MOTOROLA
SEMICONDUCTOR
TECHNICAL DATA

Order this document
by MC68HC16S2TS/D

MC68HC16S2

Technical Summary
16-Bit Modular Microcontroller

1 Introduction

The MC68HC16S2 is a high-speed 16-bit microcontroller. It is a member of the MC68300/M68HC16 family.

M68HC16 microcontrollers are built up from standard modules that interface through a common inter-module bus (IMB). Standardization facilitates rapid development of devices tailored for specific applications.

The MCU incorporates a 16-bit central processing unit (CPU16), a system integration module (SIM), and a 2-Kbyte standby RAM module (SRAM).

The MCU clock can either be synthesized from an external reference or input directly. Operation with a 32.768 kHz reference frequency is standard. The maximum system clock speed is 25.17 MHz. System hardware and software allow changes in clock rate during operation. Because MCU operation is fully static, register and memory contents are not affected by clock rate changes.

High-density complementary metal-oxide semiconductor (HCMOS) architecture makes the basic power consumption of the MCU low. Power consumption can be minimized by stopping the system clock. The M68HC16 instruction set includes a low-power stop (LPSTOP) command that efficiently implements this capability.

Table 1 MC68HC16S2 Ordering Information

Package Type	Frequency (MHz)	Temperature	Package Order Quantity	Order Number
100-pin TQFP	20.97 MHz	– 40 to + 85 °C	2	SPMC16S2CPU20
			84	MC68HC16S2CPU20
			420	MC16S2CPU20B1
	25.17 MHz	– 40 to + 85 °C	2	SPMC16S2CPU25
			84	MC68HC16S2CPU25
			420	MC16S2CPU25B1

This document contains information on a new product. Specifications and information herein are subject to change without notice.

TABLE OF CONTENTS

1.1 Features

- CPU16
  - — 16-bit architecture
  - — Full set of 16-bit instructions
  - — Three 16-bit index registers
  - — Two 16-bit accumulators
  - — Control-oriented digital signal processing capability
  - — One Mbyte of program memory and one Mbyte of data memory
  - — High-level language support
  - — Fast interrupt response time
  - — Background debugging mode
  - — Fully static operation
- System Integration Module (SIM)
  - — External bus support
  - — Programmable chip select outputs
  - — System protection logic
  - — Watchdog timer, clock monitor and bus monitor
  - — Two 8-bit dual function input/output ports
  - — One 7-bit dual function output port
  - — Phase-locked loop (PLL) clock system
- Standby RAM Module (SRAM)
  - — 2 Kbytes of static RAM
  - — External standby voltage supply input

1.2 Block Diagram

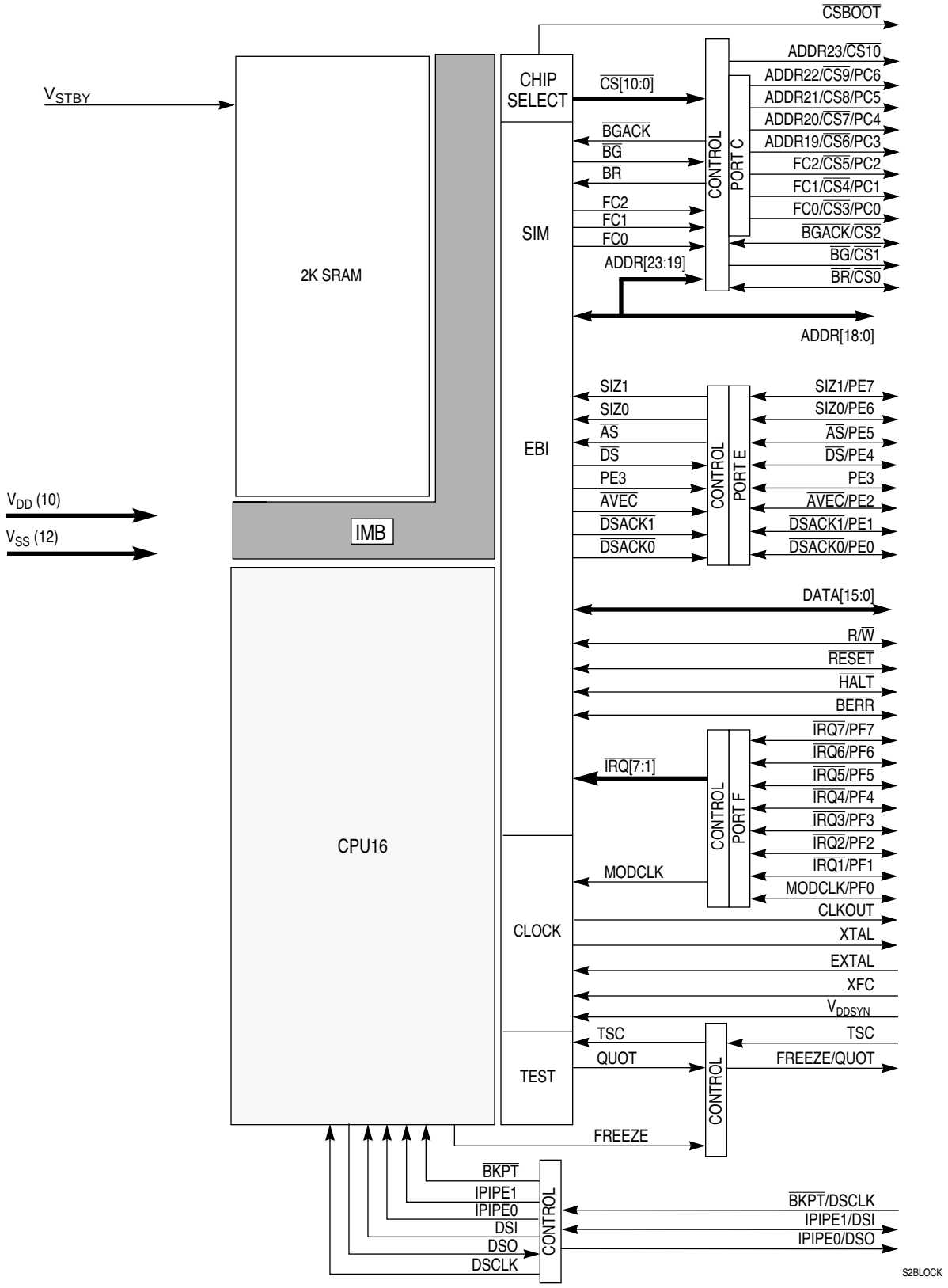


Figure 1 MC68HC16S2 Block Diagram

1.3 Pin Assignments

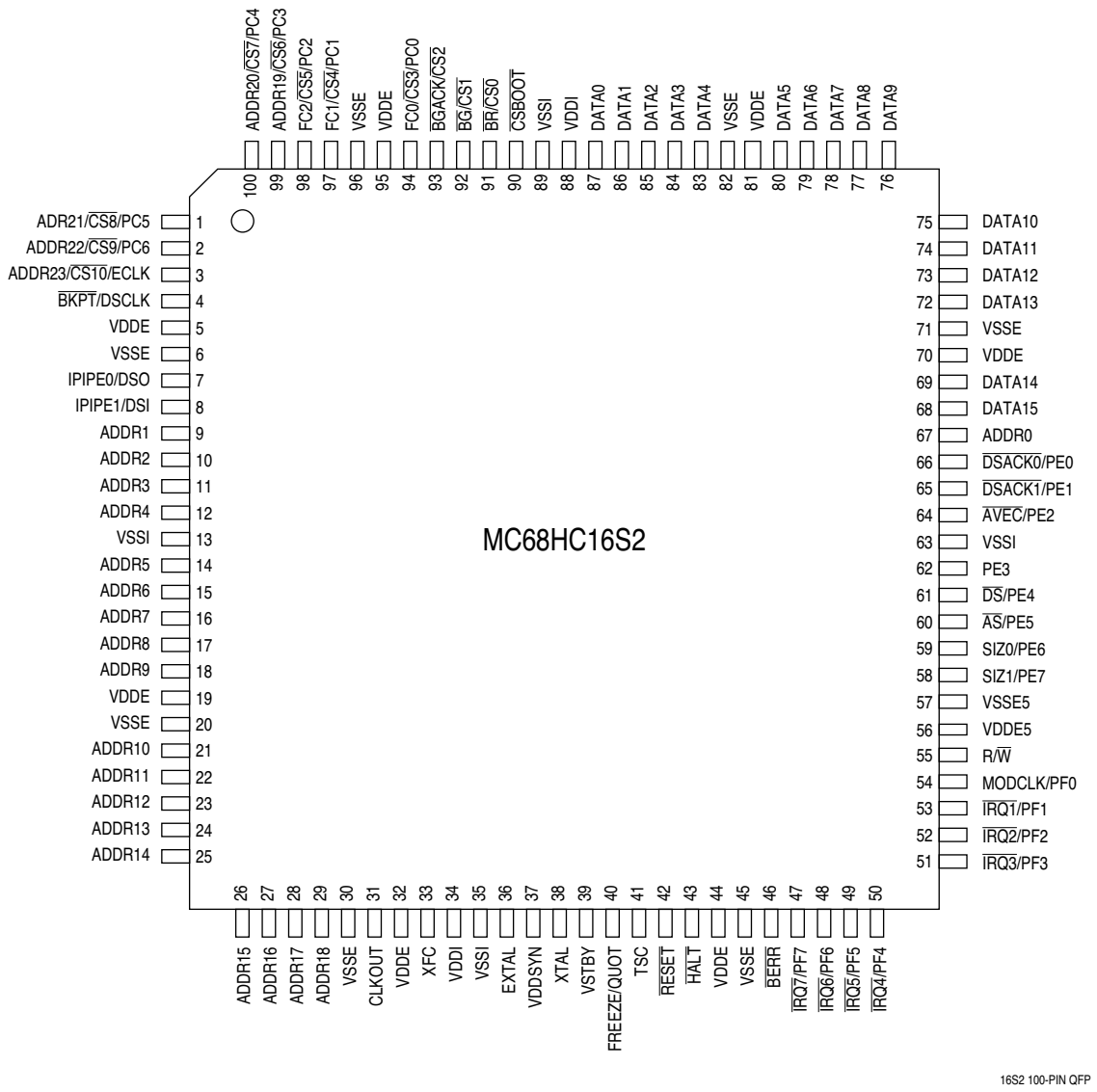


Figure 2 MC68HC16S2 Pin Assignments

1.4 Address Map

Figure 3 is a map of the MCU internal addresses. Although there are 24 intermodule bus (IMB) address lines, the CPU16 uses only ADDR[19:0]. ADDR[23:20] follow the logic state of ADDR19. Addresses $080000 to $F7FFFF are not accessible. The RAM array is positioned by the base address register in the associated RAM control block. Unimplemented blocks are mapped externally.

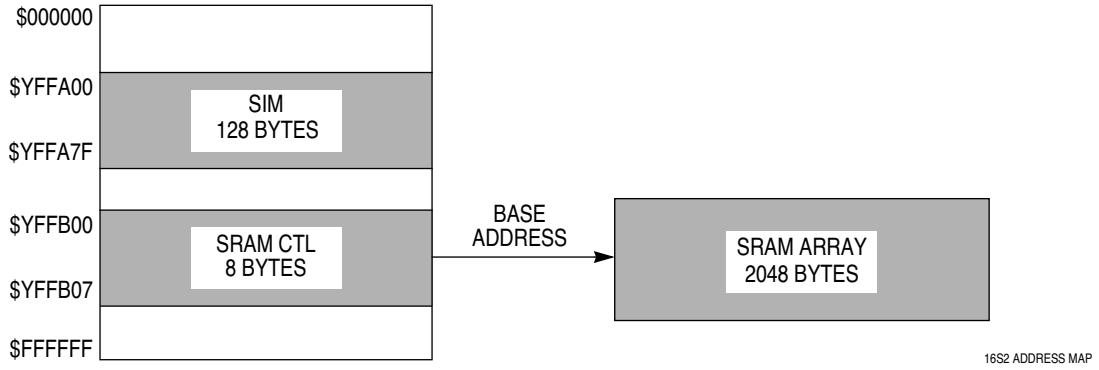


Figure 3 MC68HC16S2 Address Map

1.5 Intermodule Bus

The intermodule bus (IMB) is a standardized bus developed to facilitate both design and operation of modular microcontrollers. It contains circuitry to support exception processing, address space partitioning, multiple interrupt levels, and vectored interrupts. The standardized modules in the MC68HC16S2 communicate with one another and with external components through the IMB. Although the full IMB supports 24 address and 16 data lines, the MC68HC16S2 uses only 16 data lines and 20 address lines. Because the CPU16 uses only 20 address lines, ADDR[23:20] follow the state of ADDR19.

2 Signal Descriptions

2.1 Pin Characteristics

Table 2 shows MCU pins and their characteristics. All inputs detect CMOS logic levels. All inputs can be put in a high impedance state, but the method of doing this differs depending upon pin function. Refer to **Table 4** for a description of output drivers. An entry in the discrete I/O column of the MCU pin characteristics table indicates that a pin has an alternate I/O function. The port designation is given when it applies. Refer to the MCU block diagram for information about port organization.

Table 2 MCU Pin Characteristics

Pin Mnemonic	Output Driver	Input Synchronized	Input Hysteresis	Discrete I/O	Port Designation
ADDR23/$\overline{CS10}$/ECLK	A	Yes	No	—	—
ADDR[22:19]/$\overline{CS[9:6]}$	A	Yes	No	O	PC[6:3]
ADDR[18:0]	A	Yes	No	—	—
$\overline{AS}$	B	Yes	No	I/O	PE5
$\overline{AVEC}$	B	Yes	No	I/O	PE2
$\overline{BERR}$	B	Yes[1]	No	—	—
$\overline{BG}$/$\overline{CS1}$	B	—	—	—	—
$\overline{BGACK}$/$\overline{CS2}$	B	Yes	No	—	—
$\overline{BKPT}$/DSCLK	—	Yes	Yes	—	—
$\overline{BR}$/$\overline{CS0}$	B	Yes	No	—	—
CLKOUT	A	—	—	—	—
$\overline{CSBOOT}$	B	—	—	—	—
DATA[15:0]	Aw	Yes[2]	No	—	—
$\overline{DS}$	B	Yes	No	I/O	PE4
$\overline{DSACK[1:0]}$	B	Yes	No	I/O	PE[1:0]
EXTAL	—	—	Yes	—	—
FC[2:0]/$\overline{CS[5:3]}$	A	Yes	No	O	PC[2:0]
FREEZE/QUOT	A	—	—	—	—
$\overline{HALT}$	Bo	Yes[1]	No	—	—
IPIPE0/DSO	A	—	—	—	—
IPIPE1/DSI	A	Yes	Yes	—	—
$\overline{IRQ[7:1]}$	B	Yes	Yes	I/O	PF[7:1]
MODCLK[2]	B	Yes	No	I/O	PF0
R/$\overline{W}$	A	Yes	No	—	—
$\overline{RESET}$	Bo	Yes	Yes	—	—
PE3	B	Yes	Yes	I/O	PE3
SIZ[1:0]	B	Yes	No	I/O	PE[7:6]
TSC	—	Yes	Yes	—	—
XFC	—	—	—	—	—
XTAL	—	—	—	—	—

NOTES:
1. $\overline{HALT}$ and $\overline{BERR}$ synchronized only if late $\overline{HALT}$ or $\overline{BERR}$.
2. DATA[15:0] are synchronized during reset only. MODCLK is synchronized only when used as a port I/O pin.

2.2 Power Connections

Table 3 MCU Power Connections

Pin	Description
V_{STBY}	Standby RAM power
V_{DDSYN}	Clock synthesizer power
V_{SSE} V_{DDE}	External periphery output driver power (source and drain)
$V_{SSI,}$ V_{DDI}	Internal module power (source and drain)

2.3 Output Driver Types

Table 4 MCU Output Driver Types

Type	I/O	Description
A	O	Output only signals that are always driven; no external pull-up required
Aw	O	Type A output with weak P-channel pull-up during reset
B	O	Three-state output that includes circuitry to pull up output before high impedance is established to ensure rapid rise time. An external holding resistor is required to maintain logic level while the pin is in the high-impedance state.
Bo	O	Type B output that can be operated in an open-drain mode.

2.4 Signal Characteristics

Table 5 MCU Signal Characteristics

Signal Name	MCU Module	Signal Type	Active State
ADDR[23:0]	SIM	Bus	—
$\overline{AS}$	SIM	Output	0
$\overline{AVEC}$	SIM	Input	0
$\overline{BERR}$	SIM	Input	0
$\overline{BG}$	SIM	Output	0
$\overline{BGACK}$	SIM	Input	0
$\overline{BKPT}$	CPU16	Input	0
$\overline{BR}$	SIM	Input	0
CLKOUT	SIM	Output	—
$\overline{CS[10:0]}$	SIM	Output	0
$\overline{CSBOOT}$	SIM	Output	0
DATA[15:0]	SIM	Bus	—
$\overline{DS}$	SIM	Output	0
$\overline{DSACK[1:0]}$	SIM	Input	0
DSCLK	CPU16	Input	—
DSI	CPU16	Input	—
DSO	CPU16	Output	—
EXTAL	SIM	Input	—
FC[2:0]	SIM	Output	—
FREEZE	SIM	Output	1
$\overline{HALT}$	SIM	Input/Output	0
IPIPE[1:0]	CPU16	Output	—
$\overline{IRQ[7:1]}$	SIM	Input	0
MODCLK	SIM	Input	—

Table 5 MCU Signal Characteristics (Continued)

Signal Name	MCU Module	Signal Type	Active State
QUOT	SIM	Output	—
R/$\overline{W}$	SIM	Output	1/0
$\overline{RESET}$	SIM	Input/Output	0
PE3	SIM	Output	—
SIZ[1:0]	SIM	Output	—
TSC	SIM	Input	—
XFC	SIM	Input	—
XTAL	SIM	Output	—

2.5 Signal Functions

Table 6 MCU Signal Functions

Signal Name	Mnemonic	Function
Address Bus	ADDR[23:0]	20-bit address bus used by CPU16; ADDR[23:20] follow ADDR19
Address Strobe	$\overline{AS}$	Indicates that a valid address is on the address bus
Autovector	$\overline{AVEC}$	Requests an automatic vector during interrupt acknowledge
Bus Error	$\overline{BERR}$	Signals a bus error to the CPU
Bus Grant	$\overline{BG}$	Indicates that the MCU has relinquished the bus
Bus Grant Acknowledge	$\overline{BGACK}$	Indicates that an external device has assumed bus mastership
Breakpoint	$\overline{BKPT}$	Signals a hardware breakpoint to the CPU
Bus Request	$\overline{BR}$	Indicates that an external device requires bus mastership
System Clock Out	CLKOUT	System clock output
Chip Selects	$\overline{CS}$[10:0]	Select external devices at programmed addresses
Boot Chip Select	$\overline{CSBOOT}$	Chip-select for external boot start-up ROM
Data Bus	DATA[15:0]	16-bit data bus
Data Strobe	$\overline{DS}$	Indicates that an external device should place valid data on the data bus during a read cycle and that valid data has been placed on the bus by the CPU during a write cycle
Data and Size Acknowledge	$\overline{DSACK}$[1:0]	Acknowledges to the SIM that data has been received for a write cycle, or that data is valid on the data bus for a read cycle
Development Serial In, Out, Clock	DSI, DSO,DSCLK	Serial I/O and clock for background debug mode
Crystal Oscillator	EXTAL, XTAL	Connections for clock synthesizer circuit reference; a crystal or an external oscillator can be used
Function Codes	FC[2:0]	Identify processor state and current address space
Freeze	FREEZE	Indicates that the CPU has entered background debug mode
Halt	$\overline{HALT}$	Suspend external bus activity
Instruction Pipeline	IPIPE[1:0]	Indicate instruction pipeline activity
Interrupt Request Level	$\overline{IRQ}$[7:1]	Request interrupt service from the CPU
Clock Mode Select	MODCLK	Selects system clock source
Quotient Out	QUOT	Provides the quotient bit of the polynomial divider
Reset	$\overline{RESET}$	System reset
Read/Write	R/$\overline{W}$	Indicates the direction of data transfer on the bus
Size	SIZ[1:0]	Indicates the number of bytes to be transferred during a bus cycle
Three-State Control	TSC	Places all output drivers in a high impedance state
External Filter Capacitor	XFC	Connection for external phase-locked loop filter capacitor

3 System Integration Module

The system integration module (SIM) consists of six functional blocks that control system startup, initialization, configuration, and external bus. **Figure 4** shows the SIM block diagram.

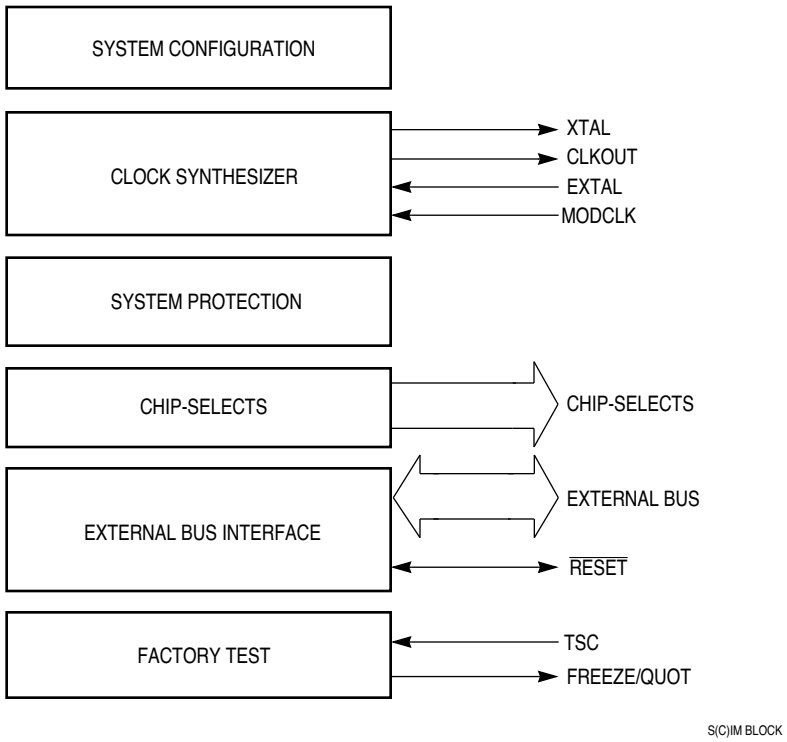


Figure 4 SIM Block Diagram

3.1 Overview

The system configuration block controls MCU configuration and operating mode.

The clock synthesizer generates clock signals used by the SIM, other IMB modules, and external devices. In addition, a periodic interrupt generator supports execution of time-critical control routines.

The system protection block provides bus and software watchdog monitors.

The chip-select block provides eleven general-purpose chip-select signals and a boot ROM chip-select signal. Both general-purpose and boot ROM chip-select signals have associated base address registers and option registers.

The external bus interface handles the transfer of information between IMB modules and external address space.

The system test block incorporates hardware necessary for testing the MCU. It is used to perform factory tests, and its use in normal applications is not supported.

Table 7 shows the SIM address map, which occupies 128 bytes. Unused registers within the 128-byte address space return zeros when read.

Table 7 SIM Address Map

Address	15 8	7 0
$YFFA00[1]	SIM Module Configuration Register (SIMCR)	
$YFFA02	SIM Test Register (SIMTR)	
$YFFA04	Clock Synthesizer Control Register (SYNCR)	
$YFFA06	Not Used	Reset Status Register (RSR)
$YFFA08	SIM Test Register E (SIMTRE)	
$YFFA0A	Not Used	
$YFFA0C	Not Used	
$YFFA0E	Not Used	
$YFFA10	Not Used	Port E Data (PORTE0)
$YFFA12	Not Used	Port E Data (PORTE1)
$YFFA14	Not Used	Port E Data Direction (DDRE)
$YFFA16	Not Used	Port E Pin Assignment (PEPAR)
$YFFA18	Not Used	Port F Data (PORTF0)
$YFFA1A	Not Used	Port F Data (PORTF1)
$YFFA1C	Not Used	Port F Data Direction (DDRF)
$YFFA1E	Not Used	Port F Pin Assignment (PFPAR)
$YFFA20	Not Used	System Protection Control (SYPCR)
$YFFA22	Periodic Interrupt Control Register (PICR)	
$YFFA24	Periodic Interrupt Timer Register (PITR)	
$YFFA26	Not Used	Software Service (SWSR)
$YFFA28	Not Used	
$YFFA2A	Not Used	
$YFFA2C	Not Used	
$YFFA2E	Not Used	
$YFFA30	Test Module Master Shift A (TSTMSRA)	
$YFFA32	Test Module Master Shift B (TSTMSRB)	
$YFFA34	Test Module Shift Count (TSTSC)	
$YFFA36	Test Module Repetition Counter (TSTRC)	
$YFFA38	Test Module Control (CREG)	
$YFFA3A	Test Module Distributed Register (DREG)	
$YFFA3C	Not Used	
$YFFA3E	Not Used	
$YFFA40	Not Used	Port C Data (PORTC)
$YFFA42	Not Used	
$YFFA44	Chip-Select Pin Assignment (CSPAR0)	
$YFFA46	Chip-Select Pin Assignment (CSPAR1)	
$YFFA48	Chip-Select Base Boot (CSBARBT)	
$YFFA4A	Chip-Select Option Boot (CSORBT)	
$YFFA4C	Chip-Select Base 0 (CSBAR0)	
$YFFA4E	Chip-Select Option 0 (CSOR0)	
$YFFA50	Chip-Select Base 1 (CSBAR1)	

Table 7 SIM Address Map (Continued)

Address	15 8	7 0
$YFFA52	Chip-Select Option 1 (CSOR1)	
$YFFA54	Chip-Select Base 2 (CSBAR2)	
$YFFA56	Chip-Select Option 2 (CSOR2)	
$YFFA58	Chip-Select Base 3 (CSBAR3)	
$YFFA5A	Chip-Select Option 3 (CSOR3)	
$YFFA5C	Chip-Select Base 4 (CSBAR4)	
$YFFA5E	Chip-Select Option 4 (CSOR4)	
$YFFA60	Chip-Select Base 5 (CSBAR5)	
$YFFA62	Chip-Select Option 5 (CSOR5)	
$YFFA64	Chip-Select Base 6 (CSBAR6)	
$YFFA66	Chip-Select Option 6 (CSOR6)	
$YFFA68	Chip-Select Base 7 (CSBAR7)	
$YFFA6A	Chip-Select Option 7 (CSOR7)	
$YFFA6C	Chip-Select Base 8 (CSBAR8)	
$YFFA6E	Chip-Select Option 8 (CSOR8)	
$YFFA70	Chip-Select Base 9 (CSBAR9)	
$YFFA72	Chip-Select Option 9 (CSOR9)	
$YFFA74	Chip-Select Base 10 (CSBAR10)	
$YFFA76	Chip-Select Option 10 (CSOR10)	
$YFFA78	Not Used	
$YFFA7A	Not Used	
$YFFA7C	Not Used	
$YFFA7E	Not Used	

NOTES:
1. Y = M111, where M is the logic state of the module mapping (MM) bit in the SIMCR.

3.2 System Configuration Block

The SIM controls MCU configuration during normal operation and during internal testing.

SIMCR — SIM Configuration Register **$YFFA00**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EXOFF	FRZSW	FRZBM	0	SLVEN	0	SHEN		SUPV	MM	0	0	IARB[3:0]			
RESET:															
0	0	0	0	DATA11	0	0	0	1	1	0	0	1	1	1	1

The SIM configuration register controls system configuration. It can be read or written at any time, except for the module mapping (MM) bit, which can be written only once.

EXOFF — External Clock Off

0 = The CLKOUT pin is driven by the MCU system clock.

1 = The CLKOUT pin is placed in a high-impedance state.

FRZSW — Freeze Software Enable

0 = When FREEZE is asserted, the software watchdog and periodic interrupt timer counters continue to run.

1 = When FREEZE is asserted, the software watchdog and periodic interrupt timer counters are disabled, preventing interrupts while the MCU is in background debug mode.

FRZBM — Freeze Bus Monitor Enable

0 = When FREEZE is asserted, the bus monitor continues to operate.

1 = When FREEZE is asserted, the bus monitor is disabled.

SLVEN — Factory Test Mode Enabled

This bit is a read-only status bit that reflects the state of DATA11 during reset.

0 = IMB is not available to an external master.

1 = An external bus master has direct access to the IMB.

SHEN[1:0] — Show Cycle Enable

This field determines what the EBI does with the external bus during internal transfer operations. A show cycle allows internal transfers to be externally monitored. **Table 8** shows whether show cycle data is driven externally, and whether external bus arbitration can occur. To prevent bus conflict, external peripherals must not be enabled during show cycles.

Table 8 Show Cycle Enable Bits

SHEN[1:0]	Action
00	Show cycles disabled, external bus arbitration allowed
01	Show cycles enabled, external bus arbitration not allowed
10	Show cycles enabled, external bus arbitration allowed
11	Show cycles enabled, external bus arbitration allowed, internal activity is halted by a bus grant

SUPV — Supervisor/Unrestricted Data Space

This bit has no effect because the CPU16 always operates in the supervisor mode.

MM — Module Mapping

0 = Internal modules are addressed from $7FF000 – $7FFFFF.

1 = Internal modules are addressed from $FFF000 – $FFFFFF.

The logic state of MM determines the value of ADDR23 for IMB module addresses. Because ADDR[23:20] are driven to the same state as ADDR19, MM must be set to one. If MM is cleared, IMB modules are inaccessible. This bit can be written only once after reset.

IARB[3:0] — Interrupt Arbitration Field

Each module that can generate interrupt requests has an interrupt arbitration (IARB) field. Arbitration between interrupt requests of the same priority is performed by serial contention between IARB field bit values. Contention must take place whenever an interrupt request is acknowledged, even when there is only a single pending request. An IARB field must have a non-zero value for contention to take place. If an interrupt request from a module with an IARB field value of %0000 is recognized, the CPU processes a spurious interrupt exception. Because the SIM routes external interrupt requests to the CPU, the SIM IARB field value is used for arbitration between internal and external interrupts of the same priority. The reset value of IARB for the SIM is %1111, and the reset value of IARB for all other modules is %0000, which prevents SIM interrupts from being discarded during initialization.

3.3 System Clock

The system clock in the SIM provides timing signals for the IMB modules and for an external peripheral bus. Because the MCU is a fully static design, register and memory contents are not affected when the clock rate changes. System hardware and software support changes in clock rate during operation.

The system clock signal can be generated in one of two ways. An internal phase-locked loop can synthesize the clock from a reference frequency, or the clock signal can be input directly from an external source. Keep these clock sources in mind while reading the rest of this section. **Figure 5** is a block diagram of the system clock.

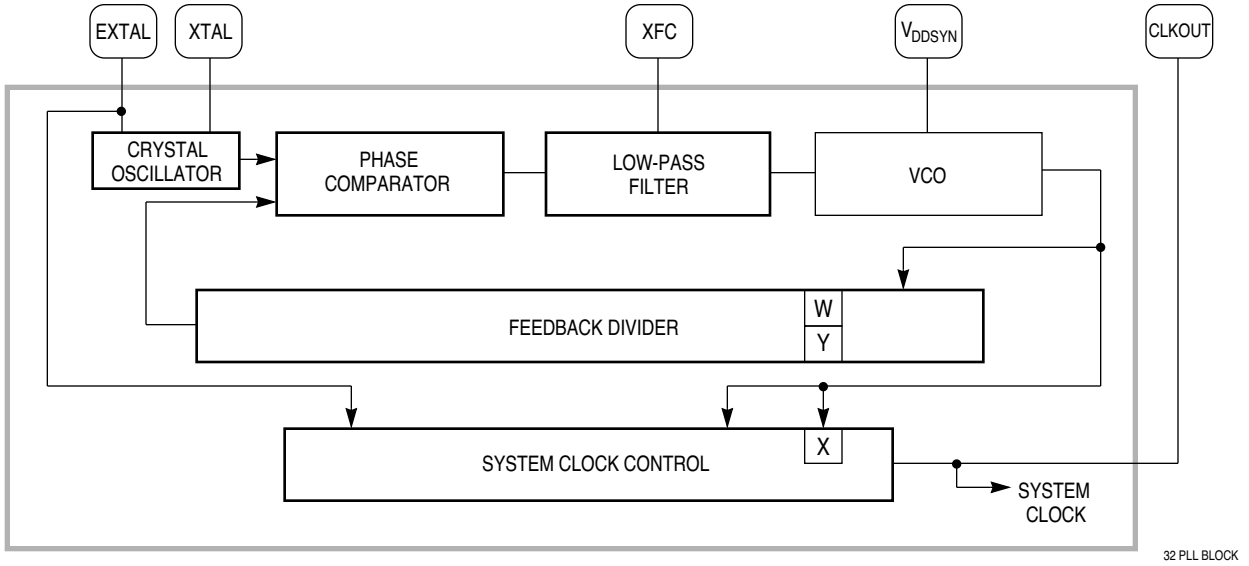


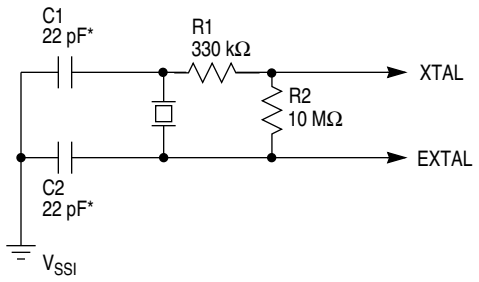
Figure 5 System Clock Block Diagram

3.3.1 Clock Sources

The state of the MODCLK pin during reset determines the system clock source. When MODCLK is held high during reset, the clock synthesizer generates a clock signal from a reference frequency connected to the EXTAL pin. The clock synthesizer control register (SYNCR) determines operating frequency and mode of operation. When MODCLK is held low during reset, the clock synthesizer is disabled and an external system clock signal must be applied. The SYNCR control bits have no effect.

The input clock is referred to as "f_{ref}", and can be either a crystal or an external clock source. The output of the clock system is referred to as "f_{sys}". Ensure that f_{ref} and f_{sys} are within normal operating limits.

The reference frequency for this MCU is typically 32.768 kHz, but can range from 25 kHz to 50 kHz. To generate a reference frequency using the crystal oscillator, a reference crystal must be connected between the EXTAL and XTAL pins. **Figure 6** shows a recommended circuit.



* RESISTANCE AND CAPACITANCE BASED ON A TEST CIRCUIT CONSTRUCTED WITH A DAISHINKU DMX-38 32.768 kHz CRYSTAL. SPECIFIC COMPONENTS MUST BE BASED ON CRYSTAL TYPE. CONTACT CRYSTAL VENDOR FOR EXACT CIRCUIT.

32 OSCILLATOR

Figure 6 System Clock Oscillator Circuit

When an external system clock signal is applied (PLL disabled, MODCLK = 0 during reset), the duty cycle of the input is critical, especially at operating frequencies close to maximum. The relationship between clock signal duty cycle and clock signal period is expressed:

$$\text{Minimum External Clock Period} = \frac{\text{Minimum External Clock High/Low Time}}{50\ \% - \text{Percentage Variation of External Clock Input Duty Cycle}}$$

When the system clock signal is applied directly to the EXTAL pin (PLL is disabled, MODCLK = 0 during reset), or the clock synthesizer reference frequency is supplied by a source other than a crystal (PLL enabled, MODCLK = 1 during reset), the XTAL pin must be left floating. In either case, the frequency of the signal applied to EXTAL may not exceed the maximum system clock frequency (PLL disabled) or the maximum clock synthesizer reference frequency (PLL enabled).

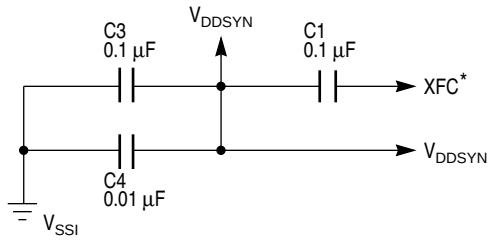
3.3.2 Clock Synthesizer Operation

V_{DDSYN} is used to power the clock circuits when the phase-locked loop is used. A separate power source increases MCU noise immunity and can be used to run the clock when the MCU is powered down. A quiet power supply must be used as the V_{DDSYN} source. Adequate external bypass capacitors should be placed as close as possible to the V_{DDSYN} pin to assure stable operating frequency. When an external system clock signal is applied and the PLL is disabled, V_{DDSYN} should be connected to the V_{DD} supply. Refer to the *SIM Reference Manual* (SIMRM/AD) for more information regarding system clock power supply conditioning.

A voltage controlled oscillator (VCO) generates the system clock signal. To maintain a 50% clock duty cycle, the VCO frequency (f_{VCO}) is either two or four times the system clock frequency, depending on the state of the X bit in SYNCR. A portion of the clock signal is fed back to a divider/counter. The divider controls the frequency of one input to a phase comparator. The other phase comparator input is the reference signal connected to the EXTAL pin. The comparator generates a control signal proportional to the difference in phase between the two inputs. The signal is low-pass filtered and used to correct the VCO output frequency.

Filter circuit implementation can vary, depending upon the external environment and required clock stability. **Figure 7** shows a recommended system clock filter network. XFC pin leakage must be kept within specified limits to maintain optimum stability and PLL performance.

An external filter network connected to the XFC pin is not required when an external system clock signal is applied and the PLL is disabled. The XFC pin must be left floating in this case.



* MAINTAIN LOW LEAKAGE ON THE XFC NODE.

32 XFC CONN

Figure 7 System Clock Filter Network

When the clock synthesizer is used, SYNCR determines the operating frequency of the MCU. The following equation relates the MCU operating frequency to the clock synthesizer reference frequency (f_{ref}) and the W, X, and Y fields in SYNCR:

$$f_{sys} = 4f_{ref}(Y + 1)(2^{2W + X})$$

The W bit controls a prescaler tap in the feedback divider. Setting W increases VCO speed by a factor of four. The Y field determines the count modulus for a modulo 64 downcounter, causing it to divide by a value of Y+1. When W or Y changes, VCO frequency (f_{VCO}) changes, and the VCO must relock.

The X bit controls a divide-by-two circuit that is not in the synthesizer feedback loop. When X=0 (reset state), the divider is enabled, and the system clock is one-fourth the VCO frequency. Setting X=1 disables the divider, doubling the clock speed without changing the VCO frequency. There is no relock delay when clock speed is changed by the X bit.

Internal VCO frequency is determined by the following equations:

$$f_{VCO} = 4f_{sys} \text{ if } X = 0$$

or

$$f_{VCO} = 2f_{sys} \text{ if } X = 1$$

For the MCU to operate correctly, system clock and VCO frequencies selected by the W, X, and Y bits must be within the limits specified for the MCU. Do not use a combination of bit values that selects either an operating frequency or a VCO frequency greater than the maximum specified values.

3.3.3 Clock Synthesizer Control

The clock synthesizer control circuits determine system clock frequency and clock operation under special circumstances, such as following loss of synthesizer reference or during low-power operation. Clock source is determined by the logic state of the MODCLK pin during reset.

SYNCR — Clock Synthesizer Control Register **$YFFA04**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
W	X	Y						EDIV	0	0	RSVD[1]	SLOCK	RSVD[1]	STSIM	STEXT
RESET:															
0	0	1	1	1	1	1	1	0	0	0	0	U	0	0	0

NOTES:
1. Ensure that initialization software does not change the value of this bit (it should always be zero).

When the on-chip clock synthesizer is used, system clock frequency is controlled by the bits in the upper byte of SYNCR. Bits in the lower byte show the status of or control the operation of internal and external clocks. SYNCR can be read or written only when the CPU is operating in supervisor mode.

W — Frequency Control (VCO)

This bit controls a prescaler tap in the synthesizer feedback loop. Setting it increases the VCO speed by a factor of four. VCO relock delay is required.

X — Frequency Control (Prescaler)

This bit controls a divide by two prescaler that is not in the synthesizer feedback loop. Setting it doubles the clock speed without changing the VCO speed. No VCO relock delay is required.

Y[5:0] — Frequency Control (Counter)

The Y field controls the modulus down counter in the synthesizer feedback loop, causing it to divide by a value of Y + 1. Values range from zero to 63. VCO relock delay is required.

EDIV — E Clock Divide Rate

0 = ECLK frequency is system clock divided by eight.
1 = ECLK frequency is system clock divided by 16.

ECLK is an external M6800 bus clock available on pin ADDR23. Refer to **3.6 Chip-Selects** for more information.

SLOCK — Synthesizer Lock Flag

0 = VCO has not locked, but is enabled on the desired frequency.
1 = VCO has locked on the desired frequency, or is disabled.

The MCU remains in reset until the synthesizer locks, but SLOCK does not indicate synthesizer lock status until after the user writes to SYNCR.

STSIM — Stop Mode SIM Clock

0 = When LPSTOP is executed, the SIM clock is driven by the crystal oscillator and the VCO is turned off to conserve power.
1 = When LPSTOP is executed, the SIM clock is driven by the VCO.

STEXT — Stop Mode External Clock

0 = When LPSTOP is executed, the CLKOUT signal is held negated to conserve power.
1 = When LPSTOP is executed, the CLKOUT signal is driven by the SIM clock, as determined by the state of the STSIM bit.

3.3.4 External MC6800 Bus Clock

The state of the ECLK division rate bit (EDIV) in SYNCR determines clock rate for the ECLK signal available on pin ADDR23. ECLK is a bus clock for MC6800 devices and peripherals. ECLK frequency can be set to system clock frequency divided by eight or system clock frequency divided by sixteen. The clock is enabled by the $\overline{\text{CS10}}$ field in chip-select pin assignment register 1 (CSPAR1). ECLK operation during low-power stop is described in the following paragraph. Refer to **3.6 Chip-Selects** for more information about the external bus clock.

3.3.5 Low-Power Operation

Low-power operation is initiated by the CPU16. To reduce power consumption selectively, the CPU16 can enter the following low-power modes:

1. The CPU16 can selectively disable a module by setting the module's STOP bit.
2. The CPU16 can execute the LPSTOP instruction to stop the operations of the entire MCU.

If the STOP bit in a module is set, then that module enters a low power mode. Some or all of that module's registers remain accessible. The module can be restarted by asserting $\overline{\text{RESET}}$ or by the CPU16 clearing the module's STOP bit.

3.3.5.1 LPSTOP Mode

This low power mode offers the greatest power reduction. To enter normal LPSTOP mode, the CPU16 executes the LPSTOP instruction after clearing the STCPU bit in SYNCR. This causes the SIM to turn off the system clock to most of the MCU.

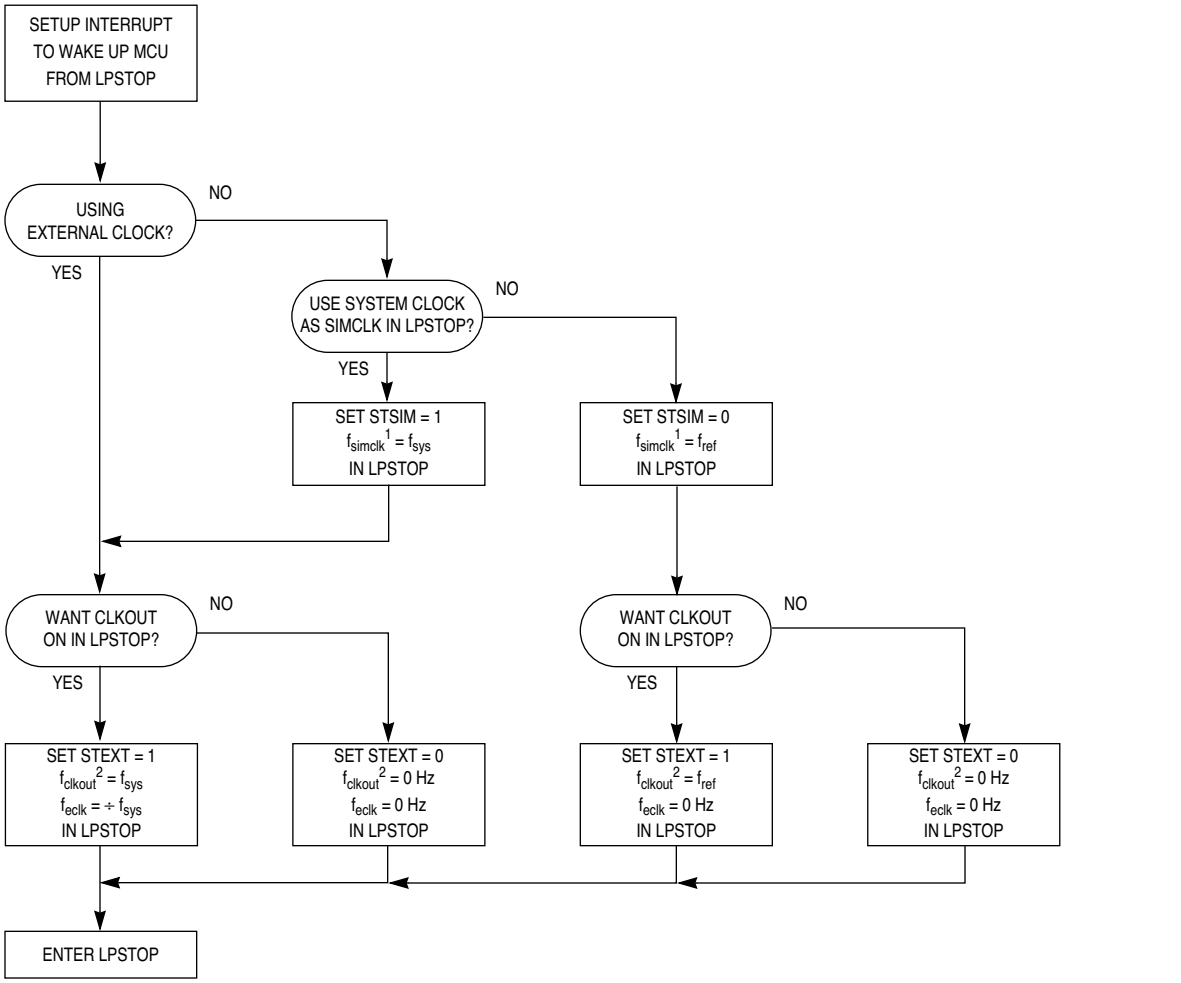
When the CPU executes LPSTOP, a special CPU space bus cycle writes a copy of the current interrupt mask into the clock control logic. The SIM brings the MCU out of normal LPSTOP mode when one of the following exceptions occurs:

- $\overline{\text{RESET}}$
- Trace
- SIM interrupt of higher priority than the stored interrupt mask

During LPSTOP, unless the system clock signal is supplied by an external source and that source is removed, the SIM clock control logic and the SIM clock signal (SIMCLK) continue to operate. The periodic interrupt timer and input logic for the $\overline{\text{RESET}}$ and $\overline{\text{IRQ}}$ pins are clocked by SIMCLK, and can be used to bring the processor out of LPSTOP. The software watchdog monitor cannot perform this function. Optionally, the SIM can also continue to generate the CLKOUT signal while in LPSTOP.

STSIM and STEXT bits in SYNCR determine clock operation during LPSTOP.

The flow chart shown in **Figure 8** summarizes the effects of the STSIM and STEXT bits when the MCU enters normal LPSTOP mode.



NOTES:
1. THE SIMCLK IS USED BY THE PIT, $\overline{IRQ}$, AND INPUT BLOCKS OF THE SIM.
2. CLKOUT CONTROL DURING LPSTOP IS OVERRIDDEN BY THE EXOFF BIT IN SIMCR. IF EXOFF = 1, THE CLKOUT PIN IS ALWAYS IN A HIGH IMPEDANCE STATE AND STEXT HAS NO EFFECT IN LPSTOP. IF EXOFF = 0, CLKOUT IS CONTROLLED BY STEXT IN LPSTOP.

LPSTOPFLOW

Figure 8 LPSTOP Flowchart

3.4 System Protection Block

System protection includes a bus monitor, a halt monitor, a spurious interrupt monitor, and a software watchdog timer. These functions reduce the number of external components required for complete system control. **Figure 9** shows the system protection block.

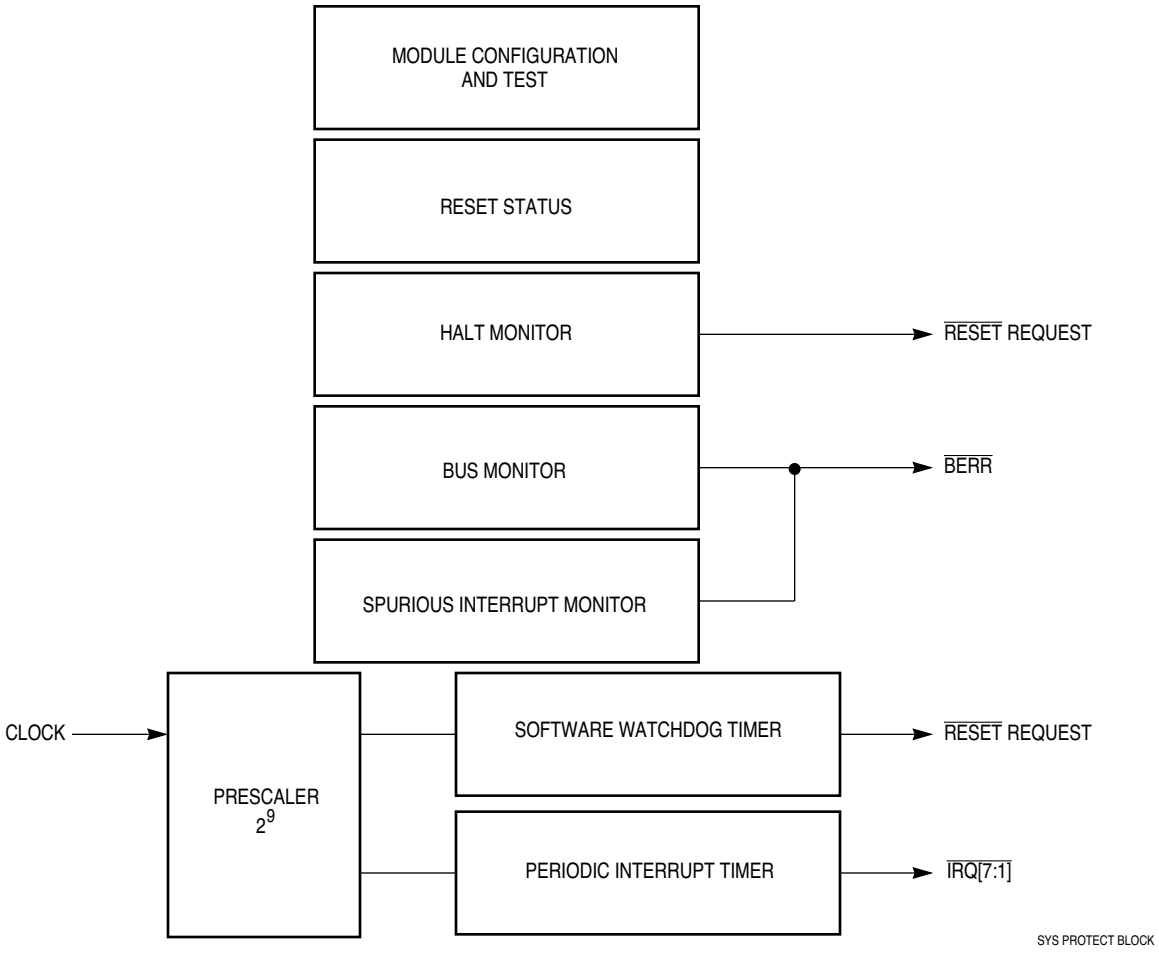


Figure 9 System Protection Block

3.4.1 System Protection Control Register

The system protection control register controls the software watchdog timer, bus monitor, and halt monitor. This register can be written only once following power-on or reset, but can be read at any time.

SYPCR — System Protection Control Register **$YFFA21**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NOT USED								SWE	SWP	SWT[1:0]		HME	BME	BMT	
RESET:															
								1	MODCLK	0	0	0	0	0	0

SWE — Software Watchdog Enable
- 0 = Software watchdog disabled
- 1 = Software watchdog enabled

SWP — Software Watchdog Prescaler

This bit controls the value of the software watchdog prescaler.
- 0 = Software watchdog clock not prescaled
- 1 = Software watchdog clock prescaled by 512

SWT[1:0] — Software Watchdog Timing

This field selects the divide ratio used to establish software watchdog time-out period. **Table 9** gives the ratio for each combination of SWP and SWT bits.

Table 9 Software Watchdog Timing Field

SWP	SWT[1:0]	Ratio
0	00	2^9
0	01	2^{11}
0	10	2^{13}
0	11	2^{15}
1	00	2^{18}
1	01	2^{20}
1	10	2^{22}
1	11	2^{24}

HME — Halt Monitor Enable

0 = Disable halt monitor function
1 = Enable halt monitor function

BME — Bus Monitor Enable

0 = Disable bus monitor function for internal to external bus cycles.
1 = Enable bus monitor function for internal to external bus cycles.

BMT[1:0] — Bus Monitor Timing

This bit field selects the time-out period in system clocks for the bus monitor. Refer to **Table 10**.

Table 10 Bus Monitor Time-Out Period

BMT[1:0]	Bus Monitor Time-Out Period
00	64 System clocks
01	32 System clocks
10	16 System clocks
11	8 System clocks

3.4.2 Bus Monitor

The internal bus monitor checks for excessively long $\overline{\text{DSACK}}$ response times during normal bus cycles and for excessively long $\overline{\text{DSACK}}$ or $\overline{\text{AVEC}}$ response times during interrupt acknowledge (IACK) cycles. The monitor asserts $\overline{\text{BERR}}$ if the response time exceeds a user-specified timeout period.

$\overline{\text{DSACK}}$ and $\overline{\text{AVEC}}$ response times are measured in clock cycles. The maximum allowable response time can be selected by setting the BMT[1:0] field.

The monitor does not check $\overline{\text{DSACK}}$ response on the external bus unless the CPU initiates the bus cycle. The BME bit in SYPCR enables the internal bus monitor for internal to external bus cycles. If a system contains external bus masters, an external bus monitor must be implemented and the internal to external bus monitor option must be disabled.

3.4.3 Halt Monitor

The halt monitor responds to assertion of the $\overline{\text{HALT}}$ signal on the internal bus caused by a double bus fault. A double bus fault occurs when:

- Bus error exception processing begins and a second $\overline{\text{BERR}}$ is detected before the first instruction of the first exception handler is executed.
- One or more bus errors occur before the first instruction after a reset exception is executed.
- A bus error occurs while the CPU is loading information from a bus error stack frame during a return from exception (RTE) instruction.

If the halt monitor is enabled by setting HME in SYPCR, the MCU will issue a reset when a double bus fault occurs, otherwise the MCU will remain halted.

A flag in the reset status register (RSR) indicates that the last reset was caused by the halt monitor.

3.4.4 Spurious Interrupt Monitor

The spurious interrupt monitor issues $\overline{\text{BERR}}$ if no interrupt arbitration occurs during an interrupt acknowledge cycle. Leaving IARB[3:0] set to %0000 in the module configuration register of any peripheral that can generate interrupts will cause a spurious interrupt.

3.4.5 Software Watchdog

The software watchdog is controlled by SWE in SYPCR. Once enabled, the watchdog requires that a service sequence be written to SWSR on a periodic basis. If servicing does not take place, the watchdog times out and issues a reset. This register can be written at any time, but returns zeros when read.

SWSR — Software Service Register **$YFFA27**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NOT USED								SWSR							
RESET:															
								0	0	0	0	0	0	0	0

Each time the service sequence is written, the software watchdog timer restarts. The servicing sequence consists of the following steps:

1. Write $55 to SWSR.
2. Write $AA to SWSR.

Both writes must occur before time-out in the order listed, but any number of instructions can be executed between the two writes.

The watchdog clock rate is affected by SWP and SWT[1:0] in SYPCR. When SWT[1:0] are modified, a watchdog service sequence must be performed before the new time-out period takes effect.

The reset value of SWP is affected by the state of the MODCLK pin on the rising edge of $\overline{\text{RESET}}$, as shown in **Table 11**.

Table 11 MODCLK Pin States

MODCLK	SWP
0	1
1	0

3.4.6 Periodic Interrupt Timer

The periodic interrupt timer (PIT) generates interrupts at user-programmable intervals. Timing for the PIT is provided by a programmable prescaler driven by the system clock.

PICR — Periodic Interrupt Control Register **$YFFA22**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	PIRQL[2:0]			PIV[7:0]							
RESET:															
0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1

This register contains information concerning periodic interrupt priority and vectoring. Bits [10:0] can be read or written at any time. Bits [15:11] are unimplemented and always return zero.

PIRQL[2:0] — Periodic Interrupt Request Level

Table 12 shows what interrupt request level is asserted when a periodic interrupt is generated. If a PIT interrupt and an external $\overline{IRQ}$ signal of the same priority occur simultaneously, the PIT interrupt is serviced first. The periodic timer continues to run when the interrupt is disabled.

Table 12 Periodic Interrupt Request Levels

PIRQL[2:0]	Interrupt Request Level
000	Periodic interrupt disabled
001	Interrupt request level 1
010	Interrupt request level 2
011	Interrupt request level 3
100	Interrupt request level 4
101	Interrupt request level 5
110	Interrupt request level 6
111	Interrupt request level 7

PIV[7:0] — Periodic Interrupt Vector

This bit field contains the vector generated in response to an interrupt from the periodic timer. When the SIM responds, the periodic interrupt vector is placed on the bus.

PITR — Periodic Interrupt Timer Register **$YFFA24**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	PTP	PITM[7:0]							
RESET:															
0	0	0	0	0	0	0	$\overline{MODCLK}$	0	0	0	0	0	0	0	0

PITR contains the count value for the periodic timer. Setting the PITM[7:0] field turns off the periodic timer. This register can be read or written at any time.

PTP — Periodic Timer Prescaler Control

0 = Periodic timer clock not prescaled

1 = Periodic timer clock prescaled by 512

The reset state of PTP is the complement of the state of the MODCLK signal at the rising edge of $\overline{RESET}$.

PITM[7:0] — Periodic Interrupt Timer Modulus

This is an 8-bit timing modulus. The period of the timer can be calculated as follows:

$$\text{PIT Period} = \frac{4(\text{PITM[7:0]})(\text{Prescaler})}{f_{ref}}$$

where

PIT Period = Periodic interrupt timer period

PITM[7:0] = Periodic interrupt timer modulus

f_{ref} = Synthesizer reference of external clock input frequency

Prescaler = 1 if PTP = 0 or 512 if PTP = 1

3.5 External Bus Interface

The external bus interface (EBI) transfers information between the internal MCU bus and external devices. The external bus has 24 address lines and 16 data lines. Because the CPU16 in the MC68HC16S2 drives only 20 of the 24 IMB address lines, ADDR[23:20] follow the output state of ADDR19.

The EBI provides dynamic sizing between 8-bit and 16-bit data accesses. It supports byte, word, and long-word transfers. Ports are accessed through the use of asynchronous cycles controlled by the size (SIZ1 and SIZ0) and data size acknowledge ($\overline{\text{DSACK1}}$ and $\overline{\text{DSACK0}}$) pins. Multiple bus cycles may be required for dynamically sized transfer.

Port width is the maximum number of bits accepted or provided during a bus transfer. External devices must follow the handshake protocol described below. Control signals indicate the beginning of the cycle, the address space, the size of the transfer, and the type of cycle. The selected device controls the length of the cycle. Strobe signals, one for the address bus and another for the data bus, indicate the validity of an address and provide timing information for data. The EBI operates in an asynchronous mode for any port width.

To add flexibility and minimize the necessity for external logic, MCU chip-select logic can be synchronized with EBI transfers. Chip-select logic can also provide internally-generated bus control signals for these accesses. Refer to **3.6 Chip-Selects** for more information.

3.5.1 Bus Control Signals

The CPU initiates a bus cycle by driving the address, size, function code, and read/write outputs. At the beginning of the cycle, size signals SIZ0 and SIZ1 are driven along with the function code signals FC[2:0]. The size signals indicate the number of bytes remaining to be transferred during an operand cycle. They are valid while the address strobe $\overline{\text{AS}}$ is asserted.

Table 13 shows SIZ0 and SIZ1 encoding. The read/write (R/$\overline{\text{W}}$) signal determines the direction of the transfer during a bus cycle. This signal changes state, when required, at the beginning of a bus cycle, and is valid while $\overline{\text{AS}}$ is asserted. The R/$\overline{\text{W}}$ signal only changes state when a write cycle is preceded by a read cycle or vice versa. The signal can remain low for two consecutive write cycles.

Table 13 Size Signal Encoding

SIZ1	SIZ0	Transfer Size
0	1	Byte
1	0	Word
1	1	Three byte
0	0	Long word

3.5.2 Function Codes

Function code signals FC[2:0] are automatically generated by the CPU16. The function codes can be considered address extensions that automatically select one of eight address spaces to which an address applies. These spaces are designated as either user or supervisor, and program or data spaces. Because the CPU16 always operates in supervisor mode (FC2 always = 1), address spaces 0 to 3 are not used. Address space 7 is designated CPU space. CPU space is used for control information not normally associated with read or write bus cycles. Function codes are valid while $\overline{AS}$ is asserted.

Table 14 displays CPU16 address space encodings.

Table 14 CPU16 Address Space Encoding

FC2	FC1	FC0	Address Space
1	0	0	Reserved
1	0	1	Data space
1	1	0	Program space
1	1	1	CPU space

3.5.3 Address Bus

Address bus signals ADDR[19:0] define the address of the most significant byte to be transferred during a bus cycle. The MCU places the address on the bus at the beginning of a bus cycle. The address is valid while $\overline{AS}$ is asserted. Because the CPU16 in the MC68HC16S2 does not drive ADDR[23:20], these lines follow the logic state of ADDR19.

3.5.4 Address Strobe

$\overline{AS}$ is a timing signal that indicates the validity of an address on the address bus and the validity of many control signals. It is asserted one-half clock after the beginning of a bus cycle.

3.5.5 Data Bus

Data bus signals DATA[15:0] make up a bidirectional, non-multiplexed parallel bus that transfers data to or from the MCU. A read or write operation can transfer eight or 16 bits of data in one bus cycle. During a read cycle, the data is latched by the MCU on the last falling edge of the clock for that bus cycle. For a write cycle, all 16 bits of the data bus are driven, regardless of the port width or operand size. The MCU places the data on the data bus one-half clock cycle after $\overline{AS}$ is asserted in a write cycle.

3.5.6 Data Strobe

Data strobe ($\overline{DS}$) is a timing signal. For a read cycle, the MCU asserts $\overline{DS}$ to signal an external device to place data on the bus. $\overline{DS}$ is asserted at the same time as $\overline{AS}$ during a read cycle. For a write cycle, $\overline{DS}$ signals an external device that data on the bus is valid. The MCU asserts $\overline{DS}$ one full clock cycle after the assertion of $\overline{AS}$ during a write cycle.

3.5.7 Bus Cycle Termination Signals

During bus cycles, external devices assert the data size acknowledge signals $\overline{DSACK1}$ and $\overline{DSACK0}$. During a read cycle, the signals tell the MCU to terminate the bus cycle and to latch data. During a write cycle, the signals indicate that an external device has successfully stored data and that the cycle can end. These signals also indicate to the MCU the size of the port for the bus cycle just completed. Alternately, chip-selects can be used to generate $\overline{DSACK1}$ and $\overline{DSACK0}$ internally. Refer to **3.5.8 Dynamic Bus Sizing** for more information.

The bus error ($\overline{BERR}$) signal is also a bus cycle termination indicator and can be used in the absence of $\overline{DSACK1}$ and $\overline{DSACK0}$ to indicate a bus error condition. It can also be asserted in conjunction with these signals, provided it meets the appropriate timing requirements. The internal bus monitor can be used to generate the $\overline{BERR}$ signal for internal-to-external transfers. When $\overline{BERR}$ and $\overline{HALT}$ are asserted simultaneously, the CPU takes a bus error exception.

The autovector signal ($\overline{AVEC}$) can terminate $\overline{IRQ}$ pin interrupt acknowledge cycles. $\overline{AVEC}$ indicates that the MCU will internally generate a vector number to locate an interrupt handler routine. If it is continuously asserted, autovectors will be generated for all external interrupt requests. $\overline{AVEC}$ is ignored during all other bus cycles.

3.5.8 Dynamic Bus Sizing

The MCU dynamically interprets the port size of the addressed device during each bus cycle, allowing operand transfers to or from 8- and 16-bit ports. During an operand transfer cycle, the slave device signals its port size and indicates completion of the bus cycle to the MCU through the use of the $\overline{DSACK1}$ and $\overline{DSACK0}$ inputs, as shown in **Table 15**.

Table 15 Effect of $\overline{DSACK}$ Signals

$\overline{DSACK1}$	$\overline{DSACK0}$	Result
1	1	Insert wait states in current bus cycle
1	0	Complete cycle — Data bus port size is 8 bits
0	1	Complete cycle — Data bus port size is 16 bits
0	0	Reserved

For example, if the MCU is executing an instruction that reads a long-word operand from a 16-bit port, the MCU latches the 16 bits of valid data and then runs another bus cycle to obtain the other 16 bits. The operation for an 8-bit port is similar, but requires four read cycles. The addressed device uses the $\overline{DSACK0}$ and $\overline{DSACK1}$ signals to indicate the port width. For instance, a 16-bit device always returns $\overline{DSACK0}$ = 1 and $\overline{DSACK1}$ = 0 for a 16-bit port, regardless of whether the bus cycle is a byte or word operation.

Dynamic bus sizing requires that the portion of the data bus used for a transfer to or from a particular port size be fixed. A 16-bit port must reside on data bus bits [15:0] and an 8-bit port must reside on data bus bits [15:8]. This minimizes the number of bus cycles needed to transfer data and ensures that the MCU transfers valid data.

The MCU always attempts to transfer the maximum amount of data on all bus cycles. For a word operation, it is assumed that the port is 16 bits wide when the bus cycle begins. Operand bytes are designated as shown in **Figure 10**. OP0 is the most significant byte of a long-word operand, and OP3 is the least significant byte. The two bytes of a word-length operand are OP0 (most significant) and OP1. The single byte of a byte-length operand is OP0.

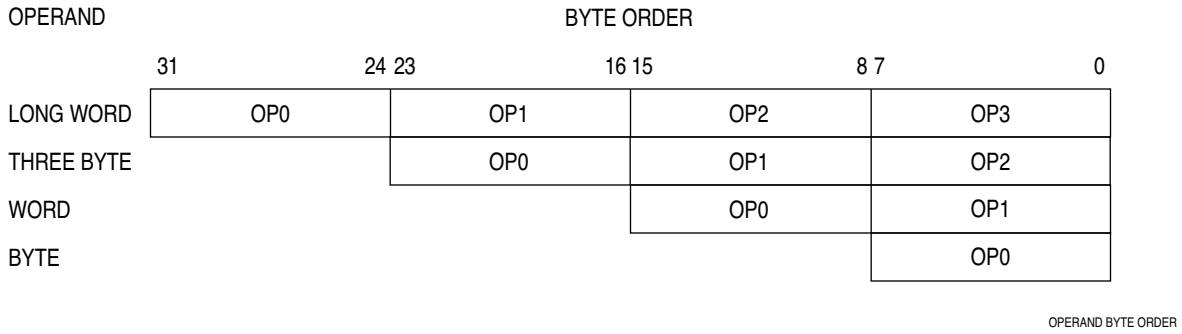


Figure 10 Operand Byte Order

3.5.9 Operand Alignment

The data multiplexer establishes the necessary connections for different combinations of address and data sizes. The multiplexer takes the two bytes of the 16-bit bus and routes them to their required positions. Positioning of bytes is determined by the size and address outputs. SIZ1 and SIZ0 indicate the remaining number of bytes to be transferred during the current bus cycle. The number of bytes transferred is equal to or less than the size indicated by SIZ1 and SIZ0, depending on port width.

ADDR0 also affects the operation of the data multiplexer. During an operand transfer, ADDR[23:1] indicate the word base address of the portion of the operand to be accessed, and ADDR0 indicates the byte offset from the base. Bear in mind the fact that ADDR[23:20] follow the state of ADDR19 in the MC68HC16S2.

3.5.10 Misaligned Operands

CPU16 processor architecture uses a basic operand size of 16 bits. An operand is misaligned when it overlaps a word boundary. This is determined by the value of ADDR0. When ADDR0 = 0 (an even address), the address is on a word and byte boundary. When ADDR0 = 1 (an odd address), the address is on a byte boundary only. A byte operand is aligned at any address; a word or long-word operand is misaligned at an odd address.

The largest amount of data that can be transferred by a single bus cycle is an aligned word. If the MCU transfers a long-word operand via a 16-bit port, the most significant operand word is transferred on the first bus cycle and the least significant operand word on a following bus cycle.

The CPU16 can perform misaligned word transfers. This capability makes it software compatible with the M68HC11 CPU. The CPU16 treats misaligned long-word transfers as two misaligned word transfers.

3.5.11 Operand Transfer Cases

Table 16 summarizes how operands are aligned for various types of transfers. OPn entries are portions of a requested operand that are read or written during a bus cycle and are defined by SIZ1, SIZ0, and ADDR0 for that bus cycle.

Table 16 Operand Alignment

Current Cycle	Transfer Case	SIZ1	SIZ0	ADDR0	$\overline{\text{DSACK1}}$	$\overline{\text{DSACK0}}$	DATA [15:8]	DATA [7:0]	Next Cycle
1	Byte to 8-bit port (even)	0	1	0	1	0	OP0	(OP0)[1]	—
2	Byte to 8-bit port (odd)	0	1	1	1	0	OP0	(OP0)	—
3	Byte to 16-bit port (even)	0	1	0	0	1	OP0	(OP0)	—
4	Byte to 16-bit port (odd)	0	1	1	0	1	(OP0)	OP0	—
5	Word to 8-bit port (aligned)	1	0	0	1	0	OP0	(OP1)	2
6	Word to 8-bit port (misaligned	1	0	1	1	0	OP0	(OP0)	1
7	Word to 16-bit port (aligned)	1	0	0	0	1	OP0	OP1	—
8	Word to 16-bit port (misaligned)	1	0	1	0	1	(OP0)	OP0	3
9	Long word to 8-bit port (aligned)	0	0	0	1	0	OP0	(OP1)	13
10	Long word to 8-bit port (misaligned)[2]	1	0	1	1	0	OP0	(OP0)	1
11	Long word to 16-bit port (aligned)	0	0	0	0	1	OP0	OP1	7
12	Long word to 16-bit port (misaligned)[2]	1	0	1	0	1	(OP0)	OP0	3
13	Three byte to 8-bit port[3]	1	1	1	1	0	OP0	(OP0)	5

NOTES:

1. Operands in parentheses are ignored by the CPU16 during read cycles.
2. The CPU16 treats misaligned long-word transfers as two misaligned word transfers.
3. Three byte transfer cases occur only as a result of an aligned long word to 8-bit port transfer.

3.6 Chip-Selects

Typical microcontrollers require additional hardware to provide external chip-select and address decode signals. The MC68HC16S2 includes 12 programmable chip-selects that can provide 2- to 16-clock-cycle access to external memory and peripherals. Address block sizes of two Kbytes to one Mbyte can be selected. However, because ADDR[23:20] = ADDR19 in the CPU16, 512 Kbyte blocks are the largest usable size. **Figure 11** is a functional diagram of a chip-select circuit.

Chip-select assertion can be synchronized with bus control signals to provide output enable, read/write strobe, or interrupt acknowledge signals. Chip-select logic can also generate $\overline{\text{DSACK}}$ and $\overline{\text{AVEC}}$ signals internally. Each signal can also be synchronized with the ECLK signal available on ADDR23.

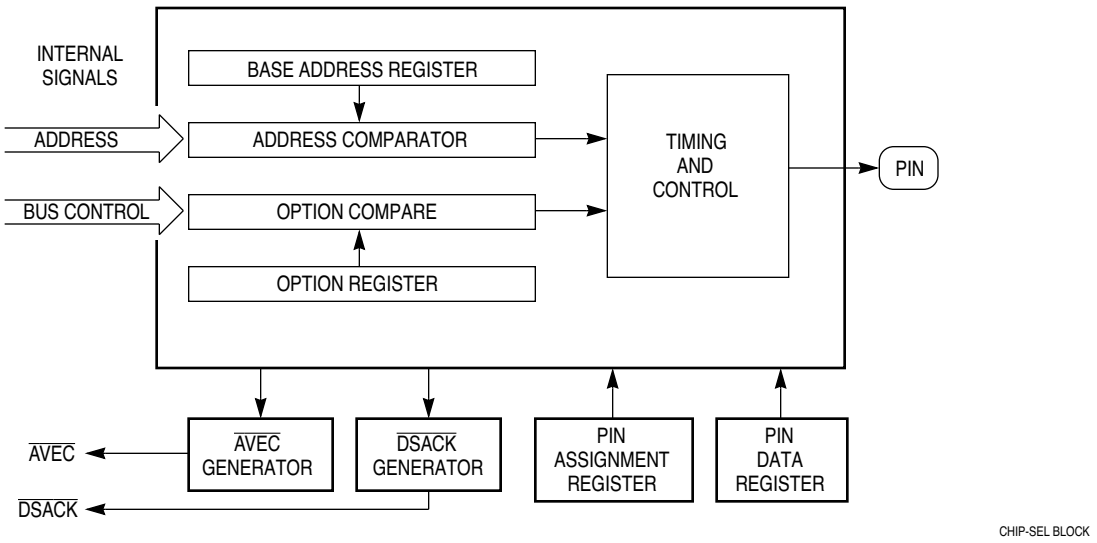


Figure 11 Chip-Select Circuit Block Diagram

When a memory access occurs, chip-select logic compares address space type, address, type of access, transfer size, and interrupt priority (in the case of interrupt acknowledge) to parameters stored in chip-select registers. If all parameters match, the appropriate chip-select signal is asserted. Select signals are active low. If a chip-select function is given the same address as a microcontroller module or an internal memory array, an access to that address goes to the module or array, and the chip-select signal is not asserted. The external address and data buses do not reflect the internal access.

All chip-select circuits except $\overline{\text{CSBOOT}}$ are disabled out of reset. Chip-select option registers must not be written until base addresses have been written to the proper base address registers. Alternate functions for chip-select pins are enabled if appropriate data bus pins are held low at the release of $\overline{\text{RESET}}$.

Table 17 lists allocation of chip-selects and discrete outputs on the pins of the MCU.

Table 17 Chip-Select and Discrete Output Allocation

Pin	Chip-Select	Discrete Outputs
$\overline{\text{CSBOOT}}$	$\overline{\text{CSBOOT}}$	—
$\overline{\text{BR}}$	$\overline{\text{CS0}}$	—
$\overline{\text{BG}}$	$\overline{\text{CS1}}$	—
$\overline{\text{BGACK}}$	$\overline{\text{CS2}}$	—
FC0	$\overline{\text{CS3}}$	PC0
FC1	$\overline{\text{CS4}}$	PC1
FC2	$\overline{\text{CS5}}$	PC2
ADDR19	$\overline{\text{CS6}}$	PC3
ADDR20	$\overline{\text{CS7}}$	PC4
ADDR21	$\overline{\text{CS8}}$	PC5
ADDR22	$\overline{\text{CS9}}$	PC6
ADDR23	$\overline{\text{CS10}}$	—

3.6.1 Chip-Select Registers

Each chip-select pin can have one or more functions. Chip-select pin assignment registers CSPAR[0:1] determine functions of the pins. Pin assignment registers also determine port size for dynamic bus allocation. A pin data register (PORTC) latches data for chip-select pins that are used for discrete output.

Blocks of addresses are assigned to each chip-select function. Block sizes of two Kbytes to one Mbyte can be selected by writing values to the appropriate base address registers CSBARBT and CSBAR[0:10]. However, because the logic state of ADDR20 is always the same as the state of ADDR19 in the MC68HC16S2, the largest usable block size is 512 Kbytes. Multiple chip-selects assigned to the same block of addresses must have the same number of wait states.

Chip-select option registers CSORBT and CSOR[0:10] determine timing of and conditions for assertion of chip-select signals. Eight parameters, including operating mode, access size, synchronization, and wait state insertion can be specified.

Initialization software usually resides in a peripheral memory device controlled by the chip-select circuits. $\overline{\text{CSBOOT}}$ and registers CSORBT and CSBARBT are provided to support bootstrap operation.

3.6.2 Pin Assignment Registers

The pin assignment registers contain twelve 2-bit fields that determine functions of the chip-select pins. Each pin has two or three possible functions, as shown in **Table 18**.

Table 18 Chip-Select Pin Functions

Assignment Register	16-Bit Chip-Select	Alternate Function	Discrete Output
CSPAR0	$\overline{\text{CSBOOT}}$	$\overline{\text{CSBOOT}}$	—
	$\overline{\text{CS0}}$	$\overline{\text{BR}}$	—
	$\overline{\text{CS1}}$	$\overline{\text{BG}}$	—
	$\overline{\text{CS2}}$	$\overline{\text{BGACK}}$	—
	$\overline{\text{CS3}}$	FC0	PC0
	$\overline{\text{CS4}}$	FC1	PC1
	$\overline{\text{CS5}}$	FC2	PC2
CSPAR1	$\overline{\text{CS6}}$	ADDR19	PC3
	$\overline{\text{CS7}}$	ADDR20	PC4
	$\overline{\text{CS8}}$	ADDR21	PC5
	$\overline{\text{CS9}}$	ADDR22	PC6
	$\overline{\text{CS10}}$	ADDR23	ECLK

Table 19 shows pin assignment field encoding. Pins that have no discrete output function do not use the %00 encoding.

Table 19 Pin Assignment Encodings

Bit Field	Description
00	Discrete output
01	Alternate function
10	Chip-select (8-bit port)
11	Chip-select (16-bit port)

CSPAR0 — Chip-Select Pin Assignment Register 0 **$YFFA44**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	CS5PA[1:0]		CS4PA[1:0]		CS3PA[1:0]		CS2PA[1:0]		CS1PA[1:0]		CS0PA[1:0]		CSBTPA[1:0]	
RESET:															
0	0	DATA2	1	DATA2	1	DATA2	1	DATA1	1	DATA1	1	DATA1	1	1	DATA0

CSPAR0 contains seven 2-bit fields that determine the functions of corresponding chip-select pins. CSPAR0[15:14] are not used. These bits always read zero; writes have no effect. CSPAR0 bit 1 always reads one; writes to CSPAR0 bit 1 have no effect. **Table 20** shows CSPAR0 pin assignments.

Table 20 CSPAR0 Pin Assignments

CSPAR0 Field	Chip-Select Signal	Alternate Signal	Discrete Output
CS5PA[1:0]	CS5	FC2	PC2
CS4PA[1:0]	CS4	FC1	PC1
CS3PA[1:0]	CS3	FC0	PC0
CS2PA[1:0]	CS2	$\overline{\text{BGACK}}$	—
CS1PA[1:0]	CS1	$\overline{\text{BG}}$	—
CS0PA[1:0]	CS0	$\overline{\text{BR}}$	—
CSBTPA[1:0]	$\overline{\text{CSBOOT}}$	—	—

CSPAR1 — Chip-Select Pin Assignment Register 1 **$YFFA46**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	CS10PA[1:0]		CS9PA[1:0]		CS8PA[1:0]		CS7PA[1:0]		CS6PA[1:0]	
RESET:															
0	0	0	0	0	0	DATA7[1]	1	DATA [7:6][1]	1	DATA [7:5][1]	1	DATA [7:4][1]	1	DATA [7:3][1]	1

NOTES:
1. Refer to **Table 21** for CSPAR1 reset state information.

The reset state of DATA[7:3] determines whether pins controlled by CSPAR1 are initially configured as high-order address lines or chip-selects. **Table 21** shows the correspondence between DATA[7:3] and the reset configuration of $\overline{\text{CS}}$[10:6]/ADDR[23:19].

Table 21 Reset Pin Function of $\overline{\text{CS}}$[10:6]

Data Bus Pins at Reset					Chip-Select/Address Bus Pin Function				
DATA7	DATA6	DATA5	DATA4	DATA3	$\overline{\text{CS10}}$/ ADDR23	$\overline{\text{CS9}}$/ ADDR22	$\overline{\text{CS8}}$/ ADDR21	$\overline{\text{CS7}}$/ ADDR20	$\overline{\text{CS8}}$/ ADDR19
1	1	1	1	1	$\overline{\text{CS10}}$	$\overline{\text{CS9}}$	$\overline{\text{CS8}}$	$\overline{\text{CS7}}$	$\overline{\text{CS6}}$
1	1	1	1	0	$\overline{\text{CS10}}$	$\overline{\text{CS9}}$	$\overline{\text{CS8}}$	$\overline{\text{CS7}}$	ADDR19
1	1	1	0	X	$\overline{\text{CS10}}$	$\overline{\text{CS9}}$	$\overline{\text{CS8}}$	ADDR20	ADDR19
1	1	0	X	X	$\overline{\text{CS10}}$	$\overline{\text{CS9}}$	ADDR21	ADDR20	ADDR19
1	0	X	X	X	$\overline{\text{CS10}}$	ADDR22	ADDR21	ADDR20	ADDR19
0	X	X	X	X	ADDR23	ADDR22	ADDR21	ADDR20	ADDR19

CSPAR1 contains five 2-bit fields that determine the functions of corresponding chip-select pins. CSPAR1[15:10] are not used. These bits always read zero; writes have no effect. **Table 22** shows CSPAR1 pin assignments.

Table 22 CSPAR1 Pin Assignments

CSPAR1 Field	Chip-Select Signal	Alternate Signal	Discrete Output
CS10PA[1:0]	$\overline{CS10}$	ADDR23	ECLK
CS9PA[1:0]	$\overline{CS9}$	ADDR22	PC6
CS8PA[1:0]	$\overline{CS8}$	ADDR21	PC5
CS7PA[1:0]	$\overline{CS7}$	ADDR20	PC4
CS6PA[1:0]	$\overline{CS6}$	ADDR19	PC3

Port size determines the way in which bus transfers to external addresses are allocated. Port size of eight bits or sixteen bits can be selected when a pin is assigned as a chip-select. Port size and transfer size affect how the chip-select signal is asserted. Refer to **3.6.4 Option Registers** for more information.

Out of reset, chip-select pin function is determined by the logic level on a corresponding data bus pin. These pins have weak internal pull-up drivers, but can be held low by external devices. Either 16-bit chip-select function (%11) or alternate function (%01) can be selected during reset. All pins except the boot ROM chip-select pin ($\overline{CSBOOT}$) are disabled out of reset.

The $\overline{CSBOOT}$ signal is enabled out of reset. The state of the DATA0 line during reset determines what port width $\overline{CSBOOT}$ uses. If DATA0 is held high (either by the weak internal pull-up driver or by an external pull-up device), 16-bit port size is selected. If DATA0 is held low, 8-bit port size is selected.

A pin programmed as a discrete output drives an external signal to the value specified in the pin data register. No discrete output function is available on pins $\overline{CSBOOT}$, $\overline{BR}$, $\overline{BG}$, or $\overline{BGACK}$. ADDR23 provides ECLK output rather than a discrete output signal.

When a pin is programmed for discrete output or alternate function, internal chip-select logic still functions and can be used to generate $\overline{DSACK}$ or $\overline{AVEC}$ internally on an address and control signal match.

3.6.3 Base Address Registers

Each chip-select has an associated base address register. A base address is the lowest address in the block of addresses enabled by a chip-select. Block size is the extent of the address block above the base address. Block size is determined by the value contained in the BLKSZ field. Multiple chip-selects may be assigned to the same block of addresses so long as each chip-select uses the same number of wait states.

The BLKSZ field determines which bits in the base address field are compared to corresponding bits on the address bus during an access. Provided other constraints determined by option register fields are also satisfied, when a match occurs, the associated chip-select signal is asserted.

After reset, the MCU fetches the address of the first instruction to be executed from the reset vector, located beginning at address $000000 in program space. To support bootstrap operation from reset, the base address field in CSBARBT has a reset value of all zeros. A memory device containing the reset vector and an initialization routine can be automatically enabled by $\overline{CSBOOT}$ after a reset. The block size field in CSBARBT has a reset value of 512 Kbytes.

CSBARBT — Chip-Select Base Address Register Boot ROM **$YFFA48**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR 23	ADDR 22	ADDR 21	ADDR 20	ADDR 19	ADDR 18	ADDR 17	ADDR 16	ADDR 15	ADDR 14	ADDR 13	ADDR 12	ADDR 11	BLKSZ[2:0]		
RESET:															
0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1

CSBAR[0:10] — Chip-Select Base Address Registers **$YFFA4C–$YFFA74**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR 23*	ADDR 22*	ADDR 21*	ADDR 20*	ADDR 19	ADDR 18	ADDR 17	ADDR 16	ADDR 15	ADDR 14	ADDR 13	ADDR 12	ADDR 11	BLKSZ[2:0]		
RESET:															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

*ADDR[23:20] follow the state of ADDR19 in the MC68HC16S2. ADDR[23:20] must match ADDR19 in the base address register for the chip select to be active.

ADDR[23:11] — Base Address Field

This field sets the starting address of a particular address space. The address compare logic uses only the most significant bits to match an address within a block. The value of the base address must be an integer multiple of the block size.

NOTE

Because ADDR[23:20] = ADDR19 in the CPU16, maximum block size is 512 Kbytes. For this same reason, addresses from $080000 to $F7FFFF are inaccessible. Blocks can be based above this dead zone, but the effect of ADDR19 must be considered.

BLKSZ[2:0] — Block Size Field

This field determines the size of the block that must be enabled by the chip-select. **Table 23** shows bit encoding for the base address registers block size field.

Table 23 Block Size Field Bit Encoding

BLKSZ[2:0]	Block Size	Address Lines Compared
000	2 Kbyte	ADDR[23:11]
001	8 Kbyte	ADDR[23:13]
010	16 Kbyte	ADDR[23:14]
011	64 Kbyte	ADDR[23:16]
100	128 Kbyte	ADDR[23:17]
101	256 Kbyte	ADDR[23:18]
110	512 Kbyte	ADDR[23:19]
111	512 Kbyte	ADDR[23:20]

ADDR[23:20] are at the same logic level as ADDR19 during normal operation.

3.6.4 Option Registers

The option registers contain eight fields that determine timing of and conditions for assertion of chip-select signals. To assert a chip-select signal, and to provide $\overline{\text{DSACK}}$ or autovector support, other constraints set by fields in the option register and in the base address register must also be satisfied.

CSORBT — Chip-Select Option Register Boot ROM **$YFFA4A**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MODE	BYTE[1:0]		R/$\overline{\text{W}}$[1:0]		STRB	$\overline{\text{DSACK}}$[3:0]				SPACE[1:0]		IPL[2:0]			$\overline{\text{AVEC}}$
RESET:															
0	1	1	1	1	0	1	1	0	1	1	1	0	0	0	0

CSOR[0:10] — Chip-Select Option Registers **$YFFA4E–YFFA76**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MODE	BYTE[1:0]		R/$\overline{\text{W}}$[1:0]		STRB	$\overline{\text{DSACK}}$[3:0]				SPACE[1:0]		IPL[2:0]			$\overline{\text{AVEC}}$
RESET:															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CSORBT, the option register for $\overline{\text{CSBOOT}}$, contains special reset values that support bootstrap operation from peripheral memory devices.

The following bit descriptions apply to both CSORBT and CSOR[0:10] option registers.

MODE — Asynchronous/Synchronous Mode

0 = Asynchronous mode (chip-select assertion determined by bus control signals)
1 = Synchronous mode (chip-select assertion synchronized with ECLK signal)

In asynchronous mode, the chip-select is asserted synchronized with $\overline{\text{AS}}$ or $\overline{\text{DS}}$.

$\overline{\text{DSACK}}$[3:0] is not used in synchronous mode because a bus cycle is only performed as a synchronous operation. When a match condition occurs on a chip-select programmed for synchronous operation, the chip-select signals the EBI that an ECLK cycle is pending.

BYTE[1:0] — Upper/Lower Byte Option

This field is used only when the chip-select 16-bit port option is selected in the pin assignment register. **Table 24** lists upper/lower byte options.

Table 24 Upper/Lower Byte Options

BYTE[1:0]	Description
00	Disable
01	Lower byte
10	Upper byte
11	Both bytes

R/$\overline{\text{W}}$[1:0] — Read/Write

This field causes a chip-select to be asserted only for reads, only for writes, or for both reads and writes. Refer to **Table 25** for options available.

Table 25 R/$\overline{W}$ Encodings

R/$\overline{W}$[1:0]	Description
00	Reserved
01	Read only
10	Write only
11	Read/Write

STRB — Address Strobe/Data Strobe

0 = Address strobe

1 = Data strobe

This bit controls the timing for assertion of a chip-select in asynchronous mode. Selecting address strobe causes chip-select to be asserted synchronized with address strobe. Selecting data strobe causes chip-select to be asserted synchronized with data strobe.

$\overline{DSACK[3:0]}$ — Data and Size Acknowledge

This field specifies the source of $\overline{DSACK}$ in asynchronous mode. It also allows the user to adjust bus timing with internal $\overline{DSACK}$ generation by controlling the number of wait states that are inserted to optimize bus speed in a particular application. **Table 26** shows the $\overline{DSACK[3:0]}$ encoding. The fast termination encoding (%1110) is used for two-cycle access to external memory.

Table 26 $\overline{DSACK}$ Field Encoding

$\overline{DSACK[3:0]}$	Clock Cycles Required Per Access	Wait States Per Access
0000	3	0 Wait states
0001	4	1 Wait state
0010	5	2 Wait states
0011	6	3 Wait states
0100	7	4 Wait states
0101	8	5 Wait states
0110	9	6 Wait states
0111	10	7 Wait states
1000	11	8 Wait states
1001	12	9 Wait states
1010	13	10 Wait states
1011	14	11 Wait states
1100	15	12 Wait states
1101	16	13 Wait states
1110	2	Fast termination
1111	—	External $\overline{DSACK}$

SPACE[1:0] — Address Space

Use this option field to select an address space for the chip-select logic. The CPU16 normally operates in supervisor space, but interrupt acknowledge cycles must take place in CPU space. **Table 27** shows address space bit encodings.

Table 27 Address Space Bit Encodings

SPACE[1:0]	Address Space
00	CPU space
01	User space
10	Supervisor space
11	Supervisor/User space

IPL[2:0] — Interrupt Priority Level

If the space field is set for CPU space, chip-select logic can be used for interrupt acknowledge. During an interrupt acknowledge cycle, the priority level on address lines ADDR[3:1] is compared to the value in the IPL field. If the values are the same, a chip-select is asserted, provided that other option register conditions are met. **Table 28** shows IPL field encoding.

Table 28 Interrupt Priority Level Field Encoding

IPL[2:0]	Interrupt Priority Level
000	Any level
001	1
010	2
011	3
100	4
101	5
110	6
111	7

This field only affects the response of chip-selects and does not affect interrupt recognition by the CPU. Any level means that chip-select is asserted regardless of the level of the interrupt acknowledge cycle.

$\overline{\text{AVEC}}$ — Autovector Enable

0 = External interrupt vector enabled

1 = Autovector enabled

This field selects one of two methods of acquiring an interrupt vector number during an external interrupt acknowledge cycle.

If the chip-select is configured to trigger on an interrupt acknowledge cycle (SPACE[1:0] = %00) and the $\overline{\text{AVEC}}$ field is set to one, the chip-select circuit generates an internal $\overline{\text{AVEC}}$ signal in response to an external interrupt cycle, and the SIM supplies an automatic vector number. Otherwise, the vector number must be supplied by the requesting device. An internal autovector is generated only in response to interrupt requests from the SIM $\overline{\text{IRQ}}$ pins. Interrupt requests from other IMB modules are ignored.

The $\overline{\text{AVEC}}$ bit must not be used in synchronous mode, as autovector response timing can vary because of ECLK synchronization.

3.6.5 Port C Data Register

Bit values in port C determine the state of chip-select pins used for discrete output. When a pin is assigned as a discrete output, the value in this register appears at the output. This is a read/write register. Bit 7 is not used. Writing to this bit has no effect, and it always returns zero when read.

PORTC — Port C Data Register **$YFFA41**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NOT USED								0	PC6	PC5	PC4	PC3	PC2	PC1	PC0
RESET:								0	1	1	1	1	1	1	1

3.7 General-Purpose Input/Output

SIM pins can be configured as two general-purpose I/O ports, E and F. The following paragraphs describe registers that control the ports.

PORTE0, PORTE1 — Port E Data Register **$YFFA11, YFFA13**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NOT USED								PE7	PE6	PE5	PE4	PE3	PE2	PE1	PE0
RESET:								U	U	U	U	U	U	U	U

A write to the port E data register is stored in the internal data latch and, if any port E pin is configured as an output, the value stored for that bit is driven on the pin. A read of the port E data register returns the value at the pin only if the pin is configured as a discrete input. Otherwise, the value read is the value stored in the register.

The port E data register is a single register that can be accessed in two locations. When accessed at $YFFA11, the register is referred to as PORTE0; when accessed at $YFFA13, the register is referred to as PORTE1. The register can be read or written at any time. It is unaffected by reset.

DDRE — Port E Data Direction Register **$YFFA15**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								DDE7	DDE6	DDE5	DDE4	DDE3	DDE2	DDE1	DDE0
RESET:								0	0	0	0	0	0	0	0

The bits in this register control the direction of the pin drivers when the pins are configured as I/O. Any bit in this register set to one configures the corresponding pin as an output. Any bit in this register cleared to zero configures the corresponding pin as an input. This register can be read or written at any time.

PEPAR — Port E Pin Assignment **$YFFA17**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								PEPA7	PEPA6	PEPA5	PEPA4	PEPA3	PEPA2	PEPA1	PEPA0
RESET:								DATA8	DATA8	DATA8	DATA8	DATA8	DATA8	DATA8	DATA8

The bits in this register control the function of each port E pin. Any bit set to one configures the corresponding pin as a bus control signal, with the function shown in **Table 29**. Any bit cleared to zero defines the corresponding pin to be an I/O pin, controlled by PORTE and DDRE.

Data bus bit 8 controls the state of this register following reset. If DATA8 is set to one during reset, the register is set to $FF, which defines all port E pins as bus control signals. If DATA8 is cleared to zero during reset, this register is set to $00, configuring all port E pins as I/O pins.

Table 29 Port E Pin Assignments

PEPAR Bit	Port E Signal	Bus Control Signal
PEPA7	PE7	SIZ1
PEPA6	PE6	SIZ0
PEPA5	PE5	$\overline{AS}$
PEPA4	PE4	$\overline{DS}$
PEPA3	PE3	—
PEPA2	PE2	$\overline{AVEC}$
PEPA1	PE1	$\overline{DSACK1}$
PEPA0	PE0	$\overline{DSACK0}$

PORTF0, PORTF1 — Port F Data Register **$YFFA19, YFFA1B**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NOT USED								PF7	PF6	PF5	PF4	PF3	PF2	PF1	PF0
RESET:								U	U	U	U	U	U	U	U

A write to the port F data register is stored in the internal data latch, and if any port F pin is configured as an output, the value stored for that bit is driven onto the pin. A read of the port F data register returns the value at the pin only if the pin is configured as a discrete input. Otherwise, the value read is the value stored in the register.

The port F data register is a single register that can be accessed in two locations. When accessed at $YFFA19, the register is referred to as PORTF0; when accessed at $YFFA1B, the register is referred to as PORTF1. The register can be read or written at any time. It is unaffected by reset.

DDRF — Port F Data Direction Register **$YFFA1D**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NOT USED								DDF7	DDF6	DDF5	DDF4	DDF3	DDF2	DDF1	DDF0
RESET:								0	0	0	0	0	0	0	0

The bits in this register control the direction of the pin drivers when the pins are configured as I/O. Any bit in this register set to one configures the corresponding pin as an output. Any bit in this register cleared to zero configures the corresponding pin as an input. This register can be read or written at any time.

PFPAR — Port F Pin Assignment Register **$YFFA1F**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NOT USED								PFPA7	PFPA6	PFPA5	PFPA4	PFPA3	PFPA2	PFPA1	PFPA0
RESET:								DATA9	DATA9	DATA9	DATA9	DATA9	DATA9	DATA9	DATA9

The bits in this register control the function of each port F pin. Any bit cleared to zero defines the corresponding pin to be an I/O pin. Any bit set to one defines the corresponding pin to be an interrupt request signal or MODCLK. The MODCLK signal has no function after reset. **Table 30** shows port F pin assignments.

Table 30 Port F Pin Assignments

PFPAR Field	Port F Signal	Alternate Signal
PFPA7	PF7	$\overline{IRQ7}$
PFPA6	PF6	$\overline{IRQ6}$
PFPA5	PF5	$\overline{IRQ5}$
PFPA4	PF4	$\overline{IRQ4}$
PFPA3	PF3	$\overline{IRQ3}$
PFPA2	PF2	$\overline{IRQ2}$
PFPA1	PF1	$\overline{IRQ1}$
PFPA0	PF0	MODCLK

Data bus pin 9 controls the state of this register following reset. If DATA9 is set to one during reset, the register is set to $FF, which defines all port F pins as interrupt request inputs. If DATA9 is cleared to zero during reset, this register is set to $00, defining all port F pins as I/O pins.

3.8 Resets

Reset procedures handle system initialization and recovery from catastrophic failure. The MCU performs resets with a combination of hardware and software. The SIM determines whether a reset is valid, asserts control signals, performs basic system configuration based on hardware mode-select inputs, then passes control to the CPU.

Reset occurs when an active low logic level on the $\overline{RESET}$ pin is clocked into the SIM. Resets are gated by the CLKOUT signal. Asynchronous resets are assumed to be catastrophic. An asynchronous reset can occur on any clock edge. Synchronous resets are timed to occur at the end of bus cycles. If there is no clock when $\overline{RESET}$ is asserted, reset does not occur until the clock starts. Resets are clocked in order to allow completion of write cycles in progress at the time $\overline{RESET}$ is asserted.

Reset is the highest-priority CPU16 exception. Any processing in progress is aborted by the reset exception, and cannot be restarted. Only essential tasks are performed during reset exception processing. Other initialization tasks must be accomplished by the exception handler routine.

3.8.1 SIM Reset Mode Selection

The logic states of certain data bus pins during reset determine SIM operating configuration. In addition, the state of the MODCLK pin determines system clock source and the state of the $\overline{BKPT}$ pin determines what happens during subsequent breakpoint assertions. **Table 31** is a summary of reset mode selection options.

Table 31 Reset Mode Selection

Mode Select Pin	Default Function (Pin Left High)	Alternate Function (Pin Pulled Low)
DATA0	$\overline{\text{CSBOOT}}$ 16-Bit	$\overline{\text{CSBOOT}}$ 8-Bit
DATA1	$\overline{\text{CS0}}$ $\overline{\text{CS1}}$ $\overline{\text{CS2}}$	$\overline{\text{BR}}$ $\overline{\text{BG}}$ $\overline{\text{BGACK}}$
DATA2	$\overline{\text{CS3}}$ $\overline{\text{CS4}}$ $\overline{\text{CS5}}$	FC0 FC1 FC2
DATA3 DATA4 DATA5 DATA6 DATA7	$\overline{\text{CS6}}$ $\overline{\text{CS}}$[7:6] $\overline{\text{CS}}$[8:6] $\overline{\text{CS}}$[9:6] $\overline{\text{CS}}$[10:6]	ADDR19 ADDR[20:19] ADDR[21:19] ADDR[22:19] ADDR[23:19]
DATA8	$\overline{\text{DSACK}}$[1:0] $\overline{\text{AVEC}}$, $\overline{\text{DS}}$, $\overline{\text{AS}}$ $\overline{\text{SIZ}}$[1:0]	PORTE
DATA9	$\overline{\text{IRQ}}$[7:1] MODCLK	PORTF
DATA11	Test mode disabled	Test mode enabled
MODCLK	VCO = System clock	EXTAL = System clock
$\overline{\text{BKPT}}$	Background mode disabled	Background mode enabled

Data lines have weak internal pull-up drivers. External bus loading can overcome the weak internal pull-up drivers on data bus lines, and hold pins low during reset. Use an active device to hold data bus lines low. Data bus configuration logic must release the bus before the first bus cycle after reset to prevent conflict with external memory devices. The first bus cycle occurs ten CLKOUT cycles after $\overline{\text{RESET}}$ is released. If external mode selection logic causes a conflict of this type, an isolation resistor on the driven lines may be required.

3.8.2 Functions of Pins for Other Modules During Reset

Generally, pins associated with modules other than the SIM default to port functions, and input/output ports are set to input state. This is accomplished by disabling pin functions in the appropriate control registers, and by clearing the appropriate port data direction registers. Refer to individual module sections in this manual for more information. **Table 32** is a summary of module pin function out of reset.

Table 32 Module Pin Functions

Module	Pin Mnemonic	Function
CPU16	DSI/IPIPE1	DSI/IPIPE1
	DSO/IPIPE0	DSO/IPIPE0
	$\overline{\text{BKPT}}$/DSCLK	$\overline{\text{BKPT}}$/DSCLK

3.8.3 Reset Timing

The $\overline{\text{RESET}}$ input must be asserted for a specified minimum period in order for reset to occur. External $\overline{\text{RESET}}$ assertion can be delayed internally for a period equal to the longest bus cycle time (or the bus monitor timeout period) in order to protect write cycles from being aborted by reset. While $\overline{\text{RESET}}$ is asserted, SIM pins are either in a disabled high-impedance state or are driven to their inactive states.

When an external device asserts $\overline{\text{RESET}}$ for the proper period, reset control logic clocks the signal into an internal latch. The control logic drives the $\overline{\text{RESET}}$ pin low for an additional 512 CLKOUT cycles after it detects that the $\overline{\text{RESET}}$ signal is no longer being externally driven, to guarantee this length of reset to the entire system.

If an internal source asserts the reset signal, the reset control logic asserts $\overline{\text{RESET}}$ for a minimum of 512 cycles. If the reset signal is still asserted at the end of 512 cycles, the control logic continues to assert $\overline{\text{RESET}}$ until the internal reset signal is negated.

After 512 cycles have elapsed, the $\overline{\text{RESET}}$ pin goes to an inactive, high-impedance state for ten cycles. At the end of this 10-cycle period, the $\overline{\text{RESET}}$ pin is tested. When the input is at logic level one, reset exception processing begins. If, however, the $\overline{\text{RESET}}$ pin is at logic level zero, the reset control logic drives the pin low for another 512 cycles. At the end of this period, the pin again goes to high-impedance state for ten cycles, then it is tested again. The process repeats until $\overline{\text{RESET}}$ is released.

3.8.4 Power-On Reset

When the SIM clock synthesizer is used to generate the system clock, power-on reset involves special circumstances related to application of system and clock synthesizer power. Regardless of clock source, voltage must be applied to clock synthesizer power input pin V_{DDSYN} in order for the MCU to operate. The following discussion assumes that V_{DDSYN} is applied before and during reset. This minimizes crystal start-up time. When V_{DDSYN} is applied at power-on, start-up time is affected by specific crystal parameters and by oscillator circuit design. V_{DD} ramp-up time also affects pin state during reset.

During power-on reset, an internal circuit in the SIM drives the IMB and external reset lines. The circuit releases the internal reset line as V_{DD} ramps up to the minimum specified value, and SIM pins are initialized. As V_{DD} reaches a specified minimum value, the clock synthesizer VCO begins operation and clock frequency ramps up to specified limp mode frequency. The external $\overline{\text{RESET}}$ line remains asserted until the clock synthesizer PLL locks and 512 CLKOUT cycles elapse.

The SIM clock synthesizer provides clock signals to the other MCU modules. After the clock is running and the internal reset signal is asserted for four clock cycles, these modules reset. V_{DD} ramp time and VCO frequency ramp time determine how long these four cycles take. Worst case is approximately 15 milliseconds. During this period, module port pins may be in an indeterminate state. While input-only pins can be put in a known state by means of external pull-up resistors, external logic on input/output or output-only pins must condition the lines during this time. Active drivers require high-impedance buffers or isolation resistors to prevent conflict.

3.8.5 Use of Three-State Control Pin

Asserting the three-state control (TSC) input causes the MCU to put all output drivers in an inactive, high-impedance state. The signal must remain asserted for ten clock cycles in order for drivers to change state. There are certain constraints on use of TSC during power-on reset:

- When the internal clock synthesizer is used (MODCLK held high during reset), synthesizer ramp-up time affects how long the ten cycles take. Worst case is approximately 20 ms from TSC assertion.
- When an external clock signal is applied (MODCLK held low during reset), pins go to high-impedance state as soon after TSC assertion as ten clock pulses have been applied to the EXTAL pin.
- When TSC assertion takes effect, internal signals are forced to values that can cause inadvertent mode selection. Once the output drivers change state, the MCU must be powered down and restarted before normal operation can resume.

3.9 Interrupts

Interrupt recognition and servicing involve complex interaction between the CPU16, the SIM, and a device or module requesting interrupt service.

The CPU16 provides seven levels of interrupt priority (1–7), seven automatic interrupt vectors, and 200 assignable interrupt vectors. All interrupts with priorities less than seven can be masked by the interrupt priority (IP) field in the condition code register. The CPU16 handles interrupts as a type of asynchronous expression.

There are seven interrupt request signals ($\overline{\text{IRQ}}$[7:1]). These signals are used internally on the IMB, and there are corresponding pins for external interrupt service requests. The CPU treats all interrupt requests as though they come from internal modules — external interrupt requests are treated as interrupt service requests from the SIM. Each of the interrupt request signals corresponds to an interrupt priority level. $\overline{\text{IRQ1}}$ has the lowest priority and $\overline{\text{IRQ7}}$ the highest.

Interrupt recognition is determined by interrupt priority level and interrupt priority mask value. The interrupt priority mask consists of three bits (IP[2:0]) in the CPU16 condition code register. Binary values %000 to %111 provide eight priority masks. Masks prevent an interrupt request of a priority less than or equal to the mask value from being recognized and processed. $\overline{\text{IRQ7}}$, however, is always recognized, even if the mask value is %111.

$\overline{\text{IRQ}}$[7:1] are active-low level-sensitive inputs. The low on the pin must remain asserted until an interrupt acknowledge cycle corresponding to that level is detected.

$\overline{\text{IRQ7}}$ is transition-sensitive as well as level-sensitive: a level 7 interrupt is not detected unless a falling edge transition is detected on the $\overline{\text{IRQ7}}$ line. This prevents redundant servicing and stack overflow. A non-maskable interrupt is generated each time $\overline{\text{IRQ7}}$ is asserted as well as each time the priority mask changes from %111 to a lower number while $\overline{\text{IRQ7}}$ is asserted.

Interrupt requests are sampled on consecutive falling edges of the system clock. Interrupt request input circuitry has hysteresis: to be valid, a request signal must be asserted for at least two consecutive clock periods. Valid requests do not cause immediate exception processing, but are left pending. Pending requests are processed at instruction boundaries or when exception processing of higher-priority exceptions is complete.

The CPU16 does not latch the priority of a pending interrupt request. If an interrupt source of higher priority makes a service request while a lower priority request is pending, the higher priority request is serviced. If an interrupt request with a priority equal to or lower than the current IP mask value is made, the CPU16 does not recognize the occurrence of the request. If simultaneous interrupt requests of different priorities are made, and both have a priority greater than the mask value, the CPU16 recognizes the higher-level request.

3.9.1 Interrupt Acknowledge and Arbitration

When the CPU16 detects one or more interrupt requests of a priority higher than the interrupt priority mask value, it places the interrupt request level on the address bus and initiates a CPU space read cycle. The request level serves two purposes: it is decoded by modules or external devices that have requested interrupt service, to determine whether the current interrupt acknowledge cycle pertains to them, and it is latched into the IP mask field in the CPU16 condition code register, to preclude further interrupts of lower priority during interrupt service.

Modules or external devices that have requested interrupt service must decode the interrupt priority mask value placed on the address bus during the interrupt acknowledge cycle and respond if the priority of the service request corresponds to the mask value. However, before modules or external devices respond, interrupt arbitration takes place.

Arbitration is performed by means of serial contention between values stored in individual module interrupt arbitration (IARB) fields. Each module that can request interrupt service, including the SIM, has an IARB field in its configuration register. IARB fields can be assigned values from %0000 to %1111. In order to implement an arbitration scheme, each module that can request interrupt service must be assigned a unique, non-zero IARB field value during system initialization. Arbitration priorities range from %0001 (lowest) to %1111 (highest) — if the CPU recognizes an interrupt service request from a source that has an IARB field value of %0000, a spurious interrupt exception is processed.

WARNING

Do not assign the same arbitration priority to more than one module. When two or more IARB fields have the same non-zero value, the CPU16 interprets multiple vector numbers at the same time, with unpredictable consequences.

Because the EBI manages external interrupt requests, the SIM IARB value is used for arbitration between internal and external interrupt requests. The reset value of IARB for the SIM is %1111, and the reset IARB value for all other modules is %0000.

Although arbitration is intended to deal with simultaneous requests of the same priority, it always takes place, even when a single source is requesting service. This is important for two reasons: the EBI does not transfer the interrupt acknowledge read cycle to the external bus unless the SIM wins contention, and failure to contend causes the interrupt acknowledge bus cycle to be terminated early, by a bus error.

When arbitration is complete, the module with the highest arbitration priority must terminate the bus cycle. Internal modules place an interrupt vector number on the data bus and generate appropriate internal cycle termination signals. In the case of an external interrupt request, after the interrupt acknowledge cycle is transferred to the external bus, the appropriate external device must decode the mask value and respond with a vector number, then generate data and size acknowledge ($\overline{\text{DSACK}}$) termination signals, or it must assert the autovector ($\overline{\text{AVEC}}$) request signal. If the device does not respond in time, the EBI bus monitor asserts the bus error signal ($\overline{\text{BERR}}$), and a spurious interrupt exception is taken.

Chip-select logic can also be used to generate internal $\overline{\text{AVEC}}$ or $\overline{\text{DSACK}}$ signals in response to interrupt requests from external devices. Chip-select address match logic functions only after the EBI transfers an interrupt acknowledge cycle to the external bus following IARB contention. If a module makes an interrupt request of a certain priority, and the appropriate chip-select registers are programmed to generate $\overline{\text{AVEC}}$ or $\overline{\text{DSACK}}$ signals in response to an interrupt acknowledge cycle for that priority level, chip-select logic does not respond to the interrupt acknowledge cycle, and the internal module supplies a vector number and generates internal cycle termination signals.

For periodic timer interrupts, the PIRQL field in the periodic interrupt control register (PICR) determines PIT priority level. A PIRQL value of %000 means that PIT interrupts are inactive. By hardware convention, when the CPU16 receives simultaneous interrupt requests of the same level from more than one SIM source (including external devices), the periodic interrupt timer is given the highest priority, followed by the $\overline{\text{IRQ}}$ pins. Refer to **3.4.6 Periodic Interrupt Timer** for more information.

3.9.2 Interrupt Processing Summary

A summary of the interrupt processing sequence follows. When the sequence begins, a valid interrupt service request has been detected and is pending.

A. The CPU finishes higher priority exception processing or reaches an instruction boundary.
B. The processor state is stacked, then the CCR PK extension field is cleared.
C. The interrupt acknowledge cycle begins:

1. FC[2:0] are driven to %111 (CPU space) encoding.
2. The address bus is driven as follows: ADDR[23:20] = %1111; ADDR[19:16] = %1111, which indicates that the cycle is an interrupt acknowledge CPU space cycle; ADDR[15:4] = %111111111111; ADDR[3:1] = the priority of the interrupt request being acknowledged; and ADDR0 = %1.
3. The request level is latched from the address bus into the IP mask field in the condition code register.

D. Modules or external peripherals that have requested interrupt service decode the priority value on ADDR[3:1]. If request priority is the same as acknowledged priority, arbitration by IARB contention takes place.
E. After arbitration, the interrupt acknowledge cycle is completed in one of the following ways:
   1. When there is no contention (IARB = %0000), the spurious interrupt monitor asserts $\overline{\text{BERR}}$, and the CPU16 generates the spurious interrupt vector number.
   2. The dominant interrupt source supplies a vector number and $\overline{\text{DSACK}}$ signals appropriate to the access. The CPU16 acquires the vector number.
   3. The internal $\overline{\text{AVEC}}$ signal is asserted by the dominant interrupt source and the CPU16 generates an autovector number corresponding to interrupt priority.
   4. The bus monitor asserts $\overline{\text{BERR}}$ and the CPU16 generates the spurious interrupt vector number.
F. The vector number is converted to a vector address.
G. The content of the vector address is loaded into the PC, and the processor transfers control to the exception handler routine.

3.10 Factory Test Block

The test submodule supports scan-based testing of the various MCU modules. It is integrated into the SIM to support production testing.

Test submodule registers are intended for Motorola use. Register names and addresses are provided to indicate that these addresses are occupied.

SIMTR — System Integration Module Test Register **$YFFA02**

SIMTRE — System Integration Module Test Register (E Clock) **$YFFA08**

TSTMSRA — Master Shift Register A **$YFFA30**

TSTMSRB — Master Shift Register B **$YFFA32**

TSTSC — Test Module Shift Count **$YFFA34**

TSTRC — Test Module Repetition Count **$YFFA36**

CREG — Test Module Control Register **$YFFA38**

DREG — Test Module Distributed Register **$YFFA3A**

4 Central Processing Unit

The CPU16 is a true 16-bit, high-speed device. It was designed to give M68HC11 users a path to higher performance while maintaining maximum compatibility with existing systems.

4.1 Overview

The CPU16 instruction set is optimized for high performance. There are two 16-bit general-purpose accumulators and three 16-bit index registers. The CPU16 supports 8-bit (byte), 16-bit (word), and 32-bit (long-word) load and store operations, as well as 16- and 32-bit signed fractional operations. Code development is simplified by the background debugging mode.

CPU16 memory space includes a one Mbyte data space and a one Mbyte program space. Twenty-bit addressing and transparent bank switching are used to implement extended memory. In addition, most instructions automatically handle bank boundaries.

The CPU16 includes instructions and hardware to implement control-oriented digital signal processing functions with a minimum of interfacing. A multiply and accumulate unit provides the capability to multiply signed 16-bit fractional numbers and store the resulting 32-bit fixed point product in a 36-bit accumulator. Modulo addressing supports finite impulse response filters.

Use of high-level languages is increasing as controller applications become more complex and control programs become larger. These languages make rapid development of portable software possible. The CPU16 instruction set supports high-level languages.

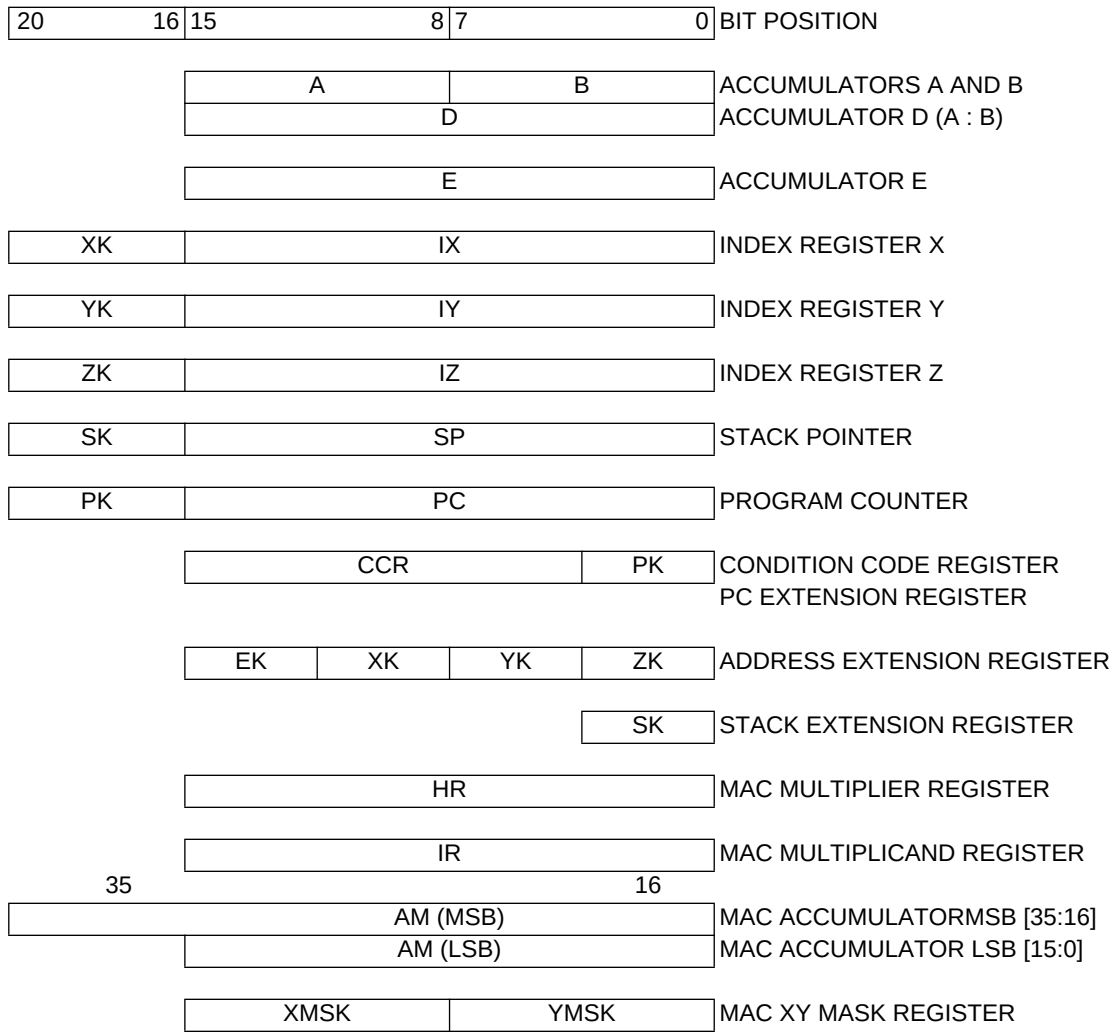
4.2 M68HC11 Compatibility

The CPU16 architecture is a superset of the M68HC11 CPU architecture. All M68HC11 CPU resources are available in the CPU16. M68HC11 CPU instructions are either directly implemented in the CPU16, or have been replaced by instructions with an equivalent form. The instruction sets are source code compatible, but some instructions are executed differently in the CPU16. These instructions are mainly related to interrupt and exception processing — M68HC11 CPU code that processes interrupts, handles stack frames, or manipulates the condition code register must be rewritten.

CPU16 execution times and number of cycles for all instructions are different from those of the M68HC11 CPU. As a result, cycle-related delays and timed control routines may be affected.

The CPU16 also has several new or enhanced addressing modes. M68HC11 CPU direct mode addressing has been replaced by a special form of indexed addressing that uses the new IZ register and a reset vector to provide greater flexibility.

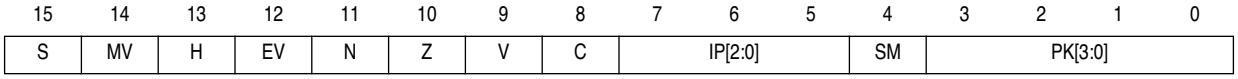
4.3 Programming Model



Accumulator A — 8-bit general-purpose register
Accumulator B — 8-bit general-purpose register
Accumulator D — 16-bit general-purpose register formed by concatenating accumulators A and B
Accumulator E — 16-bit general-purpose register
Index Register X — 16-bit indexing register, addressing extended by XK field in K register
Index Register Y — 16-bit indexing register, addressing extended by YK field in K register
Index Register Z — 16-bit indexing register, addressing extended by ZK field in K register
Stack Pointer — 16-bit dedicated register, addressing extended by the SK register
Program Counter — 16-bit dedicated register, addressing extended by PK field in CCR
Condition Code Register — 16-bit register containing condition flags, interrupt priority mask, and the program counter address extension field
K Register — 16-bit register made up of four 4-bit address extension fields
SK Register — 4-bit register containing the stack pointer address extension field
H Register — 16-bit multiply and accumulate input (multiplier) register
I Register — 16-bit multiply and accumulate input (multiplicand) register
MAC Accumulator — 36-bit multiply and accumulate result register
XMSK, YMSK — Determine which bits change when an offset is added

Figure 12 CPU16 Programming Model

4.3.1 Condition Code Register



15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S	MV	H	EV	N	Z	V	C	IP[2:0]			SM	PK[3:0]			

S — STOP Enable
0 = Stop clock when LPSTOP instruction is executed
1 = Perform NOP when LPSTOP instruction is executed

MV — Accumulator M overflow flag
MV is set when an overflow into AM35 has occurred.

H — Half Carry Flag
H is set when a carry from A3 or B3 occurs during BCD addition.

EV — Extension Bit Overflow Flag
EV is set when an overflow into AM31 has occurred.

N — Negative Flag
N is set when the MSB of a result register is set.

Z — Zero Flag
Z is set when all bits of a result register are zero.

V — Overflow Flag
V is set when a two's complement overflow occurs as the result of an operation.

C — Carry Flag
C is set when a carry or borrow occurs during an arithmetic operation. This flag is also used during shift and rotate to facilitate multiple word operations.

IP[2:0] — Interrupt Priority Field
The priority value in this field (0 to 7) is used to mask interrupts.

SM — Saturate Mode Bit
When SM is set, if either EV or MV is set, data read from AM using TMER or TMET is given maximum positive or negative value, depending on the state of the AM sign bit before overflow.

PK[3:0] — Program Counter Address Extension Field
This field is concatenated with the program counter to form a 20-bit address.

4.4 Data Types

The CPU16 supports the following data types:

- Bit data
- 8-bit (byte) and 16-bit (word) integers
- 32-bit long integers
- 16-bit and 32-bit signed fractions (MAC operations only)
- 20-bit effective address consisting of 16-bit page address plus 4-bit extension

A byte is eight bits wide and can be accessed at any byte location. A word is composed of two consecutive bytes, and is addressed at the lower byte. Instruction fetches are always accessed on word boundaries. Word operands are normally accessed on word boundaries as well, but can be accessed on odd byte boundaries, with a substantial performance penalty.

To be compatible with the M68HC11, misaligned word transfers and misaligned stack accesses are allowed. Transferring a misaligned word requires two successive byte operations.

4.5 Addressing Modes

The CPU16 provides ten types of addressing. Each type encompasses one or more addressing modes. Six CPU16 addressing types are identical to M68HC11 addressing types.

All modes generate ADDR[15:0]. This address is combined with ADDR[19:16] from an extension field to form a 20-bit effective address. Extension fields are part of a bank switching scheme that provides the CPU16 with a one Mbyte address space. Bank switching is transparent to most instructions. ADDR[19:16] of the effective address change when an access crosses a bank boundary. However, it is important to note that the value of the associated extension field is dependent on the type of instruction, and usually does not change as a result of effective address calculation.

In the immediate modes, the instruction argument is contained in bytes or words immediately following the instruction. The effective address is the address of the byte following the instruction. The AIS, AIX/Y/Z, ADDD and ADDE instructions have an extended 8-bit mode where the immediate value is an 8-bit signed number that is sign-extended to 16 bits, and then added to the appropriate register. Use of the extended 8-bit mode decreases execution time.

Extended mode instructions contain ADDR[15:0] in the word following the opcode. The effective address is formed by concatenating EK and the 16-bit extension.

In the indexed modes, registers IX, IY, and IZ, together with their associated extension fields, are used to calculate the effective address. Signed 16-bit mode and signed 20-bit mode are extensions to the M68HC11 indexed addressing mode.

For 8-bit indexed mode, an 8-bit unsigned offset contained in the instruction is added to the value contained in the index register and its associated extension field.

For 16-bit mode, a 16-bit signed offset contained in the instruction is added to the value contained in the index register and its associated extension field.

For 20-bit mode, a 20-bit signed offset is added to the value contained in the index register. This mode is used for JMP and JSR instructions.

Inherent mode instructions use information available to the processor to determine the effective address. Operands (if any) are system resources and are thus not fetched from memory.

Accumulator offset mode adds the contents of 16-bit accumulator E to one of the index registers and its associated extension field to form the effective address. This mode allows use of index registers and an accumulator within loops without corrupting accumulator D.

Relative modes are used for branch and long branch instructions. A byte or word signed two's complement offset is added to the program counter if the branch condition is satisfied. The new PC value, concatenated with the PK field, is the effective address.

Post-modified index mode is used with the MOVB and MOVW instructions. A signed 8-bit offset is added to index register X after the effective address formed by XK and IX is used.

In M68HC11 systems, direct mode can be used to perform rapid accesses to RAM or I/O mapped into page 0 ($0000 to $00FF), but the CPU16 uses the first 512 bytes of page 0 for exception vectors. To compensate for the loss of direct mode, the ZK field and index register Z have been assigned reset initialization vectors. By resetting the ZK field to a chosen page, and using 8-bit unsigned index mode with IZ, a programmer can access useful data structures anywhere in the address map.

4.6 Instruction Set

The CPU16 instruction set is based on that of the M68HC11, but the opcode map has been rearranged to maximize performance with a 16-bit data bus. All M68HC11 instructions are supported by the CPU16, although they may be executed differently. Most M68HC11 code runs on the CPU16 following reassembly. However, take into account changed instruction times, the interrupt mask, and the new interrupt stack frame.

The CPU16 has a full range of 16-bit arithmetic and logic instructions, including signed and unsigned multiplication and division. New instructions have been added to support extended addressing and digital signal processing.

Table 33 is a quick reference to the entire CPU16 instruction set. Because it is only affected by a few instructions, the LSB of the condition code register is not shown in the table. Instructions that affect the interrupt mask and PK field are noted. **Table 34** provides a key to the table nomenclature.

Table 33 Instruction Set Summary

Mnemonic	Operation	Description	Address	Instruction			Condition Codes							
			Mode	Opcode	Operand	Cycles	S	MV	H	EV	N	Z	V	C
ABA	Add B to A	(A) + (B) ⇒ A	INH	370B	—	2	—	—	Δ	—	Δ	Δ	Δ	Δ
ABX	Add B to IX	(XK : IX) + (000 : B) ⇒ XK : IX	INH	374F	—	2	—	—	—	—	—	—	—	—
ABY	Add B to IY	(YK : IY) + (000 : B) ⇒ YK : IY	INH	375F	—	2	—	—	—	—	—	—	—	—
ABZ	Add B to IZ	(ZK : IZ) + (000 : B) ⇒ ZK : IZ	INH	376F	—	2	—	—	—	—	—	—	—	—
ACE	Add E to AM	(AM[31:16]) + (E) ⇒ AM	INH	3722	—	2	—	Δ	—	Δ	—	—	—	—
ACED	Add E : D to AM	(AM) + (E : D) ⇒ AM	INH	3723	—	4	—	Δ	—	Δ	—	—	—	—
ADCA	Add with Carry to A	(A) + (M) + C ⇒ A	IND8, X	43	ff	6	—	—	Δ	—	Δ	Δ	Δ	Δ
			IND8, Y	53	ff	6								
			IND8, Z	63	ff	6								
			IMM8	73	ii	2								
			IND16, X	1743	gggg	6								
			IND16, Y	1753	gggg	6								
			IND16, Z	1763	gggg	6								
			EXT	1773	hh ll	6								
			E, X	2743	—	6								
			E, Y	2753	—	6								
			E, Z	2763	—	6								
ADCB	Add with Carry to B	(B) + (M) + C ⇒ B	IND8, X	C3	ff	6	—	—	Δ	—	Δ	Δ	Δ	Δ
			IND8, Y	D3	ff	6								
			IND8, Z	E3	ff	6								
			IMM8	F3	ii	2								
			IND16, X	17C3	gggg	6								
			IND16, Y	17D3	gggg	6								
			IND16, Z	17E3	gggg	6								
			EXT	17F3	hh ll	6								
			E, X	27C3	—	6								
			E, Y	27D3	—	6								
			E, Z	27E3	—	6								
ADCD	Add with Carry to D	(D) + (M : M + 1) + C ⇒ D	IND8, X	83	ff	6	—	—	—	—	Δ	Δ	Δ	Δ
			IND8, Y	93	ff	6								
			IND8, Z	A3	ff	6								
			IMM16	37B3	jj kk	4								
			IND16, X	37C3	gggg	6								
			IND16, Y	37D3	gggg	6								
			IND16, Z	37E3	gggg	6								
			EXT	37F3	hh ll	6								
			E, X	2783	—	6								
			E, Y	2793	—	6								
			E, Z	27A3	—	6								
ADCE	Add with Carry to E	(E) + (M : M + 1) + C ⇒ E	IMM16	3733	jj kk	4	—	—	—	—	Δ	Δ	Δ	Δ
			IND16, X	3743	gggg	6								
			IND16, Y	3753	gggg	6								
			IND16, Z	3763	gggg	6								
			EXT	3773	hh ll	6								
ADDA	Add to A	(A) + (M) ⇒ A	IND8, X	41	ff	6	—	—	Δ	—	Δ	Δ	Δ	Δ
			IND8, Y	51	ff	6								
			IND8, Z	61	ff	6								
			IMM8	71	ii	2								
			IND16, X	1741	gggg	6								
			IND16, Y	1751	gggg	6								
			IND16, Z	1761	gggg	6								
			EXT	1771	hh ll	6								
			E, X	2741	—	6								
			E, Y	2751	—	6								
			E, Z	2761	—	6								

Table 33 Instruction Set Summary (Continued)

Mnemonic	Operation	Description	Address Mode	Instruction Opcode	Instruction Operand	Instruction Cycles	Condition Codes S	MV	H	EV	N	Z	V	C
ADDB	Add to B	(B) + (M) ⇒ B	IND8, X	C1	ff	6	—	—	Δ	—	Δ	Δ	Δ	Δ
			IND8, Y	D1	ff	6								
			IND8, Z	E1	ff	6								
			IMM8	F1	ii	2								
			IND16, X	17C1	gggg	6								
			IND16, Y	17D1	gggg	6								
			IND16, Z	17E1	gggg	6								
			EXT	17F1	hh ll	6								
			E, X	27C1	—	6								
			E, Y	27D1	—	6								
			E, Z	27E1	—	6								
ADDD	Add to D	(D) + (M : M + 1) ⇒ D	IND8, X	81	ff	6	—	—	—	—	Δ	Δ	Δ	Δ
			IND8, Y	91	ff	6								
			IND8, Z	A1	ff	6								
			IMM8	FC	ii	2								
			IMM16	37B1	jj kk	4								
			IND16, X	37C1	gggg	6								
			IND16, Y	37D1	gggg	6								
			IND16, Z	37E1	gggg	6								
			EXT	37F1	hh ll	6								
			E, X	2781	—	6								
			E, Y	2791	—	6								
			E, Z	27A1	—	6								
ADDE	Add to E	(E) + (M : M + 1) ⇒ E	IMM8	7C	ii	2	—	—	—	—	Δ	Δ	Δ	Δ
			IMM16	3731	jj kk	4								
			IND16, X	3741	gggg	6								
			IND16, Y	3751	gggg	6								
			IND16, Z	3761	gggg	6								
			EXT	3771	hh ll	6								
ADE	Add D to E	(E) + (D) ⇒ E	INH	2778	—	2	—	—	—	—	Δ	Δ	Δ	Δ
ADX	Add D to IX	(XK : IX) + (20 « D) ⇒ XK : IX	INH	37CD	—	2	—	—	—	—	—	—	—	—
ADY	Add D to IY	(YK : IY) + (20 « D) ⇒ YK : IY	INH	37DD	—	2	—	—	—	—	—	—	—	—
ADZ	Add D to IZ	(ZK : IZ) + (20 « D) ⇒ ZK : IZ	INH	37ED	—	2	—	—	—	—	—	—	—	—
AEX	Add E to IX	(XK : IX) + (20 « D) ⇒ XK : IX	INH	374D	—	2	—	—	—	—	—	—	—	—
AEY	Add E to IY	(YK : IY) + (20 « D) ⇒ YK : IY	INH	375D	—	2	—	—	—	—	—	—	—	—
AEZ	Add E to IZ	(ZK : IZ) + (20 « D) ⇒ ZK : IZ	INH	376D	—	2	—	—	—	—	—	—	—	—
AIS	Add Immediate Data to Stack Pointer	(SK : SP) + (20 « IMM) ⇒ SK : SP	IMM8	3F	ii	2	—	—	—	—	—	—	—	—
			IMM16	373F	jj kk	4								
AIX	Add Immediate Value to IX	(XK : IX) + (20 « IMM) ⇒ XK : IX	IMM8	3C	ii	2	—	—	—	—	—	Δ	—	—
			IMM16	373C	jj kk	4								
AIY	Add Immediate Value to IY	(YK : IY) + (20 « IMM) ⇒ YK : IY	IMM8	3D	ii	2	—	—	—	—	—	Δ	—	—
			IMM16	373D	jj kk	4								
AIZ	Add Immediate Value to IZ	(ZK : IZ) + (20 « IMM) ⇒ ZK : IZ	IMM8	3E	ii	2	—	—	—	—	—	Δ	—	—
			IMM16	373E	jj kk	4								
ANDA	AND A	(A) • (M) ⇒ A	IND8, X	46	ff	6	—	—	—	—	Δ	Δ	0	—
			IND8, Y	56	ff	6								
			IND8, Z	66	ff	6								
			IMM8	76	ii	2								
			IND16, X	1746	gggg	6								
			IND16, Y	1756	gggg	6								
			IND16, Z	1766	gggg	6								
			EXT	1776	hh ll	6								
			E, X	2746	—	6								
			E, Y	2756	—	6								
			E, Z	2766	—	6								

Table 33 Instruction Set Summary (Continued)

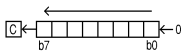
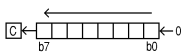
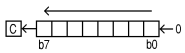
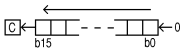
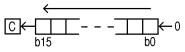
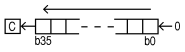
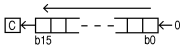
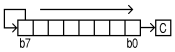
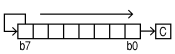
Mnemonic	Operation	Description	Address Mode	Instruction Opcode	Instruction Operand	Instruction Cycles	Condition Codes S	MV	H	EV	N	Z	V	C
ANDB	AND B	(B) • (M) ⇒ B	IND8, X	C6	ff	6	—	—	—	—	Δ	Δ	0	—
			IND8, Y	D6	ff	6								
			IND8, Z	E6	ff	6								
			IMM8	F6	ii	2								
			IND16, X	17C6	gggg	6								
			IND16, Y	17D6	gggg	6								
			IND16, Z	17E6	gggg	6								
			EXT	17F6	hh ll	6								
			E, X	27C6	—	6								
			E, Y	27D6	—	6								
			E, Z	27E6	—	6								
ANDD	AND D	(D) • (M : M + 1) ⇒ D	IND8, X	86	ff	6	—	—	—	—	Δ	Δ	0	—
			IND8, Y	96	ff	6								
			IND8, Z	A6	ff	6								
			IMM16	37B6	jj kk	4								
			IND16, X	37C6	gggg	6								
			IND16, Y	37D6	gggg	6								
			IND16, Z	37E6	gggg	6								
			EXT	37F6	hh ll	6								
			E, X	2786	—	6								
			E, Y	2796	—	6								
			E, Z	27A6	—	6								
ANDE	AND E	(E) • (M : M + 1) ⇒ E	IMM16	3736	jj kk	4	—	—	—	—	Δ	Δ	0	—
			IND16, X	3746	gggg	6								
			IND16, Y	3756	gggg	6								
			IND16, Z	3766	gggg	6								
			EXT	3776	hh ll	6								
ANDP[1]	AND CCR	(CCR) • IMM16⇒ CCR	IMM16	373A	jj kk	4	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ
ASL	Arithmetic Shift Left		IND8, X	04	ff	8	—	—	—	—	Δ	Δ	Δ	Δ
			IND8, Y	14	ff	8								
			IND8, Z	24	ff	8								
			IND16, X	1704	gggg	8								
			IND16, Y	1714	gggg	8								
			IND16, Z	1724	gggg	8								
			EXT	1734	hh ll	8								
ASLA	Arithmetic Shift Left A		INH	3704	—	2	—	—	—	—	Δ	Δ	Δ	Δ
ASLB	Arithmetic Shift Left B		INH	3714	—	2	—	—	—	—	Δ	Δ	Δ	Δ
ASLD	Arithmetic Shift Left D		INH	27F4	—	2	—	—	—	—	Δ	Δ	Δ	Δ
ASLE	Arithmetic Shift Left E		INH	2774	—	2	—	—	—	—	Δ	Δ	Δ	Δ
ASLM	Arithmetic Shift Left AM		INH	27B6	—	4	—	Δ	—	Δ	Δ	—	—	Δ
ASLW	Arithmetic Shift Left Word		IND16, X	2704	gggg	8	—	—	—	—	Δ	Δ	Δ	Δ
			IND16, Y	2714	gggg	8								
			IND16, Z	2724	gggg	8								
			EXT	2734	hh ll	8								
ASR	Arithmetic Shift Right		IND8, X	0D	ff	8	—	—	—	—	Δ	Δ	Δ	Δ
			IND8, Y	1D	ff	8								
			IND8, Z	2D	ff	8								
			IND16, X	170D	gggg	8								
			IND16, Y	171D	gggg	8								
			IND16, Z	172D	gggg	8								
			EXT	173D	hh ll	8								
ASRA	Arithmetic Shift Right A		INH	370D	—	2	—	—	—	—	Δ	Δ	Δ	Δ

Table 33 Instruction Set Summary (Continued)

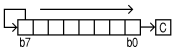
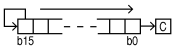
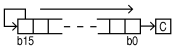
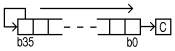
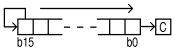
Mnemonic	Operation	Description	Address Mode	Instruction Opcode	Instruction Operand	Instruction Cycles	Condition Codes S	MV	H	EV	N	Z	V	C
ASRB	Arithmetic Shift Right B		INH	371D	—	2	—	—	—	—	Δ	Δ	Δ	Δ
ASRD	Arithmetic Shift Right D		INH	27FD	—	2	—	—	—	—	Δ	Δ	Δ	Δ
ASRE	Arithmetic Shift Right E		INH	277D	—	2	—	—	—	—	Δ	Δ	Δ	Δ
ASRM	Arithmetic Shift Right AM		INH	27BA	—	4	—	—	—	Δ	Δ	—	—	Δ
ASRW	Arithmetic Shift Right Word		IND16, X IND16, Y IND16, Z EXT	270D 271D 272D 273D	gggg gggg gggg hh ll	8 8 8 8	—	—	—	—	Δ	Δ	Δ	Δ
BCC[2]	Branch if Carry Clear	If C = 0, branch	REL8	B4	rr	6, 2	—	—	—	—	—	—	—	—
BCLR	Clear Bit(s)	(M) • ($\overline{\text{Mask}}$) ⇒ M	IND8, X IND8, Y IND8, Z IND16, X IND16, Y IND16, Z EXT	1708 1718 1728 08 18 28 38	mm ff mm ff mm ff mm gggg mm gggg mm gggg mm hh ll	8 8 8 8 8 8 8	—	—	—	—	Δ	Δ	0	—
BCLRW	Clear Bit(s) in a Word	(M : M + 1) • ($\overline{\text{Mask}}$) ⇒ M : M + 1	IND16, X IND16, Y IND16, Z EXT	2708 2718 2728 2738	gggg mmmm gggg mmmm gggg mmmm hh ll mmmm	10 10 10 10	—	—	—	—	Δ	Δ	0	—
BCS[2]	Branch if Carry Set	If C = 1, branch	REL8	B5	rr	6, 2	—	—	—	—	—	—	—	—
BEQ[2]	Branch if Equal	If Z = 1, branch	REL8	B7	rr	6, 2	—	—	—	—	—	—	—	—
BGE[2]	Branch if Greater Than or Equal to Zero	If N ⊕ V = 0, branch	REL8	BC	rr	6, 2	—	—	—	—	—	—	—	—
BGND	Enter Background Debug Mode	If BDM enabled, begin debug; else, illegal instruction trap	INH	37A6	—	—	—	—	—	—	—	—	—	—
BGT[2]	Branch if Greater Than Zero	If Z ✣ (N ⊕ V) = 0, branch	REL8	BE	rr	6, 2	—	—	—	—	—	—	—	—
BHI[2]	Branch if Higher	If C ✣ Z = 0, branch	REL8	B2	rr	6, 2	—	—	—	—	—	—	—	—
BITA	Bit Test A	(A) • (M)	IND8, X IND8, Y IND8, Z IMM8 IND16, X IND16, Y IND16, Z EXT E, X E, Y E, Z	49 59 69 79 1749 1759 1769 1779 2749 2759 2769	ff ff ff ii gggg gggg gggg hh ll — — —	6 6 6 2 6 6 6 6 6 6 6	—	—	—	—	Δ	Δ	0	—
BITB	Bit Test B	(B) • (M)	IND8, X IND8, Y IND8, Z IMM8 IND16, X IND16, Y IND16, Z EXT E, X E, Y E, Z	C9 D9 E9 F9 17C9 17D9 17E9 17F9 27C9 27D9 27E9	ff ff ff ii gggg gggg gggg hh ll — — —	6 6 6 2 6 6 6 6 6 6 6	—	—	—	—	Δ	Δ	0	—
BLE[2]	Branch if Less Than or Equal to Zero	If Z ✣ (N ⊕ V) = 1, branch	REL8	BF	rr	6, 2	—	—	—	—	—	—	—	—

Table 33 Instruction Set Summary (Continued)

Mnemonic	Operation	Description	Address Mode	Instruction Opcode	Instruction Operand	Instruction Cycles	Condition Codes S	MV	H	EV	N	Z	V	C
BLS[2]	Branch if Lower or Same	If C ✣ Z = 1, branch	REL8	B3	rr	6, 2	—	—	—	—	—	—	—	—
BLT[2]	Branch if Less Than Zero	If N ⊕ V = 1, branch	REL8	BD	rr	6, 2	—	—	—	—	—	—	—	—
BMI[2]	Branch if Minus	If N = 1, branch	REL8	BB	rr	6, 2	—	—	—	—	—	—	—	—
BNE[2]	Branch if Not Equal	If Z = 0, branch	REL8	B6	rr	6, 2	—	—	—	—	—	—	—	—
BPL[2]	Branch if Plus	If N = 0, branch	REL8	BA	rr	6, 2	—	—	—	—	—	—	—	—
BRA	Branch Always	If 1 = 1, branch	REL8	B0	rr	6	—	—	—	—	—	—	—	—
BRCLR[2]	Branch if Bit(s) Clear	If (M) • (Mask) = 0, branch	IND8, X IND8, Y IND8, Z IND16, X IND16, Y IND16, Z EXT	CB DB EB 0A 1A 2A 3A	mm ff rr mm ff rr mm ff rr mm gggg rrrr mm gggg rrrr mm gggg rrrr mm hh ll rrrr	10, 12 10, 12 10, 12 10, 14 10, 14 10, 14 10, 14	—	—	—	—	—	—	—	—
BRN	Branch Never	If 1 = 0, branch	REL8	B1	rr	2	—	—	—	—	—	—	—	—
BRSET[2]	Branch if Bit(s) Set	If (M̄) • (Mask) = 0, branch	IND8, X IND8, Y IND8, Z IND16, X IND16, Y IND16, Z EXT	8B 9B AB 0B 1B 2B 3B	mm ff rr mm ff rr mm ff rr mm gggg rrrr mm gggg rrrr mm gggg rrrr mm hh ll rrrr	10, 12 10, 12 10, 12 10, 14 10, 14 10, 14 10, 14	—	—	—	—	—	—	—	—
BSET	Set Bit(s)	(M) ✣ (Mask) ⇒ M	IND8, X IND8, Y IND8, Z IND16, X IND16, Y IND16, Z EXT	1709 1719 1729 09 19 29 39	mm ff mm ff mm ff mm gggg mm gggg mm gggg mm hh ll	8 8 8 8 8 8 8	—	—	—	—	Δ	Δ	0	Δ
BSETW	Set Bit(s) in Word	(M : M + 1) ✣ (Mask) ⇒ M : M + 1	IND16, X IND16, Y IND16, Z EXT	2709 2719 2729 2739	gggg mmmm gggg mmmm gggg mmmm hh ll mmmm	10 10 10 10	—	—	—	—	Δ	Δ	0	Δ
BSR	Branch to Subroutine	(PK : PC) - 2 ⇒ PK : PC Push (PC) (SK : SP) - 2 ⇒ SK : SP Push (CCR) (SK : SP) - 2 ⇒ SK : SP (PK : PC) + Offset ⇒ PK : PC	REL8	36	rr	10	—	—	—	—	—	—	—	—
BVC[2]	Branch if Overflow Clear	If V = 0, branch	REL8	B8	rr	6, 2	—	—	—	—	—	—	—	—
BVS[2]	Branch if Overflow Set	If V = 1, branch	REL8	B9	rr	6, 2	—	—	—	—	—	—	—	—
CBA	Compare A to B	(A) – (B)	INH	371B	—	2	—	—	—	—	Δ	Δ	Δ	Δ
CLR	Clear a Byte in Memory	$00 ⇒ M	IND8, X IND8, Y IND8, Z IND16, X IND16, Y IND16, Z EXT	05 15 25 1705 1715 1725 1735	ff ff ff gggg gggg gggg hh ll	4 4 4 6 6 6 6	—	—	—	—	0	1	0	0
CLRA	Clear A	$00 ⇒ A	INH	3705	—	2	—	—	—	—	0	1	0	0
CLRB	Clear B	$00 ⇒ B	INH	3715	—	2	—	—	—	—	0	1	0	0
CLRD	Clear D	$0000 ⇒ D	INH	27F5	—	2	—	—	—	—	0	1	0	0
CLRE	Clear E	$0000 ⇒ E	INH	2775	—	2	—	—	—	—	0	1	0	0

Table 33 Instruction Set Summary (Continued)

Mnemonic	Operation	Description	Address Mode	Instruction Opcode	Instruction Operand	Instruction Cycles	Condition Codes S	MV	H	EV	N	Z	V	C
CLRM	Clear AM	$000000000 ⇒ AM[35:0]	INH	27B7	—	2	—	0	—	0	—	—	—	—
CLRW	Clear a Word in Memory	$0000 ⇒ M : M + 1	IND16, X	2705	gggg	6	—	—	—	—	0	1	0	0
			IND16, Y	2715	gggg	6								
			IND16, Z	2725	gggg	6								
			EXT	2735	hh ll	6								
CMPA	Compare A to Memory	(A) – (M)	IND8, X	48	ff	6	—	—	—	—	Δ	Δ	Δ	Δ
			IND8, Y	58	ff	6								
			IND8, Z	68	ff	6								
			IMM8	78	ii	2								
			IND16, X	1748	gggg	6								
			IND16, Y	1758	gggg	6								
			IND16, Z	1768	gggg	6								
			EXT	1778	hh ll	6								
			E, X	2748	—	6								
			E, Y	2758	—	6								
			E, Z	2768	—	6								
CMPB	Compare B to Memory	(B) – (M)	IND8, X	C8	ff	6	—	—	—	—	Δ	Δ	Δ	Δ
			IND8, Y	D8	ff	6								
			IND8, Z	E8	ff	6								
			IMM8	F8	ii	2								
			IND16, X	17C8	gggg	6								
			IND16, Y	17D8	gggg	6								
			IND16, Z	17E8	gggg	6								
			EXT	17F8	hh ll	6								
			E, X	27C8	—	6								
			E, Y	27D8	—	6								
			E, Z	27E8	—	6								
COM	One's Complement	$FF – (M) ⇒ M, or $\overline{M}$ ⇒ M	IND8, X	00	ff	8	—	—	—	—	Δ	Δ	0	1
			IND8, Y	10	ff	8								
			IND8, Z	20	ff	8								
			IND16, X	1700	gggg	8								
			IND16, Y	1710	gggg	8								
			IND16, Z	1720	gggg	8								
			EXT	1730	hh ll	8								
COMA	One's Complement A	$FF – (A) ⇒ A, or $\overline{A}$ ⇒ A	INH	3700	—	2	—	—	—	—	Δ	Δ	0	1
COMB	One's Complement B	$FF – (B) ⇒ B, or $\overline{B}$ ⇒ B	INH	3710	—	2	—	—	—	—	Δ	Δ	0	1
COMD	One's Complement D	$FFFF – (D) ⇒ D, or $\overline{D}$ ⇒ D	INH	27F0	—	2	—	—	—	—	Δ	Δ	0	1
COME	One's Complement E	$FFFF – (E) ⇒ E, or $\overline{E}$ ⇒ E	INH	2770	—	2	—	—	—	—	Δ	Δ	0	1
COMW	One's Complement Word	$FFFF – M : M + 1 ⇒ M : M + 1, or ($\overline{M : M + 1}$) ⇒ M : M + 1	IND16, X	2700	gggg	8	—	—	—	—	Δ	Δ	0	1
			IND16, Y	2710	gggg	8								
			IND16, Z	2720	gggg	8								
			EXT	2730	hh ll	8								
CPD	Compare D to Memory	(D) – (M : M + 1)	IND8, X	88	ff	6	—	—	—	—	Δ	Δ	Δ	Δ
			IND8, Y	98	ff	6								
			IND8, Z	A8	ff	6								
			IMM16	37B8	jj kk	4								
			IND16, X	37C8	gggg	6								
			IND16, Y	37D8	gggg	6								
			IND16, Z	37E8	gggg	6								
			EXT	37F8	hh ll	6								
			E, X	2788	—	6								
			E, Y	2798	—	6								
			E, Z	27A8	—	6								
CPE	Compare E to Memory	(E) – (M : M + 1)	IMM16	3738	jjkk	4	—	—	—	—	Δ	Δ	Δ	Δ
			IND16, X	3748	gggg	6								
			IND16, Y	3758	gggg	6								
			IND16, Z	3768	gggg	6								
			EXT	3778	hhll	6								
CPS	Compare Stack Pointer to Memory	(SP) – (M : M + 1)	IND8, X	4F	ff	6	—	—	—	—	Δ	Δ	Δ	Δ
			IND8, Y	5F	ff	6								
			IND8, Z	6F	ff	6								
			IMM16	377F	jj kk	4								
			IND16, X	174F	gggg	6								
			IND16, Y	175F	gggg	6								
			IND16, Z	176F	gggg	6								
			EXT	177F	hh ll	6								

Table 33 Instruction Set Summary (Continued)

Mnemonic	Operation	Description	Address Mode	Instruction Opcode	Instruction Operand	Instruction Cycles	Condition Codes S	MV	H	EV	N	Z	V	C
CPX	Compare IX to Memory	(IX) – (M : M + 1)	IND8, X	4C	ff	6	—	—	—	—	Δ	Δ	Δ	Δ
			IND8, Y	5C	ff	6								
			IND8, Z	6C	ff	6								
			IMM16	377C	jj kk	4								
			IND16, X	174C	gggg	6								
			IND16, Y	175C	gggg	6								
			IND16, Z	176C	gggg	6								
			EXT	177C	hh ll	6								
CPY	Compare IY to Memory	(IY) – (M : M + 1)	IND8, X	4D	ff	6	—	—	—	—	Δ	Δ	Δ	Δ
			IND8, Y	5D	ff	6								
			IND8, Z	6D	ff	6								
			IMM16	377D	jj kk	4								
			IND16, X	174D	gggg	6								
			IND16, Y	175D	gggg	6								
			IND16, Z	176D	gggg	6								
			EXT	177D	hh ll	6								
CPZ	Compare IZ to Memory	(IZ) – (M : M + 1)	IND8, X	4E	ff	6	—	—	—	—	Δ	Δ	Δ	Δ
			IND8, Y	5E	ff	6								
			IND8, Z	6E	ff	6								
			IMM16	377E	jj kk	4								
			IND16, X	174E	gggg	6								
			IND16, Y	175E	gggg	6								
			IND16, Z	176E	gggg	6								
			EXT	177E	hh ll	6								
DAA	Decimal Adjust A	$(A)_{10}$	INH	3721	—	2	—	—	—	—	Δ	Δ	U	Δ
DEC	Decrement Memory	(M) – $01 ⇒ M	IND8, X	01	ff	8	—	—	—	—	Δ	Δ	Δ	—
			IND8, Y	11	ff	8								
			IND8, Z	21	ff	8								
			IND16, X	1701	gggg	8								
			IND16, Y	1711	gggg	8								
			IND16, Z	1721	gggg	8								
			EXT	1731	hh ll	8								
DECA	Decrement A	(A) – $01 ⇒ A	INH	3701	—	2	—	—	—	—	Δ	Δ	Δ	—
DECB	Decrement B	(B) – $01 ⇒ B	INH	3711	—	2	—	—	—	—	Δ	Δ	Δ	—
DECW	Decrement Memory Word	(M : M + 1) – $0001 ⇒ M : M + 1	IND16, X	2701	gggg	8	—	—	—	—	Δ	Δ	Δ	—
			IND16, Y	2711	gggg	8								
			IND16, Z	2721	gggg	8								
			EXT	2731	hh ll	8								
EDIV	Extended Unsigned Integer Divide	(E : D) / (IX) Quotient ⇒ IX Remainder ⇒ D	INH	3728	—	24	—	—	—	—	Δ	Δ	Δ	Δ
EDIVS	Extended Signed Integer Divide	(E : D) / (IX) Quotient ⇒ IX Remainder ⇒ D	INH	3729	—	38	—	—	—	—	Δ	Δ	Δ	Δ
EMUL	Extended Unsigned Multiply	(E) ∗ (D) ⇒ E : D	INH	3725	—	10	—	—	—	—	Δ	Δ	—	Δ
EMULS	Extended Signed Multiply	(E) ∗ (D) ⇒ E : D	INH	3726	—	8	—	—	—	—	Δ	Δ	—	Δ
EORA	Exclusive OR A	(A) ⊕ (M) ⇒ A	IND8, X	44	ff	6	—	—	—	—	Δ	Δ	0	—
			IND8, Y	54	ff	6								
			IND8, Z	64	ff	6								
			IMM8	74	ii	2								
			IND16, X	1744	gggg	6								
			IND16, Y	1754	gggg	6								
			IND16, Z	1764	gggg	6								
			EXT	1774	hh ll	6								
			E, X	2744	—	6								
			E, Y	2754	—	6								
			E, Z	2764	—	6								

Table 33 Instruction Set Summary (Continued)

Mnemonic	Operation	Description	Address Mode	Instruction Opcode	Operand	Cycles	Condition Codes S	MV	H	EV	N	Z	V	C
EORB	Exclusive OR B	(B) ⊕ (M) ⇒ B	IND8, X	C4	ff	6	—	—	—	—	Δ	Δ	0	—
			IND8, Y	D4	ff	6								
			IND8, Z	E4	ff	6								
			IMM8	F4	ii	2								
			IND16, X	17C4	gggg	6								
			IND16, Y	17D4	gggg	6								
			IND16, Z	17E4	gggg	6								
			EXT	17F4	hh ll	6								
			E, X	27C4	—	6								
			E, Y	27D4	—	6								
			E, Z	27E4	—	6								
EORD	Exclusive OR D	(D) ⊕ (M : M + 1) ⇒ D	IND8, X	84	ff	6	—	—	—	—	Δ	Δ	0	—
			IND8, Y	94	ff	6								
			IND8, Z	A4	ff	6								
			IMM16	37B4	jj kk	4								
			IND16, X	37C4	gggg	6								
			IND16, Y	37D4	gggg	6								
			IND16, Z	37E4	gggg	6								
			EXT	37F4	hh ll	6								
			E, X	2784	—	6								
			E, Y	2794	—	6								
			E, Z	27A4	—	6								
EORE	Exclusive OR E	(E) ⊕ (M : M + 1) ⇒ E	IMM16	3734	jj kk	4	—	—	—	—	Δ	Δ	0	—
			IND16, X	3744	gggg	6								
			IND16, Y	3754	gggg	6								
			IND16, Z	3764	gggg	6								
			EXT	3774	hh ll	6								
FDIV	Fractional Unsigned Divide	(D) / (IX) ⇒ IX Remainder ⇒ D	INH	372B	—	22	—	—	—	—	—	Δ	Δ	Δ
FMULS	Fractional Signed Multiply	(E) ∗ (D) ⇒ E : D[31:1] 0 ⇒ D[0]	INH	3727	—	8	—	—	—	—	Δ	Δ	Δ	Δ
IDIV	Integer Divide	(D) / (IX) ⇒ IX Remainder ⇒ D	INH	372A	—	22	—	—	—	—	—	Δ	0	Δ
INC	Increment Memory	(M) + $01 ⇒ M	IND8, X	03	ff	8	—	—	—	—	Δ	Δ	Δ	—
			IND8, Y	13	ff	8								
			IND8, Z	23	ff	8								
			IND16, X	1703	gggg	8								
			IND16, Y	1713	gggg	8								
			IND16, Z	1723	gggg	8								
			EXT	1733	hh ll	8								
INCA	Increment A	(A) + $01 ⇒ A	INH	3703	—	2	—	—	—	—	Δ	Δ	Δ	—
INCB	Increment B	(B) + $01 ⇒ B	INH	3713	—	2	—	—	—	—	Δ	Δ	Δ	—
INCW	Increment Memory Word	(M : M + 1) + $0001 ⇒ M : M + 1	IND16, X	2703	gggg	8	—	—	—	—	Δ	Δ	Δ	—
			IND16, Y	2713	gggg	8								
			IND16, Z	2723	gggg	8								
			EXT	2733	hh ll	8								
JMP	Jump	⟨ea⟩ ⇒ PK : PC	EXT20	7A	zb hh ll	6	—	—	—	—	—	—	—	—
			IND20, X	4B	zg gggg	8								
			IND20, Y	5B	zg gggg	8								
			IND20, Z	6B	zg gggg	8								
JSR	Jump to Subroutine	Push (PC) (SK : SP) – $0002 ⇒ SK : SP Push (CCR) (SK : SP) – $0002 ⇒ SK : SP ⟨ea⟩ ⇒ PK : PC	EXT20	FA	zb hh ll	10	—	—	—	—	—	—	—	—
			IND20, X	89	zg gggg	12								
			IND20, Y	99	zg gggg	12								
			IND20, Z	A9	zg gggg	12								
LBCC[2]	Long Branch if Carry Clear	If C = 0, branch	REL16	3784	rrrr	6, 4	—	—	—	—	—	—	—	—
LBCS[2]	Long Branch if Carry Set	If C = 1, branch	REL16	3785	rrrr	6, 4	—	—	—	—	—	—	—	—
LBEQ[2]	Long Branch if Equal to Zero	If Z = 1, branch	REL16	3787	rrrr	6, 4	—	—	—	—	—	—	—	—
LBEV[2]	Long Branch if EV Set	If EV = 1, branch	REL16	3791	rrrr	6, 4	—	—	—	—	—	—	—	—
LBGE[2]	Long Branch if Greater Than or Equal to Zero	If N ⊕ V = 0, branch	REL16	378C	rrrr	6, 4	—	—	—	—	—	—	—	—
LBGT[2]	Long Branch if Greater Than Zero	If Z ✣ (N ⊕ V) = 0, branch	REL16	378E	rrrr	6, 4	—	—	—	—	—	—	—	—
LBHI [2]	Long Branch if Higher	If C ✣ Z = 0, branch	REL16	3782	rrrr	6, 4	—	—	—	—	—	—	—	—

Table 33 Instruction Set Summary (Continued)

Mnemonic	Operation	Description	Address Mode	Instruction Opcode	Instruction Operand	Instruction Cycles	Condition Codes S	MV	H	EV	N	Z	V	C
LBLE[2]	Long Branch if Less Than or Equal to Zero	If Z ✣ (N ⊕ V) = 1, branch	REL16	378F	rrrr	6, 4	—	—	—	—	—	—	—	—
LBLS[2]	Long Branch if Lower or Same	If C ✣ Z = 1, branch	REL16	3783	rrrr	6, 4	—	—	—	—	—	—	—	—
LBLT[2]	Long Branch if Less Than Zero	If N ⊕ V = 1, branch	REL16	378D	rrrr	6, 4	—	—	—	—	—	—	—	—
LBMI[2]	Long Branch if Minus	If N = 1, branch	REL16	378B	rrrr	6, 4	—	—	—	—	—	—	—	—
LBMV[2]	Long Branch if MV Set	If MV = 1, branch	REL16	3790	rrrr	6, 4	—	—	—	—	—	—	—	—
LBNE[2]	Long Branch if Not Equal to Zero	If Z = 0, branch	REL16	3786	rrrr	6, 4	—	—	—	—	—	—	—	—
LBPL[2]	Long Branch if Plus	If N = 0, branch	REL16	378A	rrrr	6, 4	—	—	—	—	—	—	—	—
LBRA	Long Branch Always	If 1 = 1, branch	REL16	3780	rrrr	6	—	—	—	—	—	—	—	—
LBRN	Long Branch Never	If 1 = 0, branch	REL16	3781	rrrr	6	—	—	—	—	—	—	—	—
LBSR	Long Branch to Subroutine	Push (PC) (SK : SP) – 2 ⇒ SK : SP Push (CCR) (SK : SP) – 2 ⇒ SK : SP (PK : PC) + Offset ⇒ PK : PC	REL16	27F9	rrrr	10	—	—	—	—	—	—	—	—
LBVC[2]	Long Branch if Overflow Clear	If V = 0, branch	REL16	3788	rrrr	6, 4	—	—	—	—	—	—	—	—
LBVS[2]	Long Branch if Overflow Set	If V = 1, branch	REL16	3789	rrrr	6, 4	—	—	—	—	—	—	—	—
LDAA	Load A	(M) ⇒ A	IND8, X	45	ff	6	—	—	—	—	Δ	Δ	0	—
			IND8, Y	55	ff	6								
			IND8, Z	65	ff	6								
			IMM8	75	ii	2								
			IND16, X	1745	gggg	6								
			IND16, Y	1755	gggg	6								
			IND16, Z	1765	gggg	6								
			EXT	1775	hh ll	6								
			E, X	2745	—	6								
			E, Y	2755	—	6								
			E, Z	2765	—	6								
LDAB	Load B	(M) ⇒ B	IND8, X	C5	ff	6	—	—	—	—	Δ	Δ	0	Δ
			IND8, Y	D5	ff	6								
			IND8, Z	E5	ff	6								
			IMM8	F5	ii	2								
			IND16, X	17C5	gggg	6								
			IND16, Y	17D5	gggg	6								
			IND16, Z	17E5	gggg	6								
			EXT	17F5	hh ll	6								
			E, X	27C5	—	6								
			E, Y	27D5	—	6								
			E, Z	27E5	—	6								
LDD	Load D	(M : M + 1) ⇒ D	IND8, X	85	ff	6	—	—	—	—	Δ	Δ	0	—
			IND8, Y	95	ff	6								
			IND8, Z	A5	ff	6								
			IMM16	37B5	jj kk	4								
			IND16, X	37C5	gggg	6								
			IND16, Y	37D5	gggg	6								
			IND16, Z	37E5	gggg	6								
			EXT	37F5	hh ll	6								
			E, X	2785	—	6								
			E, Y	2795	—	6								
			E, Z	27A5	—	6								
LDE	Load E	(M : M + 1) ⇒ E	IMM16	3735	jj kk	4	—	—	—	—	Δ	Δ	0	—
			IND16, X	3745	gggg	6								
			IND16, Y	3755	gggg	6								
			IND16, Z	3765	gggg	6								
			EXT	3775	hh ll	6								
LDED	Load Concatenated E and D	(M : M + 1) ⇒ E (M + 2 : M + 3) ⇒ D	EXT	2771	hh ll	8	—	—	—	—	—	—	—	—
LDHI	Initialize H and I	$(M : M + 1)_X$ ⇒ H R $(M : M + 1)_Y$ ⇒ I R	EXT	27B0	—	8	—	—	—	—	—	—	—	—

Table 33 Instruction Set Summary (Continued)

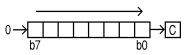
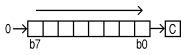
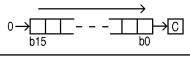
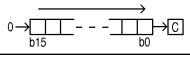
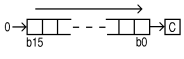
Mnemonic	Operation	Description	Address Mode	Instruction Opcode	Instruction Operand	Instruction Cycles	Condition Codes S	MV	H	EV	N	Z	V	C
LDS	Load SP	(M : M + 1) ⇒ SP	IND8, X	CF	ff	6	—	—	—	—	Δ	Δ	0	—
			IND8, Y	DF	ff	6								
			IND8, Z	EF	ff	6								
			IND16, X	17CF	gggg	6								
			IND16, Y	17DF	gggg	6								
			IND16, Z	17EF	gggg	6								
			EXT	17FF	hh ll	6								
			IMM16	37BF	jj kk	4								
LDX	Load IX	(M : M + 1) ⇒ IX	IND8, X	CC	ff	6	—	—	—	—	Δ	Δ	0	—
			IND8, Y	DC	ff	6								
			IND8, Z	EC	ff	6								
			IMM16	37BC	jj kk	4								
			IND16, X	17CC	gggg	6								
			IND16, Y	17DC	gggg	6								
			IND16, Z	17EC	gggg	6								
			EXT	17FC	hh ll	6								
LDY	Load IY	(M : M + 1) ⇒ IY	IND8, X	CD	ff	6	—	—	—	—	Δ	Δ	0	—
			IND8, Y	DD	ff	6								
			IND8, Z	ED	ff	6								
			IMM16	37BD	jj kk	4								
			IND16, X	17CD	gggg	6								
			IND16, Y	17DD	gggg	6								
			IND16, Z	17ED	gggg	6								
			EXT	17FD	hh ll	6								
LDZ	Load IZ	(M : M + 1) ⇒ IZ	IND8, X	CE	ff	6	—	—	—	—	Δ	Δ	0	—
			IND8, Y	DE	ff	6								
			IND8, Z	EE	ff	6								
			IMM16	37BE	jj kk	4								
			IND16, X	17CE	gggg	6								
			IND16, Y	17DE	gggg	6								
			IND16, Z	17EE	gggg	6								
			EXT	17FE	hh ll	6								
LPSTOP	Low Power Stop	If $\overline{S}$ then STOP else NOP	INH	27F1	—	4, 20	—	—	—	—	—	—	—	—
LSR	Logical Shift Right		IND8, X	0F	ff	8	—	—	—	—	0	Δ	Δ	Δ
			IND8, Y	1F	ff	8								
			IND8, Z	2F	ff	8								
			IND16, X	170F	gggg	8								
			IND16, Y	171F	gggg	8								
			IND16, Z	172F	gggg	8								
			EXT	173F	hh ll	8								
LSRA	Logical Shift Right A		INH	370F	—	2	—	—	—	—	0	Δ	Δ	Δ
LSRB	Logical Shift Right B		INH	371F	—	2	—	—	—	—	0	Δ	Δ	Δ
LSRD	Logical Shift Right D		INH	27FF	—	2	—	—	—	—	0	Δ	Δ	Δ
LSRE	Logical Shift Right E		INH	277F	—	2	—	—	—	—	0	Δ	Δ	Δ
LSRW	Logical Shift Right Word		IND16, X	270F	gggg	8	—	—	—	—	0	Δ	Δ	Δ
			IND16, Y	271F	gggg	8								
			IND16, Z	272F	gggg	8								
			EXT	273F	hh ll	8								
MAC	Multiply and Accumulate Signed 16-Bit Fractions	(HR) * (IR) ⇒ E : D (AM) + (E : D) ⇒ AM Qualified (IX) ⇒ IX Qualified (IY) ⇒ IY (HR) ⇒ IZ $(M : M + 1)_X$ ⇒ HR $(M : M + 1)_Y$ ⇒ IR	IMM8	7B	xoyo	12	—	Δ	—	Δ	—	—	Δ	—

Table 33 Instruction Set Summary (Continued)

Mnemonic	Operation	Description	Address Mode	Instruction Opcode	Operand	Cycles	Condition Codes S	MV	H	EV	N	Z	V	C
MOVB	Move Byte	$(M_1) \Rightarrow M_2$	IXP to EXT	30	ff hh ll	8	—	—	—	—	Δ	Δ	0	—
			EXT to IXP	32	ff hh ll	8								
			EXT to EXT	37FE	hh ll hh ll	10								
MOVW	Move Word	$(M : M + 1_1) \Rightarrow M : M + 1_2$	IXP to EXT	31	ff hh ll	8	—	—	—	—	Δ	Δ	0	—
			EXT to IXP	33	ff hh ll	8								
			EXT to EXT	37FF	hh ll hh ll	10								
MUL	Multiply	(A) ∗ (B) ⇒ D	INH	3724	—	10	—	—	—	—	—	—	—	Δ
NEG	Negate Memory	$00 – (M) ⇒ M	IND8, X	02	ff	8	—	—	—	—	Δ	Δ	Δ	Δ
			IND8, Y	12	ff	8								
			IND8, Z	22	ff	8								
			IND16, X	1702	gggg	8								
			IND16, Y	1712	gggg	8								
			IND16, Z	1722	gggg	8								
			EXT	1732	hh ll	8								
NEGA	Negate A	$00 – (A) ⇒ A	INH	3702	—	2	—	—	—	—	Δ	Δ	Δ	Δ
NEGB	Negate B	$00 – (B) ⇒ B	INH	3712	—	2	—	—	—	—	Δ	Δ	Δ	Δ
NEGD	Negate D	$0000 – (D) ⇒ D	INH	27F2	—	2	—	—	—	—	Δ	Δ	Δ	Δ
NEGE	Negate E	$0000 – (E) ⇒ E	INH	2772	—	2	—	—	—	—	Δ	Δ	Δ	Δ
NEGW	Negate Memory Word	$0000 – (M : M + 1) ⇒ M : M + 1	IND16, X	2702	gggg	8	—	—	—	—	Δ	Δ	Δ	Δ
			IND16, Y	2712	gggg	8								
			IND16, Z	2722	gggg	8								
			EXT	2732	hh ll	8								
NOP	Null Operation	—	INH	274C	—	2	—	—	—	—	—	—	—	—
ORAA	OR A	(A) ✣ (M) ⇒ A	IND8, X	47	ff	6	—	—	—	—	Δ	Δ	0	—
			IND8, Y	57	ff	6								
			IND8, Z	67	ff	6								
			IMM8	77	ii	2								
			IND16, X	1747	gggg	6								
			IND16, Y	1757	gggg	6								
			IND16, Z	1767	gggg	6								
			EXT	1777	hh ll	6								
			E, X	2747	—	6								
			E, Y	2757	—	6								
			E, Z	2767	—	6								
ORAB	OR B	(B) ✣ (M) ⇒ B	IND8, X	C7	ff	6	—	—	—	—	Δ	Δ	0	—
			IND8, Y	D7	ff	6								
			IND8, Z	E7	ff	6								
			IMM8	F7	ii	2								
			IND16, X	17C7	gggg	6								
			IND16, Y	17D7	gggg	6								
			IND16, Z	17E7	gggg	6								
			EXT	17F7	hh ll	6								
			E, X	27C7	—	6								
			E, Y	27D7	—	6								
			E, Z	27E7	—	6								
ORD	OR D	(D) ✣ (M : M + 1) ⇒ D	IND8, X	87	ff	6	—	—	—	—	Δ	Δ	0	—
			IND8, Y	97	ff	6								
			IND8, Z	A7	ff	6								
			IMM16	37B7	jj kk	4								
			IND16, X	37C7	gggg	6								
			IND16, Y	37D7	gggg	6								
			IND16, Z	37E7	gggg	6								
			EXT	37F7	hh ll	6								
			E, X	2787	—	6								
			E, Y	2797	—	6								
			E, Z	27A7	—	6								
ORE	OR E	(E) ✣ (M : M + 1) ⇒ E	IMM16	3737	jj kk	4	—	—	—	—	Δ	Δ	0	—
			IND16, X	3747	gggg	6								
			IND16, Y	3757	gggg	6								
			IND16, Z	3767	gggg	6								
			EXT	3777	hh ll	6								
ORP [1]	OR Condition Code Register	(CCR) ✣ IMM16 ⇒ CCR	IMM16	373B	jj kk	4	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ
PSHA	Push A	(SK : SP) + $0001 ⇒ SK : SP Push (A) (SK : SP) – $0002 ⇒ SK : SP	INH	3708	—	4	—	—	—	—	—	—	—	—

Table 33 Instruction Set Summary (Continued)

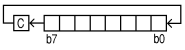
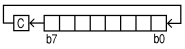
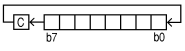
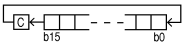
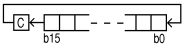
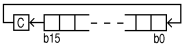
Mnemonic	Operation	Description	Address Mode	Instruction Opcode	Instruction Operand	Instruction Cycles	Condition Codes S	MV	H	EV	N	Z	V	C
PSHB	Push B	(SK : SP) + $0001 ⇒ SK : SP Push (B) (SK : SP) – $0002 ⇒ SK : SP	INH	3718	—	4	—	—	—	—	—	—	—	—
PSHM	Push Multiple Registers Mask bits: 0 = D 1 = E 2 = IX 3 = IY 4 = IZ 5 = K 6 = CCR 7 = (Reserved)	For mask bits 0 to 7: If mask bit set Push register (SK : SP) – 2 ⇒ SK : SP	IMM8	34	ii	4 + 2N N = number of iterations	—	—	—	—	—	—	—	—
PSHMAC	Push MAC Registers	MAC Registers ⇒ Stack	INH	27B8	—	14	—	—	—	—	—	—	—	—
PULA	Pull A	(SK : SP) + $0002 ⇒ SK : SP Pull (A) (SK : SP) – $0001 ⇒ SK : SP	INH	3709	—	6	—	—	—	—	—	—	—	—
PULB	Pull B	(SK : SP) + $0002 ⇒ SK : SP Pull (B) (SK : SP) – $0001 ⇒ SK : SP	INH	3719	—	6	—	—	—	—	—	—	—	—
PULM[1]	Pull Multiple Registers Mask bits: 0 = CCR[15:4] 1 = K 2 = IZ 3 = IY 4 = IX 5 = E 6 = D 7 = (Reserved)	For mask bits 0 to 7: If mask bit set (SK : SP) + 2 ⇒ SK : SP Pull register	IMM8	35	ii	4+2(N+1) N = number of iterations	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ
PULMAC	Pull MAC State	Stack ⇒ MAC Registers	INH	27B9	—	16	—	—	—	—	—	—	—	—
RMAC	Repeating Multiply and Accumulate Signed 16-Bit Fractions	Repeat until (E) < 0 (AM) + (H) ∗ (I) ⇒ AM Qualified (IX) ⇒ IX; Qualified (IY) ⇒ IY; $(M : M + 1)_X$ ⇒ H; $(M : M + 1)_Y$ ⇒ I (E) – 1 ⇒ E Until (E) < $0000	IMM8	FB	xoyo	6 + 12 per iteration	—	Δ	—	Δ	—	—	—	—
ROL	Rotate Left		IND8, X IND8, Y IND8, Z IND16, X IND16, Y IND16, Z EXT	0C 1C 2C 170C 171C 172C 173C	ff ff ff gggg gggg gggg hh ll	8 8 8 8 8 8 8	—	—	—	—	Δ	Δ	Δ	Δ
ROLA	Rotate Left A		INH	370C	—	2	—	—	—	—	Δ	Δ	Δ	Δ
ROLB	Rotate Left B		INH	371C	—	2	—	—	—	—	Δ	Δ	Δ	Δ
ROLD	Rotate Left D		INH	27FC	—	2	—	—	—	—	Δ	Δ	Δ	Δ
ROLE	Rotate Left E		INH	277C	—	2	—	—	—	—	Δ	Δ	Δ	Δ
ROLW	Rotate Left Word		IND16, X IND16, Y IND16, Z EXT	270C 271C 272C 273C	gggg gggg gggg hh ll	8 8 8 8	—	—	—	—	Δ	Δ	Δ	Δ

Table 33 Instruction Set Summary (Continued)

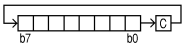
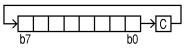
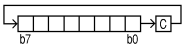
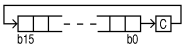
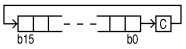
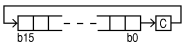
Mnemonic	Operation	Description	Address Mode	Instruction Opcode	Instruction Operand	Instruction Cycles	Condition Codes S	MV	H	EV	N	Z	V	C
ROR	Rotate Right Byte		IND8, X	0E	ff	8	—	—	—	—	Δ	Δ	Δ	Δ
			IND8, Y	1E	ff	8								
			IND8, Z	2E	ff	8								
			IND16, X	170E	gggg	8								
			IND16, Y	171E	gggg	8								
			IND16, Z	172E	gggg	8								
			EXT	173E	hh ll	8								
RORA	Rotate Right A		INH	370E	—	2	—	—	—	—	Δ	Δ	Δ	Δ
RORB	Rotate Right B		INH	371E	—	2	—	—	—	—	Δ	Δ	Δ	Δ
RORD	Rotate Right D		INH	27FE	—	2	—	—	—	—	Δ	Δ	Δ	Δ
RORE	Rotate Right E		INH	277E	—	2	—	—	—	—	Δ	Δ	Δ	Δ
RORW	Rotate Right Word		IND16, X	270E	gggg	8	—	—	—	—	Δ	Δ	Δ	Δ
			IND16, Y	271E	gggg	8								
			IND16, Z	272E	gggg	8								
			EXT	273E	hh ll	8								
RTI[3]	Return from Interrupt	(SK : SP) + 2 ⇒ SK : SP Pull CCR (SK : SP) + 2 ⇒ SK : SP Pull PC (PK : PC) – 6 ⇒ PK : PC	INH	2777	—	12	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ
RTS[4]	Return from Subroutine	(SK : SP) + 2 ⇒ SK : SP Pull PK (SK : SP) + 2 ⇒ SK : SP Pull PC (PK : PC) – 2 ⇒ PK : PC	INH	27F7	—	12	—	—	—	—	—	—	—	—
SBA	Subtract B from A	(A) – (B) ⇒ A	INH	370A	—	2	—	—	—	—	Δ	Δ	Δ	Δ
SBCA	Subtract with Carry from A	(A) – (M) – C ⇒ A	IND8, X	42	ff	6	—	—	—	—	Δ	Δ	Δ	Δ
			IND8, Y	52	ff	6								
			IND8, Z	62	ff	6								
			IMM8	72	ii	2								
			IND16, X	1742	gggg	6								
			IND16, Y	1752	gggg	6								
			IND16, Z	1762	gggg	6								
			EXT	1772	hh ll	6								
			E, X	2742	—	6								
			E, Y	2752	—	6								
			E, Z	2762	—	6								
SBCB	Subtract with Carry from B	(B) – (M) – C ⇒ B	IND8, X	C2	ff	6	—	—	—	—	Δ	Δ	Δ	Δ
			IND8, Y	D2	ff	6								
			IND8, Z	E2	ff	6								
			IMM8	F2	ii	2								
			IND16, X	17C2	gggg	6								
			IND16, Y	17D2	gggg	6								
			IND16, Z	17E2	gggg	6								
			EXT	17F2	hh ll	6								
			E, X	27C2	—	6								
			E, Y	27D2	—	6								
			E, Z	27E2	—	6								
SBCD	Subtract with Carry from D	(D) – (M : M + 1) – C ⇒ D	IND8, X	82	ff	6	—	—	—	—	Δ	Δ	Δ	Δ
			IND8, Y	92	ff	6								
			IND8, Z	A2	ff	6								
			IMM16	37B2	jj kk	4								
			IND16, X	37C2	gggg	6								
			IND16, Y	37D2	gggg	6								
			IND16, Z	37E2	gggg	6								
			EXT	37F2	hh ll	6								
			E, X	2782	—	6								
			E, Y	2792	—	6								
			E, Z	27A2	—	6								

Table 33 Instruction Set Summary (Continued)

Mnemonic	Operation	Description	Address Mode	Instruction Opcode	Instruction Operand	Instruction Cycles	Condition Codes S	MV	H	EV	N	Z	V	C
SBCE	Subtract with Carry from E	(E) – (M : M + 1) – C ⇒ E	IMM16	3732	jj kk	4	—	—	—	—	Δ	Δ	Δ	Δ
			IND16, X	3742	gggg	6								
			IND16, Y	3752	gggg	6								
			IND16, Z	3762	gggg	6								
			EXT	3772	hh ll	6								
SDE	Subtract D from E	(E) – (D)⇒ E	INH	2779	—	2	—	—	—	—	Δ	Δ	Δ	Δ
STAA	Store A	(A) ⇒ M	IND8, X	4A	ff	4	—	—	—	—	Δ	Δ	0	—
			IND8, Y	5A	ff	4								
			IND8, Z	6A	ff	4								
			IND16, X	174A	gggg	6								
			IND16, Y	175A	gggg	6								
			IND16, Z	176A	gggg	6								
			EXT	177A	hh ll	6								
			E, X	274A	—	4								
			E, Y	275A	—	4								
			E, Z	276A	—	4								
STAB	Store B	(B) ⇒ M	IND8, X	CA	ff	4	—	—	—	—	Δ	Δ	0	—
			IND8, Y	DA	ff	4								
			IND8, Z	EA	ff	4								
			IND16, X	17CA	gggg	6								
			IND16, Y	17DA	gggg	6								
			IND16, Z	17EA	gggg	6								
			EXT	17FA	hh ll	6								
			E, X	27CA	—	4								
			E, Y	27DA	—	4								
			E, Z	27EA	—	4								
STD	Store D	(D) ⇒ M : M + 1	IND8, X	8A	ff	4	—	—	—	—	Δ	Δ	0	—
			IND8, Y	9A	ff	4								
			IND8, Z	AA	ff	4								
			IND16, X	37CA	gggg	6								
			IND16, Y	37DA	gggg	6								
			IND16, Z	37EA	gggg	6								
			EXT	37FA	hh ll	6								
			E, X	278A	—	6								
			E, Y	279A	—	6								
			E, Z	27AA	—	6								
STE	Store E	(E) ⇒ M : M + 1	IND16, X	374A	gggg	6	—	—	—	—	Δ	Δ	0	—
			IND16, Y	375A	gggg	6								
			IND16, Z	376A	gggg	6								
			EXT	377A	hh ll	6								
STED	Store Concatenated D and E	(E) ⇒ M : M + 1 (D) ⇒ M + 2 : M + 3	EXT	2773	hh ll	8	—	—	—	—	—	—	—	—
STS	Store Stack Pointer	(SP) ⇒ M : M + 1	IND8, X	8F	ff	4	—	—	—	—	Δ	Δ	0	—
			IND8, Y	9F	ff	4								
			IND8, Z	AF	ff	4								
			IND16, X	178F	gggg	6								
			IND16, Y	179F	gggg	6								
			IND16, Z	17AF	gggg	6								
			EXT	17BF	hh ll	6								
STX	Store IX	(IX) ⇒ M : M + 1	IND8, X	8C	ff	4	—	—	—	—	Δ	Δ	0	—
			IND8, Y	9C	ff	4								
			IND8, Z	AC	ff	4								
			IND16, X	178C	gggg	6								
			IND16, Y	179C	gggg	6								
			IND16, Z	17AC	gggg	6								
			EXT	17BC	hh ll	6								
STY	Store IY	(IY) ⇒ M : M + 1	IND8, X	8D	ff	4	—	—	—	—	Δ	Δ	0	—
			IND8, Y	9D	ff	4								
			IND8, Z	AD	ff	4								
			IND16, X	178D	gggg	6								
			IND16, Y	179D	gggg	6								
			IND16, Z	17AD	gggg	6								
			EXT	17BD	hh ll	6								

Table 33 Instruction Set Summary (Continued)

Mnemonic	Operation	Description	Address Mode	Instruction Opcode	Operand	Cycles	Condition Codes S	MV	H	EV	N	Z	V	C
STZ	Store Z	(IZ) ⇒ M : M + 1	IND8, X	8E	ff	4	—	—	—	—	Δ	Δ	0	—
			IND8, Y	9E	ff	4								
			IND8, Z	AE	ff	4								
			IND16, X	178E	gggg	6								
			IND16, Y	179E	gggg	6								
			IND16, Z	17AE	gggg	6								
			EXT	17BE	hh ll	6								
SUBA	Subtract from A	(A) – (M) ⇒ A	IND8, X	40	ff	6	—	—	—	—	Δ	Δ	Δ	Δ
			IND8, Y	50	ff	6								
			IND8, Z	60	ff	6								
			IMM8	70	ii	2								
			IND16, X	1740	gggg	6								
			IND16, Y	1750	gggg	6								
			IND16, Z	1760	gggg	6								
			EXT	1770	hh ll	6								
			E, X	2740	—	6								
			E, Y	2750	—	6								
			E, Z	2760	—	6								
SUBB	Subtract from B	(B) – (M) ⇒ B	IND8, X	C0	ff	6	—	—	—	—	Δ	Δ	Δ	Δ
			IND8, Y	D0	ff	6								
			IND8, Z	E0	ff	6								
			IMM8	F0	ii	2								
			IND16, X	17C0	gggg	6								
			IND16, Y	17D0	gggg	6								
			IND16, Z	17E0	gggg	6								
			EXT	17F0	hh ll	6								
			E, X	27C0	—	6								
			E, Y	27D0	—	6								
			E, Z	27E0	—	6								
SUBD	Subtract from D	(D) – (M : M + 1) ⇒ D	IND8, X	80	ff	6	—	—	—	—	Δ	Δ	Δ	Δ
			IND8, Y	90	ff	6								
			IND8, Z	A0	ff	6								
			IMM16	37B0	jj kk	4								
			IND16, X	37C0	gggg	6								
			IND16, Y	37D0	gggg	6								
			IND16, Z	37E0	gggg	6								
			EXT	37F0	hh ll	6								
			E, X	2780	—	6								
			E, Y	2790	—	6								
			E, Z	27A0	—	6								
SUBE	Subtract from E	(E) – (M : M + 1) ⇒ E	IMM16	3730	jj kk	4	—	—	—	—	Δ	Δ	Δ	Δ
			IND16, X	3740	gggg	6								
			IND16, Y	3750	gggg	6								
			IND16, Z	3760	gggg	6								
			EXT	3770	hh ll	6								
SWI	Software Interrupt	(PK : PC) + $0002 ⇒ PK : PC Push (PC) (SK : SP) – $0002 ⇒ SK : SP Push (CCR) (SK : SP) – $0002 ⇒ SK : SP $0 ⇒ PK SWI Vector ⇒ PC	INH	3720	—	16	—	—	—	—	—	—	—	—
SXT	Sign Extend B into A	If B7 = 1 then $FF ⇒ A else $00 ⇒ A	INH	27F8	—	2	—	—	—	—	Δ	Δ	—	—
TAB	Transfer A to B	(A) ⇒ B	INH	3717	—	2	—	—	—	—	Δ	Δ	0	—
TAP	Transfer A to CCR	(A[7:0]) ⇒ CCR[15:8]	INH	37FD	—	4	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ
TBA	Transfer B to A	(B) ⇒ A	INH	3707	—	2	—	—	—	—	Δ	Δ	0	—
TBEK	Transfer B to EK	(B[3:0]) ⇒ EK	INH	27FA	—	2	—	—	—	—	—	—	—	—
TBSK	Transfer B to SK	(B[3:0]) ⇒ SK	INH	379F	—	2	—	—	—	—	—	—	—	—
TBXK	Transfer B to XK	(B[3:0]) ⇒ XK	INH	379C	—	2	—	—	—	—	—	—	—	—
TBYK	Transfer B to YK	(B[3:0]) ⇒ YK	INH	379D	—	2	—	—	—	—	—	—	—	—
TBZK	Transfer B to ZK	(B[3:0]) ⇒ ZK	INH	379E	—	2	—	—	—	—	—	—	—	—
TDE	Transfer D to E	(D) ⇒ E	INH	277B	—	2	—	—	—	—	Δ	Δ	0	—
TDMSK	Transfer D to XMSK : YMSK	(D[15:8]) ⇒ X MASK (D[7:0]) ⇒ Y MASK	INH	372F	—	2	—	—	—	—	—	—	—	—
TDP[1]	Transfer D to CCR	(D) ⇒ CCR[15:4]	INH	372D	—	4	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ
TED	Transfer E to D	(E) ⇒ D	INH	27FB	—	2	—	—	—	—	Δ	Δ	0	—

Table 33 Instruction Set Summary (Continued)

Mnemonic	Operation	Description	Address Mode	Instruction Opcode	Instruction Operand	Instruction Cycles	Condition Codes S	MV	H	EV	N	Z	V	C
TEDM	Transfer E and D to AM[31:0] Sign Extend AM	(E) ⇒ AM[31:16] (D) ⇒ AM[15:0] AM[35:32] = AM31	INH	27B1	—	4	—	0	—	0	—	—	—	—
TEKB	Transfer EK to B	(EK) ⇒ B[3:0] $0 ⇒ B[7:4]	INH	27BB	—	2	—	—	—	—	—	—	—	—
TEM	Transfer E to AM[31:16] Sign Extend AM Clear AM LSB	(E) ⇒ AM[31:16] $00 ⇒ AM[15:0] AM[35:32] = AM31	INH	27B2	—	4	—	0	—	0	—	—	—	—
TMER	Transfer Rounded AM to E	Rounded (AM) ⇒ Temp If (SM • (EV ✣ MV)) then Saturation Value ⇒ E else Temp[31:16] ⇒ E	INH	27B4	—	6	—	Δ	—	Δ	Δ	Δ	—	—
TMET	Transfer Truncated AM to E	If (SM • (EV ✣ MV)) then Saturation Value ⇒ E else AM[31:16] ⇒ E	INH	27B5	—	2	—	—	—	—	Δ	Δ	—	—
TMXED	Transfer AM to IX : E : D	AM[35:32] ⇒ IX[3:0] AM35 ⇒ IX[15:4] AM[31:16] ⇒ E AM[15:0] ⇒ D	INH	27B3	—	6	—	—	—	—	—	—	—	—
TPA	Transfer CCR to A	(CCR[15:8]) ⇒ A	INH	37FC	—	2	—	—	—	—	—	—	—	—
TPD	Transfer CCR to D	(CCR) ⇒ D	INH	372C	—	2	—	—	—	—	—	—	—	—
TSKB	Transfer SK to B	(SK) ⇒ B[3:0] $0 ⇒ B[7:4]	INH	37AF	—	2	—	—	—	—	—	—	—	—
TST	Test Byte Zero or Minus	(M) – $00	IND8, X IND8, Y IND8, Z IND16, X IND16, Y IND16, Z EXT	06 16 26 1706 1716 1726 1736	ff ff ff gggg gggg gggg hh ll	6 6 6 6 6 6 6	—	—	—	—	Δ	Δ	0	0
TSTA	Test A for Zero or Minus	(A) – $00	INH	3706	—	2	—	—	—	—	Δ	Δ	0	0
TSTB	Test B for Zero or Minus	(B) – $00	INH	3716	—	2	—	—	—	—	Δ	Δ	0	0
TSTD	Test D for Zero or Minus	(D) – $0000	INH	27F6	—	2	—	—	—	—	Δ	Δ	0	0
TSTE	Test E for Zero or Minus	(E) – $0000	INH	2776	—	2	—	—	—	—	Δ	Δ	0	0
TSTW	Test for Zero or Minus Word	(M : M + 1) – $0000	IND16, X IND16, Y IND16, Z EXT	2706 2716 2726 2736	gggg gggg gggg hh ll	6 6 6 6	—	—	—	—	Δ	Δ	0	0
TSX	Transfer SP to IX	(SK : SP) + $0002 ⇒ XK : IX	INH	274F	—	2	—	—	—	—	—	—	—	—
TSY	Transfer SP to IY	(SK : SP) + $0002 ⇒ YK : IY	INH	275F	—	2	—	—	—	—	—	—	—	—
TSZ	Transfer SP to IZ	(SK : SP) + $0002 ⇒ ZK : IZ	INH	276F	—	2	—	—	—	—	—	—	—	—
TXKB	Transfer XK to B	(XK) ⇒ B[3:0] $0 ⇒ B[7:4]	INH	37AC	—	2	—	—	—	—	—	—	—	—
TXS	Transfer IX to SP	(XK : IX) – $0002 ⇒ SK : SP	INH	374E	—	2	—	—	—	—	—	—	—	—
TXY	Transfer IX to IY	(XK : IX) ⇒ YK : IY	INH	275C	—	2	—	—	—	—	—	—	—	—
TXZ	Transfer IX to IZ	(XK : IX) ⇒ ZK : IZ	INH	276C	—	2	—	—	—	—	—	—	—	—
TYKB	Transfer YK to B	(YK) ⇒ B[3:0] $0 ⇒ B[7:4]	INH	37AD	—	2	—	—	—	—	—	—	—	—
TYS	Transfer IY to SP	(YK : IY) – $0002 ⇒ SK : SP	INH	375E	—	2	—	—	—	—	—	—	—	—
TYX	Transfer IY to IX	(YK : IY) ⇒ XK : IX	INH	274D	—	2	—	—	—	—	—	—	—	—
TYZ	Transfer IY to IZ	(YK : IY) ⇒ ZK : IZ	INH	276D	—	2	—	—	—	—	—	—	—	—
TZKB	Transfer ZK to B	(ZK) ⇒ B[3:0] $0 ⇒ B[7:4]	INH	37AE	—	2	—	—	—	—	—	—	—	—
TZS	Transfer IZ to SP	(ZK : IZ) – $0002 ⇒ SK : SP	INH	376E	—	2	—	—	—	—	—	—	—	—
TZX	Transfer IZ to IX	(ZK : IZ) ⇒ XK : IX	INH	274E	—	2	—	—	—	—	—	—	—	—
TZY	Transfer IZ to IY	(ZK : IZ) ⇒ ZK : IY	INH	275E	—	2	—	—	—	—	—	—	—	—
WAI	Wait for Interrupt	WAIT	INH	27F3	—	8	—	—	—	—	—	—	—	—
XGAB	Exchange A with B	(A) ⇔ (B)	INH	371A	—	2	—	—	—	—	—	—	—	—
XGDE	Exchange D with E	(D) ⇔ (E)	INH	277A	—	2	—	—	—	—	—	—	—	—
XGDX	Exchange D with IX	(D) ⇔ (IX)	INH	37CC	—	2	—	—	—	—	—	—	—	—
XGDY	Exchange D with IY	(D) ⇔ (IY)	INH	37DC	—	2	—	—	—	—	—	—	—	—
XGDZ	Exchange D with IZ	(D) ⇔ (IZ)	INH	37EC	—	2	—	—	—	—	—	—	—	—

Table 33 Instruction Set Summary (Continued)

Mnemonic	Operation	Description	Address Mode	Instruction			Condition Codes							
				Opcode	Operand	Cycles	S	MV	H	EV	N	Z	V	C
XGEX	Exchange E with IX	(E) ⇔ (IX)	INH	374C	—	2	—	—	—	—	—	—	—	—
XGEY	Exchange E with IY	(E) ⇔ (IY)	INH	375C	—	2	—	—	—	—	—	—	—	—
XGEZ	Exchange E with IZ	(E) ⇔ (IZ)	INH	376C	—	2	—	—	—	—	—	—	—	—

NOTES:

1. CCR[15:4] change according to results of operation. The PK field is not affected.
2. Cycle times for conditional branches are shown in “taken, not taken” order.
3. CCR[15:0] change according to copy of CCR pulled from stack.
4. PK field changes according to state pulled from stack. The rest of the CCR is not affected.

Table 34 Instruction Set Abbreviations and Symbols

A	—	Accumulator A
AM	—	Accumulator M
B	—	Accumulator B
CCR	—	Condition code register
D	—	Accumulator D
E	—	Accumulator E
EK	—	Extended addressing extension field
IR	—	MAC multiplicand register
HR	—	MAC multiplier register
IX	—	Index register X
IY	—	Index register Y
IZ	—	Index register Z
K	—	Address extension register
PC	—	Program counter
PK	—	Program counter extension field
SK	—	Stack pointer extension field
SL	—	Multiply and accumulate sign latch
SP	—	Stack pointer
XK	—	Index register X extension field
YK	—	Index register Y extension field
ZK	—	Index register Z extension field
XMSK	—	Modulo addressing index register X mask
YMSK	—	Modulo addressing index register Y mask
S	—	Stop disable control bit
MV	—	AM overflow indicator
H	—	Half carry indicator
EV	—	AM extended overflow indicator
N	—	Negative indicator
Z	—	Zero indicator
V	—	Two's complement overflow indicator
C	—	Carry/borrow indicator
IP	—	Interrupt priority field
SM	—	Saturation mode control bit
PK	—	Program counter extension field
—	—	Bit not affected
Δ	—	Bit changes as specified
0	—	Bit cleared
1	—	Bit set
M	—	Memory location used in operation
R	—	Result of operation
S	—	Source data

X	—	Register used in operation
M	—	Address of one memory byte
M +1	—	Address of byte at M + $0001
M : M + 1	—	Address of one memory word
(…)X	—	Contents of address pointed to by IX
(...)Y	—	Contents of address pointed to by IY
(...)Z	—	Contents of address pointed to by IZ
E, X	—	IX with E offset
E, Y	—	IY with E offset
E, Z	—	IZ with E offset
EXT	—	Extended
EXT20	—	20-bit extended
IMM8	—	8-bit immediate
IMM16	—	16-bit immediate
IND8, X	—	IX with unsigned 8-bit offset
IND8, Y	—	IY with unsigned 8-bit offset
IND8, Z	—	IZ with unsigned 8-bit offset
IND16, X	—	IX with signed 16-bit offset
IND16, Y	—	IY with signed 16-bit offset
IND16, Z	—	IZ with signed 16-bit offset
IND20, X	—	IX with signed 20-bit offset
IND20, Y	—	IY with signed 20-bit offset
IND20, Z	—	IZ with signed 20-bit offset
INH	—	Inherent
IXP	—	Post-modified indexed
REL8	—	8-bit relative
REL16	—	16-bit relative
b	—	4-bit address extension
ff	—	8-bit unsigned offset
gggg	—	16-bit signed offset
hh	—	High byte of 16-bit extended address
ii	—	8-bit immediate data
jj	—	High byte of 16-bit immediate data
kk	—	Low byte of 16-bit immediate data
ll	—	Low byte of 16-bit extended address
mm	—	8-bit mask
mmmm	—	16-bit mask
rr	—	8-bit unsigned relative offset
rrrr	—	16-bit signed relative offset
xo	—	MAC index register X offset
yo	—	MAC index register Y offset
z	—	4-bit zero extension

+	—	Addition
−	—	Subtraction or negation (two's complement)
*	—	Multiplication
/	—	Division
>	—	Greater
<	—	Less
=	—	Equal
≥	—	Equal or greater
≤	—	Equal or less
≠	—	Not equal

•	—	AND
✣	—	Inclusive OR (OR)
⊕	—	Exclusive OR (EOR)
$\overline{\text{NOT}}$	—	Complementation
:	—	Concatenation
⇒	—	Transferred
⇔	—	Exchanged
±	—	Sign bit; also used to show tolerance
«	—	Sign extension
%	—	Binary value
$	—	Hexadecimal value

4.7 Exceptions

An exception is an event that preempts normal instruction process. Exception processing makes the transition from normal instruction execution to execution of a routine that deals with an exception.

Each exception has an assigned vector that points to an associated handler routine. Exception processing includes all operations required to transfer control to a handler routine, but does not include execution of the handler routine itself. Keep the distinction between exception processing and execution of an exception handler in mind while reading this section.

4.7.1 Exception Vectors

An exception vector is the address of a routine that handles an exception. Exception vectors are contained in a data structure called the exception vector table, which is located in the first 512 bytes of bank 0. Refer to **Table 35** for the exception vector table.

All vectors except the reset vector consist of one word and reside in data space. The reset vector consists of four words that reside in program space. There are 52 predefined or reserved vectors, and 200 user-defined vectors.

Each vector is assigned an 8-bit number. Vector numbers for some exceptions are generated by external devices; others are supplied by the processor. There is a direct mapping of vector number to vector table address. The processor left shifts the vector number one place (multiplies by two) to convert it to an address.

Table 35 Exception Vector Table

Vector Number	Vector Address	Address Space	Type of Exception
0	0000	P	Reset — initial ZK, SK, and PK
	0002	P	Reset — initial PC
	0004	P	Reset — initial SP
	0006	P	Reset — initial IZ (direct page)
4	0008	D	Breakpoint
5	000A	D	Bus error
6	000C	D	Software interrupt
7	000E	D	Illegal instruction
8	0010	D	Division by zero
9 – E	0012 – 001C	D	Unassigned, reserved
F	001E	D	Uninitialized interrupt
10	0020	D	Unassigned, reserved
11	0022	D	Level 1 interrupt autovector
12	0024	D	Level 2 interrupt autovector
13	0026	D	Level 3 interrupt autovector
14	0028	D	Level 4 interrupt autovector
15	002A	D	Level 5 interrupt autovector
16	002C	D	Level 6 interrupt autovector
17	002E	D	Level 7 interrupt autovector
18	0030	D	Spurious interrupt
19 – 37	0032 – 006E	D	Unassigned, reserved
38 – FF	0070 – 01FE	D	User-defined interrupts

4.7.2 Exception Stack Frame

During exception processing, the contents of the program counter and condition code register are stacked at a location pointed to by SK:SP. Unless it is altered during exception processing, the stacked PK : PC value is the address of the next instruction in the current instruction stream, plus $0006. **Figure 13** shows the exception stack frame.

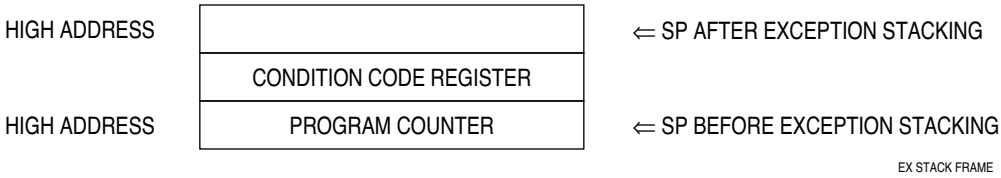


Figure 13 Exception Stack Frame Format

4.7.3 Exception Processing Sequence

Exception processing is performed in four phases.

1. Priority of all pending exceptions is evaluated, and the highest priority exception is processed first.
2. Processor state is stacked, then the CCR PK extension field is cleared.
3. An exception vector number is acquired and converted to a vector address.
4. The content of the vector address is loaded into the PC, and the processor jumps to the exception handler routine.

There are variations within each phase for differing types of exceptions. However, all vectors but the reset vectors contain 16-bit addresses, and the PK field is cleared. Exception handlers must be located within bank 0 or vectors must point to a jump table.

4.7.4 Types of Exceptions

Exceptions can be either internally or externally generated. External exceptions, which are defined as asynchronous, include interrupts, bus errors (BERR), breakpoints (BKPT), and resets (RESET). Internal exceptions, which are defined as synchronous, include the software interrupt (SWI) instruction, the background (BGND) instruction, illegal instruction exceptions, and the divide-by-zero exception.

4.7.4.1 Asynchronous Exceptions

Asynchronous exceptions occur without reference to CPU16 or IMB clocks, but exception processing is synchronized. For all asynchronous exceptions but RESET, exception processing begins at the first instruction boundary following recognition of an exception.

Because of pipelining, the stacked return PK : PC value for all asynchronous exceptions, other than reset, is equal to the address of the next instruction in the current instruction stream plus $0006. The RTI instruction, which must terminate all exception handler routines, subtracts $0006 from the stacked value to resume execution of the interrupted instruction stream.

4.7.4.2 Synchronous Exceptions

Synchronous exception processing is part of an instruction definition. Exception processing for synchronous exceptions is always completed, and the first instruction of the handler routine is always executed, before interrupts are detected.

Because of pipelining, the value of PK : PC at the time a synchronous exception executes is equal to the address of the instruction that causes the exception plus $0006. Because RTI always subtracts $0006 upon return, the stacked PK : PC must be adjusted by the instruction that caused the exception so that execution resumes with the following instruction. For this reason, $0002 is added to the PK : PC value before it is stacked.

4.7.5 Multiple Exceptions

Each exception has a hardware priority based upon its relative importance to system operation. Asynchronous exceptions have higher priorities than synchronous exceptions. Exception processing for multiple exceptions is completed by priority, from highest to lowest. Priority governs the order in which exception processing occurs, not the order in which exception handlers are executed.

Unless a bus error, a breakpoint, or a reset occurs during exception processing, the first instruction of all exception handler routines is guaranteed to execute before another exception is processed. Because interrupt exceptions have higher priority than synchronous exceptions, the first instruction in an interrupt handler are executed before other interrupts are sensed.

Bus error, breakpoint, and reset exceptions that occur during exception processing of a previous exception are processed before the first instruction of that exception's handler routine. The converse is not true. If an interrupt occurs during bus error exception processing, for example, the first instruction of the exception handler is executed before interrupts are sensed. This permits the exception handler to mask interrupts during execution.

4.7.6 RTI Instruction

The "return from interrupt instruction" (RTI) must be the last instruction in all exception handlers except the RESET handler. RTI pulls the exception stack frame that was pushed onto the system stack during exception processing, and restores processor state. Normal program flow resumes at the address of the instruction that follows the last instruction executed before exception processing began.

RTI is not used in the RESET handler because RESET initializes the stack pointer and does not create a stack frame.

5 Standby RAM Module

The standby RAM module (SRAM) provides two Kbytes of fast RAM that is especially useful for system stacks and variable storage. The SRAM has a dedicated power supply pin so that memory content can be preserved when the MCU is powered down.

5.1 Overview

The SRAM module consists of a control register block that is located at a fixed range of addresses in MCU address space, and a 2-Kbyte array of two bus cycle static RAM that can be mapped to any 2-Kbyte boundary in MCU address space. SRAM control registers are located at addresses $YFFB00–YFFB08.

The module responds to program and data space accesses. Data can be read or written in bytes, words, or long words. The RAM array must not be mapped so that array addresses overlap module control register addresses, as overlap makes the registers inaccessible.

The SRAM is powered by V_{DD} in normal operation. During power-down, SRAM contents are maintained by power from the V_{STBY} input. Power switching between sources is automatic.

Table 36 shows the SRAM address map.

Table 36 SRAM Address Map

Address	15 0
$YFFB00[1]	RAM Module Configuration Register (RAMMCR)
$YFFB02	RAM Test Register (RAMTST)
$YFFB04	RAM Array Base Address Register High (RAMBAH)
$YFFB06	RAM Array Base Address Register Low (RAMBAL)

NOTES:
1. Y = M111, where M is the logic state of the module mapping (MM) bit in SIMCR.

5.2 SRAM Register Block

There are four SRAM control registers: the SRAM module configuration register (RAMMCR), the SRAM test register (RAMTST), and the SRAM array base address registers (RAMBAH/RAMBAL).

5.3 SRAM Registers

SRAM responds to both program and data space accesses based on the value in the RASP field in RAMMCR. This allows code to be executed from RAM.

RAMMCR — RAM Module Configuration Register **$YFFB00**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STOP	0	0	0	RLCK	0	RASP[1:0]		NOT USED							
RESET:															
1	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0

Use RAMMCR to determine whether the RAM is in STOP mode or normal mode. RAMMCR can determine in which space the array resides and also controls access to the base array registers. Reads of unimplemented bits always return zeros. Writes do not affect unimplemented bits.

STOP — Stop Control

0 = RAM array operates normally.

1 = RAM array enters low-power stop mode.

This bit controls whether the RAM array is in stop mode or normal operation. Reset state is one, leaving the array configured for LPSTOP operation. In stop mode, the array retains its contents, but cannot be read or written by the CPU. This bit can be read or written at any time.

RLCK — RAM Base Address Lock

0 = SRAM base address registers can be written from IMB

1 = SRAM base address registers are locked

RLCK defaults to zero on reset. It can be written to one once.

RASP[1:0] — RAM Array Space

This field limits access to the SRAM array in microcontrollers that support separate user and supervisor operating modes. Because the CPU16 operates in supervisor mode only, RASP1 has no effect. Refer to **Table 37**.

Table 37 RASP Encoding

RASP	Space
X0	Program and data
X1	Program

RAMTST — RAM Test Register **$YFFB02**

RAMTST is for factory test only. Reads of this register return zeros and writes have no effect.

RAMBAH — Array Base Address Register High **$YFFB04**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NOT USED								ADDR 23	ADDR 22	ADDR 21	ADDR 20	ADDR 19	ADDR 18	ADDR 17	ADDR 16
RESET:								0	0	0	0	0	0	0	0

RAMBAL — Array Base Address Register Low **$YFFB06**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR 15	ADDR 14	ADDR 13	ADDR 12	ADDR 11	NOT USED										
RESET:															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

RAMBAH and RAMBAL specify an SRAM base address in the system memory map. They can only be written while the SRAM is in low-power mode (RAMMCR STOP = 1, the default out of reset) and the base address lock is disabled (RAMMCR RLCK = 0, the default out of reset). This prevents accidental remapping of the array. Because the CPU16 drives ADDR[23:20] to the same logic level as ADDR19, the values of the RAMBAH ADDR[23:20] fields must match the value of the ADDR19 field for the array to be accessible.

5.4 SRAM Operation

There are five SRAM operating modes. They include the following:

1. The RAM module is in normal mode when powered by V_{DD}. The array can be accessed by byte, word, or long word. A byte or aligned word (high-order byte is at an even address) access only takes one bus cycle or two system clocks. A long word or misaligned word access requires two bus cycles.
2. Standby mode is intended to preserve RAM contents when V_{DD} is removed. SRAM contents are maintained by a power source connected to the V_{STBY} pin. The standby voltage is referred to as V_{SB}. Circuitry within the SRAM module switches to the higher of V_{DD} or V_{SB} with no loss of data. When SRAM is powered from the V_{STBY} pin, access to the array is not guaranteed. If standby operation is not desired, connect the V_{STBY} pin to V_{SS}.
3. Reset mode allows the CPU to complete the current bus cycle before resetting. When a synchronous reset occurs while a byte or word SRAM access is in progress, the access is completed. If reset occurs during the first word access of a long-word operation, only the first word access is completed. If reset occurs during the second word access of a long word operation, the entire access is completed. Data being read from or written to the RAM may be corrupted by asynchronous reset.
4. Test mode is used for factory testing of the RAM array.
5. Writing the STOP bit of RAMMCR causes the SRAM module to enter stop mode. The RAM array is disabled which, if necessary, allows external logic to decode SRAM addresses but all data is retained. If V_{DD} falls below V_{SB}, internal circuitry switches to V_{SB}, as in standby mode. Exit the stop mode by clearing the STOP bit.

6 Electrical Characteristics

This section contains 20.97 MHz and 25.17 MHz electrical specification tables and reference timing diagrams.

Table 38 20.97/25.17 MHz Maximum Ratings

Num	Rating	Symbol	Value	Unit
1	Supply Voltage[1, 2, 3]	V_{DD}	– 0.3 to + 6.5	V
2	Input Voltage[1, 2, 3, 4]	V_{in}	– 0.3 to + 6.5	V
3	Instantaneous Maximum Current Single pin limit (applies to all pins)[1, 3, 5, 6]	I_D	25	mA
4	Operating Maximum Current Digital Input Disruptive Current[5, 6, 7] $V_{SS} - 0.3 \le V_{IN} \le V_{DD} + 0.3$	I_{ID}	– 500 to + 500	µA
5	Operating Temperature Range	T_A	T_L to T_H – 40 to + 85	°C
6	Storage Temperature Range	T_{stg}	– 55 to + 150	°C

NOTES:

1. Permanent damage can occur if maximum ratings are exceeded. Exposure to voltages or currents in excess of recommended values affects device reliability. Device modules may not operate normally while being exposed to electrical extremes.
2. Although sections of the device contain circuitry to protect against damage from high static voltages or electrical fields, take normal precautions to avoid exposure to voltages higher than maximum-rated voltages.
3. This parameter is periodically sampled rather than 100% tested.
4. All pins except TSC.
5. All functional non-supply pins are internally clamped to V_{SS} for transitions below V_{SS}. All functional pins except EXTAL and XFC are internally clamped to V_{DD} for transitions below V_{DD}.
6. Power supply must maintain regulation within operating V_{DD} range during instantaneous and operating maximum current conditions.
7. Total input current for all digital input-only and all digital input/output pins must not exceed 10 mA. Exceeding this limit can cause disruption of normal operation.

Table 39 20.97 MHz Typical Ratings

Num	Rating	Symbol	Value	Unit
1	Supply Voltage	V_{DD}	5.0	V
2	Operating Temperature	T_A	25	°C
3	V_{DD} Supply Current RUN LPSTOP, VCO Off LPSTOP, External clock, max f_{sys}	I_{DD}	 60 125 3.0	 mA µA µA
4	Clock Synthesizer Operating Voltage	V_{DDSYN}	5.0	V
5	V_{DDSYN} Supply Current VCO on, maximum f_{sys} External Clock, maximum f_{sys} LPSTOP, VCO off V_{DD} powered down	I_{DDSYN}	 1.0 4.5 100 50	 mA mA µA µA
6	RAM Standby Voltage	V_{SB}	3.0	V
7	RAM Standby Current Normal RAM Operation Standby Operation	I_{SB}	 7.0 40	 µA µA
8	Power Dissipation	P_D	300	mW

Table 40 25.17 MHz Typical Ratings

Num	Rating	Symbol	Value	Unit
1	Supply Voltage	V_{DD}	5.0	V
2	Operating Temperature	T_A	25	°C
3	V_{DD} Supply Current RUN LPSTOP, VCO Off LPSTOP, External clock, max f_{sys}	I_{DD}	 75 125 3.75	 mA µA mA
4	Clock Synthesizer Operating Voltage	V_{DDSYN}	5.0	V
5	V_{DDSYN} Supply Current VCO on, maximum f_{sys} External Clock, maximum f_{sys} LPSTOP, VCO off V_{DD} powered down	I_{DDSYN}	 1.0 5.0 100 50	 mA mA µA µA
6	RAM Standby Voltage	V_{SB}	3.0	V
7	RAM Standby Current Normal RAM Operation Standby Operation	I_{SB}	 7.0 40	 µA µA
8	Power Dissipation	P_D	375	mW

Table 41 Thermal Characteristics

Num	Characteristic	Symbol	Value	Unit
1	Thermal Resistance Plastic 100-Pin Surface Mount	Θ_{JA}	42.5	°C/W

The average chip-junction temperature (T_J) in C can be obtained from:

$$T_J = T_A + (P_D \times \Theta_{JA}) \quad (1)$$

where:

T_A = Ambient Temperature, °C

Θ_{JA}= Package Thermal Resistance, Junction-to-Ambient, °C/W

$P_D = P_{INT} + P_{I/O}$

$P_{INT} = I_{DD} \times V_{DD}$, Watts — Chip Internal Power

$P_{I/O}$= Power Dissipation on Input and Output Pins — User Determined

For most applications $P_{I/O} < P_{INT}$ and can be neglected. An approximate relationship between P_D and T_J (if $P_{I/O}$ is neglected) is:

$$P_D = K \div (T_J + 273°C) \quad (2)$$

Solving equations 1 and 2 for K gives:

$$K = P_D + (T_A + 273°C) + \Theta_{JA} \times P_{D^2} \quad (3)$$

where K is a constant pertaining to the particular part. K can be determined from equation (3) by measuring P_D (at equilibrium) for a known T_A. Using this value of K, the values of P_D and T_J can be obtained by solving equations (1) and (2) iteratively for any value of T_A.

Table 42 20.97 MHz Clock Control Timing

(V_{DD} and V_{DDSYN} = 5.0 Vdc, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H)

Num	Characteristic	Symbol	Min	Max	Unit
1	PLL Reference Frequency Range[1]	f_{ref}	20	50	kHz
2	System Frequency[2] On-Chip PLL System Frequency Range External Clock Operation	f_{sys}	dc $4f_{ref}$ dc	20.97 20.97 20.97	MHz
3	PLL Lock Time[1, 3, 4, 5, 6]	t_{lpll}	—	20	ms
4	VCO Frequency[7]	f_{VCO}	—	2 (f_{sys} max)	MHz
5	Limp Mode Clock Frequency SYNCR X bit = 0 SYNCR X bit = 1	f_{limp}	— —	(f_{sys} max)/2 f_{sys} max	MHz
6	CLKOUT Jitter[1, 4, 5, 6, 8] Short term (5 µs interval) Long term (500 µs interval)	J_{clk}	– 1.5 – 0.5	1.0 0.5	%

NOTES:

1. The base configuration of the MC68HC16S2 requires a 32.768 kHz reference.
2. All internal registers retain data at 0 Hz.
3. Assumes that stable V_{DDSYN} is applied, and that the crystal oscillator is stable. Lock time is measured from the time V_{DD} and V_{DDSYN} are valid until $\overline{\text{RESET}}$ is released. This specification also applies to the period required for PLL lock after changing the W and Y frequency control bits in the synthesizer control register (SYNCR) while the PLL is running, and to the period required for the clock to lock after LPSTOP.
4. This parameter is periodically sampled rather than 100% tested.
5. Assumes that a low-leakage external filter network is used to condition clock synthesizer input voltage. Total external resistance from the XFC pin due to external leakage must be greater than 15 MΩ to guarantee this specification. Filter network geometry can vary depending upon operating environment.
6. Proper layout procedures must be followed to achieve specifications.
7. Internal VCO frequency (f_{VCO}) is determined by SYNCR W and Y bit values.
   The SYNCR X bit controls a divide-by-two circuit that is not in the synthesizer feedback loop.
   When X = 0, the divider is enabled, and $f_{sys} = f_{VCO} \div 4$.
   When X = 1, the divider is disabled, and $f_{sys} = f_{VCO} \div 2$.
   X must equal one when operating at maximum specified f_{sys}.
8. Jitter is the average deviation from the programmed frequency measured over the specified interval at maximum f_{sys}. Measurements are made with the device powered by filtered supplies and clocked by a stable external clock signal. Noise injected into the PLL circuitry via V_{DDSYN} and V_{SS} and variation in crystal oscillator frequency increase the J_{clk} percentage for a given interval. When clock jitter is a critical constraint on control system operation, this parameter should be measured during functional testing of the final system.

Table 43 25.17 MHz Clock Control Timing

(V_{DD} and V_{DDSYN} = 5.0 Vdc, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H)

Num	Characteristic	Symbol	Min	Max	Unit
1	PLL Reference Frequency Range[1]	f_{ref}	20	50	kHz
2	System Frequency[2] On-Chip PLL System Frequency Range External Clock Operation	f_{sys}	dc $4f_{ref}$ dc	25.17 25.17 25.17	MHz
3	PLL Lock Time[1, 3, 4, 5, 6]	t_{lpll}	—	20	ms
4	VCO Frequency[7]	f_{VCO}	—	2 (f_{sys} max)	MHz
5	Limp Mode Clock Frequency SYNCR X bit = 0 SYNCR X bit = 1	f_{limp}	— —	(f_{sys} max)/2 f_{sys} max	MHz
6	CLKOUT Jitter[1, 4, 5, 6, 8] Short term (5 µs interval) Long term (500 µs interval)	J_{clk}	– 1.5 – 0.5	1.0 0.5	%

NOTES:

1. Refer to notes in **Table 42**.

Table 44 20.97 MHz DC Characteristics

(V_{DD} and V_{DDSYN} = 5.0 Vdc, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H)

Num	Characteristic	Symbol	Min	Max	Unit
1	Input High Voltage	V_{IH}	0.7 (V_{DD})	V_{DD} + 0.3	V
2	Input Low Voltage	V_{IL}	V_{SS} – 0.3	0.2 (V_{DD})	V
3	Input Hysteresis[1]	V_{HYS}	0.5	—	V
4	Input Leakage Current[2] $V_{in} = V_{DD}$ or V_{SS}	I_{IN}	–2.5	2.5	µA
5	High Impedance (Off-State) Leakage Current[2] $V_{in} = V_{DD}$ or V_{SS}	I_{OZ}	–2.5	2.5	µA
6	CMOS Output High Voltage[2, 3] I_{OH} = –10.0 µA	V_{OH}	V_{DD} – 0.2	—	V
7	CMOS Output Low Voltage[2] I_{OL} = 10.0 µA	V_{OL}	—	0.2	V
8	Output High Voltage[2, 3] I_{OH} = –0.8 mA	V_{OH}	V_{DD} – 0.8	—	V
9	Output Low Voltage[2] I_{OL} = 1.6 mA I_{OL} = 5.3 mA I_{OL} = 12 mA	V_{OL}	— — —	0.4 0.4 0.4	V
10	Three State Control Input High Voltage	V_{IHTSC}	1.6 (V_{DD})	9.1	V
11	Data Bus Mode Select Pull-up Current[4] $V_{in} = V_{IL}$ DATA[15:0] $V_{in} = V_{IH}$ DATA[15:0]	I_{MSP}	— –15	–120 —	µA
12	V_{DD} Supply Current[5, 6] Run[6], crystal reference LPSTOP, crystal reference, VCO Off (STSIM = 0) LPSTOP, external clock input = max f_{sys}	I_{DD}	— —	110 350 5	mA µA mA
13	Clock Synthesizer Operating Voltage	V_{DDSYN}	4.75	5.25	V
14	V_{DDSYN} Supply Current[5, 6] VCO on, 32.768 kHz crystal reference, maximum f_{sys} External Clock, maximum f_{sys} LPSTOP, 32.768 kHz crystal reference, VCO off (STSIM = 0) 32.768 kHz, V_{DD} powered down	I_{DDSYN}	— — — —	1 5 100 50	mA mA µA µA
15	RAM Standby Voltage[7] Specified V_{DD} applied $V_{DD} = V_{SS}$	V_{SB}	0.0 3.0	5.25 5.25	V
16	RAM Standby Current[7, 6] Normal RAM operation[8] $V_{DD} > V_{SB} – 0.5$ V Transient condition $V_{SB} – 0.5\ V \geq V_{DD} \geq V_{SS} + 0.5$ V Standby operation[7] $V_{DD} < V_{SS} + 0.5$ V	I_{SB}	— — —	10 3 50	µA mA µA
17	Power Dissipation[5, 9]	P_D	—	603	mW
18	Input Capacitance[2, 10] All input-only pins All input/output pins	C_{IN}	— —	10 20	pF

Table 44 20.97 MHz DC Characteristics (Continued)

(V_{DD} and V_{DDSYN} = 5.0 Vdc, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H)

Num	Characteristic	Symbol	Min	Max	Unit
19	Load Capacitance[2] Group 1 I/O Pins, CLKOUT, FREEZE/QUOT, IPIPE0 Group 2 I/O Pins and $\overline{CSBOOT}$, BG/CS Group 3 I/O Pins Group 4 I/O Pins	C_L	— — — —	90 100 130 200	pF

NOTES:

1. Applies to:
   SIZ[1:0], $\overline{AS}$, $\overline{DS}$, $\overline{IRQ[7:1]}$, MODCLK, $\overline{RESET}$, EXTAL, TSC, $\overline{BKPT}$/DSCLK, IPIPE1/DSI
2. Input-Only Pins: EXTAL, TSC, $\overline{BKPT}$/DSCLK
   Output-Only Pins: $\overline{CSBOOT}$, $\overline{BG}$/$\overline{CS1}$, CLKOUT, FREEZE/QUOT, IPIPE0/DSO
   Input/Output Pins:
   Group 1: DATA[15:0], IPIPE1/DSI
   Group 2: Port C[6:0] — ADDR[22:19]/$\overline{CS[9:6]}$, FC[2:0]/$\overline{CS[5:3]}$
   Port E[7:0] — SIZ[1:0], $\overline{AS}$, $\overline{DS}$, $\overline{AVEC}$, $\overline{DSACK[1:0]}$
   Port F[7:0] —$\overline{IRQ[7:1]}$, MODCLK, ADDR23/$\overline{CS10}$/ECLK, ADDR[18:0], R/$\overline{W}$, $\overline{BERR}$, $\overline{BR}$/$\overline{CS0}$, $\overline{BGACK}$/$\overline{CS2}$
   Group 3: $\overline{HALT}$, $\overline{RESET}$
3. Does not apply to $\overline{HALT}$ and $\overline{RESET}$ because they are open drain pins.
4. Use of an active pulldown device is recommended.
5. Total operating current is the sum of the appropriate V_{DD}, supply and V_{DDSYN} supply current. V_{DD} at 3.3V.
6. Current measured with system clock frequency of 20.97 MHz, all modules active.
7. The SRAM module will not switch into standby mode as long as V_{SB} does not exceed V_{DD} by more than 0.5 volts. The SRAM array cannot be accessed while the module is in standby mode.
8. When V_{SB} is more than 0.3V greater than V_{DD}, current flows between the V_{STBY} and V_{DD} pins, which causes standby current to increase toward the maximum transient condition specification. System noise on the V_{DD} and V_{STBY} pin can contribute to this condition.
9. Power dissipation is measured with a system clock frequency of 20.97 MHz, all modules active. Power dissipation is calculated using the following expression:
   $$P_D = \text{Maximum } V_{DD} (I_{DD} + I_{DDSYN} + I_{SB})$$
10. Input capacitance is periodically sampled rather than 100% tested.

Table 45 25.17 MHz DC Characteristics

(V_{DD} and V_{DDSYN} = 5.0 Vdc, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H)

Num	Characteristic	Symbol	Min	Max	Unit
1	Input High Voltage	V_{IH}	0.7 (V_{DD})	V_{DD} + 0.3	V
2	Input Low Voltage	V_{IL}	V_{SS} – 0.3	0.2 (V_{DD})	V
3	Input Hysteresis[1]	V_{HYS}	0.5	—	V
4	Input Leakage Current[2] $V_{in} = V_{DD}$ or V_{SS}	I_{IN}	–2.5	2.5	µA
5	High Impedance (Off-State) Leakage Current[2] $V_{in} = V_{DD}$ or V_{SS}	I_{OZ}	–2.5	2.5	µA
6	CMOS Output High Voltage[2, 3] I_{OH} = –10.0 µA	V_{OH}	V_{DD} – 0.2	—	V
7	CMOS Output Low Voltage[2] I_{OL} = 10.0 µA	V_{OL}	—	0.2	V
8	Output High Voltage[2, 3] I_{OH} = –0.8 mA	V_{OH}	V_{DD} – 0.8	—	V
9	Output Low Voltage[2] I_{OL} = 1.6 mA I_{OL} = 5.3 mA I_{OL} = 12 mA	V_{OL}	— — —	0.4 0.4 0.4	V
10	Three State Control Input High Voltage	V_{IHTSC}	1.6 (V_{DD})	9.1	V
11	Data Bus Mode Select Pull-up Current[4] $V_{in} = V_{IL}$ DATA[15:0] $V_{in} = V_{IH}$ DATA[15:0]	I_{MSP}	— –15	–120 —	µA
12	V_{DD} Supply Current[5, 6] Run[6], crystal reference LPSTOP, crystal reference, VCO Off (STSIM = 0) LPSTOP, external clock input = max f_{sys}	I_{DD}	— —	140 350 5	mA µA mA
13	Clock Synthesizer Operating Voltage	V_{DDSYN}	4.75	5.25	V
14	V_{DDSYN} Supply Current[5, 6] VCO on, 32.768 kHz crystal reference, maximum f_{sys} External Clock, maximum f_{sys} LPSTOP, 32.768 kHz crystal reference, VCO off (STSIM = 0) 32.768 kHz, V_{DD} powered down	I_{DDSYN}	— — — —	2 7 150 100	mA mA µA µA
15	RAM Standby Voltage[7] Specified V_{DD} applied $V_{DD} = V_{SS}$	V_{SB}	0.0 3.0	5.25 5.25	V
16	RAM Standby Current[7, 6] Normal RAM operation[8] $V_{DD} > V_{SB}$ – 0.5 V Transient condition V_{SB} – 0.5 V ≥ V_{DD} ≥ V_{SS} + 0.5 V Standby operation[7] $V_{DD} < V_{SS}$ + 0.5 V	I_{SB}	— — —	10 3 50	µA mA µA
17	Power Dissipation[5, 9]	P_D	—	766	mW
18	Input Capacitance[2, 10] All input-only pins All input/output pins	C_{IN}	— —	10 20	pF

Table 45 25.17 MHz DC Characteristics (Continued)

(V_{DD} and V_{DDSYN} = 5.0 Vdc, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H)

Num	Characteristic	Symbol	Min	Max	Unit
19	Load Capacitance[2] Group 1 I/O Pins, CLKOUT, FREEZE/QUOT, IPIPE0 Group 2 I/O Pins and $\overline{CSBOOT}$, BG/$\overline{CS}$ Group 3 I/O Pins Group 4 I/O Pins	C_L	 — — — —	 90 100 130 200	pF

NOTES:

1. Refer to notes in **Table 44**. Parameters are measured with system clock frequency of 25.17 MHz.

Table 46 20.97 MHz AC Timing

(V_{DD} and V_{DDSYN} = 5.0 Vdc ±10%, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H)[1]

Num	Characteristic	Symbol	Min	Max	Unit
F1	Frequency of Operation[2]	f_{sys}	4 f_{ref}	20.97	MHz
1	Clock Period	t_{cyc}	47.7	—	ns
1A	ECLK Period	t_{Ecyc}	381	—	ns
1B	External Clock Input Period[3]	t_{Xcyc}	47.7	—	ns
2, 3	Clock Pulse Width	t_{CW}	18.8	—	ns
2A, 3A	ECLK Pulse Width	t_{ECW}	183	—	ns
2B, 3B	External Clock Input High/Low Time[3]	t_{XCHL}	23.8	—	ns
4, 5	CLKOUT Rise and Fall Time	t_{Crf}	—	5	ns
4A, 5A	Rise and Fall Time (All Outputs except CLKOUT)	t_{rf}	—	8	ns
4B, 5B	External Clock Input Rise and Fall Time[4]	t_{XCrf}	—	5	ns
6	Clock High to ADDR, FC, SIZE Valid[5]	t_{CHAV}	0	23	ns
7	Clock High to ADDR, Data, FC, SIZE High Impedance	t_{CHAZx}	0	47	ns
8	Clock High to ADDR, FC, SIZE Invalid	t_{CHAZn}	0	—	ns
9	Clock Low to $\overline{AS}$, $\overline{DS}$, $\overline{CS}$ Asserted[5]	t_{CLSA}	0	23	ns
9A	$\overline{AS}$ to $\overline{DS}$ or $\overline{CS}$ Asserted (Read)[6]	t_{STSA}	–10	10	ns
11	ADDR, FC, SIZE Valid to $\overline{AS}$, $\overline{CS}$, (and $\overline{DS}$ Read) Asserted	t_{AVSA}	10	—	ns
12	Clock Low to $\overline{AS}$, $\overline{DS}$, $\overline{CS}$ Negated	t_{CLSN}	2	23	ns
13	$\overline{AS}$, $\overline{DS}$, $\overline{CS}$ Negated to ADDR, FC SIZE Invalid (Address Hold)	t_{SNAI}	10	—	ns
14	$\overline{AS}$, $\overline{CS}$ (and $\overline{DS}$ Read) Width Asserted	t_{SWA}	80	—	ns
14A	$\overline{DS}$, $\overline{CS}$ Width Asserted (Write)	t_{SWAW}	36	—	ns
14B	$\overline{AS}$, $\overline{CS}$ (and $\overline{DS}$ Read) Width Asserted (Fast Cycle)	t_{SWDW}	32	—	ns
15	$\overline{AS}$, $\overline{DS}$, $\overline{CS}$ Width Negated[7]	t_{SN}	32	—	ns
16	Clock High to $\overline{AS}$, $\overline{DS}$, R/$\overline{W}$ High Impedance	t_{CHSZ}	—	47	ns
17	$\overline{AS}$, $\overline{DS}$, $\overline{CS}$ Negated to R/$\overline{W}$ High	t_{SNRN}	10	—	ns
18	Clock High to R/$\overline{W}$ High	t_{CHRH}	0	23	ns
20	Clock High to R/$\overline{W}$ Low	t_{CHRL}	0	23	ns
21	R/$\overline{W}$ High to $\overline{AS}$, $\overline{CS}$ Asserted	t_{RAAA}	10	—	ns
22	R/$\overline{W}$ Low to $\overline{DS}$, $\overline{CS}$ Asserted (Write)	t_{RASA}	54	—	ns
23	Clock High to Data Out Valid	t_{CHDO}	—	23	ns
24	Data Out Valid to Negating Edge of $\overline{AS}$, $\overline{CS}$ (Fast Write Cycle)	t_{DVASN}	10	—	ns
25	$\overline{DS}$, $\overline{CS}$ Negated to Data Out Invalid (Data Out Hold)	t_{SNDOI}	10	—	ns
26	Data Out Valid to $\overline{DS}$, $\overline{CS}$ Asserted (Write)	t_{DVSA}	10	—	ns
27	Data In Valid to Clock Low (Data Setup)[5]	t_{DICL}	5	—	ns
27A	Late $\overline{BERR}$, $\overline{HALT}$ Asserted to Clock Low (Setup Time)	t_{BELCL}	15	—	ns
28	$\overline{AS}$, $\overline{DS}$ Negated to $\overline{DSACK}$[1:0], $\overline{BERR}$, $\overline{HALT}$, $\overline{AVEC}$ Negated	t_{SNDN}	0	60	ns
29	$\overline{DS}$, $\overline{CS}$ Negated to Data In Invalid (Data In Hold)[8]	t_{SNDI}	0	—	ns
29A	$\overline{DS}$, $\overline{CS}$ Negated to Data In High Impedance[8, 9]	t_{SHDI}	—	48	ns
30	CLKOUT Low to Data In Invalid (Fast Cycle Hold)[8]	t_{CLDI}	10	—	ns
30A	CLKOUT Low to Data In High Impedance[8]	t_{CLDH}	—	72	ns

Table 46 20.97 MHz AC Timing (Continued)

(V_{DD} and V_{DDSYN} = 5.0 Vdc ±10%, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H)[1]

Num	Characteristic	Symbol	Min	Max	Unit
31	$\overline{DSACK[1:0]}$ Asserted to Data In Valid[10]	t_{DADI}	—	46	ns
33	Clock Low to $\overline{BG}$ Asserted/Negated	t_{CLBAN}	—	23	ns
35	$\overline{BR}$ Asserted to $\overline{BG}$ Asserted[11]	t_{BRAGA}	1	—	t_{cyc}
37	$\overline{BGACK}$ Asserted to $\overline{BG}$ Negated	t_{GAGN}	1	2	t_{cyc}
39	$\overline{BG}$ Width Negated	t_{GH}	2	—	t_{cyc}
39A	$\overline{BG}$ Width Asserted	t_{GA}	1	—	t_{cyc}
46	R/$\overline{W}$ Width Asserted (Write or Read)	t_{RWA}	115	—	ns
46A	R/$\overline{W}$ Width Asserted (Fast Write or Read Cycle)	t_{RWAS}	70	—	ns
47A	Asynchronous Input Setup Time $\overline{BR}$, $\overline{BGACK}$, $\overline{DSACK[1:0]}$, $\overline{BERR}$, $\overline{AVEC}$, $\overline{HALT}$	t_{AIST}	5	—	ns
47B	Asynchronous Input Hold Time	t_{AIHT}	12	—	ns
48	$\overline{DSACK[1:0]}$ Asserted to $\overline{BERR}$, $\overline{HALT}$ Asserted[12]	t_{DABA}	—	30	ns
53	Data Out Hold from Clock High	t_{DOCH}	0	—	ns
54	Clock High to Data Out High Impedance	t_{CHDH}	—	23	ns
55	R/$\overline{W}$ Asserted to Data Bus Impedance Change	t_{RADC}	32	—	ns
70	Clock Low to Data Bus Driven (Show Cycle)	t_{SCLDD}	0	23	ns
71	Data Setup Time to Clock Low (Show Cycle)	t_{SCLDS}	10	—	ns
72	Data Hold from Clock Low (Show Cycle)	t_{SCLDH}	10	—	ns
73	$\overline{BKPT}$ Input Setup Time	t_{BKST}	10	—	ns
74	$\overline{BKPT}$ Input Hold Time	t_{BKHT}	10	—	ns
75	Mode Select Setup Time (DATA[15:0], MODCLK, $\overline{BKPT}$)	t_{MSS}	20	—	t_{cyc}
76	Mode Select Hold Time (DATA[15:0], MODCLK, $\overline{BKPT}$)	t_{MSH}	0	—	ns
77	$\overline{RESET}$ Assertion Time[13]	t_{RSTA}	4	—	t_{cyc}
78	$\overline{RESET}$ Rise Time[14, 15]	t_{RSTR}	—	10	t_{cyc}
100	CLKOUT High to Phase 1 Asserted[16]	t_{CHP1A}	3	40	ns
101	CLKOUT High to Phase 2 Asserted	t_{CHP2A}	3	40	ns
102	Phase 1 Valid to $\overline{AS}$ or $\overline{DS}$ Asserted	t_{P1VSA}	10	—	ns
103	Phase 2 Valid to $\overline{AS}$ or $\overline{DS}$ Asserted	t_{P2VSN}	10	—	ns
104	$\overline{AS}$ or $\overline{DS}$ Valid to Phase 1 Negated	t_{SAP1N}	10	—	ns
105	$\overline{AS}$ or $\overline{DS}$ Valid to Phase 2 Negated	t_{SNP2N}	10	—	ns

NOTES:

1. All AC timing is shown with respect to 20% V_{DD} and 70% V_{DD} levels unless otherwise noted.
2. The base configuration of the MC68HC16S2 requires a 32.768 kHz crystal reference.
3. When an external clock is used, minimum high and low times are based on a 50% duty cycle. The minimum allowable t_{Xcyc} period is reduced when the duty cycle of the external clock varies. The relationship between external clock input duty cycle and minimum t_{Xcyc} is expressed:

   Minimum t_{Xcyc} period = minimum t_{XCHL} / (50% – external clock input duty cycle tolerance).
4. Parameters for an external clock signal applied while the internal PLL is disabled (MODCLK pin held low during reset). Does not pertain to an external reference applied while the PLL is enabled (MODCLK pin held high during reset). When the PLL is enabled, the clock synthesizer detects successive transitions of the reference signal. If transitions occur within the correct clock period, rise/fall times and duty cycle are not critical.
5. Address access time = (2.5 + WS) $t_{cyc} – t_{CHAV} – t_{DICL}$
   Chip-select access time = (2 + WS) $t_{cyc} – t_{CLSA} – t_{DICL}$
   Where: WS = number of wait states. When fast termination is used (2 clock bus) WS = –1.
6. Specification 9A is the worst-case skew between $\overline{AS}$ and $\overline{DS}$ or $\overline{CS}$. The amount of skew depends on the relative loading of these signals. When loads are kept within specified limits, skew will not cause $\overline{AS}$ and $\overline{DS}$ to fall outside the limits shown in specification 9.
7. If multiple chip selects are used, $\overline{CS}$ width negated (specification 15) applies to the time from the negation of a heavily loaded chip select to the assertion of a lightly loaded chip select. The $\overline{CS}$ width negated specification between multiple chip selects does not apply to chip selects being used for synchronous ECLK cycles.
8. Hold times are specified with respect to $\overline{DS}$ or $\overline{CS}$ on asynchronous reads and with respect to CLKOUT on fast cycle reads. The user is free to use either hold time.
9. Maximum value is equal to (t_{cyc} / 2) + 25 ns.
10. If the asynchronous setup time (specification 47A) requirements are satisfied, the $\overline{DSACK[1:0]}$ low to data setup time (specification 31) and $\overline{DSACK[1:0]}$ low to $\overline{BERR}$ low setup time (specification 48) can be ignored. The data must only satisfy the data-in to clock low setup time (specification 27) for the following clock cycle. $\overline{BERR}$ must satisfy only the late $\overline{BERR}$ low to clock low setup time (specification 27A) for the following clock cycle.
11. To ensure coherency during every operand transfer, $\overline{BG}$ is not asserted in response to $\overline{BR}$ until after all cycles of the current operand transfer are complete.
12. In the absence of $\overline{DSACK[1:0]}$, $\overline{BERR}$ is an asynchronous input using the asynchronous setup time (specification 47A).
13. After external $\overline{RESET}$ negation is detected, a short transition period (approximately 2 t_{cyc}) elapses, then the SIM drives $\overline{RESET}$ low for 512 tcyc.
14. External assertion of the $\overline{RESET}$ input can overlap internally-generated resets. To ensure that an external reset is recognized in all cases, $\overline{RESET}$ must be asserted for at least 590 CLKOUT cycles.
15. External logic must pull $\overline{RESET}$ high during this period in order for normal MCU operation to begin.
16. Eight pipeline states are multiplexed into IPIPE[1:0]. The multiplexed signals have two phases.

Table 47 25.17 MHz AC Timing

(V_{DD} and V_{DDSYN} = 5.0 Vdc ±10%, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H)[1]

Num	Characteristic	Symbol	Min	Max	Unit
F1	Frequency of Operation[2]	f_{sys}	4 f_{ref}	25.166	MHz
1	Clock Period	t_{cyc}	39.7	—	ns
1A	ECLK Period	t_{Ecyc}	318	—	ns
1B	External Clock Input Period[3]	t_{Xcyc}	39.7	—	ns
2, 3	Clock Pulse Width	t_{CW}	15	—	ns
2A, 3A	ECLK Pulse Width	t_{ECW}	155	—	ns
2B, 3B	External Clock Input High/Low Time[3]	t_{XCHL}	19.8	—	ns
4, 5	CLKOUT Rise and Fall Time	t_{Crf}	—	5	ns
4A, 5A	Rise and Fall Time (All Outputs except CLKOUT)	t_{rf}	—	8	ns
4B, 5B	External Clock Input Rise and Fall Time[4]	t_{XCrf}	—	4	ns
6	Clock High to ADDR, FC, SIZE Valid[5]	t_{CHAV}	0	19	ns
7	Clock High to ADDR, Data, FC, SIZE High Impedance	t_{CHAZx}	0	39	ns
8	Clock High to ADDR, FC, SIZE Invalid	t_{CHAZn}	0	—	ns
9	Clock Low to $\overline{AS}$, $\overline{DS}$, $\overline{CS}$ Asserted[5]	t_{CLSA}	2	19	ns
9A	$\overline{AS}$ to $\overline{DS}$ or $\overline{CS}$ Asserted (Read)[6]	t_{STSA}	–10	15	ns
11	ADDR, FC, SIZE Valid to $\overline{AS}$, $\overline{CS}$, (and $\overline{DS}$ Read) Asserted	t_{AVSA}	8	—	ns
12	Clock Low to $\overline{AS}$, $\overline{DS}$, $\overline{CS}$ Negated	t_{CLSN}	2	19	ns
13	$\overline{AS}$, $\overline{DS}$, $\overline{CS}$ Negated to ADDR, FC SIZE Invalid (Address Hold)	t_{SNAI}	8	—	ns
14	$\overline{AS}$, $\overline{CS}$ (and $\overline{DS}$ Read) Width Asserted	t_{SWA}	65	—	ns
14A	$\overline{DS}$, $\overline{CS}$ Width Asserted (Write)	t_{SWAW}	25	—	ns
14B	$\overline{AS}$, $\overline{CS}$ (and $\overline{DS}$ Read) Width Asserted (Fast Cycle)	t_{SWDW}	22	—	ns
15	$\overline{AS}$, $\overline{DS}$, $\overline{CS}$ Width Negated[7]	t_{SN}	22	—	ns
16	Clock High to $\overline{AS}$, $\overline{DS}$, R/$\overline{W}$ High Impedance	t_{CHSZ}	—	39	ns
17	$\overline{AS}$, $\overline{DS}$, $\overline{CS}$ Negated to R/$\overline{W}$ High	t_{SNRN}	10	—	ns
18	Clock High to R/$\overline{W}$ High	t_{CHRH}	0	19	ns
20	Clock High to R/$\overline{W}$ Low	t_{CHRL}	0	19	ns
21	R/$\overline{W}$ High to $\overline{AS}$, $\overline{CS}$ Asserted	t_{RAAA}	10	—	ns
22	R/$\overline{W}$ Low to $\overline{DS}$, $\overline{CS}$ Asserted (Write)	t_{RASA}	40	—	ns
23	Clock High to Data Out Valid	t_{CHDO}	—	19	ns
24	Data Out Valid to Negating Edge of $\overline{AS}$, $\overline{CS}$ (Fast Write Cycle)	t_{DVASN}	7	—	ns
25	$\overline{DS}$, $\overline{CS}$ Negated to Data Out Invalid (Data Out Hold)	t_{SNDOI}	5	—	ns
26	Data Out Valid to $\overline{DS}$, $\overline{CS}$ Asserted (Write)	t_{DVSA}	8	—	ns
27	Data In Valid to Clock Low (Data Setup)[5]	t_{DICL}	5	—	ns
27A	Late $\overline{BERR}$, $\overline{HALT}$ Asserted to Clock Low (Setup Time)	t_{BELCL}	10	—	ns
28	$\overline{AS}$, $\overline{DS}$ Negated to $\overline{DSACK}$[1:0], $\overline{BERR}$, $\overline{HALT}$, $\overline{AVEC}$ Negated	t_{SNDN}	0	50	ns
29	$\overline{DS}$, $\overline{CS}$ Negated to Data In Invalid (Data In Hold)[8]	t_{SNDI}	0	—	ns
29A	$\overline{DS}$, $\overline{CS}$ Negated to Data In High Impedance[8, 9]	t_{SHDI}	—	45	ns
30	CLKOUT Low to Data In Invalid (Fast Cycle Hold)[8]	t_{CLDI}	8	—	ns

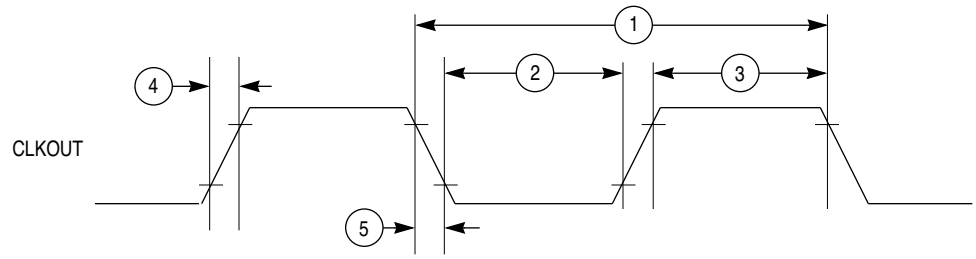
Table 47 25.17 MHz AC Timing (Continued)

(V_{DD} and V_{DDSYN} = 5.0 Vdc ±10%, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H)[1]

Num	Characteristic	Symbol	Min	Max	Unit
30A	CLKOUT Low to Data In High Impedance[8]	t_{CLDH}	—	60	ns
31	$\overline{DSACK[1:0]}$ Asserted to Data In Valid[10]	t_{DADI}	—	35	ns
33	Clock Low to $\overline{BG}$ Asserted/Negated	t_{CLBAN}	—	19	ns
35	$\overline{BR}$ Asserted to $\overline{BG}$ Asserted[11]	t_{BRAGA}	1	—	t_{cyc}
37	$\overline{BGACK}$ Asserted to $\overline{BG}$ Negated	t_{GAGN}	1	2	t_{cyc}
39	$\overline{BG}$ Width Negated	t_{GH}	2	—	t_{cyc}
39A	$\overline{BG}$ Width Asserted	t_{GA}	1	—	t_{cyc}
46	R/$\overline{W}$ Width Asserted (Write or Read)	t_{RWA}	90	—	ns
46A	R/$\overline{W}$ Width Asserted (Fast Write or Read Cycle)	t_{RWAS}	55	—	ns
47A	Asynchronous Input Setup Time $\overline{BR}$, $\overline{BGACK}$, $\overline{DSACK[1:0]}$, $\overline{BERR}$, $\overline{AVEC}$, $\overline{HALT}$	t_{AIST}	5	—	ns
47B	Asynchronous Input Hold Time	t_{AIHT}	10	—	ns
48	$\overline{DSACK[1:0]}$ Asserted to $\overline{BERR}$, $\overline{HALT}$ Asserted[12]	t_{DABA}	—	27	ns
53	Data Out Hold from Clock High	t_{DOCH}	0	—	ns
54	Clock High to Data Out High Impedance	t_{CHDH}	—	23	ns
55	R/$\overline{W}$ Asserted to Data Bus Impedance Change	t_{RADC}	25	—	ns
70	Clock Low to Data Bus Driven (Show Cycle)	t_{SCLDD}	0	19	ns
71	Data Setup Time to Clock Low (Show Cycle)	t_{SCLDS}	8	—	ns
72	Data Hold from Clock Low (Show Cycle)	t_{SCLDH}	8	—	ns
73	$\overline{BKPT}$ Input Setup Time	t_{BKST}	10	—	ns
74	$\overline{BKPT}$ Input Hold Time	t_{BKHT}	10	—	ns
75	Mode Select Setup Time (DATA[15:0], MODCLK, $\overline{BKPT}$)	t_{MSS}	20	—	t_{cyc}
76	Mode Select Hold Time (DATA[15:0], MODCLK, $\overline{BKPT}$)	t_{MSH}	0	—	ns
77	$\overline{RESET}$ Assertion Time[13]	t_{RSTA}	4	—	t_{cyc}
78	$\overline{RESET}$ Rise Time[14, 15]	t_{RSTR}	—	10	t_{cyc}
100	CLKOUT High to Phase 1 Asserted[16]	t_{CHP1A}	3	34	ns
101	CLKOUT High to Phase 2 Asserted	t_{CHP2A}	3	34	ns
102	Phase 1 Valid to $\overline{AS}$ or $\overline{DS}$ Asserted	t_{P1VSA}	9	—	ns
103	Phase 2 Valid to $\overline{AS}$ or $\overline{DS}$ Asserted	t_{P2VSN}	9	—	ns
104	$\overline{AS}$ or $\overline{DS}$ Valid to Phase 1 Negated	t_{SAP1N}	9	—	ns
105	$\overline{AS}$ or $\overline{DS}$ Valid to Phase 2 Negated	t_{SNP2N}	9	—	ns

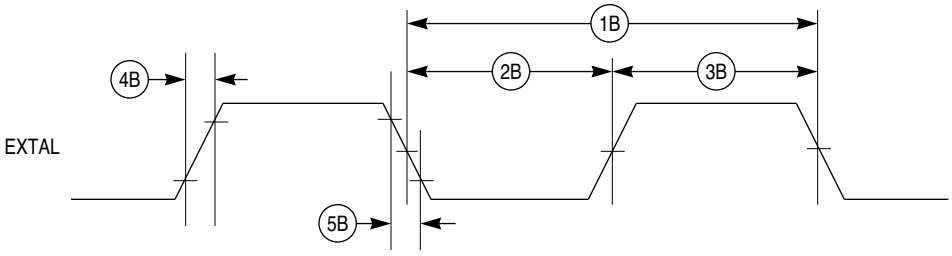
NOTES:

1. Refer to notes in **Table 46**. Parameters are measured with system clock frequency of 25.17 MHz.



16 CLKOUT TIM

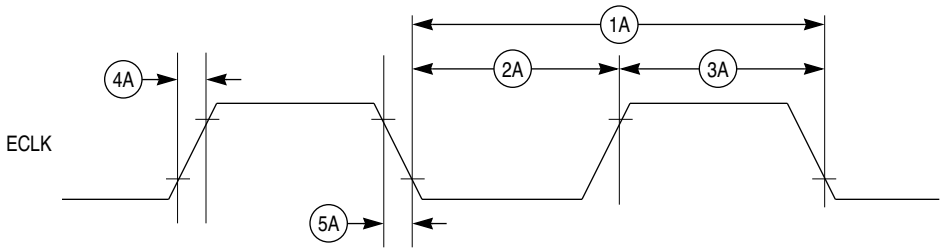
Figure 14 CLKOUT Output Timing Diagram



NOTE: TIMING SHOWN WITH RESPECT TO 20% AND 70% V_{DD}.
PULSE WIDTH SHOWN WITH RESPECT TO 50% V_{DD}.

16 EXT CLK INPUT TIM

Figure 15 External Clock Input Timing Diagram



NOTE: TIMING SHOWN WITH RESPECT TO 20% AND 70% V_{DD}.

16 ECLK OUTPUT TIM

Figure 16 ECLK Output Timing Diagram

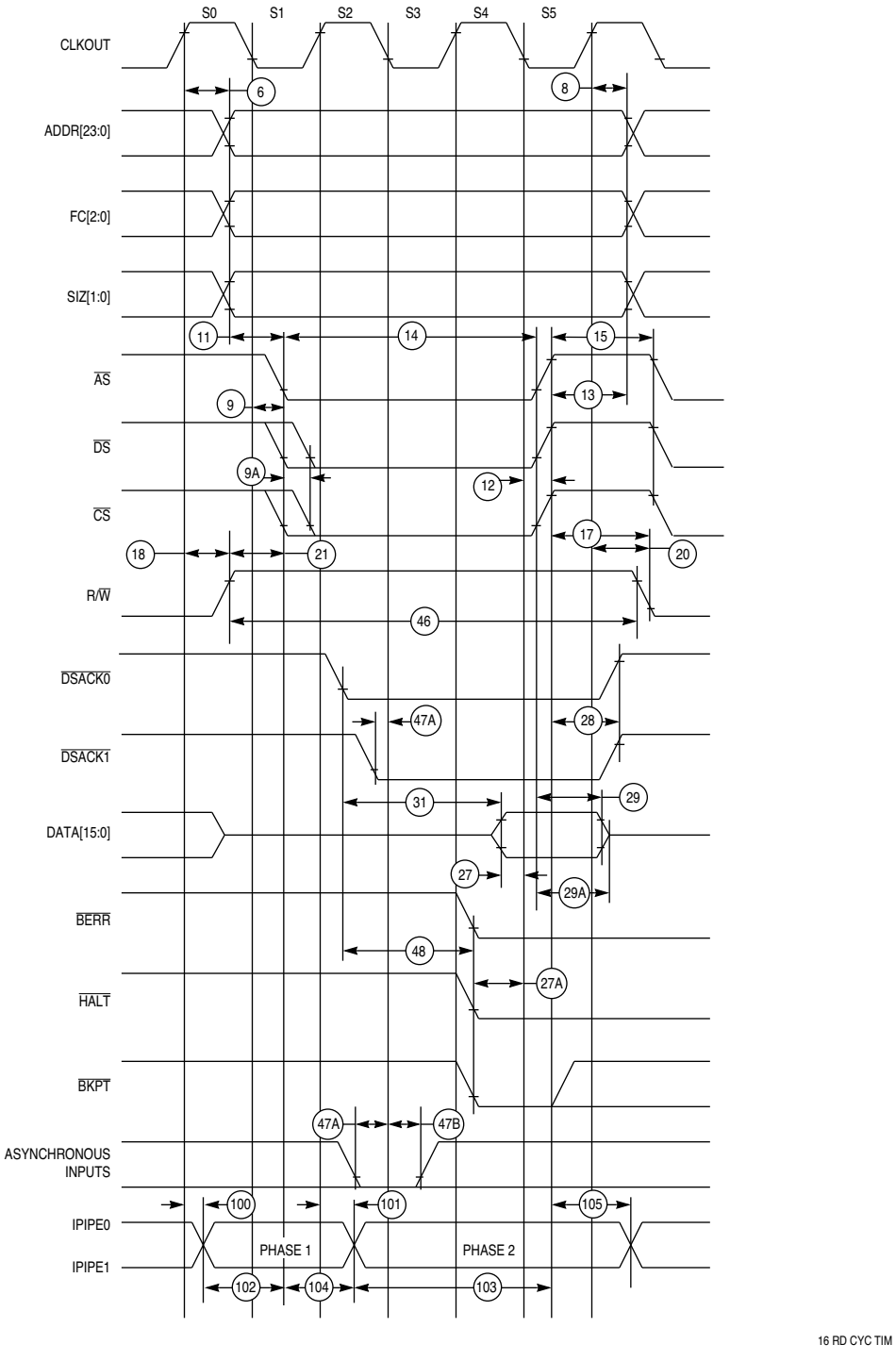


Figure 17 Read Cycle Timing Diagram

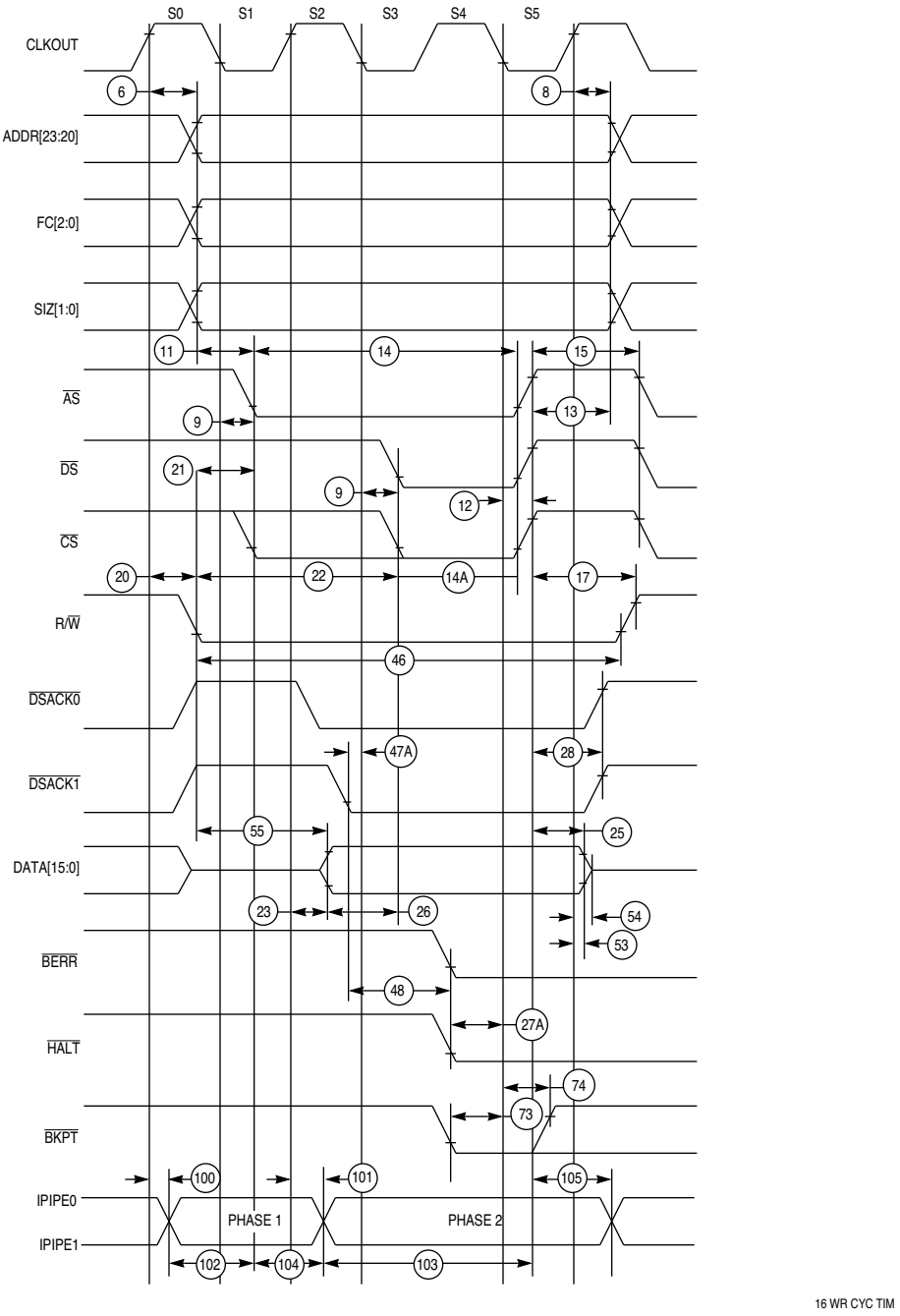


Figure 18 Write Cycle Timing Diagram

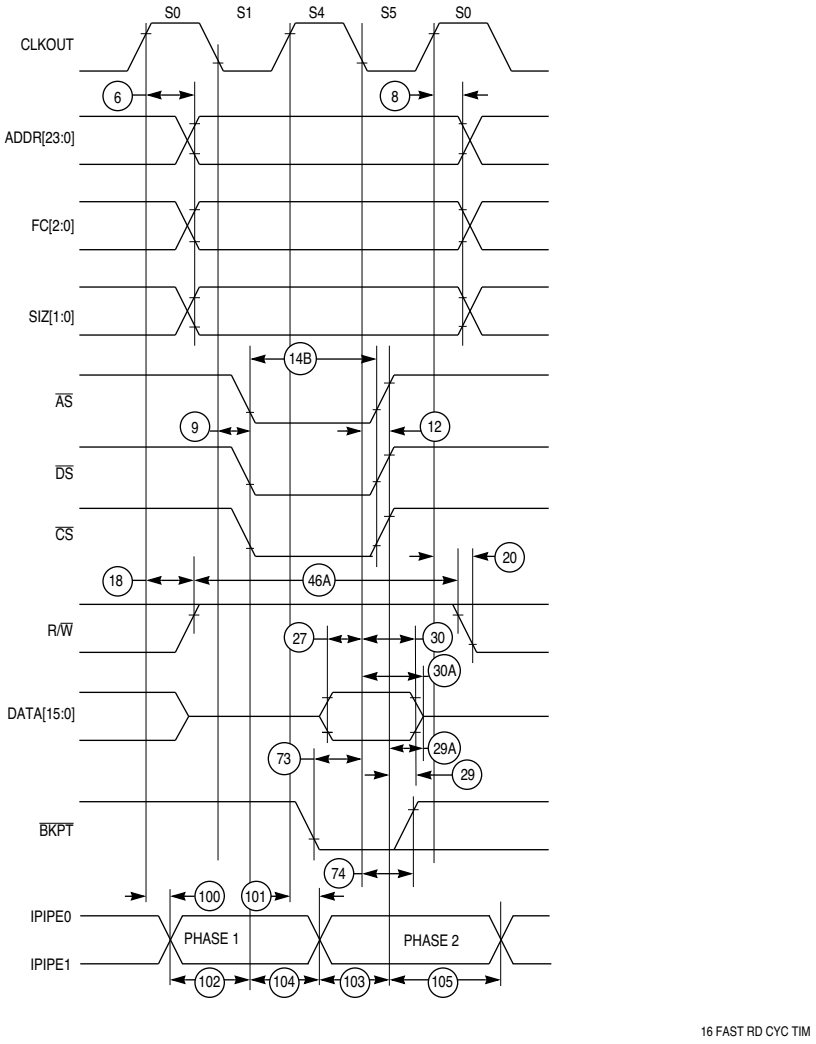


Figure 19 Fast Termination Read Cycle Timing Diagram

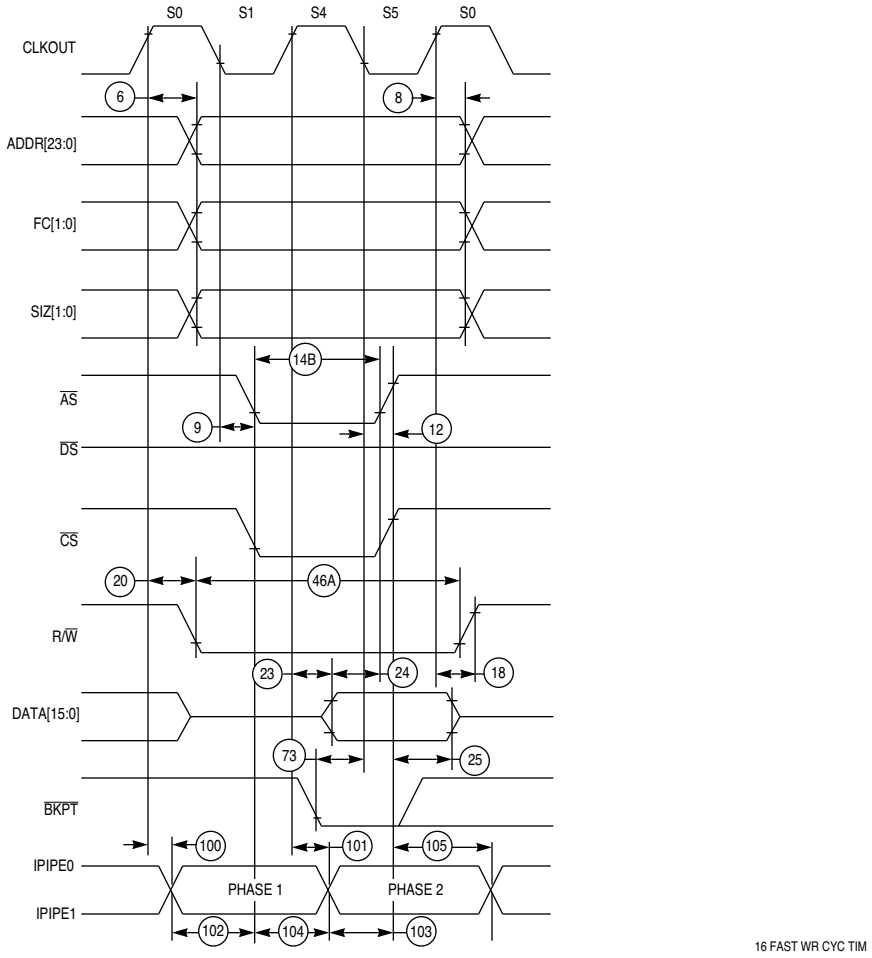


Figure 20 Fast Termination Write Cycle Timing Diagram

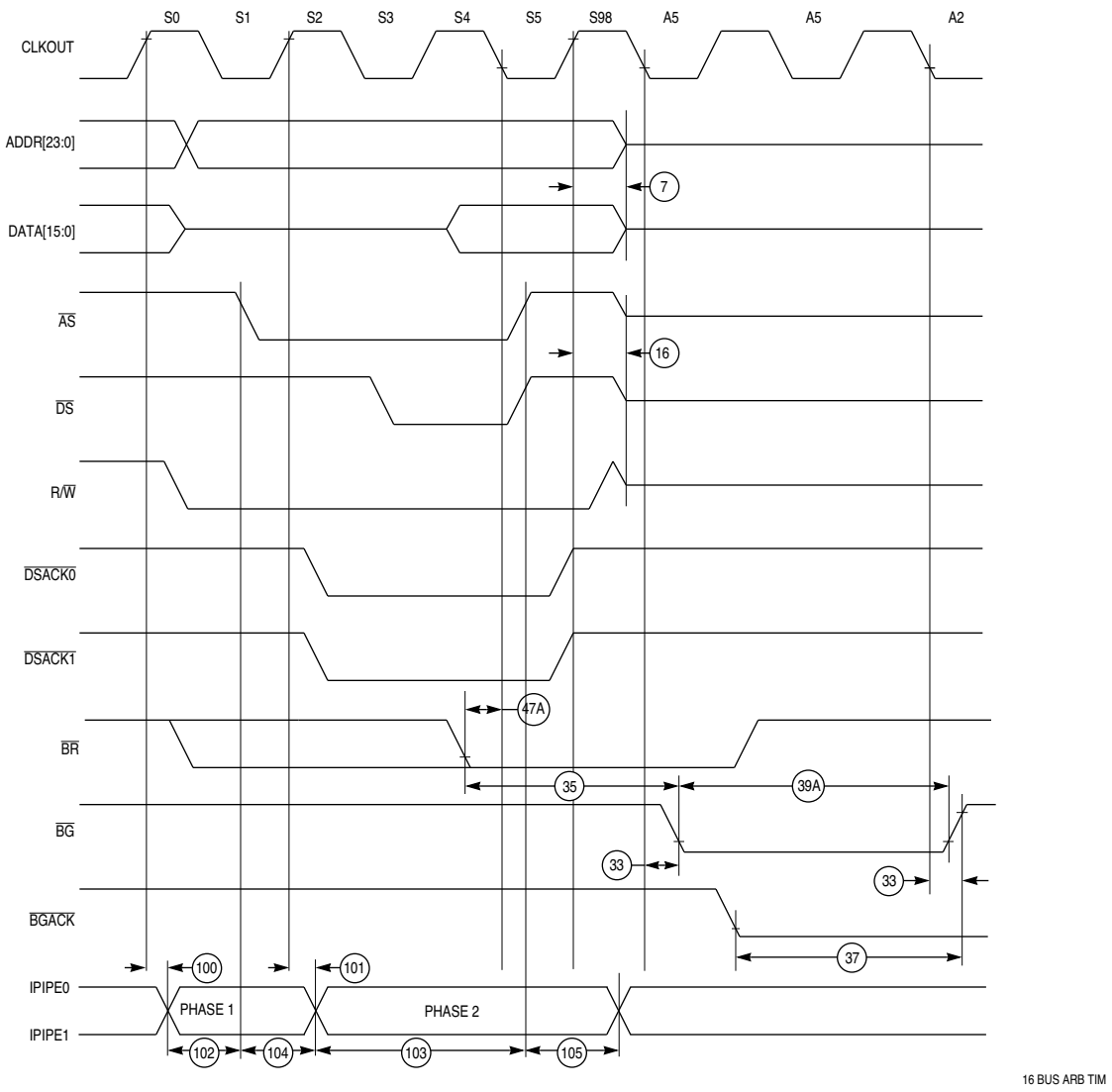


Figure 21 Bus Arbitration Timing Diagram — Active Bus Case

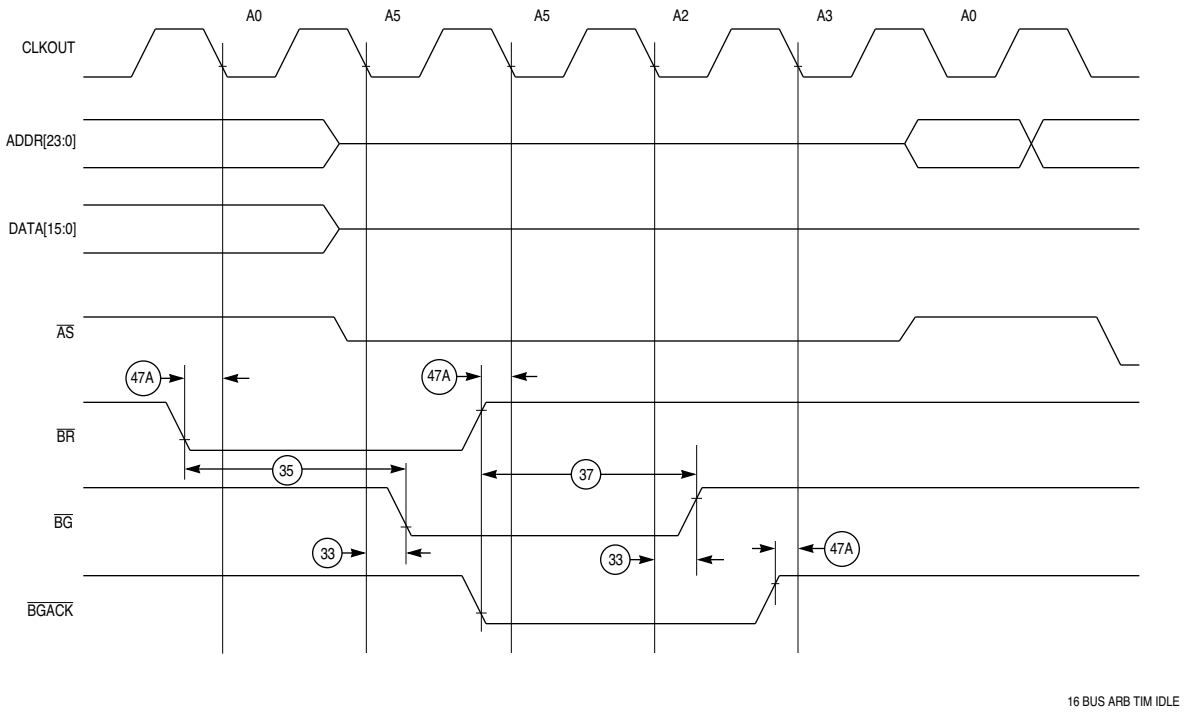
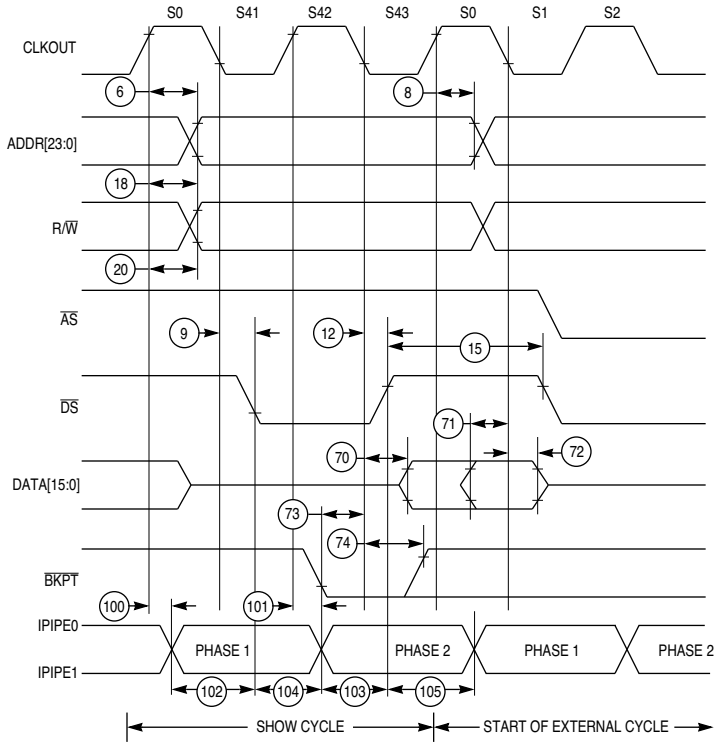


Figure 22 Bus Arbitration Timing Diagram — Idle Bus Case



NOTE:
Show cycles can stretch during clock phase S42 when bus accesses take longer than two cycles due to IMB module wait-state insertion.

16 SHW CYC TIM

Figure 23 Show Cycle Timing Diagram

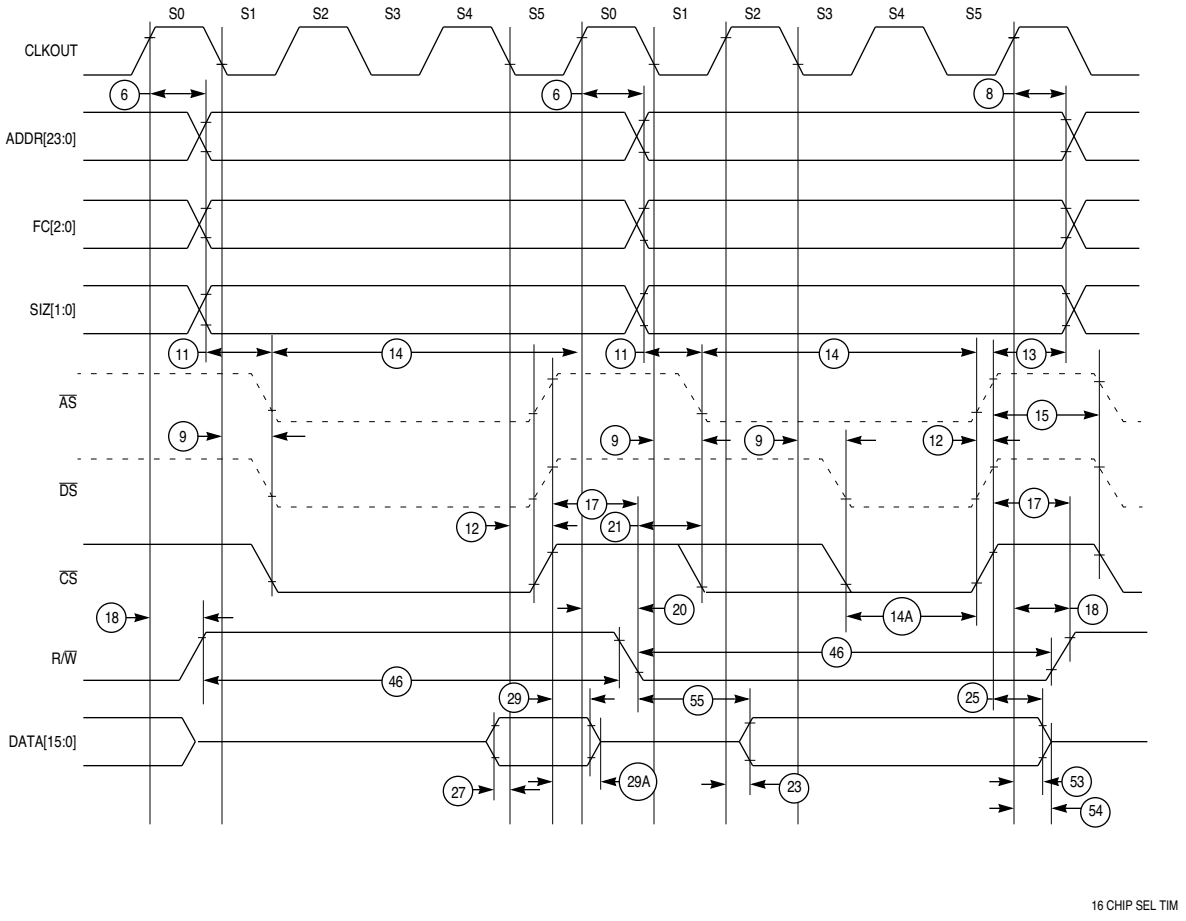


Figure 24 Chip-Select Timing Diagram

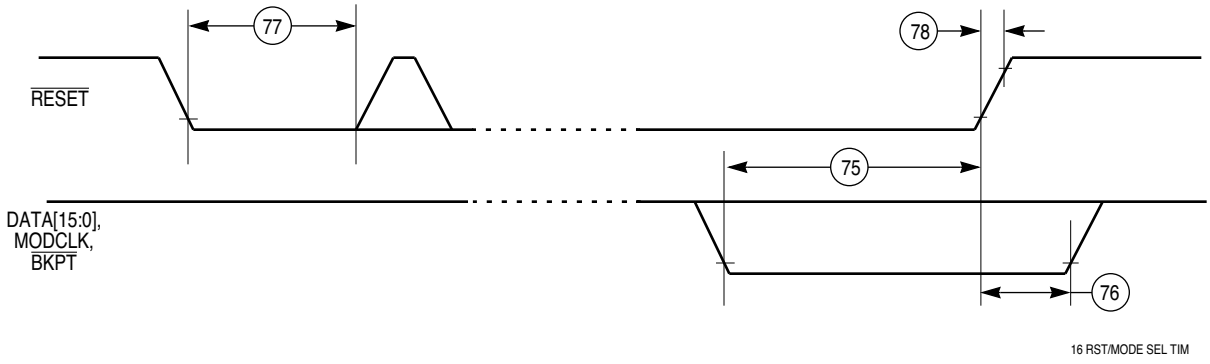


Figure 25 Reset and Mode Select Timing Diagram

Table 48 20.97 MHz Background Debugging Mode Timing

(V_{DD} and V_{DDSYN} = 5.0 Vdc ±10%, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H)[1]

Num	Characteristic	Symbol	Min	Max	Unit
B0	DSI Input Setup Time	t_{DSISU}	15	—	ns
B1	DSI Input Hold Time	t_{DSIH}	10	—	ns
B2	DSCLK Setup Time	t_{DSCSU}	15	—	ns
B3	DSCLK Hold Time	t_{DSCH}	10	—	ns
B4	DSO Delay Time	t_{DSOD}	—	25	ns
B5	DSCLK Cycle Time	t_{DSCCYC}	2	—	t_{cyc}
B6	CLKOUT High to FREEZE Asserted/Negated	t_{FRZAN}	—	50	ns
B7	CLKOUT High to IPIPE1 High Impedance	t_{IFZ}	—	50	ns
B8	CLKOUT High to IPIPE1 Valid	t_{IF}	—	50	ns
B9	DSCLK Low Time	t_{DSCLO}	1	—	t_{cyc}
B10	IPIPE1 High Impedance to FREEZE Asserted	t_{IPFA}	TBD	—	t_{cyc}
B11	FREEZE Negated to IPIPE[1:0] Active	t_{FRIP}	TBD	—	t_{cyc}

NOTES:
1. All AC timing is shown with respect to 20% V_{DD} and 70% V_{DD} levels unless otherwise noted.

Table 49 25.17 MHz Background Debugging Mode Timing

(V_{DD} and V_{DDSYN} = 5.0 Vdc ±10%, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H)[1]

Num	Characteristic	Symbol	Min	Max	Unit
B0	DSI Input Setup Time	t_{DSISU}	10	—	ns
B1	DSI Input Hold Time	t_{DSIH}	5	—	ns
B2	DSCLK Setup Time	t_{DSCSU}	10	—	ns
B3	DSCLK Hold Time	t_{DSCH}	5	—	ns
B4	DSO Delay Time	t_{DSOD}	—	20	ns
B5	DSCLK Cycle Time	t_{DSCCYC}	2	—	t_{cyc}
B6	CLKOUT High to FREEZE Asserted/Negated	t_{FRZAN}	—	20	ns
B7	CLKOUT High to IPIPE1 High Impedance	t_{IFZ}	—	20	ns
B8	CLKOUT High to IPIPE1 Valid	t_{IF}	—	20	ns
B9	DSCLK Low Time	t_{DSCLO}	1	—	t_{cyc}
B10	IPIPE1 High Impedance to FREEZE Asserted	t_{IPFA}	TBD	—	t_{cyc}
B11	FREEZE Negated to IPIPE[1:0] Active	t_{FRIP}	TBD	—	t_{cyc}

NOTES:
1. All AC timing is shown with respect to 20% V_{DD} and 70% V_{DD} levels unless otherwise noted.

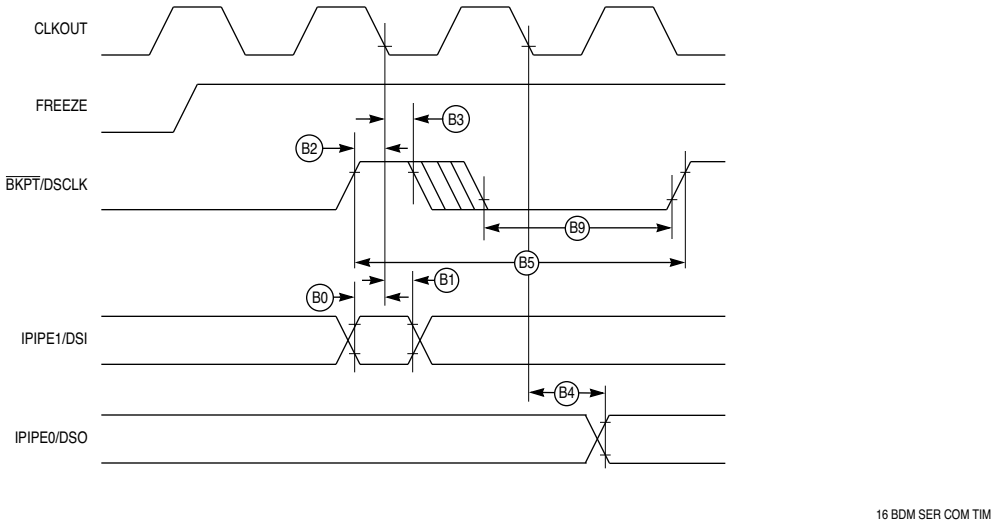


Figure 26 Background Debugging Mode Timing Diagram — Serial Communication

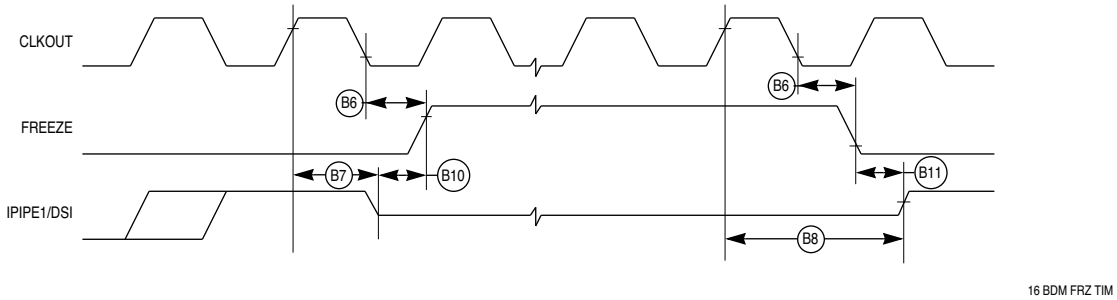


Figure 27 Background Debugging Mode Timing Diagram — Freeze Assertion

Table 50 20.97 MHz ECLK Bus Timing

(V_{DD} and V_{DDSYN} = 5.0 Vdc ± 5%, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H)[1]

Num	Characteristic	Symbol	Min	Max	Unit
E1	ECLK Low to Address Valid[2]	t_{EAD}	—	48	ns
E2	ECLK Low to Address Hold	t_{EAH}	10	—	ns
E3	ECLK Low to $\overline{CS}$ Valid ($\overline{CS}$ Delay)	t_{ECSD}	—	120	ns
E4	ECLK Low to $\overline{CS}$ Hold	t_{ECSH}	10	—	ns
E5	$\overline{CS}$ Negated Width	t_{ECSN}	25	—	ns
E6	Read Data Setup Time	t_{EDSR}	25	—	ns
E7	Read Data Hold Time	t_{EDHR}	5	—	ns
E8	ECLK Low to Data High Impedance	t_{EDHZ}	—	48	ns
E9	$\overline{CS}$ Negated to Data Hold (Read)	t_{ECDH}	0	—	ns
E10	$\overline{CS}$ Negated to Data High Impedance	t_{ECDZ}	—	1	t_{cyc}
E11	ECLK Low to Data Valid (Write)	t_{EDDW}	—	2	t_{cyc}
E12	ECLK Low to Data Hold (Write)	t_{EDHW}	10	—	ns
E13	Address Access Time (Read)[3]	t_{EACC}	308	—	ns
E14	Chip-Select Access Time (Read)[4]	t_{EACS}	236	—	ns
E15	Address Setup Time	t_{EAS}	1/2	—	t_{cyc}

NOTES:

1. All AC timing is shown with respect to 20% V_{DD} and 70% V_{DD} levels unless otherwise noted.
2. When previous bus cycle is not an ECLK cycle, the address may be valid before ECLK goes low.
3. Address access time = $t_{Ecyc} - t_{EAD} - t_{EDSR}$.
4. Chip select access time = $t_{Ecyc} - t_{ECSD} - t_{EDSR}$.

Table 51 25.17 MHz ECLK Bus Timing

(V_{DD} and V_{DDSYN} = 5.0 Vdc ± 5%, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H)[1]

Num	Characteristic	Symbol	Min	Max	Unit
E1	ECLK Low to Address Valid[2]	t_{EAD}	—	40	ns
E2	ECLK Low to Address Hold	t_{EAH}	10	—	ns
E3	ECLK Low to $\overline{CS}$ Valid ($\overline{CS}$ Delay)	t_{ECSD}	—	100	ns
E4	ECLK Low to $\overline{CS}$ Hold	t_{ECSH}	10	—	ns
E5	$\overline{CS}$ Negated Width	t_{ECSN}	20	—	ns
E6	Read Data Setup Time	t_{EDSR}	25	—	ns
E7	Read Data Hold Time	t_{EDHR}	5	—	ns
E8	ECLK Low to Data High Impedance	t_{EDHZ}	—	40	ns
E9	$\overline{CS}$ Negated to Data Hold (Read)	t_{ECDH}	0	—	ns
E10	$\overline{CS}$ Negated to Data High Impedance	t_{ECDZ}	—	1	t_{cyc}
E11	ECLK Low to Data Valid (Write)	t_{EDDW}	—	2	t_{cyc}
E12	ECLK Low to Data Hold (Write)	t_{EDHW}	5	—	ns
E13	Address Access Time (Read)[3]	t_{EACC}	255	—	ns
E14	Chip-Select Access Time (Read)[4]	t_{EACS}	195	—	ns
E15	Address Setup Time	t_{EAS}	—	1/2	t_{cyc}

NOTES:

1. All AC timing is shown with respect to 20% V_{DD} and 70% V_{DD} levels unless otherwise noted.
2. When previous bus cycle is not an ECLK cycle, the address may be valid before ECLK goes low.
3. Address access time = $t_{Ecyc} - t_{EAD} - t_{EDSR}$.
4. Chip select access time = $t_{Ecyc} - t_{ECSD} - t_{EDSR}$.

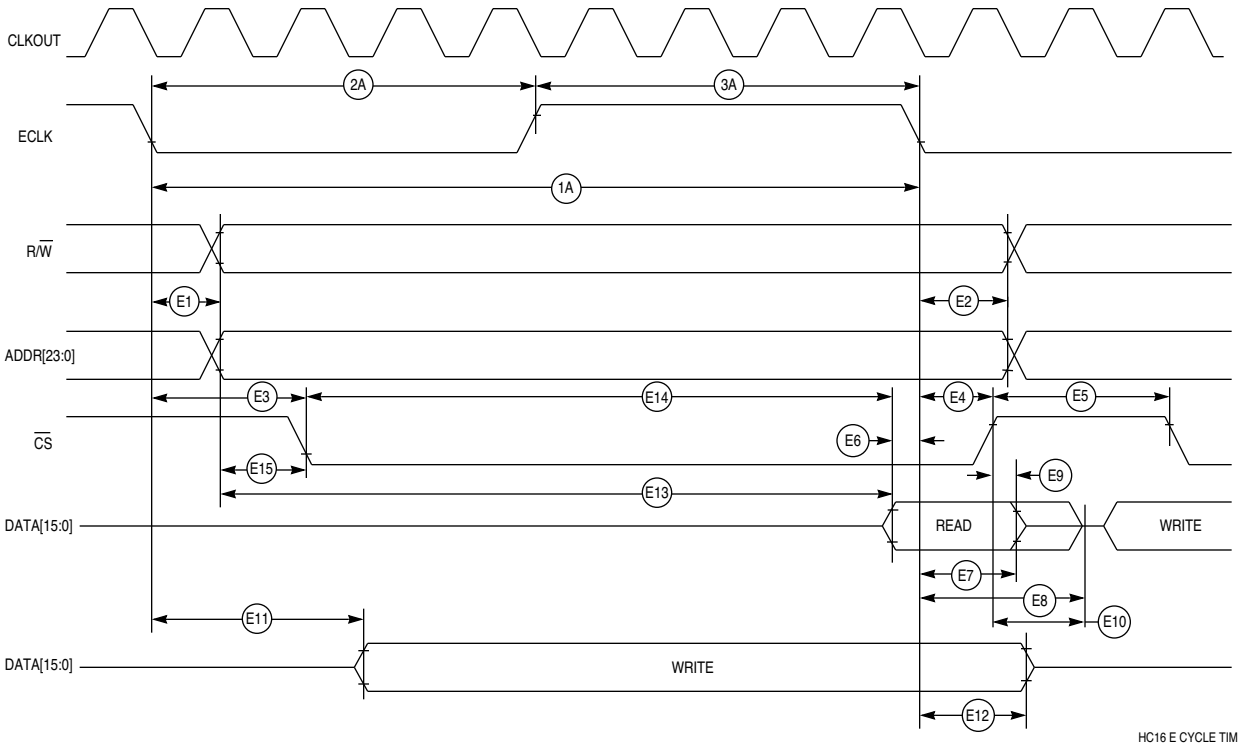


Figure 28 ECLK Timing Diagram

How to reach us:
USA/EUROPE/Locations Not Listed: Motorola Literature Distribution;
P.O. Box 20912; Phoenix, Arizona 85036. 1-800-441-2447 or 602/303-5454
MFAX: RMFAX0@email.sps.mot.com - TOUCHTONE (602) 244-6609
INTERNET: http://Design-NET.com
JAPAN: Nippon Motorola Ltd.; Tatsumi-SPD-JLDC,
6F Seibu-Butsuryu-Center, 3-14-2 Tatsumi Koto-Ku, Tokyo 135, Japan. 03-3521-8315
ASIA PACIFIC: Motorola Semiconductors H.K. Ltd.; 8B Tai Ping Industrial Park,
51 Ting Kok Road, Tai Po, N.T., Hong Kong. 852-26629298

MOTOROLA

MC68HC16S2TS/D