MC68HC916X1

Technical Summary
16-Bit Modular Microcontroller

1 Introduction

The MC68HC916X1 microcontroller (MCU) is a high-speed 16-bit device that is upwardly code compatible with M68HC11 controllers. It is a member of the M68300/68HC16 Family of modular microcontrollers.

M68HC16 controllers are built up from standard modules that interface via a common intermodule bus (IMB). Standardization facilitates rapid development of devices tailored for specific applications.

The MC68HC916X1 incorporates a true 16-bit CPU (CPU16), a single-chip integration module (SCIM), an 8/10-bit analog-to-digital converter (ADC), a queued serial module (QSM), a general-purpose timer (GPT), 2 Kbytes of standby RAM (SRAM), 48 Kbytes of flash EEPROM (FLASH), and 2 Kbytes of block-erasable flash EEPROM (BEFLASH). These modules are interconnected by the intermodule bus (IMB).

The maximum system clock for the MC68HC916X1 is 16.78 MHz. A phase-locked loop circuit synthesizes the clock from a frequency reference. Either a crystal with a nominal frequency of 4.194 MHz or an externally generated signal can be used. System hardware and software support changes in the clock rate during operation. Because the MC68HC916X1 is a fully static design, 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 MC68HC916X1 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 Ordering Information

Package Type	Frequency	Temperature	Package Order Quantity	Order Number
120 QFP	16.78 MHz	– 40 to + 85 °C	250	MC916X1CTH16B1
			50	MC68HC916X1CTH16
			2	SPMC916X1CTH16

MOTOROLA

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

1.1 Features

- CPU16
  - 16-bit architecture
  - Full set of 16-bit instructions
  - Three 16-bit index registers
  - Two 16-bit accumulators
  - 16-bit multiply and accumulate (digital signal processing support)
  - High-level language support
  - Fast interrupt response time
  - Hardware breakpoint signal/Background debugging mode
  - Fully static implementation
- Single-Chip Integration Module (SCIM)
  - Single-chip or expanded modes of operation
  - External bus support in expanded mode
  - Five programmable chip-select outputs
  - Watchdog timer, clock monitor, and bus monitor
  - Address and data bus provide 33 discrete I/O lines in single-chip mode
  - Phase-locked loop (PLL) clock system
- 8/10-Bit Analog-to-Digital Converter (ADC)
  - Six channels, eight result registers
  - Eight automated modes
  - Three result alignment formats
- Queued Serial Module (QSM)
  - Enhanced serial communication interface (SCI)
  - Queued serial peripheral interface (QSPI)
  - Dual function I/O ports
- General-Purpose Timer (GPT)
  - Two 16-bit free-running counters with prescaler
  - Three input capture channels
  - Four output compare channels
  - One input capture/output compare channel
  - One pulse accumulator/event counter input
  - Two pulse-width modulation outputs
  - Optional external clock input
- 2 Kbyte Standby RAM Module (SRAM)
  - External standby voltage supply input for low-power standby operation
- Flash EEPROM (16 and 32 Kbyte modules)
  - Bulk erase mode
  - Can provide a contiguous 48 Kbyte address space
- 2 Kbyte Flash EEPROM with Independently Erasable Blocks (BEFLASH)
  - Bulk/block erase and byte/word programming with 12 volt external input

1.2 Block Diagram
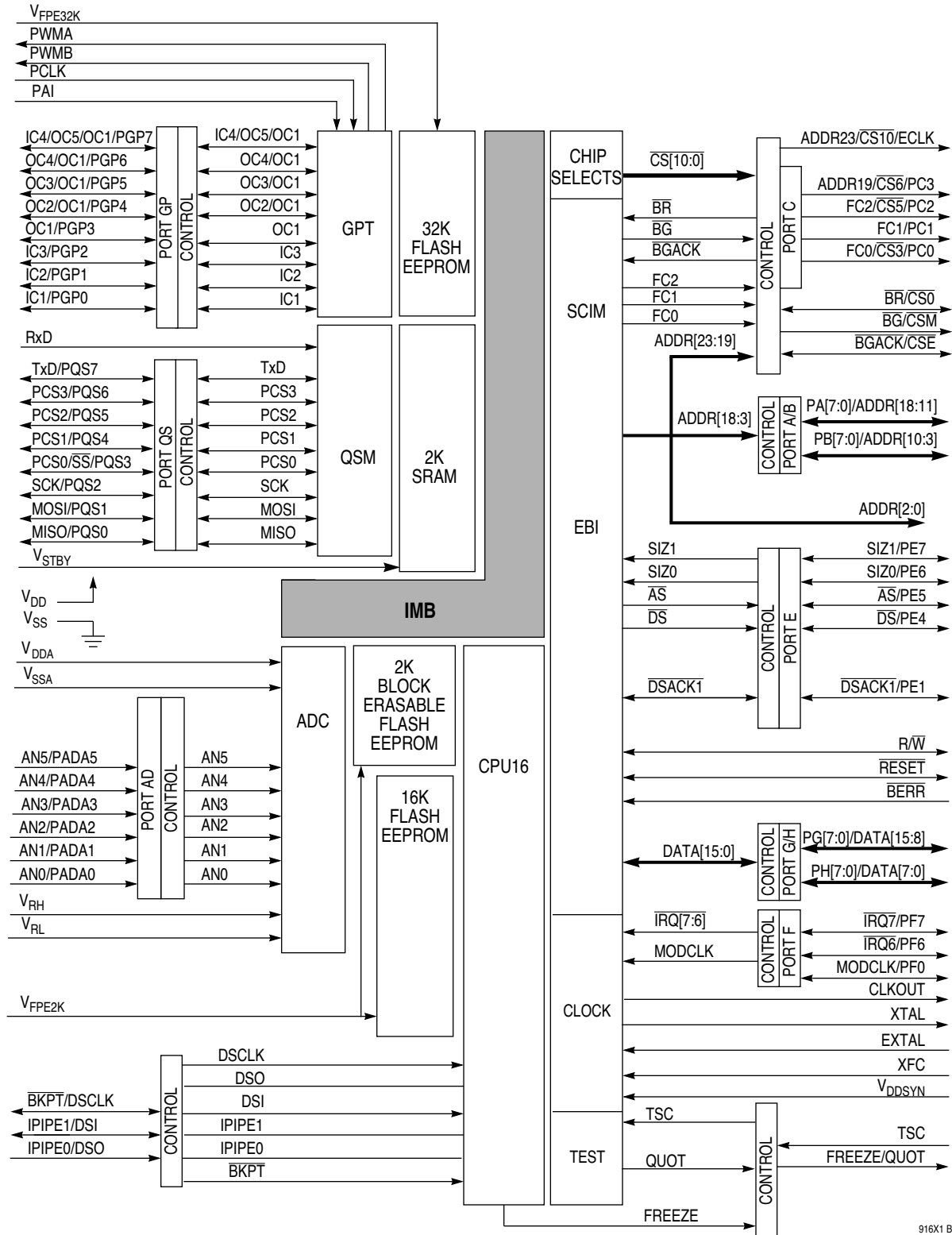
Figure 1 MC68HC916X1 Block Diagram

1.3 Pin Assignments
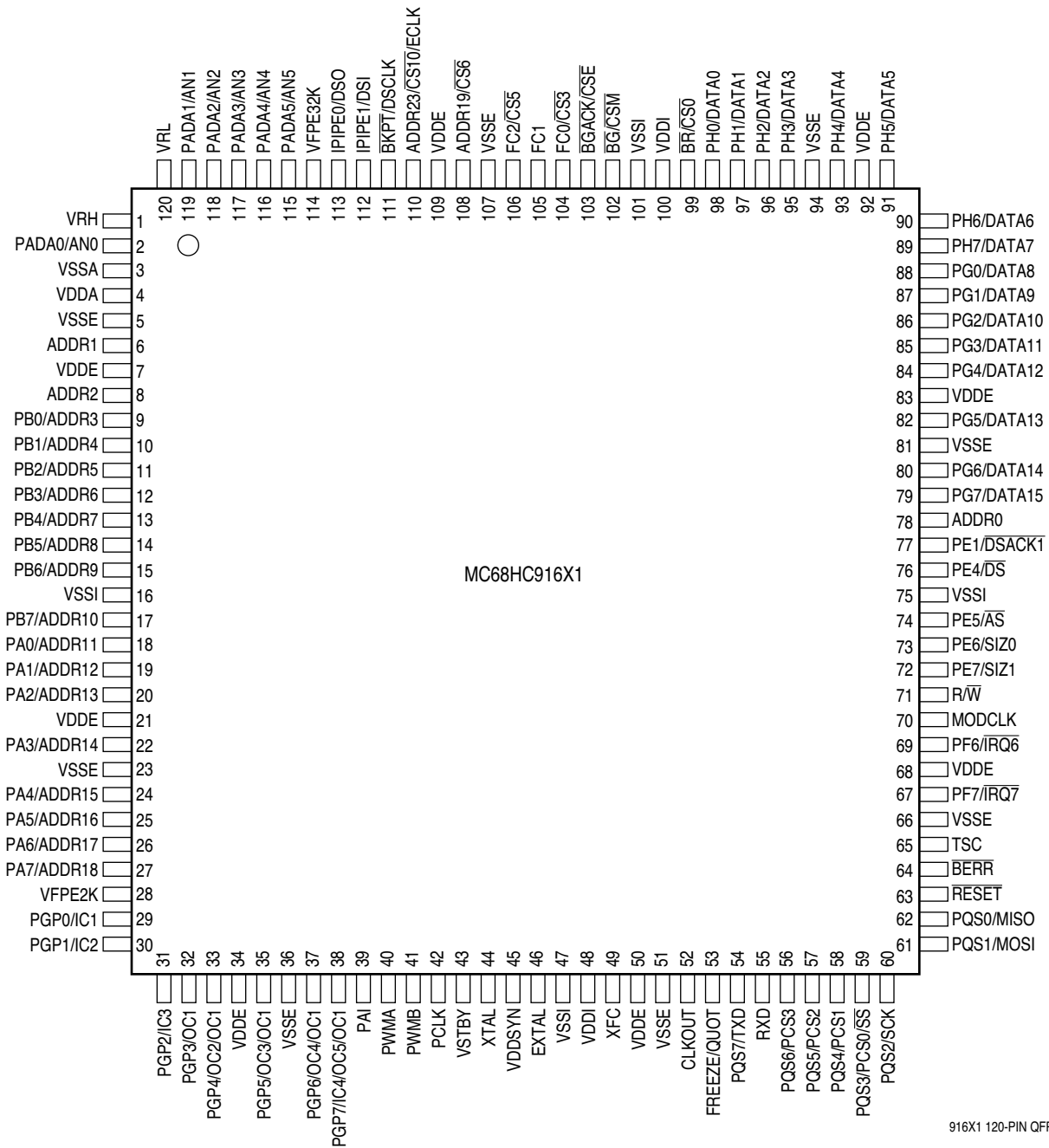
Figure 2 MC68HC916X1 Pin Assignments

1.4 Address Map

Figure 3 is a map of the MCU internal addresses. Although there are 24 IMB address lines, the CPU16 uses only ADDR[19:0]. ADDR[23:20] are driven to the same logic state as ADDR19. Addresses $080000 to $F7FFFF are not accessible. The 2-Kbyte SRAM, 16-Kbyte and 32-Kbyte flash EEPROM arrays are positioned by the base address registers in their respective control blocks. Reset disables the SRAM array. Unimplemented blocks are mapped externally.
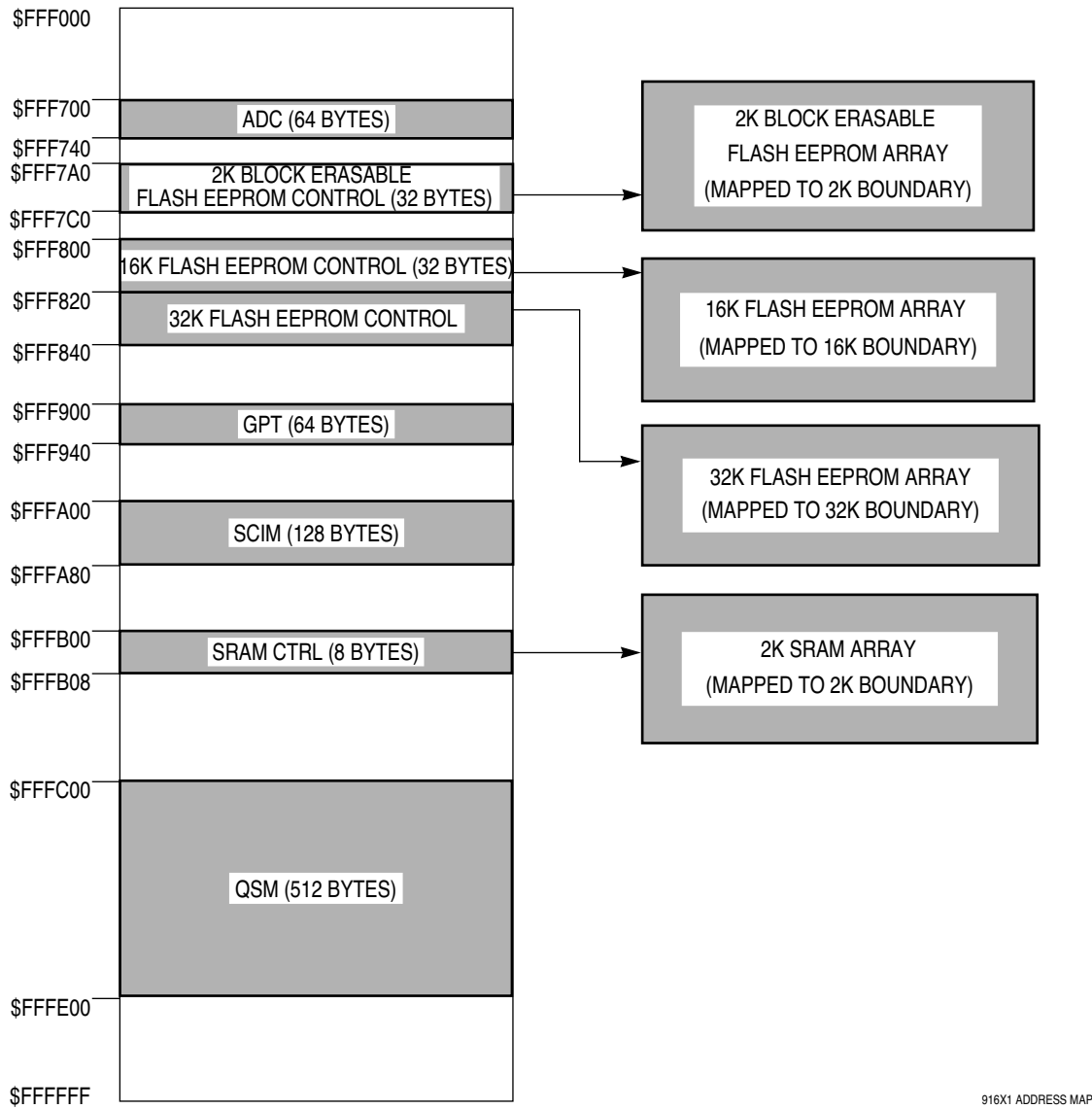
Figure 3 MC68HC916X1 Address Map

In the address map, Y = M111, where M reflects the state of the module mapping (MM) bit in the SCIM module configuration register (SCIMCR). On M68HC16 microcontrollers, Y must equal $F. If M equals 0, IMB modules become inaccessible until a reset occurs. The SCIMCR MM bit can be written only once after reset.

1.5 Intermodule Bus

The IMB is a standardized bus developed to facilitate design of modular microcontrollers. It contains circuitry that supports exception processing, address space partitioning, multiple interrupt levels, and vectored interrupts. The standardized modules in the MCU communicate with one another and with external components via the IMB. Although the full IMB supports 24 address and 16 data lines, this MCU uses only 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 **Figure 1** for information about port organization.

Table 2 MCU Pin Characteristics

Pin Mnemonic	Output Driver	Input Synchronized	Input Hysteresis	Discrete I/O	Port Designation
ADDR23/$\overline{\text{CS10}}$/ECLK	A	Y	N	—	—
ADDR19/$\overline{\text{CS6}}$	A	Y	N	O	PC3
ADDR[18:11]	A	Y	Y	I/O	PA[7:0]
ADDR[10:3]	A	Y	Y	I/O	PB[7:0]
ADDR[2:0]	A	—	—	—	—
AN[5:0][1]	—	Y	Y	I	PADA[5:0]
$\overline{\text{AS}}$	B	Y	Y	I/O	PE5
$\overline{\text{BERR}}$[2]	B	Y	N	—	—
$\overline{\text{BG}}$/$\overline{\text{CSM}}$	B	—	—	—	—
$\overline{\text{BGACK}}$/$\overline{\text{CSE}}$	B	Y	N	—	—
$\overline{\text{BKPT}}$/DSCLK	—	Y	N	—	—
$\overline{\text{BR}}$/$\overline{\text{CS0}}$	B	Y	N	—	—
CLKOUT	A	—	—	—	—
DATA[15:8][1]	Aw	Y	Y	I/O	PG[7:0]
DATA[7:0][1]	Aw	Y	Y	I/O	PH[7:0]
$\overline{\text{DS}}$	B	Y	Y	I/O	PE4
$\overline{\text{DSACK1}}$	B	Y	N	I/O	PE1
EXTAL[3]	—	—	—	—	—
FC[2:0]/$\overline{\text{CS5}}$, $\overline{\text{CS3}}$	A	—	—	O	PC[2:0]
FREEZE/QUOT	A	—	—	—	—
IC4/OC5	A	Y	Y	I/O	PGP7
IC[3:1]	A	Y	Y	I/O	PGP[2:0]
IPIPE1/DSI	A	Y	Y	—	—
IPIPE0/DSO	A	—	—	—	—
$\overline{\text{IRQ}}$[7:6]	B	Y	Y	I/O	PF[7:6]
MISO	Bo	Y	Y	I/O	PQS6
MODCLK[1]	B	Y	Y	I/O	PF0
MOSI	Bo	Y	Y	I/O	PQS7
OC[4:1]	A	Y	Y	I/O	PGP[6:3]
PAI[4]	—	Y	Y	I	—
PCLK[4]	—	Y	Y	I	—
PCS0/$\overline{\text{SS}}$	Bo	Y	Y	I/O	PQS1
PCS[3:1]	Bo	Y	Y	I/O	PQS[4:2]
PWMA, PWMB[5]	A	Y	Y	O	—
R/$\overline{\text{W}}$	A	—	—	—	—

Table 2 MCU Pin Characteristics (Continued)

Pin Mnemonic	Output Driver	Input Synchronized	Input Hysteresis	Discrete I/O	Port Designation
RESET	Bo	Y	Y	—	—
RXD	Bo	Y	Y	I/O	—
SCK	Bo	Y	Y	I/O	PQS5
SIZ[1:0]	B	Y	N	I/O	PE[7:6]
TSC	—	Y	Y	—	—
TXD	Bo	Y	Y	I/O	PQS0
XFC[3]	—	—	—	—	—
XTAL[3]	—	—	—	—	—

1. DATA[15:0] are synchronized during reset only. MODCLK and ADC pins are synchronized only if used as input port pins.
2. BERR only synchronized if late BERR.
3. EXTAL, XFC, and XTAL are clock reference connections.
4. PAI and PCLK can be used for discrete input, but are not part of an I/O port.
5. PWMA and PWMB can be used for discrete output, but are not part of an I/O port.

2.2 MCU Power Connections

Table 3 MCU Power Connections

Pin	Description
V_{DDA}/V_{SSA}	A/D converter power
V_{SSSYN}	Clock synthesizer power
V_{DDE}, V_{SSE}	External periphery power (source and drain)
V_{DDI}, V_{SSI}	Internal module power (source and drain)
V_{STBY}	Standby RAM power/clock synthesizer power
V_{FPE2K}	2K block erasable flash (shared with 16K flash EEPROM array) program/erase power
V_{FPE32K}	32K flash EEPROM array program/erase power

2.3 MCU 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 MCU Signal Characteristics

Table 5 MCU Signal Characteristics

Signal Name	MCU Module	Signal Type	Active State
ADDR[19:0]	SCIM	Bus	—
AN[5:0]	ADC	Input	—
$\overline{AS}$	SCIM	Output	0
$\overline{BERR}$	SCIM	Input	0
$\overline{BG}$	SCIM	Output	0
$\overline{BGACK}$	SCIM	Input	0
$\overline{BKPT}$	CPU16	Input	0
$\overline{BR}$	SCIM	Input	0
CLKOUT	SCIM	Output	—
$\overline{CS10}$, $\overline{CS[6:5]}$, $\overline{CS3}$, $\overline{CS0}$	SCIM	Output	0
$\overline{CSE}$	SCIM	Output	0
$\overline{CSM}$	SCIM	Output	0
DATA[15:0]	SCIM	Bus	—
$\overline{DS}$	SCIM	Output	0
$\overline{DSACK1}$	SCIM	Input	0
DSCLK	CPU16	Input	Serial Clock
DSI	CPU16	Input	Serial Data
DSO	CPU16	Output	Serial Data
ECLK	CPU16	Output	—
EXTAL	CPU16	Input	—
FREEZE	SCIM	Output	1
IC[4:1]	GPT	Input	—
IPIPE0	CPU16	Output	—
IPIPE1	CPU16	Output	—
$\overline{IRQ[7:6]}$	SCIM	Input	0
MISO	QSM	Input/Output	Serial Data
MODCLK	SCIM	Input	—
MOSI	QSM	Input/Output	Serial Data
OC[5:1]	GPT	Output	—
PAI	GPT	Input	—
PCLK	GPT	Input	—
PCS0/$\overline{SS}$	QSM	Input	0
PCS[3:1]	QSM	Input	0
PWMA, PWMB	GPT	Output	—
QUOT	SCIM	Output	—
R/$\overline{W}$	SCIM	Output	—
$\overline{RESET}$	SCIM	Input/Output	0
RXD	QSM	Input	—
SCK	QSM	Input/Output	—
SIZ[1:0]	SCIM	Output	—
TSC	SCIM	Input	1
TXD	QSM	Input/Output	—
XFC	SCIM	Input	—
XTAL	SCIM	Output	—

2.5 MCU Signal Function

Table 6 MCU Signal Function

Signal Name	Mnemonic	Function
Address Bus	ADDR[19:0]	20-bit address bus used by CPU16
ADC Analog Input	AN[5:0]	Inputs to ADC multiplexer
Address Strobe	$\overline{\text{AS}}$	Indicates that a valid address is on the address bus
Bus Grant	$\overline{\text{BG}}$	Indicates that the MCU has relinquished the bus
Bus Grant Acknowledge	$\overline{\text{BGACK}}$	Indicates that an external device has assumed bus mastership
Bus Error	$\overline{\text{BERR}}$	Indicates that a bus error has occurred
Breakpoint	$\overline{\text{BKPT}}$	Signals a hardware breakpoint to the CPU
Bus Request	$\overline{\text{BR}}$	Indicates that an external device requires bus mastership
System Clockout	CLKOUT	System clock output
Emulation Mode Chip-Selects	$\overline{\text{CSE}}$, $\overline{\text{CSM}}$	$\overline{\text{CSE}}$ selects external emulation devices at internally-mapped addresses. It is used to emulate I/O ports. $\overline{\text{CSM}}$ has no function on the MC68HC916X1. It is driven high if the SCIM is configured for emulation mode.
Chip-Selects	$\overline{\text{CS10}}$, $\overline{\text{CS[6:5]}}$, $\overline{\text{CS3}}$, $\overline{\text{CS0}}$	Select external devices at programmed addresses
Data Bus	DATA[15:0]	16-bit data bus
Data Strobe	$\overline{\text{DS}}$	During a read cycle, indicates that an external device should place valid data on the data bus. During a write cycle, indicates that valid data is on the data bus.
Data and Size Acknowledge	$\overline{\text{DSACK1}}$	Asserted by external devices during asynchronous transfers to indicate receipt of data and width of receiving port
Development Serial In, Out, Clock	DSI, DSO, DSCLK	Serial I/O and clock for background debug mode
External Clock	ECLK	M6800 bus clock output
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 CPU16 has entered background mode
Instruction Pipeline	IPIPE[1:0]	Indicates instruction pipeline activity
Interrupt Request	$\overline{\text{IRQ[7:6]}}$	Request interrupt service from the CPU16
Master In Slave Out	MISO	Serial input to SPI in master mode; serial output from SPI in slave mode
Clock Mode Select	MODCLK	Selects the source and type of system clock
Master Out Slave In	MOSI	Serial output from SPI in master mode; serial input to SPI in slave mode
Peripheral Chip-Selects	PCS[3:0]	QSPI peripheral chip selects
Port A	PA[7:0]	Port A digital input or output signals
Port B	PB[7:0]	Port B digital input or output signals
Port C	PC[3:0]	Port C digital input/output port signals
Port E	PE1, PE[7:4]	Port E digital I/O port signals
Port F	PF0, PF[7:6]	Port F digital I/O port signals
Port G	PG[7:0]	Port G digital I/O signals
Port GP	PGP[7:0]	GPT digital I/O port signal
Port H	PH[7:0]	Port H digital I/O signal
Port QS	PQS[7:0]	QSM digital I/O port signal
Pulse Accumulator Input	PAI	Input to the GPT pulse accumulator

Table 6 MCU Signal Function (Continued)

Signal Name	Mnemonic	Function
Quotient Out	QUOT	Provides the quotient bit of the polynomial divider
Read/Write	R/$\overline{W}$	Indicates the direction of data transfer on the bus
Reset	$\overline{RESET}$	System reset
Receive Data (SCI)	RXD	Serial input to the SCI
Serial Clock (QSPI)	SCK	Clock output from QSPI in master mode; clock input to QSPI in slave mode
Size	SIZ[1:0]	Indicates the number of bytes to be transferred during a bus cycle
Slave Select	$\overline{SS}$	Causes serial transmission when QSPI is in slave mode; causes mode fault in master mode
Three-State Control	TSC	Places all output drivers in a high-impedance state
Transmit Data (SCI)	TXD	Serial output from the SCI
External Filter Capacitor	XFC	Connection for external phase-locked loop filter capacitor

3 Single-Chip Integration Module

The single-chip integration module (SCIM) consists of six submodules that, with a minimum of external devices, control system startup, initialization, configuration, and the external bus. **Figure 4** shows the SCIM block diagram.
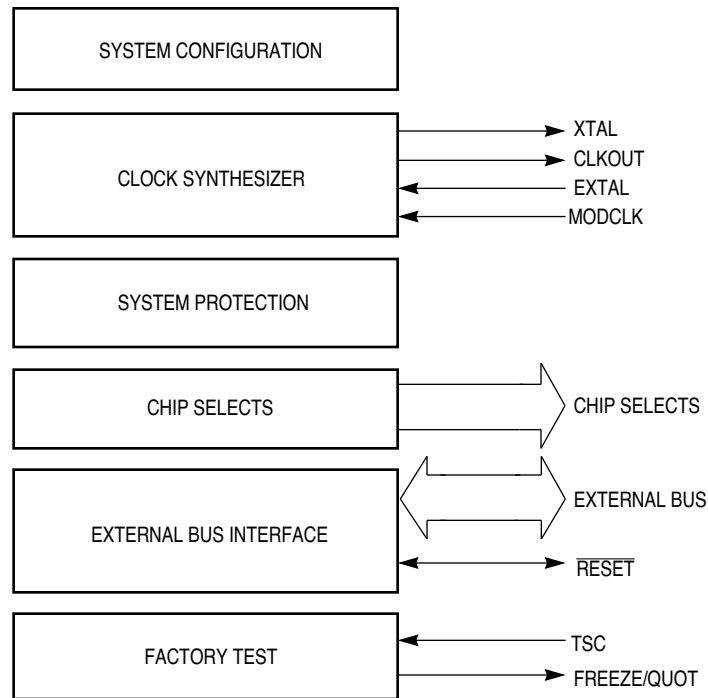
Figure 4 Single-Chip Integration Module Block Diagram

3.1 Overview

The system configuration block controls MCU configuration and operating mode.

The system clock generates clock signals used by the SCIM, 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 five general-purpose chip-select signals and two emulation-support chip-select signals. The general-purpose 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 SCIM address map, which occupies 128 bytes. Unused registers within the 128-byte address space return zeros when read.

Table 7 SCIM Address Map

Address	15 8	7 0
$YFFA00[1]	SCIM MODULE CONFIGURATION REGISTER (SCIMCR)	
$YFFA02	FACTORY TEST REGISTER (SCIMTR)	
$YFFA04	CLOCK SYNTHESIZER CONTROL REGISTER (SYNCR)	
$YFFA06	NOT USED	RESET STATUS REGISTER (RSR)
$YFFA08	MODULE TEST E (SCIMTRE)	
$YFFA0A	PORT A DATA REGISTER (PORTA)	PORT B DATA REGISTER (PORTB)
$YFFA0C	PORT G DATA REGISTER (PORTG)	PORT H DATA REGISTER (PORTH)
$YFFA0E	PORT G DATA DIRECTION (DDRG)	PORT H DATA DIRECTION (DDRH)
$YFFA10	NOT USED	PORT E DATA (PORTE0)
$YFFA12	NOT USED	PORT E DATA (PORTE1)
$YFFA14	PORT A/B DATA DIRECTION (DDRAB)	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 (PICR)	
$YFFA24	PERIODIC INTERRUPT TIMING (PITR)	
$YFFA26	NOT USED	SOFTWARE SERVICE (SWSR)
$YFFA28	NOT USED	PORT F EDGE DETECT FLAGS (PORTFE)
$YFFA2A	NOT USED	PORT F EDGE DETECT INTERRUPT VECTOR (PFIVR)
$YFFA2C	NOT USED	PORT F EDGE DETECT INTERRUPT LEVEL (PFLVR)
$YFFA2E	RESERVED	
$YFFA30	TEST MODULE MASTER SHIFT A (TSTMSRA)	
$YFFA32	TEST MODULE MASTER SHIFT B (TSTMSRB)	
$YFFA34	TEST MODULE SHIFT COUNT A (TSTSCA)	TEST MODULE SHIFT COUNT B (TSTSCB)
$YFFA36	TEST MODULE REPETITION COUNTER (TSTRC)	
$YFFA38	TEST MODULE CONTROL (CREG)	
$YFFA3A	TEST MODULE DISTRIBUTED REGISTER (DREG)	
$YFFA3C	RESERVED	
$YFFA3E	RESERVED	
$YFFA40	NOT USED	PORT C DATA (PORTC)
$YFFA42	RESERVED	
$YFFA44	CHIP-SELECT PIN ASSIGNMENT 0 (CSPAR0)	
$YFFA46	CHIP-SELECT PIN ASSIGNMENT 1 (CSPAR1)	
$YFFA48	CHIP-SELECT BASE ADDRESS BOOT (CSBARBT)	
$YFFA4A	CHIP-SELECT OPTION BOOT (CSORBT)	
$YFFA4C	CHIP-SELECT BASE 0 (CSBAR0)	
$YFFA4E	CHIP-SELECT OPTION 0 (CSOR0)	

Table 7 SCIM Address Map (Continued)

Address	15 8	7 0
$YFFA50	RESERVED	
$YFFA52	RESERVED	
$YFFA54	RESERVED	
$YFFA56	RESERVED	
$YFFA58	CHIP-SELECT BASE 3 (CSBAR3)	
$YFFA5A	CHIP-SELECT OPTION 3 (CSOR3)	
$YFFA5C	RESERVED	
$YFFA5E	RESERVED	
$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	RESERVED	
$YFFA7A	RESERVED	
$YFFA7C	RESERVED	
$YFFA7E	RESERVED	

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

3.2 System Configuration

The MCU can operate as a stand-alone device (single-chip mode), with a 20-bit external address bus and an 8-bit external data bus, or with a 20-bit external address bus and a 16-bit external data bus. SCIM pins can be configured for use as I/O ports or programmable chip select signals. Refer to **3.9 Chip Selects** and **3.10 General-Purpose Input/Output** for more information. System configuration is determined by setting bits in the SCIM configuration register (SCIMCR), and by asserting MCU pins during reset.

SCIMCR — SCIM Module Configuration Register   **$YFFA00**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EXOFF	FRZSW	FRZBM	CPUD[1]	RSVD[2]	0	SHEN		SUPV	MM	ABD[1]	RWD[1]	IARB			

RESET:

| 0 | 0 | 0 | * | 0 | 0 | 0 | 0 | 1 | 1 | * | * | 1 | 1 | 1 | 1 |

1. Reset state is mode-dependent. Refer to the following bit descriptions.
2. This bit is reserved for future use. Ensure that initialization software does not change its value (it should always read zero).

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

EXOFF — External Clock Off
    0 = The CLKOUT pin is driven from an internal clock source.
    1 = The CLKOUT pin is placed in a high-impedance state.

FRZSW — Freeze Software Watchdog Enable
    0 = When FREEZE is asserted, the software watchdog continues to run.
    1 = When FREEZE is asserted, the software watchdog is disabled.

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.

CPUD — CPU Development Support Disable
    0 = Instruction pipeline signals available on pins IPIPE0 and IPIPE1
    1 = Pins IPIPE0 and IPIPE1 placed in high-impedance state unless a breakpoint occurs
    CPUD is cleared to zero when the MCU is in an expanded mode, and set to one in single-chip mode.

SHEN[1:0] — Show Cycle Enable
    This field determines how the external bus is driven during internal transfer operations. A show cycle allows internal transfers to be monitored externally. **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 Field

SHEN	Action
00	Show cycles disabled, external arbitration enabled
01	Show cycles enabled, external arbitration disabled
10	Show cycles enabled, external arbitration enabled
11	Show cycles enabled, external arbitration enabled, 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 AD-DR[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.

ABD — Address Bus Disable
    0 = Pins ADDR[2:0] operate normally.
    1 = Pins ADDR[2:0] are disabled.
    ABD is cleared to zero when the MCU is in an expanded mode, and set to one in single-chip mode. ABD can be written only once after reset.

RWD — Read/Write Disable

        0 = R/$\overline{\text{W}}$ signal operates normally

        1 = R/$\overline{\text{W}}$ signal placed in high-impedance state.

   RWD is cleared to zero when the MCU is in an expanded mode, and set to one in single-chip mode. RWD can be written only once after reset.

IARB[3:0] — Interrupt Arbitration

   Each module that can generate interrupts, including the SCIM, has an IARB field. Each IARB field can be assigned a value from $0 to $F. During an interrupt acknowledge cycle, IARB permits arbitration among simultaneous interrupts of the same priority level. The reset value of the SCIM IARB field is $F. This prevents SCIM interrupts from being discarded. Initialization software must set the IARB field to a lower value if lower priority interrupts are to be arbitrated.

3.2.1 Operating Modes

   During reset, the SCIM configures itself according to the states of the DATA[11:0], $\overline{\text{BERR}}$, MOD-CLK, and $\overline{\text{BKPT}}$ pins. DATA[11:0] provide pin configuration information. $\overline{\text{BERR}}$, MODCLK, and $\overline{\text{BKPT}}$ determine basic operation.

   The SCIM can be configured to operate in one of three modes: 16-bit expanded, 8-bit expanded, and single chip. Operating mode is determined by the value of the DATA1 and $\overline{\text{BERR}}$ signals coming out of reset. **Table 9** shows the basic configuration options.

Table 9 Basic Configuration Options

Select Pin	Default Function (Pin Left High)	Alternate Function (Pin Pulled Low)
MODCLK	Synthesized System Clock	External System Clock
$\overline{\text{BKPT}}$	Background Mode Disabled	Background Mode Enabled
$\overline{\text{BERR}}$	Expanded Mode	Single-Chip Mode
DATA1 (if $\overline{\text{BERR}}$ = 1)	8-Bit Expanded Mode	16-Bit Expanded Mode

   $\overline{\text{BERR}}$, $\overline{\text{BKPT}}$, and MODCLK do not have internal pull-ups and must be driven to the desired state during reset.

   Operating mode determines which address and data bus lines are used and which general-purpose I/O ports are available. **Table 10** is a summary of bus and port configuration options.

Table 10 Bus and Port Configuration Options

Mode	Address Bus	Data Bus	I/O Ports
16-Bit Expanded	ADDR[18:3]	DATA[15:0]	—
8-Bit Expanded	ADDR[18:3]	DATA[15:8]	DATA[7:0] = Port H
Single Chip	None	None	ADDR[18:11] = Port A ADDR[10:3] = Port B DATA[15:8] = Port G DATA[7:0] = Port H

   Many pins on the MC68HC916X1, including data and address bus pins, have multiple functions. Reset value for these pins depends on operating mode. In expanded mode, the values of DATA[11:0] during reset determine the function of these pins. The functions of some pins can be changed by writing to the appropriate pin assignment register. Data bus pins have internal pull-ups and must be pulled low to achieve the alternate configuration desired. The following tables contain a summary of pin configuration options for each operating mode.

3.2.2 16-Bit Expanded Mode

In 16-bit expanded mode, ($\overline{\text{BERR}}$ = 1, DATA1 = 0) pins ADDR[18:3] and DATA[15:0] are configured as address and data pins, respectively. The alternate functions for these pins as ports A, B, G, and H are unavailable. **Table 11** is a summary of pin function options for 16-bit data bus operation.

Table 11 16-Bit Expanded Mode Reset Configuration

Pin(s) Affected	Select Pin	Default Function (Pin Held High)	Alternate Function (Pin Held Low)
$\overline{\text{BR}}$/$\overline{\text{CS0}}$ FC0/$\overline{\text{CS3}}$/PC0 FC1/PC1 FC2/$\overline{\text{CS5}}$/PC2	DATA2	$\overline{\text{CS0}}$ $\overline{\text{CS3}}$ FC1 $\overline{\text{CS5}}$	BR FC0 FC1 FC2
ADDR19/$\overline{\text{CS6}}$/PC3 ADDR23/$\overline{\text{CS10}}$/ECLK	DATA3[1] DATA7[1]	$\overline{\text{CS6}}$ $\overline{\text{CS10}}$	ADDR19 ADDR23
$\overline{\text{DSACK1}}$/PE1 $\overline{\text{DS}}$/PE4 $\overline{\text{AS}}$/PE5 SIZ0/PE6 SIZ1/PE7	DATA8	$\overline{\text{DSACK1}}$ $\overline{\text{DS}}$ $\overline{\text{AS}}$ SIZ0 SIZ1	PE1 PE4 PE5 PE6 PE7
MODCLK/PF0 $\overline{\text{IRQ}}$[7:6]/PF[7:6]	DATA9	MODCLK $\overline{\text{IRQ}}$[7:6]	PF0 PF[7:6]
$\overline{\text{BGACK}}$/$\overline{\text{CSE}}$ $\overline{\text{BG}}$/$\overline{\text{CSM}}$	DATA10	$\overline{\text{BGACK}}$ $\overline{\text{BG}}$	$\overline{\text{CSE}}$[2] $\overline{\text{CSM}}$[3]
Reserved	DATA11[4]	Normal Operation	Reserved
Emulation Mode (SCIM)	DATA10	Disabled	Enabled
STOP Mode (BEFLASH)	DATA15	Array Enabled[5]	Array Disabled

1. $\overline{\text{CS}}$[9:7] outputs are not available on the MC68HC916X1. Corresponding pin assignment register fields (CSPA1[3:1]) are affected by the state of DATA[6:4] during reset.
2. $\overline{\text{CSE}}$ is enabled when DATA10 and DATA1 = 0 during reset.
3. $\overline{\text{CSM}}$ is enabled when DATA13, DATA10 and DATA1 = 0 during reset.
4. DATA11 must remain high during reset to ensure normal operation of MCU.
5. Driven to put BEFLASH in STOP mode. STOP mode disabled when DATA15 is held high and STOP shadow bit is cleared.

3.2.3 8-Bit Expanded Mode

In 8-bit expanded mode ($\overline{\text{BERR}}$ = 1, DATA1 = 1), pins DATA[7:0] are configured as an 8-bit I/O port. Pins DATA[15:8] are configured as data pins. Pins ADDR[18:3] are configured as address pins. Emulator mode is always disabled. **Table 12** is a summary of pin function selections for 8-bit data bus operation.

Table 12 8-Bit Expanded Mode Reset Configuration

Pin(s) Affected	Select Pin	Default Function (Pin Held High)	Alternate Function (Pin Held Low)
$\overline{BR}$/$\overline{CS0}$ FC0/$\overline{CS3}$/PC0 FC1/PC1 FC2/$\overline{CS5}$/PC2	N/A[1]	$\overline{CS0}$ $\overline{CS3}$ FC1 $\overline{CS5}$	$\overline{CS0}$ $\overline{CS3}$ FC1 $\overline{CS5}$
ADDR19/$\overline{CS6}$/PC0 ADDR23/$\overline{CS10}$/ECLK	N/A[1]	$\overline{CS6}$ $\overline{CS10}$	$\overline{CS6}$ $\overline{CS10}$
$\overline{DSACK1}$/PE1 $\overline{DS}$/PE4 $\overline{AS}$/PE5 SIZ0/PE6 SIZ1/PE7	DATA8	$\overline{DSACK1}$ $\overline{DS}$ $\overline{AS}$ $\overline{SIZ0}$ $\overline{SIZ1}$	PE1 PE4 PE5 PE6 PE7
MODCLK/PF0 $\overline{IRQ[7:6]}$/PF[7:6]	DATA9	MODCLK $\overline{IRQ[7:6]}$	PF0 PF[7:6]
$\overline{BGACK}$/$\overline{CSE}$ $\overline{BG}$/CSM	N/A[1]	$\overline{BGACK}$ $\overline{BG}$	$\overline{BGACK}$ $\overline{BG}$

1. These pins have only one reset configuration in 8-bit expanded mode.

3.2.4 Single-Chip Mode

In single-chip mode, when $\overline{BERR}$ = 0 during reset, ADDR[18:3] are configured as 8-bit I/O ports (port A and port B) and DATA[15:0] are configured as 8-bit I/O ports (port G and port H). There is no external data bus path. Expanded mode configuration options are not available: I/O ports A, B, E, F, G, and H are always selected. $\overline{BERR}$ can be tied low permanently to select single-chip mode. **Table 13** is a summary of SCIM pin functions during single-chip operation.

Table 13 Single-Chip Mode Reset Configuration

Pin(s) Affected	Function
ADDR[18:11]	PA[7:0]
ADDR[10:3]	PB[7:0]
BR/$\overline{CS0}$	$\overline{CS0}$
FC0/$\overline{CS3}$/PC0 FC1/PC1 FC2/$\overline{CS5}$/PC2 ADDR19/$\overline{CS6}$/PC3	PC[3:0]
ADDR23/$\overline{CS10}$/ECLK	—
$\overline{DSACK1}$/PE1 $\overline{DS}$/PE4 $\overline{AS}$/PE5 SIZ0/PE6 SIZ1/PE7	PE[7:4], PE1
MODCLK/PF0 $\overline{IRQ[7:6]}$/PF[7:6]	PF0 PF[7:6]
DATA[15:8]	PG[7:0]
DATA[7:0]	PH[7:0]
$\overline{BGACK}$, $\overline{CSE}$ $\overline{BG}$/CSM	$\overline{BGACK}$ $\overline{BG}$

3.2.5 Emulation Support

The SCIM contains logic that can be used to replace on-chip ports externally. The SCIM also contains special support logic that allows external emulation of internal ROM. This emulation support feature enables the development of a single-chip application in expanded mode.

Emulator mode is a special type of 16-bit expanded operation. It is entered by holding DATA10 low, $\overline{\text{BERR}}$ high, and DATA1 low during reset. In emulator mode, all port A, B, E, G, and H data and data direction registers and the port E pin assignment register are mapped externally. Port C data, port F data and data direction registers, and port F pin assignment register are accessible normally in emulator mode.

An emulator chip select ($\overline{\text{CSE}}$) is asserted whenever any of the externally-mapped registers are addressed. The signal is asserted on the falling edge of $\overline{\text{AS}}$. The SCIM does not respond to these accesses, allowing external logic, such as a port replacement unit (PRU) to respond. Accesses to externally mapped registers require three clock cycles.

External ROM emulation is enabled by holding DATA1, DATA10, and DATA13 low during reset ($\overline{\text{BERR}}$ must be held high during reset to enable the ROM module). While ROM emulation mode is enabled, memory chip select signal $\overline{\text{CSM}}$ is asserted whenever a valid access to an address assigned to the masked ROM array is made.

The ROM module does not acknowledge IMB accesses while in emulation mode. This causes the SCIM to run an external bus cycle for each access.

NOTE

The MC68HC916X1 flash modules do not yet support the emulator mode. If ROM emulation is enabled, the $\overline{\text{CSM}}$ chip-select will be driven high at all times.

3.3 System Clock

The system clock 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 from one of two sources. An internal phase-locked loop (PLL) can synthesize the clock from a fast reference, or the clock signal can be input directly from an external frequency source. The fast reference is typically a 4.194 MHz crystal, but may be generated by sources other than a crystal. Keep these sources in mind while reading the rest of this section.

Figure 5 is a block diagram of the clock submodule.

16/32 PLL BLOCK 4M
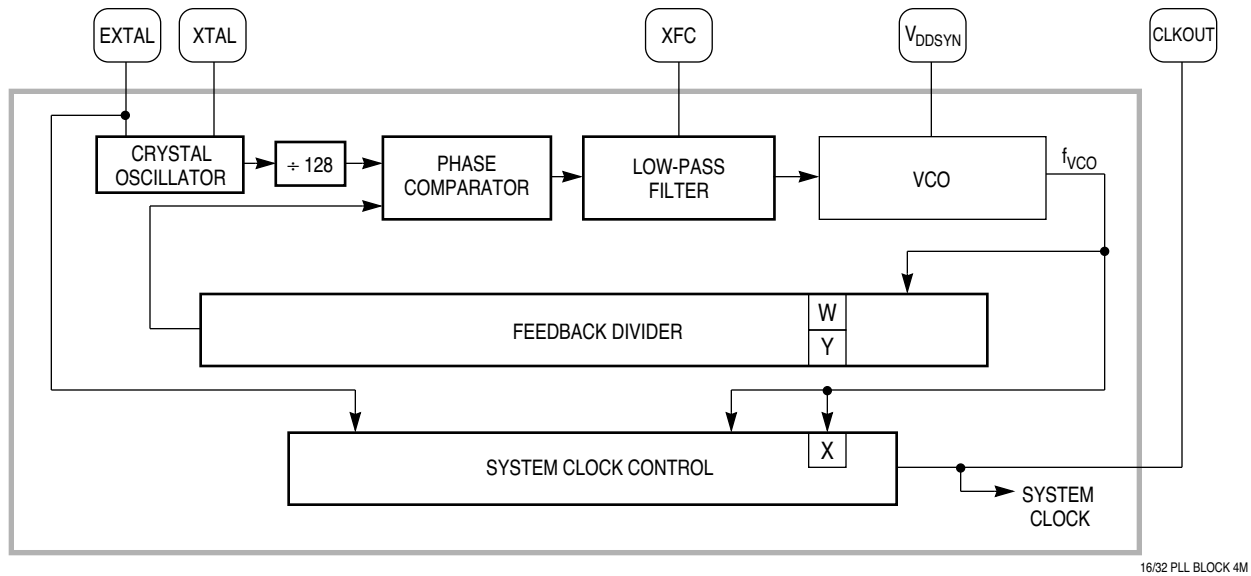
Figure 5 System Clock Block Diagram

3.3.1 Clock Sources

The state of the clock mode (MODCLK) pin during reset determines the system clock source. When MODCLK is held high during reset, the clock synthesizer generates a clock signal from an external reference frequency. 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 driven onto the EXTAL pin.

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.

To generate a reference frequency using the internal oscillator, a reference crystal must be connected between the EXTAL and XTAL pins. Typically, a 4.194 MHz crystal is used, but the frequency may vary between 1 and 6 MHz. **Figure 6** shows a recommended circuit.
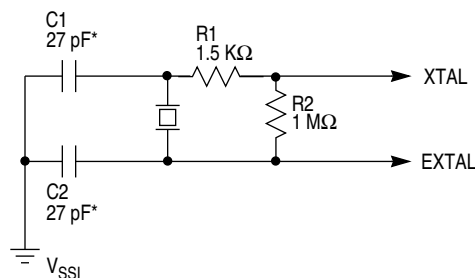
* RESISTANCE AND CAPACITANCE BASED ON A TEST CIRCUIT CONSTRUCTED WITH A KDS041-18 4.194 MHz CRYSTAL. SPECIFIC COMPONENTS MUST BE BASED ON CRYSTAL TYPE. CONTACT CRYSTAL VENDOR FOR EXACT CIRCUIT.

16 OSCILLATOR 4M

Figure 6 System Clock Oscillator Circuit

If a fast reference frequency is provided to the PLL from a source other than a crystal, or an external system clock signal is applied through the EXTAL pin, the XTAL pin must be left floating.

When an external system clock signal is applied (MODCLK = 0 during reset), the PLL is disabled. 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 as follows:

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

3.3.2 Clock Synthesizer Operation

V_{DDSYN} is used to power the clock circuits when the system clock is synthesized from either a crystal or an externally supplied reference frequency. 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_{SS} supply. Refer to the *SCIM Reference Manual* (SCIMRM/AD) for more information regarding system clock power supply conditioning.

A voltage controlled oscillator (VCO) in the PLL 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. 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 a reference signal, either from the crystal oscillator or from an external source. The comparator generates a control signal proportional to the difference in phase between the two inputs. This 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 as low as possible 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 ($V_{DDSYN} = 0$). The XFC pin must be left floating in this case.
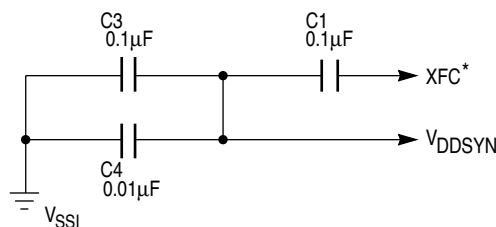
* MAINTAIN LOW LEAKAGE ON THE XFC NODE.

16/32 XFC CONN

Figure 7 System Clock Filter Network

The synthesizer locks when the VCO frequency is equal to f_{ref}. Lock time is affected by the filter time constant and by the amount of difference between the two comparator inputs. Whenever a comparator input changes, the synthesizer must relock. Lock status is shown by the SLOCK bit in SYNCR. During power-up, the MCU does not come out of reset until the synthesizer locks. Crystal type, characteristic frequency, and layout of external oscillator circuitry affect lock time.

When the clock synthesizer is used, SYNCR determines operating frequency and certain operating parameters. The W and Y[5:0] bits are located in the PLL feedback path, enabling frequency multiplication by a factor of up to 256. When the W or Y values change, VCO frequency changes, and there is a VCO relock delay. The SYNCR X bit controls a divide-by circuit that is not in the synthesizer feedback loop. When X = 0 (reset state), a divide-by-four circuit is enabled, and the system clock frequency is one-fourth the VCO frequency (f_{VCO}). When X = 1, a divide-by-two circuit is enabled, and system clock frequency is one-half the VCO frequency (f_{VCO}). There is no relock delay when clock speed is changed by the X bit.

Clock frequency is determined by SYNCR bit settings as follows:

$$f_{sys} = \frac{f_{ref}}{128}[4(Y + 1)(2^{(2W + X)})]$$

The reset state of SYNCR ($3F00) results in a power-on f_{sys} of 16.78 MHz when the f_{ref} is 4.194 MHz.

For the device to perform correctly, the clock and frequency selected by the W, X, and Y bits must be within the limits specified for the MCU.

Internal VCO frequency is determined by the following equations:

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

$$\text{or}$$

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

3.3.3 Clock Control

The clock control circuits determine system clock frequency and clock operation under special circumstances, such as loss of synthesizer reference or low-power mode. 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[5:0]				EDIV	0	0	RSVD[1]	SLOCK	RSVD[1]	STSCIM	STEXT

RESET:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	1	1	1	1	1	1	0	0	0	0	U	0	0	0

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. Because the CPU16 always operates in supervisor mode, SYNCR can be read or written at any time.

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

X — Frequency Control Bit
This bit controls a divide by two prescaler that is not in the synthesizer feedback loop. Setting it doubles clock speed without changing VCO speed. There is no VCO relock delay.

Y[5:0] — Frequency Control
The Y field controls the modulus down counter in the synthesizer feedback loop, which effectively allows frequency multiplication by a value of Y + 1. VCO relock delay is required.

EDIV — ECLK Divide Rate
    0 = ECLK frequency is system clock divided by 8
    1 = ECLK frequency is system clock divided by 16
ECLK is an external M6800 bus clock available on pin ADDR23. Refer to **3.9 Chip Selects** for more information.

SLOCK — Synthesizer Lock Flag
    0 = VCO is enabled, but has not locked
    1 = VCO has locked on the desired frequency (or system clock is external)
The MCU remains in reset until the synthesizer locks, but SLOCK does not indicate synthesizer lock status until after the user writes to SYNCR.

STSCIM — Stop Mode SCIM Clock
    0 = When LPSTOP is executed, the SCIM clock is driven from the crystal oscillator and the VCO is turned off to conserve power.
    1 = When LPSTOP is executed, the SCIM clock is driven from 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 from the SCIM clock, as determined by the state of the STSCIM bit.

3.3.4 Periodic Interrupt Timer

The periodic interrupt timer (PIT) generates interrupts of specified priorities at specified 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 14 shows what interrupt request level is asserted when a periodic interrupt is generated. If a PIT interrupt and an external interrupt request of the same priority occur simultaneously, the PIT interrupt is serviced first. The periodic timer continues to run when the interrupt is disabled.

Table 14 Periodic Interrupt Request Levels

PIRQL	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
The bits of this field contain the vector generated in response to an interrupt from the periodic timer. When the SCIM 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	$\overline{\text{MODCLK}}$	0	0	0	0	0	0	0	0

PITR contains the count value for the periodic timer. A zero value turns off the periodic timer. This register can be read or written at any time.

PTP — Periodic Timer Prescaler Control
        1 = Periodic timer clock prescaled by a value of 512
        0 = Periodic timer clock not prescaled
The reset state of PTP is the complement of the state of the MODCLK signal during reset.

PITM[7:0] — Periodic Interrupt Timing Modulus Field
The PIT period can be calculated as follows:

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

where

PIT Period = Periodic interrupt timer period

PITM[7:0] = Periodic interrupt timer register modulus

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

Prescale = 512 or 1 depending on the state of the PTP bit in PITR

3.4 System Protection

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 a complete control system. **Figure 8** shows the SCIM system protection block diagram.
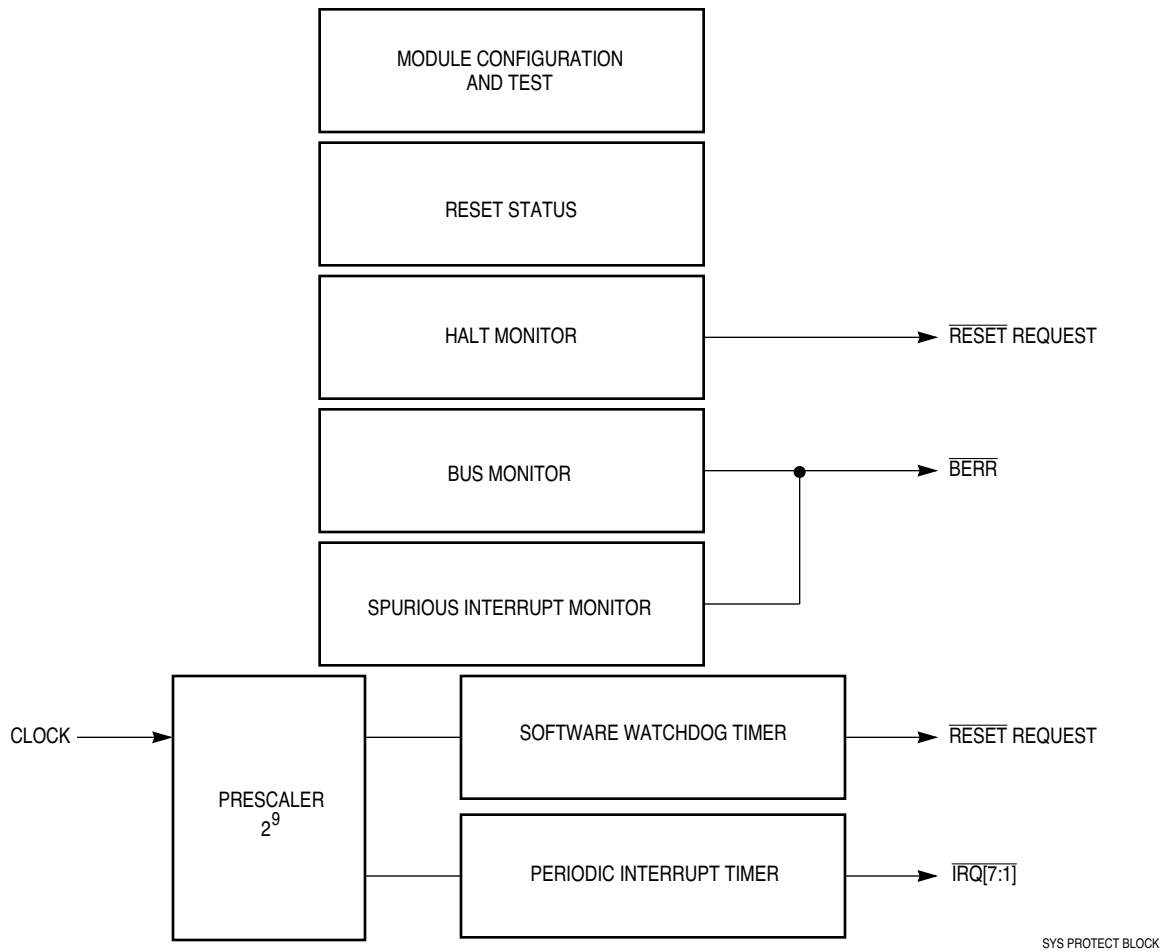
Figure 8 System Protection Block

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[1:0]	

RESET:

| | | | | | | | | 1 | $\overline{\text{MODCLK}}$ | 0 | 0 | 0 | 0 | 0 | 0 |

The system protection control register controls system monitor functions, software watchdog clock prescaling, and bus monitor timing. This register can only be written once following reset, but can be read at any time.

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

SWP — Software Watchdog Prescale
  This bit controls the value of the software watchdog prescaler.
    0 = Software watchdog clock not prescaled
    1 = Software watchdog clock prescaled by 512
  The reset value of SWP is the complement of the state of the MODCLK pin during reset.

SWT[1:0] — Software Watchdog Timing
This field selects the divide ratio used to establish software watchdog time-out period. **Table 15** gives the ratio for each combination of SWP and SWT bits.

Table 15 Software Watchdog Timeout Period Divide Ratio

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 field selects a bus monitor time-out period as shown in **Table 16**.

Table 16 Bus Monitor Timeout Period

BMT[1:0]	Time-Out
0 0	64 System Clocks
0 1	32 System Clocks
1 0	16 System Clocks
1 1	8 System Clocks

3.4.1 Bus Monitor

The internal bus monitor checks for excessively long response times during normal bus cycles ($\overline{DSACK1}$) and during IACK cycles. The monitor asserts the internal $\overline{BERR}$ signal if response time is excessive.

$\overline{DSACK1}$ response times are measured in clock cycles. The maximum allowable response time can be selected by setting BMT[1:0].

The monitor does not check $\overline{DSACK1}$ response on the external bus unless the CPU16 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.

3.4.2 Halt Monitor

The halt monitor responds to an assertion of $\overline{\text{HALT}}$ on the internal bus, caused by a double bus fault. This signal is asserted by the CPU after a double bus fault occurs. A flag in the reset status register (RSR) indicates that the last reset was caused by the halt monitor. The halt monitor reset can be inhibited by the HME bit in the SYPCR.

3.4.3 Spurious Interrupt Monitor

The spurious interrupt monitor causes a bus error exception if no interrupt arbitration occurs during an interrupt acknowledge cycle. The most common cause of spurious interrupts is failure to set the module configuration register IARB[3:0] to a non-zero value for modules that can generate interrupts.

3.4.4 Software Watchdog

SWSR — Software Watchdog Service Register   **$YFFA27**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UNUSED								SWSR							

RESET:

| | | | | | | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

The software watchdog service register (SWSR) 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.

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

- Write $55 to SWSR
- Write $AA to SWSR

Both writes must occur before timeout in the order listed, but any number of instructions, up to the end of the timeout period, can be executed between the two writes.

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 will take effect.

The reset value of SWP is the complement of the state of the MODCLK pin on the rising edge of reset.

Software watchdog time-out period is given in the following equation:

$$\text{Timeout Period} = \frac{\text{Divide Ratio Specified by SWP and SWT[1:0]}}{f_{\text{ref}}}$$

3.5 External Bus Interface

The external bus interface (EBI) transfers information between the internal MCU bus and external devices when the MCU is operating in expanded modes. In 16-bit expanded mode, the external bus has 24 address lines and 16 data lines. In 8-bit expanded mode, the external bus has 24 address lines and 8 data lines. Because the CPU16 drives only 20 of the 24 IMB address lines, ADDR[23:20] follow the output state of ADDR19.

The EBI supports byte, word, and long-word transfers. Ports are accessed through the use of asynchronous cycles controlled by the data transfer size (SIZ1 and SIZ0) and the data size acknowledge pin ($\overline{\text{DSACK1}}$).

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 **3.9 Chip Selects** for more information.

3.6 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. 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 17** 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. R/$\overline{\text{W}}$ only transitions when a write cycle is preceded by a read cycle or vice versa. The signal can remain low for two consecutive write cycles.

Table 17 Size Signal Encoding

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

3.6.1 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 $\overline{\text{AS}}$ is asserted.

Table 18 displays CPU16 address space encodings.

Table 18 CPU16 Address Space Encoding

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

3.6.2 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. The CPU16 drives ADDR[23:20] to the same logic state as ADDR19.

3.6.3 Address Strobe

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

3.6.4 Data Bus

Data bus signals DATA[15:0] comprise a bidirectional, non-multiplexed parallel bus that transfers data to or from the MCU. A read or write operation can transfer 8 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.

3.6.5 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.

3.6.6 Bus Cycle Termination Signals

During bus cycles, external devices assert a data transfer and size acknowledge signal ($\overline{DSACK1}$). During a read cycle, the signal tells the MCU to terminate the bus cycle and to latch data. During a write cycle, the signal indicates that an external device has successfully stored data and that the cycle can terminate. The $\overline{DSACK1}$ signal also indicates to the MCU the size of the port for the bus cycle just completed. In the MC68HC916X1, the $\overline{DSACK0}$ pin is not provided and an external device indicates the availability of data by asserting $\overline{DSACK1}$ regardless of port size.

The bus error ($\overline{BERR}$) signal is also a bus cycle termination indicator and can be used in the absence of $\overline{DSACK1}$ to indicate a bus error condition. $\overline{BERR}$ can also be asserted in conjunction with $\overline{DSACK1}$, provided $\overline{BERR}$ meets the appropriate timing requirements. The internal bus monitor can be used to generate the $\overline{BERR}$ signal for internal and internal-to-external transfers. When $\overline{BERR}$ is asserted, the CPU16 takes a bus error exception.

3.6.7 Data Transfer Mechanism

MCU architecture supports byte, word, and long-word operands, allowing access to 8- and 16-bit data ports through the use of asynchronous cycles.

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 9**. 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.

	Operand			**Byte Order**		
	31	24	23	16	15	8	7	0

Long Word	OP0	OP1	OP2	OP3
Three Byte		OP0	OP1	OP2
Word			OP0	OP1
Byte				OP0
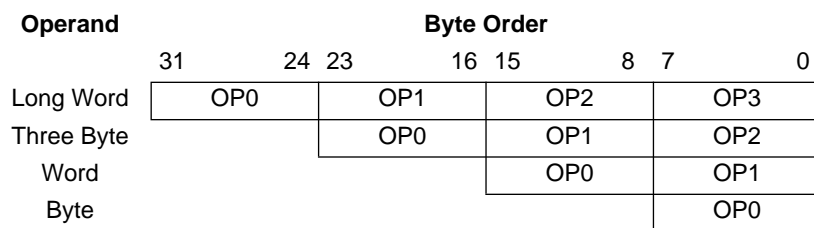
Figure 9 Operand Byte Order

3.6.8 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] are driven to the same logic state as ADDR19.

3.6.9 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.

3.6.10 Operand Transfer Cases

Table 19 shows 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 19 Operand Alignment

Transfer Case	SIZ1	SIZ0	ADDR0	DATA[15:8]	DATA[7:0]
Byte to 16-Bit Port (Even)	0	1	0	OP0	(OP0)[1]
Byte to 16-Bit Port (Odd)	0	1	1	(OP0)[1]	OP0
Word to 16-Bit Port (Aligned)	1	0	0	OP0	OP1
Word to 16-Bit Port (Misaligned)	1	0	1	(OP0)[1]	OP0
3 Byte to 16-Bit Port (Aligned)[2]	1	1	0	OP0	OP1
3 Byte to 16-Bit Port (Misaligned)[2]	1	1	1	(OP0)[1]	OP0
Long Word to 16-Bit Port (Aligned)	0	0	0	OP0	OP1
Long Word to 16-Bit Port (Misaligned)[3]	1	0	1	(OP0)[1]	OP0

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

It is not possible to perform transfers of word operands to an 8-bit port on the MC68HC916X1 because the $\overline{DSACK0}$ pin is not present and therefore cannot be asserted to acknowledge the transfer. This limitation can be overcome by using SCIM chip-select logic to generate $\overline{DSACK}$ for such transfers.

3.7 Resets

Reset procedures handle system initialization and recovery from catastrophic failure. The MCU performs resets with a combination of hardware and software. The SCIM determines whether a reset is valid, asserts control signals, performs basic system configuration and boot ROM selection based on hardware mode-select inputs, then passes control to the CPU16.

Reset occurs when an active low logic level on the $\overline{RESET}$ pin is clocked into the SCIM. 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 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.

3.7.1 SCIM Reset Mode Selection

The logic states of certain MCU pins during reset determine SCIM operating configuration. Refer to **3.2.1 Operating Modes** for more information.

3.7.2 MCU Module Pin Function During Reset

Module pins usually 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. For more information, refer to the sections of this technical summary that present information about the individual modules. **Table 20** is a summary of module pin functions out of reset.

Table 20 Module Pin Functions

Module	Pin Mnemonic	Function
ADC	PADA[7:0]/AN[7:0]	Discrete Input
	VRH	Reference Voltage
	VRL	Reference Voltage
CPU	DSI/IPIPE1	DSI/IPIPE1
	DSO/IPIPE0	DSO/IPIPE0
	$\overline{\text{BKPT}}$/DSCLK	$\overline{\text{BKPT}}$/DSCLK
GPT	PGP7/IC4/OC5	Discrete Input
	PGP[6:3]/OC[4:1]	Discrete Input
	PGP[2:0]/IC[3:1]	Discrete Input
	PAI	Discrete Input
	PCLK	Discrete Input
	PWMA, PWMB	Discrete Output
QSM	PQS7/MOSI	Discrete Input
	PQS6/MISO	Discrete Input
	PQS5/SCK	Discrete Input
	PQS4/PCS3	Discrete Input
	PQS3/PCS2	Discrete Input
	PQS2/PCS1	Discrete Input
	PQS1/PCS0/$\overline{\text{SS}}$	Discrete Input
	PQS0/TXD	Discrete Input
	RXD	RXD

3.7.3 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 time-out period) to protect write cycles from being aborted by reset. While $\overline{\text{RESET}}$ is asserted, SCIM pins are either in an inactive, 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 a 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 reset input pin goes to an inactive, high-impedance state for 10 cycles. At the end of this 10-cycle period, the reset input is tested. When the input is at logic level one, reset exception processing begins. If, however, the reset input 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.

3.7.4 Power-On Reset

When the SCIM clock synthesizer is used to generate system clocks, power-on reset involves special circumstances related to the application of system and clock synthesizer power. Regardless of clock source, voltage must be applied to the clock synthesizer power input pin V_{DDSYN}, so that the MCU can 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 SCIM drives the IMB internal and external reset lines. The circuit releases the internal reset line as V_{DD} ramps up to the minimum specified value, and SCIM pins are initialized. When V_{DD} reaches the specified minimum value, the clock synthesizer VCO begins operation. Clock frequency ramps up to the specified limp mode frequency. The external $\overline{RESET}$ line remains asserted until the clock synthesizer PLL locks and 512 CLKOUT cycles elapse.

The SCIM 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.

3.7.5 Use of Three-State Control Pin

Asserting the three-state control (TSC) input causes all MCU output drivers to go to an inactive, high-impedance condition. Although TSC is an active-high input, it does not have an internal pull-down and must be tied low when not in use.

TSC must remain asserted for ten system clock cycles for drivers to change state. There are certain constraints on use of TSC during power-up reset.

When the internal clock synthesizer is used (MODCLK held high during reset), synthesizer ramp-up time affects how long ten clock cycles take. Worst case is approximately 20 milliseconds 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.

NOTE

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.

3.8 Interrupts

Interrupt recognition and servicing involve complex interaction between the central processing unit, the single-chip integration module, and a device or module requesting interrupt service.

The CPU16 provides for 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 exception.

Interrupt recognition is based on the states of interrupt request signals $\overline{\text{IRQ7}}$ and the IP mask value. Each of the signals corresponds to an interrupt priority. $\overline{\text{IRQ1}}$ has the lowest priority, and $\overline{\text{IRQ7}}$ has the highest priority.

NOTE

On the MC68HC916X1, the only external interrupts available are $\overline{\text{IRQ6}}$ and $\overline{\text{IRQ7}}$.

The IP field consists of three bits (CCR[7:5]). 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 (except for $\overline{\text{IRQ7}}$) from being recognized and processed. When IP contains %000, no interrupt is masked. During exception processing, the IP field is set to the priority of the interrupt being serviced.

Interrupt request signals can be asserted by external devices or by microcontroller modules. Request lines are connected internally by a wired-NOR. Simultaneous requests with different priorities can be made. Internal assertion of an interrupt request signal does not affect the logic state of the corresponding MCU pin.

External interrupt requests are routed to the CPU16 through the external bus interface and SCIM interrupt control logic. The CPU treats external interrupt requests as though they had come from the SCIM.

External $\overline{\text{IRQ6}}$ is an active-low level-sensitive input. External $\overline{\text{IRQ7}}$ is an active-low transition-sensitive input. It requires both an edge and a voltage level for validity.

$\overline{\text{IRQ6}}$ is maskable. $\overline{\text{IRQ7}}$ is non-maskable. The $\overline{\text{IRQ7}}$ input is transition-sensitive to prevent redundant servicing and stack overflow. A non-maskable interrupt is generated each time $\overline{\text{IRQ7}}$ is asserted, and 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 of equal or lower priority than the current IP mask value is made, the CPU does not recognize the occurrence of the request in any way.

3.8.1 Interrupt Acknowledge and Arbitration

Interrupt acknowledge bus cycles are generated during exception processing. When the CPU16 detects one or more interrupt requests of a priority higher than the interrupt priority mask value, it performs a CPU space read from address $FFFFF: [IP] : 1.

The CPU space read cycle performs two functions: it places a mask value corresponding to the highest priority interrupt request on the address bus, and it acquires an exception vector number from the interrupt source. The mask value also serves two purposes: it is latched into the CCR IP field to mask lower-priority interrupts during exception processing, and it is decoded by modules that have requested interrupt service to determine whether the current interrupt acknowledge cycle pertains to them.

Modules that have requested interrupt service decode the IP value placed on the address bus at the beginning of the interrupt acknowledge cycle. If their requests are at the specified IP level, they respond to the cycle. Arbitration between simultaneous requests of the same priority is performed by serial contention between module interrupt arbitration (IARB) field bit values.

Each module that can make an interrupt service request, including the SCIM, has an IARB field in its configuration register. An IARB field can be assigned a value from %0001 (lowest priority) to %1111 (highest priority). A value of %0000 in an IARB field causes the CPU16 to process a spurious interrupt exception when an interrupt from that module is recognized.

Because the EBI manages external interrupt requests, the SCIM IARB value is used for arbitration between internal and external interrupt requests. The reset value of IARB for the SCIM is %1111. The reset IARB value for all other modules is %0000. Initialization software must assign different IARB values to implement an arbitration scheme.

Each module must have a unique IARB value. When two or more IARB fields have the same non-zero value, the CPU16 interprets multiple vector numbers simultaneously, with unpredictable consequences.

Arbitration must always take place, even when a single source requests service. This point is important for two reasons: the CPU interrupt acknowledge cycle is not driven on the external bus unless the SCIM wins contention, and failure to contend causes an interrupt acknowledge bus cycle to be terminated by a bus error, which causes a spurious interrupt exception to be taken.

When arbitration is complete, the dominant module must place an interrupt vector number on the data bus and terminate the bus cycle. In the case of an external interrupt request, because the interrupt acknowledge cycle is transferred to the external bus, an external device must decode the mask value and respond with a vector number, then generate bus cycle termination signals. If the device does not respond in time, a spurious interrupt exception is taken.

The periodic interrupt timer (PIT) in the SCIM can generate internal interrupt requests of specific priority at predetermined intervals. By hardware convention, PIT interrupts are serviced before external interrupt service requests of the same priority. Refer to **3.3.4 Periodic Interrupt Timer** for more information.

3.8.2 Interrupt Processing

The following summary outlines the interrupt processing sequence. 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. 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. Request priority is latched into the CCR IP field from the address bus.

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 the priority value in the address, IARB contention takes place. When there is no contention, the spurious interrupt monitor asserts $\overline{\text{BERR}}$, and a spurious interrupt exception is processed.

E. After arbitration, the interrupt acknowledge cycle can be completed in one of three ways:

1. The dominant interrupt source supplies a vector number and $\overline{\text{DSACK1}}$ signals appropriate to the access. The CPU16 acquires the vector number.

2. Chip-select logic asserts $\overline{\text{AVEC}}$ internally and the CPU16 generates an autovector number corresponding to interrupt priority.

3. 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.

3.9 Chip Selects

Typical microcontrollers require additional hardware to provide external chip-select signals. The MC68HC916X1 includes five general-purpose programmable chip select circuits that can provide 2- to 13-clock cycle access to external memory and peripherals. Two additional chip select signals, $\overline{\text{CSE}}$ and $\overline{\text{CSM}}$, provide emulation support. Address block sizes of 2 Kbytes to 1 Mbyte can be selected. However, because the CPU16 drives ADDR[23:20] to the same logic state as ADDR19, 512-Kbyte blocks are the largest usable size. Refer to **3.2.5 Emulation Support** for more information.

Chip select assertion can be synchronized with bus control signals to provide output enable, read/write strobes, or interrupt acknowledge signals. Logic can also generate $\overline{\text{DSACK}}$ and $\overline{\text{AVEC}}$ signals internally. A single $\overline{\text{DSACK}}$ generator is shared by all circuits. Multiple chip selects assigned to the same address and control must have the same number of wait states. Chip selects can also be synchronized with the ECLK signal available on ADDR23.

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, a chip select signal is asserted. Select signals are active low. **Figure 10** shows a single chip-select circuit.
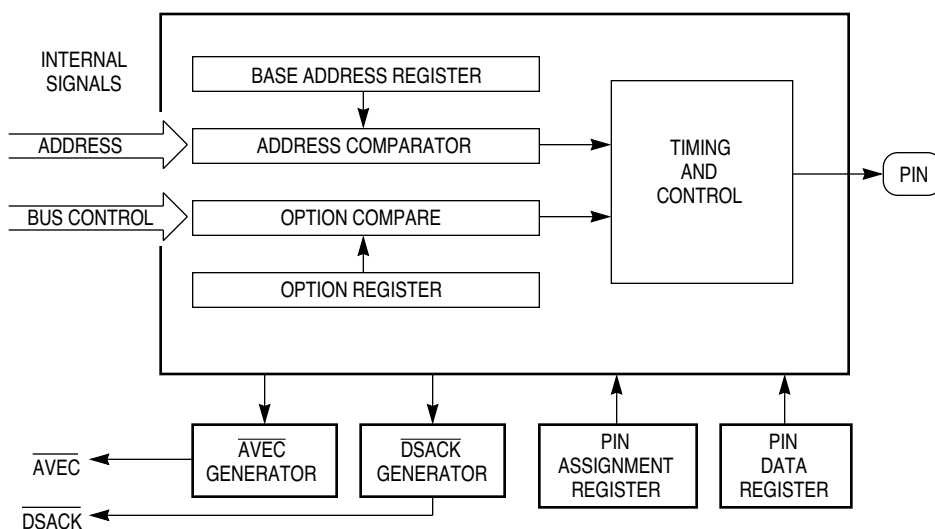
Figure 10 Chip-Select Circuit Block Diagram

If a chip select function is given the same address as a microcontroller module or memory array, an access to that address goes to the module or array, and the chip select signal is not asserted.

Each chip-select pin has two or more functions. Configuration out of reset is determined by operating mode. In single-chip mode, all chip select pins except $\overline{CS10}$ and $\overline{CS0}$ are configured for alternate functions or discrete output. In expanded modes, appropriate pins are configured for chip select operation, but chip select signals cannot be asserted until a transfer size is chosen. In fully expanded mode, data bus pins can be held low to enable alternate chip-select pin functions.

Table 21 shows allocation of chip selects and discrete outputs to MCU pins.

Table 21 Chip Select Pin Allocation

Chip Select Function	Alternate Function	Discrete Outputs Function
$\overline{CS0}$	$\overline{BR}$	—
$\overline{CSM}$	$\overline{BG}$	—
$\overline{CSE}$	$\overline{BGACK}$	—
$\overline{CS3}$	FC0	PC0
—	FC1	PC1
$\overline{CS5}$	FC2	PC2
$\overline{CS6}$	ADDR19	PC3
$\overline{CS10}$	ADDR23	ECLK

3.9.1 Chip Select Registers

Pin assignment registers (CSPAR) determine functions of chip select pins. Pin assignment registers also determine port size (8- or 16-bit) for dynamic bus allocation.

A pin data register (PORTC) latches discrete output data.

Blocks of addresses are assigned to each chip select function. Block sizes of 2 Kbytes to 1 Mbyte can be selected by writing values to the appropriate base address register (CSBAR). However, because the logic state of ADDR20 is always the same as the state of ADDR19, the largest usable block size is 512 Kbytes. Address blocks for separate chip-select functions can overlap, however, they must have the same number of wait states if they do.

Chip-select option registers (CSOR) 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.

3.9.2 Pin Assignment Registers

The pin assignment registers contain pairs of bits that determine the functions of chip-select pins. **Table 22** shows pin assignment field encoding. Pin functions are shown in **Tables 23** and **24** following the register diagrams.

Reset state of the pin assignment registers depends on operating mode. In the register diagrams, reset values are shown in the following order: single-chip mode, 8-bit expanded mode, and 16-bit expanded mode. The notation DATA# indicates that a bit goes to the logic level of that data bus pin on reset. Data bus lines have weak pull-ups. During reset in 16-bit expanded mode, an active external device can pull the data lines low to select alternate functions.

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	CSPA0[13:12]		CSPA0[11:10]		CSPA0[9:8]		CSPA0[7:6]		CSPA0[5:4]		CSPA0[3:2]		RESERVED	

RESET:

SINGLE-CHIP MODE

0	0	0	0	0	0	0	0	0	1	0	1	1	1	0	0

8-BIT EXPANDED MODE

0	0	1	0	0	1	1	0	0	1	0	1	1	0	0	0

16-BIT EXPANDED MODE

0	0	DATA2	1	0	1	DATA2	1	$\overline{\text{DATA10}}$	1	$\overline{\text{DATA10}}$	1	DATA2	1	0	0

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	CSPA1[9:8]		CSPA1[7:6]		CSPA1[5:4]		CSPA1[3:2]		CSPA1[1:0]	

RESET:

SINGLE-CHIP MODE

0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0

8-BIT EXPANDED MODE

0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0

16-BIT EXPANDED MODE

0	0	0	0	0	0	DATA7	1	$\overline{\text{DATA6}}$	1	DATA5	1	DATA4	1	DATA3	1

Clearing both CS10 select bits (CSPAR1[9:8]) enables the M6800 bus clock (ECLK) on ADDR23.

Table 22 Pin Assignment Field Encoding

Bit Pair	Description
00	Discrete Output
01	Alternate Function
10	Chip Select (8-Bit Port)
11	Chip Select (16-Bit Port)

Table 23 CSPAR0 Pin Functions

CSPAR0 Field	Chip-Select Signal	Alternate Signal	Discrete Output
CSPA0[13:12]	$\overline{\text{CS5}}$	FC2	PC2
CSPA0[11:10]	—	FC1	PC1
CSPA0[9:8]	$\overline{\text{CS3}}$	FC0	PC0
CSPA0[7:6]	$\overline{\text{CSE}}$	$\overline{\text{BGACK}}$	—
CSPA0[5:4]	$\overline{\text{CSM}}$	$\overline{\text{BG}}$	—
CSPA0[3:2]	$\overline{\text{CS0}}$	$\overline{\text{BR}}$	—

Table 24 CSPAR1 Pin Functions

CSPAR1 Field	Chip-Select Signal	Alternate Signal	Discrete Output
CSPA1[9:8]	$\overline{\text{CS10}}$	ADDR23	ECLK
CSPA1[7:6]	—	—	—
CSPA1[5:4]	—	—	—
CSPA1[3:2]	—	—	—
CSPA1[1:0]	$\overline{\text{CS6}}$	ADDR19	PC3

A pin programmed as a discrete output drives an external signal to the value specified in the port C data register (PORTC), with the following exceptions:

• No discrete output function is available on pins $\overline{\text{BR}}$, $\overline{\text{BG}}$, or $\overline{\text{BGACK}}$.
• ADDR23 provides ECLK output rather than a discrete output signal.

Internal chip select logic is inhibited when discrete output or alternate function are assigned.

Port size is determined when a pin is assigned as a chip select. When a pin is assigned to an 8-bit port, the chip select is asserted at all addresses within the block range. If a pin is assigned to a 16-bit port, the upper/lower byte field of the option register selects the byte with which the chip select is associated.

3.9.3 Base Address Registers

A base address is the starting address for the block enabled by a given chip select. Block size determines the extent of the block above the base address. Each chip select has an associated base register so that an efficient address map can be constructed for each application. If a chip select is assigned an address used by a microcontroller module, the module has priority. The chip select does not respond to an access.

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] is at the same logic level as ADDR19 during internal CPU master operation. ADDR[23:20] must match ADDR19 for the chip select to be active.

NOTE

$\overline{\text{CSBOOT}}$ is not present on the MC68HC916X1, however, the $\overline{\text{CSBOOT}}$ chip-select logic is still present and should be disabled before other chip-selects are initialized.

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 a multiple of block size. Base address register diagrams show how base register bits correspond to address lines.

Since ADDR20 = ADDR19 in the CPU16, the maximum block size is 512 Kbytes. Because ADDR[23:20] follow the logic state of ADDR19, if all 24 address lines are used, addresses from $080000 to $F7FFFF are inaccessible.

BLKSZ[2:0] — Block Size Field
This field determines the size of the block above the base address that must be enabled by the chip select. **Table 25** shows bit encoding for the base address registers block size field.

Table 25 Block Size Field Bit Encoding

BLKSZ[2:0]	Block Size	Address Lines Compared
000	2 K	ADDR[23:11]
001	8 K	ADDR[23:13]
010	16 K	ADDR[23:14]
011	64 K	ADDR[23:16]
100	128 K	ADDR[23:17]
101	256 K	ADDR[23:18]
110	512 K	ADDR[23:19]
111	512 K	ADDR[23:20][1]

1. During normal operation ADDR[23:20] is at the same logic level as ADDR19.

3.9.4 Option Registers

The option registers contain eight fields that determine timing of and conditions for assertion of chip select signals. These make the chip selects useful for generating peripheral control signals. Certain constraints set by fields in the base address register and in the option register must be satisfied to assert a chip select signal and to provide $\overline{\text{DSACK}}$ or autovector support.

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[2:0]		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

NOTE
$\overline{\text{CSBOOT}}$ is not present on the MC68HC916X1, however, the $\overline{\text{CSBOOT}}$ chip-select logic is still present and should be disabled before other chip-selects are initialized.

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

MODE — Asynchronous/Synchronous Mode
      0 = Asynchronous mode selected
      1 = Synchronous mode selected
In asynchronous mode, the chip select is asserted synchronized with $\overline{AS}$ or $\overline{DS}$.

In synchronous mode, the $\overline{DSACK}$ field is not used 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 E-clock 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 26** lists upper/lower byte options.

Table 26 Byte Field Bit Encoding

BYTE[1:0]	Description
00	Disable
01	Lower Byte
10	Upper Byte
11	Both Bytes

R/$\overline{W}$[1:0] — Read/Write
This field causes a chip select to be asserted only for a read, only for a write, or for both read and write. **Table 27** shows the options.

Table 27 Read/Write Bit Encodings

R/$\overline{W}$[1:0]	Description
00	Disable
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 28** shows the $\overline{DSACK}$[3:0] field encoding. A no-wait encoding (%0000) corresponds to a three clock-cycle bus. The fast termination encoding (%1110) corresponds to a two clock-cycle bus. Microcontroller modules typically respond at this rate, but fast termination can also be used to access fast external memory.

Table 28 $\overline{\text{DSACK}}$ Field Encoding

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

SPACE[1:0] — Address Space Select
　　This option field is used to select an address space to be used by the chip select logic. The CPU16 normally operates in supervisor space. All space types can be used. Interrupt acknowledge cycles take place in CPU space. **Table 29** shows address space bit encodings.

Table 29 Address Space Bit Encodings

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

　　1. The CPU16 executes code only in supervisor mode, therefore this space field encoding has no effect. Supervisor/User space is equivalent to supervisor space encoding.

IPL[2:0] — Interrupt Priority Level
　　When the SPACE[1:0] field is set for CPU space (%00), 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[2:0] field. If the values are the same, then a chip-select can be asserted, provided other option register conditions are met. When the SPACE[1:0] field has any value except %00, the IPL[2:0] field determines whether an access takes place in program or data space. **Table 30** shows IPL[2:0] field encoding.

Table 30 Interrupt Priority Level Field Encoding

IPL[2:0]	SPACE[1:0] = 00	SPACE[1:0] = 01, 10, 11
000	All[1]	Data or Program
001	IPL1	Data
010	IPL2	Program
011	IPL3	Reserved
100	IPL4	Reserved
101	IPL5	Data
110	IPL6	Program
111	IPL7	Reserved

1. "All" means that a chip select signal is asserted regardless of the priority of the interrupt.

This field only affects the response of chip selects and does not affect interrupt recognition by the CPU.

$\overline{\text{AVEC}}$ — Autovector Enable
    0 = External interrupt vector enabled
    1 = Autovector enabled
This field selects one of two methods of acquiring the interrupt vector during the interrupt acknowledge cycle. It is not usually used with a chip select pin.

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 automatically generates an $\overline{\text{AVEC}}$ in response to the interrupt acknowledge cycle. Otherwise, the vector must be supplied by the requesting device.

3.9.5 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		RESERVED		PC3	PC2	PC1	PC0

RESET:

| | | | | | | | | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

3.9.6 Chip Select Reset Operation

The reset values of the chip select pin assignment fields in CSPAR0 and CSPAR1 depend on the operating mode selected. Refer to **3.2.1 Operating Modes** and to the discussion of these registers for more information.

The BYTE[1:0] field in option register CSORBT has a reset value of both bytes, but CSOR[10:0] have a reset value of disable, as they should not select external devices until an initial program sets up the base and option registers.

3.10 General-Purpose Input/Output

The SCIM contains six general-purpose input/output ports: ports A, B, E, F, G, and H. (Port C, an output-only port, is included under the discussion of chip selects). Ports A, B, and G are available in single-chip mode only and port H is available in single-chip or 8-bit expanded modes only. Ports E, F, G, and H have an associated data direction register to configure each pin as input or output. Ports A and B share a data direction register that configures each port as input or output. Ports E and F have associated pin assignment registers that configure each pin as digital I/O or an alternate function. Port F has an edge-detect flag register that indicates whether a transition has occurred on any of its pins.

Table 31 shows the shared functions of the general-purpose I/O ports and the modes in which they are available.

Table 31 General-Purpose I/O Ports

Port	Shared Function	Modes
A	ADDR[18:11]	Single Chip
B	ADDR[10:3]	Single Chip
E	Bus Control	All
F	$\overline{IRQ[7:6]}$/MODCLK	All
G	DATA[15:8]	Single Chip
H	DATA[7:0]	Single Chip, 8-Bit Expanded

Access to the port A, B, E, F, G, and H data and data direction registers, and the port C, E, and F pin assignment registers require three clock cycles to ensure timing compatibility with external port replacement logic. Port registers are byte-addressable and are grouped to allow coherent word access to port data register pairs A-B and G-H, as well as word-aligned long word coherency of A-B-G-H port data registers.

If emulation mode is enabled, the emulation mode chip-select signal $\overline{CSE}$ is asserted whenever an access to ports A, B, E, G, and H data and data direction registers or the port E pin assignment register is made. The SCIM does not respond to these accesses, but allows external logic, such as a Motorola port replacement unit (PRU) MC68HC33 to respond. Port C data and data direction register, port F data and data direction register, and the port F pin assignment register remain accessible.

A write to the port A, B, E, F, G, or H data register is stored in the internal data latch. If any port pin is configured as an output, the value stored for that bit is driven on the pin. A read of the port 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.

3.10.1 Ports A and B

Ports A and B are available in single-chip mode only. One data direction register controls data direction for both ports. Port A and B registers can be read or written at any time the MCU is not in emulator mode.

PORTA — Port A Data Register   **$YFFA0A**
PORTB — Port B Data Register   **$YFFA0B**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0

RESET:

U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U

DDRAB — Port A/B Data Direction Register   **$YFFA14**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	DDA	DDB	DDRE							

RESET:

U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U

DDA and DDB control the direction of the pin drivers for ports A and B, respectively, when the pins are configured for I/O. Setting DDA or DDB to one configures all pins in the corresponding port as outputs. Clearing DDA or DDB to zero configures all pins in the corresponding port as inputs.

3.10.2 Port E

Port E can be made available in all operating modes. The state of $\overline{BERR}$ and DATA8 during reset controls whether the port E pins are used as bus control signals or discrete I/O lines.

If the MCU is in emulator mode, an access of the port E data, data direction, or pin assignment registers (PORTE, DDRE, PEPAR) is forced to go external. This allows port replacement logic to be supplied externally, giving an emulator access to the bus control signals.

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	RESERVED		PE1	RSVD[1]

RESET:

								U	U	U	U	U	U	U	U

1. Reserved

PORTE is a single register that can be accessed in two locations. It can be read or written at any time the MCU is not in emulator mode.

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	RESERVED		DDE1	RSVD[1]

RESET:

								0	0	0	0	0	0	0	0

1. Reserved

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 the MCU is not in emulator mode.

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	RESERVED		PEPA1	RSVD[1]

RESET:

8- AND 16-BIT EXPANDED MODES

| | | | | | | | | DATA8 | DATA8 | DATA8 | DATA8 | DATA8 | DATA8 | DATA8 | DATA8 |

SINGLE-CHIP MODE

| | | | | | | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

1. Reserved

The bits in PEPAR control the function of each port E pin. Any bit set to one defines the corresponding pin to be a bus control signal, with the function shown in **Table 32**. Any bit cleared to zero defines the corresponding pin to be an I/O pin, controlled by PORTE and DDRE.

Table 32 Port E Pin Assignments

PEPAR Bit	Port E Signal	Bus Control Signal
PEPA7	PE7	SIZ1
PEPA6	PE6	SIZ0
PEPA5	PE5	$\overline{AS}$
PEPA4	PE4	$\overline{DS}$
PEPA1	PE1	$\overline{DSACK1}$

$\overline{BERR}$ and DATA8 control the state of this register following reset. If $\overline{BERR}$ and/or DATA8 are low during reset, this register is set to $00, defining all port E pins as I/O pins. If $\overline{BERR}$ and DATA8 are both high during reset, the register is set to $FF, which defines all port E pins as bus control signals.

3.10.3 Port F

Port F pins can be configured as interrupt request inputs, edge-detect input/outputs, or discrete input/outputs. When port F pins are configured for edge detection, and a priority level is specified by writing a value to the port F edge-detect interrupt level register (PFLVR), port F control logic generates an interrupt request when the specified edge is detected. Interrupt vector assignment is made by writing a value to the port F edge-detect interrupt vector register (PFIVR). The edge-detect interrupt has the lowest arbitration priority in the SCIM.

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	RESERVED					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 on the pin. A read of PORTF returns the value on a pin only if the pin is configured as a discrete input. Otherwise, the value read is the value stored in the data register.

Port F is a single register that can be accessed in two locations. It can be read or written at any time, including when the MCU is in emulator mode.

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	RESERVED						DDF0

RESET:

| | | | | | | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

The bits in this register control the direction of port F pin drivers when the pins are configured for I/O. Setting any bit in this register configures the corresponding pin as an output. Clearing any bit in this register configures the corresponding pin as an input.

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								PFPA[7:6]		RESERVED				PFPA[1:0]	

RESET:

8- AND 16-BIT EXPANDED MODES

| | | | | | | | | DATA9 | DATA9 | DATA9 | DATA9 | DATA9 | DATA9 | DATA9 | DATA9 |

SINGLE-CHIP MODE

| | | | | | | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

The fields in this register determine the functions of pairs of port F pins. **Table 33** shows port F pin assignments. **Table 34** shows PFPAR pin functions.

In single-chip mode ($\overline{\text{BERR}}$ = 0 during reset), this register is set to $00, defining all port F pins to be I/O pins. In 8- and 16-bit expanded modes, the state of DATA9 during reset determines the default value for PFPAR.

Table 33 Port F Pin Assignments

PFPAR Field	Port F Signal	Alternate Signal
PFPA[7:6][1]	PF[7:6]	$\overline{\text{IRQ}}$[7:6]
PFPA[1:0][1]	PF[1:0]	MODCLK[2]

1. PF[5:2] are not connected. These bits have no meaning.
2. MODCLK signal is only recognized during reset.

Table 34 PFPAR Pin Functions

PFPAx Bits	Port F Signal
00	I/O pin without edge detect
01	Rising edge detect
10	Falling edge detect
11	Interrupt request

PORTFE — Port F Edge-Detect Flag Register $YFFA2B

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NOT USED								PEF7	PEF6	RESERVED					PEF0

RESET:

| | | | | | | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

When the corresponding pin is configured for edge detection, a PORTFE bit is set if an edge is detected. PORTFE bits remain set, regardless of the subsequent state of the corresponding pin, until cleared. To clear a bit, first read PORTFE, then write the bit to zero. When a pin is configured for general-purpose I/O or for use as an interrupt request input, PORTFE bits do not change state.

PFIVR — Port F Edge-Detect Interrupt Vector Register $YFFA2B

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

RESET:

| | | | | | | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

This register determines which vector in the exception vector table is used for interrupts generated by the port F edge-detect logic. Program PFIVR[7:0] to the value pointing to the appropriate interrupt vector. Refer to **4 Central Processing Unit** for interrupt vector assignments.

PFLVR — Port F Edge-Detect Interrupt Level Register $YFFA2D

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

RESET:

| | | | | | | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

PFLVR determines the priority level of the port F edge-detect interrupt. The reset value is $00, indicating that the interrupt is disabled. When several sources of interrupts from the SCIM are arbitrating for the same level, the port F edge-detect interrupt has the lowest arbitration priority.

3.10.4 Port G

Port G is available in single-chip mode only. These pins are always configured for use as general-purpose I/O in single-chip mode.

3.10.5 Port H

Port H is available in single-chip and 8-bit expanded modes only. The function of these pins is determined by the operating mode. There is no pin assignment register associated with this port.

PORTG — Port G Data Register $YFFA0C
PORTH — Port H Data Register $YFFA0D

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PG7	PG6	PG5	PG4	PG3	PG2	PG1	PG0	PH7	PH6	PH5	PH4	PH3	PH2	PH1	PH0

RESET:

| U | U | U | U | U | U | U | U | U | U | U | U | U | U | U | U |

These port data registers can be read or written any time the MCU is not in emulation mode. Reset has no effect.

DDRG — Port G Data Direction Register   **$YFFA0E**
DDRH — Port H Data Direction Register   **$YFFA0F**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DDG7	DDG6	DDG5	DDG4	DDG3	DDG2	DDG1	DDG0	DDH7	DDH6	DDH5	DDH4	DDH3	DDH2	DDH1	DDH0

RESET:

| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

The bits in this register control the direction of the port pin drivers when pins are configured as I/O. Setting a bit configures the corresponding pin as an output. Clearing a bit configures the corresponding pin as an input.

3.11 Factory Test

Test functions are integrated into the SCIM to support scan-based testing of the various MCU modules during production. Test submodule registers are intended for Motorola use. Register names and addresses are provided to show the user that these addresses are occupied.

SCIMTR — Single-Chip Integration Module Test Register   **$YFFA02**

SCIMTRE — Single-Chip 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 Submodule Control Register   **$YFFA38**

DREG — 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

Ease of programming is an important consideration when using a microcontroller. 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 (longword) 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

CPU16 architecture is a superset of 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

| 20 | 16 | 15 | 8 | 7 | 0 | BIT POSITION |

| A | B | ACCUMULATORS A AND B |
| D | ACCUMULATOR D (A : B) |

| E | ACCUMULATOR E |

| XK | IX | INDEX REGISTER X |

| YK | IY | INDEX REGISTER Y |

| ZK | IZ | INDEX REGISTER Z |

| SK | SP | STACK POINTER |

| PK | PC | PROGRAM COUNTER |

| CCR | PK | CONDITION CODE REGISTER / PC EXTENSION REGISTER |

| EK | XK | YK | ZK | ADDRESS EXTENSION REGISTER |

| SK | STACK EXTENSION REGISTER |

| HR | MAC MULTIPLIER REGISTER |

| IR | MAC MULTIPLICAND REGISTER |

35 16

| AM (MSB) | MAC ACCUMULATOR MSB [35:16] |
| AM (LSB) | MAC ACCUMULATOR LSB [15:0] |

| XMSK | YMSK | MAC XY MASK REGISTER |

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 11 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 1 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 35 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 36** provides a key to the table nomenclature.

Table 35 Instruction Set Summary

Mnemonic	Operation	Description	Address 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 35 Instruction Set Summary (Continued)

Mnemonic	Operation	Description	Address Mode	Opcode	Operand	Cycles	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 35 Instruction Set Summary (Continued)
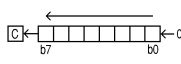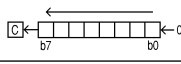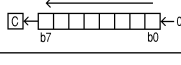
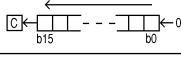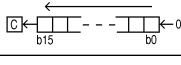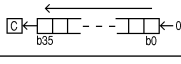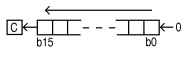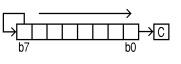| Mnemonic	Operation	Description	Address	Instruction			Condition Codes							
			Mode	Opcode	Operand	Cycles	S	MV	H	EV	N	Z	V	C
ANDB	AND B	(B) • (M) ⇒ B	IND8, X IND8, Y IND8, Z IMM8 IND16, X IND16, Y IND16, Z EXT E, X E, Y E, Z	C6 D6 E6 F6 17C6 17D6 17E6 17F6 27C6 27D6 27E6	ff ff ff ii gggg gggg gggg hh ll — — —	6 6 6 2 6 6 6 6 6 6 6	—	—	—	—	Δ	Δ	0	—
ANDD	AND D	(D) • (M : M + 1) ⇒ D	IND8, X IND8, Y IND8, Z IMM16 IND16, X IND16, Y IND16, Z EXT E, X E, Y E, Z	86 96 A6 37B6 37C6 37D6 37E6 37F6 2786 2796 27A6	ff ff ff jj kk gggg gggg gggg hh ll — — —	6 6 6 4 6 6 6 6 6 6 6	—	—	—	—	Δ	Δ	0	—
ANDE	AND E	(E) • (M : M + 1) ⇒ E	IMM16 IND16, X IND16, Y IND16, Z EXT	3736 3746 3756 3766 3776	jj kk gggg gggg gggg hh ll	4 6 6 6 6	—	—	—	—	Δ	Δ	0	—
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 IND8, Y IND8, Z IND16, X IND16, Y IND16, Z EXT	04 14 24 1704 1714 1724 1734	ff ff ff gggg gggg gggg hh ll	8 8 8 8 8 8 8	—	—	—	—	Δ	Δ	Δ	Δ
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 IND16, Y IND16, Z EXT	2704 2714 2724 2734	gggg gggg gggg hh ll	8 8 8 8	—	—	—	—	Δ	Δ	Δ	Δ
ASR	Arithmetic Shift Right		IND8, X IND8, Y IND8, Z IND16, X IND16, Y IND16, Z EXT	0D 1D 2D 170D 171D 172D 173D	ff ff ff gggg gggg gggg hh ll	8 8 8 8 8 8 8	—	—	—	—	Δ	Δ	Δ	Δ

Table 35 Instruction Set Summary (Continued)
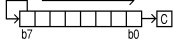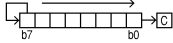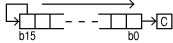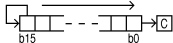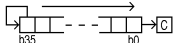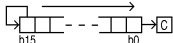
Mnemonic	Operation	Description	Address	Instruction			Condition Codes							
			Mode	Opcode	Operand	Cycles	S	MV	H	EV	N	Z	V	C
ASRA	Arithmetic Shift Right A		INH	370D	—	2	—	—	—	—	Δ	Δ	Δ	Δ
ASRB	Arithmetic Shift Right B		INH	371D	—	2	—	—	—	—	Δ	Δ	Δ	Δ
ASRD	Arithmetic Shift Right D		INH	27FD	—	2	—	—	—	—	Δ	Δ	Δ	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)	$(M) \bullet (\overline{Mask}) \Rightarrow M$	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	$(M : M + 1) \bullet (\overline{Mask}) \Rightarrow$ $M : M + 1$	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	$(A) \bullet (M)$	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	—

Table 35 Instruction Set Summary (Continued)

Mnemonic	Operation	Description	Address Mode	Opcode	Operand	Cycles	S	MV	H	EV	N	Z	V	C
BITB	Bit Test B	(B) • (M)	IND8, X	C9	ff	6	—	—	—	—	Δ	Δ	0	—
			IND8, Y	D9	ff	6								
			IND8, Z	E9	ff	6								
			IMM8	F9	ii	2								
			IND16, X	17C9	gggg	6								
			IND16, Y	17D9	gggg	6								
			IND16, Z	17E9	gggg	6								
			EXT	17F9	hh ll	6								
			E, X	27C9	—	6								
			E, Y	27D9	—	6								
			E, Z	27E9	—	6								
BLE[2]	Branch if Less Than or Equal to Zero	If Z ✛ (N ⊕ V) = 1, branch	REL8	BF	rr	6, 2	—	—	—	—	—	—	—	—
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	CB	mm ff rr	10, 12	—	—	—	—	—	—	—	—
			IND8, Y	DB	mm ff rr	10, 12								
			IND8, Z	EB	mm ff rr	10, 12								
			IND16, X	0A	mm gggg rrrr	10, 14								
			IND16, Y	1A	mm gggg rrrr	10, 14								
			IND16, Z	2A	mm gggg rrrr	10, 14								
			EXT	3A	mm hh ll rrrr	10, 14								
BRN	Branch Never	If 1 = 0, branch	REL8	B1	rr	2	—	—	—	—	—	—	—	—
BRSET[2]	Branch if Bit(s) Set	If ($\overline{M}$) • (Mask) = 0, branch	IND8, X	8B	mm ff rr	10, 12	—	—	—	—	—	—	—	—
			IND8, Y	9B	mm ff rr	10, 12								
			IND8, Z	AB	mm ff rr	10, 12								
			IND16, X	0B	mm gggg rrrr	10, 14								
			IND16, Y	1B	mm gggg rrrr	10, 14								
			IND16, Z	2B	mm gggg rrrr	10, 14								
			EXT	3B	mm hh ll rrrr	10, 14								
BSET	Set Bit(s)	(M) ✛ (Mask) ⇒ M	IND8, X	1709	mm ff	8	—	—	—	—	Δ	Δ	0	Δ
			IND8, Y	1719	mm ff	8								
			IND8, Z	1729	mm ff	8								
			IND16, X	09	mm gggg	8								
			IND16, Y	19	mm gggg	8								
			IND16, Z	29	mm gggg	8								
			EXT	39	mm hh ll	8								
BSETW	Set Bit(s) in Word	(M : M + 1) ✛ (Mask) ⇒ M : M + 1	IND16, X	2709	gggg mmmm	10	—	—	—	—	Δ	Δ	0	Δ
			IND16, Y	2719	gggg mmmm	10								
			IND16, Z	2729	gggg mmmm	10								
			EXT	2739	hh ll mmmm	10								

Table 35 Instruction Set Summary (Continued)

Mnemonic	Operation	Description	Address Mode	Opcode	Operand	Cycles	S	MV	H	EV	N	Z	V	C
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
CLRM	Clear AM	$000000000 ⇒ AM[35:0]	INH	27B7	—	2	—	0	—	0	—	—	—	—
CLRW	Clear a Word in Memory	$0000 ⇒ M : M + 1	IND16, X IND16, Y IND16, Z EXT	2705 2715 2725 2735	gggg gggg gggg hh ll	6 6 6 6	—	—	—	—	0	1	0	0
CMPA	Compare A to Memory	(A) – (M)	IND8, X IND8, Y IND8, Z IMM8 IND16, X IND16, Y IND16, Z EXT E, X E, Y E, Z	48 58 68 78 1748 1758 1768 1778 2748 2758 2768	ff ff ff ii gggg gggg gggg hh ll — — —	6 6 6 2 6 6 6 6 6 6 6	—	—	—	—	Δ	Δ	Δ	Δ
CMPB	Compare B to Memory	(B) – (M)	IND8, X IND8, Y IND8, Z IMM8 IND16, X IND16, Y IND16, Z EXT E, X E, Y E, Z	C8 D8 E8 F8 17C8 17D8 17E8 17F8 27C8 27D8 27E8	ff ff ff ii gggg gggg gggg hh ll — — —	6 6 6 2 6 6 6 6 6 6 6	—	—	—	—	Δ	Δ	Δ	Δ
COM	One's Complement	$FF – (M) ⇒ M, or M̄ ⇒ M	IND8, X IND8, Y IND8, Z IND16, X IND16, Y IND16, Z EXT	00 10 20 1700 1710 1720 1730	ff ff ff gggg gggg gggg hh ll	8 8 8 8 8 8 8	—	—	—	—	Δ	Δ	0	1
COMA	One's Complement A	$FF – (A) ⇒ A, or Ā ⇒ A	INH	3700	—	2	—	—	—	—	Δ	Δ	0	1
COMB	One's Complement B	$FF – (B) ⇒ B, or B̄ ⇒ B	INH	3710	—	2	—	—	—	—	Δ	Δ	0	1
COMD	One's Complement D	$FFFF – (D) ⇒ D, or D̄ ⇒ D	INH	27F0	—	2	—	—	—	—	Δ	Δ	0	1
COME	One's Complement E	$FFFF – (E) ⇒ E, or Ē ⇒ E	INH	2770	—	2	—	—	—	—	Δ	Δ	0	1
COMW	One's Complement Word	$FFFF – M : M + 1 ⇒ M : M + 1, or (M̄ : M̄ + 1) ⇒ M : M + 1	IND16, X IND16, Y IND16, Z EXT	2700 2710 2720 2730	gggg gggg gggg hh ll	8 8 8 8	—	—	—	—	Δ	Δ	0	1

Table 35 Instruction Set Summary (Continued)

Mnemonic	Operation	Description	Address Mode	Opcode	Operand	Cycles	S	MV	H	EV	N	Z	V	C
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								
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

Table 35 Instruction Set Summary (Continued)

Mnemonic	Operation	Description	Address	Instruction			Condition Codes								
			Mode	Opcode	Operand	Cycles	S	MV	H	EV	N	Z	V	C
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 IND8, Y IND8, Z IMM8 IND16, X IND16, Y IND16, Z EXT E, X E, Y E, Z	44 54 64 74 1744 1754 1764 1774 2744 2754 2764	ff ff ff ii gggg gggg gggg hh ll — — —	6 6 6 2 6 6 6 6 6 6 6	—	—	—	—	Δ	Δ	0	—
EORB	Exclusive OR B	(B) ⊕ (M) ⇒ B	IND8, X IND8, Y IND8, Z IMM8 IND16, X IND16, Y IND16, Z EXT E, X E, Y E, Z	C4 D4 E4 F4 17C4 17D4 17E4 17F4 27C4 27D4 27E4	ff ff ff ii gggg gggg gggg hh ll — — —	6 6 6 2 6 6 6 6 6 6 6	—	—	—	—	Δ	Δ	0	—
EORD	Exclusive OR D	(D) ⊕ (M : M + 1) ⇒ D	IND8, X IND8, Y IND8, Z IMM16 IND16, X IND16, Y IND16, Z EXT E, X E, Y E, Z	84 94 A4 37B4 37C4 37D4 37E4 37F4 2784 2794 27A4	ff ff ff jj kk gggg gggg gggg hh ll — — —	6 6 6 4 6 6 6 6 6 6 6	—	—	—	—	Δ	Δ	0	—
EORE	Exclusive OR E	(E) ⊕ (M : M + 1) ⇒ E	IMM16 IND16, X IND16, Y IND16, Z EXT	3734 3744 3754 3764 3774	jj kk gggg gggg gggg hh ll	4 6 6 6 6	—	—	—	—	Δ	Δ	0	—
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 IND8, Y IND8, Z IND16, X IND16, Y IND16, Z EXT	03 13 23 1703 1713 1723 1733	ff ff ff gggg gggg gggg hh ll	8 8 8 8 8 8 8	—	—	—	—	Δ	Δ	Δ	—
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 IND16, Y IND16, Z EXT	2703 2713 2723 2733	gggg gggg gggg hh ll	8 8 8 8	—	—	—	—	Δ	Δ	Δ	—

Table 35 Instruction Set Summary (Continued)

Mnemonic	Operation	Description	Address Mode	Opcode	Operand	Cycles	S	MV	H	EV	N	Z	V	C
JMP	Jump	$\langle ea \rangle \Rightarrow PK : PC$	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) $(SK : SP) - \$0002 \Rightarrow SK : SP$ Push (CCR) $(SK : SP) - \$0002 \Rightarrow SK : SP$ $\langle ea \rangle \Rightarrow PK : PC$	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 $\oplus$ V = 0, branch	REL16	378C	rrrr	6, 4	—	—	—	—	—	—	—	—
LBGT[2]	Long Branch if Greater Than Zero	If Z $\div$ (N $\oplus$ V) = 0, branch	REL16	378E	rrrr	6, 4	—	—	—	—	—	—	—	—
LBHI[2]	Long Branch if Higher	If C $\div$ Z = 0, branch	REL16	3782	rrrr	6, 4	—	—	—	—	—	—	—	—
LBLE[2]	Long Branch if Less Than or Equal to Zero	If Z $\div$ (N $\oplus$ V) = 1, branch	REL16	378F	rrrr	6, 4	—	—	—	—	—	—	—	—
LBLS[2]	Long Branch if Lower or Same	If C $\div$ Z = 1, branch	REL16	3783	rrrr	6, 4	—	—	—	—	—	—	—	—
LBLT[2]	Long Branch if Less Than Zero	If N $\oplus$ 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) $(SK : SP) - 2 \Rightarrow SK : SP$ Push (CCR) $(SK : SP) - 2 \Rightarrow SK : SP$ $(PK : PC) + Offset \Rightarrow PK : PC$	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	$(M) \Rightarrow A$	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								

Table 35 Instruction Set Summary (Continued)

Mnemonic	Operation	Description	Address	Instruction			Condition Codes							
			Mode	Opcode	Operand	Cycles	S	MV	H	EV	N	Z	V	C
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	—	—	—	—	—	—	—	—
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 S̄ then STOP else NOP	INH	27F1	—	4, 20	—	—	—	—	—	—	—	—

Table 35 Instruction Set Summary (Continued)

Mnemonic	Operation	Description	Address	Instruction			Condition Codes							
			Mode	Opcode	Operand	Cycles	S	MV	H	EV	N	Z	V	C
LSR	Logical Shift Right		IND8, X IND8, Y IND8, Z IND16, X IND16, Y IND16, Z EXT	0F 1F 2F 170F 171F 172F 173F	ff ff ff gggg gggg gggg hh ll	8 8 8 8 8 8 8	—	—	—	—	0	Δ	Δ	Δ
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 IND16, Y IND16, Z EXT	270F 271F 272F 273F	gggg gggg gggg hh ll	8 8 8 8	—	—	—	—	0	Δ	Δ	Δ
MAC	Multiply and Accumulate Signed 16-Bit Fractions	$(HR) * (IR) \Rightarrow E : D$ $(AM) + (E : D) \Rightarrow AM$ Qualified $(IX) \Rightarrow IX$ Qualified $(IY) \Rightarrow IY$ $(HR) \Rightarrow IZ$ $(M : M + 1)_X \Rightarrow HR$ $(M : M + 1)_Y \Rightarrow IR$	IMM8	7B	xoyo	12	—	Δ	—	Δ	—	—	Δ	—
MOVB	Move Byte	$(M_1) \Rightarrow M_2$	IXP to EXT EXT to IXP EXT to EXT	30 32 37FE	ff hh ll ff hh ll hh ll hh ll	8 8 10	—	—	—	—	Δ	Δ	0	—
MOVW	Move Word	$(M : M + 1_1) \Rightarrow M : M + 1_2$	IXP to EXT EXT to IXP EXT to EXT	31 33 37FF	ff hh ll ff hh ll hh ll hh ll	8 8 10	—	—	—	—	Δ	Δ	0	—
MUL	Multiply	$(A) * (B) \Rightarrow D$	INH	3724	—	10	—	—	—	—	—	—	—	Δ
NEG	Negate Memory	$\$00 - (M) \Rightarrow M$	IND8, X IND8, Y IND8, Z IND16, X IND16, Y IND16, Z EXT	02 12 22 1702 1712 1722 1732	ff ff ff gggg gggg gggg hh ll	8 8 8 8 8 8 8	—	—	—	—	Δ	Δ	Δ	Δ
NEGA	Negate A	$\$00 - (A) \Rightarrow A$	INH	3702	—	2	—	—	—	—	Δ	Δ	Δ	Δ
NEGB	Negate B	$\$00 - (B) \Rightarrow B$	INH	3712	—	2	—	—	—	—	Δ	Δ	Δ	Δ
NEGD	Negate D	$\$0000 - (D) \Rightarrow D$	INH	27F2	—	2	—	—	—	—	Δ	Δ	Δ	Δ
NEGE	Negate E	$\$0000 - (E) \Rightarrow E$	INH	2772	—	2	—	—	—	—	Δ	Δ	Δ	Δ
NEGW	Negate Memory Word	$\$0000 - (M : M + 1)$ $\Rightarrow M : M + 1$	IND16, X IND16, Y IND16, Z EXT	2702 2712 2722 2732	gggg gggg gggg hh ll	8 8 8 8	—	—	—	—	Δ	Δ	Δ	Δ
NOP	Null Operation	—	INH	274C	—	2	—	—	—	—	—	—	—	—

Table 35 Instruction Set Summary (Continued)

Mnemonic	Operation	Description	Address Mode	Opcode	Operand	Cycles	S	MV	H	EV	N	Z	V	C
ORAA	OR A	$(A) + (M) \Rightarrow A$	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	$(B) + (M) \Rightarrow B$	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	$(D) + (M : M + 1) \Rightarrow D$	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	$(E) + (M : M + 1) \Rightarrow E$	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	$(CCR) + IMM16 \Rightarrow CCR$	IMM16	373B	jj kk	4	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ
PSHA	Push A	$(SK : SP) + \$0001 \Rightarrow SK : SP$ Push (A) $(SK : SP) - \$0002 \Rightarrow SK : SP$	INH	3708	—	4	—	—	—	—	—	—	—	—
PSHB	Push B	$(SK : SP) + \$0001 \Rightarrow SK : SP$ Push (B) $(SK : SP) - \$0002 \Rightarrow SK : SP$	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 $(SK : SP) - 2 \Rightarrow SK : SP$	IMM8	34	ii	4 + 2N N = number of iterations	—	—	—	—	—	—	—	—
PSHMAC	Push MAC Registers	MAC Registers $\Rightarrow$ Stack	INH	27B8	—	14	—	—	—	—	—	—	—	—
PULA	Pull A	$(SK : SP) + \$0002 \Rightarrow SK : SP$ Pull (A) $(SK : SP) - \$0001 \Rightarrow SK : SP$	INH	3709	—	6	—	—	—	—	—	—	—	—
PULB	Pull B	$(SK : SP) + \$0002 \Rightarrow SK : SP$ Pull (B) $(SK : SP) - \$0001 \Rightarrow SK : SP$	INH	3719	—	6	—	—	—	—	—	—	—	—

Table 35 Instruction Set Summary (Continued)

Mnemonic	Operation	Description	Address Mode	Opcode	Operand	Cycles	S	MV	H	EV	N	Z	V	C
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
ROR	Rotate Right Byte		IND8, X IND8, Y IND8, Z IND16, X IND16, Y IND16, Z EXT	0E 1E 2E 170E 171E 172E 173E	ff ff ff gggg gggg gggg hh ll	8 8 8 8 8 8 8	—	—	—	—	Δ	Δ	Δ	Δ
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

Table 35 Instruction Set Summary (Continued)

Mnemonic	Operation	Description	Address Mode	Opcode	Operand	Cycles	S	MV	H	EV	N	Z	V	C
RORW	Rotate Right Word		IND16, X IND16, Y IND16, Z EXT	270E 271E 272E 273E	gggg gggg gggg hh ll	8 8 8 8	—	—	—	—	Δ	Δ	Δ	Δ
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 IND8, Y IND8, Z IMM8 IND16, X IND16, Y IND16, Z EXT E, X E, Y E, Z	42 52 62 72 1742 1752 1762 1772 2742 2752 2762	ff ff ff ii gggg gggg gggg hh ll — — —	6 6 6 2 6 6 6 6 6 6 6	—	—	—	—	Δ	Δ	Δ	Δ
SBCB	Subtract with Carry from B	(B) – (M) – C ⇒ B	IND8, X IND8, Y IND8, Z IMM8 IND16, X IND16, Y IND16, Z EXT E, X E, Y E, Z	C2 D2 E2 F2 17C2 17D2 17E2 17F2 27C2 27D2 27E2	ff ff ff ii gggg gggg gggg hh ll — — —	6 6 6 2 6 6 6 6 6 6 6	—	—	—	—	Δ	Δ	Δ	Δ
SBCD	Subtract with Carry from D	(D) – (M : M + 1) – C ⇒ D	IND8, X IND8, Y IND8, Z IMM16 IND16, X IND16, Y IND16, Z EXT E, X E, Y E, Z	82 92 A2 37B2 37C2 37D2 37E2 37F2 2782 2792 27A2	ff ff ff jj kk gggg gggg gggg hh ll — — —	6 6 6 4 6 6 6 6 6 6 6	—	—	—	—	Δ	Δ	Δ	Δ
SBCE	Subtract with Carry from E	(E) – (M : M + 1) – C ⇒ E	IMM16 IND16, X IND16, Y IND16, Z EXT	3732 3742 3752 3762 3772	jj kk gggg gggg gggg hh ll	4 6 6 6 6	—	—	—	—	Δ	Δ	Δ	Δ
SDE	Subtract D from E	(E) – (D) ⇒ E	INH	2779	—	2	—	—	—	—	Δ	Δ	Δ	Δ
STAA	Store A	(A) ⇒ M	IND8, X IND8, Y IND8, Z IND16, X IND16, Y IND16, Z EXT E, X E, Y E, Z	4A 5A 6A 174A 175A 176A 177A 274A 275A 276A	ff ff ff gggg gggg gggg hh ll — — —	4 4 4 6 6 6 6 4 4 4	—	—	—	—	Δ	Δ	0	—

Table 35 Instruction Set Summary (Continued)

Mnemonic	Operation	Description	Address		Instruction		Condition Codes							
			Mode	Opcode	Operand	Cycles	S	MV	H	EV	N	Z	V	C
STAB	Store B	$(B) \Rightarrow M$	IND8, X IND8, Y IND8, Z IND16, X IND16, Y IND16, Z EXT E, X E, Y E, Z	CA DA EA 17CA 17DA 17EA 17FA 27CA 27DA 27EA	ff ff ff gggg gggg gggg hh ll — — —	4 4 4 6 6 6 6 4 4 4	—	—	—	—	Δ	Δ	0	—
STD	Store D	$(D) \Rightarrow M : M + 1$	IND8, X IND8, Y IND8, Z IND16, X IND16, Y IND16, Z EXT E, X E, Y E, Z	8A 9A AA 37CA 37DA 37EA 37FA 278A 279A 27AA	ff ff ff gggg gggg gggg hh ll — — —	4 4 4 6 6 6 6 6 6 6	—	—	—	—	Δ	Δ	0	—
STE	Store E	$(E) \Rightarrow M : M + 1$	IND16, X IND16, Y IND16, Z EXT	374A 375A 376A 377A	gggg gggg gggg hh ll	6 6 6 6	—	—	—	—	Δ	Δ	0	—
STED	Store Concatenated D and E	$(E) \Rightarrow M : M + 1$ $(D) \Rightarrow M + 2 : M + 3$	EXT	2773	hh ll	8	—	—	—	—	—	—	—	—
STS	Store Stack Pointer	$(SP) \Rightarrow M : M + 1$	IND8, X IND8, Y IND8, Z IND16, X IND16, Y IND16, Z EXT	8F 9F AF 178F 179F 17AF 17BF	ff ff ff gggg gggg gggg hh ll	4 4 4 6 6 6 6	—	—	—	—	Δ	Δ	0	—
STX	Store IX	$(IX) \Rightarrow M : M + 1$	IND8, X IND8, Y IND8, Z IND16, X IND16, Y IND16, Z EXT	8C 9C AC 178C 179C 17AC 17BC	ff ff ff gggg gggg gggg hh ll	4 4 4 6 6 6 6	—	—	—	—	Δ	Δ	0	—
STY	Store IY	$(IY) \Rightarrow M : M + 1$	IND8, X IND8, Y IND8, Z IND16, X IND16, Y IND16, Z EXT	8D 9D AD 178D 179D 17AD 17BD	ff ff ff gggg gggg gggg hh ll	4 4 4 6 6 6 6	—	—	—	—	Δ	Δ	0	—
STZ	Store Z	$(IZ) \Rightarrow M : M + 1$	IND8, X IND8, Y IND8, Z IND16, X IND16, Y IND16, Z EXT	8E 9E AE 178E 179E 17AE 17BE	ff ff ff gggg gggg gggg hh ll	4 4 4 6 6 6 6	—	—	—	—	Δ	Δ	0	—
SUBA	Subtract from A	$(A) - (M) \Rightarrow A$	IND8, X IND8, Y IND8, Z IMM8 IND16, X IND16, Y IND16, Z EXT E, X E, Y E, Z	40 50 60 70 1740 1750 1760 1770 2740 2750 2760	ff ff ff ii gggg gggg gggg hh ll — — —	6 6 6 2 6 6 6 6 6 6 6	—	—	—	—	Δ	Δ	Δ	Δ

Table 35 Instruction Set Summary (Continued)

Mnemonic	Operation	Description	Address Mode	Opcode	Operand	Cycles	S	MV	H	EV	N	Z	V	C
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	—
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	—	—

Table 35 Instruction Set Summary (Continued)

Mnemonic	Operation	Description	Address	Instruction			Condition Codes							
			Mode	Opcode	Operand	Cycles	S	MV	H	EV	N	Z	V	C
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	—	—	—	—	—	—	—	—

Table 35 Instruction Set Summary (Continued)

Mnemonic	Operation	Description	Address	Instruction			Condition Codes							
			Mode	Opcode	Operand	Cycles	S	MV	H	EV	N	Z	V	C
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	—	—	—	—	—	—	—	—
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	—	—	—	—	—	—	—	—

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 36 Instruction Set Abbreviations and Symbols

A	—	Accumulator A		
AM	—	Accumulator M		
B	—	Accumulator B		
CCR	—	Condition code register		
D	—	Accumulator D		
E	—	Accumulator E		

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 37** 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 37 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 12** shows the exception stack frame.

High Address		⇐ SP After Exception Stacking
	Condition Code Register	
High Address	Program Counter	⇐ SP Before Exception Stacking
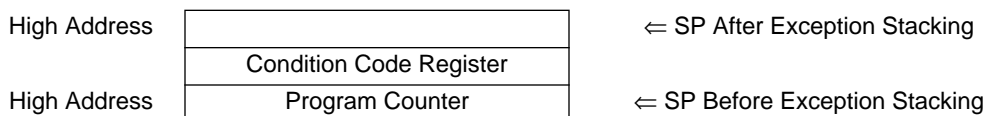
Figure 12 Exception Stack Frame Format

4.7.3 Exception Processing Sequence

Exception processing is performed in four phases.

    A. Priority of all pending exceptions is evaluated, and the highest priority exception is processed first.

    B. Processor state is stacked, then the CCR PK extension field is cleared.

    C. An exception vector number is acquired and converted to a vector address.

    D. 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 Analog-to-Digital Converter Module

The analog-to-digital converter module (ADC) is a unipolar, successive-approximation converter with eight modes of operation. It has selectable 8- or 10-bit resolution. Monotonicity is guaranteed in both modes.

ADC module conversion functions can be grouped into three basic subsystems: an analog front end, a digital control section, and result storage. In addition, the six analog inputs can be used as a general-purpose digital input (port ADA), provided signals are within logic level specification.

Figure 13 shows a block diagram of the ADC converter module.

5.1 Analog Subsystem

The analog front end consists of a multiplexer, a buffer amplifier, a resistor-capacitor array, and a high-gain comparator. The multiplexer selects one of six internal or six external signal sources for conversion. The buffer amplifier protects the input channel from the relatively large capacitance of the resistor capacitor (RC) array. The RC array performs two functions: it acts as a sample/hold circuit, and it provides the digital-to-analog comparison output necessary for successive approximation conversion. The comparator indicates whether each successive output of the RC array is higher or lower than the sampled input.

5.2 Digital Control Subsystem

The digital control section includes conversion sequence control logic, channel and reference select logic, a successive approximation register, eight result registers, a port data register, and control/status registers. It controls the multiplexer and the output of the RC array during the sample and conversion periods, stores the results of comparison in the successive-approximation register, and transfers the result to a result register.

5.3 Bus Interface Subsystem

The ADC bus interface unit contains logic necessary to interface the ADC to the intermodule bus. The ADC is designed to act as a slave device on the bus. The interface must respond with appropriate bus cycle termination signals and supply appropriate interface timing to the other submodules.
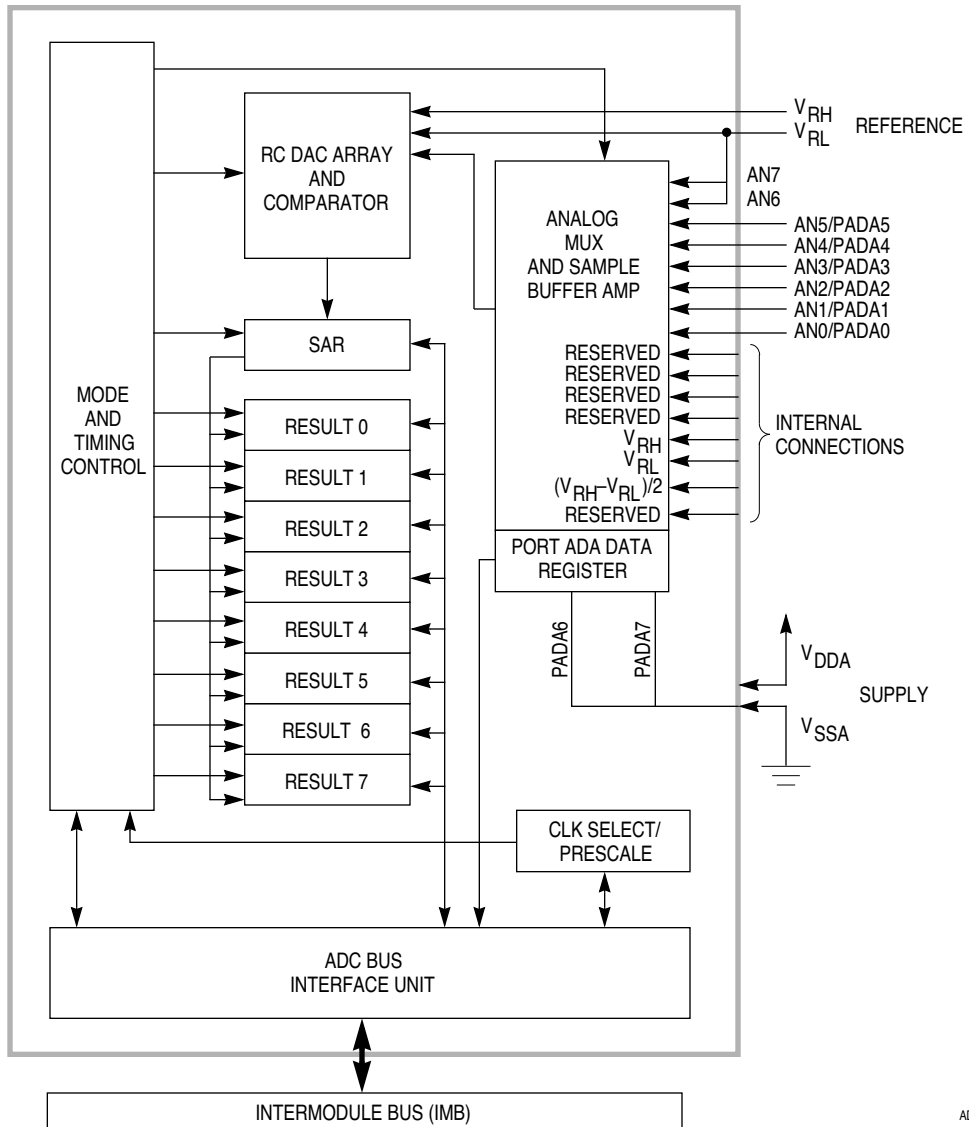
Figure 13 Analog-to-Digital Converter Block Diagram

5.4 ADC Address Map

Table 38 shows the ADC address map.

Table 38 ADC Address Map

Address	15 8	7 0
$YFF700[1]	ADC MODULE CONFIGURATION REGISTER (ADCMCR)	
$YFF702	ADC TEST REGISTER (ADCTEST)	
$YFF704	RESERVED	
$YFF706	PORT ADA DATA (PORTADA)	
$YFF708	RESERVED	
$YFF70A	ADC CONTROL REGISTER 0 (ADCTL0)	
$YFF70C	ADC CONTROL REGISTER 1 (ADCTL1)	
$YFF70E	ADC STATUS REGISTER (ADSTAT)	
$YFF710	RIGHT-JUSTIFIED UNSIGNED ADC RESULT REGISTER 0 (RJURR0)	
$YFF712	RIGHT-JUSTIFIED UNSIGNED ADC RESULT REGISTER 1 (RJURR1)	
$YFF714	RIGHT-JUSTIFIED UNSIGNED ADC RESULT REGISTER 2 (RJURR2)	
$YFF716	RIGHT-JUSTIFIED UNSIGNED ADC RESULT REGISTER 3 (RJURR3)	
$YFF718	RIGHT-JUSTIFIED UNSIGNED ADC RESULT REGISTER 4 (RJURR4)	
$YFF71A	RIGHT-JUSTIFIED UNSIGNED ADC RESULT REGISTER 5 (RJURR5)	
$YFF71C	RIGHT-JUSTIFIED UNSIGNED ADC RESULT REGISTER 6 (RJURR6)	
$YFF71E	RIGHT-JUSTIFIED UNSIGNED ADC RESULT REGISTER 7 (RJURR7)	
$YFF720	LEFT-JUSTIFIED SIGNED ADC RESULT REGISTER 0 (LJSRR0)	
$YFF722	LEFT-JUSTIFIED SIGNED ADC RESULT REGISTER 1 (LJSRR1)	
$YFF724	LEFT-JUSTIFIED SIGNED ADC RESULT REGISTER 2 (LJSRR2)	
$YFF726	LEFT-JUSTIFIED SIGNED ADC RESULT REGISTER 3 (LJSRR3)	
$YFF728	LEFT-JUSTIFIED SIGNED ADC RESULT REGISTER 4 (LJSRR4)	
$YFF72A	LEFT-JUSTIFIED SIGNED ADC RESULT REGISTER 5 (LJSRR5)	
$YFF72C	LEFT-JUSTIFIED SIGNED ADC RESULT REGISTER 6 (LJSRR6)	
$YFF72E	LEFT-JUSTIFIED SIGNED ADC RESULT REGISTER 7 (LJSRR7)	
$YFF730	LEFT-JUSTIFIED UNSIGNED ADC RESULT REGISTER 0 (LJURR0)	
$YFF732	LEFT-JUSTIFIED UNSIGNED ADC RESULT REGISTER 1 (LJURR1)	
$YFF734	LEFT-JUSTIFIED UNSIGNED ADC RESULT REGISTER 2 (LJURR2)	
$YFF736	LEFT-JUSTIFIED UNSIGNED ADC RESULT REGISTER 3 (LJURR3)	
$YFF738	LEFT-JUSTIFIED UNSIGNED ADC RESULT REGISTER 4 (LJURR4)	
$YFF73A	LEFT-JUSTIFIED UNSIGNED ADC RESULT REGISTER 5 (LJURR5)	
$YFF73C	LEFT-JUSTIFIED UNSIGNED ADC RESULT REGISTER 6 (LJURR6)	
$YFF73E	LEFT-JUSTIFIED UNSIGNED ADC RESULT REGISTER 7 (LJURR7)	

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

5.5 ADC Registers

The following section provides a summary of ADC registers and their contents.

ADCMCR — ADC Module Configuration Register   **$YFF700**

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

RESET:

| 1 | 0 | 0 | | | | | | 0 | | | | | | | |

The ADCMCR is used to initialize the ADC.

STOP — STOP Mode
    0 = Normal operation
    1 = Low-power operation
STOP places the ADC in low-power state by disabling the ADC clock and powering down the analog circuitry. Setting STOP aborts any conversion in progress. STOP is set to logic level one at reset, and may be cleared to logic level zero by the CPU.

Clearing STOP enables normal ADC operation. However, because analog circuitry bias current has been turned off, there is a period of recovery before output stabilization.

FRZ[1:0] — FREEZE Assertion Response
The FRZ field is used to determine ADC response to assertion of the IMB FREEZE signal. **Table 39** shows possible responses.

Table 39 FREEZE Assertion Response

FRZ[1:0]	Response
00	Ignore FREEZE
01	Reserved
10	Finish conversion, then freeze
11	Freeze immediately

SUPV — Supervisor/Unrestricted Data Space
This bit has no effect because the CPU16 always operates in the supervisor mode.

ADTEST — ADC Test Register   **$YFF702**

ADTEST is used during factory test of the ADC.

PORTADA — Port ADA Data Register   **$YFF706**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NOT USED								PADA7	PADA6	PADA5	PADA4	PADA3	PADA2	PADA1	PADA0

RESET:

| | | | | | | | | 0 | 0 | REFLECTS STATE OF THE INPUT PINS | | | | | |

Port ADA is an input port that shares pins with the A/D converter inputs.

PADA[7:6] — Port ADA Data Pins
PADA[7:6] digital inputs to the ADC module are internally connected to V_{SSA}. The corresponding bits in PORTADA will read zero at all times.

PADA[5:0] — Port ADA Data Pins

A read of PADA[5:0] returns the logic level of the port ADA pins. If an input is not at an appropriate logic level (i.e. outside the defined levels), the read is indeterminate. Use of a port ADA pin for digital input does not preclude use as an analog input.

ADCTL0 — A/D Control Register 0   **$YFF70A**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NOT USED								RES10	STS[1:0]		PRS[4:0]				

RESET:

| | | | | | | | | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |

ADCTL0 is used to select ADC clock source and to set up prescaling. Writes to it have immediate effect.

RES10 — 10-Bit Resolution
    0 = 8-bit resolution
    1 = 10-bit resolution
Conversion results are appropriately aligned in result registers to reflect conversion status.

STS[1:0] — Sample Time Select
    Total conversion time depends on initial sample time, transfer time, final sample time, and resolution time. Initial sample time is fixed at two clocks. Transfer time is fixed at two clocks. Resolution time is fixed at 10 ADC clock cycles for an 8-bit conversion and 12 ADC clock cycles for a 10-bit conversion. Final sample time depends on the STS[1:0] field. Refer to **Table 40**.

Table 40 Sample Time Select Field

STS[1:0]	Sample Time
00	2 A/D Clock Periods
01	4 A/D Clock Periods
10	8 A/D Clock Periods
11	16 A/D Clock Periods

PRS[4:0] — Prescaler Rate Selection Field
    The ADC clock is generated from the system clock using a modulo counter and a divide-by-two circuit. The binary value of this field is the counter modulus. System clock is divided by the PRS[4:0] value plus one, then sent to the divide-by-two circuit. Refer to **Table 41**.

Table 41 Prescaler Rate Selection Field

PRS[4:0]	Divisor Value
00000	4
00001	4
00010	6
...	...
11101	60
11110	62
11111	64

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NOT USED									SCAN	MULT	S8CM	CD	CC	CB	CA

RESET:

									0	0	0	0	0	0	0

ADCTL1 is used to initiate an A/D conversion and to select conversion modes and a conversion channel. It can be written or read at any time. A write to ADCTL1 initiates a conversion sequence. If a conversion sequence is already in progress, a write to ADCTL1 aborts it and resets the SCF and CCF flags in the ADC status register.

SCAN — Scan Mode Selection Bit
  0 = Single conversion sequence
  1 = Continuous conversion
Length of conversion sequence(s) is determined by S8CM.

MULT — Multichannel Conversion Bit
  0 = Conversion sequence(s) run on single channel (channel selected via [CD:CA])
  1 = Sequential conversion of a block of four or eight channels (block selected via [CD:CA])
Length of conversion sequence(s) is determined by S8CM.

S8CM — Select Eight-Conversion Sequence Mode
  0 = Four-conversion sequence
  1 = Eight-conversion sequence
This bit determines the number of conversions in a conversion sequence.

[CD:CA] — Channel Selection Field
  The bits in this field are used to select an input or block of inputs for A/D conversion.

Table 42 summarizes the operation of S8CM and [CD:CA] when MULT is cleared (single channel mode). Number of conversions per channel is determined by SCAN.

Table 42 Single-Channel Conversions

S8CM	CD	CC	CB	CA	Input	Result Register
0	0	0	0	0	AN0	RSLT[0:3]
0	0	0	0	1	AN1	RSLT[0:3]
0	0	0	1	0	AN2	RSLT[0:3]
0	0	0	1	1	AN3	RSLT[0:3]
0	0	1	0	0	AN4	RSLT[0:3]
0	0	1	0	1	AN5	RSLT[0:3]
0	1	1	1	0	AN6[1]	RSLT[0:3][1]
0	0	1	1	1	AN7[1]	RSLT[0:3][1]
0	1	0	0	0	Reserved	RSLT[0:3]
0	1	0	0	1	Reserved	RSLT[0:3]
0	1	0	1	0	Reserved	RSLT[0:3]
0	1	0	1	1	Reserved	RSLT[0:3]
0	1	1	0	0	V_{RH}	RSLT[0:3]
0	1	1	0	1	V_{RL}	RSLT[0:3]
0	1	1	1	0	$(V_{RH} - V_{RL}) / 2$	RSLT[0:3]
0	1	1	1	1	Test/Reserved	RSLT[0:3]
1	0	0	0	0	AN0	RSLT[0:7]
1	0	0	0	1	AN1	RSLT[0:7]
1	0	0	1	0	AN2	RSLT[0:7]
1	0	0	1	1	AN3	RSLT[0:7]
1	0	1	0	0	AN4	RSLT[0:7]
1	0	1	0	1	AN5	RSLT[0:7]
1	0	1	1	0	AN6[1]	RSLT[0:7][1]
1	0	1	1	1	AN7[1]	RSLT[0:7][1]
1	1	0	0	0	Reserved	RSLT[0:7]
1	1	0	0	1	Reserved	RSLT[0:7]
1	1	0	1	0	Reserved	RSLT[0:7]
1	1	0	1	1	Reserved	RSLT[0:7]
1	1	1	0	0	V_{RH}	RSLT[0:7]
1	1	1	0	1	V_{RL}	RSLT[0:7]
1	1	1	1	0	$(V_{RH} - V_{RL}) / 2$	RSLT[0:7]
1	1	1	1	1	Test/Reserved	RSLT[0:7]

1. AN6 and AN7 are internally connected to V_{RL}. Corresponding result register values always read $0000.

Table 43 summarizes the operation of S8CM and [CD:CA] when MULT is set (multichannel mode). Number of conversions per channel is determined by SCAN. Channel numbers are given in order of conversion.

Table 43 Multiple-Channel Conversions

S8CM	CD	CC	CB	CA	Input	Result Register
0	0	0	X	X	AN0	RSLT0
					AN1	RSLT1
					AN2	RSLT2
					AN3	RSLT3
0	0	1	X	X	AN4	RSLT0
					AN5	RSLT1
					AN6[1]	RSLT2[1]
					AN7[1]	RSLT3[1]
0	1	0	X	X	Reserved	RSLT0
					Reserved	RSLT1
					Reserved	RSLT2
					Reserved	RSLT3
0	1	1	X	X	V_{RH}	RSLT0
					V_{RL}	RSLT1
					$(V_{RH} - V_{RL}) / 2$	RSLT2
					Test/Reserved	RSLT3
1	0	X	X	X	AN0	RSLT0
					AN1	RSLT1
					AN2	RSLT2
					AN3	RSLT3
					AN4	RSLT4
					AN5	RSLT5
					AN6[1]	RSLT6[1]
					AN7[1]	RSLT7[1]
1	1	X	X	X	Reserved	RSLT0
					Reserved	RSLT1
					Reserved	RSLT2
					Reserved	RSLT3
					V_{RH}	RSLT4
					V_{RL}	RSLT5
					$(V_{RH} - V_{RL}) / 2$	RSLT6
					Test/Reserved	RSLT7

1. AN6 and AN7 are internally connected to V_{RL}. Corresponding result register values always read $0000.

ADSTAT — ADC Status Register $YFF70E

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCF	NOT USED				CCTR[2:0]			CCF[7:0]							

RESET:

| 0 | | | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

ADSTAT contains information related to the status of a conversion sequence.

SCF — Sequence Complete Flag
0 = Sequence not complete
1 = Sequence complete
SCF is set at the end of the conversion sequence when SCAN is cleared, and at the end of the first conversion sequence when SCAN is set. SCF is cleared when ADCTL1 is written and a new conversion sequence begins.

CCTR[2:0] — Conversion Counter Field
This field reflects the contents of the conversion counter pointer in either four or eight count conversion sequence. The value corresponds to the number of the next result register to be written, and thus indicates which channel is being converted.

CCF[7:0] — Conversion Complete Field
Each bit in this field corresponds to an A/D result register (CCF7 to RSLT7, etc.). A bit is set when conversion for the corresponding channel is complete, and remains set until the result register is read.

RSLT[0:7] — A/D Result Registers $YFF710–$YFF73E

The result registers are used to store data after conversion is complete. Each register can be read from three different addresses in the register block. Data format depends on the address from which it is read.

RJURR[0:7] — Unsigned Right-Justified Format $YFF710–$YFF71F

Conversion result is unsigned right-justified data. Bits [9:0] are used for 10-bit resolution. For 8-bit conversions, bits [7:0] contain data and bits [9:8] are zero. Bits [15:10] always return zero when read.

LJSRR[0:7] — Signed Left-Justified Format $YFF720–$YFF72F

Conversion result is signed left-justified data. Bits [15:6] are used for 10-bit resolution. For 8-bit conversions, bits [15:8] contain data and bits [7:6] are zero. Although the ADC is unipolar, it is assumed that the zero point is halfway between low and high reference when this format is used — for positive input, bit 15 = 0, for negative input, bit 15 = 1. Bits [5:0] always return zero when read.

LJURR[0:7] — Unsigned Left-Justified Format $YFF730–$YFF73F

Conversion result is unsigned left-justified data. Bits [15:6] are used for 10-bit resolution. For 8-bit conversions, bits [15:8] contain data and bits [7:6] are zero. Bits [5:0] always return zero when read.

6 General-Purpose Timer Module

The GPT is a simple, yet flexible 11-channel timer used in systems where a moderate degree of external visibility and control is required. The GPT consists of a capture/compare unit, a pulse accumulator, and two pulse-width modulators. A bus interface unit connects the GPT to the intermodule bus.

Figure 14 shows a block diagram of the GPT.
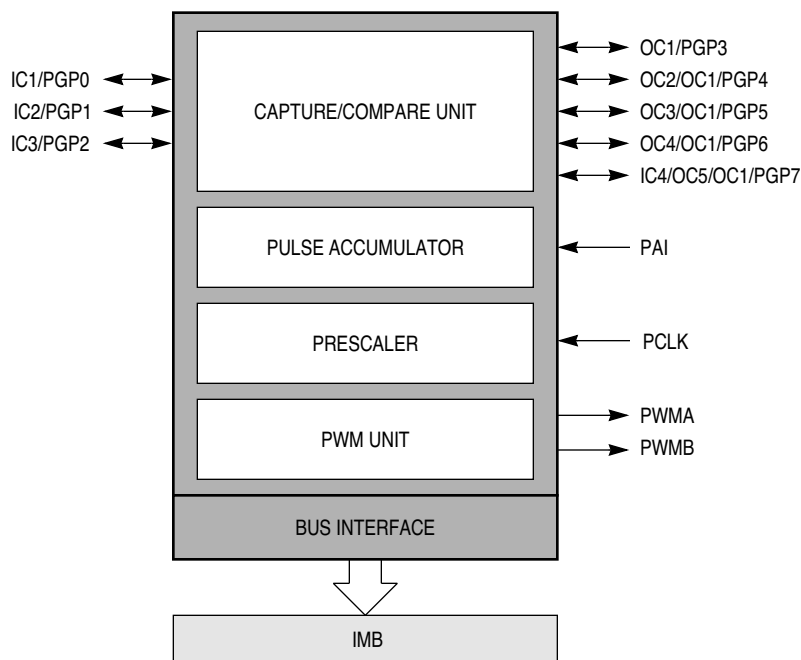
GPT BLOCK

Figure 14 GPT Block Diagram

6.1 Overview

The capture/compare unit features three input capture channels, four output compare channels and one selectable input capture/output compare channel. These channels share a 16-bit free-running counter (TCNT), which derives its clock from a nine-stage prescaler or from the external clock input signal (PCLK).

Pulse accumulator channel logic includes an 8-bit counter; the pulse accumulator can operate in either event counting mode or gated time accumulation mode.

Pulse-width modulator outputs are periodic waveforms whose duty cycles can be independently selected and modified by user software. The PWM circuits share a 16-bit free-running counter that can be clocked by the same nine-stage prescaler used by the capture/compare unit or by the PCLK input.

All GPT pins can also be used for general-purpose input/output. The input capture and output compare pins form a bidirectional 8-bit parallel port (PORTGP). PWM pins are outputs only. PAI and PCLK pins are inputs only.

GPT input capture/output compare pins are bidirectional and can be used to form an 8-bit parallel port. The pulse-width modulator outputs can be used as general-purpose outputs. The PAI and PCLK inputs can be used as general-purpose inputs.

6.2 GPT Address Map

Table 44 shows the GPT address map.

Table 44 GPT Address Map

Address	15　　　　　　　　　　　　　　　　8	7　　　　　　　　　　　　　　　　0
$YFF900[1]	GPT MODULE CONFIGURATION REGISTER (GPTMCR)	
$YFF902	GPT MODULE TEST REGISTER (GPTMTR)	
$YFF904	INTERRUPT CONFIGURATION REGISTER (ICR)	
$YFF906	PGP DATA DIRECTION (DDRGP)	PGP DATA (PORTGP)
$YFF908	OC1 ACTION MASK (OC1M)	OC1 ACTION DATA (OC1D)
$YFF90A	TIMER COUNTER (TCNT)	
$YFF90C	PULSE ACCUMULATOR CONTROL (PACTL)	PULSE ACCUMULATOR COUNTER (PACNT)
$YFF90E	TIMER INPUT CAPTURE 1 (TIC1)	
$YFF910	TIMER INPUT CAPTURE 2 (TIC2)	
$YFF912	TIMER INPUT CAPTURE 3 (TIC3)	
$YFF914	TIMER OUTPUT COMPARE 1 (TOC1)	
$YFF916	TIMER OUTPUT CAPTURE 2 (TOC2)	
$YFF918	TIMER OUTPUT CAPTURE 3 (TOC3)	
$YFF91A	TIMER OUTPUT CAPTURE 4 (TOC4)	
$YFF91C	TIMER INPUT CAPTURE 4/OUTPUT COMPARE 5 (TI4/O5)	
$YFF91E	TIMER CONTROL 1 (TCTL1)	TIMER CONTROL 2 (TCTL2)
$YFF920	TIMER MASK 1 (TMSK1)	TIMER MASK 2 (TMSK2)
$YFF922	TIMER FLAG 1 (TFLG1)	TIMER FLAG 2 (TFLG2)
$YFF924	FORCE COMPARE (CFORC)	PWM CONTROL C (PWMC)
$YFF926	PWMA DUTY CYCLE (PWMA)	PWMB DUTY CYCLE (PWMB)
$YFF928	PWM COUNT REGISTER (PWMCNT)	
$YFF92A	PWMA BUFFER (PWMBUFA)	PWMB BUFFER (PWMBUFB)
$YFF92C	GPT PRESCALER (PRESCL)	
$YFF92E–$YFF93F	RESERVED	

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

6.3 Capture/Compare Unit

The capture/compare unit features three input capture channels, four output compare channels, and one input capture/output compare channel (function selected by control register). Refer to **Figure 15**.

These channels share a 16-bit free-running counter (TCNT), which derives its clock from seven stages of a 9-stage prescaler. Refer to **Figure 16**.

This section also contains one pulse accumulator channel. The pulse accumulator logic includes its own 8-bit counter and can operate in either event counting mode or gated time accumulation mode. Refer to **Figure 17**.
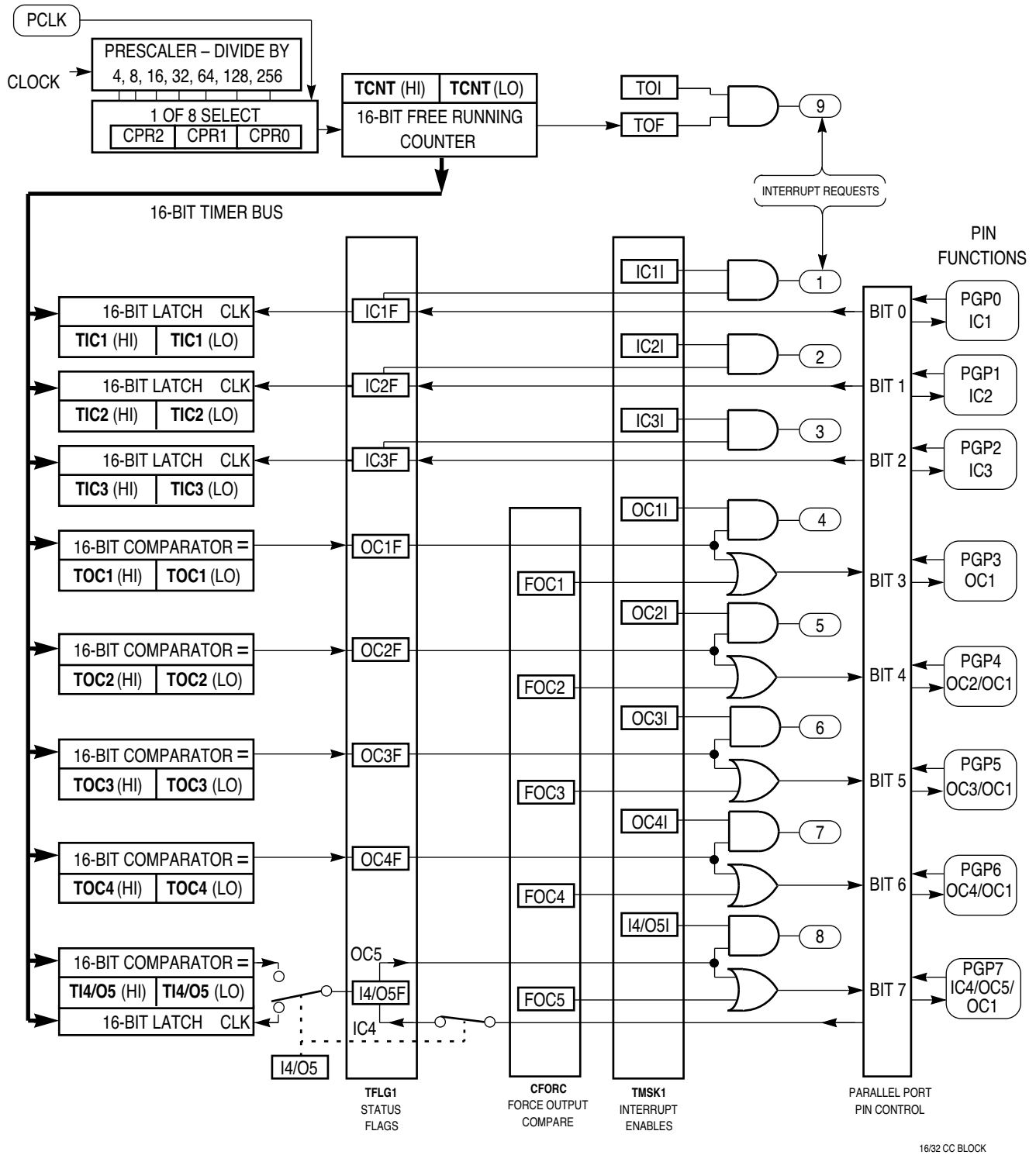
Figure 15 GPT Capture/Compare Block Diagram
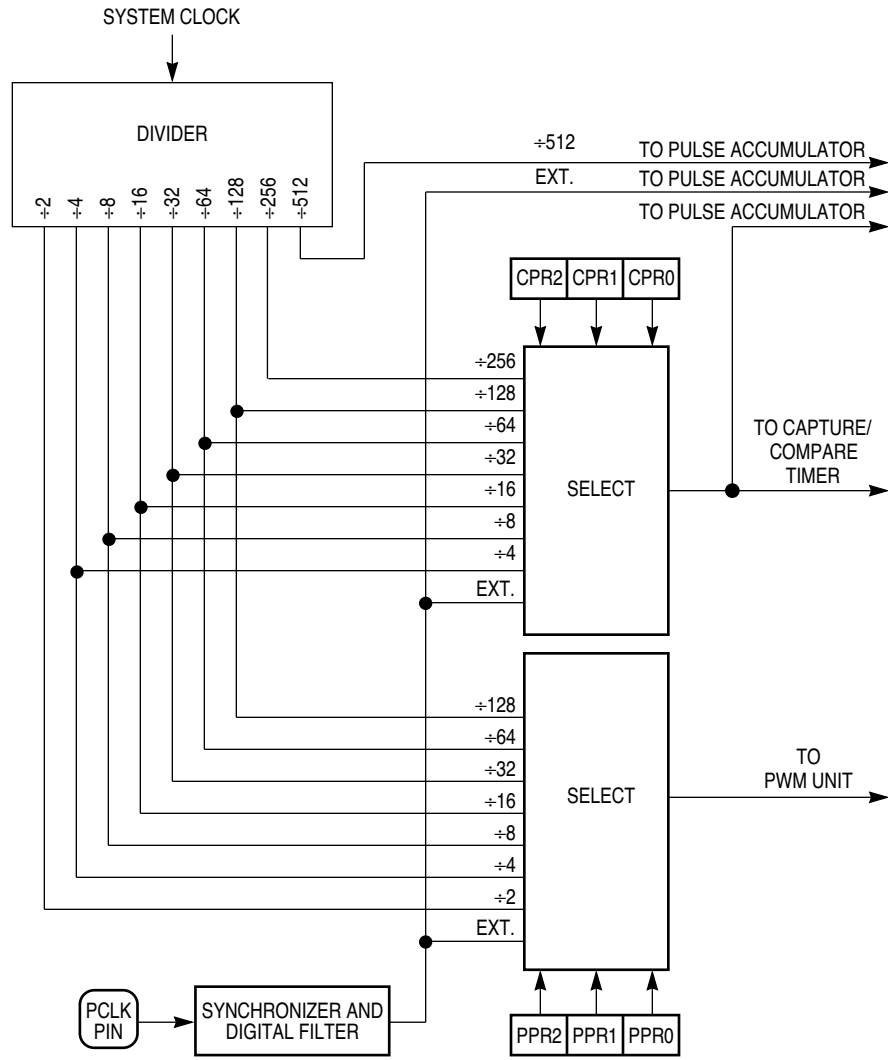
Figure 16 Prescaler Block Diagram
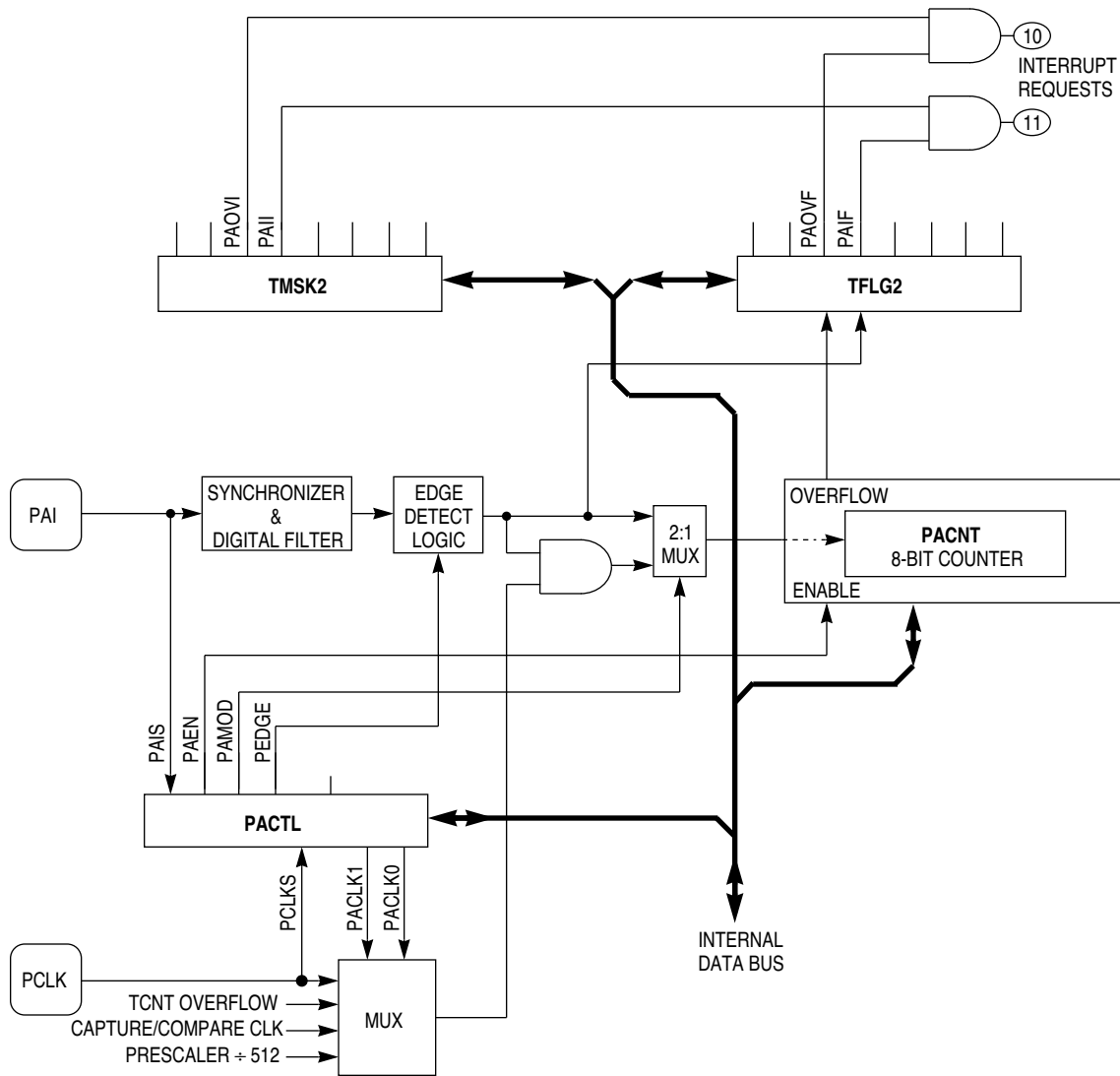
Figure 17 Pulse Accumulator Block Diagram

6.4 Pulse-Width Modulator

The pulse-width modulation submodule has two output pins. The outputs are periodic waveforms controlled by a single frequency whose duty cycles can be independently selected and modified by user software. Each PWM can be independently programmed to run in fast or slow mode. The PWM unit has its own 16-bit free-running counter, which is clocked by an output of the nine-stage prescaler (the same prescaler used by the capture/compare unit) or by the clock input pin, PCLK.

Figure 18 displays a block diagram of the PWM unit.

16-BIT DATA BUS

PWMA REGISTER | PWMB REGISTER

PWMABUF REGISTER | PWMBBUF REGISTER

"A" COMPARATOR | "B" COMPARATOR

PWMA PIN

R LATCH S

PWMB PIN

R LATCH S

F1A BIT

ZERO DETECTOR

8-BIT

8-BIT

ZERO DETECTOR

F1B BIT

SFA BIT

"A" MULTIPLEXER | "B" MULTIPLEXER

SFB BIT

16-BIT COUNTER   [14:0]

FROM PRESCALER CLOCK
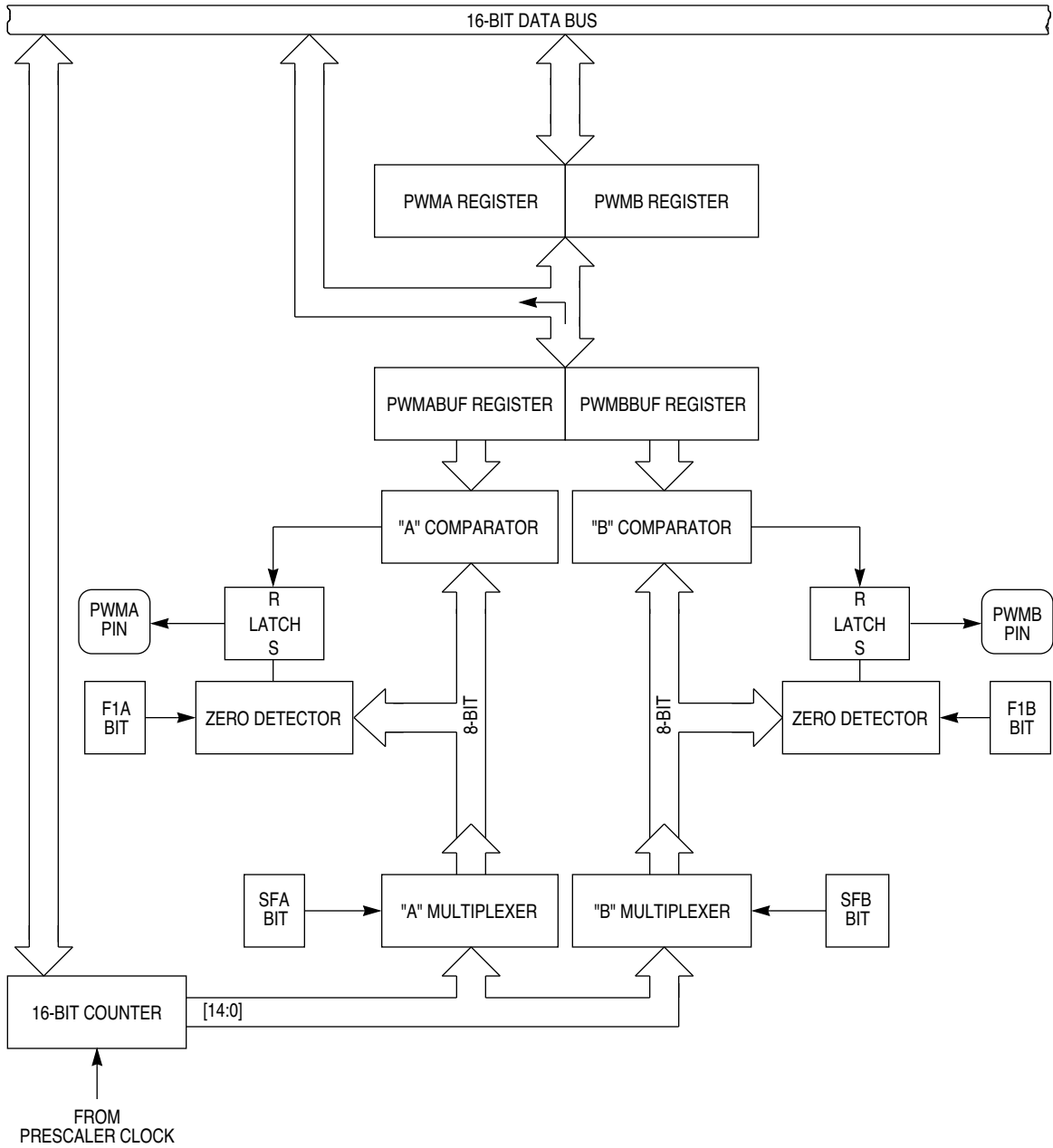
16/32 PWM BLOCK

Figure 18 PWM Unit Block Diagram

6.5 GPT Registers

The following section provides a summary of GPT registers and their contents.

GPTMCR — GPT Module Configuration Register $YFF900

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STOP	FRZ1	FRZ0	STOPP	INCP	0	0	0	SUPV	0	0	0	IARB[3:0]			

RESET:

| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

The GPTMCR contains parameters for configuring the GPT.

STOP — Stop Clocks
    0 = Internal clocks not shut down
    1 = Internal clocks shut down

FRZ1 — Not Implemented

FRZ0 — FREEZE Assertion Response
    0 = Ignore IMB FREEZE Signal
    1 = FREEZE the current state of the GPT

STOPP — Stop Prescaler
    0 = Normal operation
    1 = Stop prescaler and pulse accumulator from incrementing. Ignore changes to input pins.

INCP — Increment Prescaler
    0 = Has no meaning
    1 = If STOPP is asserted, increment prescaler once and clock input synchronizers once.

SUPV — Supervisor/Unrestricted Data Space
    This bit has no effect because the CPU16 always operates in the supervisor mode.

IARB[3:0] — Interrupt Arbitration Identification
    The value in this field is used to arbitrate between simultaneous interrupt service requests of the same priority. Each module that can generate interrupts has an IARB field. In order to implement an arbitration scheme, each IARB field must be set to a different non-zero value. If an interrupt request from a module that has an IARB field value of $0 is recognized, the CPU16 processes a spurious interrupt exception. The reset value of all IARB fields other than that of the SCIM is $0 (no priority), to preclude interrupt processing during reset.

GPTMTR — GPT Module Test Register $YFF902

This address is reserved for GPT factory test.

ICR — GPT Interrupt Configuration Register $YFF904

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IPA[3:0]				0	IPL[2:0]			IVBA[3:0]				0	0	0	0

RESET:

| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

IPA[3:0] — Interrupt Priority Adjust
    This field specifies which GPT interrupt source is given highest internal priority. Refer to **Table 45**.

Table 45 GPT Interrupt Sources

Name	Source Number	Source	Vector Number
—	0000	Adjusted Channel	IVBA : 0000
IC1	0001	Input Capture 1	IVBA : 0001
IC2	0010	Input Capture 2	IVBA : 0010
IC3	0011	Input Capture 3	IVBA : 0011
OC1	0100	Output Compare 1	IVBA : 0100
OC2	0101	Output Compare 2	IVBA : 0101
OC3	0110	Output Compare 3	IVBA : 0110
OC4	0111	Output Compare 4	IVBA : 0111
IC4/OC5	1000	Input Capture 4/Output Compare 5	IVBA : 1000
TO	1001	Timer Overflow	IVBA : 1001
PAOV	1010	Pulse Accumulator Overflow	IVBA : 1010
PAI	1011	Pulse Accumulator Input	IVBA : 1011

IPL[2:0] — Interrupt Priority Level
   This field specifies the priority level of interrupts generated by the GPT.

IVBA[3:0] — Interrupt Vector Base Address
   Most significant nibble of interrupt vector numbers generated by the GPT. Refer to **Table 45**.

DDRGP/PORTGP — Port GP Data Direction Register/Port GP Data Register   **$YFF906**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
			DDRGP								PORTGP				

RESET:

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

   When GPT pins are used as an 8-bit port, DDRGP determines whether pins are input or output and PORTGP holds the 8-bit data.

DDRGP[7:0] — Port GP Data Direction Register
      0 = Input only
      1 = Output

OC1M/OC1D — OC1 Action Mask Register/OC1 Action Data Register   **$YFF908**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	OC1M[5:1]				0	0	0		OC1D[5:1]				0	0	0

RESET:

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

   All OC outputs can be controlled by the action of OC1. OC1M contains a mask that determines which pins are affected. OC1D determines what the outputs are.

OC1M[5:1] — OC1 Mask Field
      0 = Corresponding output compare pin is not affected by OC1 compare.
      1 = Corresponding output compare pin is affected by OC1 compare.
   OC1M[5:1] correspond to OC[5:1].

OC1D[5:1] — OC1 Data Field
      0 = If OC1 mask bit is set, clear the corresponding output compare pin on OC1 match.
      1 = If OC1 mask bit is set, set the corresponding output compare pin on OC1 match.
   OC1D[5:1] correspond to OC[5:1].

TCNT — Timer Counter Register   **$YFF90A**

TCNT is the 16-bit free-running counter associated with the input capture, output compare, and pulse accumulator functions of the GPT module.

PACTL/PACNT — Pulse Accumulator Control Register/Counter   **$YFF90C**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PAIS	PAEN	PAMOD	PEDGE	PCLKS	I4/O5	PACLK[1:0]		PULSE ACCUMULATOR COUNTER							

RESET:

| U | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

PACTL enables the pulse accumulator and selects either event counting or gated mode. In event counting mode, PACNT is incremented each time an event occurs. In gated mode, it is incremented by an internal clock.

PAIS — PAI Pin State (Read Only)

PAEN — Pulse Accumulator Enable
  0 = Pulse accumulator disabled
  1 = Pulse accumulator enabled

PAMOD — Pulse Accumulator Mode
  0 = External event counting
  1 = Gated time accumulation

PEDGE — Pulse Accumulator Edge Control
  The effects of PAMOD and PEDGE are shown in **Table 46**.

Table 46 PAMOD/PEDGE Effects

PAMOD	PEDGE	Effect
0	0	PAI falling edge increments counter
0	1	PAI rising edge increments counter
1	0	Zero on PAI inhibits counting
1	1	One on PAI inhibits counting

PCLKS — PCLK Pin State (Read Only)

I4/O5 — Input Capture 4/Output Compare 5
  0 = Output compare 5 enabled
  1 = Input capture 4 enabled

PACLK[1:0] — Pulse Accumulator Clock Select (Gated Mode)
  Table 47 shows the PACLK[1:0] bit field effects.

Table 47 PACLK[1:0] Bit Field

PACLK[1:0]	Pulse Accumulator Clock Selected
00	System clock divided by 512
01	Same clock used to increment TCNT
10	TOF flag from TCNT
11	External clock, PCLK

PACNT — Pulse Accumulator Counter
  Eight-bit read/write counter used for external event counting or gated time accumulation.

TIC[1:3] — Input Capture Registers 1–3   **$YFF90E, $YFF910, $YFF912**

The input capture registers are 16-bit read-only registers which are used to latch the value of TCNT when a specified transition is detected on the corresponding input capture pin. They are reset to $FFFF.

TOC[1:4] — Output Compare Registers 1–4   **$YFF914, $YFF916, $YFF918, $YFF91A**

The output compare registers are 16-bit read/write registers which can be used as output waveform controls or as elapsed time indicators. For output compare functions, they are written to a desired match value and compared against TCNT to control specified pin actions. They are reset to $FFFF.

TI4/O5 — Input Capture 4/Output Compare 5 Register   **$YFF91C**

This register serves either as input capture register 4 or output compare register 5, depending on the state of I4/O5 in PACTL.

TCTL1/TCTL2 — Timer Control Registers 1–2   **$YFF91E**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OM5	OL5	OM4	OL4	OM3	OL3	OM2	OL2	EDG4B	EDG4A	EDG3B	EDG3A	EDG2B	EDG2A	EDG1B	EDG1A

RESET:

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

TCTL1 determines output compare mode and output logic level. TCTL2 determines the type of input capture to be performed.

OM/OL[5:2] — Output Compare Mode Bits and Output Compare Level Bits
Each pair of bits specifies an action to be taken when output comparison is successful. Refer to **Table 48**.

Table 48 OM/OL[5:2] Bit Field Effects

OM/OL[5:2]	Action Taken
00	Timer disconnected from output logic
01	Toggle OCx output line
10	Clear OCx output line to 0
11	Set OCx output line to 1

EDG[4:1]B/A — Input Capture Edge Control Bits
Each pair of bits configures input sensing logic for the corresponding input capture. Refer to **Table 49**.

Table 49 EDG[4:1]B/A Bit Field Effects

EDG[4:1]B/A	Configuration
00	Capture disabled
01	Capture on rising edge only
10	Capture on falling edge only
11	Capture on any (rising or falling) edge

TMSK1/TMSK2 — Timer Interrupt Mask Registers 1–2   **$YFF920**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I4/O5I		OCI[4:1]				ICI[3:1]		TOI	0	PAOVI	PAII	CPROUT		CPR[2:0]	

RESET:

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

TMSK1 enables OC and IC interrupts. TMSK2 controls pulse accumulator interrupts and TCNT functions.

I4/O5I — Input Capture 4/Output Compare 5 Interrupt Enable
    0 = IC4/OC5 interrupt disabled.
    1 = IC4/OC5 interrupt requested when I4/O5F flag in TFLG1 is set.

OCI[4:1] — Output Compare Interrupt Enable
    0 = OC interrupt disabled
    1 = OC interrupt requested when OC flag set
  OCI[4:1] correspond to OC[4:1].

ICI[3:1] — Input Capture Interrupt Enable
    0 = IC interrupt disabled
    1 = IC interrupt requested when IC flag set
  ICI[3:1] correspond to IC[3:1].

TOI — Timer Overflow Interrupt Enable
    0 = Timer overflow interrupt disabled
    1 = Interrupt requested when TOF flag is set

PAOVI — Pulse Accumulator Overflow Interrupt Enable
    0 = Pulse accumulator overflow interrupt disabled
    1 = Interrupt requested when PAOVF flag is set

PAII — Pulse Accumulator Input Interrupt Enable
    0 = Pulse accumulator interrupt disabled
    1 = Interrupt requested when PAIF flag is set

CPROUT — Compare/Capture Unit Clock Output Enable
    0 = Normal operation for OC1 pin
    1 = TCNT clock driven out OC1 pin

CPR[2:0] — Timer Prescaler/PCLK Select Field
  This field selects one of seven prescaler taps or PCLK to be TCNT input. Refer to **Table 50**.

Table 50 CPR[2:0] Bit Field Effects

CPR[2:0]	System Clock Divide-By Factor
000	4
001	8
010	16
011	32
100	64
101	128
110	256
111	PCLK

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I4/O5F		OCF[4:1]				ICF[3:1]		TOF	0	PAOVF	PAIF	0	0	0	0
RESET:															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

These registers show condition flags that correspond to various GPT events. If the corresponding interrupt enable bit in TMSK1/TMSK2 is set, an interrupt occurs.

I4/O5F — Input Capture 4/Output Compare 5 Flag
    When I4/O5 in PACTL is zero, this flag is set each time TCNT matches the value in TOC5. When I4/O5 in PACTL is one, the flag is set each time a selected edge is detected at the I4/O5 pin.

OCF[4:1] — Output Compare Flags
    An output compare flag is set each time TCNT matches the corresponding TOC register. OCF[4:1] correspond to OC[4:1].

ICF[3:1] — Input Capture Flags
    A flag is set each time a selected edge is detected at the corresponding input capture pin. ICF[3:1] correspond to IC[3:1].

TOF — Timer Overflow Flag
    This flag is set each time TCNT advances from a value of $FFFF to $0000.

PAOVF — Pulse Accumulator Overflow Flag
    This flag is set each time the pulse accumulator counter advances from a value of $FF to $00.

PAIF — Pulse Accumulator Flag
    In event counting mode, this flag is set when an active edge is detected on the PAI pin. In gated time accumulation mode, PAIF is set at the end of the timed period.

CFORC/PWMC — Compare Force Register/PWM Control Register **$YFF924**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FOC[5:1]					0	FPWMA	FPWMB	PPROUT	PPR[2:0]			SFA	SFB	F1A	F1B
RESET:															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Setting a bit in CFORC causes a specific output on OC. PWMC controls operation of the GPT PWM submodule.

FOC[5:1] — Force Output Compare
        0 = Causes no action on corresponding OC pin.
        1 = Causes pin action programmed for corresponding OC pin, but the OC flag is not set.
    FOC[5:1] correspond to OC[5:1].

FPWMA/B — Force PWM Value
        0 = PWM pin A/B is used for PWM functions; normal operation.
        1 = PWM pin A/B is used for discrete output. The value of the F1A/B bit will be driven out on the PWMA/B pin. This is true for PWMA regardless of the state of the PPROUT bit.

PPROUT — PWM Clock Output Enable
        0 = Normal PWM operation on PWMA
        1 = Clock selected by PPR[2:0] is driven out PWMA pin.

PPR[2:0] — PWM Prescaler/PCLK Select

This field selects one of seven prescaler taps, or PCLK, to be PWMCNT input. Refer to **Table 51**.

Table 51 PWM Prescaler/PCLK Select Taps

PPR[2:0]	System Clock Divide-By Factor
000	2
001	4
010	8
011	16
100	32
101	64
110	128
111	PCLK

SFA/B — PWMA/B Slow/Fast Select

    0 = PWMA/B period is 256 PWMCNT increments long.
    1 = PWMA/B period is 32768 PWMCNT increments long.

Table 52 shows the effects of settings on PWM frequency for a 16.78-MHz system clock.

Table 52 Effects of SFA/B Settings on PWM Frequency

PPR[2:0]	Prescaler Tap	SFA/B = 0	SFA/B = 1
000	Div 2 = 8.39 MHz	32.8 kHz	256 Hz
001	Div 4 = 4.19 MHz	16.4 kHz	128 Hz
010	Div 8 = 2.10 MHz	8.19 kHz	64.0 Hz
011	Div 16 = 1.05 MHz	4.09 kHz	32.0 Hz
100	Div 32 = 524 kHz	2.05 kHz	16.0 Hz
101	Div 64 = 262 kHz	1.02 kHz	8.0 Hz
110	Div 128 = 131 kHz	512 Hz	4.0 Hz
111	PCLK	PCLK/256	PCLK/32768

F1A/B — Force Logic Level One on PWMA/B

    0 = Force logic level zero output on PWMA/B pin
    1 = Force logic level one output on PWMA/B pin

PWMA/PWMB — PWM Duty Cycle Registers A/B   **$YFF926, $YFF927**

These registers are associated with the pulse-width value of the PWM output on the corresponding PWM pin. A value of $00 loaded into one of these registers results in a continuously low output on the corresponding pin. A value of $80 results in a 50% duty cycle output. Maximum value ($FF) selects an output that is high for 255/256 of the period.

PWMCNT — PWM Count Register   **$YFF928**

PWMCNT is the 16-bit free-running counter associated with the PWM functions of the GPT module.

PWMBUFA/B — PWM Buffer Registers A/B   **$YFF92A, $YFF92B**

These read-only registers contain values associated with the duty cycles of the corresponding PWM. Reset state for both registers is $00.

PRESCL — GPT Prescaler   **$YFF92C**

The 9-bit prescaler value can be read from bits [8:0] at this address. Bits [15:9] always read as zeros. Reset state is $0000.

7 Queued Serial Module

The QSM contains two serial interfaces, the queued serial peripheral interface (QSPI) and the serial communication interface (SCI). **Figure 19** shows the QSM block diagram.
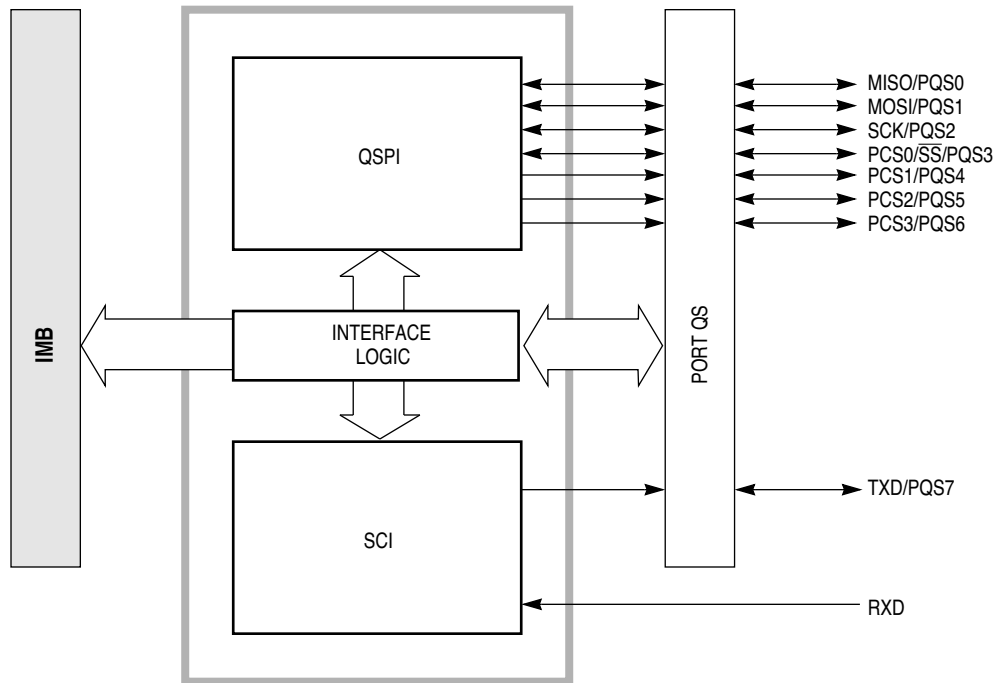
QSM BLOCK

Figure 19 QSM Block Diagram

7.1 Overview

The QSPI provides peripheral expansion or interprocessor communication through a full-duplex, synchronous, three-line bus: data in, data out, and a serial clock. Four programmable peripheral chip-select pins provide addressability for up to 16 peripheral devices. A self-contained RAM queue allows up to 16 serial transfers of 8 to 16 bits each, or transmission of a 256-bit data stream without CPU intervention. A special wraparound mode supports continuous sampling of a serial peripheral, with automatic QSPI RAM updating, which makes the interface to A/D converters more efficient.

The SCI provides a standard nonreturn to zero (NRZ) mark/space format. It operates in either full- or half-duplex mode. There are separate transmitter and receiver enable bits and dual data buffers. A modulus-type baud rate generator provides rates from 64 to 524 kbaud with a 16.78 MHz system clock. Word length of either 8 or 9 bits is software selectable. Optional parity generation and detection provide either even or odd parity check capability. Advanced error detection circuitry catches glitches of up to 1/16 of a bit time in duration. Wake-up functions allow the CPU to run uninterrupted until meaningful data is available.

Table 53 shows the address map of the QSM.

Table 53 QSM Address Map

Address	15 8	7 0
$YFFC00[1]	QSM MODULE CONFIGURATION (QSMCR)	
$YFFC02	QSM TEST (QTEST)	
$YFFC04	QSM INTERRUPT LEVEL (QILR)	QSM INTERRUPT VECTOR (QIVR)
$YFFC06	RESERVED	
$YFFC08	SCI CONTROL 0 REGISTER (SCCR0)	
$YFFC0A	SCI CONTROL 1 REGISTER (SCCR1)	
$YFFC0C	SCI STATUS REGISTER (SCSR)	
$YFFC0E	SCI DATA REGISTER (SCDR)	
$YFFC10	RESERVED	
$YFFC12	RESERVED	
$YFFC14	NOT USED	PQS DATA (PORTQS)
$YFFC16	PQS PIN ASSIGNMENT (PQSPAR)	PQS DATA DIRECTION (DDRQS)
$YFFC18	SPI CONTROL 0 REGISTER (SPCR0)	
$YFFC1A	SPI CONTROL 1 REGISTER (SPCR1)	
$YFFC1C	SPI CONTROL 2 REGISTER (SPCR2)	
$YFFC1E	SPI CONTROL 3 REGISTER (SPCR3)	SPI STATUS REGISTER (SPSR)
$YFFC20–$YFFCFF	RESERVED	
$YFFD00–$YFFD1F	RECEIVE RAM (RR[0:F])	
$YFFD20–$YFFD3F	TRANSMIT RAM (TR[0:F])	
$YFFD40–$YFFD4F	COMMAND RAM (CR[0:F])	

1. Y = M111, where M is the logic state of the MM bit in the SCIMCR.

7.2 Pin Function

Table 54 is a summary of the functions of the QSM pins when they are not configured for general-purpose I/O. The QSM data direction register (DDRQS) designates each pin except RXD as an input or output.

Table 54 QSM Pin Functions

Pin	Mode	Pin Function
MISO	Master	Serial Data Input to QSPI
	Slave	Serial Data Output from QSPI
MOSI	Master	Serial Data Output from QSPI
	Slave	Serial Data Input to QSPI
SCK	Master	Clock Output from QSPI
	Slave	Clock Input to QSPI
PCS0/$\overline{SS}$	Master	Input: Assertion Causes Mode Fault Output: Selects Peripherals
	Slave	Input: Selects the QSPI
PCS[3:1]	Master	Output: Selects Peripherals
	Slave	None
TXD	Transmit	Serial Data Output from SCI
RXD	Receive	Serial Data Input to SCI

7.3 QSM Registers

QSM registers are divided into four categories: QSM global registers, QSM pin control registers, QSPI submodule registers, and SCI submodule registers. The QSPI and SCI registers are defined in separate sections below. Writes to unimplemented register bits have no meaning or effect, and reads from unimplemented bits always return a logic zero value.

7.3.1 Global Registers

The QSM global registers contain system parameters used by both the QSPI and the SCI submodules. These registers contain the bits and fields used to configure the QSM.

QSMCR — QSM Configuration Register   **$YFFC00**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STOP	FRZ1	FRZ0	0	0	0	0	0	SUPV	0	0	0	IARB[3:0]			

RESET:

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

The QSMCR contains parameters for the QSM/CPU/intermodule bus (IMB) interface.

STOP — Stop Enable
    0 = Normal QSM clock operation
    1 = QSM clock operation stopped
STOP places the QSM in a low-power state by disabling the system clock in most parts of the module. The QSMCR is the only register guaranteed to be readable while STOP is asserted. The QSPI RAM is not readable. However, writes to RAM or any register are guaranteed to be valid while STOP is asserted. STOP can be negated by the CPU and by reset.

The system software must stop each submodule before asserting STOP to avoid complications at restart and to avoid data corruption. The SCI submodule receiver and transmitter should be disabled, and the operation should be verified for completion before asserting STOP. The QSPI submodule should be stopped by asserting the HALT bit in SPCR3 and by asserting STOP after the HALTA flag is set.

FRZ1 — FREEZE Assertion Response
    0 = Ignore the FREEZE signal on the IMB
    1 = Halt the QSPI (on a transfer boundary)
FRZ1 determines what action is taken by the QSPI when the FREEZE signal of the IMB is asserted. FREEZE is asserted whenever the CPU enters the background mode.

FRZ0 — Not Implemented

Bits [12:8] — Not Implemented

SUPV — Supervisor/Unrestricted Data Space
    This bit has no effect because the CPU16 always operates in the supervisor mode.

IARB[3:0] — Interrupt Arbitration Identification Number
    The IARB field is used to arbitrate between simultaneous interrupt requests of the same priority. Each module that can generate interrupt requests must be assigned a unique, non-zero IARB field value.

QTEST — QSM Test Register   **$YFFC02**

QTEST is used during factory testing of the QSM.

QILR — QSM Interrupt Levels Register   **$YFFC04**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	ILQSPI[2:0]			ILSCI[2:0]			QIVR							

RESET:

0	0	0	0	0	0	0	0

QILR determines the priority level of interrupts requested by the QSM.

ILQSPI[2:0] — Interrupt Level for QSPI
   ILQSPI determines the priority of QSPI interrupts. This field must be given a value between $0 (interrupts disabled) to $7 (highest priority).

ILSCI[2:0] — Interrupt Level of SCI
   ILSCI determines the priority of SCI interrupts. This field must be given a value between $0 (interrupts disabled) to $7 (highest priority).

   If ILQSPI and ILSCI are the same nonzero value, and both submodules simultaneously request interrupt service, QSPI has priority.

QIVR — QSM Interrupt Vector Register   **$YFFC05**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QILR								INTV[7:0]							

RESET:

0	0	0	0	1	1	1	1

At reset, QIVR is initialized to $0F, which corresponds to the uninitialized interrupt vector in the exception table. This vector is selected until QIVR is written. A user-defined vector ($40–$FF) should be written to QIVR during QSM initialization.

After initialization, QIVR determines which two vectors in the exception vector table are to be used for QSM interrupts. The QSPI and SCI submodules have separate interrupt vectors adjacent to each other. Both submodules use the same interrupt vector with the least significant bit (LSB) determined by the submodule causing the interrupt.

The value of INTV0 used during an interrupt-acknowledge cycle is supplied by the QSM. During an interrupt-acknowledge cycle, INTV[7:1] are driven on DATA[7:1] IMB lines. DATA0 is negated for an SCI interrupt and asserted for a QSPI interrupt. Writes to INTV0 have no meaning or effect. Reads of INTV0 return a value of one.

7.3.2 Pin Control Registers

The QSM uses nine pins, eight of which form a parallel port (PORTQS) on the MCU. Although these pins are used by the serial subsystems, any pin can alternately be assigned as general-purpose I/O on a pin-by-pin basis.

Pins used for general-purpose I/O must not be assigned to the QSPI by register PQSPAR. To avoid driving incorrect data, the first byte to be output must be written before DDRQS is configured. DDRQS must then be written to determine the direction of data flow and to output the value contained in register PORTQS. Subsequent data for output is written to PORTQS.

PORTQS — Port QS Data Register   **$YFFC14**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
			NOT USED					PQS7	PQS6	PQS5	PQS4	PQS3	PQS2	PQS1	PQS0

RESET:

| | | | | | | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

PORTQS latches I/O data. Writes drive pins defined as outputs. Reads return data present on the pins. To avoid driving undefined data, first write a byte to PORTQS, then configure DDRQS.

PQSPAR — PORT QS Pin Assignment Register   **$YFFC16**
DDRQS — PORT QS Data Direction Register   **$YFFC17**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	PQSPA6	PQSPA5	PQSPA4	PQSPA3	0	PQSPA1	PQSPA0	DDQS7	DDQS6	DDQS5	DDQS4	DDQS3	DDQS2	DDQS1	DDQS0

RESET:

| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Clearing a bit in PQSPAR assigns the corresponding pin to general-purpose I/O; setting a bit assigns the pin to the QSPI. PQSPAR does not affect operation of the SCI. **Table 55** displays PQSPAR pin assignments.

Table 55 PQSPAR Pin Assignments

PQSPAR Field	PQSPAR Bit	Pin Function
PQSPA0	0	PQS0
	1	MISO
PQSPA1	0	PQS1
	1	MOSI
—	—	PQS2[1]
	—	SCK
PQSPA3	0	PQS3
	1	PCS0/$\overline{SS}$
PQSPA4	0	PQS4
	1	PCS1
PQSPA5	0	PQS5
	1	PCS2
PQSPA6	0	PQS6
	1	PCS3
—	—	PQS7[2]
	—	TXD

1. PQS2 is a digital I/O pin unless the SPI is enabled (SPE in SPCR1 set), in which case it becomes SPI serial clock SCK.
2. PQS7 is a digital I/O pin unless the SCI transmitter is enabled (TE in SCCR1 = 1), in which case it becomes SCI serial output TXD.

DDRQS determines whether pins are inputs or outputs. Clearing a bit makes the corresponding pin an input; setting a bit makes the pin an output. DDRQS affects both QSPI function and I/O function. **Table 56** shows the effect of DDRQS on QSM pin functions.

Table 56 Effect of DDRQS on QSM Pin Function

QSM Pin	Mode	DDRQS Bit	Bit State	Pin Function
MISO	Master	DDQS0	0	Serial Data Input to QSPI
			1	Disables Data Input
	Slave		0	Disables Data Output
			1	Serial Data Output from QSPI
MOSI	Master	DDQS1	0	Disables Data Output
			1	Serial Data Output from QSPI
	Slave		0	Serial Data Input to QSPI
			1	Disables Data Input
SCK[1]	Master	DDQS2	0	Disables Clock Output
			1	Clock Output from QSPI
	Slave		0	Clock Input to QSPI
			1	Disables Clock Input
PCS0/$\overline{SS}$	Master	DDQS3	0	Assertion Causes Mode Fault
			1	Chip-Select Output
	Slave		0	QSPI Slave Select Input
			1	Disables Select Input
PCS[3:1]	Master	DDQS[4:6]	0	Disables Chip-Select Output
			1	Chip-Select Output
	Slave		0	Inactive
			1	Inactive
TXD[2]	Transmit	DDQS7	X	Serial Data Output from SCI
RXD	Receive	None	NA	Serial Data Input to SCI

1. PQS2 is a digital I/O pin unless the SPI is enabled (SPE in SPCR1 set), in which case it becomes SPI serial clock SCK.
2. PQS7 is a digital I/O pin unless the SCI transmitter is enabled (TE in SCCR1 = 1), in which case it becomes SCI serial output TXD.

DDRQS determines the direction of the TXD pin only when the SCI transmitter is disabled. When the SCI transmitter is enabled, the TXD pin is an output.

7.4 QSPI Submodule

The QSPI submodule communicates with external devices through a synchronous serial bus. The QSPI is fully compatible with the serial peripheral interface (SPI) systems found on other Motorola products. **Figure 20** shows a block diagram of the QSPI.
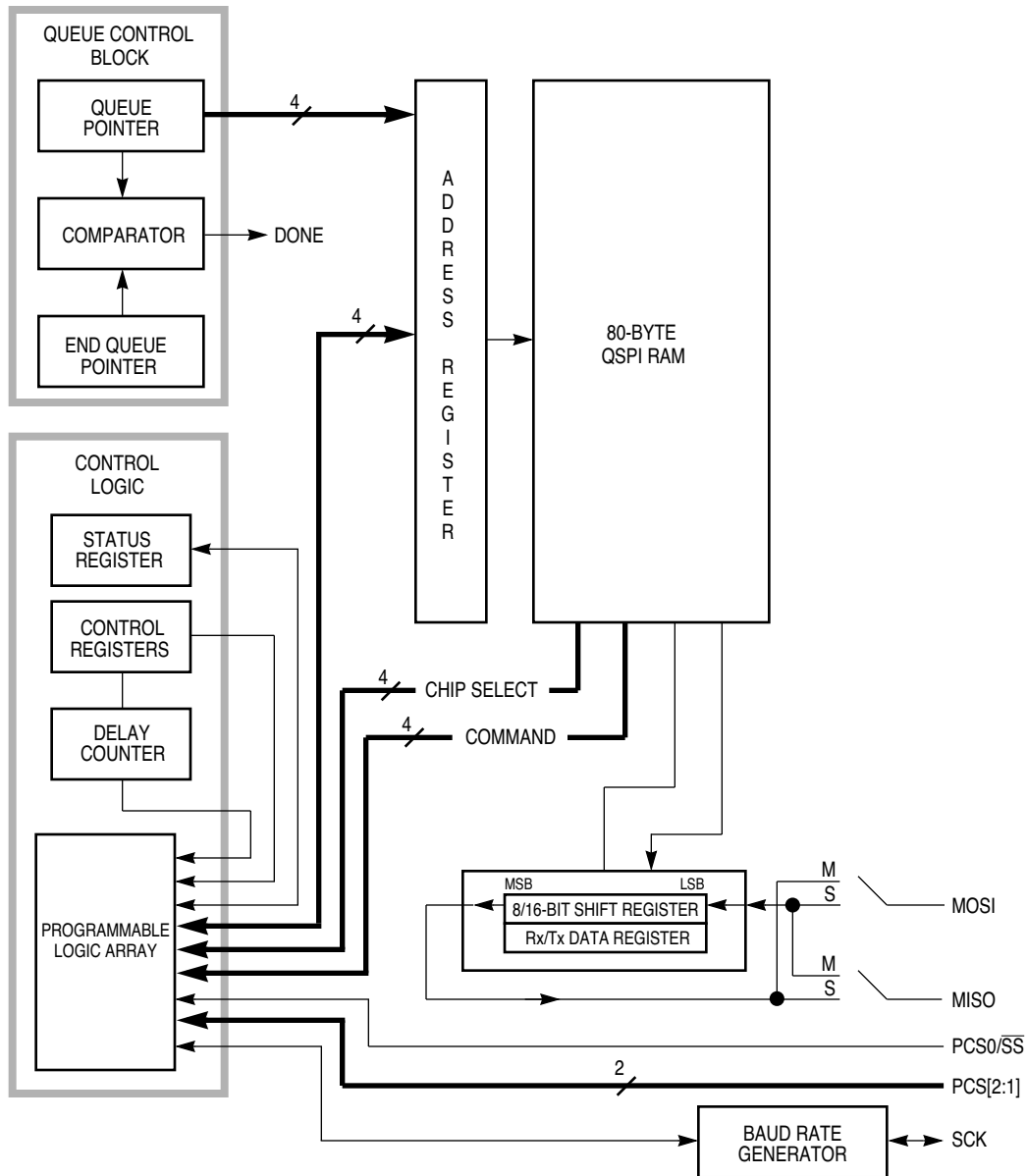
QSPI BLOCK

Figure 20 QSPI Block Diagram

7.4.1 QSPI Pins

Seven pins are associated with the QSPI. When not needed for a QSPI function, they can be configured as general-purpose I/O pins. The PCS0/$\overline{SS}$ pin can function as a peripheral chip select output, slave select input, or general-purpose I/O. Refer to **Table 57** for QSPI input and output pins and their functions.

Table 57 QSPI Pins

Pin Name(s)	Mnemonic(s)	Mode	Function
Master In Slave Out	MISO	Master Slave	Serial Data Input to QSPI Serial Data Output from QSPI
Master Out Slave In	MOSI	Master Slave	Serial Data Output from QSPI Serial Data Input to QSPI
Serial Clock	SCK	Master Slave	Clock Output from QSPI Clock Input to QSPI
Peripheral Chip Selects	PCS[3:1]	Master	Select Peripherals
Peripheral Chip Select Slave Select	PCS0 $\overline{SS}$	Master Master Slave	Selects Peripheral Causes mode fault Initiates Serial Transfer

7.4.2 QSPI Registers

The programmer's model for the QSPI submodule consists of the QSM global and pin control registers, four QSPI control registers, one status register, and the 80-byte QSPI RAM.

The CPU can read and write to registers and RAM. The four control registers must be initialized before the QSPI is enabled to ensure defined operation. SPCR1 should be written last because it contains QSPI enable bit SPE. Asserting this bit starts the QSPI. The QSPI control registers are reset to a defined state and can then be changed by the CPU. Reset values are shown below each register.

Table 58 shows a memory map of the QSPI.

Table 58 QSPI Memory Map

Address	Name	Usage
$YFFC18	SPCR0	QSPI Control Register 0
$YFFC1A	SPCR1	QSPI Control Register 1
$YFFC1C	SPCR2	QSPI Control Register 2
$YFFC1E	SPCR3	QSPI Control Register 3
$YFFC1F	SPSR	QSPI Status Register
$YFFD00	RR[0:F]	QSPI Receive Data (16 Words)
$YFFD20	TR[0:F]	QSPI Transmit Data (16 Words)
$YFFD40	CR[0:F]	QSPI Command Control (8 Words)

Writing a different value into any control register except SPCR2 while the QSPI is enabled disrupts operation. SPCR2 is buffered to prevent disruption of the current serial transfer. After completion of the current serial transfer, the new SPCR2 values become effective.

Writing the same value into any control register except SPCR2 while the QSPI is enabled has no effect on QSPI operation. Rewriting NEWQP[3:0] in SPCR2 causes execution to restart at the designated location.

SPCR0 — QSPI Control Register 0   **$YFFC18**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MSTR	WOMQ			BITS[3:0]		CPOL	CPHA				SPBR[7:0]				

RESET:

| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |

SPCR0 contains parameters for configuring the QSPI before it is enabled. The CPU can read and write this register. The QSM has read-only access.

MSTR — Master/Slave Mode Select
    0 = QSPI is a slave device and only responds to externally generated serial data.
    1 = QSPI is system master and can initiate transmission to external SPI devices.
MSTR configures the QSPI for either master or slave mode operation. This bit is cleared on reset and may only be written by the CPU.

WOMQ — Wired-OR Mode for QSPI Pins
    0 = Outputs have normal MOS drivers.
    1 = Pins designated for output by DDRQS have open-drain drivers.
WOMQ allows the wired-OR function to be used on QSPI pins, regardless of whether they are used as general-purpose outputs or as QSPI outputs. WOMQ affects the QSPI pins regardless of whether the QSPI is enabled or disabled.

BITS[3:0] — Bits Per Transfer
In master mode, when BITSE in a command is set, the BITS[3:0] field determines the number of data bits transferred. When BITSE is cleared, eight bits are transferred. Reserved values default to eight bits. In slave mode, the command RAM is not used and the setting of BITSE has no effect on QSP1 transfers. Instead, the BITS[3:0] field determines the number of bits the QSPI will receive during each transfer before storing the received data.

Table 59 shows the number of bits per transfer.

Table 59 Bits Per Transfer

BITS[3:0]	Bits per Transfer
0000	16
0001	Reserved
0010	Reserved
0011	Reserved
0100	Reserved
0101	Reserved
0110	Reserved
0111	Reserved
1000	8
1001	9
1010	10
1011	11
1100	12
1101	13
1110	14
1111	15

CPOL — Clock Polarity
> 0 = The inactive state value of SCK is logic level zero.
> 1 = The inactive state value of SCK is logic level one.

CPOL is used to determine the inactive state value of the serial clock (SCK). It is used with CPHA to produce a desired clock/data relationship between master and slave devices.

CPHA — Clock Phase
> 0 = Data is captured on the leading edge of SCK and changed on the following edge of SCK.
> 1 = Data is changed on the leading edge of SCK and captured on the following edge of SCK.

CPHA determines which edge of SCK causes data to change and which edge causes data to be captured. CPHA is used with CPOL to produce a desired clock/data relationship between master and slave devices. CPHA is set at reset.

SPBR[7:0] — Serial Clock Baud Rate

The QSPI uses a modulus counter to derive SCK baud rate from the MCU system clock. Baud rate is selected by writing a value from 2 to 255 into the SPBR[7:0] field. The following equation determines the SCK baud rate:

$$\text{SCK Baud Rate} = \frac{\text{System Clock}}{2 \times \text{SPBR[7:0]}}$$

or

$$\text{SPBR[7:0]} = \frac{\text{System Clock}}{2 \times \text{SCK Baud Rate Desired}}$$

Giving SPBR[7:0] a value of zero or one disables the baud rate generator. SCK is disabled and assumes its inactive state value. No serial transfers occur. At reset, baud rate is initialized to one eighth of the system clock frequency.

SPRC1 — QSPI Control Register 1   **$YFFC1A**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SPE				DSCKL[6:0]							DTL[7:0]				

RESET:

0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0

SPCR1 contains parameters for configuring the QSPI before it is enabled. The CPU can read and write this register, but the QSM has read access only, except for SPE, which is automatically cleared by the QSPI after completing all serial transfers, or when a mode fault occurs.

SPE — QSPI Enable
> 0 = QSPI is disabled. QSPI pins can be used for general-purpose I/O.
> 1 = QSPI is enabled. Pins allocated by PQSPAR are controlled by the QSPI.

DSCKL[6:0] — Delay before SCK

When the DSCK bit in command RAM is set, this field determines the length of delay from PCS valid to SCK transition. PCS can be any of the four peripheral chip-select pins. The following equation determines the actual delay before SCK:

$$\text{PCS to SCK Delay} = \frac{\text{DSCKL[6:0]}}{\text{System Clock}}$$

where DSCKL[6:0] equals {1, 2, 3,..., 127}.

When the DSCK value of a queue entry equals zero, then DSCKL[6:0] is not used. Instead, the PCS valid-to-SCK transition is one-half SCK period.

DTL[7:0] — Length of Delay after Transfer
   When the DT bit in command RAM is set, this field determines the length of delay after serial transfer.
   The following equation is used to calculate the delay:

$$\text{Delay after Transfer} = \frac{32 \times \text{DTL[7:0]}}{\text{System Clock}}$$

where DTL equals {1, 2, 3,..., 255}.

A zero value for DTL[7:0] causes a delay-after-transfer value of 8192/System Clock.

If DT equals zero, a standard delay is inserted.

$$\text{Standard Delay after Transfer} = \frac{17}{\text{System Clock}}$$

Delay after transfer can be used to provide a peripheral deselect interval. A delay can also be inserted between consecutive transfers to allow serial A/D converters to complete conversion.

SPCR2 — QSPI Control Register 2   **$YFFC1C**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SPIFIE	WREN	WRTO	0		ENDQP[3:0]			0	0	0	0		NEWQP[3:0]		

RESET:

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SPCR2 contains QSPI configuration parameters. The CPU can read and write this register; the QSM has read access only. Writes to SPCR2 are buffered. A write to SPCR2 that changes a bit value while the QSPI is operating is ineffective on the current serial transfer, but becomes effective on the next serial transfer. Reads of SPCR2 return the current value of the register, not of the buffer.

SPIFIE — SPI Finished Interrupt Enable
   0 = QSPI interrupts disabled
   1 = QSPI interrupts enabled
   SPIFIE enables the QSPI to generate a CPU interrupt upon assertion of the status flag SPIF.

WREN — Wrap Enable
   0 = Wraparound mode disabled
   1 = Wraparound mode enabled
   WREN enables or disables wraparound mode.

WRTO — Wrap To
   When wraparound mode is enabled, after the end of queue has been reached, WRTO determines which address the QSPI executes.

Bit 12 — Not Implemented

ENDQP[3:0] — Ending Queue Pointer
   This field contains the last QSPI queue address.

Bits [7:4] — Not Implemented

NEWQP[3:0] — New Queue Pointer Value
   This field contains the first QSPI queue address.

SPCR3 — QSPI Control Register   **$YFFC1E**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	LOOPQ	HMIE	HALT				SPSR				

RESET:

0	0	0	0	0	0	0	0

SPCR3 contains QSPI configuration parameters. The CPU can read and write SPCR3, but the QSM has read-only access.

Bits [15:11] — Not Implemented

LOOPQ — QSPI Loop Mode
    0 = Feedback path disabled
    1 = Feedback path enabled
    LOOPQ controls feedback on the data serializer for testing.

HMIE — HALTA and MODF Interrupt Enable
    0 = HALTA and MODF interrupts disabled
    1 = HALTA and MODF interrupts enabled
    HMIE controls CPU interrupts caused by the HALTA status flag or the MODF status flag in SPSR.

HALT — Halt
    0 = Halt not enabled
    1 = Halt enabled
    When HALT is asserted, the QSPI stops on a queue boundary. It is in a defined state from which it can later be restarted.

SPSR — QSPI Status Register   **$YFFC1F**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
			SPCR3					SPIF	MODF	HALTA	0		CPTQP[3:0]		

RESET:

0	0	0	0	0	0	0	0

SPSR contains QSPI status information. Only the QSPI can assert the bits in this register. The CPU reads this register to obtain status information and writes it to clear status flags.

SPIF — QSPI Finished Flag
    0 = QSPI not finished
    1 = QSPI finished
    SPIF is set after execution of the command at the address in ENDQP[3:0].

MODF — Mode Fault Flag
    0 = Normal operation
    1 = Another SPI node requested to become the network SPI master while the QSPI was enabled
        in master mode ($\overline{SS}$ input taken low).
    The QSPI asserts MODF when the QSPI is the serial master (MSTR = 1) and the $\overline{SS}$ input pin is negated by an external driver.

HALTA — Halt Acknowledge Flag
    0 = QSPI not halted
    1 = QSPI halted
    HALTA is asserted when the QSPI halts in response to CPU assertion of HALT.

Bit 4 — Not Implemented

CPTQP[3:0] — Completed Queue Pointer

CPTQP[3:0] points to the last command executed. It is updated when the current command is complete. When the first command in a queue is executing, CPTQP[3:0] contains either the reset value ($0) or a pointer to the last command completed in the previous queue.

7.4.3 QSPI RAM

The QSPI contains an 80-byte block of dual-access static RAM that is used by both the QSPI and the CPU. The RAM is divided into three segments: receive data, transmit data, and command control data. Receive data is information received from a serial device external to the MCU. Transmit data is information stored by the CPU for transmission to an external peripheral. Command control data is used to perform the transfer.

Figure 21 displays the organization of the RAM.

```
 500 |  RR0  |        520 |  TR0  |        540 |  CR0  |
     |- RR1 -|            |- TR1 -|            |- CR1 -|
     |- RR2 -|            |- TR2 -|            |- CR2 -|
     |       |            |       |            |       |
     |RECEIVE|            |TRANSMIT|           |COMMAND|
     |  RAM  |            |  RAM  |            |  RAM  |
     |       |            |       |            |       |
     |- RRD -|            |- TRD -|            |- CRD -|
     |- RRE -|            |- TRE -|            |- CRE -|
 51E |  RRF  |        53E |  TRF  |        54F |  CRF  |
        WORD                 WORD                BYTE
```
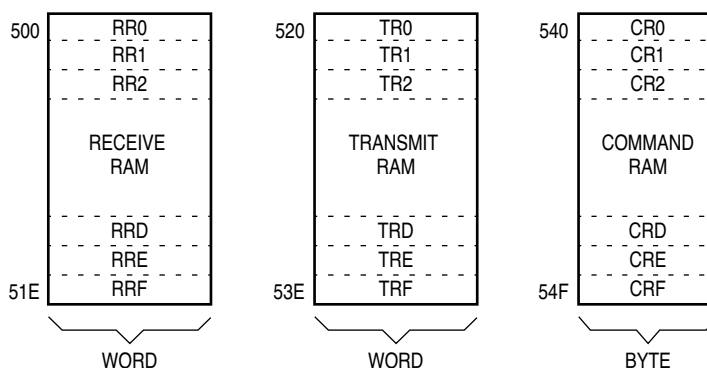
QSPI RAM MAP

Figure 21 QSPI RAM

Once the CPU has set up the queue of QSPI commands and enabled the QSPI, the QSPI can operate independently of the CPU. The QSPI executes all of the commands in its queue, sets a flag indicating that it is finished, and then either interrupts the CPU or waits for CPU intervention. It is possible to execute a queue of commands repeatedly without CPU intervention.

RR[0:F] — Receive Data RAM   **$YFFD00**

Data received by the QSPI is stored in this segment. The CPU reads this segment to retrieve data from the QSPI. Data stored in receive RAM is right-justified. Unused bits in a receive queue entry are set to zero by the QSPI upon completion of the individual queue entry. The CPU can access the data using byte, word, or long-word addressing.

The CPTQP[3:0] value in SPSR shows which queue entries have been executed. The CPU uses this information to determine which locations in receive RAM contain valid data before reading them.

TR[0:F] — Transmit Data RAM   **$YFFD20**

Data that is to be transmitted by the QSPI is stored in this segment. The CPU usually writes one word of data into this segment for each queue command to be executed.

Information to be transmitted must be written to transmit data RAM in a right-justified format. The QSPI cannot modify information in the transmit data RAM. The QSPI copies the information to its data serializer for transmission. Information remains in transmit RAM until overwritten.

7	6	5	4	3	2	1	0
CONT	BITSE	DT	DSCK	PCS3	PCS2	PCS1	PCS0[1]

–	–	–	–	–	–	–	–

CONT	BITSE	DT	DSCK	PCS3	PCS2	PCS1	PCS0[1]

COMMAND CONTROL   PERIPHERAL CHIP SELECT

1. The PCS0 bit represents the dual-function PCS0/$\overline{SS}$.

Command RAM is used by the QSPI when in master mode. The CPU writes one byte of control information to this segment for each QSPI command to be executed. The QSPI cannot modify information in command RAM.

Command RAM consists of 16 bytes. Each byte is divided into two fields. The peripheral chip-select field enables peripherals for transfer. The command control field provides transfer options.

A maximum of 16 commands can be in the queue. Queue execution by the QSPI proceeds from the address in NEWQP[3:0] through the address in ENDQP[3:0]. (Both of these fields are in SPCR2).

CONT — Continue
    0 = Control of chip selects returned to PORTQS after transfer is complete.
    1 = Peripheral chip selects remain asserted after transfer is complete.

BITSE — Bits per Transfer Enable
    0 = 8 bits
    1 = Number of bits set in BITS[3:0] field of SPCR0

DT — Delay after Transfer
The QSPI provides a variable delay at the end of serial transfer to facilitate the interface with peripherals that have a latency requirement. The delay between transfers is determined by the SPCR1 DTL[6:0] field.

DSCK — PCS to SCK Delay
    0 = PCS valid to SCK transition is one-half SCK.
    1 = SPCR1 DSCKL[6:0] field specifies delay from PCS valid to SCK.

PCS[3:0] — Peripheral Chip Select
Use peripheral chip-select bits to select an external device for serial data transfer. More than one peripheral chip select can be activated at a time, and more than one peripheral chip can be connected to each PCS pin, provided that proper fanout is observed.

7.4.4 Operating Modes

The QSPI operates in either master or slave mode. Master mode is used when the MCU originates data transfers. Slave mode is used when an external device initiates serial transfers to the MCU through the QSPI. Switching between the modes is controlled by MSTR in SPCR0. Before entering either mode, appropriate QSM and QSPI registers must be properly initialized.

In master mode, the QSPI executes a queue of commands defined by control bits in each command RAM queue entry. Chip-select pins are activated, data is transmitted from transmit RAM and received into receive RAM.

In slave mode, operation proceeds in response to $\overline{SS}$ pin activation by an external bus master. Operation is similar to master mode, but no peripheral chip selects are generated, and the number of bits transferred is controlled in a different manner. When the QSPI is selected, it automatically executes the next queue transfer to exchange data with the external device correctly.

Although the QSPI inherently supports multi-master operation, no special arbitration mechanism is provided. A mode fault flag (MODF) indicates a request for SPI master arbitration. System software must provide arbitration. Note that unlike previous SPI systems, MSTR is not cleared by a mode fault being set, nor are the QSPI pin output drivers disabled. The QSPI and associated output drivers must be disabled by clearing SPE in SPCR1.

7.5 SCI Submodule

The SCI submodule is used to communicate with external devices through an asynchronous serial bus. The SCI is fully compatible with the SCI systems found on other Motorola MCUs, such as the M68HC11 and M68HC05 Families.

7.5.1 SCI Pins

There are two unidirectional pins associated with the SCI. The SCI controls the transmit data (TXD) pin when enabled, whereas the receive data (RXD) pin remains a dedicated input pin to the SCI. TXD is available as a general-purpose I/O pin when the SCI transmitter is disabled. When used for I/O, TXD can be configured either as input or output, as determined by QSM register DDRQS.

Table 60 shows SCI pins and their functions.

Table 60 SCI Pins

Pin Names	Mnemonics	Mode	Function
Receive Data	RXD	Receiver Disabled Receiver Enabled	Not Used Serial Data Input to SCI
Transmit Data	TXD	Transmitter Disabled Transmitter Enabled	General-Purpose I/O Serial Data Output from SCI

7.5.2 SCI Registers

The SCI programming model includes QSM global and pin control registers, and four SCI registers. There are two SCI control registers, one status register, and one data register. All registers can be read or written at any time by the CPU.

Changing the value of SCI control bits during a transfer operation may disrupt operation. Before changing register values, allow the transmitter to complete the current transfer, then disable the receiver and transmitter. Status flags in the SCSR may be cleared at any time.

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

RESET:

| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |

SCCR0 contains a baud rate selection parameter. Baud rate must be set before the SCI is enabled. The CPU can read and write this register at any time.

Bits [15:13] — Not Implemented

SCBR[12:0] — Baud Rate
SCI baud rate is programmed by writing a 13-bit value to BR. The baud rate is derived from the MCU system clock by a modulus counter.

The SCI receiver operates asynchronously. An internal clock is necessary to synchronize with an incoming data stream. The SCI baud rate generator produces a receiver sampling clock with a frequency 16 times that of the expected baud rate of the incoming data. The SCI determines the position of bit boundaries from transitions within the received waveform, and adjusts sampling points to the proper positions within the bit period. Receiver sampling rate is always 16 times the frequency of the SCI baud rate, which is calculated as follows:

$$\text{SCI Baud Rate} = \frac{\text{System Clock}}{32 \times \text{SCBR[12:0]}}$$

or

$$\text{SCBR[12:0]} = \frac{\text{System Clock}}{32 \times \text{SCI Baud Rate Desired}}$$

where SCBR[12:0] is in the range {1, 2, 3, …, 8191}

Writing a value of zero to SCBR[12:0] disables the baud rate generator.

Table 61 lists the SCBR[12:0] settings for standard and maximum baud rates using a 16.78 MHz system clock.

Table 61 SCI Baud Rates

Nominal Baud Rate	Actual Rate with 16.78 MHz Clock	SCBR[12:0] Value
64	64.0	$1B8B
110	110.0	$1006
300	300.0	$05E0
600	600.1	$02F0
1200	1200.1	$0178
2400	2400.3	$00BC
4800	4800.5	$005E
9600	9601.1	$002F
19200	18802.1	$0018
38400	37604.2	$000C
76800	75208.3	$0006
Maximum Rate	451250.0	$0001

SCCR1 — SCI Control Register 1   **$YFFC0A**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	LOOPS	WOMS	ILT	PT	PE	M	WAKE	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK

RESET:

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SCCR1 contains SCI configuration parameters. The CPU can read and write this register at any time. The SCI can modify RWU in some circumstances. In general, interrupts enabled by these control bits are cleared by reading SCSR, then reading (for receiver status bits) or writing (for transmitter status bits) SCDR.

Bit 15 — Not Implemented

LOOPS — Loop Mode
      0 = Normal SCI operation, no looping, feedback path disabled
      1 = Test SCI operation, looping, feedback path enabled
LOOPS controls a feedback path on the data serial shifter. When loop mode is enabled, SCI transmitter output is fed back into the receive serial shifter. TXD is asserted (idle line). Both transmitter and receiver must be enabled before entering loop mode.

WOMS — Wired-OR Mode for SCI Pins
      0 = If configured as an output, TXD is a normal CMOS output.
      1 = If configured as an output, TXD is an open-drain output.
WOMS determines whether the TXD pin is an open-drain output or a normal CMOS output. This bit is used only when TXD is an output. If TXD is used as a general-purpose input pin, WOMS has no effect.

ILT — Idle-Line Detect Type
      0 = Short idle-line detect (start count on first one)
      1 = Long idle-line detect (start count on first one after stop bit(s))

PT — Parity Type
      0 = Even parity
      1 = Odd parity
When parity is enabled, PT determines whether parity is even or odd for both the receiver and the transmitter.

PE — Parity Enable
      0 = SCI parity disabled
      1 = SCI parity enabled
PE determines whether parity is enabled or disabled for both the receiver and the transmitter. If the received parity bit is not correct, the SCI sets the PF error flag in SCSR.

When PE is set, the most significant bit (MSB) of the data field is used for the parity function, which results in either seven or eight bits of user data, depending on the condition of M bit. **Table 62** lists the available choices.

Table 62 Parity Enable Data Bit Selection

M	PE	Result
0	0	8 Data Bits
0	1	7 Data Bits, 1 Parity Bit
1	0	9 Data Bits
1	1	8 Data Bits, 1 Parity Bit

M — Mode Select
   0 = SCI frame: 1 start bit, 8 data bits, 1 stop bit (10 bits total)
   1 = SCI frame: 1 start bit, 9 data bits, 1 stop bit (11 bits total)

WAKE — Wakeup by Address Mark
   0 = SCI receiver awakened by idle-line detection
   1 = SCI receiver awakened by address mark (last bit set)

TIE — Transmit Interrupt Enable
   0 = SCI TDRE interrupts inhibited
   1 = SCI TDRE interrupts enabled

TCIE — Transmit Complete Interrupt Enable
   0 = SCI TC interrupts inhibited
   1 = SCI TC interrupts enabled

RIE — Receiver Interrupt Enable
   0 = SCI RDRF interrupt inhibited
   1 = SCI RDRF interrupt enabled

ILIE — Idle-Line Interrupt Enable
   0 = SCI IDLE interrupts inhibited
   1 = SCI IDLE interrupts enabled

TE — Transmitter Enable
   0 = SCI transmitter disabled (TXD pin may be used as I/O)
   1 = SCI transmitter enabled (TXD pin dedicated to SCI transmitter)
The transmitter retains control of the TXD pin until completion of any character transfer that was in progress when TE is cleared.

RE — Receiver Enable
   0 = SCI receiver disabled (status bits inhibited)
   1 = SCI receiver enabled

RWU — Receiver Wakeup
   0 = Normal receiver operation (received data recognized)
   1 = Wakeup mode enabled (received data ignored until awakened)
Setting RWU enables the wakeup function, which allows the SCI to ignore received data until awakened by either an idle line or address mark (as determined by WAKE). When in wakeup mode, the receiver status flags are not set, and interrupts are inhibited. This bit is cleared automatically (returned to normal mode) when the receiver is awakened.

SBK — Send Break
   0 = Normal operation
   1 = Break frame(s) transmitted after completion of current frame
SBK provides the ability to transmit a break code from the SCI. If the SCI is transmitting when SBK is set, it will transmit continuous frames of zeros after it completes the current frame, until SBK is cleared. If SBK is toggled (one to zero in less than one frame interval), the transmitter sends only one or two break frames before reverting to idle line or beginning to send data.

SCSR — SCI Status Register   **$YFFC0C**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NOT USED							TDRE	TC	RDRF	RAF	IDLE	OR	NF	FE	PF

RESET:

| | | | | | | | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SCSR contains flags that show SCI operational conditions. These flags can be cleared either by hardware or by a special acknowledgment sequence. The sequence consists of SCSR read with flags set, followed by SCDR read (write in the case of TDRE and TC). A long-word read can consecutively access both SCSR and SCDR. This action clears receive status flag bits that were set at the time of the read, but does not clear TDRE or TC flags.

If an internal SCI signal for setting a status bit comes after the CPU has read the asserted status bits, but before the CPU has written or read register SCDR, the newly set status bit is not cleared. SCSR must be read again with the bit set. Also, SCDR must be written or read before the status bit is cleared.

Reading either byte of SCSR causes all 16 bits to be accessed. Any status bit already set in either byte will be cleared on a subsequent read or write of register SCDR.

TDRE — Transmit Data Register Empty Flag
    0 = Register TDR still contains data to be sent to the transmit serial shifter.
    1 = A new character can now be written to the transmit data register.
TDRE is set when the byte in the transmit data register is transferred to the transmit serial shifter. If TDRE is zero, transfer has not occurred and a write to the transmit data register will overwrite the previous value. New data is not transmitted if the transmit data register is written without first clearing TDRE.

TC — Transmit Complete Flag
    0 = SCI transmitter is busy
    1 = SCI transmitter is idle
TC is set when the transmitter finishes shifting out all data, queued preambles (mark/idle line), or queued breaks (logic zero). The interrupt can be cleared by reading SCSR when TC is set and then by writing the transmit data register of SCDR.

RDRF — Receive Data Register Full Flag
    0 = Receive data register is empty or contains previously read data.
    1 = Receive data register contains new data.
RDRF is set when the content of the receive serial shifter is transferred to the receive data register. If one or more errors are detected in the received word, flag(s) NF, FE, and/or PF are set within the same clock cycle.

RAF — Receiver Active Flag
    0 = SCI receiver is idle
    1 = SCI receiver is busy
RAF indicates whether the SCI receiver is busy. It is set when the receiver detects a possible start bit and is cleared when the chosen type of idle line is detected. RAF can be used to reduce collisions in systems with multiple masters.

IDLE — Idle-Line Detected Flag
    0 = SCI receiver did not detect an idle-line condition.
    1 = SCI receiver detected an idle-line condition.
IDLE is disabled when RWU in SCCR1 is set. IDLE is set when the SCI receiver detects the idle-line condition specified by ILT in SCCR1. If cleared, IDLE will not set again until after RDRF is set. RDRF is set when a break is received, so that a subsequent idle line can be detected.

OR — Overrun Error Flag

    0 = RDRF is cleared before new data arrives.

    1 = RDRF is not cleared before new data arrives.

OR is set when a new byte is ready to be transferred from the receive serial shifter to the receive data register, and RDRF is still set. Data transfer is inhibited until OR is cleared. Previous data in receive data register remains valid, but data received during overrun condition (including the byte that set OR) is lost.

NF — Noise Error Flag

    0 = No noise detected on the received data

    1 = Noise occurred on the received data

NF is set when the SCI receiver detects noise on a valid start bit, on any data bit, or on a stop bit. It is not set by noise on the idle line or on invalid start bits. Each bit is sampled three times. If none of the three samples are the same logic level, the majority value is used for the received data value, and NF is set. NF is not set until an entire frame is received and RDRF is set.

FE — Framing Error Flag

    0 = No framing error on the received data.

    1 = Framing error or break occurred on the received data.

FE is set when the SCI receiver detects a zero where a stop bit was to have occurred. FE is not set until the entire frame is received and RDRF is set. A break can also cause FE to be set. It is possible to miss a framing error if RXD happens to be at logic level one at the time the stop bit is expected.

PF — Parity Error Flag

    0 = No parity error on the received data

    1 = Parity error occurred on the received data

PF is set when the SCI receiver detects a parity error. PF is not set until the entire frame is received and RDRF is set.

SCDR — SCI Data Register   **$YFFC0E**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	R8/T8	R7/T7	R6/T6	R5/T5	R4/T4	R3/T3	R2/T2	R1/T1	R0/T0

RESET:

0	0	0	0	0	0	0	U	U	U	U	U	U	U	U	U

SCDR contains two data registers at the same address. The receive data register is a read-only register that contains data received by the SCI. The data comes into the receive serial shifter and is transferred to the receive data register. The transmit data register is a write-only register that contains data to be transmitted. The data is first written to the transmit data register, then transferred to the transmit serial shifter, where additional format bits are added before transmission. R[7:0]/T[7:0] contain either the first eight data bits received when SCDR is read, or the first eight data bits to be transmitted when SCDR is written. R8/T8 are used when the SCI is configured for 9-bit operation. When it is configured for 8-bit operation, they have no meaning or effect.

8 Standby RAM Module

The standby RAM module (SRAM) provides 2 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.

8.1 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.

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 63 shows the SRAM address map.

Table 63 SRAM Address Map

Address	15 8	7 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)	

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

8.2 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).

8.3 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 64**.

Table 64 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.

8.4 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.

9 Flash EEPROM Module

The MC68HC916X1 contains two flash electrically erasable programmable read-only memory (EE-PROM) modules: a 16 Kbyte module and a 32 Kbyte module.

9.1 Overview

The flash EEPROM modules serve as nonvolatile, fast-access, electrically erasable and programmable ROM-emulation memory. The modules can contain program code (e.g., operating system kernels and standard subroutines) which must execute at high speed or is frequently executed, or static data which is read frequently. The flash EEPROM supports both byte and word reads. It is capable of responding to back-to-back IMB accesses to provide two bus cycle (four system clock) access for aligned long words. It can also be programmed to insert up to three wait states to accommodate migration from slower external development memory to onboard flash EEPROM without the need for retiming the system.

The 16 Kbyte flash EEPROM array can begin on any 16 Kbyte boundary, and the 32 Kbyte array can begin on any 32 Kbyte boundary. The two arrays can be configured to appear as a single contiguous memory block, with the 16 Kbyte array immediately preceding or immediately following the 32 Kbyte array.

Pulling data bus pin DATA14 low during reset disables both the 16- and 32-Kbyte flash EEPROM modules and places them in stop mode.

Either of the flash EEPROM modules can be configured to generate bootstrap information on system reset. Bootstrap information consists of the initial program counter and stack pointer values for the CPU16.

The flash EEPROM and its control bits are erasable and programmable under software control. Program/erase voltage must be supplied via external V_{FPE} pins. Data is programmed in byte or word aligned fashion. Multiple word programming is not supported. The flash EEPROM modules support bulk erase only, and have a minimum program-erase life of 100 cycles.

The flash EEPROM modules have hardware interlocks which protect stored data from corruption by accidental enabling of the program/erase voltage to the flash EEPROM arrays. With the hardware interlocks, inadvertent programming or erasure is highly unlikely.

9.2 Address Map

Table 65 shows the flash EEPROM module address map.

Table 65 Flash EEPROM Address Map

Address	Register	Module
$YFF800[1]	FLASH EEPROM MODULE CONFIGURATION (FEE1MCR)	16 KBYTE FLASH EEPROM
$YFF802	FLASH EEPROM TEST REGISTER (FEE1TST)	
$YFF804	FLASH EEPROM BASE ADDRESS HIGH (FEE1BAH)	
$YFF806	FLASH EEPROM BASE ADDRESS LOW (FEE1BAL)	
$YFF808	FLASH EEPROM CONTROL REGISTER (FEE1CTL)	
$YFF80A	RESERVED	
$YFF80C	RESERVED	
$YFF80E	RESERVED	
$YFF810	FLASH EEPROM BOOTSTRAP WORD 0 (FEE1BS0)	
$YFF812	FLASH EEPROM BOOTSTRAP WORD 1 (FEE1BS1)	
$YFF814	FLASH EEPROM BOOTSTRAP WORD 2 (FEE1BS2)	
$YFF816	FLASH EEPROM BOOTSTRAP WORD 3 (FEE1BS3)	
$YFF818	RESERVED	
$YFF81A	RESERVED	
$YFF81C	RESERVED	
$YFF81E	RESERVED	
$YFF820	FLASH EEPROM MODULE CONFIGURATION (FEE2MCR)	32 KBYTE FLASH EEPROM
$YFF822	FLASH EEPROM TEST REGISTER (FEE2TST)	
$YFF824	FLASH EEPROM BASE ADDRESS HIGH (FEE2BAH)	
$YFF826	FLASH EEPROM BASE ADDRESS LOW (FEE2BAL)	
$YFF828	FLASH EEPROM CONTROL REGISTER (FEE2CTL)	
$YFF82A	RESERVED	
$YFF82C	RESERVED	
$YFF82E	RESERVED	
$YFF830	FLASH EEPROM BOOTSTRAP WORD 0 (FEE2BS0)	
$YFF832	FLASH EEPROM BOOTSTRAP WORD 1 (FEE2BS1)	
$YFF834	FLASH EEPROM BOOTSTRAP WORD 2 (FEE2BS2)	
$YFF836	FLASH EEPROM BOOTSTRAP WORD 3 (FEE2BS3)	
$YFF838	RESERVED	
$YFF83A	RESERVED	
$YFF83C	RESERVED	
$YFF83E	RESERVED	

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

9.3 Flash EEPROM Control Block

Each flash EEPROM module has a 32 byte control block with five registers to control flash EE-PROM operation: the flash EEPROM module configuration register (FEE1MCR, FEE2MCR), the flash EEPROM test register (FEE1TST, FEE2TST), the flash EEPROM array base address registers (FEE1BAH, FEE2BAH, and FEE1BAL, FEE2BAL), and the flash EEPROM control register (FEE1CTL, FEE2CTL). Four additional flash EEPROM words in the control block can contain bootstrap information for use during reset. Control registers are located in supervisor data space.

The control register blocks for the 16 Kbyte and 32 Kbyte flash EEPROM modules start at locations $YFF800 and $YFF820, respectively. The following register descriptions apply to the corresponding register in either control block. References to FEExMCR, for example, apply to both FEE1MCR (in the 16 Kbyte module) and FEE2MCR (in the 32 Kbyte module.)

A number of control register bits have associated bits in "shadow" registers. The values of the shadow bits determine the reset states of the control register bits. Shadow registers are programmed or erased in the same manner as a location in the array, using the address of the corresponding control registers. When a shadow register is programmed, the data is not written to the corresponding control register. The new data is not copied into the control register until the next reset. The contents of shadow registers are erased when the array is erased.

Configuration information is specified and programmed independently of the array. After reset, registers in the control block that contain writable bits can be modified. Writes to these registers do not affect the associated shadow register. Certain registers can be written only when LOCK = 0 or STOP = 1 in FEExMCR.

9.4 Flash EEPROM Array

The base address registers specify the starting address of the flash EEPROM array. The user programs the reset base address. The base address of the 16 Kbyte array must be on a 16 Kbyte boundary; the base address of the 32 Kbyte array must be on a 32 Kbyte boundary. Behavior will be indeterminate if one flash EEPROM array overlaps the other.

The base address must also be set so that an array does not overlap a flash EEPROM control block in the data space memory map. If an array does overlap a control block, accesses to the 32 bytes in the array that is overlapped are ignored, allowing the flash EEPROM control blocks to remain accessible. If the array overlaps the control block of another module, the results will be indeterminate.

9.5 Flash EEPROM Registers

In the following register diagrams, bits with reset states determined by shadow bits are shaded, and the reset state is annotated "SB".

FEE1MCR, FEE2MCR — Flash EEPROM Module Configuration Registers   **$YFF800, $YFF820**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STOP	FRZ	0	BOOT	LOCK	0	ASPC[1:0]		WAIT[1:0]		0	0	0	0	0	0

RESET:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
$\overline{\text{DATA14}}$ + SB	0	0	SB	SB	0	SB	SB	SB	SB	0	0	0	0	0	0

The flash EEPROM module configuration registers (FEE1MCR, FEE2MCR) control module configuration. This register can be written only when LOCK = 0. All active bits in the FEExMCR take values from the associated shadow register during reset.

STOP — Stop Mode Control
        0 = Normal operation
        1 = Low-power stop operation
STOP can be set either by pulling data bus pin DATA14 low during reset (for both flash EEPROM modules) or by the corresponding shadow bit. The array can be re-enabled by clearing STOP. If STOP is set during programming or erasing, the program/erase voltage is automatically turned off. However, the ENPE control bit in FEExCTL remains set. When STOP is cleared, the program/erase voltage is automatically turned back on if ENPE is set.

FRZ — Freeze Mode Control
    0 = Disable program/erase voltage while FREEZE is asserted
    1 = Allow ENPE bit to turn on the program/erase voltage while FREEZE is asserted

BOOT — Boot Control
    0 = Flash EEPROM module responds to bootstrap addresses after reset
    1 = Flash EEPROM module does not respond to bootstrap addresses after reset
    On reset, BOOT takes on the value stored in its associated shadow bit. If BOOT = 0 and STOP = 0, the module responds to program space accesses to IMB addresses $000000 to $000006 following reset, and the contents of FEExBS[3:0] are used as bootstrap vectors. After address $000006 is read, the module responds normally to control block or array addresses only.

LOCK — Lock Registers
    0 = Write-locking disabled
    1 = Write-locked registers protected
    If the reset state of LOCK is zero, it can be set once after reset to allow protection of the registers after initialization. Once the LOCK bit is set by software, it cannot be cleared again until after a reset.

ASPC[1:0] — Flash EEPROM Array Space
    ASPC[1:0] assigns the array to supervisor or user space, and to program or data space. The state of ASPC[1:0] out of reset is determined by the value stored in the associated shadow bits. Since the CPU16 runs only in supervisor mode, ASPC1 must remain set to one for array accesses to take place. The field can be written only when LOCK = 0 and STOP = 1. Refer to **Table 66**.

Table 66 Array Space Encoding

ASPC[1:0]	Type of Access
10	Supervisor program and data space
11	Supervisor program space

WAIT[1:0] — Wait States
    The state of WAIT[1:0] out of reset is determined by the value stored in the associated shadow bits. WAIT[1:0] specifies the number of wait states inserted during accesses to the flash EEPROM module. A wait state has the duration of one system clock cycle. WAIT[1:0] affects both control block and array accesses, and can be written only if LOCK = 0 and STOP = 1. Refer to **Table 67**.

Table 67 Wait State Encoding

WAIT[1:0]	Wait States	Clocks Per Transfer
00	0	3
01	1	4
10	2	5
11	−1	2

The value of WAIT[1:0] is compatible with the lower two bits of the $\overline{\text{DSACK}}$ field in the SCIM chip-select option registers. An encoding of %11 in WAIT[1:0] corresponds to an encoding for fast termination.

FEE1TST, FEE2TST — Flash EEPROM Test Registers   **$YFF802, $YFF822**

    These registers are used for factory test only.

FEE1BAH, FEE2BAH — Flash EEPROM Base Address High Registers   **$YFF804, $YFF824**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	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	SB	SB	SB	SB	SB	SB	SB	SB

FEE1BAL — Flash EEPROM Base Address Low Register   **$YFF806**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR 15	ADDR 14	0	0	0	0	0	0	0	0	0	0	0	0	0	0

RESET:

SB	SB	0	0	0	0	0	0	0	0	0	0	0	0	0	0

FEE2BAL — Flash EEPROM Base Address Low Register   **$YFF826**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

RESET:

SB	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

The base address high registers (FEE1BAH, FEE2BAH) contain the 8 high-order bits of the array base address; the base address low registers (FEE1BAL, FEE2BAL) contain the active low-order bits of the array base address. During reset, both FEExBAH and FEExBAL take on default values programmed into associated shadow registers. After reset, if LOCK = 0 and STOP = 1, software can write to FEExBAH and FEExBAL to relocate the array.

FEE1CTL, FEE2CTL — Flash EEPROM Control Register   **$YFF808, $YFF828**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	VFPE	ERAS	LAT	ENPE

RESET:

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

The flash EEPROM control registers (FEE1CTL, FEE2CTL) control programming and erasure of the arrays. FEExCTL is accessible in supervisor mode only.

VFPE — Verify Program/Erase
        0 = Normal read cycles
        1 = Invoke program verify circuit
The VFPE bit invokes a special program-verify circuit. During programming sequences (ERAS = 0), VFPE is used in conjunction with the LAT bit to determine when programming of a location is complete. If VFPE and LAT are both set, a bit-wise exclusive-OR of the latched data with the data in the location being programmed occurs when any valid FLASH location is read. If the location is completely programmed, a value of zero is read. Any other value indicates that the location is not fully programmed. When VFPE is cleared, normal reads of valid FLASH locations occur. The value of VFPE cannot be changed while ENPE = 1.

ERAS — Erase Control
        0 = Flash EEPROM configured for programming
        1 = Flash EEPROM configured for erasure
The ERAS bit configures the array for either programming or erasure. Setting ERAS causes all locations in the array and all control bits in the control block to be configured for erasure at the same time.
When the LAT bit is set, ERAS also determines whether a read returns the data in the addressed location (ERAS = 1) or the address itself (ERAS = 0). ERAS cannot be changed while ENPE = 1.

LAT — Latch Control
      0 = Programming latches disabled
      1 = Programming latches enabled
The LAT bit configures the EEPROM array for normal reads or for programming. When LAT is cleared, the FLASH module address and data buses are connected to the IMB address and data buses and the module is configured for normal reads. When LAT is set, module address and data buses are connected to parallel internal latches and the array is configured for programming or erasing.

Once LAT is set, the next write to a valid FLASH module address causes the programming circuitry to latch both address and data. Unless control register shadow bits are to be programmed, the write must be to an array address. The value of LAT cannot be changed while ENPE = 1.

ENPE — Enable Programming/Erase
      0 = Disable program/erase voltage
      1 = Apply program/erase voltage to flash EEPROM
Setting the ENPE bit applies the program/erase voltage to the array. ENPE can be set only after LAT has been set and a write to the data and address latches has occurred. ENPE remains cleared if these conditions are not met. While ENPE is set, the LAT, VFPE, and ERAS bits cannot be changed, and attempts to read an array location are ignored.

FEE1BS[3:0] — Flash EEPROM Bootstrap Words   **$YFF810–$YFF816**
FEE2BS[3:0] — Flash EEPROM Bootstrap Words   **$YFF830–$YFF836**

The flash EEPROM bootstrap words (FEE1BS[3:0], FEE2BS[3:0]) can be used as system bootstrap vectors. When BOOT = 1 in FEExMCR during reset, the flash module responds to program space accesses of IMB addresses $000000 to $000006 after reset. When BOOT = 0, the flash module responds only to normal array and register accesses. FEExBS[3:0] can be read at any time, but it can only be changed by programming the appropriate locations. **Table 68** shows bootstrap word addresses in program space.

Table 68 Bootstrap Words

Bootstrap Word	Corresponding Boot Address	Corresponding Vector Content
FEE1BS0, FEE2BS0	$000000	Initial ZK, SK, and PC
FEE1BS1, FEE2BS1	$000002	Initial PC
FEE1BS2, FEE2BS2	$000004	Initial SP
FEE1BS3, FEE2BS3	$000006	Initial IZ

9.6 Flash EEPROM Operation

The following paragraphs describe the operation of the flash EEPROM module during reset, system boot, normal operation, and while it is being programmed or erased.

9.6.1 Reset Operation

Reset initializes all registers to certain default values. Some of these reset values are programmable by the user and are contained in flash EEPROM shadow registers. If the state of the STOP shadow bit is zero, and bus pin DATA14 is pulled high during reset, the STOP bit in the FEExMCR is cleared during reset. The array responds normally to the bootstrap address range and the flash EEPROM array base address. If the STOP shadow bit is one, or the module's associated data bus pin is pulled low during reset, the STOP bit in the FEExMCR is set. The flash EEPROM array is disabled until the STOP bit is cleared by software. It will not respond to the bootstrap address range, or the flash EEPROM array base address in FEExBAH and FEExBAL, allowing an external device to respond to the flash EEPROM array's address space or bootstrap information. Since the erased state of the shadow bits is one, erased flash EEPROM modules (which include the shadow registers in the control blocks) come out of reset in STOP mode.

9.6.2 Bootstrap Operation

After reset, the CPU begins bootstrap operation by fetching initial values for its internal registers from special bootstrap word addresses $000000 through $000006. If BOOT = 0 and STOP = 0 in FEExMCR, the flash EEPROM module is configured to recognize these addresses after a reset and provide this information from the FEExBS[3:0] bootstrap registers in the flash EEPROM control block. The information in these registers is programmed by the user.

9.6.3 Normal Operation

The flash EEPROM module allows a byte or aligned-word read in one bus cycle. Long-word reads require two bus cycles.

The module checks function codes to verify access privileges. All control block addresses must be in supervisor data space. Array accesses are defined by the state of ASPC[1:0] in FEExMCR. Access time is governed by the WAIT[1:0] field in FEExMCR.

Accesses to any address in the address block defined by FEExBAH and FEExBAL which does not fall within the array are ignored, allowing external devices to adjoin flash EEPROM arrays which do not entirely fill the entire address space specified by FEExBAH and FEExBAL.

9.6.4 Program/Erase Operation

An erased flash bit has a logic state of one. A bit must be programmed to change its state from one to zero. Erasing a bit returns it to a logic state of one. Programming and erasing the flash module requires a series of control register writes and a write to an array address. The same procedure is used to program control registers that contain flash shadow bits. Programming is restricted to a single byte or aligned word at a time. The entire array and the shadow register bits are erased at the same time.

When multiple flash modules share a single V_{FPE} pin, do not program or erase more than one flash module at a time. Normal accesses to modules that are not being programmed are not affected by programming or erasure of another flash module.

The following paragraphs give step-by-step procedures for programming and erasure of flash EE-PROM arrays. Refer to **11 Electrical Characteristics** for information on programming and erasing specifications for the flash EEPROM module.

9.6.4.1 Programming

The following steps are used to program a flash EEPROM array. **Figure 22** is a flowchart of the programming operation. Refer to **Figures 45** and **46** in **11 Electrical Characteristics** for V_{FPE} to V_{DD} relationships during programming.

1. Increase voltage applied to the V_{FPE} pin to program/erase/verify level.
2. Clear the ERAS bit and set the LAT bit in FEExCTL. This enables the programming address and data latches.
3. Write data to the address to be programmed. This latches the address to be programmed and the programming data.
4. Set the ENPE bit in FEExCTL. This starts the program pulse.
5. Delay the proper amount of time for one programming pulse to take place. Delay is specified by parameter pw_{pp}.
6. Clear the ENPE bit in FEExCTL. This stops the program pulse.
7. Delay while high voltage to array is turned off. Delay is specified by parameter t_{pr}.
8. Read the address to verify that it has been programmed.
9. If the location is not programmed, repeat steps 4 through 7 until the location is programmed, or until the specified maximum number of program pulses has been reached. Maximum number of pulses is specified by parameter n_{pp}.

10. If the location is programmed, repeat the same number of pulses as required to program the location. This provides 100% program margin.
11. Read the address to verify that it remains programmed.
12. Clear the LAT bit in FEExCTL. This disables the programming address and data latches.
13. If more locations are to be programmed, repeat steps 2 through 10.
14. Reduce voltage applied to the V_{FPE} pin to normal read level.

9.6.4.2 Erasure

The following steps are used to erase a flash EEPROM array. **Figure 23** is a flowchart of the erasure operation. Refer to **Figures 45** and **46** in **11 Electrical Characteristics** for V_{FPE} to V_{DD} relationships during erasure.

1. Increase voltage applied to the V_{FPE} pin to program/erase/verify level.
2. Set the ERAS bit and the LAT bit in FEExCTL. This configures the module for erasure.
3. Perform a write to any valid address in the control block or array. The data written does not matter.
4. Set the ENPE bit in FEExCTL. This applies the erase voltage to the array.
5. Delay the proper amount of time for one erase pulse. Delay is specified by parameter t_{epk}.
6. Clear the ENPE bit in FEExCTL. This turns off erase voltage to the array.
7. Delay while high voltage to array is turned off. Delay is specified by parameter t_{er}.
8. Read the entire array and control block to ensure all locations are erased.
9. If all locations are not erased, calculate a new value for t_{epk} ($t_{ei} \times$ pulse number) and repeat steps 3 through 10 until all locations erase, or the maximum number of pulses has been applied.
10. If all locations are erased, calculate the erase margin (e_m) and repeat steps 3 through 10 for the single margin pulse.
11. Clear the LAT and ERAS bits in FEExCTL. This allows normal access to the flash.
12. Reduce voltage applied to the V_{FPE} pin to normal read level.
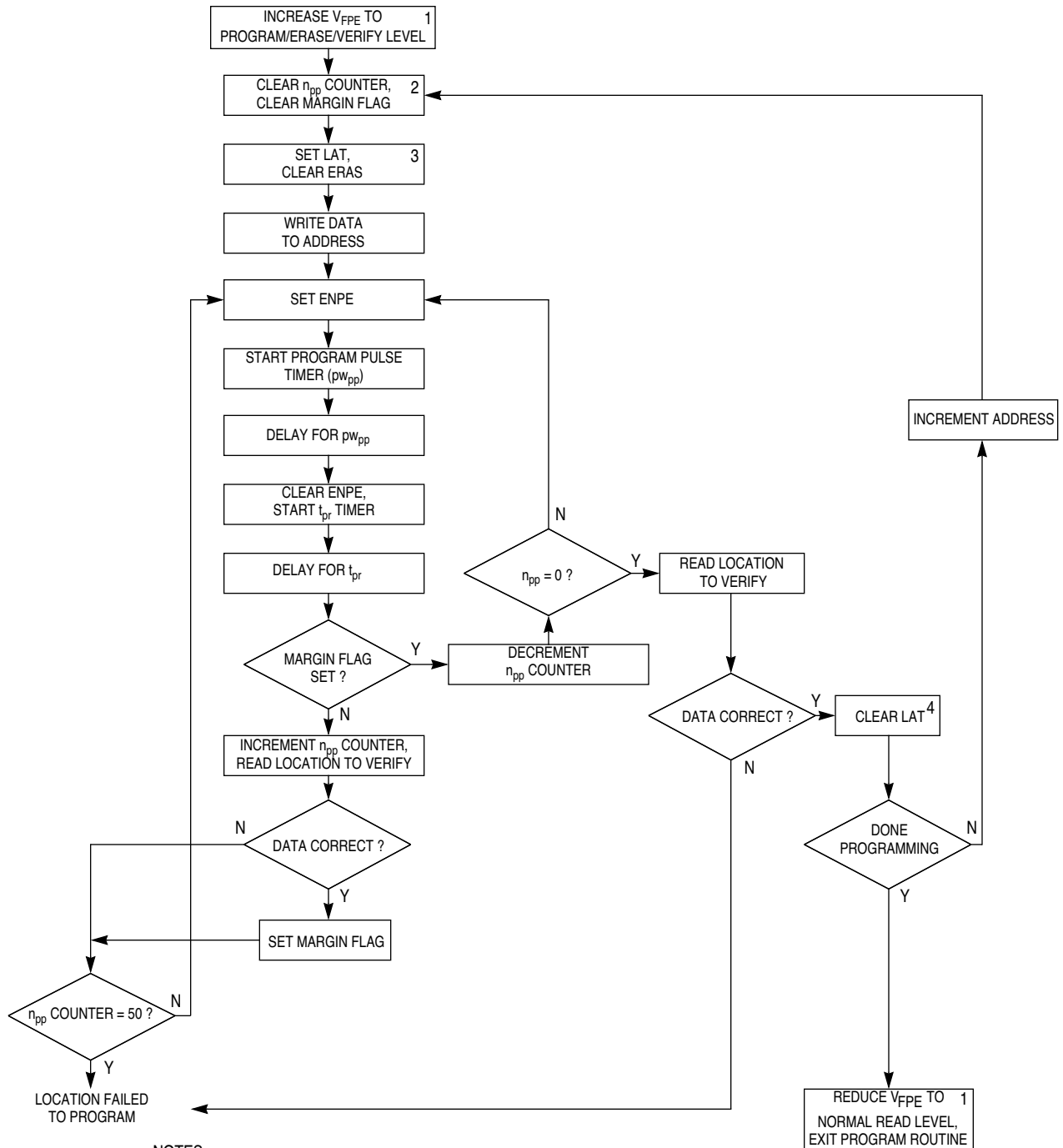
INCREASE V_{FPE} TO   1
PROGRAM/ERASE/VERIFY LEVEL

CLEAR n_{pp} COUNTER,   2
CLEAR MARGIN FLAG

SET LAT,   3
CLEAR ERAS

WRITE DATA
TO ADDRESS

SET ENPE

START PROGRAM PULSE
TIMER (pw_{pp})

DELAY FOR pw_{pp}

CLEAR ENPE,
START t_{pr} TIMER

DELAY FOR t_{pr}

MARGIN FLAG SET ?

DECREMENT n_{pp} COUNTER

n_{pp} = 0 ?

READ LOCATION TO VERIFY

INCREMENT n_{pp} COUNTER, READ LOCATION TO VERIFY

DATA CORRECT ?

DATA CORRECT ?

CLEAR LAT 4

SET MARGIN FLAG

INCREMENT ADDRESS

n_{pp} COUNTER = 50 ?

DONE PROGRAMMING

LOCATION FAILED TO PROGRAM

REDUCE V_{FPE} TO   1
NORMAL READ LEVEL,
EXIT PROGRAM ROUTINE

NOTES:
1. SEE ELECTRICAL CHARACTERISTICS FOR V_{FPE} PIN VOLTAGE SEQUENCING.
2. THE MARGIN FLAG IS A SOFTWARE-DEFINED FLAG THAT INDICATES WHETHER THE PROGRAM SEQUENCE IS GENERATING PROGRAM PULSES OR MARGIN PULSES.
3. TO SIMPLIFY THE PROGRAM OPERATION, THE V_{FPE} BIT IN FEExCTL CAN BE SET.
4. CLEAR V_{FPE} BIT ALSO IF ROUTINE USES THIS FUNCTION.

FEEPROM PGM FLOW1 TD

Figure 22 Programming Flow
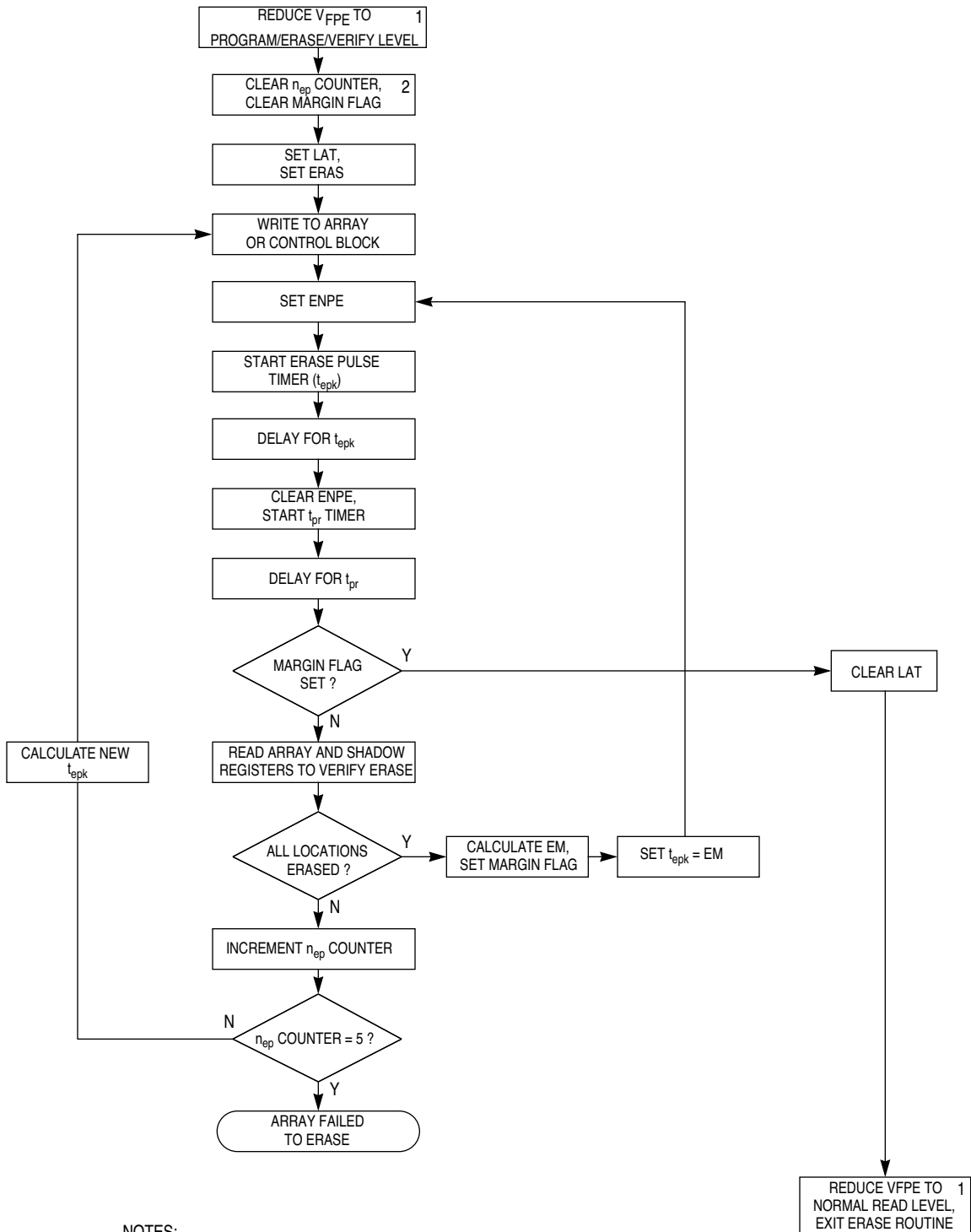
Figure 23 Erasure Flow

NOTES:
1. SEE ELECTRICAL CHARACTERISTICS FOR V_{FPE} PIN VOLTAGE SEQUENCING.
2. THE MARGIN FLAG IS A SOFTWARE-DEFINED FLAG THAT INDICATES WHETHER THE PROGRAM SEQUENCE IS GENERATING ERASE PULSES OR MARGIN PULSES.

FEEPROM PGM FLOW2 TD

10 Block-Erasable Flash EEPROM

The 2 Kbyte block-erasable flash EEPROM module (BEFLASH) serves as nonvolatile, fast-access ROM-emulation memory. The module can be used for program code that must either execute at high speed or is frequently executed, such as operating system kernels and standard subroutines, or it can be used for static data that is read frequently. The module can also be configured to provide bootstrap vectors for system reset.

10.1 Overview

The BEFLASH module consists of a control register block that occupies a fixed position in MCU address space and a 2 Kbyte flash EEPROM array that can be mapped to any 2 Kbyte boundary in MCU address space. The array can be configured to reside in both program and data space, or in program space alone.

The flash EEPROM array can be read as either bytes, words, or long-words. The module responds to back-to-back IMB accesses, providing two bus cycle (four system clocks) access for aligned long words. The module can also be programmed to insert up to three wait states per access, to accommodate migration from slower external development memory without re-timing the system.

Both the array and the individual control bits are programmable and erasable under software control. Program/erase voltage must be supplied via the external V_{FPE2K} pin. Data is programmed in byte or word aligned fashion. The module supports both block and bulk erase modes, and has a minimum program/erase life of 100 cycles. Hardware interlocks protect stored data from corruption if the program/erase voltage to the BEFLASH EEPROM array is enabled accidently. The BEFLASH array is enabled/disabled by a combination of DATA15 and the STOP shadow bit after reset.

Table 69 shows the BEFLASH address map.

Table 69 BEFLASH Address Map

Address	15 8	7 0
$YFF7A0[1]	BEFLASH MODULE CONFIGURATION REGISTER (BFEMCR)	
$YFF7A2	BEFLASH TEST REGISTER (BFETST)	
$YFF7A4	BEFLASH BASE ADDRESS HIGH REGISTER (BFEBAH)	
$YFF7A6	BEFLASH BASE ADDRESS LOW REGISTER (BFEBAL)	
$YFF7A8	BEFLASH CONTROL REGISTER (BFECTL)	
$YFF7AA	RESERVED	
$YFF7AC	RESERVED	
$YFF7AE	RESERVED	
$YFF7B0	BEFLASH BOOTSTRAP WORD 0 (BFEBS0)	
$YFF7B2	BEFLASH BOOTSTRAP WORD 1 (BFEBS1)	
$YFF7B4	BEFLASH BOOTSTRAP WORD 2 (BFEBS2)	
$YFF7B6	BEFLASH BOOTSTRAP WORD 3 (BFEBS3)	
$YFF7B8	RESERVED	
$YFF7BA	RESERVED	
$YFF7BC	RESERVED	
$YFF7BE	RESERVED	

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

10.2 BEFLASH Control Block

The BEFLASH module control block contains five registers: the BEFLASH module configuration register (BFEMCR), the BEFLASH test register (BFETST), the BEFLASH array base address registers (BFEBAH and BFEBAL), and the BEFLASH control register (BFECTL). Four additional words in the control block can contain bootstrap information when the BEFLASH is used as bootstrap memory.

Each register in the control block has an associated shadow register that is physically located in a spare BEFLASH row. During reset, fields within the registers are loaded with default information from the shadow registers. Shadow registers are programmed or erased in the same manner as locations in the BEFLASH array, using the address of the corresponding control registers. When a shadow register is programmed, the data is not written to the corresponding control register. The new data is not copied into the control register until the next reset. The contents of shadow registers are erased whenever the BEFLASH array is erased.

Configuration information is specified and programmed independently of the BEFLASH array. After reset, registers in the control block that contain writable bits can be modified. Writes to these registers do not affect the associated shadow register. Certain registers are writable only when the LOCK bit in BFEMCR is disabled or when the STOP bit in BFEMCR is set. These restrictions are noted in the individual register descriptions.

10.3 BEFLASH Array

The base address registers specify the starting address of the BEFLASH array. A default base address can be programmed into the base address shadow registers. The array base address must be on a 2 Kbyte boundary. Because the states of ADDR[23:20] follow the state of ADDR19, addresses in the range $080000 to $F7FFFF cannot be accessed by the CPU16. If the BEFLASH array is mapped to these addresses, the system must be reset before the array can be accessed.

Avoid using a base address value that causes the array to overlap control registers. If a portion of the array overlaps the EEPROM register block, the registers remain accessible, but accesses to that portion of the array are ignored. If the array overlaps the control block of another module, however, those registers may become inaccessible. If the BEFLASH array overlaps another memory array (RAM or flash EEPROM), proper access to one or both arrays may not be possible.

10.4 BEFLASH Registers

In the following register diagrams, the reset value SB indicates that a bit assumes the value of its associated shadow bit during reset.

BFEMCR — BEFLASH Module Configuration Register   **$YFF7A0**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STOP	FRZ	0	BOOT	LOCK	0	ASPC[1:0]		0	0	0	0	0	0	0	0

RESET:

$\overline{\text{DATA15}}$+ SB	0	0	SB	SB	0	SB	SB	0	0	0	0	0	0	0	0

This register can be written only when the control block is not write-locked (when LOCK = 0). All active bits take values from the associated shadow register during reset.

STOP — Stop Mode Control
    0 = Normal operation
    1 = Low-power stop operation
STOP can be set either by pulling data bus pin DATA15 low during reset or by the corresponding shadow bit. The EEPROM array is inaccessible during low-power stop. The array can be re-enabled by

clearing STOP. If STOP is set during programming or erasing, the program/erase voltage is automatically turned off. However, the enable program/erase bit (ENPE) remains set. If STOP is cleared, program/erase voltage is automatically turned back on unless ENPE is cleared.

FRZ — Freeze Mode Control
    0 = Disable program/erase voltage while FREEZE is asserted
    1 = Allow ENPE bit to turn on the program/erase voltage while FREEZE signal is asserted

BOOT — Boot Control
    0 = BEFLASH responds to bootstrap vector addresses after reset
    1 = BEFLASH does not respond to bootstrap vector addresses after reset
On reset, BOOT takes on the value stored in its associated shadow bit. If BOOT = 0 and STOP = 0, the module responds to program space accesses of IMB addresses $000000 to $000006 following reset, and the contents of BFEBS[3:0] are used as bootstrap vectors. After address $000006 is read, the module responds normally to control block or array addresses only.

LOCK — Lock Registers
    0 = Write-locking disabled
    1 = Write-locked registers protected
If the reset state of the LOCK is zero, it can be set once to protect the registers after initialization. When set, LOCK cannot be cleared until reset occurs.

ASPC[1:0] — BEFLASH Array Space
The CPU16 operates only in supervisory mode, and as a result, ASPC1 must remain set to one for array accesses to take place. The field can be written only if LOCK = 0 and STOP = 1. During reset, ASPC[1:0] takes on the default value programmed into the associated shadow register. Refer to **Table 70**.

Table 70 Array Space Encoding

ASPC[1:0]	Type of Access
10	Supervisor program and data space
11	Supervisor program space

BFETST — BEFLASH Test Register   **$YFF7A2**

      This register is used for factory test purposes only.

BFEBAH — BEFLASH Base Address High Register   **$YFF7A4**

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:

								SB	SB	SB	SB	SB	SB	SB	SB

BFEBAL — BEFLASH Base Address Low Register   **$YFF7A6**

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	0	0	0	0	0	0	0	0	0	0	0

RESET:

SB	SB	SB	SB	SB	0	0	0	0	0	0	0	0	0	0	0

BFEBAH and BFEBAL contain the 13 high-order bits of the BEFLASH array base address. During reset, BFEBAH and BFEBAL take on the default values programmed into the associated shadow registers. After reset, if LOCK = 0 and STOP = 1, software can write to BFEBAH and BFEBAL to relocate the BEFLASH array. Because the states of ADDR[23:20] follow the state of ADDR19, addresses in the range $080000 to $F7FFFF cannot be accessed by the CPU16. If the BEFLASH array is mapped to these addresses, the system must be reset before the array can be accessed.

BFECTL — BEFLASH Control Register   **$YFF7A8**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	VFPE	ERAS	LAT	ENPE

RESET:

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

BFECTL contains the bits needed to control programming and erasing the BEFLASH.

VFPE — Verify Program/Erase
    0 = Normal read cycles
    1 = Invoke program-verify circuit
This bit invokes a special program-verify circuit. During programming sequences (ERAS = 0), VFPE is used in conjunction with the LAT bit to determine when programming of a location is complete. If VFPE and LAT are both set, a bit-wise exclusive-OR of the latched data with the data in the location being programmed occurs when any valid BEFLASH location is read. If the location is completely programmed, a value of zero is read. Any other value indicates that the location is not fully programmed. When VFPE is cleared, normal reads of valid BEFLASH locations occur.

ERAS — Erase Control
    0 = BEFLASH configured for programming
    1 = BEFLASH configured for erasure
The ERAS bit in BFECTL configures the BEFLASH array for programming or erasure. Setting ERAS causes all locations in the array and all BEFLASH shadow bits in the control block to be configured for erasure. **Table 71** shows the address ranges that must be written to during an erase operation in order to erase specific blocks of the BEFLASH array.

Table 71 BEFLASH Erase Operation Address Ranges

Block	Addresses Affected	Address Bits Used to Specify Block for Erasure							
		ADDR[23:11]	ADDR[10:6]	ADDR5	ADDR4	ADDR3	ADDR2	ADDR1	ADDR0
0	$0000 – $007F	BFEBAH/ BFEBAL[1]	X^2	1	0	0	0	0	X^2
1	$0080 – $0100			1	0	0	0	1	
2	$0100 – $017F			1	0	0	1	0	
3	$0180 – $01FF			1	0	0	1	1	
4	$0200 – $02FF			1	0	1	0	0	
5	$0300 – $03FF			1	0	1	0	1	
6	$0400 – $05FF			1	0	1	1	0	
7	$0600 – $07FF			1	0	1	1	1	
Reserved				1	1	X	X	X	
Entire Array[3]	$0600 – $07FF			0	X	X	X	X	

1. The block erasable flash base address high and low registers (BFEBAH and BFEBAL) specify ADDR[23:11] of the block to be erased.
2. These address bits are "don't cares" when specifying the block to be erased.
3. Erasing the entire array also erases the BEFLASH control register shadow bits.

When the LAT bit is set, ERAS also determines whether a read returns the value of the addressed location (ERAS = 1) or the location being programmed (ERAS = 0).

The value of ERAS cannot be changed if the program/erase voltage is turned on (ENPE = 1).

LAT — Latch Control
    0 = Programming latches disabled
    1 = Programming latches enabled
When LAT is cleared, the BEFLASH address and data buses are connected to the IMB address and data buses. The BEFLASH is configured for normal reads. When LAT is set, the BEFLASH address and data buses are connected to parallel internal latches. The BEFLASH array is configured for programming or erasing.

Once LAT is set, the next write to a valid BEFLASH address causes the programming circuitry to latch both address and data. Unless control register shadow bits are to be programmed, the write must be to an array address.

The value of LAT cannot be changed when program/erase voltage is turned on (ENPE = 1).

ENPE — Enable Program/Erase
    0 = Disable program/erase voltage
    1 = Apply program/erase voltage
ENPE can be set only after LAT has been set, and a write to the data and address latches has occurred. ENPE remains cleared if these conditions are not met. While ENPE is set, the LAT, VFPE, and ERAS bits cannot be changed, and attempts to read a BEFLASH array location in BEFLASH are ignored.

BFEBS[3:0] — BEFLASH Bootstrap Words   **$YFF7B0 – $YFF7B6**

15	0
BOOTSTRAP VECTOR	

RESET:

PROGRAMMED VALUE
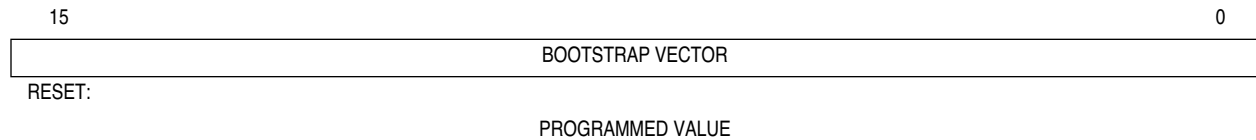
The BEFLASH bootstrap words (BFEBS[3:0]) can be used as system bootstrap vectors. When BOOT = 0 in BFEMCR during reset, the BEFLASH responds to program space accesses of IMB addresses $000000 to $000006 after reset. When BOOT = 1, the BEFLASH responds only to normal array and register accesses. BFEBS[3:0] can be read at any time, but the values in the words can only be changed by programming the appropriate locations.

10.5 BEFLASH Operation

The following paragraphs describe the operation of the BEFLASH during reset, system boot, normal operation, and while it is being programmed or erased.

10.5.1 Reset Operation

Reset initializes all BEFLASH control registers. Some bits have fixed default values, and some take values that are programmed into the associated BEFLASH shadow registers.

If the state of the STOP shadow bit is zero, and data bus pin DATA15 is pulled high during reset, the STOP bit in BFEMCR is cleared during reset, and the module responds to accesses in the range specified by BFEBAH and BFEBAL. When the BOOT bit is cleared, the module also responds to bootstrap vector accesses.

If the state of the STOP shadow bit is one, or data bus pin DATA15 is pulled low during reset, the STOP bit in BFEMCR is set during reset and the BEFLASH array is disabled. The module does not respond to array or bootstrap vector accesses until the STOP bit is cleared. This allows an external device to respond to accesses to the BEFLASH array address space or to bootstrap accesses. The erased state of the shadow bits is one. An erased module comes out of reset in STOP mode.

10.5.2 Bootstrap Operation

After reset, the CPU16 begins bootstrap operation by fetching initial values for its internal registers from IMB addresses $000000 through $000006 in program space. These are the addresses of the bootstrap vectors in the exception vector table. If BOOT = 0 and STOP = 0 in BFEMCR during reset, the BEFLASH module is configured to respond to bootstrap vector accesses. Vector assignments are shown in **Table 72**.

Table 72 Bootstrap Vector Assignments

EEPROM Bootstrap Word	IMB Vector Address	MCU Reset Vector Content
BFEBS0	$000000	Initial ZK, SK, and PK
BFEBS1	$000002	Initial PC
BFEBS2	$000004	Initial SP
BFEBS3	$000006	Initial IZ

As soon as address $000006 has been read, BEFLASH operation returns to normal, and the module no longer responds to bootstrap vector accesses.

10.5.3 Normal Operation

The BEFLASH module allows a byte or aligned-word read in one bus cycle. Long-word reads require two bus cycles.

The module checks function codes to verify address space access type. Array accesses are defined by the state of ASPC[1:0] in BFEMCR.

10.5.4 Program/Erase Operation

An unprogrammed flash bit has a logic state of one. A bit must be programmed to change its state from one to zero. Erasing a bit returns it to a logic state of one. Programming or erasing the BEFLASH array requires a series of control register writes and a write to an array address. The same procedure is used to program control registers that contain flash bits. Programming is restricted to a single byte or aligned word at a time. Erasure of BEFLASH array blocks and control shadow bits are dependent on the setting of ADDR[3:1] of the address written to during an erase operation. Refer to **Table 71** for the address bit patterns corresponding to specific BEFLASH blocks.

NOTE

In order to program the array, programming voltage must be applied to the V_{FPE2K} pin. $V_{FPE2K} \geq (V_{DD} - 0.3 \text{ V})$ must be applied at all times or damage to the BEFLASH module can occur.

Refer to **11 Electrical Characteristics** for information on programming and erasing specifications for the BEFLASH module.

10.5.4.1 Programming Sequence

Use the following procedure to program the BEFLASH. Refer to **Figures 45** and **46** in **11 Electrical Characteristics** for V_{FPE} to V_{DD} relationships during programming.

1. Turn on V_{FPE2K} (apply program/erase voltage to V_{FPE2K} pin).
2. Clear ERAS and set LAT and VFPE bits in BFECTL to set program mode, enable programming address and data latches, and invoke special verification read circuitry. Set initial value of t_{ppulse} to t_{pmin}.
3. Write new data to the desired address. This causes the address and data of the location to be programmed to be latched in the programming latches.
4. Set ENPE to apply programming voltage.
5. Delay long enough for one programming pulse to occur (t_{ppulse}).
6. Clear ENPE to remove programming voltage.
7. Delay while high voltage is turning off (t_{vprog}).
8. Read the location just programmed. If the value read is all zeros, proceed to step 9. If not, calculate a new value for t_{ppulse} and repeat steps 4 through 7 until either the location is verified or the total programming time ($t_{progmax}$) has been exceeded. If $t_{progmax}$ has been exceeded, the location may be bad and should not be used.
9. If the location is programmed, calculate $t_{pmargin}$ and repeat steps 4 through 7. If the location does not remain programmed, the location is bad.
10. Clear VFPE and LAT.
11. If there are more locations to program, repeat steps 2 through 10.
12. Turn off V_{FPE2K} (reduce voltage on V_{FPE2K} pin to V_{DD}).
13. Read the entire array to verify that all locations are correct. If any locations are incorrect, the array is bad.

10.5.4.2 Erasure Sequence

Use the following procedure to erase the BEFLASH. Refer to **Figures 45** and **46** in **11 Electrical Characteristics** for V_{FPE} to V_{DD} relationships during erasure.

1. Turn on V_{FPE2K} (apply program/erase voltage to V_{FPE2K} pin).
2. Set initial value of t_{epulse} to t_{emin}.
3. Set LAT, VFPE, and ERAS bits to configure the BEFLASH module for erasure.
4. Write to any valid address in the control block or array. This allows the erase voltage to be turned on. The data written and the address written to are of no consequence.
5. Set ENPE to apply programming voltage.
6. Delay long enough for one erase pulse to occur (t_{epulse}).
7. Clear ENPE to remove programming voltage.
8. Delay while high voltage is turning off (t_{verase}).
9. Clear LAT, ERAS, and VFPE to allow normal access to the BEFLASH.
10. Read the entire array and control block to ensure that the entire module is erased.
11. If all of the locations are not erased, calculate a new value for t_{epulse} and repeat steps 3 through 10 until either the remaining locations are erased or the maximum erase time ($t_{erasemax}$) has expired.
12. If all locations are erased, calculate $t_{emargin}$ and repeat steps 3 through 10. If all locations do not remain erased, the BEFLASH module may be bad.
13. Turn off V_{FPE2K} (reduce voltage on V_{FPE2K} pin to V_{DD}).

11 Electrical Characteristics

This section contains electrical specification tables and reference timing diagrams.

Table 73 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, 5]	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[3, 5, 5, 6, 7] $V_{SS} - 0.3 \leq V_{IN} \leq V_{DD} + 0.3$	I_{ID}	− 500 to + 500	μA
5	Flash EEPROM Program/Erase Supply Voltage[8, 9]	V_{FPE}	$(V_{DD} - 0.5)$ to +12.6	V
6	Operating Temperature Range C Suffix	T_A	T_L to T_H − 40 to + 85	°C
7	Storage Temperature Range	T_{stg}	− 55 to + 150	°C

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. Input must be current limited to the value specified. To determine the value of the required current-limiting resistor, calculate resistance values for positive and negative clamp voltages, then use the larger of the two values.

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.

8. V_{FPE} must not be raised to programming level while V_{DD} is below specified minimum value. V_{FPE} must not be reduced below minimum specified value while V_{DD} is applied.

9. Flash EEPROM modules can be damaged by power-on and power-off V_{FPE} transients. Maximum power-on overshoot tolerance is 13.5 V for periods of less than 30 ns.

Table 74 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}	116 500 3	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 4.0 125 60	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 40.0	µA µA
8	Power Dissipation	P_D	600	mW

Table 75 Thermal Characteristics

Num	Characteristic	Symbol	Value	Unit
1	Thermal Resistance Plastic 120-Pin Surface Mount	Θ_{JA}	61.2	°C/W

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

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

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:

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

Solving equations 1 and 2 for K gives:

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

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 76 Clock Control Timing

(V_{DD} and V_{DDSYN} = 5.0 Vdc $\pm$ 10 %, 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}	3.2	4.2	MHz
2	System Frequency[2] On-Chip PLL System Frequency Range External Clock Operation	f_{sys}	dc $4(f_{ref})$ /128 dc	16.78 16.78 16.78	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}	– 0.5 – 0.05	0.5 0.05	%

1. Tested with a 4.194 MHz 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 $\overline{RESET}$ 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 77 DC Characteristics

(V_{DD} and V_{DDSYN} = 5.0 Vdc ±10%, 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, 2]	V_{HYS}	0.5	—	V
4	Input Leakage Current[3] $V_{in} = V_{DD}$ or V_{SS} All input-only pins except ADC pins	I_{IN}	–2.5	2.5	µA
5	High Impedance (Off-State)[3] Leakage Current $V_{in} = V_{DD}$ or V_{SS} All input/output and output pins	I_{OZ}	–2.5	2.5	µA
6	CMOS Output High Voltage[3, 4] $I_{OH} = –10.0$ µA Group 1, 2, 4 input/output and all output pins	V_{OH}	V_{DD} –0.2	—	V
7	CMOS Output Low Voltage[3] $I_{OL} = 10.0$ µA Group 1, 2, 4 input/output and all output pins	V_{OL}	—	0.2	V
8	Output High Voltage[3, 4] $I_{OH} = –0.8$ mA Group 1, 2, 4 input/output and all output pins	V_{OH}	V_{DD} –0.8	—	V
9	Output Low Voltage[3] $I_{OL} = 1.6$ mA Group 1 I/O Pins, CLKOUT, FREEZE/QUOT, IPIPE0 $I_{OL} = 5.3$ mA Group 2 and Group 4 I/O Pins, $\overline{BG}$/CSM $I_{OL} = 12$ mA Group 3	V_{OL}	— — —	0.4 0.4 0.4	V
10	Data Bus Mode Select[5] Pull-up Current $V_{in} = V_{IL}$ DATA[15:0] $V_{in} = V_{IH}$ DATA[15:0]	I_{MSP}	— –15	–120 —	µA
11	V_{DD} Supply Current[6] Run[7] LPSTOP, 4.194 MHz crystal, VCO Off (STSIM = 0) LPSTOP, External clock input frequency = maximum f_{sys}	I_{DD}	— — —	150 2 10	mA mA mA
12	Clock Synthesizer Operating Voltage	V_{DDSYN}	4.5	5.5	V
13	V_{DDSYN} Supply Current[6] 4.194 MHz crystal, VCO on, maximum f_{sys} External Clock, maximum f_{sys} LPSTOP, 4.194 MHz crystal, VCO off (STSIM = 0) 4.194 MHz crystal, V_{DD} powered down	I_{DDSYN}	— — — —	2 7 2 2	mA mA mA mA
14	RAM Standby Voltage[8] Specified V_{DD} applied $V_{DD} = V_{SS}$	V_{SB}	0.0 3.0	5.5 5.5	V
15	RAM Standby Current[6] Normal RAM operation[9] $V_{DD} > V_{SB} – 0.5$ V Transient condition $V_{SB} – 0.5$ V $\geq V_{DD} \geq V_{SS} + 0.5$ V Standby operation[8] $V_{DD} < V_{SS} + 0.5$ V	I_{SB}	— — —	50 3 100	µA mA µA
16	Power Dissipation[10]	P_D	—	865	mW
17	Input Capacitance[2, 3] All input-only pins except ADC pins All input/output pins	C_{IN}	— —	10 20	pF

Table 77 DC Characteristics (Continued)

(V_{DD} and V_{DDSYN} = 5.0 Vdc $\pm$10%, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H)

Num	Characteristic	Symbol	Min	Max	Unit

18	Load Capacitance[3]
	    Group 1 I/O Pins, CLKOUT, FREEZE/QUOT, IPIPE0
	    Group 2 I/O Pins and $\overline{BG}$/$\overline{CSM}$
	    Group 3 I/O Pins
	    Group 4 I/O Pins
		C_L	—
			—
			—
			—
				90
				100
				130
				200
					pF

1. Applies to :
   Port ADA[5:0] — AN[5:0]
   Port E[7:4], 1 — SIZ[1:0], $\overline{AS}$, $\overline{DS}$
   Port F[7:6], 0 — $\overline{IRQ}$[7:6], MODCLK
   Port GP[7:0] — IC4/OC5/OC1, IC[3:1], OC[4:1]/OC1
   Port QS[7:0] — TXD, PCS[3:1], PCS0/$\overline{SS}$, SCK, MOSI, MISO
   $\overline{BKPT}$/DSCLK, DSI/IPIPE1, PAI, PCLK, $\overline{RESET}$, RXD, TSC
   EXTAL (when PLL enabled)

2. This parameter is periodically sampled rather than 100% tested.

3. Input-Only Pins: EXTAL, TSC, $\overline{BKPT}$/DSCLK, PAI, PCLK, RXD
   Output-Only Pins: ADDR[2:0], $\overline{BG}$/$\overline{CSM}$, CLKOUT, FREEZE/QUOT, DSO/IPIPE0, PWMA, PWMB
         Group 1: Port GP[7:0] — IC4/OC5/OC1, IC[3:1], OC[4:1]/OC1
                 DATA[15:0], DSI/IPIPE1
         Group 2: Port C[3:0] — ADDR19/$\overline{CS6}$, FC[2:0]/$\overline{CS5}$/$\overline{CS3}$
                 Port E[7:4], 1 — SIZ[1:0], $\overline{AS}$, $\overline{DS}$, $\overline{DSACK1}$
                 Port F[7:6], 0 — $\overline{IRQ}$[7:6], MODCLK
                 Port QS[7:3] — TXD, PCS[3:1], PCS0/$\overline{SS}$
                 ADDR23/$\overline{CS10}$/ECLK, ADDR[18:0], R/$\overline{W}$, $\overline{BERR}$, $\overline{BR}$/$\overline{CS0}$, $\overline{BG}$/$\overline{CSM}$, $\overline{BGACK}$/$\overline{CSE}$

4. Does not apply to RESET because it is an open drain pin. Does not apply to Port QS[7:0] (TXD,
   PCS[3:1], PCS0/$\overline{SS}$, SCK, MOSI, MISO) in wired-OR mode.

5. Use of an active pulldown device is recommended.

6. Total operating current is the sum of the appropriate I_{DD}, I_{DDSYN}, and I_{SB} values, plus I_{DDA}. I_{DD} values
   include supply currents for device modules powered by V_{DDE} and V_{DDI} pins.

7. Current measured at maximum system clock frequency, all modules active.

8. 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.

9. When V_{SB} is more than 0.3 V 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.

10. Power dissipation is measured at maximum system clock frequency, all modules active. Power dissipation can be
    calculated using the following expression:

$$P_D = \text{Maximum } V_{DD} (I_{DD} + I_{DDSYN} + I_{SB}) + \text{Maximum } V_{DDA} (I_{DDA})$$

    I_{DD} includes supply currents for all device modules powered by V_{DDE} and V_{DDI} pins.

Table 78 AC Timing

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

Num	Characteristic	Symbol	Min	Max	Unit
F1	Frequency of Operation[2]	f	4 f(ref)/128	16.78	MHz
1	Clock Period	t_{cyc}	59.6	—	ns
1A	ECLK Period	t_{Ecyc}	476	—	ns
1B	External Clock Input Period[3]	t_{Xcyc}	59.6	—	ns
2, 3	Clock Pulse Width	t_{CW}	28	—	ns
2A, 3A	ECLK Pulse Width	t_{ECW}	236	—	ns
2B, 3B	External Clock Input High/Low Time[3]	t_{XCHL}	29.8	—	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	t_{CHAV}	0	29	ns
7	Clock High to ADDR, Data, FC, SIZE, High Impedance	t_{CHAZx}	0	59	ns
8	Clock High to ADDR, FC, SIZE, Invalid	t_{CHAZn}	0	—	ns
9	Clock Low to $\overline{AS}$, $\overline{DS}$, $\overline{CS}$ Asserted	t_{CLSA}	2	25	ns
9A	$\overline{AS}$ to $\overline{DS}$ or $\overline{CS}$ Asserted (Read)[5]	t_{STSA}	-15	15	ns
11	ADDR, FC, SIZE Valid to $\overline{AS}$, $\overline{CS}$, (and $\overline{DS}$ Read) Asserted	t_{AVSA}	15	—	ns
12	Clock Low to $\overline{AS}$, $\overline{DS}$, $\overline{CS}$ Negated	t_{CLSN}	2	29	ns
13	$\overline{AS}$, $\overline{DS}$, $\overline{CS}$ Negated to ADDR, FC SIZE Invalid (Address Hold)	t_{SNAI}	15	—	ns
14	$\overline{AS}$, $\overline{CS}$ (and $\overline{DS}$ Read) Width Asserted	t_{SWA}	100	—	ns
14A	$\overline{DS}$, $\overline{CS}$ Width Asserted (Write)	t_{SWAW}	45	—	ns
14B	$\overline{AS}$, $\overline{CS}$ (and $\overline{DS}$ Read) Width Asserted (Fast Cycle)	t_{SWDW}	40	—	ns
15	$\overline{AS}$, $\overline{DS}$, $\overline{CS}$ Width Negated[6]	t_{SN}	40	—	ns
16	Clock High to $\overline{AS}$, $\overline{DS}$, R/$\overline{W}$ High Impedance	t_{CHSZ}	—	59	ns
17	$\overline{AS}$, $\overline{DS}$, $\overline{CS}$ Negated to R/$\overline{W}$ High	t_{SNRN}	15	—	ns
18	Clock High to R/$\overline{W}$ High	t_{CHRH}	0	29	ns
20	Clock High to R/$\overline{W}$ Low	t_{CHRL}	0	29	ns
21	R/$\overline{W}$ High to $\overline{AS}$, $\overline{CS}$ Asserted	t_{RAAA}	15	—	ns
22	R/$\overline{W}$ Low to $\overline{DS}$, $\overline{CS}$ Asserted (Write)	t_{RASA}	70	—	ns
23	Clock High to Data Out Valid	t_{CHDO}	—	29	ns
24	Data Out Valid to Negating Edge of $\overline{AS}$, $\overline{CS}$ (Fast Write Cycle)	t_{DVASN}	15	—	ns
25	$\overline{DS}$, $\overline{CS}$ Negated to Data Out Invalid (Data Out Hold)	t_{SNDOI}	15	—	ns
26	Data Out Valid to $\overline{DS}$, $\overline{CS}$ Asserted (Write)	t_{DVSA}	15	—	ns
27	Data In Valid to Clock Low (Data Setup)	t_{DICL}	5	—	ns
27A	Late $\overline{BERR}$ Asserted to Clock Low (Setup Time)	t_{BELCL}	20	—	ns
28	$\overline{AS}$, $\overline{DS}$ Negated to $\overline{DSACK1}$, $\overline{BERR}$, Negated	t_{SNDN}	0	80	ns
29	$\overline{DS}$, $\overline{CS}$ Negated to Data In Invalid (Data In Hold)[7]	t_{SNDI}	0	—	ns
29A	$\overline{DS}$, $\overline{CS}$ Negated to Data In High Impedance[7, 8]	t_{SHDI}	—	55	ns
30	CLKOUT Low to Data In Invalid (Fast Cycle Hold)[7]	t_{CLDI}	15	—	ns
30A	CLKOUT Low to Data In High Impedance[7]	t_{CLDH}	—	90	ns

Table 78 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	$\overline{DSACK1}$ Asserted to Data In Valid[9]	t_{DADI}	—	50	ns
33	Clock Low to $\overline{BG}$ Asserted/Negated	t_{CLBAN}	—	29	ns
35	$\overline{BR}$ Asserted to $\overline{BG}$ Asserted[10]	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}	150	—	ns
46A	R/$\overline{W}$ Width Asserted (Fast Write or Read Cycle)	t_{RWAS}	90	—	ns
47A	Asynchronous Input Setup Time $\overline{BR}$, $\overline{BGACK}$, $\overline{DSACK1}$, $\overline{BERR}$	t_{AIST}	5	—	ns
47B	Asynchronous Input Hold Time	t_{AIHT}	15	—	ns
48	$\overline{DSACK1}$ Asserted to $\overline{BERR}$ Asserted[11]	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}	—	28	ns
55	R/W Asserted to Data Bus Impedance Change	t_{RADC}	40	—	ns
70	Clock Low to Data Bus Driven (Show Cycle)	t_{SCLDD}	0	29	ns
71	Data Setup Time to Clock Low (Show Cycle)	t_{SCLDS}	15	—	ns
72	Data Hold from Clock Low (Show Cycle)	t_{SCLDH}	10	—	ns
73	$\overline{BKPT}$ Input Setup Time	t_{BKST}	15	—	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[12]	t_{RSTA}	4	—	t_{cyc}
78	$\overline{RESET}$ Rise Time[13]	t_{RSTR}	—	10	t_{cyc}
100	CLKOUT High to Phase 1 Asserted[14]	t_{CHP1A}	3	40	ns
101	CLKOUT High to Phase 2 Asserted[14]	t_{CHP2A}	3	40	ns
102	Phase 1 Valid to $\overline{AS}$ or $\overline{DS}$ Asserted[14]	t_{P1VSA}	10	—	ns
103	Phase 2 Valid to $\overline{AS}$ or $\overline{DS}$ Asserted[14]	t_{P2VSN}	10	—	ns
104	$\overline{AS}$ or $\overline{DS}$ Valid to Phase 1 Negated[14]	t_{SAP1N}	10	—	ns
105	$\overline{AS}$ or $\overline{DS}$ Negated to Phase 2 Negated[14]	t_{SNP2N}	10	—	ns

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 MC68HC916X1 requires a 4.194 MHz 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 VCO 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. 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.

6. 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.

7. 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.

8. Maximum value is equal to $(t_{cyc} / 2) + 25$ ns.

9. If the asynchronous setup time (specification 47A) requirements are satisfied, the $\overline{DSACK1}$ low to data setup time (specification 31) and $\overline{DSACK1}$ 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.

10. 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.

11. In the absence of $\overline{DSACK1}$, $\overline{BERR}$ is an asynchronous input using the asynchronous setup time (specification 47A).

12. 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 t_{cyc}.

13. External logic must pull $\overline{RESET}$ high during this period in order for normal MCU operation to begin.

14. Eight pipeline states are multiplexed into IPIPE[1:0]. The multiplexed signals have two phases.

15. 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.
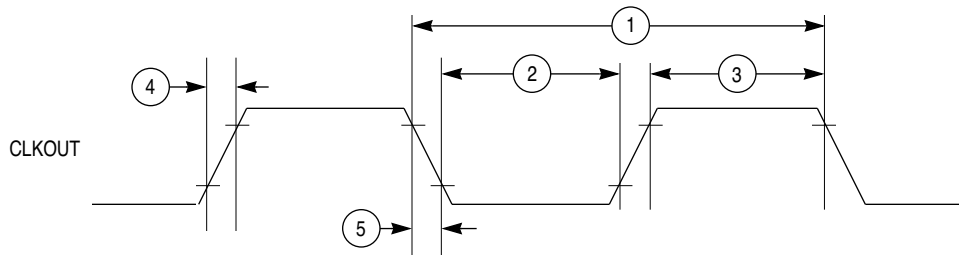
NOTE: TIMING SHOWN WITH RESPECT TO 20% AND 70% V$_{DD}$

16 CLKOUT TIM

Figure 24 CLKOUT Output Timing Diagram
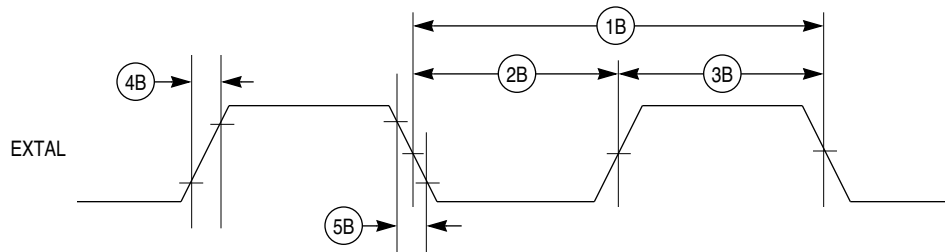
NOTE: TIMING SHOWN WITH RESPECT TO 20% AND 70% V$_{DD}$
PULSE WIDTH SHOWN WITH RESPECT TO 50% V$_{DD}$

16 EXT CLK INPUT TIM

Figure 25 External Clock Input Timing Diagram
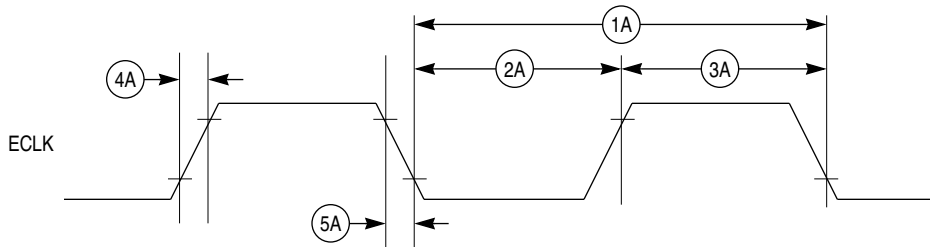
NOTE: TIMING SHOWN WITH RESPECT TO 20% AND 70% V$_{DD}$

16 ECLK OUTPUT TIM

Figure 26 ECLK Output Timing Diagram
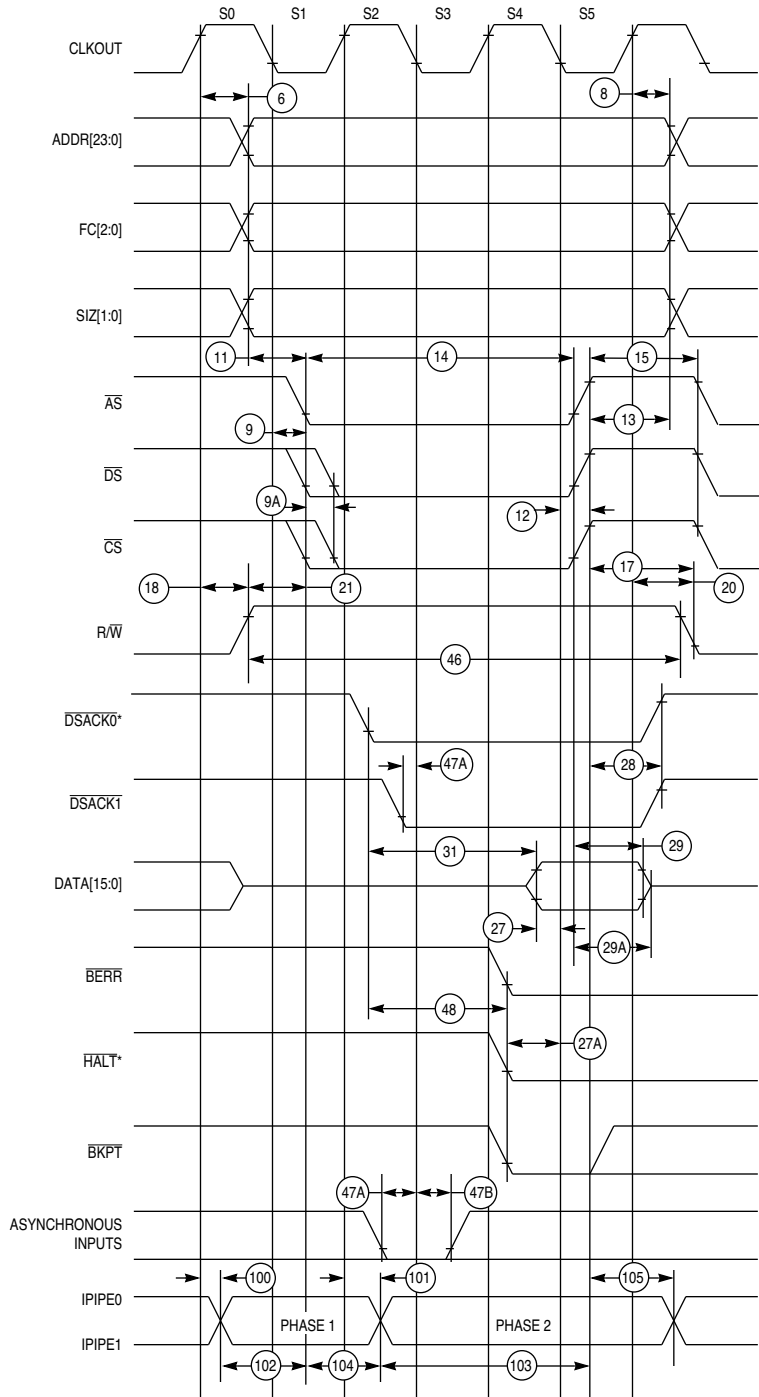
Figure 27 Read Cycle Timing Diagram
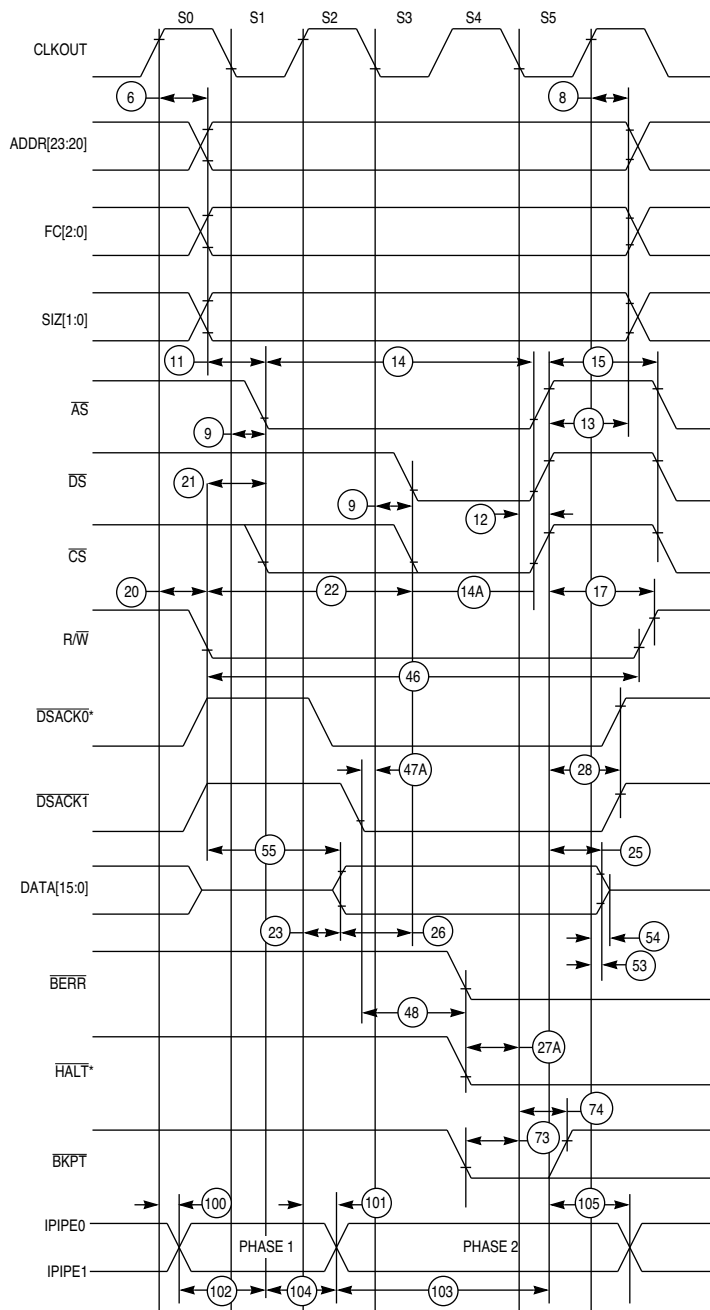
* ON THE MC68HC916X1, THE $\overline{\text{HALT}}$ AND $\overline{\text{DSACK0}}$ PINS ARE NOT BONDED AND ARE INTERNALLY PULLED UP TO V_{DD}.

916X1 RD CYC TIM

* ON THE MC68HC916X1, THE HALT AND DSACK0 PINS ARE NOT BONDED AND ARE INTERNALLY PULLED UP TO V_DD.

916X1 WR CYC TIM

Figure 28 Write Cycle Timing Diagram
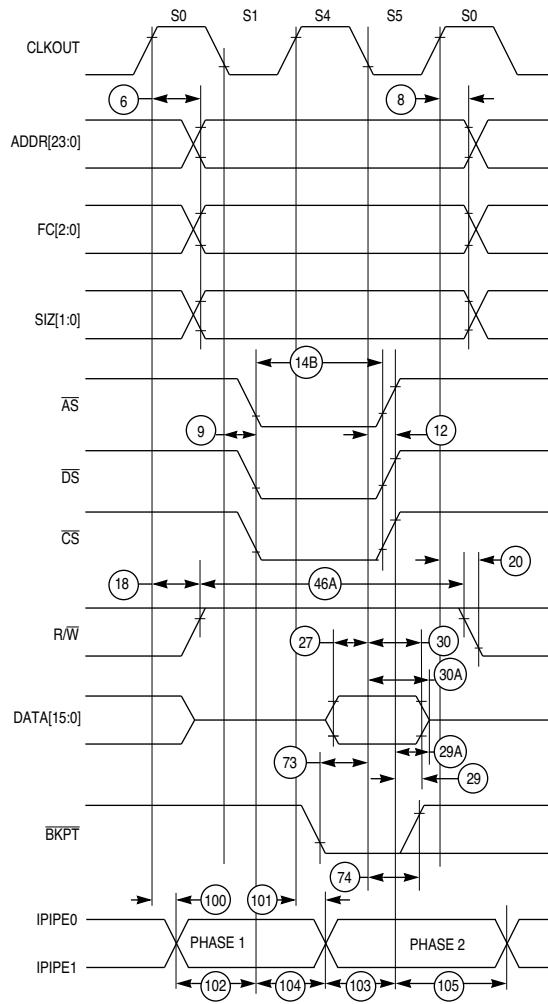
Figure 29 Fast Termination Read Cycle Timing Diagram

16 FAST RD CYC TIM
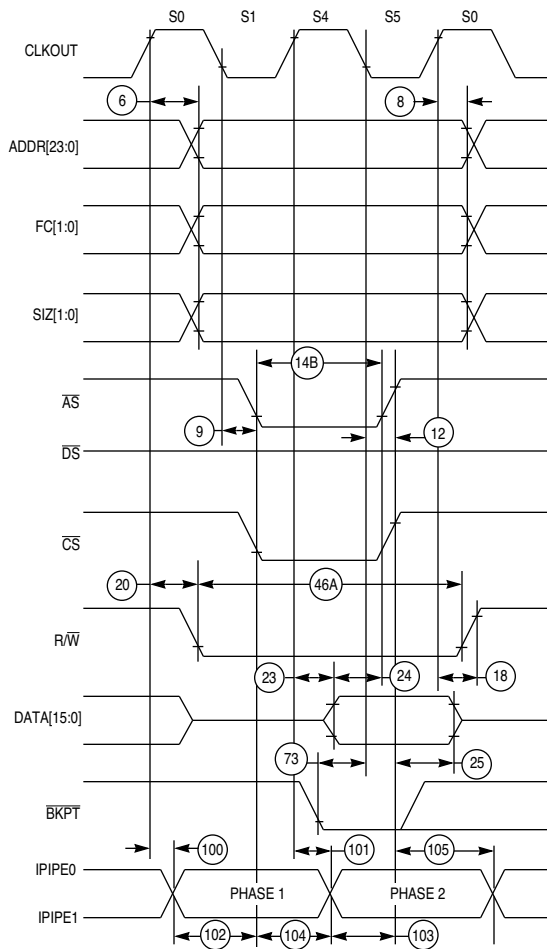
16 FAST WR CYC TIM

Figure 30 Fast Termination Write Cycle Timing Diagram
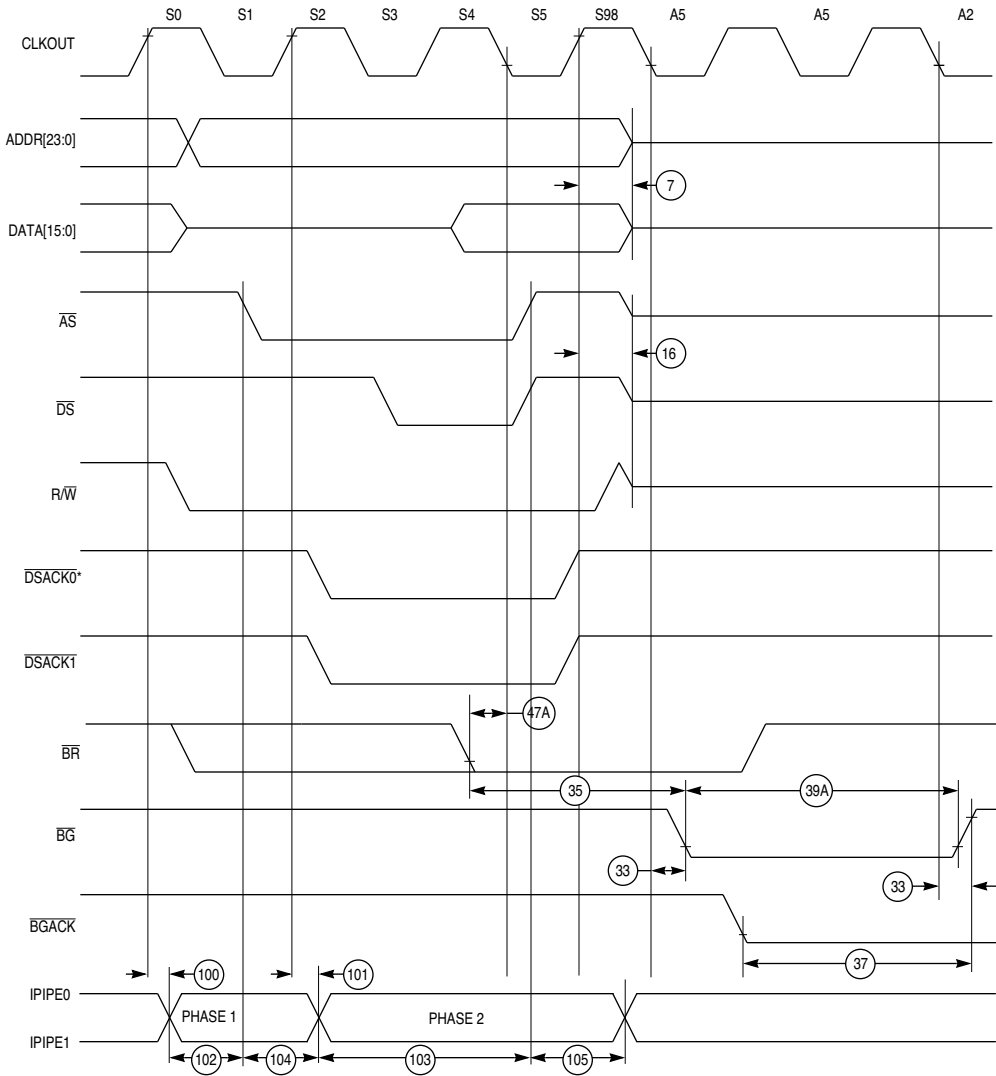
* ON THE MC68HC916X1 THE $\overline{\text{DSACK0}}$ PIN IS NOT BONDED AND IS INTERNALLY PULLED UP TO V_{DD}.

916X1 BUS ARB TIM

Figure 31 Bus Arbitration Timing Diagram — Active Bus Case
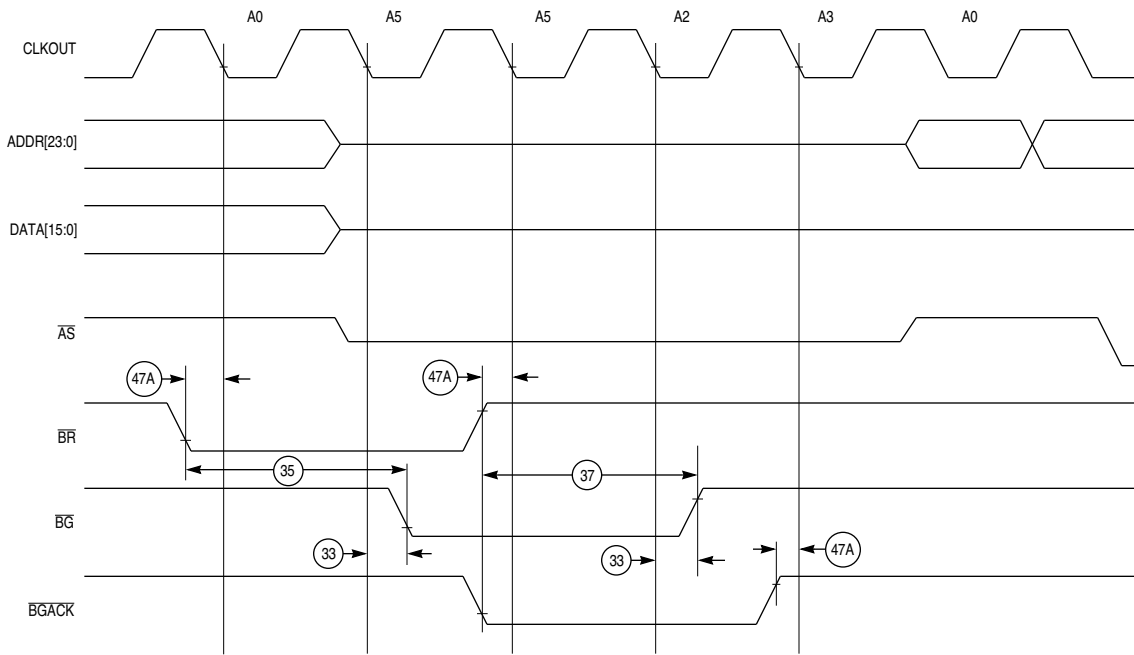
16 BUS ARB TIM IDLE

Figure 32 Bus Arbitration Timing Diagram — Idle Bus Case
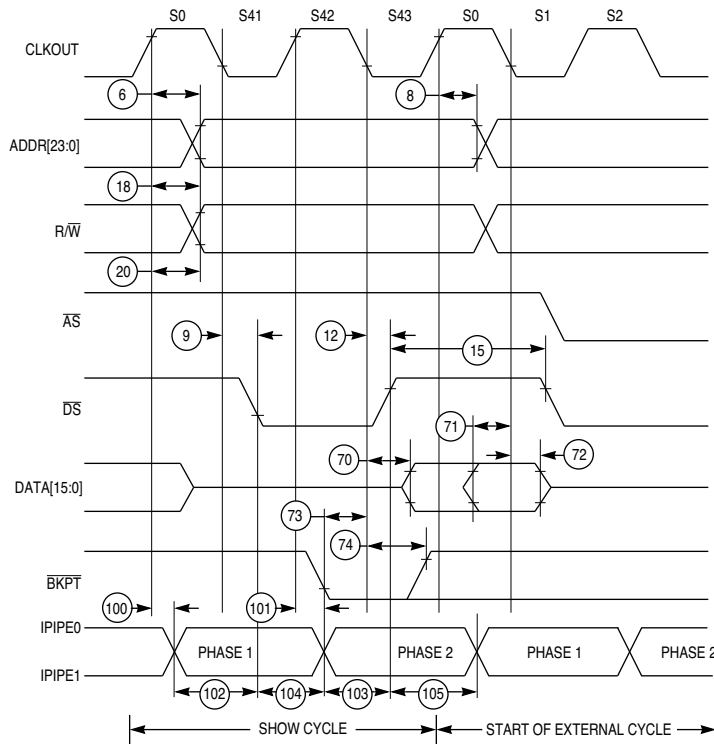
NOTE:
SHOW CYCLES CAN STRETCH DURING CLOCK PHASE S42 WHEN BUS ACCESSES TAKE LONGER THAN TWO CYCLES DUE TO IMB MODULE WAIT-STATE INSERTION.

16 SHW CYC TIM

Figure 33 Show Cycle Timing Diagram
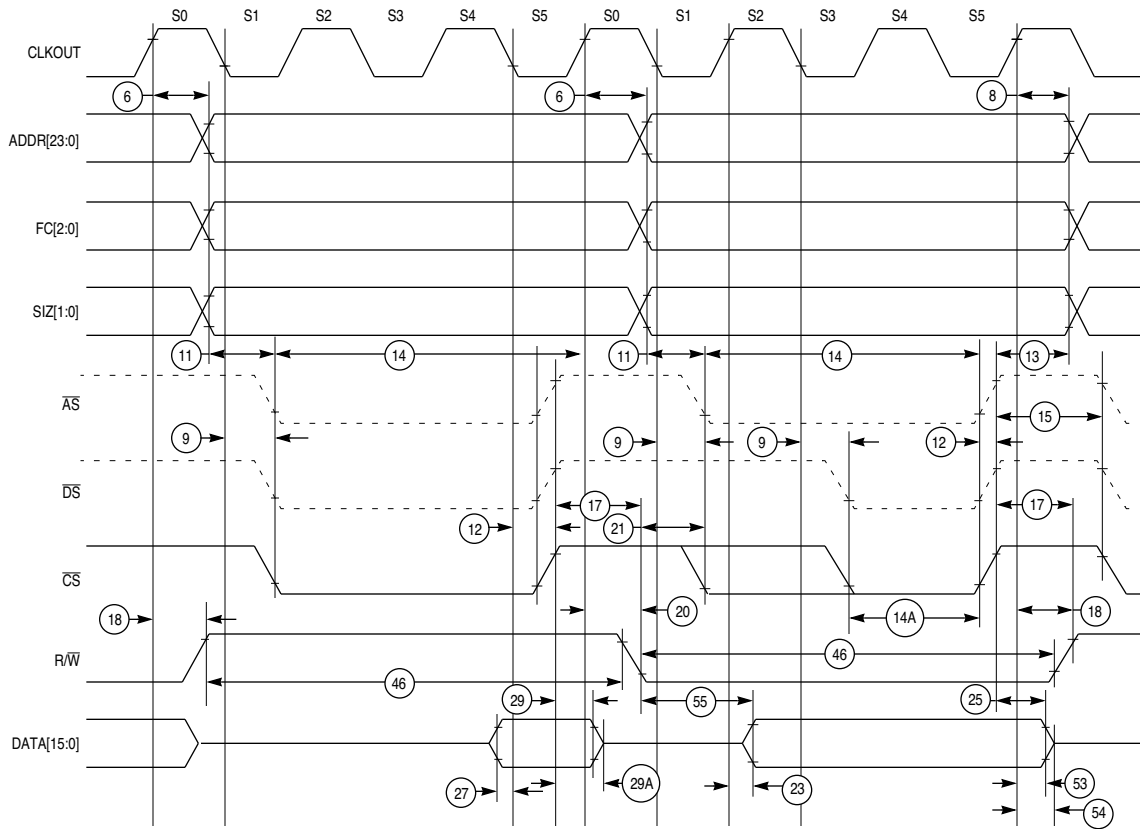
Figure 34 Chip Select Timing Diagram
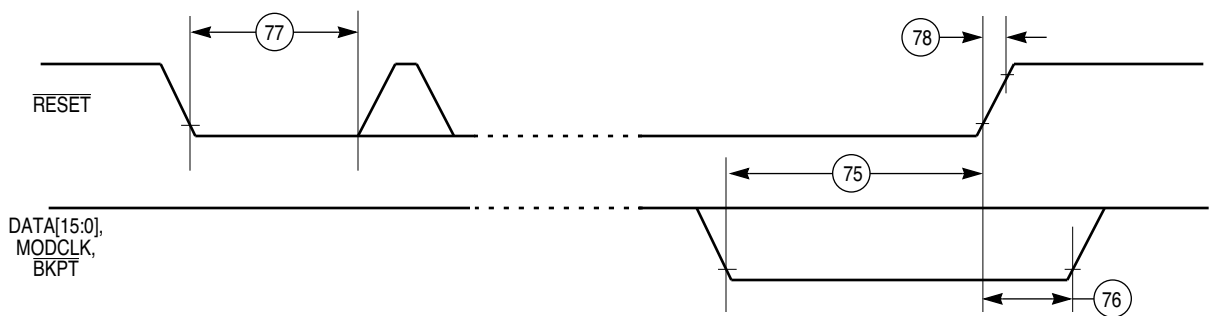
Figure 35 Reset and Mode Select Timing Diagram

Table 79 Background Debugging Mode Timing

$(V_{DD}$ and V_{DDSYN} = 5.0 Vdc $\pm$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 Low 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}

1. All AC timing is shown with respect to 20% V_{DD} and 70% V_{DD} levels unless otherwise noted.
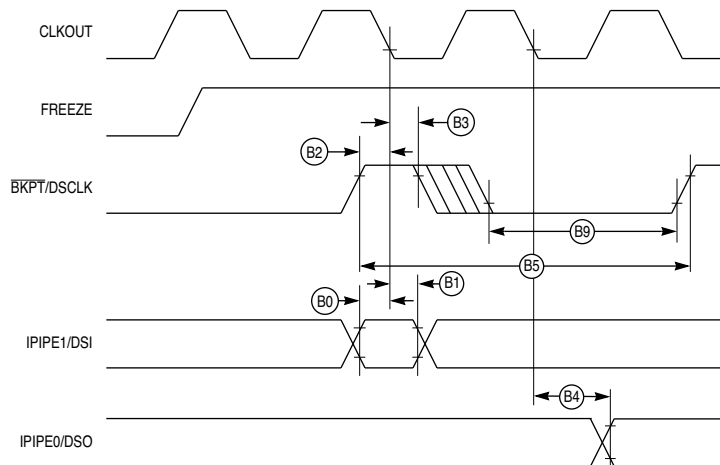
16 BDM SER COM TIM

**Figure 36 Background Debugging Mode Timing Diagram —
Serial Communication**
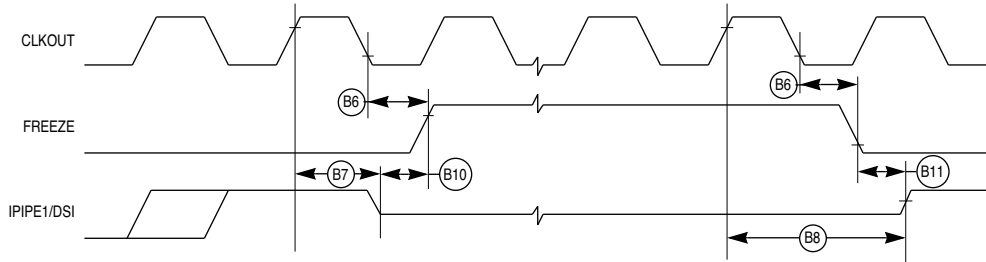
**Figure 37 Background Debugging Mode Timing Diagram —
Freeze Assertion**

Table 80 ECLK Bus 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
E1	ECLK Low to Address Valid[2]	t_{EAD}	—	60	ns
E2	ECLK Low to Address Hold	t_{EAH}	15	—	ns
E3	ECLK Low to $\overline{CS}$ Valid ($\overline{CS}$ Delay)	t_{ECSD}	—	150	ns
E4	ECLK Low to $\overline{CS}$ Hold	t_{ECSH}	15	—	ns
E5	$\overline{CS}$ Negated Width	t_{ECSN}	30	—	ns
E6	Read Data Setup Time	t_{EDSR}	30	—	ns
E7	Read Data Hold Time	t_{EDHR}	5	—	ns
E8	ECLK Low to Data High Impedance	t_{EDHZ}	—	60	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}	15	—	ns
E13	$\overline{CS}$ Negated to Data Hold (Write)	t_{ECHW}	0	—	ns
E14	Address Access Time (Read)[3]	t_{EACC}	386	—	ns
E15	Chip-Select Access Time (Read)[4]	t_{EACS}	296	—	ns
E16	Address Setup Time	t_{EAS}	—	1/2	t_{cyc}

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 = $t_{Ecyc} - t_{ECSD} - t_{EDSR}$.
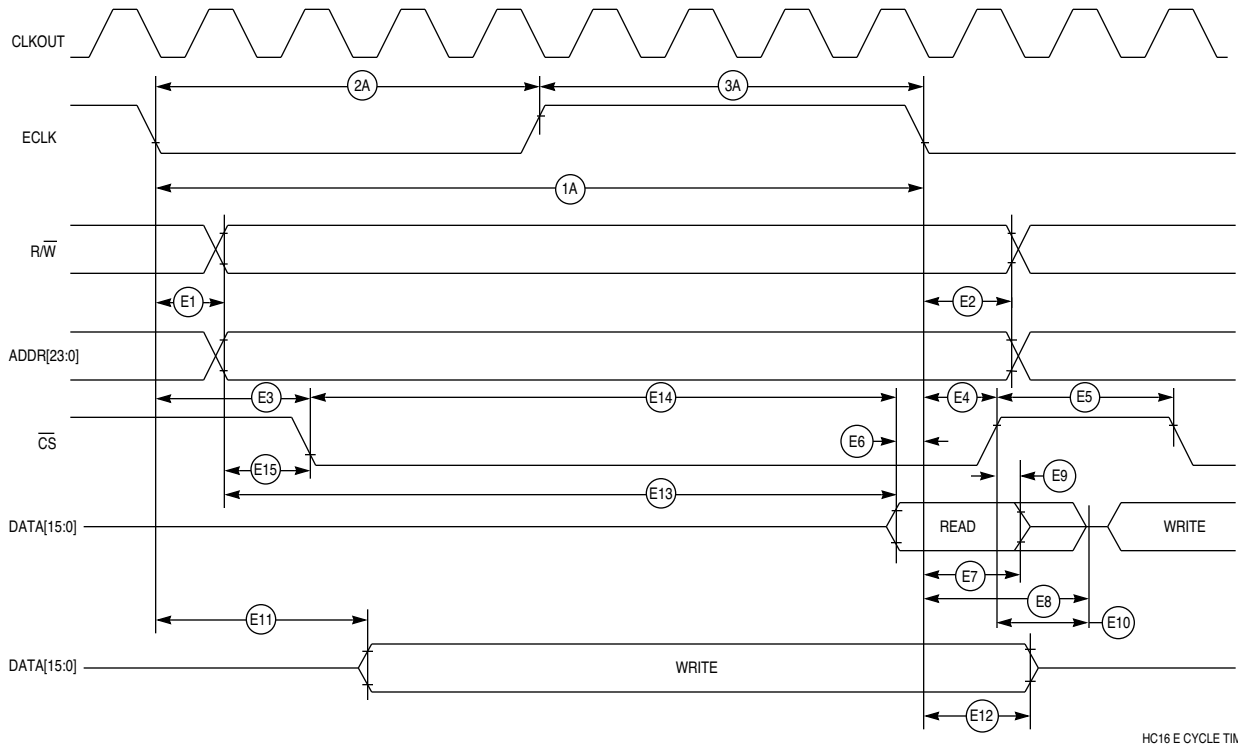
Figure 38 ECLK Timing Diagram

Table 81 QSPI Timing

(V_{DD} and V_{DDSYN} = 5.0 Vdc ±10%, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H 200 pF load on all QSPI pins)[1]

Num	Function	Symbol	Min	Max	Unit
0	Operating Frequency Master Slave	f_{op}	DC DC	1/4 1/4	System Clock Frequency System Clock Frequency
1	Cycle Time Master Slave	t_{qcyc}	4 4	510 —	t_{cyc} t_{cyc}
2	Enable Lead Time Master Slave	t_{lead}	2 2	128 —	t_{cyc} t_{cyc}
3	Enable Lag Time Master Slave	t_{lag}	— 2	1/2 —	SCK t_{cyc}
4	Clock (SCK) High or Low Time Master Slave[2]	t_{sw}	$2\,t_{cyc} - 60$ $2\,t_{cyc} - n$	$255\,t_{cyc}$ —	ns ns
5	Sequential Transfer Delay Master Slave (Does Not Require Deselect)	t_{td}	17 13	8192 —	t_{cyc} t_{cyc}
6	Data Setup Time (Inputs) Master Slave	t_{su}	30 20	— —	ns ns
7	Data Hold Time (Inputs) Master Slave	t_{hi}	0 20	— —	ns ns
8	Slave Access Time	t_a	—	1	t_{cyc}
9	Slave MISO Disable Time	t_{dis}	—	2	t_{cyc}
10	Data Valid (after SCK Edge) Master Slave	t_v	— —	50 50	ns ns
11	Data Hold Time (Outputs) Master Slave	t_{ho}	0 0	— —	ns ns
12	Rise Time Input Output	t_{ri} t_{ro}	— —	2 30	µs ns
13	Fall Time Input[3] Output	t_{fi} t_{fo}	— —	2 30	µs ns

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

2. For high time, n = External SCK rise time; for low time, n = External SCK fall time.

3. Data can be recognized properly with longer transition times as long as MOSI/MISO signals from external sources are at valid V_{OH}/V_{OL} prior to SCK transitioning between valid V_{OL} and V_{OH}. Due to process variation, logic decision point voltages of the data and clock signals can differ, which can corrupt data if slower transition times are used.
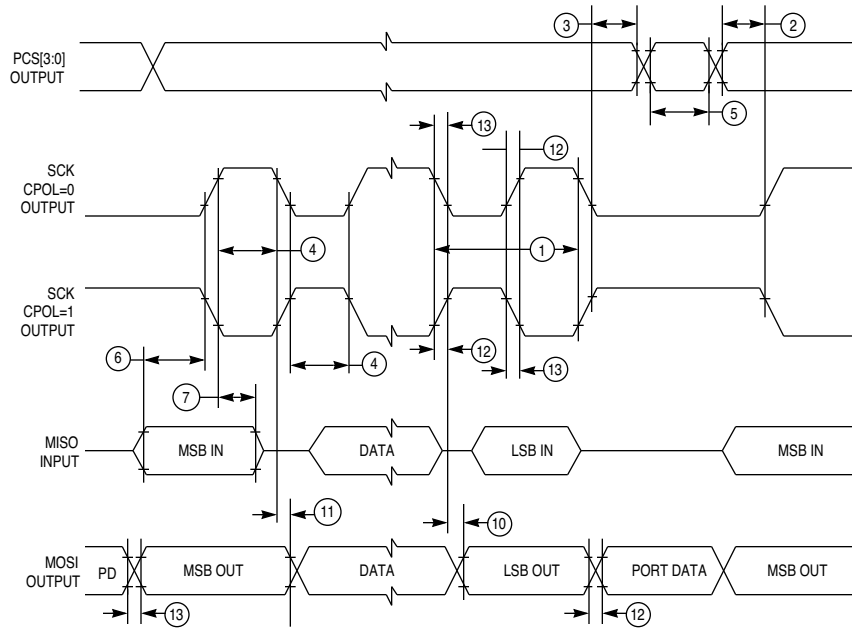
16 QSPI MAST CPHA0

Figure 39 QSPI Timing — Master, CPHA = 0
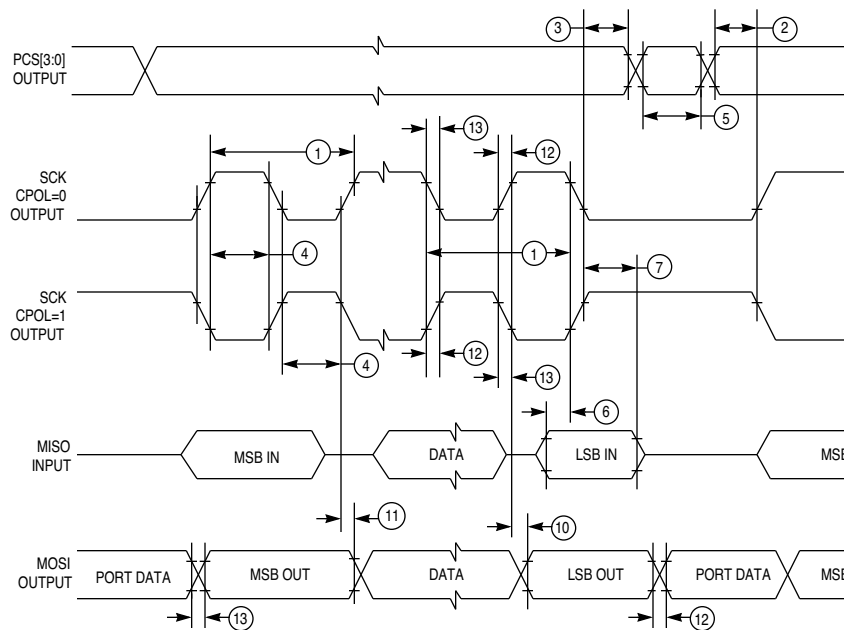
16 QSPI MAST CPHA1

Figure 40 QSPI Timing — Master, CPHA = 1
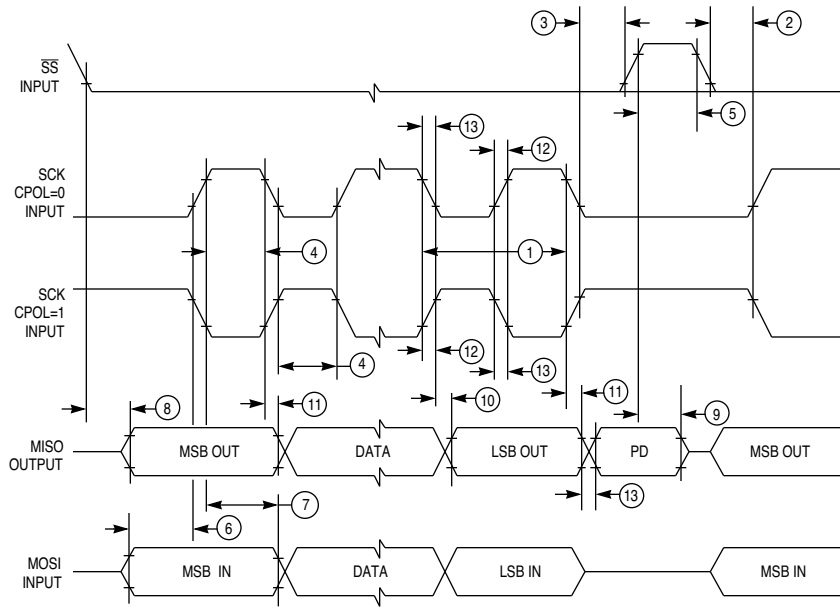
16 QSPI SLV CPHA0

Figure 41 QSPI Timing — Slave, CPHA = 0
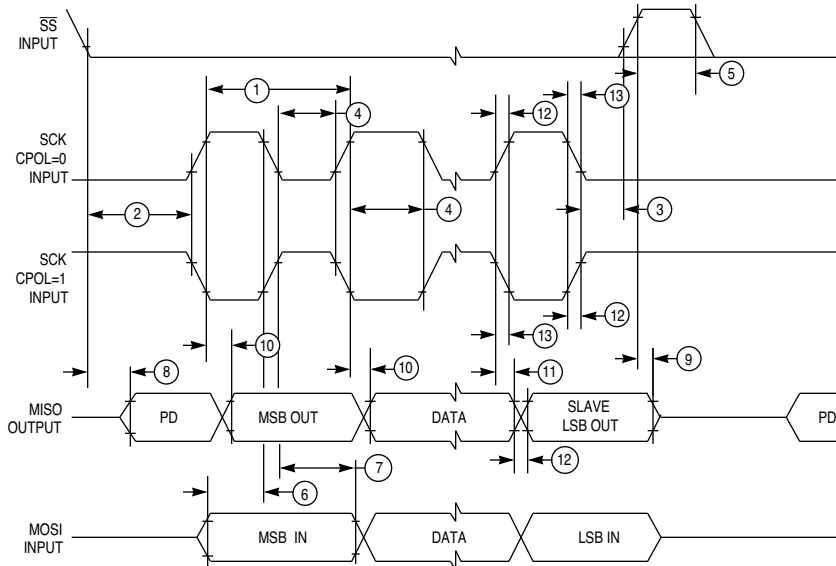
16 QSPI SLV CPHA1

Figure 42 QSPI Timing — Slave, CPHA = 1

Table 82 ADC Maximum Ratings

Num	Parameter	Symbol	Min	Max	Unit
1	Analog Supply	V_{DDA}	-0.3	6.5	V
2	Internal Digital Supply, with reference to V_{SSI}	V_{DDI}	-0.3	6.5	V
3	Reference Supply, with reference to V_{SSI}	V_{RH}, V_{RL}	-0.3	6.5	V
4	V_{SS} Differential Voltage	$V_{SSI} - V_{SSA}$	-0.1	0.1	V
5	V_{DD} Differential Voltage	$V_{DDI} - V_{DDA}$	-6.5	6.5	V
6	V_{REF} Differential Voltage	$V_{RH} - V_{RL}$	-6.5	6.5	V
7	V_{RH} to V_{DDA} Differential Voltage	$V_{RH} - V_{DDA}$	-6.5	6.5	V
8	V_{RL} to V_{SSA} Differential Voltage	$V_{RL} - V_{SSA}$	-6.5	6.5	V
9	Disruptive Input Current[1, 2, 3, 4, 5, 6, 7] $V_{NEGCLAMP} \cong -0.3V$ $V_{POSCLAMP} \cong 8$ V	I_{NA}	-500	500	µA
10	Positive Overvoltage Current Coupling Ratio[1, 5, 6, 8]	K_P	2000	—	—
11	Negative Overvoltage Current Coupling Ratio[1, 5, 6, 8]	K_N	500	—	—
12	Maximum Input Current [3, 4, 6] $V_{NEGCLAMP} \cong -0.3$ V $V_{POSCLAMP} \cong 8$ V	I_{MA}	-25	25	mA

1. Below disruptive current conditions, a stressed channel will store the maximum conversion value for analog inputs greater than V_{RH} and the minimum conversion value for inputs less than V_{RL}. This assumes that $V_{RH} \leq V_{DDA}$ and $V_{RL} \geq V_{SSA}$ due to the presence of the sample amplifier. Other channels are not affected by non-disruptive conditions

2. Input signals with large slew rates or high frequency noise components cannot be converted accurately. These signals also interfere with conversion of other channels.

3. Exceeding limit may cause conversion error on stressed channels and on unstressed channels. Transitions within the limit do not affect device reliability or cause permanent damage.

4. Input must be current limited to the value specified. To determine the value of the required current-limiting resistor, calculate resistance values using positive and negative clamp values, then use the larger of the calculated values.

5. This parameter is periodically sampled rather than 100% tested.

6. Applies to single pin only.

7. The values of external system components can change the maximum input current value, and affect operation. A voltage drop may occur across the external source impedances of the adjacent pins, impacting conversions on these adjacent pins. The actual maximum may need to be determined by testing the complete design.

8. Current coupling is the ratio of the current induced from overvoltage (positive or negative, through an external series coupling resistor), divided by the current induced on adjacent pins. A voltage drop may occur across the external source impedances of the adjacent pins, impacting conversions on these adjacent pins.

Table 83 ADC DC Electrical Characteristics (Operating)

(V_{SS} = 0 Vdc, ADCLK = 2.1 MHz, $T_A = T_L$ to T_H)

Num	Parameter	Symbol	Min	Max	Unit
1	Analog Supply[1]	V_{DDA}	4.5	5.5	V
2	Internal Digital Supply[1]	V_{DDI}	4.5	5.5	V
3	V_{SS} Differential Voltage	$V_{SSI} - V_{SSA}$	− 1.0	1.0	mV
4	V_{DD} Differential Voltage	$V_{DDI} - V_{DDA}$	− 1.0	1.0	V
5	Reference Voltage Low[2, 3]	V_{RL}	V_{SSA}	V_{DDA} / 2	V
6	Reference Voltage High[2, 3]	V_{RH}	V_{DDA} / 2	V_{DDA}	V
7	V_{REF} Differential Voltage[3]	$V_{RH} - V_{RL}$	4.5	5.5	V
8	Input Voltage[2]	V_{INDC}	V_{SSA}	V_{DDA}	V
9	Input High, Port ADA	V_{IH}	0.7 (V_{DDA})	V_{DDA} + 0.3	V
10	Input Low, Port ADA	V_{IL}	V_{SSA} − 0.3	0.2 (V_{DDA})	V
11	Analog Supply Current Normal Operation[4] Low-Power Stop	I_{DDA}	— —	1.0 200	mA μA
12	Reference Supply Current	I_{REF}	—	250	μA
13	Input Current, Off Channel[5]	I_{OFF}	—	150	nA
14	Total Input Capacitance, Not Sampling	C_{INN}	—	10	pF
15	Total Input Capacitance, Sampling	C_{INS}	—	15	pF

1. Refers to operation over full temperature and frequency range.
2. To obtain full-scale, full-range results, $V_{SSA} \leq V_{RL} \leq V_{INDC} \leq V_{RH} \leq V_{DDA}$.
3. Accuracy tested and guaranteed at $V_{RH} - V_{RL} \leq 5.0$ V ± 10%.
4. Current measured at maximum system clock frequency with ADC active.
5. Maximum leakage occurs at maximum operating temperature. Current decreases by approximately one-half for each 10 °C decrease from maximum temperature.

Table 84 ADC AC Characteristics (Operating)

Num	Parameter	Symbol	Min	Max	Unit
1	ADC Clock Frequency	f_{ADCLK}	0.5	2.1	MHz
2	8-bit Conversion Time[1] f_{ADCLK} = 1.0 MHz f_{ADCLK} = 2.1 MHz	t_{CONV}	15.2 7.6	—	μs
3	10-bit Conversion Time[1] f_{ADCLK} = 1.0 MHz f_{ADCLK} = 2.1 MHz	t_{CONV}	17.1 8.6	—	μs
4	Stop Recovery Time	t_{SR}	—	10	μs

1. Conversion accuracy varies with f_{ADCLK} rate. Reduced conversion accuracy occurs at maximum.

Table 85 ADC Conversion Characteristics (Operating)

$(V_{DD}$ and V_{DDA} = 5.0 Vdc $\pm$ 5%, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H,

0.5 MHz $\leq f_{ADCLK} \leq$ 1.0 MHz, 2 Clock Input Sample Time)

Num	Parameter	Symbol	Min	Typ	Max	Unit
1	8-bit Resolution[1]	1 Count	—	20	—	mV
2	8-bit Differential Nonlinearity	DNL	–0.5	—	–0.5	Counts
3	8-bit Integral Nonlinearity	INL	–1	—	1	Counts
4	8-bit Absolute Error[2]	AE	–1	—	1	Counts
5	10-bit Resolution[1]	1 Count	—	5	—	mV
6	10-bit Differential Nonlinearity[3]	DNL	–0.5	—	–0.5	Counts
7	10-bit Integral Nonlinearity[3]	INL	–2.0	—	2.0	Counts
8	10-bit Absolute Error[3,4]	AE	–2.5	—	2.5	Counts
9	Source Impedance at Input[5]	R_S	—	20	—	kΩ

1. At $V_{RH} - V_{RL}$ = 5.12 V, one 10-bit count = 5 mV and one 8-bit count = 20 mV.
2. 8-bit absolute error of 1 count (20 mV) includes 1/2 count (10 mV) inherent quantization error and 1/2 count (10 mV) circuit (differential, integral, and offset) error.
3. Conversion accuracy varies with f_{ADCLK} rate. Reduced conversion accuracy occurs at maximum f_{ADCLK}. Assumes that minimum sample time (2 ADC Clocks) is selected.
4. 10-bit absolute error of 2.5 counts (12.5 mV) includes 1/2 count (2.5 mV) inherent quantization error and 2 counts (10 mV) circuit (differential, integral, and offset) error.
5. Maximum source impedance is application-dependent. Error resulting from pin leakage depends on junction leakage into the pin and on leakage due to charge-sharing with internal capacitance.
   Error from junction leakage is a function of external source impedance and input leakage current. Expected error in result value due to junction leakage is expressed in voltage (V_{ERRJ}):

$$V_{ERRJ} = R_S \times I_{OFF}$$

   where I_{OFF} is a function of operating temperature, as shown in **Table 83**.
   Charge-sharing leakage is a function of input source impedance, conversion rate, change in voltage between successive conversions, and the size of the decoupling capacitor used. Error levels are best determined empirically. In general, continuous conversion of the same channel may not be compatible with high source impedance.
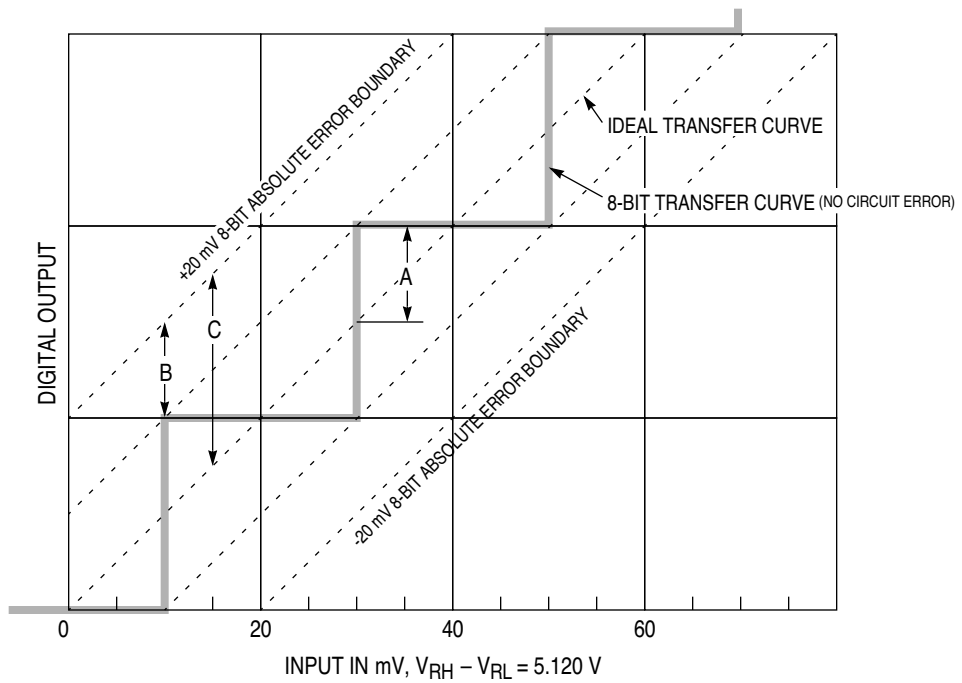
A – +1/2 COUNT (10 mV) INHERENT QUANTIZATION ERROR
B – CIRCUIT-CONTRIBUTED +10 mV ERROR
C – + 20 mV ABSOLUTE ERROR (ONE 8-BIT COUNT)

ADC 8-BIT ACCURACY

Figure 43 8-Bit ADC Conversion Accuracy
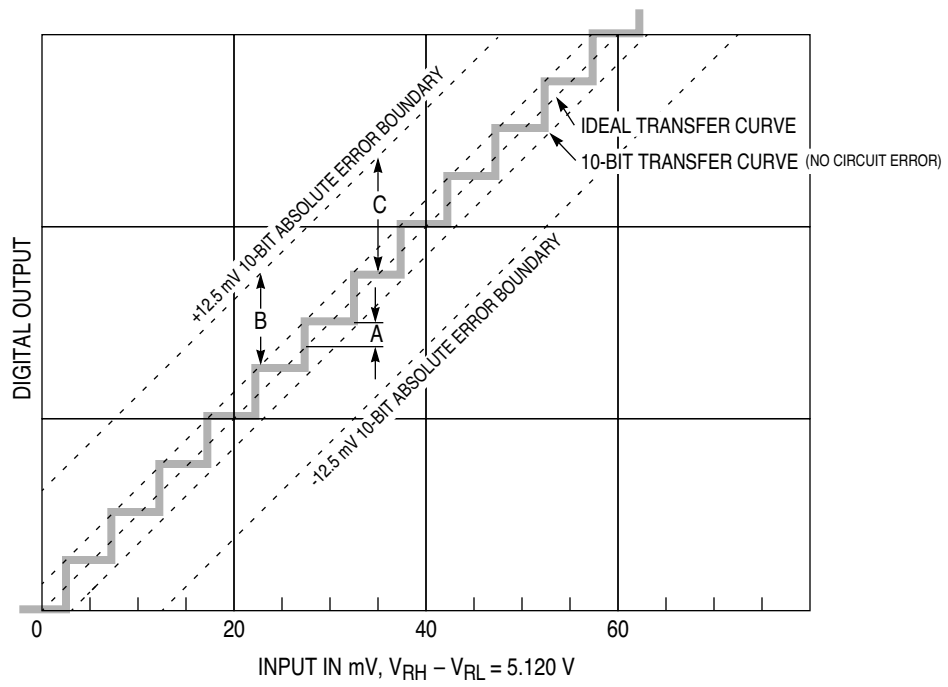
A – +.5 COUNT (2.5 mV) INHERENT QUANTIZATION ERROR
B – CIRCUIT-CONTRIBUTED +10 mV ERROR
C – +12.5 mV ABSOLUTE ERROR (2.5 10-BIT COUNTS)

ADC 10-BIT ACCURACY

Figure 44 10-Bit ADC Conversion Accuracy

Table 86 BEFLASH/Flash EEPROM Module Specifications

Num	Characteristic	Symbol	Min	Max	Unit

1	Program/Erase Supply Voltage[1]
	  Read Operation
	  Program/Erase/Verify Operation
		V_{FPE}	$V_{DD} - 0.5$
			11.4
				5.5
				12.6
					V

2	Program/Erase Supply Current[2]
	  Read Operation
	  Program/Erase/Verify Operation
	    Verify (ENPE = 0)
	    Program Byte (ENPE = 1)
	    Program Word (ENPE = 1)
	    Erase (ENPE = 1)
		I_{FPE}	—
			
			—
			—
			—
			—
				15
				
				50
				15
				30
				4
					µA
					
					µA
					mA
					mA
					mA

3	Program Recovery Time	t_{pr}	—	3	cycles
4	Program Pulse Width	pw_{pp}	20	25	µA
5	Number of Program Pulses[3]	n_{pp}	—	50	—
6	Program Margin[4]	p_m	100	—	%
7	Number of Erase Pulses[3]	n_{ep}	—	5	—
8	Erase Pulse Time	t_{epk}	—	$t_{ei} \times k$	ms
9	Amount to Increment t_{ep}	t_{ei}	90	110	ms
10	Erase Margin	e_m	—	$\sum_{k=1}^{n_{ep}} t_{ei} \times k$	ms
11	Erase Recovery Time	t_{er}	1	—	ms
12	Low-Power Stop Recovery Time[5]	t_{sb}	1	—	µA

1. V_{FPE} must not be raised to programming voltage while V_{DD} is below specified minimum value. V_{FPE} must not be reduced below minimum specified value while V_{DD} is applied.
2. Current parameters apply to each individual EEPROM module.
3. Without margin.
4. At 100% margin, the number of margin pulses required is the same as the number of pulses used to program the byte or word.
5. Parameter measured from end of write cycle that clears STOP bit in FEEMCR.

Table 87 BEFLASH/Flash EEPROM Module Life

Num	Parameter	Symbol	Value	Unit
1	Program-Erase Endurance[1]	e_{pe}	100	cyc
2	Data Retention[2]	r_d	10	yr

1. Number of program-erase cycles (1 to 0, 0 to 1) per bit.
2. Parameter based on accelerated-life testing with standard test pattern.

13.5 V

12.6 V

11.4 V

6.5 V

4.5 V
4.0 V

0 V
− 0.30 V

30 ns MAXIMUM

V$_{FPE}$ ENVELOPE

V$_{DD}$ ENVELOPE

COMBINED V$_{DD}$ AND V$_{FPE}$

POWER
ON

NORMAL
READ

PROGRAM
ERASE
VERIFY

POWER
DOWN
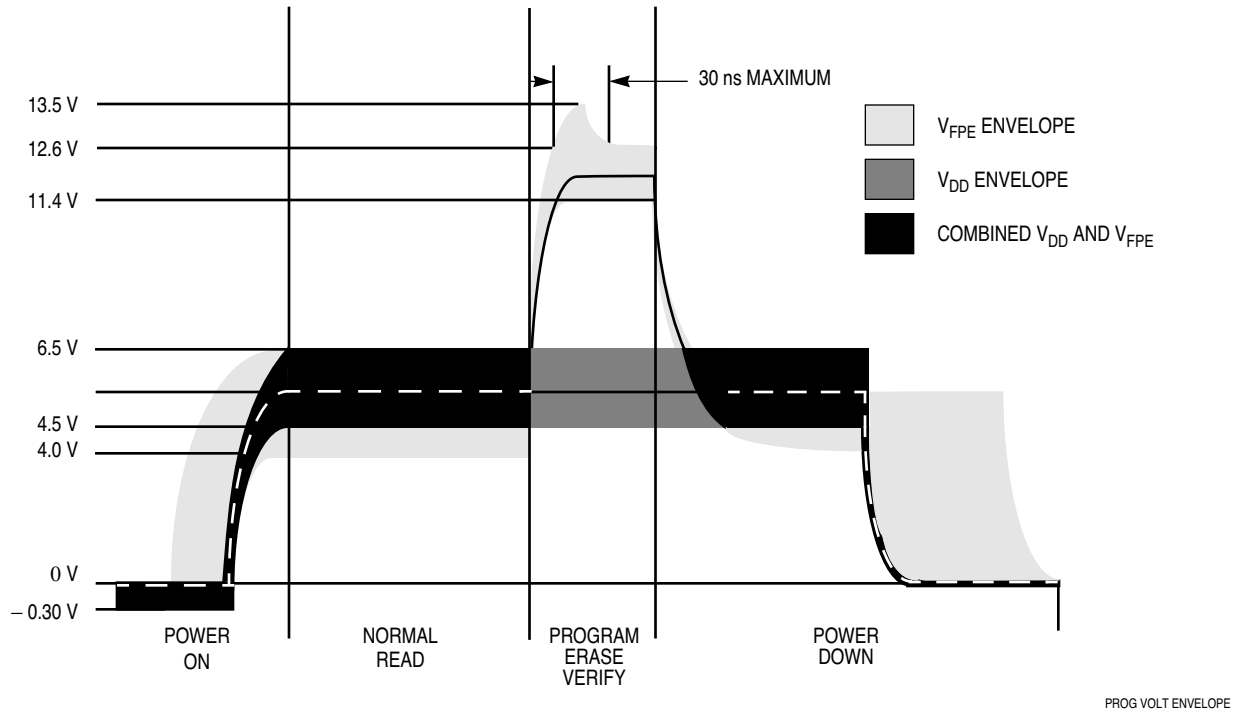
PROG VOLT ENVELOPE

Figure 45 Programming Voltage Envelope

PROGRAMMING VOLTAGE
POWER SUPPLY

D1

R1
10 kΩ

D2

V$_{DD}$ 4.5 V

V$_{FPE}$
PIN

R2
22 kΩ

C1
0.1 µF
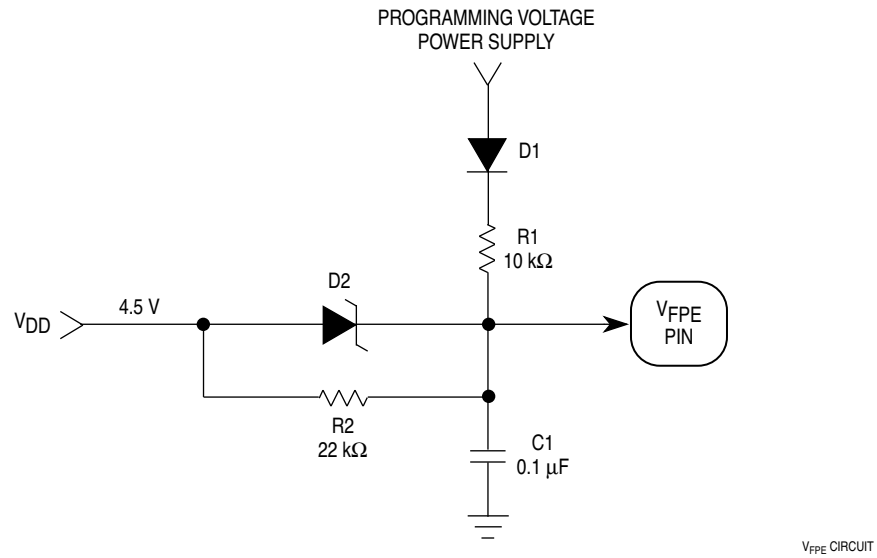
V$_{FPE}$ CIRCUIT

Figure 46 V$_{FPE}$ Conditioning Circuit

NOTES

Ⓜ *MOTOROLA*