68HC705V8

SPECIFICATION

REV 2.1

(General Release)

MCU System Design Group
Oak Hill, Texas

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

SECTION 1 GENERAL DESCRIPTION

The Motorola MC68HC705V8 microcontroller is a custom HC05 based MCU featuring a Message Data Link Controller (MDLC) module and on-chip power regulation. The device is available packaged in a 56-pin SDIP, 68-pin PLCC, or 64-pin QFP. A functional block diagram of the MC68HC705V8 is shown in Figure 1-6.

1.1 FEATURES

- Low cost, HC05 Core
- 12K Bytes of User EPROM
- 512 Bytes of User RAM
- 128 Bytes of byte erasable EEPROM
  - Byte writeable
  - Byte, Block or Bulk erasable
- Message Data Link Controller (MDLC) module
- 16-channel 8-bit A/D converter (8 channels available in 56 SDIP version)
- Full function Serial Peripheral Interface (SPI)
- On-Chip Oscillator for Crystal/Ceramic Resonator
- 8-bit timer with Real Time Interrupt
- 16-bit timer with 1 input capture and 1 output compare
- 38 frequency 6-bit PWM
- COP Watchdog System
- 22 general purpose I/O pins, 16 with interrupt wake-up capability
  6 I/O pins muxed with Timer and SPI pins. 16 Input Only pins muxed with A/D.
- Mask option Low Voltage Reset (LVR).
- On-chip 5V MCU power regulator (mask option to disable)
- Power Saving STOP and WAIT Mode Instructions
  (Mask Selectable STOP Instruction Disable)

NOTE: A line over a signal name indicates an active low signal. For example, RESET is active high and $\overline{\text{RESET}}$ is active low. Any reference to voltage, current, or frequency specified in the following sections will refer to the nominal values. The exact values and their tolerance or limits are specified in **SECTION 17 ELECTRICAL SPECIFICATIONS**.

1.2 MASK OPTIONS

The following mask options are available:

- Sensitivity on $\overline{\text{IRQ}}$ Interrupt, Edge- and Level-Sensitive or Edge-Sensitive Only
- Selectable COP Watchdog System
- Selectable Low Voltage Reset (LVR) to Hold CPU in Reset
- Selectable STOP Instruction Disable
- Selectable on-chip power supply regulator disconnect
- Selectable option to allow regulator wakeup on a rising edge of the MDLC Bus
- Selectable option to tie the regulator output to V_{SS} when disabled

These mask options are provided through individual bits within the Mask Option Register which is located and programmed as part of the EPROM array.

1.3 PIN ASSIGNMENTS

The MC68HC705V8 requires 64 bond pads for all I/O and power supply functions. The device is available in a 56-pin SDIP package where the 8 A/D channels associated with Port E are not bonded out (AN8 through AN15). The device is also available in the 68 pin PLCC and 64-pin QFP packages.

The pin out for the 56-pin SDIP, the 68-pin PLCC, and the 64-pin QFP packages are shown in the following figures. The pad positions are also shown in the following figures.

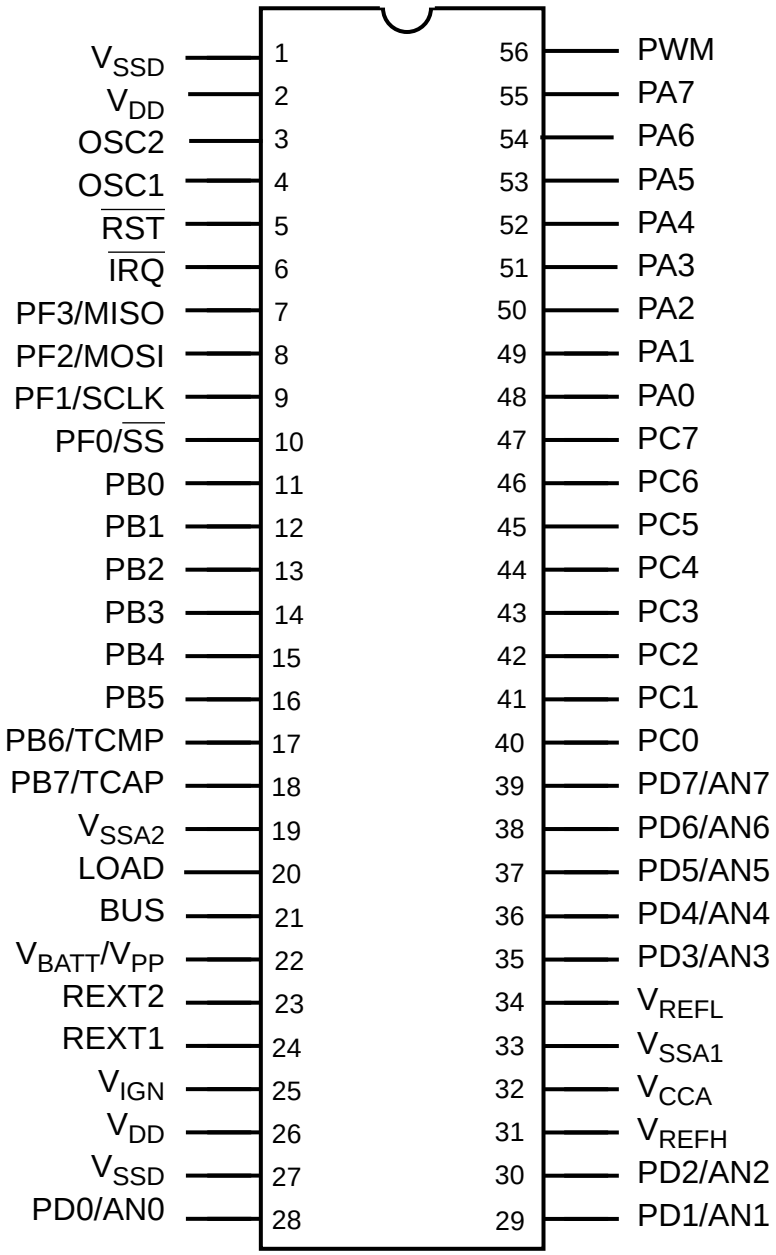


Figure 1-1: MC68HC705V8 Pin Assignments (56 SDIP pin package)

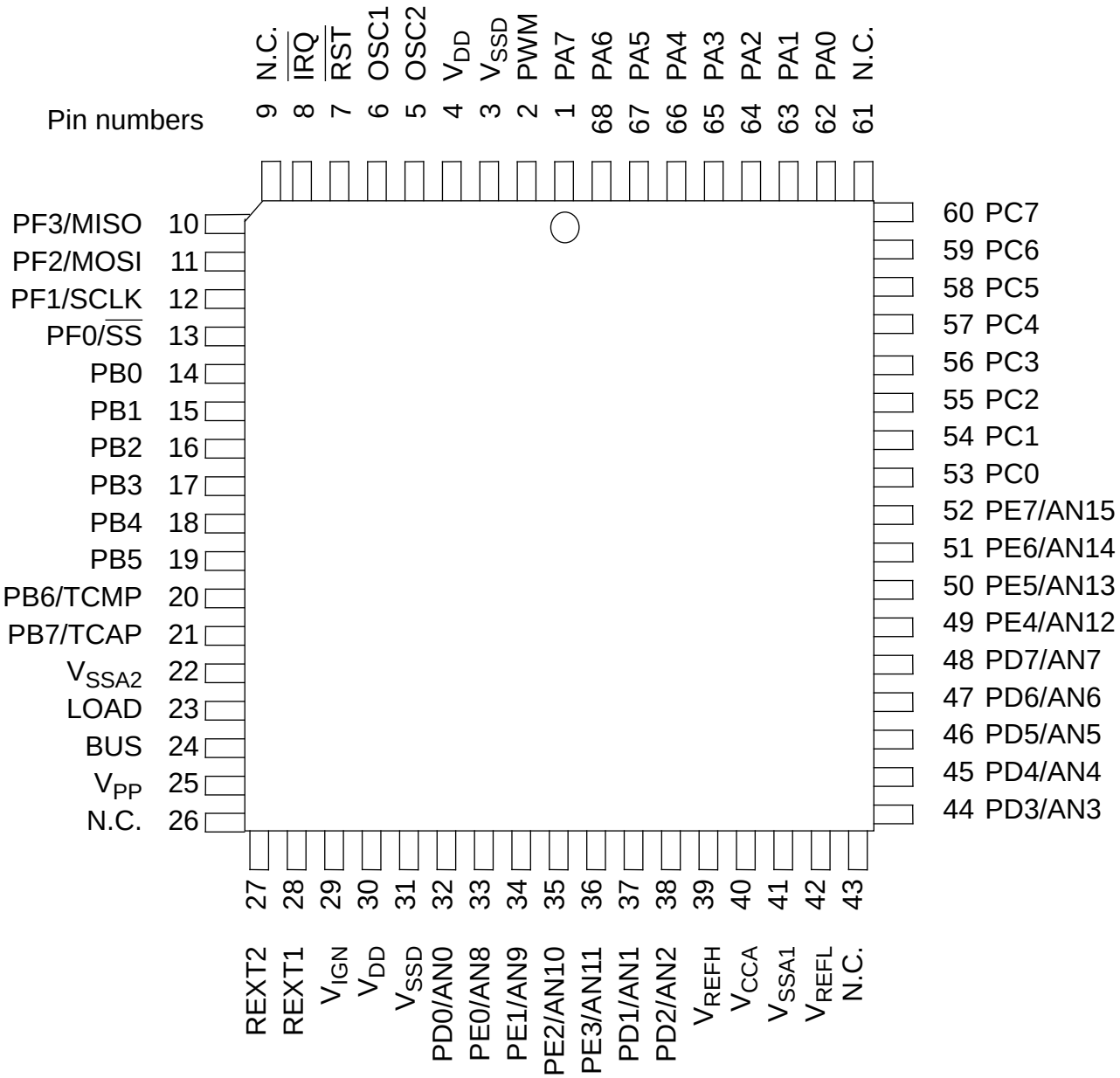


Figure 1-2: MC68HC705V8 Pin Assignments (68-pin PLCC package)

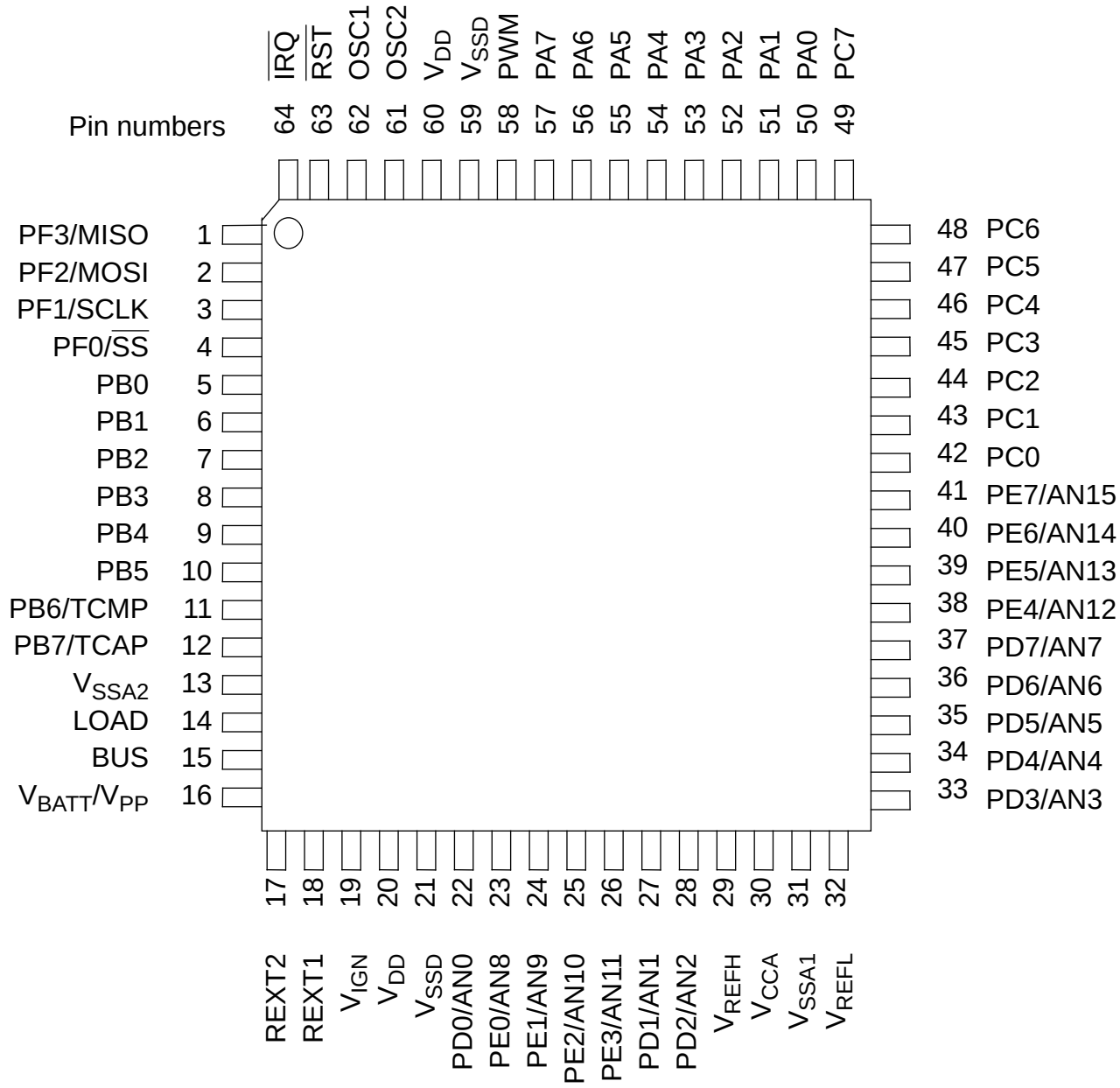


Figure 1-3: MC68HC705V8 Pin Assignments (64-pin QFP)

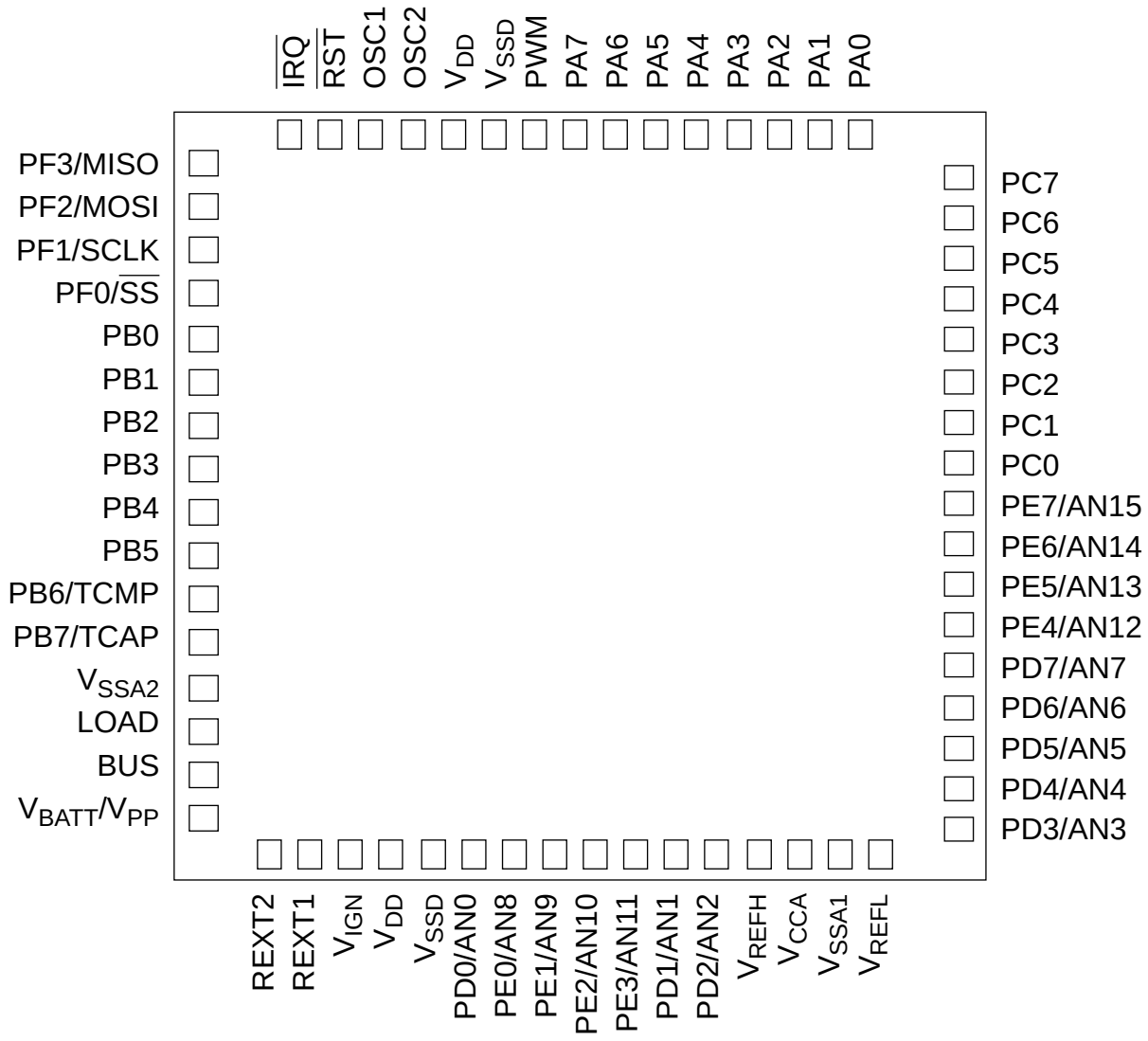


Figure 1-4: MC68HC705V8 Bonding Diagram
Circuit Side Up

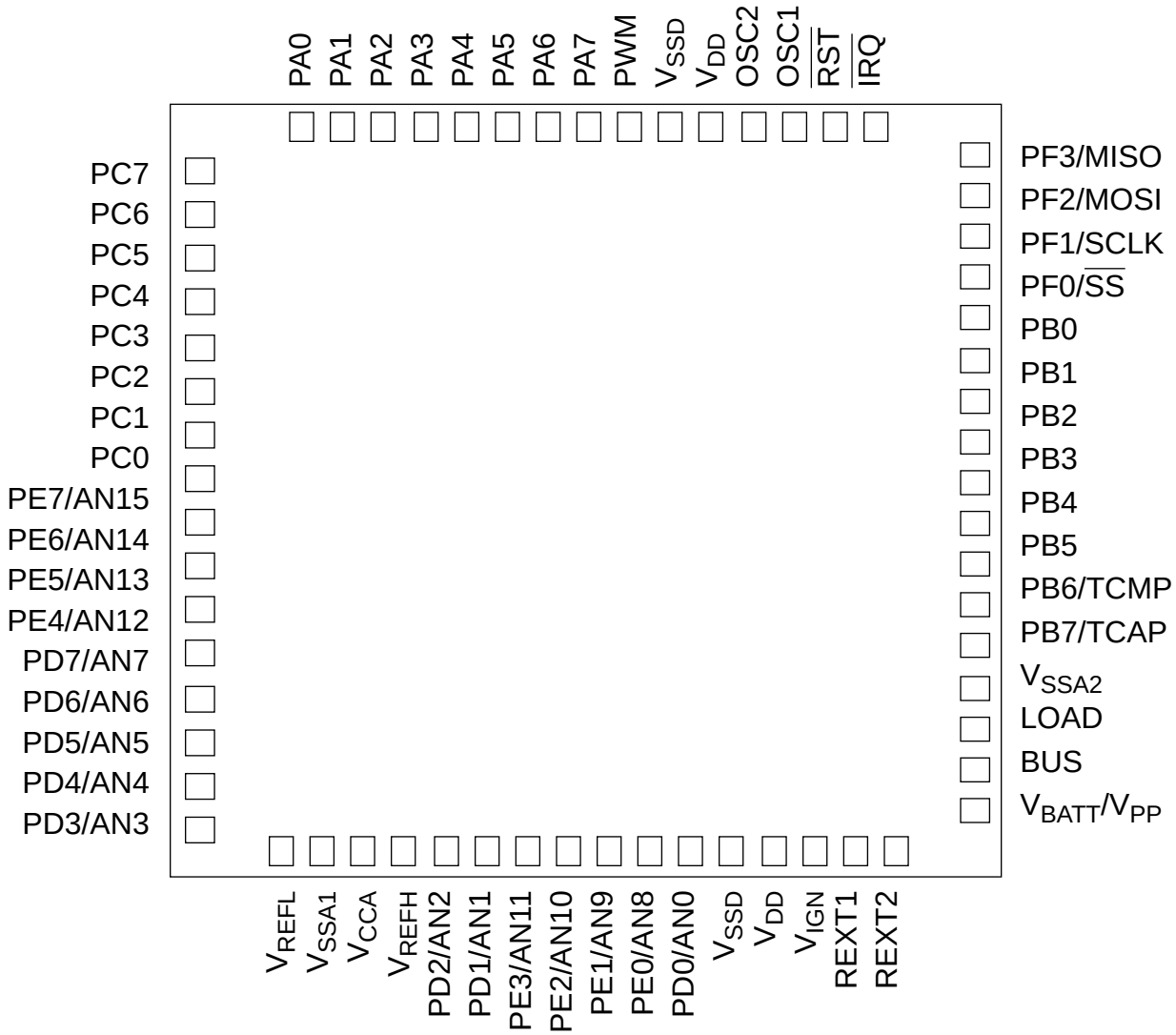


Figure 1-5: MC68HC705V8 Bonding Diagram Circuit Side Down

1.4 MCU STRUCTURE

The overall block diagram of the MC68HC705V8 is shown in Figure 1-6.

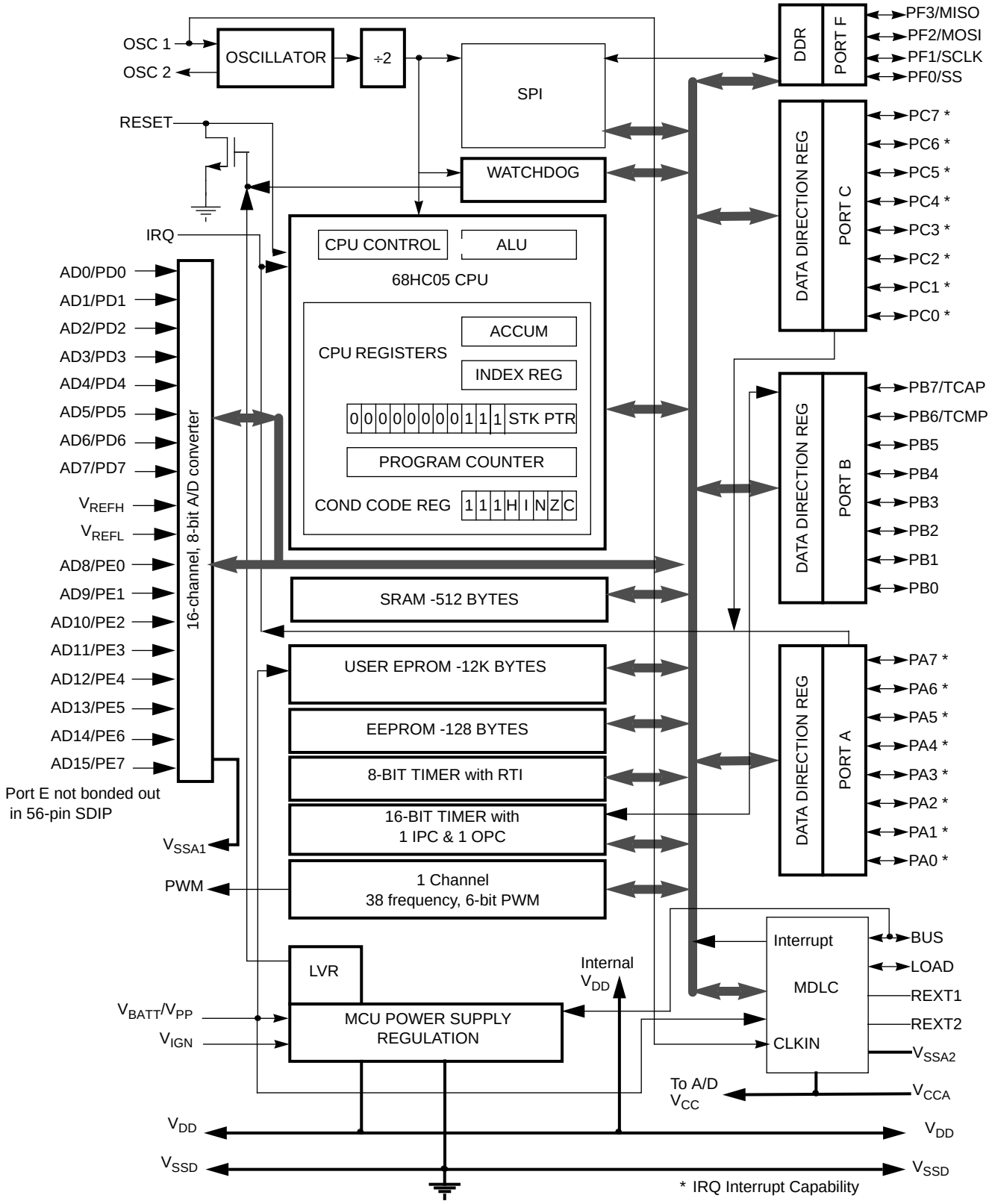


Figure 1-6: MC68HC705V8 Block Diagram

1.5 FUNCTIONAL PIN DESCRIPTION

The following paragraphs give a description of the general function of each pin.

1.5.1 V_{BATT}/V_{PP}

Power is supplied to the device through the V_{BATT} pin. The on-chip voltage regulator uses this voltage to derive the internal V_{DD} supply for the MCU. The V_{BATT} pin is also used to power the MDLC and should still be connected even when the Voltage Regulator is not used. This pin is also used to provide the programming voltage for the EPROM array. See **SECTION 3 EPROM AND EEPROM** for more details on EPROM programming.

See SECTION 7 POWER SUPPLY AND REGULATION.

1.5.2 V_{DD} AND V_{SS}

These pins are provided to allow the internally generated V_{DD} supply to be externally decoupled. The short rise and fall times of the MCU supply current transients place very high short-duration current demands on the internal power supply. To prevent noise problems, special care should be taken to provide good power supply bypassing at the MCU by using bypass capacitors with good high-frequency characteristics that are positioned as close to the MCU supply pins as possible. Two sets of V_{DD} and V_{SS} pins are required to maintain on-chip supply noise to within acceptable limits. Each supply pin pair will require its own decoupling capacitor. See **SECTION 7 POWER SUPPLY AND REGULATION**.

1.5.3 V_{SSA1}

V_{SSA1} is a separate ground pad which provides a ground return for the A/D subsystem. To prevent digital noise contamination, this pin should be connected directly to a low impedance ground reference point.

1.5.4 V_{SSA2}

V_{SSA2} is a separate ground pad that provides a ground return for the MDLC analog subsystem. To prevent digital noise contamination, this pin should be connected directly to a low impedance ground reference point.

1.5.5 V_{CCA}

V_{CCA} is a separate supply pin providing power to the analog subsystems of the MDLC and A/D converter. This pin must be connected to the V_{DD} pin externally. To prevent contamination from the digital supply, this pin should be adequately decoupled to a low impedance ground reference. See **SECTION 7 POWER SUPPLY AND REGULATION**.

1.5.6 V_{REFH} AND V_{REFL}

V_{REFH} is the positive (high) reference voltage for the A/D subsystem. V_{REFL} is the negative (low) reference voltage for the A/D subsystem. V_{REFH} and V_{REFL} should be isolated from the digital supplies to prevent any loss of accuracy from the A/D converter.

1.5.7 OSC1, OSC2

The OSC1 and OSC2 pins are the connections for the on-chip oscillator. OSC1 is the input to the oscillator inverter. The output (OSC2) will always reflect OSC1 inverted except when the device is in STOP mode, which forces OSC2 high. The OSC1, and OCS2 pins can accept the following sets of components:

1. A crystal as shown in Figure 1-7(a)
2. A ceramic resonator as shown in Figure 1-7(a)
3. An external clock signal as shown in Figure 1-7(b)

The frequency, f_{OSC}, of the oscillator or external clock source is divided by two to produce the internal operating frequency, f_{OP}.

1.5.7.1 Crystal Oscillator

The circuit in Figure 1-7(a) shows a typical oscillator circuit for an AT-cut, parallel resonant crystal. The crystal manufacturer's recommendations should be followed, as the crystal parameters determine the external component values required to provide maximum stability and reliable start-up. The load capacitance values used in the oscillator circuit design should include all stray capacitances. The crystal and components should be mounted as close as possible to the pins for start-up stabilization and to minimize output distortion and radiated emissions.

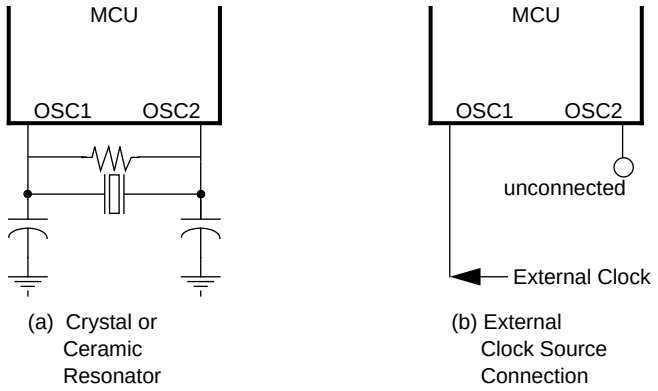


Figure 1-7: Oscillator Connections

1.5.7.2 Ceramic Resonator Oscillator

In cost-sensitive applications, a ceramic resonator can be used in place of the crystal. The circuit in Figure 1-7(a) is for a ceramic resonator. The resonator manufacturer's recommendations should be followed, as the resonator parameters determine the external component values required for maximum stability and reliable starting. The load capacitance values used in the oscillator circuit design should include all stray capacitances. The ceramic resonator and components should be mounted as close as possible to the pins for start-up stabilization and to minimize output distortion and radiated emissions.

1.5.7.3 External Clock

An external clock from another CMOS-compatible device can be connected to the OSC1 input. The OSC2 pin should be left unconnected, as shown in Figure 1-7(b).

1.5.8 $\overline{\text{RESET}}$

This pin can be used as an input to reset the MCU to a known start-up state by pulling it to the low state. The $\overline{\text{RESET}}$ pin contains an internal Schmitt trigger to improve its noise immunity as an input. The $\overline{\text{RESET}}$ pin has an internal pulldown device that pulls the $\overline{\text{RESET}}$ pin low when there is an internal COP Watchdog reset, POR, illegal address reset, a disabled STOP instruction reset, or an internal low voltage reset. Refer to **SECTION 6 RESETS**.

1.5.9 $\overline{\text{IRQ}}$ (MASKABLE INTERRUPT REQUEST)

This input pin drives the asynchronous IRQ interrupt function of the CPU. The IRQ interrupt function has a mask option to select either negative edge-sensitive triggering or both negative edge-sensitive and low level-sensitive triggering. The IRQ input requires an external resistor to V_{DD} for "wire-OR" operation, if desired. If the $\overline{\text{IRQ}}$ pin is not used, it must be tied to the V_{DD} supply. The $\overline{\text{IRQ}}$ pin contains an internal Schmitt trigger as part of its input to improve noise immunity. Each of the PA0 through PA7 and PC0 through PC7 I/O pins may be connected as an OR function with the IRQ interrupt function. This capability allows keyboard scan applications where the transitions on the I/O pins will behave the same as the $\overline{\text{IRQ}}$ pin. The edge or level sensitivity selected by a mask option for the $\overline{\text{IRQ}}$ pin does not apply to the I/O pin interrupt. The I/O pin interrupt is always negative edge sensitive.

See **SECTION 5 INTERRUPTS** for more details on the interrupts.

1.5.10 PA0-PA7

These eight I/O lines comprise Port A. The state of any pin is software programmable and all Port A lines are configured as inputs during power-on or reset. All eight pins are connected via an internal gate to the IRQ interrupt function. When the IRQ interrupt function is enabled, all the Port A pins will act as negative edge sensitive IRQ sources. See **SECTION 9 PARALLEL I/O** for more details on the I/O ports.

1.5.11 PB0-PB5, PB6/TCMP, PB7/TCAP

These eight I/O lines comprise Port B. The state of any pin is software programmable and all Port B lines are configured as inputs during power-on or reset. See **SECTION 9 PARALLEL I/O** for more details on the I/O ports. PB6 and PB7 are also shared with timer functions. The TCAP pin controls the input capture feature for the on-chip 16-bit timer. The TCMP pin provides an output for the output compare feature of the on-chip 16-bit timer. See **SECTION 11 16-BIT TIMER** for more details on the operation of the timer subsystem.

1.5.12 PC0-PC7

These eight I/O lines comprise Port C. The state of any pin is software programmable and all Port C lines are configured as inputs during power-on or reset. All eight pins are connected via an internal gate to the $\overline{\text{IRQ}}$ interrupt function. When the $\overline{\text{IRQ}}$ interrupt function

is enabled, all the Port C pins will act as negative edge sensitive IRQ sources. See **SECTION 9 PARALLEL I/O** for more details on the I/O ports.

1.5.13 AD0-AD7 / PD0-PD7: AD8-AD15/PE0-PE7

When the A/D converter is disabled, PD0-PD7 and PE0-PE7 are general purpose input pins. The A/D converter is disabled upon exiting from reset. When the A/D converter is enabled, one of these pins is the analog input to the A/D converter. The A/D control register contains control bits to direct which of the analog inputs are to be converted at any one time. A digital read of this pin when the A/D converter is enabled results in a read of logical zero from the selected analog pin. A digital read of the remaining pins gives their correct (digital) values. A/D inputs AD8-AD15 (Port E) are not bonded out in the 56-pin package. See **SECTION 10 A/D CONVERTER** for more details on the operation of the A/D subsystem.

1.5.14 PWM

This pin provides an output for the pulse width modulation feature of the on-chip programmable timer. See **SECTION 13 PULSE WIDTH MODULATOR** for more details on the operation of the PWM subsystem.

1.5.15 PF0/$\overline{SS}$, PF1/SCK, PF2/MOSI, PF3/MISO

These four I/O lines comprise Port F. The state of any pin is software programmable and all Port F lines are configured as inputs during power-on or reset. See **SECTION 9 PARALLEL I/O** for more details on the I/O ports. When the SPI subsystem is enabled, PF0-PF3 become the data, clock and select lines for the SPI. See **SECTION 14 SERIAL PERIPHERAL INTERFACE** for more details on the operation of the SPI subsystem.

1.5.16 BUS, LOAD, REXT1, REXT2

These pins provide the I/O interface and external biasing functions for the MDLC subsystem. See **SECTION 15 MESSAGE DATA LINK CONTROLLER** for more details on the operation of the MDLC subsystem. The regulator control logic monitors the BUS pin and latches a rising edge to re-enable the primary voltage regulator if this mask option is enabled.

1.5.17 V_{IGN}

The V_{IGN} pin is used by the internal voltage regulator power moding circuitry to indicate that the regulator should power up. Its state is reflected in the IGNS bit of the MISC register. The input voltage levels on this pin are ratioed to the voltage on V_{BATT}. See the electrical specifications. A smaller secondary regulator remains alive to power the V_{IGN} pin logic when the primary voltage regulator is powered down, allowing the power moding logic to recognize and latch a rising edge on this pin and re-enable the primary regulator (power-up). See **SECTION 7 POWER SUPPLY AND REGULATION** for a more detailed description.

The V_{IGN} pin has no function when the option to disable the on-chip voltage regulator is selected and should be tied low.

SECTION 2 MEMORY MAP

2.1 SINGLE-CHIP MODE MEMORY MAP

When the MC68HC705V8 is in the Single-Chip Mode the I/O and peripherals, user RAM, EEPROM and user EPROM are all active as shown in Figure 2-1.

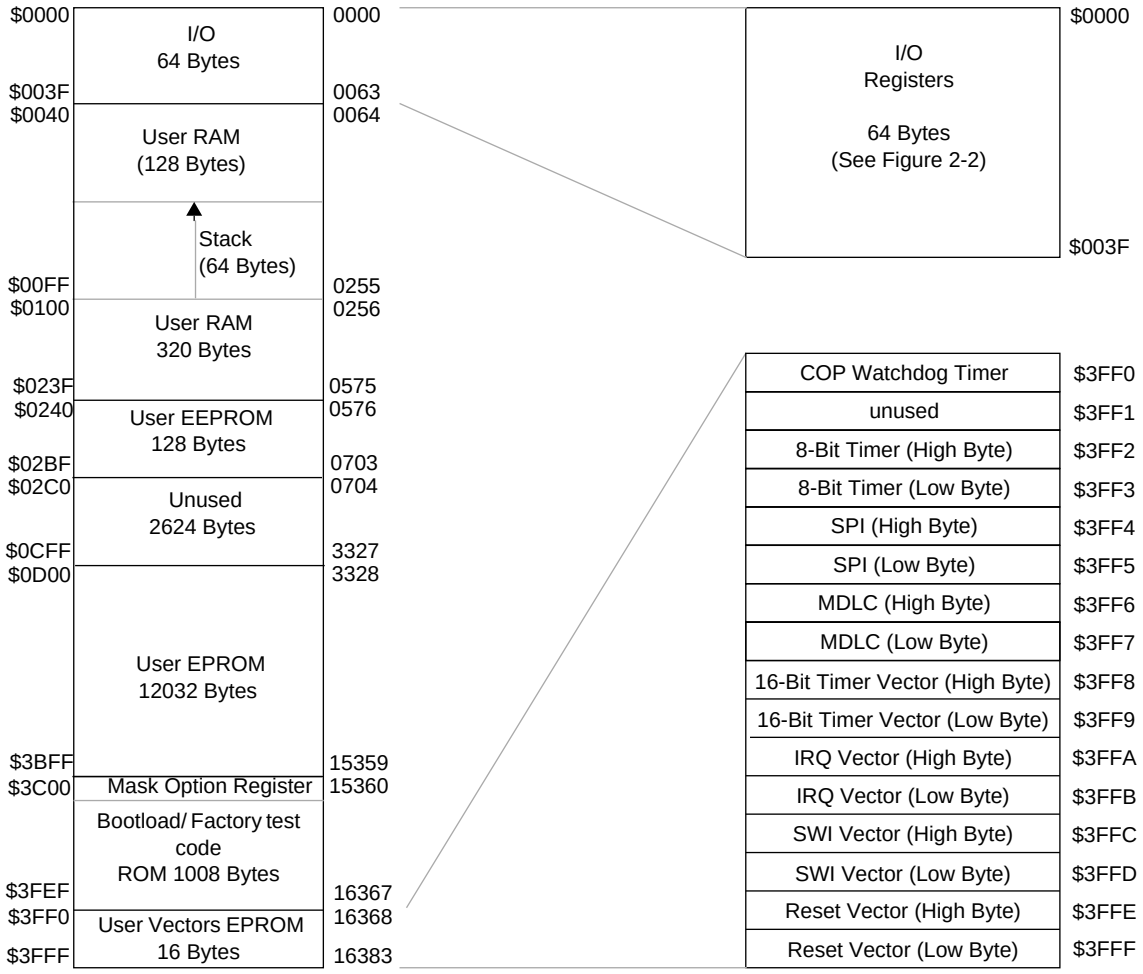


Figure 2-1: MC68HC705V8 Single-Chip Mode Memory Map

2.2 I/O AND CONTROL REGISTERS

The I/O and Control Registers reside in locations $0000-$003F. The overall organization of these registers is shown in Figure 2-2. The bit assignments for each register are shown in Figure 2-3, Figure 2-4, Figure 2-5, and Figure 2-6. Reading from unimplemented bits will return unknown states, and writing to unimplemented bits will be ignored.

Port A Data Register	$0000
Port B Data Register	$0001
Port C Data Register	$0002
Port D Data Register	$0003
Port A Data Direction Register	$0004
Port B Data Direction Register	$0005
Port C Data Direction Register	$0006
Unused	$0007
8-Bit Timer Status & Control	$0008
8-Bit Timer Counter Register	$0009
SPI Control Register	$000A
SPI Status Register	$000B
SPI Data Register	$000C
EPROM Program Register	$000D
MDLC Control Register	$000E
MDLC Status Register	$000F
MDLC Tx Control Register	$0010
MDLC Rx Status Register	$0011
16-Bit Timer Control Register	$0012
16-Bit Timer Status Register	$0013
Input Capture Register (high)	$0014
Input Capture Register (low)	$0015
Output Compare Register (high)	$0016
Output Compare Register (low)	$0017
16-Bit Timer Count Register (high)	$0018
16-Bit Timer Count Register (low)	$0019
Alt. Count Register (high)	$001A
Alt. Count Register (low)	$001B
EEPROM Program Register	$001C
A/D Data Register	$001D
A/D Status & Control Register	$001E
IRQ Control and Status Register	$001F

MDLC Tx Data Register 0	$0020
MDLC Tx Data Register 1	$0021
MDLC Tx Data Register 2	$0022
MDLC Tx Data Register 3	$0023
MDLC Tx Data Register 4	$0024
MDLC Tx Data Register 5	$0025
MDLC Tx Data Register 6	$0026
MDLC Tx Data Register 7	$0027
MDLC Tx Data Register 8	$0028
MDLC Tx Data Register 9	$0029
MDLC Tx Data Register 10	$002A
Port E Data Register	$002B
Port F Data Register	$002C
Unused	$002D
Port F Data Direction Register	$002E
Miscellaneous Register	$002F
PWM Data Register	$0030
PWM Control Register	$0031
Unused	$0032
Unused	$0033
MDLC Rx Data Register 0	$0034
MDLC Rx Data Register 1	$0035
MDLC Rx Data Register 2	$0036
MDLC Rx Data Register 3	$0037
MDLC Rx Data Register 4	$0038
MDLC Rx Data Register 5	$0039
MDLC Rx Data Register 6	$003A
MDLC Rx Data Register 7	$003B
MDLC Rx Data Register 8	$003C
MDLC Rx Data Register 9	$003D
MDLC Rx Data Register10	$003E
Reserved	$003F

Figure 2-2: MC68HC705V8 I/O Registers Memory Map

2.3 RAM

The total RAM consists of 512 bytes (including the stack). The stack begins at address $00FF and proceeds down to $00C0 (64 bytes). Using the stack area for data storage or temporary work locations requires care to prevent it from being overwritten due to stacking from an interrupt or subroutine call.

NOTE: The stack is located in the middle of the RAM address space. Data written to addresses within the stack address range could be overwritten during stack activity.

2.4 BOOTROM

The ROM space in the MC68HC705V8 consists of 1008 bytes of EPROM Bootload code, EEPROM test code, burn-in code, and 16 bytes of selfcheck vectors. The MOR register is located in this space.

2.5 EPROM

There are 12032 bytes of user EPROM and 16 bytes of EPROM for user vectors and the COP update location. Refer to **SECTION 3 EPROM AND EEPROM** for programming details.

2.6 EEPROM

This device contains 128 bytes of EEPROM. Programming the EEPROM is performed by the user on a single-byte basis by manipulating the Programming Register, located at address $001C. Refer to **SECTION 3 EPROM AND EEPROM** for programming details.

ADDR	REGISTER	READ / WRITE	7	6	5	4	3	2	1	0
$0000	PORT A DATA	R	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
		W								
$0001	PORT B DATA	R	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
		W								
$0002	PORT C DATA	R	PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0
		W								
$0003	PORT D DATA	R	PD7	PD6	PD5	PD4	PD3	PD2	PD1	PD0
		W								
$0004	PORT A DATA DIRECTION	R	DDRA7	DDRA6	DDRA5	DDRA4	DDRA3	DDRA2	DDRA1	DDRA0
		W								
$0005	PORT B DATA DIRECTION	R	DDRB7	DDRB6	DDRB5	DDRB4	DDRB3	DDRB2	DDRB1	DDRB0
		W								
$0006	PORT C DATA DIRECTION	R	DDRC7	DDRC6	DDRC5	DDRC4	DDRC3	DDRC2	DDRC1	DDRC0
		W								
$0007	UNUSED	R								
		W								
$0008	8 BIT TIMER STATUS/CONTROL	R	CTOF	RTIF	TOFE	RTIE			RT1	RT0
		W					TOFC	RTFC		
$0009	8 BIT TIMER COUNTER	R	D7	D6	D5	D4	D3	D2	D1	D0
		W								
$000A	SPI CONTROL REGISTER	R	SPIE	SPE		MSTR	CPOL	CPHA	SPR1	SPR0
		W								
$000B	SPI STATUS REGISTER	R	SPIF	WCOL		MODF				
		W								
$000C	SPI DATA REGISTER	R	SPD7	SPD6	SPD5	SPD4	SPD3	SPD2	SPD1	SPD0
		W								
$000D	EPROM PROGRAM REGISTER	R	MORON					LATCH		EPGM
		W								
$000E	MDLC CONTROL REGISTER	R	RXBM	TXAB	R1	R0			IE	WCM
		W								
$000F	MDLC STATUS REGISTER	R					TXMS	RXMS		
		W								

UNIMPLEMENTED ONE TIME WRITE

Figure 2-3: MC68HC705V8 I/O Registers $0000-$000F

ADDR	REGISTER	READ WRITE	7	6	5	4	3	2	1	0
$0010	MDLC TX CONTROL REGISTER	R					TC3	TC2	TC1	TC0
		W								
$0011	MDLC RX STATUS REGISTER	R					RC3	RC2	RC1	RC0
		W								
$0012	TIMER CONTROL REGISTER	R	ICIE	OCIE	TOIE			TON	IEDG	OLVL
		W								
$0013	TIMER STATUS REGISTER	R	ICF	OCF	TOF					
		W								
$0014	INPUT CAPTURE HIGH	R	15	14	13	12	11	10	9	8
		W								
$0015	INPUT CAPTURE LOW	R	7	6	5	4	3	2	1	0
		W								
$0016	OUTPUT COMPARE HIGH	R	15	14	13	12	11	10	9	8
		W								
$0017	OUTPUT COMPARE LOW	R	7	6	5	4	3	2	1	0
		W								
$0018	TIMER COUNTER HIGH	R	15	14	13	12	11	10	9	8
		W								
$0019	TIMER COUNTER LOW	R	7	6	5	4	3	2	1	0
		W								
$001A	ALT. COUNTER HIGH	R	15	14	13	12	11	10	9	8
		W								
$001B	ALT. COUNTER LOW	R	7	6	5	4	3	2	1	0
		W								
$001C	EEPROM PROGRAMMING	R		CPEN		ER1	ER0	LATCH	EERC	EEPGM
		W								
$001D	A/D DATA	R	D7	D6	D5	D4	D3	D2	D1	D0
		W								
$001E	A/D STATUS AND CONTROL	R	COCO	ADRC	ADON	CH4	CH3	CH2	CH1	CH0
		W								
$001F	IRQ CONTROL AND STATUS	R	IRQE	IRQPAE	IRQPCE		IRQF	IRQPAF	IRQPCF	0
		W								IRQA

UNIMPLEMENTED ONE TIME WRITE

Figure 2-4: MC68HC705V8 I/O Registers $0010-$001F

ADDR	REGISTER	READ WRITE	7	6	5	4	3	2	1	0
$0020	MDLC TX DATA REGISTER 0	R								
		W	D7	D6	D5	D4	D3	D2	D1	D0
$0021	MDLC TX DATA REGISTER 1	R								
		W	D7	D6	D5	D4	D3	D2	D1	D0
$0022	MDLC TX DATA REGISTER 2	R								
		W	D7	D6	D5	D4	D3	D2	D1	D0
$0023	MDLC TX DATA REGISTER 3	R								
		W	D7	D6	D5	D4	D3	D2	D1	D0
$0024	MDLC TX DATA REGISTER 4	R								
		W	D7	D6	D5	D4	D3	D2	D1	D0
$0025	MDLC TX DATA REGISTER 5	R								
		W	D7	D6	D5	D4	D3	D2	D1	D0
$0026	MDLC TX DATA REGISTER 6	R								
		W	D7	D6	D5	D4	D3	D2	D1	D0
$0027	MDLC TX DATA REGISTER 7	R								
		W	D7	D6	D5	D4	D3	D2	D1	D0
$0028	MDLC TX DATA REGISTER 8	R								
		W	D7	D6	D5	D4	D3	D2	D1	D0
$0029	MDLC TX DATA REGISTER 9	R								
		W	D7	D6	D5	D4	D3	D2	D1	D0
$002A	MDLC TX DATA REGISTER 10	R								
		W	D7	D6	D5	D4	D3	D2	D1	D0
$002B	PORT E DATA	R	PE7	PE6	PE5	PE4	PE3	PE2	PE1	PE0
		W								
$002C	PORT F DATA	R					PF3	PF2	PF1	PF0
		W								
$002D	UNUSED	R								
		W								
$002E	PORT F DATA DIRECTION	R					DDRF3	DDRF2	DDRF1	DDRF0
		W								
$002F	MISCELLANEOUS	R	IGNS	OCE						PDC
		W								

Figure 2-5: MC68HC705V8 I/O Registers $0020-$002F

ADDR	REGISTER	READ WRITE	7	6	5	4	3	2	1	0
$0030	PWM DATA	R	POL		D5	D4	D3	D2	D1	D0
		W								
$0031	PWM CONTROL	R	PSA1	PSA0			PSB3	PSB2	PSB1	PSB0
		W								
$0032	UNUSED	R								
		W								
$0033	UNUSED	R								
		W								
$0034	MDLC RX DATA REGISTER 0	R	D7	D6	D5	D4	D3	D2	D1	D0
		W								
$0035	MDLC RX DATA REGISTER 1	R	D7	D6	D5	D4	D3	D2	D1	D0
		W								
$0036	MDLC RX DATA REGISTER 2	R	D7	D6	D5	D4	D3	D2	D1	D0
		W								
$0037	MDLC RX DATA REGISTER 3	R	D7	D6	D5	D4	D3	D2	D1	D0
		W								
$0038	MDLC RX DATA REGISTER 4	R	D7	D6	D5	D4	D3	D2	D1	D0
		W								
$0039	MDLC RX DATA REGISTER 5	R	D7	D6	D5	D4	D3	D2	D1	D0
		W								
$003A	MDLC RX DATA REGISTER 6	R	D7	D6	D5	D4	D3	D2	D1	D0
		W								
$003B	MDLC RX DATA REGISTER 7	R	D7	D6	D5	D4	D3	D2	D1	D0
		W								
$003C	MDLC RX DATA REGISTER 8	R	D7	D6	D5	D4	D3	D2	D1	D0
		W								
$003D	MDLC RX DATA REGISTER 9	R	D7	D6	D5	D4	D3	D2	D1	D0
		W								
$003E	MDLC RX DATA REGISTER 10	R	D7	D6	D5	D4	D3	D2	D1	D0
		W								
$003F	RESERVED	R								
		W								

UNIMPLEMENTED

Figure 2-6: MC68HC705V8 I/O Registers $0030-$003F

SECTION 3 EPROM AND EEPROM

The MC68HC705V8 contains EPROM and EEPROM Memory. This section describes the programming mechanisms for each type of memory.

3.1 EPROM BOOTLOADER

Bootloader Programming Mode is entered upon the rising edge of $\overline{\text{RESET}}$ if the $\overline{\text{IRQ}}$ pin is at V_{TST} and the PB7 pin is at logic one. The Bootloader code resides in the ROM from $3C00 to $3FEF. This program handles copying of user code from an external EPROM into the on-chip EPROM.

The user code must be a one-to-one correspondence with the internal EPROM addresses.

3.1.1 BOOTLOADER FUNCTIONS

Three pins are used to select various bootloader functions. These pins are PC2, PC1 and PC0. PC3 and PC4 are tied to logic zero. Two other pins, PC7 and PC6 are used to drive the PROG LED and the VERF LED respectively. The programming modes are shown in Table 3-1.

Table 3-1: Bootloader Functions

PC2	PC1	PC0	MODE
0	0	0	Program/Verify EPROM
0	0	1	Verify Only
0	1	0	Factory use
0	1	1	Jump to top of EEPROM
1	0	0	Jump to top of RAM
1	0	1	Jump to top of EPROM

3.2 EPROM PROGRAMMING

The EPROM array is programmed through manipulation of the Programming Register located at $000D. It maybe programmed in user, bootstrap or test mode. The schematic of the EPROM programmer using the bootstrap firmware is shown in Figure 3-1. In addition to the main EPROM array, the Mask Option Register must also be programmed appropriately by the programming software.

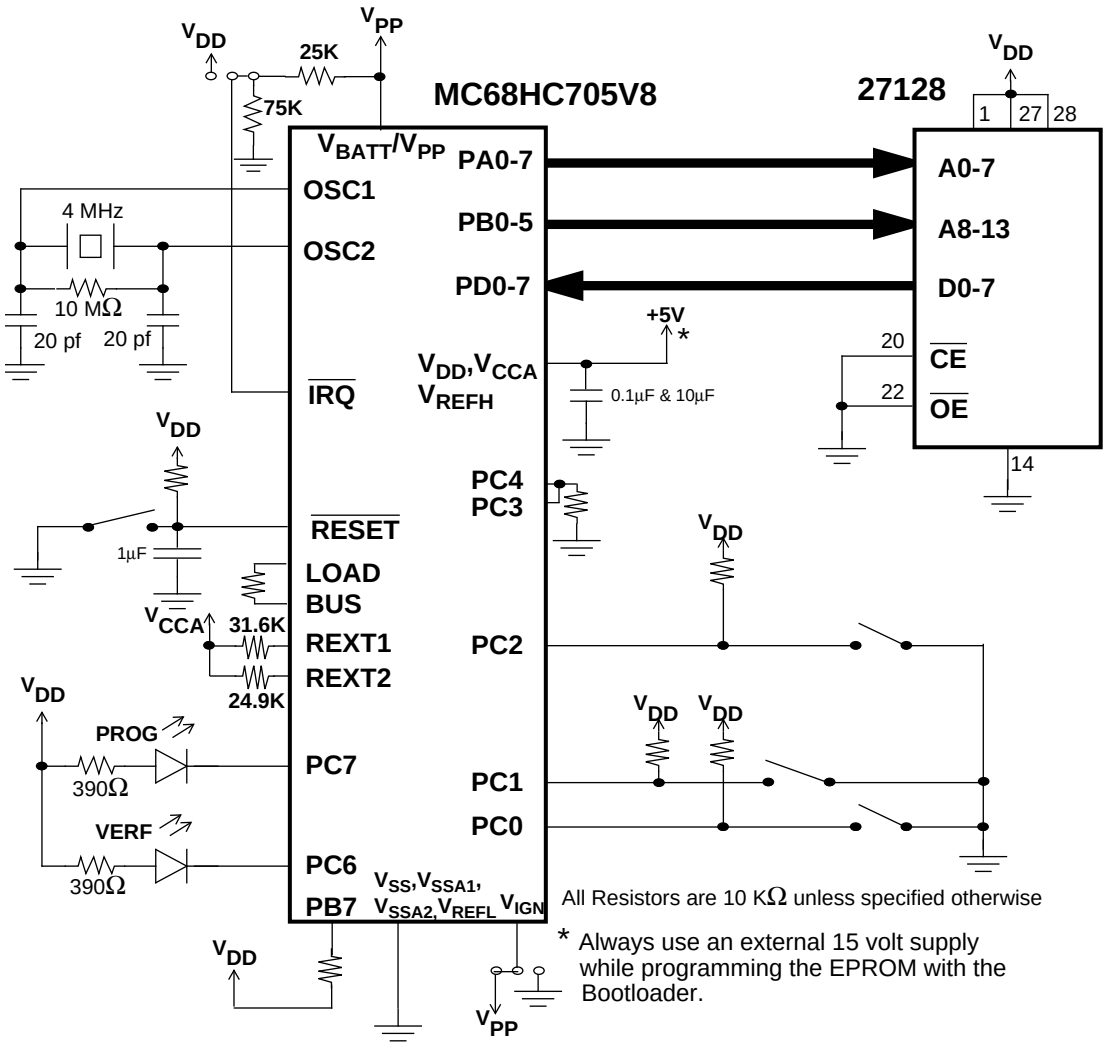


Figure 3-1: Bootstrap EPROM Programmer Schematic

3.2.1 PROGRAMMING REGISTER $0D

This register is used to program the EPROM array. To program a byte of EPROM, set LATCH, then write data to the desired address, then set EPGM for t_{EPGM}.

$0D	MORON	0	0	0	0	LATCH	0	EPGM
RESET:	0	0	0	0	0	0	0	0

Figure 3-2: Programming Register

MORON - Mask Option Register On

This bit enables and disables the decoding of the MOR register.

0- The first byte of Boot ROM will be read from location $3C00.

1- The MOR register contents are placed into the memory map at location $3C00.

The contents of the MOR register can be read/written only if this bit is set and is available in any operating mode. This bit must be set when the MOR byte is being programmed.

LATCH - EPROM Latch Control

This bit latches the address and data bus when a write to the EPROM array is performed.

NOTE: The EPROM array cannot be read while this bit is set.

---

0 - EPROM address and data bus configured for normal reads.

1 - EPROM address and data bus configured for programming.

EPGM - EPROM Program Control

READ: Any time

WRITE: Cleared any time, Set only if LATCH=1

This bit controls the programming voltage to the EPROM array. EPGM can not be set if LATCH is not already set. EPGM is automatically cleared when LATCH = 0.

NOTE: LATCH and EPGM cannot both be set on the same write.

---

0 - Programming power switched off to the EPROM array.

1 - Programming power switched on to the EPROM array.

The sequence for programming the EPROM is as follows:

1. Set the LATCH bit. If programming the MOR byte, also set the MORON bit.

2. Write the data to be programmed to the EPROM (or MOR byte) location to be programmed.

3. Set the EPGM bit.

4. Wait a time t_{EPGM}

5. Clear the LATCH, MORON (if programming the MOR byte) and EPGM bits.

6. Repeat for each byte.

3.3 MASK OPTION REGISTER (MOR) $3C00

The Mask Option Register (MOR) is used to select all mask options available on the MC68HC705V8. When in the erased state, the EPROM cells will read as a logic zero

which will therefore represent the value transferred from the MOR at reset if it is left unprogrammed.

.

$3C00	-	REGEN	VDDC	MDLCPU	LVR	STOPEN	IRQ	COPEN
ERASED:	0	0	0	0	0	0	0	0

Figure 3-3: Option Register

REGEN - Regulator Enable or Disable

1 = internal voltage regulator is enabled.
0 = internal voltage regulator is disabled.

COPEN - COP Timer Enable or Disable

1 = COP timer enabled.
0 = COP timer disabled.

IRQ - Interrupt Request Pin Sensitivity

1 = $\overline{\text{IRQ}}$ pin is both negative edge and level-sensitive.

0 = $\overline{\text{IRQ}}$ pin is negative edge-sensitive only.

LVR - Low Voltage Reset Enable or Disable

1 = LVR enabled
0 = LVR disabled

STOPEN - Enable the entry into STOP mode via the execution of the STOP instruction.

1 = STOP mode enabled
0 = STOP mode disabled. If STOP instruction is executed, a chip reset will result.

MDLCPU - Enable the MDLC BUS pin to power up the on board regulator on a rising edge

1 = enable
0 = disable

VDDC - V_{DD} Clamp. When this option is enabled and the on board regulator has been disabled through software, the V_{DD} pin will be clamped to V_{SS} through an active device.

1 = enable
0 = disable

NOTE: All options are disabled while the MOR register is being programmed. (MORON=LATCH=EPGM=1 in EPROM Programming Register with the exception of the REGEN bit.)

3.4 EEPROM PROGRAMMING REGISTER $1C

The contents and use of the programming register are discussed below.

$1C	-	CPEN	-	ER1	ER0	LATCH	EERC	EEPGM
RESET:	0	0	0	0	0	0	0	0

Figure 3-4: Programming Register

NOTE: Any reset including LVR will abort any write in progress when it is asserted. Data written to the addressed byte will therefore be indeterminate.

3.4.1 CPEN - Charge Pump Enable

When set, CPEN enables the charge pump which produces the internal programming voltage. This bit should be set with the LATCH bit. The programming voltage will not be available until EEPGM is set. The charge pump should be disabled when not in use. CPEN is readable and writable and is cleared by reset.

3.4.2 ER1:ER0 - Erase Select Bits

ER1 and ER0 form a 2-bit field which is used to select one of three erase modes: byte, block, or bulk. Table 3-2 shows the modes selected for each bit configuration. These bits are readable and writable and are cleared by reset.

In byte erase mode, only the selected byte is erased. In block mode, a 32-byte block of EEPROM is erased. The EEPROM memory space is divided into four 32-byte blocks ($0240-$025F, $0260-$027F, $0280-$029F, $02A0-$02BF), and doing a block erase to any address within a block will erase the entire block. In bulk erase mode, the entire 128 byte EEPROM section is erased.

Table 3-2: Erase Mode Select

ER1	ER0	MODE
0	0	No Erase
0	1	Byte Erase
1	0	Block Erase
1	1	Bulk Erase

3.4.3 LATCH

When set, LATCH configures the EEPROM address and data bus for programming. When LATCH is set, writes to the EEPROM array cause the data bus and the address bus to be latched. This bit is readable and writable, but reads from the array are inhibited if the LATCH bit is set and a write to the EEPROM space has taken place. When clear, address and data buses are configured for normal operation. Reset clears this bit.

3.4.4 EERC - EEPROM RC Oscillator Control

When this bit is set, the EEPROM section uses the internal RC oscillator instead of the CPU clock. After setting the EERC bit, delay a time t_{RCON} to allow the RC oscillator to stabilize. This bit is readable and writable and should be set by the user when the internal bus frequency falls below 1.5 MHz. Reset clears this bit.

3.4.5 EEPGM - EEPROM Programming Power Enable

EEPGM must be written to enable (or disable) the EEPGM function. When set, EEPGM turns on the charge pump and enables the programming (or erasing) power to the EEPROM array. When clear, this power is switched off. This will enable pulsing of the programming voltage to be controlled internally. This bit can be read at any time, but can only be written to if LATCH=1. If LATCH is not set, then EEPGM cannot be set. Reset clears this bit.

3.4.6 PROGRAMMING/ERASING PROCEDURES

To program a byte of EEPROM, set LATCH = CPEN = 1, set ER1 = ER0 = 0, write data to the desired address and then set EEPGM for a time t_{EEPGM}.

In general, all bits should be erased before being programmed. However, if write/erase cycling is a concern, a procedure can be followed to minimize the cycling of each bit in each EEPROM byte. The erased state is 1; therefore, if any bits within the byte need to be changed from a 0 to a 1, the byte must be erased before programming. The decision whether to erase a byte before programming is summarized in Table 3-3.

Table 3-3: EEPROM Write/Erase Cycle Reduction

EEPROM Data To Be Programmed	EEPROM Data Before Programming	Erase Before Programming?
0	0	No
0	1	No
1	0	Yes
1	1	No

To erase a **byte** of EEPROM set LATCH = 1, CPEN = 1, ER1 = 0 and ER0 = 1, write to the address to be erased, and set EEPGM for a time t_{EBYT}.

To erase a **block** of EEPROM set LATCH = 1, CPEN = 1, ER1 = 1 and ER0 = 0, write to any address in the block, and set EEPGM for a time t_{EBLOCK}.

For a **bulk** erase set LATCH = 1, CPEN = 1, ER1 = 1, and ER0 = 1, write to any address in the array, and set EEPGM for a time t_{EBULK}.

To terminate the programming or erase sequence, clear EEPGM, delay for a time t_{FPV} to allow the programming voltage to fall, and then clear LATCH and CPEN to free up the buses. Following each erase or programming sequence, clear all programming control bits.

The following program is an example of the EEPROM programming sequence using the timer to implement the required delay and assuming a 2.1 MHz bus frequency.

```
TSR      EQU     $0013          TIMER STATUS REGISTER
TCNTH    EQU     $0018          TIMER COUNTER REGISTER (HIGH)
TCNTL    EQU     $0019          TIMER COUNTER REGISTER (LOW)
TCMPH    EQU     $0016          TIMER OUTPUT COMPARE REGISTER (HIGH)
TCMPL    EQU     $0017          TIMER OUTPUT COMPARE REGISTER (LOW)
OCF      EQU     6              OCF BIT OF TSR
PROG     EQU     $001C          PROGRAM REGISTER
CPEN     EQU     6              CHARGE PUMP ENABLE
ER1      EQU     4              ERASE SELECT 1
ER0      EQU     3              ERASE SELECT 0
LATCH    EQU     2              LATCH BIT
EERC     EQU     1              RC/OSC SELECTOR
EEPGM    EQU     0              EE PROGRAM BIT
EESTART  EQU     $0240          STARTING ADDRESS OF EEPROM
SUMPIN   EQU     $55            DUMMY DATA

ORG      $0F00
START    EQU     *
         BSET    EERC,PROG      SELECT RC OSC
         BSR     DELAY          RC OSC STABILIZATION
         BSET    CPEN,PROG      TURN ON CHARGE PUMP
         BSET    LATCH,PROG     ENABLE LATCH BIT
```

```
        BCLR    ER1,PROG        ENSURE PROGRAM (NOT ERASE)
        BCLR    ER0,PROG        ENSURE PROGRAM (NOT ERASE)
        LDA     #SUMPIN         GET DATA
        STA     EESTART
        BSET    EEPGM, PROG     ENABLE PROGRAMMING POWER
        JSR     DELAY           WAIT FOR PROGRAM TIME
        BCLR    EEPGM,PROG      CLEAR EEPGM
        JSR     DELAY           WAIT FOR PROG VOLTAGE TO FALL
        BCLR    LATCH,PROG      CLEAR LATCH
        BCLR    CPEN,PROG       DISABLE CHARGE PUMP
        LDA     #SUMPIN
        CMP     EESTART         VERIFY
        BNE     OUT1
        CLC                     CLEAR CARRY BIT FOR NO ERROR
OUT     RTS
OUT1    SEC                     FLAG AN ERROR
        RTS

* THIS ROUTINE GIVES ABOUT A 10 MS DELAY FOR A 2.1Mhz BUS.
* THE SAME DELAY ROUTINE IS USED IN THIS EXAMPLE FOR SIMPLICITY,
* USING THE LONGEST DELAY TIME. USERS WILL WANT TO WRITE SHORTER
* DELAY ROUTINES FOR APPLICATIONS IN WHICH SPEED IS IMPORTANT.

DELAY   EQU     *
        LDA     TCNTH           READ MS BYTE OF TIMER COUNTER
        ADD     #$15            ADD OFFSET
        STA     TCMPH           STORE BACK IN O/P COMPARE MS BYTE
        LDA     TCNTL           READ LS BYTE OF TCR
        STA     TCMPL           STORE INTO O/P COMPARE LS BYTE
        BRCLR   OCF,TSR,*       WAIT FOR OCF FLAG TO OCCUR
        RTS
```

3.5 OPERATION IN STOP AND WAIT

The RC oscillator for the EEPROM is automatically disabled when entering STOP mode. The user may want to ensure that the RC oscillator is disabled before entering WAIT mode to help conserve power.

SECTION 4 CPU CORE

The MC68HC705V8 has a 16K memory map. Therefore it uses only the lower 14 bits of the address bus. In the following discussion the upper 2 bits of the address bus can be ignored. Also, the STOP instruction may be selected to act as either the normal STOP instruction or pull a reset by means of a mask option. All other instructions and registers behave as described in this chapter.

4.1 REGISTERS

The MCU contains five registers that are hard-wired within the CPU and are not part of the memory map. These five registers are shown in Figure 4-1 and are described in the following paragraphs.

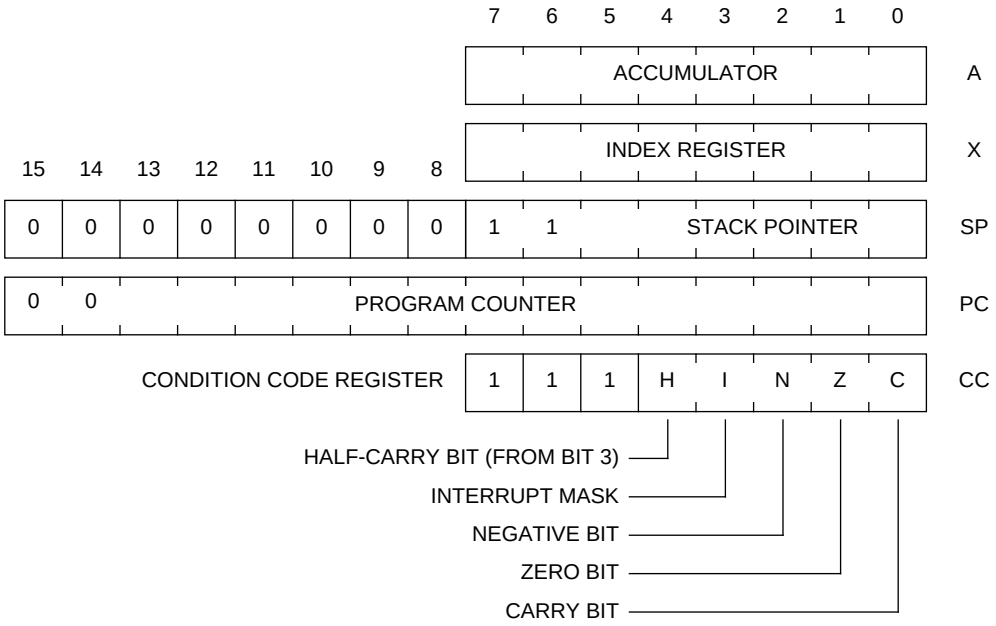


Figure 4-1: MC68HC05 Programming Model

4.1.1 ACCUMULATOR (A)

The accumulator is a general purpose 8-bit register, as shown in Figure 4-1. The CPU uses the accumulator to hold operands and results of arithmetic calculations or nonarithmetic operations. The accumulator is unaffected by a reset of the device.

4.1.2 INDEX REGISTER (X)

The index register shown in Figure 4-1 is an 8-bit register that can perform two functions:

- Indexed addressing
- Temporary storage

In indexed addressing with no offset, the index register contains the low byte of the operand address, and the high byte is assumed to be $00. In indexed addressing with an 8-bit offset, the CPU finds the operand address by adding the index register contents to an 8-bit immediate value. In indexed addressing with a 16-bit offset, the CPU finds the operand address by adding the index register contents to a 16-bit immediate value

The index register can also serve as an auxiliary accumulator for temporary storage. The index register is unaffected by a reset of the device.

4.1.3 STACK POINTER (SP)

The stack pointer shown in Figure 4-1 is a 16-bit register internally. In devices with memory maps less than 64 Kbytes the unimplemented upper address lines are ignored. The stack pointer contains the address of the next free location on the stack. During a reset or the reset stack pointer (RSP) instruction, the stack pointer is set to $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. The six least significant register bits are appended to these ten fixed bits to produce an address within the range of $00FF to $00C0. Subroutines and interrupts may use up to 64 ($40) locations. If 64 locations are exceeded, the stack pointer wraps around and writes over the previously stored information. A subroutine call occupies two locations on the stack and an interrupt uses five locations.

4.1.4 PROGRAM COUNTER (PC)

The program counter shown in Figure 4-1 is a 16-bit register internally. In devices with memory maps less than 64 Kbytes the unimplemented upper address lines are ignored. The program counter contains the address of the next instruction or operand to be fetched.

Normally, the address in the program counter increments to the next sequential memory location every time an instruction or operand is fetched. Jump, branch, and interrupt operations load the program counter with an address other than that of the next sequential location.

4.1.5 CONDITION CODE REGISTER (CCR)

The CCR shown in Figure 4-1 is a 5-bit register in which four bits are used to indicate the results of the instruction just executed. The fifth bit is the interrupt mask. These bits can be individually tested by a program, and specific actions can be taken as a result of their state. The condition code register should be thought of as having three additional upper bits that are always ones. Only the interrupt mask is affected by a reset of the device. The following paragraphs explain the functions of the lower five bits of the condition code register.

4.1.5.1 Half Carry Bit (H-Bit)

When the half-carry bit is set, it means that a carry occurred between bits 3 and 4 of the accumulator during the last ADD or ADC (add with carry) operation. The half-carry bit is required for binary-coded decimal (BCD) arithmetic operations.

4.1.5.2 Interrupt Mask (I-Bit)

When the interrupt mask is set, the internal and external interrupts are disabled. Interrupts are enabled when the interrupt mask is cleared. When an interrupt occurs, the interrupt mask is automatically set after the CPU registers are saved on the stack, but before the interrupt vector is fetched. If an interrupt request occurs while the interrupt mask is set, the interrupt request is latched. Normally, the interrupt is processed as soon as the interrupt mask is cleared.

A return from interrupt (RTI) instruction pulls the CPU registers from the stack, restoring the interrupt mask to its state before the interrupt was encountered. After any reset, the interrupt mask is set and can only be cleared by the Clear I-Bit (CLI), STOP, or WAIT instructions.

4.1.5.3 Negative Bit (N-Bit)

The negative bit is set when the result of the last arithmetic operation, logical operation, or data manipulation was negative. (Bit 7 of the result was a logical one.)

The negative bit can also be used to check an often-tested flag by assigning the flag to bit 7 of a register or memory location. Loading the accumulator with the contents of that register or location then sets or clears the negative bit according to the state of the flag.

4.1.5.4 Zero Bit (Z-Bit)

The zero bit is set when the result of the last arithmetic operation, logical operation, data manipulation, or data load operation was zero.

4.1.5.5 Carry/Borrow Bit (C-Bit)

The carry/borrow bit is set when a carry out of bit 7 of the accumulator occurred during the last arithmetic operation, logical operation, or data manipulation. The carry/borrow bit is also set or cleared during bit test and branch instructions and during shifts and rotates. This bit is not set by an INC or DEC instruction.

SECTION 5 INTERRUPTS

The MCU can be interrupted seven different ways:

1. Nonmaskable Software Interrupt Instruction (SWI)
2. External Asynchronous Interrupt (IRQ)
3. External Interrupt via IRQ on PA0-PA7, PC0-PC7
4. Internal 16-bit Timer Interrupt (TIMER)
5. Internal Serial Peripheral Interface Interrupt (SPI)
6. Internal MDLC Interrupt (MDLC)
7. Internal 8-bit Timer Interrupt

5.1 CPU INTERRUPT PROCESSING

Interrupts cause the processor to save register contents on the stack and to set the interrupt mask (I-bit) to prevent additional interrupts. Unlike RESET, hardware interrupts do not cause the current instruction execution to be halted, but are considered pending until the current instruction is complete.

If interrupts are not masked (I-bit in the CCR is clear) and the corresponding interrupt enable bit is set the processor will proceed with interrupt processing. Otherwise, the next instruction is fetched and executed. If an interrupt occurs the processor completes the current instruction, then stacks the current CPU register states, sets the I-bit to inhibit further interrupts, and finally checks the pending hardware interrupts. If more than one interrupt is pending following the stacking operation, the interrupt with the highest vector location shown in Table 5-1 will be serviced first. The SWI is executed the same as any other instruction, regardless of the I-bit state.

When an interrupt is to be processed the CPU fetches the address of the appropriate interrupt software service routine from the vector table at locations $3FF2 through $3FFF as defined in Table 5-1.

Table 5-1. Vector Address for Interrupts and Reset

Register	Flag Name	Interrupts	CPU Interrupt	Vector Address
N/A	N/A	Reset	RESET	$3FFE-$3FFF
N/A	N/A	Software	SWI	$3FFC-$3FFD
N/A	N/A	External Interrupts **	IRQ	$3FFA-$3FFB
TSR	OCF,ICF,TOF	16-bit Timer Interrupts	TIMER	$3FF8-$3FF9
MSR	TXMS,RXMS	MDLC Interrupt	MDLC	$3FF6-$3FF7
SPSR	SPIF	SPI Interrupt	SPI	$3FF4-$3FF5
CTCSR	TOFE,RTIE	8 bit Timer Interrupts	TIMER,RTI	$3FF2-$3FF3

** External interrupts include IRQ, PORTA and PORTC sources.

The M68HC05 CPU does not support interruptible instructions, therefore, the maximum latency to the first instruction of the interrupt service routine must include the longest instruction execution time plus stacking overhead.

$$\text{Latency} = (\text{Longest instruction execution time} + 10) \times t_{CYC} \text{ secs}$$

An RTI instruction is used to signify when the interrupt software service routine is completed. The RTI instruction causes the register contents to be recovered from the stack and normal processing to resume at the next instruction that was to be executed when the interrupt took place. Figure 5-1 shows the sequence of events that occur during interrupt processing.

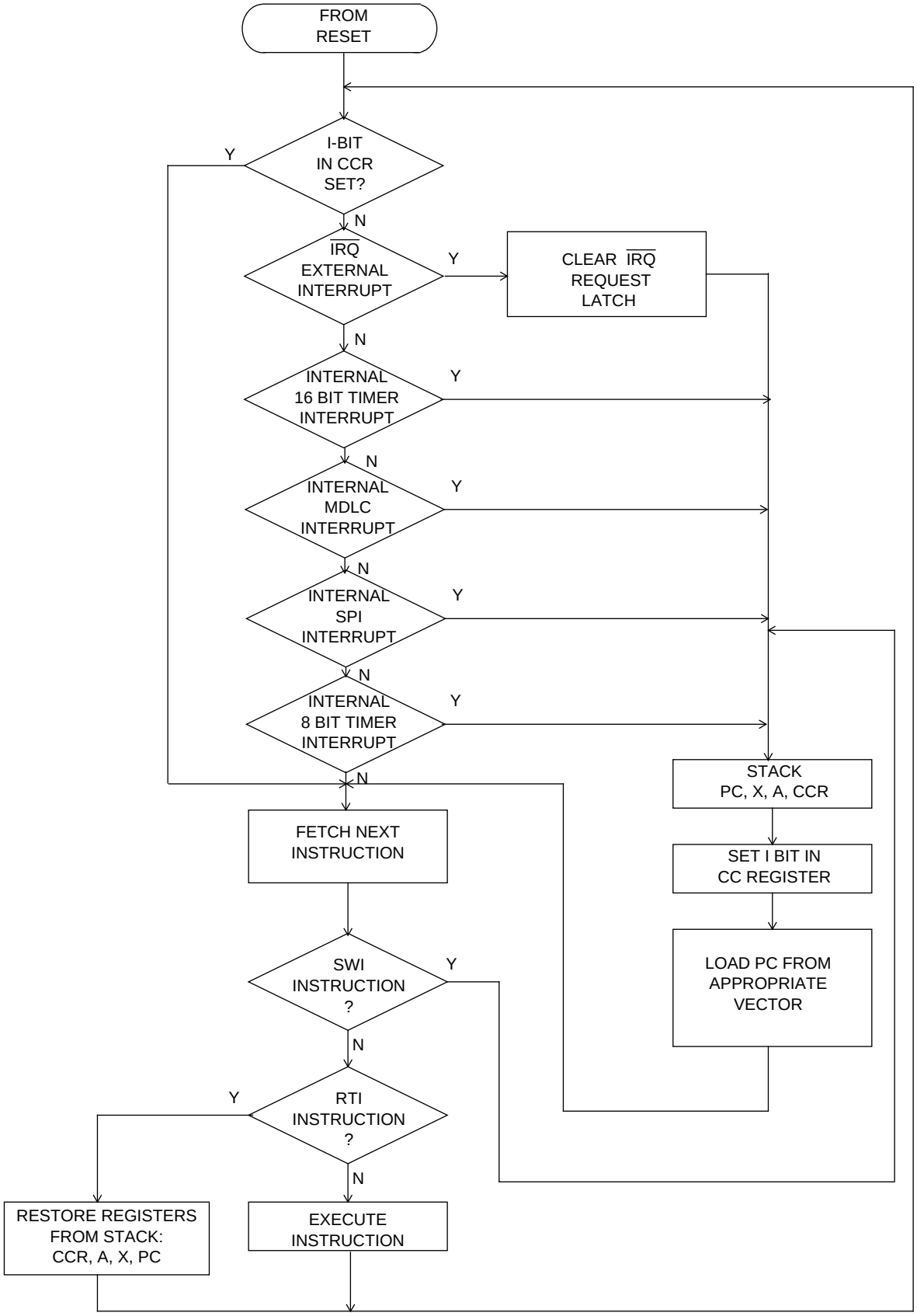


Figure 5-1: Interrupt Processing Flowchart

5.2 RESET INTERRUPT SEQUENCE

The RESET function is not in the strictest sense an interrupt; however, it is acted upon in a similar manner, as shown in Figure 5-1. A low level input on the $\overline{\text{RESET}}$ pin or internally generated RST signal causes the program to vector to its starting address which is specified by the contents of memory locations $3FFE and $3FFF. The I-bit in the condition code register is also set. The MCU is configured to a known state during this type of reset as described in **SECTION 6 RESETS**.

5.3 SOFTWARE INTERRUPT (SWI)

The SWI is an executable instruction and a non-maskable interrupt since it is executed regardless of the state of the I-bit in the CCR. If the I-bit is zero (interrupts enabled), the SWI instruction executes after interrupts that were pending before the SWI was fetched, or before interrupts generated after the SWI was fetched. The interrupt service routine address is specified by the contents of memory locations $3FFC and $3FFD.

5.4 HARDWARE INTERRUPTS

All hardware interrupts except RESET are maskable by the I-bit in the CCR. If the I-bit is set, all hardware interrupts (internal and external) are disabled. Clearing the I-bit enables the hardware interrupts. There are two types of hardware interrupts that are explained in the following sections.

5.5 EXTERNAL INTERRUPT (IRQ)

The IRQ pin provides an asynchronous interrupt to the CPU. A block diagram of the IRQ function is shown in Figure 5-2.

NOTE: The BIH and BIL instructions will only apply to the level on the $\overline{\text{IRQ}}$ pin itself, and not to the output of the logic OR function with the Port A and Port C IRQ interrupts. The state of the individual Port A and Port C pins can be checked by reading the appropriate Port A and Port C pins as inputs.

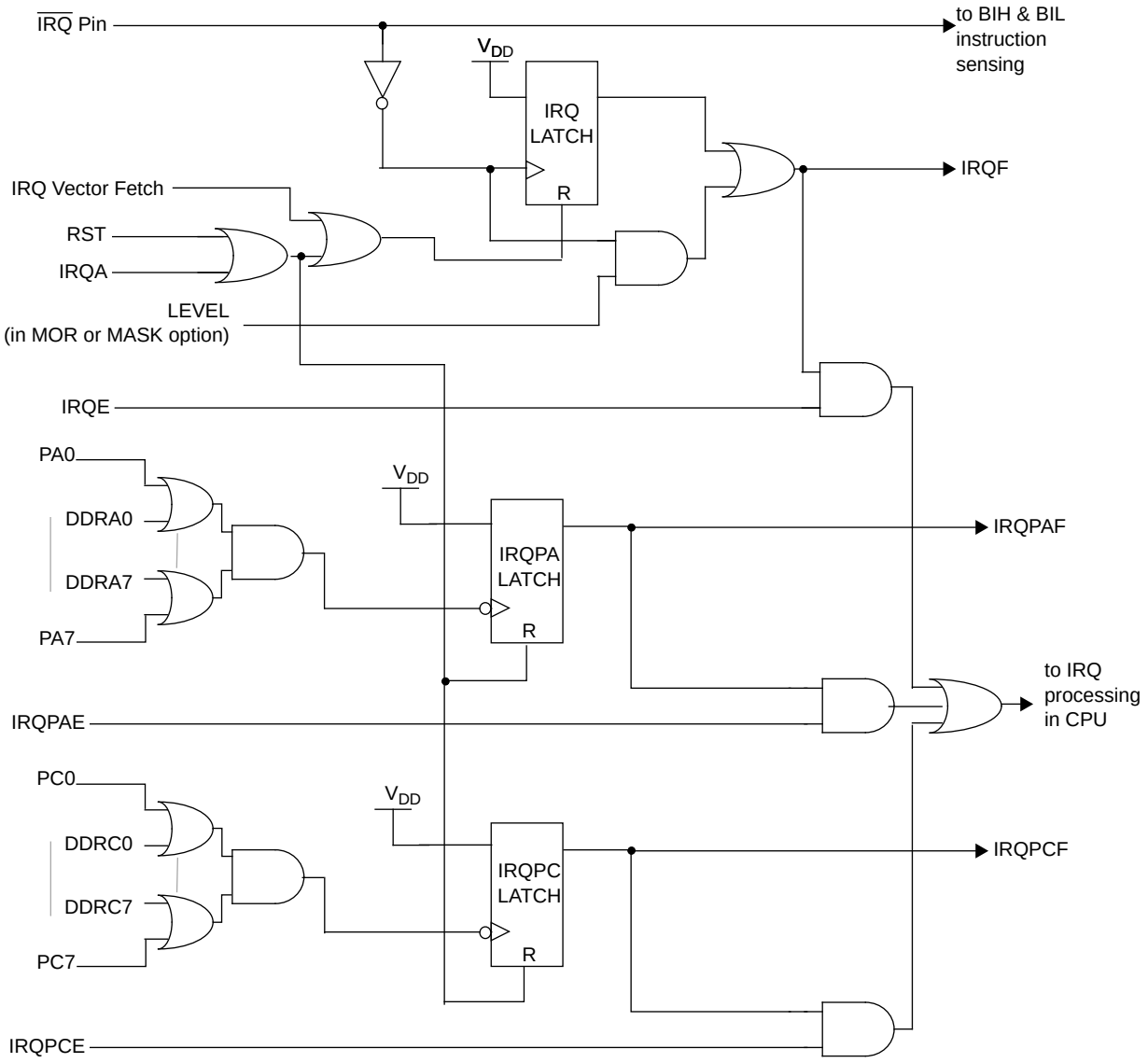


Figure 5-2: IRQ Function Block Diagram

The $\overline{\text{IRQ}}$ pin is one source of an external interrupt. All Port A pins (PA0 through PA7) and/or all Port C pins (PC0 through PC7) act as other external interrupt sources. These sources each have their own interrupt latch but are all combined into a single external interrupt request.

The Port A and Port C interrupt sources are **negative (falling) edge sensitive only**. Note that all Port A pins and all Port C pins are ANDed together, if configured as an input, to form the negative edge signal which sets the corresponding flag bits. A high to low transition on any Port A or Port C pin configured as an input will therefore set the respective flag bit. If a Port A or Port C pin is an input and is not used as a switch input in the system, it must remain high to prevent spurious interrupts from occurring. Refer to **9.1.3 PORT A/C I/O PIN INTERRUPTS** for more detail on Port A/Port C interrupts. The $\overline{\text{IRQ}}$ pin interrupt source may be selected to be either edge sensitive or edge and level sensitive through a mask option

or an MOR bit. If the EDGE sensitive interrupt option is selected for the IRQ pin, only the IRQ latch output can activate an IRQF flag which creates an interrupt request to the CPU to generate the external interrupt sequence.

When edge sensitivity is selected for the IRQ interrupt, it is sensitive to the following cases:

- Falling edge on the $\overline{\text{IRQ}}$ pin.
- Falling edge on any Port A or Port C pin with IRQ enabled.

If the LEVEL select bit in the MOR is set, the active low state of the $\overline{\text{IRQ}}$ pin can also activate an IRQF flag which creates an IRQ request to the CPU to generate the IRQ interrupt sequence.

When edge and level sensitivity is selected for the IRQ interrupt, it is sensitive to the following cases:

- Low level on the $\overline{\text{IRQ}}$ pin.
- Falling edge on the $\overline{\text{IRQ}}$ pin.
- Falling edge on any Port A or Port C pin with IRQ enabled.

The IRQE enable bit controls whether an active IRQF flag ($\overline{\text{IRQ}}$ pin interrupt) can generate an IRQ interrupt sequence. The IRQPAE enable bit controls whether an active IRQPAF flag (Port A interrupt) can generate an IRQ interrupt sequence. The IRQPCE enable bit controls whether an active IRQPCF flag (Port C interrupt) can generate an IRQ interrupt sequence. The IRQ interrupt is serviced by the interrupt service routine located at the address specified by the contents of $3FFA and $3FFB.

The IRQF latch is automatically cleared by entering the interrupt service routine to maintain compatibility with existing M6805 interrupt servicing protocol. To allow the user to identify the source of the interrupt, the port interrupt flags (IRQPAF and IRQPCF) are not cleared automatically. These flags must be cleared within the interrupt handler prior to exit in order to prevent repeated re-entry. This is achieved by writing a logic one to the IRQA (IRQ acknowledge) bit, which will clear all pending IRQ interrupts (including a pending $\overline{\text{IRQ}}$ pin interrupt).

The interrupt request flags (IRQPAF and IRQPCF) are read only and cannot be cleared by writing to them. The acknowledge flag always reads as a logic 0. Together, these features permit the safe use of read-modify-write instructions (for example, BSET, BCLR) on the ICSR.

NOTE: Although read modify write instruction use is allowable on the ICSR, shift operations should be avoided due to the possibility of inadvertently setting the IRQA.

5.5.1 IRQ CONTROL/STATUS REGISTER (ICSR) $1F

The IRQ interrupt function is controlled by the ICSR located at $001F. All unused bits in the ICSR will read as logic zeros. The IRQF bit is cleared and IRQE bit is set by reset.

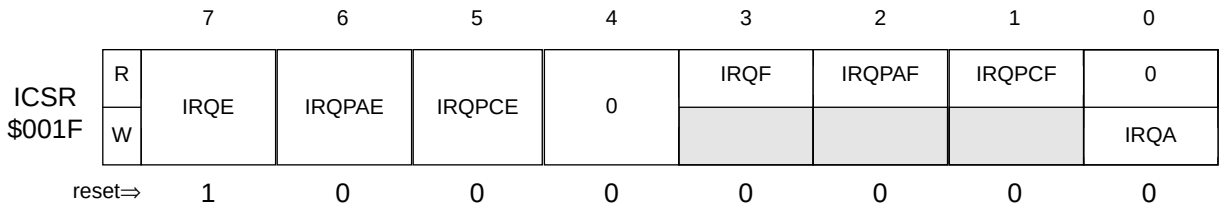


Figure 5-3: IRQ Status & Control Register

5.5.1.1 IRQA - IRQ Interrupt Acknowledge

The IRQA acknowledge bit clears an IRQ interrupt by clearing the IRQ, IRQPA and IRQPC latches. This is achieved by writing a logic one to the IRQA acknowledge bit. Writing a logic zero to the IRQA acknowledge bit will have no effect on the any of the IRQ latches. If either the IRQ, IRQPA or IRQPC latch is not cleared within the IRQ service routine the CPU will re-enter the IRQ interrupt sequence continuously until the IRQ latches are all cleared. The IRQA is useful for cancelling unwanted or spurious interrupts which may have occurred while servicing the initial IRQ interrupt. The IRQA acknowledge bit will always read as a logic zero. The IRQA function is activated by reset.

NOTE: The IRQ latch is cleared automatically during the IRQ vector fetch. The IRQPA and IRQPC latches are not cleared automatically (to permit interrupt source differentiation) and must be cleared from within the IRQ service routine.

5.5.1.2 IRQPCF - Port C IRQ Interrupt Request

The IRQPCF flag bit indicates that a Port C IRQ request is pending. Writing to the IRQPCF flag bit will have no effect on it. The IRQPCF flag bit must be cleared by writing a logic one to the IRQA acknowledge bit. In this way an additional IRQPCF flag bit that is set while in the service routine can be ignored by clearing the IRQPCF flag bit before exiting the service routine. If the additional IRQPCF flag bit is not cleared in the IRQ service routine and the IRQPCE enable bit remains set, the CPU will re-enter the IRQ interrupt sequence continuously until either the IRQPCF flag bit or the IRQPCE enable bit is clear. This bit is operational regardless of the state of the IRQPCE bit. The IRQPCF bit is cleared by reset.

5.5.1.3 IRQPAF - Port A IRQ Interrupt Request

The IRQPAF flag bit indicates that a Port A IRQ request is pending. Writing to the IRQPAF flag bit will have no effect on it. The IRQPAF flag bit must be cleared by writing a logic one to the IRQA acknowledge bit. In this way an additional IRQPAF flag bit that is set while in the service routine can be ignored by clearing the IRQPAF flag bit before exiting the service routine. If the additional IRQPAF flag bit is not cleared in the IRQ service routine and the IRQPAE enable bit remains set, the CPU will re-enter the IRQ interrupt sequence continuously until either the IRQPAF flag bit or the IRQPAE enable bit is clear. This bit is operational regardless of the state of the IRQPAE bit. The IRQPAF bit is cleared by reset.

5.5.1.4 IRQF - IRQ Interrupt Request

The IRQF flag bit indicates that an IRQ request is pending. Writing to the IRQF flag bit will have no effect on it. The IRQF flag bit is cleared when the IRQ vector is fetched prior to the service routine being entered. The IRQF flag bit can also be cleared by writing a logic one to the IRQA acknowledge bit to clear the IRQ latch. In this way any additional IRQF flag bit that is set while in the service routine can be ignored by clearing the IRQF flag bit before exiting the service routine. If the additional IRQF flag bit is not cleared in the IRQ service routine and the IRQE enable bit remains set, the CPU will re-enter the IRQ interrupt sequence continuously until either the IRQF flag bit or the IRQE enable bit is clear. This flag can be set only when the IRQE enable is set. The IRQ latch is cleared by reset.

5.5.1.5 IRQPCE - Port C IRQ Interrupt Enable

The IRQPCE bit controls whether the IRQPCF flag bit can or cannot initiate an IRQ interrupt sequence. If the IRQPCE enable bit is set the IRQPCF flag bit can generate an interrupt sequence. If the IRQPCE enable bit is cleared the IRQPCF flag bit cannot generate an interrupt sequence. Reset clears the IRQPCE enable bit, thereby disabling Port C IRQ interrupts. In addition, reset also sets the I-bit, which masks all interrupt sources. Execution of the STOP or WAIT instructions does not effect the IRQPCE bit.

5.5.1.6 IRQPAE - Port A IRQ Interrupt Enable

The IRQPAE bit controls whether the IRQPAF flag bit can or cannot initiate an IRQ interrupt sequence. If the IRQPAE enable bit is set the IRQPAF flag bit can generate an interrupt sequence. If the IRQPAE enable bit is cleared the IRQPAF flag bit cannot generate an interrupt sequence. Reset clears the IRQPAE enable bit, thereby disabling Port A IRQ interrupts. In addition, reset also sets the I-bit, which masks all interrupt sources. Execution of the STOP or WAIT instructions does not effect the IRQPAE bit.

NOTE: The IRQPAE and IRQPCE mask bits must be set prior to entering STOP or WAIT modes if Port IRQ interrupts are to be enabled.

5.5.1.7 IRQE - IRQ Interrupt Enable

The IRQE bit controls whether the IRQF flag bit can or cannot initiate an IRQ interrupt sequence. If the IRQE enable bit is set the IRQF flag bit can generate an interrupt sequence. If the IRQE enable bit is cleared the IRQF flag bit cannot generate an interrupt sequence. Reset sets the IRQE enable bit, thereby enabling IRQ interrupts once the I-bit is cleared. Execution of the STOP or WAIT instructions causes the IRQE bit to be set in order to allow the external IRQ to exit these modes. In addition, reset also sets the I-bit, which masks all interrupt sources.

5.5.2 EXTERNAL INTERRUPT TIMING

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. The interrupt request is latched immediately following the falling edge of the $\overline{\text{IRQ}}$ source. It is then synchronized internally

and serviced as specified by the contents of $3FFA and $3FFB.The IRQ timing diagram is shown in Figure 5-4.

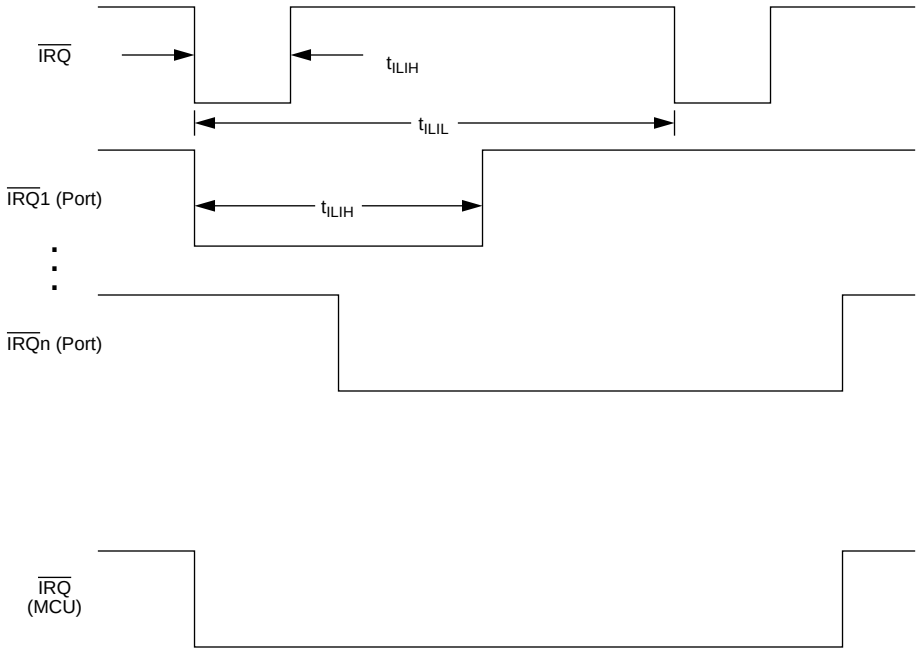


Figure 5-4: External Interrupts Timing Diagram

Either a level-sensitive and edge-sensitive trigger, or an edge-sensitive-only trigger is available as a mask option for the IRQ pin only.

5.6 16-BIT TIMER INTERRUPT

There are three different timer interrupt flags that cause a timer interrupt whenever they are set and enabled. The interrupt flags are in the Timer Status Register (TSR), and the enable bits are in the Timer Control Register (TCR). Any of these interrupts will vector to the same interrupt service routine, located at the address specified by the contents of memory location $3FF8 and $3FF9.

5.7 MDLC INTERRUPT

The MDLC Transmit Message Successful and Receive Message Successful interrupts are respectively generated by the TXMS and RXMS bits of the MDLC Status Register (MSR). In addition a wake-up from STOP or WAIT mode can also generate an interrupt from the MDLC. The MDLC Interrupt Enable (IE) bit is located in the MDLC Control Register (MCR). Provided the IE bit is set (and the CCR I bit is clear), all interrupts will vector to the same interrupt service routine, located at the address specified by the contents of memory location $3FF6 and $3FF7.

5.8 SPI INTERRUPT

There are two different SPI interrupt flags that cause an SPI interrupt whenever they are set and enabled. The interrupt flags are in the SPI Status Register (SPSR), and the enable bits are in the SPI Control Register (SPCR). Either of these interrupts will vector to the

same interrupt service routine, located at the address specified by the contents of memory location $3FF4 and $3FF5.

5.9 8-BIT TIMER INTERRUPT

This timer can create two types of interrupts. A timer overflow interrupt will occur whenever the 8 bit timer rolls over from $FF to $00 and the enable bit TOFE is set. A real time interrupt will occur whenever the programmed time elapses and the enable bit RTIE is set. See SECTION 12 CORE TIMER. This interrupt will vector to the interrupt service routine located at the address specified by the contents of memory location $3FF2 and $3FF3.

SECTION 6 RESETS

The MCU can be reset from six sources: one external input and five internal restart conditions. The RESET pin is an input with a Schmitt trigger as shown in Figure 6-1. All the internal peripheral modules will be reset by the internal reset signal (RST). Refer to Figure 6-2 for reset timing detail.

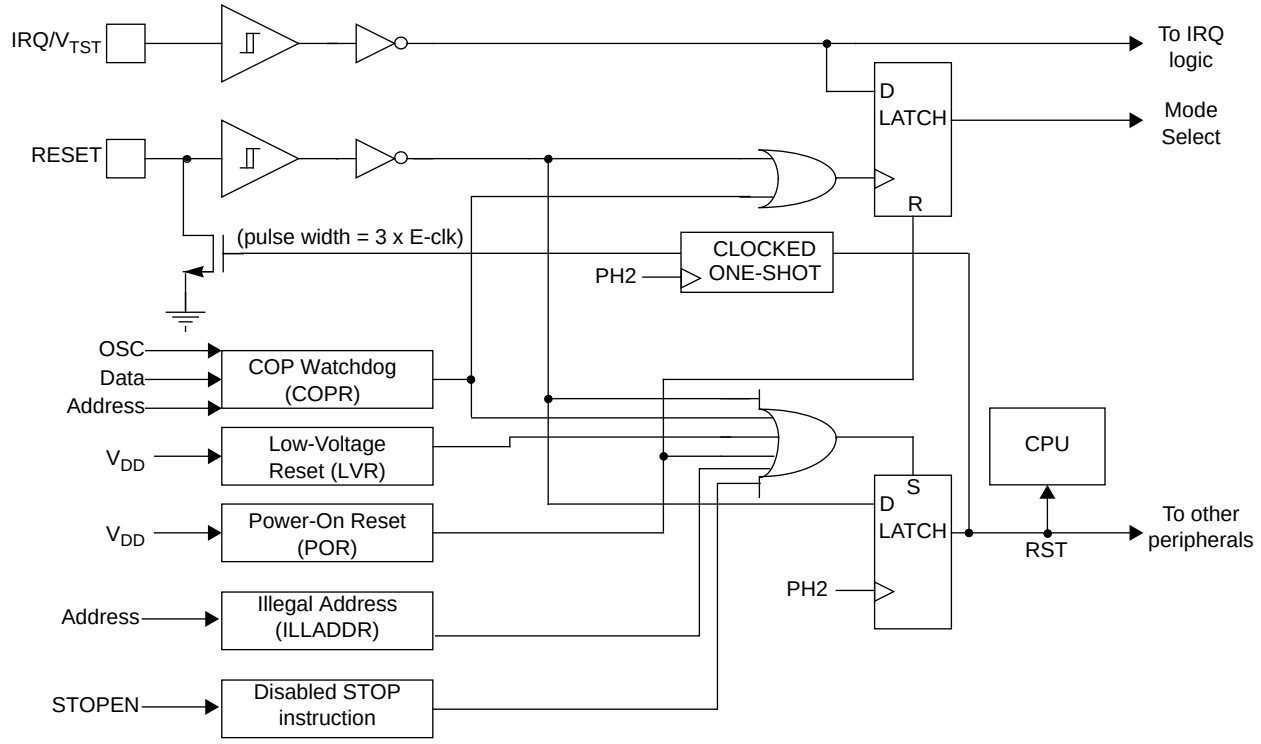


Figure 6-1: Reset Block Diagram

6.1 EXTERNAL RESET ($\overline{\text{RESET}}$)

The RESET pin is the only external source of a reset. This pin is connected to a Schmitt trigger input gate to provide an upper and lower threshold voltage separated by a minimum amount of hysteresis. This external reset occurs whenever the RESET pin is pulled below the lower threshold and remains in reset until the RESET pin rises above the upper threshold. This active low input will generate the RST signal and reset the CPU and peripherals. Termination of the external RESET input or the internal COP Watchdog reset are the only reset sources that can alter the operating mode of the MCU.

NOTE: Activation of the RST signal is generally referred to as **reset** of the device, unless otherwise specified.

The $\overline{\text{RESET}}$ pin can also act as an open drain output. It will be pulled to a low state by an internal pulldown that is activated by any reset source. This RESET pulldown device will only be asserted for 3-4 cycles of the internal clock, PH2 (PH2 period = E clock period) or as long as an internal reset source is asserted. When the external $\overline{\text{RESET}}$ pin is asserted, the pulldown device will be turned on for only the 3-4 internal clock cycles.

6.2 INTERNAL RESETS

The five internally generated resets are the initial power-on reset function, the COP Watchdog Timer reset, the illegal address detector, the low-voltage reset, and the disabled STOP instruction. Termination of the external RESET input or the internal COP Watchdog Timer are the only reset sources that can alter the operating mode of the MCU. The other internal resets will not have any effect on the mode of operation when their reset state ends. All internal resets will also assert (pull to logic zero) the external $\overline{\text{RESET}}$ pin for the duration of the reset or 3-4 internal clock cycles, whichever is longer.

6.2.1 POWER-ON RESET (POR)

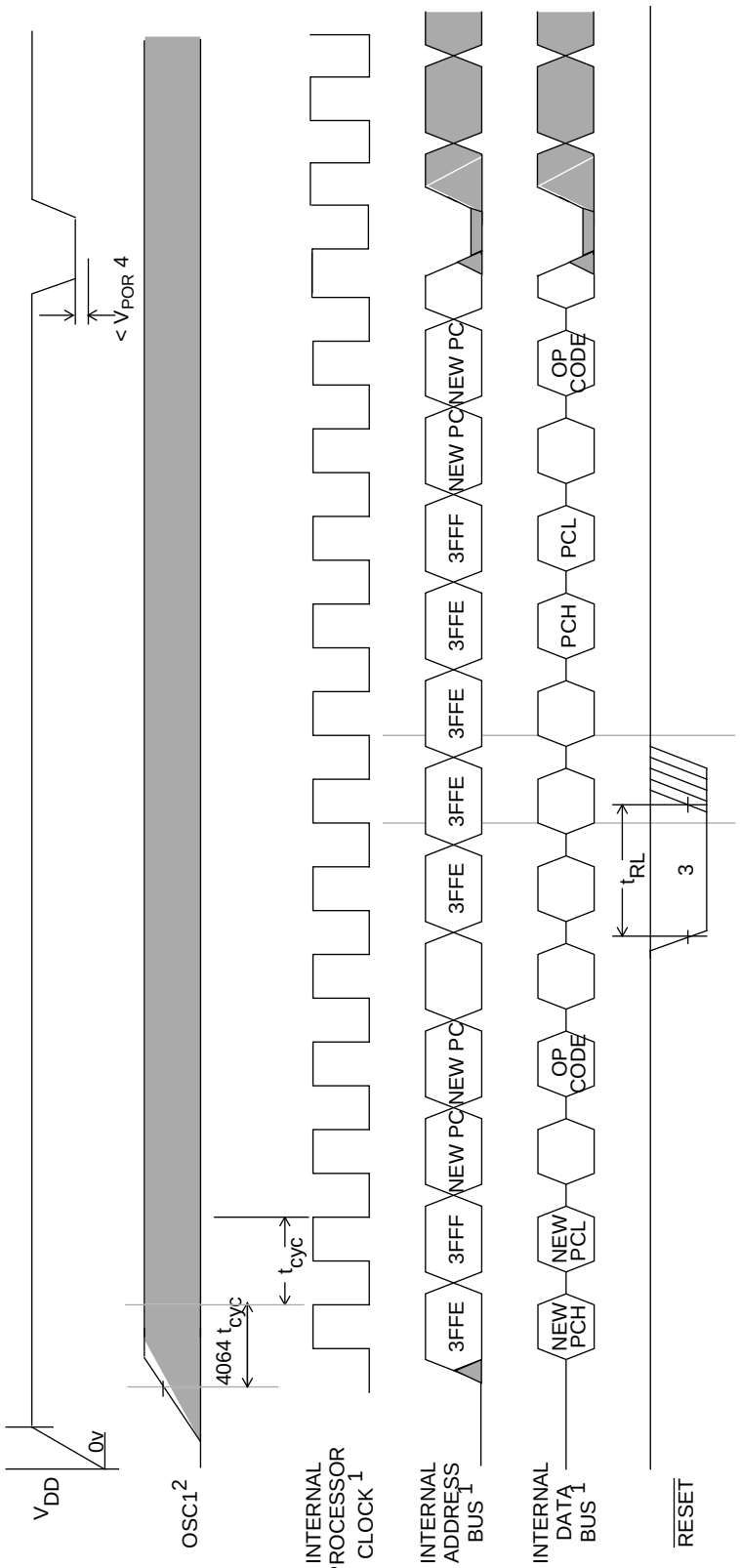
The internal POR is generated on power-up to allow the clock oscillator to stabilize. The POR is strictly for power turn-on conditions and is not able to detect a drop in the power supply voltage (brown-out). There is an oscillator stabilization delay of 4064 internal processor bus clock cycles (PH2) after the oscillator becomes active.

The POR will generate the RST signal which will reset the CPU. If any other reset function is active at the end of this 4064 cycle delay, the RST signal will remain in the reset condition until the other reset condition(s) end.

POR will activate the $\overline{\text{RESET}}$ pin pulldown device connected to the pin. V_{DD} must drop below V_{POR} in order for the internal POR circuit to detect the next rise of V_{DD}.

6.2.2 OPERATION IN STOP AND WAIT

If enabled, the LVR supply voltage sense option is active during STOP and WAIT. Any reset source can bring the MCU out of STOP or WAIT modes.



NOTES:
1. Internal timing signal and bus information not available externally.
2. OSC1 line is not meant to represent frequency. It is only used to represent time.
3. The next rising edge of the internal processor clock following the rising edge of $\overline{\text{RESET}}$ initiates the reset sequence.
4. V_{DD} must fall to a level lower than V_{POR} in order to recognized as a power on reset.

Figure 6-2: RESET and POR Timing Diagram

6.2.3 COMPUTER OPERATING PROPERLY RESET (COPR)

The MCU contains a watchdog timer that automatically times out if not reset (cleared) within a specific time by a program reset sequence. If the COP watchdog timer is allowed to time-out, an internal reset is generated to reset the MCU. Regardless of an internal or external RESET, the MCU comes out of a COP reset according to the standard rules of mode selection.

The COP reset function is enabled or disabled by the mask option bit and is verified during production testing.

The COP Watchdog reset will activate the internal pulldown device connected to the RESET pin.

6.2.3.1 RESETTING THE COP

Preventing a COP reset is done by writing a “0” to the COPF bit. This action will reset the counter and begin the time-out period again. The COPF bit is bit 0 of address $3FF0. A read of address $3FF0 will return user data programmed at that location.

6.2.3.2 COP DURING WAIT MODE

The COP will continue to operate normally during WAIT mode. The software should pull the device out of WAIT mode periodically and reset the COP by writing to the COPF bit to prevent a COP reset.

6.2.3.3 COP DURING STOP MODE

When the STOP enable mask option is selected, STOP mode disables the oscillator circuit and thereby turns the clock off for the entire device. The COP counter will be reset when STOP mode is entered. If a reset is used to exit STOP mode, the COP counter will be held in reset during the 4064 cycles of start up delay. If any operable interrupt is used to exit STOP mode, the COP counter will not be reset during the 4064 cycle start up delay and will have that many cycles already counted when control is returned to the program.

6.2.3.4 COP WATCHDOG TIMER CONSIDERATIONS

The COP Watchdog Timer is active in all modes of operation if enabled by a mask option. If the COP Watchdog Timer is selected by a mask option, any execution of the STOP instruction (either intentional or inadvertent due to the CPU being disturbed) will cause the oscillator to halt and prevent the COP Watchdog Timer from timing out. Therefore, it is recommended that the STOP instruction should be **disabled** if the COP Watchdog Timer is enabled.

If the COP Watchdog Timer is selected by a mask option, the COP will reset the MCU when it times out. Therefore, it is recommended that the COP Watchdog should be **disabled** for a system that must have intentional uses of the WAIT Mode for periods longer than the COP time-out period.

The recommended interactions and considerations for the COP Watchdog Timer, STOP instruction, and WAIT instruction are summarized in Table 6-1.

Table 6-1: COP Watchdog Timer Recommendations

IF the following conditions exist:		THEN the COP Watchdog Timer should be as follows:
STOP Instruction	WAIT Time	
converted to reset	WAIT Time **Less than** COP Time-Out	Enable or disable COP by mask option
converted to reset	WAIT Time **More than** COP Time-Out	Disable COP by mask option
Acts as STOP	any length WAIT Time	Disable COP by mask option

6.2.3.5 COP REGISTER

The COP register is shared with the MSB of an unimplemented User Interrupt Vector as shown in Figure 6-3. Reading this location will return whatever user data has been programmed at this location. Writing a “0” to the COPR bit in this location will clear the COP watchdog timer.

ADDR	REGISTER	READ / WRITE	7	6	5	4	3	2	1	0
$3FF0	UNIMPLEMENTED VECTOR & COP WATCHDOG TIMER	R	x	x	x	x	x	x	x	x
		W								COPR

UNIMPLEMENTED

Figure 6-3: COP Watchdog Timer Location

6.2.4 LOW-VOLTAGE RESET (LVR)

The internal low voltage (LVR) reset is generated when V_{DD} falls below the LVR threshold V_{LVRI} and will be released following a POR delay starting when V_{DD} rises above V_{LVRR}. The LVR threshold is tested to be above the minimum operating voltage of the microcontroller and is intended to assure that the CPU will be held in reset when the V_{BATT} supply voltage is below reasonable operating limits. However, it should be noted that the LVR monitors V_{DD} not V_{BATT}. A mask option is provided to disable the LVR when the device is expected to normally operate at low voltages. Note that the V_{DD} rise and fall slew rates (S_{VDDR} and S_{VDDF}) must be within the specification for proper LVR operation. If the specification is not met, the circuit will operate properly following a delay of V_{DD}/Slew rate.

The LVR will generate the RST signal which will reset the CPU and other peripherals. As it is not possible to determine the level of the internal V_{DD} at the point V_{BATT} recovers (and POR is not intended to detect a loss of V_{DD}), an LVR will always recover using a POR delay. The low voltage reset will activate the internal pulldown device connected to the RESET pin.

If any other reset function is active at the end of the LVR reset signal, the RST signal will remain in the reset condition until the other reset condition(s) end.

6.2.5 ILLEGAL ADDRESS

An illegal address reset is generated when the CPU attempts to fetch an instruction from either unimplemented address space ($02C0 to $0CFF) or I/O address space ($0000 to $003F).

The illegal address reset will activate the internal pulldown device connected to the $\overline{\text{RESET}}$ pin.

6.2.6 DISABLED STOP INSTRUCTION

When the mask option is selected to disable the STOP instruction, execution of a STOP instruction results in an internal reset. This activates the internal pulldown device connected to the $\overline{\text{RESET}}$ pin.

SECTION 7 POWER SUPPLY AND REGULATION

The MC68HC705V8 contains a low-power CMOS on-chip fixed voltage regulator to provide internal power to the MCU from an external DC source. The MCU features separate analog and digital power and ground pins to help decrease common-mode noise contamination of the analog subsystems. The supply pins are also grouped in adjacent pairs to permit optimal decoupling and to minimize radiated RF emissions.

7.1 INTERNAL POWER SUPPLY

The on-chip voltage regulation and power supply control circuitry is comprised of four elements: the primary regulator, the secondary regulator, the power mode control, and the LVR circuitry. The primary internal regulator can be disabled through register control bits. When it is desired to use an external voltage regulator, regulated power must be provided to both sets of V_{DD}/V_{SS} pins. When the primary regulator is disabled, the regulator input pin (V_{BATT}) is still used to power the MDLC's physical interface.

7.1.1 PRIMARY 5V REGULATOR

The primary 5V regulator accepts a regulated input supply and provides a regulated 5V supply to all the digital sections of the device with the exception of the V_{IGN} and power supply control logic. The output of this regulator is also connected to the V_{DD} pins to allow for decoupling of the digital supply pins (V_{DD}), to supply a decoupled and/or filtered supply to the analog supply pin (V_{CCA}) and to provide an external power source for off chip usage. The regulator requires an external 10 uF capacitor for stability.

See **Figure 7-6: MC68HC705V8 On-chip Power Supply Configuration** for the most suitable supply pin and external capacitor connections.

The primary regulator can be disabled through software by clearing the PDC bit of the MISCELLANEOUS register (see Figure 7-1). Once disabled, the regulator can only be enabled by a rising edge on the V_{IGN} or BUS (if mask option enabled) pins. When disabled, the primary regulator output will be connected to V_{SS} if the V_{DDC} mask option has been selected. This prevents unintentional device operation using current sourced through external pullup components.

Any loss of V_{BATT} sufficient to trigger an LVR will cause the device to be reset. The device will remain in the reset state for the duration of the LVR condition or until the internal V_{DD} drops below the functional level of the device, at which point reset no longer has meaning. If the drop in V_{BATT} that triggers an LVR is transient, the internal RST will be asserted for a minimum 4064 cycles of the CPU bus clock, PH2 (the POR delay).

7.1.2 SECONDARY REGULATOR

The secondary regulator provides an independent supply to the small amount of power supply control logic required to provide some of the power moding functions. The

secondary regulator is always enabled (regardless of the mask option selection) but consumes significantly less power than the primary regulator. The secondary regulator will provide sufficient voltage to the control logic provided V_{BATT} remains above the minimum voltage.

7.2 MISCELLANEOUS REGISTER

The software power supply management functions are performed through the MISCELLANEOUS register located at $002F. Although the OCE bit of the MISCELLANEOUS register is unrelated to the power supply, it will be described here.

ADDR	REGISTER	READ WRITE	7	6	5	4	3	2	1	0
$002F	MISCELLANEOUS REGISTER	R	IGNS	OCE	0	0	0	0	0	PDC
		W								
		reset⇒	-	0	-	-	-	-	-	-

UNIMPLEMENTED

Figure 7-1: Miscellaneous Register

7.2.1 IGNS - Ignition status bit

This is a read only status bit which indicates the state of the V_{IGN} pin.

0 - The V_{IGN} pin is less than the V_{IL} level for this pin.

1 - The V_{IGN} pin is greater than the V_{IH} level for this pin.

WARNING: There is not hysteresis or hardware debounce on the V_{IGN} pin and hence in the state of this bit. This bit cannot be read unless the REGEN bit in the MOR is set.

See **17.3 DC ELECTRICAL CHARACTERISTICS** for V_{IH} and V_{IL} levels.

7.2.2 OCE - Output compare enable

This bit controls the function of the PB6 pin.

0 - PB6 functions as a normal I/O pin.

1 - PB6 becomes the TCMP output pin for the 16-bit timer.

See **SECTION 11 16-BIT TIMER** for a description of the TCMP function.

Reset clears this bit.

7.2.3 PDC - Power Down Control

This bit controls the internal primary voltage regulator.

0 - Disables the internal primary voltage regulator.

1 - Enables the internal primary voltage regulator.

This bit has no affect unless the REGEN bit in the MOR is set. This bit is not affected by reset and must be initialized by software.

7.3 POWER MODING

A flow diagram depicting all conditions that the MCU could enter is shown in Figure 7-2. The chip can be operated in one of three distinct power modes. The first mode is "Power off". In this mode, the chip is not being powered from any source. The V_{BATT} pin voltage is below that required by the regulator circuit to provide an output voltage that is within regulation limits.

The second mode is "Standby power". In this mode, the V_{BATT} voltage is above the required minimum, but the user has not turned the primary regulator on. The secondary regulator and the power mode logic are powered up. Standby mode can be exited by either reducing the V_{BATT} voltage or by turning the primary regulator on.

The third mode is "Power up" mode. In this mode the CPU and peripherals can perform their normal processing tasks. The only way to exit from this mode is to lose regulator output power. Losing regulator output power can happen in one of three ways: 1) the V_{BATT} voltage can fall below the required minimum level causing V_{DD} to fall below minimum, 2) software can turn the primary regulator off, and 3) some sort of fault on the external V_{DD} pin. In any case, the LVR reset will be asserted to protect the system as the V_{DD} voltage is falling.

Transitioning between these states is referred to as power moding.

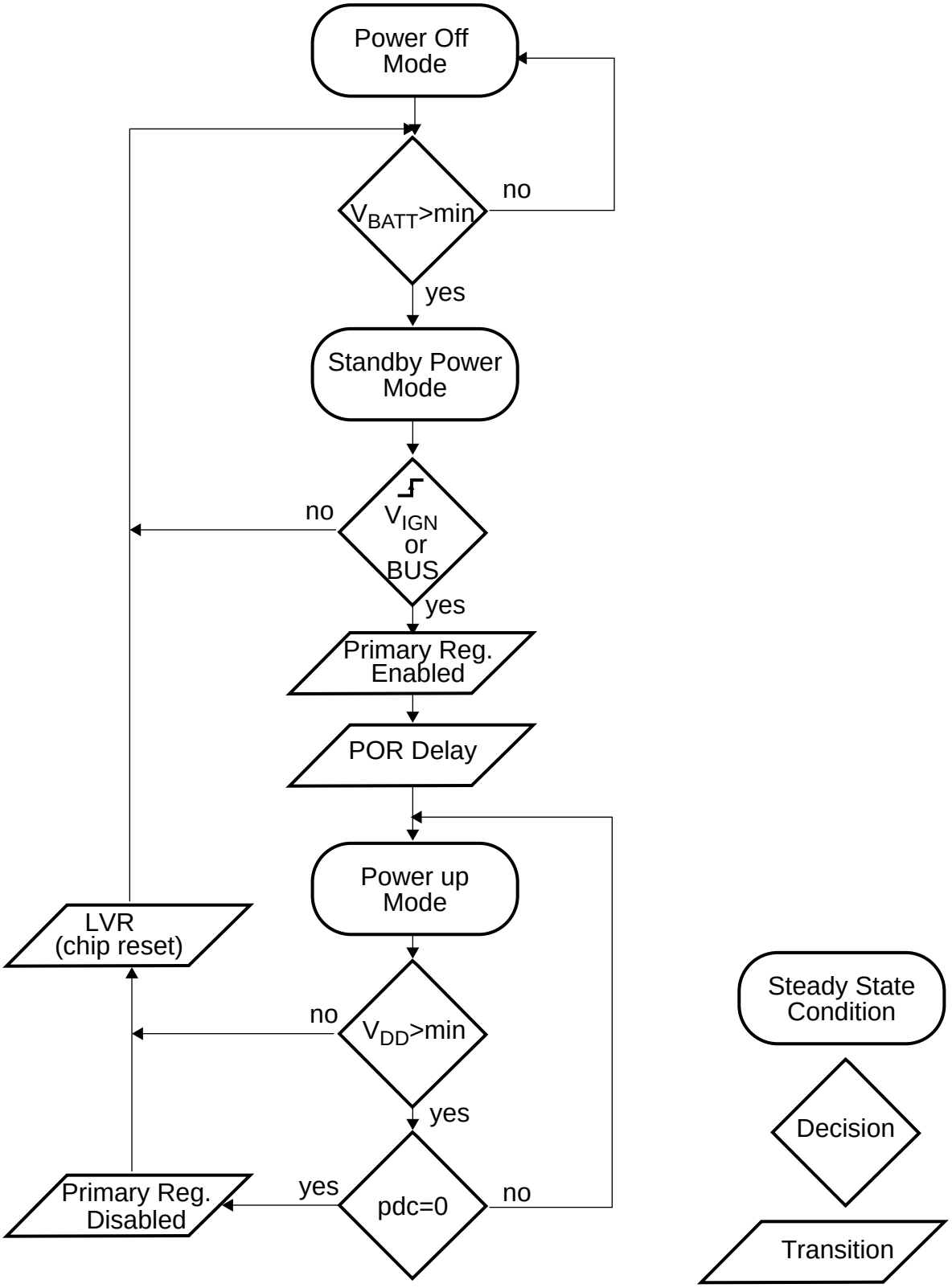


Figure 7-2: MC68HC705V8 Power Moding Flow Diagram

7.4 REGULATOR CONTROL LOGIC

The ignition state monitor pin, V_{IGN}, or the MDLC BUS pin (provided the MDLCPU mask option is selected) allows the user to wake-up the primary power supply and subsequently the MCU. A positive transition applied to either pin will enable the primary regulator. The state of the V_{IGN} pin can be determined by reading the IGNS bit of the MISCELLANEOUS register. The primary regulator can be disabled by clearing the PDC bit in the MISCELLANEOUS register. The BUS or V_{IGN} pin need not be low to write the PDC bit. Once the regulator has been disabled, a rising edge on V_{IGN} or BUS will re-enable the regulator.

To ensure the validity of the PDC bit during a V_{DD} transition, the PDC bit is always copied into a latch powered by the secondary regulator. The bit is copied during a write to the MISCELLANEOUS register. When the regulator is powered up by a rising edge of BUS or V_{IGN}, the latch that contains the copy of the PDC bit will be initialized to a 1 and may have a different value than the PDC bit itself. Software should always initialize the PDC bit to a 1. The latch will be set on the rising edge of V_{IGN} or BUS. This will give the primary regulator, CPU, and software ample time to start up and initialize the PDC bit to a 1. At this point, the MCU will transition to the Power Up Mode. (See Figure 7-2.)

In order to ensure that the primary regulator starts up when V_{BATT} voltage is applied to the MCU, a low to high transition should be provided on the V_{IGN} pin once V_{BATT} reaches minimum level, shown in Figure 7-3.

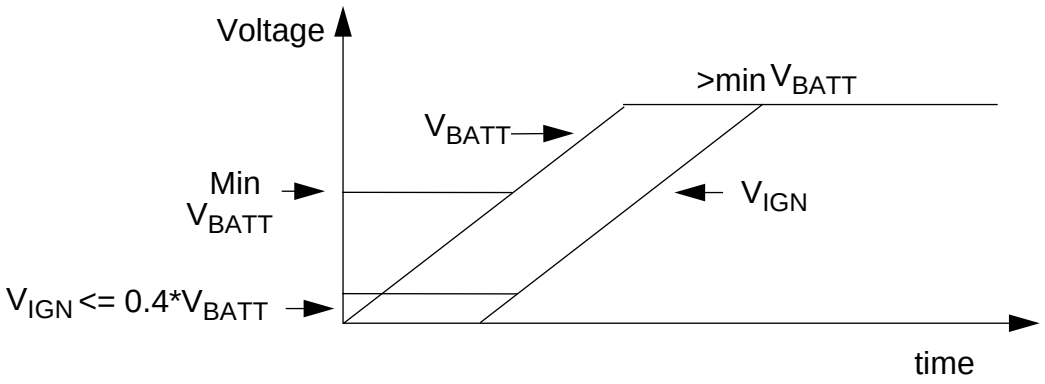


Figure 7-3: Regulator Startup

NOTE: Once enabled, the regulator can only be disabled through software.

NOTE: If the LVR option is enabled, reset will be asserted when V_{DD} drops to the appropriate level.

NOTE: Before the PDC bit is written to a 0, the state of the IGNS bit should be debounced in software. This will ensure that the primary regulator does not power up with any bounce that may be present on V_{IGN}.

7.4.1 MASK OPTIONS

The regulator has three mask options associated with it. The first one is the MDLCPU option. If this option is enabled, a rising edge on the MDLC BUS pin or the V_{IGN} pin will bring the primary regulator out of the Standby power mode. If this option is not enabled, only the V_{IGN} pin will bring the primary regulator out of the Standby power mode.

The second mask option is the V_{DDC} option. This option, if enabled, will enable an active pulldown device, connected between V_{DD} and V_{SS}, to turn on when the primary regulator is disabled. This option should not be selected if an external voltage regulator is being used.

The third mask option is the regulator enable option. If it is not desired to use the on-board voltage regulator, this option should not be selected. In this case the secondary regulator will still remain active in order to properly keep the primary regulator turned off. While the regulator is not enabled the MDLCPU option will remain operational but will serve no purpose. The V_{DDC} option and the MISCELLANEOUS register will continue to operate normally. The V_{DDC} function (V_{DD} to V_{SS} clamping device) uses the PDC bit to determine when to turn the clamping device on. If this option is enabled and the PDC bit is written to a "0", the clamping device will turn on. It will not turn off until a rising edge is detected on V_{IGN} or a rising edge on the BUS pin is detected provided the MDLCPU option is selected.

If the configuration shown in Figure 7-4 is used with the regulator enable option selected, the MDLCPU and V_{DDC} options may be selected. In addition, software must write a ’1’ to the PDC bit in the miscellaneous register to initialize it after power-up. This prevents read-modify-write instructions to the MISCELLANEOUS register from accidentally clearing the PDC bit.

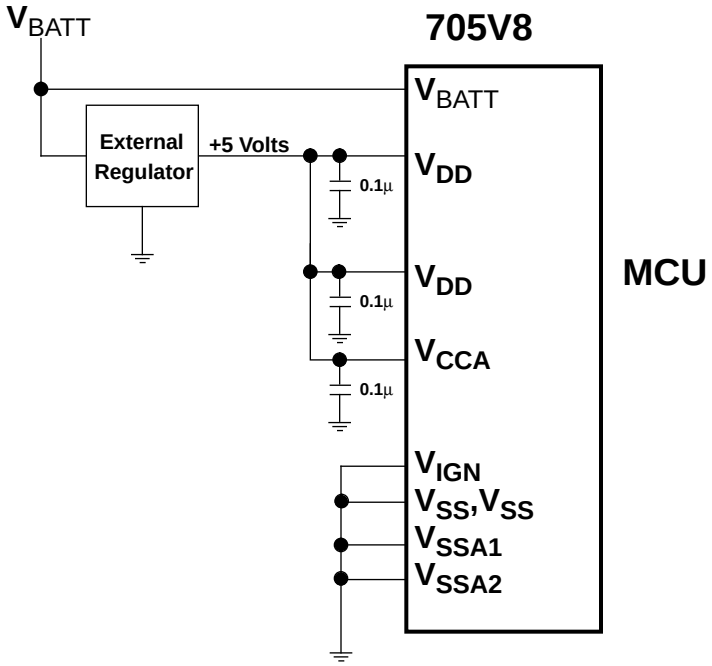


Figure 7-4: Using the 68HC705V8 with an External Regulator

If the regulator enable option is not selected, the MDLCPU option and the PDC bit have no affect. The V_{DDC} option will remain available.

Figure 7-5 is a block diagram of the regulator control circuit. It illustrates how the MISCELLANEOUS register bits, V_{IGN}, BUS and the Mask options interact in the power moding scheme.

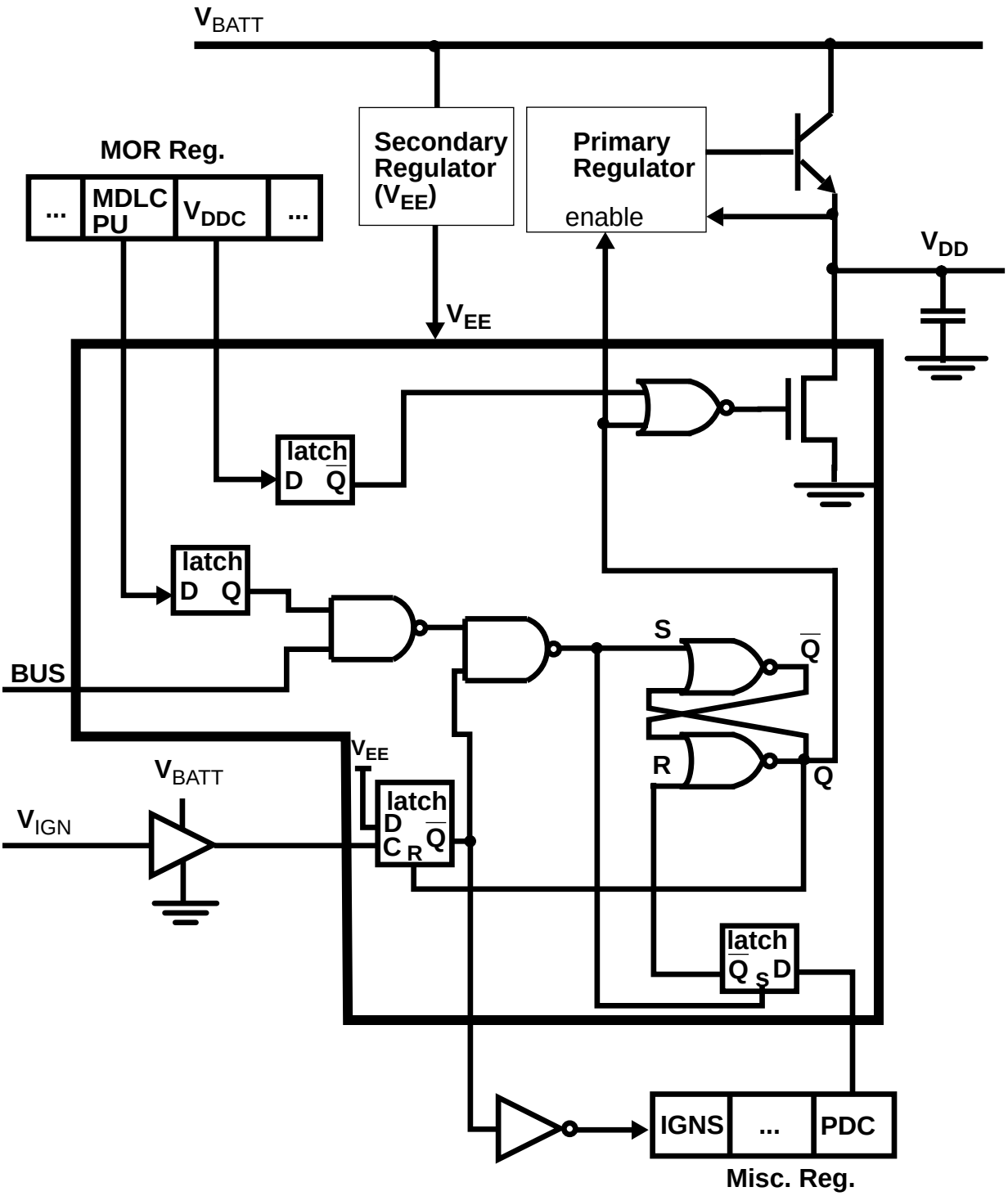
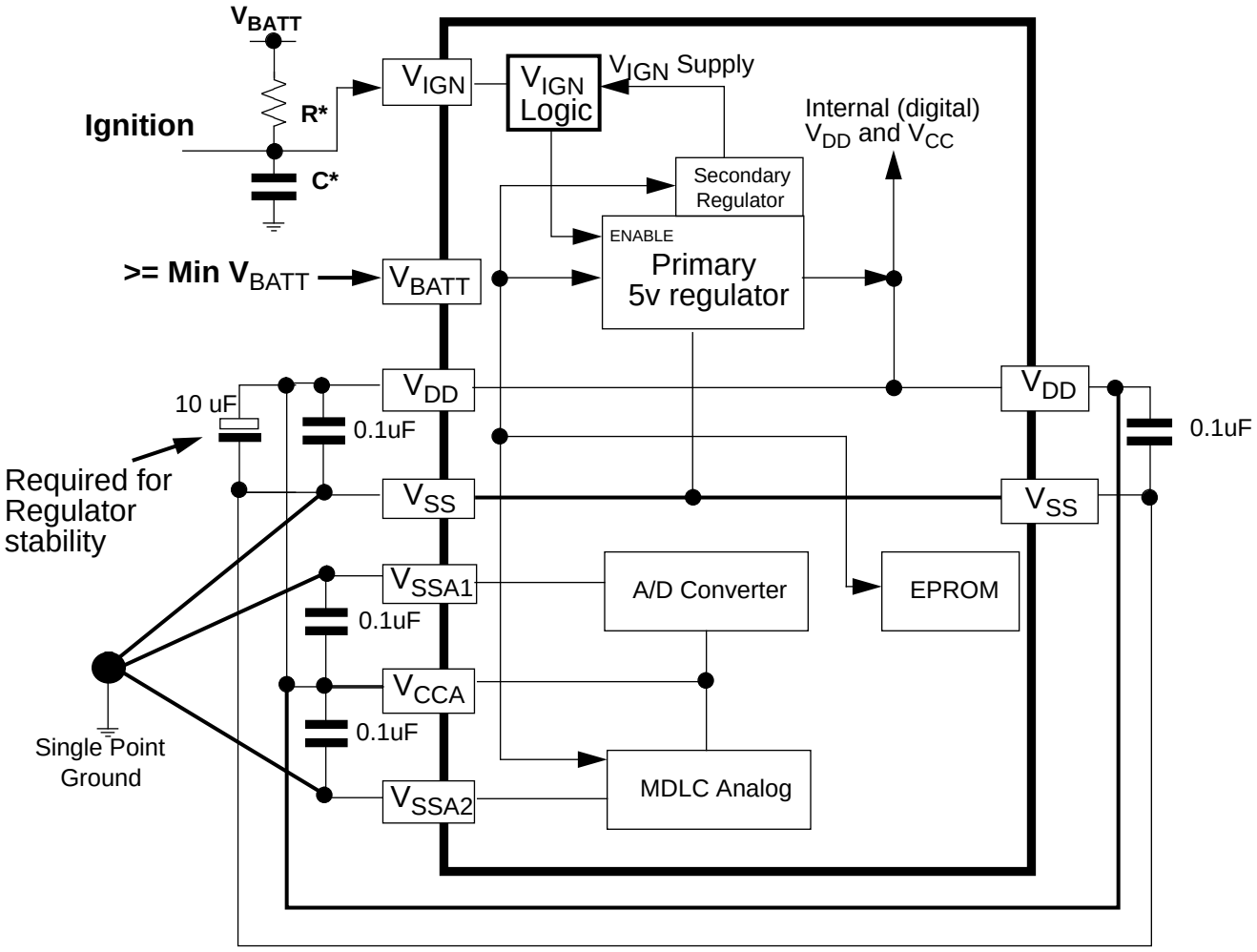


Figure 7-5: Power Moding Block Diagram

7.5 POWER SUPPLY CONFIGURATION

The recommended decoupling and interconnection of the various power supply pins is shown in Figure 7-6. Note that this diagram does not reflect actual pin location or order.



* RC values should be chosen to provide a V_{IGN} voltage of <= 0.4*V_{BATT} once V_{BATT} reaches minimum level.

Note: The 10 uF capacitor should be placed on the V_{DD} pin closest to the V_{IGN} pin.

Figure 7-6: MC68HC705V8 On-chip Power Supply Configuration

7.5.1 DECOUPLING RECOMMENDATIONS

To provide effective decoupling and to reduce radiated RF emissions, the small decoupling capacitors must be located as close to the supply pins as possible. The self-inductance of these capacitors and the parasitic inductance and capacitance of the interconnecting traces determines the self-resonant frequency of the decoupling network. Too low a frequency will reduce decoupling effectiveness and could increase radiated RF emissions from the system. A low value capacitor (470 pF to 0.01 uF) placed in parallel with the other capacitors will improve the

bandwidth and effectiveness of the network.

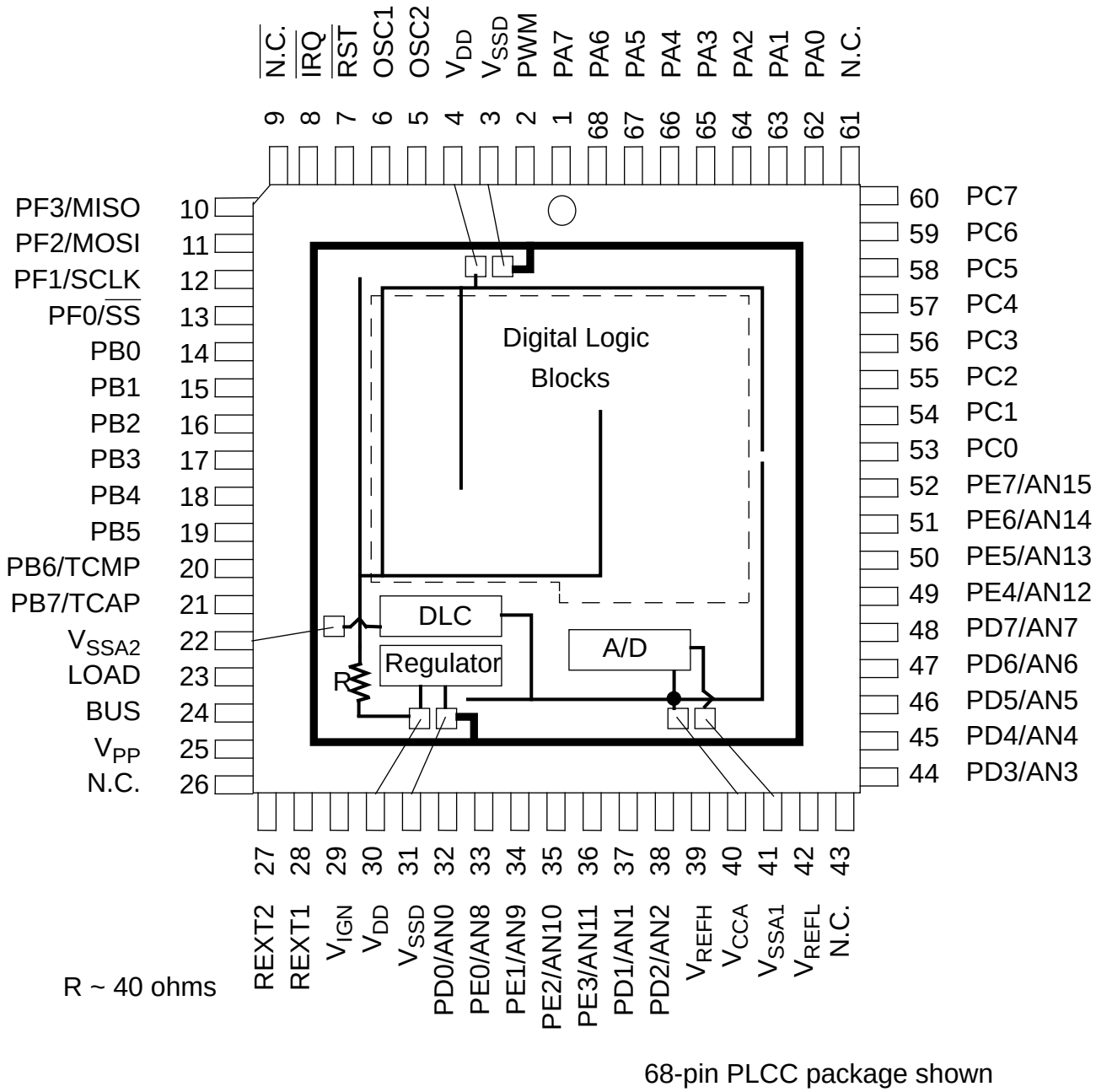


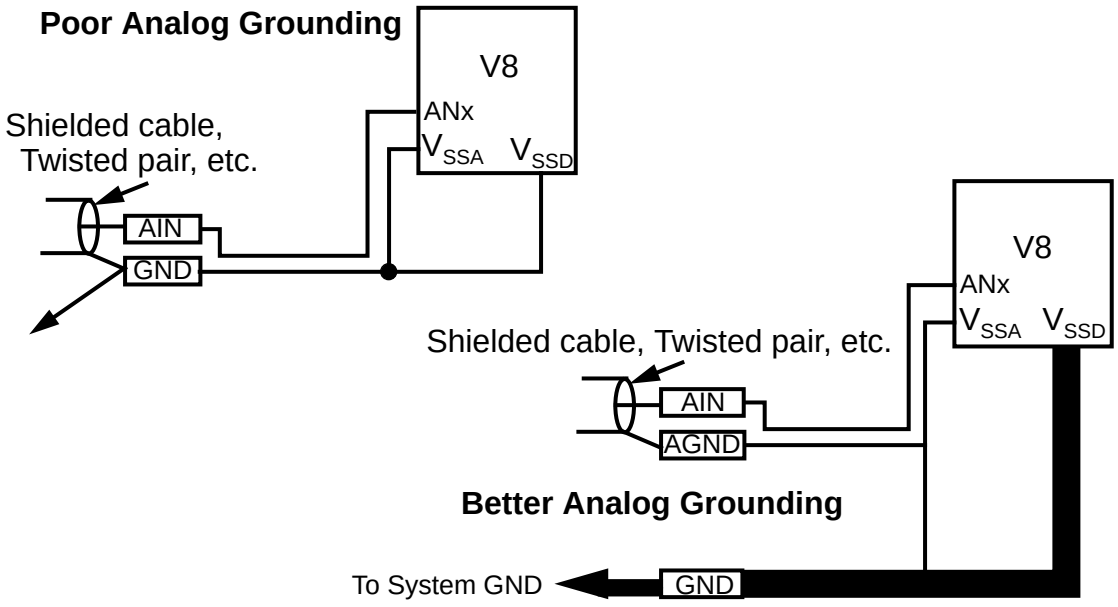
Figure 7-7: HC705V8 Internal Power Routing

1) V_{DD} to V_{SSD} (pins 30 & 31): On-chip regulator output decoupling. As with any regulator, the output must be adequately decoupled to maintain stability of the regulator. Decouple with at least 10 uF tantalum or aluminum electrolytic bulk in parallel with 0.1 uF ceramic or polystyrene capacitor for high frequency (HF) stability. Locate both capacitors as close as possible to

the V_{DD} and V_{SSD} pins, ensuring the HF capacitor is closest. The two V_{SSD} pins should be externally connected together.
The effect of not connecting the two V_{DD} pins together is under evaluation and it is recommended that they be connected at this time.

2) V_{DD} to V_{SSD} (pins 3 & 4): MCU internal digital power decoupling. These pins are connected to the on-chip regulator output via a small resistance. Decouple with a 0.1uF ceramic or polystyrene capacitator. If the self-resonance frequency of the decoupling circuit (assume 4nH per bond wire) is too low, add a 0.01uF or smaller capacitor in parallel to increase the bandwidth of the decoupling network. Ensure the smaller capacitor is located closest to the V_{DD} and V_{SSD} pins.

3) V_{CCA} to V_{SSA1} and V_{SSA2}: Analog subsystem power supply pins. These pins are internally isolated from the digital V_{DD} and V_{SS} supplies. The V_{SSA1} pin provides a ground return for the A/D subsystem. The V_{SSA2} pin provides a ground return for the DLC subsystem. The analog supply pins should be appropriately filtered to prevent any external noise effecting the analog subsystems. The V_{SSA1} and V_{SSA2} pins should be brought together with the digital ground at a single point which has a low (HF) impedance to ground to prevent common mode noise problems. If this is not practical, then the V_{SSA1} and V_{SSA2} printed circuit board (PCB) traces should be routed in such a manner that digital ground return current is impeded from passing through the analog input ground reference as shown in the single sided PCB example below.



SECTION 8 LOW-POWER MODES

The MC68HC705V8 is capable of running in one of several low-power operational modes. The WAIT and STOP instructions provide two modes that reduce the power required for the MCU by stopping various internal clocks and/or the on-chip oscillator. The STOP and WAIT instructions are not normally used if the COP Watchdog Timer is enabled. A mask option is provided to convert the STOP instruction to an internal reset. The flow of the STOP and WAIT modes is shown in Figure 8-2.

8.1 STOP INSTRUCTION

The STOP instruction can result in one of two operations depending on the state of the MOR's STOP bit. If the STOP option is enabled, the STOP instruction operates like the STOP in normal MC68HC05 family members and places the device in the low power STOP Mode. If the STOP option is disabled, the STOP instruction will cause a chip reset when executed.

8.1.1 STOP MODE

Execution of the STOP instruction, with the proper mask option, places the MCU in its lowest power consumption mode. In the STOP Mode the internal oscillator is turned off, halting *all* internal processing, including the COP Watchdog Timer.

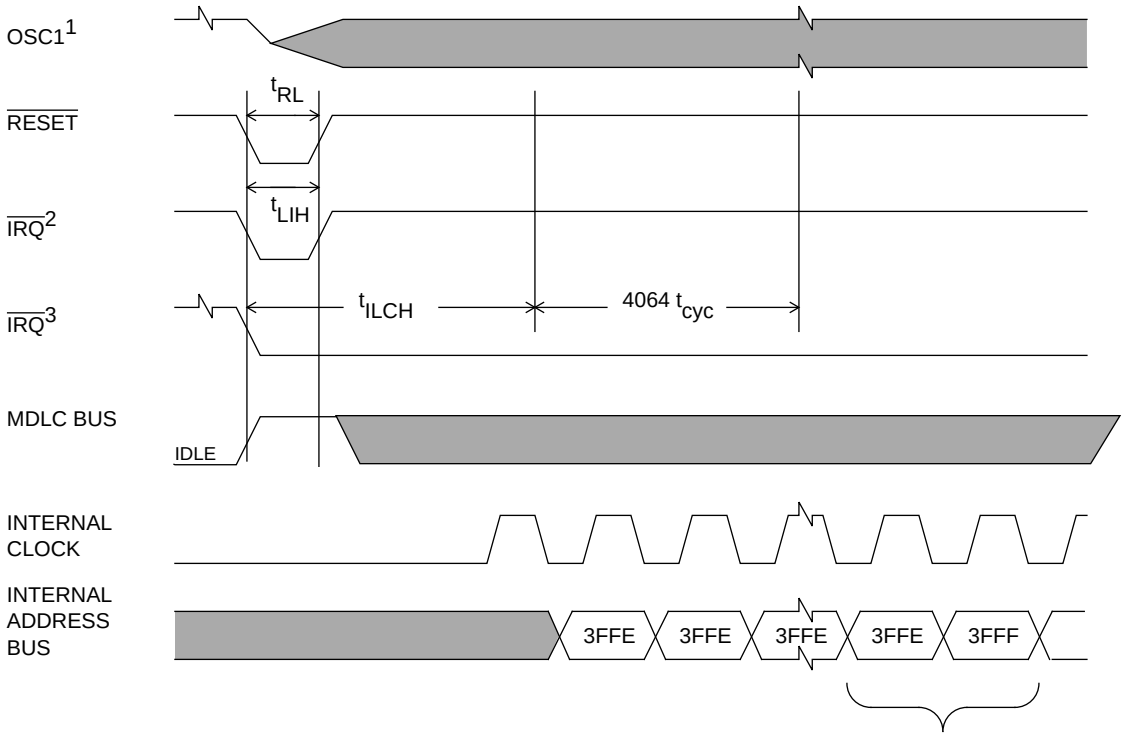
During the STOP mode, the TCR bits are altered to remove any pending timer interrupt request and to disable any further timer interrupts. The timer prescaler is cleared. The I bit in the CCR is cleared and the IRQE mask is set in the ICSR to enable external interrupts. All other registers and memory remain unaltered. All input/output lines remain unchanged. The processor can be brought out of the STOP mode only by an external interrupt or reset.

The MCU can be brought out of the STOP Mode by only one of the following:

- $\overline{\text{IRQ}}$ pin external interrupt
- Externally generated RESET
- Falling edge on any Port A or Port C pin (if enabled)
- Rising edge on the MDLC BUS pin

When exiting the STOP Mode, the internal oscillator will resume after a 4064 internal processor clock cycle oscillator stabilization delay, as shown in Figure 8-1.

NOTE: Execution of the STOP instruction with the proper mask option will cause the oscillator to stop and therefore disable the COP Watchdog Timer. If the COP Watchdog Timer is to be used, the STOP Mode should be disabled by selecting the proper mask option. See **6.2.3.4 COP WATCHDOG TIMER CONSIDERATIONS** for more details.



Notes:
1. Represents the internal gating of the OSC1 pin.
2. $\overline{IRQ}$ pin edge-sensitive mask option or Port A/C pin.
3. $\overline{IRQ}$ pin level and edge sensitive mask option.

Figure 8-1: Stop Recovery Timing Diagram

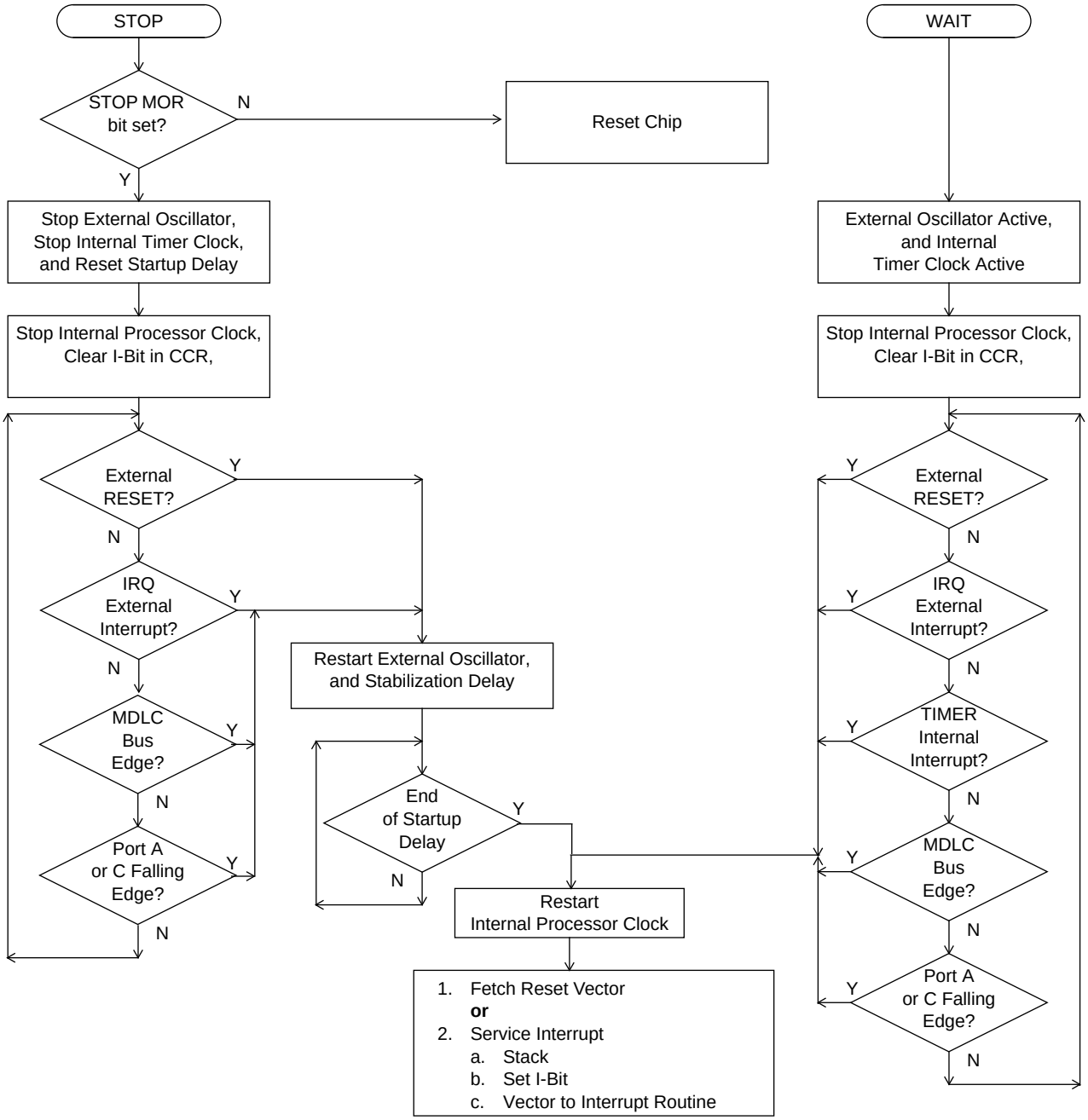


Figure 8-2: STOP/WAIT Flowcharts

8.1.2 WAIT INSTRUCTION

The WAIT instruction places the MCU in a low-power mode, which consumes more power than the STOP Mode. In the WAIT Mode the internal processor clock is halted, suspending all processor and internal bus activity. Internal timer clocks remain active, permitting interrupts to be generated from the timer or a reset to be generated from the COP Watchdog Timer. Execution of the WAIT instruction automatically clears the I-bit in the Condition Code Register. All other registers, memory, and input/output lines remain in their previous states.

If timer interrupts are enabled, a TIMER interrupt will cause the processor to exit the WAIT Mode and resume normal operation. The Timer may be used to generate a periodic exit from the WAIT Mode. The MCU can be brought out of the WAIT Mode by one of the following:

- TIMER interrupt from either timer
- $\overline{\text{IRQ}}$ pin external interrupt
- Externally generated RESET
- Falling edge on any Port A or Port C pin (if enabled)
- Rising edge on the MDLC BUS pin

SECTION 9 PARALLEL I/O

In the Single-Chip Mode there are 28 bidirectional I/O lines arranged as three 8-bit I/O ports (Ports A, B and C) and one 4-bit I/O port (Port F), and 16 input only lines arranged as two 8-bit ports (Ports D and E). The individual bits in the I/O ports are programmable as either inputs or outputs under software control by the data direction registers (DDRs). The Port A and Port C pins also have the additional properties of acting as additional IRQ interrupt input sources.

9.1 PORT A AND PORT C

Port A and Port C are 8-bit bidirectional ports which share all of their pins with the IRQ interrupt system as shown in Figure 9-1. Each pin is controlled by the corresponding bits in a data direction register and a data register. The Port A Data Register is located at address $0000.The Port C Data Register is located at address $0002. The Port A Data Direction Register (DDRA) is located at address $0004. The Port C Data Direction Register (DDRC) is located at address $0006. Reset clears DDRA and DDRC. The Data Registers are unaffected by reset.

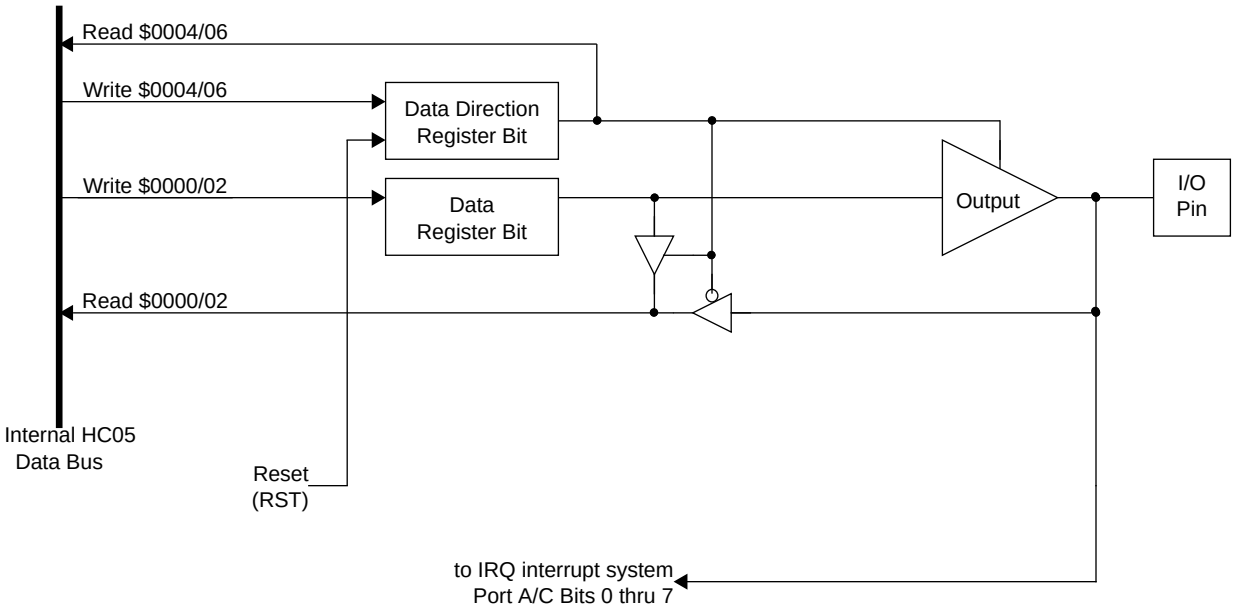


Figure 9-1: Port A and Port C I/O Circuitry

9.1.1 PORT A/C DATA REGISTERS

Each Port A/C I/O pin has a corresponding bit in the Port A/C Data Register. When a Port A/C pin is programmed as an output the state of the corresponding data register bit determines the state of the output pin. When a Port A/C pin is programmed as an input, any

read of the Port A/C Data Register will return the logic state of the corresponding I/O pin. The Port A/C data register is unaffected by reset.

9.1.2 PORT A/C DATA DIRECTION REGISTER

Each Port A/C I/O pin may be programmed as an input by clearing the corresponding bit in the DDRA/C, or programmed as an output by setting the corresponding bit in the DDRA/C. The DDRA can be accessed at address $0004. The DDRC can be accessed at address $0006. The DDRA and DDRC are cleared by reset.

9.1.3 PORT A/C I/O PIN INTERRUPTS

The inputs of all eight bits of Port A and Port C are ANDed into the IRQ input of the CPU. Each port has its own interrupt request latch to enable the user to differentiate between the IRQ sources. The port IRQ inputs are falling edge sensitive only. Any Port A or Port C pin can be disabled as an interrupt input by setting the corresponding DDR bit. Any Port A or Port C pins that are outputs will not cause a port interrupt when the pin transitions from a 1 to a 0. However, since all inputs pins are ANDed together to form the interrupt signal, any input pin that remains low will inhibit the interrupt flag bit from being set when subsequent pins transition low.

NOTE: The BIH and BIL instructions will only apply to the level on the IRQ pin itself, and not to the internal IRQ input to the CPU. Therefore BIH and BIL cannot be used to obtain the result of the logical combination of the eight pins of Port A or Port C.

9.2 PORT B

Port B is an 8-bit bidirectional port. Each Port B pin is controlled by the corresponding bits in a data direction register and a data register as shown in Figure 9-2. PB6 and PB7 are shared with 16 Bit Timer functions. See **SECTION 11 16-BIT TIMER**. The Port B Data Register is located at address $0001. The Port B Data Direction Register (DDRB) is located at address $0005. Reset clears the DDRB register. The Port B Data Register is unaffected by reset.

.

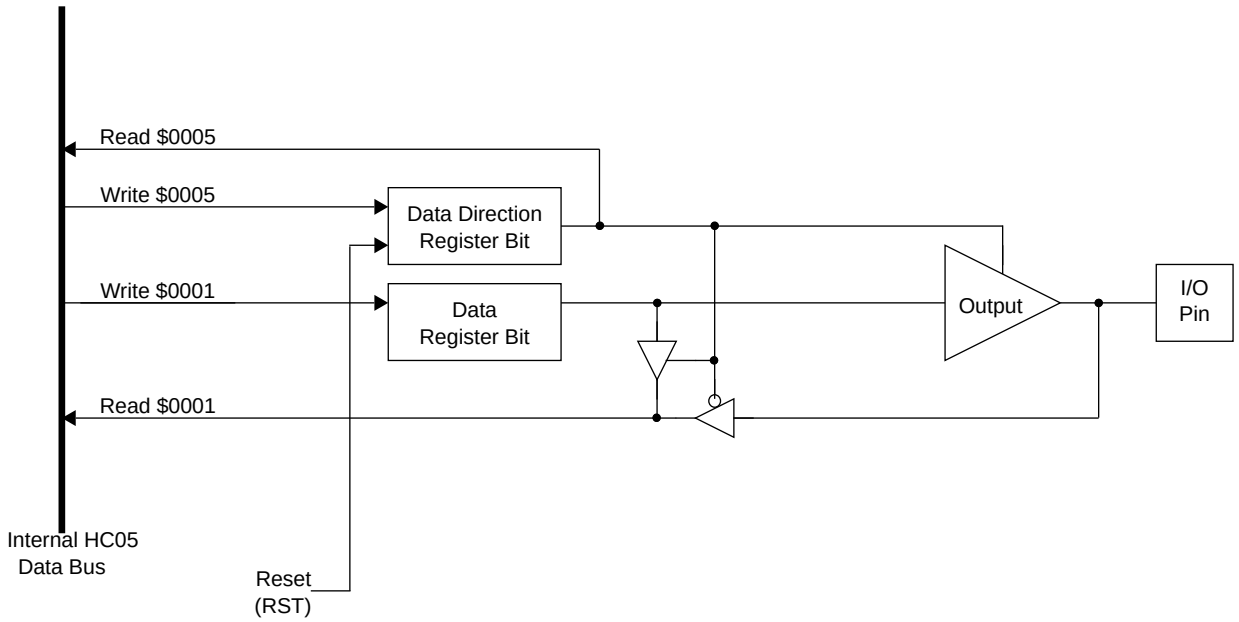


Figure 9-2: Port B I/O Circuitry

9.2.1 PORT B DATA REGISTER

Each Port B I/O pin has a corresponding bit in the Port B Data Register. When a Port B pin is programmed as an output the state of the corresponding data register bit determines the state of the output pin. When a Port B pin is programmed as an input, any read of the Port B Data Register will return the logic state of the corresponding I/O pin. The Port B data register is unaffected by reset. PB7 serves as the TCAP input and PB6 serves as the TCMP output if the OCE bit in the MISC Register is set. See **7.2 MISCELLANEOUS REGISTER**.

9.2.2 PORT B DATA DIRECTION REGISTER

Each Port B I/O pin may be programmed as an input by clearing the corresponding bit in the DDRB, or programmed as an output by setting the corresponding bit in the DDRB. The DDRB can be accessed at address $0005. The DDRB is cleared by reset.

9.3 PORT D AND PORT E

Port D is an 8-bit input only port that shares all of its pins with the A/D converter (AN0 through AN7) as shown in Figure 1-6. The Port D Data Register is located at address $0003. Port E is an 8-bit input only port which also shares all of its pins with the A/D converter (AN8 through AN15). The Port E Data Register is located at address $002B. When the A/D converter is active, one of these 16 input ports may be selected by the A/D multiplexer for conversion. A logical read of a selected input port will always return 0.

NOTE: Port E is not bonded out in the 56-pin SDIP packaged version.

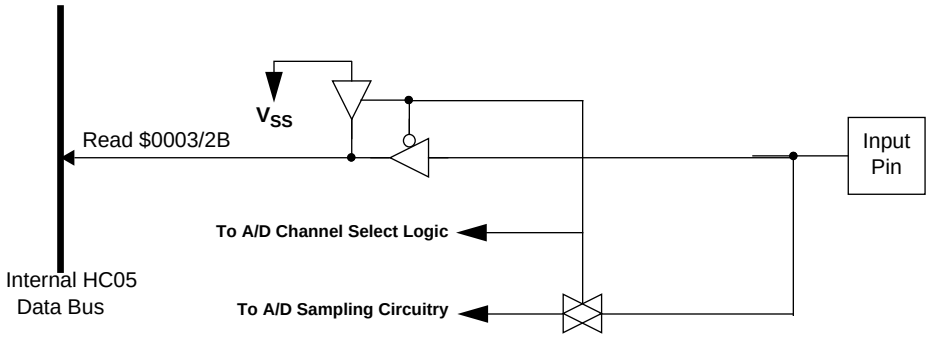


Figure 9-3: Port D and Port E Circuitry

9.4 PORT F

Port F is a 4-bit bidirectional port which shares all of its pins with the SPI subsystem. If the SPI is enabled, Port F3-1 are disabled as regular I/O pins. PF0 can still function as an output if the DDR bit is set. Each Port F pin is controlled by the corresponding bits in a data direction register and a data register. The Port F Data Register is located at address $002C. The Port F Data Direction Register (DDRF) is located at address $002E. Reset clears the DDRF register. The Port F Data Register is unaffected by reset. Port F is shared with the SPI.

9.4.1 PORT F DATA REGISTER

Each Port F I/O pin has a corresponding bit in the Port F Data Register. When a Port F pin is programmed as an output the state of the corresponding data register bit determines the state of the output pin. When a Port F pin is programmed as an input, any read of the Port F Data Register will return the logic state of the corresponding I/O pin. The Port F data register may be read even if the SPI subsystem is enabled. The Port F data register is unaffected by reset.

9.4.2 PORT F DATA DIRECTION REGISTER

Each Port F I/O pin may be programmed as an input by clearing the corresponding bit in the DDRF, or programmed as an output by setting the corresponding bit in the DDRF. The DDRF can be accessed at address $002E. If the SPI subsystem is enabled (SPE bit is set), the DDRF bit corresponding to PF0 will determine the function of the PF0 pin. If DDRF0 is cleared, the PF0 pin functions as a SS (slave select) input to the SPI. If DDRF0 is set, then PF0 can serve as a normal output pin. When the SPI subsystem is enabled, the DDRF3-1 bits are set or cleared according to the required pin function. The DDRF is cleared by reset.

SECTION 10 A/D CONVERTER

The MC68HC705V8 includes a 16 channel, 8-bit, multiplexed input, successive approximation A/D converter. When the device is packaged (in a 56-pin SDIP), pin number restrictions require that 8 channels, AN8 through AN15, are not bonded out.

10.1 ANALOG SECTION

10.1.1 RATIOMETRIC CONVERSION

The A/D is ratiometric, with two dedicated pins supplying the reference voltages (V_{REFH} and V_{REFL}). An input voltage equal to V_{REFH} converts to $FF (full scale) and an input voltage equal to V_{REFL} converts to $00. An input voltage greater than V_{REFH} will convert to $FF with no overflow indication. For ratiometric conversions, the source of each analog input should use V_{REFH} as the supply voltage and be referenced to V_{REFL}.

10.1.2 V_{REFH} AND V_{REFL}

The reference supply for the converter uses two dedicated pins rather than being driven by the system power supply lines because the voltage drops in the bonding wires of those heavily loaded pins would degrade the accuracy of the A/D conversion. V_{REFH} and V_{REFL} can be any voltage between V_{SSA1} and V_{CCA}, as long as $V_{REFH} > V_{REFL}$; however, the accuracy of conversions is tested and guaranteed only for $V_{REFL} = V_{SSA1}$ and $V_{REFH} = V_{CCA}$.

10.1.3 ACCURACY AND PRECISION

The 8-bit conversions shall be accurate to within $\pm 1\ ^1/_2$ LSB including quantization.

10.2 CONVERSION PROCESS

The A/D reference inputs are applied to a precision internal digital-to-analog converter. Control logic drives this D/A and the analog output is successively compared to the selected analog input which was sampled at the beginning of the conversion time. The conversion process is monotonic and has no missing codes.

10.3 DIGITAL SECTION

10.3.1 CONVERSION TIMES

Each channel of conversion takes 32 clock cycles, which must be at a frequency equal to or greater than 1 MHz.

10.3.2 INTERNAL VS. MASTER OSCILLATOR

If the MCU bus (E clock) frequency is less than 1.0 MHz, an internal RC oscillator (nominally 1.5 MHz) must be used for the A/D conversion clock. This selection is made by setting the ADRC bit in the A/D Status and Control Registers to 1. In STOP mode, the internal RC oscillator is turned off automatically, though the A/D subsystem remains enabled (ADON remains set). In WAIT mode the A/D subsystem remains functional. See **10.6 A/D DURING WAIT MODE**.

When the internal RC oscillator is being used as the conversion clock, three limitations apply:

1. The Conversion Complete Flag (COCO) must be used to determine when a conversion sequence has been completed, due to the frequency tolerance of the RC oscillator and its asynchronism with regard to the MCU E clock.
2. The conversion process runs at the nominal 1.5 MHz rate but the conversion results must be transferred to the MCU result registers synchronously with the MCU E clock so conversion time is limited to a maximum of one channel per E cycle.
3. If the system clock is running faster than the RC oscillator, the RC oscillator should be turned off, and the system clock used as the conversion clock.

10.3.3 MULTI-CHANNEL OPERATION

A multiplexer allows the A/D converter to select one of sixteen external analog signals and four internal reference sources.

10.4 A/D STATUS AND CONTROL REGISTER (ADSCR)

The following paragraphs describe the function of the A/D Status and Control Register.

$1E	COCO	ADRC	ADON	CH4	CH3	CH2	CH1	CH0
RESET:	0	0	0	0	0	0	0	0

Figure 10-1: A/D Status and Control Register

10.4.1 COCO - Conversions Complete

This read-only status bit is set when a conversion is completed, indicating that the A/D Data Register contains valid results. This bit is cleared whenever the A/D Status and Control Register is written and a new conversion is automatically started, or whenever the A/D Data Register is read. Once a conversion has been started by writing to the A/D Status and Control Register, conversions of the selected channel will continue every 32 cycles until the A/D Status and Control Register is written again. In this continuous conversion mode the A/D Data Register will be filled with new data, and the COCO bit set, every 32 cycles. Data from the previous conversion will be overwritten regardless of the state of the COCO bit prior to writing.

10.4.2 ADRC - RC Oscillator Control

When ADRC is set, the A/D section runs on the internal RC oscillator instead of the CPU clock. The RC oscillator requires a time t_{RCON} to stabilize, and results can be inaccurate during this time.

10.4.3 ADON - A/D On

When the A/D is turned on (ADON = 1), it requires a time t_{ADON} for the current sources to stabilize, and results can be inaccurate during this time. This bit turns on the charge pump.

10.4.4 CH4:CH0 - Channel Select Bits

CH4, CH3, CH2, CH1, and CH0 form a 5-bit field which is used to select one of twenty A/D channels, including four internal references. Channels $0-7 correspond to Port D input pins on the MCU. Channels $8-F correspond to Port E input pins on the MCU and are not bonded out in 56 pin packaged parts. Channels $10-13 are used for internal reference points. In Single-Chip Mode, channel $13 is reserved and converts to $00. The following table shows the signals selected by the channel select field.

Table 10-1: A/D Channel Assignments*

CH4:CH0	Signal	
00-07	AN0-7	
08-0F	AN8-15	
10	V_{REFH}	*
11	$(V_{REFH}-V_{REFL})/2$	*
12	V_{REFL}	*
13	Factory Test	
14	Unused	

* Accuracy not guaranteed for internal channels

10.5 A/D DATA REGISTERS

An 8-bit result register is provided. This register is updated each time the COCO bit is set.

$1D	D7	D6	D5	D4	D3	D2	D1	D0

Figure 10-2: A/D Data Register

10.6 A/D DURING WAIT MODE

The A/D converter continues normal operation during WAIT mode. To decrease power consumption during WAIT, it is recommended that both the ADON and ADRC bits in the A/D Status and Control Registers be cleared if the A/D converter is not being used. If the A/D converter is in use and the system clock rate is above 1.0 MHz, it is recommended that the ADRC bit be cleared.

NOTE: As the A/D converter continues to function normally in WAIT mode, the COCO bit is not cleared.

10.7 A/D DURING STOP MODE

In STOP mode the comparator and charge pump are turned off and the A/D ceases to function. Any pending conversion is aborted. When the clocks begin oscillation upon leaving the STOP mode, a finite amount of time passes before the A/D circuits stabilize enough to provide conversions to the specified accuracy. Normally, the delays built into the device when coming out of STOP mode are sufficient for this purpose, therefore no explicit delays need to be built into the software.

NOTE: Although the comparator and charge pump are disabled in STOP mode, the A/D Data and Status/Control registers are not modified. Disabling the A/D prior to entering STOP mode will not effect the STOP mode current consumption.

SECTION 11 16-BIT TIMER

The timer consists of a 16-bit, free-running counter driven by a fixed divide-by-four prescaler. This timer can be used for many purposes, including input waveform measurements while simultaneously generating an output waveform. Pulse widths can vary from several microseconds to many seconds. See Figure 11-1.

Because the timer has a 16-bit architecture, each specific functional segment (capability) is represented by two registers. These registers contain the high and low byte of that functional segment. Access of the high byte inhibits that specific timer function until the low byte is also accessed.

NOTE: The I bit in the CCR should be set while manipulating both the high and low byte register of a specific timer function to ensure that an interrupt does not occur.

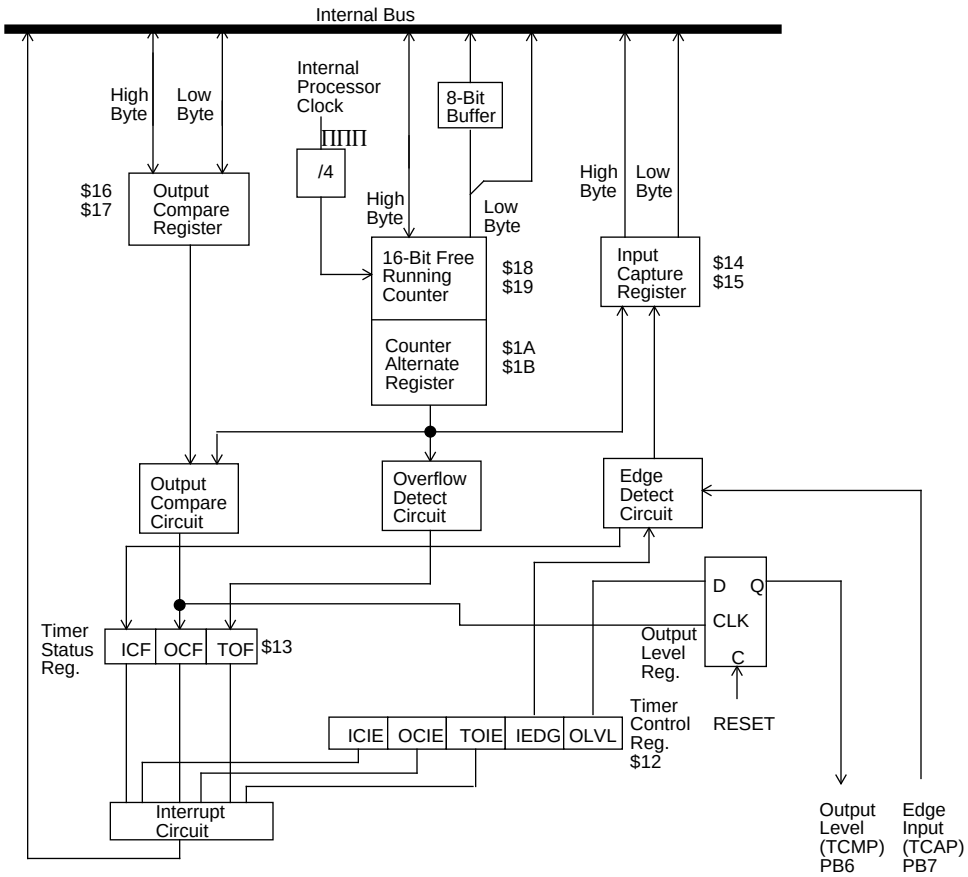


Figure 11-1: 16-Bit Timer Block Diagram

11.1 COUNTER REGISTER - $18:$19, $1A:$1B

The key element in the programmable timer is a 16-bit, free-running counter or counter register, preceded by a prescaler that divides the internal processor clock by four. The prescaler gives the timer a resolution of 2.0 microseconds if the internal bus clock is 2.0 MHz. The counter is incremented during the low portion of the internal bus clock. Software can read the counter at any time without affecting its value.

The double-byte, free-running counter can be read from either of two locations, $18-$19 (counter register) or $1A-$1B (counter alternate register). A read from only the least significant byte (LSB) of the free-running counter ($19, $1B) receives the count value at the time of the read. If a read of the free-running counter or counter alternate register first addresses the most significant byte (MSB) ($18, $1A), the LSB ($19, $1B) is transferred to a buffer. This buffer value remains fixed after the first MSB read, even if the user reads the MSB several times. This buffer is accessed when reading the free-running counter or counter alternate register LSB ($19 or $1B) and, thus, completes a read sequence of the total counter value. In reading either the free-running counter or counter alternate register, if the MSB is read, the LSB must also be read to complete the sequence.

The counter alternate register differs from the counter register in one respect: a read of the counter register MSB can clear the timer overflow flag (TOF). Therefore, the counter alternate register can be read at any time without the possibility of missing timer overflow interrupts due to clearing of the TOF.

The free-running counter is configured to $FFFC during reset and is a read-only register but only when the timer is enabled. During a power-on reset, the counter is also preset to $FFFC and begins running only after the TON bit in the TIMER control register is set. Because the free-running counter is 16 bits preceded by a fixed divided-by-four prescaler, the value in the free-running counter repeats every 262,144 internal bus clock cycles. When the counter rolls over from $FFFF to $0000, the TOF bit is set. An interrupt can also be enabled when counter roll-over occurs by setting its interrupt enable bit (TOIE).

NOTE: The I bit in the CCR should be set while manipulating both the high and low byte register of a specific timer function to ensure that an interrupt does not occur.

11.2 OUTPUT COMPARE REGISTER - $16:$17

The 16-bit output compare register is made up of two 8-bit registers at locations $16 (MSB) and $17 (LSB). The output compare register is used for several purposes, such as indicating when a period of time has elapsed. All bits are readable and writable and are not altered by the timer hardware or reset. If the compare function is not needed, the two bytes of the output compare register can be used as storage locations.

The output compare register contents are continually compared with the contents of the free-running counter. If a match is found, the corresponding output compare flag (OCF) bit is set and the corresponding output level (OLVL) bit is clocked to an output level register. The output compare register values and the output level bit should be changed after each successful comparison to establish a new elapsed time-out. An interrupt can also

accompany a successful output compare provided the corresponding interrupt enable bit (OCIE) is set. After a processor write cycle to the output compare register containing the MSB ($16), the output compare function is inhibited until the LSB ($17) is also written. The user must write both bytes (locations) if the MSB is written first. A write made only to the LSB ($17) will not inhibit the compare function. The free-running counter is updated every four internal bus clock cycles. The minimum time required to update the output compare register is a function of the program rather than the internal hardware.

The processor can write to either byte of the output compare register without affecting the other byte. The output level (OLVL) bit is clocked to the output level register regardless of whether the output compare flag (OCF) is set or clear.

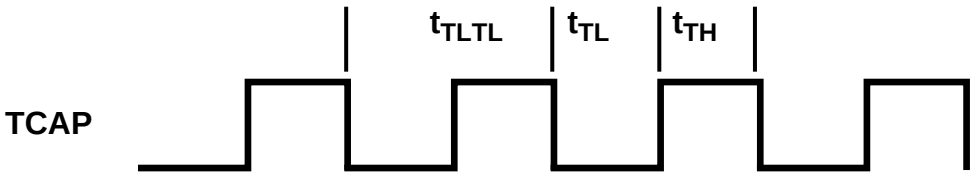
11.3 INPUT CAPTURE REGISTER - $14:$15

Two 8-bit registers, which make up the 16-bit input capture register, are read-only and are used to latch the value of the free-running counter after the corresponding input capture edge detector senses a defined transition. The level transition that triggers the counter transfer is defined by the corresponding input edge bit (IEDG). Reset does not affect the contents of the input capture register.

The result obtained by an input capture will be one more than the value of the free-running counter on the rising edge of the internal bus clock preceding the external transition. This delay is required for internal synchronization. Resolution is one count of the free-running counter, which is four internal bus clock cycles.

The free-running counter contents are transferred to the input capture register on each proper signal transition regardless of whether the input capture flag (ICF) is set or clear. The input capture register always contains the free-running counter value that corresponds to the most recent input capture.

After a read of the input capture register MSB ($14), the counter transfer is inhibited until the LSB ($15) is also read. This characteristic causes the time used in the input capture software routine and its interaction with the main program to determine the minimum pulse period. A read of the input capture register LSB ($15) does not inhibit the free-running counter transfer since they occur on opposite edges of the internal bus clock.



See control timing specifications for TCAP timing requirements.

Figure 11-2: TCAP Timing

NOTE: The Input Capture pin (TCAP) and the Output Compare pin (TCMP) are shared with PB7 and PB6 respectively. The timer's TCAP input is always connected to PB7. PB6 is the timer's TCMP pin if the OCE bit in the MISCELLANEOUS control register is set. See **7.2.2 OCE - Output compare enable**.

11.4 TIMER CONTROL REGISTER (TCR) - $12

The TCR is a read/write register containing six control bits. Three bits control interrupts associated with the timer status register flags ICF, OCF and TOF.

$12	ICIE	OCIE	TOIE	0	0	TON	IEDG	OLVL
RESET	0	0	0	0	0	0	0	0

Figure 11-3: Timer Control Register - $12

11.4.1 ICIE - Input Capture Interrupt Enable

1 - Interrupt enabled
0 - Interrupt disabled

11.4.2 OCIE - Output Compare Interrupt Enable

1 - Interrupt enabled
0 - Interrupt disabled

11.4.3 TOIE - Timer Overflow Interrupt Enable

1 - Interrupt enabled
0 - Interrupt disabled

11.4.4 TON - Timer On

When disabled, the timer is initialized to the reset condition.
1 - Timer enabled
0 - Timer disabled

11.4.5 IEDG - Input Edge

Value of input edge determines which level transition on TCAP pin will trigger free-running counter transfer to the input capture register. Reset clears this bit.
1 - Positive edge
0 - Negative edge

11.4.6 OLVL - Output Level

Value of output level is clocked into output level register by the next successful output compare and will appear on the TCMP pin

1 - High output
0 - Low output

11.5 TIMER STATUS REGISTER (TSR) - $13

The TSR is a read-only register containing three status flag bits.

$13	ICF	OCF	TOF	0	0	0	0	0
RESET	0	0	0	0	0	0	0	0

Figure 11-4: Timer Status Register - $13

11.5.1 ICF - Input Capture Flag

1 - Flag set when selected polarity edge is sensed by input capture edge detector
0 - Flag cleared when TSR and input capture low register ($15) are accessed

Reset clears this bit.

11.5.2 OCF - Output Compare Flag

1 - Flag set when output compare register contents match the free-running counter contents
0 - Flag cleared when TSR and output compare low register ($17) are accessed

Reset clears this bit.

11.5.3 TOF - Timer Overflow Flag

1 - Flag set when free-running counter transition from $FFFF to $0000 occurs
0 - Flag cleared when TSR and counter low register ($19) are accessed

Reset clears this bit.

11.5.4 Bits 0-4 - Not used

Always read zero.

Accessing the timer status register satisfies the first condition required to clear status bits. The remaining step is to access the register corresponding to the status bit.

A problem can occur when using the timer overflow function and reading the free-running counter at random times to measure an elapsed time. Without incorporating the proper precautions into software, the timer overflow flag could unintentionally be cleared if:

1. The timer status register is read or written when TOF is set, and
2. The MSB of the free-running counter is read but not for the purpose of servicing the flag.

The counter alternate register at address $1A and $1B contains the same value as the free-running counter (at address $18 and $19); therefore, this alternate register can be read at any time without affecting the timer overflow flag in the timer status register.

11.6 TIMER DURING WAIT MODE

The CPU clock halts during the WAIT mode, but the timer remains active if turned on prior to entering wait mode. If interrupts are enabled, a timer interrupt will cause the processor to exit the WAIT mode.

11.7 TIMER DURING STOP MODE

In the STOP mode, the timer stops counting and holds the last count value if STOP is exited by an interrupt. If RESET is used, the counter is forced to $FFFC. During STOP, if the TIMER is on and at least one valid input capture edge occurs at the TCAP pin, the input capture detect circuit is armed. This does not set any timer flags nor wake up the MCU, but when the MCU does wake up, there is an active input capture flag and data from the first valid edge that occurred during the STOP mode. If RESET is used to exit STOP mode, then no input capture flag or data remains, even if a valid input capture edge occurred.

SECTION 12 CORE TIMER

The Core Timer for this device is a 12-stage multifunctional ripple counter. The features include Timer Over Flow, Power-On Reset (POR), Real Time Interrupt (RTI), and COP Watchdog Timer.

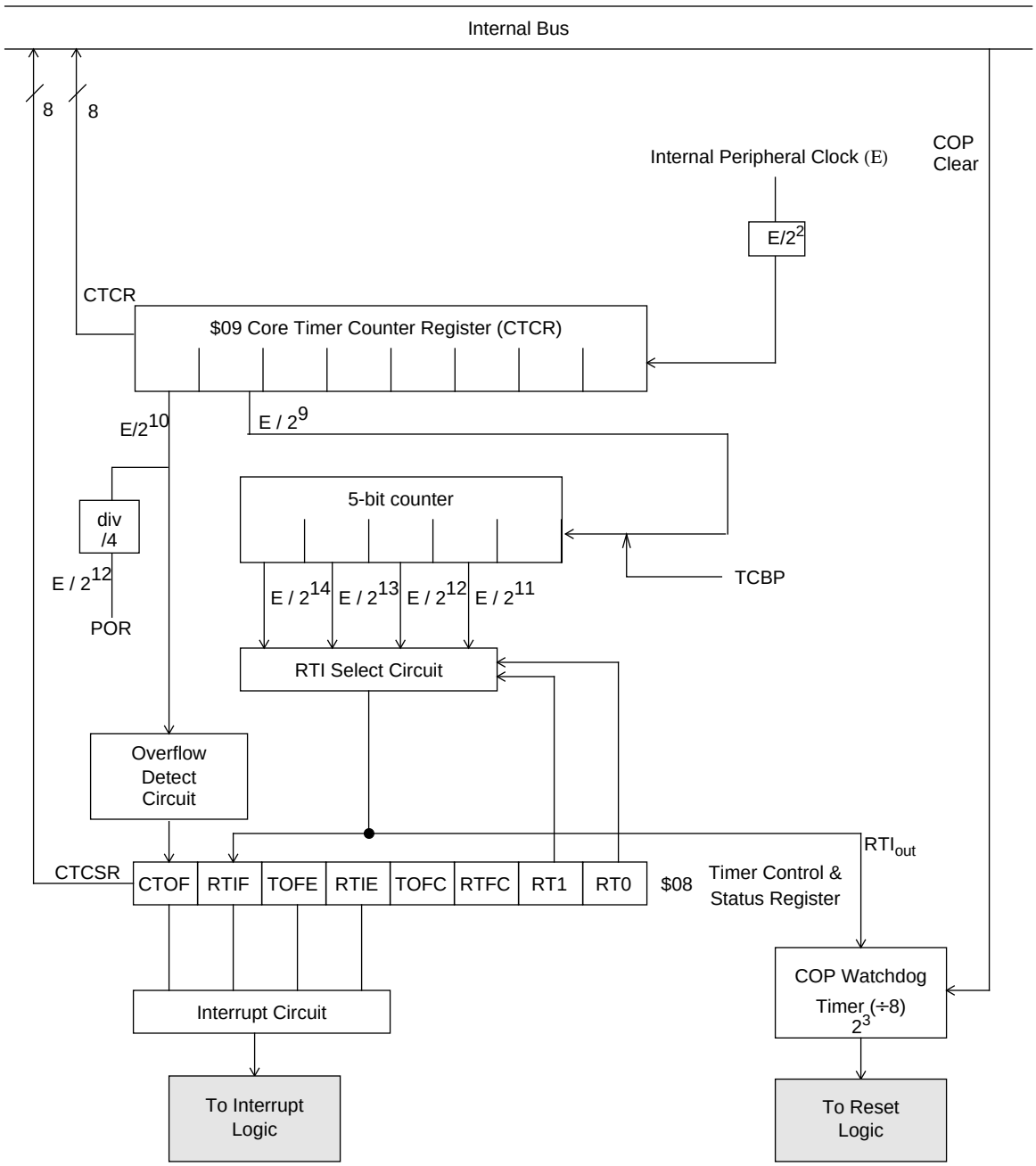


Figure 12-1: Core Timer Block Diagram

As seen in Figure 6-1, the internal peripheral clock is divided by four then drives an 8-bit ripple counter. The value of this 8-bit ripple counter can be read by the CPU at any time by accessing the Core Timer Counter Register (CTCR) at address $09. A timer overflow function is implemented on the last stage of this counter, giving a possible interrupt rate of the Internal Peripheral clock(E)/1024. This point is then followed by two more stages, with the resulting clock (E/2048) driving the Real Time Interrupt circuit (RTI). The RTI circuit consists of three divider stages with a 1 of 4 selector. The output of the RTI circuit is further divided by eight to drive the mask optional COP Watchdog Timer circuit. The RTI rate selector bits, and the RTI and CTOF enable bits and flags are located in the Timer Control and Status Register at location $08.

12.1 CORE TIMER CTRL & STATUS REGISTER (CTCSR) $08

The CTCSR contains the timer interrupt flag, the timer interrupt enable bits, and the real time interrupt rate select bits. Figure 12-2 shows the value of each bit in the CTCSR when coming out of reset.

$08	CTOF	RTIF	TOFE	RTIE	TOFC	RTFC	RT1	RT0
RESET:	0	0	0	0	0	0	1	1

Figure 12-2: Core Timer Control and Status Register

12.1.1 CTOF - Core Timer Over Flow

CTOF is a read-only status bit set when the 8-bit ripple counter rolls over from $FF to $00. Clearing the CTOF is done by writing a ‘1’ to TOFC. Writing to this bit has no effect. Reset clears CTOF.

12.1.2 RTIF - Real Time Interrupt Flag

The Real Time Interrupt circuit consists of a three stage divider and a 1 of 4 selector. The clock frequency that drives the RTI circuit is E/2**11 (or E/2048) with three additional divider stages giving a maximum interrupt period of 7.8 milliseconds at a bus rate of 2.1 MHz. RTIF is a clearable, read-only status bit and is set when the output of the chosen (1 of 4 selection) stage goes active. Clearing the RTIF is done by writing a “1” to RTFC. Writing has no effect on this bit. Reset clears RTIF.

12.1.3 TOFE - Timer Over Flow Enable

When this bit is set, a CPU interrupt request is generated when the CTOF bit is set. Reset clears this bit.

12.1.4 RTIE - Real Time Interrupt Enable

When this bit is set, a CPU interrupt request is generated when the RTIF bit is set. Reset clears this bit.

12.1.5 TOFC - Timer Over Flow Flag Clear

When a “1” is written to this bit, CTOF is cleared. Writing a “0” has no effect on the CTOF bit. This bit always reads as zero.

12.1.6 RTFC - Real Time Interrupt Flag Clear

When a “1” is written to this bit, RTIF is cleared. Writing a “0” has no effect on the RTIF bit. This bit always reads as zero.

12.1.7 RT1:RT0 - Real Time Interrupt Rate Select

These two bits select one of four taps from the Real Time Interrupt circuit. Table 12-1 shows the available interrupt rates with a 2.1 and 1.05MHz bus clock. Reset sets these two bits which selects the lowest periodic rate and gives the maximum time in which to alter these bits if necessary. Care should be taken when altering RT0 and RT1 if the time-out period is imminent or uncertain. If the selected tap is modified during a cycle in which the counter is switching, an RTIF could be missed or an additional one could be generated. To avoid problems, the COP should be cleared before changing RTI taps.

Table 12-1: RTI and COP Rates at 2.1 MHz

RTI RATE			RT1:RT0	MIN. COP RATES		
2.1 MHz	1.05 MHz				2.1 MHz	1.05 MHz
0.97 ms	1.95 ms	$2^{11}/E$	00	$(2^{14}-2^{11})/E$	6.83 ms	13.65ms
1.95 ms	3.90 ms	$2^{12}/E$	01	$(2^{15}-2^{12})/E$	13.65 ms	27.31ms
3.90 ms	7.80 ms	$2^{13}/E$	10	$(2^{16}-2^{13})/E$	27.31 ms	54.61ms
7.80 ms	15.60 ms	$2^{14}/E$	11	$(2^{17}-2^{14})/E$	54.61 ms	109.23ms

12.2 COMPUTER OPERATING PROPERLY (COP) RESET

The COP watchdog timer function is implemented on this device by using the output of the RTI circuit and further dividing it by eight. The minimum COP reset rates are listed in Table 12-1. If the COP circuit times out, an internal reset is generated and the normal reset vector is fetched. Preventing a COP time-out, or clearing the COP, is accomplished by writing a “0” to bit 0 of address $3FF0. When the COP is cleared, only the final divide by eight stage (output of the RTI) is cleared.

If the COP watchdog timer is allowed to time-out, an internal reset is generated to reset the MCU. In addition the $\overline{\text{RESET}}$ pin will be pulled low for a minimum of 3 E Clock cycle for emulation purposes. During a chip reset (regardless of the source), the entire Core Timer counter chain is cleared.

The COP will remain enabled after execution of the WAIT instruction and all associated operations apply. If the STOP instruction is disabled, execution of STOP instruction will cause an internal reset.

This COP’s objective is to make it impossible for this part to become “stuck” or “locked-up” and to be sure the COP is able to “rescue” the part from any situation where it might entrap itself in an abnormal or unintended behavior. This function is a mask option.

12.3 CORE TIMER COUNTER REGISTER (CTCR) $09

The Timer Counter Register is a read-only register which contains the current value of the 8-bit ripple counter at the beginning of the timer chain. This counter is clocked by the CPU clock (E/4) and can be used for various functions including a software input capture. Extended time periods can be attained using the TOF function to increment a temporary RAM storage location thereby simulating a 16-bit (or more) counter.

$09	D7	D6	D5	D4	D3	D2	D1	D0

Figure 12-3: Timer Counter Register

The power-on cycle clears the entire counter chain and begins clocking the counter. After 4064 cycles, the power-on reset circuit is released which again clears the counter chain and allows the device to come out of reset. At this point, if $\overline{\text{RESET}}$ is not asserted, the timer will start counting up from zero and normal device operation will begin. When $\overline{\text{RESET}}$ is asserted anytime during operation (other than POR), the counter chain will be cleared.

12.4 TIMER DURING WAIT MODE

The CPU clock halts during the WAIT mode, but the timer remains active. If interrupts are enabled, a timer interrupt will cause the processor to exit the WAIT mode. The COP is always enabled while in user mode.

SECTION 13 PULSE WIDTH MODULATOR

The pulse width modulator (PWM) system has one 6-bit channel (PWMA). Preceding the 6-bit (÷64) PWM are two programmable prescalers.The PWM frequency is selected by choosing the desired divide option from the programmable prescalers. Note that the PWM clock input is f_{osc}, which is twice the bus frequency. The PWM frequency will be f_{osc}/(PSA+PSB+64) where PSA and PSB are the values selected by the A and B prescaler and 64 comes from the 6-bit modulus counter. See Table 13-1 for precise values. E is the internal bus frequency fixed to half of the external oscillator frequency.

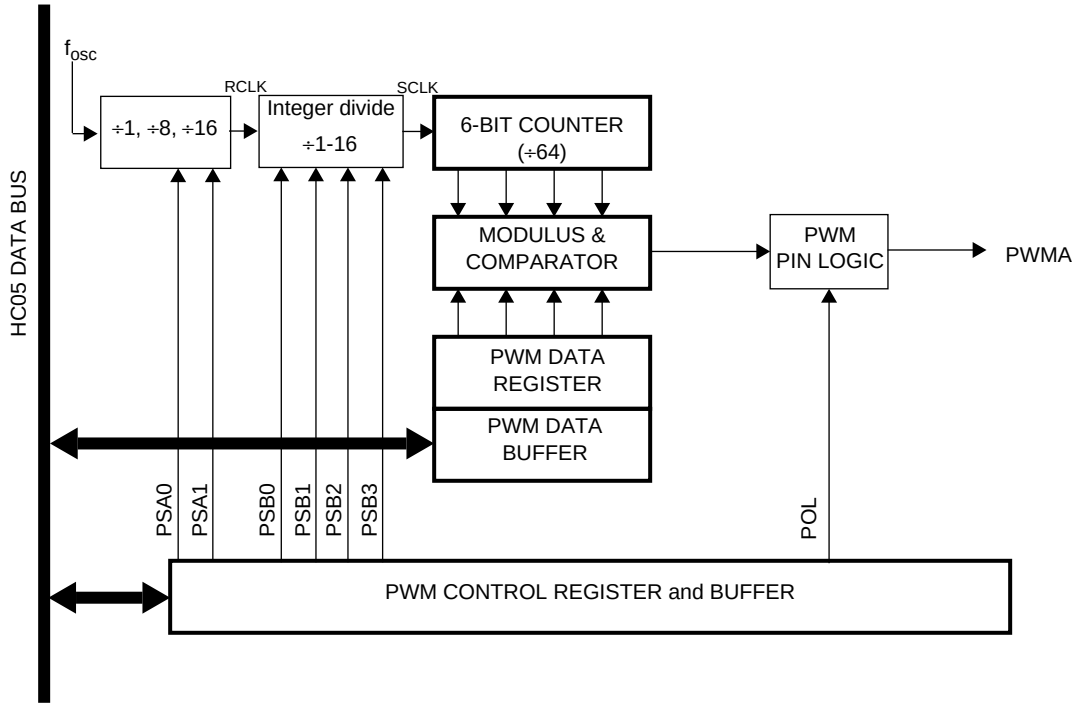


Figure 13-1: PWM Block Diagram

13.1 FUNCTIONAL DESCRIPTION

The PWM is capable of generating signals from 0% to 100% duty cycle. A $00 in the PWM Data Register yields an “low” output (0%), but a $3F yields a duty of 63/64. To achieve the 100% duty (“high” output), the polarity control bit is set to zero while the data register has $00 in it.

When not in use, the PWM system can be shut off to save power by clearing the clock rate select bits PSA0 and PSA1 in PWMCR.

Writes to the PWM data register are buffered and can therefore be performed at any time without affecting the output signal. When the PWM subsystem is enabled, a write to the PWM Control Register will become effective immediately.

When the PWM subsystem is enabled, a write to the PWM Data Register will not become effective until the end of the current PWM period has occurred, at which time the new data value is loaded into the PWM Data Register.

However, should a write to the registers be performed when the PWM subsystem is disabled, the data is transferred immediately. All Registers are updated after the PWM Data Register is written to and the end of a PWM cycle occurs.

The PWM output can have an active high or an active low pulse under software control using the POL (polarity) bit as shown in Figure 13-2 and Figure 13-3.

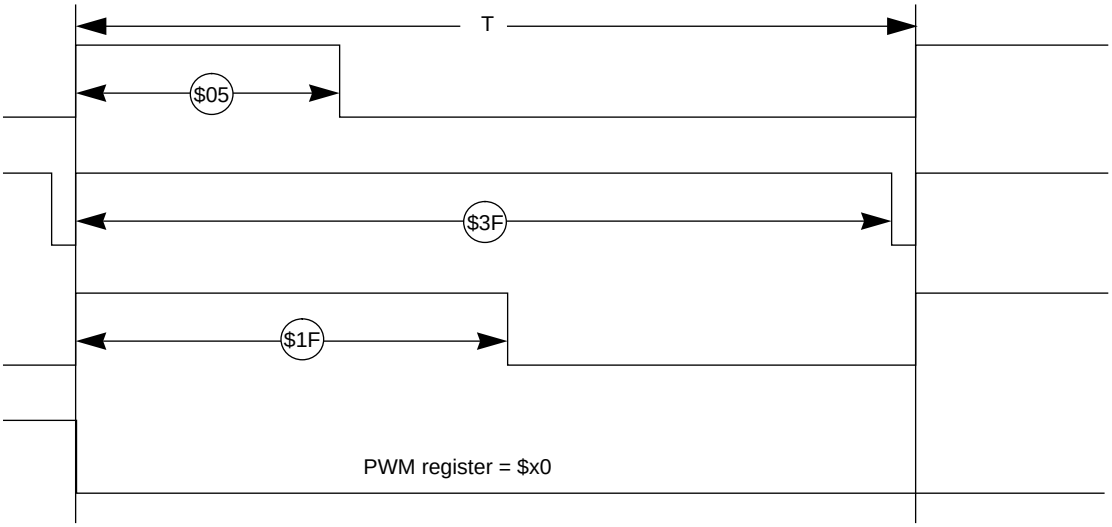


Figure 13-2: PWM Waveform Examples (POL = 1)

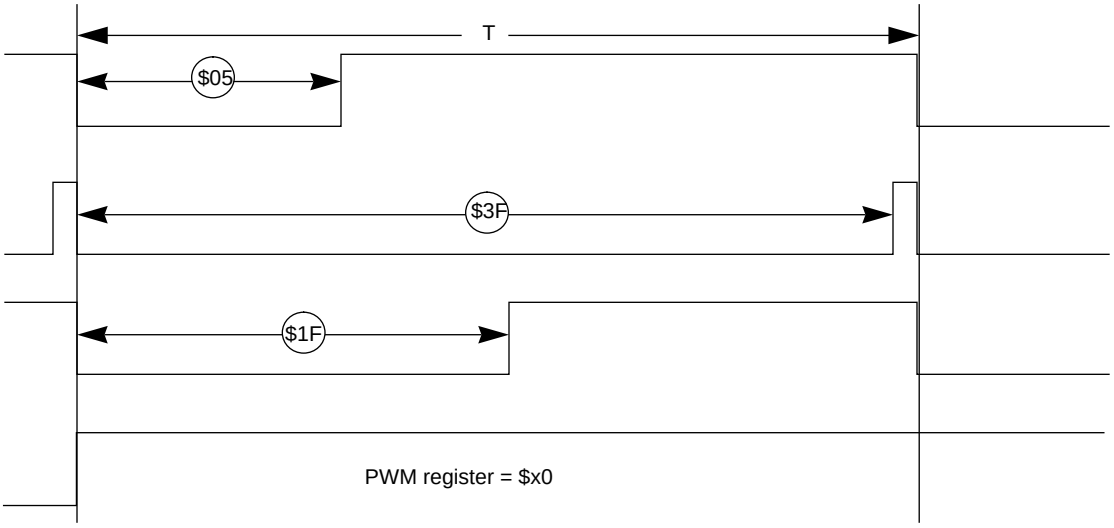


Figure 13-3: PWM Waveform Examples (POL = 0)

13.2 REGISTERS

Associated with the PWM system, there is a PWM data register and a control register. The following registers can be written to and read at any time. Data written to the data register is held in a buffer and transferred to the PWM Data Register at the end of a PWM cycle.

Reads of this register will always result in the read of the PWM Data Register and not the buffer.

Upon RESET the user should write to the data register prior to enabling the PWM system (for example, prior to setting the PSA and PSB bits for PWM input clock rate). This will avoid an erroneous duty cycle from being driven. During regular user mode the user should write to the PWM Data data register after writing the PWM Control Register.

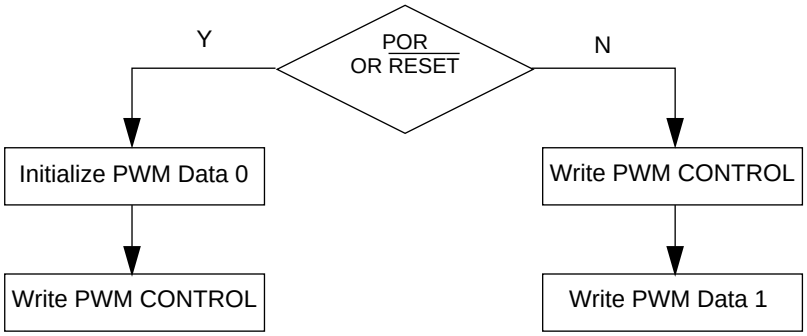


Figure 13-4: PWM Write Sequences

13.2.1 PWM CONTROL

$31	PSA1	PSA0	--	--	PSB3	PSB2	PSB1	PSB0
RESET:	0	0	--	--	0	0	0	0

Figure 13-5: PWM Control Register

PSA1, PSA0, PSB3-PSB0- PWM Clock Rate

These bits select the input clock rate and determines the period, as shown in Table 13-1. Note that some output frequencies can be obtained with more than one combination of PSA

Table 13-1: PWM Clock Rate

PSA1:PSA0	PSB3-PSB0	RCLK	SCLK	PWM OUT
00	xxxx	off	off	off
01	0000-1111	$f_{osc}/1$	$f_{osc}/1$-$f_{osc}/16$	$f_{osc}/64$-$f_{osc}/1024$
10	0000-1111	$f_{osc}/8$	$f_{osc}/8$-$f_{osc}/128$	$f_{osc}/512$-$f_{osc}/8192$
11	0000-1111	$f_{osc}/16$	$f_{osc}/16$-$f_{osc}/256$	$f_{osc}/1024$-$f_{osc}/16384$

and PSB values. For instance a PWM output of E/512 can be obtained with either PSA1:PSA0 = 10 and PSB3-PSB0 = 0001 or PSA1:PSA0 = 01 and PSB3-PSB0 = 1000. This scheme allows for 38 unique frequency selections.

13.2.2 PWM DATA REGISTERS

The PWM system has one 8-bit data register which hold the duty cycle for each PWM output in the least significant 6 bits. The data bits in this register are unaffected by RESET.

$30	POL	-	D5	D4	D3	D2	D1	D0
RESET:	1	-	-	-	-	-	-	-

Figure 13-6: PWM Data Register

POL - PWM Polarity

1 = PWM pulse is active high.
0 = PWM pulse is active low.

13.3 PWM DURING WAIT MODE

The PWM continues normal operation during WAIT mode. To decrease power consumption during WAIT, it is recommended that the rate select bits in the PWM Control Register be cleared if the PWM is not being used.

13.4 PWM DURING STOP MODE

In STOP mode the oscillator is stopped causing the PWM to cease functioning. Any signal in process is aborted in whatever phase the signal happens to be in.

13.5 PWM DURING RESET

Upon RESET the PSA0 and PSA1 bits in PWM CONTROL are cleared. This disables the PWM system and sets the PWMA output low. The user should write to the data registers prior to enabling the PWM system (for example, prior to setting PSA1 or PSA0). This will avoid an erroneous duty cycle from being driven.

SECTION 14 SERIAL PERIPHERAL INTERFACE

The Serial Peripheral Interface (SPI) is an interface which allows several MC68HC05 MCUs, or MC68HC05 MCU plus peripheral devices, to be interconnected within a single printed circuit board. In an SPI, separate wires are required for data and clock. In the SPI format, the clock is not included in the data stream and must be furnished as a separate signal. An SPI system may be configured in one containing one master MCU and several slave MCUs, or in a system in which an MCU is capable of being a master or a slave.

Features include:

- Full duplex, three-wire synchronous transfers
- Master or slave operation
- Internal MCU clock divided by 2 (maximum) master bit frequency
- Internal MCU clock (maximum) slave bit frequency
- Four programmable master bit rates
- Programmable clock polarity and phase
- End of transmission interrupt flag
- Write collision flag protection
- Master-Master mode fault protection capability

14.1 SPI SIGNAL DESCRIPTION

The four basic signals (MOSI, MISO, SCK, $\overline{SS}$) are described in the following paragraphs. Each signal function is described for both the master and slave mode.

The SPI forces the direction on some of the pin to output in order to function properly.

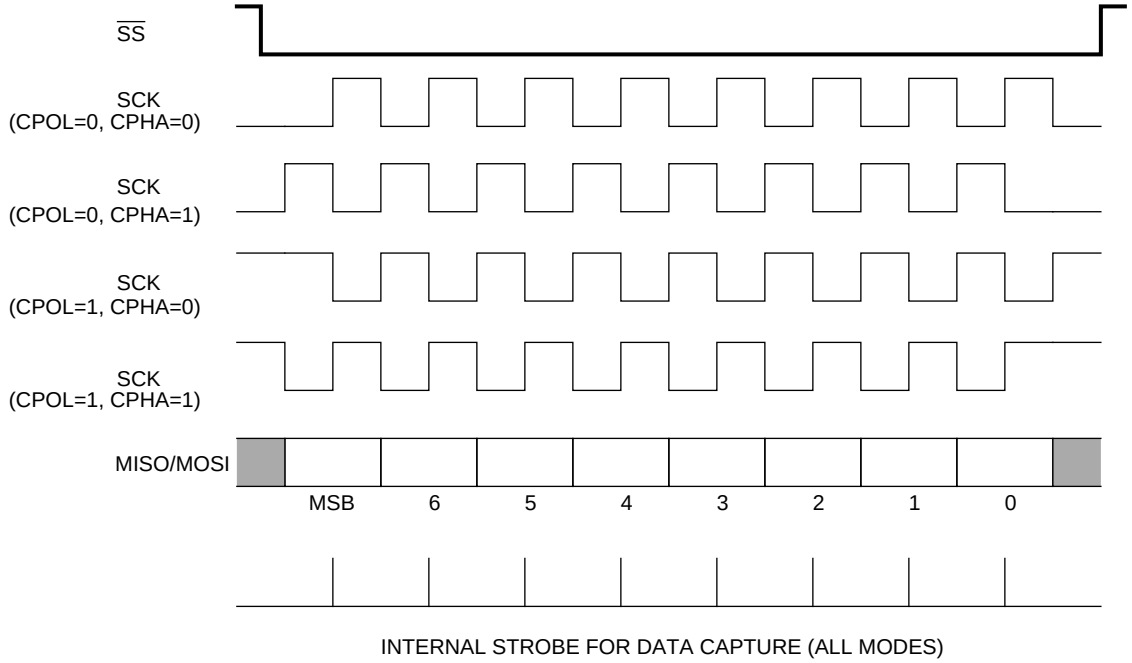


Figure 14-1: Data Clock Timing Diagram

14.1.1 Master In Slave Out (MISO/PF3)

The MISO line is configured as an input in a master device and as an output in a slave device. It is one of the two lines that transfer serial data in one direction, with the most significant bit sent first. The MISO line of a slave device is placed in the high-impedance state if the slave is not selected.

14.1.2 Master Out Slave In (MOSI/PF2)

The MOSI line is configured as an output in a master device and as an input in a slave device. It is one of the two lines that transfer serial data in one direction with the most significant bit sent first.

14.1.3 Serial Clock (SCK/PF1)

The master clock is used to synchronize data movement both in and out of the device through its MOSI and MISO lines. The master and slave devices are capable of exchanging a byte of information during a sequence of eight clock cycles. Since SCK is generated by the master device, this line becomes an input on a slave device.

As shown in Figure 14-1, four possible timing relationships may be chosen by using control bits CPOL and CPHA in the serial peripheral control register (SPCR). Both master and slave devices must operate with the same timing. The master device always places data on the MOSI line a half-cycle before the clock edge (SCK), in order for the slave device to latch the data.

Two bits (SPR0 and SPR1) in the SPCR of the master device select the clock rate. In a slave device, SPR0 and SPR1 have no effect on the operation of the SPI.

14.1.4 Slave Select ($\overline{SS}$/PF0)

The slave select ($\overline{SS}$) input line is used to select a slave device. It has to be low prior to data transactions and must stay low for the duration of the transaction.The $\overline{SS}$ line on the master must be tied high. If it goes low, a mode fault error flag (MODF) is set in the SPSR.

When CPHA=0, the shift clock is the OR of $\overline{SS}$ with SCK. In this clock phase mode, $\overline{SS}$ must go high between successive characters in an SPI message. When CPHA=1, $\overline{SS}$ may be left low for several SPI characters. In cases where there is only one SPI slave MCU, its $\overline{SS}$ line could be tied to V_{SS} as long as CPHA=1 clock modes are used. The $\overline{SS}$ pin is shared with the PF0 pin. See **9.4 PORT F** for additional information on the $\overline{SS}$ pin.

14.2 FUNCTIONAL DESCRIPTION

Figure 14-2 shows a block diagram of the serial peripheral interface circuitry. When a master device transmits data to a slave via the MOSI line, the slave device responds by sending data to the master device via the master's MISO line. This implies full duplex transmission with both data out and data in synchronized with the same clock signal. Thus, the byte transmitted is replaced by the byte received and eliminates the need for separate transmit-empty and receive-full status bits. A single status bit (SPIF) is used to signify that the I/O operation has been completed.

The SPI is double buffered on read, but not on write. If a write is performed during data transfer, the transfer occurs uninterrupted, and the write will be unsuccessful. This condition will cause the write collision (WCOL) status bit in the SPSR to be set. After a data byte is shifted, the SPIF flag of the SPSR is set.

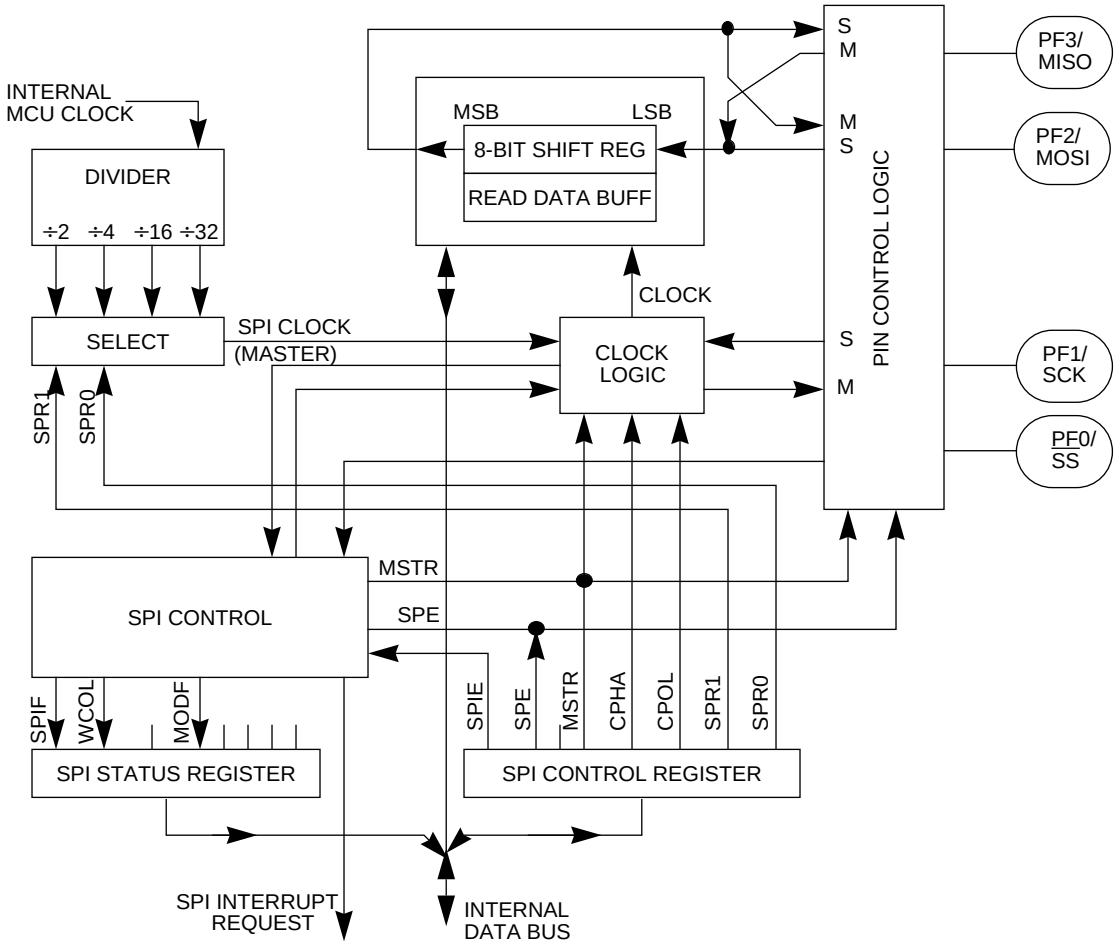


Figure 14-2: Serial Peripheral Interface Block Diagram

In the master mode, the SCK pin is an output. It idles high or low, depending on the CPOL bit in the SPCR, until data is written to the shift register, at which point eight clocks are generated to shift the eight bits of data and then SCK goes idle again.

In a slave mode, the slave select start logic receives a logic low at the $\overline{SS}$ pin and a clock at the SCK pin. Thus, the slave is synchronized with the master. Data from the master is received serially at the MOSI line and loads the 8-bit shift register. After the 8-bit shift register is loaded, its data is parallel transferred to the read buffer. During a write cycle, data is written into the shift register, then the slave waits for a clock train from the master to shift the data out on the slave's MISO line.

Figure 14-3 illustrates the MOSI, MISO, SCK, and $\overline{SS}$ master-slave interconnections.

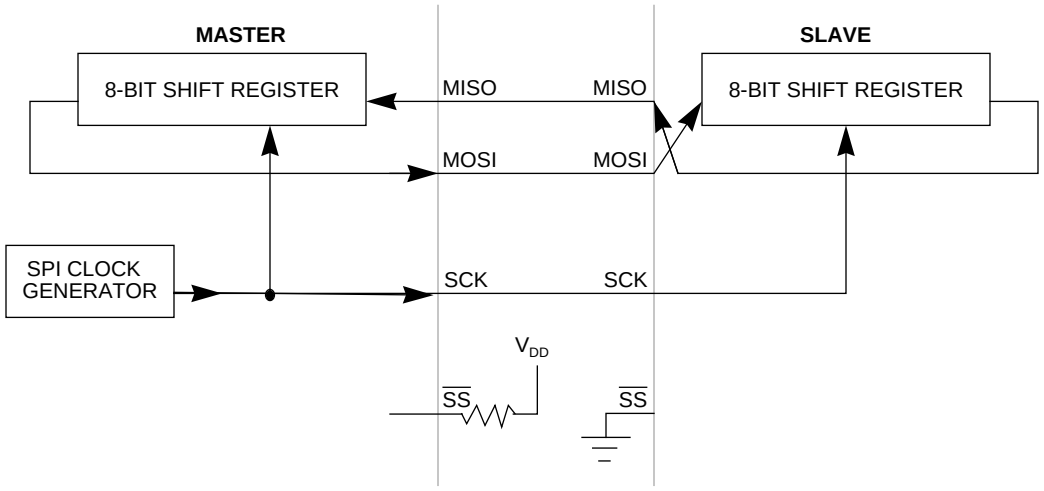


Figure 14-3: Serial Peripheral Interface Master-Slave Interconnection

14.3 SPI REGISTERS

There are three registers in the SPI which provide control, status, and data storage functions. These registers are called the serial control register (SPCR), serial peripheral status register (SPSR), and serial peripheral data I/O register (SPDR) and are described in the following paragraphs.

14.3.1 Serial Peripheral Control Register (SPCR)

$0A	SPIE	SPE	—	MSTR	CPOL	CPHA	SPR1	SPR0
RESET:	0	0	0	0	0	1	U	U

Figure 14-4: SPI Control Register (SPCR)

SPIE - Serial Peripheral Interrupt Enable

0 = SPIF interrupts disabled
1 = SPI interrupt is SPIF=1

SPE - Serial Peripheral System Enable

0 = SPI system off
1 = SPI system on. Port F becomes SPI pins. $\overline{SS}$ is a special case. See **9.4 PORT F** for additional information.

MSTR - Master Mode Select

0 = Slave mode
1 = Master mode

CPOL - Clock Polarity

When the clock polarity bit is cleared and data is not being transferred, a steady state low value is produced at the SCK pin of the master device. Conversely, if this bit is set, the SCK pin will idle high. This bit is also used in conjunction with the clock phase control bit to produce the desired clock-data relationship between master and slave. **Figure 14-1 : Data Clock Timing Diagram.**

CPHA - Clock Phase

The clock phase bit, in conjunction with the CPOL bit, controls the clock-data relationship between master and slave. The CPOL bit can be thought of as simply inserting an inverter in series with the SCK line. The CPHA bit selects one of two fundamentally different clocking protocols. When CPHA=0, the shift clock is the OR of SCK with $\overline{SS}$. As soon as $\overline{SS}$ goes low, the transaction begins and the first edge on SCK invokes the first data sample. When CPHA=1, the $\overline{SS}$ pin may be thought of as a simple output enable control. **Figure 14-1 : Data Clock Timing Diagram.**

SPR1 and SPR0 - SPI Clock Rate Selects

These two bits select one of four baud rates (Figure 14-5) to be used as SCK if the device is a master; however, they have no effect in the slave mode.

SPR1	SPR0	INT MCU CLOCK DIVIDED BY
0	0	2
0	1	4
1	0	16
1	1	32

Figure 14-5: Serial Peripheral Rate Selection

14.3.2 Serial Peripheral Status Register (SPSR)

$0B	SPIF	WCOL	—	MODF	—	—	—	—
RESET:	0	0	0	0	0	0	0	0

Figure 14-6: SPI Status Register (SPSR)

SPIF - SPI Transfer Complete Flag

The serial peripheral data transfer flag bit is set upon completion of data transfer between the processor and external device. If SPIF goes high, and if SPIE is set, a serial peripheral interrupt is generated. Clearing the SPIF bit is accomplished by reading the SPSR (with SPIF set) followed by an access of the SPDR. Unless SPSR is read (with SPIF set) first, attempts to write to SPDR are inhibited.

WCOL - Write Collision

The write collision bit is set when an attempt is made to write to the serial peripheral data register while data transfer is taking place. If CPHA is zero, a transfer is said to begin when $\overline{SS}$ goes low and the transfer ends when $\overline{SS}$ goes high after eight clock cycles on SCK. When CPHA is one, a transfer is said to begin the first time SCK becomes active while $\overline{SS}$ is low and the transfer ends when the SPIF flag gets set. Clearing the WCOL bit is accomplished by reading the SPSR (with WCOL set) followed by an access to SPDR.

Bit 5 - Not implemented

This bit always reads zero.

MODF - Mode Fault

The mode fault flag indicates that there may have been a multi-master conflict for system control and allows a proper exit from system operation to a reset or default system state. The MODF bit is normally clear, and is set only when the master device has its $\overline{SS}$ pin pulled low. Setting the MODF bit affects the internal serial peripheral interface system in the following ways:

- An SPI interrupt is generated if SPIE=1.
- The SPE bit is cleared. This disables the SPI.
- The MSTR bit is cleared, thus forcing the device into the slave mode.

Clearing the MODF bit is accomplished by reading the SPSR (with MODF set), followed by a write to the SPCR. Control bits SPE and MSTR may be restored to their original state by user software after the MODF bit has been cleared. It is also necessary to restore DDRD after a mode fault.

Bits 3-0 - Not Implemented

These bits always read zero.

14.3.3 Serial Peripheral Data I/O Register (SPDR)

$0C	SPD7	SPD6	SPD5	SPD4	SPD3	SPD2	PSD1	SPD0
RESET:	-	-	-	-	-	-	-	-

The serial peripheral data I/O register is used to transmit and receive data on the serial bus. Only a write to this register will initiate transmission/reception of another byte, and this will only occur in the master device. At the completion of transmitting a byte of data, the SPIF status bit is set in both the master and slave devices.

When the user reads the serial peripheral data I/O register, a buffer is actually being read. The first SPIF must be cleared by the time a second transfer of the data from the shift register to the read buffer is initiated, or an overrun condition will exist. In cases of overrun, the byte that causes the overrun is lost.

A write to the serial peripheral data I/O register is not buffered and places data directly into the shift register for transmission.

14.4 SPI IN STOP MODE

When the MCU enters the STOP mode, the baud rate generator which drives the SPI shuts down. This essentially stops all master mode SPI operation; thus, the master SPI is unable to transmit or receive any data. If the STOP instruction is executed during an SPI transfer, that transfer is halted until the MCU exits the stop mode (provided it is an exit resulting from a viable interrupt source). If the stop mode is exited by a reset, then the appropriate control/ status bits are cleared and the SPI is disabled. If the device is in the slave mode when the STOP instruction is executed, the slave SPI will still operate. It can still accept data and clock information in addition to transmitting its own data back to a master device.

At the end of a possible transmission with a slave SPI in the stop mode, no flags are set until a viable interrupt results in an MCU "wake up". Caution should be observed when operating the SPI (as a slave) during the stop mode because none of the protection circuitry (write collision, mode fault, etc.) is active.

It should also be noted that when the MCU enters the stop mode all enabled output drivers (MISO, MOSI, and SCLK ports) remain active and any sourcing currents from these outputs will be part of the total supply current required by the device.

14.5 SPI IN WAIT MODE

The SPI subsystem remains active in wait mode. Therefore, it is consuming power. If it is desired to reduce power, the SPI should be shut off prior to entering wait mode. A non-resetexit from wait mode will result in the state of the SPI being unchanged. A reset exit will return the SPI to its reset state, which is disabled.

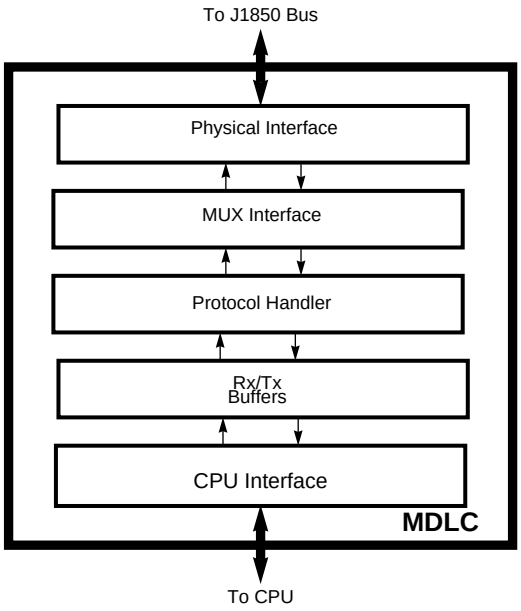
SECTION 15 MESSAGE DATA LINK CONTROLLER

The Message Data Link Controller (MDLC) provides access to an external serial communication multiplex bus, operating according to the SAE J1850 protocol.

Key features of the MDLC include:

- SAE J1850 compatible
- GM Class 2 compatible
- 10.4 kbps VPW bit format
- Digital noise filter
- Collision detection
- Two 11-byte Receive Buffers, one 11-byte Transmit Buffer
- Hardware CRC generation & checking
- Two power saving modes with automatic wake up on network activity
- Polling and CPU interrupts available
- Receive Block mode supported
- Coexists with devices supporting 4x mode
- Auto retry following loss of arbitration
- In-frame Response **not** supported
- Low Voltage Protection

15.1 OUTLINE



The CPU Interface contains the software addressable registers and provides the link between the CPU and the Tx and Rx Buffers. The Tx and Rx Buffers provide storage for data received and data to be transmitted onto the J1850 bus. The Protocol Handler is responsible for the encoding and decoding of data bits and special message symbols during transmission and reception. The MUX Interface provides the link between the MDLC digital section and the analog Physical Interface. The wave shaping, driving, and digitizing of data is performed by the Physical Interface.

NOTE: The bus data rate depends upon the microcontroller oscillator frequency (f_{OSC}) and the MDLC rate selection control bits (R0, R1). The correct combination for the application must be chosen in order for J1850 bus communications to take place. See **15.2.2.3 R1, R0 - Rate Select**.

15.1.1 MDLC OPERATING MODES

The MDLC has five main modes of operation which interact with the power supplies, pins and rest of the MCU as shown below.

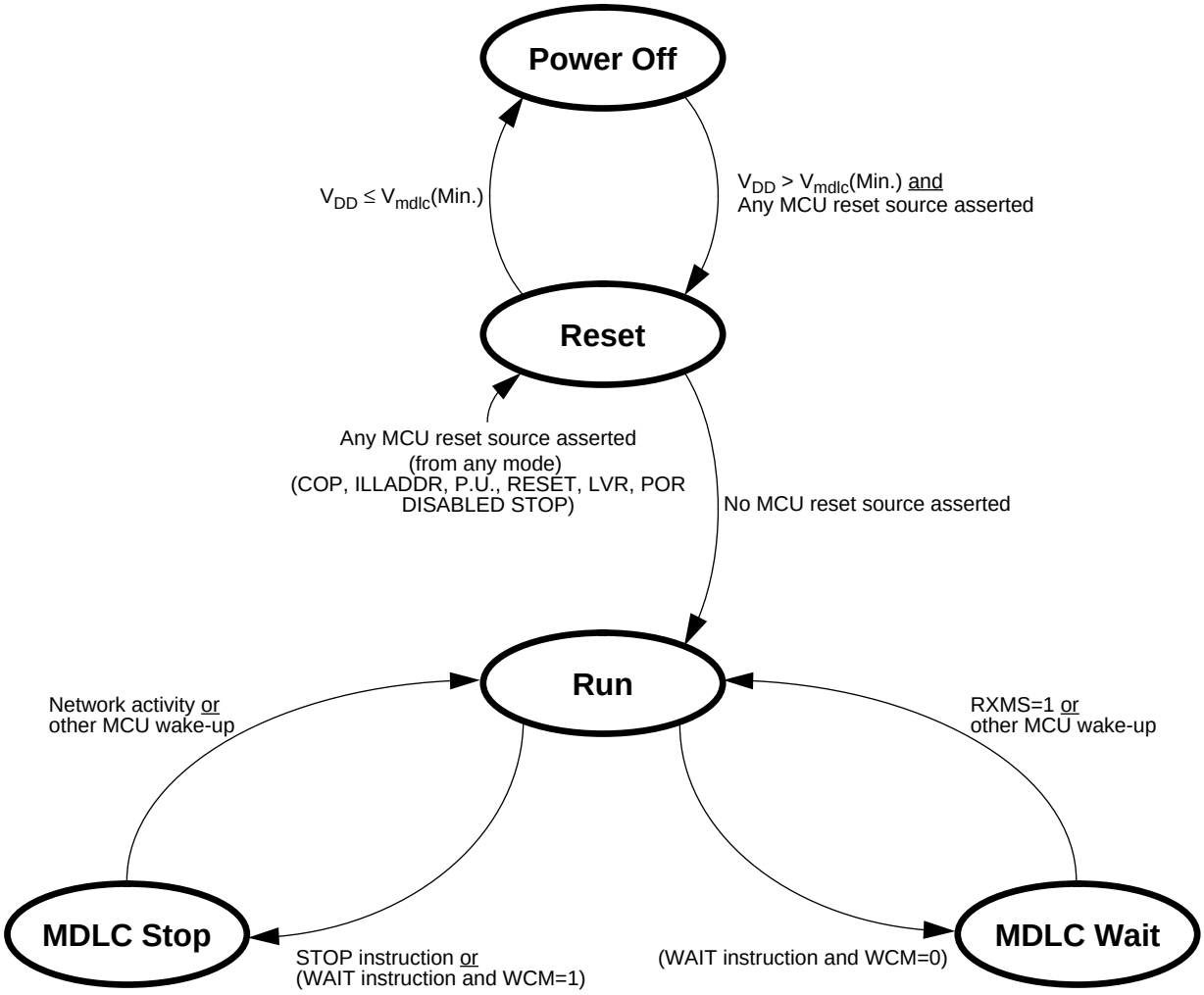


Figure 15-1: MDLC Operating Modes State Diagram

15.1.2 MODE DESCRIPTIONS

15.1.2.1 Power Off

This mode is entered from the Reset mode whenever the MCU supply voltage V_{DD} drops below the minimum specified value to guarantee correct MDLC operation. In order to ensure that the MDLC does not enter an unknown state, it will first be placed in the Reset mode before being powered down. In this mode, the pin input and output specifications are not guaranteed.

15.1.2.2 Reset

This mode is entered from the Power Off mode whenever the MCU supply voltage V_{DD} rises above the minimum specified value and some MCU reset source is asserted. In order to prevent an unknown state from being entered and to guarantee correct operation, the internal MCU reset will be asserted while the MDLC module is being powered up.

This mode is entered from any other mode whenever the MCU supply voltage V_{DD} drops below the minimum value for correct MDLC operation. When this occurs, the MDLC module will reenter the Reset mode before being powered down to prevent an unknown state from being entered.

The Reset mode is also entered from any other mode as soon as one of the MCU's possible reset sources (for example, POR, COP watchdog, Reset pin etc.) is asserted.

In this mode, the internal MDLC voltage references are operative, V_{DD} is supplied to the internal circuits, which are held in their reset state and the internal MDLC system clock is running. Registers will assume their reset condition. Outputs are held in their programmed reset state, inputs and network activity are ignored.

15.1.2.3 Run

This mode is entered from the Reset mode after all MCU reset sources are no longer asserted. It is entered from the MDLC Wait mode whenever a message is successfully received.

It is entered from the MDLC Stop mode whenever network activity is sensed though messages will not be received properly until the clocks have stabilized and the CPU is also in the Run mode.

In this mode, normal network operation takes place. The user should ensure that all MDLC transmissions have ceased before exiting this mode.

15.1.2.4 MDLC Wait

This power conserving mode is automatically entered from the Run mode whenever the CPU executes a WAIT instruction and if the WCM bit in the MCR register is previously cleared.

In this mode, the MDLC internal clocks continue to run but the physical interface circuitry is placed in a low power mode and awaits a valid network message. If a valid network message is successfully received (RXMS=1) a CPU interrupt request will be generated.

15.1.2.5 MDLC Stop

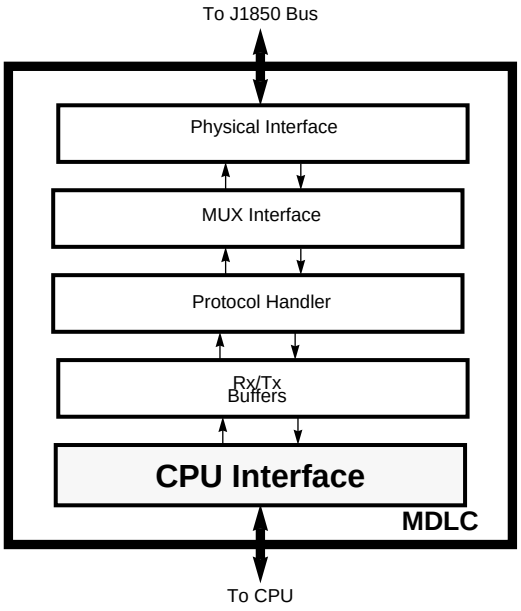
This power conserving mode is automatically entered from the Run mode whenever the CPU executes a STOP instruction, or if the CPU executes a WAIT instruction and the WCM bit in the MCR register is previously set.

In this mode, the MDLC internal clocks are stopped but the physical interface circuitry is placed in a low power mode and awaits network activity. If network activity is sensed, then a CPU interrupt request will be generated, restarting the MDLC internal clocks.

15.2 MDLC CPU INTERFACE

15.2.1 OUTLINE

The CPU Interface provides the interface between the CPU and the MDLC. It consists of 4 user registers. A full description of each register follows.



Addr	Name
$000E	MCR
$000F	MSR
$0010	MTCR
$0011	MRSR

Figure 15-2: MDLC User Registers

15.2.2 MDLC CONTROL REGISTER (MCR) $0E

This register is used to configure and control the MDLC.

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	RXBM	TXAB	R1	R0	-	-	IE	WCM
RESET	0	0	1	0	0	0	0	0

Figure 15-3: MDLC Control Register (MCR)

All bits may be read in all modes of MCU operation.

Bits 0, 4 and 5 may be written to only once after reset after which they become read only bits.

Bit 1 may always be written.

Bits 6 and 7 may only be set by the CPU, clears are ignored. These bits are cleared automatically by the MDLC when the appropriate action is completed.

15.2.2.1 RXBM - Receive Block Mode

This bit is set to indicate to the MDLC that the next incoming message will be a block message. (See **15.7.4 RECEIVING A MESSAGE IN BLOCK MODE**).

If this bit is set and the Rx Buffer being loaded with data bytes received from the multiplex bus is filled or the last byte of the current block message has been received, a CPU interrupt request is generated (if the IE bit is set) and the Rx Message Successful (RXMS) bit will be set, just as in normal message reception. Additional incoming bytes will be placed into the next available Rx Buffer.

The RC3-0 bits of the MRSR register only reflect the byte count of the Rx Buffer available to the CPU and **not** a cumulative value of the number of bytes in the block message.

Once set, this bit can only be cleared automatically by the MDLC. It will be cleared following the successful reception of a message or immediately upon the receiver detecting an error (CRC, invalid bit etc.), reverting the MDLC to its usual mode of operation in which normal message lengths are expected. This bit will also be cleared if both Rx Buffers fill, they are not serviced and further data bytes are received (Block mode 'overflow' occurs).

The receiver will compute a cumulative CRC throughout the reception of a block message, and will not treat the CRC calculation as complete until an End of Data (EOD) symbol is received. At this time the MDLC will automatically clear the RXBM bit indicating to the programmer that the block message has ended.

15.2.2.2 TXAB - Transmit Abort

This bit is set by the programmer to abort an in-progress transmission. The subsequent actions depend upon what the transmitter is doing at the point in time when the CPU interface recognizes this command.

If this command is recognized by the MDLC while one of the first seven bits of a byte is being transmitted (non-byte boundary) then the transmission of the bit will be completed and no further bits will be sent.

If this command is recognized while the last bit of a byte is being transmitted (byte boundary) then the transmission of the bit will first be completed. In order to avoid receiving nodes from interpreting the aborted transmission at a byte boundary as being the end of a normal message, the MDLC will then send two extra logic one bits.

This guarantees that no aborted message will be an integral number of bytes in length and cannot be mistaken for a valid transmission.

Arbitration will be performed on any extra bits sent, allowing any other nodes which are also transmitting to continue. Should the MDLC lose arbitration during the first extra logic one sent, the second extra logic one will not be sent and no further attempts to send the message will be made.

If this command is recognized after a write to the MDLC Transmit Control Register (MTCR), thus initiating a transmission, but before the message has begun to be sent on the J1850 bus, then no attempt to transmit that message will be made until a subsequent write to the MTCR is made.

If this command is recognized between automatic transmission retry attempts, such as those following a loss of arbitration, all further attempts to resend the message will be suppressed.

This bit is automatically cleared after the abort takes place or after it is determined that the MDLC was not transmitting anyway. Refer to **Table 15-1: MDLC Transmit Abort Function Summary** for a summary of the operation of the TXAB bit.

The programmer must never write to the MTCR register while the TXAB bit is set, as this may have unpredictable results.

Table 15-1: MDLC Transmit Abort Function Summary

Transmit Abort Command	MDLC Action
Command recognized on non-byte boundary	MDLC will terminate transmission immediately
Command recognized on byte boundary	MDLC will transmit up to two extra ones
Command recognized before the message has started on the J1850 bus	MDLC will not transmit the message
Command recognized during retry	MDLC will not retry to transmit the message

15.2.2.3 R1, R0 - Rate Select

These bits determine the operating frequency of the MDLC.

The nominal frequency (f_{MDLC}) of the MUX Interface clock **must** always be 1.048576 MHz in order for J1850 bus communication to take place. Therefore, the value programmed into these bits is dependant upon the chosen MCU system clock frequency per the following table.

Table 15-2: MDLC Rate Selection

Clock Frequency	R1	R0	Division	f_{MDLC}
f_{osc} =1.048576 MHz (f_{OP} =524Khz)	0	0	1	1.048576 MHz
f_{OP} =1.048576 MHz	0	1	1	1.048576 MHz
f_{OP} =2.097152 MHz	1	0	2	1.048576 MHz
f_{OP} =4.194304 MHz	1	1	4	1.048576 MHz

In the case of R1=R0=0, the oscillator clock is used instead of E clock.

The case of R1=R0=1 must **not** be selected for the MC68HC05V7 and MC68HC705V8.

15.2.2.4 IE - Interrupt Enable

This bit determines whether the MDLC will generate CPU interrupt requests in the Run mode. Clearing the IE bit will **not** prevent CPU interrupt requests when exiting the MDLC Stop or MDLC Wait modes.

When set, the MDLC will generate CPU interrupt requests. Interrupt requests will be maintained until all of the interrupt request sources are cleared, by performing the specified actions upon the MDLC's registers.

When clear, the MDLC will not generate CPU interrupt requests. Interrupts that were pending at the time that this bit is cleared may be lost.

15.2.2.5 WCM - Wait Clock Mode

This bit determines the operation of the MDLC during CPU Wait mode. See **15.7.6 MDLC WAIT MODE** for more details on its use.

When set, the MDLC internal operating clocks will be stopped during CPU Wait mode.

When clear, the MDLC internal operating clocks will stay running during CPU Wait mode.

15.2.3 MDLC STATUS REGISTER (MSR) $0F

This register indicates the transmit and receive status of the MDLC.

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	0	0	0	0	TXMS	RXMS	0	0
RESET	0	0	0	0	0	0	0	0

Figure 15-4: MDLC Status Register (MSR)

All bits may be read in all modes of MCU operation.

Bits 0,1, 4, 5, 6, and 7 will always read as zeros and can never be written to.

Bits 2 and 3 are reflective of CPU interrupt request signals and are asserted as long as their respective CPU interrupt request is pending. Neither bit is affected by CPU interrupt requests generated when the MDLC 'wakes up' from MDLC Stop or MDLC Wait mode.

All writes to this register are ignored in all modes of MCU operation.

15.2.3.1 TXMS - Transmitted Message Successfully

When set, this bit indicates that the Tx Buffer contents have been sent successfully.

An access of the Tx Control Register (MTCR) will clear this bit.

15.2.3.2 RXMS - Received Message Successfully

When set this bit indicates that one of the Rx Buffers contains a new message.

This bit can only be cleared if both Rx Buffers are empty. If only one message has been received into an Rx Buffer by the MDLC, any access (read or write) of the MDLC Rx Status Register (MRSR) will clear the RXMS bit.

If a second message is received after the RXMS bit has been cleared by the CPU as it is retrieving the first message, the RXMS bit will immediately be set again, and will remain set until the MRSR is accessed once the last message received is made available to the CPU.

If both Rx Buffers contain a received message, the RXMS bit will simply remain set until the MRSR register is accessed once the last message received is made available to the CPU.

15.2.3.3 TXMS and RXMS Interrupts

If either the TXMS or the RXMS bit is set, and if interrupts are enabled, an interrupt request will be made to the CPU. The interrupt service routine may poll these bits to establish the cause of the interrupt. The mechanism for clearing each of these bits will also clear the associated CPU interrupt request.

Alternatively, if interrupts are disabled, the CPU may poll these bits periodically to see if a change in status has occurred.

The MDLC will attempt to receive every message that it sends, so the RXMS bit will usually be set a short time after the TXMS bit is set.

Because these bits act independently, cannot be written to, and have separate clearing mechanisms, it is not possible to inadvertently miss an interrupt.

15.2.3.4 Clearing Wake Up Interrupts

Although no flag bit is affected by the MDLC waking up from MDLC Stop Mode (entered by 'STOP' or 'WAIT' with WCM bit set previously) due to network activity, the CPU interrupt request that is generated by the wake up is cleared by an access of the MRSR register.

15.2.4 MDLC TX CONTROL REGISTER (MTCR) $10

This register controls the operation of the MDLC's transmitter, including the Tx Buffer.

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	0	0	0	0	TC3	TC2	TC1	TC0
RESET	0	0	0	0	0	0	0	0

Figure 15-5: MDLC Tx Control Register (MTCR)

All bits may be read in all modes of MCU operation.

Bits 4, 5, 6, and 7 will always read as zeros and can never be written to.

Bits 0, 1, 2, and 3 can be written to in all modes of MCU operation.

15.2.4.1 TC0,1,2,3 - Transmit Count

These bits determine the length of the message body (not including the CRC byte) to be sent. Internally, they are reset to $00 following a reset.

The programmer should first determine that the MDLC is ready to transmit, then load the message header and data bytes into the Tx Buffer, and finally write the length of the body of the message (excluding CRC byte) into this register.

This will cause the MDLC to initiate transmission of the contents of the Tx Buffer according to the J1850 protocol. Once the last byte has been sent, a Cyclic Redundancy Check (CRC) byte will automatically be appended to the end of the message body.

Once the CRC has been successfully sent, a CPU interrupt request will be generated (if the Interrupt Enable (IE) bit is set) and the Tx Message Successful (TXMS) bit will be set in the MSR register.

An access of this register will clear the CPU interrupt request and the TXMS bit.

The valid range of values that may be written to this register is $01 to $0B. Since the Tx Buffer is 11 bytes long, any value greater than or equal to $0C written to this register will create a Tx Buffer overflow error and cause the MDLC to not transmit that message.

Attempts to send a zero byte message body length (value of $00 written to MTCR) will also prevent the MDLC from transmitting that message.

Writing a value within the valid range to this register will initiate a transmission regardless of whether any new data has been loaded into the Tx Buffer. Failure to supply new data before this register is written to will result in the MDLC transmitting whatever data happens to be in the Tx Buffer at the time of the write.

Do not write to this register while the TXAB bit in the MCR register is set.

This register will be automatically cleared at the end of any transmission, whether it was successful, unsuccessful or aborted.

15.2.5 MDLC RX STATUS REGISTER (MRSR) $11

This register reports the status of the MDLC's receiver, including the Rx Buffer.

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	0	0	0	0	RC3	RC2	RC1	RC0
RESET	0	0	0	0	0	0	0	0

Figure 15-6: MDLC Rx Status Register (MRSR)

All bits may be read in all modes of MCU operation.

Bits 4, 5, 6, and 7 will always read as zeros and can never be written to.

Bits 0, 1, 2, and 3 can be written to in all modes of MCU operation.

15.2.5.1 RC0,1,2,3 - Receive Count

These bits reflect the number of bytes of the message body in the Rx Buffer available to the CPU. As a message is being received, the data is placed into successive locations of one of the two Rx Buffers. The Rx Buffer being filled is not visible to the programmer.

A running count of the number of bytes received is kept internally. The maximum count is 11 bytes. The CRC is **not** placed in the Rx Buffer and is not counted in the final indicated message length.

Once the message has been received error-free, the Rx Buffer is placed into the MCU's memory map, the MRSR is updated with the count of the number of data bytes received, a CPU interrupt request is generated if the Interrupt Enable (IE) bit is set and the Rx Message Successful (RXMS) bit will be set in the MSR.

When the programmer has finished analyzing the received message, a write to this register will signal to the MDLC that the programmer no longer needs the contents of the Rx Buffer.

The Rx Buffer will then be released back to the MDLC. If another message is available in the second Rx Buffer at this time, the new message will then be made available to the CPU.

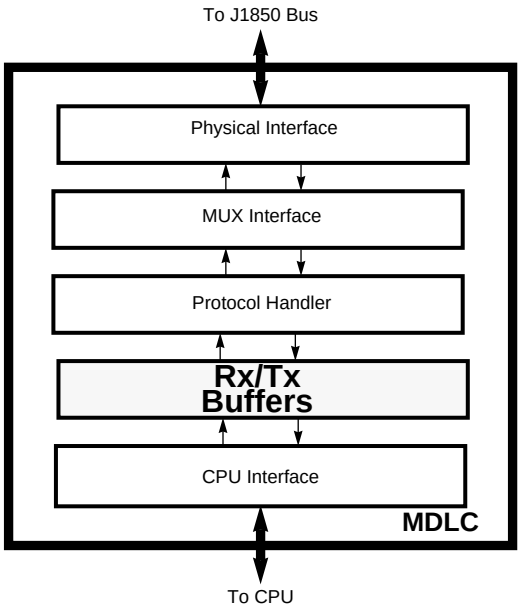
Following a write to the MRSR to release the Rx Buffer, this register should not be accessed again until the RXMS bit is again verified to be set. This procedure will ensure that the MRSR value does not change while it is being accessed by the CPU.

If, while performing the above operations, another message begins to arrive then the second message will be placed in a duplicate Rx Buffer. The MDLC will always present the oldest available message to the programmer, therefore only one Rx Buffer appears in the memory map while the other one is filled with a newer message.

If both Rx Buffers are filled when a third new message arrives, the third message will be ignored and the prior messages within the Rx Buffers will not be affected.

15.3 MDLC Rx/Tx BUFFERS

The Rx and Tx Buffers provide the interface between the CPU Interface and the Protocol Handler. They provide buffering of data being sent and received, reducing the frequency of interrupts to the CPU. Please refer to the MCU memory map for details on the starting and ending addresses of these buffers within the overall memory map.



15.3.1 OUTLINE

During all modes of MCU operation, the Rx and Tx Buffers are mapped as registers. Memory mapping these buffers requires the MDLC to supply the user with a count of the number of bytes in each received message and allows the user to tell the MDLC the length of a message to be transmitted.

Access arbitration is simplified by the following rules:

- The user must **not** read the Rx Buffer until the MDLC signals that an entire message has been received (Receive Message Successful bit (RXMS) in the MDLC Status Register (MSR) is set).
- The user returns the Rx Buffer resource to the MDLC by writing any value to the MDLC Receive Status Register (MRSR).
- The user must **not** write to the Tx Buffer after the MDLC has been told to transmit the message (length of message has been written to the MDLC Transmit Control Register (MTCR)).
- The user must wait until the MDLC has sent the message (indicated by a CPU interrupt request and the Transmit Message Successful (TXMS) bit being set), or the message has been aborted (Transmit Abort bit (TXAB) in the MDLC Control Register (MCR) is toggled) before reloading the Tx Buffer.

The MDLC has three 11 byte buffer arrays. The transmit circuitry uses a single Tx Buffer, that for certain periods of time may have either the CPU or the MDLC access "locked out". The receiver has two Rx Buffers but only one can appear in the MCU memory map at a time. The CPU will have access to whichever Rx Buffer was filled the earliest. While the CPU is reading data from one Rx Buffer, the MDLC may place incoming data into the other Rx Buffer. When the CPU writes to the MRSR register, the Rx Buffer which the CPU had access to is "given back" to the MDLC making it available for another incoming message.

The MDLC Tx Buffer is writable by the CPU at all times but it is not readable. A read of the Tx buffer will result in an unknown value. The Rx Buffers can be read by the CPU at anytime. However, writes to the Rx Buffer will not have an effect on the buffer’s content.

NOTE: The contents of the Tx Buffer are not guaranteed to remain intact following a transmission so the Tx Buffer should be completely reloaded for each new message to be sent.

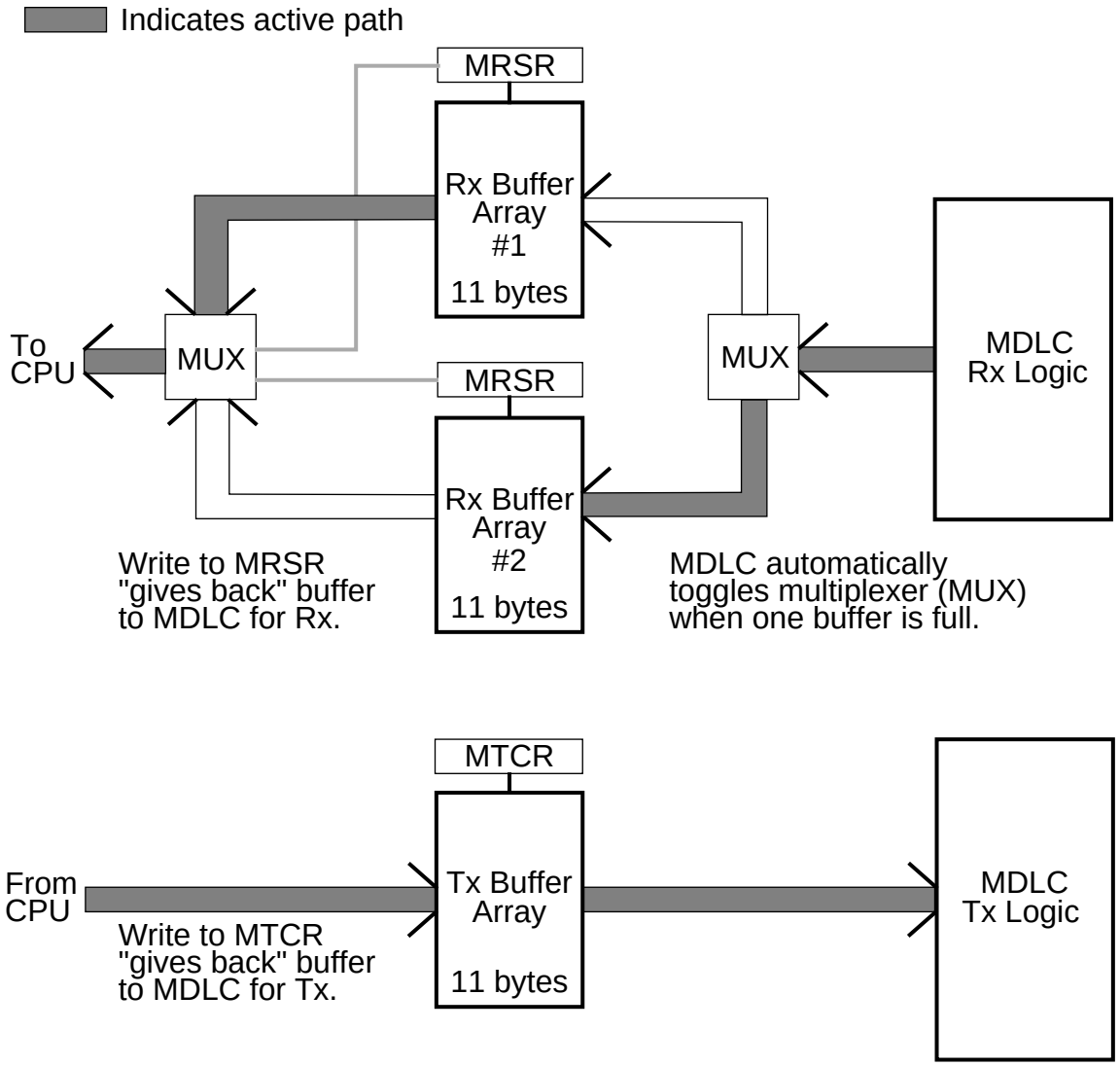


Figure 15-7: MDLC Rx/Tx Buffers Outline

15.3.2 RX BUFFERS

The Rx Buffers consist of two 11-byte arrays, with 8 bits per byte. Only one message may occupy each Rx Buffer array at any time. Each Rx Buffer array has an internal register (the 'Rx Buffer pointer') associated with it in which the length of the message body is recorded.

Following a reset, the MDLC has "possession" of the two empty Rx Buffers and their associated Rx Buffer pointers are both reset to zero. Neither Rx Buffer appears in the CPU's memory map at this time, attempts to read them will return undefined data.

As each data byte of an incoming message is received by the MDLC, it is placed into the next available entry of one of the Rx Buffers and the Rx Buffer pointer is incremented. This continues until the end of the message is detected or an error occurs.

If the received message is verified as error-free then the CRC is discarded, the filled Rx Buffer and the final value of the Rx Buffer pointer are placed into the CPU's memory map, the Received Message Successfully (RXMS) bit in the MDLC Status Register (MSR) is set, a CPU interrupt request is made (if interrupts are enabled) and the other Rx Buffer and its pointer are readied for reception of the next message by the MDLC.

The CPU may now randomly read the Rx Buffer. The first data byte received will be located in the lowest address entry of the Rx Buffer. The rest of the message is located in contiguous ascending address entries up to the last data byte received, which will be located in the nth entry, where 'n' is the final value of the Rx Buffer pointer which now appears in the MDLC Rx Status Register (MRSR).

As long as the CPU has possession of this Rx Buffer, the MDLC will have possession of the other Rx Buffer and can concurrently receive a second message without conflict with, or intervention by, the CPU. Once the CPU has finished with the contents of the Rx Buffer, it may be "given back" to the MDLC by writing **any** value to the MRSR register.

The above sequence is then repeated as long as normal data reception takes place.

If the CPU is unable to return an Rx Buffer **and** the second Rx Buffer becomes filled **and** a further new message arrives from the J1850 bus then the new message will be ignored by the MDLC until an Rx Buffer becomes available to receive new data.

If any type of reception error is detected by the MDLC as a message is being received from the J1850 bus, the internal Rx Buffer pointer will be reset to zero, effectively flushing the data received so far, and the MDLC will ignore the rest of the message. When this occurs the MDLC will **not** generate a CPU interrupt request and will silently wait for the next valid SOF symbol.

This will also happen if a message received from the J1850 bus is longer than 12 bytes (including CRC byte), and the Receive Block mode (RXBM) bit is **not** set. In both cases, when the internal Rx Buffer pointer is reset to zero, the associated Rx Buffer array bytes will not be cleared or changed.

If a message received from the J1850 bus is longer than 12 bytes (including CRC byte), and the Receive Block mode (RXBM) bit **is** set then reception of the message will be continued into the next available Rx Buffer. This will repeat until the end of the message is detected, at which time the residual data bytes will remain in the last Rx Buffer to be filled.

15.3.3 TX BUFFER

The Tx Buffer consists of one 11-byte array, with 8 bits per byte. Only one message may occupy the Tx Buffer at any time. The Tx Buffer array has an internal register (the 'Tx Buffer pointer') associated with it in which the length of the message body is stored.

Following a reset, the CPU has "possession" of the empty Tx Buffer and its associated Tx Buffer pointer is reset to zero. The Tx Buffer appears in the CPU's memory map at this time, where the CPU may randomly access it. The first data byte of the message to be sent must be placed in the lowest address entry of the Tx Buffer. The rest of the message must be placed in contiguous ascending address entries up to the last data byte to be sent, which is located in the nth entry, where 'n' is the length of the message body to be sent.

As long as the CPU has possession of the Tx Buffer, the MDLC cannot transmit. Once the CPU has finished with the contents of the Tx Buffer, it may be "given back" to the MDLC by writing the message body length ’n’ to the MDLC Tx Control Register (MTCR).

Starting from the first entry of the Tx Buffer, the MDLC fetches each data byte from a filled entry of the Tx Buffer, the Tx Buffer pointer is incremented and the MDLC attempts to transmit the data byte onto the J1850 bus. This continues until the last byte of the message body has been fetched, arbitration is lost or an error occurs. If the transmitted message is sent error-free then a CRC is appended, the filled Tx Buffer is given back to the CPU, the Transmitted Message Successfully (TXMS) bit in the MDLC Status Register (MSR) is set, and a CPU interrupt request is made (if interrupts are enabled). This sequence is then repeated as long as normal data transmission takes place.

If any type of transmission error is detected by the MDLC as a message is being transmitted onto the J1850 bus, the internal Tx Buffer pointer will be reset to zero, effectively flushing the data to be sent, and the MDLC will automatically abort transmission. When this occurs the MDLC will **not** generate a CPU interrupt request and will silently wait for the next write to the MTCR register.

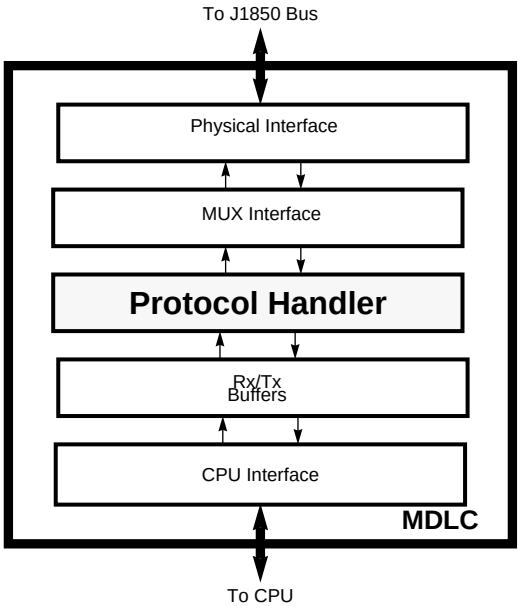
If a loss of arbitration occurs while the MDLC is attempting to transmit a message, the MDLC will immediately halt transmission and become a receiver. As soon as an idle bus condition is detected, the MDLC will again attempt to transmit the message onto the multiplex bus. This automatic retry will continue until the message is transmitted successfully, an error is detected during transmission, or the TXAB bit is set by the CPU.

Since the Tx Buffer is 11 bytes long, any value greater than or equal to $0C written to the MTCR register will create a Tx Buffer overflow error and cause the MDLC to not transmit that message. Attempts to send a zero byte message body length (value of $00 written to MTCR) will also cause the MDLC to not transmit that message.

At the end of a transmission, successful, unsuccessful or aborted, the MTCR register is automatically cleared. If interrupts are enabled, the programmer should not poll the MTCR register to detect this action since each access will clear the TXMS bit in the MSR register! Instead, since the MDLC receives every message that it sends, the programmer should check that each message that is attempted to be sent is received back within some acceptable amount of time before attempting to send another message. Failure to receive back a message that has been sent in a timely fashion might indicate that some network fault exists.

15.4 MDLC PROTOCOL HANDLER

The Protocol Handler is responsible for framing, collision detection, arbitration, CRC generation/checking, and error detection. The Protocol Handler conforms to SAE J1850 - Class B Data Communications Network Interface.



15.4.1 OUTLINE

The Protocol Handler contains three main blocks: the State Machine, Rx Shift Register and Tx Shift Register, as shown below. Each block will now be described in more detail.

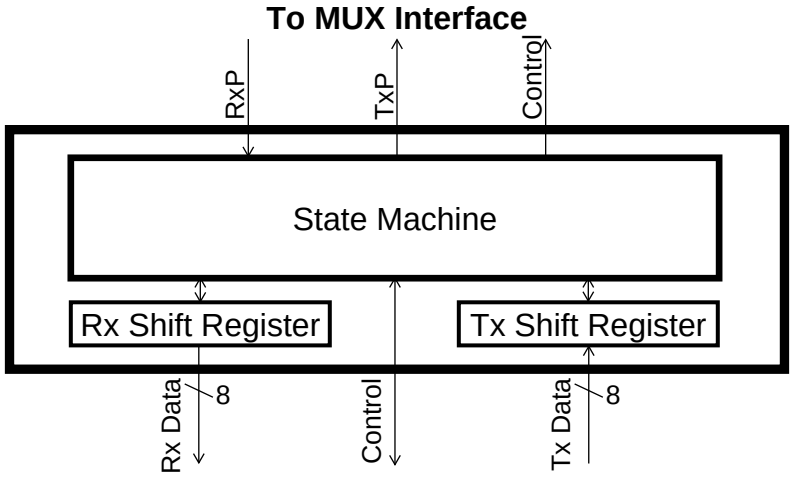


Figure 15-8: MDLC Protocol Handler Outline

15.4.2 Rx & Tx SHIFT REGISTERS

The Rx Shift Register gathers received serial data bits from the J1850 bus and makes them available in parallel form to the Rx Buffer. The Tx Shift Register takes data, in parallel form, from the Tx Buffer and presents it serially to the State Machine so that it can be transmitted onto the J1850 bus.

15.4.3 STATE MACHINE

All of the functions associated with performing the protocol are executed or controlled by the State Machine. The State Machine is responsible for framing, collision detection, arbitration, CRC generation/checking, and error detection. The following sections describe the MDLC's actions in a variety of situations.

15.4.3.1 In-Frame Response

The MDLC does not support the In-Frame Response (IFR) feature of J1850. If the MDLC receives an IFR, it will ignore the response (**not** placing it in an Rx Buffer) and wait for the End of Frame (EOF) symbol.

15.4.3.2 4x Speed Mode

The MDLC can exist on the same J1850 bus as modules which use a special 4x mode of J1850 VPW operation. The MDLC treats 4x mode messages as noise and ignores them. The MDLC will wait for a valid SOF or BREAK at 10.4 kbps to resume normal operation.

15.4.3.3 Block Mode

While the MDLC can only transmit a maximum of 12 bytes (including CRC byte) in a given message, it has the ability to receive a message of unlimited length. Like 4x mode, the programmer will have to determine from a received message that Block mode is to be enabled. The programmer then sets the "Receive Block Message" (RXBM) bit in the MDLC Control Register (MCR) to indicate to the MDLC that the following message will be longer than 12 bytes and not to treat it as a message over length error.

15.4.3.4 Arbitration

Arbitration is performed by the MDLC simply by comparing the data being received from the J1850 bus with the data being transmitted. This is done by latching the data being transmitted into a comparator, where it is compared with the data being received from the J1850 bus. If a valid data bit is received which has a higher priority over what was transmitted (0 over 1), arbitration is then lost, and the transmitter stops transmitting until a valid EOF symbol is received from the J1850 bus. See 15.5.5 MESSAGE ARBITRATION for more information.

If an invalid bit or a symbol in a non-byte boundary is detected, the transmitter will also stop transmitting.

In VPW, an active signal will always dominate a passive one, and a shorter passive symbol will dominate a longer passive symbol. This ensures that a logic zero will dominate a logic one, and the message with the highest priority (lowest value) will win arbitration. See **15.5.5 MESSAGE ARBITRATION** for more details.

15.4.3.5 J1850 Bus Errors

The MDLC detects several types of transmit and receive errors which can occur during the transmission of a message onto the J1850 bus.

If the MDLC is transmitting a message and the message received contains invalid bits, or framing symbols on non-byte boundaries, this constitutes a transmission error. When a transmission error is detected, the MDLC will immediately cease transmitting.

If the MDLC is receiving a message and it detects an error during the message, the message is discarded. No message with an error contained in it is passed to the CPU and no error status information is available.

15.4.3.5.1 CRC Error

A CRC error is detected when the data bytes and CRC byte of a received message are processed, and the CRC calculation result is not equal to $C4. The CRC code should detect any single and 2 bit errors, as well as all 8 bit burst errors, and almost all other types of errors.

15.4.3.5.2 Bit Format Error

A Bit Format error is detected when an abnormal (invalid) bit is detected in a message being received from the J1850 bus. However, if the MDLC is transmitting when this happens, it will be treated as a loss of arbitration rather than a transmitter error and the automatic retry mechanism will apply.

15.4.3.5.3 Message Format Error

A Message Format error is detected if an EOD or EOF symbol is detected on a non-byte boundary from the J1850 bus.

15.4.3.5.4 Bus Fault

If a bus fault occurs, the response of the MDLC will depend upon the type of bus fault.

If the bus is shorted to V_{BATT}, the MDLC will wait for the bus to fall to a passive state before it will attempt to transmit a message. As long as the short remains, the MDLC will never attempt to transmit a message onto the J1850 bus.

If the bus is shorted to ground, the MDLC will see an idle bus, begin to transmit the message, and then detect a transmission error, since the short to ground would not allow the bus to be driven to the dominant state. The MDLC will abort that transmission and wait for the next CPU command to transmit.

In any case, if the bus fault is temporary, as soon as the fault is cleared, the MDLC will resume normal operation. If the bus fault is permanent, it may result in permanent loss of communication on the J1850 bus.

15.4.3.5.5 Message Under Length Error

Any message received by the MDLC must be at least two bytes long. Since the MDLC does not have message filtering, these two bytes could just be one data byte and a CRC byte. Any messages received with less than the minimum number of bytes will be discarded by the MDLC.

15.4.3.5.6 Message Over Length Error

A Message Over Length error is detected when a message being received exceeds the maximum length allowed for a message on the J1850 bus. If the Receive Block mode (RXBM) bit is not set, the maximum received length from SOF to the EOD is 12 bytes including the CRC byte. Any messages received with more than the maximum number of bytes will be discarded by the MDLC.

15.4.3.5.7 Invalid Active Level

If the MDLC transmits an active symbol but senses that the bus remains active longer than 163 us, then the MDLC will stop transmitting. The MDLC will treat this invalid active level just like a received Break symbol. The MDLC will try to transmit the message again after an idle bus is detected.

15.4.3.5.8 BREAK - Break

Any MDLC transmitting at the time a BREAK is detected will treat the BREAK as if a loss of arbitration had occurred, and halt transmission. The MDLC will then retry to transmit the message.

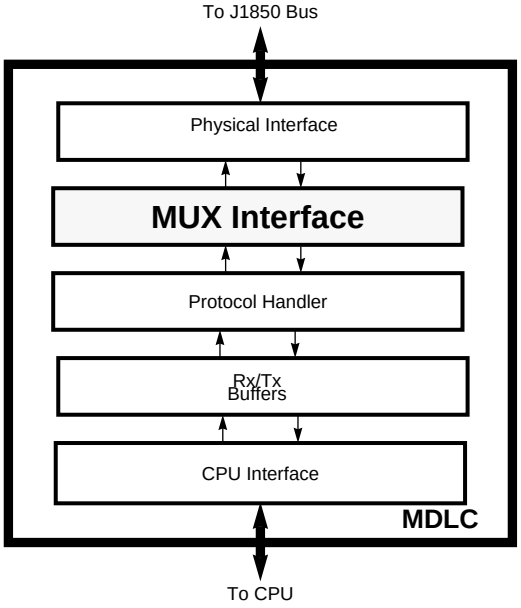
If while receiving a message the MDLC detects a BREAK symbol, it will treat the BREAK as a reception error and clear the current Rx Buffer pointer.

Table 15-3: MDLC J1850 Bus Error Summary

Error Condition	MDLC Function
Bus short to V_{BATT}.	The MDLC will not transmit until the bus is idle.
Bus short to Gnd.	The MDLC will try only once to send the message. No interrupt will be generated.
MDLC transmits but receives invalid bits (noise).	The MDLC will abort transmission immediately. No interrupt will be generated.
MDLC receives invalid bit.	The MDLC will reset the Rx Buffer pointer and wait for the next valid SOF. No interrupt will be generated.
MDLC receives message framing error.	The MDLC will reset the Rx Buffer pointer and wait for the next valid SOF. No interrupt will be generated.
MDLC receives under length or over length message.	The MDLC will reset the Rx Buffer pointer and wait for the next valid SOF. No interrupt will be generated.
MDLC receives invalid CRC	The MDLC will reset the Rx Buffer pointer and wait for the next valid SOF. No interrupt will be generated.
MDLC receives BREAK symbol	The MDLC will reset the Rx Buffer pointer and wait for the next valid SOF. No interrupt will be generated.
MDLC sends an EOD but receives an active symbol.	The MDLC will reset the Rx Buffer pointer and try to resend that message after an idle bus.

15.5 MDLC MUX INTERFACE

The MUX Interface is responsible for bit encoding/decoding and digital noise filtering between the Protocol Handler and the Physical Interface.



15.5.1 Rx DIGITAL FILTER

The Receiver section of the MDLC includes a digital low pass filter to remove narrow noise pulses from the incoming message. An outline of the digital filter is shown in the following diagram.

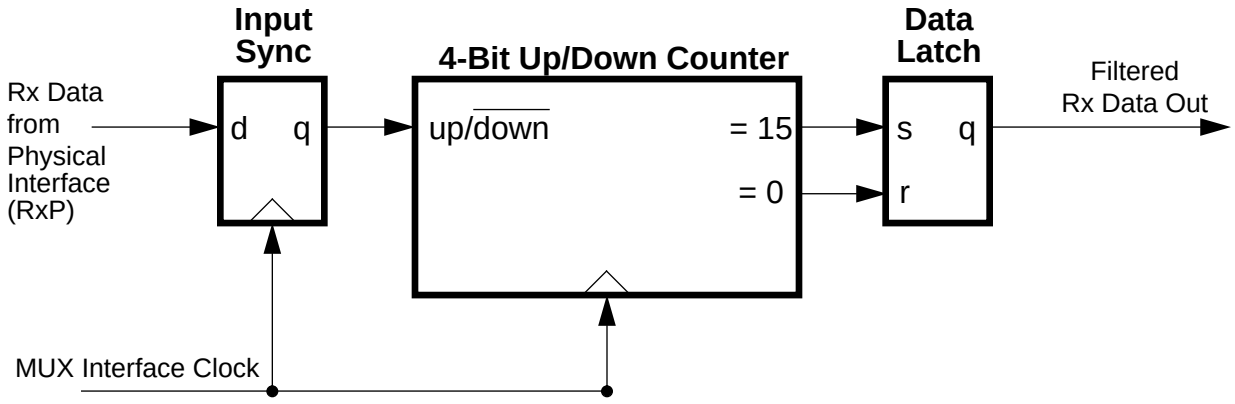


Figure 15-9: MDLC Rx Digital Filter Block

15.5.1.1 Operation

The clock for the digital filter is provided by the MUX Interface clock. At each positive edge of the clock signal, the current state of the Receiver Physical Interface (RxP) signal is sampled. The RxP signal state is used to determine whether the counter should increment or decrement at the next negative edge of the clock signal.

The counter will increment if the input data sample is high but decrement if the input sample is low. The counter will thus progress up towards ’15’ if, on average, the RxP signal remains high or progress down towards ’0’ if, on average, the RxP signal remains low.

When the counter eventually reaches the value ’15’, the digital filter decides that the condition of the RxP signal is at a stable logic level one and the Data Latch is set, causing the Filtered Rx Data signal to become a logic level one. Furthermore, the counter is prevented from overflowing and can only be decremented from this state.

Alternatively, should the counter eventually reach the value ’0’, the digital filter decides that the condition of the RxP signal is at a stable logic level zero and the Data Latch is reset, causing the Filtered Rx Data signal to become a logic level zero. Furthermore, the counter is prevented from underflowing and can only be incremented from this state.

The Data Latch will retain its value until the counter next reaches the opposite end point, signifying a definite transition of the RxP signal.

15.5.1.2 Performance

The performance of the digital filter is best described in the time domain rather than the frequency domain.

If the level of the RxP signal transitions, then there will be a delay before that transition appears at the Filtered Rx Data output signal. This delay will be between 15 and 16 clock periods, depending on where the transition occurs with respect to the sampling points. This ’filter delay’ must be taken into account when performing message arbitration.

For example, if the frequency of the MUX Interface clock ($f_{MDLC)}$ is 1.0486 MHz, then the period ($t_{MDLC)}$ is 954ns and the maximum filter delay in the absence of noise will be 15.259us.

The effect of random noise on the RxP signal depends on the characteristics of the noise itself. Narrow noise pulses on the RxP signal will be completely ignored if they are shorter than the filter delay. This provides a degree of low pass filtering.

If noise occurs during a symbol transition, the detection of that transition may be delayed by an amount equal to the length of the noise burst. This is just a reflection of the uncertainty of where the transition is truly occurring within the noise.

Noise pulses that are wider than the filter delay, but narrower than the shortest allowable symbol length will be detected by the next stage of the MDLC receiver as an invalid symbol.

Noise pulses that are longer than the shortest allowable symbol length will normally be detected as an invalid symbol or as invalid data when the frame’s CRC is checked.

15.5.2 J1850 FRAME FORMAT

All messages transmitted on the J1850 bus are structured using the format:

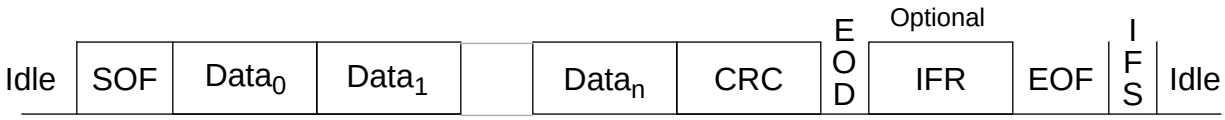


Figure 15-10: J1850 Bus Message Format (VPW)

Each message has a maximum length of 12 bytes (excluding SOF, EOD, NB and EOF).

15.5.2.1 SOF - Start of Frame Symbol

All messages transmitted onto the J1850 bus must begin with an SOF symbol. This indicates to any listeners on the J1850 bus the start of a new message transmission. The SOF symbol is not used in the CRC calculation.

15.5.2.2 Data - In Message Data Bytes

The data bytes contained in the message include the message header bytes and any actual data being transmitted to the receiving node. The MDLC can be used to transmit messages using any of the three header formats outlined in the SAE J1850 document. However, the MDLC does not utilize the IFR, Header type or functional/physical addressing bits defined in the 1st byte of the 3-byte consolidated header message format. See *SAE J1850 - Class B Data Communications Network Interface*, for more information about 1- and 3-byte headers.

Messages transmitted by the MDLC onto the J1850 bus must contain at least one data byte, and therefore can be as short as one data byte and one CRC byte. Each data byte in the message is 8 bits in length, transmitted MSB to LSB.

15.5.2.3 CRC - Cyclical Redundancy Check Byte

This byte is used by the receiver(s) of each message to determine if any errors have occurred during the transmission of the message. The MDLC calculates the CRC byte and appends it onto any messages transmitted onto the J1850 bus, and also performs CRC detection on any messages it receives from the J1850 bus.

CRC generation uses the divisor polynomial $X^8+X^4+X^3+X^2+1$. The remainder polynomial is initially set to all ones, and then each byte in the message after the SOF symbol is serially processed through the CRC generation circuitry. The one's complement of the remainder then becomes the 8-bit CRC byte, which is appended to the message after the data bytes, in MSB to LSB order.

When receiving a message, the MDLC uses the same divisor polynomial. All data bytes, excluding the SOF and EOD symbols, but including the CRC byte, are used to check the CRC. If the message is error free, the remainder polynomial will equal $X^7+X^6+X^2$ ($C4), regardless of the data contained in the message. If the calculated CRC does not equal $C4, the MDLC will recognize this as a CRC error, and will discard the message, not informing the CPU of the failure.

15.5.2.4 EOD - End of Data Symbol

The EOD symbol is a short passive period on the J1850 bus used to signify to any recipients of a message that the transmission by the originator has completed.

15.5.2.5 IFR - In Frame Response Bytes

The IFR section of the J1850 message format is optional. The MDLC does not support this option. The MDLC will not transmit an IFR under any circumstances. If an IFR is received from another node, the MDLC will ignore this part of the message and wait for the EOF symbol before resuming normal operation.

15.5.2.6 EOF - End of Frame Symbol

This symbol is a passive period on the J1850 bus, longer than an EOD symbol, which signifies the end of a message. Since an EOF symbol is longer than an EOD symbol, if no response is transmitted after an EOD symbol, it becomes an EOF, and the message is assumed to be completed.

15.5.2.7 IFS - Inter-Frame Separation Symbol

The IFS symbol is a passive period on the J1850 bus that allows proper synchronization between nodes during continuous message transmission. The IFS symbol is transmitted by a node following the completion of the EOF period.

When the last byte of a message has been transmitted onto the J1850 bus, and the EOF symbol time has expired, all nodes must then wait for the IFS symbol time to expire before transmitting an SOF, marking the beginning of another message.

However, if the MDLC is waiting for the IFS period to expire before beginning a transmission and a rising edge is detected before the IFS time has expired, it must internally synchronize to that edge. If a write to the MTCR register (initiate transmission) occurred on or before $104 \cdot t_{MDLC}$ from the received rising edge, then the MDLC will transmit and arbitrate for the bus. If a CPU write to the MTCR register occurred after $104 \cdot t_{MDLC}$ from the detection of the rising edge, then the MDLC will not transmit, but will wait for the next IFS period to expire before attempting to transmit the message.

A rising edge may occur during the IFS period because of varying clock tolerances and loading of the J1850 bus, causing different nodes to observe the completion of the IFS period at different times. Receivers must synchronize to any SOF occurring during an IFS period to allow for individual clock tolerances.

15.5.2.8 BREAK - Break

Any MDLC transmitting at the time a BREAK is detected will treat the BREAK as if a loss of arbitration had occurred, and halt transmission. The MDLC cannot transmit a BREAK symbol. If while receiving a message the MDLC detects a BREAK symbol, it will treat the BREAK as a reception error and clear any partially received message from the Rx buffer.

15.5.2.9 Idle Bus

An idle condition exists on the bus during any passive period after expiration of the IFS period. Any node sensing an idle bus condition can begin transmission immediately.

15.5.3 J1850 VPW SYMBOLS

Variable Pulse Width Modulation (VPW) is an encoding technique in which each bit is defined by the time between successive transitions, and by the level of the bus between transitions, active or passive. Active and passive bits are used alternately.

Each logic one or logic zero contains a single transition, and can be at either the active or passive level and one of two lengths, either 64 µs or 128 µs (T_{NOM} at 10.4kbps baud rate), depending upon the encoding of the previous bit. The SOF, EOD, EOF and IFS symbols will always be encoded at an assigned level and length. See Figure 15-11.

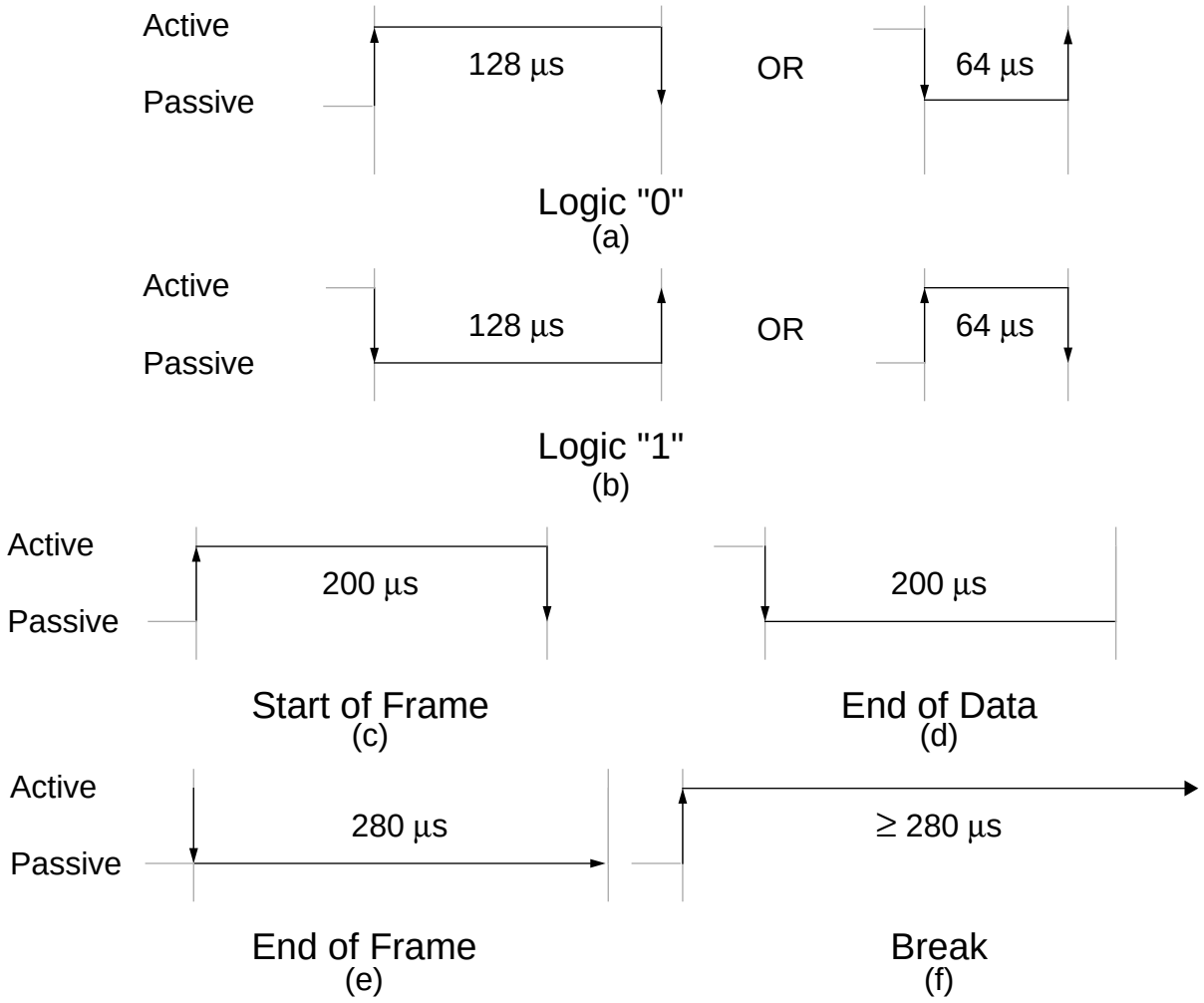


Figure 15-11: J1850 VPW Symbols

Each message will begin with an SOF symbol, an **active** symbol, and therefore each data byte (including the CRC byte) will begin with a **passive** bit, regardless of whether it is a logic one or a logic zero.

All VPW bit lengths stated in the following descriptions are typical values at a 10.4 kbps bit rate.

15.5.3.1 Logic "0"

A logic zero is defined as either an active to passive transition followed by a passive period 64 µs in length, or a passive to active transition followed by an active period 128 µs in length (Figure 15-11(a)).

15.5.3.2 Logic "1"

A logic one is defined as either an active to passive transition followed by a passive period 128 µs in length, or a passive to active transition followed by an active period 64 µs in length (Figure 15-11(b)).

15.5.3.3 SOF - Start of Frame Symbol

The SOF symbol is defined as a passive to active transition followed by an active period 200 µs in length (Figure 15-11(c)). This allows the data bytes that follow the SOF symbol to begin with a passive bit, regardless of whether it is a logic one or a logic zero.

15.5.3.4 EOD - End of Data Symbol

The EOD symbol is defined as an active to passive transition followed by a passive period 200 µs in length (Figure 15-11(d)).

15.5.3.5 EOF - End of Frame Symbol

The EOF symbol is defined as an active to passive transition followed by a passive period of at least 280 µs in length (Figure 15-11(e)). If there is no IFR byte transmitted after an EOD symbol is transmitted, after another 80 µs the EOD becomes an EOF, indicating the completion of the message.

15.5.3.6 IFS - Inter-Frame Separation Symbol

The IFS symbol is defined as a passive period 300 µs in length. The IFS symbol contains no transition, since when used it always follows an EOF symbol.

15.5.3.7 BREAK - Break Signal

The BREAK signal is defined as a passive to active transition followed by an active period of at least 280 µs (Figure 15-11(f)).

15.5.4 J1850 VPW VALID/INVALID BITS & SYMBOLS

The timing tolerances for receiving data bits and symbols from the J1850 bus have been defined to allow for variations in oscillator frequencies. In many cases the maximum time allowed to define a data bit or symbol is equal to the minimum time allowed to define another data bit or symbol.

Since the minimum resolution of the MDLC for determining what symbol is being received is equal to a single period of the MUX Interface clock (t_{MDLC}), an apparent separation in these maximum time/minimum time concurrences equal to one cycle of t_{MDLC} occurs.

This one clock resolution allows the MDLC to properly differentiate between the different bits and symbols, without reducing the valid window for receiving bits and symbols from transmitters onto the J1850 bus having varying oscillator frequencies.

In VPW bit encoding, the tolerances for the both passive and active data bits and symbols are defined with no gaps between definitions. For example, the maximum length of a passive logic zero is equal to the minimum length of a passive logic one, and the maximum length of an active logic zero is equal to the minimum length of a valid SOF symbol.

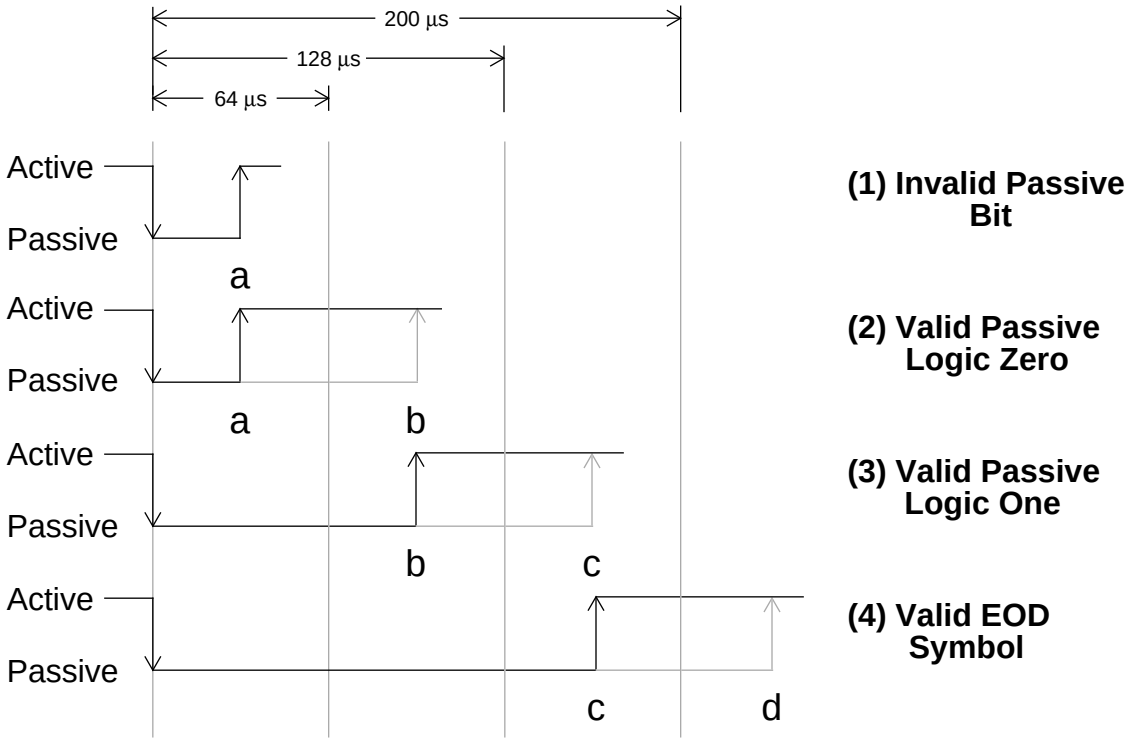


Figure 15-12: J1850 VPW Passive Symbols

15.5.4.1 Invalid Passive Bit

If the passive to active transition beginning the next data bit or symbol occurs between the active to passive transition beginning the current data bit or symbol and **a**, the current bit would be invalid. See Figure 15-12(1).

15.5.4.2 Valid Passive Logic Zero

If the passive to active transition beginning the next data bit or symbol occurs between **a** and **b**, the current bit would be considered a logic zero. See Figure 15-12(2).

15.5.4.3 Valid Passive Logic One

If the passive to active transition beginning the next data bit or symbol occurs between **b** and **c**, the current bit would be considered a logic one. See Figure 15-12(3).

15.5.4.4 Valid EOD Symbol

If the passive to active transition beginning the next data bit or symbol occurs between **c** and **d**, the current symbol would be considered a valid EOD symbol. See Figure 15-12(4).

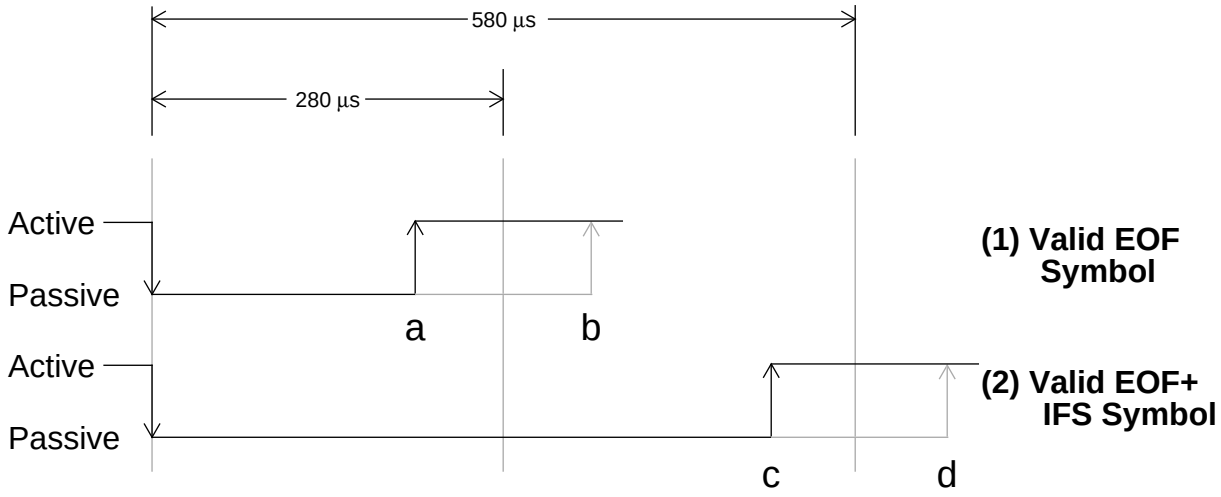


Figure 15-13: J1850 VPW EOF and IFS Symbols

15.5.4.5 Valid EOF & IFS Symbol

If the passive to active transition beginning the SOF symbol of the next message occurs between **a** and **b**, the current symbol will be considered a valid EOF symbol. If the passive to active transition beginning the SOF symbol of the next message occurs between **c** and **d**, the current symbol will be considered a valid EOF symbol followed by a valid IFS symbol. All nodes must wait until a valid IFS symbol time has expired before beginning transmission. However, due to variations in clock frequencies and bus loading, some nodes may recognize a valid IFS symbol before others, and immediately begin transmitting. Therefore, anytime a node waiting to transmit detects a passive to active transition once a valid EOF has been detected, it should immediately begin transmission, initiating the arbitration process. See Figure 15-13(1&2).

15.5.4.6 Idle Bus

If the passive to active transition beginning the SOF symbol of the next message does not occur before **d,** the bus is considered to be idle, and any node wishing to transmit a message may do so immediately.

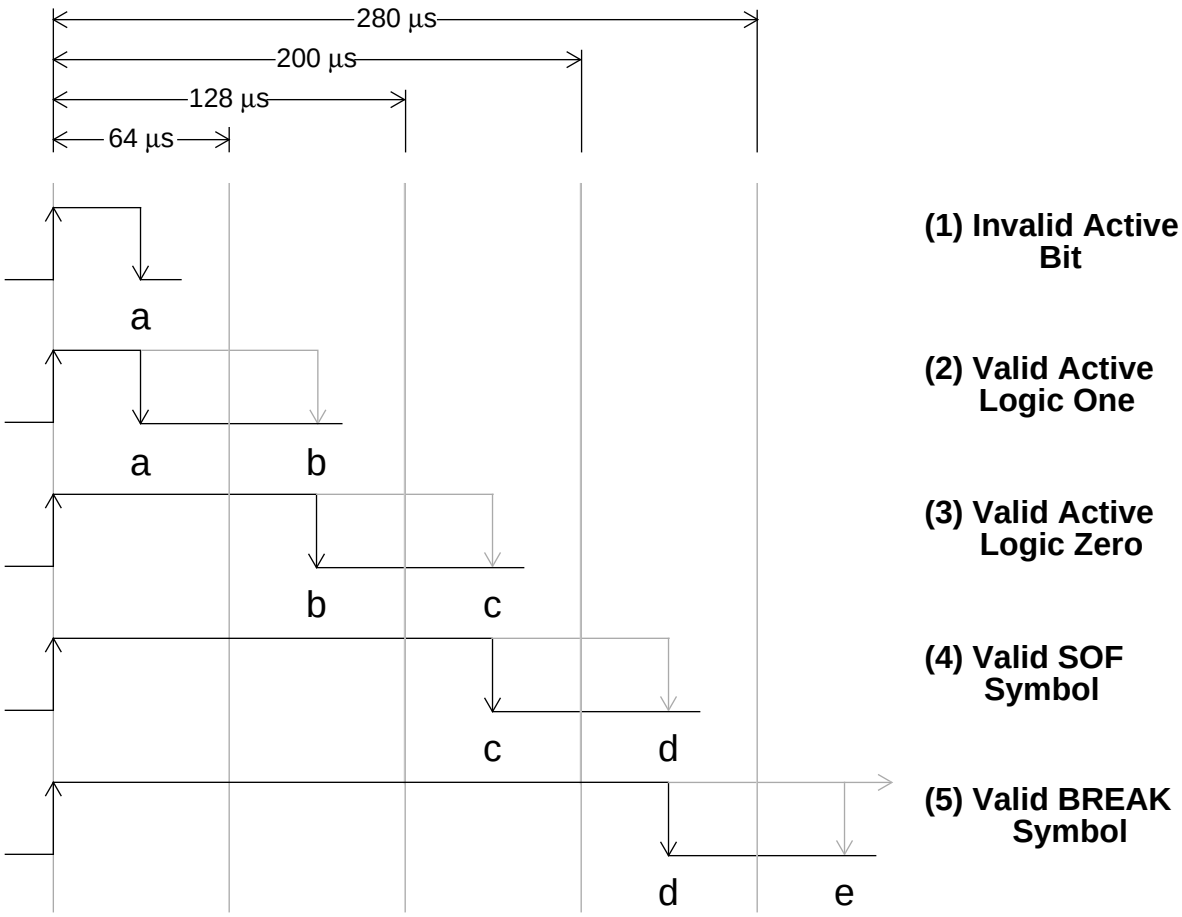


Figure 15-14: J1850 VPW Active Symbols

15.5.4.7 Invalid Active Bit

If the active to passive transition beginning the next data bit or symbol occurs between the passive to active transition beginning the current data bit or symbol and **a**, the current bit would be invalid. See Figure 15-14(1).

15.5.4.8 Valid Active Logic One

If the active to passive transition beginning the next data bit or symbol occurs between **a** and **b**, the current bit would be considered a logic one. See Figure 15-14(2).

15.5.4.9 Valid Active Logic Zero

If the active to passive transition beginning the next data bit or symbol occurs between **b** and **c**, the current bit would be considered a logic zero. See Figure 15-14(3).

15.5.4.10 Valid SOF Symbol

If the active to passive transition beginning the next data bit or symbol occurs between **c** and **d**, the current symbol would be considered a valid SOF symbol. See Figure 15-14(4).

15.5.4.11 Valid BREAK Symbol

If the next active to passive transition does not occur until after **d**, the current symbol will be considered a valid BREAK symbol. Following the BREAK symbol, an IFS period must be observed, after which normal communication can resume on the J1805 bus. See Figure 15-14(5).

15.5.5 MESSAGE ARBITRATION

Message arbitration on the J1850 bus is accomplished in a non-destructive manner, allowing the message with the highest priority to be transmitted, while any transmitters which lose arbitration simply stop transmitting and wait for an idle bus to begin transmitting again.

If the MDLC wishes to transmit onto the J1850 bus, but detects that another message is in progress, it must wait until the bus is idle. However, if multiple nodes begin to transmit in the same synchronization window, message arbitration will occur beginning with the first bit after the SOF symbol and continue with each bit thereafter.

The VPW symbols and J1850 bus electrical characteristics are carefully chosen so that a logic zero (active or passive type) will always dominate over a logic one (active or passive type) simultaneously transmitted. Hence logic zeroes are said to be 'dominant' and logic ones are said to be 'recessive'. Whenever a node detects a dominant bit when it transmits a recessive bit, it loses arbitration, and immediately stops transmitting. This is known as 'bitwise arbitration'.

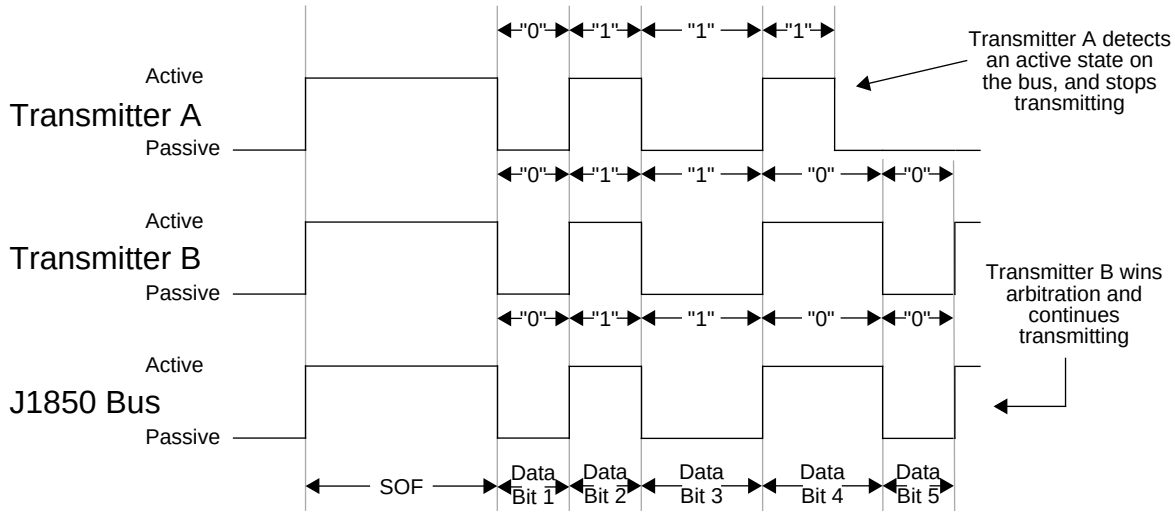


Figure 15-15: J1850 VPW Bitwise Arbitration

During arbitration, or even throughout the message being transmitted, when an opposite bit is detected, transmission is immediately stopped unless it occurs on the 8th bit of a byte. In this case the MDLC will automatically append up to two extra 1 bits and then stop transmitting. These two extra bits will be arbitrated normally and thus will not interfere with another message. The second 1 bit will not be sent if the first loses arbitration. If the MDLC has lost arbitration to another valid message then the two extra ones will not corrupt the current message. However, if the MDLC has lost arbitration due to noise on the bus, then the two extra ones will ensure that the current message will be detected and ignored as a noise-corrupted message.

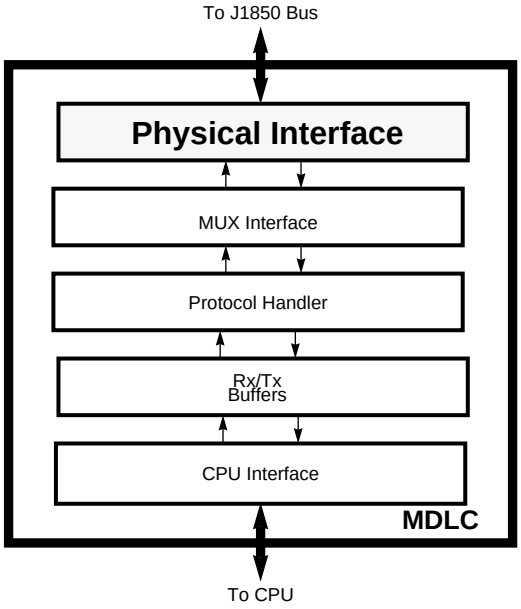
Since a "0" dominates a "1", the message with the lowest value will have the highest priority, and will always win arbitration, i.e. a message with priority 000 will win arbitration over a message with priority 001. This method of arbitration will work no matter how many bits of priority encoding are contained in the message.

If the MDLC loses arbitration during the transmission of a message, it will attempt to retransmit as soon as a valid EOF is detected on the J1850 bus. The MDLC will attempt to retransmit the message indefinitely, as long as messages with higher priorities continue to win arbitration.

The only way to stop transmission retry due to loss of arbitration is to abort the transmission by setting the Tx Abort (TXAB) bit in the MDLC Control register. This will reset the internal Tx Buffer pointer and halt transmission.

15.6 MDLC PHYSICAL INTERFACE

All analog functions involved in transmitting and receiving from the J1850 bus are performed by the Physical Interface. The Physical Interface serves to buffer the incoming messages, wave-shape the messages being transmitted, and protect the MDLC from transients occurring on the J1850 bus.



15.6.1 OUTLINE

The receiver analog comparator and transmitter drivers are included in the MDLC Physical Interface and together are referred to as the "transceiver".

The transceiver provides a waveshaped 7V bus waveform in response to a timed logic signal from the MUX Interface. The transceiver actively drives the bus high, and passively lets an RC network pull the bus low. In order to achieve the 7V level necessary for the bus signal, the transceiver uses a battery supply (V_{BATT}) which is nominally 12V. The transceiver also receives bus waveforms and provides the MUX Interface with unfiltered input data

A low power mode of operation is automatically entered if the CPU executes a STOP or WAIT instruction.

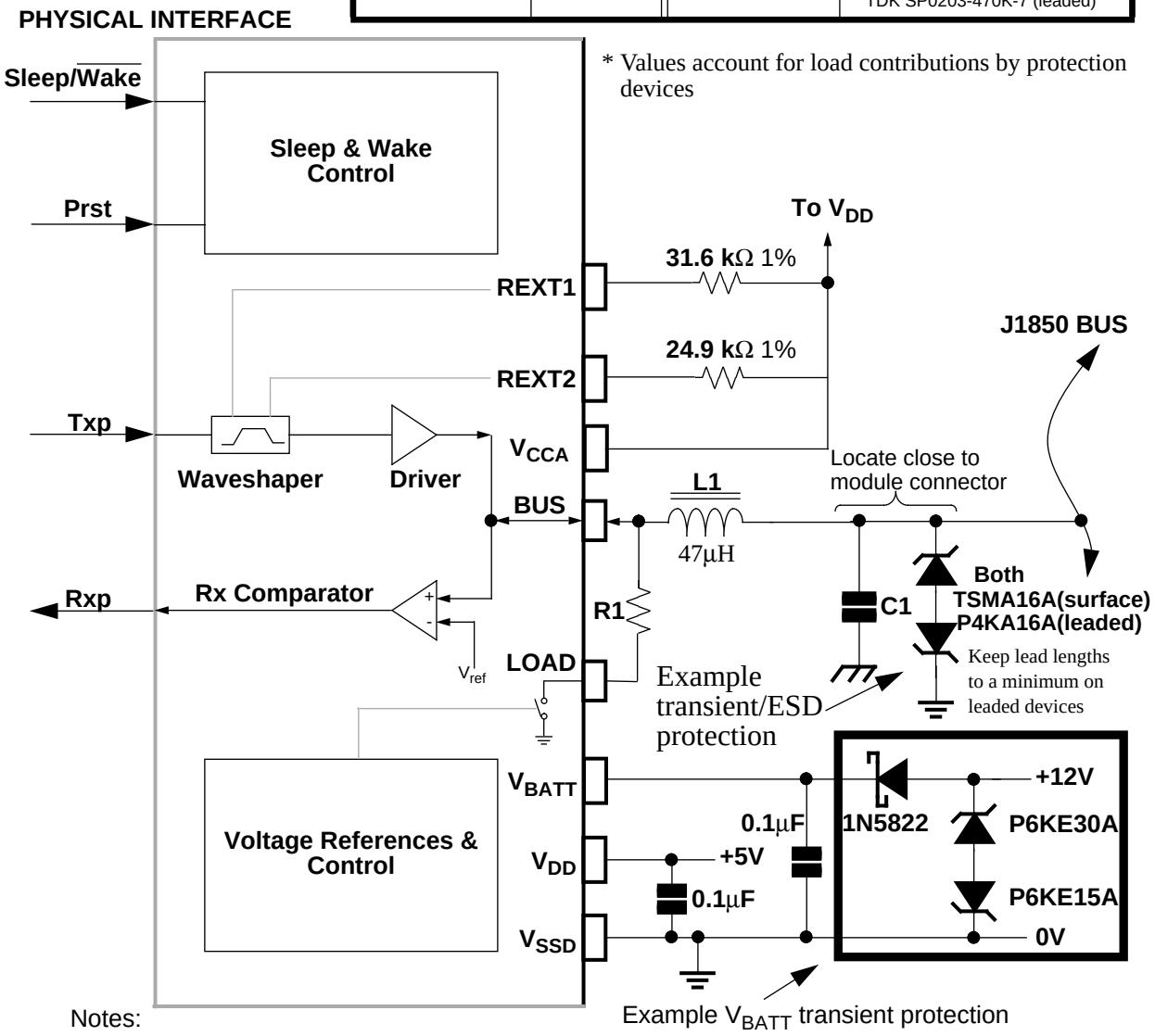
In the event that ground is lost, the transceiver senses this condition and releases the bus by switching the LOAD pin to a high impedance instead of a low impedance to ground.The transceiver also protects the MDLC by not passing on any standard disruptive or non-disruptive signals seen on the bus to the rest of the MDLC.

The Physical Interface contains its own power-on reset circuit that is used to reset its control circuitry whenever a power-on condition is detected.

See **Figure 15-16 : MDLC Physical Interface Outline** for an example mechanization for interfacing the MDLC module to the J1850 multiplex bus physical layer. Although this example mechanization is designed to provide some transient protection beyond what the

MDLC module can withstand directly, the user should ensure that their circuit design meets the transient protection requirements of their application.

MDLC Use	R1	C1	L1
1st node in system	1.5kΩ ±1%	3300pF/2700pF*	TDK NL322522T-470J-3 (surface) TDK SP0203-470K-7 (leaded)
All other nodes	10.7kΩ ±1%	470pF/220pF*	TDK NL322522T-470J-3 (surface) TDK SP0203-470K-7 (leaded)



Notes:

All circuit board grounds (⏚) should be tied to a single point on the circuit board.

The chassis ground (⏚) should be tied to a single point on the chassis as close to the Bus connector as possible.

The L1,C1 filter should be located as close to the Bus connector as possible.

R1, REXT1, REXT2 and decoupling capacitors should be located as close as possible to the MDLC

Transient protection circuits are optional and only examples of how a protection scheme might be implemented, these devices may not be sufficient for all applications and under all circumstances.

Figure 15-16 : MDLC Physical Interface Outline

15.6.2 MUX INTERFACE SIGNALS

There are five signals which pass between the Physical Interface and the MUX Interface section of the MDLC as shown in **Figure 15-16 : MDLC Physical Interface Outline**. These signals are now described.

15.6.2.1 Sleep/$\overline{\text{Wake}}$

This input signal is used to control the power consumption of the Physical Interface transmitter.

When driven **high** by the MUX interface, the transmitter will enter a low power consumption **Sleep** mode. This will happen whenever the CPU executes a STOP or WAIT instruction. The transmitter is not capable of transmitting while it is in the Sleep mode.

When driven **low** by the MUX interface, the transmitter will '**wake-up**' and become capable of transmitting messages. This will happen when the MDLC enters the Run mode.

15.6.2.2 Wake

This input signal is used to **exit** the low power consumption mode of the Physical Interface transmitter.

The MUX interface will drive this signal **high and then low** to notify the transmitter to **wake** up from Sleep mode.

This will happen whenever the transmitter is in the Sleep mode and a passive to active transition occurs on the J1850 bus, causing the RxP signal to transition from low to high.

15.6.2.3 Prst

This is the reset signal input to the Physical Interface.

This signal is **normally low** and the MUX Interface will drive this signal **high** to **reset** the Physical Interface control circuits.

15.6.2.4 Txp

This input signal represents the digital serial transmit data from the MUX Interface to the Physical Interface.

When sending a **passive** phase, this signal will be **driven low**.

When sending an **active** phase, this signal will be **driven high**.

15.6.2.5 Rxp

This output signal represents the digital serial receive data from the Physical Interface to the MUX Interface.

When receiving an **active** phase, this signal will be **driven high.**

When receiving a **passive** phase this signal will be **driven low**.

15.6.3 PINS

There are seven pins associated with the MDLC module. **Figure 15-16 : MDLC Physical Interface Outline** gives an example mechanization. A brief description of each pin follows.

15.6.3.1 REXT1, REXT2

These are external biasing resistor tie points for transmitter waveshaping and they must be referenced to the V_{DD} pin through lines which are as low impedance and low noise as possible to ensure consistent operation.

15.6.3.2 BUS

This is the half-duplex data line for J1850 communications.

15.6.3.3 LOAD

This is the switched ground connection for the J1850 bus, via an RC load.

15.6.3.4 V_{BATT}

This is the source for battery voltage for the MDLC. This power supply pin must be protected against transient and fault conditions.

To protect against loss of battery, a low voltage protect circuit within the transceiver disables drive from the BUS pin to prevent erroneous messages from being transmitted. Nominal trip points are V_{TIVB} Volts on V_{BATT} and V_{RIVD} Volts on V_{CCA}.

15.6.3.5 V_{CCA}

This is the MDLC analog transceiver power supply input and is nominally 5V. This supply must be kept as low impedance and low noise as possible. It may be supplied from a different voltage regulator source than V_{DD}.

Whenever the voltage on this pin rises from zero to its nominal operating value, the Physical Interface detects this power-on condition and resets its control circuits. Also, whenever the voltage on this pin falls below its minimum operating value, the MDLC analog circuitry will return to its reset state.

15.6.3.6 V_{SSA2}

This is the Physical Interface power supply ground. This supply must be kept as low impedance and low noise as possible. It must be connected directly to the same grounding point that the main MCU V_{SS} uses.

15.6.3.7 Decoupling

Good supply decoupling, as close as possible to the supply pins, is absolutely essential to guarantee correct operation, minimize RFI and maximize analog performance. MDLC and non-MDLC supply lines should be kept separate to minimize interaction. Emphasis should be placed on keeping supply and ground line impedances as low as possible, with preference given to ground lines if some compromise is necessitated. Ground planes and wide, short, PCB traces can help significantly here. Decoupling capacitors should be of low

impedance construction, with regard to high frequencies, and placed as close as is physically possible to the supply pins of the MDLC. See Motorola application note *Designing for EMC with HCMOS Microcontrollers* (AN1050) for more help on this matter.

15.7 MDLC APPLICATION NOTES

15.7.1 INITIALIZATION

The MCU will first write to the MDLC Control Register (MCR). This byte should configure the rate select bits to configure the MUX Interface clock to its nominal value, and set the Interrupt Enable bit if desired.

15.7.2 TRANSMITTING A MESSAGE

The MDLC is ready for loading of a new message for transmission when the TXMS bit in the MDLC Status Register (MSR) is set to a 1, or had been set but was cleared by a subsequent access of the MDLC Transmit Control Register (MTCR). The MCU will first load bytes for transmission into the Tx Buffer. Once the data bytes have been loaded, the MCU will then write the transmit count to the MTCR register. This will command the MDLC to begin transmission on the J1850 bus at the next idle bus period. The MDLC will automatically calculate and append a CRC to the last byte of the transmitted message.

If the IE bit was set in the MCR register an interrupt request will be generated for the receipt of this message from the J1850 bus. The MCU should clear the interrupt by an access (read or write) to the MTCR register. The TXMS and RXMS bits in the MSR register will be set by the MDLC upon reception of this message. The TXMS bit is cleared by any access (read or write) of the MTCR register. The RXMS bit is cleared by any access (read or write) of the MDLC Receive Status Register (MRSR). The MDLC is now available to transmit another message.

A transmitting node should verify the correct transmission of its message by waiting for the Transmit Message Successful (TXMS) bit to be set in the MSR register. The only way to check for incorrect transmission of a message is for the MCU to provide a time-out if the TXMS bit is not set in a certain length of time. This is the only way to detect the unsuccessful transmission of a message, because any received message with errors is neither stored nor indicated as received in error by the MDLC.

15.7.3 RECEIVING A MESSAGE

When a complete message is received by the MDLC error-free from the SAE J1850 bus, the RXMS bit in the MSR will be set. In addition, if the IE bit in the MCR is set, a CPU interrupt request will be generated. The RXMS bit and the CPU interrupt are cleared by an access (read or write) of the MRSR register, unless a second message has also been received from the SAE J1850 bus and is waiting to be retrieved.

The MCU may extract received message bytes from the MDLC by reading successive Rx Buffer locations beginning at the lowest address location of the Rx Buffer. The received message bytes can be accessed any number of times, in any order, by the CPU. The CPU can determine the number of bytes to read by the Receive Count indicated in the MRSR

register. If in Block mode (RXBM bit set by MCU in MCR register) the MCU should treat the received bytes as part of an in-progress message until the MDLC clears the RXBM bit.

Once the received data bytes of interest to the CPU have been analyzed, the MCU must write any quantity to the MRSR register to free it for the next received message.

15.7.4 RECEIVING A MESSAGE IN BLOCK MODE

Although not a part of the SAE J1850 protocol, the MDLC does allow for a special ’Block mode’ of operation for the receiver only. The MDLC cannot transmit Block mode messages. As far as the MDLC is concerned, a Block mode message is simply a long J1850 frame that contains an indefinite number of data bytes. All of the other features of the frame remain the same, including the SOF, CRC and EOD symbols.

Another node wishing to send a Block mode transmission must first inform all other nodes on the network that this is about to happen. This is usually accomplished by sending a special predefined message. MDLC nodes wishing to receive the message should set the Receive Block Mode (RXBM) bits in the MDLC Control Register (MCR).

Since the MDLC only has a finite amount of received data buffering available, the programmer must ensure that received data is moved from the Rx Buffers to the application memory throughout the duration of the Block mode message. The MDLC aids the user by utilizing both Rx Buffers to buffer the arriving data bytes in Block mode. As soon as one Rx Buffer fills, the incoming data ’spills over’ into the second Rx Buffer, the Received Message Successfully (RXMS) bit in the MDLC Status Register (MSR) will be set and a CPU interrupt request is generated, signalling to the user that an Rx Buffer must be emptied and "given back" in the same manner as explained for receiving normal messages. This alternate filling of Rx Buffers gives plenty of time for one Rx Buffer to be emptied by the user, while the other one is being filled by the MDLC.

The Rx Buffers will continue to be alternately filled and emptied until an EOD symbol, or error, is detected. Throughout the reception of the Block mode message, the MDLC will calculate a running CRC. When an EOD symbol is finally detected, the CRC will be checked and, if correct, the Received Message Successfully (RXMS) bit in the MDLC Status Register (MSR) will be set, the RXBM bit will be cleared and a CPU interrupt request will be generated.

Should the second Rx Buffer ever fill, then it will attempt to spill over into the first Rx Buffer which (hopefully) has been emptied in time. If not, an overflow condition is detected, both Rx Buffer pointers are reset (discarding the data currently held in the Rx Buffers), the RXBM bit will be cleared, and the MDLC will silently wait for the next normal J1850 message to be received. Any other errors detected during the reception of a Block mode message (e.g. invalid symbols) will result in these same actions.

MDLC nodes not wishing to receive a Block mode message can leave the RXBM bit clear, causing all Block mode messages to be completely ignored. The next normal J1850 message to appear on the bus will be automatically received as long as it contains no errors.

15.7.5 MDLC STOP MODE

This power conserving mode is automatically entered from the Run mode whenever the CPU executes a STOP instruction, or if the CPU executes a WAIT instruction and the WCM bit in the MCR register is previously set. This is the lowest power mode that the MDLC can enter.

A subsequent passive to active transition on the J1850 bus will cause the MDLC to ’wake up’ and generate a non-maskable CPU interrupt request. The MDLC **is not** guaranteed to correctly receive the message which woke it up since it may take some time for the MDLC internal operating clocks to restart and stabilize.

When MDLC Stop mode is entered, the analog circuitry within the transmitter will be put into a power conserving ’sleep’ mode. In MDLC Stop mode the MDLC can neither do wave shaping nor drive any data. Therefore, it is important that the programmer ensures that all transmissions are complete or aborted before putting the MDLC into MDLC Stop mode.

If this mode is entered while the MDLC is receiving a message, the first subsequent received edge will cause the MDLC to immediately wake up, generate a CPU interrupt request, and wait for the MDLC internal operating clocks to restart and stabilize before normal communication can resume. Therefore, the MDLC is not guaranteed to correctly receive that message.

15.7.6 MDLC WAIT MODE

This power conserving mode is automatically entered from the Run mode whenever the CPU executes a WAIT instruction and the WCM bit in the MCR register is previously clear.

A subsequent successfully received message, including one that is in progress at the time that this mode is entered, will cause the MDLC to ’wake up’ and generate a CPU interrupt request if the Interrupt Enable (IE) bit in the MCR register is previously set. This results in less of a power saving, but the MDLC **is** guaranteed to correctly receive the message which woke it up since the MDLC internal operating clocks are kept running.

When MDLC Wait mode is entered, the analog circuitry within the transmitter will be put into a power conserving ’sleep’ mode. In MDLC Wait mode the MDLC can neither do wave shaping nor drive any data. Therefore, it is important the programmer ensures that all transmissions are complete or aborted before putting the MDLC into MDLC Stop or MDLC Wait mode.

15.7.7 CONTROLLING EXTERNAL VOLTAGE REGULATORS

If the application node contains other, off-chip, supply voltage regulators that need to be controlled by J1850 network activity then an output port pin of the MCU must be reserved by the programmer for use as a Power Sense (PSEN) signal. This pin will provide a logical indication of when the MDLC is active through software.

Whenever the MCU is in the normal (Run) mode of operation, the programmer should **assert** the PSEN output on the chosen port pin. The programmer should **negate** the PSEN output just prior to placing the microcontroller in the Stop or Wait mode.

When the MDLC is in the MDLC Stop or MDLC Wait mode of operation, and network activity is sensed, a CPU interrupt request will be generated. As part of the service routine for this interrupt, the programmer should once again **assert** the PSEN output. Note that both V_{CCA} and V_{DD} for the MDLC must be maintained in the MDLC Stop and MDLC Wait modes of operation in order for this to work properly.

SECTION 16 INSTRUCTION SET

The MCU has a set of 62 basic instructions. They can be divided into five different types: register/memory, read-modify-write, branch, bit manipulation, and control. The following paragraphs briefly explain each type. For more information on the instruction set, refer to the *M6805 Family User's Manual* (M6805UM/AD2) or the associated MC68HC05 Data Sheet.

16.1 REGISTER/MEMORY INSTRUCTIONS

Most of these instructions use two operands. One operand is either the accumulator or the index register. The other 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 the following instruction list.

Function	Mnemonic
Load A from Memory	LDA
Load X from Memory	LDX
Store A in Memory	STA
Store X in Memory	STX
Add Memory to A	ADD
Add Memory and Carry to A	ADC
Subtract Memory	SUB
Subtract Memory from A with Borrow	SBC
AND Memory to A	AND
OR Memory with A	ORA
Exclusive OR Memory with A	EOR
Arithmetic Compare A with Memory	CMP
Arithmetic Compare X with Memory	CPX
Bit Test Memory with A (Logical Compare)	BIT
Jump Unconditional	JMP
Jump to Subroutine	JSR
Multiply	MUL

16.2 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 the read-modify-write sequence since it does not modify the value. Do not use these read-modify-write instructions on write-only locations. Refer to the following list of instructions.

Function	Mnemonic
Increment	INC
Decrement	DEC
Clear	CLR
Complement	COM
Negate (Two's Complement)	NEG
Rotate Left Thru Carry	ROL
Rotate Right Thru Carry	ROR
Logical Shift Left	LSL
Logical Shift Right	LSR
Arithmetic Shift Right	ASR
Test for Negative or Zero	TST

16.3 BRANCH INSTRUCTIONS

This set of instructions branches if a particular condition is met; otherwise, no operation is performed. Branch instructions are 2-byte instructions. Refer to the following list for branch instructions.

Function	Mnemonic
Branch Always	BRA
Branch Never	BRN
Branch if Higher	BHI
Branch if Lower or Same	BLS
Branch if Carry Clear	BCC
Branch if Higher or Same	BHS
Branch if Carry Set	BCS
Branch if Lower	BLO
Branch if Not Equal	BNE
Branch if Equal	BEQ
Branch if Half Carry Clear	BHCC
Branch if Half Carry Set	BHCS
Branch if Plus	BPL
Branch if Minus	BMI
Branch if Interrupt Mask Bit is Clear	BMC
Branch if Interrupt Mask Bit is Set	BMS
Branch if Interrupt Line is Low	BIL
Branch if Interrupt Line is High	BIH
Branch to Subroutine	BSR

16.4 BIT MANIPULATION INSTRUCTIONS

The MCU is capable of setting or clearing any read/write bit that resides in the first 256 bytes of the memory space where all port registers, port DDRs, timer, timer control, and on-chip RAM reside. An additional feature allows the software to test and branch on the state of any bit within these 256 locations. The bit set, bit clear, and bit test and branch functions are each implemented with a single instruction. For test and branch instructions, the value

of the bit tested is also placed in the carry bit of the condition code register. The bit set and bit clear instructions are also read-modify-write instructions and should not be used to manipulate write-only locations. Refer to the following list for bit manipulation instructions.

Function	Mnemonic
Set Bit n	BSET n (n = 0. . .7)
Clear Bit n	BCLR n (n = 0. . .7)
Branch if Bit n is Set	BRSET n (n = 0. . .7)
Branch if bit n is Clear	BRCLR n (n = 0. . .7)

16.5 CONTROL INSTRUCTIONS

These instructions are register reference instructions and are used to control processor operation during program execution. Refer to the following list for control instructions.

Function	Mnemonic
Transfer A to X	TAX
Transfer X to A	TXA
Set Carry Bit	SEC
Clear Carry Bit	CLC
Set Interrupt Mask Bit	SEI
Clear Interrupt Mask Bit	CLI
Software Interrupt	SWI
Return from Subroutine	RTS
Return from Interrupt	RTI
Reset Stack Pointer	RSP
No-Operation	NOP
Stop	STOP
Wait	WAIT

16.6 ADDRESSING MODES

The MCU uses ten different addressing modes to provide the programmer with an opportunity to optimize the 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 (3 bytes) permit accessing tables throughout memory. Short and long absolute addressing is also included. One- or 2-byte direct addressing instructions access all data bytes in most applications. Extended addressing permits jump instructions to reach all memory.

The term “effective address” (EA) is used in describing the various addressing modes. Effective address is defined as the address from which the argument for an instruction is fetched or stored.

16.6.1 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 (for example, a constant used to initialize a loop counter).

16.6.2 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 single 2-byte instructions.

16.6.3 EXTENDED

In the extended addressing mode, the effective address of the argument is contained in the 2 bytes following the opcode byte. Instructions with extended addressing mode are capable of referencing arguments anywhere in memory with a single 3-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 shortest form of the instruction.

16.6.4 RELATIVE

The relative addressing mode is only used in branch instructions. In relative addressing, the contents of the 8-bit signed offset byte (which is the last byte of the instruction) is 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 -128 to +127 from the address of the next opcode. 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.

16.6.5 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.

16.6.6 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. The addressing mode is useful for selecting the Kth element in an element table. With this 2-byte instruction, K would typically be in X with the address of the beginning of the table in the instruction. As such, tables may begin anywhere within the first 256 addressable locations and could extend as far as location 510. $1FE is the highest location which can be accessed in this way.

16.6.7 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 2 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.

16.6.8 BIT SET/CLEAR

In the bit set/clear addressing mode, the bit to be set or cleared is part of the opcode, and the byte following the opcode specifies the direct address of the byte in which the specified bit is to be set or cleared. Any read/write register bit in the first 256 locations of memory, including I/O, can be selectively set or cleared with a single 2-byte instruction.

16.6.9 BIT TEST AND BRANCH

The bit test and branch addressing mode is a combination of direct addressing and relative addressing. The bit that is 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. The signed relative 8-bit offset in the third byte is added to the PC if the specified bit is set or cleared in the specified memory location. This single 3-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 branching is from -128 to +127 from the address of the next opcode. The state of the tested bit is also transferred to the carry bit of the condition code register.

16.6.10 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 and/or accumulator as well as the control instructions with no other arguments are included in this mode. These instructions are 1 byte long.

SECTION 17 ELECTRICAL SPECIFICATIONS

17.1 MAXIMUM RATINGS

(Voltages referenced to V_{SS})

Rating	Symbol	Value	Unit
Supply Voltage	V_{BATT}	-0.5 to +42.0	V
Input Voltage	V_{IN}	V_{SS} -0.3 to V_{DD} +0.3	V
Current Drain Per Pin Excluding V_{DD} and V_{SS}	I	25	mA
Operating Temperature Range MC68HC705V8XX (Standard) MC68HC705V8CXX (Extended) XX=B for SDIP, XX=FN for PLCC, XX=FU for QFP	T_A	T_L to T_H 0 to +70 -40 to +85	°C
Storage Temperature Range	T_{STG}	-65 to +150	°C
Write/Erase Cycles (@ 10 ms write time & -40°C, +25°C, +85°C)		10,000	cycles
Data Retention EPROM & EEPROM		10	years

This device contains circuitry to protect the inputs against damage due to high static voltages or electric fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum-rated voltages to this high-impedance circuit. For proper operation, it is recommended that V_{IN} and V_{OUT} be constrained to the range $V_{SS} \le (V_{IN}$ or $V_{OUT}) \le V_{DD}$. Reliability of operation is enhanced if unused inputs are connected to an appropriate logic voltage level (for example, either V_{SS} or V_{DD}).

17.2 THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Unit
Thermal Resistance			
SDIP (56 pin)	θ_{JA}	60	°C/W
PLCC (68 pin)	θ_{JA}	50	°C/W
QFP (64 pin)	θ_{JA}	85	°C/W

17.3 DC ELECTRICAL CHARACTERISTICS

(V_{DD} = 5.0 Vdc ±10%, V_{SS} = 0 Vdc, T_A = -40°C to +85°C, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Output voltage I_{Load} = 10.0 µA	V_{OL} V_{OH}	— V_{DD}-0.1	— —	0.1 —	V
Output High Voltage (I_{Load} = -0.8 mA) PA0-7, PB0-7, PC0-7, PF0-3, PWM	V_{OH}	V_{DD}-0.8	—	—	V
Output Low Voltage (I_{Load} = 1.6 mA) PA0-7, PB0-7, PC0-7, PF0-3,PWM	V_{OL}	---	—	0.4	V
Input High Voltage Ports A,B C V_{IGN}	V_{IH}	0.8×V_{DD} 0.8×V_{BATT}	—	V_{DD} V_{BATT}	V
Input High Voltage Ports D, E, F, $\overline{IRQ}$, $\overline{RESET}$, OSC1	V_{IH}	0.7×V_{DD}	—	V_{DD}	V
Input Low Voltage Ports A, B, C V_{IGN}	V_{IL}	V_{SS} V_{SS}	—	0.4×V_{DD} 0.4×V_{BATT}	V
Input Low Voltage PortsD, E, F, $\overline{IRQ}$, $\overline{RESET}$, OSC1	V_{IL}	V_{SS}	—	0.3×V_{DD}	V
EPROM programming voltage	V_{PP}	19.75	20	20.25	V
EPROM programming current	I_{PP}	—	5	10	mA
V_{DD} Supply Current (see Notes) Run Wait SPI, TIMER, A/D, PWM, COP and LVR enabled Wait above modules off Stop (Regulator disabled, LVR enabled) 25°C 0°C to +70°C (Standard) -40°C to +85°C (Extended) 25°C (LVR disabled)	 I_{DD} I_{DD} I_{DD} I_{DD} I_{DD} I_{DD} I_{DD}	 --- --- --- --- --- --- ---	 5 3 1.2 325 TBD TBD 200	 --- --- --- --- TBD TBD ---	 mA mA mA µA µA µA µA
V_{BATT} Supply Current at 12V Stop Regulator disabled Run Regulator disabled (Transceiver current) Stop Regulator enabled (includes STOP I_{DD} , LVR disabled)	 I_{BATT} I_{BATT} I_{BATT}	 --- --- ---	 25 800 600	 TBD TBD TBD	 µA µA µA
I/O Ports Hi-Z Leakage Current PA0-7, PB0-7, PC0-7, PF0-3	I_{IL}	—	—	±1	µA
Input Current $\overline{RESET}$, $\overline{IRQ}$, OSC1, TCAP, PD0-7,PE0-7, $\overline{SS}$	I_{IN}	—	—	±1	µA
Capacitance * Ports (as Input or Output) $\overline{RESET}$, $\overline{IRQ}$, OSC1, OSC2, TCAP, PD0-7,PE0-7, $\overline{SS}$	C_{OUT} C_{IN}	— —	— —	12 8	pF pF

Low Voltage Reset Inhibit	V_{LVRI}	3.5	—	4.2	V
Low Voltage Reset Recover	V_{LVRR}	3.6	—	4.5	V
Low Voltage Reset Inhibit/Recover Hysteresis	H_{LVR}	0.1	0.2	0.3	V
V_{DD} slew rate rising *	S_{VDDR}	—	—	0.1	V/µs
V_{DD} slew rate falling*	S_{VDDF}	—	—	0.05	V/µs
POR Reset Voltage *	V_{POR}	—	—	100	mV

NOTES:

1. All values shown reflect average measurements.
2. Typical values at midpoint of voltage range, 25°C only.
3. Run (Operating) I_{DD}, Wait I_{DD}: Measured using external square wave clock source to OSC1 (f_{OSC} = 4.2 MHz), all inputs 0.2 Vdc from rail; no DC loads, less than 50 pF on all outputs, C_L = 20 pF on OSC2.
4. Wait, Stop I_{DD}: All ports configured as inputs, V_{IL} = 0.2 Vdc, V_{IH} = V_{DD}-0.2 Vdc.
5. Stop I_{DD} measured with OSC1 = V_{SS}.
6. Wait I_{DD} is affected linearly by the OSC2 capacitance.
7. Total.

* Not Tested.

17.4 REGULATOR ELECTRICAL CHARACTERISTICS

(V_{BATT} = 12 Vdc ±10%, V_{SS} = 0 Vdc, T_A = -40°C to +85°C, unless otherwise noted)

Characteristic	Symbol	Min	Max	Units
Input Voltage[1]	V_{BATT}	8	16	V
Maximum Input Voltage transient amplitude [2,4]	$V_{BATTMAX}$	26.5	40	V
Maximum Input Voltage transient duration [2,4]	$V_{BATTDUR}$	--	200	mS
Output Voltage	V_{DD}	4.75	5.25	V
Output Current [3]	I_O	-	20	mA
Input Bias Current	I_{BIAS}	TBD	TBD	uA
Output Noise Voltage [2]	V_N	-	TBD	uVrms
Ripple Rejection Ratio [2]	RR	-	65	dB
Input - Output Differential Voltage [5] V_{BATT} to V_{DD} with V_{BATT} < min.	V_{DIFF}	1.5*	2.50	V
Short Circuit Current Limit	I_{SHORT}	75	95	mA
Temperature Coefficient [2]	T_C	-	TBD	mV/°C
Line regulation	$V_{LINEREG}$	-	50	mV
Load regulation at 20 mA	$V_{LOADREG}$	-	50	mV

* max I_{DD} of 20mA at V_{DD}=3.5V.

NOTES: (All values shown reflect average measurements.)

1. Maximum voltage of 26.5V for 5 minutes only. If V_{BATT} exceeds $V_{BATTMAX}$, V_{DD} may deviate from min. max limits.
2. Guaranteed by design. Not tested in productions tests.
3. This is the available current out of the V_{DD} pin for off chip usage.
4. This transient may be disruptive but will not be destructive.
5. Total regulator load of 20 mA. Lighter load currents result in lower V_{DIFF}.

17.5 CONTROL TIMING

(V_{DD} = 5.0 Vdc ±10%, V_{SS} = 0 Vdc, T_A = -40°C to +85°C, unless otherwise noted)

Characteristic	Symbol	Min	Max	Units
Frequency of Operation Crystal Oscillator Option External Clock Source	 f_{OSC} f_{OSC}	 0.1 DC	 4.2 4.2	 MHz MHz
Internal Operating Frequency Crystal Oscillator (f_{OSC} ÷ 2) External Clock (f_{OSC} ÷ 2)	 f_{OP} f_{OP}	 --- DC	 2.1 2.1	 MHz MHz
Cycle Time (1 ÷ f_{OP})	t_{CYC}	476	---	nS
Crystal Oscillator Startup Time (Crystal Oscillator option)	t_{OXON}	---	TBD	t_{CYC}
Stop Recovery Startup Time (Crystal Oscillator option)	t_{ILCH}	---	TBD	t_{CYC}
RESET Pulse Width Low (See Figure 8-1)	t_{RL}	120	---	ns
IRQ Interrupt Pulse width low (Edge-Triggered)	t_{ILHI}	120	---	nS
IRQ Interrupt Pulse Period (See Figure 8-1)	t_{ILIL}	note 3	---	t_{CYC}
PA0-7, PC0-7 Interrupt Pulse Width High (Edge-Triggered)	t_{IHIL}	120	---	nS
PA0-7, PC0-7 Interrupt Pulse Period	t_{IHIH}	note 3	---	t_{CYC}
OSC1 Pulse Width	t_{OSC1}	90	---	nS
EPROM Programming Time per Byte	t_{EPGM}	4	---	mS
EEPROM Programming Time per Byte	t_{EEPGM}	10	---	mS
EEPROM Erase Time per Byte	t_{EBYT}	10	---	mS
EEPROM Erase Time per Block	t_{EBLOCK}	10	---	mS
EEPROM Bulk Erase Time	t_{EBULK}	10	---	mS
EEPROM Programming Voltage Discharge Period	t_{FPV}		10	µS
RC Oscillator Stabilization Time	t_{RCON}	--	5.0	µs
16-Bit Timer Resolution (note 2) Input Capture Pulse Width (See Figure 11-2.) Input Capture Period	 t_{RESL} t_{TH},t_{TL} t_{TLTL}	 4.0 125 note 4	 -- -- --	 t_{CYC} nS t_{CYC}

NOTES:
1. V_{DD} = 5.0 Vdc ± 10%, V_{SS} = 0 Vdc, T_A = T_L to T_H
2. The 2-bit timer prescaler is the limiting factor in determining timer resolution.
3. The minimum period t_{ILIL} or t_{IHIH} should not be less than the number of cycles it takes to execute the interrupt service routine plus 19 t_{CYC}.
4. The minimum period t_{TLTL} should not be less than the number of cycles it takes to execute the capture interrupt service routine plus 24 t_{CYC}.

17.6 A/D CONVERTER CHARACTERISTICS

(V_{CCA} = 5.0 Vdc ±10%, V_{SSA1} = 0 Vdc, T_A = -40°C to +85°C, unless otherwise noted)

Characteristic	Min	Max	Unit	Comments
Resolution	8	8	Bits	
Absolute Accuracy (V_{REFL} = 0 V, V_{REFH} = V_{DD})	—	±1 1/2	LSB	Including quantization
Conversion Range V_{REFH} V_{REFL}	V_{REFL} V_{REFL} -0.1	V_{REFH} V_{DD} V_{REFH}	V V V	A/D accuracy may decrease proportionately as V_{REFH} is reduced below 4.0
Power-up Time	—	100	µs	
Input Leakage PD0-PD7, PE0-PE7 V_{REFL},V_{REFH}	 — —	 ±1 ±1	 µA µA	
Conversion Time (Includes Sampling Time)	32	32	T_{AD}*	
Monotonicity	Inherent (Within Total Error)			
Zero Input Reading	00	01	Hex	V_{IN} = 0V
Full-scale Reading	FE	FF	Hex	V_{IN} = V_{REFH}
Sample Time	12	12	T_{AD}*	
Input Capacitance	—	8	pF	Not Tested
Analog Input Voltage	V_{REFL}	V_{REFH}	V	
A/D On Current Stabilization Time (t_{ADON})		100	µs	
RC Ocillator Stabilization Time (t_{RCON})		5	µs	

*T_{AD} = t_{CYC} if clock source equals MCU.

17.7 DC ELECTRICAL CHARACTERISTICS

(V_{DD}=3.5-4.75 Vdc ±10%, V_{SS}=0 Vdc, T_A=-40°C to +85°C, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Output voltage I_{Load} = 10.0 µA	V_{OL} V_{OH}	— V_{DD}-0.1	— —	0.1 —	V V
Output High Voltage (I_{Load} = -0.2 mA) PA0-7, PB0-7,PC0-7,PF0-3,PWM	V_{OH}	V_{DD}-0.3	—	—	V
Output Low Voltage (I_{Load} = 0.4 mA) PA0-7, PB0-7,PC0-7,PF0-3,PWM	V_{OL}	—	—	0.3	V
Input High Voltage Ports A, B, C	V_{IH}	0.8×V_{DD}	—	V_{DD}	V
Input High Voltage Ports D, E,F, $\overline{IRQ}$, $\overline{RESET}$, OSC1	V_{IH}	0.7×V_{DD}	—	V_{DD}	V
Input Low Voltage Ports A, B, C	V_{IL}	V_{SS}	—	0.4×V_{DD}	V
Input Low Voltage Ports D, E,F, $\overline{IRQ}$, $\overline{RESET}$, OSC1	V_{IL}	V_{SS}	—	0.3×V_{DD}	V
V_{DD} Supply Current (see Notes) Run Wait SPI, TIMER, A/D, PWM, COP and LVR enabled Wait above modules off Stop (Regulator disabled, LVR enabled) 25°C 0°C to +70°C (Standard) -40°C to +85°C (Extended) 25°C (LVR disabled)	 I_{DD} I_{DD} I_{DD} I_{DD} I_{DD} I_{DD} I_{DD}	 TBD TBD TBD TBD TBD TBD TBD	 TBD TBD TBD TBD TBD TBD TBD	 --- --- --- --- TBD TBD ---	 mA mA mA µA µA µA µA
I/O Ports Hi-Z Leakage Current PA0-7, PB0-7, PC0-7, PF0-3	I_{OZ}	—	—	±1	µA
Input Current $\overline{RESET}$, $\overline{IRQ}$, OSC1, TCAP,$\overline{SS}$ PD0-PD7,PE0-7	I_{IN} I_{IN}	— —	— —	±1 ±1	µA µA
Capacitance* Ports (as Input or Output) $\overline{RESET}$, $\overline{IRQ}$, OSC1,OSC2,TCAP,$\overline{SS}$	 C_{OUT} C_{IN}	 — —	 — —	 12 8	 pF pF

NOTES:

1. All values shown reflect average measurements.
2. Typical values at midpoint of voltage range, 25°C only.
3. Wait I_{DD}: Only timer system active.
4. Run (Operating) I_{DD}, Wait I_{DD}: Measured using external square wave clock source (f_{osc}=4.2 MHz), all inputs 0.2 V from rail; no dc loads, less than 50pF on all outputs, C_L=20 pF on OSC2.
5. Wait, Stop I_{DD}: All ports configured as inputs, V_{IL}=0.2 V, V_{IH}=V_{DD}-0.2 V.
6. Stop I_{DD} measured with OSC1=V_{SS}.
7. Wait I_{DD} is affected linearly by the OSC2 capacitance.
8. The MDLC module will not operate below 4.65 volts.

* Not Tested.

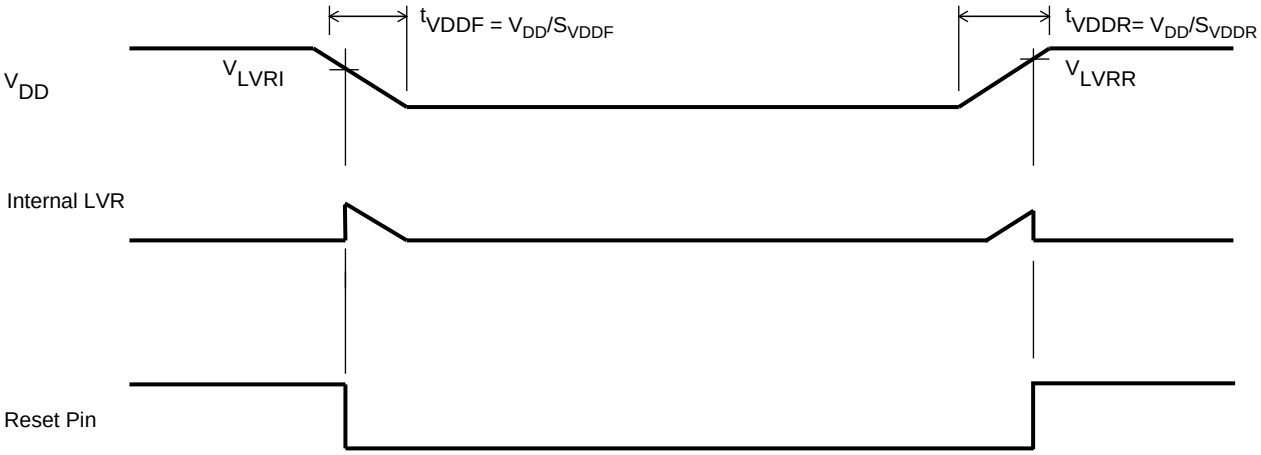
17.8 CONTROL TIMING

(V_{DD} =3.5-4.75 Vdc ±10%, V_{SS} = 0 Vdc, T_A = -40°C to +85°C, unless otherwise noted)

Characteristic	Symbol	Min	Max	Units
Frequency of Operation Crystal Oscillator Option External Clock Source	 f_{OSC} f_{OSC}	 0.1 DC	 4.2 4.2	 MHz MHz
Internal Operating Frequency Crystal Oscillator (f_{OSC} ÷ 2) External Clock (f_{OSC} ÷ 2)	 f_{OP} f_{OP}	 --- DC	 2.1 2.1	 MHz MHz
Cycle Time (1 ÷ f_{OP})	t_{CYC}	476	---	nS
Crystal Oscillator Startup Time (Crystal Oscillator option)	t_{OXON}	---	TBD	t_{CYC}
Stop Recovery Startup Time (Crystal Oscillator option)	t_{ILCH}	---	TBD	t_{CYC}
RESET Pulse Width Low	t_{RL}	1.5	---	t_{CYC}
IRQ Interrupt Pulse Width Low (Edge-Triggered)	t_{ILIH}	120	---	nS
IRQ Interrupt Pulse Period	t_{ILIL}	note 3	---	t_{CYC}
PA0-7, PC0-7 Interrupt Pulse Width High (Edge-Triggered)	t_{IHIL}	120	---	nS
PA0-7, PC0-7 Interrupt Pulse Period	t_{IHIH}	note 3	---	t_{CYC}
OSC1 Pulse Width	t_{osc1}	90	---	nS
EEPROM Programming Time per Byte	t_{EEPGM}	15	---	mS
EEPROM Erase Time per Byte	t_{EBYT}	15	---	mS
EEPROM Erase Time per Block	t_{EBLOCK}	15	---	mS
EEPROM Bulk Erase Time	t_{EBULK}	15	---	mS
EEPROM Programming Voltage Discharge Period	t_{FPV}		10	µS
RC Oscillator Stabilization Time	t_{RCON}	--	5.0	µs
16-Bit Timer Resolution (note 2) Input Capture Pulse Width (See Figure 11-2) Input Capture Period	 t_{RESL} t_{TH},t_{TL} t_{TLTL}	 4.0 250 note 4	 -- -- --	 t_{CYC} nS t_{CYC}

NOTES:
1. The 2-bit timer prescaler is the limiting factor in determining timer resolution.
2. The minimum period t_{ILIL} or t_{IHIH} should not be less than the number of cycles it takes to execute the interrupt service routine plus 19 t_{CYC}.
3. The minimum period t_{TLTL} should not be less than the number of cycles it takes to execute the capture interrupt service routine plus 24 t_{CYC}.

17.9 LVR TIMING DIAGRAM



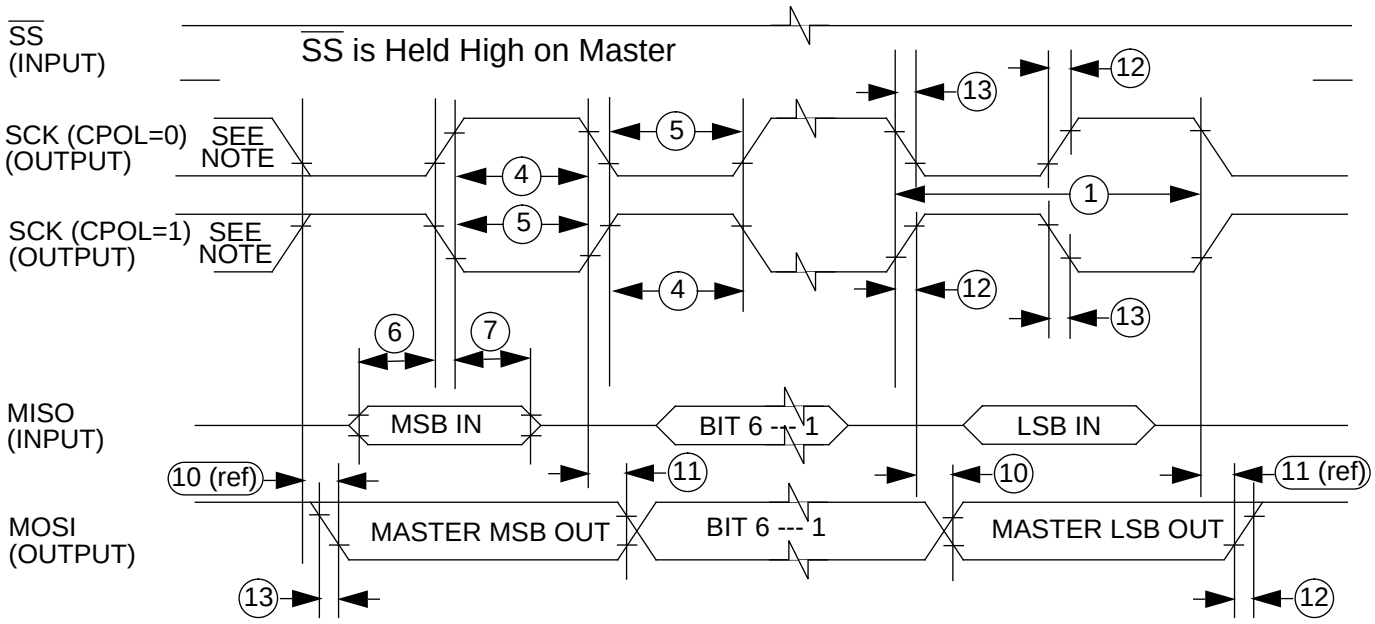
17.10 SPI TIMING

SERIAL PERIPHERAL INTERFACE (SPI) TIMING

(V_{DD} = 5.0 Vdc ±10%, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H)

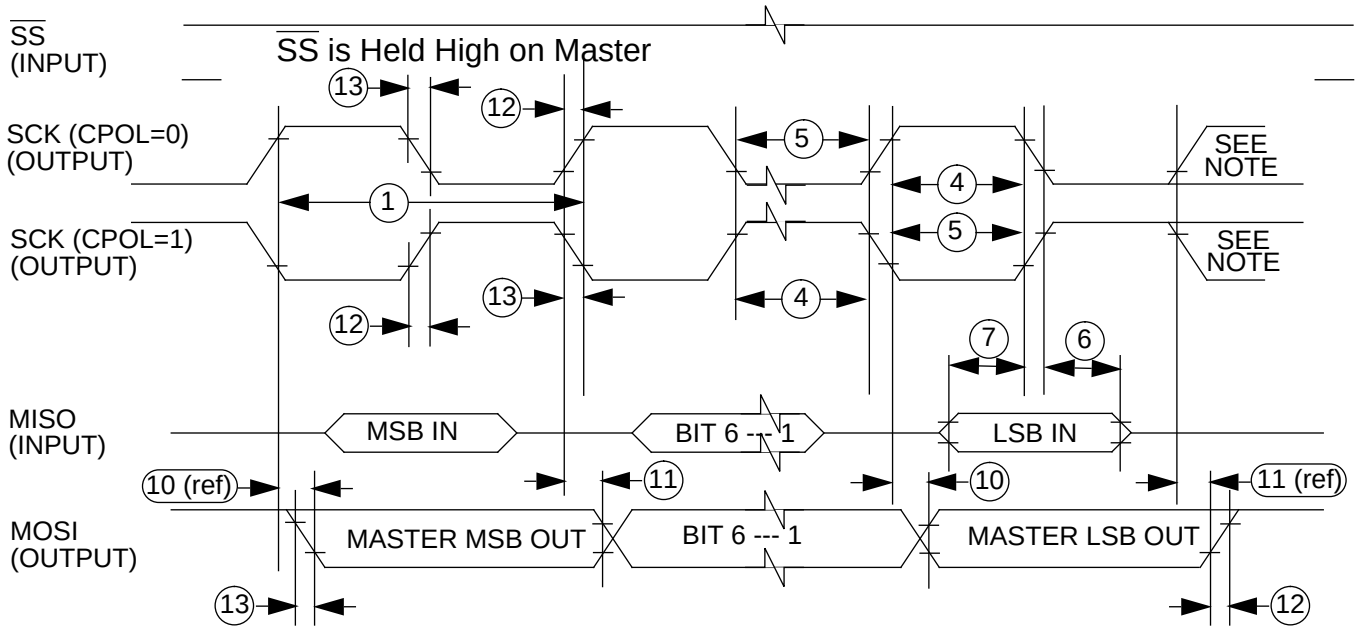
Num	Characteristic	Symbol	Min	Max	Unit
	Operating Frequency Master Slave	 $f_{OP(M)}$ $f_{OP(S)}$	 dc dc	 0.5 4.2	 f_{op} MHz
1	Cycle Time Master Slave	 $t_{CYC(M)}$ $t_{cyc(s)}$	 2.0 240	 — —	 t_{cyc} ns
2	Enable Lead Time Master Slave	 $t_{LEAD(M)}$ $t_{LEAD(S)}$	 * 240	 — —	 ns ns
3	Enable Lag Time Master Slave	 $t_{LAG(M)}$ $t_{LAG(S)}$	 * 240	 — —	 ns ns
4	Clock (SCK) High Time Master Slave	 $t_{W(SCKH)M}$ $t_{W(SCKH)S}$	 340 190	 — —	 ns ns
5	Clock (SCK) Low Time Master Slave	 $t_{W(SCKL)M}$ $t_{W(SCKL)M}$	 340 190	 — —	 ns ns
6	Data Setup Time (Inputs) Master Slave	 $t_{SU(M)}$ $t_{SU(S)}$	 100 100	 — —	 ns ns
7	Data Hold Time (Inputs) Master Slave	 $t_{H(M)}$ $t_{H(S)}$	 100 100	 — —	 ns ns
8	Access Time (Time to Data Active from High-Impedance State) Slave	 t_A	 0	 120	 ns
9	Disable Time (Hold Time to High-Impedance State) Slave	 t_{DIS}	 —	 240	 ns
10	Data Valid (After Enable Edge)**	$t_{V(S)}$	—	240	ns
11	Data Hold Time (Output) (After Enable Edge)	t_{HO}	0	—	ns
12	Rise Time (20% V_{DD} to 70% V_{DD}, C_L =200 pF) SPI Outputs (SCK, MOSI, and MISO) SPI Inputs (SCK, MOSI, MISO, and $\overline{SS}$)	 t_{RM} t_{RS}	 — —	 100 2.0	 ns us
13	Fall Time (70% V_{DD} to 20% V_{DD}, C_L =200 pF) SPI Outputs (SCK, MOSI, and MISO) SPI Inputs (SCK, MOSI, MISO, and $\overline{SS}$)	 t_{FM} t_{FS}	 — —	 100 2.0	 ns us

* Signal production depends on software.
** Assumes 200 pF load on all SPI pins.



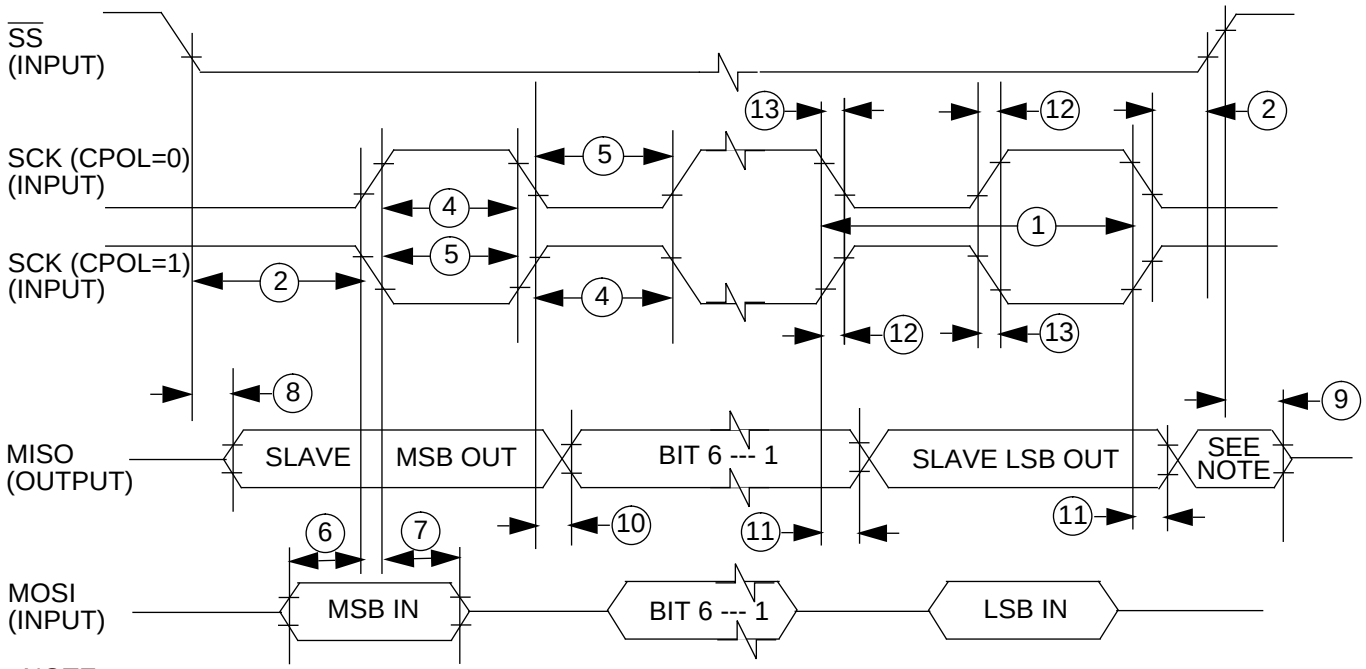
NOTE: This first clock edge is generated internally but is not seen at the SCK pin.

Figure 17-1: SPI MASTER TIMING (CPHA = 0)



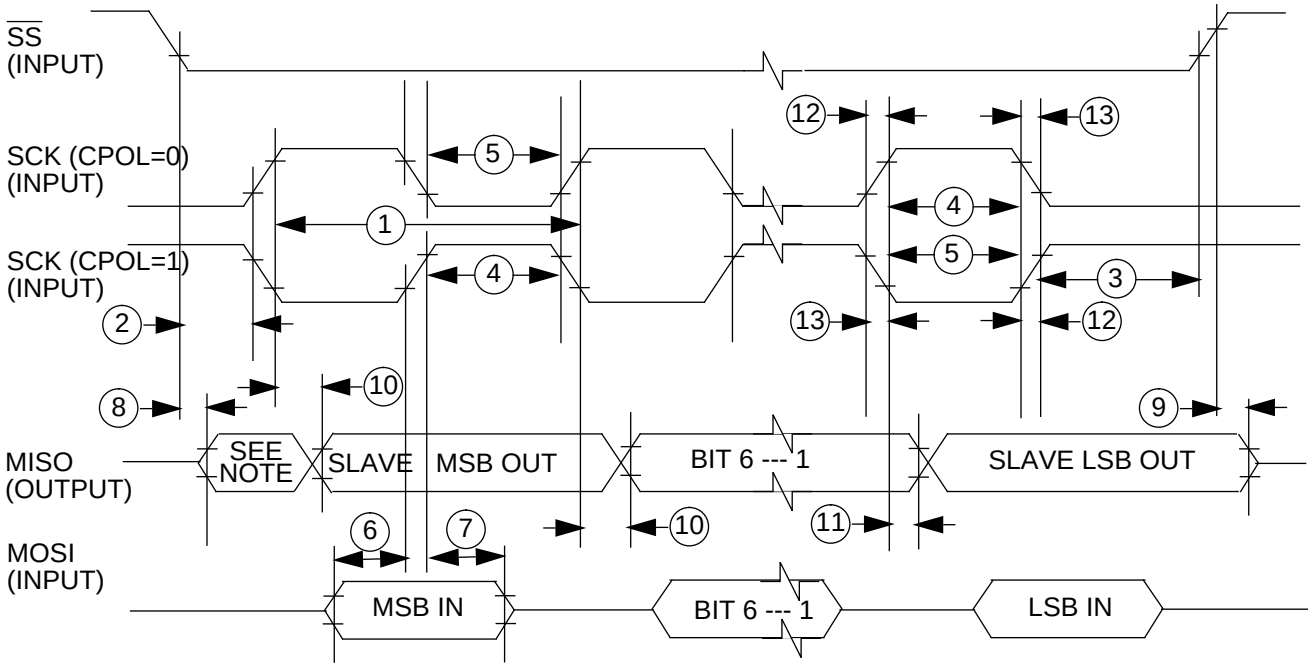
NOTE: This last clock edge is generated internally but is not seen at the SCK pin.

Figure 17-2: SPI MASTER TIMING (CPHA = 1)



NOTE: Not defined but normally LSB of character previously transmitted.

Figure 17-3: SPI SLAVE TIMING (CPHA = 0)



NOTE: Not defined but normally LSB of character previously transmitted.

Figure 17-4: SPI SLAVE TIMING (CPHA = 1)

17.11 MDLC ELECTRICAL SPECIFICATIONS

17.11.1 ABSOLUTE MAXIMUM RATINGS

(Voltages referenced to V_{SSA2} unless otherwise indicated.)

Rating	Symbol	Min.	Typ.	Max.	Unit
Battery Voltage (V_{BATT})	V_{BATT}	-0.3	12.0	16.0	V
Reference Supply Voltage (V_{CC})	V_{CCA}	-0.3	5.0	7.0	V
Bus Voltage (BUS)	V_{BUS}	-2.0	0 to 7.0	18.0	V
Transient Voltage (V_{BATT}, BUS, TBD max. time)	V_{TRAN}	-2.0	—	+24.0	V
Bus Voltage with respect to V_{BATT}	V_{BUS}	--	--	10.0	V

17.11.2 OPERATING CONDITIONS

(Voltages referenced to V_{SSA})

Rating	Symbol	Min.	Typ.	Max.	Unit
Battery Voltage (V_{BATT}) for proper MDLC operation	V_{BATT}	9.0	12.0	16.0	V
Battery Supply Current (Run)	I_{BATT}	—	5	50	mA
Battery Supply Current (Stop)	I_{BATT}	—	5	50	µA
Reference Supply Voltage (V_{CC})	V_{CCA}	4.75	5.00	5.25	V
Reference Supply Current (Run)	I_{CCA}	—	0.5	2	mA
Reference Supply Current (Stop)	I_{CCA}	—	40	50	µA

17.11.3 TRANSMITTER D.C. ELECTRICAL CHARACTERISTICS

(Total network Bus to V_{ssa2} resistance = 245Ω to 15kΩ,
(V_{BATT} = 9 to 16V, V_{CC} = 5.0 V ±5%, V_{SSA} = 0 V, T_A = -40°C to +85°C, unless otherwise noted)

Characteristic	Symbol	Min.	Max.	Unit
Output High Voltage (BUS)	V_{OH}	6.25	8.0	V
Output Low Voltage (BUS)	V_{OL}	0	1.5	V
Output Current (BUS, Loss of V_{BATT} or V_{SSA}, V_{OUT}= 0V, V_{IN} = 0 to 8V)	I_{OUT}	—	±1	mA
Output Current (BUS, Tx = passive, V_{IN} = 12V)	I_{OUT}	—	-200	mA
Output Current (BUS, Tx = active, V_{IN} = 0V)	I_{OUT}	—	200	mA
Low voltage inhibit of transmitter for V_{BATT}	V_{TIVB}	6.0	7.5	V
Low voltage inhibit of transmitter for V_{DD}	V_{RIVD}	4.5	4.75	V

17.11.4 TRANSMITTER A.C. ELECTRICAL CHARACTERISTICS

(V_{BATT} = 9 to 16V, V_{CCA} = 5.0V ±5%, V_{SSA2} = 0V, T_A = -40°C to +85°C, unless otherwise noted)

Characteristic	Number	Symbol	Min.	Typ.	Max.	Unit
Transmitter Rise Time (REXT1 = 31.6kΩ, REXT2 = 24.9kΩ)	1	t_R	13.0	14.5	16.0	µs
Transmitter Fall Time (REXT1 = 31.6kΩ, REXT2 = 24.9kΩ)	2	t_F	13.0	14.5	16.0	µs

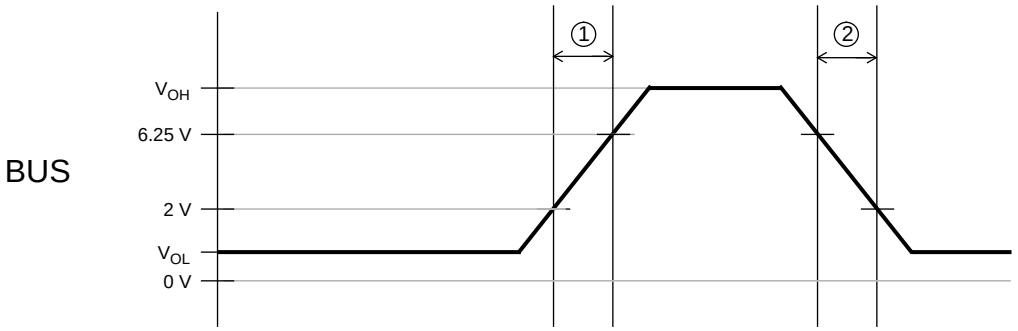


Figure 17-5: Transmitter A.C. Electrical Characteristics

17.11.5 RECEIVER D.C. ELECTRICAL CHARACTERISTICS

(V_{BATT} = 9 to 16V, V_{CCA} = 5.0V ±5%, V_{SSA2} = 0V, T_A = -40°C to +85°C, unless otherwise noted)

Characteristic	Symbol	Min.	Max.	Unit
Input High Voltage (BUS)	V_{IH}	4.25	—	V
Input Low Voltage (BUS)	V_{IL}	—	3.5	V
Input Current (BUS, Normal operation, Tx = passive, V_{in} = 0 to 8V)	I_{IN}	—	±10	µA

17.11.6 TRANSMITTER VPW SYMBOL TIMINGS

(f_{MDLC} = 1.048576 MHz, V_{BATT} = 12V, V_{CC} = 5.0V, V_{SSA} = 0V, T_A = 25°C, unless otherwise noted)

Characteristic	Number	Symbol	Min.	Typ.	Max.	Unit
Passive logic 0	10	T_{tvp1}	62.0	64.0	66.0	µs
Passive logic 1	11	T_{tvp2}	126.0	128.0	130.0	µs
Active logic 0	12	T_{tva1}	126.0	128.0	130.0	µs
Active logic 1	13	T_{tva2}	62.0	64.0	66.0	µs
Start of Frame (SOF)	14	T_{tva3}	198.0	200.0	202.0	µs
End of Data (EOD)	15	T_{tvp3}	198.0	200.0	202.0	µs
End of Frame	16	T_{tv4}	278.0	280.0	282.0	µs
Inter-Frame Separator (IFS)	17	T_{tv6}	298.0	300.0	302.0	µs

17.11.7 RECEIVER VPW SYMBOL TIMINGS

(f_{MDLC} = 1.048576 MHz, V_{BATT} =12V, V_{CCA} = 5.0V, V_{SSA2} = 0V, T_A = 25°C, unless otherwise noted)

Characteristic	Number	Symbol	Min.	Typ.	Max.	Unit
Passive logic 0	10	T_{rvp1}	34.0	64.0	96.0	µs
Passive logic 1	11	T_{rvp2}	96.0	128.0	163.0	µs
Active logic 0	12	T_{rva1}	96.0	128.0	163.0	µs
Active logic 1	13	T_{rva2}	34.0	64.0	96.0	µs
Start of Frame (SOF)	14	T_{rva3}	163.0	200.0	239.0	µs
End of Data (EOD)	15	T_{rvp3}	163.0	200.0	239.0	µs
End of Frame	16	T_{rv4}	239.0	280.0	320.0	µs
Break	18	T_{rv7}	239.0	800.0	—	µs

Note: The receiver symbol timing boundaries are subject to an uncertainty of ± 1 T_{mdlc} µs due to sampling considerations.

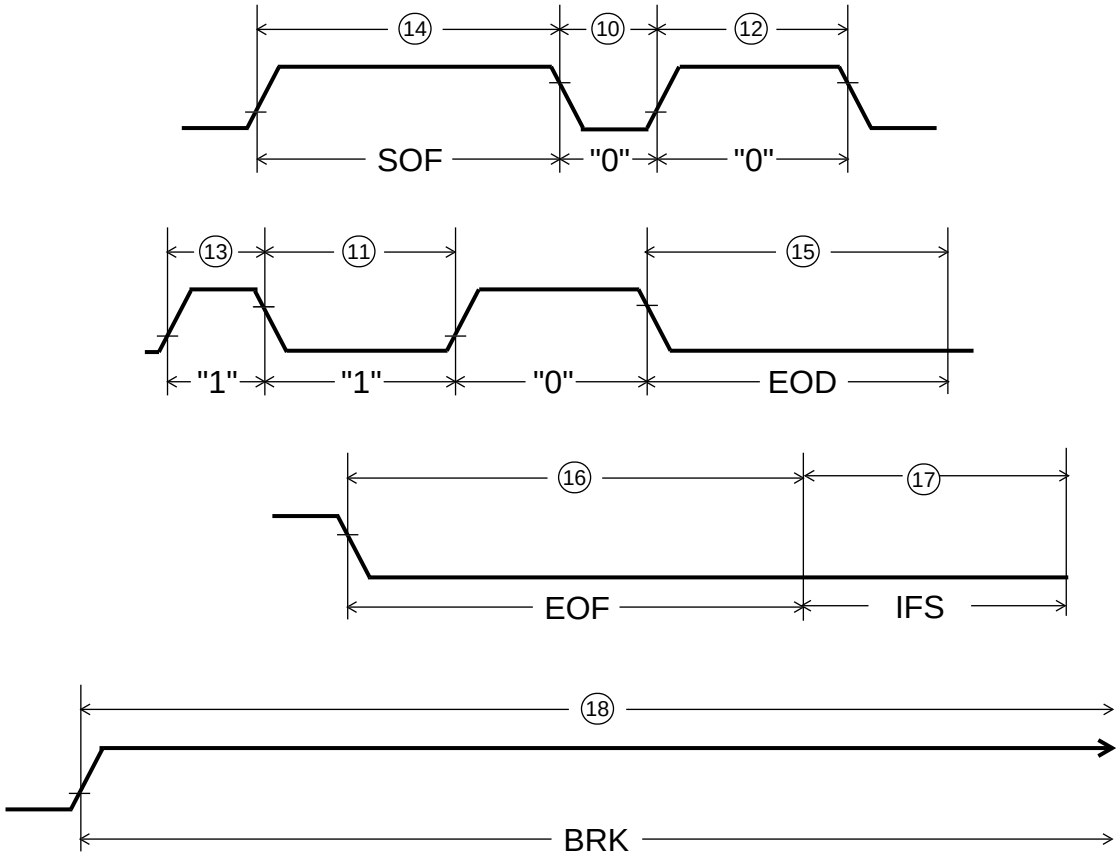


Figure 17-6: Variable Pulse Width Modulation (VPW) Symbol Timings

SECTION 18 705V8 BEHAVIOR DURING EMULATION

This section covers the functions of the MCU that behave differently during emulation using either the 705V8/05V7 EVS or MMDS emulation systems. Since the 705V8/05V7 device itself is used in the emulator system and operates in TEST mode, some extra software operations must be performed to make the device behave as it will in the users application (USER mode). The circuits that require special treatment are described in the sections that follow.

18.1 COP

A value of $04 should be written to location $003F to enable the COP circuitry. The COP circuit is defaulted off to allow ease of factory testing. It should be noted that enabling the COP with this bit will enable the COP regardless of the state of the COPEN bit in the MOR register.

18.2 MDLC

The MDLC requires bit 2 and 3 to be set in the MDLC Control Register ($0E) to allow normal operation of the MDLC. If these two bits are not set during emulation the MDLC will not transmit or receive.

18.3 REGULATOR

The voltage regulator should not be enabled in the device used in the top board of the emulator system. This may result in a supply voltage contention and may overheat and damage the emulator system.

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.