chosen for its low power consumption and good noise immunity.
I/O	Input/output; used to describe a bidirectional pin or function.
Input capture	(IC) This is a function provided by the timing system, whereby an external event is 'captured' by storing the value of a counter at the instant the event is detected.
$\overline{\text{IRQ}}$	Interrupt request. The overline indicates that this is an active-low signal format.
K byte	A kilo-byte (of memory); 1024 bytes.
LCD	Liquid crystal display.
LSB	Least significant byte.
M68HC05	Motorola's family of 8-bit MCUs.
MCU	Microcontroller unit.
MI BUS	Motorola interconnect bus. A single wire, medium speed serial communications protocol.
MSB	Most significant byte.
Nibble	Half a byte; four bits.
NRZ	Non-return to zero.
Opcode	The opcode is a byte which identifies the particular instruction and operating mode to the CPU.
Operand	The operand is a byte containing information the CPU needs to execute a particular instruction.
Output compare	(OC) This is a function provided by the timing system, whereby an external event is generated when an internal counter value matches a predefined value.
PLCC	Plastic leaded chip carrier package.
PLL	Phase-locked loop circuit. This provides a method of frequency multiplication, to enable the use of a low frequency crystal in a high frequency circuit.

Pull-down, pull-up	These terms refer to resistors, sometimes internal to the device, which are permanently connected to either ground or V_{DD}.
PWM	Pulse width modulation. This term is used to describe a technique where the width of the high and low periods of a waveform is varied, usually to enable a representation of an analog value.
QFP	Quad flat pack package.
RAM	Random access memory. Fast read and write, but contents are lost when the power is removed.
RFI	Radio frequency interference.
RTI	Real-time interrupt.
ROM	Read-only memory. This type of memory is programmed during device manufacture and cannot subsequently be altered.
RS-232C	A standard serial communications protocol.
SAR	Successive approximation register.
SCI	Serial communications interface.
Set	'1' — the logic one state; the opposite of 'clear'.
Silicon glen	An area in the central belt of Scotland, so called because of the concentration of semiconductor manufacturers and users found there.
Single chip mode	In this mode the device functions as a self contained unit, requiring only I/O devices to complete a system.
SPI	Serial peripheral interface.
Test mode	This mode is intended for factory testing.
TTL	Transistor-transistor logic.
UART	Universal asynchronous receiver transmitter.
VCO	Voltage controlled oscillator.
Watchdog	*see* 'COP'.
Wired-OR	A means of connecting outputs together such that the resulting composite output state is the logical OR of the state of the individual outputs.
Word	Two bytes; 16 bits.
XIRQ	Non-maskable interrupt request. The overline indicates that this has an active-low signal format.

THIS PAGE LEFT BLANK INTENTIONALLY

Home Page:
www.freescale.com
email:
support@freescale.com
USA/Europe or Locations Not Listed:
Freescale Semiconductor
Technical Information Center, CH370
1300 N. Alma School Road
Chandler, Arizona 85224
(800) 521-6274
480-768-2130
support@freescale.com
Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
support@freescale.com
Japan:
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku
Tokyo 153-0064, Japan
0120 191014
+81 2666 8080
support.japan@freescale.com
Asia/Pacific:
Freescale Semiconductor Hong Kong Ltd.
Technical Information Center
2 Dai King Street
Tai Po Industrial Estate,
Tai Po, N.T., Hong Kong
+800 2666 8080
support.asia@freescale.com
For Literature Requests Only:
Freescale Semiconductor
Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
(800) 441-2447
303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor
@hibbertgroup.com

RoHS-compliant and/or Pb- free versions of Freescale products have the functionality and electrical characteristics of their non-RoHS-compliant and/or non-Pb- free counterparts. For further information, see http://www.freescale.com or contact your Freescale sales representative.

For information on Freescale.s Environmental Products program, go to http://www.freescale.com/epp.

freescale™
semiconductor