

M68HC16 Family

MC68HC16Y1

**USER'S
MANUAL**



MOTOROLA

INTRODUCTION	1
NOMENCLATURE	2
OVERVIEW	3
SINGLE-CHIP INTEGRATION MODULE	4
CENTRAL PROCESSING UNIT	5
ANALOG-TO-DIGITAL CONVERTER	6
MULTICHANNEL COMMUNICATION INTERFACE	7
GENERAL-PURPOSE TIMER	8
TIME PROCESSOR UNIT	9
STANDBY RAM WITH TPU EMULATION	10
MASKED ROM MODULE	11
ELECTRICAL CHARACTERISTICS	A
MECHANICAL DATA AND ORDERING INFORMATION	B
DEVELOPMENT SUPPORT	C
REGISTER SUMMARY	D
INDEX	I

- 1 INTRODUCTION**
- 2 NOMENCLATURE**
- 3 OVERVIEW**
- 4 SINGLE-CHIP INTEGRATION MODULE**
- 5 CENTRAL PROCESSING UNIT**
- 6 ANALOG-TO-DIGITAL CONVERTER**
- 7 MULTICHANNEL COMMUNICATION INTERFACE**
- 8 GENERAL-PURPOSE TIMER**
- 9 TIME PROCESSOR UNIT**
- 10 STANDBY RAM WITH TPU EMULATION**
- 11 MASKED ROM MODULE**
- A ELECTRICAL CHARACTERISTICS**
- B MECHANICAL DATA AND ORDERING INFORMATION**
- C DEVELOPMENT SUPPORT**
- D REGISTER SUMMARY**
- I INDEX**

MC68HC16Y1

USER'S MANUAL


Motorola reserves the right to make changes without further notice to any products herein. Motorola makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Motorola assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters can and do vary in different applications. All operating parameters, including "Typicals" must be validated for each customer application by customer's technical experts. Motorola does not convey any license under its patent rights nor the rights of others. Motorola products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Motorola product could create a situation where personal injury or death may occur. Should Buyer purchase or use Motorola products for any such unintended or unauthorized application, Buyer shall indemnify and hold Motorola and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Motorola was negligent regarding the design or manufacture of the part. Motorola and  are registered trademarks of Motorola, Inc. Motorola, Inc. is an Equal Opportunity/Affirmative Action Employer.

TABLE OF CONTENTS

Paragraph	Title	Page
Section 1		
Introduction		
Section 2		
Nomenclature		
2.1	Symbols and Operators.....	2-1
2.2	CPU16 Registers	2-2
2.3	Pin and Signal Mnemonics.....	2-3
2.4	Register Mnemonics.....	2-5
2.5	Conventions.....	2-8
Section 3		
Overview		
3.1	MC68HC16Y1 Features.....	3-1
3.1.1	Single-Chip Integration Module.....	3-1
3.1.2	CPU16	3-1
3.1.3	Analog-to-Digital Converter.....	3-2
3.1.4	Multichannel Communication Interface.....	3-2
3.1.5	General-Purpose Timer.....	3-2
3.1.6	Time Processor Unit.....	3-2
3.1.7	Standby RAM with TPU Emulation.....	3-2
3.1.8	Masked ROM	3-2
3.2	System Block and Pin Assignment Diagrams.....	3-2
3.3	Pin Descriptions.....	3-5
3.4	Signal Descriptions.....	3-7
3.5	Intermodule Bus.....	3-10
3.6	System Memory Maps.....	3-10
3.6.1	Internal Register Map.....	3-10
3.6.2	Pseudolinear Address Maps	3-11
3.7	System Reset.....	3-15

TABLE OF CONTENTS (Continued)

Paragraph	Title	Page
Section 4		
Single-Chip Integration Module		
4.1	General.....	4-1
4.2	System Configuration and Protection.....	4-3
4.2.1	Module Mapping.....	4-5
4.2.2	Interrupt Arbitration.....	4-5
4.2.3	Show Internal Cycles.....	4-5
4.2.4	Factory Test Mode.....	4-6
4.2.5	Register Access.....	4-6
4.2.6	Bus Monitor.....	4-6
4.2.7	Halt Monitor.....	4-7
4.2.8	Spurious Interrupt Monitor.....	4-7
4.2.9	Software Watchdog.....	4-7
4.2.10	Periodic Interrupt Timer.....	4-9
4.2.11	Monitor Low-Power Operation.....	4-11
4.2.12	Freeze Operation.....	4-11
4.3	System Clock.....	4-12
4.3.1	Clock Sources.....	4-13
4.3.2	Clock Synthesizer Operation.....	4-13
4.3.3	External Bus Clock.....	4-19
4.3.4	Clock Low-Power Operation.....	4-19
4.3.5	Loss of Reference Signal.....	4-20
4.4	External Bus Interface.....	4-21
4.4.1	Bus Signals.....	4-22
4.4.1.1	Address Bus.....	4-22
4.4.1.2	Address Strobe.....	4-22
4.4.1.3	Data Bus.....	4-22
4.4.1.4	Data Strobe.....	4-23
4.4.1.5	Read/Write Signal.....	4-23
4.4.1.6	Size Signals.....	4-23
4.4.1.7	Function Codes.....	4-23
4.4.1.8	Data and Size Acknowledge Signals.....	4-24
4.4.1.9	Bus Error Signal.....	4-24
4.4.1.10	Halt Signal.....	4-24
4.4.1.11	Autovector Signal.....	4-24
4.4.2	Dynamic Bus Sizing.....	4-25
4.4.3	Operand Alignment.....	4-26
4.4.4	Misaligned Operands.....	4-26
4.4.5	Operand Transfer Cases.....	4-27

TABLE OF CONTENTS (Continued)

Paragraph	Title	Page
4.5	Bus Operation.....	4-28
4.5.1	Synchronization to CLKOUT	4-28
4.5.2	Regular Bus Cycles.....	4-29
4.5.2.1	Read Cycle.....	4-30
4.5.2.2	Write Cycle.....	4-31
4.5.3	Fast Termination Cycles.....	4-32
4.5.4	CPU Space Cycles.....	4-33
4.5.4.1	Breakpoint Acknowledge Cycle	4-34
4.5.4.2	LPSTOP Broadcast Cycle	4-35
4.5.5	Bus Exception Control Cycles.....	4-36
4.5.5.1	Bus Errors.....	4-38
4.5.5.2	Double Bus Fault	4-38
4.5.5.3	Halt Operation	4-39
4.5.6	External Bus Arbitration	4-39
4.5.6.1	Slave (Factory Test) Mode Arbitration	4-41
4.5.6.2	Show Cycles.....	4-42
4.6	Reset.....	4-42
4.6.1	Reset Exception Processing	4-42
4.6.2	Reset Control Logic.....	4-43
4.6.3	Operating Configuration out of Reset.....	4-44
4.6.3.1	Address and Data Bus Pin Functions	4-45
4.6.3.2	Configuration for 16-Bit Data Bus Operation.....	4-45
4.6.3.3	Configuration for 8-Bit Data Bus Operation	4-47
4.6.3.4	Configuration for Single-Chip Operation	4-48
4.6.3.5	Clock Mode Selection.....	4-49
4.6.3.6	Breakpoint Mode Selection	4-50
4.6.4	MCU Module Pin Function During Reset	4-50
4.6.5	Pin State During Reset	4-51
4.6.5.1	Reset States of SCIM Pins.....	4-51
4.6.5.2	Reset States of Pins Assigned to Other MCU Modules.....	4-52
4.6.6	Reset Timing.....	4-53
4.6.7	Power-On Reset.....	4-53
4.6.8	Use of Three-State Control Pin.....	4-55
4.6.9	Reset Processing.....	4-55
4.6.10	Reset Status Register.....	4-56
4.7	Interrupts.....	4-57
4.7.1	Interrupt Exception Processing	4-57
4.7.2	Interrupt Priority and Recognition	4-57
4.7.3	Interrupt Acknowledge and Arbitration	4-58
4.7.4	Interrupt Processing Summary.....	4-60

TABLE OF CONTENTS (Continued)

Paragraph	Title	Page
4.7.5	Interrupt Acknowledge Bus Cycles.....	4-61
4.8	Chip Selects	4-62
4.8.1	Chip-Select Registers.....	4-64
4.8.1.1	Chip-Select Pin Assignment Registers.....	4-64
4.8.1.2	Chip-Select Base Address Registers.....	4-66
4.8.1.3	Chip-Select Option Registers.....	4-67
4.8.1.4	PORTC Data Register	4-69
4.8.2	Chip-Select Operation.....	4-69
4.8.3	Using Chip-Select Signals for Interrupt Acknowledge.....	4-69
4.8.4	Chip-Select Reset Operation.....	4-71
4.8.5	Emulation-Support Chip Selects.....	4-72
4.8.5.1	External Port Emulation.....	4-72
4.8.5.2	External ROM Emulation	4-72
4.9	Factory Test.....	4-72

Section 5 Central Processing Unit

5.1	General.....	5-1
5.2	Register Model.....	5-2
5.2.1	Accumulators.....	5-3
5.2.2	Index Registers.....	5-3
5.2.3	Stack Pointer	5-3
5.2.4	Program Counter	5-4
5.2.5	Condition Code Register.....	5-4
5.2.6	Address Extension Register and Address Extension Fields.....	5-5
5.2.7	Multiply and Accumulate Registers	5-5
5.3	Memory Management.....	5-6
5.3.1	Address Extension.....	5-6
5.3.2	Extension Fields.....	5-6
5.4	Data Types.....	5-7
5.5	Memory Organization.....	5-8
5.6	Addressing Modes.....	5-10
5.6.1	Immediate Addressing Modes.....	5-11
5.6.2	Extended Addressing Modes	5-11
5.6.3	Indexed Addressing Modes.....	5-11
5.6.4	Inherent Addressing Mode.....	5-11
5.6.5	Accumulator Offset Addressing Mode.....	5-12
5.6.6	Relative Addressing Modes.....	5-12
5.6.7	Post-Modified Index Addressing Mode.....	5-12

TABLE OF CONTENTS (Continued)

Paragraph	Title	Page
5.6.8	Use of CPU16 Indexed Mode to Replace M68HC11 Direct Mode	5-12
5.7	Instruction Set.....	5-12
5.7.1	Instruction Set Summary.....	5-13
5.8	Comparison of CPU16 and M68HC11 CPU Instruction Sets	5-30
5.9	Instruction Format	5-32
5.10	Execution Model	5-34
5.10.1	Microsequencer	5-34
5.10.2	Instruction Pipeline	5-35
5.10.3	Execution Unit	5-35
5.11	Execution Process.....	5-35
5.11.1	Changes in Program Flow	5-35
5.12	Instruction Timing.....	5-36
5.13	Exceptions.....	5-37
5.13.1	Exception Vectors.....	5-37
5.13.2	Exception Stack Frame.....	5-38
5.13.3	Exception Processing Sequence	5-39
5.13.4	Types of Exceptions	5-39
5.13.4.1	Asynchronous Exceptions.....	5-39
5.13.4.2	Synchronous Exceptions	5-39
5.13.5	Multiple Exceptions	5-40
5.13.6	RTI Instruction.....	5-40
5.14	Development Support.....	5-41
5.14.1	Deterministic Opcode Tracking.....	5-41
5.14.1.1	IPIPE0/IPIPE1 Multiplexing	5-41
5.14.1.2	Combining Opcode Tracking with Other Capabilities.....	5-42
5.14.2	Breakpoints.....	5-42
5.14.3	Opcode Tracking and Breakpoints.....	5-42
5.14.4	Background Debugging Mode.....	5-43
5.14.4.1	Enabling BDM	5-43
5.14.4.2	BDM Sources	5-43
5.14.4.2.1	BKPT Signal	5-43
5.14.4.2.2	BGND Instruction	5-44
5.14.4.3	Entering BDM	5-44
5.14.4.4	BDM Commands.....	5-44
5.14.4.5	Returning from BDM.....	5-45
5.14.4.6	BDM Serial Interface	5-45
5.15	Recommended BDM Connection.....	5-46
5.16	Digital Signal Processing.....	5-47

TABLE OF CONTENTS (Continued)

Paragraph	Title	Page
Section 6		
Analog-to-Digital Converter		
6.1	General.....	6-1
6.2	External Connections.....	6-1
6.2.1	Analog Input Pins.....	6-3
6.2.2	Analog Reference Pins.....	6-3
6.2.3	Analog Supply Pins.....	6-3
6.3	Programmer's Model.....	6-3
6.4	ADC Bus Interface Unit.....	6-4
6.5	Special Operating Modes.....	6-4
6.5.1	Low-Power Stop Mode.....	6-4
6.5.2	Freeze Mode.....	6-4
6.6	Analog Subsystem.....	6-5
6.6.1	Multiplexer.....	6-5
6.6.2	Sample Capacitor and Buffer Amplifier.....	6-6
6.6.3	RC DAC Array.....	6-6
6.6.4	Comparator.....	6-7
6.7	Digital Control Subsystem.....	6-7
6.7.1	Control/Status Registers.....	6-7
6.7.2	Clock and Prescaler Control.....	6-7
6.7.3	Sample Time.....	6-8
6.7.4	Resolution.....	6-8
6.7.5	Conversion Control Logic.....	6-9
6.7.5.1	Conversion Parameters.....	6-9
6.7.5.2	Conversion Modes.....	6-9
6.7.6	Conversion Timing.....	6-14
6.7.7	Successive Approximation Register.....	6-16
6.7.8	Result Registers.....	6-16

Section 7 **Multichannel Communication Interface**

7.1	General.....	7-2
7.2	MCCI Registers and Address Map.....	7-2
7.2.1	MCCI Global Registers.....	7-2
7.2.1.1	Low-Power Stop Operation.....	7-3
7.2.1.2	MCCI Interrupts.....	7-3
7.2.2	MCCI Pin Control.....	7-4
7.3	Serial Communication Interface.....	7-5

TABLE OF CONTENTS (Continued)

Paragraph	Title	Page
7.3.1	SCI Pins.....	7-5
7.3.2	SCI Registers.....	7-6
7.3.3	Serial Formats.....	7-6
7.3.4	Parity Checking.....	7-7
7.3.5	Baud Clock.....	7-8
7.3.6	SCI Transmitter.....	7-8
7.3.6.1	Transmitter Status Flags and Interrupts.....	7-8
7.3.7	SCI Receiver.....	7-9
7.3.7.1	Receiver Status Flags and Interrupts.....	7-10
7.3.7.2	Receiver Wakeup.....	7-10
7.3.7.3	Frame Detection and Synchronization.....	7-11
7.4	Serial Peripheral Interface.....	7-12
7.4.1	SPI Pins.....	7-12
7.4.2	SPI Registers.....	7-12
7.4.3	SPI Operation.....	7-13
7.4.3.1	Write Collision.....	7-13
7.4.3.2	Mode Fault.....	7-14

Section 8 General-Purpose Timer

8.1	General.....	8-1
8.2	GPT Registers and Address Map.....	8-2
8.3	Special Modes of Operation.....	8-3
8.3.1	Low-Power Stop Mode.....	8-3
8.3.2	Freeze Mode.....	8-3
8.3.3	Single-Step Mode.....	8-4
8.3.4	Test Mode.....	8-4
8.4	Polled and Interrupt-Driven Operation.....	8-4
8.4.1	Polled Operation.....	8-5
8.4.2	GPT Interrupts.....	8-5
8.5	Pin Descriptions.....	8-7
8.5.1	Input Capture Pins (IC[1:3]).....	8-7
8.5.2	Input Capture/Output Compare Pin (IC4/OC5).....	8-7
8.5.3	Output Compare Pins (OC[1:4]).....	8-8
8.5.4	Pulse Accumulator Input Pin (PAI).....	8-8
8.5.5	Pulse-Width Modulation (PWMA, PWMB).....	8-8
8.5.6	Auxiliary Timer Clock Input (PCLK).....	8-8
8.6	General-Purpose I/O.....	8-8
8.7	Prescaler.....	8-9

TABLE OF CONTENTS (Continued)

Paragraph	Title	Page
8.8	Capture/Compare Unit.....	8-11
8.8.1	Timer Counter.....	8-11
8.8.2	Input Capture Functions.....	8-11
8.8.3	Output Compare Functions.....	8-14
8.8.3.1	Output Compare 1.....	8-15
8.8.3.2	Forced Output Compare.....	8-15
8.9	Input Capture 4/Output Compare 5.....	8-15
8.10	Pulse Accumulator.....	8-16
8.11	Pulse-Width Modulation Unit.....	8-17
8.11.1	PWM Counter.....	8-18
8.11.2	PWM Function.....	8-19

Section 9 Time Processor Unit

9.1	Overview.....	9-2
9.2	TPU Components.....	9-2
9.2.1	Time Bases.....	9-2
9.2.2	Timer Channels.....	9-2
9.2.3	Scheduler.....	9-3
9.2.4	Microengine.....	9-3
9.2.5	Host Interface.....	9-3
9.2.6	Parameter RAM.....	9-3
9.3	TPU Operation.....	9-4
9.3.1	Event Timing.....	9-4
9.3.2	Channel Orthogonality.....	9-4
9.3.3	Interchannel Communication.....	9-5
9.3.4	Programmable Channel Service Priority.....	9-5
9.3.5	Coherency.....	9-5
9.3.6	Emulation Support.....	9-5
9.3.7	TPU Interrupts.....	9-6
9.4	Time Functions.....	9-7
9.4.1	Discrete Input/Output.....	9-7
9.4.2	Input Capture/Input Transition Counter.....	9-7
9.4.3	Output Compare.....	9-7
9.4.4	Pulse-Width Modulation.....	9-8
9.4.5	Synchronized Pulse-Width Modulation.....	9-8
9.4.6	Period Measurement with Additional Transition Detect.....	9-8
9.4.7	Period Measurement with Missing Transition Detect.....	9-9
9.4.8	Position-Synchronized Pulse Generator.....	9-9

TABLE OF CONTENTS (Continued)

Paragraph	Title	Page
9.4.9	Stepper Motor.....	9-9
9.4.10	Period/Pulse-Width Accumulator.....	9-10
9.5	Host Interface Registers.....	9-11
9.5.1	System Configuration Registers.....	9-11
9.5.1.1	Prescaler Control for TCR1.....	9-11
9.5.1.2	Prescaler Control for TCR2.....	9-12
9.5.1.3	Emulation Control.....	9-13
9.5.1.4	Low-Power Stop Control.....	9-13
9.5.2	Channel Control Registers.....	9-13
9.5.2.1	Channel Interrupt Enable and Status Registers.....	9-13
9.5.2.2	Channel Function Select Registers.....	9-14
9.5.2.3	Host Sequence Registers.....	9-14
9.5.2.4	Host Service Registers.....	9-14
9.5.2.5	Channel Priority Registers.....	9-16
9.5.3	Development Support and Test Registers.....	9-16

Section 10 Standby RAM with TPU Emulation

10.1	General.....	10-1
10.2	TPURAM Register Block.....	10-1
10.3	TPURAM Array Address Mapping.....	10-2
10.4	SRAM Array Address Space Type.....	10-2
10.5	Normal Access.....	10-2
10.6	Standby and Low-Power Stop Operation.....	10-2
10.7	TPU ROM Emulation.....	10-4
10.8	Reset.....	10-4

Section 11 Masked ROM Module

11.1	General.....	11-1
11.2	MRM Register Block.....	11-1
11.3	MRM Array Address Mapping.....	11-2
11.4	MRM Array Address Space Type.....	11-2
11.5	Normal Access.....	11-3
11.6	Emulation Mode Operation.....	11-3
11.7	Low-Power Stop Operation.....	11-3
11.8	ROM Signature.....	11-3
11.9	Reset.....	11-3

TABLE OF CONTENTS (Continued)

Paragraph Title Page

Appendix A Electrical Characteristics

Appendix B Mechanical Data and Ordering Information

B.1	Pin Assignments and Package Dimensions.....	B-1
B.2	Standard Part Ordering Information.....	B-1
B.3	Custom MCU Ordering Information.....	B-1
B.4	Items Needed to Make a Custom MCU Order.....	B-8
B.5	How to Obtain an Ordering Form.....	B-8
B.6	Information Necessary To Complete Ordering Form.....	B-8
B.7	Application Program Media.....	B-10
B.8	Application Program Labeling.....	B-10
B.9	Application Program Format.....	B-10
B.10	ROM Program Verification.....	B-10
B.11	Submitting an MCU ROM Pattern.....	B-11
B.12	Technical Information.....	B-12
B.12.1	Clock Reference Option.....	B-12
B.12.2	Masked ROM Module Options.....	B-12
B.12.2.1	Base Address.....	B-12
B.12.2.2	Configuration Options.....	B-12
B.12.2.3	Bootstrap Options.....	B-13
B.13	Time Processor Unit Options.....	B-14

Appendix C Development Support

C.1	M68HC16Y1MEVB Modular Evaluation Board.....	C-1
C.2	M68HC16Y1MPB Modular Development System.....	C-2

Appendix D Register Summary

D.1	Central Processing Unit.....	D-1
D.1.1	CPU16 Register Model.....	D-2
D.1.2	CCR — Condition Code Register.....	D-3
D.2	Analog-to-Digital Converter Module.....	D-4
D.2.1	ADCMCR — ADC Module Configuration Register.....	D-5
D.2.2	ADCTST — ADC Test Register.....	D-5
D.2.3	PORTADA — Port ADA Data Register.....	D-5

TABLE OF CONTENTS (Continued)

Paragraph	Title	Page
D.2.4	ADCTL0 — A/D Control Register 0.....	D-6
D.2.5	ADCTL1 — ADC Control Register 1.....	D-7
D.2.6	ADSTAT — ADC Status Register	D-9
D.2.7	RSLT[0:7] — ADC Result Registers.....	D-10
D.2.7.1	RJURR — Unsigned Right-Justified Result Registers.....	D-10
D.2.7.2	LJSRR — Signed Left-Justified Result Registers	D-10
D.2.7.3	LJURR — Unsigned Left-Justified Result Registers.....	D-10
D.3	Masked ROM Module.....	D-11
D.3.1	Masked ROM Control Registers.....	D-11
D.3.1.1	MRMCR — Masked ROM Module Configuration Register..	D-11
D.3.1.2	ROMBAH — Array Base Address Register High.....	D-13
D.3.1.3	ROMBAL — Array Base Address Register Low	D-13
D.3.1.4	RSIGHI — ROM Signature High Register	D-13
D.3.1.5	RSIGLO — ROM Signature Low Register.....	D-13
D.3.1.6	ROMBS0 — ROM Bootstrap Word 0	D-13
D.3.1.7	ROMBS1 — ROM Bootstrap Word 1	D-13
D.3.1.8	ROMBS2 — ROM Bootstrap Word 2	D-13
D.3.1.9	ROMBS3 — ROM Bootstrap Word 3.....	D-13
D.4	General-Purpose Timer.....	D-14
D.4.1	GPTMCR — GPT Module Configuration Register	D-14
D.4.2	GPTMTR — GPT Module Test Register (Reserved).....	D-15
D.4.3	ICR — GPT Interrupt Configuration Register	D-15
D.4.4	DDRGP — Port GP Data Direction Register	D-16
	PORTGP — Port GP Data Register.....	D-16
D.4.5	OC1M — OC1 Action Mask Register.....	D-16
	OC1D — OC1 Action Data Register.....	D-16
D.4.6	TCNT — Timer Counter Register	D-16
D.4.7	PACTL — Pulse Accumulator Control Register	D-17
	PACNT — Pulse Accumulator Counter	D-17
D.4.8	TIC[1:3] — Input Capture Registers 1-3	D-18
D.4.9	TOC[1:4] — Output Compare Registers 1-4	D-18
D.4.10	TI4/O5 — Input Capture 4/Output Compare 5 Register	D-18
D.4.11	TCTL1/TCTL2 — Timer Control Registers 1 and 2.....	D-18
D.4.12	TMSK1/TMSK2 — Timer Interrupt Mask Registers 1 and 2.....	D-19
D.4.13	TFLG1/TFLG2 — Timer Interrupt Flag Registers 1 and 2.....	D-20
D.4.14	CFORC — Compare Force	D-21
	PWMC — PWM Control.....	D-21
D.4.15	PWMA/PWMB — PWM Registers A/B	D-22
D.4.16	PWCNT — PWM Count Register	D-22
D.4.17	PWMBUFA — PWM Buffer Register A	D-22

TABLE OF CONTENTS (Continued)

Paragraph	Title	Page
	PWMBUFB — PWM Buffer Register B	D-22
D.4.18	PRESCL — GPT Prescaler	D-22
D.5	Single-Chip Integration Module.....	D-23
D.5.1	SCIMCR — SCIM Configuration Register	D-24
D.5.2	SCIMTR — SCIM Test Register	D-26
D.5.3	SYNCR — Clock Synthesizer Control Register	D-26
D.5.4	RSR — Reset Status Register	D-27
D.5.5	SCIMTRE — Module Test Register E	D-28
D.5.6	PORTA — Port A Data Register.....	D-28
	PORTB — Port B Data Register.....	D-28
D.5.7	PORTG — Port G Data Register.....	D-28
	PORTH — Port H Data Register.....	D-28
D.5.8	DDRG — Port G Data Direction Register.....	D-28
	DDRH — Port H Data Direction Register.....	D-28
D.5.9	PORTE0/PORTE1 — Port E Data Register.....	D-28
D.5.10	DDRAB — Port A/B Data Direction Register.....	D-29
	DDRE — Port E Data Direction Register	D-29
D.5.11	PEPAR — Port E Pin Assignment Register.....	D-29
D.5.12	PORTF0/PORTF1— Port F Data Register.....	D-30
D.5.13	DDRF — Port F Data Direction Register.....	D-30
D.5.14	PFPAR — Port F Pin Assignment Register.....	D-31
D.5.15	SYPCR — System Protection Control Register	D-31
D.5.16	PICR — Periodic Interrupt Control Register.....	D-33
D.5.17	PITR — Periodic Interrupt Timer Register	D-33
D.5.18	SWSR — Software Service Register.....	D-34
D.5.19	PORTFE — Port F Edge-Detect Flag Register.....	D-34
D.5.20	PFIVR — Port F Edge-Detect Interrupt Vector Register	D-34
D.5.21	PFLVR — Port F Edge-Detect Interrupt Level Register.....	D-34
D.5.22	TSTMSRA — Master Shift Register A.....	D-34
D.5.23	TSTMSRB — Master Shift Register B.....	D-35
D.5.24	TSTSC — Test Module Shift Count	D-35
D.5.25	TSTRC — Test Module Repetition Count.....	D-35
D.5.26	CREG — Test Submodule Control Register	D-35
D.5.27	DREG — Distributed Register.....	D-35
D.5.28	PORTC — Port C Data Register.....	D-35
D.5.29	CSPAR0 — Chip Select Pin Assignment Register 0	D-35
D.5.30	CSPAR1 — Chip Select Pin Assignment Register 1	D-36
D.5.31	CSBARBT — Chip Select Base Address Register Boot ROM	D-37
D.5.32	CSBAR[0:10] — Chip Select Base Address Registers	D-38
D.5.33	CSORBT — Chip Select Option Register Boot ROM	D-38

TABLE OF CONTENTS (Continued)

Paragraph	Title	Page
D.5.34	CSOR[0:10] — Chip Select Option Registers.....	D-38
D.6	Standby RAM with TPU Emulation.....	D-40
D.6.1	TRAMMCR — TPU RAM Module Configuration Register.....	D-40
D.6.2	TRAMTST — TPURAM Test Register.....	D-41
D.6.3	TRAMBAR — TPURAM Array Base Address Register High	D-41
D.7	Multichannel Communications Interface.....	D-42
D.7.1	MMCR — MCCI Configuration Register	D-43
D.7.2	MTEST — MCCI Test	D-43
D.7.3	ILSCI — SCI Interrupt Request Level	D-44
	MIVR — MCCI Interrupt Vector	D-44
D.7.4	ILSPI — SPI Interrupt Level	D-44
D.7.5	PORTMC — MCCI Port Data Register.....	D-44
D.7.6	PORTMCP — MCCI Port Pin State Register.....	D-45
D.7.7	PMCPAR — MCCI Pin Assignment Register.....	D-45
D.7.8	DDRMC — MCCI Data Direction Register.....	D-45
D.7.9	SCCR0A, SCCR0B — SCI Control Register 0	D-46
D.7.10	SCCR1A, SCCR1B — SCI Control Register 1.....	D-46
D.7.11	SCSRA, SCSR B — SCI Status Register	D-49
D.7.12	SCDRA, SCDRB — SCI Data Register	D-51
D.7.13	SPCR — SPI Control Register	D-51
D.7.14	SPSR — SPI Status Register.....	D-53
D.7.15	SPDR — SPI Data Register.....	D-53
D.8	Time Processor Unit.....	D-54
D.8.1	TMCR — TPU Module Configuration Register.....	D-54
D.8.2	TCR — Test Configuration Register	D-55
D.8.3	DSCR — Development Support Control Register.....	D-56
D.8.4	DSSR — Development Support Status Register	D-57
D.8.5	TICR — TPU Interrupt Configuration Register.....	D-57
D.8.6	CIER — Channel Interrupt Enable Register.....	D-58
D.8.7	CFSR0–CFSR3 — Channel Function Select Registers.....	D-58
D.8.8	HSQR0 — Host Sequence Register 0.....	D-58
D.8.9	HSQR1 — Host Sequence Register 1.....	D-58
D.8.10	HSRR0 — Host Service Request Register 0	D-58
D.8.11	HSRR1 — Host Service Request Register 1	D-59
D.8.12	CPR0 — Channel Priority Register 0	D-59
D.8.13	CPR1 — Channel Priority Register 1	D-59
D.8.14	CISR — Channel Interrupt Status Register.....	D-59
D.8.15	LR — Link Register.....	D-61
D.8.16	SGLR — Service Grant Latch Register.....	D-61
D.8.17	DCNR — Decoded Channel Number Register.....	D-61

LIST OF ILLUSTRATIONS

Figure	Title	Page
3-1	MC68HC16Y1 Block Diagram	3-3
3-2	MC68HC16Y1 Pin Assignment for 160-Pin Package.....	3-4
3-3	Internal Register Addresses.....	3-12
3-4	Pseudolinear Addressing/Combined Program and Data Spaces	3-13
3-5	Pseudolinear Addressing/Separated Program and Data Spaces.....	3-14
4-1	Single-Chip Integration Module Block Diagram.....	4-2
4-2	System Configuration and Protection.....	4-4
4-3	Periodic Interrupt Timer and Software Watchdog Timer	4-9
4-4	System Clock Block Diagram.....	4-12
4-5	MC68HC16Y1 Basic System	4-21
4-6	Operand Byte Order.....	4-26
4-7	Word Read Cycle.....	4-30
4-8	Write Cycle.....	4-31
4-9	Fast-Termination Timing.....	4-33
4-10	CPU Space Address Encoding.....	4-34
4-11	Breakpoint Operation.....	4-35
4-12	LPSTOP Interrupt Mask Level.....	4-36
4-13	Bus Arbitration.....	4-41
4-14	Power-On Reset Timing.....	4-54
4-15	Basic M68HC16 System	4-62
4-16	Chip-Select Circuit Block Diagram.....	4-63
4-17	CPU Space Encoding for Interrupt Acknowledge	4-70
5-1	CPU16 Register Model.....	5-2
5-2	Condition Code Register.....	5-4
5-3	Data Types and Memory Organization.....	5-9
5-4	Instruction Execution Model.....	5-34
5-5	Exception Stack Frame Format.....	5-38
5-6	BDM Serial I/O Block Diagram.....	5-46
5-7	BDM Connector Pinout.....	5-47
6-1	ADC Block Diagram	6-2
6-2	8-Bit Conversion Timing.....	6-15
6-3	10-Bit Conversion Timing.....	6-15
7-1	MCCI Block Diagram.....	7-1

LIST OF ILLUSTRATIONS (Continued)

Paragraph	Title	Page
8-1	GPT Block Diagram.....	8-2
8-2	Prescaler Block Diagram.....	8-10
8-3	Capture/Compare Unit Block Diagram.....	8-12
8-4	Input Capture Timing Example.....	8-14
8-5	Pulse Accumulator Block Diagram.....	8-17
8-6	PWM Block Diagram	8-18
9-1	TPU Block Diagram.....	9-1
9-2	TCR1 Prescaler Control.....	9-11
9-3	TCR2 Prescaler Control.....	9-12
A-1	CLKOUT Output Timing Diagram	A-16
A-2	External Clock Input Timing Diagram	A-16
A-3	ECLK Output Timing Diagram.....	A-16
A-4	Read Cycle Timing Diagram.....	A-18
A-5	Write Cycle Timing Diagram.....	A-20
A-6	Show Cycle Timing Diagram.....	A-22
A-7	Reset and Mode Select Timing Diagram	A-24
A-8	Bus Arbitration Timing Diagram — Active Bus Case	A-26
A-9	Bus Arbitration Timing Diagram — Idle Bus Case.....	A-28
A-10	Fast Termination Read Cycle Timing Diagram	A-30
A-11	Fast Termination Write Cycle Timing Diagram.....	A-32
A-12	ECLK Timing Diagram.....	A-34
A-13	Chip Select Timing Diagram	A-36
A-14	BDM Timing Diagram — Serial Communication	A-38
A-15	BDM Timing Diagram — Freeze Assertion.....	A-38
A-16	SPI Timing Master, CPHA = 0	A-40
A-17	SPI Timing Master, CPHA = 1	A-40
A-18	SPI Timing Slave, CPHA = 0.....	A-42
A-19	SPI Timing Slave, CPHA = 1	A-42
B-1	160-Pin Plastic Quad Flat Package Pin Assignments	B-2
B-2	160-Pin QFP Package Dimensions.....	B-3
B-3	160-Pin QFP (with Molded Carrier Ring) Package Dimensions.....	B-4

LIST OF TABLES

Table	Title	Page
3-1	MC68HC16Y1 Driver Types.....	3-5
3-2	MC68HC16Y1 Pin Characteristics.....	3-5
3-3	MC68HC16Y1 Power Connections.....	3-7
3-4	MC68HC16Y1 Signal Characteristics.....	3-7
3-5	MC68HC16Y1 Signal Function.....	3-9
3-6	Basic Configuration Options.....	3-15
3-7	Bus and Port Configuration Options.....	3-16
4-1	Show Cycle Enable Bits.....	4-6
4-2	Bus Monitor Period.....	4-6
4-3	MODCLK Pin and SWP Bit During Reset.....	4-8
4-4	Software Watchdog Ratio.....	4-8
4-5	MODCLK Pin and PTP Bit at Reset.....	4-10
4-6	Periodic Interrupt Priority.....	4-10
4-7	Clock Control Multipliers.....	4-15
4-8	System Frequencies from 4.194-MHz Reference.....	4-17
4-9	Clock Control.....	4-20
4-10	Size Signal Encoding.....	4-23
4-11	Address Space Encoding.....	4-24
4-12	Effect of DSACK Signals.....	4-25
4-13	Operand Alignment.....	4-27
4-14	DSACK, BERR, and HALT Assertion Results.....	4-37
4-15	Reset Source Summary.....	4-43
4-16	Basic Configuration Options.....	4-44
4-17	Bus and I/O Port Pin Functions.....	4-45
4-18	16-Bit Data Bus Mode Reset Configuration.....	4-47
4-19	8-Bit Expanded Mode Reset Configuration.....	4-48
4-20	Single-Chip Mode Reset Configuration.....	4-49
4-21	Module Pin Function.....	4-51
4-22	Pin Reset States.....	4-52
4-23	Chip-Select Pin Functions.....	4-65
4-24	Pin Assignment Field Encoding.....	4-65
4-25	Block Size Encoding.....	4-66
4-26	Option Register Function Summary.....	4-67
4-27	CSBOOT Base and Option Register Reset Values.....	4-71

LIST OF TABLES (Continued)

Table	Title	Page
5-1	Addressing Modes.....	5-10
5-2	CPU16 Implementation of M68HC11 CPU Instructions.....	5-31
5-3	Basic Instruction Formats.....	5-33
5-4	Exception Vector Table.....	5-38
5-5	IPIPE0/IPIPE1 Encoding.....	5-41
5-6	Command Summary.....	5-45
6-1	FRZ Field Selection.....	6-5
6-2	Multiplexer Channels.....	6-6
6-3	Prescaler Output.....	6-8
6-4	STS Field Selection.....	6-8
6-5	ADC Conversion Modes.....	6-10
6-6	Single-Channel Conversions.....	6-12
6-7	Multiple-Channel Conversions.....	6-13
7-1	MCCI Pin Assignments.....	7-4
7-2	SCI Pin Function.....	7-5
7-3	Data Frame Formats.....	7-7
7-4	SPI Pin Function.....	7-12
8-1	GPT Status Flags.....	8-5
8-2	GPT Interrupt Sources.....	8-6
8-3	PWM Frequency Range.....	8-19
9-1	TCR1 Prescaler Control.....	9-12
9-2	TCR2 Prescaler Control.....	9-13
9-3	Host Sequence Code/Host Service Request Code.....	9-15
9-4	Channel Priority Encodings.....	9-16
A-1	Maximum Ratings.....	A-2
A-2	Thermal Characteristics.....	A-3
A-3	Clock Control Timing.....	A-4
A-4	DC Characteristics.....	A-5
A-5	AC Timing.....	A-7
A-6	Background Debugging Mode Timing.....	A-10
A-7	ECLK Bus Timing.....	A-10
A-8	MCCI SPI Timing.....	A-11
A-9	ADC Maximum Ratings.....	A-12
A-10	ADC DC Electrical Characteristics (Operating).....	A-13
A-11	ADC AC Characteristics (Operating).....	A-14
A-12	ADC Conversion Characteristics (Operating).....	A-14

LIST OF TABLES (Continued)

Table	Title	Page
B-1	Ordering Information	B-7
B-2	Custom MCU Ordering Information Checklist	B-9
C-1	MC68HC16Y1 Development Tools.....	C-1
D-1	MC68HC16Y1 Module Address Map	D-1
D-2	ADC Module Address Map	D-4
D-3	MRM Control Register Address Map.....	D-11
D-4	General-Purpose Timer Address Map.....	D-14
D-5	SCIM Address Map	D-23
D-6	Standby RAM Control Register Address Map	D-40
D-7	MCCI Address Map	D-42
D-8	TPU Address Map.....	D-54
D-9	MC68HC16Y1 Module Address Map	D-62
D-10	Register Bit and Field Mnemonics.....	D-69

SECTION 1 INTRODUCTION

The MC68HC16Y1 is a high-speed 16-bit control unit that is upwardly code compatible with M68HC11 controllers. It is a member of the M68HC16 Family of modular microcontrollers.

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

The MC68HC16Y1 incorporates a 16-bit central processing unit (CPU16), a single-chip integration module (SCIM), an 8/10-bit analog-to-digital converter (ADC), a multichannel communication interface (MCCI), a general-purpose timer (GPT), a time processing unit (TPU), a 2-Kbyte standby RAM module with TPU emulation capability (TPURAM), and a 48-Kbyte masked ROM module (MRM). All of these modules are interconnected by the intermodule bus (IMB).

The MC68HC16Y1 can either synthesize an internal clock signal from an external reference, or use an external clock input directly. Operation with a 4.194-MHz reference frequency is standard, but operation with a 32.768-kHz reference is available as an option. Contact your Motorola representative for more information. System hardware and software allow changes in clock rate during operation. Because the MC68HC16Y1 is a fully-static device, 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 MC68HC16Y1 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.

Documentation for the Modular Microcontroller Family follows the modular construction of the devices in the product line. Each device has a comprehensive user's manual that provides sufficient information for normal operation of the device. The user's manual is supplemented by module reference manuals that provide detailed information about module operation and applications. Refer to Motorola publication *Advanced Microcontroller Unit (AMCU) Literature* (BR1116/D) for a list of additional documentation for this device.

SECTION 2 NOMENCLATURE

The following nomenclature is used throughout the manual. Nomenclature used only in certain sections, such as register bit mnemonics, is defined in those sections.

2.1 Symbols and Operators

- + — 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

2.2 CPU16 Registers

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

2.3 Pin and Signal Mnemonics

AN[7:0]	—	ADC Analog Inputs
ADDR[23:0]	—	Address Bus
\overline{AS}	—	Address Strobe
\overline{AVEC}	—	Autovector Request
\overline{BERR}	—	Bus Error
\overline{BG}	—	Bus Grant
\overline{BGACK}	—	Bus Grant Acknowledge
\overline{BKPT}	—	Breakpoint Request
\overline{BR}	—	Bus Request
CLKOUT	—	System Clock Output
$\overline{CS[10:5]}$ $\overline{CS3}; \overline{CS0}$	—	Chip Selects
\overline{CSBOOT}	—	Boot ROM Chip Select
\overline{CSE}	—	External PRU Chip Select
\overline{CSM}	—	External ROM Chip Select
DATA[15:0]	—	Data Bus
\overline{DS}	—	Data Strobe
$\overline{DSACK[1:0]}$	—	Data and Size Acknowledge
DSCLK	—	Development Serial Clock
ECLK	—	6800 Bus Clock Output
DSI	—	Development Serial Input
DSO	—	Development Serial Output
EXTAL	—	External Clock/Reference Connection
FC[2:0]	—	Function Code Output
FREEZE	—	Freeze Request
\overline{HALT}	—	Halt Request
IC[4:1]	—	GPT Input Capture
IPIPE0/IPIPE1	—	Instruction Pipeline MUX
$\overline{IRQ[7:0]}$	—	Interrupt Request
MISO	—	SPI Master In Slave Out
MODCLK	—	System Clock Mode Select
MOSI	—	SPI Master Out Slave In
OC[5:1]	—	GPT Output Compare
PA[7:0]	—	SCIM I/O Port A
PADA[7:0]	—	ADC Port A
PAI	—	GPT Pulse Accumulator Input
PB[7:0]	—	SCIM I/O Port B

PC[6:0]	—	SCIM Port C
PCLK	—	GPT Pulse Accumulator Clock
PCS[3:0]	—	GPT Peripheral Chip Selects
PE[7:0]	—	SCIM I/O Port E
PF[7:0]	—	SCIM I/O Port F
PG[7:0]	—	SCIM I/O Port G
PGP[7:0]	—	GPT I/O Port
PH[7:0]	—	SCIM I/O Port H
PMC[7:0]	—	MCCI I/O Port
PWMA, PWMB	—	GPT Pulse Width Modulator Output
QUOT	—	Quotient Out
\overline{RW}	—	Read/Write
\overline{RESET}	—	Reset
RXDA	—	SCI A Receive Data
RXDB	—	SCI B Receive Data
SCK	—	SPI Serial Clock
SIZ[1:0]	—	Size
\overline{SS}	—	SPI Slave Select
TPUCH[0:15]	—	TPU Channels
\overline{TSC}	—	SCIM Three-State Control
\overline{TSTME}	—	SCIM Test Mode Enable
TXDA	—	SCI A Transmit Data
TXDB	—	SCI B Transmit Data
V _{DDA} /V _{SSA}	—	ADC Power
V _{DDSYN}	—	Clock Synthesizer Power
V _{RH} /V _{RL}	—	ADC Reference Voltage
V _{SSE} /V _{DDE}	—	External Peripheral Power (Source and Drain)
V _{SSI} /V _{DDI}	—	Internal Module Power (Source and Drain)
V _{STBY}	—	Standby RAM Power
XFC	—	Clock Filter Capacitor Connection
XTAL	—	Clock Crystal Oscillator Connection

2.4 Register Mnemonics

ADCMCR	—	ADC Module Configuration Register
ADCTL[0:1]	—	ADC Control Registers
ADSTAT	—	ADC Status Register
ADTEST	—	ADC Test Register
CFORC	—	GPT Compare Force Register
CFSR[0:3]	—	TPU Channel Function Select Registers
CIER	—	TPU Channel Interrupt Enable Register
CISR	—	TPU Channel Interrupt Status Register
CPR[0:1]	—	TPU Channel Priority Registers
CREG	—	SCIM Test Submodule Control Register
CSBARBT	—	SCIM Chip Select Base Address Register Boot ROM
CSBAR[0:10]	—	SCIM Chip Select Base Address Registers
CSORBT	—	SCIM Chip Select Option Register Boot ROM
CSPAR[0:1]	—	SCIM Chip Select Pin Assignment Registers
DDRAB	—	SCIM Port A and B Data Direction Register
DDRE	—	SCIM Port E Data Direction Register
DDRF	—	SCIM Port F Data Direction Register
DDRG	—	SCIM Port G Data Direction Register
DDRGP	—	GPT Port GP Data Direction Register
DDRH	—	SCIM Port H Data Direction Register
DDRMC	—	MCCI PORT MC Data Direction Register
DNCR	—	TPU Decoded Channel Number Register
DREG	—	SCIM Distributed Register
DSCR	—	TPU Development Support Control Register
DSSR	—	TPU Development Support Status Register
GPTMCR	—	GPT Module Configuration Register
GPTMTR	—	GPT Module Test Register
HSQR[0:1]	—	TPU Host Sequence Registers
ICR	—	GPT Interrupt Configuration Register
ILSCI	—	MCCI SCI Interrupt Level Register
ILSPI	—	MCCI SPI Interrupt Level Register
LR	—	TPU LINK Register
MIVR	—	MCCI Interrupt Vector Register
MMCR	—	MCCI Configuration Register
MRMCR	—	MRM Configuration Register
MTEST	—	MCCI Test Register
OC1D	—	GPT OC1 Action Data Register
OC1M	—	GPT OC1 Action Mask Register
PACNT	—	GPT Pulse Accumulator Counter
PACTL	—	GPT Pulse Accumulator Control Register
PEPAR	—	SCIM Port E Pin Assignment Register
PFIVR	—	SCIM Port F Interrupt Vector Register
PFLVR	—	SCIM Port F Interrupt Level Register

PFPAR	—	SCIM Port F Pin Assignment Register
PICR	—	SCIM Periodic Interrupt Control Register
PITR	—	SCIM Periodic Interrupt Timer Register
PMCPAR	—	MCCI Port MC Pin Assignment Register
PORTA	—	SCIM Port A Data Register
PORTADA	—	ADC Port ADA Data Register
PORTB	—	SCIM Port B Data Register
PORTC	—	SCIM Port C Data Register
PORTE	—	SCIM Port E Data Register
PORTF	—	SCIM Port F Data Register
PORTFE	—	SCIM Port F Edge Detection Register
PORTG	—	SCIM Port G Data Register
PORTGP	—	GPT Port GP Data Register
PORTH	—	SCIM Port H Data Register
PORTMC	—	MCCI Port MC Data Register
PORTMCP	—	MCCI Port MC Pin State Register
PRESCL	—	GPT Prescaler Register
PWMA	—	GPT PWM Control Register A
PWMB	—	GPT PWM Control Register B
PWMBUFA	—	GPT PWM Buffer Register A
PWMBUFB	—	GPT PWM Buffer Register B
PWMC	—	GPT PWM Control Register C
PWMCNT	—	GPT PWM Count Register
ROMBAH	—	ROM Array Base Address Register High
ROMBAL	—	ROM Array Base Address Register Low
ROMBS[0:3]	—	ROM Bootstrap Words
RSIGHI	—	ROM Signature High
RSIGLO	—	ROM Signature Low
RSLT[0:7]	—	ADC Result Registers
RSR	—	SCIM Reset Status Register
SCCR[0:1]A	—	MCCI SCI A Control Registers
SCCR[0:1]B	—	MCCI SCI B Control Registers
SCDRA	—	MCCI SCI A Data Register
SCDRB	—	MCCI SCI B Data Register
SCIMCR	—	SCIM Configuration Register
SCIMTR	—	SCIM Test Register
SCIMTRE	—	SCIM Test Register (ECLK)
SCSRA	—	MCCI SCI A Status Register
SCSRB	—	MCCI SCI B Status Register
SGLR	—	TPU Service Grant Latch Register
SPCR	—	MCCI SPI Control Register
SPDR	—	MCCI SPI Data Register
SPSR	—	MCCI SPI Status Register
SWSR	—	SCIM Software Watchdog Service Register

SYNCR	—	SCIM Clock Synthesizer Control Register
SYPCR	—	SCIM System Protection Control Register
TCNT	—	GPT Timer Counter Register
TCR	—	TPU Module Test Configuration Register
TCTL[1:2]	—	GPT Timer Control Registers
TFLG[1:2]	—	GPT Timer Interrupt Flag Registers
TI4/O5	—	GPT Input Capture 4/Output Compare 5 Register
TICR	—	TPU Interrupt Control Register
TIC[1:3]	—	GPT Input Capture Registers
TMSK[1:2]	—	GPT Timer Interrupt Mask Registers
TOC[1:4]	—	GPT Output Compare Registers
TPUMCR	—	TPU Module Configuration Register
TPURAM[0:15]	—	GPT TPU Channel Parameter RAM
TRAMBAH	—	RAM Array Base Address Register High
TRAMBAL	—	RAM Array Base Address Register Low
TRAMMCR	—	RAM Module Configuration Register
TRAMTST	—	RAM Test Register
TSTMSRA	—	SCIM Master Shift Register A
TSTMSRB	—	SCIM Master Shift Register B
TSTRC	—	SCIM Test Module Repetition Count
TSTSC	—	SCIM Test Module Shift Count

2.5 Conventions

Logic level one is the voltage that corresponds to a Boolean true (1) state.

Logic level zero is the voltage that corresponds to a Boolean false (0) state.

Set refers specifically to establishing logic level one on a bit or bits.

Clear refers specifically to establishing logic level zero on a bit or bits.

Asserted means that a signal is in active logic state. An active low signal changes from logic level one to logic level zero when asserted, and an active high signal changes from logic level zero to logic level one.

Negated means that an asserted signal changes logic state. An active low signal changes from logic level zero to logic level one when negated, and an active high signal changes from logic level one to logic level zero.

A specific mnemonic within a range is referred to by mnemonic and number: A15 is bit 15 of Accumulator A; ADDR7 is line 7 of the address bus; CSOR0 is chip-select option register 0. **A range of mnemonics** is referred to by mnemonic and the numbers that define the range: AM[35:30] are bits 35 to 30 of Accumulator M; CSOR[0:5] are the first six option registers

Parentheses are used to indicate the content of a register or memory location, rather than the register or memory location itself. (A) is the content of Accumulator A. (M : M + 1) is the content of the word at address M.

LSB means least significant bit or bits. **MSB** means most significant bit or bits. References to low and high bytes are spelled out.

LSW means least significant word or words. **MSW** means most significant word or words.

ADDR is the address bus. ADDR[7:0] are the eight LSB of the address bus.

DATA is the data bus. DATA[15:8] are the eight MSB of the data bus.

SECTION 3 OVERVIEW

This section contains information about the entire MC68HC16Y1 modular microcontroller. It lists the features of each module, shows device functional divisions and pinouts, summarizes signal and pin functions, discusses the intermodule bus, and provides system memory maps. Timing and electrical specifications for the entire microcontroller and for individual modules are provided in **APPENDIX A ELECTRICAL CHARACTERISTICS**. Comprehensive module register descriptions and memory maps are provided in **APPENDIX D REGISTER SUMMARY**.

3.1 MC68HC16Y1 Features

The following paragraphs highlight capabilities of each of the microcontroller modules. Each module is discussed separately in a subsequent section of this reference manual.

3.1.1 Single-Chip Integration Module

- Single Chip or Expanded Modes of Operation
- External Bus Support in Expanded Mode
- Eight General-Purpose Chip-Select Outputs
- Two Emulation-Mode Chip-Select Outputs
- Boot ROM Chip-Select Output
- System Protection Logic
- Watchdog Timer, Clock Monitor, and Bus Monitor
- Parallel Port Option on Address and Data Bus in Single-Chip Mode
- Phase-Locked Loop (PLL) Clock System

3.1.2 CPU16

- 16-Bit Architecture
- Full Set of 16-Bit Instructions
- Three 16-Bit Index Registers
- Two 16-Bit Accumulators
- Control-Oriented Digital Signal Processing Capability
- One Megabyte of Program Memory and One Megabyte of Data Memory
- High-Level Language Support
- Fast Interrupt Response Time
- Background Debugging Mode

3.1.3 Analog-to-Digital Converter

- Eight Channels, Eight Result Registers
- Eight Automated Modes
- Three Result Alignment Modes

3.1.4 Multichannel Communication Interface

- Dual Serial Communication Interface
- Serial Peripheral Interface

3.1.5 General-Purpose Timer

- 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

3.1.6 Time Processor Unit

- Dedicated Microengine Operating Independently of CPU16
- Sixteen Independently Programmable Channels and Pins
- Two Timer Count Registers with Programmable Prescalers
- Selectable Channel Priority Levels

3.1.7 Standby RAM with TPU Emulation

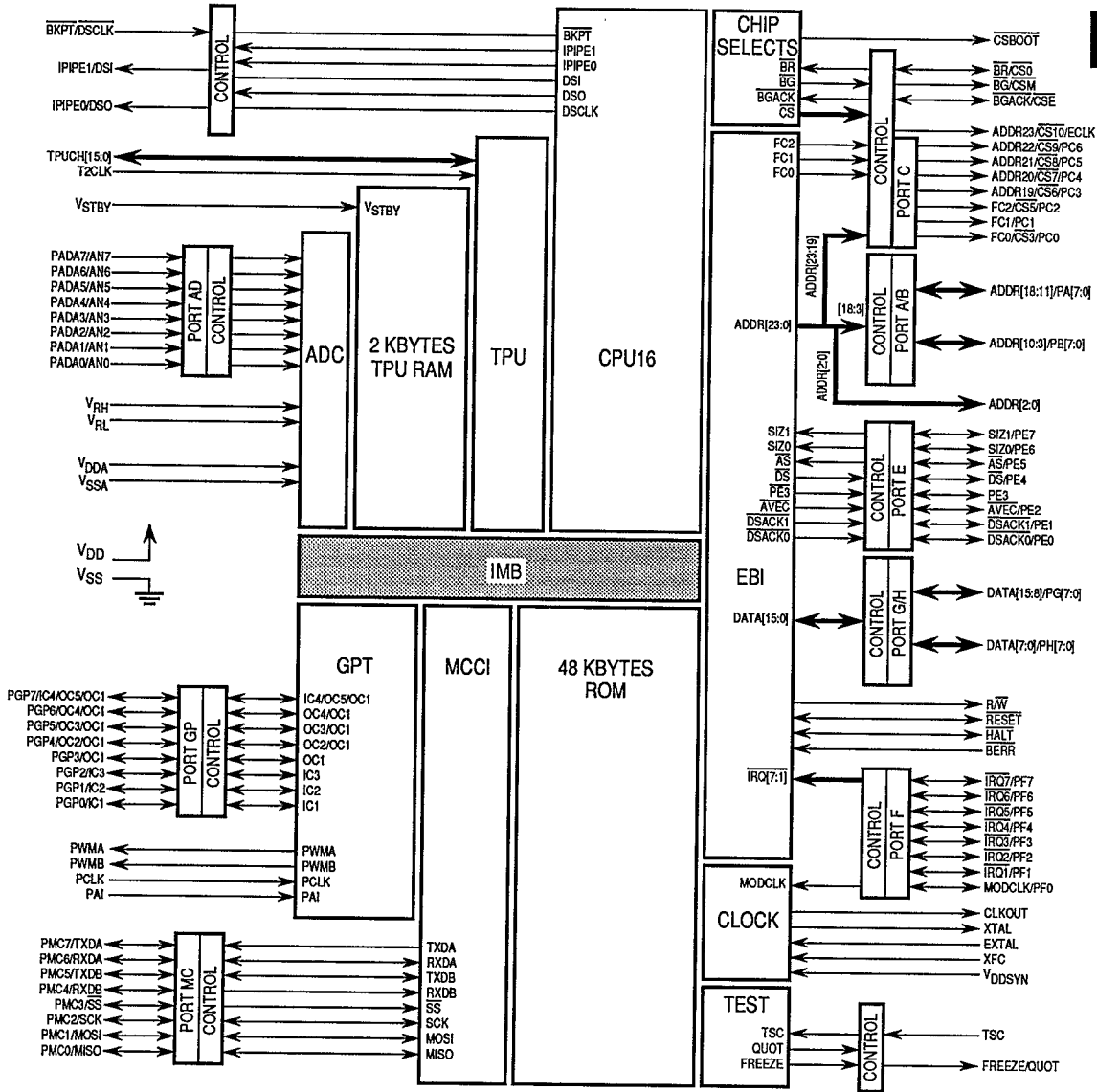
- Two Kbyte Standby RAM
- External Standby Voltage Supply Input
- Power Down Status Flag

3.1.8 Masked ROM

- 48 Kbyte 16-Bit Array
- User-Selectable Default Base Address
- User-Selectable Bootstrap ROM Function
- User-Selectable ROM Verification Code

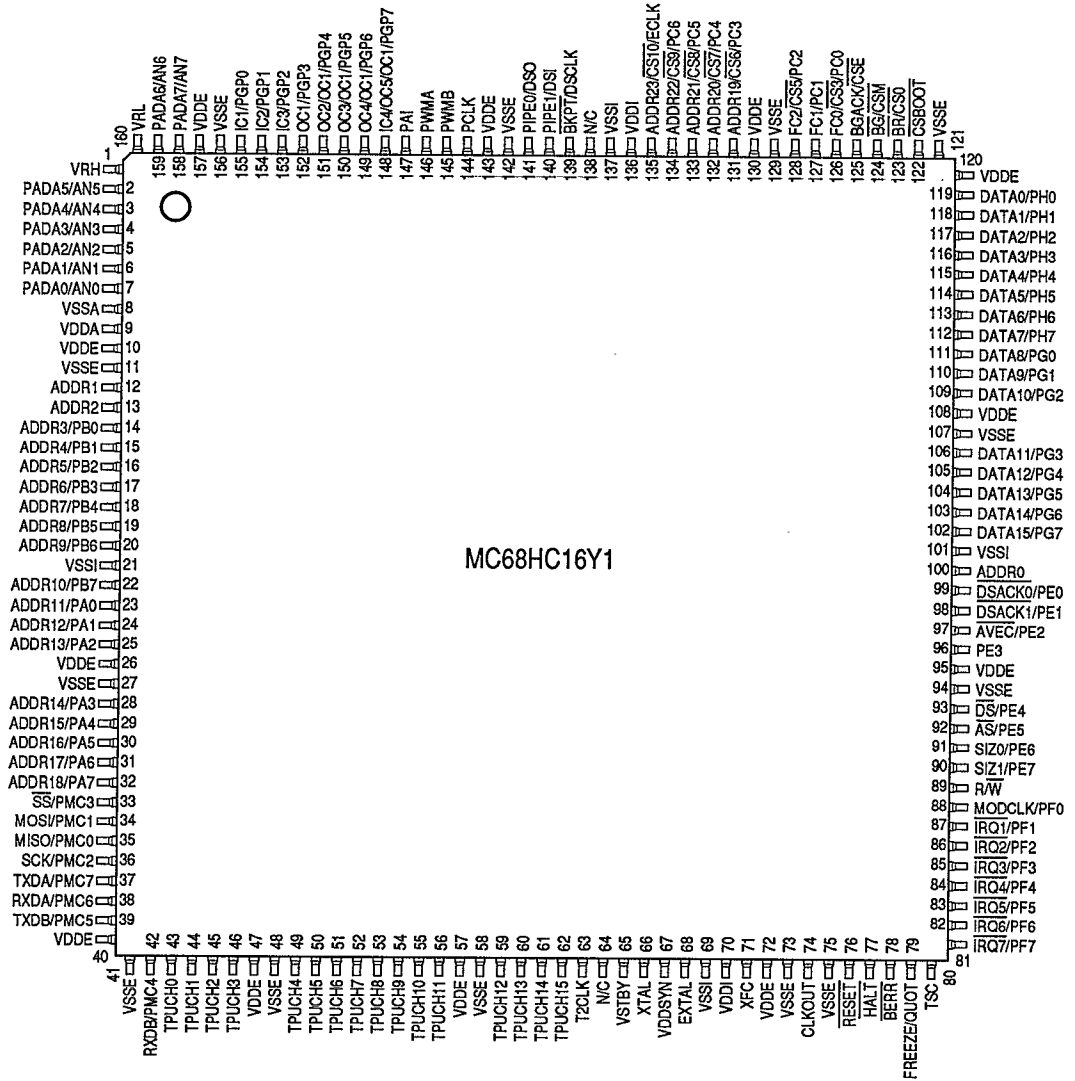
3.2 System Block and Pin Assignment Diagrams

Figure 3–1 is a functional diagram of the MC68HC16Y1. Although the blocks in the diagram represent the physical modules, there is not a one-to-one correspondence between location and size of blocks in the diagram and location and size of integrated-circuit modules. Figure 3–2 shows the pin assignments. Refer to **APPENDIX B MECHANICAL DATA AND ORDERING INFORMATION** for package dimensions. All pin functions and signal names are shown in these drawings. Refer to subsequent paragraphs in this section for pin and signal descriptions.



Y1 BLOCK

Figure 3-1. MC68HC16Y1 Block Diagram



Y1 160-PIN OPP

Figure 3–2. MC68HC16Y1 Pin Assignment for 160-Pin Package

3.3 Pin Descriptions

The following tables summarize the functional characteristics of MC68HC16Y1 pins. Table 3-1 shows types of output drivers. Table 3-2 shows all inputs and outputs. Digital inputs and outputs use CMOS logic levels. An entry in the Discrete I/O column indicates that a pin can also be used for general-purpose input, output, or both. The I/O port designation is given when it applies. Table 3-3 shows characteristics of power pins. Refer to Figure 3-1 for port organization.

Table 3-1. MC68HC16Y1 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 ¹	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

NOTE: Pins with this type of driver can only go into high impedance state under certain conditions. The TSC signal can put all pins with this type of driver in high-impedance state.

Table 3-2. MC68HC16Y1 Pin Characteristics

Pin Mnemonic	Output Driver	Input Synchronized	Input Hysteresis	Discrete I/O	Port Designation
ADDR23/CS10/ECLK	A	Y	N	—	—
ADDR[22:19]/CS[9:6]	A	Y	N	O	C[6:3]
ADDR[18:11]	A	Y	N	I/O	A[7:0]
ADDR[10:3]	A	Y	N	I/O	B[7:0]
ADDR[2:0]	A	Y	N	—	—
AN[7:0] ¹	—	Y*	N	I	AD[7:0]
AS	B	Y	N	I/O	E4
AVEC	B	Y	N	I/O	E2
BERR	—	Y	N	—	—
BGACK/CSE	B	Y	N	—	—
BG/CSM	B	—	—	—	—
BKPT/DSCLK	—	Y	Y	—	—
BR/CS0	B	Y	N	—	—
CLKOUT	A	—	—	—	—
CSBOOT	B	—	—	—	—
DATA[15:8] ¹	AW	Y	N	I/O	G[7:0]
DATA[7:0] ¹	AW	Y	N	I/O	H[7:0]
DS	B	Y	N	I/O	E5
DSACK1	B	Y	N	I/O	E1
DSACK0	B	Y	N	I/O	E0

Table 3–2. MC68HC16Y1 Pin Characteristics (Continued)

Pin Mnemonic	Output Driver	Input Synchronized	Input Hysteresis	Discrete I/O	Port Designation
DSI/PIPE1	A	Y	Y	—	—
DSO/PIPE0	A	Y	Y	—	—
EXTAL ²	—	—	—	—	—
FC2/CS5	A	Y	N	O	C0
FC1	A	Y	N	O	C1
FC0/CS3	A	Y	N	O	C2
FREEZE/QUOT	A	—	—	—	—
HALT	Bo	Y	N	—	—
IC4/OC5	A	Y	Y	I/O	GP4
IC[3:1]	A	Y	Y	I/O	GP[7:5]
IRQ[7:1]	B	Y	Y	I/O	F[7:1]
MISO	Bo	Y	Y	I/O	MC0
MODCLK ¹	B	Y	N	I/O	F0
MOSI	Bo	Y	Y	I/O	MC1
OC[4:1]	A	Y	Y	I/O	GP[3:0]
PAI ³	—	Y	Y	I	—
PCLK ³	—	Y	Y	I	—
SS	Bo	Y	Y	I/O	MC3
PE3	Bo	Y	Y	I/O	E3
PWMA, PWMB ⁴	A	—	—	O	—
R/W	A	—	—	—	—
RESET	Bo	Y	Y	—	—
RXDA	Bo	Y	Y	I/O	MC6
RXDB	Bo	Y	Y	I/O	MC4
SCK	Bo	Y	Y	I/O	MC2
SIZ[1:0]	B	Y	N	I/O	E[7:6]
TSC	—	Y	Y	—	—
TPUCH[15:0]	A	Y	Y	—	—
T2CLK	—	Y	Y	—	—
TXDA	Bo	Y	Y	I/O	MC7
TXDB	Bo	Y	Y	I/O	MC5
VRH ⁴	—	—	—	—	—
VRL ⁴	—	—	—	—	—
XFC ²	—	—	—	—	—
XTAL ²	—	—	—	—	—

NOTES

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

Table 3–3. MC68HC16Y1 Power Connections

V _{STBY}	Standby RAM Power/Clock Synthesizer Power
V _{DDSYN}	Clock Synthesizer Power
V _{DDA} /V _{SSA}	A/D Converter Power
V _{SSE} /V _{DDE}	External Periphery Power (Source and Drain)

3.4 Signal Descriptions

The following tables define the MC68HC16Y1 signals. Table 3–4 shows signal origin, type, and active state. Table 3–5 describes signal functions. Both tables are sorted alphabetically by mnemonic. MCU pins often have multiple functions. More than one description can apply to a pin.

Table 3–4. MC68HC16Y1 Signal Characteristics

Signal Name	MCU Module	Signal Type	Active State
ADDR[23:0]	SCIM	Bus	—
AN[7:0]	ADC	Input	—
\overline{AS}	SCIM	Output	0
\overline{AVEC}	SCIM	Input	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	—
$\overline{CS[10:5]}$, $\overline{CS3}$, $\overline{CS0}$	SCIM	Output	0
\overline{CSBOOT}	SCIM	Output	0
\overline{CSE}	SCIM	Output	0
\overline{CSM}	SCIM	Output	0
DATA[15:0]	SCIM	Bus	—
\overline{DS}	SCIM	Output	0
$\overline{DSACK[1:0]}$	SCIM	Input	0
DSCLK	CPU16	Input	Serial Clock
DSI	CPU16	Input	(Serial Data)
DSO	CPU16	Output	(Serial Data)
ECLK	CPU16	Output	—
EXTAL	SCIM	Input	—
FC[2:0]	SCIM	Output	—
FREEZE	SCIM	Output	1
\overline{HALT}	SCIM	Input/Output	0
IC[4:1]	GPT	Input	—
IPIPE0	CPU16	Output	—
IPIPE1	CPU16	Output	—

Table 3–4. MC68HC16Y1 Signal Characteristics (Continued)

Signal Name	MCU Module	Signal Type	Active State
IRQ[7:1]	SCIM	Input	0
MISO	MCCI	Input/Output	—
MODCLK	SCIM	Input	—
MOSI	MCCI	Input/Output	—
OC[5:1]	GPT	Output	—
PADA[7:0]	ADC	Input	—
PAI	GPT	Input	—
PA[7:0]	SCIM	Input/Output	—
PB[7:0]	SCIM	Input/Output	—
PCLK	GPT	Output	—
PC[6:0]	SCIM	Input/Output	—
PE[7:0] ¹	SCIM	Input/Output	—
PF[7:0]	SCIM	Input/Output	—
PG[7:0]	SCIM	Input/Output	—
PGP[7:0]	GPT	Input/Output	—
PH[7:0]	SCIM	Input/Output	—
PMC[7:0]	MCCI	Input/Output	—
PWMA, PWMB	GPT	Output	—
QUOT	SCIM	Output	—
R \bar{W}	SCIM	Output	—
$\overline{\text{RESET}}$	SCIM	Input/Output	0
RXDA	MCCI	Input	—
RXDB	MCCI	Input	—
SCK	MCCI	Input/Output	—
SIZ[1:0]	SCIM	Output	—
$\overline{\text{SS}}$	MCCI	Input	0
T2CLK	TPU	Input	—
TPUCH[15:0]	TPU	Input/Output	—
TSC	SCIM	Input	1
TXDA	MCCI	Output	—
TXDB	MCCI	Output	—
XFC	SCIM	Input	—
XTAL	SCIM	Output	—

Table 3–5. MC68HC16Y1 Signal Function

Signal Name	Mnemonic	Function
Address Bus	ADDR[19:0]	20-bit address bus used by CPU16
Address Bus	ADDR[23:20]	4 MSB of IMB, logic states same as state of ADDR19
ADC Analog Input	AN[7:0]	Inputs to ADC MUX
Address Strobe	\overline{AS}	Indicates that a valid address is on the address bus
Autovector	\overline{AVEC}	Requests an automatic vector during interrupt acknowledge
Bus Error	\overline{BERR}	Indicates that a bus error has occurred
Bus Grant	\overline{BG}	Indicates that the MCU has relinquished the bus
Bus Grant Acknowledge	\overline{BGACK}	Indicates that an external device has assumed bus mastership
Breakpoint	\overline{BKPT}	Signals a hardware breakpoint to the CPU
Bus Request	\overline{BR}	Indicates that an external device requires bus mastership
System Clockout	CLKOUT	System clock output
General-Purpose Chip Selects	$\overline{CS[10:5]}$, CS3, CS0	Select external devices at programmed addresses
Boot ROM Chip Select	\overline{CSBOOT}	Chip select for external boot startup ROM
Emulation Mode Chip Selects	\overline{CSE} , \overline{CSM}	Select external emulation devices at internally mapped addresses. CSE is used for I/O port emulation, and CSM is used for ROM emulation.
Data Bus	DATA[15:0]	16-bit data bus
Data Strobe	\overline{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	$\overline{DSACK[1:0]}$	Provide asynchronous data transfers and dynamic bus sizing
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 CPU has entered background mode
Halt	\overline{HALT}	Suspend external bus activity
Instruction Pipeline	IPIPE[1:0]	Indicate instruction pipeline activity
Interrupt Request	$\overline{IRQ[7:1]}$	Provides an interrupt priority level to the CPU
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
Port ADA	PADA[7:0]	ADC general-purpose input port
Port A	PA[7:0]	SCIM general-purpose input/output port
Port B	PB[7:0]	SCIM general-purpose input/output port
Timer Clock Input	PCLK	Allows GPT to be clocked by an external source
Port C	PC[6:0]	SCIM general-purpose input/output port

Table 3–5. MC68HC16Y1 Signal Function (Continued)

Signal Name	Mnemonic	Function
Port E	PE[7:0]	SCIM general-purpose input/output port
Port F	PF[7:0]	SCIM general-purpose input/output port
Port G	PG[7:0]	SCIM general-purpose input/output port
Port GP	PGP[7:0]	GPT general-purpose input/output port
Port H	PH[7:0]	SCIM general-purpose input/output port
Port MC	PMC[7:0]	MCCI general-purpose input/output port
Quotient Out	QUOT	Provides the quotient bit of the polynomial divider
Read/Write	$\overline{R\overline{W}}$	Indicates the direction of data transfer on the bus
Reset	$\overline{\text{RESET}}$	System reset input. Clocked into SCIM.
SCI Transmit Data	TXDA, TXDB	Serial output from SCI
SCI Receive Data	RXDA, RXDB	Serial input to SCI
SPI Serial Clock	SCK	Clock output from SPI in master mode; clock input to SPI in slave mode
Size	SIZ[1:0]	Indicates the number of bytes to be transferred during a bus cycle
Slave Select	$\overline{\text{SS}}$	Selects SPI slave device; assertion while a device is in master mode causes mode fault
Three-State Control	TSC	Places all output drivers in a high-impedance state
TPU Channels	TPUCH[15:0]	Independently programmable timer channels
TPU Clock	T2CLK	External TPU clock input
External Filter Capacitor	XFC	Connection for external phase-locked loop filter capacitor

3.5 Intermodule Bus

The intermodule bus (IMB) is a standard bus developed to facilitate design of modular microcontrollers. MC68HC16Y1 modules communicate with one another and with external components via the IMB. Although the full IMB supports 24 address and 16 data lines, the CPU16 uses only 16 data lines and 20 address lines — ADDR[23:20] are driven to the same value as ADDR19. ADDR[23:20] are brought out to pins for test purposes.

3.6 System Memory Maps

Figures 3–3 through 3–5 are system memory maps. Figure 3–3 shows IMB addresses of internal registers and memory arrays. Figures 3–4 and 3–5 show system memory maps for two external address decoding schemes.

3.6.1 Internal Register Map

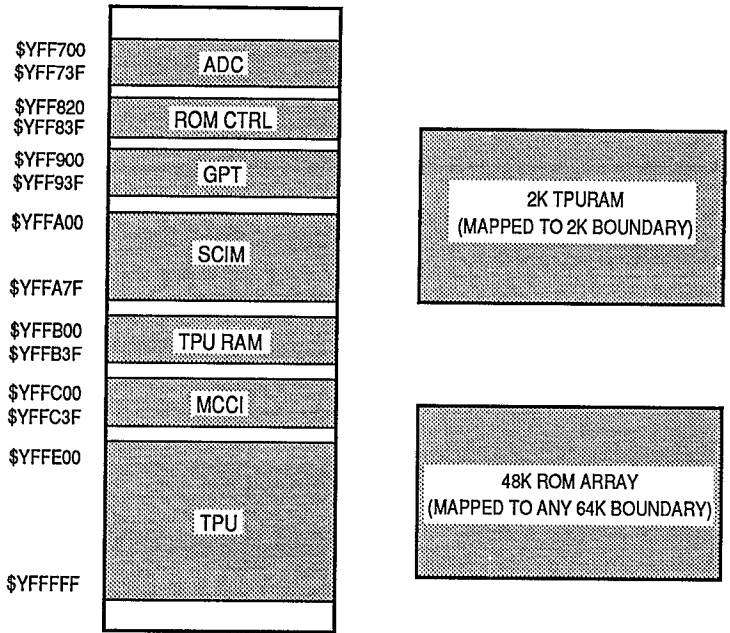
In Figure 3–3, IMB ADDR[23:20] are represented by the letter Y. The value represented by Y determines the base address of MCU module control registers. For the MC68HC16Y1, Y is equal to M111, where M is the logic state of the module mapping (MM) bit in the SCIMCR. Because the CPU16 uses only

ADDR[19:0], and ADDR[23:20] are driven to the same logic state as ADDR19, the CPU cannot access IMB addresses from \$080000 to \$F7FFFF. MM must be set (Y must equal \$F) for the MCU to function correctly. If M is cleared, internal registers are mapped to base address \$700000, and are inaccessible until a reset occurs. RAM and ROM arrays are mapped by base address registers in their respective control blocks. Unimplemented blocks are mapped externally.

3.6.2 Pseudolinear Address Maps

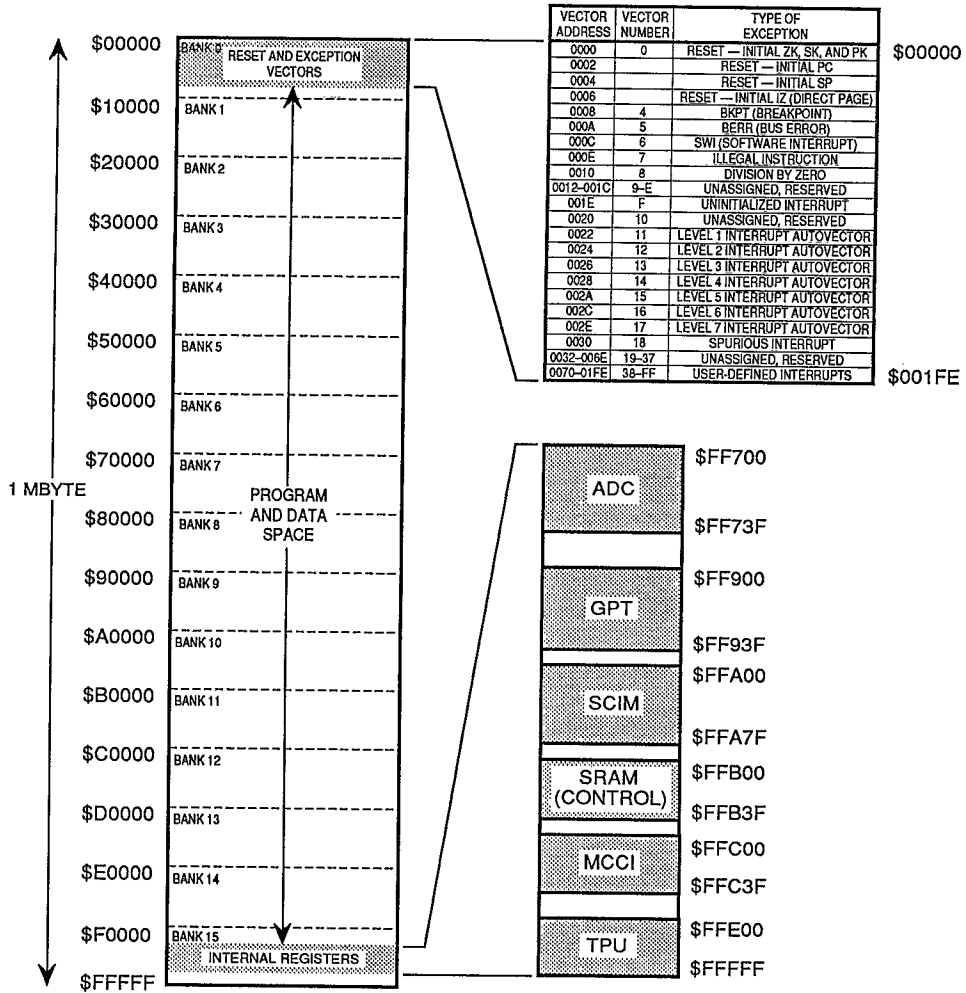
Figures 3–4 and 3–5 show the complete CPU16 pseudolinear address space. Address space can be split into physically distinct program and data spaces by decoding the MCU function code outputs. Figure 3–4 shows the memory map of a system that has combined program and data spaces. Figure 3–5 shows the memory map when MCU function code outputs are decoded.

Reset and exception vectors are mapped into bank 0 and cannot be relocated. The CPU16 program counter, stack pointer, and Z index register can be initialized to any address in pseudolinear memory, but exception vectors are limited to 16-bit addresses. To access locations outside of bank 0 during exception handler routines (including interrupt exceptions), a jump table must be used. Refer to **SECTION 5 CENTRAL PROCESSING UNIT** for more information concerning memory management, extended addressing, and exception processing. For more information concerning function codes, address space types, resets, and interrupts, refer to **SECTION 4 SINGLE-CHIP INTEGRATION MODULE**.



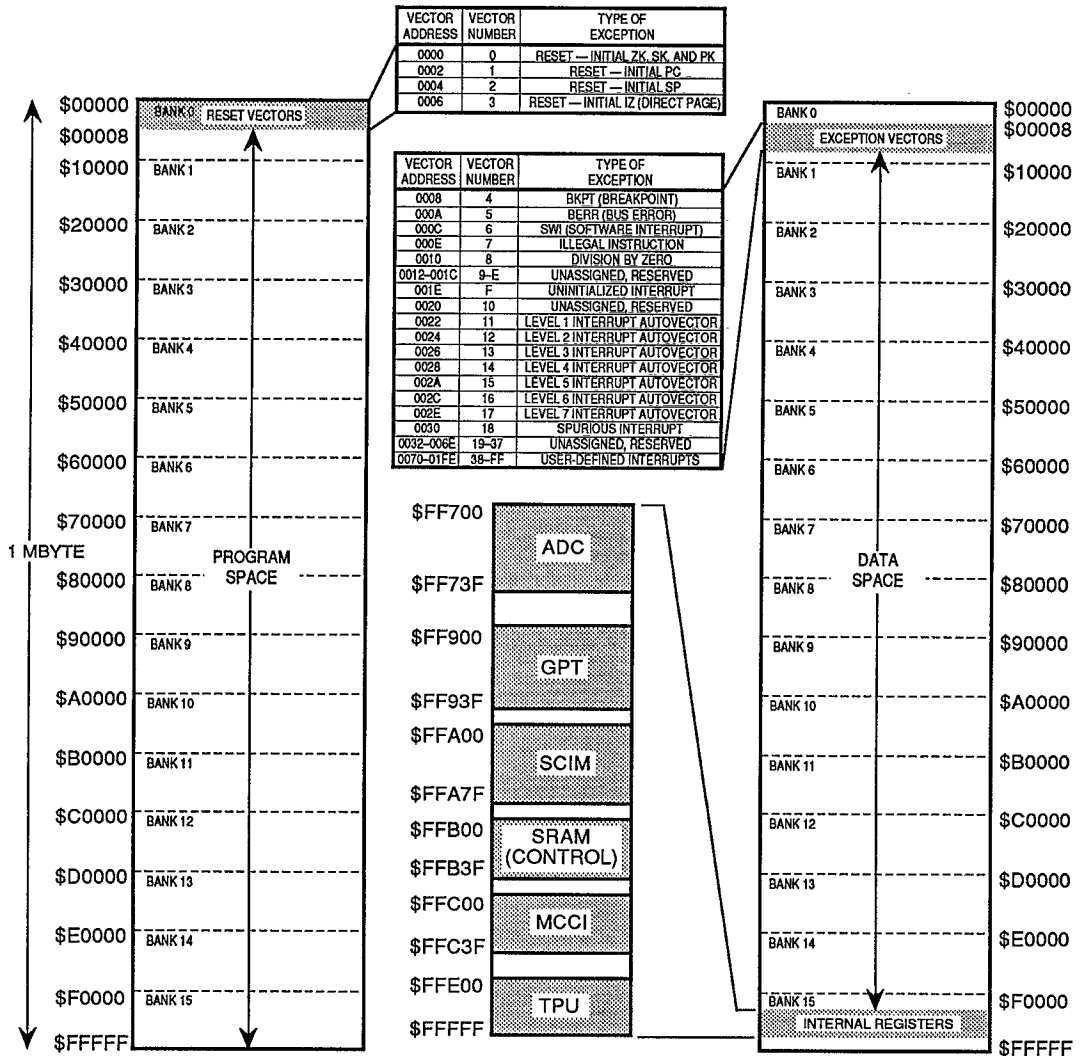
Y1 ADDRESS MAP

Figure 3-3. Internal Register Addresses



Y1 MEM MAP (COMBINED)

Figure 3-4. Pseudolinear Addressing With Combined Program and Data Spaces



Y1 MEM.MAP (SEPARATED)

Figure 3-5. Pseudolinear Addressing With Separated Program and Data Spaces

3.7 System Reset

System reset is a complex operation. The following paragraphs summarize reset operations. For a detailed description of SCIM operation during and after reset, refer to **SECTION 4 SINGLE-CHIP INTEGRATION MODULE**.

The logic states of certain pins during reset determines SCIM operating configuration. The SCIM reads pin configuration from DATA[11:0], internal module configuration from DATA[15:12], and basic operating information from $\overline{\text{BERR}}$, MODCLK, and $\overline{\text{BKPT}}$.

The $\overline{\text{BERR}}$, MODCLK, $\overline{\text{BKPT}}$, and DATA pins are normally pulled high internally during reset, causing the MCU to default to a specific configuration. However, a user can drive pins low during reset to achieve alternate configurations.

Basic operating options include system clock selection, background mode disable/enable, and external bus configuration. The MC68HC16Y1 can be configured as a fully-expanded MCU with a 24-bit external address bus, a 16-bit data bus, and separate chip-select signals; as a partially-expanded MCU with a 24-bit address bus and an 8-bit external data bus; or as a single-chip MCU with no external buses.

Table 3–6 is a summary of basic configuration options.

Table 3–6. 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

Many SCIM pins, including data and address bus pins, have multiple functions. The functions of these pins depend on the configuration selected during reset.

In expanded modes, the values of DATA[11:0] during reset determine the functions of pins with multiple functions. The functions of some pins can be changed subsequently by writing to a pin assignment register. Data bus pins have internal pull-ups and must be pulled low to achieve the desired alternate configuration.

External bus configuration determines which address and data bus lines are used and which general-purpose I/O ports are available. ADDR[18:3] serve as pins for ports A and B when the MCU is operating in single-chip mode. DATA[7:0] serve as port H pins in partially-expanded and single-chip modes, and DATA[15:8] serve as port G pins during single-chip operation. Table 3–7 is a summary of bus and port configuration.

Table 3–7. Bus and Port Configuration Options

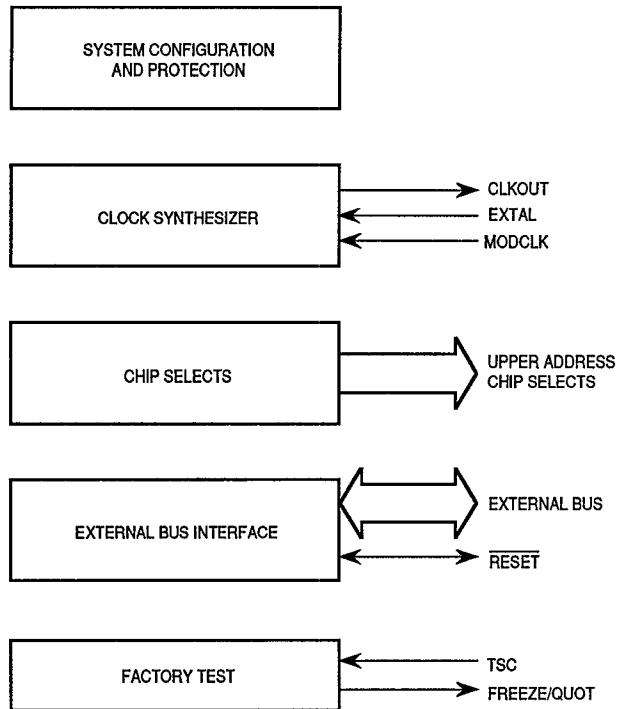
Bus Configuration	Address Bus Pins [18:3]	Data Bus Pins [15:0]	I/O Ports
Fully Expanded	ADDR[18:3]	DATA[15:0]	—
Partially Expanded	ADDR[18:3]	DATA[15:8]	DATA[7:0] = Port H
Single Chip	None	None	ADDR[18:11] = Port A ADDR[10:3] = Port B DATA[15:8] = Port G DATA[7:0] = Port H

SECTION 4 SINGLE-CHIP INTEGRATION MODULE

The single-chip integration module (SCIM) determines system configuration and operating mode, monitors operation, controls clock source and speed, manages internal and external bus interfaces, provides chip-select signals, and performs system test. This section provides sufficient information for normal use of the SCIM. Refer to the *SCIM Reference Manual (SCIMRM/AD)* for detailed information.

4.1 General

The SCIM consists of five functional blocks, shown in Figure 4-1.



SCIM BLOCK

Figure 4–1. Single-Chip Integration Module Block Diagram

The configuration and protection block controls basic system configuration and provides bus and software watchdog monitors. It also includes a periodic interrupt generator to support the execution of time-critical control routines.

The system clock generates clock signals used by the SCIM, other IMB modules, and external devices.

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

The chip-select block provides eight general-purpose chip-select signals, two emulation chip-select signals, and a boot ROM chip-select signal. Each general-purpose chip-select signal has an associated base register and option register that contain the programmable characteristics of that chip select.

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

The SCIM has three basic operating modes.

In fully-expanded mode, the SCIM provides a 24-bit external address bus and a 16-bit external data bus, eight general-purpose chip-select lines, a boot ROM chip-select line, and seven interrupt-request inputs. The bus-control pins, the chip-select pins, and the interrupt-request pins can be configured as general-purpose I/O ports. In addition, two emulation chip-select lines are available — the CSE line can be used to select an external port-replacement unit, and the CSM line can be used to select an external ROM-emulation device.

In partially-expanded mode, the SCIM provides a single general-purpose I/O port, a 24-bit external address bus, an 8-bit external data bus, seven general-purpose chip-select lines, a boot ROM chip-select line, and seven interrupt-request inputs. The bus-control pins, the chip-select pins, and the interrupt-request pins can be configured as general-purpose I/O ports.

In single-chip mode, the SCIM provides seven general-purpose I/O ports, no external address or data buses, one general-purpose chip-select line, and a boot ROM chip-select line.

Although the SCIM makes up to 24 address lines available, the CPU16 module drives only ADDR[19:0]. Since ADDR[23:20] are driven to the same state as ADDR19, the external bus size is effectively 20 bits.

Operating mode is determined by the logic states of specific MCU pins during reset. Refer to 4.6 Reset for a detailed description of reset and operating modes.

4.2 System Configuration and Protection

The system configuration functional block controls device module mapping, monitors internal activity, and provides periodic interrupt generation. Figure 4-2 is a block diagram of the submodule.

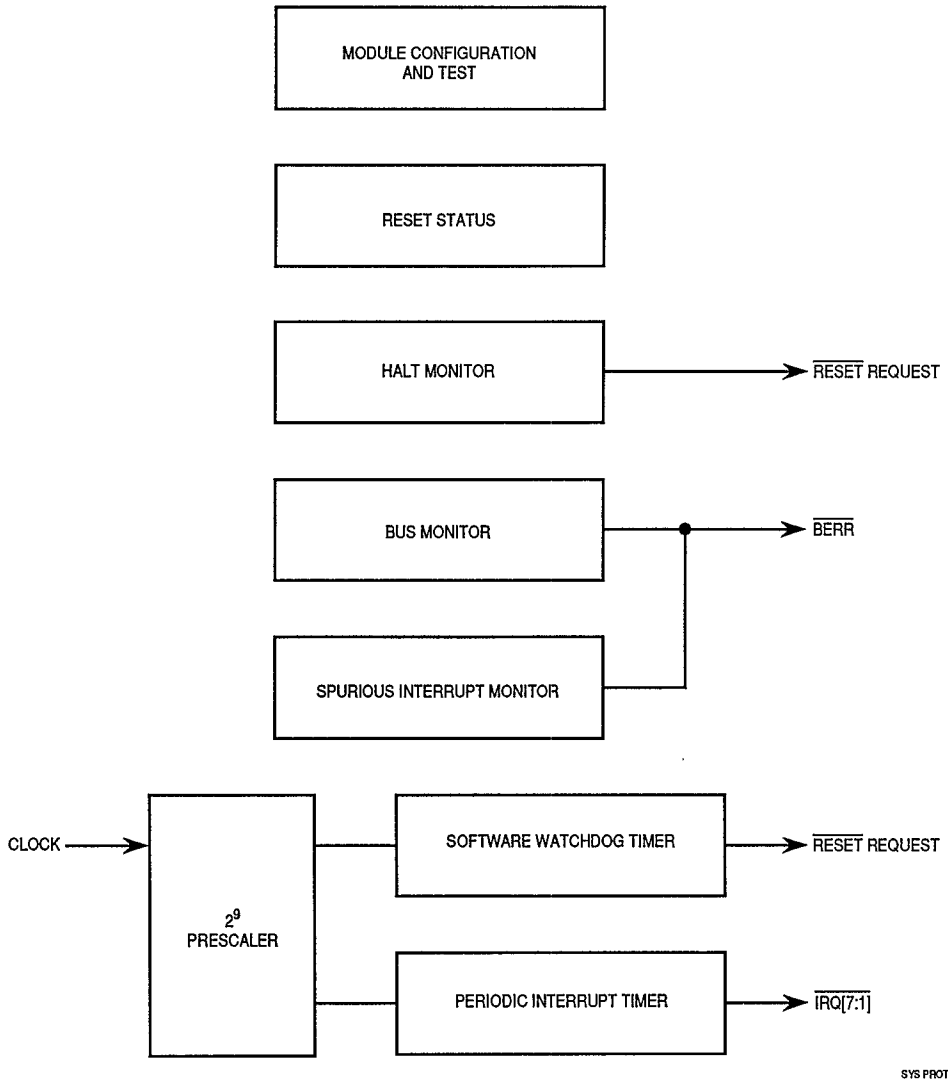


Figure 4–2. System Configuration and Protection

4.2.1 Module Mapping

Control registers for all the modules in the microcontroller are mapped into a four Kbyte block. The state of the module mapping bit (MM) in the SCIM configuration register (SCIMCR) determines where the control register block is located in the system memory map. When MM = 0, register addresses range from \$7FF000 to \$7FFFFFF; when MM = 1, register addresses range from \$FFF000 to \$FFFFFF.

ADDR[23:20] are driven to the same logic state as ADDR19. MM corresponds to IMB ADDR23. If MM is cleared, the SCIM maps IMB modules into address space \$7FF000–\$7FFFFFF, which is inaccessible to the CPU16. Modules remain inaccessible until reset occurs. The reset state of MM is one, but the bit can be written once. Initialization software should make certain that MM remains set.

4.2.2 Interrupt Arbitration

Each module that can generate interrupt requests has an interrupt arbitration (IARB) field. Arbitration between interrupt requests of the same priority is performed by serial contention between IARB field bit values. Contention must take place whenever an interrupt request is acknowledged, even when there is only a single request pending. For contention to take place, an IARB field must have a non-zero value. If an interrupt request from a module with an IARB field value of %0000 is recognized, the CPU16 processes a spurious interrupt exception.

Because the SCIM routes external interrupt requests to the CPU16, the SCIM IARB field value is used for arbitration between internal and external interrupts of the same priority. The reset value of IARB for the SCIM is %1111, and the reset IARB value for all other modules is %0000, which prevents SCIM interrupts from being discarded during initialization. Refer to **4.7 Interrupts** for a discussion of interrupt arbitration.

4.2.3 Show Internal Cycles

A show cycle allows internal bus transfers to be monitored externally. The SHEN field in the MCR determines what the external bus interface does during internal transfer operations. Table 4–1 shows whether data is driven externally, and whether external bus arbitration can occur. Refer to **4.5.6.2 Show Cycles** for more information.

Table 4–1. Show Cycle Enable Bits

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

4.2.4 Factory Test Mode

The IMB can be slaved to an external master. This mode is reserved for factory test. Test mode is enabled by holding DATA11 low during reset. The read-only slave enabled (SLVEN) bit shows the reset state of DATA11.

4.2.5 Register Access

Although the module configuration register contains an access control bit (SUPV), the CPU16 always operates in the supervisor access mode. The state of SUPV has no meaning.

4.2.6 Bus Monitor

The internal bus monitor checks data and size acknowledge (\overline{DSACK}) or autovector (\overline{AVEC}) signal response times during normal bus cycles. The monitor asserts the internal bus error (\overline{BERR}) signal when the response time is excessively long.

\overline{DSACK} and \overline{AVEC} response times are measured in clock cycles. Maximum allowable response time can be selected by setting the bus monitor timing (BMT) field in the system protection control register (SYPCR). Table 4–2 shows the periods allowed.

Table 4–2. Bus Monitor Period

BMT	Bus Monitor Timeout Period
00	64 System Clocks
01	32 System Clocks
10	16 System Clocks
11	8 System Clocks

The monitor does not check \overline{DSACK} response on the external bus unless the CPU16 initiates a 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.

When monitoring transfers to an 8-bit port, the bus monitor does not reset until both byte accesses of a word transfer are completed. Monitor timeout period must be at least twice the number of clocks that a single byte access requires.

4.2.7 Halt Monitor

The halt monitor responds to an assertion of the $\overline{\text{HALT}}$ signal on the internal bus. Refer to **4.5.5.2 Double Bus Fault** for more information. Halt monitor reset can be inhibited by the halt monitor (HME) bit in SYPCR.

4.2.8 Spurious Interrupt Monitor

During interrupt exception processing, the CPU16 normally acknowledges an interrupt request, arbitrates among various sources of interrupt, recognizes the highest priority source, and then acquires a vector or responds to a request for autovectoring. The spurious interrupt monitor asserts the internal bus error signal ($\overline{\text{BERR}}$) if no interrupt arbitration occurs during interrupt exception processing. The assertion of $\overline{\text{BERR}}$ causes the CPU16 to load the spurious interrupt exception vector into the program counter. The spurious interrupt monitor cannot be disabled. Refer to **4.7 Interrupts** for further information. Refer to **SECTION 5 CENTRAL PROCESSING UNIT** for information about interrupt exception processing.

4.2.9 Software Watchdog

The software watchdog is controlled by the software watchdog enable (SWE) bit in SYPCR. When enabled, the watchdog requires that a service sequence be written to software service register SWSR on a periodic basis. If servicing does not take place, the watchdog times out and asserts the external reset signal.

Perform a software watchdog service sequence as follows:

- a. Write \$55 to SWSR.
- b. Write \$AA to SWSR.

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

Watchdog clock rate is affected by the software watchdog prescale (SWP) and software watchdog timing (SWT) fields in SYPCR.

SWP determines system clock prescaling for the watchdog timer. One of two options, either no prescaling or prescaling by a factor of 512, can be selected. The value of SWP is affected by the state of the MODCLK pin during reset, as shown in Table 4-3. System software can change SWP value.

**Table 4-3.
MODCLK Pin and
SWP Bit During Reset**

MODCLK	SWP
0 (External Clock)	1 (+ 512)
1 (Internal Clock)	0 (+ 1)

The SWT field selects the divide ratio used to establish software watchdog timeout period. Timeout period is given by the following equations.

$$\text{Timeout Period} = \left(\frac{(128) (\text{Divide Ratio})}{\text{EXTAL Frequency}} \right)$$

or

$$\text{Timeout Period} = \left(\frac{\text{System Clock Frequency}}{\text{Divide Ratio}} \right)$$

Table 4-4 shows the ratio for each combination of SWP and SWT bits. When SWT[1:0] are modified, a watchdog service sequence must be performed before the new timeout period can take effect.

**Table 4-4.
Software Watchdog Ratio**

SWP	SWT	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

Figure 4-3 is a block diagram of the watchdog timer and the clock control for the periodic interrupt timer.

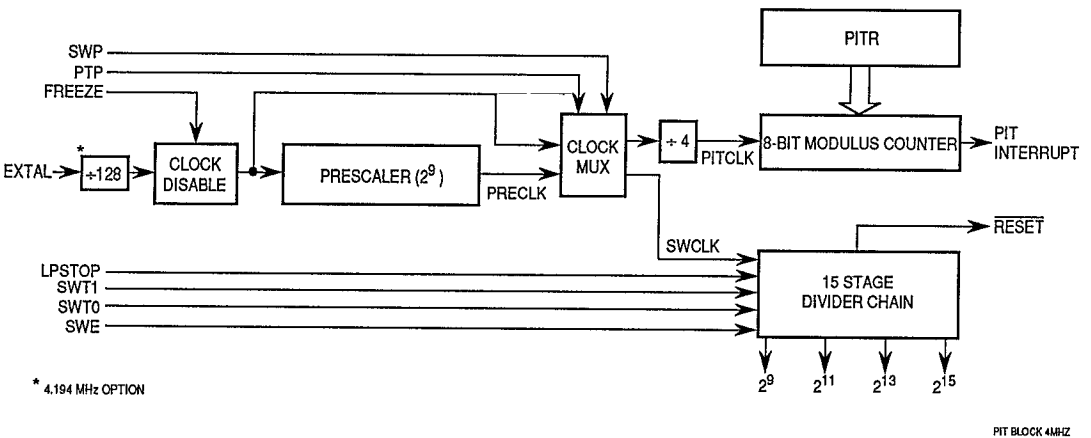


Figure 4-3. Periodic Interrupt Timer and Software Watchdog Timer

4.2.10 Periodic Interrupt Timer

The periodic interrupt timer allows the generation of interrupts of specific priority at predetermined intervals. This capability is often used to schedule control system tasks that must be performed within time constraints. The timer consists of a prescaler, a modulus counter, and registers that determine interrupt timing, priority, and vector assignment. For further information about interrupt exception processing refer to **SECTION 5 CENTRAL PROCESSING UNIT**.

The periodic interrupt modulus counter is clocked by a signal (PITCLK) derived from the system clock. The value of the periodic timer prescaler (PTP) bit in the periodic interrupt timer register (PITR) determines whether the system clock signal fed to the PIT is prescaled. When PTP = 0, the signal is not prescaled; when PTP = 1, the signal is prescaled by a factor of 512. Whether prescaled or not, the system clock signal is divided by four to generate PITCLK.

Since system clock frequency is equal to the frequency applied to the EXTAL pin \div 128, use the following expression to calculate timer period.

$$\text{PIT Period} = \frac{(\text{PIT Modulus}) (\text{Prescaler Value}) (512)}{\text{EXTAL Frequency}}$$

As shown in Table 4–5, the initial value of PTP is determined by the state of the MODCLK pin during reset, but the value can be changed by system software.

Table 4–5.
MODCLK Pin and
PTP Bit at Reset

MODCLK	PTP
0 (External Clock)	1 (+ 512)
1 (Internal Clock)	0 (+ 1)

The periodic interrupt timer begins to run when a non-zero value is written to the periodic interrupt timer modulus (PITM) field. The timer runs whether or not interrupts are enabled. Each time it counts down to zero, the modulus counter is reloaded with the value in PITM, and counting repeats. If interrupts are enabled, an interrupt service request is made when time counter value reaches zero. When a new value is written to PITR, it is loaded into the modulus counter when the current count is completed. Writing a zero value to PITM turns off the timer.

PIT interrupts are controlled by two fields in the periodic interrupt control register (PICR). The periodic interrupt request level (PIRQL) field enables interrupts and determines interrupt priority. The periodic interrupt vector (PIV) field contains the vector number used when the interrupt is acknowledged.

When PIRQL value is %000, the PIT interrupt is disabled. When PIRQL contains a non-zero value, that value is compared to the CPU16 interrupt priority mask to determine whether an interrupt request is recognized. Table 4–6 shows PIRQL values. By hardware convention, a PIT interrupt is serviced before an external interrupt request of the same priority.

Table 4–6.
Periodic Interrupt Priority

PIRQL	Priority Level
000	Periodic Interrupt Disabled
001	Interrupt Priority Level 1
010	Interrupt Priority Level 2
011	Interrupt Priority Level 3
100	Interrupt Priority Level 4
101	Interrupt Priority Level 5
110	Interrupt Priority Level 6
111	Interrupt Priority Level 7

The PIV field contains the periodic interrupt vector. When a PIT interrupt is acknowledged, the vector number is multiplied by two to form the vector offset, which is added to \$0000 to obtain the address of the vector. Reset value of the PIV field is \$0F, which generates the uninitialized interrupt vector.

4.2.11 Monitor Low-Power Operation

When the CPU16 executes the LPSTOP instruction, the current interrupt priority mask is stored in the clock control logic, internal clocks are disabled according to the state of the STSCIM bit in the MCR, and the MCU enters low-power stop mode. The bus monitor, halt monitor, and spurious interrupt monitor are all inactive during low-power stop.

During low-power stop, the clock input to the software watchdog timer is disabled and the timer stops. The software watchdog begins to run again on the first rising clock edge after low-power stop ends (refer to **4.3.4 Clock Low-Power Operation**). The watchdog is not reset by low-power stop. A service sequence must be performed to reset the timer.

The periodic interrupt timer does not respond to the LPSTOP instruction, and continues to run at the programmed PITCLK frequency during LPSTOP. A PIT interrupt can bring the MCU out of the low-power stop condition if it has a higher priority than the interrupt mask value stored in the clock control logic when low-power stop is initiated. To stop the periodic interrupt timer, PITR must be loaded with a zero value before the LPSTOP instruction is executed.

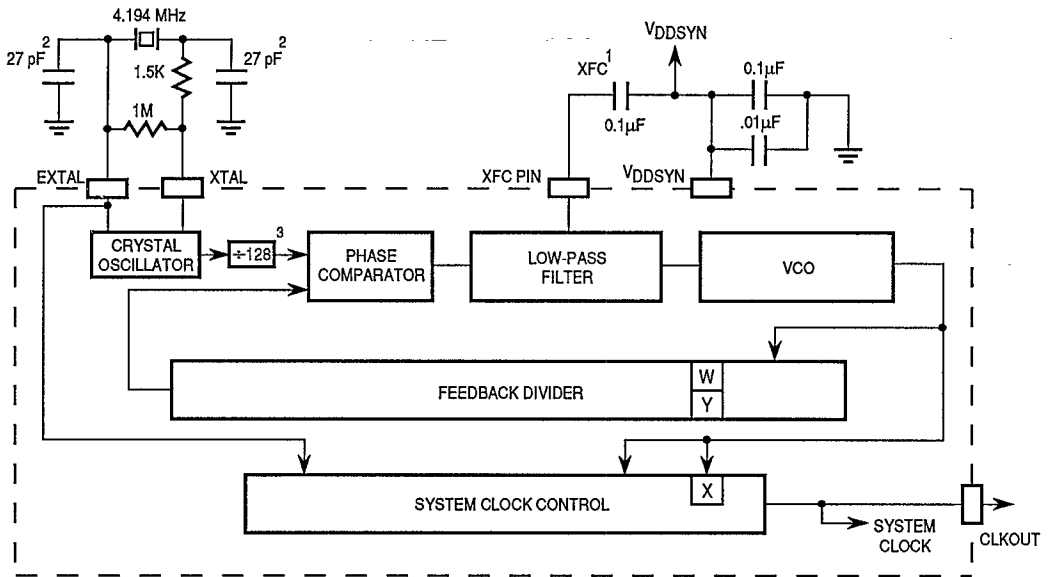
4.2.12 Freeze Operation

The FREEZE signal halts MCU operations during debugging. FREEZE is asserted internally by the CPU16 if a breakpoint occurs while background mode is enabled. When FREEZE is asserted, only the bus monitor, software watchdog, and periodic interrupt timer are affected. The halt monitor and spurious interrupt monitor continue to operate normally. Setting the freeze bus monitor (FRZBM) bit in the MCR disables the bus monitor when FREEZE is asserted, and setting the freeze software watchdog (FRZSW) bit disables the software watchdog and the periodic interrupt timer when FREEZE is asserted. When FRZSW is set, FREEZE assertion must be at least two times the PIT clock source period to ensure an accurate number of PIT counts.

4.3 System Clock

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

The system clock signal can be generated in three ways. An internal phase-locked loop can synthesize the clock from either an internal reference or an external reference, or an external clock signal can be input. Keep the distinction between an external reference and an external clock signal in mind while reading the rest of this section. Figure 4-4 is a block diagram of the system clock.



1. MUST BE LOW-LEAKAGE CAPACITOR (INSULATION RESISTANCE 30,000 MΩ OR GREATER).
2. 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.
3. 4 MHz DIVIDED TO 32 KHz.

18 SYS CLOCK BLOCK 4MHz

Figure 4-4. System Clock Block Diagram

4.3.1 Clock Sources

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

A reference crystal must be connected between the EXTAL and XTAL pins to use the internal oscillator. Clock synthesizer specifications are provided in **APPENDIX A ELECTRICAL CHARACTERISTICS**, Table A-3, Clock Control Timing.

If either an external reference signal or an external system clock signal is applied through the EXTAL pin, the XTAL pin must be left floating. External reference signal frequency must be less than or equal to maximum specified reference frequency. External system clock signal frequency must be less than or equal to maximum specified system clock frequency.

When an external system clock signal is applied, 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:

$$\text{Minimum external clock period} = \frac{\text{minimum external clock high/low time}}{50\% - \text{percentage variation of external clock input duty cycle}}$$

4.3.2 Clock Synthesizer Operation

A voltage controlled oscillator (VCO) generates the system clock signal. A portion of the clock signal is fed back to a divider/counter. The divider controls the frequency of one input to a phase comparator. The other phase comparator input is a reference signal, either from the internal crystal oscillator or from an external source. The reference frequency is divided by 128 before being fed to the comparator. The comparator generates a control signal proportional to the difference in phase between its two inputs. The signal is low-pass filtered and used to correct VCO output frequency.

The synthesizer locks when VCO frequency is equal to reference frequency divided by 128. Lock time is affected by the filter time constant and by the amount of difference between the two comparator inputs. Whenever comparator input changes, the synthesizer must relock. Lock status is shown by the SLOCK bit in SYNCR.

To maintain a 50% clock duty cycle, VCO frequency is either two or four times clock frequency, depending on the state of the X bit in SYNCR.

The MCU does not come out of reset state until the synthesizer locks. Crystal type, characteristic frequency, and layout of external oscillator circuitry affect lock time.

The low-pass filter requires an external low-leakage capacitor, typically 0.1 μF with an insulation resistance specification of 30,000 $\text{M}\Omega$ or greater, connected between the XFC and V_{DDSYN} pins.

V_{DDSYN} is used to power the clock circuits. 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 the clock synthesizer is used, control register SYNCR determines operating frequency and various modes of operation. Because the CPU16 operates only in supervisor mode, SYNCR can be read or written at any time.

The SYNCR X bit controls a divide by two prescaler that is not in the synthesizer feedback loop. When $X = 0$ (reset state), the divider is enabled, and system clock frequency is one-fourth VCO frequency; setting X disables the divider, doubling clock speed without changing VCO speed. There is no VCO relock delay. The SYNCR W bit controls a three-bit prescaler in the feedback divider. Setting W increases VCO speed by a factor of four. The SYNCR Y field determines the count modulus for a modulo 64 down counter, causing it to divide by a value of $Y + 1$. When either the W or Y value change, there is a VCO relock delay (**APPENDIX A ELECTRICAL CHARACTERISTICS**, Table A-3, Clock Control Timing).

Clock frequency is determined by SYNCR bit settings as follows:

$$F_{\text{SYSTEM}} = \frac{F_{\text{REFERENCE}}}{128} [4(Y + 1)(2^{2W+X})]$$

For the device to perform correctly, the clock frequency selected by the W, X, and Y bits must be within the limits specified for the MCU. Refer to **APPENDIX A ELECTRICAL CHARACTERISTICS** for the maximum specified clock frequency.

The reset state of SYNCR (\$3F00) produces a modulus-64 count. System frequency is two times reference frequency.

Table 4-7 shows multipliers for combinations of SYNCR bits. The range of possible system frequencies can exceed the maximum specified system clock frequency. For instance, with a 4.194-MHz reference and a maximum system frequency of 16.78 MHz (**APPENDIX A**, Table A-3, Clock Control Timing), W and X must not both be set at any count modulus greater than $Y = \%001111$. Table 4-7 shows available clock frequencies for a 16.78-MHz system with a 4.194-MHz reference.

Table 4-7. Clock Control Multipliers

To obtain clock frequency, find counter modulus in the left column, then multiply reference frequency by value in appropriate prescaler cell

Modulus Y	Prescalers			
	[W:X] = 00	[W:X] = 01	[W:X] = 10	[W:X] = 11
000000	.03125	.0625	.125	.25
000001	.0625	.125	.25	.5
000010	.09375	.1875	.375	.75
000011	.125	.25	.5	1
000100	.15625	.3125	.625	1.25
000101	.1875	.375	.75	1.5
000110	.21875	.4375	.875	1.75
000111	.25	.5	1	2
001000	.21875	.5625	1.125	2.25
001001	.3125	.625	1.25	2.5
001010	.34375	.6875	1.375	2.75
001011	.375	.75	1.5	3
001100	.40625	.8125	1.625	3.25
001101	.4375	.875	1.75	3.5
001110	.46875	.9375	1.875	3.75
001111	.5	1	2	4
010000	.53125	1.0625	2.125	4.25
010001	.5625	1.125	2.25	4.5
010010	.59375	1.1875	2.375	4.75
010011	.625	1.25	2.5	5
010100	.65625	1.3125	2.625	5.25
010101	.6875	1.375	2.75	5.5
010110	.71875	1.4375	2.875	5.75
010111	.75	1.5	3	6
011000	.78125	1.5625	3.125	6.25
011001	.8125	1.625	3.25	6.5
011010	.84375	1.6875	3.375	6.75
011011	.875	1.75	3.5	7
011100	.90625	1.8125	3.625	7.25
011101	.9375	1.875	3.75	7.5
011110	.96875	1.9375	3.875	7.75
011111	1	2	4	8

Table 4–7. Clock Control Multipliers (Continued)

To obtain clock frequency, find counter modulus in the left column, then multiply reference frequency by value in appropriate prescaler cell

Modulus Y	Prescalers			
	[W:X] = 00	[W:X] = 01	[W:X] = 10	[W:X] = 11
10000	1.03125	2.0625	4.125	8.25
10001	1.0625	2.125	4.25	8.5
10010	1.09375	2.1875	4.375	8.75
10011	1.125	2.25	4.5	9
100100	1.15625	2.3125	4.675	9.25
100101	1.1875	2.375	4.75	9.5
100110	1.21875	2.4375	4.875	9.75
100111	1.25	2.5	5	10
101000	1.28125	2.5625	5.125	10.25
101001	1.3125	2.625	5.25	10.5
101010	1.34375	2.6875	5.375	10.75
101011	1.375	2.75	5.5	11
101100	1.40625	2.8125	5.625	11.25
101101	1.4375	2.875	5.75	11.5
101110	1.46875	2.9375	5.875	11.75
101111	1.5	3	6	12
110000	1.53125	3.0625	6.125	12.25
110001	1.5625	3.125	6.25	12.5
110010	1.59375	3.1875	6.375	12.75
110011	1.625	3.25	6.5	13
110100	1.65625	3.3125	6.625	13.25
110101	1.6875	3.375	6.75	13.5
110110	1.71875	3.4375	6.875	13.75
110111	1.75	3.5	7	14
111000	1.78125	3.5625	7.125	14.25
111001	1.8125	3.625	7.25	14.5
111010	1.84375	3.6875	7.375	14.75
111011	1.875	3.75	7.5	15
111100	1.90625	3.8125	7.625	15.25
111101	1.9375	3.875	7.75	15.5
111110	1.96875	3.9375	7.875	15.75
111111	2	4	8	16

Table 4–8. System Frequencies from 4.194-MHz Reference

To obtain clock frequency in kHz, find counter modulus in the left column, then look in appropriate prescaler cell. Shaded cells are values that exceed specifications.

Modulus Y	Prescaler			
	[W:X] = 00	[W:X] = 01	[W:X] = 10	[W:X] = 11
000000	131	262	524	1049
000001	262	524	1049	2097
000010	393	786	1573	3146
000011	524	1049	2097	4194
000100	655	1311	2621	5243
000101	786	1573	3146	6291
000110	918	1835	3670	7340
000111	1049	2097	4194	8389
001000	1180	2359	4719	9437
001001	1311	2621	5243	10486
001010	1442	2884	5767	11534
001011	1573	3146	6291	12583
001100	1704	3408	6816	13631
001101	1835	3670	7340	14680
001110	1966	3932	7864	15729
001111	2097	4194	8389	16777
010000	2228	4456	8913	17826
010001	2359	4719	9437	18874
010010	2490	4981	9961	19923
010011	2621	5243	10486	20972
010100	2753	5505	11010	22020
010101	2884	5767	11534	23069
010110	3015	6029	12059	24117
010111	3146	6291	12583	25166
011000	3277	6554	13107	26214
011001	3408	6816	13631	27263
011010	3539	7078	14156	28312
011011	3670	7340	14680	29360
011100	3801	7602	15204	30409
011101	3932	7864	15729	31457
011110	4063	8126	16253	32506
011111	4194	8389	16777	33554

4

**Table 4–8. System Frequencies from 4.194-MHz Reference
(Continued)**

To obtain clock frequency in kHz, find counter modulus in the left column, then look in appropriate prescaler cell. Shaded cells are values that exceed specifications.

Y	[W:X] = 00	[W:X] = 01	[W:X] = 10	[W:X] = 11
100000	4325	8651	17302	34603
100001	4456	8913	17826	35652
100010	4588	9175	18350	36700
100011	4719	9437	18874	37749
100100	4850	9699	19399	38797
100101	4981	9961	19923	39846
100110	5112	10224	20447	40894
100111	5243	10486	20972	41943
101000	5374	10748	21496	42992
101001	5505	11010	22020	44040
101010	5636	11272	22544	45089
101011	5767	11534	23069	46137
101100	5898	11796	23593	47186
101101	6029	12059	24117	48234
101110	6160	12321	24642	49283
101111	6291	12583	25166	50332
110000	6423	12845	25690	51380
110001	6554	13107	26214	52428
110010	6685	13369	26739	53477
110011	6816	13631	27263	54526
110100	6947	13894	27787	55575
110101	7078	14156	28312	56623
110110	7209	14418	28836	57672
110111	7340	14680	29360	58720
111000	7471	14942	2988	59769
111001	7602	15204	30409	60817
111010	7733	15466	30933	61866
111011	7864	15729	31457	62915
111100	7995	15991	31982	63963
111101	8126	16253	32506	65011
111110	8258	16515	33030	66060
111111	8389	16777	33554	67109

4.3.3 External Bus Clock

The state of the external clock division bit (EDIV) in SYNCR determines clock rate for the external bus clock signal (ECLK) available on pin ADDR23. ECLK is a bus clock for MC6800 devices and peripherals. ECLK frequency can be set to system clock frequency divided by eight or system clock frequency divided by sixteen. The clock is enabled by the CSPA[1:4] field in chip select pin assignment register 1 (CSPAR1). The operation of the external bus clock during low-power stop is described in the following paragraph. Refer to **4.8 Chip Selects** for more information about the external bus clock.

4.3.4 Clock Low-Power Operation

Low-power operation is initiated by the CPU16. To reduce power consumption selectively, the CPU can set the STOP bits in each module configuration register. To minimize overall microcontroller power consumption, the CPU can execute the LPSTOP instruction, which causes the SCIM to turn off the system clock.

When individual module STOP bits are set, clock signals inside each module are turned off, but module registers are still accessible.

When the CPU executes LPSTOP, a special CPU space bus cycle writes a copy of the current interrupt mask into the clock control logic. The SCIM brings the MCU out of low-power operation when either an interrupt of higher priority than the stored mask or a reset occurs. Refer to **4.5.4.2 LPSTOP Broadcast Cycle** and **SECTION 5 CENTRAL PROCESSING UNIT** for more information.

During a low-power stop, unless the system clock signal is supplied by an external source and that source is removed, the SCIM clock control logic and the SCIM clock signal (SCIMCLK) continue to operate. The periodic interrupt timer and input logic for the RESET and IRQ pins are clocked by SCIMCLK. The SCIM can also continue to generate the CLKOUT signal while in low-power mode.

The stop mode single-chip integration module clock (STSCIM) and stop mode external clock (STEXT) bits in SYNCR determine clock operation during low-power stop. Table 4–9 is a summary of the effects of STSCIM and STEXT with various clock sources. MODCLK value is the logic level on the MODCLK pin during the last reset before LPSTOP execution. Any clock in the off state is held low.

Table 4–9. Clock Control

Mode	Pins		SYNCR Bits		Clock Source	
LPSTOP	MODCLK	EXTAL	STSCIM	STEXT	SCIMCLK	CLKOUT
No	0	External Clock	X	X	External Clock	External Clock
Yes	0	External Clock	0	0	External Clock	Off
Yes	0	External Clock	0	1	External Clock	External Clock
Yes	0	External Clock	1	0	External Clock	Off
Yes	0	External Clock	1	1	External Clock	External Clock
No	1	Crystal/Reference	X	X	VCO	VCO
Yes	1	Crystal/Reference	0	0	Crystal/Reference	Off
Yes	1	Crystal/Reference	0	1	Crystal/Reference	Crystal/Reference
Yes	1	Crystal/Reference	1	0	VCO	Off
Yes	1	Crystal/Reference	1	1	VCO	VCO

4

4.3.5 Loss of Reference Signal

The state of the reset enable (RSTEN) bit in SYNCR determines what happens when clock logic detects a reference failure.

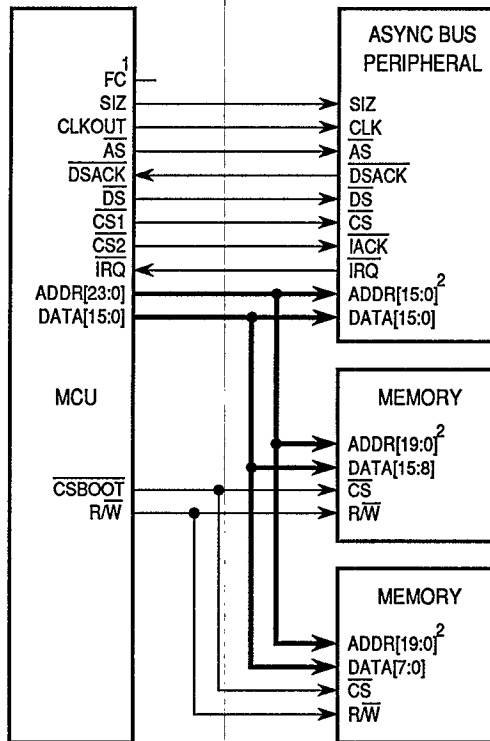
When RSTEN is cleared (default state out of reset), the clock synthesizer is forced into an operating condition referred to as limp mode. Limp mode frequency varies from device to device, but maximum limp frequency does not exceed one half maximum system clock when $X = 0$, or maximum system clock frequency when $X = 1$.

When RSTEN is set, the SCIM resets the MCU.

The limp status bit (SLIMP) in SYNCR indicates whether the synthesizer has a reference signal. It is set when a reference failure is detected.

4.4 External Bus Interface

In fully expanded mode, the external bus interface (EBI) transfers information between the internal MCU bus and external devices. Figure 4-5 shows a basic system with external memory and peripherals.



- 1. CAN BE DECODED TO PROVIDE ADDITIONAL ADDRESS SPACE.
- 2. VARIES DEPENDING UPON PERIPHERAL MEMORY SIZE.

16 EXAMPLE SYS BLOCK

Figure 4-5. MC68HC16Y1 Basic System

The external bus has 20 address lines and 16 data lines. ADDR[19:0] are normal address outputs, ADDR[23:20] are driven to the same logic state as ADDR19.

A three-line handshaking interface performs external bus arbitration. The interface supports byte, word, and long-word transfers. The EBI performs dynamic sizing for data accesses.

The maximum number of bits transferred during an access is referred to as port width. Widths of eight and sixteen bits can be accessed by asynchronous bus cycles controlled by the data size ($SIZ0$ and $SIZ1$) and the data and size acknowledge ($DSACK0$ and $DSACK1$) signals. Multiple bus cycles may be required for a dynamically sized transfer.

To add flexibility and minimize the necessity for external logic, MCU chip-select logic can be synchronized with EBI transfers. Refer to **4.8 Chip Selects** for more information.

4.4.1 Bus Signals

The address bus provides addressing information to external devices. The data bus transfers 8-bit and 16-bit data between the MCU and external devices. 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.

Control signals indicate the beginning of each bus cycle, the address space it is to take place in, the size of the transfer, and the type of cycle. External devices decode these signals, respond to transfer data and terminate the bus cycle. The EBI operates in an asynchronous mode for any port width.

4.4.1.1 Address Bus

Bus signals $ADDR[19:0]$ define the address of the byte (or 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.

4.4.1.2 Address Strobe

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.

4.4.1.3 Data Bus

Bus signals $DATA[15:0]$ comprise a bidirectional, nonmultiplexed parallel bus that transfers data to or from the MCU. A read or write operation can transfer eight or sixteen 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.

4.4.1.4 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.

4.4.1.5 Read/Write Signal

The read/write (R/\overline{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{AS} is asserted. R/\overline{W} only transitions when a write cycle is preceded by a read cycle or vice versa. The signal may remain low for two consecutive write cycles.

4.4.1.6 Size Signals

Size signals ($SIZ[1:0]$) indicate the number of bytes remaining to be transferred during an operand cycle. They are valid while the address strobe (\overline{AS}) is asserted. Table 4–10 shows $SIZ0$ and $SIZ1$ encoding.

Table 4–10.
Size Signal Encoding

SIZ1	SIZ0	Transfer Size
0	1	Byte
1	0	Word
1	1	3 Byte
0	0	Long Word

4.4.1.7 Function Codes

Function code signals $FC[2:0]$ are generated by the CPU16. The function codes can be considered address extensions that designate which of eight external address spaces is accessed during a bus cycle.

Because the CPU16 always operates in supervisor mode ($FC2 = 1$), address spaces 0 to 3 are not used. Address space 7 is designated CPU space. CPU space is used for control information not normally associated with read or write bus cycles. Function codes are valid while \overline{AS} is asserted. Table 4–11 shows address space encoding.

**Table 4–11.
Address Space Encoding**

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

4.4.1.8 Data and Size Acknowledge Signals

During normal bus transfers, external devices assert the data and size acknowledge signals ($\overline{DSACK1}$ and $\overline{DSACK0}$) to indicate port width to the MCU. During a read cycle, these signals tell the MCU to terminate the bus cycle and to latch data. During a write cycle, the signals indicate that an external device has successfully stored data and that the cycle can terminate. $\overline{DSACK1}$ and $\overline{DSACK0}$ can also be supplied internally by chip-select logic. Refer to **4.8 Chip Selects** for more information.

4.4.1.9 Bus Error Signal

The bus error (\overline{BERR}) signal is asserted in the absence of \overline{DSACK} to indicate a bus error condition. \overline{BERR} can also be asserted with \overline{DSACK} to indicate a bus error condition, provided it meets the appropriate timing requirements. Refer to **4.5.5 Bus Exception Control Cycles** for more information.

The internal bus monitor can generate the \overline{BERR} signal for internal and internal-to-external transfers. An external bus master must provide its own \overline{BERR} generation and drive the \overline{BERR} pin, because the internal \overline{BERR} monitor has no information about transfers initiated by an external bus master. Refer to **4.5.6 External Bus Arbitration** for more information.

4.4.1.10 Halt Signal

The halt signal (\overline{HALT}) can be asserted by an external device to cause single bus cycle operation. Usually \overline{HALT} is used for debugging purposes. When \overline{BERR} and \overline{HALT} are asserted simultaneously, the MCU acts as though \overline{BERR} alone is asserted. Refer to **4.5.5 Bus Exception Control Cycles** for further information.

4.4.1.11 Autovector Signal

The autovector signal (\overline{AVEC}) can be used to terminate external interrupt acknowledge cycles. Assertion of \overline{AVEC} causes the CPU16 to generate vector numbers to locate an interrupt handler routine. If it is continuously asserted,

autovectors are generated for all external interrupt requests. $\overline{\text{AVEC}}$ is ignored during all other bus cycles. Refer to **4.7 Interrupts** for more information. $\overline{\text{AVEC}}$ for external interrupt requests can also be supplied internally by chip-select logic. Refer to **4.8 Chip Selects** for more information. The autovector function is disabled when there is an external bus master. Refer to **4.5.6 External Bus Arbitration** for more information.

4.4.2 Dynamic Bus Sizing

The MCU dynamically interprets the port size of an addressed device during each bus cycle, allowing operand transfers to or from 8-bit and 16-bit ports.

During an operand transfer cycle, an external device signals its port size and indicates completion of the bus cycle to the MCU through the use of the $\overline{\text{DSACK}}$ inputs, as shown in Table 4–12. Chip-select logic can generate data and size acknowledge signals for an external device. Refer to **4.8 Chip Selects** for further information.

Table 4–12. Effect of $\overline{\text{DSACK}}$ Signals

$\overline{\text{DSACK1}}$	$\overline{\text{DSACK0}}$	Result
1	1	Insert Wait States in Current Bus Cycle
1	0	Complete Cycle — Data Bus Port Size is 8 Bits
0	1	Complete Cycle — Data Bus Port Size is 16 Bits
0	0	Reserved

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

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

The MCU always attempts to transfer the maximum amount of data on all bus cycles. For a word operation, it is assumed that the port is 16 bits wide when the bus cycle begins.

Operand bytes are designated as shown in Figure 4–6. OP[0:3] represent the order of access. For instance, OP0 is the most significant byte of a long-word operand, and is accessed first, while OP3, the least significant byte, is accessed last. The two bytes of a word-length operand are OP0 (most significant) and OP1. The single byte of a byte-length operand is OP0.

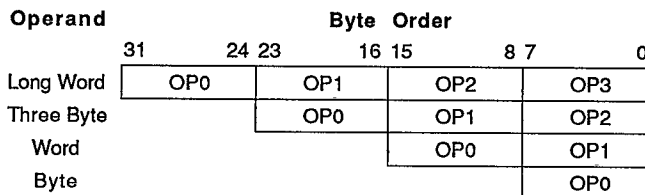


Figure 4–6. Operand Byte Order

4.4.3 Operand Alignment

The EBI 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. Note that ADDR[23:20] are driven to the same logic state as ADDR19.

4.4.4 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 through a 16-bit port, the most significant operand word is transferred on the first bus cycle and the least significant operand word is transferred on a following bus cycle.

The CPU16 can perform misaligned word transfers. This capability makes it compatible with the M68HC11 CPU. The CPU16 treats misaligned long-word transfers as two misaligned word transfers. Other modular microcontrollers have wider CPU architectures that support different long-word transfer cases.

4.4.5 Operand Transfer Cases

Table 4–13 is a summary of 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. The following paragraphs discuss all the allowable transfer cases in detail.

Table 4–13. Operand Alignment

Transfer Case	SIZ1	SIZ0	ADDR0	DSACK1	DSACK0	DATA [15:8]	DATA [7:0]
Byte to 8-Bit Port (Even/Odd)	0	1	X	1	0	OP0	(OP0)
Byte to 16-Bit Port (Even)	0	1	0	0	X	OP0	(OP0)
Byte to 16-Bit Port (Odd)	0	1	1	0	X	(OP0)	OP0
Word to 8-Bit Port (Aligned)	1	0	0	1	0	OP0	(OP1)
Word to 8-Bit Port (Misaligned)	1	0	1	1	0	OP0	(OP0)
Word to 16-Bit Port (Aligned)	1	0	0	0	X	OP0	OP1
Word to 16-Bit Port (Misaligned)	1	0	1	0	X	(OP0)	OP0
3 Byte to 8-Bit Port (Aligned) ³	1	1	0	1	0	OP0	(OP1)
3 Byte to 8-Bit Port (Misaligned) ³	1	1	1	1	0	OP0	(OP0)
3 Byte to 16-Bit Port (Aligned) ⁴	1	1	0	0	X	OP0	OP1
3 Byte to 16-Bit Port (Misaligned) ³	1	1	1	0	X	(OP0)	OP0
Long Word to 8-Bit Port (Aligned)	0	0	0	1	0	OP0	(OP1)
Long Word to 8-Bit Port (Misaligned) ⁴	1	0	1	1	0	OP0	(OP0)
Long Word to 16-Bit Port (Aligned)	0	0	0	0	X	OP0	OP1
Long Word to 16-Bit Port (Misaligned) ⁴	1	0	1	0	X	(OP0)	OP0

NOTES:

1. X in a column means that the state of the signal has no effect.
2. Operands in parentheses are ignored by the CPU16 during read cycles.
3. Three-byte transfer cases occur only as a result of an aligned long word to byte transfer.
4. The CPU16 treats misaligned long-word transfers as two misaligned word transfers.

4.5 Bus Operation

Internal microcontroller modules are typically accessed in two system clock cycles, with no wait states. Regular external bus cycles use handshaking between the MCU and external peripherals to manage transfer size and data. These accesses take three system clock cycles, again with no wait states. During regular cycles, wait states can be inserted as needed by bus control logic. Refer to **4.5.2 Regular Bus Cycles** for more information. Fast-termination cycles, which are two-cycle external accesses with no wait states, use chip-select logic to generate handshaking signals internally. Chip-select logic can also be used to insert wait states before internal generation of handshaking signals. Refer to **4.5.3 Fast Termination Cycles** and **4.8 Chip Selects** for more information. Bus control signal timing, as well as chip-select signal timing, are specified in **APPENDIX A ELECTRICAL CHARACTERISTICS**. Refer to the *SCIM Reference Manual (SCIMRM/AD)* for more information about each type of bus cycle.

The MCU is responsible for de-skewing signals it issues at both the start and the end of a cycle. In addition, the MCU is responsible for de-skewing acknowledge and data signals from peripheral devices.

4.5.1 Synchronization to CLKOUT

External devices connected to the MCU bus can operate at a clock frequency different from the frequencies of the MCU as long as the external devices satisfy the interface signal timing constraints. Although bus cycles are classified as asynchronous, they are interpreted relative to the MCU system clock output (CLKOUT).

Descriptions are made in terms of individual system clock states, labelled {S0, S1, S2, ..., SN} in the appropriate timing diagrams. The designation **state** refers to the logic level of the clock signal, and does not correspond to any implemented machine state. A clock cycle consists of two successive states. Refer to **APPENDIX A ELECTRICAL CHARACTERISTICS** for more information.

Bus cycles terminated by $\overline{\text{DSACK}}$ assertion normally require a minimum of three CLKOUT cycles. To support systems that use CLKOUT to generate $\overline{\text{DSACK}}$ and other inputs, asynchronous input setup time and asynchronous input hold times are specified. When these specifications are met, the MCU is guaranteed to recognize the appropriate signal on a specific edge of the CLKOUT signal.

For a read cycle, when assertion of $\overline{\text{DSACK}}$ is recognized on a particular falling edge of the clock, valid data is latched into the MCU on the next falling clock

edge, provided that the data meets the data setup time. In this case, the parameter for asynchronous operation can be ignored.

When a system asserts \overline{DSACK} for the required window around the falling edge of S2 and obeys the bus protocol by maintaining \overline{DSACK} and \overline{BERR} or \overline{HALT} until and throughout the clock edge that negates \overline{AS} (with the appropriate asynchronous input hold time), no wait states are inserted. The bus cycle runs at the maximum speed of three clocks per cycle.

To ensure proper operation in a system synchronized to CLKOUT, when either \overline{BERR} , or \overline{BERR} and \overline{HALT} , is asserted after \overline{DSACK} , \overline{BERR} (or \overline{BERR} and \overline{HALT}) assertion must satisfy the appropriate data-in setup and hold times before the falling edge of the clock cycle after \overline{DSACK} is recognized.

4.5.2 Regular Bus Cycles

The following paragraphs contain a discussion of cycles that use external bus control logic. Refer to **4.5.3 Fast Termination Cycles** for information about fast cycles.

To initiate a transfer, the MCU asserts an address and the SIZ[1:0] signals. The SIZ signals and ADDR0 are externally decoded to select the active portion of the data bus (refer to **4.4.2 Dynamic Bus Sizing**). When \overline{AS} , \overline{DS} , and R/\overline{W} are valid, a peripheral device either places data on the bus (read cycle) or latches data from the bus (write cycle), then asserts a $\overline{DSACK}[1:0]$ combination that indicates port size.

The $\overline{DSACK}[1:0]$ signals can be asserted before the data from a peripheral device is valid on a read cycle. To ensure valid data is latched into the MCU, a maximum period between \overline{DSACK} assertion and \overline{DS} assertion is specified.

There is no specified maximum for the period between the assertion of \overline{AS} and \overline{DSACK} . Although the MCU can transfer data in a minimum of three clock cycles when the cycle is terminated with \overline{DSACK} , the MCU inserts wait cycles in clock period increments until either \overline{DSACK} signal goes low.

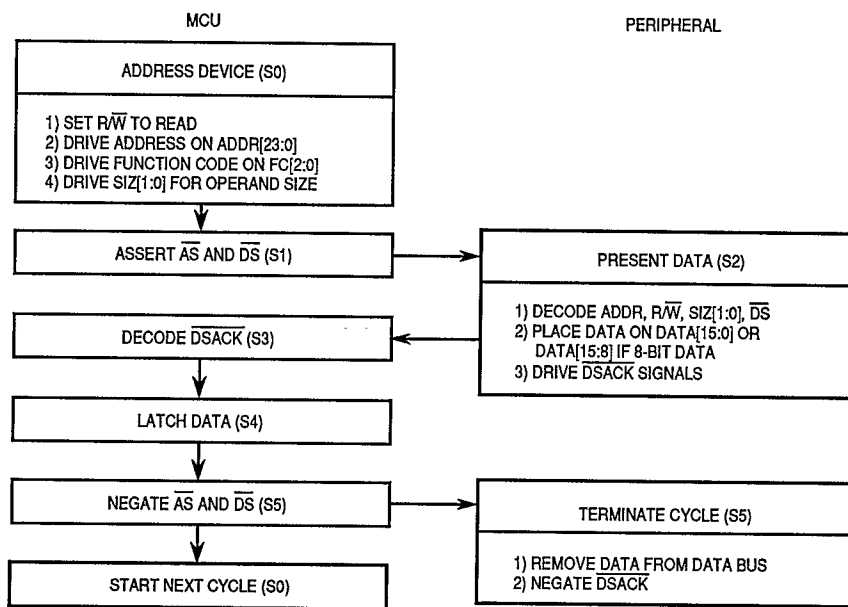
NOTE

The SCIM bus monitor asserts internal \overline{BERR} when response time exceeds a predetermined limit. Bus monitor period is determined by the BMT field in SYPCR. The bus monitor cannot be disabled; maximum monitor period is 64 system clock cycles.

If no peripheral responds to an access, or if an access is invalid, external logic should assert the $\overline{\text{BERR}}$ or $\overline{\text{HALT}}$ signals to abort the bus cycle (when $\overline{\text{BERR}}$ and $\overline{\text{HALT}}$ are asserted simultaneously, the CPU16 acts as though only $\overline{\text{BERR}}$ is asserted). If bus termination signals are not asserted within a specified period, the bus monitor terminates the cycle.

4.5.2.1 Read Cycle

During a read cycle, the MCU transfers data from an external memory or peripheral device. If the instruction specifies a long-word or word operation, the MCU attempts to read two bytes at once. For a byte operation, the MCU reads one byte. The portion of the data bus from which each byte is read depends on operand size, peripheral address, and peripheral port size. Figure 4–7 is a flowchart of a word read cycle. Refer to **4.4.2 Dynamic Bus Sizing**, **4.4.4 Misaligned Operands**, and the *SCIM Reference Manual* (SCIMRM/AD) for more information.

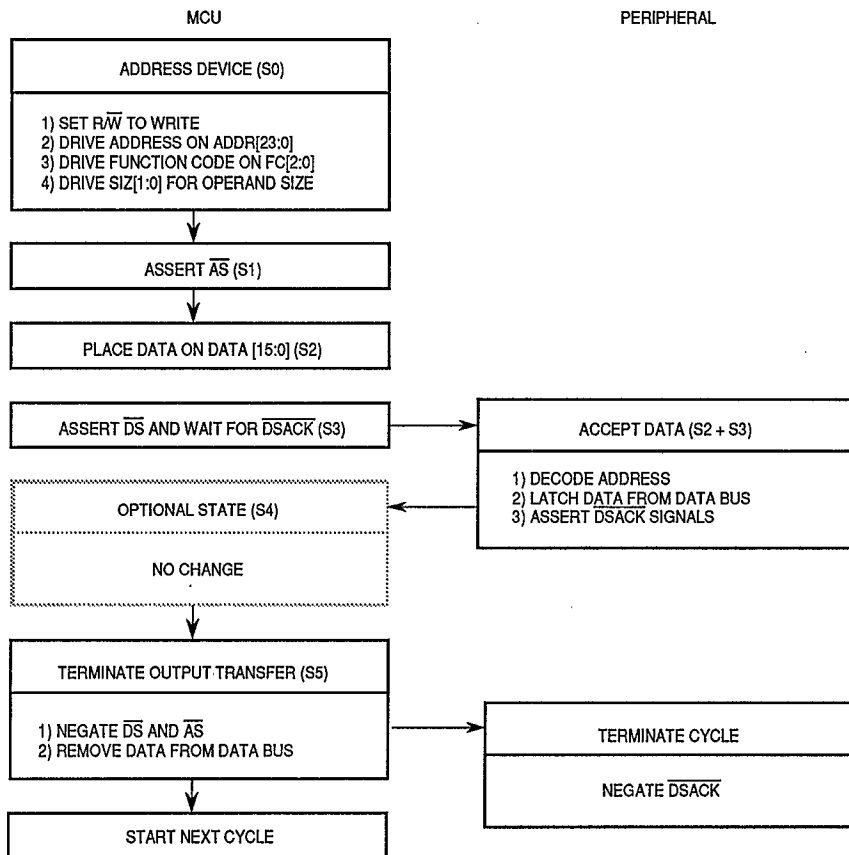


RD CYC FLOW

Figure 4–7. Word Read Cycle

4.5.2.2 Write Cycle

During a write cycle, the MCU transfers data to an external memory or peripheral device. If the instruction specifies a long-word or word operation, the MCU attempts to write two bytes at once. For a byte operation, the MCU writes one byte. The portion of the data bus upon which each byte is written depends on operand size, peripheral address, and peripheral port size. Refer to **4.4.2 Dynamic Bus Sizing**, **4.4.4 Misaligned Operands**, and the *SCIM Reference Manual* (SCIMRM/AD) for more information. Figure 4–8 is a flowchart of a write-cycle operation for a word transfer.



WR CYC FLOW

Figure 4–8. Write Cycle

4.5.3 Fast Termination Cycles

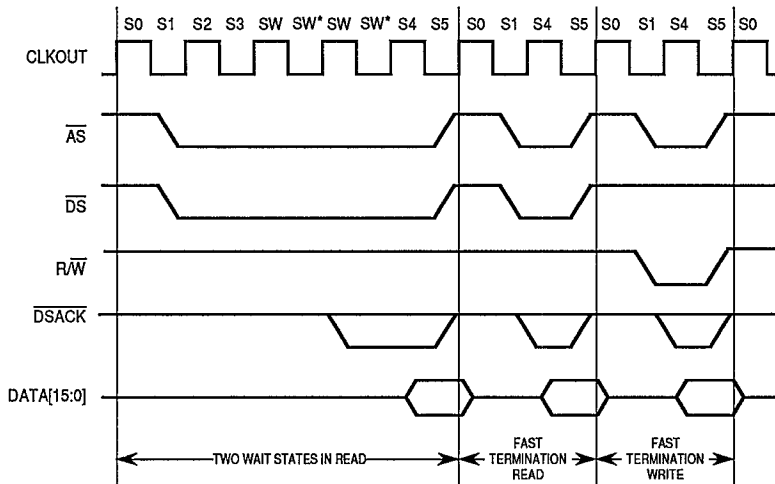
When an external device has a fast access time, the chip-select circuit fast-termination option can provide a two-cycle external bus transfer. Because the chip-select circuits are driven from the system clock, the bus cycle termination is inherently synchronized with the system clock.

Fast termination cycles use internal handshaking signals generated by the chip-select logic. To initiate a transfer, the MCU asserts an address and the $SIZ[1:0]$ signals. When \overline{AS} , \overline{DS} , and R/\overline{W} are valid, a peripheral device either places data on the bus (read cycle) or latches data from the bus (write cycle). At the appropriate time, chip-select logic asserts data and size acknowledge signals.

The \overline{DSACK} option fields in the chip-select option registers determine whether internally generated \overline{DSACK} or externally generated \overline{DSACK} are used. Refer to **4.8.1 Chip-Select Registers** for information about fast-termination setup.

To use fast-termination, an external device must be fast enough to have data ready, within the specified setup time, by the falling edge of $S4$. Figure 4-9 shows the \overline{DSACK} timing for two wait states in read, and a fast-termination read and write.

When fast termination is in use, \overline{DS} is asserted during read cycles but not during write cycles. The $STRB$ field in the chip-select option register used must be programmed with the address strobe encoding to assert the chip select signal for a fast-termination write.



* DSACK ONLY INTERNALLY ASSERTED FOR FAST TERMINATION

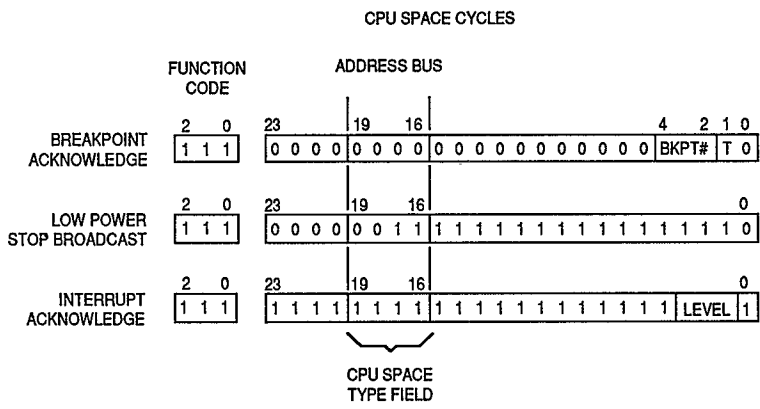
FAST TERM TIM

Figure 4–9. Fast-Termination Timing

4.5.4 CPU Space Cycles

Function code signals FC[2:0] designate which of eight external address spaces is accessed during a bus cycle. 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 only while AS is asserted. Refer to **4.4.1.7 Function Codes** for more information on codes and encoding.

During a CPU space access, ADDR[19:16] are encoded to reflect the type of access being made. Three encodings are used by the M68HC16 devices, as shown in Figure 4–10. These encodings represent breakpoint acknowledge (Type \$0) cycles, low power stop broadcast (Type \$3) cycles, and interrupt acknowledge (Type \$F) cycles. Refer to **4.7 Interrupts** for information about interrupt acknowledge bus cycles.



CPU SPACE CYC TIM

Figure 4–10. CPU Space Address Encoding

4.5.4.1 Breakpoint Acknowledge Cycle

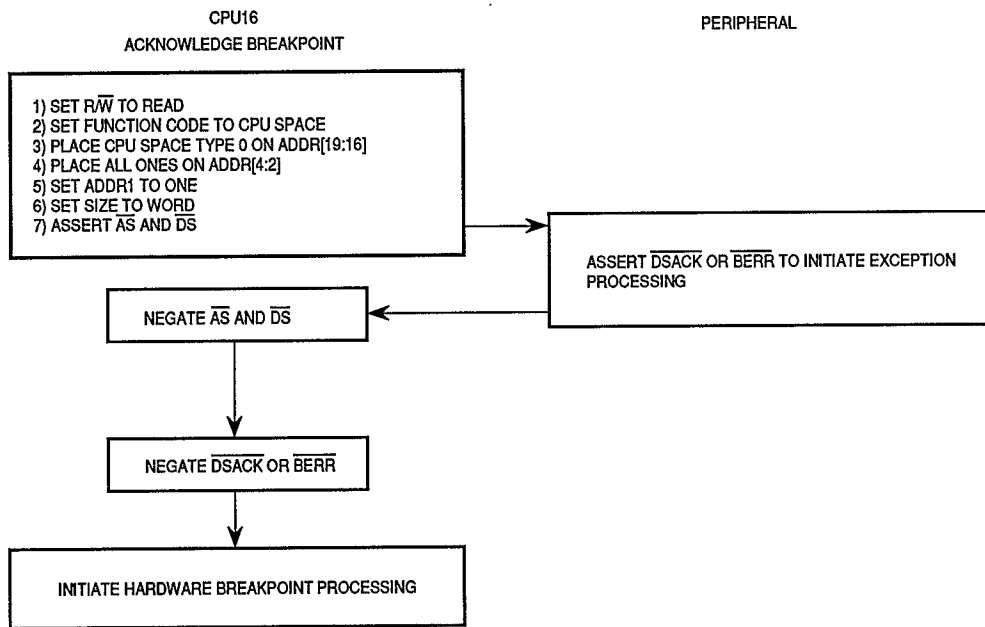
Breakpoints stop program execution at a predefined point during system development. Breakpoints are handled as a type of exception. Breakpoints can be used alone or in conjunction with the background debugging mode. Refer to **SECTION 5 CENTRAL PROCESSING UNIT** for more information on exception processing and the background debugging mode.

A hardware breakpoint is initiated by assertion of the \overline{BKPT} input. M68HC16 devices do not support breakpoints initiated by software. The breakpoint acknowledge cycle discussed here is the bus cycle that occurs as a part of breakpoint exception processing when a breakpoint is initiated while background debugging mode is not enabled.

\overline{BKPT} is sampled on the same clock phase as data. If \overline{BKPT} is valid, the data is tagged as it enters the CPU pipeline. When \overline{BKPT} is asserted while data is valid during an instruction prefetch, the acknowledge cycle occurs immediately after that instruction has executed. When \overline{BKPT} is asserted while data is valid during an operand fetch, the acknowledge cycle occurs immediately after execution of the instruction during which it is latched. If \overline{BKPT} is asserted for only one bus cycle and a pipe flush occurs before \overline{BKPT} is detected by the CPU, no acknowledge cycle occurs. To ensure detection, \overline{BKPT} can be asserted until a breakpoint acknowledge cycle is recognized.

When $\overline{\text{BKPT}}$ assertion is acknowledged by the CPU, the SCIM performs a word read from CPU space address \$00001E. This corresponds to the breakpoint number field (ADDR[4:2]) and the type bit (T) being set to all ones (source 7, type 1). If this bus cycle is terminated by $\overline{\text{BERR}}$ or by $\overline{\text{DSACK}}$, the MCU performs breakpoint exception processing. Refer to Figure 4–11 for a flowchart of the breakpoint operation.

Refer to the *SCIM Reference Manual (SCIMRM/AD)* for further information.



CPU16 BKPT FLOW

Figure 4–11. Breakpoint Operation

4.5.4.2 LPSTOP Broadcast Cycle

Low-power stop is initiated by the CPU16. Individual modules can be stopped by setting the STOP bits in each module configuration register, or the SCIM can turn off system clocks after execution of the LPSTOP instruction. When the CPU executes LPSTOP, the LPSTOP broadcast cycle is generated. The SCIM brings the MCU out of low-power mode when either an interrupt of higher priority than the stored mask or a reset occurs. Refer to **4.2.11 Monitor Low-Power Operation**, **4.3.4 Clock Low-Power Operation** and **SECTION 5 CENTRAL PROCESSING UNIT** for more information.

During an LPSTOP broadcast cycle, the CPU performs a CPU space write to address \$3FFFE. This write puts a copy of the interrupt mask value in the clock control logic. The mask is encoded on the data bus, as shown in Figure 4-12.

The LPSTOP CPU space cycle is shown externally, if the bus is available, as an indication to external devices that the MCU is going into low-power stop mode. The SCIM provides an internally generated \overline{DSACK} response to this cycle. The timing of this bus cycle is the same as for a fast write cycle.

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	IP MASK

Figure 4-12. LPSTOP Interrupt Mask Level

4.5.5 Bus Exception Control Cycles

An external device or a chip-select circuit must assert at least one of the $\overline{DSACK}[1:0]$ signals or the \overline{AVEC} signal to terminate an external bus cycle normally (refer to **4.4.1 Bus Signals**). Bus exception control cycles are used when bus cycles are not terminated in the expected manner. There are two sources of bus exception control cycles, \overline{BERR} and \overline{HALT} .

When neither \overline{DSACK} nor \overline{AVEC} is asserted within a specified period after assertion of \overline{AS} , the internal bus monitor asserts internal \overline{BERR} .

The spurious interrupt monitor asserts internal \overline{BERR} when an interrupt request is acknowledged and no IARB contention occurs. \overline{BERR} assertion terminates a cycle and causes the MCU to process a bus error exception. External devices can assert \overline{BERR} to indicate an external bus error.

\overline{HALT} can be asserted by an external device to cause single bus cycle operation. \overline{HALT} is typically used for debugging purposes.

To control termination of a bus cycle for a bus error condition properly, \overline{DSACK} , \overline{BERR} , and \overline{HALT} must be asserted and negated synchronously with the rising edge of CLKOUT. This ensures that setup time and hold time requirements are met for the same falling edge of the MCU clock when two signals are asserted simultaneously. (For further information refer to **APPENDIX A ELECTRICAL CHARACTERISTICS**.) External circuitry that provides these signals must be designed with these constraints in mind, or the internal bus monitor must be used.

The acceptable bus cycle terminations for asynchronous cycles in relation to \overline{DSACK} assertion are summarized in Table 4–14.

Table 4–14. \overline{DSACK} , \overline{BERR} , and \overline{HALT} Assertion Results

Type of Termination	Control Signal	Asserted on Rising Edge of State		Description of Result
		S	S+2	
Normal	\overline{DSACK} \overline{BERR} \overline{HALT}	A NA NA	RA NA X	Normal cycle terminate and continue.
Halt	\overline{DSACK} \overline{BERR} \overline{HALT}	A NA A/RA	RA NA RA	Normal cycle terminate and halt. Continue when \overline{HALT} is negated.
Bus Error 1	\overline{DSACK} \overline{BERR} \overline{HALT}	NA/A A NA	X RA X	Terminate and take bus error exception.
Bus Error 2	\overline{DSACK} \overline{BERR} \overline{HALT}	A A NA	X RA NA	Terminate and take bus error exception.
Bus Error 3	\overline{DSACK} \overline{BERR} \overline{HALT}	NA/A A A/S	X RA RA	Terminate and take bus error exception.
Bus Error 4	\overline{DSACK} \overline{BERR} \overline{HALT}	A NA NA	X A A	Terminate and take bus error exception.

NOTES:

- A = Signal is asserted in this bus state.
- NA = Signal is not asserted in this state.
- RA = Signal was asserted in previous state and remains asserted in this state.
- S = The number of current even bus state (e.g., S2, S4, etc.)

4.5.5.1 Bus Errors

The CPU16 treats bus errors as a type of exception. Bus error exception processing begins when the CPU detects assertion of the IMB $\overline{\text{BERR}}$ signal.

$\overline{\text{BERR}}$ assertion does not force immediate exception processing. The signal is synchronized with normal bus cycles and is latched into the CPU16 at the end of the bus cycle in which it was asserted. Because bus cycles can overlap instruction boundaries, bus error exception processing may not occur at the end of the instruction in which the bus cycle begins. Timing of $\overline{\text{BERR}}$ detection/acknowledge is dependent upon several factors:

Which bus cycle of an instruction is terminated by assertion of $\overline{\text{BERR}}$

The number of bus cycles in the instruction during which $\overline{\text{BERR}}$ is asserted

The number of bus cycles in the instruction following the instruction in which $\overline{\text{BERR}}$ is asserted

Whether $\overline{\text{BERR}}$ is asserted during a program space access or a data space access

Because of these factors, it is impossible to predict precisely how long after occurrence of a bus error the bus error exception is processed.

CAUTION

The external bus interface does not latch data when an external bus cycle is terminated by a bus error. When this occurs during an instruction prefetch, the IMB precharge state (bus pulled high, or \$FF) is latched into the CPU16 instruction register, with indeterminate results.

4.5.5.2 Double Bus Fault

Exception processing for bus error exceptions follows the standard exception processing sequence. Refer to **SECTION 5 CENTRAL PROCESSING UNIT** for more information about exceptions. However, two special cases of bus error, called double bus faults, can abort exception processing.

$\overline{\text{BERR}}$ assertion is not detected until an instruction is complete. The $\overline{\text{BERR}}$ latch is cleared by the first instruction of the $\overline{\text{BERR}}$ exception handler. Double bus fault occurs in two ways:

1. When bus error exception processing begins and a second $\overline{\text{BERR}}$ is detected before the first instruction of the first exception handler is executed.
2. When one or more bus errors occur before the first instruction after a reset exception is executed.

Multiple bus errors within a single instruction that can generate multiple bus cycles cause a single bus error exception after the instruction has been executed.

Immediately after $\overline{\text{BERR}}$ is asserted a second time, the MCU halts and drives the $\overline{\text{HALT}}$ line low. Only a reset can restart a halted MCU. However, bus arbitration can still occur (refer to **4.5.6 External Bus Arbitration**). A bus error or address error that occurs after exception processing has been completed (during the execution of the exception handler routine, or later) does not cause a double bus fault.

4.5.5.3 Halt Operation

When $\overline{\text{HALT}}$ is asserted while $\overline{\text{BERR}}$ is not asserted, the MCU halts external bus activity after negation of $\overline{\text{DSACK}}$. The MCU may complete the current word transfer in progress. For a long-word to byte transfer, this could be after S2 or S4. For a word to byte transfer, activity ceases after S2.

Negating and reasserting $\overline{\text{HALT}}$ according to timing requirements provides single-step (bus cycle to bus cycle) operation. The $\overline{\text{HALT}}$ signal affects external bus cycles only, so that a program that does not use external bus can continue executing. During dynamically-sized 8-bit transfers, external bus activity may not stop at the next cycle boundary. Occurrence of a bus error while $\overline{\text{HALT}}$ is asserted causes the CPU16 to process a bus error exception.

When the MCU completes a bus cycle while the $\overline{\text{HALT}}$ signal is asserted, the data bus goes to high-impedance state and the AS and DS signals are driven to their inactive states. Address, function code, size, and read/write signals remain in the same state.

The halt operation has no effect on bus arbitration (refer to **4.5.6 External Bus Arbitration**). However, when external bus arbitration occurs while the MCU is halted, address and control signals go to high-impedance state. If $\overline{\text{HALT}}$ is still asserted when the MCU regains control of the bus, address, function code, size, and read/write signals revert to the previous driven states. The MCU cannot service interrupt requests while halted.

4.5.6 External Bus Arbitration

MCU bus design provides for a single bus master at any one time. When the MCU is not configured for single-chip operation either the MCU or an external device can be master. Bus arbitration protocols determine when an external device can become bus master. Bus arbitration requests are recognized during normal processing, $\overline{\text{HALT}}$ assertion, and when the CPU has halted because of a double bus fault.

4

The bus controller in the MCU manages bus arbitration signals so that the MCU has the lowest priority. External devices that need to obtain the bus must assert bus arbitration signals in the sequences described in the following paragraphs.

Systems that include several devices that can become bus master require external circuitry to assign priorities to the devices, so that when two or more external devices attempt to become bus master at the same time, the one having the highest priority becomes bus master first. The protocol sequence is:

- A. An external device asserts the bus request signal ($\overline{\text{BR}}$);
- B. The MCU asserts the bus grant signal ($\overline{\text{BG}}$) to indicate that the bus is available;
- C. An external device asserts the bus grant acknowledge ($\overline{\text{BGACK}}$) signal to indicate that it has assumed bus mastership.

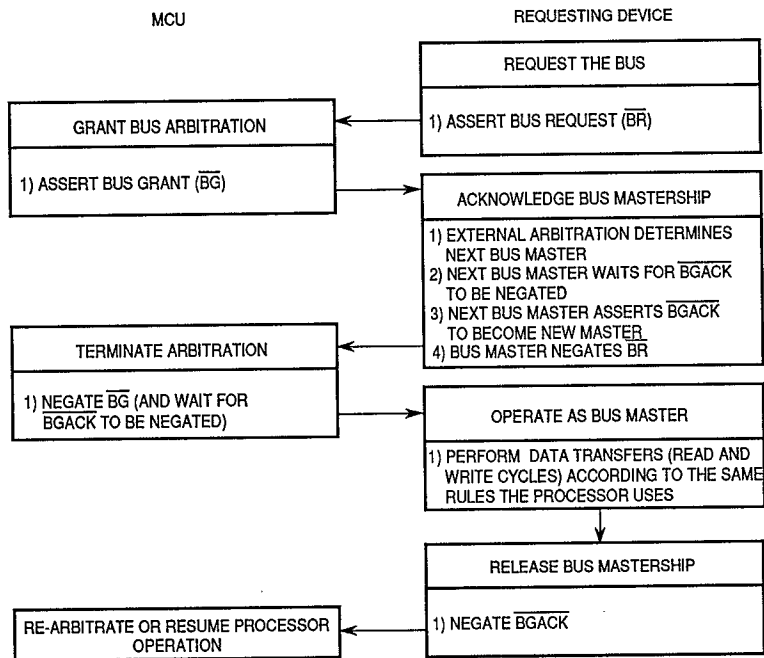
$\overline{\text{BR}}$ can be asserted during a bus cycle or between cycles. $\overline{\text{BG}}$ is asserted in response to $\overline{\text{BR}}$. To guarantee operand coherency, $\overline{\text{BG}}$ is only asserted at the end of operand transfer.

If more than one external device can be bus master, required external arbitration must begin when a requesting device receives $\overline{\text{BG}}$. An external device must assert $\overline{\text{BGACK}}$ when it assumes mastership, and must maintain $\overline{\text{BGACK}}$ assertion as long as it is bus master.

Two conditions must be met for an external device to assume bus mastership. The device must receive $\overline{\text{BG}}$ through the arbitration process, and $\overline{\text{BGACK}}$ must be inactive, indicating that no other bus master is active. This technique allows the processing of bus requests during data transfer cycles.

$\overline{\text{BG}}$ is negated a few clock cycles after $\overline{\text{BGACK}}$ transition. However, if bus requests are still pending after $\overline{\text{BG}}$ is negated, the MCU asserts $\overline{\text{BG}}$ again within a few clock cycles. This additional $\overline{\text{BG}}$ assertion allows external arbitration circuitry to select the next bus master before the current master has released the bus.

Refer to Figure 4-13, which shows bus arbitration for a single device. The flowchart shows $\overline{\text{BR}}$ negated at the same time $\overline{\text{BGACK}}$ is asserted.



BUS ARB FLOW

Figure 4–13. Bus Arbitration for Single Request

State changes occur on the next rising edge of CLKOUT after the internal signal is valid. The BG signal transitions on the falling edge of the clock after a state is reached during which G changes. The bus control signals (controlled by T) are driven by the MCU immediately following a state change, when bus mastership is returned to the MCU. State 0, in which G and T are both negated, is the state of the bus arbiter while the MCU is bus master. Request (R) and acknowledge (A) keep the arbiter in state 0 as long as they are both negated.

4.5.6.1 Slave (Factory Test) Mode Arbitration

This mode is used for factory production testing of internal modules. It is not supported as a user operating mode. Slave mode is enabled by holding DATA11 low during reset. In slave mode, when BG is asserted, the MCU is slaved to an external master that has full access to all internal registers.

4.5.6.2 Show Cycles

The MCU normally performs internal data transfers without affecting the external bus, but it is possible to show these transfers during debugging, provided the MCU is not configured for single-chip operation. \overline{AS} is not asserted externally during show cycles.

Show cycles are controlled by the SHEN field in the SCIMCR (refer to 4.2.3 **Show Internal Cycles**). This field is cleared by reset. When show cycles are disabled, the address bus, function codes, size, and read/write signals reflect internal bus activity, but \overline{AS} and \overline{DS} are not asserted externally and external data bus pins are in high-impedance state during internal accesses.

When show cycles are enabled, \overline{DS} is asserted externally during internal cycles, and internal data is driven out on the external data bus. Because internal cycles normally continue to run when the external bus is granted, one SHEN encoding halts internal bus activity while there is an external master.

SIZ[1:0] signals reflect bus allocation during show cycles. Only the appropriate portion of the data bus is valid during the cycle. During a byte write to an internal address, the portion of the bus that represents the byte that is not written reflects internal bus conditions, and is indeterminate. During a byte write to an external address, the data multiplexer in the SCIM causes the value of the byte that is written to be driven out on both bytes of the data bus.

4.6 Reset

Reset occurs when an active low logic level on the \overline{RESET} pin is clocked into the SCIM. The \overline{RESET} input is synchronized to the system clock. 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 procedures handle system initialization and recovery from catastrophic failure. Resets are performed by a combination of hardware and software. The single-chip integration module 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.

4.6.1 Reset Exception Processing

The CPU16 processes resets as a type of asynchronous exception. An exception is an event that preempts normal processing. 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. These vectors are stored in a vector table

located in the first 512 bytes of address bank 0. The CPU16 uses vector numbers to calculate displacement into the table. For more information concerning exceptions refer to **SECTION 5 CENTRAL PROCESSING UNIT**.

Reset is the highest-priority CPU16 exception. Unlike all other exceptions, a reset occurs at the end of a bus cycle, and not at an instruction boundary. Handling resets in this way prevents write cycles in progress at the time the reset signal is asserted from being corrupted. However, any processing in progress is aborted by the reset exception, and cannot be restarted. Only essential reset tasks are performed during exception processing. Other initialization tasks must be accomplished by the exception handler routine.

4.6.9 Reset Processing Summary contains details of exception processing.

4.6.2 Reset Control Logic

SCIM reset control logic determines the cause of a reset, synchronizes request signals to CLKOUT, and asserts reset control signals. Reset control logic can drive three different internal signals.

EXTRST (external reset) drives the external reset pin.

CLKRST (clock reset) resets the clock module.

MSTRST (master reset) goes to all other internal circuits.

All resets are gated by CLKOUT. Asynchronous resets are assumed to be catastrophic. An asynchronous reset can occur on any clock edge. Synchronous resets are timed (CLKOUT) to occur at the end of bus cycles. The internal bus monitor is automatically enabled for synchronous resets. When a bus cycle does not terminate normally, the bus monitor terminates it. Table 4–15 is a summary of reset sources.

Table 4–15. Reset Source Summary

Type	Source	Timing	Reset Lines Asserted by Controller		
External	External	Synchronous	MSTRST	CLKRST	EXTRST
Power Up	EBI	Asynchronous	MSTRST	CLKRST	EXTRST
Software Watchdog	Monitor	Asynchronous	MSTRST	CLKRST	EXTRST
Halt	Monitor	Asynchronous	MSTRST	CLKRST	EXTRST
Loss Of Clock	Clock	Synchronous	MSTRST	CLKRST	EXTRST
Test	Test	Synchronous	MSTRST	—	EXTRST

Internal single byte or aligned word writes are guaranteed valid for synchronous resets. External writes are also guaranteed to complete, provided the external configuration logic on the data bus is conditioned as shown in Figure 4–28.

4.6.3 Operating Configuration out of Reset

The logic states of certain pins during reset determine SCIM operating mode and configuration. During reset, the SCIM reads pin configuration from DATA[11:0], internal module configuration from DATA[15:12], and basic operating information from $\overline{\text{BERR}}$, MODCLK, and $\overline{\text{BKPT}}$.

Data bus pins are normally pulled high internally during reset, causing the MCU to default to a specific configuration. However, the user can drive selected pins low during reset to achieve alternate configurations. $\overline{\text{BERR}}$, $\overline{\text{BKPT}}$, and MODCLK do not have internal pull-ups and must be driven to the desired state during reset.

Basic operating options include system clock selection, background mode disable/enable, and external bus configuration. The SCIM supports fully-expanded operation with an M68020 style 24-bit address bus and 16-bit data bus with chip selects, partially-expanded operation with a 24-bit address bus and an 8-bit external data bus, and single-chip operation with no external address and data bus. Refer to Table 4–16.

Table 4–16. Basic Configuration Options

Select Pin	Pin Held High	Pin Held 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

When $\overline{\text{BERR}}$ is high during reset, the MCU is configured for partially-expanded or fully-expanded operation. DATA2 is then decoded to select 8- or 16-bit data bus operation, DATA8 is decoded to configure pins for bus control or port E operation, and DATA9 is decoded to configure pins for clock selection and interrupt request or port F operation. If DATA2 is held low at reset selecting 16-bit data bus operation, DATA11, DATA10, DATA[7:2] and DATA0 are also decoded. The following subsections explain the process in greater detail.

4.6.3.1 Address and Data Bus Pin Functions

External bus configuration determines whether certain address and data pins are used for those functions or for general-purpose I/O. ADDR[18:3] serve as pins for ports A and B when the MCU is operating in single-chip mode. DATA[7:0] serve as port H pins in partially-expanded and single-chip modes, and DATA[15:8] serve as port G pin during single-chip operation. Refer to Table 4–17.

Table 4–17. Bus and I/O Port Pin Functions

Bus Configuration	Mode-Select Pins		Bus and Port Distribution		
	$\overline{\text{BERR}}$	DATA1	Address Bus Pins	Data Bus Pins	I/O Ports
16-Bit Expanded	1	0	ADDR[18:3]	DATA[15:0]	—
8-Bit Expanded	1	1	ADDR[18:3]	DATA[15:8]	DATA[7:0] = Port H
Single Chip	0	X	None	None	ADDR[18:11] = Port A ADDR[10:3] = Port B DATA[15:8] = Port G DATA[7:0] = Port H

The ABD bit in the SCIM configuration register controls ADDR[2:0] function. When ABD is set, the pins operate normally; when ABD is cleared, the pins are placed in a high-impedance state. The reset state of ABD in expanded modes is zero; in single-chip mode the reset state is one. ABD can be written once after reset. Refer to **APPENDIX D REGISTER SUMMARY** for more information.

Because ADDR[23:20] are driven to the same state as ADDR19 their use as address bus pins is limited; however, ADDR[23:19] can also be used as chip-select or discrete output pins, depending on the external bus configuration selected at reset. The following paragraphs contain a summary of pin configuration options for each external bus configuration.

4.6.3.2 Configuration for 16-Bit Data Bus Operation

16-bit data bus operation is selected when $\overline{\text{BERR}} = 1$ and DATA1 = 0 during reset. In this configuration, 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. ADDR[23:20] can be configured as chip selects or address bus pins, as shown in the following table. ADDR[2:0] are configured as address bus pins.

DATA2 determines the functions of $\overline{\text{CS0/BR}}$, $\overline{\text{CS3/FC0}}$, and $\overline{\text{CS5/FC2}}$.

DATA[7:3] determine the functions of ADDR[23:19]/ $\overline{\text{CS[10:6]}}$. A data bus pin pulled low selects the associated chip select and all lower-numbered chip-

selects down through $\overline{CS6}$. For example, if DATA5 is pulled low during reset, CS[8:6] are configured as address bus signals ADDR[21:19], and CS[10:9] are configured as chip selects. ADDR[23:20] are driven to the same logic state as ADDR19, and DATA[7:4] have limited use. Refer to **4.8.4 Chip-Select Reset Operation** for more information.

DATA8 determines the function of the $\overline{DSACK0}$, $\overline{DSACK1}$, \overline{AVEC} , \overline{DS} , \overline{AS} , and SIZ[1:0] pins. If DATA8 is held low during reset, these pins are used for discrete I/O (port E).

DATA9 determines the function of interrupt request pins ($\overline{IRQ[7:0]}$) and the clock mode select pin (MODCLK). When DATA9 is held low during reset, these pins are used for discrete I/O (port F). MODCLK functions during reset regardless of DATA9 logic state.

DATA11 determines whether the SCIM operates in test mode out of reset. This capability is used for factory testing of the MCU.

DATA0 determines the port size of the boot ROM chip-select signal \overline{CSBOOT} . Unlike other chip-select signals, \overline{CSBOOT} is active at the release of reset. When DATA0 is held low, port size is 8 bits; when DATA0 is held high, either by the weak internal pull-up driver or by an external pull-up, port size is 16 bits. Refer to **4.8.4 Chip-Select Reset Operation** for more information. Refer to Table 4-18.

Table 4–18. 16-Bit Data Bus Mode Reset Configuration

Pin(s) Affected	Select Pin	Default Function (Pin Left High)	Alternate Function (Pin Pulled Low)
$\overline{\text{CSBOOT}}$	DATA0	$\overline{\text{CSBOOT}}$ 16-Bit	$\overline{\text{CSBOOT}}$ 8-Bit
$\overline{\text{BR/CS0}}$ $\overline{\text{FC0/CS3}}$ $\overline{\text{FC1/PC1}}$ $\overline{\text{FC2/CS5/PC2}}$	DATA2	$\overline{\text{CS0}}$ $\overline{\text{CS3}}$ $\overline{\text{FC1}}$ $\overline{\text{CS5}}$	BR FC0 FC1 FC2
$\overline{\text{ADDR19/CS6/PC3}}$ $\overline{\text{ADDR20/CS7/PC4}}$ $\overline{\text{ADDR21/CS8/PC5}}$ $\overline{\text{ADDR22/CS9/PC6}}$ $\overline{\text{ADDR23/CS10/ECLK}}$	DATA3 DATA4 DATA5 DATA6 DATA7	$\overline{\text{CS6}}$ $\overline{\text{CS[7:6]}}$ $\overline{\text{CS[8:6]}}$ $\overline{\text{CS[9:6]}}$ $\overline{\text{CS[10:6]}}$	ADDR19 ADDR[20:19] ADDR[21:19] ADDR[22:19] ADDR[23:19]
$\overline{\text{DSACK0/PE0}}$ $\overline{\text{DSACK1/PE1}}$ $\overline{\text{AVEC/PE2}}$ PE3 $\overline{\text{DS/PE4}}$ $\overline{\text{AS/PE5}}$ $\overline{\text{SIZ0/PE6}}$ $\overline{\text{SIZ1/PE7}}$	DATA8	$\overline{\text{DSACK0}}$ $\overline{\text{DSACK1}}$ AVEC PE3 $\overline{\text{DS}}$ $\overline{\text{AS}}$ $\overline{\text{SIZ0}}$ SIZ1	PE0 PE1 PE2 PE3 PE4 PE5 PE6 PE7
$\overline{\text{MODCLK/PF0}}$ $\overline{\text{IRQ[7:1]/PF[7:1]}}$	DATA9	MODCLK IRQ[7:1]	PF0 PF[7:1]
$\overline{\text{BGACK/CSE}}$ $\overline{\text{BG/CSM}}$	DATA10	$\overline{\text{BGACK}}$ BG	$\overline{\text{CSE}}^1$ CSM ²
DATA11	DATA11	Slave Mode Disabled ³	Slave Mode Enabled ³
DATA14	DATA14	ROM Enabled (ROMMCR STOP = 0)	ROM Disabled (ROMMCR STOP = 1)

Notes:

1. $\overline{\text{CSE}}$ is enabled when DATA10 and DATA1 = 0 during reset.
2. CSM is enabled when DATA13, DATA10 and DATA1 = 0 during reset (DB14 must equal 1 to enable ROM).
3. Slave mode is used for factory testing only.

4.6.3.3 Configuration for 8-Bit Data Bus Operation

The SCIM uses an 8-bit data bus when $\overline{\text{BERR}} = 1$ and DATA1 = 1 during reset. In this configuration, pins DATA[7:0] are configured as port H, an 8-bit I/O port. Pins DATA[15:8] are configured as data bus pins, and ADDR[18:3] are configured as address bus pins. The alternate functions for these address and data bus pins as ports A, B, and G are unavailable. ADDR[23:19]/CS[10:6] are configured as chip selects. ADDR[2:0] are configured as address bus pins. Emulator mode is always disabled.

DATA8 determines the function of the $\overline{\text{DSACK0}}$, $\overline{\text{DSACK1}}$, $\overline{\text{AVEC}}$, $\overline{\text{DS}}$, $\overline{\text{AS}}$, and SIZ[1:0] pins. If DATA8 is held low during reset, these pins are used for discrete I/O (port E).

DATA9 determines the function of interrupt request pins ($\overline{\text{IRQ[7:0]}}$) and the clock mode select pin (MODCLK). When DATA9 is held low during reset, these pins

are used for discrete I/O (port F). MODCLK functions during reset regardless of DATA9 logic state.

When DATA14 is driven low during reset, ROM is placed in STOP mode.

Table 4–19. 8-Bit Expanded Mode Reset Configuration

Pin(s) Affected	Select Pin	Default Function (Pin Left High)	Alternate Function (Pin Pulled Low)
$\overline{\text{CSBOOT}}$	N/A ¹	$\overline{\text{CSBOOT}}$ 8-Bit	$\overline{\text{CSBOOT}}$ 8-Bit
$\overline{\text{BF/CS0}}$ FC0/CS3/PC0 FC1/PC1 FC2/CS5/PC2	N/A ¹	$\overline{\text{CS0}}$ CS3 FC1 CS5	$\overline{\text{CS0}}$ CS3 FC1 CS5
ADDR19/ $\overline{\text{CS6/PC3}}$ ADDR20/ $\overline{\text{CS7/PC4}}$ ADDR21/ $\overline{\text{CS8/PC5}}$ ADDR22/ $\overline{\text{CS9/PC6}}$ ADDR23/ $\overline{\text{CS10/ECLK}}$	N/A ¹	CS[10:6]	CS[10:6]
$\overline{\text{DSACK0/PE0}}$ $\overline{\text{DSACK1/PE1}}$ AVEC/PE2 PE3 $\overline{\text{DS/PE4}}$ AS/PE5 $\overline{\text{SIZ0/PE6}}$ $\overline{\text{SIZ1/PE7}}$	DATA8	$\overline{\text{DSACK0}}$ $\overline{\text{DSACK1}}$ AVEC PE3 DS AS $\overline{\text{SIZ0}}$ $\overline{\text{SIZ1}}$	PE0 PE1 PE2 PE3 PE4 PE5 PE6 PE7
$\overline{\text{MODCLK/PF0}}$ $\overline{\text{IRQ[7:1]/PF[7:1]}}$	DATA9	MODCLK $\overline{\text{IRQ[7:1]}}$	PF0 PF[7:1]
$\overline{\text{BGACK/CSE}}$ BG/CSM	N/A ¹	$\overline{\text{BGACK}}$ BG	$\overline{\text{BGACK}}$ BG
DATA14	DATA14	ROM Enabled (ROMMCR STOP = 0)	ROM Disabled (ROMMCR STOP = 1)

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

4.6.3.4 Configuration for Single-Chip Operation

Single-chip operation is selected when $\overline{\text{BERR}} = 0$ during reset. $\overline{\text{BERR}}$ can be tied low permanently to select this configuration. In single-chip configuration, pins DATA[15:0] are configured as two 8-bit I/O ports, ports G and H. ADDR[18:3] are configured as two 8-bit I/O ports, ports A and B. There is no external data bus path. Expanded mode configuration options are not available: I/O ports A, B, C, E, F, G, and H are always selected. ADDR[2:0] come out of reset in a high-impedance state. After reset, clearing the ABD bit in the SCIM configuration register enables these pins, and leaving the bit set (its single-chip reset state) leaves the pins in a disabled (high-impedance) state. For more information, refer to **APPENDIX D REGISTER SUMMARY** for a discussion of the ABD bit in the SCIM configuration register.

Table 4–20 is a summary of SCIM pin function during single-chip operation.

Table 4–20. Single-Chip Mode Reset Configuration

Pin(s) Affected	Function
$\overline{\text{CSBOOT}}$	$\overline{\text{CSBOOT}}$ 8-Bit
ADDR[18:10]	PA[7:0]
ADDR[9:3]	PB[7:0]
$\text{BR}/\overline{\text{CS0}}$	$\overline{\text{CS0}}$
FC0/ $\overline{\text{CS3}}$ / $\overline{\text{PC0}}$ FC1/ $\overline{\text{PC1}}$ FC2/ $\overline{\text{CS5}}$ / $\overline{\text{PC2}}$ ADDR19/ $\overline{\text{CS6}}$ / $\overline{\text{PC3}}$ ADDR20/ $\overline{\text{CS7}}$ / $\overline{\text{PC4}}$ ADDR21/ $\overline{\text{CS8}}$ / $\overline{\text{PC5}}$ ADDR22/ $\overline{\text{CS9}}$ / $\overline{\text{PC6}}$	$\overline{\text{PC}}[6:0]$
ADDR23/ $\overline{\text{CS10}}$ / $\overline{\text{ECLK}}$	—
$\overline{\text{DSACK0}}/\overline{\text{PE0}}$ $\overline{\text{DSACK1}}/\overline{\text{PE1}}$ AVEC/ $\overline{\text{PE2}}$ $\overline{\text{PE3}}$ $\overline{\text{DS}}/\overline{\text{PE4}}$ AS/ $\overline{\text{PE5}}$ $\overline{\text{SIZ0}}/\overline{\text{PE6}}$ $\overline{\text{SIZ1}}/\overline{\text{PE7}}$	$\overline{\text{PE}}[7:0]$
MODCLK/ $\overline{\text{PF0}}$ IRQ[7:1]/ $\overline{\text{PF}}[7:1]$	$\overline{\text{PF0}}$ $\overline{\text{PF}}[7:1]$
DATA[15:8]	PG[7:0]
DATA[7:0]	PH[7:0]
$\overline{\text{BGACK}}/\overline{\text{CSE}}$ BG/ $\overline{\text{CSM}}$	$\overline{\text{BGACK}}$ BG

4.6.3.5 Clock Mode Selection

The state of the clock mode (MODCLK) pin during reset determines what clock source the MCU uses. When MODCLK is held high during reset, the clock signal is generated from a reference frequency. When MODCLK is held low during reset, the clock synthesizer is disabled, and an external system clock signal must be applied. Refer to **4.3 System Clock** and **APPENDIX A ELECTRICAL CHARACTERISTICS** for more information.

NOTE

The MODCLK pin can also be used as parallel I/O pin PF0. Function is determined by the state of DATA9 during reset. MODCLK function is enabled during reset regardless of DATA9 logic state. To prevent inadvertent clock mode selection by logic connected to port F, drive MODCLK high during reset.

4.6.3.6 Breakpoint Mode Selection

M68HC16 devices use internal and external breakpoint ($\overline{\text{BKPT}}$) signals. During reset exception processing, at the release of the $\overline{\text{RESET}}$ signal, the CPU16 samples these signals to determine how to handle breakpoints.

If either $\overline{\text{BKPT}}$ signal is at logic level zero when sampled, an internal BDM flag is set, and the CPU16 enters background debugging mode whenever either $\overline{\text{BKPT}}$ input is subsequently asserted.

If both $\overline{\text{BKPT}}$ inputs are at logic level one when sampled, BKPT exception processing begins whenever either $\overline{\text{BKPT}}$ signal is subsequently asserted.

Refer to **SECTION 5 CENTRAL PROCESSING UNIT** for more information on background debugging mode and exceptions. Refer to **4.5.4 CPU Space Cycles** for information concerning breakpoint acknowledge bus cycles.

4.6.4 MCU Module Pin Function During Reset

Usually, module pins default to port functions, and input/output ports are set to input state. This is accomplished by disabling pin functions in the appropriate control registers, and by clearing the appropriate port data direction registers. Refer to individual module sections in this manual for more information. Table 4-21 is a summary of module pin function out of reset. Refer to **APPENDIX D REGISTER SUMMARY** for register function and reset state.

Table 4–21. Module Pin Function

Module	Pin Mnemonic	Function
ADC	PADA[7:0]/AN[7:0]	Discrete Input
	V _{RH}	Reference Voltage
	V _{RL}	Reference Voltage
CPU	DSI/IPIPE1	DSI/IPIPE1
	DSO/IPIPE0	DSO/IPIPE0
	BKPT/DSCLK	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
MCCI	PMC7/TXDA	Discrete Input
	PMC6/RXDA	Discrete Input
	PMC5/TXDB	Discrete Input
	PMC4/RXDB	Discrete Input
	PMC3/ \overline{SS}	Discrete Input
	PMC2/SCK	Discrete Input
	PMC1/MOSI	Discrete Input
	PMC0/MISO	Discrete Input

4.6.5 Pin State During Reset

It is important to keep the distinction between pin function and pin electrical state clear. Although control register values and mode select inputs determine pin function, a pin driver can be active, inactive or in high-impedance state while reset occurs. During power-up reset, pin state is subject to the constraints discussed in 4.6.7 Power-On Reset.

NOTE

Pins that are not used should either be configured as outputs, or, if configured as inputs, pulled to the appropriate inactive state. This decreases additional I_{DD} caused by digital inputs floating near mid-supply level.

4.6.5.1 Reset States of SCIM Pins

While \overline{RESET} is asserted, SCIM pins either go to an inactive high-impedance state or are driven to their inactive states. After \overline{RESET} is released, mode selection occurs, and reset exception processing begins. Pins configured as inputs during reset become active high-impedance loads after \overline{RESET} is

released. Inputs must be driven to the desired active state. Pull-up or pull-down circuitry may be necessary. Pins configured as outputs begin to function after $\overline{\text{RESET}}$ is released. Table 4–22 is a summary of SCIM pin states during reset.

4.6.5.2 Reset States of Pins Assigned to Other MCU Modules

As a rule, module pins that are assigned to general-purpose I/O ports go to active high-impedance state following reset. However, during power-up reset, module port pins may be in an indeterminate state for a short period. Refer to **4.6.7 Power-On Reset** for more information.

Table 4–22. Pin Reset States

Pin Mnemonic	Pin State While $\overline{\text{RESET}}$ Asserted	Pin State After $\overline{\text{RESET}}$ Released			
		Default Function (Pin held high during reset)		Alternate Function (Pin held low during reset)	
		Pin Function	Pin State	Pin Function	Pin State
$\overline{\text{CS10}}/\text{ADDR23}$	1	$\overline{\text{CS10}}$	1	ADDR23	Unknown
$\overline{\text{CS}}[9:6]/\text{ADDR}[22:19]/\text{PC}[6:3]$	1	$\overline{\text{CS}}[9:6]$	1	ADDR[22:19]	Unknown
ADDR[18:0]	High-Z Output	ADDR[18:0]	Unknown	ADDR[18:0]	Unknown
$\overline{\text{AS}}/\text{PE5}$	High-Z Output	$\overline{\text{AS}}$	Output	PE5	Input
$\overline{\text{AVEC}}/\text{PE2}$	Disabled	$\overline{\text{AVEC}}$	Input	PE2	Input
$\overline{\text{BERR}}$	Disabled	$\overline{\text{BERR}}$	Input	$\overline{\text{BERR}}$	Input
$\overline{\text{CSM}}/\text{BG}$	1	$\overline{\text{CSM}}$	1	$\overline{\text{BG}}$	1
$\overline{\text{CSE}}/\text{BGACK}$	1	$\overline{\text{CSE}}$	1	$\overline{\text{BGACK}}$	Input
$\overline{\text{CS0}}/\text{BR}$	1	$\overline{\text{CS0}}$	1	BR	Input
CLKOUT	Output	CLKOUT	Output	CLKOUT	Output
$\overline{\text{CSBOOT}}$	1	$\overline{\text{CSBOOT}}$	0	$\overline{\text{CSBOOT}}$	0
DATA[15:0]	Mode Select	DATA[15:0]	Input	DATA[15:0]	Input
$\overline{\text{DS}}/\text{PE4}$	Disabled	$\overline{\text{DS}}$	Output	PE4	Input
$\overline{\text{DSACK0}}/\text{PE0}$	Disabled	$\overline{\text{DSACK0}}$	Input	PE0	Input
$\overline{\text{DSACK1}}/\text{PE1}$	Disabled	$\overline{\text{DSACK1}}$	Input	PE1	Input
$\overline{\text{CS5}}/\text{FC2}/\text{PC2}$	1	$\overline{\text{CS5}}$	1	FC2	Unknown
$\overline{\text{FC}}[2:0]/\text{PC}[2:0]$	1	FC1	1	FC1	Unknown
$\overline{\text{CS3}}/\text{FC}[2:0]/\text{PC}[2:0]$	1	$\overline{\text{CS3}}$	1	FC0	Unknown
$\overline{\text{HALT}}$	Disabled	$\overline{\text{HALT}}$	Input	$\overline{\text{HALT}}$	Input
$\overline{\text{IRQ}}[7:1]/\text{PF}[7:1]$	Disabled	$\overline{\text{IRQ}}[7:1]$	Input	PF[7:1]	Input
MODCLK/PF0	Mode Select	MODCLK	Input	PF0	Input
$\overline{\text{RW}}$	Disabled	$\overline{\text{RW}}$	Output	$\overline{\text{RW}}$	Output
$\overline{\text{RESET}}$	Asserted	$\overline{\text{RESET}}$	Input	$\overline{\text{RESET}}$	Input
PE3	Disabled	PE3	Output	PE3	Input
$\text{SIZ}[1:0]/\text{PE}[7:6]$	Disabled	$\text{SIZ}[1:0]$	Unknown	PE[7:6]	Input
TSC	Mode Select	TSC	Input	TSC	Input

4.6.6 Reset Timing

The $\overline{\text{RESET}}$ input must be asserted for a specified minimum period for reset to occur. External $\overline{\text{RESET}}$ assertion can be delayed internally for a period equal to the longest bus cycle time (or the bus monitor timeout period) in order to protect write cycles from being aborted by reset. While $\overline{\text{RESET}}$ is asserted, 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 ten cycles. At the end of this ten-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.

4.6.7 Power-On Reset

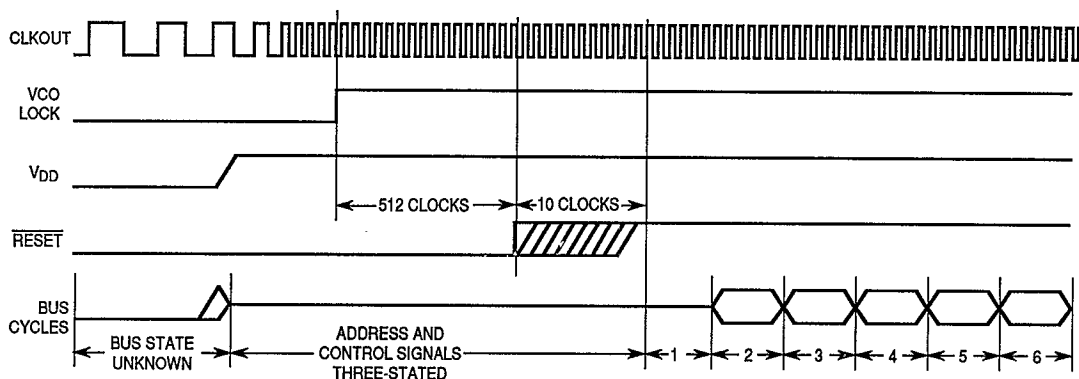
When the SCIM clock synthesizer is used to generate system clocks, power-on reset involves special circumstances related to application of system and clock synthesizer power. Regardless of clock source, voltage must be applied to clock synthesizer power input pin V_{DDSYN} for the MCU to operate. The following discussion assumes that V_{DDSYN} is applied before and during reset, which 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. Refer to **APPENDIX A ELECTRICAL CHARACTERISTICS** for voltage and timing specifications.

During power-on reset, an internal circuit in the SCIM drives the IMB internal (MSTRST) and external (EXTRST) reset lines. The circuit releases MSTRST as V_{DD} ramps up to the minimum specified value, and SCIM pins are initialized as shown in Table 4–22. As V_{DD} reaches specified minimum value, the clock synthesizer VCO begins operation and clock frequency ramps up to specified

limp mode frequency. The external $\overline{\text{RESET}}$ line remains asserted until the clock synthesizer PLL locks and 512 CLKOUT cycles elapse.

The SCIM clock synthesizer provides clock signals to the other MCU modules. After the clock is running and MSTRST is asserted for at least four clock cycles, these modules reset. V_{DD} ramp time and VCO frequency ramp time determine how long the 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 external pull-up resistors, external logic on input/output or output-only pins during this time must condition the lines. Active drivers require high-impedance buffers or isolation resistors to prevent conflict.

Figure 4–14 is a timing diagram of power-up reset. It shows the relationships between $\overline{\text{RESET}}$, V_{DD} , and bus signals.



NOTES:

1. INTERNAL START-UP TIME
2. INITIAL ZK, SK, PK FETCHED
3. INITIAL PC FETCHED
4. INITIAL SP FETCHED
5. INITIAL IZ FETCHED
6. FIRST INSTRUCTION FETCHED

FOR TM

Figure 4–14. Power-On Reset Timing

4.6.8 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.

4.6.9 Reset Processing Summary

To prevent write cycles in progress from being corrupted, a reset is recognized at the end of a bus cycle, and not at an instruction boundary. Any processing in progress at the time a reset occurs is aborted. After SCIM reset control logic has synchronized an internal or external reset request, it asserts the MSTRST signal.

The following events take place when MSTRST is asserted.

- A. Instruction execution is aborted.
- B. The condition code register is initialized.
 - 1. The IP field is set to \$7, disabling all interrupts below priority 7.
 - 2. The S bit is set, disabling LPSTOP mode.
 - 3. The SM bit is cleared, disabling MAC saturation mode.
- C. The K register is cleared.

It is important to be aware that all CCR bits that are not initialized are not affected by reset. However, out of power-on reset, these bits are indeterminate.

The following events take place when MSTRST is negated after assertion.

- A. The CPU16 samples the $\overline{\text{BKPT}}$ input.
- B. The CPU16 fetches reset vectors in the following order:
 1. Initial ZK, SK, and PK extension field values
 2. Initial PC
 3. Initial SP
 4. Initial IZ value

Vectors can be fetched from internal ROM, or from external ROM enabled by the $\overline{\text{CSBOOT}}$ signal.

- C. The CPU16 begins fetching instructions pointed to by the initial PK : PC.

4.6.10 Reset Status Register

The reset status register (RSR) contains a bit for each reset source in the MCU. When a reset occurs, a bit corresponding to the reset type is set. When multiple causes of reset occur at the same time, more than one bit in RSR can be set. The reset status register is updated by the reset control logic when the $\overline{\text{RESET}}$ signal is released. Refer to **APPENDIX D REGISTER SUMMARY**.

4.7 Interrupts

Interrupt recognition and servicing involve complex interaction between the SCIM, the central processing unit, and a device or module requesting interrupt service. This discussion provides an overview of the entire interrupt process. Chip-select logic can also be used to respond to interrupt requests. Refer to **4.8 Chip Selects** for more information.

4.7.1 Interrupt Exception Processing

The CPU16 handles interrupts as a type of asynchronous exception. An exception is an event that preempts normal processing. 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. These vectors are stored in a vector table located in the first 512 bytes of address bank 0. The CPU16 uses vector numbers to calculate displacement into the table. Refer to **SECTION 5 CENTRAL PROCESSING UNIT** for more information concerning exceptions.

4.7.2 Interrupt Priority and Recognition

The CPU16 provides for eight levels of interrupt priority (0–7), seven automatic interrupt vectors, and 200 assignable interrupt vectors. All interrupts with priorities less than 7 can be masked by the interrupt priority (IP) field in the condition code register.

Interrupt recognition is based on the states of interrupt request signals $\overline{\text{IRQ}}[7:1]$ and the IP mask value. Each of the signals corresponds to an interrupt priority. $\overline{\text{IRQ}}1$ has the lowest priority, and $\overline{\text{IRQ}}7$ has the highest priority.

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 $\overline{\text{IRQ}}7$) 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 of 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 via the external bus interface and SCIM interrupt control logic. The CPU treats external interrupt requests as if they had come from the SCIM.

External $\overline{\text{IRQ}}[6:1]$ are active-low level-sensitive inputs. External $\overline{\text{IRQ}}7$ is an active-low transition-sensitive input. It requires both an edge and a voltage level for validity.

$\overline{\text{IRQ}}[6:1]$ are maskable. $\overline{\text{IRQ}}7$ is nonmaskable. The $\overline{\text{IRQ}}7$ input is transition-sensitive to prevent redundant servicing and stack overflow. A nonmaskable interrupt is generated each time $\overline{\text{IRQ}}7$ is asserted, and each time the priority mask changes from %111 to a lower number while $\overline{\text{IRQ}}7$ is asserted.

Interrupt request signals 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.

4.7.3 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. (Refer to 4.4.1.7 **Function Codes** and 4.5.4 **CPU Space Cycles** for more information.)

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 decoded by modules that have requested interrupt service to determine whether the current interrupt acknowledge cycle pertains to them, and it is latched into the CCR IP field in order to mask lower-priority interrupts during exception processing.

Modules that have requested interrupt service decode the IP value placed on the address bus at the beginning of the interrupt acknowledge cycle, and if their requests are at the specified IP level, 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, and the reset IARB value for all other modules is %0000. Initialization software must assign different IARB values to implement an arbitration scheme.

NOTE

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

Arbitration must always take place, even when a single source is requesting service. This point is important for two reasons: the EBI does not transfer the CPU interrupt acknowledge cycle to the external bus unless the SCIM wins contention, and failure to contend causes the interrupt acknowledge bus cycle to be terminated early, by a bus error.

When arbitration is complete, the 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 data and size acknowledge (DSACK) or AVEC cycle termination signals. If the device does not respond in time, the EBI bus monitor asserts the bus error signal ($\overline{\text{BERR}}$), and a spurious interrupt exception is taken.

Chip-select logic can also be used to generate internal $\overline{\text{AVEC}}$ or $\overline{\text{DSACK}}$ signals in response to interrupt requests from external $\overline{\text{IRQ}}$ pins (refer to **4.8.3 Using Chip-Select Signals for Interrupt Acknowledge**). Chip-select address match logic functions only after the EBI transfers an interrupt acknowledge cycle to the external address bus following IARB contention. If an internal module makes an interrupt request of a certain priority, and the appropriate chip-select registers are programmed to generate $\overline{\text{AVEC}}$ or $\overline{\text{DSACK}}$ signals in response to an interrupt acknowledge cycle for that priority level, chip-select logic does not respond to the interrupt acknowledge cycle, and the internal module supplies a vector number and generates internal cycle termination signals.

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, or port F edge-detect interrupt requests, of the same priority. External interrupt requests are serviced before requests from port F edge-detect logic, as well. Refer to **4.2.10 Periodic Interrupt Timer** for more information.

4.7.4 Interrupt Processing Summary

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

- A. The CPU finishes higher priority exception processing or reaches an instruction boundary.
- B. 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 (or chip-select circuits programmed to respond to interrupts) decode the priority value in 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 $\overline{\text{DSACK}}$ signals appropriate to the access. The CPU16 acquires the vector number.
 2. The $\overline{\text{AVEC}}$ signal is asserted (the signal can be asserted by the dominant interrupt source or chip-select circuit programmed to assert $\overline{\text{AVEC}}$ internally, or the pin can be tied low), 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.

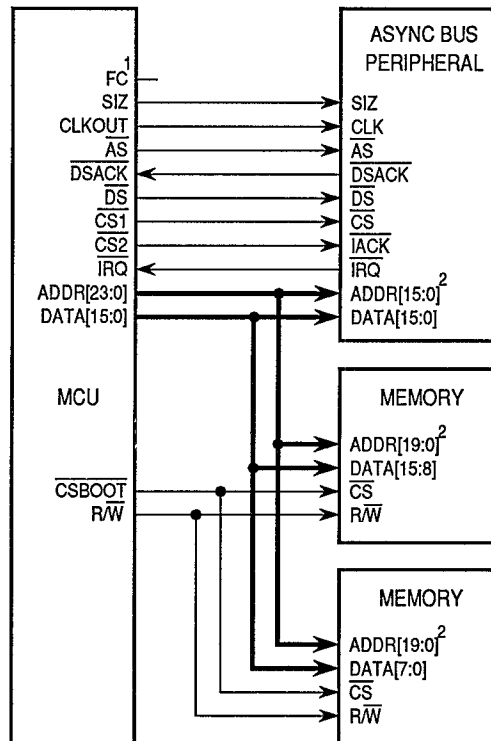
4.7.5 Interrupt Acknowledge Bus Cycles

Interrupt acknowledge bus cycles are CPU space cycles that are generated during exception processing. Refer to the *SCIM Reference Manual* (SCIMRM/AD) and **APPENDIX A ELECTRICAL CHARACTERISTICS** for further information about the types of interrupt acknowledge bus cycles that can be executed as part of interrupt exception processing.

4.8 Chip Selects

Typical microcontrollers require additional hardware to provide external chip-select signals. The MCU includes nine programmable chip-select circuits that can provide from two- to thirteen-cycle access to external memory and peripherals. Address block sizes of two Kbytes to one Mbyte can be selected. However, because ADDR[23:20] are driven to the same logic state as ADDR19, 512-Kbyte blocks are the largest usable size. In addition there are two chip-select signals for emulation support. Figure 4–15 is a diagram of a basic system that uses chip selects.

4



1. CAN BE DECODED TO PROVIDE ADDITIONAL ADDRESS SPACE.
2. VARIES DEPENDING UPON PERIPHERAL MEMORY SIZE.

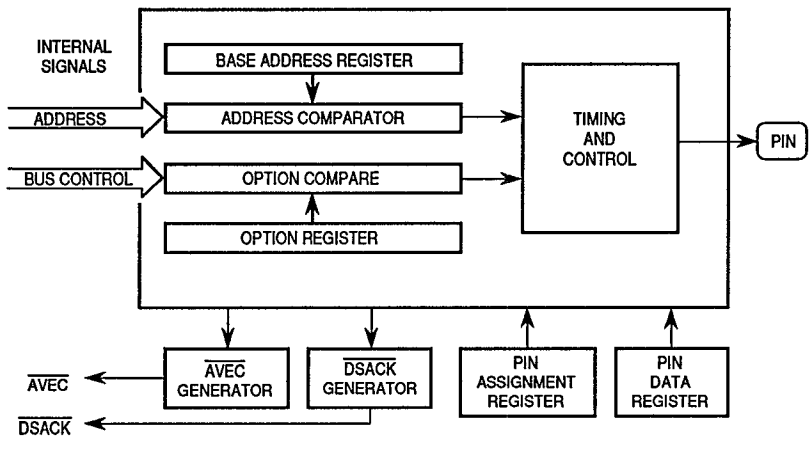
16 EXAMPLE SYS BLOCK

Figure 4–15. Basic M68HC16 System

Chip-select assertion can be synchronized with bus control signals to provide output enable, read/write strobe, or interrupt acknowledge signals. Logic can also generate \overline{DSACK} and \overline{AVEC} signals internally. A single \overline{DSACK} generator is shared by all chip-select circuits. Multiple chip selects assigned to the same address must have the same number of wait states. Each signal 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, the appropriate chip-select signal is asserted. Select signals are active low. If a chip-select function is given the same address as a microcontroller module or an internal memory array, an access to that address goes to the module or array, and the chip-select signal is not asserted. The external address and data buses do not reflect the internal access.

All chip-select circuits are configured for operation out of reset. However, all chip-select signals except \overline{CSBOOT} are disabled, and cannot be asserted until a transfer size is chosen. \overline{CSBOOT} is automatically asserted out of reset. Alternate functions for chip-select pins are enabled if appropriate data bus pins are held low at the release of the reset signal. Figure 4-16 is a functional diagram of a single chip-select circuit.



CHIP SEL BLOCK

Figure 4-16. Chip-Select Circuit Block Diagram

4.8.1 Chip-Select Registers

Each chip-select pin can have one or more functions. Chip-select pin assignment registers (CSPAR[1:0]) determine functions of the pins. Pin assignment registers also determine port size (8- or 16-bit) for dynamic bus allocation. A pin data register (CSPDR) latches data for chip-select pins that are used for discrete output.

Blocks of addresses are assigned to each chip-select function. Block sizes of two Kbytes to one Mbyte can be selected by writing values to the appropriate base address register (CSBAR[10:0], CSBARBT). However, because ADDR20 is always driven to the same logic state as ADDR19, the largest usable block size is 512 Kbytes. Address blocks for separate chip-select functions can overlap.

Chip select option registers (CSOR[10:5], CSOR3, CSOR0, CSORBT) determine timing of and conditions for assertion of chip-select signals. Eight parameters, including operating mode, access size, synchronization, and wait state insertion can be specified.

Initialization software usually resides in a peripheral memory device controlled by the chip-select circuits. A set of special chip-select functions and registers (CSORBT, CSBARBT) is provided to support bootstrap operation.

Comprehensive address maps and register diagrams are provided in **APPENDIX D REGISTER SUMMARY**.

4.8.1.1 Chip-Select Pin Assignment Registers

The pin assignment registers contain eleven 2-bit fields ($\overline{\text{CS}}[10:0]$ and CSBOOT) that determine the functions of the chip-select pins. Each pin has two or three possible functions, as shown in Table 4–23. For further information, refer to **4.6.3 Operating Configuration Out of Reset**.

**Table 4–23.
Chip-Select Pin Functions**

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

Table 4–24 shows pin assignment field encoding. Pins that have no discrete output function do not use the %00 encoding.

**Table 4–24.
Pin Assignment Field Encoding**

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

Port size determines the way in which bus transfers to an external address are allocated. Port size of eight bits or sixteen bits can be selected when a pin is assigned as a chip select. Port size and transfer size affect how the chip-select signal is asserted. Refer to **4.8.1.3 Chip-Select Option Registers** for more information.

Either 16-bit chip-select function (%11) or alternate function (%01) can be selected during reset. All chip select logic, except the boot ROM select logic ($\overline{\text{CSBOOT}}$), are disabled out of reset. Refer to **4.6.3.1 Address and Data Bus Pin Functions** and **4.8.4 Chip-Select Reset Operation** for more information.

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

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

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

4.8.1.2 Chip-Select Base Address Registers

Each chip select has an associated base address register. A base address is the lowest address in the block of addresses enabled by a chip select. Block size is the extent of the address block above the base address. Block size is determined by the value contained in a BLKSZ field. Block addresses for different chip selects can overlap.

The BLKSZ field determines which bits in the base address field are compared to corresponding bits on the address bus during an access. Provided other constraints determined by option register fields are also satisfied, when a match occurs, the associated chip-select signal is asserted. Table 4–25 shows BLKSZ encoding.

Table 4–25. Block Size 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]

ADDR[23:20] = ADDR19 during normal operation.

The chip-select 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.

The **BYTE** field controls bus allocation for chip-select transfers. Port size, set when a chip select is enabled by a pin assignment register, affects signal assertion. When an 8-bit port is assigned, any BYTE field value other than %00 enables the chip select signal. When a 16-bit port is assigned, however, BYTE field value determines when the chip select is enabled. The BYTE fields for $\overline{\text{CS}}[10:0]$ are cleared during reset. However, both bits in the boot ROM option register (CSORBT) BYTE field are set (%11) when the reset signal is released.

The $\overline{\text{R/W}}$ field causes a chip-select signal to be asserted only for a read, only for a write, or for both read and write. Use this field in conjunction with the STRB bit to generate asynchronous control signals for external devices.

The **STRB** bit controls the timing of a chip-select assertion in asynchronous mode. Selecting address strobe causes a chip-select signal to be asserted synchronized with the address strobe. Selecting data strobe causes a chip-select signal to be asserted synchronized with the data strobe. This bit has no effect in synchronous mode.

The $\overline{\text{DSACK}}$ field specifies the source of data strobe acknowledge signals used in asynchronous mode. It also allows the user to optimize bus speed in a particular application by controlling the number of wait states that are inserted.

The **SPACE** field determines the address space in which a chip select is asserted. An access must have the space type represented by SPACE encoding in order for a chip-select signal to be asserted.

When SPACE = %00 (CPU space) the **IPL** field contains an interrupt priority mask that is used when chip-select logic is set to trigger on external interrupt acknowledge cycles. When the SPACE field is set to %00 (CPU space), interrupt priority (ADDR[3:1]) is compared to IPL value. If the values are the same (and other option register constraints are satisfied), a chip select signal is asserted. This field only affects the response of chip selects and does not affect interrupt recognition by the CPU. Encoding %000 causes a chip select signal to be asserted regardless of the priority of the interrupt acknowledge cycle, provided all other constraints are met. When SPACE = %01, %10, or %11, this field specifies whether to assert the chip select during accesses to data space, program space, or both.

The $\overline{\text{AVEC}}$ bit selects one of two methods of acquiring an interrupt vector during an external interrupt acknowledge cycle. The internal autovector signal is generated only in response to interrupt requests from the SCIM $\overline{\text{IRQ}}$ pins.

4.8.1.4 PORTC Data Register

The PORTC data register latches data for PORTC pins programmed as discrete outputs. When a pin is assigned as a discrete output, the value in this register appears at the output. PC[6:0] correspond to CS[9:3]. This is a read/write register. Bit 7 is not used. Writing to this bit has no effect and it always reads zero.

4.8.2 Chip-Select Operation

When the MCU makes an access, enabled chip-select circuits compare the following items:

1. Function codes to SPACE fields, and to the IPL field if the SPACE field encoding is not for CPU space
2. Appropriate ADDR bits to base address fields
3. Read/write status to R/W fields
4. ADDR0 and/or SIZ bits to the BYTE field (16-bit ports only)
5. Priority of the interrupt being acknowledged (ADDR[3:1]) to IPL fields (when the access is an interrupt acknowledge cycle)

When a match occurs, the chip-select signal is asserted. Assertion occurs at the same time as \overline{AS} or \overline{DS} assertion in asynchronous mode. Assertion is synchronized with \overline{ECLK} in synchronous mode. In asynchronous mode, the value of the \overline{DSACK} field determines whether \overline{DSACK} is generated internally. \overline{DSACK} also determines the number of wait states inserted before internal \overline{DSACK} assertion.

The speed of an external device determines whether internal wait states are needed. Normally, wait states are inserted into the bus cycle during S3 until a peripheral asserts \overline{DSACK} . If a peripheral does not generate \overline{DSACK} , internal \overline{DSACK} generation must be selected and a predetermined number of wait states can be programmed into the chip-select option register.

Refer to the *SCIM Reference Manual (SCIMRM/AD)* for further information.

4.8.3 Using Chip-Select Signals for Interrupt Acknowledge

Ordinary I/O bus cycles use supervisor space access, but interrupt acknowledge bus cycles use CPU space access. Refer to 4.5.4 CPU Space Cycles and 4.7 Interrupts for more information. There are no differences in flow for chip selects in each type of space, but base and option registers must be properly programmed for each type of external bus cycle.

4.8.4 Chip-Select Reset Operation

The least significant bits of each of the 2-bit pin assignment fields in CSPAR0 and CSPAR1 each have a reset value of one. The reset values of the most significant bits of each field are determined by the states of DATA[7:1] during reset. There are weak internal pull-up drivers for each of the data lines, but these drivers can be overcome by bus loading effects.

The CSBOOT assignment field in CSPAR0 is configured differently from the other assignment fields. The MSB, bit 1 of CSPAR0, has a reset value of one. This enables the $\overline{\text{CSBOOT}}$ signal to select a boot ROM containing initialization firmware. The LSB value, determined by the logic level of DATA0 during reset, selects boot ROM port size. When DATA0 is held low, port size is 8 bits. When DATA0 is held high, port size is 16 bits.

After reset, the MCU fetches initialization values from word addresses \$0000 to \$0006 in bank 0 of program space. To support bootstrap operation from reset, the base address field in chip-select base address register boot (CSBARBT) has a reset value of all zeros. A ROM device containing vectors located at these addresses can be enabled by $\overline{\text{CSBOOT}}$ after a reset. Alternately, the on-chip ROM module can be used for bootstrap operation. Refer to **SECTION 11 MASKED ROM MODULE** for more information. The block size field in CSBARBT has a reset value of 512 Kbytes.

The byte 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. Refer to Table 4-27.

Table 4-27.
CSBOOT Base and Option Register
Reset Values

Fields	Reset Values
Base Address	\$0000 0000
Block Size	512 Kbytes
Async/Sync Mode	Asynchronous Mode
Upper/Lower Byte	Both Bytes
Read/Write	Read/Write
$\overline{\text{AS/DS}}$	$\overline{\text{AS}}$
$\overline{\text{DSACK}}$	13 Wait States
Address Space	Supervisor/User Space
IPL	Any Level
Autovector	Interrupt Vector Externally

4.8.5 Emulation-Support Chip Selects

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

Two special chip selects, $\overline{\text{CSE}}$ and $\overline{\text{CSM}}$, support external emulation of on-chip ports and on-chip ROM, respectively. Chip select pin assignment register 0 (CSPAR0) assigns the $\overline{\text{CSE/BGACK}}$ and $\overline{\text{CSM/BG}}$ pins for chip-select or alternate function. $\overline{\text{CSE}}$ and $\overline{\text{CSM}}$ do not have associated base and option registers.

4.8.5.1 External Port Emulation

Emulator mode is a special type of 16-bit expanded operation. It is entered by holding DATA10 low, 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 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. Refer to Motorola Technical Summary *Port-Replacement Unit* (MC68HC33TS/D).

4.8.5.2 External ROM Emulation

External ROM emulation is enabled by holding DATA10 and DATA13 low during reset. (DATA14 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 array does not acknowledge IMB accesses while in emulation mode. This causes the SCIM to run an external bus cycle for each access. An internal $\overline{\text{DSACK}}$ signal is generated by the ROM module after it has inserted the number of wait states specified by the WAIT field in the MRMCR. Refer to **SECTION 11 MASKED ROM MODULE** for more information

4.9 Factory Test

The test submodule supports scan-based testing of the various MCU modules. It is integrated into the SCIM to support production test. Test submodule registers are intended for Motorola use. Register names and addresses are provided in **APPENDIX D REGISTER SUMMARY** to show the user that these addresses are occupied. The QUOT pin is also used for factory test.

SECTION 5 CENTRAL PROCESSING UNIT

This section is an overview of the CPU16. For detailed information, refer to the *CPU16 Reference Manual (CPU16RM/AD)*.

5.1 General

The central processing unit (CPU16) provides compatibility with the M68HC11 CPU and also provides additional capabilities associated with 16- and 32-bit data sizes, 20-bit addressing, and digital signal processing. CPU16 registers are an integral part of the CPU and are not addressed as memory locations.

The CPU16 treats all peripheral, I/O, and memory locations as parts of a pseudolinear 1 Megabyte address space. There are no special instructions for I/O that are separate from instructions for addressing memory. Address space is made up of sixteen 64-Kbyte banks. Specialized bank addressing techniques and support registers provide transparent access across bank boundaries.

The CPU16 interacts with external devices and with other modules within the microcontroller via a standardized bus and bus interface. There are bus protocols used for memory and peripheral accesses, as well as for managing an hierarchy of interrupt priorities.

5.2 Register Model

Figure 5–1 shows the CPU16 register model. Refer to the paragraphs that follow for a detailed description of each register.

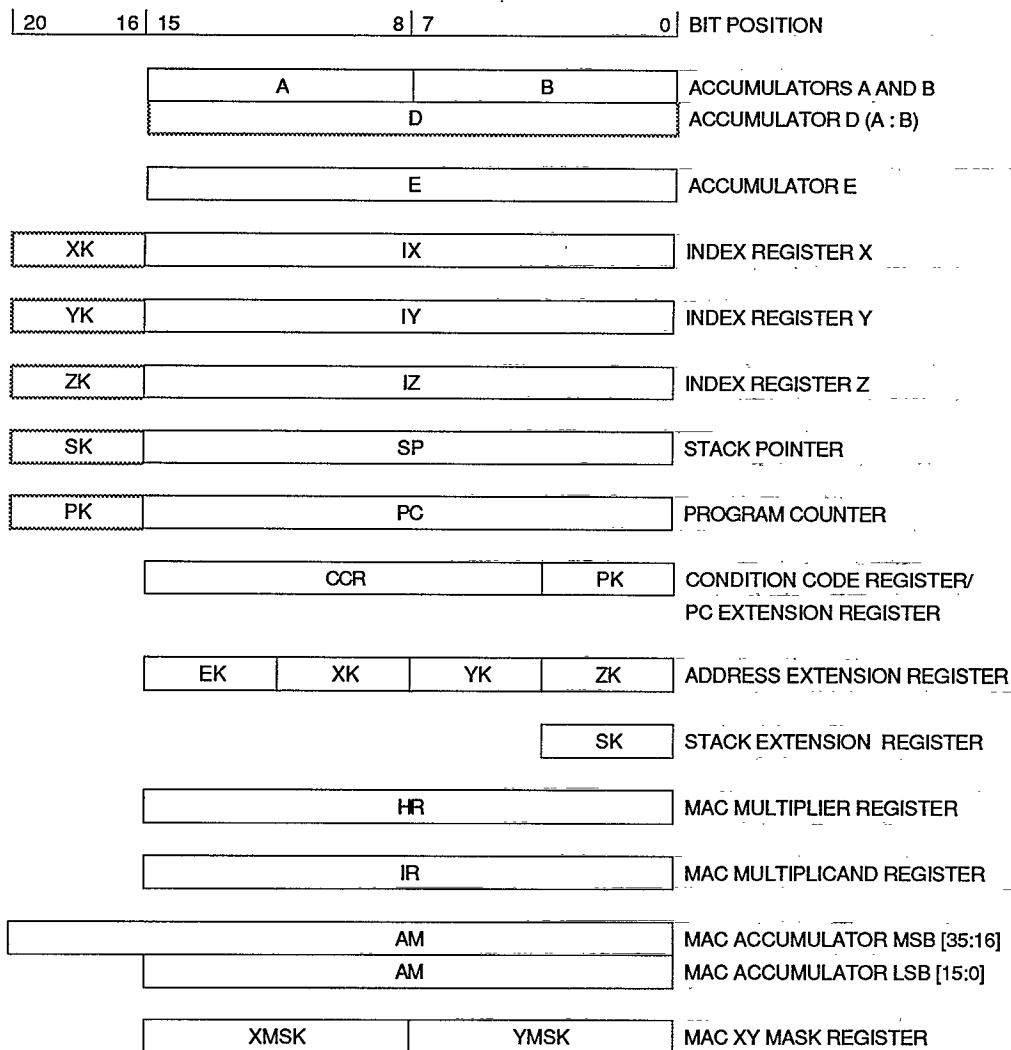


Figure 5–1. CPU16 Register Model

5.2.1 Accumulators

The CPU16 has two 8-bit accumulators (A and B) and one 16-bit accumulator (E). In addition, accumulators A and B can be concatenated into a second 16-bit double accumulator (D).

Accumulators A, B, and D are general-purpose registers that hold operands and results during mathematic and data manipulation operations.

Accumulator E, which can be used in the same way as accumulator D, also extends CPU16 capabilities. It allows more data to be held within the CPU16 during operations, simplifies 32-bit arithmetic and digital signal processing, and provides a practical 16-bit accumulator offset indexed addressing mode.

5.2.2 Index Registers

The CPU16 has three 16-bit index registers (IX, IY, and IZ). Each index register has an associated 4-bit extension field (XK, YK, and ZK).

Concatenated registers and extension fields provide 20-bit indexed addressing and support data structure functions anywhere in the CPU16 address space.

IX and IY can perform the same operations as M68HC11 registers of the same names, but the CPU16 instruction set provides additional indexed operations.

IZ can perform the same operations as IX and IY, and also provides an additional indexed addressing capability that replaces M68HC11 direct addressing mode. Initial IZ and ZK extension field values are included in the RESET exception vector, so that ZK : IZ can be used as a direct page pointer out of reset.

5.2.3 Stack Pointer

The CPU16 stack pointer (SP) is 16 bits wide. An associated 4-bit extension field (SK) provides 20-bit stack addressing.

Stack implementation in the CPU16 is from high to low memory. The stack grows downward as it is filled. SK : SP are decremented each time data is pushed on the stack, and incremented each time data is pulled from the stack.

SK : SP point to the next available stack address, rather than to the address of the latest stack entry. Although the stack pointer is normally incremented or decremented by word address, it is possible to push and pull byte-sized data. Setting the stack pointer to an odd value causes misalignment, which affects performance.

5.2.4 Program Counter

The CPU16 program counter (PC) is 16 bits wide. An associated 4-bit extension field (PK) provides 20-bit program addressing.

CPU16 instructions are fetched from even word boundaries. PC0 always has a value of zero, to ensure that instruction fetches are made from word-aligned addresses.

5.2.5 Condition Code Register

The 16-bit condition code register is composed of two functional blocks. The eight MSB, which correspond to the CCR in the M68HC11, contain the low-power stop control bit and processor status flags. The eight LSB contain the interrupt priority field, the DSP saturation mode control bit, and the program counter address extension field.

Figure 5–2 shows the condition code register. Detailed descriptions of each status indicator and field in the register follow the figure.

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			SM	PK			

Figure 5–2. Condition Code Register

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.

5.2.6 Address Extension Register and Address Extension Fields

There are six 4-bit address extension fields. EK, XK, YK, and ZK are contained by the address extension register, PK is part of the CCR, and SK stands alone.

Extension fields are the bank portions of 20-bit concatenated bank : byte addresses used in the CPU16 pseudolinear memory management scheme.

All extension fields except EK correspond directly to a register. XK, YK, and ZK extend registers IX, IY, and IZ; PK extends the PC; and SK extends the SP. EK holds the four MSB of the 20-bit address used by extended addressing mode.

5.2.7 Multiply and Accumulate Registers

The multiply and accumulate (MAC) registers are part of a CPU submodule that performs repetitive signed fractional multiplication and stores the cumulative result. These operations are part of control-oriented digital signal processing.

There are four MAC registers. Register H contains the 16-bit signed fractional multiplier. Register I contains the 16-bit signed fractional multiplicand. Accumulator M is a specialized 36-bit product accumulation register. XMSK and YMSK contain 8-bit mask values used in modulo addressing.

The CPU16 has a special subset of signal processing instructions that manipulate the MAC registers and perform signal processing calculation.

5.3 Memory Management

The CPU16 uses bank switching to provide a 1-Mbyte address space. There are 16 banks within the address space. Each bank is made up of 64 Kbytes addressed from \$0000 to \$FFFF. Banks are selected by means of address extension fields associated with individual CPU16 registers.

In addition, address space can be split into discrete 1-Mbyte program and data spaces by externally decoding the SCIM function code outputs. When this technique is used, instruction fetches and reset vector fetches access program space, while exception vector fetches (other than for reset), data accesses, and stack accesses are made in data space.

5

5.3.1 Address Extension

All CPU16 resources used to generate addresses are effectively 20 bits wide. These resources include extended index registers, program counter, and stack pointer. All addressing modes use 20-bit addresses. The CPU16 also drives ADDR[23:20] to the same logic state as ADDR19.

Twenty-bit addresses are formed from a 16-bit byte address generated by an individual CPU16 register and a 4-bit bank address contained in an associated extension field. The byte address corresponds to ADDR[15:0] and the bank address corresponds to ADDR[19:16].

5.3.2 Extension Fields

Each of the six address extension fields is used in a different type of access. All but EK are associated with particular CPU16 registers. There are several ways to manipulate extension fields and the address map. Refer to the *CPU16 Reference Manual* (CPU16RM/AD) for detailed information.

5.4 Data Types

The CPU16 uses the following types of data:

- Bits
- 4-bit signed integers
- 8-bit (byte) signed and unsigned integers
- 8-bit, 2-digit binary coded decimal numbers
- 16-bit (word) signed and unsigned integers
- 32-bit (long word) signed and unsigned integers
- 16-bit signed fractions
- 32-bit signed fractions
- 36-bit signed fixed-point numbers
- 20-bit effective address consisting of 16-bit byte address and 4-bit extension

There are 8 bits in a byte, 16 bits in a word. Bit set and clear instructions use both byte and word operands. Bit test instructions use byte operands.

Negative integers are represented in two's complement form. Four-bit signed integers, packed two to a byte, are used only as X and Y offsets in MAC and RMAC operations. Thirty-two-bit integers are used only by extended multiply and divide instructions, and by the associated LDED and STED instructions.

Binary coded decimal (BCD) numbers are packed, two digits per byte. BCD operations use byte operands.

Signed 16-bit fractions are used by fractional multiplication instructions, and as multiplicand and multiplier operands in the MAC unit. Bit 15 is the sign bit, and there is an implied radix point between bits 15 and 14. There are 15 bits of magnitude. The range of values is -1 (\$8000) to $1 - 2^{-15}$ (\$7FFF).

Signed 32-bit fractions are used only by fractional multiplication and division instructions. Bit 31 is the sign bit. An implied radix point lies between bits 31 and 30. There are 31 bits of magnitude. The range of values is -1 (\$80000000) to $1 - 2^{-31}$ (\$7FFFFFFF).

Signed 36-bit fixed-point numbers are used only by the MAC unit. Bit 35 is the sign bit. Bits [34:31] are sign extension bits. There is an implied radix point between bits 31 and 30. There are 31 bits of magnitude, but use of the extension bits allows representation of numbers in the range -16 (\$80000000) to 15.999969482 (\$7FFFFFFF).

Twenty-bit addresses are formed by combining a 16-bit byte address with a 4-bit address extension.

5.5 Memory Organization

Both program and data memory are divided into sixteen 64-Kbyte banks. Addressing is pseudolinear. A 20-bit extended address can access any byte location in the appropriate address space.

A word is composed of two consecutive bytes. A word address is normally an even byte address. Byte 0 of a word has a lower 16-bit address than byte 1. Long words and 32-bit signed fractions consist of two consecutive words, and are normally accessed at the address of byte 0 in word 0.

Instruction fetches always access word addresses. Word operands are normally accessed at even byte addresses, but can be accessed at odd byte addresses, with a substantial performance penalty.

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

Figure 5-3 shows how each CPU16 data type is organized in memory. Consecutive even addresses show size and alignment.

Address	Type															
	BIT 15	BIT 14	BIT 13	BIT 12	BIT 11	BIT 10	BIT 9	BIT 8	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
\$0000																
\$0002	BYTE0								BYTE1							
\$0004	±	X OFFSET			±	Y OFFSET			±	X OFFSET			±	Y OFFSET		
\$0006	BCD1				BCD0				BCD1				BCD0			
\$0008	WORD 0															
\$000A	WORD1															
\$000C	MSW LONG WORD 0															
\$000E	LSW LONG WORD 0															
\$0010	MSW LONG WORD 1															
\$0012	LSW LONG WORD 1															
\$0014	±	← (Radix Point)			16-BIT SIGNED FRACTION 0											
\$0016	±	← (Radix Point)			16-BIT SIGNED FRACTION 1											
\$0018	±	← (Radix Point)			MSW 32-BIT SIGNED FRACTION 0											
\$001A	LSW 32-BIT SIGNED FRACTION 0															0
\$001C	±	← (Radix Point)			MSW 32-BIT SIGNED FRACTION 1											
\$001E	LSW 32-BIT SIGNED FRACTION 1															0

MAC Data Types

35	32	31	16
±	«	«	«
		«	← (Radix Point)
			MSW 32-BIT SIGNED FRACTION
		15	0
			LSW 32-BIT SIGNED FRACTION
±			← (Radix Point)
			16-BIT SIGNED FRACTION

Address Data Type

19	16	15	0
4-Bit Extension		16-Bit Address	

Figure 5-3. Data Types and Memory Organization

5.6 Addressing Modes

The CPU16 uses nine types of addressing. There are one or more addressing modes within each type. Table 5–1 shows the addressing modes.

Table 5–1. Addressing Modes

Mode	Mnemonic	Description
Accumulator Offset	E,X	Index register X with accumulator E offset
	E,Y	Index register Y with accumulator E offset
	E,Z	Index register Z with accumulator E offset
Extended	EXT	Extended
	EXT20	20-bit extended
Immediate	IMM8	8-bit immediate
	IMM16	16-bit immediate
Indexed 8-Bit	IND8, X	Index register X with unsigned 8-bit offset
	IND8, Y	Index register Y with unsigned 8-bit offset
	IND8, Z	Index register Z with unsigned 8-bit offset
Indexed 16-Bit	IND16, X	Index register X with signed 16-bit offset
	IND16, Y	Index register Y with signed 16-bit offset
	IND16, Z	Index register Z with signed 16-bit offset
Indexed 20-Bit	IND20, X	Index register X with signed 20-bit offset
	IND20, Y	Index register Y with signed 20-bit offset
	IND20, Z	Index register Z with signed 20-bit offset
Inherent	INH	Inherent
Post-Modified Index	IXP	Signed 8-bit offset added to index register X after effective address is used
Relative	REL8	8-bit relative
	REL16	16-bit relative

All modes generate ADDR[15:0]. This address is combined with ADDR[19:16] from an operand or an extension field to form a 20-bit effective address.

NOTE

Bank switching is transparent to most instructions. ADDR[19:16] of the effective address are changed to make an access across a bank boundary. However, extension field values do not change as a result of effective address computation.

5.6.1 Immediate Addressing Modes

In the immediate modes, an argument is contained in a byte or word immediately following the instruction. For IMM8 and IMM16 modes, the effective address is the address of the argument.

There are three specialized forms of IMM8 addressing.

The AIS, AIX/Y/Z, ADDD and ADDE instructions decrease execution time by sign-extending the 8-bit immediate operand to 16 bits, then adding it to an appropriate register.

The MAC and RMAC instructions use an 8-bit immediate operand to specify two signed 4-bit index register offsets.

The PSHM and PULM instructions use an 8-bit immediate mask operand to indicate which registers must be pushed to or pulled from the stack.

5.6.2 Extended Addressing Modes

Regular extended mode instructions contain ADDR[15:0] in the word following the opcode. The effective address is formed by concatenating the EK field and the 16-bit byte address. EXT20 mode is used only by the JMP and JSR instructions. These instructions contain a 20-bit effective address that is zero-extended to 24 bits to give the instruction an even number of bytes.

5.6.3 Indexed Addressing Modes

In the indexed modes, registers IX, IY, and IZ, together with their associated extension fields, are used to calculate the effective address.

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

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

For 20-bit modes, a 20-bit signed offset (zero-extended to 24 bits) is added to the value contained in an index register. These modes are used for JMP and JSR instructions only.

5.6.4 Inherent Addressing Mode

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

5.6.5 Accumulator Offset Addressing Mode

Accumulator offset modes form an effective address by sign-extending the content of accumulator E to 20 bits, then adding the result to an index register and its associated extension field. This mode allows use of an index register and an accumulator within a loop without corrupting accumulator D.

5.6.6 Relative Addressing Modes

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

5.6.7 Post-Modified Index Addressing Mode

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

5.6.8 Use of CPU16 Indexed Mode to Replace M68HC11 Direct Mode

In M68HC11 systems, the direct addressing mode can be used to perform rapid accesses to RAM or I/O mapped into bank 0 (\$0000 to \$00FF), but the CPU16 uses the first 512 bytes of bank 0 for exception vectors. To provide an enhanced replacement for 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 indexed mode addressing, a programmer can access useful data structures anywhere in the address map.

5.7 Instruction Set

The CPU16 instruction set is based on the M68HC11 instruction set, but the opcode map has been rearranged to maximize performance with a 16-bit data bus. Most M68HC11 code can run on the CPU16 following reassembly. The user must take into account changed instruction times, the interrupt mask, and the changed interrupt stack frame (see Motorola Programming Note M68HC16PN01/D, *Transporting M68HC11 Code to M68HC16 Devices*, for more information).

5.7.1 Instruction Set Summary

The following summary is a quick reference to the entire CPU16 instruction set. Refer to the *CPU16 Reference Manual (CPU16RM/AD)* for detailed information about each instruction, assembler syntax, and condition code evaluation. A key to table nomenclature is provided on the last page of the table.

Instruction Set Summary

Mnemonic	Operation	Description	Address Mode	Instruction			Condition Codes												
				Opcode	Operand	Cycles	S	MV	H	EV	N	Z	V	C					
ABA	Add B to A	$(A) + (B) \Rightarrow A$	INH	370B	—	2	—	—	Δ	—	—	—	—	Δ	Δ	Δ	Δ		
ABX	Add B to X	$(XK : IX) + (000 : B) \Rightarrow XK : IX$	INH	374F	—	2	—	—	—	—	—	—	—	—	—	—	—		
ABY	Add B to Y	$(YK : IY) + (000 : B) \Rightarrow YK : IY$	INH	375F	—	2	—	—	—	—	—	—	—	—	—	—	—		
ABZ	Add B to Z	$(ZK : IZ) + (000 : B) \Rightarrow ZK : IZ$	INH	376F	—	2	—	—	—	—	—	—	—	—	—	—	—		
ACE	Add E to AM[31:15]	$(AM[31:15]) + (E) \Rightarrow AM$	INH	3722	—	2	—	Δ	—	Δ	—	—	—	—	—	—	—		
ACED	Add concatenated E and D to AM	$(E : D) + (AM) \Rightarrow AM$	INH	3723	—	4	—	Δ	—	Δ	—	—	—	—	—	—	—		
ADCA	Add with Carry to A	$(A) + (M) + C \Rightarrow A$	IND8, X	43	ff	6	—	—	Δ	—	Δ	Δ	Δ	Δ	Δ	Δ	Δ		
			IND8, Y	53	ff	6	—	—	—	—	—	—	—	—	—	—	—	—	
			IND8, Z	63	ff	6	—	—	—	—	—	—	—	—	—	—	—	—	—
			IMM8	73	ii	2	—	—	—	—	—	—	—	—	—	—	—	—	—
			IND16, X	1743	9999	6	—	—	—	—	—	—	—	—	—	—	—	—	—
			IND16, Y	1753	9999	6	—	—	—	—	—	—	—	—	—	—	—	—	—
			IND16, Z	1763	9999	6	—	—	—	—	—	—	—	—	—	—	—	—	—
			EXT	1773	hh ll	6	—	—	—	—	—	—	—	—	—	—	—	—	—
			E, X	2743	—	6	—	—	—	—	—	—	—	—	—	—	—	—	—
			E, Y	2753	—	6	—	—	—	—	—	—	—	—	—	—	—	—	—
E, Z	2763	—	6	—	—	—	—	—	—	—	—	—	—	—	—	—			
ADCB	Add with Carry to B	$(B) + (M) + C \Rightarrow B$	IND8, X	C3	ff	6	—	—	Δ	—	Δ	Δ	Δ	Δ	Δ	Δ	Δ		
			IND8, Y	D3	ff	6	—	—	—	—	—	—	—	—	—	—	—	—	
			IND8, Z	E3	ff	6	—	—	—	—	—	—	—	—	—	—	—	—	
			IMM8	F3	ii	2	—	—	—	—	—	—	—	—	—	—	—	—	—
			E, X	27C3	—	6	—	—	—	—	—	—	—	—	—	—	—	—	—
			E, Y	27D3	—	6	—	—	—	—	—	—	—	—	—	—	—	—	—
			E, Z	27E3	—	6	—	—	—	—	—	—	—	—	—	—	—	—	—
			IND16, X	17C3	9999	6	—	—	—	—	—	—	—	—	—	—	—	—	—
			IND16, Y	17D3	9999	6	—	—	—	—	—	—	—	—	—	—	—	—	—
			IND16, Z	17E3	9999	6	—	—	—	—	—	—	—	—	—	—	—	—	—
EXT	17F3	hh ll	6	—	—	—	—	—	—	—	—	—	—	—	—	—			
ADCD	Add with Carry to D	$(D) + (M : M + 1) + C \Rightarrow D$	IND8, X	83	ff	6	—	—	—	—	Δ	Δ	Δ	Δ	Δ	Δ	Δ		
			IND8, Y	93	ff	6	—	—	—	—	—	—	—	—	—	—	—	—	
			IND8, Z	A3	ff	6	—	—	—	—	—	—	—	—	—	—	—	—	
			E, X	2783	—	6	—	—	—	—	—	—	—	—	—	—	—	—	
			E, Y	2793	—	6	—	—	—	—	—	—	—	—	—	—	—	—	—
			E, Z	27A3	—	6	—	—	—	—	—	—	—	—	—	—	—	—	—
			IMM16	37B3	jj kk	4	—	—	—	—	—	—	—	—	—	—	—	—	—
			IND16, X	37C3	9999	6	—	—	—	—	—	—	—	—	—	—	—	—	—
			IND16, Y	37D3	9999	6	—	—	—	—	—	—	—	—	—	—	—	—	—
			IND16, Z	37E3	9999	6	—	—	—	—	—	—	—	—	—	—	—	—	—
EXT	37F3	hh ll	6	—	—	—	—	—	—	—	—	—	—	—	—	—			
ADCE	Add with Carry to E	$(E) + (M : M + 1) + C \Rightarrow E$	IMM16	3733	jj kk	4	—	—	—	—	Δ	Δ	Δ	Δ	Δ	Δ	Δ		
			IND16, X	3743	9999	6	—	—	—	—	—	—	—	—	—	—	—	—	
			IND16, Y	3753	9999	6	—	—	—	—	—	—	—	—	—	—	—	—	
			IND16, Z	3763	9999	6	—	—	—	—	—	—	—	—	—	—	—	—	
			EXT	3773	hh ll	6	—	—	—	—	—	—	—	—	—	—	—	—	
ADDA	Add to A	$(A) + (M) \Rightarrow A$	IND8, X	41	ff	6	—	—	Δ	—	Δ	Δ	Δ	Δ	Δ	Δ	Δ		
			IND8, Y	51	ff	6	—	—	—	—	—	—	—	—	—	—	—	—	
			IND8, Z	61	ff	6	—	—	—	—	—	—	—	—	—	—	—	—	
			IMM8	71	ii	2	—	—	—	—	—	—	—	—	—	—	—	—	—
			E, X	2741	—	6	—	—	—	—	—	—	—	—	—	—	—	—	—
			E, Y	2751	—	6	—	—	—	—	—	—	—	—	—	—	—	—	—
			E, Z	2761	—	6	—	—	—	—	—	—	—	—	—	—	—	—	—
			IND16, X	1741	9999	6	—	—	—	—	—	—	—	—	—	—	—	—	
			IND16, Y	1751	9999	6	—	—	—	—	—	—	—	—	—	—	—	—	
			IND16, Z	1761	9999	6	—	—	—	—	—	—	—	—	—	—	—	—	
EXT	1771	hh ll	6	—	—	—	—	—	—	—	—	—	—	—	—	—			

Instruction Set Summary (Continued)

Mnemonic	Operation	Description	Address Mode	Instruction			Condition Codes												
				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	—	—	—	—	—	—	—	—	—	—	—	—	—
			E, X	27C1	—	6	—	—	—	—	—	—	—	—	—	—	—	—	—
			E, Y	27D1	—	6	—	—	—	—	—	—	—	—	—	—	—	—	—
			E, Z	27E1	—	6	—	—	—	—	—	—	—	—	—	—	—	—	—
			IND16, X	17C1	9999	6	—	—	—	—	—	—	—	—	—	—	—	—	—
			IND16, Y	17D1	9999	6	—	—	—	—	—	—	—	—	—	—	—	—	—
			IND16, Z	17E1	9999	6	—	—	—	—	—	—	—	—	—	—	—	—	—
			EXT	17F1	hh ll	6	—	—	—	—	—	—	—	—	—	—	—	—	—
			ADDD	Add to D	(D) + (M : M + 1) ⇒ D	IND8, X	81	ff	6	—	—	—	—	—	—	—	Δ	Δ	Δ
IND8, Y	91	ff				6	—	—	—	—	—	—	—	—	—	—	—	—	
IND8, Z	A1	ff				6	—	—	—	—	—	—	—	—	—	—	—	—	—
IMM8	FC	ii				2	—	—	—	—	—	—	—	—	—	—	—	—	—
E, X	2781	—				6	—	—	—	—	—	—	—	—	—	—	—	—	—
E, Y	2791	—				6	—	—	—	—	—	—	—	—	—	—	—	—	—
E, Z	27A1	—				6	—	—	—	—	—	—	—	—	—	—	—	—	—
IMM16	37B1	—				4	—	—	—	—	—	—	—	—	—	—	—	—	—
IND16, X	37C1	jjkk				6	—	—	—	—	—	—	—	—	—	—	—	—	—
IND16, Y	37D1	9999				6	—	—	—	—	—	—	—	—	—	—	—	—	—
IND16, Z	37E1	9999				6	—	—	—	—	—	—	—	—	—	—	—	—	—
EXT	37F1	hh ll				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	9999	6	—	—	—	—	—	—	—	—	—	—	—	—	
			IND16, Y	3751	9999	6	—	—	—	—	—	—	—	—	—	—	—	—	—
			IND16, Z	3761	9999	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 X	(XK : IX) + (←D) ⇒ XK : IX	INH	37CD	—	2	—	—	—	—	—	—	—	—	—	—	—		
ADY	Add D to Y	(YK : IY) + (←D) ⇒ YK : IY	INH	37DD	—	2	—	—	—	—	—	—	—	—	—	—	—		
ADZ	Add D to Z	(ZK : IZ) + (←D) ⇒ ZK : IZ	INH	37ED	—	2	—	—	—	—	—	—	—	—	—	—	—		
AEX	Add E to X	(XK : IX) + (←E) ⇒ XK : IX	INH	374D	—	2	—	—	—	—	—	—	—	—	—	—	—		
AEY	Add E to Y	(YK : IY) + (←E) ⇒ YK : IY	INH	375D	—	2	—	—	—	—	—	—	—	—	—	—	—		
AEZ	Add E to Z	(ZK : IZ) + (←E) ⇒ ZK : IZ	INH	376D	—	2	—	—	—	—	—	—	—	—	—	—	—		
AIS	Add Immediate Data to SP	SK : SP + ←IMM ⇒ SK : SP	IMM8 IMM16	3F 373F	ii jj kk	2 4	—	—	—	—	—	—	—	—	—	—	—		
AIX	Add Immediate Value to X	XK : IX + ←IMM ⇒ XK : IX	IMM8 IMM16	3C 373C	ii jj kk	2 4	—	—	—	—	—	—	—	Δ	—	—	—		
AIY	Add Immediate Value to Y	YK : IY + ←IMM ⇒ YK : IY	IMM8 IMM16	3D 373D	ii jj kk	2 4	—	—	—	—	—	—	—	Δ	—	—	—		
AIZ	Add Immediate Value to Z	ZK : IZ + ←IMM ⇒ ZK : IZ	IMM8 IMM16	3E 373E	ii jj kk	2 4	—	—	—	—	—	—	—	Δ	—	—	—		
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	9999	6	—	—	—	—	—	—	—	—	—	—	—	—	—
			IND16, Y	1756	9999	6	—	—	—	—	—	—	—	—	—	—	—	—	—
			IND16, Z	1766	9999	6	—	—	—	—	—	—	—	—	—	—	—	—	—
			EXT	1776	hh ll	6	—	—	—	—	—	—	—	—	—	—	—	—	—
			E, X	2746	—	6	—	—	—	—	—	—	—	—	—	—	—	—	—
			E, Y	2756	—	6	—	—	—	—	—	—	—	—	—	—	—	—	—
			E, Z	2766	—	6	—	—	—	—	—	—	—	—	—	—	—	—	—
			ANDB	AND B	(B) • (M) ⇒ B	IND8, X	C6	ff	6	—	—	—	—	—	—	—	Δ	Δ	0
IND8, Y	D6	ff				6	—	—	—	—	—	—	—	—	—	—	—	—	
IND8, Z	E6	ff				6	—	—	—	—	—	—	—	—	—	—	—	—	
IMM8	F6	ii				2	—	—	—	—	—	—	—	—	—	—	—	—	—
IND16, X	17C6	9999				6	—	—	—	—	—	—	—	—	—	—	—	—	—
IND16, Y	17D6	9999				6	—	—	—	—	—	—	—	—	—	—	—	—	—
IND16, Z	17E6	9999				6	—	—	—	—	—	—	—	—	—	—	—	—	—
EXT	17F6	hh ll				6	—	—	—	—	—	—	—	—	—	—	—	—	—
E, X	27C6	—				6	—	—	—	—	—	—	—	—	—	—	—	—	—
E, Y	27D6	—				6	—	—	—	—	—	—	—	—	—	—	—	—	—
E, Z	27E6	—				6	—	—	—	—	—	—	—	—	—	—	—	—	—

5

Instruction Set Summary (Continued)

Mnemonic	Operation	Description	Address Mode	Instruction			Condition Codes									
				Opcode	Operand	Cycles	S	MV	H	EV	N	Z	V	C		
ANDD	AND D	$(D) \cdot (M : M + 1) \Rightarrow D$	IND8, X	86	ff	6	—	—	—	—	—	—	—	—	—	
			IND8, Y	96	ff	6	—	—	—	—	—	—	—	—	—	
			IND8, Z	A6	ff	6	—	—	—	—	—	—	—	—	—	—
			E, X	2786	—	6	—	—	—	—	—	—	—	—	—	—
			E, Y	2796	—	6	—	—	—	—	—	—	—	—	—	—
			E, Z	27A6	—	6	—	—	—	—	—	—	—	—	—	—
			IMM16	37B6	jj kk	4	—	—	—	—	—	—	—	—	—	—
			IND16, X	37C6	gggg	6	—	—	—	—	—	—	—	—	—	—
			IND16, Y	37D6	gggg	6	—	—	—	—	—	—	—	—	—	—
			IND16, Z	37E6	gggg	6	—	—	—	—	—	—	—	—	—	—
EXT	37F6	hh ll	6	—	—	—	—	—	—	—	—	—	—			
ANDE	AND E	$(E) \cdot (M : M + 1) \Rightarrow E$	IMM16	3736	jj kk	4	—	—	—	—	—	—	—	—		
			IND16, X	3746	gggg	6	—	—	—	—	—	—	—	—		
			IND16, Y	3756	gggg	6	—	—	—	—	—	—	—	—	—	
			IND16, Z	3766	gggg	6	—	—	—	—	—	—	—	—	—	
			EXT	3776	hh ll	6	—	—	—	—	—	—	—	—	—	
ANDP ¹	AND CCR	$(CCR) \cdot IMM16 \Rightarrow CCR$	IMM16	373A	jj kk	4	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ		
ASL	Arithmetic Shift Left		IND8, X	04	ff	8	—	—	—	—	—	—	—	—		
			IND8, Y	14	ff	8	—	—	—	—	—	—	—	—		
			IND8, Z	24	ff	8	—	—	—	—	—	—	—	—	—	
			IND16, X	1704	gggg	8	—	—	—	—	—	—	—	—	—	
			IND16, Y	1714	gggg	8	—	—	—	—	—	—	—	—	—	
			IND16, Z	1724	gggg	8	—	—	—	—	—	—	—	—	—	
ASLA	Arithmetic Shift Left A		INH	3704	—	2	—	—	—	—	—	—	—			
ASLB	Arithmetic Shift Left B		INH	3714	—	2	—	—	—	—	—	—	—			
ASLD	Arithmetic Shift Left D		INH	27F4	—	2	—	—	—	—	—	—	—			
ASLE	Arithmetic Shift Left E		INH	2774	—	2	—	—	—	—	—	—	—			
ASLM	Arithmetic Shift Left AM		INH	27B6	—	4	—	Δ	—	—	—	—	—			
ASLW	Arithmetic Shift Left Word		IND16, X	2704	gggg	8	—	—	—	—	—	—	—			
			IND16, Y	2714	gggg	8	—	—	—	—	—	—	—	—		
			IND16, Z	2724	gggg	8	—	—	—	—	—	—	—	—		
			EXT	2734	hh ll	8	—	—	—	—	—	—	—	—		
ASR	Arithmetic Shift Right		IND8, X	0D	ff	8	—	—	—	—	—	—	—			
			IND8, Y	1D	ff	8	—	—	—	—	—	—	—	—		
			IND8, Z	2D	ff	8	—	—	—	—	—	—	—	—		
			IND16, X	170D	gggg	8	—	—	—	—	—	—	—	—		
			IND16, Y	171D	gggg	8	—	—	—	—	—	—	—	—		
			IND16, Z	172D	gggg	8	—	—	—	—	—	—	—	—		
ASRA	Arithmetic Shift Right A		INH	370D	—	2	—	—	—	—	—	—				
ASRB	Arithmetic Shift Right B		INH	371D	—	2	—	—	—	—	—	—				
ASRD	Arithmetic Shift Right D		INH	27FD	—	2	—	—	—	—	—	—				
ASRE	Arithmetic Shift Right E		INH	277D	—	2	—	—	—	—	—	—				
ASRM	Arithmetic Shift Right AM		INH	27BA	—	4	—	—	—	—	—	—				
ASRW	Arithmetic Shift Right Word		IND16, X	270D	gggg	8	—	—	—	—	—	—	—			
			IND16, Y	271D	gggg	8	—	—	—	—	—	—	—			
			IND16, Z	272D	gggg	8	—	—	—	—	—	—	—			
			EXT	273D	hh ll	8	—	—	—	—	—	—	—			
BCC ⁴	Branch if Carry Clear	If C = 0, branch	REL8	B4	rr	6, 2	—	—	—	—	—	—	—			

Instruction Set Summary (Continued)

Mnemonic	Operation	Description	Address Mode	Instruction			Condition Codes									
				Opcod	Operand	Cycles	S	MV	H	EV	N	Z	V	C		
BRSET ⁴	Branch if Bit(s) Set	If (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 rrr	10, 14										
			IND16, Y	1B	mm gggg rr	10, 14										
			IND16, Z	2B	mm gggg rr	10, 14										
EXT	3B	mm hh ll rr	10, 14													
BSET	Set Bit(s)	(M) • (Mask) ⇒ M	IND16, X	09	mm gggg	8										
			IND16, Y	19	mm gggg	8										
			IND16, Z	29	mm gggg	8										
			EXT	39	mm hh ll	8										
			IND8, X	1709	mm ff	8										
			IND8, Y	1719	mm ff	8										
IND8, Z	1729	mm ff	8													
BSETW	Set Bit(s) in Word	(M : M + 1) • (Mask) ⇒ M : M + 1	IND16, X	2709	gggg mmmm	10										
			IND16, Y	2719	gggg mmmm	10										
			IND16, Z	2729	gggg mmmm	10										
			EXT	2739	hh ll mmmm	10										
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 ⁴	Branch if Overflow Clear	If V = 0, branch	REL8	B8	rr	6, 2										
BVS ⁴	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 Memory	\$00 ⇒ M	IND8, X	05	ff	4										
			IND8, Y	15	ff	4										
			IND8, Z	25	ff	4										
			IND16, X	1705	gggg	6										
			IND16, Y	1715	gggg	6										
			IND16, Z	1725	gggg	6										
EXT	1735	hh ll	6													
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	\$00000000 ⇒ AM[32:0]	INH	27B7	—	2		0		0						
CLRW	Clear Memory Word	\$0000 ⇒ M : M + 1	IND16, X	2705	gggg	6										
			IND16, Y	2715	gggg	6										
			IND16, Z	2725	gggg	6										
			EXT	2735	hh ll	6										
CMPA	Compare A to Memory	(A) - (M)	IND8, X	48	ff	6										
			IND8, Y	58	ff	6										
			IND8, Z	68	ff	6										
			IMM8	78	ii	2										
			IND16, X	1748	gggg	6										
			IND16, Y	1758	gggg	6										
			IND16, Z	1768	gggg	6										
			EXT	1778	hh ll	6										
			E, X	2748	—	6										
			E, Y	2758	—	6										
E, Z	2768	—	6													

Instruction Set Summary (Continued)

Mnemonic	Operation	Description	Address Mode	Instruction			Condition Codes								
				Opcode	Operand	Cycles	S	MV	H	EV	N	Z	V	C	
CMPB	Compare B to Memory	(B) - (M)	IND8, X	C8	ff	6						Δ	Δ	Δ	Δ
			IND8, Y	D8	ff	6									
			IND8, Z	E8	ff	6									
			IMM8	F8	ii	2									
			IND16, X	17C8	gggg	6									
			IND16, Y	17D8	gggg	6									
			IND16, Z	17E8	gggg	6									
			EXT	17F8	hh ll	6									
			E, X	27C8	—	6									
			E, Y	27D8	—	6									
			E, Z	27E8	—	6									
			COM	Ones Complement	\$FF - (M) ⇒ M	IND8, X	00	ff	8						Δ
IND8, Y	10	ff				8									
IND8, Z	20	ff				8									
IND16, X	1700	gggg				8									
IND16, Y	1710	gggg				8									
IND16, Z	1720	gggg				8									
EXT	1730	hh ll	8												
COMA	Ones Complement A	\$FF - (A) ⇒ A	INH	3700	—	2					Δ	Δ	0	1	
COMB	Ones Complement B	\$FF - (B) ⇒ B	INH	3710	—	2					Δ	Δ	0	1	
COMD	Ones Complement D	\$FFF - (D) ⇒ D	INH	27F0	—	2					Δ	Δ	0	1	
COME	Ones Complement E	\$FFF - (E) ⇒ E	INH	2770	—	2					Δ	Δ	0	1	
COMW	Ones Complement Word	\$FFF - M : M + 1 ⇒ M : M + 1	IND16, X	2700	gggg	8						Δ	Δ	0	1
			IND16, Y	2710	gggg	8									
			IND16, Z	2720	gggg	8									
			EXT	2730	hh ll	8									
CPD	Compare D to Memory	(D) - (M : M + 1)	IND8, X	88	ff	6						Δ	Δ	Δ	Δ
			IND8, Y	98	ff	6									
			IND8, Z	A8	ff	6									
			E, X	2788	—	6									
			E, Y	2798	—	6									
			E, Z	27A8	—	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									
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	hh ll	6									
CPS	Compare SP 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									
			IND16, X	174F	gggg	6									
			IND16, Y	175F	gggg	6									
			IND16, Z	176F	gggg	6									
			EXT	177F	hh ll	6									
IMM16	377F	jj kk	4												
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									
			IND16, X	174C	gggg	6									
			IND16, Y	175C	gggg	6									
			IND16, Z	176C	gggg	6									
			EXT	177C	hh ll	6									
			IMM16	377C	jj kk	4									
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									
			IND16, X	174D	gggg	6									
			IND16, Y	175D	gggg	6									
			IND16, Z	176D	gggg	6									
			EXT	177D	hh ll	6									
			IMM16	377D	jj kk	4									

5

Instruction Set Summary (Continued)

Mnemonic	Operation	Description	Address Mode	Instruction			Condition Codes													
				Opcod	Operand	Cycles	S	MV	H	EV	N	Z	V	C						
CPZ	Compare IZ to Memory	$(IZ) \sim (M : M + 1)$	IND8, X	4E	ff	6	—	—	—	—	—	—	—	—	—	—	—	—		
			IND8, Y	5E	ff	6	—	—	—	—	—	—	—	—	—	—	—	—		
			IND8, Z	6E	ff	6	—	—	—	—	—	—	—	—	—	—	—	—	—	
			IND16, X	174E	9999	6	—	—	—	—	—	—	—	—	—	—	—	—	—	
			IND16, Y	175E	9999	6	—	—	—	—	—	—	—	—	—	—	—	—	—	
			IND16, Z	176E	9999	6	—	—	—	—	—	—	—	—	—	—	—	—	—	—
			EXT	177E	hh ll	6	—	—	—	—	—	—	—	—	—	—	—	—	—	—
IMM16	377E	jj kk	4	—	—	—	—	—	—	—	—	—	—	—	—	—	—			
DAA	Decimal Adjust A	$(A)_{10}$	INH	3721	—	2	—	—	—	—	—	—	—	—	—	—	—	—		
DEC	Decrement Memory	$(M) - \$01 \Rightarrow M$	IND8, X	01	ff	8	—	—	—	—	—	—	—	—	—	—	—	—		
			IND8, Y	11	ff	8	—	—	—	—	—	—	—	—	—	—	—	—	—	
			IND8, Z	21	ff	8	—	—	—	—	—	—	—	—	—	—	—	—	—	—
			IND16, X	1701	9999	8	—	—	—	—	—	—	—	—	—	—	—	—	—	—
			IND16, Y	1711	9999	8	—	—	—	—	—	—	—	—	—	—	—	—	—	—
			IND16, Z	1721	9999	8	—	—	—	—	—	—	—	—	—	—	—	—	—	—
			EXT	1731	hh ll	8	—	—	—	—	—	—	—	—	—	—	—	—	—	—
DECA	Decrement A	$(A) - \$01 \Rightarrow A$	INH	3701	—	2	—	—	—	—	—	—	—	—	—	—	—	—		
DECB	Decrement B	$(B) - \$01 \Rightarrow B$	INH	3711	—	2	—	—	—	—	—	—	—	—	—	—	—	—		
DECW	Decrement Memory Word	$(M : M + 1) - \$0001 \Rightarrow M : M + 1$	IND16, X	2701	9999	8	—	—	—	—	—	—	—	—	—	—	—	—		
			IND16, Y	2711	9999	8	—	—	—	—	—	—	—	—	—	—	—	—	—	
			IND16, Z	2721	9999	8	—	—	—	—	—	—	—	—	—	—	—	—	—	
			EXT	2731	hh ll	8	—	—	—	—	—	—	—	—	—	—	—	—	—	
EDIV	Extended Unsigned Divide	$(E : D) / (IX)$ Quotient $\Rightarrow IX$ Remainder $\Rightarrow D$	INH	3728	—	24	—	—	—	—	—	—	—	—	—	—	—	—		
EDIVS	Extended Signed Divide	$(E : D) / (IX)$ Quotient $\Rightarrow IX$ Remainder $\Rightarrow ACCD$	INH	3729	—	38	—	—	—	—	—	—	—	—	—	—	—	—		
EMUL	Extended Unsigned Multiply	$(E) * (D) \Rightarrow E : D$	INH	3725	—	10	—	—	—	—	—	—	—	—	—	—	—	—		
EMULS	Extended Signed Multiply	$(E) * (D) \Rightarrow E : D$	INH	3726	—	8	—	—	—	—	—	—	—	—	—	—	—	—		
EORA	Exclusive OR A	$(A) \oplus (M) \Rightarrow A$	IND8, X	44	ff	6	—	—	—	—	—	—	—	—	—	—	—	—		
			IND8, Y	54	ff	6	—	—	—	—	—	—	—	—	—	—	—	—	—	
			IND8, Z	64	ff	6	—	—	—	—	—	—	—	—	—	—	—	—	—	
			IMM8	74	ii	2	—	—	—	—	—	—	—	—	—	—	—	—	—	—
			IND16, X	1744	9999	6	—	—	—	—	—	—	—	—	—	—	—	—	—	—
			IND16, Y	1754	9999	6	—	—	—	—	—	—	—	—	—	—	—	—	—	—
			IND16, Z	1764	9999	6	—	—	—	—	—	—	—	—	—	—	—	—	—	—
			EXT	1774	hh ll	6	—	—	—	—	—	—	—	—	—	—	—	—	—	—
			E, X	2744	—	6	—	—	—	—	—	—	—	—	—	—	—	—	—	—
			E, Y	2754	—	6	—	—	—	—	—	—	—	—	—	—	—	—	—	—
			E, Z	2764	—	6	—	—	—	—	—	—	—	—	—	—	—	—	—	—
			IND8, X	C4	ff	6	—	—	—	—	—	—	—	—	—	—	—	—	—	—
			IND8, Y	D4	ff	6	—	—	—	—	—	—	—	—	—	—	—	—	—	—
IND8, Z	E4	ff	6	—	—	—	—	—	—	—	—	—	—	—	—	—	—			
IMM8	F4	ii	2	—	—	—	—	—	—	—	—	—	—	—	—	—	—			
IND16, X	17C4	9999	6	—	—	—	—	—	—	—	—	—	—	—	—	—	—			
IND16, Y	17D4	9999	6	—	—	—	—	—	—	—	—	—	—	—	—	—	—			
IND16, Z	17E4	9999	6	—	—	—	—	—	—	—	—	—	—	—	—	—	—			
EXT	17F4	hh ll	6	—	—	—	—	—	—	—	—	—	—	—	—	—	—			
E, X	27C4	—	6	—	—	—	—	—	—	—	—	—	—	—	—	—	—			
E, Y	27D4	—	6	—	—	—	—	—	—	—	—	—	—	—	—	—	—			
E, Z	27E4	—	6	—	—	—	—	—	—	—	—	—	—	—	—	—	—			
EORD	Exclusive OR D	$(D) \oplus (M : M + 1) \Rightarrow D$	IND8, X	84	ff	6	—	—	—	—	—	—	—	—	—	—	—	—		
			IND8, Y	94	ff	6	—	—	—	—	—	—	—	—	—	—	—	—	—	
			IND8, Z	A4	ff	6	—	—	—	—	—	—	—	—	—	—	—	—	—	
			E, X	2784	—	6	—	—	—	—	—	—	—	—	—	—	—	—	—	—
			E, Y	2794	—	6	—	—	—	—	—	—	—	—	—	—	—	—	—	—
			E, Z	27A4	—	6	—	—	—	—	—	—	—	—	—	—	—	—	—	—
			IMM16	37B4	jjkk	4	—	—	—	—	—	—	—	—	—	—	—	—	—	—
			IND16, X	37C4	9999	6	—	—	—	—	—	—	—	—	—	—	—	—	—	—
			IND16, Y	37D4	9999	6	—	—	—	—	—	—	—	—	—	—	—	—	—	—
			IND16, Z	37E4	9999	6	—	—	—	—	—	—	—	—	—	—	—	—	—	—
			EXT	37F4	hhll	6	—	—	—	—	—	—	—	—	—	—	—	—	—	—
EORE	Exclusive OR E	$(E) \oplus (M : M + 1) \Rightarrow E$	IMM16	3734	jj kk	4	—	—	—	—	—	—	—	—	—	—	—	—		
			IND16, X	3744	9999	6	—	—	—	—	—	—	—	—	—	—	—	—	—	
			IND16, Y	3754	9999	6	—	—	—	—	—	—	—	—	—	—	—	—	—	
			IND16, Z	3764	9999	6	—	—	—	—	—	—	—	—	—	—	—	—	—	
			EXT	3774	hh ll	6	—	—	—	—	—	—	—	—	—	—	—	—	—	—

Instruction Set Summary (Continued)

Mnemonic	Operation	Description	Address Mode	Instruction			Condition Codes													
				Opcode	Operand	Cycles	S	MV	H	EV	N	Z	V	C						
LDAA	Load A	(M) ⇒ A	IND8, X	45	ff	6	—	—	—	—	—	—	—	Δ	Δ	0	—			
			IND8, Y	55	ff	6	—	—	—	—	—	—	—	—	—	—	—	—		
			IND8, Z	65	ff	6	—	—	—	—	—	—	—	—	—	—	—	—		
			IMM8	75	ii	2	—	—	—	—	—	—	—	—	—	—	—	—		
			IND16, X	1745	gggg	6	—	—	—	—	—	—	—	—	—	—	—	—		
			IND16, Y	1755	gggg	6	—	—	—	—	—	—	—	—	—	—	—	—		
			IND16, Z	1765	gggg	6	—	—	—	—	—	—	—	—	—	—	—	—		
			EXT	1775	hh ll	6	—	—	—	—	—	—	—	—	—	—	—	—		
			E, X	2745	—	6	—	—	—	—	—	—	—	—	—	—	—	—		
			E, Y	2755	—	6	—	—	—	—	—	—	—	—	—	—	—	—		
			E, Z	2765	—	6	—	—	—	—	—	—	—	—	—	—	—	—		
LDAB	Load B	(M) ⇒ B	IND8, X	C5	ff	6	—	—	—	—	—	—	—	Δ	Δ	0	—			
			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	—	—	—	—	—	—	—	—	—	—	—			
			E, X	2785	—	6	—	—	—	—	—	—	—	—	—	—	—			
			E, Y	2795	—	6	—	—	—	—	—	—	—	—	—	—	—			
			E, Z	27A5	—	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	—	—	—	—	—	—	—	—	—	—	—			
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	—	—	—	—	—	—	—	—	—	—				
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	—	—	—	—	—	—	—	—	—	—	—
IND16, X	17CC	gggg				6	—	—	—	—	—	—	—	—	—	—	—			
IND16, Y	17DC	gggg				6	—	—	—	—	—	—	—	—	—	—	—			
IND16, Z	17EC	gggg				6	—	—	—	—	—	—	—	—	—	—	—			
EXT	17FC	hh ll				6	—	—	—	—	—	—	—	—	—	—	—			
IMM16	37BC	jj kk				4	—	—	—	—	—	—	—	—	—	—	—			
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	—	—	—	—	—	—	—	—	—	—	—			
			IND16, X	17CD	gggg	6	—	—	—	—	—	—	—	—	—	—	—			
			IND16, Y	17DD	gggg	6	—	—	—	—	—	—	—	—	—	—	—			
			IND16, Z	17ED	gggg	6	—	—	—	—	—	—	—	—	—	—	—			
			EXT	17FD	hh ll	6	—	—	—	—	—	—	—	—	—	—	—			
			IMM16	37BD	jj kk	4	—	—	—	—	—	—	—	—	—	—	—			
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	—	—	—	—	—	—	—	—	—	—	—			
			IND16, X	17CE	gggg	6	—	—	—	—	—	—	—	—	—	—	—			
			IND16, Y	17DE	gggg	6	—	—	—	—	—	—	—	—	—	—	—			
			IND16, Z	17EE	gggg	6	—	—	—	—	—	—	—	—	—	—	—			
			EXT	17FE	hh ll	6	—	—	—	—	—	—	—	—	—	—	—			
			IMM16	37BE	jj kk	4	—	—	—	—	—	—	—	—	—	—	—			

Instruction Set Summary (Continued)

Mnemonic	Operation	Description	Address Mode	Instruction			Condition Codes									
				Opcode	Operand	Cycles	S	MV	H	EV	N	Z	V	C		
LPSTOP	Low Power Stop	If S then STOP else NOP	INH	27F1	—	4, 20	—	—	—	—	—	—	—	—	—	—
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 9999 9999 9999 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	9999 9999 9999 hh ll	8 8 8 8	—	—	—	—	—	—	0	Δ	Δ	Δ
MAC	Multiply and Accumulate Signed 16-Bit Fractions	(HR) * (IR) ⇒ E : D (AM) + (E : D) ⇒ AM Qualified (IX) ⇒ IX Qualified (IY) ⇒ IY (HR) ⇒ IZ (M : M + 1)X ⇒ HR (M : M + 1)Y ⇒ IR	IMM8	7B	xoyo	12	—	Δ	—	Δ	—	—	Δ	—	—	—
MOVB	Move Byte	(M ₁) ⇒ 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) ⇒ 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) ⇒ D	INH	3724	—	10	—	—	—	—	—	—	—	—	—	Δ
NEG	Negate Memory	\$00 - (M) ⇒ M	IND8, X IND8, Y IND8, Z IND16, X IND16, Y IND16, Z EXT	02 12 22 1702 1712 1722 1732	ff ff ff 9999 9999 9999 hh ll	8 8 8 8 8 8 8	—	—	—	—	—	—	Δ	Δ	Δ	Δ
NEGA	Negate A	\$00 - (A) ⇒ A	INH	3702	—	2	—	—	—	—	—	—	Δ	Δ	Δ	Δ
NEGB	Negate B	\$00 - (B) ⇒ B	INH	3712	—	2	—	—	—	—	—	—	Δ	Δ	Δ	Δ
NEGD	Negate D	\$0000 - (D) ⇒ D	INH	27F2	—	2	—	—	—	—	—	—	Δ	Δ	Δ	Δ
NEGE	Negate E	\$0000 - (E) ⇒ E	INH	2772	—	2	—	—	—	—	—	—	Δ	Δ	Δ	Δ
NEGW	Negate Memory Word	\$0000 - (M : M + 1) ⇒ M : M + 1	IND16, X IND16, Y IND16, Z EXT	2702 2712 2722 2732	9999 9999 9999 hh ll	8 8 8 8	—	—	—	—	—	—	Δ	Δ	Δ	Δ
NOP	Null Operation	—	INH	274C	—	2	—	—	—	—	—	—	—	—	—	—
ORAA	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	47 57 67 77 1747 1757 1767 1777 2747 2757 2767	ff ff ff ii 9999 9999 9999 hh ll — — —	6 6 6 2 6 6 6 6 6 6 6	—	—	—	—	—	Δ	Δ	0	—	

Instruction Set Summary (Continued)

Mnemonic	Operation	Description	Address Mode	Instruction			Condition Codes								
				Opcode	Operand	Cycles	S	MV	H	EV	N	Z	V	C	
ROL	Rotate Left		IND8, X	0C	ff	8	—	—	—	—	Δ	Δ	Δ	Δ	
			IND8, Y	1C	ff	8	—	—	—	—	—	—	—	—	
			IND8, Z	2C	ff	8	—	—	—	—	—	—	—	—	
			IND16, X	170C	9999	8	—	—	—	—	—	—	—	—	
			IND16, Y	171C	9999	8	—	—	—	—	—	—	—	—	
IND16, Z	172C	9999	8	—	—	—	—	—	—	—	—	—			
EXT	173C	hh ll	8	—	—	—	—	—	—	—	—	—	—		
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	270C	9999	8	—	—	—	—	Δ	Δ	Δ	Δ	
			IND16, Y	271C	9999	8	—	—	—	—	—	—	—	—	
			IND16, Z	272C	9999	8	—	—	—	—	—	—	—	—	
			EXT	273C	hh ll	8	—	—	—	—	—	—	—	—	—
ROR	Rotate Right		IND8, X	0E	ff	8	—	—	—	—	Δ	Δ	Δ	Δ	
			IND8, Y	1E	ff	8	—	—	—	—	—	—	—	—	
			IND8, Z	2E	ff	8	—	—	—	—	—	—	—	—	
			IND16, X	170E	9999	8	—	—	—	—	—	—	—	—	
			IND16, Y	171E	9999	8	—	—	—	—	—	—	—	—	
			IND16, Z	172E	9999	8	—	—	—	—	—	—	—	—	
EXT	173E	hh ll	8	—	—	—	—	—	—	—	—	—			
RORA	Rotate Right A		INH	370E	—	2	—	—	—	—	Δ	Δ	Δ	Δ	
RORB	Rotate Right B		INH	371E	—	2	—	—	—	—	Δ	Δ	Δ	Δ	
RORD	Rotate Right D		INH	27FE	—	2	—	—	—	—	Δ	Δ	Δ	Δ	
RORE	Rotate Right E		INH	277E	—	2	—	—	—	—	Δ	Δ	Δ	Δ	
RORW	Rotate Right Word		IND16, X	270E	9999	8	—	—	—	—	Δ	Δ	Δ	Δ	
			IND16, Y	271E	9999	8	—	—	—	—	—	—	—	—	
			IND16, Z	272E	9999	8	—	—	—	—	—	—	—	—	
			EXT	273E	hh ll	8	—	—	—	—	—	—	—	—	—
RTI ²	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		
RTS ³	Return from Subroutine	(SK : SP) + 2 ⇒ SK : SP Pull PK (SK : SP) + 2 ⇒ SK : SP Pull PC (PK : PC) - 2 ⇒ PK : PC	INH	27F7	—	12	—	—	—	—	—	—	—		
SBA	Subtract B from A	(A) - (B) ⇒ A	INH	370A	—	2	—	—	—	—	Δ	Δ	Δ	Δ	
SBCA	Subtract with Carry from A	(A) - (M) - C ⇒ A	IND8, X	42	ff	6	—	—	—	—	Δ	Δ	Δ	Δ	
			IND8, Y	52	ff	6	—	—	—	—	—	—	—	—	
			IND8, Z	62	ff	6	—	—	—	—	—	—	—	—	
			IMM8	72	ii	2	—	—	—	—	—	—	—	—	
			IND16, X	1742	9999	6	—	—	—	—	—	—	—	—	
			IND16, Y	1752	9999	6	—	—	—	—	—	—	—	—	
			IND16, Z	1762	9999	6	—	—	—	—	—	—	—	—	
			EXT	1772	hh ll	6	—	—	—	—	—	—	—	—	—
			E, X	2742	—	6	—	—	—	—	—	—	—	—	—
E, Y	2752	—	6	—	—	—	—	—	—	—	—	—			
E, Z	2762	—	6	—	—	—	—	—	—	—	—	—			

Instruction Set Summary (Continued)

Mnemonic	Operation	Description	Address Mode	Instruction			Condition Codes							
				Opcode	Operand	Cycles	S	MV	H	EV	N	Z	V	C
SBCB	Subtract with Carry from B	$(B) - (M) - C \Rightarrow B$	IND8, X	C2	ff	6	—	—	—	—	Δ	Δ	Δ	Δ
			IND8, Y	D2	ff	6								
			IND8, Z	E2	ff	6								
			IMM8	F2	ii	2								
			IND16, X	17C2	9999	6								
			IND16, Y	17D2	9999	6								
			IND16, Z	17E2	9999	6								
			EXT	17F2	hh ll	6								
			E, X	27C2	—	6								
			E, Y	27D2	—	6								
			E, Z	27E2	—	6								
SBCD	Subtract with Carry from D	$(D) - (M : M + 1) - C \Rightarrow D$	IND8, X	82	ff	6	—	—	—	—	Δ	Δ	Δ	Δ
			IND8, Y	92	ff	6								
			IND8, Z	A2	ff	6								
			E, X	2782	—	6								
			E, Y	2792	—	6								
			E, Z	27A2	—	6								
			IMM16	37B2	jj kk	4								
			IND16, X	37C2	9999	6								
			IND16, Y	37D2	9999	6								
			IND16, Z	37E2	9999	6								
			EXT	37F2	hh ll	6								
SBCE	Subtract with Carry from E	$(E) - (M : M + 1) - C \Rightarrow E$	IMM16	3732	jj kk	4	—	—	—	—	Δ	Δ	Δ	Δ
			IND16, X	3742	9999	6								
			IND16, Y	3752	9999	6								
			IND16, Z	3762	9999	6								
EXT	3772	hh ll	6											
SDE	Subtract D from E	$(E) - (D) \Rightarrow E$	INH	2779	—	2	—	—	—	—	Δ	Δ	Δ	Δ
STAA	Store A	$(A) \Rightarrow M$	IND8, X	4A	ff	4	—	—	—	—	Δ	Δ	0	—
			IND8, Y	5A	ff	4								
			IND8, Z	6A	ff	4								
			IND16, X	174A	9999	6								
			IND16, Y	175A	9999	6								
			IND16, Z	176A	9999	6								
			EXT	177A	hh ll	6								
			E, X	274A	—	4								
			E, Y	275A	—	4								
			E, Z	276A	—	4								
			STAB	Store B	$(B) \Rightarrow M$	IND8, X	CA	ff	4	—	—	—	—	Δ
IND8, Y	DA	ff				4								
IND8, Z	EA	ff				4								
IND16, X	17CA	9999				6								
IND16, Y	17DA	9999				6								
IND16, Z	17EA	9999				6								
EXT	17FA	hh ll				6								
E, X	27CA	—				4								
E, Y	27DA	—				4								
E, Z	27EA	—				4								
STD	Store D	$(D) \Rightarrow M : M + 1$				IND8, X	8A	ff	6	—	—	—	—	Δ
			IND8, Y	9A	ff	6								
			IND8, Z	AA	ff	6								
			E, X	278A	—	6								
			E, Y	279A	—	6								
			E, Z	27AA	—	6								
			IND16, X	37CA	9999	4								
			IND16, Y	37DA	9999	4								
			IND16, Z	37EA	9999	4								
			EXT	37FA	hh ll	6								
			STE	Store E	$(E) \Rightarrow M : M + 1$	IND16, X	374A	9999	6	—	—	—	—	Δ
IND16, Y	375A	9999				6								
IND16, Z	376A	9999				6								
EXT	377A	hh ll				6								
STED	Store Concatenated D and E	$(E) \Rightarrow M : M + 1$ $(D) \Rightarrow M + 2 : M + 3$	EXT	2773	hh ll	8	—	—	—	—	—	—	—	
STS	Store SP	$(SP) \Rightarrow M : M + 1$	IND8, X	8F	ff	4	—	—	—	—	Δ	Δ	0	—
			IND8, Y	9F	ff	4								
			IND8, Z	AF	ff	4								
			IND16, X	178F	9999	6								
			IND16, Y	179F	9999	6								
			IND16, Z	17AF	9999	6								
			EXT	17BF	hh ll	6								

5

Instruction Set Summary (Continued)

Mnemonic	Operation	Description	Address Mode	Instruction			Condition Codes							
				Opcode	Operand	Cycles	S	MV	H	EV	N	Z	V	C
STX	Store IX	(IX) ⇒ M : M + 1	IND8, X	8C	ff	4	—	—	—	—	Δ	Δ	0	—
			IND8, Y	9C	ff	4	—	—	—	—	Δ	Δ	0	—
			IND8, Z	AC	ff	4	—	—	—	—	Δ	Δ	0	—
			IND16, X	178C	9999	6	—	—	—	—	Δ	Δ	0	—
			IND16, Y	179C	9999	6	—	—	—	—	Δ	Δ	0	—
			IND16, Z	17AC	9999	6	—	—	—	—	Δ	Δ	0	—
			EXT	17BC	hh ll	6	—	—	—	—	Δ	Δ	0	—
STY	Store IY	(IY) ⇒ M : M + 1	IND8, X	8D	ff	4	—	—	—	—	Δ	Δ	0	—
			IND8, Y	9D	ff	4	—	—	—	—	Δ	Δ	0	—
			IND8, Z	AD	ff	4	—	—	—	—	Δ	Δ	0	—
			IND16, X	178D	9999	6	—	—	—	—	Δ	Δ	0	—
			IND16, Y	179D	9999	6	—	—	—	—	Δ	Δ	0	—
			IND16, Z	17AD	9999	6	—	—	—	—	Δ	Δ	0	—
			EXT	17BD	hh ll	6	—	—	—	—	Δ	Δ	0	—
STZ	Store Z	(IZ) ⇒ M : M + 1	IND8, X	8E	ff	4	—	—	—	—	Δ	Δ	0	—
			IND8, Y	9E	ff	4	—	—	—	—	Δ	Δ	0	—
			IND8, Z	AE	ff	4	—	—	—	—	Δ	Δ	0	—
			IND16, X	178E	9999	6	—	—	—	—	Δ	Δ	0	—
			IND16, Y	179E	9999	6	—	—	—	—	Δ	Δ	0	—
			IND16, Z	17AE	9999	6	—	—	—	—	Δ	Δ	0	—
			EXT	17BE	hh ll	6	—	—	—	—	Δ	Δ	0	—
SUBA	Subtract from A	(A) - (M) ⇒ A	IND8, X	40	ff	6	—	—	—	—	Δ	Δ	Δ	Δ
			IND8, Y	50	ff	6	—	—	—	—	Δ	Δ	Δ	Δ
			IND8, Z	60	ff	6	—	—	—	—	Δ	Δ	Δ	Δ
			IMM8	70	ii	2	—	—	—	—	Δ	Δ	Δ	Δ
			IND16, X	1740	9999	6	—	—	—	—	Δ	Δ	Δ	Δ
			IND16, Y	1750	9999	6	—	—	—	—	Δ	Δ	Δ	Δ
			IND16, Z	1760	9999	6	—	—	—	—	Δ	Δ	Δ	Δ
			EXT	1770	hh ll	6	—	—	—	—	Δ	Δ	Δ	Δ
			E, X	2740	—	6	—	—	—	—	Δ	Δ	Δ	Δ
			E, Y	2750	—	6	—	—	—	—	Δ	Δ	Δ	Δ
SUBB	Subtract from B	(B) - (M) ⇒ B	IND8, X	C0	ff	6	—	—	—	—	Δ	Δ	Δ	Δ
			IND8, Y	D0	ff	6	—	—	—	—	Δ	Δ	Δ	Δ
			IND8, Z	E0	ff	6	—	—	—	—	Δ	Δ	Δ	Δ
			IMM8	F0	ii	2	—	—	—	—	Δ	Δ	Δ	Δ
			IND16, X	17C0	9999	6	—	—	—	—	Δ	Δ	Δ	Δ
			IND16, Y	17D0	9999	6	—	—	—	—	Δ	Δ	Δ	Δ
			IND16, Z	17E0	9999	6	—	—	—	—	Δ	Δ	Δ	Δ
			EXT	17F0	hh ll	6	—	—	—	—	Δ	Δ	Δ	Δ
			E, X	27C0	—	6	—	—	—	—	Δ	Δ	Δ	Δ
			E, Y	27D0	—	6	—	—	—	—	Δ	Δ	Δ	Δ
SUBD	Subtract from D	(D) - (M : M + 1) ⇒ D	IND8, X	80	ff	6	—	—	—	—	Δ	Δ	Δ	Δ
			IND8, Y	90	ff	6	—	—	—	—	Δ	Δ	Δ	Δ
			IND8, Z	A0	ff	6	—	—	—	—	Δ	Δ	Δ	Δ
			E, X	2780	—	6	—	—	—	—	Δ	Δ	Δ	Δ
			E, Y	2790	—	6	—	—	—	—	Δ	Δ	Δ	Δ
			E, Z	27A0	—	6	—	—	—	—	Δ	Δ	Δ	Δ
			IMM16	37B0	jj kk	4	—	—	—	—	Δ	Δ	Δ	Δ
			IND16, X	37C0	9999	6	—	—	—	—	Δ	Δ	Δ	Δ
			IND16, Y	37D0	9999	6	—	—	—	—	Δ	Δ	Δ	Δ
			IND16, Z	37E0	9999	6	—	—	—	—	Δ	Δ	Δ	Δ
SUBE	Subtract from E	(E) - (M : M + 1) ⇒ E	EXT	37F0	hh ll	6	—	—	—	—	Δ	Δ	Δ	Δ
			IMM16	3730	jj kk	4	—	—	—	—	Δ	Δ	Δ	Δ
			IND16, X	3740	9999	6	—	—	—	—	Δ	Δ	Δ	Δ
			IND16, Y	3750	9999	6	—	—	—	—	Δ	Δ	Δ	Δ
			IND16, Z	3760	9999	6	—	—	—	—	Δ	Δ	Δ	Δ
SWI	Software Interrupt	(PK : PC) + 2 ⇒ PK : PC Push (PC) (SK : SP) - 2 ⇒ SK : SP Push (CCR) (SK : SP) - 2 ⇒ SK : SP \$0 ⇒ PK SWI Vector ⇒ PC	INH	3720	—	16	—	—	—	—	—	—	—	—
SXT	Sign Extend B into A	If B7 = 1 then A = \$FF else A = \$00	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) ⇒ EK	INH	27FA	—	2	—	—	—	—	—	—	—	—

5

Instruction Set Summary (Continued)

Mnemonic	Operation	Description	Address Mode	Instruction			Condition Codes											
				Opcode	Operand	Cycles	S	MV	H	EV	N	Z	V	C				
TBSK	Transfer B to SK	(B) ⇒ SK	INH	379F	—	2	—	—	—	—	—	—	—	—	—	—	—	
TBXK	Transfer B to XK	(B) ⇒ XK	INH	379C	—	2	—	—	—	—	—	—	—	—	—	—	—	
TBYK	Transfer B to YK	(B) ⇒ YK	INH	379D	—	2	—	—	—	—	—	—	—	—	—	—	—	
TBZK	Transfer B to ZK	(B) ⇒ ZK	INH	379E	—	2	—	—	—	—	—	—	—	—	—	—	—	
TDE	Transfer D to E	(D) ⇒ E	INH	277B	—	2	—	—	—	—	—	—	—	—	—	—	—	
TDMASK	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	Δ	Δ	Δ	
TED	Transfer E to D	(E) ⇒ D	INH	27FB	—	2	—	—	—	—	—	—	—	—	—	—	—	
TEDM	Transfer E and D to AM[31:0] Sign Extend AM	(D) ⇒ AM[15:0] (E) ⇒ AM[31:16] AM[35:32] = AM31	INH	27B1	—	4	—	0	—	0	—	—	—	—	—	—	—	
TEKB	Transfer EK to B	\$0 ⇒ B[7:4] (EK) ⇒ B[3:0]	INH	27BB	—	2	—	—	—	—	—	—	—	—	—	—	—	
TEM	Transfer E to AM[31:16] Sign Extend AM Clear AM LSB	(E) ⇒ AM[31:16] \$0 ⇒ AM[15:0] AM[35:32] = AM31	INH	27B2	—	4	—	0	—	0	—	—	—	—	—	—	—	
TMER	Transfer AM to E Rounded	Rounded (AM) ⇒ Temp If (SM • (EV + MV)) then Saturation ⇒ E else Temp[31:16] ⇒ E	INH	27B4	—	6	—	—	Δ	—	—	Δ	Δ	—	—	—	—	
TMET	Transfer AM to E Truncated	If (SM • (EV + MV)) then Saturation ⇒ E else AM[31:16] ⇒ E	INH	27B5	—	2	—	—	—	—	—	—	—	—	—	—	—	
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 MSB 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 for 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 9999 9999 9999 hh ll	6 6 6 6 6 6 6	—	—	—	—	—	—	—	—	—	—	—	—
TSTA	Test A for Zero or Minus	(A) - \$00	INH	3706	—	2	—	—	—	—	—	—	—	—	—	—	—	—
TSTB	Test B for Zero or Minus	(B) - \$00	INH	3716	—	2	—	—	—	—	—	—	—	—	—	—	—	—
TSTD	Test D for Zero or Minus	(D) - \$0000	INH	27F6	—	2	—	—	—	—	—	—	—	—	—	—	—	—
TSTE	Test E for Zero or Minus	(E) - \$0000	INH	2776	—	2	—	—	—	—	—	—	—	—	—	—	—	—
TSTW	Test for Zero or Minus Word	(M : M + 1) - \$0000	IND16, X IND16, Y IND16, Z EXT	2706 2716 2726 2736	9999 9999 9999 hh ll	6 6 6 6	—	—	—	—	—	—	—	—	—	—	—	—
TSX	Transfer SP to X	(SK : SP) + 2 ⇒ XK : IX	INH	274F	—	2	—	—	—	—	—	—	—	—	—	—	—	—
TSY	Transfer SP to Y	(SK : SP) + 2 ⇒ YK : IY	INH	275F	—	2	—	—	—	—	—	—	—	—	—	—	—	—
TSZ	Transfer SP to Z	(SK : SP) + 2 ⇒ ZK : IZ	INH	276F	—	2	—	—	—	—	—	—	—	—	—	—	—	—
TXKB	Transfer XK to B	\$0 ⇒ B[7:4] (XK) ⇒ B[3:0]	INH	37AC	—	2	—	—	—	—	—	—	—	—	—	—	—	—
TXS	Transfer X to SP	(XK : IX) - 2 ⇒ SK : SP	INH	374E	—	2	—	—	—	—	—	—	—	—	—	—	—	—
TXY	Transfer X to Y	(XK : IX) ⇒ YK : IY	INH	275C	—	2	—	—	—	—	—	—	—	—	—	—	—	—
TXZ	Transfer X to Z	(XK : IX) ⇒ ZK : IZ	INH	276C	—	2	—	—	—	—	—	—	—	—	—	—	—	—
TYKB	Transfer YK to B	\$0 ⇒ B[7:4] (YK) ⇒ B[3:0]	INH	37AD	—	2	—	—	—	—	—	—	—	—	—	—	—	—
TYS	Transfer Y to SP	(YK : IY) - 2 ⇒ SK : SP	INH	375E	—	2	—	—	—	—	—	—	—	—	—	—	—	—
TYX	Transfer Y to X	(YK : IY) ⇒ XK : IX	INH	274D	—	2	—	—	—	—	—	—	—	—	—	—	—	—

Instruction Set Summary (Concluded)

Mnemonic	Operation	Description	Address Mode	Instruction			Condition Codes									
				Opcod	Operand	Cycles	S	MV	H	EV	N	Z	V	C		
TYZ	Transfer Y to Z	(YK : IY) ⇒ ZK : IZ	INH	276D	—	2	—	—	—	—	—	—	—	—	—	—
TZKB	Transfer ZK to B	\$0 ⇒ B[7:4] (ZK) ⇒ B[3:0]	INH	37AE	—	2	—	—	—	—	—	—	—	—	—	—
TZS	Transfer Z to SP	(ZK : IZ) - 2 ⇒ SK : SP	INH	376E	—	2	—	—	—	—	—	—	—	—	—	—
TZX	Transfer Z to X	(ZK : IZ) ⇒ XK : IX	INH	274E	—	2	—	—	—	—	—	—	—	—	—	—
TZY	Transfer Z to Y	(ZK : IZ) ⇒ ZK : IY	INH	275E	—	2	—	—	—	—	—	—	—	—	—	—
WAI	Wait for Interrupt	WAIT	INH	27F3	—	8	—	—	—	—	—	—	—	—	—	—
XGAB	Exchange A with B	(A) ⇔ (B)	INH	371A	—	2	—	—	—	—	—	—	—	—	—	—
XGDE	Exchange D with E	(D) ⇔ (E)	INH	277A	—	2	—	—	—	—	—	—	—	—	—	—
XGDX	Exchange D with X	(D) ⇔ (IX)	INH	37CC	—	2	—	—	—	—	—	—	—	—	—	—
XGDY	Exchange D with Y	(D) ⇔ (IY)	INH	37DC	—	2	—	—	—	—	—	—	—	—	—	—
XGDZ	Exchange D with Z	(D) ⇔ (IZ)	INH	37EC	—	2	—	—	—	—	—	—	—	—	—	—
XGEX	Exchange E with X	(E) ⇔ (IX)	INH	374C	—	2	—	—	—	—	—	—	—	—	—	—
XGEY	Exchange E with Y	(E) ⇔ (IY)	INH	375C	—	2	—	—	—	—	—	—	—	—	—	—
XGEZ	Exchange E with Z	(E) ⇔ (IZ)	INH	376C	—	2	—	—	—	—	—	—	—	—	—	—

NOTES:

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

Instruction Set Abbreviations and Symbols

A	— Accumulator A	X	— Register used in operation
AM	— Accumulator M	M	— Address of one memory byte
B	— Accumulator B	M + 1	— Address of byte at M + \$0001
CCR	— Condition code register	M : M + 1	— Address of one memory word
D	— Accumulator D	(...)X	— Contents of address pointed to by IX
E	— Accumulator E	(...)Y	— Contents of address pointed to by IY
EK	— Extended addressing extension field	(...)Z	— Contents of address pointed to by IZ
IR	— MAC multiplicand register	E, X	— IX with E offset
HR	— MAC multiplier register	E, Y	— IY with E offset
IX	— Index register X	E, Z	— IZ with E offset
IY	— Index register Y	EXT	— Extended
IZ	— Index register Z	EXT20	— 20-bit extended
K	— Address extension register	IMM8	— 8-bit immediate
PC	— Program counter	IMM16	— 16-bit immediate
PK	— Program counter extension field	IND8, X	— IX with unsigned 8-bit offset
SK	— Stack pointer extension field	IND8, Y	— IY with unsigned 8-bit offset
SL	— Multiply and accumulate sign latch	IND8, Z	— IZ with unsigned 8-bit offset
SP	— Stack pointer	IND16, X	— IX with signed 16-bit offset
XK	— Index register X extension field	IND16, Y	— IY with signed 16-bit offset
YK	— Index register Y extension field	IND16, Z	— IZ with signed 16-bit offset
ZK	— Index register Z extension field	IND20, X	— IX with signed 20-bit offset
XMSK	— Modulo addressing index register X mask	IND20, Y	— IY with signed 20-bit offset
YMSK	— Modulo addressing index register Y mask	IND20, Z	— IZ with signed 20-bit offset
S	— Stop disable control bit	INH	— Inherent
MV	— AM overflow indicator	IXP	— Post-modified indexed
H	— Half carry indicator	REL8	— 8-bit relative
EV	— AM extended overflow indicator	REL16	— 16-bit relative
N	— Negative indicator	b	— 4-bit address extension
Z	— Zero indicator	ff	— 8-bit unsigned offset
V	— Two's complement overflow indicator	gggg	— 16-bit signed offset
C	— Carry/borrow indicator	hh	— High byte of 16-bit extended address
IP	— Interrupt priority field	ii	— 8-bit immediate data
SM	— Saturation mode control bit	jj	— High byte of 16-bit immediate data
PK	— Program counter extension field	kk	— Low byte of 16-bit immediate data
—	— Bit not affected	ll	— Low byte of 16-bit extended address
Δ	— Bit changes as specified	mm	— 8-bit mask
0	— Bit cleared	mmmm	— 16-bit mask
1	— Bit set	rr	— 8-bit unsigned relative offset
M	— Memory location used in operation	rrrr	— 16-bit signed relative offset
R	— Result of operation	xo	— MAC index register X offset
S	— Source data	yo	— MAC index register Y offset
		z	— 4-bit zero extension
+	— Addition	•	— AND
-	— Subtraction or negation (two's complement)	+	— Inclusive OR (OR)
*	— Multiplication	⊕	— Exclusive OR (EOR)
/	— Division	NOT	— Complementation
>	— Greater	:	— Concatenation
<	— Less	⇒	— Transferred
=	— Equal	↔	— Exchanged
≥	— Equal or greater	±	— Sign bit; also used to show tolerance
≤	— Equal or less	“	— Sign extension
≠	— Not equal	%	— Binary value
		\$	— Hexadecimal value

5.8 Comparison of CPU16 and M68HC11 CPU Instruction Sets

Most M68HC11 CPU instructions are a source-code compatible subset of the CPU16 instruction set. However, certain M68HC11 CPU instructions have been replaced by functionally equivalent CPU16 instructions, and some CPU16 instructions with the same mnemonics as M68HC11 CPU instructions operate differently.

Table 5-2 shows M68HC11 CPU instructions that either have been replaced by CPU16 instructions or that operate differently in the CPU16. Replacement instructions are not identical to M68HC11 CPU instructions. M68HC11 code must be altered to establish proper preconditions.

All CPU16 instruction execution times differ from those of the M68HC11. Motorola Programming Note M68HC16PN01/D, *Transporting M68HC11 Code to M68HC16 Devices*, contains detailed information about differences between the two instruction sets. Refer to the *CPU16 Reference Manual (CPU16RM/AD)* for further details about CPU operations.

Table 5-2. CPU16 Implementation of M68HC11 CPU Instructions

M68HC11 Instruction	CPU16 Implementation
BHS	BCC only
BLO	BCS only
BSR	Generates a different stack frame
CLC	Replaced by ANDP
CLI	Replaced by ANDP
CLV	Replaced by ANDP
DES	Replaced by AIS
DEX	Replaced by AIX
DEY	Replaced by AIY
INS	Replaced by AIS
INX	Replaced by AIX
INY	Replaced by AIY
JMP	IND8 addressing modes replaced by IND20 and EXT modes
JSR	IND8 addressing modes replaced by IND20 and EXT modes Generates a different stack frame
LSL, LSLD	Use ASL instructions*
PSHX	Replaced by PSHM
PSHY	Replaced by PSHM
PULX	Replaced by PULM
PULY	Replaced by PULM
RTI	Reloads PC and CCR only
RTS	Uses two-word stack frame
SEC	Replaced by ORP
SEI	Replaced by ORP
SEV	Replaced by ORP
STOP	Replaced by LPSTOP
TAP	CPU16 CCR bits differ from M68HC11 CPU16 interrupt priority scheme differs from M68HC11
TPA	CPU16 CCR bits differ from M68HC11 CPU16 interrupt priority scheme differs from M68HC11
TSX	Adds 2 to SK : SP before transfer to XK : IX
TSY	Adds 2 to SK : SP before transfer to YK : IY
TXS	Subtracts 2 from XK : IX before transfer to SK : SP
TXY	Transfers XK field to YK field
TYS	Subtracts 2 from YK : IY before transfer to SK : SP
TYX	Transfers YK field to XK field
WAI	Waits indefinitely for interrupt or reset Generates a different stack frame

*Motorola assemblers automatically translate ASL mnemonics

5.9 Instruction Format

CPU16 instructions consist of an 8-bit opcode that may be preceded by an 8-bit prebyte and followed by one or more operands.

Opcodes are mapped in four 256-instruction pages. Page 0 opcodes stand alone, but page 1, 2, and 3 opcodes are pointed to by a prebyte code on page 0. The prebytes are \$17 (page 1), \$27 (page 2), and \$37 (page 3).

Operands can be four bits, eight bits or sixteen bits in length. However, because the CPU16 fetches 16-bit instruction words from even-byte boundaries, each instruction must contain an even number of bytes.

Operands are organized as bytes, words, or a combination of bytes and words. Operands of four bits are either zero-extended to eight bits, or packed two to a byte. The largest instructions are six bytes in length. Size, order, and function of operands are evaluated when an instruction is decoded.

A page 0 opcode and an 8-bit operand can be fetched simultaneously. Instructions that use 8-bit indexed, immediate, and relative addressing modes have this form. Code written with these instructions is very compact.

Table 5-3 shows basic CPU16 instruction formats.

Table 5-3. Basic Instruction Formats

8-Bit Opcode with 8-Bit Operand

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

8-Bit Opcode with 4-Bit Index Extensions

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

8-Bit Opcode, Argument(s)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Opcode								Operand							
Operand(s)															
Operand(s)															

8-Bit Opcode with 8-Bit Prebyte, No Argument

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

8-Bit Opcode with 8-Bit Prebyte, Argument(s)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Prebyte								Opcode							
Operand(s)															
Operand(s)															

8-Bit Opcode with 20-Bit Argument

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Opcode								\$0				Extension			
Operand															

5

5.10 Execution Model

This description builds up a conceptual model of the mechanism the CPU16 uses to fetch and execute instructions. The functional divisions in the model do not necessarily correspond to physical subunits of the microprocessor.

As shown in Figure 5-4, there are three functional blocks involved in fetching, decoding, and executing instructions. These are the microsequencer, the instruction pipeline, and the execution unit. These elements function concurrently. All three may be active at any given time.

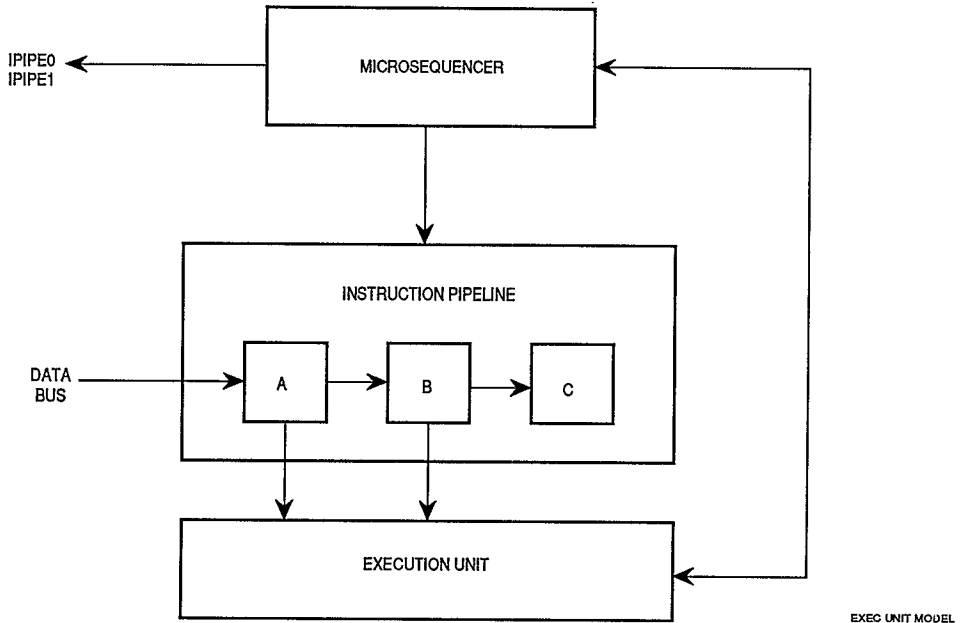


Figure 5-4. Instruction Execution Model

5.10.1 Microsequencer

The microsequencer controls the order in which instructions are fetched, advanced through the pipeline, and executed. It increments the program counter and generates multiplexed external tracking signals IPIPE0 and IPIPE1 from internal signals that control execution sequence.

5.10.2 Instruction Pipeline

The pipeline is a three stage FIFO that holds instructions while they are decoded and executed. Depending upon instruction size, as many as three instructions can be in the pipeline at one time (single-word instructions, one held in stage C, one being executed in stage B, and one latched in stage A).

5.10.3 Execution Unit

The execution unit evaluates opcodes, interfaces with the microsequencer to advance instructions through the pipeline, and performs instruction operations.

5.11 Execution Process

Fetches opcodes are latched into stage A, then advanced to stage B. Opcodes are evaluated in stage B. The execution unit can access operands in either stage A or stage B (stage B accesses are limited to 8-bit operands). When execution is complete, opcodes are moved from stage B to stage C, where they remain until the next instruction is complete.

A prefetch mechanism in the microsequencer reads instruction words from memory and increments the program counter. When instruction execution begins, the program counter points to an address six bytes after the address of the first word of the instruction being executed.

The number of machine cycles necessary to complete an execution sequence varies according to the complexity of the instruction. Refer to the *CPU16 Reference Manual* (CPU16RM/AD) for details.

5.11.1 Changes in Program Flow

When program flow changes, instructions are fetched from a new address. Before execution can begin at the new address, instructions and operands from the previous instruction stream must be removed from the pipeline. If a change in flow is temporary, a return address must be stored, so that execution of the original instruction stream can resume after the change in flow.

At the time an instruction that causes a change in program flow executes, PK : PC point to the address of the first word of the instruction + \$0006. During execution of the instruction, PK : PC is loaded with the address of the first instruction word in the new instruction stream. However, stages A and B still contain words from the old instruction stream. Extra processing steps must be performed before execution from the new instruction stream.

5.12 Instruction Timing

CPU16 instruction execution time has three components:

- Bus cycles required to prefetch the next instruction
- Bus cycles required for operand accesses
- Time required for internal operations

A bus cycle requires a minimum of two system clock periods. If the access time of a memory device is greater than two clock periods, bus cycles are longer. However, all bus cycles must be an integer number of clock periods. CPU16 internal operations are always an integer multiple of two clock periods.

5

Dynamic bus sizing affects bus cycle time. The single-chip integration module manages all accesses. For more information refer to **SECTION 4 SINGLE-CHIP INTEGRATION MODULE**

The CPU16 does not execute more than one instruction at a time. The total time required to execute a particular instruction stream can be calculated by summing the individual execution times of each instruction in the stream.

Total execution time is calculated using the expression

$$(CL_T) = (CL_P) + (CL_O) + (CL_I)$$

Where:

- (CL_T) = Total clock periods per instruction
- (CL_I) = Clock periods used for internal operation
- (CL_P) = Clock periods used for program access
- (CL_O) = Clock periods used for operand access

Refer to the *CPU16 Reference Manual (CPU16RM/AD)* for more information on this topic.

5.13 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.

5.13.1 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 5-4 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. (Refer to **SECTION 4 SINGLE-CHIP INTEGRATION MODULE** for information concerning address space types and the function code outputs.) 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 5-4. 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

5

5.13.2 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 5-5 shows the exception stack frame.

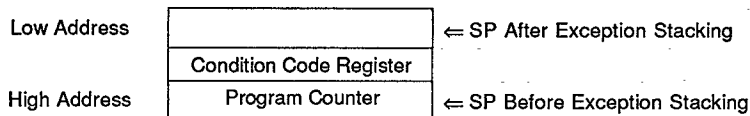


Figure 5-5. Exception Stack Frame Format

5.13.3 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.

5.13.4 Types of Exceptions

Exceptions can be either internally or externally generated. External exceptions, which are defined as asynchronous, include interrupts, bus errors, breakpoints, and resets. 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.

5.13.4.1 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. Refer to **SECTION 4 SINGLE-CHIP INTEGRATION MODULE** for more information concerning asynchronous exceptions.

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.

5.13.4.2 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.

5.13.5 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 is 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.

Refer to **SECTION 4 SINGLE-CHIP INTEGRATION MODULE** for detailed information concerning interrupts and system reset. Refer to the *CPU16 Reference Manual (CPU16RM/AD)* for information concerning processing of specific exceptions.

5.13.6 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.

5.14 Development Support

The CPU16 incorporates powerful tools for tracking program execution and for system debugging. These tools are deterministic opcode tracking, breakpoint exceptions, and background debugging mode. Judicious use of CPU16 capabilities permits in-circuit emulation and system debugging using a bus state analyzer, a simple serial interface, and a terminal.

5.14.1 Deterministic Opcode Tracking

The CPU16 has two multiplexed outputs, IPIPE0 and IPIPE1, that enable external hardware to monitor the instruction pipeline during normal program execution. The signals IPIPE0 and IPIPE1 can be demultiplexed into six pipeline state signals that allow a state analyzer to synchronize with instruction stream activity.

5.14.1.1 IPIPE0/IPIPE1 Multiplexing

Six types of information are required to track pipeline activity. To generate the six state signals, eight pipeline states are encoded and multiplexed into IPIPE0 and IPIPE1. The multiplexed signals have two phases. State signals are active low. Table 5-5 shows the encoding scheme.

Table 5-5. IPIPE0/IPIPE1 Encoding

Phase	IPIPE1 State	IPIPE0 State	State Signal Name
1	0	0	START and FETCH
	0	1	FETCH
	1	0	START
	1	1	NULL
2	0	0	INVALID
	0	1	ADVANCE
	1	0	EXCEPTION
	1	1	NULL

IPIPE0 and IPIPE1 are timed so that a logic analyzer can capture all six pipeline state signals and address, data, or control bus state in any single bus cycle. See **APPENDIX A ELECTRICAL CHARACTERISTICS** for specifications.

State signals can be latched asynchronously on the falling and rising edges of either address strobe (\overline{AS}) or data strobe (\overline{DS}). They can also be latched synchronously using the microcontroller CLKOUT signal. Refer to the *CPU16 Reference Manual* (CPU16RM/AD) for detailed information about state signals and state signal demultiplexing logic.

5.14.1.2 Combining Opcode Tracking with Other Capabilities

Pipeline state signals are useful during normal instruction execution and execution of exception handlers. The signals provide a complete model of the pipeline up to the point a breakpoint is acknowledged.

Breakpoints are acknowledged after an instruction has executed, when it is in pipeline stage C. A breakpoint can initiate either exception processing or background debugging mode. IPIPE0/IPIPE1 are not usable when the CPU16 is in background debugging mode.

5.14.2 Breakpoints

Breakpoints are set by internal assertion of the $\overline{\text{IMB BKPT}}$ signal or by external assertion of the microcontroller $\overline{\text{BKPT}}$ pin. In the MC68HC16Y1, no internal module can assert the $\overline{\text{IMB BKPT}}$ signal. The CPU16 supports breakpoints on any memory access. Acknowledged breakpoints can initiate either exception processing or background debugging mode. After BDM has been enabled, the CPU16 enters BDM when either $\overline{\text{BKPT}}$ input is asserted.

If $\overline{\text{BKPT}}$ assertion is synchronized with an instruction prefetch, the instruction is tagged with the breakpoint when it enters the pipeline, and the breakpoint occurs after the instruction executes.

If $\overline{\text{BKPT}}$ assertion is synchronized with an operand fetch, breakpoint processing occurs at the end of the instruction during which $\overline{\text{BKPT}}$ is latched.

Breakpoints on instructions that are flushed from the pipeline before execution are not acknowledged, but operand breakpoints are always acknowledged. There is no breakpoint acknowledge bus cycle when BDM is entered. Refer to **SECTION 4 SINGLE-CHIP INTEGRATION MODULE** for more information about breakpoints.

5.14.3 Opcode Tracking and Breakpoints

Breakpoints are acknowledged after a tagged instruction has executed, when it is copied from pipeline stage B to stage C. Stage C contains the opcode of the previous instruction when execution of the current instruction begins.

When an instruction is tagged, IPIPE0/IPIPE1 reflect the start of execution and the appropriate number of pipeline advances and operand fetches before the breakpoint is acknowledged. If background debugging mode is enabled, these signals model the pipeline before BDM is entered.

The CPU development support disable bit (CPUD) in the single-chip integration module configuration register (SCIMCR) affects IPIPE0 and IPIPE1 operation. When CPUD is cleared (reset state), the pipeline signals are available unless

the MCU is in background mode. Setting CPUD places the IPIPE0 and IPIPE1 pins in a high-impedance state until a breakpoint occurs.

5.14.4 Background Debugging Mode

Microprocessor debugging programs are generally implemented in external software. CPU16 BDM provides a debugger implemented in CPU microcode.

BDM incorporates a full set of debug options. Registers can be viewed and altered, memory can be read or written, and test features can be invoked.

BDM is an alternate CPU16 operating mode. While the CPU16 is in BDM, normal instruction execution is suspended, and special microcode performs debugging functions under external control. While in BDM, the CPU16 ceases to fetch instructions through the parallel bus and communicates with the development system through a dedicated serial interface.

5

5.14.4.1 Enabling BDM

The CPU16 samples the internal and external $\overline{\text{BKPT}}$ signals during reset to determine whether to enable BDM. If either $\overline{\text{BKPT}}$ input is at logic level zero when sampled, an internal BDM enabled flag is set.

When $\overline{\text{BKPT}}$ is asserted at the rising edge of the $\overline{\text{RESET}}$ signal, BDM operation is enabled. BDM remains enabled until the next system reset. If $\overline{\text{BKPT}}$ is at logic level one on the trailing edge of $\overline{\text{RESET}}$, BDM is disabled. $\overline{\text{BKPT}}$ is relatched on each rising transition of $\overline{\text{RESET}}$. $\overline{\text{BKPT}}$ is synchronized internally, and must be asserted for at least two clock cycles before negation of $\overline{\text{RESET}}$.

5.14.4.2 BDM Sources

When BDM is enabled, external breakpoint hardware, internal IMB module breakpoints, and the BGND instruction can cause the CPU16 to enter BDM. If BDM is not enabled when a breakpoint occurs, a breakpoint exception is processed.

5.14.4.2.1 $\overline{\text{BKPT}}$ Signal

If enabled, BDM is initiated when assertion of $\overline{\text{BKPT}}$ is acknowledged. There is no breakpoint acknowledge bus cycle when BDM is entered. For more information concerning breakpoint acknowledge cycles refer to **SECTION 4 SINGLE-CHIP INTEGRATION MODULE**. For timing specifications, refer to **APPENDIX A ELECTRICAL CHARACTERISTICS**.

5.14.4.2.2 BGND Instruction

If BDM has been enabled, executing BGND causes the CPU16 to suspend normal operation and enter BDM. If BDM has not been correctly enabled, an illegal instruction exception is generated.

5.14.4.3 Entering BDM

When the processor detects a breakpoint or decodes a BGND instruction, it suspends instruction execution and asserts the FREEZE signal. Once FREEZE has been asserted, the CPU enables the serial communication hardware and awaits a command.

5

Assertion of FREEZE causes opcode tracking signals IPIPE0 and IPIPE1 to change definition and become serial communication signals DSO and DSI. FREEZE is asserted at the next instruction boundary after \overline{BKPT} is asserted. IPIPE0 and IPIPE1 change function before an exception signal can be generated. The development system must use FREEZE assertion as an indication that BDM has been entered. When BDM is exited, FREEZE is negated before initiation of normal bus cycles. IPIPE0 and IPIPE1 are valid when normal instruction prefetch begins.

5.14.4.4 BDM Commands

Commands consist of one 16-bit operation word and can include one or more 16-bit extension words. Each incoming word is read as it is assembled by the serial interface. The microcode routine corresponding to a command is executed as soon as the command is complete. Result operands are loaded into the output shift register to be shifted out as the next command is read. This process is repeated for each command until the CPU returns to normal operating mode. The BDM command set is summarized in Table 5-6. Refer to the *CPU16 Reference Manual (CPU16RM/AD)* for a BDM command glossary.

Table 5–6. Command Summary

Command	Mnemonic	Description
Read Registers from Mask	RREGM	Read contents of registers specified by command word register mask
Write Registers from Mask	WREGM	Write to registers specified by command word register mask
Read MAC Registers	RDMAC	Read contents of entire multiply and accumulate register set
Write MAC Registers	WRMAC	Write to entire multiply and accumulate register set
Read PC and SP	RPCSP	Read contents of program counter and stack pointer
Write PC and SP	WPCSP	Write to program counter and stack pointer
Read Data Memory	RDMEM	Read byte from specified 20-bit address in data space
Write Data Memory	WDMEM	Write byte to specified 20-bit address in data space
Read Program Memory	RPMEM	Read word from specified 20-bit address in program space
Write Program Memory	WPMEM	Write word to specified 20-bit address in program space
Execute from Current PK : PC	GO	Instruction pipeline flushed and refilled; instructions executed from current PC – \$0006
Null Operation	NOP	Null command performs no operation

5.14.4.5 Returning from BDM

BDM is terminated when a resume execution (GO) command is received. GO refills the instruction pipeline from address (PK : PC – \$0006). FREEZE is negated before the first prefetch. Upon negation of FREEZE, the serial subsystem is disabled, and the DSO/DSI signals revert to IPIPE0/IPIPE1 functionality.

5.14.4.6 BDM Serial Interface

The serial interface uses a synchronous protocol similar to that of the Motorola serial peripheral interface (SPI). Figure 5–6 is a development system serial logic diagram.

The development system serves as the master of the serial link, and is responsible for the generation of serial interface clock signal DSCLK.

Serial clock frequency range is from DC to one-half the CPU16 clock frequency. If DSCLK is derived from the CPU16 system clock, development system serial logic can be synchronized with the target processor.

The serial interface operates in full-duplex mode. Data transfers occur on the falling edge of DSCLK and are stable by the following rising edge of DSCLK. Data is transmitted MSB first, and is latched on the rising edge of DSCLK.

The serial data word is 17 bits wide, which includes 16 data bits and a status/control bit. Bit 16 indicates status of CPU-generated messages.

Command and data transfers initiated by the development system must clear bit 16. All commands that return a result return 16 bits of data plus one status bit.

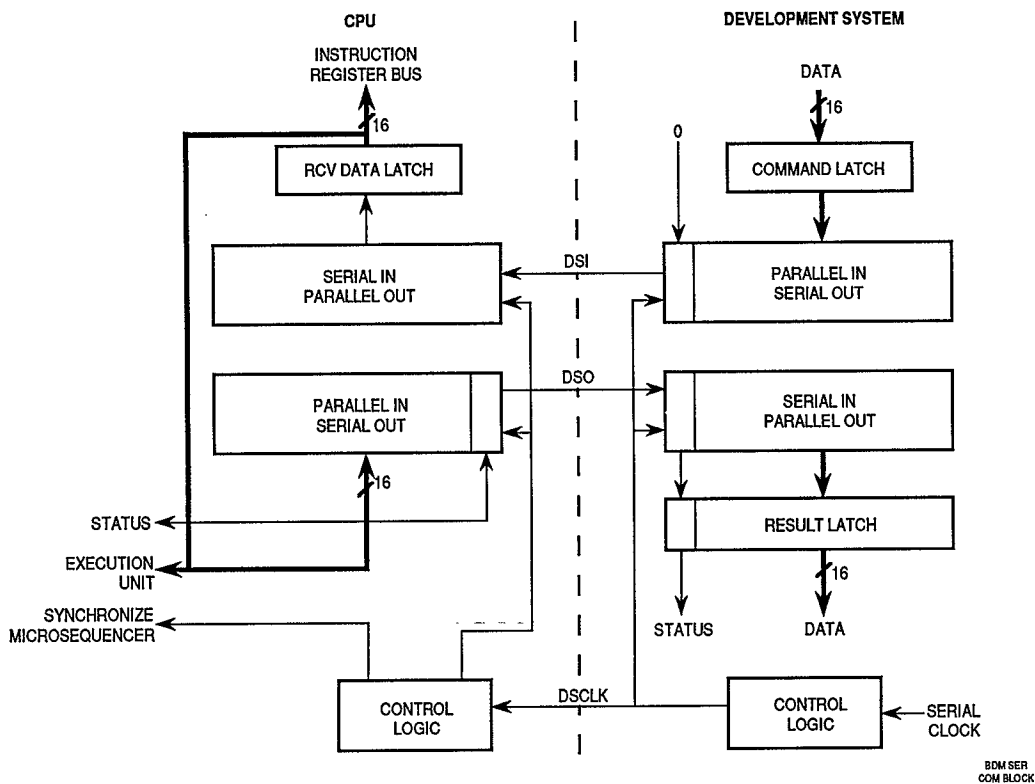
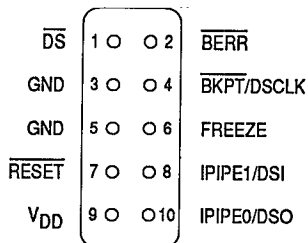


Figure 5-6. BDM Serial I/O Block Diagram

5.15 Recommended BDM Connection

In order to provide for use of development tools when an MCU is installed in a system, Motorola recommends that appropriate signal lines be routed to a male Berg connector or double-row header installed on the circuit board with the MCU, as shown in the following figure.



HC16BERG

5

Figure 5–7. BDM Connector Pinout

5.16 Digital Signal Processing

The CPU16 performs low-frequency digital signal processing (DSP) algorithms in real time. The most common DSP operation in embedded control applications is filtering, but the CPU16 can perform several other useful DSP functions. These include autocorrelation (detecting a periodic signal in the presence of noise), cross-correlation (determining the presence of a defined periodic signal), and closed-loop control routines (selective filtration in a feedback path).

Although derivation of DSP algorithms is often a complex mathematical task, the algorithms themselves typically consist of a series of multiply and accumulate (MAC) operations. The CPU16 contains a dedicated set of registers that perform MAC operations. As a group, these registers are called the MAC unit.

DSP operations generally require a large number of MAC iterations. The CPU16 instruction set includes instructions that perform MAC setup and repetitive MAC operations. Other instructions, such as 32-bit load and store instructions, can also be used in DSP routines.

Many DSP algorithms require extensive data address manipulation. To increase throughput, the CPU16 performs effective address calculations and data prefetches during MAC operations. In addition, the MAC unit provides modulo addressing to implement circular DSP buffers efficiently.

Refer to the *CPU16 Reference Manual* (CPU16RM/AD) for detailed information concerning the MAC unit and execution of DSP instructions. Motorola Application Note AN1213/D, *16-Bit DSP Servo Control with the MC68HC16Z1*, provides a practical control-oriented DSP tutorial.

SECTION 6 ANALOG-TO-DIGITAL CONVERTER

This section is an overview of ADC function. Refer to the *ADC Reference Manual* (ADCRM/AD) for a comprehensive discussion of ADC capabilities. Refer to **APPENDIX A ELECTRICAL CHARACTERISTICS** for ADC timing and electrical specifications. Refer to **APPENDIX D REGISTER SUMMARY** for register address mapping and bit/field definitions.

6.1 General

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.

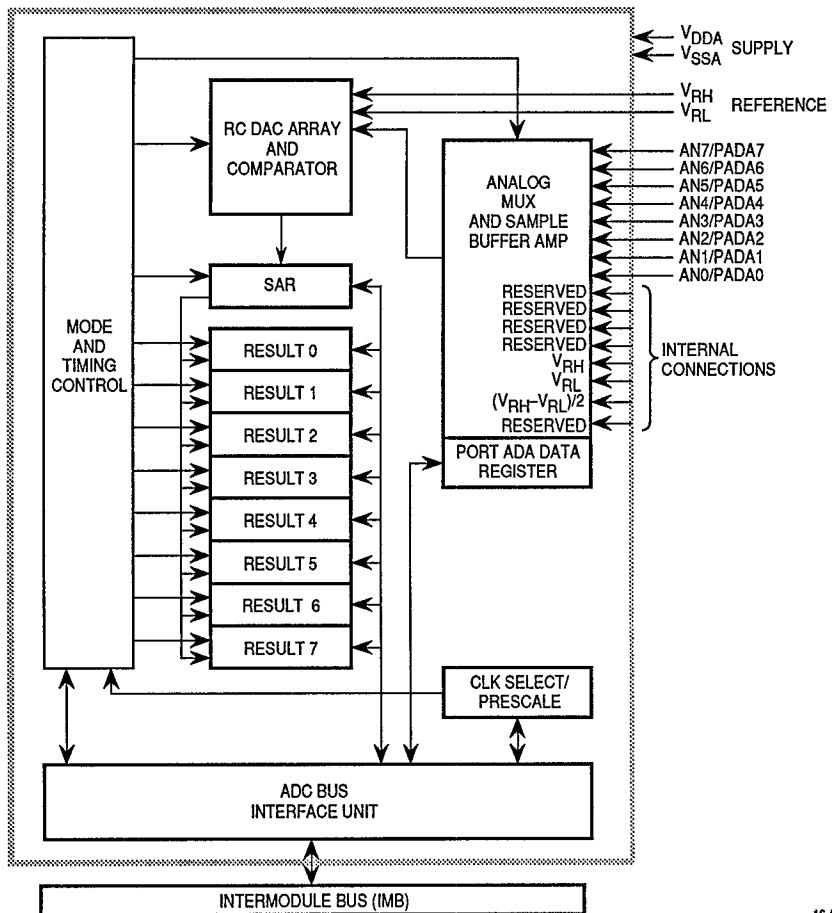
A bus interface unit handles communication between the ADC and other microcontroller modules, and supplies IMB timing signals to the ADC. Special operating modes and test functions are controlled by a module configuration register (ADCMCR) and a factory test register (ADCTST).

ADC module conversion functions can be grouped into three basic subsystems: an analog front end, a digital control section, and result storage. Figure 6–1 is a functional block diagram of the ADC module.

In addition to use as multiplexer inputs, the eight analog inputs can be used as a general-purpose digital input port (Port ADA), provided signals are within logic level specification. A port data register (PORTADA) is used to access input data.

6.2 External Connections

The ADC uses 12 pins on the MCU package. Eight pins are analog inputs (which can also be used as digital inputs), two pins are analog reference connections, and two pins are analog supply connections.



16 ADC BLOCK 2

Figure 6-1. ADC Block Diagram

6.2.1 Analog Input Pins

Each of the eight analog input pins (AN[7:0]) is connected to a multiplexer in the ADC. The multiplexer selects an analog input for conversion to digital data.

Analog input pins can also be read as digital inputs, provided the applied voltage meet V_{IH} and V_{IL} specification. When used as digital inputs, the pins are organized into an 8-bit port (Port ADA), and referred to as ADA[7:0]. There is no data direction register because port pins are used only for input.

6.2.2 Analog Reference Pins

Separate high (V_{RH}) and low (V_{RL}) analog reference voltages are connected to the analog reference pins. The pins permit connection of regulated and filtered supplies that allow the ADC to achieve its highest degree of accuracy.

6.2.3 Analog Supply Pins

Pins V_{DDA} and V_{SSA} supply power to analog circuitry associated with the RC DAC. Other circuitry in the ADC is powered from the digital power bus (pins V_{DDI} and V_{SSI}). Dedicated analog power supplies are necessary to isolate sensitive ADC circuitry from noise on the digital power bus.

6.3 Programmer's Model

The ADC module is mapped into 32 words of address space. Five words are control/status registers, one word is digital port data, and 24 words provide access to the results of AD conversion (eight addresses for each type of converted data). Two words are reserved for expansion. See **APPENDIX D REGISTER SUMMARY** for detailed information concerning the ADC address map and register structure.

The ADC module base address is determined by the value of the MM bit in the single-chip integration module configuration register (SCIMCR). The base address is normally \$FFF700.

Internally, the ADC has both a differential data bus and a buffered IMB data bus. Registers not directly associated with conversion functions, such as the module configuration register, the module test register, and the port data register, reside on the buffered bus, while conversion registers and result registers reside on the differential bus.

Registers that reside on the buffered bus are updated immediately when written. However, writes to ADC control registers abort any conversion in progress.

6.4 ADC Bus Interface Unit

The ADC is designed to act as a slave device on the intermodule bus. The ADC bus interface unit (ABIU) provides IMB bus cycle termination and synchronizes internal ADC signals with IMB signals. The ABIU also manages data bus routing to accommodate the three conversion data formats, and controls the interface to the module differential data bus.

6.5 Special Operating Modes

Low-power stop mode and freeze mode are ADC operating modes associated with assertion of IMB signals by other microcontroller modules or by external sources. These modes are controlled by the values of bits in the ADC module configuration register (ADCMCR).

6

6.5.1 Low-Power Stop Mode

When the STOP bit in ADCMCR is set, the IMB clock signal to the ADC is disabled. This places the module in an idle state, and power consumption is minimized. The ABIU does not shut down and ADC registers are still accessible. If a conversion is in progress when STOP is set, it is aborted.

STOP is set during system reset, and must be cleared before the ADC can be used. Because analog circuit bias currents are turned off during low-power stop, the ADC requires recovery time after STOP is cleared.

Execution of the CPU16 LPSTOP command places the entire modular microcontroller in low-power stop mode. For more information regarding low-power stop operation refer to **SECTION 4 SINGLE-CHIP INTEGRATION MODULE** and **SECTION 5 CENTRAL PROCESSING UNIT**.

6.5.2 Freeze Mode

When the CPU16 in the modular microcontroller enters background debugging mode, the FREEZE signal is asserted. The ADC can respond to internal assertion of FREEZE in one of three different ways. It can ignore FREEZE assertion, finish the current conversion and then freeze, or freeze immediately.

Type of response is determined by the value of the FRZ[1:0] field in the module configuration register (refer to Table 6–1).

Table 6–1.
FRZ Field Selection

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

When the ADC freezes, the ADC clock stops and all sequential activity ceases. Contents of control and status registers remain valid while frozen. When the FREEZE signal is negated, ADC activity resumes.

If the ADC freezes during a conversion, activity resumes with the next step in the conversion sequence. However, capacitors in the analog conversion circuitry discharge while the ADC is frozen; as a result, the conversion will be inaccurate.

Refer to **SECTION 5 CENTRAL PROCESSING UNIT** for more information on background debugging mode.

6.6 Analog Subsystem

The analog subsystem consists of a multiplexer, sample capacitors, a buffer amplifier, an RC DAC array, and a high-gain comparator. Comparator output sequences the successive approximation register (SAR). The interface between the comparator and the SAR is the boundary between ADC analog and digital subsystems.

6.6.1 Multiplexer

The multiplexer selects one of 16 sources for conversion. Eight sources are internal and eight are external. Multiplexer operation is controlled by channel selection field CD:CA in register ADCTL1 (refer to Table 6–2). The multiplexer contains positive and negative stress protection circuitry. This circuitry prevents voltages on other input channels from affecting the current conversion.

**Table 6-2.
Multiplexer Channels**

[CD:CA] Value	Input Source
%0000	AN0
%0001	AN1
%0010	AN2
%0011	AN3
%0100	AN4
%0101	AN5
%0110	AN6
%0111	AN7
%1000	Reserved
%1001	Reserved
%1010	Reserved
%1011	Reserved
%1100	V_{RH}
%1101	V_{RL}
%1110	$(V_{RH} - V_{RL}) / 2$
%1111	Test/Reserved

6.6.2 Sample Capacitor and Buffer Amplifier

All of the input channels share a single buffer amplifier and sample capacitor. After a channel is selected, for the first two ADC clock cycles of a sampling period, multiplexer output is connected to the input of the sample buffer amplifier through the sample capacitor. The sample amplifier buffers the input channel from the relatively large capacitance of the RC DAC array.

During the second two clock cycles of a sampling period, the sample capacitor is disconnected from the multiplexer, and the sample buffer amplifier charges the RC DAC array with the value stored in the sample capacitor.

During the third portion of a sampling period, both sample capacitor and buffer amplifier are bypassed, and multiplexer input charges the DAC array directly. The length of this third portion of a sampling period is determined by the value of the STS field in ADCTL0.

6.6.3 RC DAC Array

The RC DAC array consists of binary-weighted capacitors and a resistor-divider chain. The array performs two functions: it acts as a sample hold circuit during conversion, and it provides each successive digital-to-analog comparison voltage to the comparator. Conversion begins with MSB comparison and ends with LSB comparison. Array switching is controlled by the digital subsystem.

6.6.4 Comparator

The comparator indicates whether each approximation output from the RC DAC array during resolution is higher or lower than the sampled input voltage. Comparator output is fed to the digital control logic, which sets or clears each bit in the successive approximation register in sequence, MSB first.

6.7 Digital Control Subsystem

The digital control subsystem includes control and status registers, clock and prescaler control logic, channel and reference select logic, conversion sequence control logic, and the successive approximation register.

The subsystem controls the multiplexer and the output of the RC array during sample and conversion periods, stores the results of comparison in the successive-approximation register, then transfers results to the result registers.

6.7.1 Control/Status Registers

There are two control registers (ADCTL0, ADCTL1) and one status register (ADSTAT). ADCTL0 controls conversion resolution, sample time, and clock/prescaler value. ADCTL1 controls analog input selection, conversion mode, and initiation of conversion. A write to ADCTL0 aborts the current conversion sequence and halts the ADC. Conversion must be restarted by writing to ADCTL1. A write to ADCTL1 aborts the current conversion sequence and starts a new sequence with parameters altered by the write. ADSTAT shows conversion sequence status, conversion channel status, and conversion completion status.

The following paragraphs are a general discussion of control function. **APPENDIX D REGISTER SUMMARY** shows the ADC address map and discusses register bits and fields.

6.7.2 Clock and Prescaler Control

The ADC clock is derived from the system clock by a programmable prescaler. ADC clock period is determined by the value of the PRS field in ADCTL0.

The prescaler has two stages. The first stage is a 5-bit modulus counter. It divides the system clock by any value from 2 to 32 (PRS[4:0] = %00001 to %11111). The second stage is a divide-by-two circuit. Table 6-3 shows prescaler output values.

**Table 6–3.
Prescaler Output**

PRS[4:0]	ADC Clock
%00000	Reserved
%00001	Sys Clk/4
%00010	Sys Clk/6
...	...
%11101	Sys Clk/60
%11110	Sys Clk/62
%11111	Sys Clk/64

ADC clock speed must be between 0.5 MHz and 2.1 MHz. The reset value of the PRS field is %00011, which divides a nominal 16.78-MHz system clock by eight, yielding maximum ADC clock frequency. There are a minimum of four IMB clock cycles for each ADC clock cycle.

6

6.7.3 Sample Time

The first two portions of all sample periods require four ADC clock cycles. During the third portion of a sample period, the selected channel is connected directly to the RC DAC array for a specified number of clock cycles. The value of the STS field in ADCTL0 determines the number of cycles (refer to Table 6–4). The number of clock cycles required for a sample period is the value specified by STS plus four. Sample time is determined by PRS value.

**Table 6–4.
STS Field Selection**

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

6.7.4 Resolution

ADC resolution can be either eight or ten bits. Resolution is determined by the state of the RES10 bit in ADCTL0. Both 8-bit and 10-bit conversion results are automatically aligned in the result registers.

6.7.5 Conversion Control Logic

Analog-to-digital conversions are performed in sequences. Sequences are initiated by any write to ADCTL1. If a conversion sequence is already in progress, a write to either control register will abort it and reset the SCF and CCF flags in the A/D status register. There are eight conversion modes. Conversion mode is determined by ADCTL1 control bits. Each conversion mode affects the bits in status register ADSTAT differently. Result storage differs from mode to mode.

6.7.5.1 Conversion Parameters

The following conversion parameters are controlled by bits in ADCTL1.

Conversion channel — the value of the channel selection field ([CD:CA]) in ADCTL1 determines which multiplexer inputs are used in a conversion sequence. There are 16 possible inputs. Eight inputs are external pins (AN[7:0]), and eight are internal.

Length of sequence — A conversion sequence consists of either four or eight conversions. The number of conversions in a sequence is determined by the state of the S8CM bit in ADCTL1.

Single or continuous conversion — Conversion can be limited to a single sequence or a sequence can be performed continuously. The state of the SCAN bit in ADCTL1 determines whether single or continuous conversion is performed.

Single or multiple channel conversion — Conversion sequence(s) can be run on a single channel or on a block of four or eight channels. Channel conversion is controlled by the state of the MULT bit in ADCTL1.

6.7.5.2 Conversion Modes

Conversion modes are defined by the state of the SCAN, MULT, and S8CM bits in ADCTL1. Table 6–5 shows mode numbering.

**Table 6-5.
ADC Conversion Modes**

SCAN	MULT	S8CM	Mode
0	0	0	0
0	0	1	1
0	1	0	2
0	1	1	3
1	0	0	4
1	0	1	5
1	1	0	6
1	1	1	7

6

Mode 0 — A single four-conversion sequence is performed on a single input channel specified by the value in CD:CA. Each result is stored in a separate result register (RSLT0 to RSLT3). The appropriate CCF bit in ADSTAT is set as each register is filled. The SCF bit in ADSTAT is set when the conversion sequence is complete.

Mode 1 — A single eight-conversion sequence is performed on a single input channel specified by the value in CD:CA. Each result is stored in a separate result register (RSLT0 to RSLT7). The appropriate CCF bit in ADSTAT is set as each register is filled. The SCF bit in ADSTAT is set when the conversion sequence is complete.

Mode 2 — A single conversion is performed on each of four sequential input channels, starting with the channel specified by the value in CD:CA. Each result is stored in a separate result register (RSLT0 to RSLT3). The appropriate CCF bit in ADSTAT is set as each register is filled. The SCF bit in ADSTAT is set when the last conversion is complete.

Mode 3 — A single conversion is performed on each of eight sequential input channels, starting with the channel specified by the value in CD:CA. Each result is stored in a separate result register (RSLT0 to RSLT7). The appropriate CCF bit in ADSTAT is set as each register is filled. The SCF bit in ADSTAT is set when the last conversion is complete.

Mode 4 — Continuous four-conversion sequences are performed on a single input channel specified by the value in CD:CA. Each result is stored in a separate result register (RSLT0 to RSLT3). Previous results are overwritten when a sequence repeats. The appropriate CCF bit in ADSTAT is set as each register is filled. The SCF bit in ADSTAT is set when the first four-conversion sequence is complete.

Mode 5 — Continuous eight-conversion sequences are performed on a single input channel specified by the value in CD:CA. Each result is stored in a separate result register (RSLT0 to RSLT7). Previous results are overwritten when a sequence repeats. The appropriate CCF bit in ADSTAT is set as each register is filled. The SCF bit in ADSTAT is set when the first eight-conversion sequence is complete.

Mode 6 — Continuous conversions are performed on each of four sequential input channels, starting with the channel specified by the value in CD:CA. Each result is stored in a separate result register (RSLT0 to RSLT3). The appropriate CCF bit in ADSTAT is set as each register is filled. The SCF bit in ADSTAT is set when the first four-conversion sequence is complete.

Mode 7 — Continuous conversions are performed on each of eight sequential input channels, starting with the channel specified by the value in CD:CA. Each result is stored in a separate result register (RSLT0 to RSLT7). The appropriate CCF bit in ADSTAT is set as each register is filled. The SCF bit in ADSTAT is set when the first eight-conversion sequence is complete.

Table 6–6 summarizes ADC operation when MULT is cleared (single channel modes). Table 6–7 is a summary of ADC operation when MULT is set (multi-channel modes). Number of conversions per channel is determined by SCAN. Channel numbers are given in order of conversion.

Table 6-6. 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	0	1	1	0	AN6	RSLT[0:3]
0	0	1	1	1	AN7	RSLT[0:3]
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	RSLT[0:7]
1	0	1	1	1	AN7	RSLT[0:7]
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]

Table 6-7. Multiple-Channel Conversions

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

6.7.6 Conversion Timing

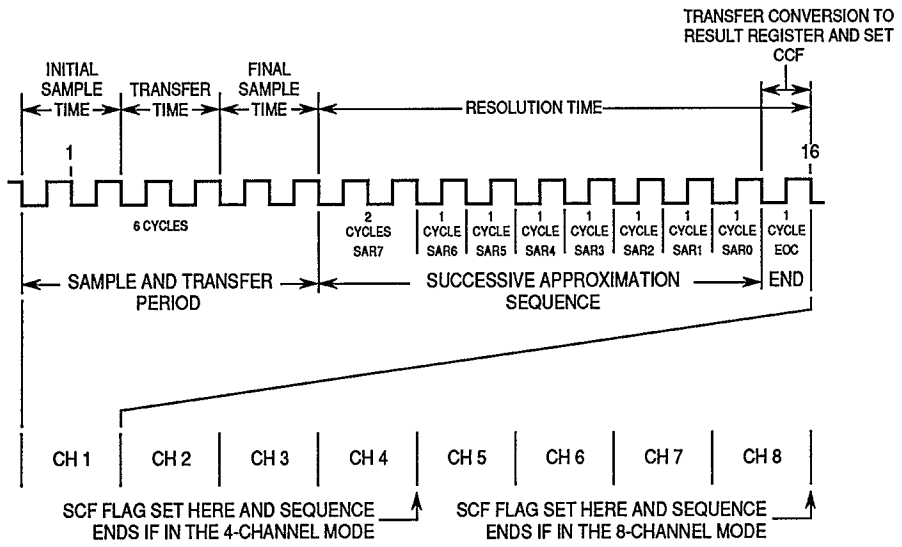
Total conversion time is made up of initial sample time, transfer time, final sample time, and resolution time. Initial sample time is the time during which a selected input channel is connected to the sample buffer amplifier through a sample capacitor. During transfer time, the sample capacitor is disconnected from the multiplexer, and the RC DAC array is driven by the sample buffer amp. During final sampling time, the sample capacitor and amplifier are bypassed, and the multiplexer input charges the RC DAC array directly. During resolution time, the voltage in the RC DAC array is converted to a digital value, and the value is stored in the SAR.

Initial sample time and transfer time are fixed at two ADC clock cycles each. Final sample time can be 2, 4, 8, or 16 ADC clock cycles, depending on the value of the STS field in ADCTL0. Resolution time is ten cycles for 8-bit conversion and twelve cycles for 10-bit conversion.

6

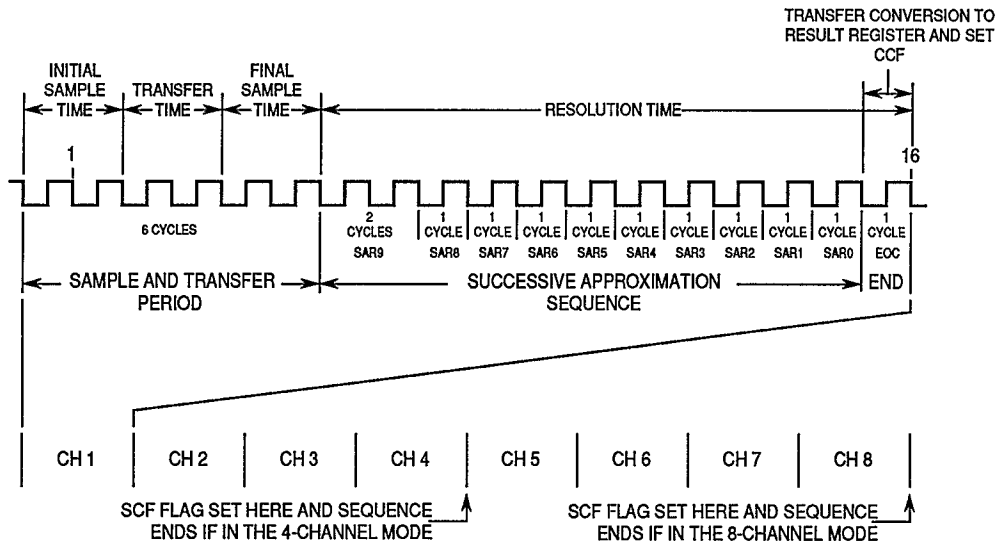
Transfer and resolution require a minimum of 16 ADC clocks (8 μ s with a 2.1-MHz ADC clock) for 8-bit resolution or 18 ADC clocks (9 μ s with a 2.1-MHz ADC clock) for 10-bit resolution. If maximum final sample time (16 ADC clocks) is used, total conversion time is 15 μ s for an 8-bit conversion or 16 μ s for a 10-bit conversion (with a 2.1-MHz ADC clock).

Figures 6-2 and 6-3 illustrate the timing for 8- and 10-bit conversions, respectively. These diagrams assume a final sampling period of two ADC clocks.



ADC 8-BIT CONVERSION TIMING

Figure 6-2. 8-Bit Conversion Timing



ADC 10-BIT CONVERSION TIMING

Figure 6-3. 10-Bit Conversion Timing

6.7.7 Successive Approximation Register

The successive approximation register accumulates the result of each conversion one bit at a time, starting with the most significant bit.

At the start of the resolution period, the MSB of the SAR is set, and all less significant bits are cleared. Depending on the result of the first comparison, the MSB is either left set or cleared. Each successive bit is set or left cleared in descending order until all eight or ten bits have been resolved.

When conversion is complete, the content of the SAR is transferred to the appropriate result register. Refer to **APPENDIX D REGISTER SUMMARY** for register mapping and configuration.

6.7.8 Result Registers

Result registers are used to store data after conversion is complete. The registers can be accessed from the IMB under ABIU control. Each register can be read from three different addresses in the ADC memory map. The format of the result data depends on the address from which it is read.

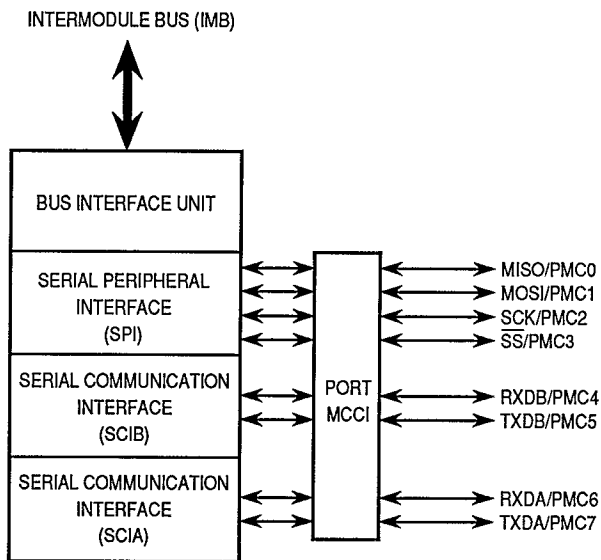
Unsigned Right-Justified Format — Conversion result is unsigned right-justified data. Bits [9:0] are used for 10-bit resolution, bits [7:0] are used for 8-bit conversion (bits [9:8] are zero). Bits [15:10] always return zero when read.

Signed Left-Justified Format — Conversion result is signed left-justified data. Bits [15:6] are used for 10-bit resolution, bits [15:8] are used for 8-bit conversion (bits [7:6] are zero). Although the ADC is unipolar, it is assumed that the zero point is $(V_{RH} - V_{RL}) / 2$ when this format is used. The value read from the register is an offset two's-complement number; for positive input, bit 15 = 0, for negative input, bit 15 = 1. Bits [5:0] always return zero when read.

Unsigned Left-Justified Format — Conversion result is unsigned left-justified data. Bits [15:6] are used for 10-bit resolution, bits [15:8] are used for 8-bit conversion (bits [7:6] are zero). Bits [5:0] always return zero when read.

SECTION 7 MULTICHANNEL COMMUNICATION INTERFACE

The multichannel communication interface (MCCI) is a general-purpose serial communication module. It contains two serial communication interfaces (SCI) and a serial peripheral interface (SPI). This section provides sufficient information for normal use of the MCCI. For detailed information, refer to the *MCCI Reference Manual* (MCCIRM/AD). Figure 7-1 is a block diagram of the MCCI.



MCCI BLOCK

Figure 7-1. MCCI Block Diagram

7.1 General

The two SCI provide standard nonreturn to zero (NRZ) mark/space format. Either SCI operates in full- or half-duplex mode. There are separate transmitter and receiver enable bits and dual data buffers for each interface. A modulus-type baud rate generator provides rates from 64 to 524 kbaud with a 16.78-MHz system clock. Word length of either eight or nine 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. Wakeup functions allow the CPU to run uninterrupted until meaningful data is available.

The SPI provides easy peripheral expansion or interprocessor communication through a full-duplex, synchronous, three-line bus. The SPI is compatible with SPI interfaces found in other Motorola devices, but contains additional features, such as programmable shift direction.

MCCI pins can also be configured for use as a general-purpose I/O port (port MC).

7

7.2 MCCI Registers and Address Map

The MCCI has four types of registers: global registers, pin control registers, SCI registers, and SPI registers. Global registers and pin control registers are discussed in **7.2.1 MCCI Global Registers** and **7.2.2 MCCI Pin Control**. SCI and SPI registers are discussed in **7.3 Serial Communication Interface** and **7.4 Serial Peripheral Interface**. Writes to unimplemented register bits have no meaning or effect, and reads from unimplemented bits always return a logic zero.

The modmap (MM) bit in the single-chip integration module configuration register (SCIMCR) defines the most significant bit (ADDR23) of the IMB address of each register in the MCU. Because the CPU16 drives ADDR[23:20] to the same logic state as ADDR[19:0], MM must equal 1.

Refer to **APPENDIX D REGISTER SUMMARY** for an MCCI address map and register bit/field definition. **SECTION 4 SINGLE-CHIP INTEGRATION MODULE** contains more information concerning module mapping.

7.2.1 MCCI Global Registers

The MCCI configuration register (MMCR) contains parameters for interfacing to the CPU16 and the intermodule bus. The MCCI test register (MTEST) is used during factory test of the MCCI. The SCI and SPI interrupt level register (ILSCI and ILSPI) determine the priority of interrupts requested by the MCCI and the vector used when an interrupt is acknowledged. The MCCI interrupt vector

register (MIVR) contains the interrupt vector for all three MCCI interfaces. ILSCI and MIVR are 8-bit registers located at the same word address. Refer to **APPENDIX D REGISTER SUMMARY** for register bit and field definitions.

7.2.1.1 Low-Power Stop Operation

When the STOP bit in MMCR is set, the system clock input to the MCCI is disabled and the module enters a low-power operating state. MMCR is the only MCCI register guaranteed to be readable while STOP is asserted. STOP can be set by the CPU and by reset.

System software must stop the SPI and SCI before asserting STOP to prevent data corruption and simplify restart. Disable SCI receivers and transmitters after transfers in progress are complete. Halt the SPI by setting the SPE bit in SPCR and then setting STOP after the SPIF flag is set. Refer to **SECTION 4 SINGLE-CHIP INTEGRATION MODULE** for more information about low-power operation.

7.2.1.2 MCCI Interrupts

Both the SCI and the SPI can make interrupt requests on the IMB. Each interface has a separate interrupt request priority register, but a single vector register is used to generate exception vector numbers.

The values of the ILSCIA, ILSCIB and ILSPI fields in the ILSCI and ILSPI registers determine the priorities of interrupt requests. The values in these fields correspond to internal interrupt request signals $\overline{IRQ}[7:1]$. A value of %111 causes $\overline{IRQ7}$ to be asserted when an MCCI interrupt request is made; lower field values cause corresponding lower-numbered interrupt request signals to be asserted. Setting field value to %000 disables interrupts. If ILSPI, ILSCIA, and ILSCIB have the same nonzero value, and simultaneous interrupt requests are made, the priority ranking is SPI, SCIA, SCIB.

When the CPU16 acknowledges an interrupt request, it places the value in the condition code register interrupt priority (IP) mask on the address bus. The MCCI compares IP mask value to request priority to determine whether it should contend for arbitration priority. Arbitration priority is determined by the value of the IARB field in MMCR. Each module that generates interrupts must have a nonzero IARB value. Arbitration is performed by means of serial assertion of IARB field bit values.

When the MCCI wins interrupt arbitration, it responds to the CPU interrupt acknowledge cycle by placing an interrupt vector number on the data bus. The vector number is used to calculate displacement into the CPU16 exception vector table. SCI and SPI vector numbers are generated from the value in the MIVR INTV field. The values of bits INTV[7:2] are the same for SPI SCIA and

SCIB, but the value of INTV[1:0] is supplied by the MCCI when an interrupt request is made. INTV[1:0] = 00 for SCIA interrupt requests; INTV[1:0] = 01 for SCIB interrupt requests; and INTV[1:0] = 10 for SPI interrupt requests.

At reset, INTV is initialized to \$0F, the uninitialized interrupt vector number. To enable interrupt-driven serial communication, a value that represents the six MSB of a user-defined vector number (\$40–\$FF) must be written to MIVR, and interrupt handler routines must be located at the addresses pointed to by the corresponding vectors. CPU writes to INTV[1:0] have no meaning or effect. Reads of INTV[1:0] return a value of one.

Refer to **SECTION 5 CENTRAL PROCESSING UNIT** and **SECTION 4 SINGLE-CHIP INTEGRATION MODULE** for more information about exceptions and interrupts.

7.2.2 MCCI Pin Control

The eight pins used by the MCCI SCI and SPI subsystems have alternate functions as general-purpose I/O pins. Configuring the MCCI submodule includes programming each pin for either general-purpose I/O or its serial interface function. In either function, each pin must also be programmed as input or output.

The MCCI data direction register (MDDR) assigns each MCCI pin as either input or output. The MCCI pin assignment register (MPAR) assigns the MOSI, MISO, and \overline{SS} pins as either SPI pins or general-purpose I/O. (The fourth pin, SCK, is automatically assigned to the SPI whenever the SPI is enabled, i.e., when the SPE bit in the SPI control register is set.) The receiver enable (RE) and transmit enable (TE) bits in the SCI control registers (SCCR0A, SCCR0B) automatically assign the associated pin as an SCI pin when set or general-purpose I/O when cleared. Table 7–1 summarizes how pin function and direction are assigned.

Table 7–1. MCCI Pin Assignments

Pin	Function Assigned By	Direction Assigned By
TXDA/PMC7	TE Bit in SCCR0A	MDDR7
RXDA/PMC6	RE Bit in SCCR0A	MDDR6
TXDB/PMC5	TE Bit in SCCR0B	MDDR5
RXDB/PMC4	RE Bit in SCCR0B	MDDR4
\overline{SS} /PMC3	\overline{SS} Bit in MPAR	MDDR3
SCK/PMC2	SPE Bit in SPCR	MDDR2
MOSI/PMC1	MOSI Bit in MPAR	MDDR1
MISO/PMC0	MISO Bit in MPAR	MDDR0

MCCI port data register PORTMC latches I/O data; MCCI pin state register PORTMCP allows pin state to be read regardless of data direction configuration. Refer to **APPENDIX D REGISTER SUMMARY** for detailed information about MCCI registers.

7.3 Serial Communication Interface

There are two identical independent SCI systems, SCIA and SCIB, in the MCCI. Each is a full-duplex universal asynchronous receiver transmitter (UART). Each SCI system is fully compatible with the SCI systems found on other Motorola devices, such as the M68HC11 and M68HC05 families. The following discussions apply to both SCIA and SCIB. Differences in register addresses and pin names are noted.

The SCI uses a standard nonreturn to zero (NRZ) transmission format. An on-chip baud-rate generator derives standard baud-rate frequencies from the MCU oscillator. Both the transmitter and the receiver are double buffered, so that back-to-back characters can be handled easily even if the CPU is delayed in responding to the completion of an individual character. The SCI transmitter and receiver are functionally independent but use the same data format and baud rate.

SCI operation can be polled by means of status flags in the SCI status register (SCSR), or interrupt-driven operation can be employed by means of the interrupt-enable bits in SCI control register one (SCCR1A and SCCR1B).

7.3.1 SCI Pins

Two unidirectional transmit data pins, TXDA and TXDB, and two unidirectional receive data pins, RXDA and RXDB, are associated with each SCI. Each pin can be used by the associated SCI or for general-purpose I/O.

Table 7–2 shows SCI pins and their functions.

Table 7–2. SCI Pin Function

Pin Names	Mnemonics	Mode	Function
Receive Data A and B	RXDA, RXDB	Receiver Disabled Receiver Enabled	General-Purpose I/O Serial Data Input to SCI
Transmit Data A and B	TXDA, TXDB	Transmitter Disabled Transmitter Enabled	General-Purpose I/O Serial Data Output from SCI

7.3.2 SCI Registers

The SCI programming model includes the MCCI global and pin control registers, and eight SCI registers. Each of the two SCI units contains two SCI control registers (SCCR0A, SCCR0B, SCCR1A and SCCR1B), one status register (SCSRA and SCSRB), and one data register (SCDRA and SCDRB).

All registers can be read or written at any time by the CPU. Rewriting the same value to any SCI register does not disrupt operation; however, writing a different value into an SCI register when the SCI is running can disrupt operation. To change register values, the receiver and transmitter should be disabled with the transmitter allowed to finish first. The status flags in register SCSR can be cleared at any time. Refer to **APPENDIX D REGISTER SUMMARY** for detailed information about SCI registers.

7.3.3 Serial Formats

Data can be transmitted and received in a number of formats. The following terms concerning data format are used in this section:

Bit Time — The time required to transmit or receive one bit of data; one cycle of the baud frequency.

Start Bit — One bit time of logic zero that indicates the beginning of a data frame. A start bit must begin with a one-to-zero transition.

Stop Bit — One bit time of logic one that indicates the end of a data frame.

Frame — A complete unit of serial information. The SCI can use 10- or 11-bit frames.

Data Frame — A start bit, a specified number of data bits (one of which may be used as an address mark or parity bit or an extra stop bit) and a final stop bit.

Idle Frame — A frame that consists of all ones. An idle frame has no start bit.

Break Frame — A frame that consists of all zeros. A break frame has no stop bits.

MSB — Most significant bit; in a data frame, the bit preceding the hardware-generated stop bit. Depending on the hardware and software configuration, the MSB may represent a data bit, parity bit, address mark bit, or extra stop bit.

The SCI transmitter automatically provides a start bit as the first bit of each frame and a stop bit as the final bit. In addition, it generates a parity bit

preceding the stop bit, if parity is enabled. The SCI receiver automatically strips the start and stop bits. If parity is enabled, the receiver strips the parity bit as well. Receiving and transmitting devices must use the same data frame format.

The SCI provides hardware support for both 10- and 11-bit frames. The most common 10-bit data frame format for NRZ serial interface is one start bit, eight data bits (LSB first), and one stop bit. The most common 11-bit data frame contains one start bit, eight data bits, a parity or address mark bit, and one stop bit.

The serial mode (M) bit in SCCR1 specifies the number of bits per frame. The PE bit determines whether the MSB is a parity bit. (For transmitted data, a parity bit is generated; for received data, the parity bit is checked.) When parity is disabled, software can use the MSB as an address mark bit or a second stop bit. When parity is enabled, the address mark or second stop bit options are not available.

Table 7-3 lists the possible frame formats. Notice that hardware provides four frame formats, according to the values of M and PE. Within these four formats, software can provide additional variations.

Table 7-3. Data Frame Formats

M	PE	Frame Format
0	0	1 start bit, 8 data bits, 1 stop bit 1 start bit, 7 data bits, 1 address mark bit, 1 stop bit 1 start bit, 7 data bits, 2 stop bits
0	1	1 start bit, 7 data bits, 1 parity bit, 1 stop bit
1	0	1 start bit, 8 data bits, 1 address mark bit, 1 stop bit 1 start bit, 8 data bits, 2 stop bits
1	1	1 start bit, 8 data bits, 1 parity bit, 1 stop bit

7.3.4 Parity Checking

The parity type (PT) bit in SCCR1 selects even (PT = 0) or odd (PT = 1) parity. PT affects received and transmitted data. The parity enable (PE) bit in SCCR1 determines whether parity checking is enabled (PE = 1) or disabled (PE = 0). When PE is set, the MSB in a frame is used for the parity function. For transmitted data, a parity bit is generated; for received data, the parity bit is checked. When parity checking is enabled, the parity flag (PF) in the SCI status register (SCSR) is set if a parity error is detected.

7.3.5 Baud Clock

The SCI baud clock is programmed by writing a 13-bit value to the baud rate (SCBR) field in SCI control register zero (SCCR0). The baud clock is derived from the MCU system clock by a modulus counter. Writing a value of zero to SCBR disables the baud rate generator.

Baud clock rate is calculated as follows:

$$\text{SCI Baud Clock Rate} = \text{System Clock}/(32 \text{ SCBR})$$

where SCBR is in the range {1, 2, 3, ... 8191}.

The SCI receiver operates asynchronously. To synchronize with an incoming data stream, the SCI baud clock generator produces a receive time (RT) sampling clock with a frequency 16 times that of the SCI baud clock. 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.

7

7.3.6 SCI Transmitter

The SCI transmitter consists of a transmit serial shifter and a parallel transmit data register (TDR) located in the SCDR. The transmitter is double buffered: one byte can be loaded into the TDR while another character is being shifted out from the serial shifter to the TXD pin.

Transmitter logic adds a start bit (logic zero) and a stop bit (logic one) to the data characters presented by the CPU for transmission. The transmitter can be configured to send characters with eight ($M = 0$) or nine ($M = 1$) data bits. When the TDR is able to accept a new data character, the TDRE status flag is set, and an interrupt can optionally be generated. Another status flag (TC) and optional interrupt are produced when the transmitter has finished sending everything in its queue. In addition to data characters, the transmitter is capable of sending idle-line characters and break characters, which are useful in multidrop SCI networks. Two characters can normally be in the transmit queue, but three characters can be queued when at least one of them is an idle-line or break character.

7.3.6.1 Transmitter Status Flags and Interrupts

Two status flags are associated with the SCI transmitter. These flags are read (polled) by software to tell when the corresponding condition exists. Alternately, an interrupt-enable bit can be set to enable each of these status conditions to generate interrupt requests when the corresponding condition is present. Status flags are automatically set by hardware logic conditions, but software clears these flags, providing an interlock mechanism for logic to know when

software has noticed the status indication. The software clearing sequence for these flags consists of steps that are normally performed in response to the flags.

When the transmitter is first enabled, the TDRE and TC flags are normally already set. To prevent an immediate interrupt from occurring from these sources, read the SCSR and then write to the SCDR before enabling the transmitter. This procedure clears the TDRE and TC flags.

The TDRE flag indicates that there is room in the transmit queue to store another data character in the TDR. The TIE bit is the interrupt-enable bit for TDRE. When TIE equals zero, TDRE must be polled; when TIE equals one, an interrupt is requested whenever TDRE is set.

The TC flag indicates that the transmitter has finished transmitting everything in its queue, including any idle preamble or break character that has been queued. The TCIE bit is the interrupt-enable bit for the TC. When TCIE equals zero, TC must be polled; when TCIE equals one, an interrupt is requested whenever TC is set.

One interrupt vector is associated with each SCI subsystem; therefore, the interrupt service routine must begin by reading the SCSR to determine which interrupt or interrupts caused the service routine to be called. Possible interrupt sources include the two transmitter sources previously discussed and two receiver-related sources.

7.3.7 SCI Receiver

Each SCI receiver contains a receive serial shifter and a parallel receive data register (RDR) located in the SCI data register (SCDR). The receiver is double buffered: while one character is shifted into the receive serial shift register, another character can be held in the RDR. This double-buffered arrangement gives software some time to notice a received character and read it before the next serial character is finished. Without double buffering, the transmitting device would be required to insert delays between transmitted characters to avoid a receiver overrun.

The receiver enable (RE) bit in SCCR1 enables the receiver. The M bit in SCCR1 determines frame size (10 or 11 bits). After a stop bit is detected, the received data is transferred from the shifter to the SCDR, and the receive data register full (RDRF) status flag is set. When a character is ready to be transferred to the receive buffer but the previous character has not yet been read, an overrun condition results. When this occurs, data is not transferred and the overrun (OR) status flag is set to indicate the error.

The wakeup block uses the WAKE control bit in SCCR1 to decide whether to use address mark or the all ones signal (idle line) to wake up the receiver. When the selected condition is detected, the wakeup logic clears the receiver wakeup (RWU) bit in SCCR1, waking up the receiver.

The SCSR contains two receiver-related flags that can be polled by software or optionally cause an SCI interrupt request. The receiver interrupt enable (RIE) control bit enables the RDRF status flag to generate hardware interrupt requests. The idle line interrupt enable (ILIE) control bit allows the IDLE status flag to generate SCI interrupt requests.

The input of the receive serial shifter is connected to the sampling logic of the receiver bit processor. The receiver bit processor logic drives a state machine that determines the logic level for each bit time. This state machine controls when the bit processor logic is to sample the RXD pin and also controls when data is to be passed to the receive serial shifter. Once data is shifted into the receive serial shifter it is moved synchronously with the MCU system clock.

7.3.7.1 Receiver Status Flags and Interrupts

Six status flags are associated with the SCI receiver. RDRF is set when a character has been received and transferred into the parallel RDR. The OR flag is set if RDRF is already set when a new character is ready to be transferred into the parallel RDR. The PF, NF, and FE flags alert the user to parity, noise, and framing error conditions, respectively. The IDLE flag is set when the MCU detects an idle line condition (a frame of all ones).

All status flags associated with a serially received frame are set simultaneously. When a completed frame is received, either the RDRF or OR flag is always set. The receiver status flags NF, FE, and PF are set simultaneously with RDRF as appropriate. (Although error conditions are detected as bits are received, these flags are not set until data is transferred from the serial shifter to RDR.)

Status flags can be polled at any time by software. RDRF and IDLE can optionally generate an automatic interrupt request. Because NF, FE, and PF are set at the same time as RDRF, they do not have separate interrupt enables.

To clear all receiver status flags, read the SCSR and then read the SCDR.

7.3.7.2 Receiver Wakeup

The receiver wakeup function allows a transmitting device to direct a transmission to a single receiver or to a group of receivers by sending an address frame at the start of a message. Receivers not addressed become dormant for the remainder of the message, eliminating any further software overhead to service the remaining characters of the message.

Software places the receiver in wakeup mode by setting the receiver wakeup (RWU) bit in SCCR1. While RWU is set, receiver status flags cannot be set. For this reason idle-line detection cannot be used with receiver wakeup. Hardware clears RWU using one of two methods: idle-line wakeup or address-mark wakeup. (It is possible to clear RWU with software, but this is not the normal procedure).

The WAKE bit in SCCR1 determines which type of wakeup is used. When WAKE = 0, idle-line wakeup is selected. When WAKE = 1, address-mark wakeup is selected. Both types require a software-based device addressing and recognition scheme.

Idle-line wakeup allows a receiver to sleep until an idle line is detected. When an idle-line is detected, the receiver clears RWU and wakes up. The receiver waits for the first frame of the next transmission. The byte is received normally and transferred to the RDR, and the RDRF flag is set. If software does not recognize the address, it can set RWU and put the receiver back to sleep until the next idle line. This method of receiver wakeup requires a minimum of one idle-line frame time between messages and allows no idle time between frames in a message. The ILT bit determines whether short or long idle-line detection is invoked.

Address-mark wakeup uses a special frame format. The first frame of each transmission must be an address frame, indicated by a logic level one in the MSB. For all other characters, the MSB must equal zero.

When the MSB of a frame is set, the receiver clears RWU and wakes up. The byte is received normally and transferred to the RDR, and the RDRF flag is set. If software does not recognize the address, it can set RWU and put the receiver back to sleep. Address-mark wakeup allows idle time between frames and eliminates idle time between transmissions. However, there is a loss of efficiency due to an additional bit time per frame.

7.3.7.3 Frame Detection and Synchronization

After RE is set, bit processor logic begins to sample the signal on the RXD pin. The sampling process identifies a valid start bit and synchronizes the receiver with the incoming frame. A receive time (RT) clock is used to control sampling and synchronization. Data is shifted into the receive serial shifter according to the most recent synchronization of the RT clock with the incoming data stream. From this point on, the data movement is synchronized with the MCU system clock. This process determines how well the SCI receiver can handle noise. Understanding the process can be useful in determining the amount of baud-rate frequency mismatch that can be tolerated.

RT clock rate is 16 times the baud clock rate. Each bit time of a received frame is divided into 16 sample periods designated RT1–RT16. Designations are assigned relative to the time a start bit is detected. The receiver active flag (RAF) in SCSR is set when a valid start bit is identified.

Synchronicity is maintained throughout the frame. Sampling continues and the receiver resynchronizes on each one-to-zero transition in the frame. Bit time normally begins at RT1 and ends at RT16. After the frame ends, the receiver begins the process of identifying the next start bit.

Synchronization at the beginning of each incoming frame eliminates cumulative timing errors due to small differences between the baud rates of the receiver and the transmitter. Synchronization on each one-to-zero transition in the frame increases tolerance to small frequency variations in the received data stream.

7.4. Serial Peripheral Interface

The SPI submodule communicates with external devices through a synchronous serial bus. The SPI is fully compatible with SPI systems found on other Motorola products, but has enhanced capabilities. The SPI can perform full-duplex three-wire transfers, or half-duplex two-wire transfers.

7.4.1 SPI Pins

The SPI uses four bidirectional pins. These pins can be configured for general-purpose I/O when not needed for SPI application. Table 7–4 shows SPI pin functions. When used for SPI functions, the pins should have pull-up resistors.

Table 7–4. SPI Pin Function

Pin Names	Mode	Function
Master In Slave Out (MISO)	Master Slave	Provides serial data input to the SPI Provides serial data output from the SPI
Master Out Slave In (MOSI)	Master Slave	Provides serial output from the SPI Provides serial input to the SPI
Serial Clock (SCK)	Master Slave	Provides clock output from SPI Provides clock input to SPI
Slave Select (SS)	Master Slave	Causes mode fault Initiates serial transfer

7.4.2 SPI Registers

The programmer's model for the SPI consists of the MCCI global and pin control registers, the SPI control register (SPCR), the SPI status register (SPSR), and the SPI data register (SPDR). All SPI registers can be read and written by the CPU. SPCR must be initialized before the SPI is enabled to ensure defined

operation. The SPI is enabled by setting the SPE bit in SPCR. Refer to **APPENDIX D REGISTER SUMMARY** for detailed information about SPI registers.

7.4.3 SPI Operation

The SPI operates in either master or slave mode. Master mode is used when the SPI originates data transfers. Slave mode is used when an external device initiates serial transfers to the SPI. Switching between the modes is controlled by MSTR in SPCR. Before either mode can be entered, appropriate MCCI and SPI registers must be properly initialized.

In master mode, transmission parameters are set by writing to SPCR, the SPI is enabled by setting SPE, then operation is initiated by writing data to SPDR. In slave mode, operation proceeds in response to \overline{SS} signal assertion by an external bus master. Slave operation is similar to that of master mode.

Normally, the SPI bus performs synchronous bidirectional transfers. The serial clock on the SPI bus master supplies the clock signal (SCK) to time the transfer of data. Four possible combinations of clock phase and polarity can be specified by the CPHA and CPOL bits in SPCR. Data can be transferred either LSB or MSB first, depending on the value of the LSBF bit in SPCR. The number of bits transferred per command defaults to eight, but can be set to 16 bits by setting the SIZE bit in SPCR.

When the SPI finishes a transmission, it sets the SPIF flag, clears SPE and stops. If the SPIE bit in SPCR is set, an interrupt request is generated when SPIF is set.

Although the SPI supports multimaster operation, no special arbitration mechanism is provided. A mode fault flag (MODF) indicates a request for SPI master arbitration. System software must provide arbitration.

Typically, SPI bus outputs are not open-drain unless multiple SPI masters are in the system. If needed, the WOMP bit in SPCR can be set to provide wired-OR open-drain outputs. An external pull-up resistor should be used on each output line. WOMP affects all SPI pins regardless of whether they are assigned to the SPI or used as general-purpose I/O.

7.4.3.1 Write Collision

A write collision occurs when an attempt is made to write the SPDR while a transfer is in progress. Since the SPDR is not double buffered in the transmit direction, a write to SPDR would cause data to be written directly into the SPI shift register, corrupting any transfer in progress.

A write collision is normally a slave error because a slave has no control over when a master initiates a transfer. Since a master is in control of the transfer, software can avoid a write collision error generated by the master. The SPI logic can, however, detect a write collision in a master as well as in a slave.

What constitutes a transfer in progress depends on the SPI configuration. For a master, a transfer starts when data is written to the SPDR and ends when SPIF is set. For a slave, the beginning and ending points of a transfer depend on the value of CPHA. When CPHA = 0, the transfer begins when \overline{SS} is asserted and ends when it is negated. When CPHA = 1, a transfer begins at the edge of the first SCK cycle and ends when SPIF is set.

When a write collision occurs, the WCOL bit in the SPSR is set, the data that caused the error is not written to the shifter, and the transfer continues undisturbed. No SPI interrupt is generated. To clear WCOL, read the SPSR while WCOL is set, and then either read the SPDR (either before or after SPIF is set) or write the SPDR after SPIF is set. (Writing the SPDR before SPIF is set results in a second write collision error.) This process clears SPIF as well as WCOL.

7

7.4.3.2 Mode Fault

When the SPI system is configured as a master and the \overline{SS} input line is asserted, a mode fault error occurs, and the MODF bit in the SPSR is set. Only an SPI master can experience a mode fault error. To avoid latchup caused by contention between two pin drivers, the MCU does the following when it detects a mode fault:

1. Forces the MSTR control bit to zero to reconfigure the SPI as a slave.
2. Forces the SPE control bit to zero to disable the SPI system.
3. Sets the MODF status flag and generates an SPI interrupt if SPIE = 1.
4. Clears the appropriate bits in the DDRMC to configure all SPI pins except the \overline{SS} pin as inputs.

After correcting the problems that led to the mode fault, clear MODF by reading the SPSR while MODF is set and then writing to the SPCR. Control bits SPE and MSTR can be restored to their original set state during this clearing sequence or after the MODF bit has been cleared. Hardware does not allow the user to set the SPE and MSTR bits while MODF is a logic one except during the proper clearing sequence.

SECTION 8 GENERAL-PURPOSE TIMER

This section is an overview of GPT function. Refer to the *GPT Reference Manual* (GPTRM/AD) for complete information about the GPT module.

8.1 General

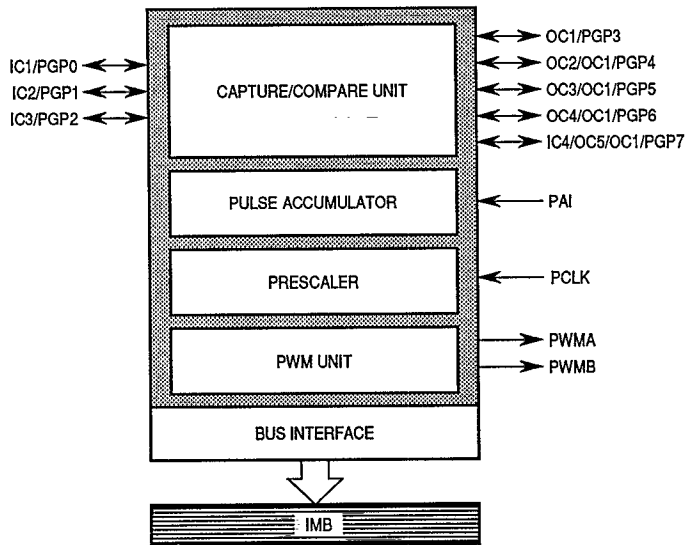
The 11-channel general-purpose timer (GPT) is used in systems where a moderate level of CPU control is required. The GPT consists of a capture/compare unit, a pulse accumulator, and two pulse-width modulators. A bus interface unit connects the GPT to the intermodule bus (IMB). Figure 8–1 is a block diagram of the GPT.

The capture/compare unit features three input capture channels, four output compare channels, and one channel that can be selected as an input capture or 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 (port GP). PWM pins are outputs only. PAI and PCLK pins are inputs only.



GPT BLOCK

Figure 8-1. GPT Block Diagram

8.2 GPT Registers and Address Map

The GPT programming model consists of a configuration register (GPTMCR), parallel I/O registers (DDRGP, PORTGP), capture/compare registers (TCNT, TCTL1, TCTL2, TIC[1:3], TOC[1:4], TI4/O5, CFORC), pulse accumulator registers (PACNT, PACTL), pulse-width modulation registers (PWMA, PWMB, PWMC, PWMCNT, PWMBUFA, PWMBUFB), status registers (TFLG1, TFLG2) and interrupt control registers (TMSK1, TMSK2). Functions of the module configuration register are discussed in **8.3 Special Modes of Operation** and **8.4 Polled and Interrupt-Driven Operation**. Other register functions are discussed in the appropriate sections.

All registers can be accessed using byte or word operations. Certain capture/compare registers and pulse-width modulation registers must be accessed by word operations to ensure coherency. If byte accesses are used to read a register such as the timer counter register (TCNT), there is a possibility that data in the byte not being accessed will change while the other byte is read. Both bytes must be accessed at the same time.

The modmap (MM) bit in the single-chip integration module configuration register (SCIMCR) defines the most significant bit (ADDR23) of the IMB address for each register in the MCU. Because the CPU16 drives ADDR[23:20] to the same logic state as ADDR[19:0], MM must equal one.

Refer to **APPENDIX D REGISTER SUMMARY** for a GPT address map and register bit/field descriptions. **SECTION 4 SINGLE-CHIP INTEGRATION MODULE** contains more information about how the state of MM affects the system.

8.3 Special Modes of Operation

The GPT module configuration register (GPTMCR) is used to control special GPT operating modes. These include low-power stop mode, freeze mode, single-step mode, and test mode. Normal GPT operation can be polled or interrupt-driven. Refer to **8.4 Polled and Interrupt-Driven Operation** for more information.

8.3.1 Low-Power Stop Mode

Low-power stop operation is initiated by setting the STOP bit in GPTMCR. In stop mode the system clock to the module is turned off. The clock remains off until STOP is negated or a reset occurs. All counters and prescalers within the timer stop counting while the STOP bit is set. Only the module configuration register (GPTMCR) and the interrupt configuration register (ICR) should be accessed while in the stop mode. Accesses to other GPT registers cause unpredictable behavior. Low-power stop can also be used to disable module operation during debugging.

8.3.2 Freeze Mode

The freeze (FRZ[1:0]) bits in GPTMCR are used to determine what action is taken by the GPT when the IMB FREEZE signal is asserted. FREEZE is asserted when the CPU enters background debugging mode. At the present time, FRZ1 has no effect; setting FRZ0 causes the GPT to enter freeze mode. Refer to **SECTION 5 CENTRAL PROCESSING UNIT** for more information on background debugging mode.

Freeze mode freezes the current state of the timer. The prescaler and the pulse accumulator do not increment and changes to the pins are ignored (input pin synchronizers are not clocked). All of the other timer functions that are controlled by the CPU will operate normally; for example, registers can be written to change pin directions, force output compares, and read or write I/O pins.

While the FREEZE signal is asserted, the CPU has write access to registers and bits that are normally read-only, or write-once. The write-once bits can be written to as often as needed. The prescaler and the pulse accumulator remain stopped and the input pins are ignored until the FREEZE signal is negated (the CPU is no longer in BDM), the FRZ0 bit is cleared, or the MCU is reset.

Activities that are in progress prior to FREEZE assertion are completed. For example, if an input edge on an input capture pin is detected just as the FREEZE signal is asserted, the capture occurs and the corresponding interrupt flag is set.

8.3.3 Single-Step Mode

Two bits in GPTMCR support GPT debugging without using BDM. When the STOPP bit is asserted, the prescaler and the pulse accumulator stop counting and changes at input pins are ignored. Reads of the GPT pins return the state of the pin when STOPP was set. After STOPP is set, the INCP bit can be set to increment the prescaler and clock the input synchronizers once. The INCP bit is self-negating after the prescaler is incremented. INCP can be set repeatedly. The INCP bit has no effect when the STOPP bit is not set.

8.3.4 Test Mode

Test mode is used during Motorola factory testing. The GPT has no dedicated test-mode control register; all GPT testing is done under control of the single-chip integration module.

8.4 Polled and Interrupt-Driven Operation

Normal GPT function can be polled or interrupt-driven. All GPT functions have an associated status flag and an associated interrupt. The timer interrupt flag registers (TFLG1 and TFLG2) contain status flags used for polled and interrupt-driven operation. The timer mask registers (TMSK1 and TMSK2) contain interrupt control bits. Control routines can monitor GPT operation by polling the status registers. When an event occurs, the control routine transfers control to a service routine that handles that event. If interrupts are enabled for an event, the GPT requests interrupt service when the event occurs. Using interrupts does not require continuously polling the status flags to see if an event has taken place. However, status flags must be cleared after an interrupt is serviced, in order to disable the interrupt request.

8.4.1 Polled Operation

When an event occurs in the GPT, that event sets a status flag in TFLG1 or TFLG2. The GPT sets the flags; they cannot be set by the CPU. TFLG1 and TFLG2 are 8-bit registers that can be accessed individually or as one 16-bit register. The registers are initialized to zero at reset. Table 8–1 shows status flag assignment.

Table 8–1. GPT Status Flags

Flag Mnemonic	Register Assignment	Source
IC1F	TFLG1	Input Capture 1
IC2F	TFLG1	Input Capture 2
IC3F	TFLG1	Input Capture 3
OC1F	TFLG1	Output Compare 1
OC2F	TFLG1	Output Compare 2
OC3F	TFLG1	Output Compare 3
OC4F	TFLG1	Output Compare 4
I4/O5F	TFLG1	Input Capture 4/Output Compare 5
TOF	TFLG2	Timer Overflow
PAOVF	TFLG2	Pulse Accumulator Overflow
PAIF	TFLG2	Pulse Accumulator Input

For each bit in TFLG1 and TFLG2 there is a corresponding bit in TMSK1 and TMSK2 in the same bit position. If a mask bit is set and an associated event occurs, a hardware interrupt request is generated.

To re-enable a status flag after an event occurs, the status flags must be cleared. Status registers are cleared in a particular sequence. The register must first be read for set flags, then zeros must be written to the flags that are to be cleared. If a new event occurs between the time that the register is read and the time that it is written, the associated flag is not cleared.

8.4.2 GPT Interrupts

The GPT has 11 internal sources that can cause it to request interrupt service (refer to Table 8–2). Setting bits in TMSK1 and TMSK2 enables specific interrupt sources. TMSK1 and TMSK2 are 8-bit registers that can be addressed individually or as one 16-bit register. The registers are initialized to zero at reset. For each bit in TMSK1 and TMSK2 there is a corresponding bit in TFLG1 and TFLG2 in the same bit position. TMSK2 also controls the operation of the timer prescaler. Refer to **8.7 Prescaler** for more information.

The value of the interrupt level (IRL) field in the interrupt control register (ICR) determines the priority of GPT interrupt requests. IRL values correspond to MCU interrupt request signals $\overline{\text{IRQ}}[7:1]$. $\overline{\text{IRQ}}7$ is the highest priority interrupt request signal; $\overline{\text{IRQ}}1$ is the lowest-priority signal. A value of %111 causes $\overline{\text{IRQ}}7$ to be asserted when a GPT interrupt request is made; lower field values cause corresponding lower-priority interrupt request signals to be asserted. Setting field value to %000 disables interrupts.

Table 8–2. 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

The CPU16 recognizes only interrupt request signals of a priority greater than the condition code register interrupt priority (IP) mask value. When the CPU acknowledges an interrupt request, the priority of the acknowledged request is written to the IP mask and driven out on the IMB address lines.

When the IP mask value driven out on the address lines is the same as the IRL value, the GPT contends for arbitration priority. GPT arbitration priority is determined by the value of the IARB field in GPTMCR. Each MCU module that can make interrupt requests must be assigned a nonzero IARB value in order to implement an arbitration scheme. Arbitration is performed by means of serial assertion of IARB field bit values.

When the GPT wins interrupt arbitration, it responds to the CPU interrupt acknowledge cycle by placing an interrupt vector number on the data bus. The vector number is used to calculate displacement into the CPU16 exception vector table. Vector numbers are formed by concatenating the value in the ICR IVBA field with a 4-bit value supplied by the GPT when an interrupt request is made.

Hardware prevents the vector number from changing while it is being driven out on the IMB. Vector number assignment is shown in Table 8–2.

At reset, IVBA is initialized to \$0. To enable interrupt-driven timer operation, the upper nibble of a user-defined vector number (\$40–\$FF) must be written to IVBA, and interrupt handler routines must be located at the addresses pointed to by the corresponding vector. Note that IVBA must be written before GPT interrupts are enabled, or the GPT could supply a vector number (\$00 to \$0F) that corresponds to an assigned or reserved exception vector.

The internal GPT interrupt priority hierarchy is shown in Table 8–2. The lower the interrupt source number, the higher the priority. A single GPT interrupt source can be given priority over all other GPT interrupt sources by assigning the priority adjust field (PAB) in the ICR a value equal to its source number.

Interrupt requests are asserted until associated status flags are cleared. Status flags must be cleared in a particular sequence. The status register must first be read for set flags, then zeros must be written to the flags that are to be cleared. If a new event occurs between the time that the register is read and the time that it is written, the associated flag is not cleared.

Refer to **SECTION 4 SINGLE-CHIP INTEGRATION MODULE** and **SECTION 5 CENTRAL PROCESSING UNIT** for more information about exceptions and interrupts.

8.5 Pin Descriptions

The GPT uses 12 of the MCU pins. Each pin can perform more than one function. Descriptions of GPT pins divided into functional groups follow.

8.5.1 Input Capture Pins (IC[1:3])

Each of these pins is associated with a single GPT input capture function. Each pin has hysteresis. Any pulse longer than two system clocks is guaranteed to be valid and any pulse shorter than one system clock is ignored. Each pin has an associated 16-bit capture register that holds the captured counter value. These pins can also be used for general-purpose I/O. Refer to **8.8.2 Input Capture Functions** for more information.

8.5.2 Input Capture/Output Compare Pin (IC4/OC5)

This pin can be configured for use by either an input capture or an output compare function. It has an associated 16-bit register that is used for holding either the input capture value or the output match value. When used for input capture the pin has the same hysteresis as other input capture pins. The pin can be used for general-purpose I/O. Refer to **8.8.2 Input Capture Functions** and **8.8.3 Output Compare Functions** for more information.

8.5.3 Output Compare Pins (OC[1:4])

These pins are used for GPT output compare functions. Each pin has an associated 16-bit compare register and a 16-bit comparator. Pins OC2, OC3, and OC4 are associated with a specific output compare function. The OC1 function can affect the output of all compare pins. If the OC1 pin is not needed for an output compare function it can be used to output the clock selected for the timer counter register. Any of these pins can also be used for general-purpose I/O. Refer to **8.8.3 Output Compare Functions** for more information.

8.5.4 Pulse Accumulator Input Pin (PAI)

The PAI pin connects a discrete signal to the pulse accumulator for timed or gated pulse accumulation. PAI has hysteresis. Any pulse longer than two system clocks is guaranteed to be valid and any pulse shorter than one system clock is ignored. It can be used as a general-purpose input pin. Refer to **8.10 Pulse Accumulator** for more information.

8.5.5 Pulse-Width Modulation (PWMA, PWMB)

PWMA and PWMB pins carry pulse-width modulator outputs. The modulators can be programmed to generate a periodic waveform of variable frequency and duty cycle. PWMA can be used to output the clock selected as the input to the PWM counter. These pins can also be used for general-purpose output. Refer to **8.11 Pulse-Width Modulation Unit** for more information.

8.5.6 Auxiliary Timer Clock Input (PCLK)

PCLK connects an external clock to the GPT. The external clock can be used as the clock source for the capture/compare unit or the PWM unit in place of one of the prescaler outputs. PCLK has hysteresis. Any pulse longer than two system clocks is guaranteed to be valid and any pulse shorter than one system clock is ignored. This pin can also be used as a general-purpose input pin. Refer to **8.7 Prescaler** for more information.

8.6 General-Purpose I/O

Any GPT pin can be used for general-purpose I/O when it is not used for another purpose. Capture/compare pins are bidirectional, others can be used only for output or input. I/O direction is controlled by a data direction bit in the port GP data direction register (DDRGP).

Parallel data is read from and written to the port GP data register (PORTGP). Pin data can be read even when pins are configured for a timer function. Data read from PORTGP always reflects the state of the external pin, while data written to PORTGP may not always affect the external pin.

Data written to PORTGP does not immediately affect pins used for output compare functions, but the data is latched. When an output compare function is disabled, the last data written to PORTGP is driven out on the associated pin if it is configured as an output. Data written to PORTGP can cause input captures if the corresponding pin is configured for input capture function.

The pulse accumulator input (PAI) and the external clock input (PCLK) pins provide general-purpose input. The state of these pins can be read by accessing the PAIS and PCLKS bits in the pulse accumulator control register (PACTL).

Pulse-width modulation A and B (PWMA/PWMB) output pins can serve as general-purpose outputs. The force PWM value (FPWMx) and the force logic one (F1x) bits in the compare force (CFORC) and PWM control (PWMC) registers, respectively, control their operation.

8.7 Prescaler

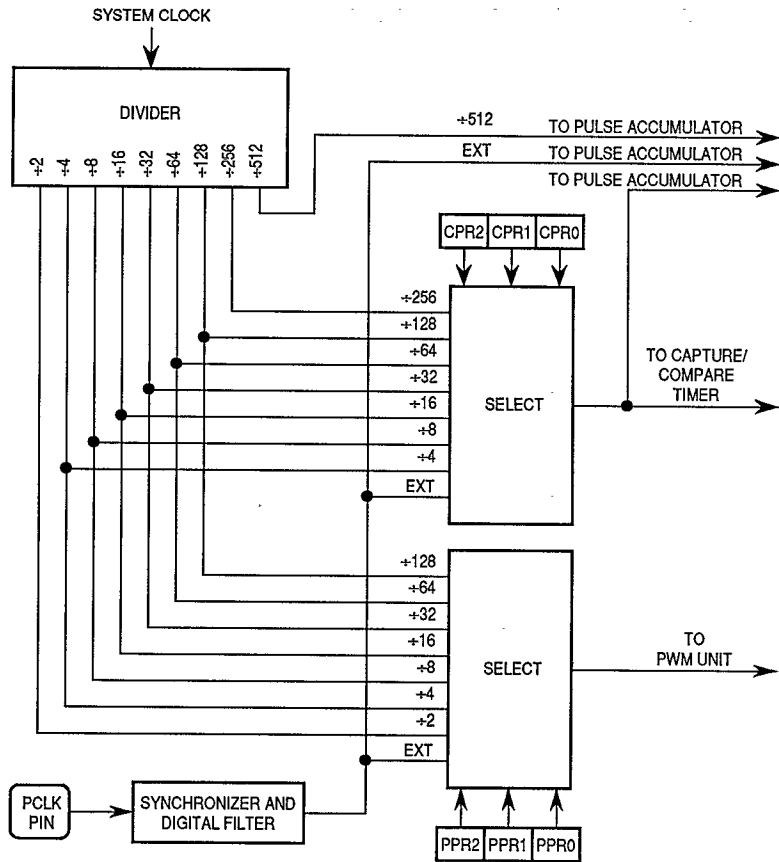
Capture/compare and PWM units have independent 16-bit free-running counters as a main timing component. These counters derive their clocks from the prescaler or from the PCLK input. Figure 8–2 is a prescaler block diagram.

In the prescaler, the system clock is divided by a nine-stage divider chain. Prescaler outputs equal to system clock divided by 2, 4, 8, 16, 32, 64, 128, 256 and 512 are provided. Connected to these outputs are two multiplexers, one for the capture/compare unit, the other for the PWM unit.

Multiplexers can each select one of seven prescaler taps or an external input from the PCLK pin. Multiplexer output for the timer counter (TCNT) is selected by bits CPR[2:0] in timer interrupt mask register 2 (TMSK2). Multiplexer output for the PWM counter (PWMCNT) is selected by bits PPR[2:0] in PWM control register C (PWMC).

After reset, the GPT is configured to use system clock divided by four for TCNT and system clock divided by two for PWMCNT. Initialization software can change the division factor. The PPR bits can be written at any time but the CPR bits can only be written once after reset unless the GPT is in test or freeze mode.

The prescaler can be read at any time. In freeze mode the prescaler can also be written. Word accesses must be used to ensure coherency. If coherency is not needed byte accesses can be used. The prescaler value is contained in bits [8:0] while bits [15:9] are unimplemented and are read as zeros.



OPT PRESCALER BLOCK

Figure 8–2. Prescaler Block Diagram

Multiplexer outputs (including the PCLK signal) can be connected to external pins. The CPROUT bit in the TMSK2 register configures the OC1 pin to output the TCNT clock and the PPROUT bit in the PWMC register configures the PWMA pin to output the PWMC clock. CPROUT and PPROUT can be written at any time. Clock signals on OC1 and PWMA do not have a 50% duty cycle. They have the period of the selected clock but are high for only one system clock time.

The prescaler also supplies three clock signals to the pulse accumulator clock select mux. These are the system clock divided by 512, the external clock signal from the PCLK pin and the capture/compare clock signal.

8.8 Capture/Compare Unit

The capture/compare unit contains the timer counter (TCNT), the input capture (IC) functions and the output compare (OC) functions. Figure 8-3 is a block diagram of the capture/compare unit.

8.8.1 Timer Counter

The timer counter (TCNT) is the key timing component in the capture/compare unit. The timer counter is a 16-bit free-running counter that starts counting after the processor comes out of reset. The counter cannot be stopped during normal operation. After reset, the GPT is configured to use the system clock divided by four as the input to the counter. The prescaler divides the system clock and provides selectable input frequencies. User software can configure the system to use one of seven prescaler outputs or an external clock.

The counter can be read any time without affecting its value. Because the GPT is interfaced to the IMB and the IMB supports a 16-bit bus, a word read gives a coherent value. If coherency is not needed, byte accesses can be made. The counter is set to \$0000 during reset and is normally a read-only register. In test mode and freeze mode, any value can be written to the timer counter.

When the counter rolls over from \$FFFF to \$0000, the timer overflow flag (TOF) in timer interrupt flag register 2 (TFLG2) is set. An interrupt can be enabled by setting the corresponding interrupt enable bit (TOI) in timer interrupt mask register 2 (TMSK2). Refer to **8.4.2 GPT Interrupts** for more information.

8.8.2 Input Capture Functions

All GPT input capture functions use the same 16-bit timer counter (TCNT). Each input capture pin has a dedicated 16-bit latch and input edge-detection/selection logic. Each input capture function has an associated status flag, and can cause the GPT to make an interrupt service request.

When a selected edge transition occurs on an input capture pin, the associated 16-bit latch captures the content of TCNT and sets the appropriate status flag. An interrupt request can be generated when the transition is detected.

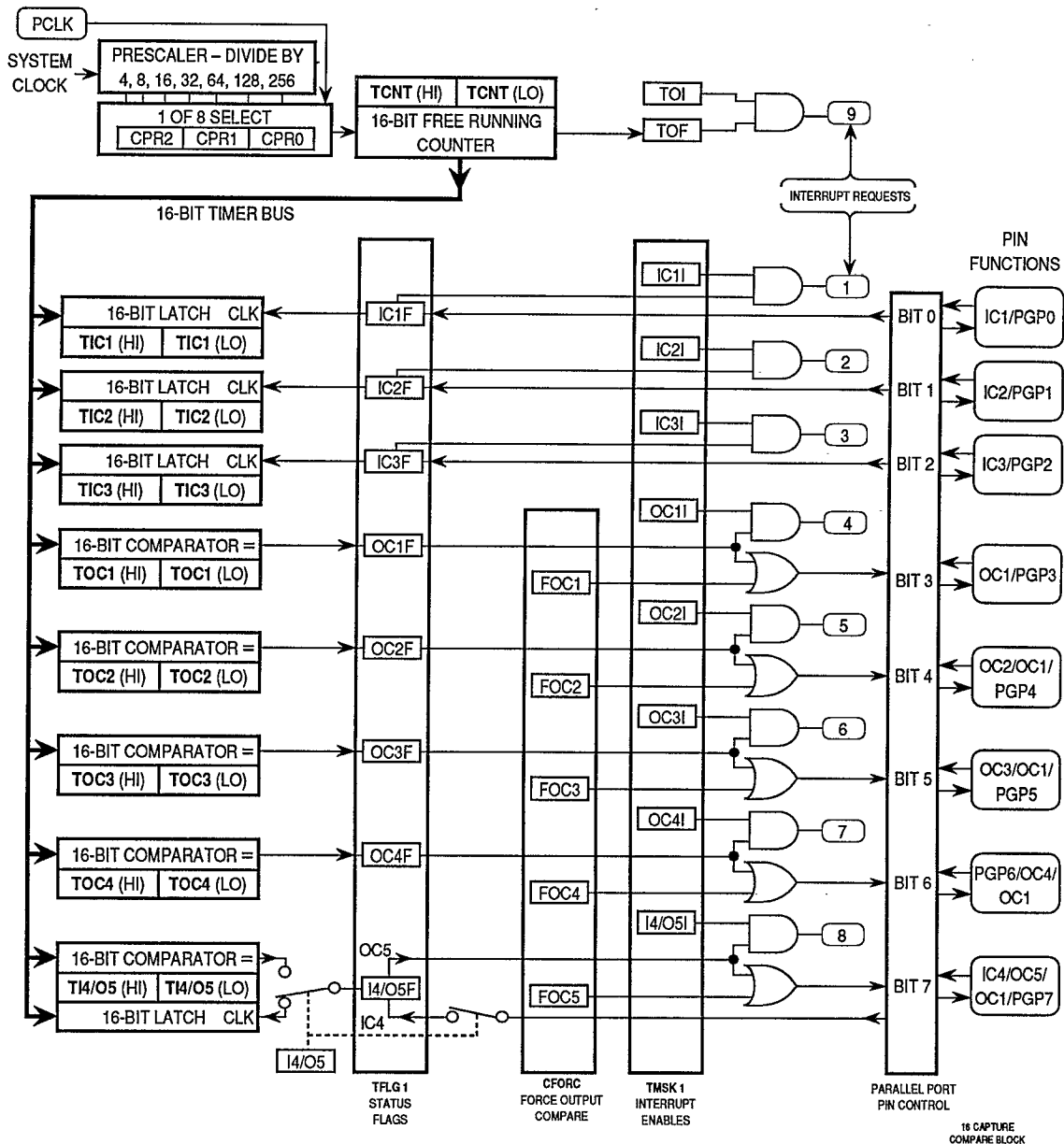


Figure 8-3. Capture/Compare Unit Block Diagram

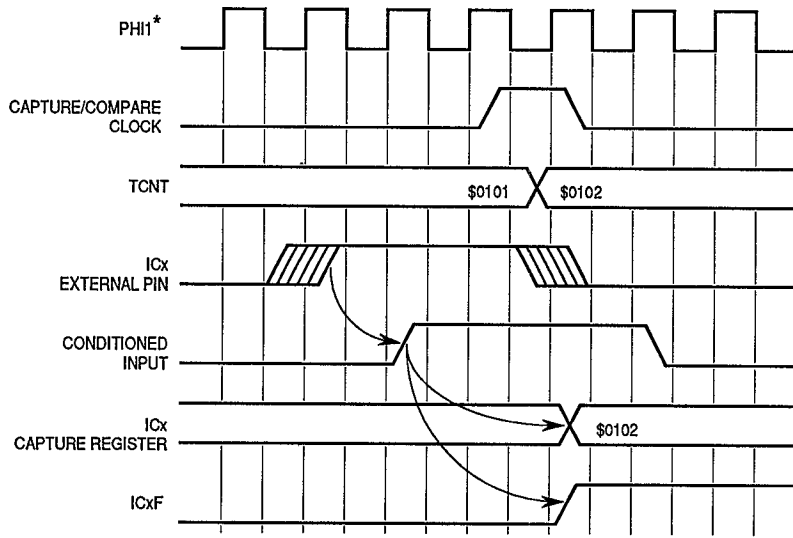
Edge-detection logic consists of control bits that enable edge detection and select a transition to detect. The EDGExA and EDGExB bits in timer control register 2 (TCTL2) determine whether the input capture functions detect rising edges only, falling edges only, or both rising and falling edges. Clearing both bits disables the input capture function. Input capture functions operate independently of each other and can capture the same TCNT value if individual input edges are detected within the same timer count cycle.

Input capture interrupt logic includes a status flag, which indicates that an edge has been detected, and an interrupt enable bit. An input capture event sets the ICxF bit in the timer interrupt flag register 1 (TFLG1) and causes the GPT to make an interrupt request if the corresponding ICxI bit is set in the timer interrupt mask register 1 (TMSK1). If the ICxI bit is cleared, software must poll the status flag to determine that an event has occurred. Refer to **8.4 Polled and Interrupt-Driven Operation** for more information.

Input capture events are generally asynchronous to the timer counter. Because of this, input capture signals are conditioned by a synchronizer and digital filter. Events are synchronized with the system clock so that latching of TCNT content and counter incrementation occur on opposite half-cycles of the system clock. Inputs have hysteresis. Capture of any transition longer than two system clocks is guaranteed; any transition shorter than one system clock has no effect.

Figure 8–4 shows the relationship of system clock to synchronizer output. The value latched into the capture register is the value of the counter several system clock cycles after the transition that triggers the edge detection logic. There can be up to one clock cycle of uncertainty in latching of the input transition. Maximum time is determined by the system clock frequency.

The input capture register is a 16-bit register. A word access is required to ensure coherency. If coherency is not required, byte accesses can be used to read the register. Input capture registers can be read at any time without affecting their values.



NOTES:

The conditioned input signal causes the current value of TCNT to be latched by the ICx capture register. The ICxF flag is set at the same time.

*PHI1 is the same frequency as the system clock; however, it does not have the same timing.

16 INPUT CAPTURE TIM

Figure 8–4. Input Capture Timing Example

An input capture occurs every time a selected edge is detected, even when the input capture status flag is set. This means that the value read from the input capture register corresponds to the most recent edge detected, which may not be the edge that caused the status flag to be set.

8.8.3 Output Compare Functions

Each GPT output compare pin has an associated 16-bit compare register and a 16-bit comparator. Each output compare function has an associated status flag, and can cause the GPT to make an interrupt service request. Output compare logic is designed to prevent false compares during data transition times.

When the programmed content of an output compare register matches the value in TCNT, an output compare status flag (OCxF) bit in TFLG1 is set. If the appropriate interrupt enable bit (OCxI) in TMSK1 is set, an interrupt request is made when a match occurs. Refer to **8.4.2 GPT Interrupts** for more information.

Operation of output compare 1 differs from that of the other output compare functions. OC1 control logic can be programmed to make state changes on other OC pins when an OC1 match occurs. Control bits in the timer compare force register (CFORC) allow for early forced compares.

8.8.3.1 Output Compare 1

Output compare 1 can affect any or all of OC[1:5] when an output match occurs. In addition to allowing generation of multiple control signals from a single comparison operation, this function makes it possible for two or more output compare functions to control the state of a single OC pin. Output pulses as short as one timer count can be generated in this way.

The OC1 action mask register (OC1M) and the OC1 action data register (OC1D) control OC1 function. Setting a bit in OC1M selects a corresponding bit in the GPT parallel data port. Bits in OC1D determine whether selected bits are to be set or cleared when an OC1 match occurs. Pins must be configured as outputs in order for the data in the register to be driven out on the corresponding pin. If an OC1 match and another output match occur at the same time and both attempt to alter the same pin, the OC1 function controls the state of the pin.

8.8.3.2 Forced Output Compare

Timer compare force register (CFORC) is used to make forced compares. The action taken as a result of a forced compare is the same as when an output compare match occurs, except that status flags are not set. Forced channels take programmed actions immediately after the write to CFORC.

The CFORC register is implemented as the upper byte of a 16-bit register which also contains the PWM control register C (PWMC). It can be accessed as eight bits or a word access can be used. Reads of force compare bits (FOC) have no meaning and always return zeros. These bits are self-negating.

8.9 Input Capture 4/Output Compare 5

The IC4/OC5 pin can be used for input capture, output compare, or general-purpose I/O. A function enable bit (I4/O5) in the pulse accumulator control register (PACTL) configures the pin for input capture (IC4) or output compare function (OC5). Both bits are cleared during reset, configuring the pin as an input, but also enabling the OC5 function. IC4/OC5 I/O functions are controlled by DDGP7 in the port GP data direction register (DDRGP).

The 16-bit register (TI4/O5) used with the IC4/OC5 function acts as an input capture register or as an output compare register depending on which function is selected. When used as the input capture 4 register, it cannot be written to except in test or freeze mode.

8.10 Pulse Accumulator

The pulse accumulator counter (PACNT) is an 8-bit read/write up-counter. PACNT can operate in external event counting or gated time accumulation modes. Figure 8–5 is a block diagram of the pulse accumulator.

In event counting mode, the counter increments each time a selected transition of the pulse accumulator input (PAI) pin is detected. The maximum clocking rate is the system clock divided by four.

In gated time accumulation mode a clock increments PACNT while the PAI pin is in the active state. There are four possible clock sources.

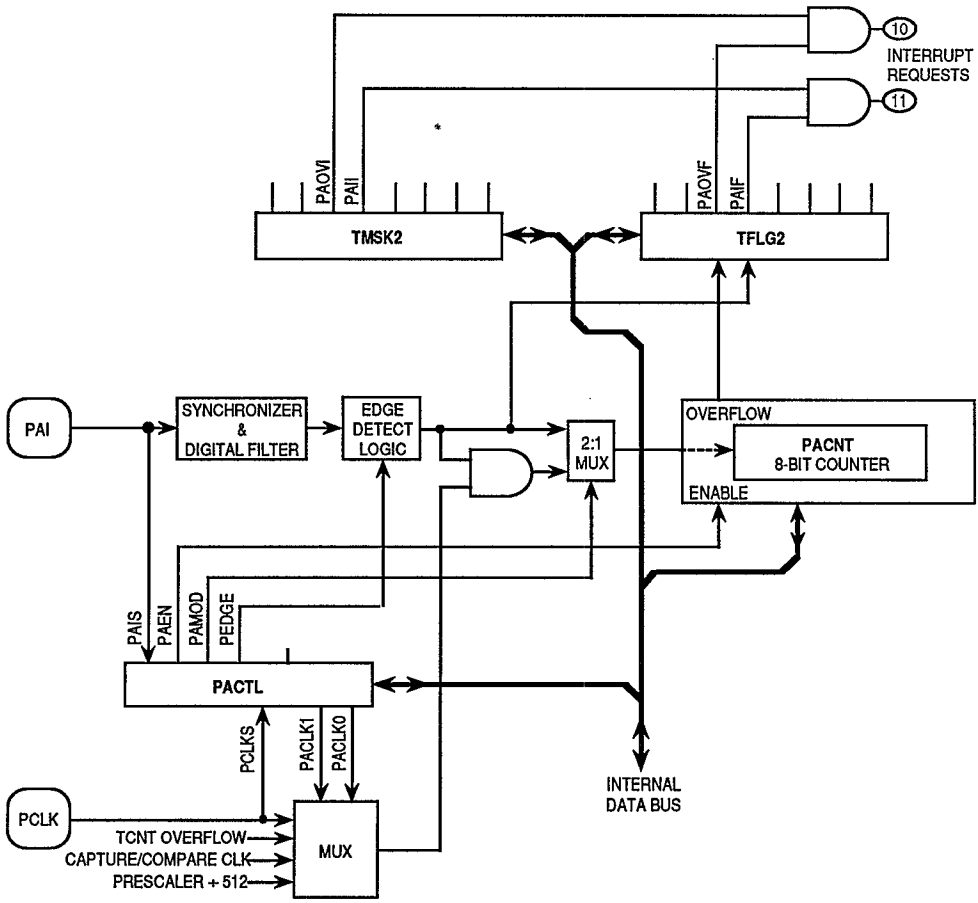
Two bits in the TFLG2 register show pulse accumulator status. The pulse accumulator flag (PAIF) indicates that a selected edge has been detected at the PAI pin. The pulse accumulator overflow flag (PAOVF) indicates that the pulse accumulator count has rolled over from \$FF to \$00. This can be used to extend the range of the counter beyond eight bits.

An interrupt request can be made when each of the status flags is set. However, operation of the PAI interrupt depends on operating mode. In event counting mode, an interrupt is requested when the edge being counted is detected. In gated mode, the request is made when the PAI input changes from active to inactive state. Interrupt requests are enabled by the PAOVI and PAII bits in the TMSK2 register.

Bits in the pulse accumulator control register (PACTL) control the operation of PACNT. The PAMOD bit selects event counting or gated operation. In event counting mode, the PEDGE control bit determines whether a rising or falling edge is detected; in gated mode, PEDGE specifies the active state of the gate signal. Bits PACLK[1:0] select the clock source used in gated mode.

PACTL and PACNT are implemented as one 16-bit register, but can be accessed with byte or word access cycles. Both registers are cleared at reset, but the PAIS and PCLKS bits show the state of the PAI and PCLK pins.

The PAI pin can also be used for general-purpose input. The logic state of the PAIS bit in PACTL shows the state of the pin.



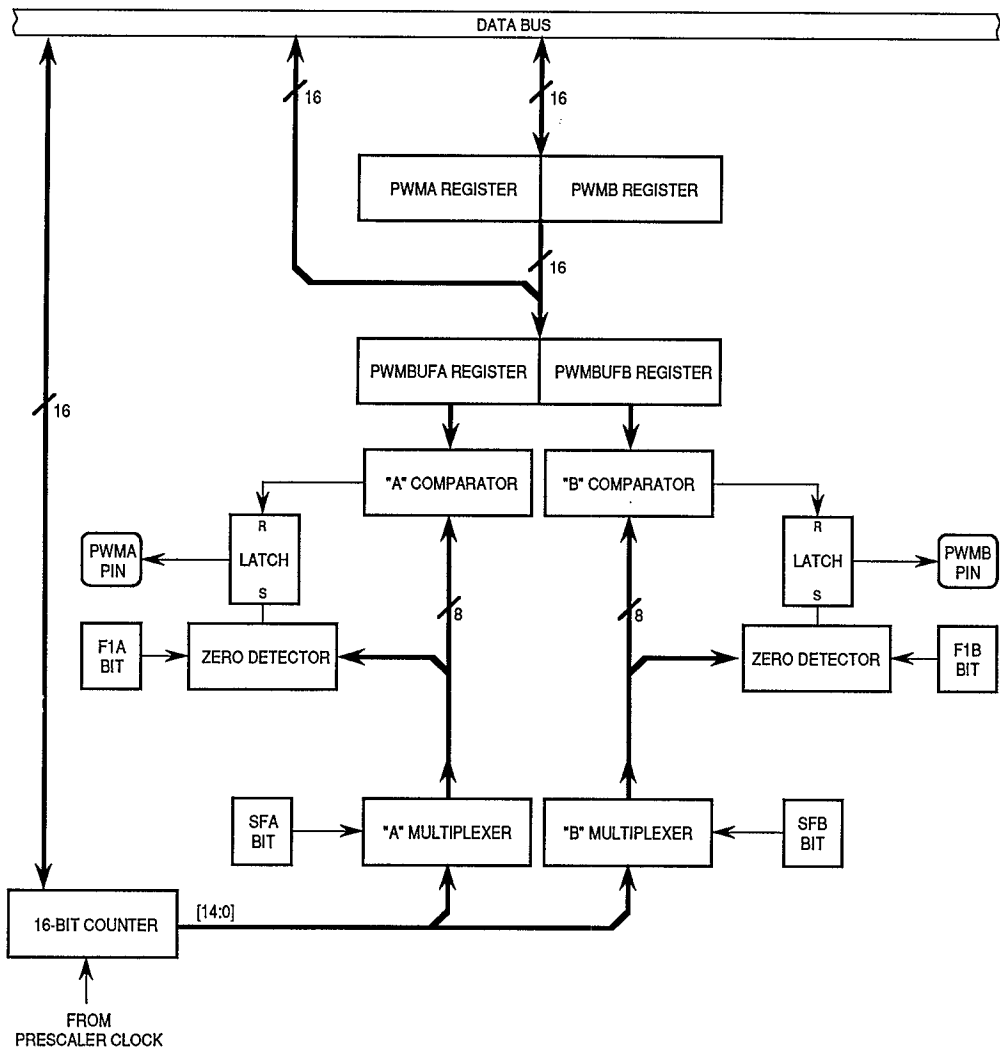
16 PULSE ACC BLOCK

Figure 8-5. Pulse Accumulator Block Diagram

8.11 Pulse-Width Modulation Unit

The pulse-width modulation (PWM) unit has two output channels, PWMA and PWMB. A single clock output from the prescaler multiplexer drives a 16-bit counter that is used to control both channels. Figure 8-6 is a block diagram of the pulse-width modulation unit.

The PWM unit has two operational modes. Fast mode uses a clocking rate equal to 1/256 of the prescaler output rate; slow mode uses a rate equal to 1/32768 of the prescaler output rate. The duty cycle ratios of the two PWM channels can be individually controlled by software. The PWMA pin can also output the clock that drives the PWM counter. PWM pins can also be used as output pins.



16 PWM BLOCK

Figure 8-6. PWM Block Diagram

8.11.1 PWM Counter

The 16-bit counter in the PWM unit is similar to the timer counter in the capture/compare unit. During reset, the GPT is configured to use the system clock divided by two to drive the counter. Initialization software can reconfigure the counter to use one of seven prescaler outputs or an external clock input from the PCLK pin.

The PWM count register (PWMCNT) can be read at any time without affecting its value. A read must be a word access to ensure coherence, but byte accesses can be made if coherence is not needed. The counter is cleared to \$0000 during reset and is a read-only register except in freeze or test mode.

Fifteen of the sixteen counter bits are output to multiplexers A and B. The multiplexers provide the fast and slow modes of the PWM unit. Mode for PWMA is selected by the SFA bit in the PWM control register C (PWMC). Mode for PWMB is selected by the SFB bit in the same register.

PWMA, PWMB, and PPR[2:0] bits in PWMC control PWM output frequency. In fast mode, bits [7:0] of PWMCNT are used to clock the PWM logic; in slow mode, bits [14:7] are used. The period of a PWM output in slow mode is 128 times longer than the fast mode period. Table 8–3 shows a range of PWM output frequencies using a 16.78-MHz system clock.

**Table 8–3. PWM Frequency Range
Using 16.78-MHz System Clock**

PPR[2:0]	Prescaler Tap	Fast Mode	Slow Mode
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

8.11.2 PWM Function

The pulse width values of the PWM outputs are determined by control registers PWMA and PWMB. PWMA and PWMB are 8-bit registers implemented as two bytes of a 16-bit register. PWMA and PWMB can be accessed as separate bytes or as one 16-bit register. A value of \$00 loaded into either register causes the corresponding output pin to output a continuous logic level zero signal. A value of \$80 causes the corresponding output signal to have a 50% duty cycle, and so on, to the maximum value of \$FF, which corresponds to an output which is at logic level one for 255/256 of the cycle.

Setting the F1A (for PWMA) or F1B (for PWMB) bits in the CFORC register causes the corresponding pin to output a continuous logic level one signal. The logic level of the associated pin does not change until the end of the current cycle. F1A and F1B are the lower two bits of CFORC, but can be accessed at the same word address as PWMC.

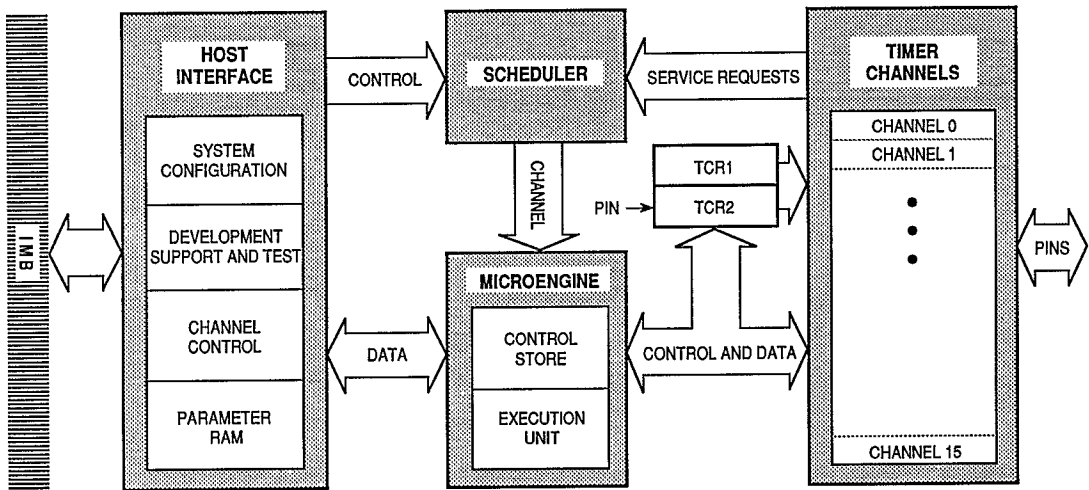
Data written to PWMA and PWMB is not used until the end of a complete cycle. This prevents spurious short or long pulses when register values are changed. The current duty cycle value is stored in the appropriate PWM buffer register (PWMBUFA or PWMBUFB). The new value is transferred from the PWM register to the buffer register at the end of the current cycle.

Registers PWMA, PWMB, and PWMC are reset to \$00 during reset. These registers may be written or read at any time. PWMC is implemented as the lower byte of a 16-bit register. The upper byte is the CFORC register. The buffer registers, PWMBUFA and PWMBUFB, are read-only at all times and may be accessed as separate bytes or as one 16-bit register.

Pins PWMA and PWMB can also be used for general-purpose output. The values of the F1A and F1B bits in PWMC are driven out on the corresponding PWM pins when normal PWM operation is disabled. When read, the F1A and F1B bits reflect the states of the PWMA and PWMB pins.

SECTION 9 TIME PROCESSOR UNIT

The time processor unit (TPU) is an intelligent, semi-autonomous microcontroller designed for timing control. Operating simultaneously with the CPU, the TPU schedules tasks, processes ROM instructions, accesses shared data, and performs input and output. Figure 9-1 is a simplified block diagram of the TPU.



TPU BLOCK

Figure 9-1. TPU Block Diagram

9.1 Overview

The TPU can be viewed as a special-purpose microcomputer that performs a programmable series of two operations, match and capture. Each occurrence of either operation is called an event. A programmed series of events is called a function. TPU functions replace software functions that would require host CPU interrupt service. The following pre-programmed timing functions are currently available:

- Input capture/input transition counter
- Output compare
- Pulse-width modulation
- Synchronized pulse-width modulation
- Period measurement with additional transition detect
- Period measurement with missing transition detect
- Position-synchronized pulse generator
- Stepper motor
- Period/pulse-width accumulator

9.2 TPU Components

The TPU module consists of two 16-bit time bases, sixteen independent timer channels, a task scheduler, a microengine, and a host interface. In addition, a dual-port parameter RAM is used to pass parameters between the module and the host CPU.

9.2.1 Time Bases

Two 16-bit counters provide reference time bases for all output compare and input capture events. Prescalers for both time bases are controlled by the host CPU via bit fields in the TPU module configuration register (TPUMCR). Timer count registers TCR1 and TCR2 provide access to current counter values. TCR1 and TCR2 can be read/write accessed in microcode, but are not directly available to the host CPU. The TCR1 clock is derived from the system clock. The TCR2 clock can be derived from the system clock or from an external clock input via the T2CLK pin.

9.2.2 Timer Channels

The TPU has 16 independent channels, each connected to an MCU pin. The channels have identical hardware. Each channel consists of an event register and pin control logic. The event register contains a 16-bit capture register, a 16-bit compare/match register, and a 16-bit greater-than-or-equal-to comparator. The direction of each pin, either output or input, is determined by the TPU microengine. Each channel can either use the same time base for match and capture, or can use one time base for match and the other for capture.

9.2.3 Scheduler

When a service request is received, the scheduler determines which TPU channel is serviced by the microengine. A channel can request service for one of four reasons: for host service, for a link to another channel, for a match event, or for a capture event. The host system assigns each active channel one of three priorities: high, middle, or low. When multiple service requests are received simultaneously, a priority-scheduling mechanism grants service based on channel number and assigned priority.

9.2.4 Microengine

The microengine is composed of a control store and an execution unit. Control-store ROM holds the microcode for each factory-masked time function. When assigned to a channel by the scheduler, the execution unit executes microcode for a function assigned to that channel by the host CPU. Microcode can also be executed from the TPURAM module instead of the control store. The TPURAM module allows emulation and development of custom TPU microcode without the generation of a microcode ROM mask. Refer to **9.3.6 Emulation Support** for more information.

9.2.5 Host Interface

Host interface registers allow communication between the host CPU and the TPU, both before and during execution of a time function. The registers are accessible from the IMB through the TPU bus interface unit. Refer to **9.5 Host Interface Registers**, and **APPENDIX D REGISTER SUMMARY** for register bit/field definitions and address mapping.

9.2.6 Parameter RAM

Parameter RAM occupies 256 bytes at the top of the system address map. Channel parameters are organized as 128 16-bit words. Although all parameter word locations in RAM can be accessed by all channels, only 100 are normally used: channels 0 to 13 use six parameter words, while channels 14 and 15 each use eight parameter words. The parameter RAM address map in **APPENDIX D REGISTER SUMMARY** shows how parameter words are organized in memory.

The host CPU specifies function parameters by writing the appropriate RAM address. The TPU reads the RAM to determine channel operation. The TPU can also store information to be read by the CPU in RAM. Detailed descriptions of the parameters required by each time function are beyond the scope of this manual. Refer to the *TPU Reference Manual (TPURM/AD)* for more information.

For pre-programmed functions, one of the parameter words associated with each channel contains three channel control fields. These fields perform the following functions:

- PSC — Forces the output level of the pin.
- PAC — For input capture, PAC specifies the edge transition to be detected. For output comparison, PAC specifies the logic level to be output when a match occurs.
- TBS — Specifies channel direction (input or output) and assigns a time base to the input capture and output compare functions of the channel.

9.3 TPU Operation

All TPU functions are related to one of the two 16-bit timebases. Functions are synthesized by combining sequences of match events and capture events. Because the primitives are implemented in hardware, the TPU can determine precisely when a match or capture event occurs, and respond rapidly. An event register for each channel provides for simultaneity of match/capture event occurrences on all channels.

When a match or input capture event requiring service occurs, the affected channel generates a service request to the scheduler. The scheduler determines the priority of the request and assigns the channel to the microengine at the first available time. The microengine performs the function defined by the content of the control store or emulation RAM, using parameters from the parameter RAM.

9.3.1 Event Timing

Match and capture events are handled by independent channel hardware. This provides an event accuracy of one time-base clock period, regardless of the number of channels that are active. An event normally causes a channel to request service. However, before an event can be serviced, any pending previous requests must be serviced. The time needed to respond to and service an event is determined by the number of channels requesting service, the relative priorities of the channels requesting service, and the microcode execution time of the active functions. Worst-case event service time (latency) determines TPU performance in a given application. Latency can be closely estimated — see *Motorola TPU Reference Manual* (TPURM/AD) for more information.

9.3.2 Channel Orthogonality

Most timer systems are limited by the fixed number of functions assigned to each pin. All TPU channels contain identical hardware and are functionally

equivalent in operation, so that any channel can be configured to perform any time function. Any function can operate on the calling channel, and, under program control, on another channel determined by the program or by a parameter. The user controls the combination of time functions.

9.3.3 Interchannel Communication

The autonomy of the TPU is enhanced by the ability of a channel to affect the operation of one or more other channels without CPU intervention. Interchannel communication can be accomplished by issuing a link service request to another channel, by controlling another channel directly, or by accessing the parameter RAM of another channel.

9.3.4 Programmable Channel Service Priority

The TPU provides a programmable service priority level to each channel. Three priority levels are available. When more than one channel of a given priority requests service at the same time, arbitration is accomplished according to channel number. To prevent a single high-priority channel from permanently blocking other functions, other service requests of the same priority are performed in channel order after the lowest-numbered, highest-priority channel is serviced.

9.3.5 Coherency

For data to be coherent, all available portions of it must be identical in age, or must be logically related. As an example, consider a 32-bit counter value that is read and written as two 16-bit words. The 32-bit value is read-coherent only if both 16-bit portions are updated at the same time, and write-coherent only if both portions take effect at the same time. Parameter RAM hardware supports coherent access of two adjacent 16-bit parameters. The host CPU must use a long-word operation to guarantee coherency.

9.3.6 Emulation Support

Although factory-programmed time functions can perform a wide variety of control tasks, they may not be ideal for all applications. The TPU provides emulation capability that allows the user to develop new time functions. Emulation mode is entered by setting the EMU bit in the TPUMCR. In emulation mode, an auxiliary bus connection is made between TPURAM and the TPU module, and access to TPURAM via the intermodule bus is disabled. A 9-bit address bus, a 32-bit data bus, and control lines transfer information between the modules. To ensure exact emulation, RAM module access timing remains consistent with access timing of the TPU ROM control store.

To support changing TPU application requirements, Motorola has established a TPU function library. The function library is a collection of TPU functions written for easy assembly in combination with each other or with custom functions. Refer to Motorola Programming Note TPUPN00/D, *Using the TPU Function Library and TPU Emulation Mode* for information about developing custom functions and accessing the TPU function library. Refer to the *TPU Reference Manual* (TPURM/AD) for more information about specific functions.

9.3.7 TPU Interrupts

Each of the TPU channels can generate an interrupt service request. Interrupts for each channel must be enabled by writing to the appropriate control bit in the channel interrupt enable register (CIER). The channel interrupt status register (CISR) contains one interrupt status flag per channel. Time functions set the flags. Setting a flag bit causes the TPU to make an interrupt service request if the corresponding channel interrupt enable bit is set and the interrupt request level is nonzero.

The value of the channel interrupt request level (CIRL) field in TICR determines the priority of all TPU interrupt service requests. CIRL values correspond to MCU interrupt request signals $\overline{\text{IRQ}}[7:1]$. $\overline{\text{IRQ}}7$ is the highest-priority request signal; $\overline{\text{IRQ}}1$ has the lowest priority. Assigning a value of %111 to CIRL causes $\overline{\text{IRQ}}7$ to be asserted when a TPU interrupt request is made; lower field values cause corresponding lower-priority interrupt request signals to be asserted. Assigning CIRL a value of %000 disables all interrupts.

The CPU recognizes only interrupt requests of a priority greater than the value contained in the interrupt priority (IP) mask in the condition code register. When the CPU acknowledges an interrupt request, the priority of the acknowledged interrupt is written to the IP mask and is driven out onto the IMB address lines.

When the IP mask value driven out on the address lines is the same as the CIRL value, the TPU contends for arbitration priority. The IARB field in TPUMCR contains the TPU arbitration number. Each module that can make an interrupt service request must be assigned a unique non-zero IARB value in order to implement an arbitration scheme. Arbitration is performed by means of serial assertion of IARB field bit values. IARB is initialized to \$0 during reset.

When the TPU wins arbitration, it must respond to the CPU interrupt acknowledge cycle by placing an interrupt vector number on the data bus. The vector number is used to calculate displacement into the exception vector table. Vectors are formed by concatenating the 4-bit value of the CIBV field in the TPU interrupt configuration register with the 4-bit number of the channel requesting interrupt service. Since the CIBV field has a reset value of %00, it must be assigned a value corresponding to the upper nibble of a block of 16 user-

defined vector numbers before TPU interrupts are enabled, or a TPU interrupt service request could cause the CPU to take one of the reserved vectors in the exception vector table.

Refer to **SECTION 4 SINGLE-CHIP INTEGRATION MODULE** for more information about interrupts. For more information about the exception vector table refer to **SECTION 5 CENTRAL PROCESSING UNIT**.

9.4 Time Functions

The following paragraphs describe factory-programmed time functions implemented in TPU microcode ROM. A complete description of the functions is beyond the scope of this manual. Refer to the *TPU Reference Manual* (TPURM/AD) for additional information.

9.4.1 Discrete Input/Output

When a pin is used as a discrete input, a parameter indicates the current input level and the previous 15 levels of a pin. Bit 15, the most significant bit of the parameter, indicates the most recent state. Bit 14 indicates the next most recent state, and so on. The programmer can choose one of the three following conditions to update the parameter: 1) when a transition occurs, 2) when the CPU makes a request, or 3) when a rate specified in another parameter is matched. When a pin is used as a discrete output, it is set high or low only upon request by the CPU.

9.4.2 Input Capture/Input Transition Counter

Any channel of the TPU can capture the value of a specified TCR upon the occurrence of each transition or specified number of transitions, and then generate an interrupt request to notify the CPU. A channel can perform input captures continually, or a channel can detect a single transition or specified number of transitions, then cease channel activity until reinitialization. After each transition or specified number of transitions, the channel can generate a link to a sequential block of up to eight channels. The user specifies a starting channel of the block and the number of channels within the block. The generation of links depends on the mode of operation. In addition, after each transition or specified number of transitions, one byte of the parameter RAM (at an address specified by channel parameter) can be incremented and used as a flag to notify another channel of a transition.

9.4.3 Output Compare

The output compare (OC) function generates a rising edge, falling edge, or a toggle of the previous edge in one of three ways:

1. Immediately upon CPU initiation, thereby generating a pulse with a length equal to a programmable delay time.
2. At a programmable delay time from a user-specified time.
3. Continuously. Upon receiving a link from a channel, OC references, without CPU interaction, a specifiable period and calculates an offset:

$$\text{OFFSET} = \text{PERIOD} * \text{RATIO}$$

where RATIO is a parameter supplied by the user.

This algorithm generates a 50% duty-cycle continuous square wave with each high/low time equal to the calculated OFFSET. Due to offset calculation, there is an initial link time before continuous pulse generation begins.

9.4.4 Pulse-Width Modulation

The TPU can generate a pulse-width modulation (PWM) waveform with any duty cycle from zero to 100% (within the resolution and latency capability of the TPU). To define the PWM, the CPU provides one parameter that indicates the period and another parameter that indicates the high time. Updates to one or both of these parameters can direct the waveform change to take effect immediately, or coherently beginning at the next low-to-high transition of the pin.

9

9.4.5 Synchronized Pulse-Width Modulation

The TPU generates a PWM waveform in which the CPU can change the period and/or high time at any time. When synchronized to a time function on a second channel, the synchronized PWM (SPWM) low-to-high transitions have a time relationship to transitions on the second channel.

9.4.6 Period Measurement with Additional Transition Detect

This function and the following function are used primarily in toothed-wheel speed-sensing applications, such as monitoring rotational speed of an engine. The period measurement with additional transition detect (PMA) function allows for a special-purpose 23-bit period measurement. It can detect the occurrence of an additional transition (caused by an extra tooth on the sensed wheel) indicated by a period measurement that is less than a programmable ratio of the previous period measurement. Once detected, this condition can be counted and compared to a programmable number of additional transitions detected before TCR2 is reset to \$FFFF. Alternatively, a byte at an address specified by a channel parameter can be read and used as a flag. A nonzero value of the flag indicates that TCR2 is to be reset to \$FFFF once the next additional transition is detected.

9.4.7 Period Measurement with Missing Transition Detect

Period measurement with missing transition detect (PMM) allows a special-purpose 23-bit period measurement. It detects the occurrence of a missing transition (caused by a missing tooth on the sensed wheel), indicated by a period measurement that is greater than a programmable ratio of the previous period measurement. Once detected, this condition can be counted and compared to a programmable number of additional transitions detected before TCR2 is reset to \$FFFF. In addition, one byte at an address specified by a channel parameter can be read and used as a flag. A nonzero value of the flag indicates that TCR2 is to be reset to \$FFFF once the next missing transition is detected.

9.4.8 Position-Synchronized Pulse Generator

Any channel of the TPU can generate an output transition or pulse, which is a projection in time based on a reference period previously calculated on another channel. Both TCRs are used in this algorithm: TCR1 is internally clocked, and TCR2 is clocked by a position indicator in the user's device. An example of a TCR2 clock source is a sensor that detects special teeth on the flywheel of an automobile using PMA or PMM. The teeth are placed at known degrees of engine rotation; hence, TCR2 is a coarse representation of engine degrees, i.e., each count represents some number of degrees.

Up to 15 position-synchronized pulse generator (PSP) function channels can operate with a single input reference channel executing a PMA or PMM input function. The input channel measures and stores the time period between the flywheel teeth and resets TCR2 when the engine reaches a reference position. The output channel uses the period calculated by the input channel to project output transitions at specific engine degrees. Because the flywheel teeth might be 30 or more degrees apart, a fractional multiplication operation resolves down to the desired degrees. Two modes of operation allow pulse length to be determined either by angular position or by time.

9.4.9 Stepper Motor

The stepper motor (SM) control algorithm provides for linear acceleration and deceleration control of a stepper motor with a programmable number of step rates of up to 14. Any group of channels, up to eight, can be programmed to generate the control logic necessary to drive a stepper motor.

The time period between steps (P) is defined as

$$P(r) = K1 - K2 * r$$

where r is the current step rate (1–14), and $K1$ and $K2$ are supplied as parameters.

After providing the desired step position in a 16-bit parameter, the CPU issues a step request. Next, the TPU steps the motor to the desired position through an acceleration/deceleration profile defined by parameters. The parameter indicating the desired position can be changed by the CPU while the TPU is stepping the motor. This algorithm changes the control state every time a new step command is received.

A 16-bit parameter initialized by the CPU for each channel defines the output state of the associated pin. The bit pattern written by the CPU defines the method of stepping, such as full stepping or half stepping. With each transition, the 16-bit parameter rotates one bit. The period of each transition is defined by the programmed step rate.

9.4.10 Period/Pulse-Width Accumulator

The period/pulse-width accumulator (PPWA) algorithm accumulates a 16-bit or 24-bit sum of either the period or the pulse width of an input signal over a programmable number of periods or pulses (from 1 to 255). After an accumulation period, the algorithm can generate a link to a sequential block of up to eight channels. The user specifies a starting channel of the block and number of channels within the block. Generation of links depends on the mode of operation.

Any channel can be used to measure an accumulated number of periods of an input signal. A maximum of 24 bits can be used for the accumulation parameter. From 1 to 255 period measurements can be made and summed with the previous measurement(s) before the TPU interrupts the CPU, allowing instantaneous or average frequency measurement, and the latest complete accumulation (over the programmed number of periods).

The pulse width (high-time portion) of an input signal can be measured (up to 24 bits) and added to a previous measurement over a programmable number of periods (1 to 255). This provides an instantaneous or average pulse-width measurement capability, allowing the latest complete accumulation (over the specified number of periods) to always be available in a parameter.

By using the output compare function in conjunction with PPWA, an output signal can be generated that is proportional to a specified input signal. The ratio of the input and output frequency is programmable. One or more output signals with different frequencies, yet proportional and synchronized to a single input signal, can be generated on separate channels.

9.5 Host Interface Registers

The TPU memory map contains three groups of registers:

- System Configuration Registers
- Channel Control and Status Registers
- Development Support and Test Verification Registers

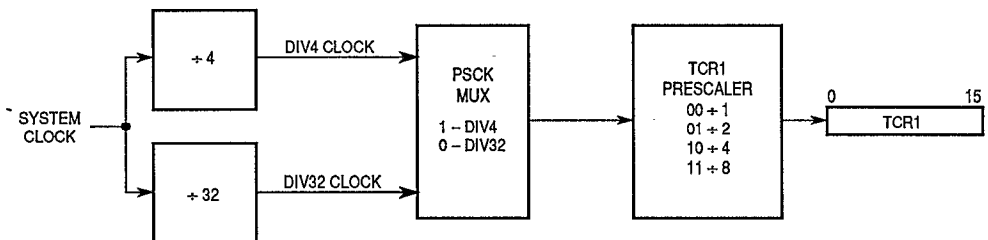
All registers except the channel interrupt status register (CISR) must be read or written by means of word accesses. The address space of the TPU memory map occupies 512 bytes. Unused registers within the 512-byte address space return zeros when read.

9.5.1 System Configuration Registers

The TPU configuration control registers, TPUMCR and TICR, determine the value of the prescaler, perform emulation control, specify whether the external TCR2 pin functions as a clock source or as gate of the DIV8 clock for TCR2, and determine interrupt request level and interrupt vector number assignment. Refer to **APPENDIX D REGISTER SUMMARY** for more information about TPUMCR and TICR.

9.5.1.1 Prescaler Control for TCR1

Timer control register one (TCR1) is clocked from the output of a prescaler. Two fields in the TPUMCR control TCR1. The prescaler's input is the internal TPU system clock divided by either 4 or 32, depending on the value of the PSCK (prescaler clock) bit. The prescaler divides this input by 1, 2, 4, or 8, depending on the value of TCR1P (timer count register 1 prescaler control). Channels using TCR1 have the capability to resolve down to the TPU system clock divided by 4. Refer to Figure 9–2 and Table 9–1.



PRESCALER CTL BLOCK 1

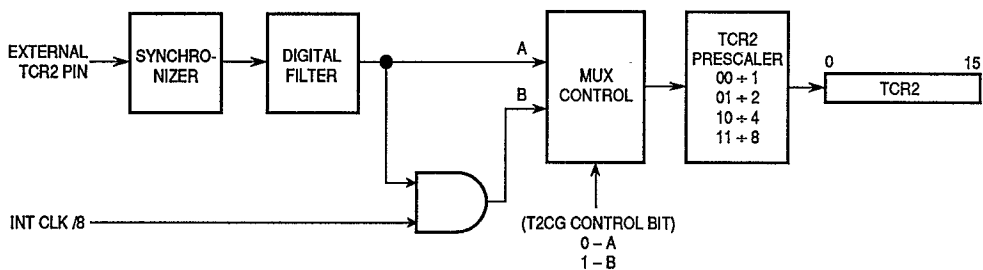
Figure 9–2. TCR1 Prescaler Control

Table 9–1. TCR1 Prescaler Control

TCR1 Prescaler	Divide By	PSCK = 0		PSCK = 1	
		Number of Clocks	Rate at 16 MHz	Number of Clocks	Rate at 16 MHz
00	1	32	2 ms	4	250 ns
01	2	64	4 ms	8	500 ns
10	4	128	8 ms	16	1 ms
11	8	256	16 ms	32	2 ms

9.5.1.2 Prescaler Control for TCR2

Timer control register two (TCR2), like TCR1, is clocked from the output of a prescaler. The T2CG (TCR2 clock/gate control) bit in TPUMCR determines whether the external TCR2 pin functions as an external clock source for TCR2 or as the gate in the use of TCR2 as a gated pulse accumulator. The function of the T2CG bit is shown in Figure 9–3.



PRESCALER CTL BLOCK 2

Figure 9–3. TCR2 Prescaler Control

When the T2CG bit is set, the external TCR2 pin functions as a gate of the DIV8 clock (the TPU system clock divided by 8). In this case, when the external TCR2 pin is low, the DIV8 clock is blocked, preventing it from incrementing TCR2. When the external TCR2 pin is high, TCR2 is incremented at the frequency of the DIV8 clock. When T2CG is cleared, an external clock from the TCR2 pin, which has been synchronized and fed through a digital filter, increments TCR2.

The TCR2 field in TPUMCR specifies the value of the prescaler: 1, 2, 4, or 8. Channels using TCR2 have the capability to resolve down to the TPU system clock divided by 8. Table 9–2 is a summary of prescaler output.

Table 9–2. TCR2 Prescaler Control

TCR2 Prescaler	Divide By	Internal Clock Divided By	External Clock Divided By
00	1	8	1
01	2	16	2
10	4	32	4
11	8	64	8

9.5.1.3 Emulation Control

Asserting the EMU bit in the TPUMCR places the TPU in emulation mode. In emulation mode, the TPU executes microinstructions from TPURAM exclusively. Access to the TPURAM module through the IMB by a host is blocked, and the TPURAM module is dedicated for use by the TPU. After reset, EMU can be written only once.

9.5.1.4 Low-Power Stop Control

If the STOP bit in the TPUMCR is set, the TPU shuts down its internal clocks, shutting down the internal microengine. TCR1 and TCR2 cease to increment and retain the last value before the stop condition was entered. The TPU asserts the stop flag (STF) in the TPUMCR to indicate that it has stopped.

9.5.2 Channel Control Registers

The channel control and status registers enable the TPU to control channel interrupts, assign time functions to be executed on a specified channel, or select the mode of operation or the type of host service request for the time function specified. Refer to Table 9–3.

9.5.2.1 Channel Interrupt Enable and Status Registers

The channel interrupt enable register (CIER) allows the CPU to enable or disable the ability of individual TPU channels to request interrupt service. Setting the appropriate bit in the register enables a channel to make an interrupt service request; clearing a bit disables the interrupt.

The channel interrupt status register (CISR) contains one interrupt status flag per channel. Time functions specify via microcode when an interrupt flag is set. Setting a flag causes the TPU to make an interrupt service request if the corresponding CIER bit is set and the CIRL field has a nonzero value. To clear a status flag, read CISR, then write a zero to the appropriate bit. CISR is the only TPU register that can be accessed on a byte basis.

9.5.2.2 Channel Function Select Registers

Encoded 4-bit fields within the channel function select registers specify one of 16 time functions to be executed on the corresponding channel. Encodings for predefined functions in the TPU ROM are found in Table 9–3.

9.5.2.3 Host Sequence Registers

The host sequence field selects the mode of operation for the time function selected on a given channel. The meaning of the host sequence bits depends on the time function specified. Refer to Table 9–3, which is a summary of the host sequence and host service request bits for each time function.

9.5.2.4 Host Service Registers

The host service request field selects the type of host service request for the time function selected on a given channel. The meaning of the host service request bits depends on the time function specified. Refer to Table 9–3.

A host service request field cleared to %00 signals the host that service is completed by the microengine on that channel. The host can request service on a channel by writing the corresponding host service request field to one of three nonzero states. It is a good practice to monitor the host service request register and wait until the TPU clears the service request before changing any parameters or issuing a new service request to the channel.

Table 9-3. Host Sequence Code/Host Service Request Code

Function Name	FunctionCode	Host Service Request Code	Host Sequence Code*
Discrete Input/Output	\$8	1 = Force Output High 2 = Force Output Low 3 = Initialization, Input Specified 3 = Initialization, Periodic Input 3 = Update Pin Status Parameter	0 = Trans Mode — Record Pin on Transition 0 = Trans Mode — Record Pin on Transition 0 = Trans Mode — Record Pin on Transition 1 = Match Mode — Record Pin at Match_Rate 2 = Record Pin State on HSR 11
ITC Input Capture/ Input Transition Counter	\$A	0 = None 1 = Initialization 2 = (Not Implemented) 3 = (Not Implemented)	0 = No Link, Single Mode 1 = No Link, Continuous Mode 2 = Link, Single Mode 3 = Link, Continuous Mode
OC Output Compare	\$E	0 = None 1 = Host-Initiated Pulse Mode 2 = (Not Implemented) 3 = Continuous Pulse Mode	0 = Execute All Functions 1 = Execute All Functions 2 = Only Update TCRn Parameters 3 = Only Update TCRn Parameters
PWM Pulse-Width Modulation	\$9	0 = None 1 = Immediate Update Request 2 = Initialization 3 = (Not Implemented)	(None Implemented)
SPWM Synchronized Pulse-Width Modulation	\$7	0 = None 1 = (Not Implemented) 2 = Initialization 3 = Immediate Update Request	0 = Mode 0 1 = Mode 1 2 = Mode 2 3 = (Not Implemented)
PMA/PMM Period Measurement with Additional/Missing Transition Detect	\$B	0 = None 1 = Initialization 2 = (Not Implemented) 3 = (Not Implemented)	0 = PMA Bank Mode 1 = PMA Count Mode 2 = PMM Bank Mode 3 = PMM Count Mode
PSP Position-Synchronized Pulse Generator	\$C	0 = None 1 = Immediate Update Request 2 = Initialization 3 = Force Change	0 = Pulse Width Set by Angle 1 = Pulse Width Set by Time 2 = Pulse Width Set by Angle 3 = Pulse Width Set by Time
SM Stepper Motor	\$D	0 = None 1 = None 2 = Initialization 3 = Step Request	(None Implemented)
PPWA Period/Pulse-Width Accumulator	\$F	0 = None 1 = (Not Implemented) 2 = Initialization 3 = (Not Implemented)	0 = 24-Bit Period 1 = 16-Bit Period + Link 2 = 24-Bit Pulse Width 3 = 16-Bit Pulse Width + Link

9.5.2.5 Channel Priority Registers

The channel priority registers (CPR1, CPR2) assign one of three priority levels to a channel or disable the channel. Table 9–4 indicates the number of time slots guaranteed for each channel priority encoding.

Table 9–4. Channel Priority Encodings

CHX[1:0]	Service	Guaranteed Time Slots
00	Disabled	—
01	Low	1 out of 7
10	Middle	2 out of 7
11	High	4 out of 7

9.5.3 Development Support and Test Registers

These registers are used for custom microcode development or for factory test. Describing the use of the registers is beyond the scope of this manual. Register descriptions are provided in **APPENDIX D REGISTER SUMMARY**. Please refer to the *TPU Reference Manual* (TPURM/AD) for more information.

SECTION 10 STANDBY RAM WITH TPU EMULATION

The standby RAM with TPU emulation (TPURAM) module consists of a control register block and a two Kbyte array of fast (two bus cycle) static RAM, which is especially useful for system stacks and variable storage. The RAM can be used to emulate TPU microcode ROM.

10.1 General

The TPURAM can be mapped to any two Kbyte boundary in the address map, but must not overlap the module control registers, as overlapping makes the registers inaccessible. TPURAM responds to both program and data space accesses. Data can be read or written in bytes, words, or long words. The RAM is powered by V_{DD} in normal operation. During power-down, the RAM contents are maintained by power on standby voltage pin V_{STBY} . Power switching between sources is automatic. A power-down status flag (PDS) indicates when voltage applied to V_{STBY} has fallen below a reference level for a specified period of time.

10.2 TPURAM Register Block

TPURAM control registers occupy a 64-byte block. There are three registers in the block: the RAM module configuration register (TRAMMCR), the RAM test register (TRAMTST), and the RAM array base address register (TRAMBAR). The rest of the block consists of unimplemented register locations. Unimplemented register addresses are read as zeros. Writes to them have no effect. Refer to **APPENDIX D REGISTER SUMMARY** for the register block address map and register bit/field definitions.

The modmap (MM) bit in the single-chip integration module configuration register (SCIMCR) defines the most significant bit (ADDR23) of the IMB address for each module in the MCU. Because the CPU16 drives ADDR[23:20] to the same logic state as ADDR19, MM must equal one. Refer to **SECTION 4 SINGLE-CHIP INTEGRATION MODULE** for more information about how the state of MM affects the system.

10.3 TPURAM Array Address Mapping

Base address register TRAMBAR specifies the RAM array base address in the MCU memory map. Writing a base address into TRAMBAR enables access to the RAM array. TRAMBAR can be written only once after reset. This prevents accidental remapping of the array. The RAM array disable status (RAMDS) flag in TRAMBAR is set when a base address is written to TRAMBAR.

NOTE

ADDR[23:20] are driven to the same logic state as ADDR19. The RAM array must not be mapped to addresses \$7FF000–\$7FFFFF, which are inaccessible to the CPU16. If mapped to these addresses, the array remains inaccessible until a reset occurs.

10.4 SRAM Array Address Space Type

The RASP field in TRAMMCR determines RAM array address space type. The TPURAM module can respond to both program and data space accesses or to program space accesses only. This allows code to be executed from RAM, and permits use of program counter relative addressing mode for operand fetches from the array. Refer to **SECTION 4 SINGLE-CHIP INTEGRATION MODULE** and **SECTION 5 CENTRAL PROCESSING UNIT** for more information concerning address space types and program/data space access.

10.5 Normal Access

The array can be accessed by byte, word, or long word. A byte or aligned word access takes one bus cycle or two system clocks. A long word or misaligned word access requires two bus cycles. Refer to **SECTION 4 SINGLE-CHIP INTEGRATION MODULE** for more information concerning access times.

10.6 Standby and Low-Power Stop Operation

Standby and low-power operation should not be confused. Standby mode maintains the RAM array when the MCU main power supply is turned off. Low-power mode allows the central processing unit to control MCU power consumption.

Relative voltage levels of the MCU V_{DD} and V_{STBY} pins determine whether the TPURAM is in standby mode. TPURAM circuitry switches to the standby power source when specified limits are exceeded. If specified standby supply voltage levels are maintained during the transition, there is no loss of memory when switching occurs. The RAM array cannot be accessed while the module is powered from V_{STBY} . If standby operation is not desired, connect the V_{STBY} pin to the V_{SS} pin.

A power-down status (PDS) flag in TRAMMCR indicates when the voltage level on the V_{STBY} pin falls below the specified minimum level. PDS is cleared when a loss of standby power occurs. PDS must be set and monitored by software in order to detect power loss.

I_{SB} exceeds specified maximum standby current during the time V_{DD} transitions from normal operating level to the level specified for standby operation. This occurs within the voltage range $V_{SB} - 0.5 \text{ V} \geq V_{DD} \geq V_{SS} + 0.5 \text{ V}$. Typically, I_{SB} peaks when $V_{DD} \approx V_{SB} - 1.5 \text{ V}$, and averages 1.0 mA over the transition period.

Refer to **APPENDIX A ELECTRICAL CHARACTERISTICS** for standby switching and power consumption specifications.

To prevent standby supply voltage from going below the specified minimum, a filter capacitor must be attached between the V_{STBY} and V_{SS} pins. To calculate filter capacitance, V_{DD} supply ramp time, available standby voltage, and available standby current must be known. Assuming that the rate of change is constant as V_{DD} transitions from 0.0 V to 5.5 V (nominal V_{SS} to nominal V_{DD}) and that V_{SB} also drops during this period, capacitance is calculated using the expression:

$$C = \frac{It}{V}$$

Where:

C = Desired capacitance

I = I_{SB} differential (Transient I_{SB} – Available supply current)

t = time of maximum I_{SB} (Typically in the range $V_{SB} - 1.5 \text{ V} \pm 0.5 \text{ V}$)

V = V_{SB} differential (Available supply voltage – Specified minimum V_{SB})

Setting the STOP bit in RAMMCR switches the TPURAM module to low-power mode. In low-power mode, the array retains its contents, but cannot be read or written by the CPU. Because the CPU16 always operates in supervisor mode, STOP can be read or written at any time. STOP is set during reset. Stop mode is exited by clearing STOP.

The TPURAM module will switch to standby mode while it is in low-power mode, provided the operating constraints discussed above are met.

10.7 TPU ROM Emulation

The TPURAM array can be used to emulate the microcode ROM in the TPU module. This provides a means of developing custom TPU code. The TPU selects TPU emulation mode. While in TPU emulation mode, the access timing of the TPURAM module matches the timing of the TPU microinstruction ROM to ensure accurate emulation. Normal accesses through the IMB are inhibited and the control registers have no effect, allowing external RAM to emulate the TPURAM at the same addresses. Refer to **SECTION 9 TIME PROCESSOR UNIT** for more information.

10.8 Reset

Reset places the TPURAM in normal mode and clears the base address register. A new base address must be written into TRAMBAR before the RAM array can be accessed.

When a synchronous reset occurs while a byte or word RAM 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. Refer to **SECTION 4 SINGLE-CHIP INTEGRATION MODULE** for more information concerning resets.

SECTION 11 MASKED ROM MODULE

The masked ROM module (MRM) consists of a control register block and an 8 Kbyte mask-programmed read-only memory array. The default base address of the array is specified by the customer, but the array can be remapped to any 64 Kbyte boundary in the system memory map. The primary function of the MRM is to serve as nonvolatile memory for the microcontroller. It can be configured to support system bootstrap during reset. The MRM can also operate in a special emulator mode that simplifies emulation of the array by an external device.

11.1 General

The ROM array in the MCU contains 48 Kbytes. It is arranged in 16-bit words, and is accessed via the intermodule bus. Bytes, words, and misaligned words can be accessed. The MRM can be programmed to insert wait states to accommodate migration from slow external development memory. Access time depends upon the number of wait states specified at mask programming time, but can be as fast as two system clocks for byte and aligned words. The MRM also responds to back-to-back IMB accesses to provide two bus cycle long word access.

The array base address must be on a 64 Kbyte boundary, must not overlap the control registers of other microcontroller modules, and should not overlap the control register block. The array occupies the low-order locations in the 64 Kbyte block — accesses to the remaining 16 Kbytes of unimplemented locations in the block are ignored by the MRM, allowing other system resources or external devices to respond to the access. If the array is mapped to overlap the control registers of other modules, accesses to those registers will be indeterminate; if the array is mapped to overlap the MRM control registers, accesses to the registers are still possible, but accesses to the overlapping 32 bytes of ROM will be ignored.

11.2 MRM Register Block

There are three MRM control registers: the masked ROM module configuration register (MRMCR), and the ROM array base address registers (ROMBAH and ROMBAL). In addition, the MRM register block contains signature registers (RSIGHI and RSIGLO), and ROM bootstrap words (ROMBS[0:3]).

The modmap (MM) bit in the single-chip integration module configuration register (SCIMCR) defines the most significant bit (ADDR23) of the IMB address for each module in the MCU. Because the CPU16 drives ADDR[23:20] to the same logic state as ADDR19, MM must equal one. Refer to **SECTION 4 SINGLE-CHIP INTEGRATION MODULE** for more information about how the state of MM affects the system.

The control register block size for the MRM consists of 32 bytes, but not all locations are implemented. Unimplemented register addresses are read as zeros, and writes have no effect. Refer to **APPENDIX D REGISTER SUMMARY** for the register bit/field definitions.

11.3 MRM Array Address Mapping

Base address registers ROMBAH and ROMBAL are used to specify the ROM array base address in the MCU memory map. Although the base address contained in ROMBAH and ROMBAL is mask-programmed, these registers can be written after reset to change the default array address.

NOTE

ADDR[23:20] are driven to the same logic state as ADDR19. The ROM array must not be mapped to addresses \$7FF000–\$7FFFFFFF, which are inaccessible to the CPU16. If mapped to these addresses, the array remains inaccessible until a reset occurs.

ROMBAH and ROMBAL can only be written while the ROM is in low-power mode (MRMCR STOP = 1) and the base address lock is disabled (MRMCR LOCK = 0). LOCK can only be written once to a value of one. This prevents accidental remapping of the array.

11.4 MRM Array Address Space Type

The ASPC field in MRMCR determines ROM array address space type. The module can respond to both program and data space accesses or to program space accesses only. This allows code to be executed from ROM, and permits use of program counter relative addressing mode for operand fetches from the array. The default value of the ASPC field is established during mask programming, but field value can be changed after reset. Refer to **SECTION 4 SINGLE-CHIP INTEGRATION MODULE** and **SECTION 5 CENTRAL PROCESSING UNIT** for more information about address space types and program/data space access.

11.5 Normal Access

The array can be accessed by byte, word, or long word. A byte or aligned word access takes a minimum of one bus cycle (two system clocks). A long word or misaligned word access requires a minimum of two bus cycles. Access time can be optimized for a particular application by inserting wait states into each access. The number of wait states inserted is determined by the value of the MRMCR WAIT field. Two, three, four, or five clock bus-cycle accesses can be specified. The default value of the WAIT field is established during mask programming, but field value can be changed after reset. Refer to **SECTION 4 SINGLE-CHIP INTEGRATION MODULE** for more information about access times.

11.6 Emulation Mode Operation

Emulation mode is enabled by the EMUL bit in the MRMCR. EMUL state is determined by the state of the DATA10 and DATA13 lines during reset. If both data lines are held low, EMUL is set, and ROM emulation mode is enabled.

While emulation mode is enabled, the internal module chip select signal ($\overline{\text{CSM}}$) is asserted whenever a valid access to an address assigned to the masked ROM module is made. To be valid, an access must be within the range specified by the ROM base array registers and must meet the address space requirements defined by the ASPC field in MRMCR. $\overline{\text{CSM}}$ is asserted for all valid read accesses; it is asserted for write accesses only in background debug mode. The MRM does not acknowledge an access on the IMB while in emulation mode — this causes the SCIM to run an external bus cycle. The $\overline{\text{CSM}}$ signal is asserted on the falling edge of $\overline{\text{AS}}$. Internal $\overline{\text{DSACK}}$ is generated by the ROM module after it has inserted the number of wait states specified by the WAIT field in the MRMCR.

11.7 Low-Power Stop Operation

Low-power mode minimizes MCU power consumption. Setting the STOP bit in MRMCR switches the MRM to low-power mode. In low-power mode, the array cannot be read by the CPU. The reset state of STOP is the complement of the logic state of DATA14 during reset. Exit stop mode by clearing STOP.

11.8 ROM Signature

Signature registers RSIGHI and RSIGLO contain a factory-specified mask-programmed signature pattern. A custom signature algorithm can then allow the user to verify ROM array content.

11.9 Reset

The state of the MRM following reset is determined by the default values programmed into MRMCR BOOT, LOCK, ASPC, and WAIT. The default array base address is determined by the value programmed into ROMBAL and ROMBAH.

When the default value of the MRMCR $\overline{\text{BOOT}}$ bit is zero, the contents of MRM bootstrap words ROMBS[0:3] are used as CPU16 reset vectors. When the default value of the MRMCR $\overline{\text{BOOT}}$ bit is one, reset vectors are fetched from external memory, and system integration module chip-select logic can be used to assert an external ROM select signal ($\overline{\text{CSBOOT}}$) during reset. Refer to **SECTION 4 SINGLE-CHIP INTEGRATION MODULE** for more information about external boot ROM selection.

When a synchronous reset occurs while a byte or word MRM 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 the ROM array can be corrupted by asynchronous reset. Refer to **SECTION 4 SINGLE-CHIP INTEGRATION MODULE** for more information about resets.

APPENDIX A ELECTRICAL CHARACTERISTICS

This appendix contains electrical specification tables and reference timing diagrams. Each timing diagram has an associated key table made up of parameters abstracted from the specification tables. Pertinent notes have been included in the key tables.

A

Table A-1. Maximum Ratings

Num	Rating	Symbol	Value	Unit
1	Supply Voltage ^{1,2,7}	V_{DD}	-0.3 to +6.5	V
2	Input Voltage ^{1,2,3,5,7}	V_{in}	-0.3 to +6.5	V
3	Instantaneous Maximum Current Single Pin Limit (applies to all pins) ^{1,5,6,7}	I_D	25	mA
4	Operating Maximum Current Digital Input Disruptive Current ^{4,5,6,7} $V_{NEGCLAMP} \cong -0.3$ V $V_{POSCLAMP} \cong V_{DD} + 0.3$ V	I_{iD}	-500 to +500	μ A
5	Operating Temperature Range MC68HC16Y1 "C" Suffix MC68HC16Y1 "V" Suffix MC68HC16Y1 "M" Suffix	T_A	TL to TH -40 to +85 -40 to +105 -40 to +125	$^{\circ}$ C
6	Storage Temperature Range	T_{stg}	-55 to +150	$^{\circ}$ C

NOTES:

1. Permanent damage can occur if maximum ratings are exceeded. Exposure to voltages or currents in excess of recommended values affects device reliability. Device modules may not operate normally while being exposed to electrical extremes.
2. Although sections of the device contain circuitry to protect against damage from high static voltages or electrical fields, take normal precautions to avoid exposure to voltages higher than maximum-rated voltages.
3. All functional non-supply pins are internally clamped to V_{SS} . All functional pins except EXTAL and XFC are internally clamped to V_{DD} .
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. Power supply must maintain regulation within operating V_{DD} range during instantaneous and operating maximum current condition.
6. This parameter is periodically sampled rather than 100% tested.
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.

A

Table A-2. Thermal Characteristics

Num	Characteristic	Symbol	Value	Unit
1	Thermal Resistance Plastic 160-Pin Surface Mount	Θ_{JA}	37	$^{\circ}\text{C/W}$

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

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

where

T_A = Ambient Temperature, $^{\circ}\text{C}$

Θ_{JA} = Package Thermal Resistance, Junction-to-Ambient, $^{\circ}\text{C/W}$

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:

$$P_D = K + (T_J + 273^{\circ}\text{C}) \quad (2)$$

Solving equations 1 and 2 for K gives:

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

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

A

Table A-3. Clock Control Timing
 (V_{DD} and $V_{DDSYN} = 5.0 \text{ Vdc} \pm 10\%$, $V_{SS} = 0 \text{ Vdc}$, $T_A = T_L$ to T_H ,
 4.194 MHz reference)

Num	Characteristic	Symbol	Min	Max	Unit
1	PLL Reference Frequency Range	f_{ref}	3.2	4.2	MHz
2	System Frequency ¹ On-Chip PLL Frequency External Clock Operation	f_{sys}	dc 0.131 dc	16.78 16.78 16.78	MHz
3	PLL Startup Time ^{2,3,4,5}	t_{spll}	—	50	ms
4	PLL Lock Time ^{2,4,5,6}	t_{lpll}	—	20	ms
5	Limp Mode Clock Frequency ⁷ SYNCR X bit = 0 SYNCR X bit = 1	f_{limp}	— —	$f_{sys} \text{ max} / 2$ $f_{sys} \text{ max}$	MHz
6	CLKOUT Stability ^{2,4,5,8} Short term (5 μs interval) Long term (500 μs interval)	C_{stab}	-0.5 -0.05	0.5 0.05	%

NOTES:

- All internal registers retain data at 0 Hz.
- This parameter is periodically sampled rather than 100% tested.
- Parameter measured from the time V_{DD} and V_{DDSYN} are valid to $\overline{\text{RESET}}$ release.
- Assumes that a low-leakage external filter capacitor with a value of 0.1 μF is attached to the XFC pin. Total external resistance from the XFC pin due to external leakage sources must be greater than 15 $\text{M}\Omega$ to guarantee this specification.
- Proper layout procedures must be followed to achieve specifications.
- 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 to $\overline{\text{RESET}}$ release. 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.
- Determined by the initial control voltage applied to the on-chip VCO. The X bit in SYNCR controls a divide by two scaler on the system clock output.
- Stability 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 C_{stab} percentage for a given interval.

Table A-4. DC Characteristics

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

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 ^{1,9}	V_{HYS}	0.5	—	V
4	Input Leakage Current ² $V_{in} = V_{DD}$ or V_{SS} All input-only pins except ADC pins	I_{in}	-2.5	2.5	μA
5	High Impedance (Off-State) Leakage Current ² $V_{in} = V_{DD}$ or V_{SS} All input/output and output pins	I_{OZ}	-2.5	2.5	μA
6	CMOS Output High Voltage ^{2,3} $I_{OH} = -10.0 \mu\text{A}$ Group 1,2,4 input/output and all output pins	V_{OH}	$V_{DD} - 0.2$	—	V
7	CMOS Output Low Voltage ² $I_{OL} = 10.0 \mu\text{A}$ Group 1,2,4 input/output and all output pins	V_{OL}	—	0.2	V
8	Output High Voltage ^{2,3} $I_{OH} = -0.8 \text{ mA}$ Group 1,2,4 input/output and all output pins	V_{OH}	$V_{DD} - 0.8$	—	V
9	Output Low Voltage ² $I_{OL} = 1.6 \text{ mA}$ Group 1 I/O Pins, CLKOUT, FREEZE/QUOT, IPIPE0 $I_{OL} = 5.3 \text{ mA}$ Group 2 and Group 4 I/O Pins, CSBOOT, BG/CS $I_{OL} = 12 \text{ mA}$ Group 3	V_{OL}	—	0.4	V
10	Data Bus Mode Select Pull-Up Current ⁵ $V_{in} = V_{IL}$ DATA[15:0] $V_{in} = V_{IH}$ DATA[15:0]	I_{MSP}	— -15	-120 —	μA
11	V_{DD} Supply Current ⁶ RUN ⁴ RUN, TPU emulation mode LPSTOP, 4.194-MHz crystal, VCO Off (STSCIM = 0) LPSTOP (External clock input frequency = maximum f_{sys})	I_{DD}	— — — —	132 142 2 10	mA 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 (STSCIM = 0) 4.194 MHz crystal, V_{DD} powered down	I_{DDSYN}	— — — —	2 7 2 2	mA mA mA mA
14	RAM Standby Voltage ⁷ Specified V_{DD} applied $V_{DD} = V_{SS}$ Power down status negation (PDS Flag)	V_{SB}	0.0 3.0 —	5.5 5.5 3.5	V
15	RAM Standby Current ⁶ Normal RAM operation ^{9,10} $V_{DD} > V_{SB} - 0.5 \text{ V}$ Transient condition $V_{SB} - 0.5 \text{ V} \geq V_{DD} \geq V_{SS} + 0.5 \text{ V}$ Standby operation ⁷ $V_{DD} < V_{SS} + 0.5 \text{ V}$	I_{SB}	— — —	50 3 100	μA mA μA
16	Power Dissipation ⁸	P_D	—	770	mW
17	Input Capacitance ^{2,9} All input-only pins except ADC pins All input/output pins	C_{in}	— —	10 20	pF
18	Load Capacitance ² Group 1 I/O Pins and CLKOUT, FREEZE/QUOT, IPIPE0 Group 2 I/O Pins and CSBOOT, BG/CS Group 3 I/O pins Group 4 I/O pins	C_L	— — — —	90 100 130 200	pF

A

Table A-4. DC Characteristics (Continued)

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

NOTES:

- Parameter applies to the following pins:

Port ADA: PADA[7:0]/AN[7:0]

Port A: PA[7:0]/ADDR[18:11]

Port B: PB[7:0]/ADDR[10:3]

Port E: PE[7:6]/SIZ[1:0], PE5/AS, PE4/DS, PE3

Port F: PF[7:1]/IRQ[7:1], PF0/MODCLK

Port GP: PGP7/IC4/OC5, PGP[6:3]/OC[4:1], PGP[2:0]/IC[3:1]

Port G: PG[7:0]/DATA[15:8]

Port H: PH[7:0]/DATA[7:0]

Port MC: PMC7/TXDA, PMC6/RXDA, PMC5/TXDB, PMC4/RXDB, PMC3/SS, PMC2/SCK, PMC1/MOSI, PMC0/MISO

Other: BKPT/DSCLK, PAI, PCLK, RESET, T2CLK, TP[15:0], TSC

- Input-Only Pins: BKPT/DSCLK, EXTAL, PAI, PCLK, TSC

Output-Only Pins: ADDR[2:0], CSBOOT, BG/CS1, CLKOUT, FREEZE/QUOT, DSO/PIPE0, PWMA, PWMB

Input/Output Pins:

Group 1: Port GP: PGP7/IC4/OC5, PGP[6:3]/OC[4:1], PGP[2:0]/IC[3:1]

Port G: PG[7:0]/DATA[15:8]

Port H: PH[7:0]/DATA[7:0]

Other: DSI/PIPE1, TP[15:0]

Group 2: Port A: PA[7:0]/ADDR[18:11]

Port B: PB[7:0]/ADDR[10:3]

Port C: PC[6:3]/ADDR[22:19]/CS[9:6], PC2/FC2/CS5, PC1/FC, PC0/FC0/CS3

Port E: PE[7:6]/SIZ[1:0], PE5/AS, PE4/DS, PE3

Port F: PF[7:1]/IRQ[7:1], PF0/MODCLK

Port MC: PMC7/TXDA, PMC6/RXDA, PMC5/TXDB, PMC4/RXDB, PMC3/SS,

Other: ADDR23/CS10/ECLK, R/W, BERR, BR/CS0, BGACK/CS2

Group 3: HALT, RESET

Group 4: Port MC: PMC2/SCK, PMC1/MOSI, PMC0/MISO

- Does not apply to HALT, RESET (open-drain pins), and Port MC in wired-OR mode.

- Current measured with system clock frequency of 16.78 MHz, all modules active.

- Use of an active pulldown device is recommended.

- 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.

- The TPURAM module cannot switch into standby mode as long as V_{SB} does not exceed V_{DD} by more than 0.5 Volt. The TPURAM array cannot be accessed while the module is in standby mode.

- Power dissipation measured with system clock frequency of 16.78 MHz, all modules active. Specification does not apply to TPU emulation mode. Power dissipation can be calculated using the 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.

- This parameter is periodically sampled rather than 100% tested.

- 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.

A

Table A-5. AC Timing

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

Num	Characteristic	Symbol	Min	Max	Unit
F1	Frequency of Operation (4.194 MHz crystal) ²	f	0.13	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}	24	—	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	24	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} , $\overline{R/W}$ High Impedance	t_{CHSZ}	—	59	ns
17	\overline{AS} , \overline{DS} , \overline{CS} Negated to $\overline{R/W}$ High	t_{SNRN}	15	—	ns
18	Clock High to $\overline{R/W}$ High	t_{CHRH}	0	29	ns
20	Clock High to $\overline{R/W}$ Low	t_{CHRL}	0	29	ns
21	$\overline{R/W}$ High to \overline{AS} , \overline{CS} Asserted	t_{RAAA}	15	—	ns
22	$\overline{R/W}$ Low to \overline{DS} , \overline{CS} Asserted (Write)	t_{RASAA}	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} , \overline{HALT} Asserted to Clock Low (Setup Time)	t_{BELCL}	20	—	ns

A

Table A-5. AC Timing (Continued)
(V_{DD} and V_{DSDYN} = 5.0 Vdc ± 10%, V_{SS} = 0 Vdc, T_A = T_L to T_H)

Num	Characteristic	Symbol	Min	Max	Unit
28	\overline{AS} , \overline{DS} Negated to $\overline{DSACK}[1:0]$, \overline{BERR} , \overline{HALT} , \overline{AVEC} 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
31	$\overline{DSACK}[1:0]$ 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	$\overline{R\overline{W}}$ Width Asserted (Write or Read)	t _{RWA}	150	—	ns
46A	$\overline{R\overline{W}}$ Width Asserted (Fast Write or Read Cycle)	t _{RWAS}	90	—	ns
47A	Asynchronous Input Setup Time \overline{BR} , \overline{BGACK} , $\overline{DSACK}[1:0]$, \overline{BERR} , \overline{AVEC} , \overline{HALT}	t _{AISt}	5	—	ns
47B	Asynchronous Input Hold Time	t _{AIHT}	15	—	ns
48	$\overline{DSACK}[1:0]$ Asserted to \overline{BERR} , \overline{HALT} 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	$\overline{R\overline{W}}$ 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	t _{MSS}	20	—	t _{cyc}
76	Mode Select Hold Time	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} Negated 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

A

Table A–5. AC Timing (Continued) $(V_{DD}$ and $V_{DDSYN} = 5.0 \text{ Vdc} \pm 10\%$, $V_{SS} = 0 \text{ Vdc}$, $T_A = T_L$ to T_H)**NOTES:**

1. All AC timing is shown with respect to 20% V_{DD} and 70% V_{DD} levels unless otherwise noted.
2. Minimum system clock frequency is four times the crystal frequency, subject to specified limits.
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 signal varies. The relationship between external clock input duty cycle and minimum t_{Xcyc} is expressed:

$$\text{Minimum } t_{Xcyc} \text{ period} = \text{minimum } t_{XCHL} / (50\% - \text{external clock input duty cycle tolerance}).$$
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 AS and 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 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 $(t_{cyc} / 2) + 25 \text{ ns}$.
9. If the asynchronous setup time (specification 47A) requirements are satisfied, the $\overline{DSACK}[1:0]$ low to data setup time (specification 31) and $\overline{DSACK}[1:0]$ low to 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 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 $\overline{DSACK}[1:0]$, \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 SCIM 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 = $(2.5 + WS) t_{cyc} - t_{CHAV} - t_{D1CL}$
 Chip select access time = $(2 + WS) t_{cyc} - t_{CLSA} - t_{D1CL}$
 Where: WS = number of wait states. When fast termination is used (2 clock bus) WS = -1.



Table A-6. Background Debugging Mode Timing $(V_{DD} = 5.0 \text{ Vdc} \pm 10\%, V_{SS} = 0 \text{ Vdc}, T_A = T_L \text{ to } T_H)$

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

NOTES:

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

Table A-7. ECLK Bus Timing $(V_{DD} = 5.0 \text{ Vdc} \pm 10\%, V_{SS} = 0 \text{ Vdc}, T_A = T_L \text{ to } T_H)$

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}	10	—	ns
E3	ECLK Low to \overline{CS} Valid (\overline{CS} delay)	$t_{ECS D}$	—	150	ns
E4	ECLK Low to \overline{CS} Hold	$t_{ECS H}$	15	—	ns
E5	\overline{CS} Negated Width	$t_{ECS N}$	30	—	ns
E6	Read Data Setup Time	t_{EDSR}	30	—	ns
E7	Read Data Hold Time	t_{EDHR}	15	—	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}	5	—	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}

NOTES:

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

A

Table A-8. MCCI SPI Timing

($V_{DD} = 5.0 \text{ Vdc} \pm 10\%$, $V_{SS} = 0 \text{ Vdc}$, $T_A = T_L$ to T_H , 200 pF load on all QSPI pins)

Num	Parameter	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_{cyc}	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 Output	t_{fi} t_{fo}	— —	2 30	μs ns

NOTES:

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; for low time, n = External SCK fall time



Table A-9. ADC Maximum Ratings

Num	Parameter	Symbol	Min	Max	Unit
1	Analog Supply	V_{DDA}	-0.3	6.5	V
2	Internal Digital Supply	V_{DDI}	-0.3	6.5	V
3	Reference Supply	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_{REF} to V_{DDA} Differential Voltage	$V_{RH} - V_{DDA}$	-6.5	6.5	V
8	Disruptive Input Current ^{1, 2, 3, 4, 5, 6} $V_{NEGCLAMP} \cong -0.3$ V $V_{POSCLAMP} \cong V_{DDA} + 2$	I_{NA}	-44	44	μ A
9	Maximum Input Current ^{3, 4, 6} $V_{NEGCLAMP} \cong -0.3$ V $V_{POSCLAMP} \cong V_{DDA} + 2$	I_{MA}	-500	500	μ A

NOTES:

1. Below disruptive current conditions, the channel being stressed will have conversion values of \$3FF for analog inputs greater than V_{RH} and \$000 for values 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.

A

Table A-10. ADC DC Electrical Characteristics (Operating)(V_{SS} = 0 Vdc, ADCLK = 2.1 MHz, T_A within operating temperature range)

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, 5}	V _R L	V _{SSA}	V _{DDA} / 2	V
6	Reference Voltage High ^{2, 5}	V _R H	V _{DDA} / 2	V _{DDA}	V
7	V _{REF} Differential Voltage ⁵	V _R H - V _R L	4.5	5.5	V
8	Input Voltage ²	V _I NC	V _{SSA}	V _{DDA}	V
9	Input High, Port ADA	V _I H	0.7 (V _{DDA})	V _{DDA} + 0.3	V
10	Input Low, Port ADA	V _I L	V _{SSA} - 0.3	0.2 (V _{DDA})	V
15	Analog Supply Current Normal Operation ³ Low-power stop	I _{DDA}	— —	1.0 TBD	mA μA
16	Reference Supply Current	I _{REF}	—	250	μA
17	Input Current, Off Channel ⁴	I _{OFF}	—	150	nA
18	Total Input Capacitance, Not Sampling	C _I NN	—	10	pF
19	Total Input Capacitance, Sampling	C _I NS	—	15	pF

NOTES:

- Refers to operation over full temperature and frequency range.
- To obtain full-scale, full-range results, V_{SSA} ≤ V_RL ≤ V_INC ≤ V_RH ≤ V_{DDA}.
- Current measured at maximum system clock frequency with ADC active.
- Maximum leakage occurs at maximum operating temperature. Current decreases by approximately one-half for each 10° C decrease from maximum temperature.
- Accuracy tested and guaranteed at V_RH - V_RL ≤ 5.0 V ± 10%.

A

Table A–11. ADC AC Characteristics (Operating)

(V_{DD} and $V_{DDA} = 5.0 \text{ Vdc} \pm 10\%$, $V_{SS} = 0 \text{ Vdc}$, T_A within operating temperature range)

Num	Parameter	Symbol	Min	Max	Unit
1	On-chip PLL Frequency	f_{sys}	2.0	16.78	MHz
2	ADC Clock Frequency	F_{ADCLK}	0.5	2.1	MHz
3	8-bit Conversion Time (16 ADC Clocks) ¹	T_{CONV}	7.62	—	μs
4	10-bit Conversion Time (18 ADC Clocks) ¹	T_{CONV}	8.58	—	μs
5	Stop Recovery Time	T_{SR}	—	10	μs

NOTES:

1. Assumes 2.1 MHz ADC clock and selection of minimum sample time (2 ADC clocks).

Table A–12. ADC Conversion Characteristics (Operating)

(V_{DD} and $V_{DDA} = 5.0 \text{ Vdc} \pm 10\%$, $V_{SS} = 0 \text{ Vdc}$, $T_A = T_L$ to T_H , $\text{ADCLK} = 2.1 \text{ MHz}$)

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	DNL	–0.5	—	0.5	Counts
7	10-bit Integral Nonlinearity	INL	–2.0	—	2.0	Counts
8	10-bit Absolute Error ³	AE	–2.5	—	2.5	Counts
9	Source Impedance at Input ⁴	R_S	—	20	—	$\text{k}\Omega$

NOTES:

1. At $V_{RH} - V_{RL} = 5.12 \text{ V}$, 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. 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.
4. 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. In the following expression, expected error in result value due to junction leakage is expressed in voltage (V_{errj}):

$$V_{\text{errj}} = R_S \times I_{\text{OFF}}$$

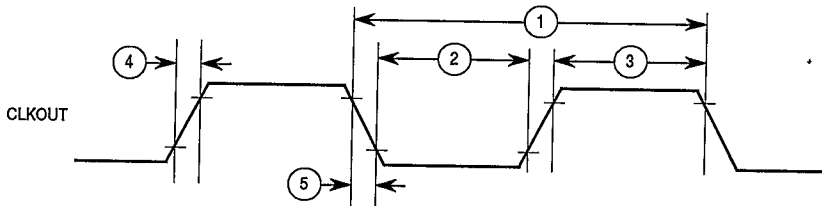
where I_{OFF} is a function of operating temperature. (See Table A–10, note 4).

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

Timing Diagrams

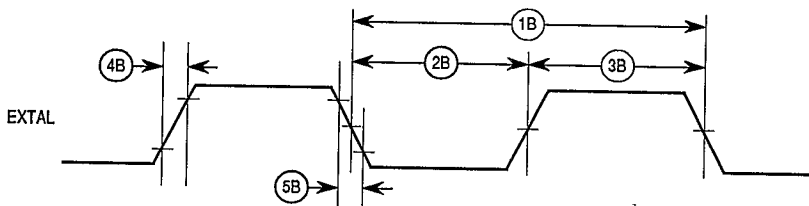
A



16 CLKOUT TIM

NOTE: Timing shown with respect to 20% and 70% V_{DD} .

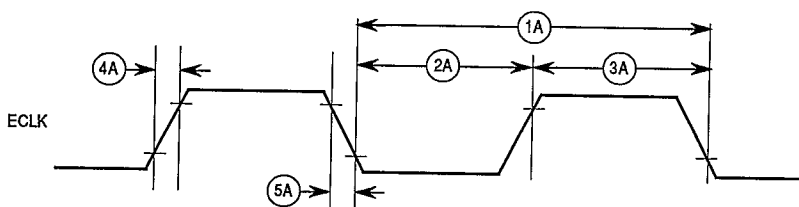
Figure A-1. CLKOUT Output Timing Diagram



18 EXT CLK INPUT TIM

NOTE: Timing shown with respect to 20% and 70% V_{DD} . Pulse width shown with respect to 50% V_{DD} .

Figure A-2. External Clock Input Timing Diagram



16 ECLK OUTPUT TIM

NOTE: Timing shown with respect to 20% and 70% V_{DD} .

Figure A-3. ECLK Output Timing Diagram

A

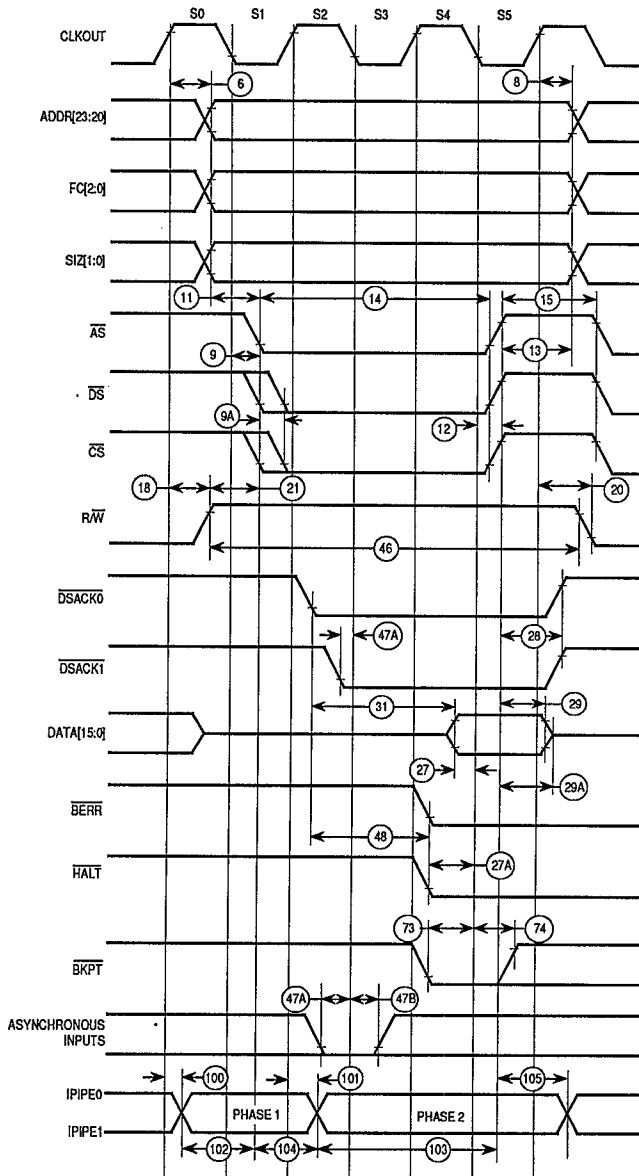
Key to Figures A-1, A-2, A-3
(Abstracted from Table A-5; see table for complete notes)

Num	Characteristic	Symbol	Min	Max	Units
1	Clock Period	t_{cyc}	59.6	—	ns
1A	ECLK Period	$t_{E_{cyc}}$	476	—	ns
1B	External Clock Input Period ³	$t_{X_{cyc}}$	59.6	—	ns
2, 3	Clock Pulse Width	t_{CW}	24	—	ns
2A, 3A	ECLK Pulse Width	t_{ECW}	236	—	ns
2B,3B	External Clock Input High/Low Time ³	$t_{X_{CHL}}$	29.8	—	ns
4, 5	CLOCKOUT 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 Rise and Fall Time ⁴	t_{XCrf}	—	5	ns

NOTES:

- All AC timing is shown with respect to 20% V_{DD} and 70% V_{DD} levels unless otherwise noted.
- When an external clock is used, minimum high and low times are based on a 50% duty cycle. The minimum allowable $t_{X_{cyc}}$ period is reduced when the duty cycle of the external clock signal varies. The relationship between external clock input duty cycle and minimum $t_{X_{cyc}}$ is expressed:
 Minimum $t_{X_{cyc}}$ period = minimum $t_{X_{CHL}}$ / (50% – external clock input duty cycle tolerance).
- 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.





16 RD CYC TIM

Figure A-4. Read Cycle Timing Diagram

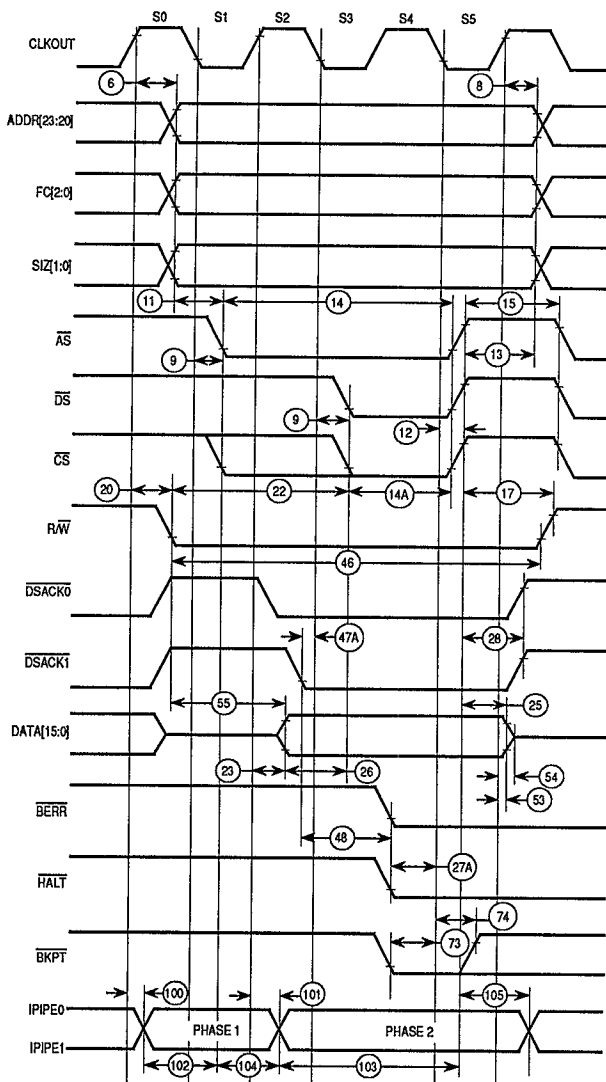
A

Key to Figure A-4
(Abstracted from Table A-5; see table for complete notes)

Num	Characteristic	Symbol	Min	Max	Units
6	Clock High to ADDR, FC, SIZE Valid	tCHAV	0	29	ns
8	Clock High to ADDR, FC, SIZE Invalid	tCHAZn	0	—	ns
9	Clock Low to AS, DS, CS Asserted	tCLSA	2	24	ns
9A	AS to DS or CS Asserted (Read) ⁵	tSTSA	-15	15	ns
11	ADDR, FC, SIZE Valid to AS, CS (and DS Read) Asserted	tAVSA	15	—	ns
12	Clock Low to AS, DS, CS Negated	tCLSN	2	29	ns
13	AS, DS, CS Negated to ADDR, FC, SIZE Invalid (Address Hold)	tSNAI	15	—	ns
14	AS, CS (and DS Read) Width Asserted	tSWA	100	—	ns
15	AS, DS, CS Width Negated ⁶	tSN	40	—	ns
18	Clock High to R/W High	tCHRH	0	29	ns
20	Clock High to R/W Low	tCHRL	0	29	ns
21	R/W High to AS, CS Asserted	tRAAA	15	—	ns
27	Data In Valid to Clock Low (Data Setup)	tDIDL	5	—	ns
27A	Late BERR, HALT Asserted to Clock Low (Setup Time)	tBELCL	20	—	ns
28	AS, DS Negated to DSACK[1:0], BERR, HALT, AVEC Negated	tSNDN	0	80	ns
29	DS, CS Negated to Data In Invalid (Data In Hold) ⁷	tSNDI	0	—	ns
29A	DS, CS Negated to Data In High-Impedance ^{7,8}	tSHDI	—	55	ns
31	DSACK[1:0] Asserted to Data In Valid ⁹	tDADI	—	50	ns
46	R/W Width Asserted (Write or Read)	tRWA	150	—	ns
47A	Asynchronous Input Setup Time BR, BG, DSACK[1:0], BERR, AVEC, HALT	tAIST	5	—	ns
47B	Asynchronous Input Hold Time	tAIHT	15	—	ns
48	DSACK[1:0] Asserted to BERR, HALT Asserted ¹¹	tDABA	—	30	ns
73	BKPT Input Setup Time	tBKST	15	—	ns
74	BKPT Input Hold Time	tBKHT	10	—	ns
100	CLKOUT High to Phase 1 Asserted ¹⁴	tCHP1A	3	40	ns
101	CLKOUT High to Phase 2 Asserted ¹⁴	tCHP2A	3	40	ns
102	Phase 1 Valid to AS or DS Asserted ¹⁴	tP1VSA	10	—	ns
103	Phase 2 Valid to AS or DS Negated ¹⁴	tP2VSN	10	—	ns
104	AS or DS Valid to Phase 1 Negated ¹⁴	tSAP1N	10	—	ns
105	AS or DS Negated to Phase 2 Negated ¹⁴	tSNP2N	10	—	ns

NOTES:

1. All AC timing is shown with respect to 20% V_{DD} and 70% V_{DD} levels unless otherwise noted.
5. Specification 9A is the worst-case skew between AS and DS or CS. The amount of skew depends on the relative loading of these signals. When loads are kept within specified limits, skew will not cause AS and DS to fall outside the limits shown in specification 9.
6. If multiple chip selects are used, 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 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 DS or CS on asynchronous reads and with respect to CLKOUT on fast 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 DSACK[1:0] low to data setup time (specification 31) and DSACK[1:0] low to 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. BERR must satisfy only the late BERR low to clock low setup time (specification 27A) for the following clock cycle.
11. In the absence of DSACK[1:0], BERR is an asynchronous input — use specification 47A.
14. Eight pipeline states are multiplexed into IPIPE[1:0]. The multiplexed signals have two phases.



18 WR CYC TIM

Figure A-5. Write Cycle Timing Diagram

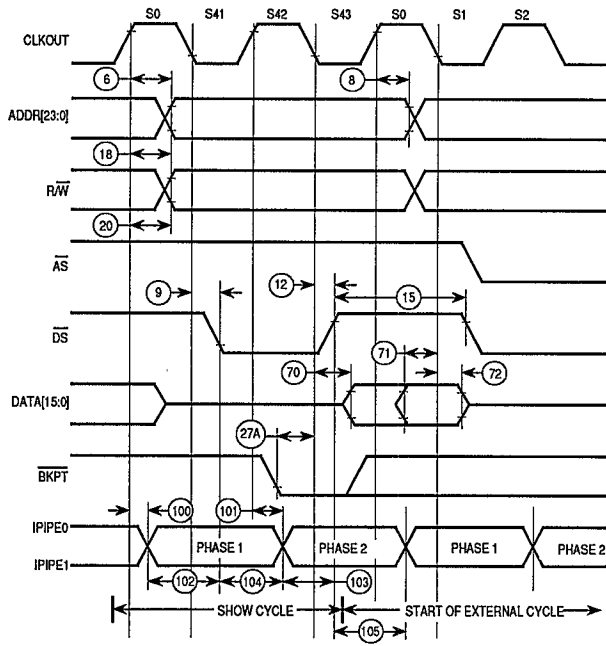
A

Key to Figure A-5
(Abstracted from Table A-5; see table for complete notes)

Num	Characteristic	Symbol	Min	Max	Units
6	Clock High to ADDR, FC, SIZE Valid	t _{CHAV}	0	29	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	24	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
15	\overline{AS} , \overline{DS} , \overline{CS} Width Negated ⁶	t _{SN}	40	—	ns
17	\overline{AS} , \overline{DS} , \overline{CS} Negated to R/W High	t _{SNRN}	15	—	ns
20	Clock High to R/W Low	t _{CHRL}	0	29	ns
22	R/W Low to \overline{DS} , \overline{CS} Asserted (Write)	t _{PRASA}	70	—	ns
23	Clock High to Data Out Valid	t _{CHDO}	—	29	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
27A	Late BERR, HALT Asserted to Clock Low (Setup Time)	t _{BELCL}	20	—	ns
28	\overline{AS} , \overline{DS} Negated to DSACK[1:0], BERR, HALT, AVEC Negated	t _{SNDN}	0	80	ns
46	R/W Width Asserted (Write or Read)	t _{RWA}	150	—	ns
47A	Asynchronous Input Setup Time BR, BG, DSACK[1:0], BERR, AVEC, HALT	t _{AIST}	5	—	ns
48	DSACK[1:0] Asserted to BERR, HALT 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
73	BKPT Input Setup Time	t _{BKST}	15	—	ns
74	BKPT Input Hold Time	t _{BKHT}	10	—	ns
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} Negated 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

NOTES:

1. All AC timing is shown with respect to 20% V_{DD} and 70% V_{DD} levels unless otherwise noted.
6. If multiple chip selects are used, 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 CS width negated specification between multiple chip selects does not apply to chip selects being used for synchronous ECLK cycles.
11. In the absence of DSACK[1:0], BERR is an asynchronous input — use specification 47A.
14. Eight pipeline states are multiplexed into IPIPE[1:0]. The multiplexed signals have two phases.



16 SHW CYC TIM

Figure A-6. Show Cycle Timing Diagram

A

Key to Figure A-6
(Abstracted from Table A-5; see table for complete notes)

Num	Characteristic	Symbol	Min	Max	Units
6	Clock High to ADDR, FC, SIZE Valid	t _{CHAV}	0	29	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	24	ns
12	Clock Low to \overline{AS} , \overline{DS} , \overline{CS} Negated	t _{CLSN}	2	29	ns
15	\overline{AS} , \overline{DS} , \overline{CS} Width Negated ⁶	t _{SN}	40	—	ns
18	Clock High to R/W High	t _{CHRH}	0	29	ns
20	Clock High to R/W Low	t _{CHRL}	0	29	ns
70	Clock Low to Data Bus Driven (Show)	t _{SCLDD}	0	29	ns
71	Data Setup Time to Clock Low (Show)	t _{SCLDS}	15	—	ns
72	Data Hold from Clock Low (Show)	t _{SCLDH}	10	—	ns
73	\overline{BKPT} Input Setup Time	t _{BKST}	15	—	ns
74	\overline{BKPT} Input Hold Time	t _{BKHT}	10	—	ns
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} Negated 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

NOTES:

- All AC timing is shown with respect to 20% V_{DD} and 70% V_{DD} levels unless otherwise noted.
- 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.
- Eight pipeline states are multiplexed into IPIPE[1:0]. The multiplexed signals have two phases.



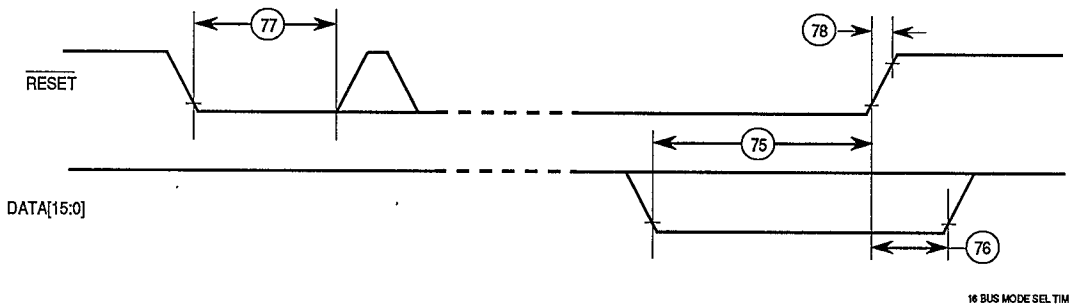


Figure A-7. Reset and Mode Select Timing Diagram

A

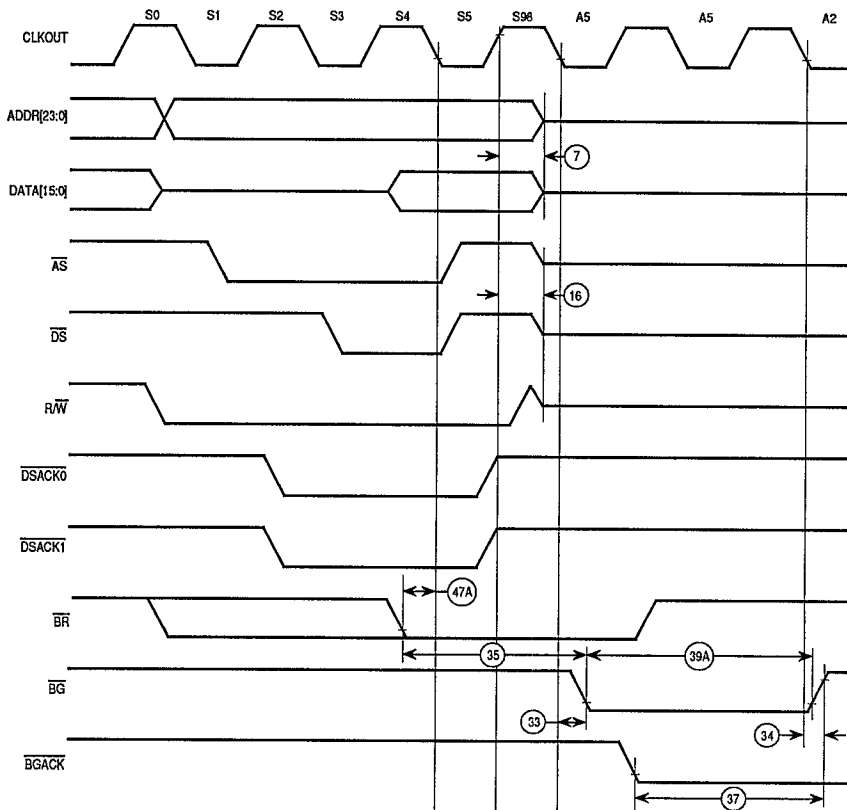
Key to Figure A-7

(Abstracted from Table A-5; see table for complete notes)

Num	Characteristic	Symbol	Min	Max	Units
75	Mode Select Setup Time	t_{MSS}	20	—	t_{cyc}
76	Mode Select Hold Time	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}

NOTES:

1. All AC timing is shown with respect to 20% V_{DD} and 70% V_{DD} levels unless otherwise noted.
12. After external \overline{RESET} negation is detected, a short transition period (approximately $2 t_{cyc}$) elapses, then the SCIM 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.



16 BUS ARB TIM

Figure A-8. Bus Arbitration Timing Diagram — Active Bus Case

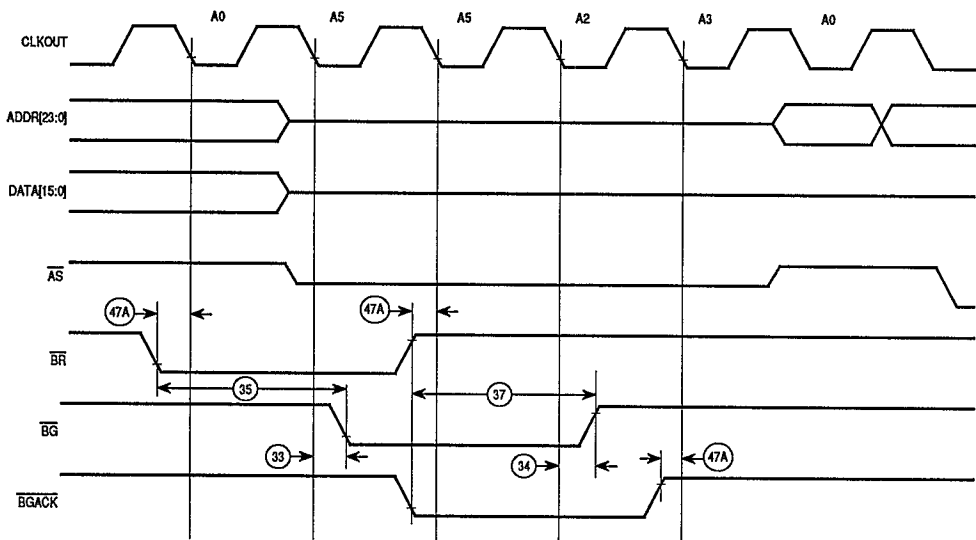
Key to Figure A-8
(Abstracted from Table A-5; see table for complete notes)

Num	Characteristic	Symbol	Min	Max	Units
7	Clock High to ADDR, Data, FC, SIZE High Impedance	t _{CHAZx}	0	59	ns
16	Clock High to \overline{AS} , \overline{DS} , \overline{RW} High Impedance	t _{CHSZ}	—	59	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}
39A	\overline{BG} Width Asserted	t _{GA}	1	—	t _{cyc}
47A	Asynchronous Input Setup Time \overline{BR} , \overline{BG} , $\overline{DSACK}[1:0]$, \overline{BERR} , \overline{AVEC} , \overline{HALT}	t _{AIST}	5	—	ns

NOTES:

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

A



16 BUS ARB TIM IDLE

Figure A-9. Bus Arbitration Timing Diagram — Idle Bus Case

A

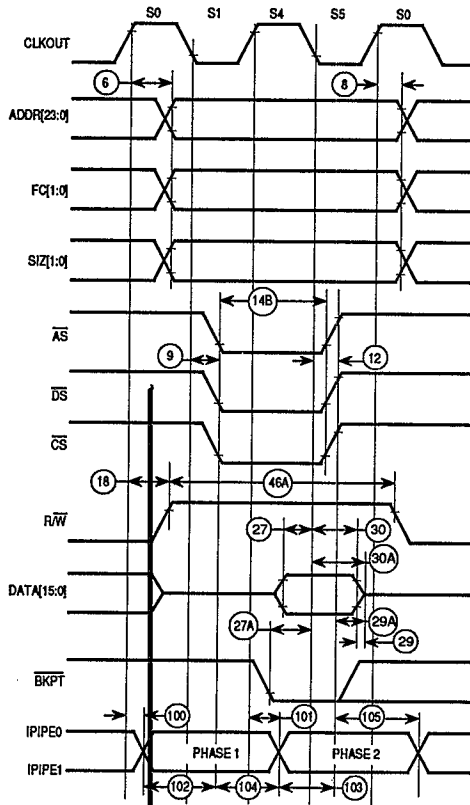
Key to Figure A-9
(Abstracted from Table A-5; see table for complete notes)

Num	Characteristic	Symbol	Min	Max	Units
33	Clock Low to $\overline{\text{BG}}$ Asserted/Negated	t_{CLBAN}	—	29	ns
35	$\overline{\text{BR}}$ Asserted to $\overline{\text{BG}}$ Asserted ¹⁰	t_{BRAGA}	1	—	t_{cyc}
37	$\overline{\text{BGACK}}$ Asserted to $\overline{\text{BG}}$ Negated	t_{GAGN}	1	2	t_{cyc}
47A	Asynchronous Input Setup Time $\overline{\text{BR}}$, $\overline{\text{BG}}$, $\overline{\text{DSACK}}[1:0]$, $\overline{\text{BERR}}$, $\overline{\text{AVEC}}$, $\overline{\text{HALT}}$	t_{AIST}	5	—	ns

NOTES:

- All AC timing is shown with respect to 20% V_{DD} and 70% V_{DD} levels unless otherwise noted.
- To ensure coherency during every operand transfer, $\overline{\text{BG}}$ is not asserted in response to $\overline{\text{BR}}$ until after all cycles of the current operand transfer are complete.

A



A

Figure A-10. Fast Termination Read Cycle Timing Diagram

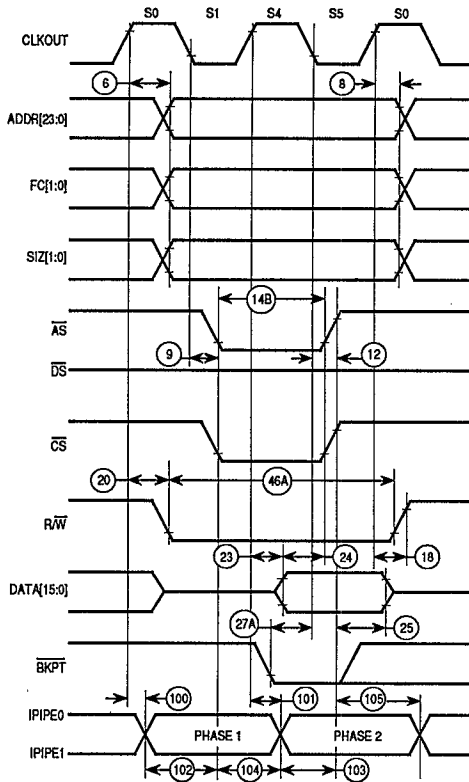
Key to Figure A-10
(Abstracted from Table A-5; see table for complete notes)

Num	Characteristic	Symbol	Min	Max	Units
6	Clock High to ADDR, FC, SIZE Valid	t _{CHAV}	0	29	ns
8	Clock High to ADDR, FC, SIZE Invalid	t _{CHAZ_n}	0	—	ns
9	Clock Low to \overline{AS} , \overline{DS} , \overline{CS} Asserted	t _{CLSA}	2	24	ns
12	Clock Low to \overline{AS} , \overline{DS} , \overline{CS} Negated	t _{CLSN}	2	29	ns
14B	\overline{AS} , \overline{CS} (and \overline{DS} Read) Width Asserted (Fast Cycle)	t _{SWDW}	40	—	ns
18	Clock High to $\overline{R/\overline{W}}$ High	t _{CHRH}	0	29	ns
20	Clock High to $\overline{R/\overline{W}}$ Low	t _{CHRL}	0	29	ns
27	Data In Valid to Clock Low (Data Setup)	t _{DICL}	5	—	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 ⁷	CLKOUT Low to Data In Invalid (Fast Hold)	t _{CLDI}	15	—	ns
30A ⁷	CLKOUT Low to Data In High Impedance	t _{CLDH}	—	90	ns
46A	$\overline{R/\overline{W}}$ Width Asserted (Fast Write or Read)	t _{RWAS}	90	—	ns
73	BKPT Input Setup Time	t _{BKST}	15	—	ns
74	BKPT Input Hold Time	t _{BKHT}	10	—	ns
100	CLKOUT High to Phase 1 Asserted	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} Negated 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

NOTES:

- All AC timing is shown with respect to 20% V_{DD} and 70% V_{DD} levels unless otherwise noted.
- 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.
- Maximum value is equal to (t_{cyc} / 2) + 25 ns.
- Eight pipeline states are multiplexed into IPIPE[1:0]. The multiplexed signals have two phases.





16 FAST WR CYC TIM

A

Figure A-11. Fast Termination Write Cycle Timing Diagram

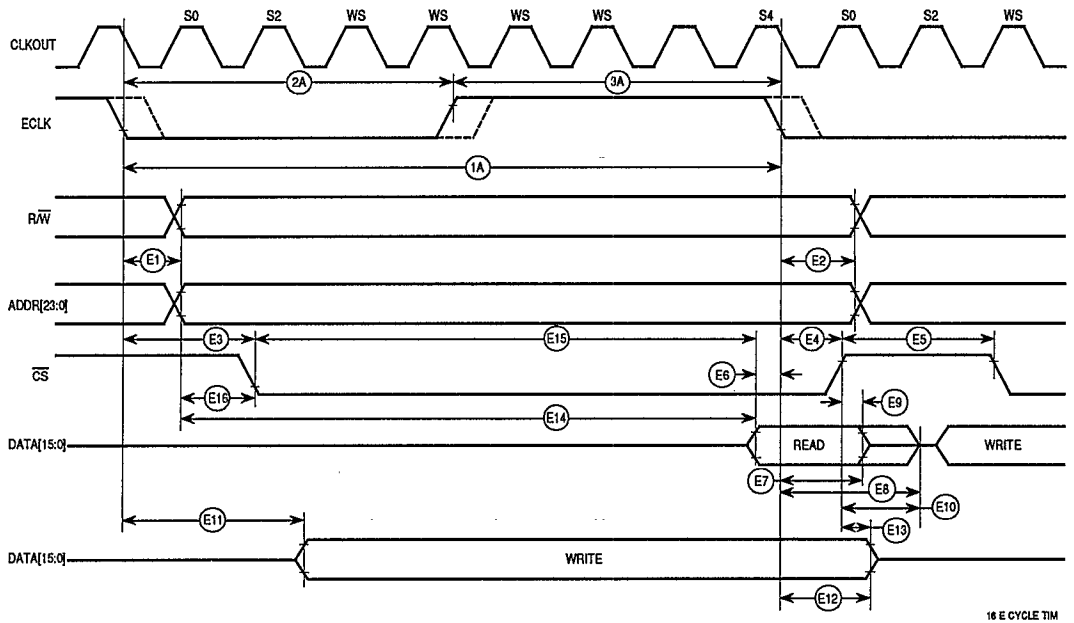
Key to Figure A-11
(Abstracted from Table A-5; see table for complete notes)

Num	Characteristic	Symbol	Min	Max	Units
6	Clock High to ADDR, FC, SIZE Valid	tCHAV	0	29	ns
8	Clock High to ADDR, FC, SIZE Invalid	tCHAZn	0	—	ns
9	Clock Low to \overline{AS} , \overline{DS} , \overline{CS} Asserted	tCLSA	2	24	ns
12	Clock Low to \overline{AS} , \overline{DS} , \overline{CS} Negated	tCLSN	2	29	ns
14B	\overline{AS} , \overline{CS} (and \overline{DS} Read) Width Asserted (Fast Cycle)	tSWDW	40	—	ns
18	Clock High to R/W High	tCHRH	0	29	ns
20	Clock High to R/W Low	tCHRL	0	29	ns
23	Clock High to Data Out Valid	tCHDO	—	29	ns
24	Data Out Valid to Negating Edge of \overline{AS} , \overline{CS} (Fast Write)	tDVASN	15	—	ns
25	\overline{DS} , \overline{CS} Negated to Data Out Invalid (Data Out Hold)	tSNDOI	15	—	ns
46A	R/W Width Asserted (Fast Write or Read)	tRWAS	90	—	ns
73	BKPT Input Setup Time	tBKST	15	—	ns
74	BKPT Input Hold Time	tBKHT	10	—	ns
100	CLKOUT High to Phase 1 Asserted ¹⁴	tCHP1A	3	40	ns
101	CLKOUT High to Phase 2 Asserted ¹⁴	tCHP2A	3	40	ns
102	Phase 1 Valid to \overline{AS} or \overline{DS} Asserted ¹⁴	tP1VSA	10	—	ns
103	Phase 2 Valid to \overline{AS} or \overline{DS} Negated ¹⁴	tP2VSN	10	—	ns
104	\overline{AS} or \overline{DS} Valid to Phase 1 Negated ¹⁴	tSAP1N	10	—	ns
105	\overline{AS} or \overline{DS} Negated to Phase 2 Negated ¹⁴	tSNP2N	10	—	ns

NOTES:

1. All AC timing is shown with respect to 20% V_{DD} and 70% V_{DD} levels unless otherwise noted.
14. Eight pipeline states are multiplexed into IPIPE[1:0]. The multiplexed signals have two phases.

A



NOTE: Shown with ECLK = system clock/8 — EDIV bit in clock synthesizer control register (SYNCR) = 0.

Figure A-12. ECLK Timing Diagram

A

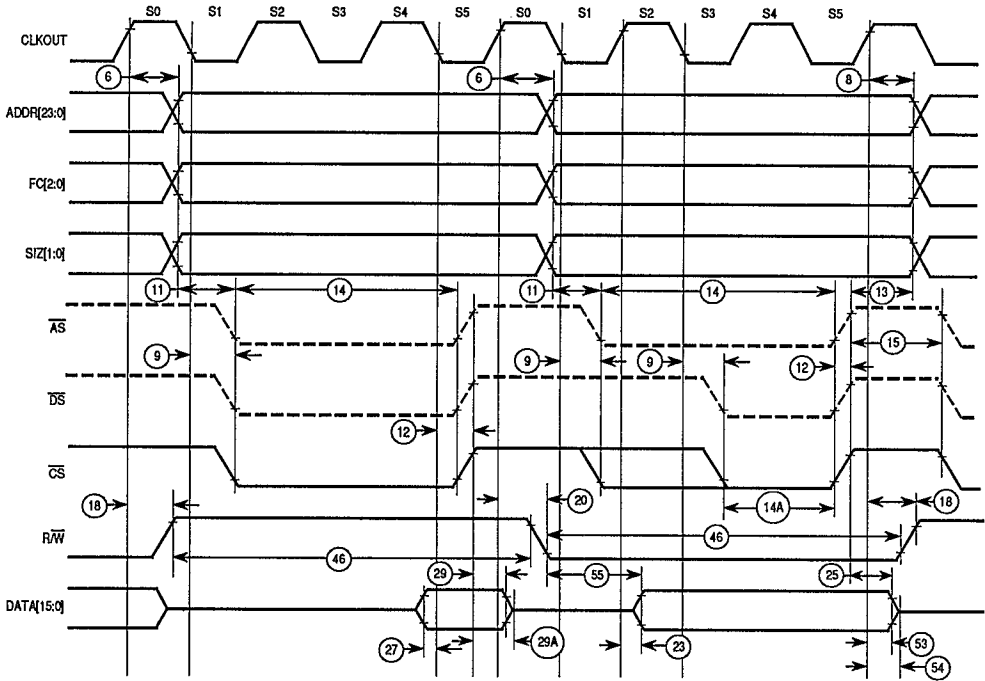
Key to Figure A-12

(Abstracted from Tables A-5 and A-7; see tables for complete notes)

Num	Characteristic	Symbol	Min	Max	Units
1A	ECLK Period	$t_{E_{cyc}}$	476	—	ns
2A, 3A	ECLK Pulse Width	$t_{E_{CW}}$	236	—	ns
4A, 5A	Rise and Fall Time	t_{rf}	—	8	ns
E1	ECLK Low to ADDR and R/W Valid ²	t_{EAD}	—	60	ns
E2	ECLK Low to ADDR and R/W Hold	t_{EAH}	10	—	ns
E3	ECLK Low to CS Valid (CS delay)	$t_{ECS D}$	—	150	ns
E4	ECLK Low to CS Hold	$t_{ECS H}$	15	—	ns
E5	CS Negated Width	$t_{ECS N}$	30	—	ns
E6	Read Data Setup Time	t_{EDSR}	30	—	ns
E7	Read Data Hold Time	t_{EDHR}	15	—	ns
E8	ECLK Low to Data High Impedance	t_{EDHZ}	—	60	ns
E9	CS Negated to Data Hold (Read)	t_{ECDH}	0	—	ns
E10	CS Negated to Data High Impedance	t_{ECDZ}	—	1	t_{cyc}
E11	ECLK Low to Data Valid (Write)	t_{EDDW}	—	2	t_{cyc}
E12	ECLK Low to Data Hold (Write)	t_{EDHW}	5	—	ns
E13	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}

NOTES:

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



16 CHIP SEL TIM

NOTE: \overline{AS} and \overline{DS} timing shown for reference only.

Figure A-13. Chip Select Timing Diagram

A

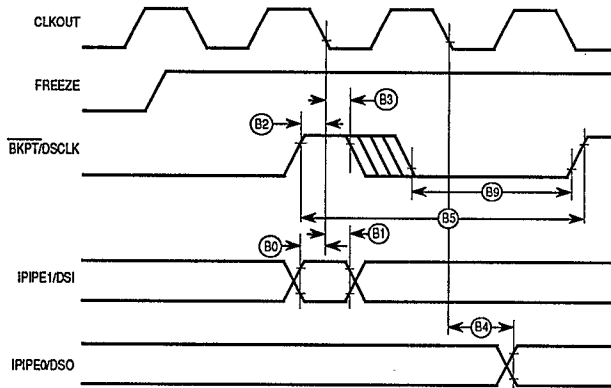
Key to Figure A-13

(Abstracted from Table A-5; see table for complete notes)

Num	Characteristic	Symbol	Min	Max	Units
6	Clock High to ADDR, FC, SIZE Valid	t _{CHAV}	0	29	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	24	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
15	\overline{AS} , \overline{DS} , \overline{CS} Width Negated ⁶	t _{SN}	40	—	ns
18	Clock High to $\overline{R/W}$ High	t _{CHRH}	0	29	ns
20	Clock High to $\overline{R/W}$ Low	t _{CHRL}	0	29	ns
23	Clock High to Data Out Valid	t _{CHDO}	—	29	ns
25	\overline{DS} , \overline{CS} Negated to Data Out Invalid (Data Out Hold)	t _{SNDIO}	15	—	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
46	$\overline{R/W}$ Width Asserted (Write or Read)	t _{RWA}	150	—	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	$\overline{R/W}$ Asserted to Data Bus Impedance Change	t _{RADC}	40	—	ns

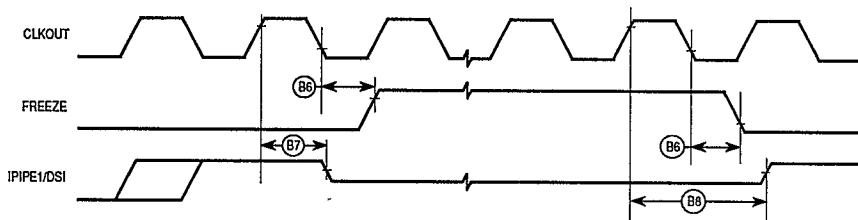
NOTES:

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



16 BDM SERZ TIM

Figure A-14. Background Debugging Mode Timing Diagram — Serial Communication



16 BDM FRZ TIM

Figure A-15. Background Debugging Mode Timing Diagram — Freeze Assertion

A

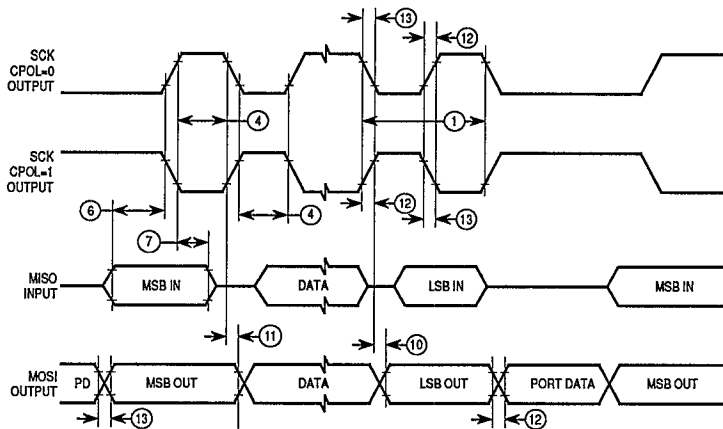
Key to Figures A-14 and A-15
(Abstracted from Table A-6; see table for complete notes)

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}

NOTES:

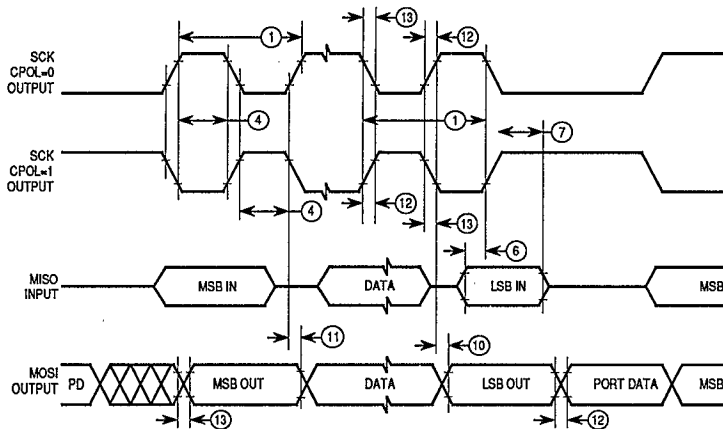
1. All AC timing is shown with respect to 20% V_{DD} and 70% V_{DD} levels unless otherwise noted.





18 MCC1 MAST CPHA0

Figure A-16. SPI Timing Master, CPHA = 0



16 MCC1 MAST CPHA1

Figure A-17. SPI Timing Master, CPHA = 1

A

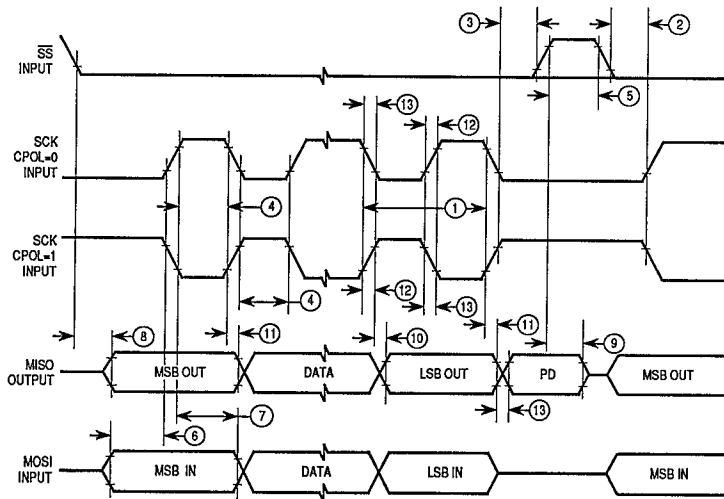
Key to Figures A-16 and A-17
(Abstracted from Table A-8; see table for complete notes)

Num	Function	Symbol	Min	Max	Unit
1	Master Cycle Time	t_{qyc}	4	510	t_{cyc}
4	Master Clock (SCK) High or Low Time	t_{sw}	$2 t_{cyc}-60$	$255 t_{cyc}$	ns
5	Master Sequential Transfer Delay	t_{td}	17	8192	t_{cyc}
6	Master Data Setup Time (Inputs)	t_{su}	30	—	ns
7	Master Data Hold Time (Inputs)	t_{hi}	0	—	ns
10	Master Data Valid (after SCK Edge)	t_v	—	50	ns
11	Master Data Hold Time (Outputs)	t_{ho}	0	—	ns
12	Rise Time Input Output	t_{ri} t_{ro}	— —	2 30	μs ns
13	Fall Time Input Output	t_{fi} t_{fo}	— —	2 30	μs ns

NOTES:

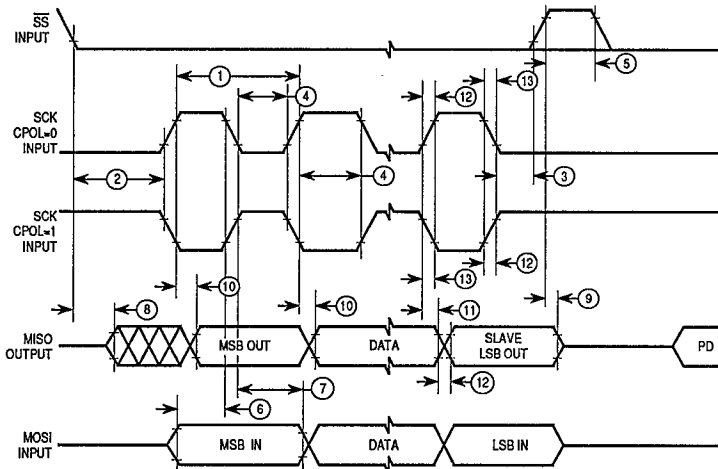
1. All AC timing is shown with respect to 20% V_{DD} and 70% V_{DD} levels unless otherwise noted.

A



16 MCC1 SLV CPHA0

Figure A-18. SPI Timing Slave, CPHA = 0



16 MCC1 SLV CPHA1

Figure A-19. SPI Timing Slave, CPHA = 1

A

Key to Figures A-18 and A-19
(Abstracted from Table A-8; see table for complete notes)

Num	Function	Symbol	Min	Max	Unit
1	Slave Cycle Time	t_{cyc}	4	—	t_{cyc}
2	Slave Enable Lead Time	t_{lead}	2	—	t_{cyc}
3	Slave Enable Lag Time	t_{lag}	2	—	t_{cyc}
4	Slave Clock (SCK) High or Low Time ²	t_{sw}	$2 t_{cyc} - n$	—	ns
5	Slave Sequential Transfer Delay (Does Not Require Deselect)	t_{td}	13	—	t_{cyc}
6	Slave Data Setup Time (Inputs)	t_{su}	20	—	ns
7	Slave Data Hold Time (Inputs)	t_{hi}	20	—	ns
8	Slave Access Time	t_a	—	1	t_{cyc}
9	Slave MISO Disable Time	t_{dis}	—	2	t_{cyc}
10	Slave Data Valid (after SCK Edge)	t_v	—	50	ns
11	Slave Data Hold Time (Outputs)	t_{ho}	0	—	ns
12	Rise Time Input Output	t_{ri} t_{ro}	— —	2 30	μs ns
13	Fall Time Input Output	t_{fi} t_{fo}	— —	2 30	μs ns

NOTES:

1. All AC timing is shown with respect to 20% V_{DD} and 70% V_{DD} levels unless otherwise noted.
2. In formula, n = External SCK rise + External SCK fall time



A

APPENDIX B MECHANICAL DATA AND ORDERING INFORMATION

This appendix provides package pin assignment drawings, dimensional drawings, and ordering information for standard and custom parts.

B.1 Pin Assignments and Package Dimensions

The MC68HC16Y1 is available in either a 160-pin quad flat package, or a 160-pin plastic surface mount package in a molded carrier ring. Figure B-1 shows pin assignments. Figures B-2 and B-3 are dimensional drawings.

B.2 Standard Part Ordering Information

Table B-1 is a matrix of standard part ordering numbers. Temperature ranges, packaging material options, and shipping quantity options are shown. Contact a Motorola sales representative for pricing and availability information.

B.3 Custom MCU Ordering Information

The MC68HC16Y1 is available as a special-order part with a number of options. In addition to application-specific masked ROM code and masked ROM module operating modes, the MCU can be ordered with an alternate system clock reference frequency and can also be programmed with custom TPU microcode.

To have Motorola manufacture an MCU with application-specific masked ROM code or TPU microcode, the customer must first develop and debug the code, then submit an MCU order along with a copy of the application program. For clock-speed options, only the ordering form is necessary.

Paragraphs following Table B-1 provide information concerning the process of ordering and manufacturing an MCU with customer-specified options. The information is a guide to requirements, and is intended to help a customer prepare to make a custom-ROM order. Contact a Motorola representative for additional information and assistance.

B

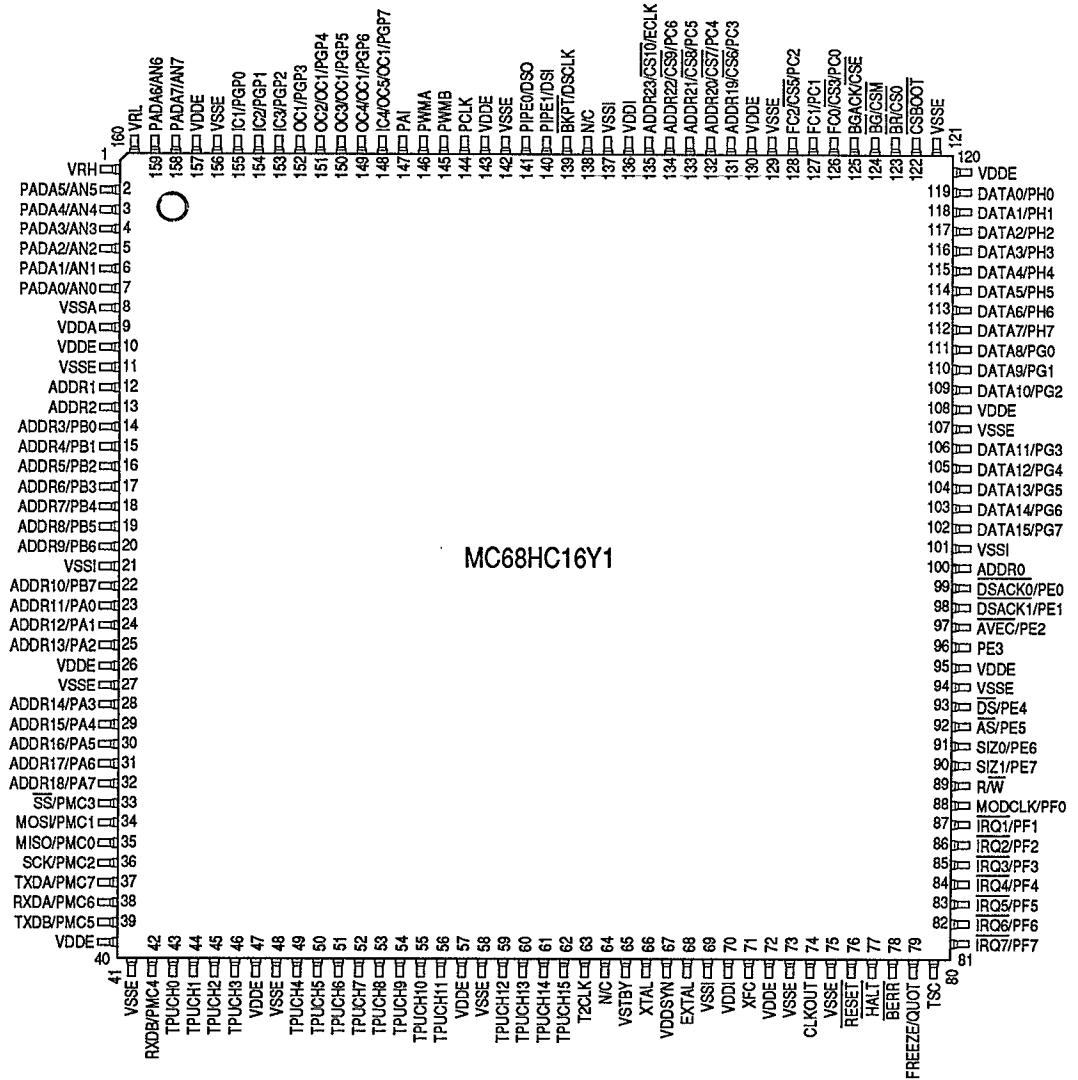


Figure B-1. 160-Pin Plastic Quad Flat Package Pin Assignments

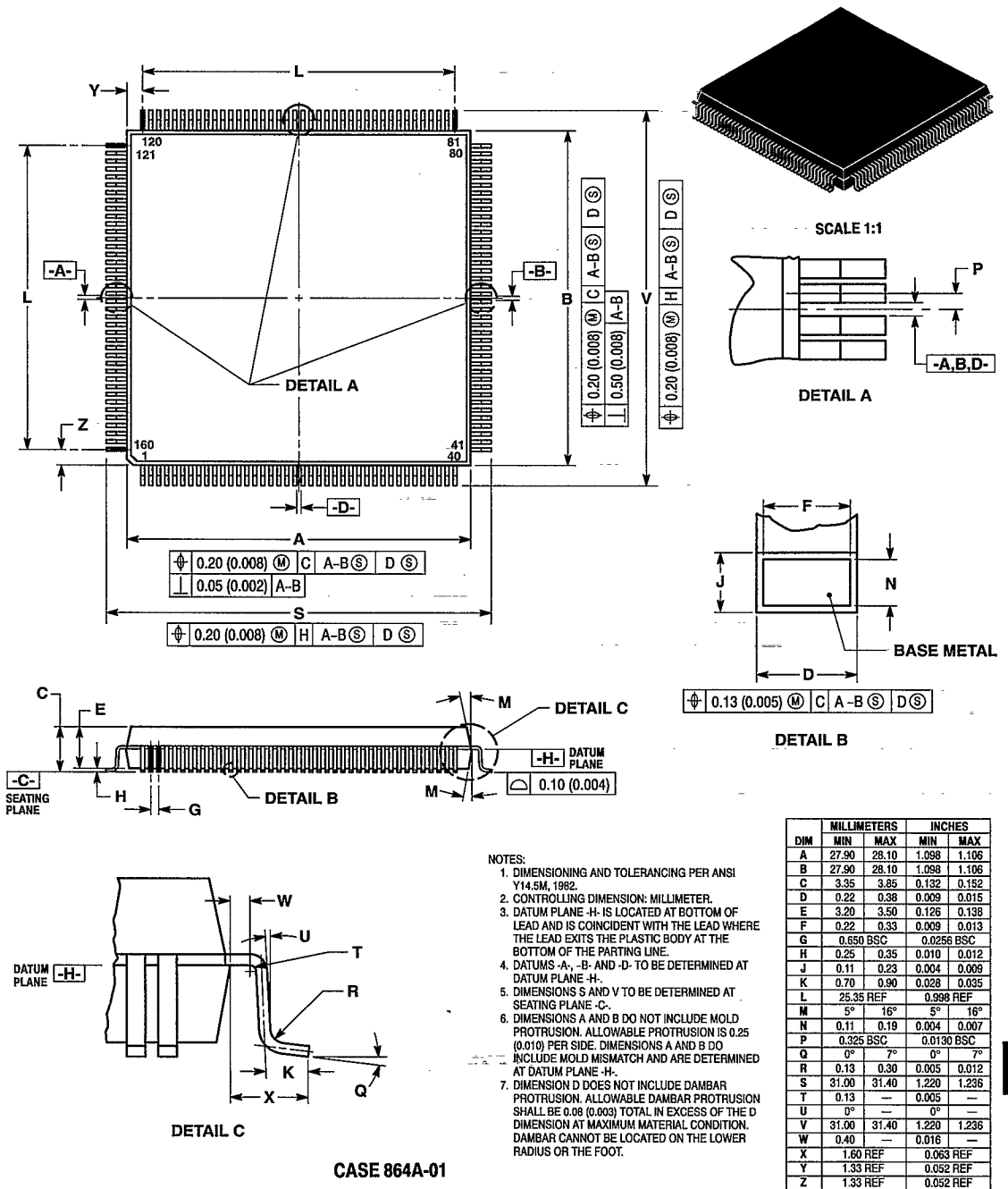
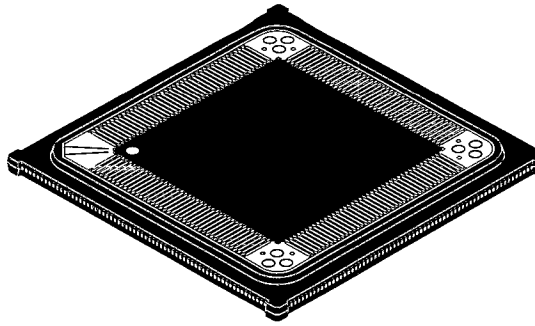
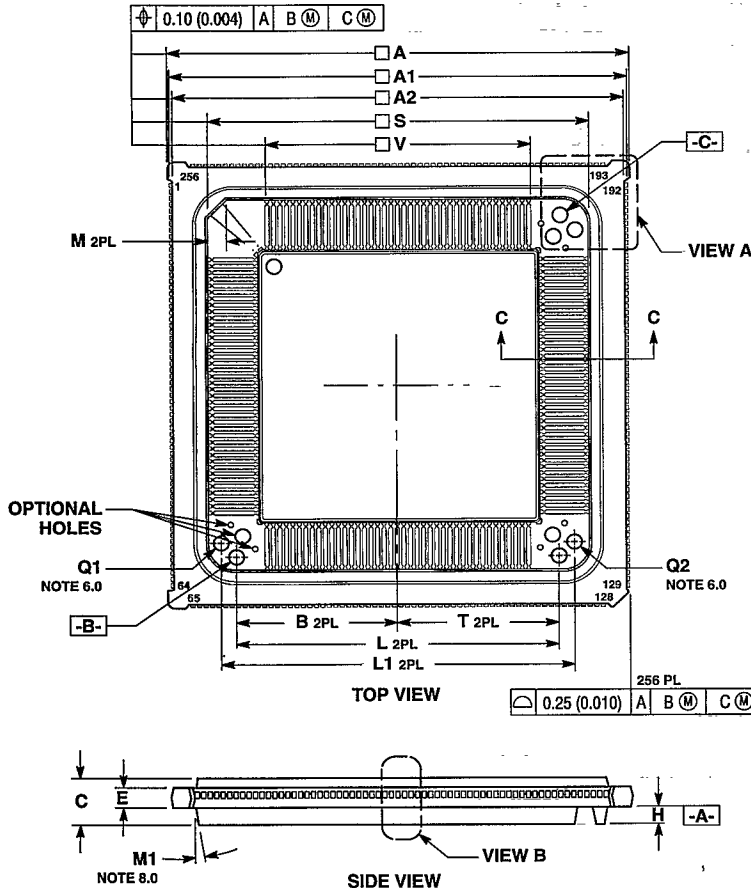


Figure B-2. 160-Pin QFP Package Dimensions



SCALE 1:1



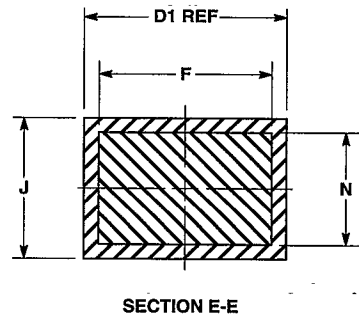
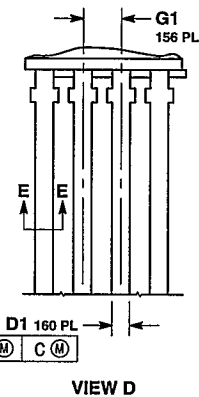
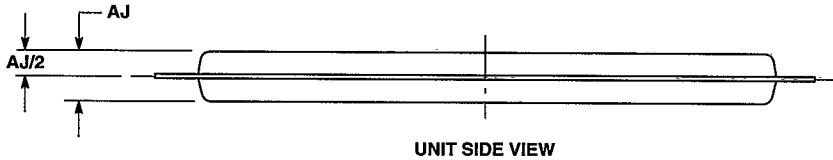
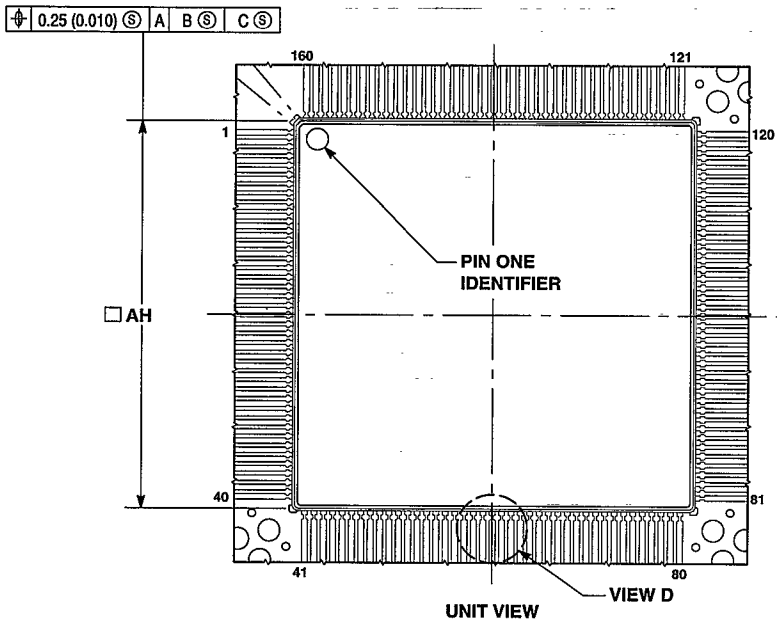
NOTES:

1. ALL DIMENSIONS AND TOLERANCES CONFORM TO ANSI Y14.5M, 1982.
2. CONTROLLING DIMENSION: MILLIMETER.
3. A DIMENSION DOES NOT INCLUDE MOLD PROTRUSION. ALLOWABLE MOLD PROTRUSION IS 0.20 (0.008) PER SIDE.
4. A AND S DIMENSIONS INCLUDE MOLD MISMATCH, AND ARE MEASURED AT THE PARTING LINE.
5. UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE SYMMETRICAL ABOUT CENTERLINES.
6. B AND C DATUM HOLES ARE TO BE USED FOR TRIM, FORM AND EXCISE OF THE MOLDED PACKAGE ONLY. HOLES Q1 AND Q2 ARE TO BE USED FOR ELECTRICAL TESTING ONLY.
7. NON-DATUM HOLES ONLY.
8. APPLIES TO RING AND PACKAGE FEATURES.

DIM	MILLIMETERS		INCHES	
	MIN	MAX	MIN	MAX
A	45.87	46.13	1.806	1.816
A1	45.70 BSC		1.799 BSC	
A2	45.17	45.43	1.778	1.789
B	16.10 BSC		0.634 BSC	
C	4.70	4.90	0.185	0.193
D	0.40	0.50	0.016	0.020
D1	0.22	0.38	0.009	0.015
E	1.90	2.10	0.075	0.083
F	0.22	0.33	0.009	0.013
G	0.65 BSC		0.026 BSC	
G1	0.65 BSC		0.026 BSC	
H	1.70	1.90	0.067	0.075
J	0.13	0.23	0.005	0.009
L	32.20 BSC		1.268 BSC	
L1	35.20 BSC		1.386 BSC	
M	1.30	2.30	0.051	0.091
M1	8° MAX		8° MAX	
N	0.13	0.17	0.005	0.007
P	1.77	2.03	0.070	0.080
R	0.40	0.60	0.016	0.024
R1	3.50	4.50	0.138	0.177
R2	2.00	3.00	0.079	0.118
S	37.87	38.13	1.491	1.501
T	16.10 BSC		0.634 BSC	
V	26.30	26.40	1.035	1.039
W	1.45	1.55	0.057	0.061
AA	0.45	0.55	0.018	0.033
AB	0.30	0.50	0.012	0.024
AC	1.37	1.63	0.054	0.064
AD	1.37	1.63	0.054	0.064
AH	27.90	28.10	1.098	1.106
AJ	3.20	3.40	0.125	0.134

CASE 887-01

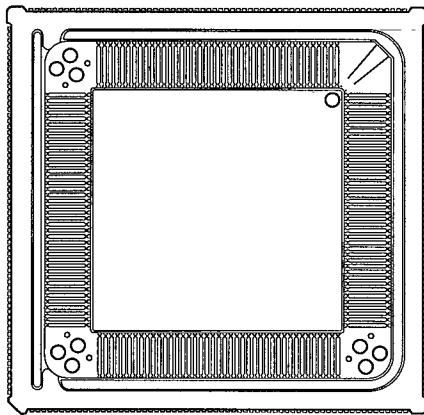
Figure B-3. 160-Pin QFP (with Molded Carrier Ring) Package Dimensions (1 of 3)



B

CASE 887-01

Figure B-3. 160-Pin QFP (with Molded Carrier Ring) Package Dimensions (2 of 3)



BOTTOM VIEW OF UNIT
IN MOLDED CARRIER RING

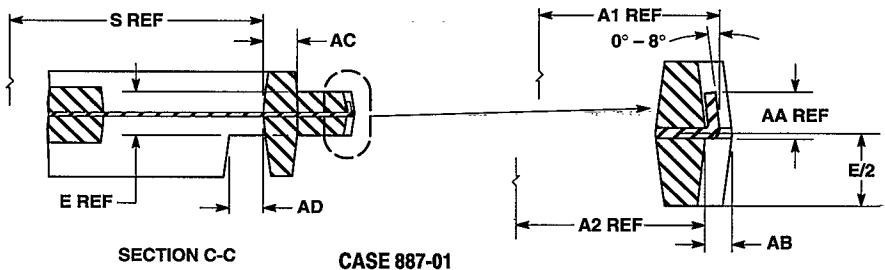
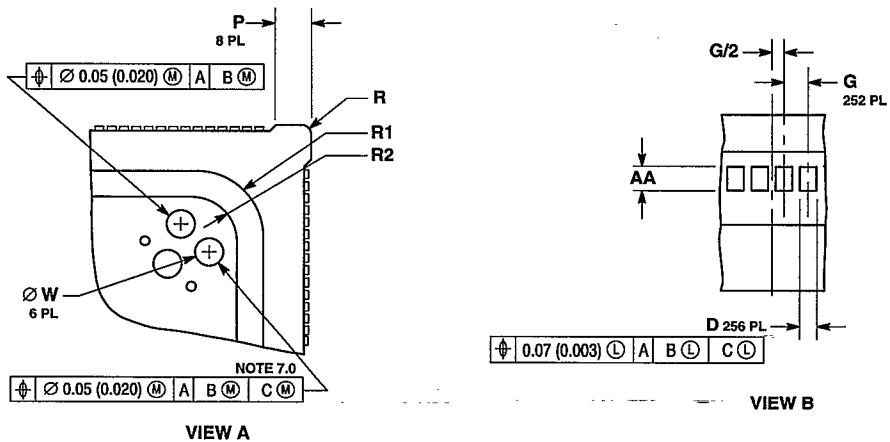


Figure B-3. 160-Pin QFP (with Molded Carrier Ring) Package Dimensions (3 of 3)

Table B-1. Ordering Information

Package	Temp	Frequency	Shipping Method	Order Number
160-Pin Quad Flat Pack	-40° to +85°C	16.78 MHz	24 pc tray	XC16Y1CFT16
			2 pc tray	SPAKXC16Y1CFT16
		20.97 MHz	24 pc tray	XC16Y1CFT20
			2 pc tray	SPAKXC16Y1CFT20
		25.17 MHz	24 pc tray	XC16Y1CFT25
			2 pc tray	SPAKXC16Y1CFT25
	-40° to +105°C	16.78 MHz	24 pc tray	XC16Y1VFT16
			2 pc tray	SPAKXC16Y1VFT16
		20.97 MHz	24 pc tray	XC16Y1VFT20
			2 pc tray	SPAKXC16Y1VFT20
		25.17 MHz	24 pc tray	XC16Y1VFT25
			2 pc tray	SPAKXC16Y1VFT25
	-40° to +125°C	16.78 MHz	24 pc tray	XC16Y1MFT16
			2 pc tray	SPAKXC16Y1MFT16
		20.97 MHz	24 pc tray	XC16Y1MFT20
2 pc tray			SPAKXC16Y1MFT20	
25.17 MHz		24 pc tray	XC16Y1MFT25	
		2 pc tray	SPAKXC16Y1MFT25	
160-Pin Quad Flat Pack with MCR	-40° to +85°C	16.78 MHz	10/tube	XC16Y1CFM16
		20.97 MHz	10/tube	XC16Y1CFM20
		25.17 MHz	10/tube	XC16Y1CFM25
	-40° to +105°C	16.78 MHz	10/tube	XC16Y1VFM16
		20.97 MHz	10/tube	XC16Y1VFM20
		25.17 MHz	10/tube	XC16Y1VFM25
	-40° to +125°C	16.78 MHz	10/tube	XC16Y1MFM16
		20.97 MHz	10/tube	XC16Y1MFM20
		25.17 MHz	10/tube	XC16Y1MFM25

B

B.4 Items Needed to Make a Custom MCU Order

A complete custom MCU order includes the following items:

A current order form that is completely filled out.

A signed "XC" status letter, when applicable.

Copies of customer specifications for any item that differs from the Motorola specification for the MCU. Customer specifications must be approved before an order can be made.

An application program stored on one of the media listed in **B.7 Application Program Media**.

An order can also include customer-supplied ROM code verification media.

B.5 How to Obtain an Ordering Form

The current MCU ordering form can be supplied by a Motorola representative or can be obtained from Motorola FREEWARE Data Services. The FREEWARE number is (512) 891-FREE. After making the connection, type bbs in lowercase letters and press the return key to access bulletin board software.

B.6 Information Necessary To Complete Ordering Form

The following list of questions is intended to help you obtain and organize the material needed to complete the ordering form. Subsequent paragraphs contain details of the ordering and verification process that will help to explain items in the table. Please contact a Motorola representative before filling out the form.

B.12 Technical Information covers technical requirements for an order.

Table B–2. Custom MCU Ordering Information Checklist

✓	Type of Information
	What is your ship-to address and phone number?
	What is the name and phone number of your designated customer contact?
	What is the name of your salesperson?
	Where is your Motorola or distributor sales office?
	Does the MCU have a pre-assigned Motorola SC number?
	Will the MCU have standard Motorola marking?
	Is additional customer-specified marking required?
	What is the MCU function and end use?
	Does the software or application include an encryption/decryption algorithm?
	Is there an end-use ECCN?
	Is the MCU being ordered for a military application?
	Are there special electrical provisions?
	What kind of pattern submission media are to be used?
	Are media labeled as specified?
	Is code on diskette in Motorola S19 record format?
	Are non-user areas blank or filled with zeros?
	Is verification listing media to be supplied?
	Is the verification listing to be shipped to you or to a Motorola sales office?
	Have you reviewed an errata sheet for the MCU?
	What is the default ROM array base address?
	What bootstrap information words are to be used?
	What are default values of masked ROM module configuration register bits?
	Is custom TPU microcode to be used?
	Does the microcode have a pre-assigned TPU S19 file identification number?
	What is MCU oscillator reference frequency?
	What is the MCU temperature range?
	What is the MCU package type?
	Are you requesting RVU center samples?
	What package type is to be used for the samples?
	Are samples to be shipped to you or to a Motorola sales office?

B

B.7 Application Program Media

Media listed in the order of Motorola's preference:

Macintosh 3-1/2 inch diskette (DSDD 800K or DSHD 1.4M)

MS-DOS 3-1/2 inch diskette (DSDD 720K or DSHD 1.44M)

MS-DOS 5-1/4 inch diskette (DSDD 360K or DSHD 1.2M)

Other media require prior factory approval.

B.8 Application Program Labeling

Clearly label application program diskettes with the following information:

Customer name

Customer part number

Project or product name

Filename of object code

Date

Operating system that formatted diskette

Formatted capacity of diskette

B.9 Application Program Format

Application programs must be submitted in S19 record format, a character-based object file format generated by Motorola cross assemblers and linkers.

Begin the application program at the first user ROM location. Program addresses must correspond exactly to available on-chip user ROM addresses. Write \$0000 in all unused ROM locations or leave unused ROM locations blank. See the current MCU ordering form for additional requirements. If an MCU has two user ROM areas with the same addresses, then write the two areas in separate files on the diskette. Label the diskette with both file names.

B.10 ROM Program Verification

Motorola inputs customer application code into a program that generates a listing verification file. The listing verification file represents the actual content of the masked-ROM array in the MCU. The verification file contains customer ROM code and may also contain other ROM code, such as self-check code. Motorola sends the customer a copy of the listing verification file along with a listing verification form. The customer must carefully check the verification file, complete and sign the verification form, and then return the form to Motorola.

A signed listing verification form constitutes a contractual agreement for the creation of a custom mask. By signing the listing verification form, the customer also agrees to accept liability for the minimum order quantity and further agrees to comply with the Motorola cancellation policy and other conditions outlined on the front and back of the listing verification form.

All original pattern media are filed for contractual purposes and are not returned. To aid the customer in checking the listing verification file, Motorola will return a copy of the verification listing on customer supplied media. This is in addition to the standard hard copy printout.

Diskettes must be pre-formatted and blank. Only one format can be used.

Media listed in the order of Motorola's preference:

Macintosh 3-1/2 inch diskette (DSDD 800K or DSHD 1.4M)

MS-DOS 3-1/2 inch diskette (DSDD 720K or DSHD 1.44M)

MS-DOS 5-1/4 inch diskette (DSDD 360K or DSHD 1.2M)

Other media require prior factory approval.

B.11 Submitting an MCU ROM Pattern

Send the pattern media and the current ordering form to your sales representative, who will send them to the appropriate marketer. The marketer assigns a special custom number (SC#), which serves as a unique identifier for the customer code in our product, then generates an internal specification that indicates customer requirements (information taken from the ordering form). The marketer then sends the pattern media and the internal documentation to the RVU Center. Production orders cannot be placed until the internal document has been released.

The RVU Center dumps the code from the pattern media, checks the options and code and runs the ROM listing. The listing and additional diskettes provided by the customer are then returned, with the listing verification form. The customer must check the listing verification file thoroughly, complete and sign the listing verification form, then return the form to Motorola. No action can be taken by Motorola until the form is returned. Once the listing is verified, the mask process begins. At this time, the customer must pay the mask charge and place an order for the minimum order quantity. After receiving the signed listing form, Motorola manufactures a custom photographic mask that is used to process silicon wafers. An application program cannot be changed after mask manufacture begins.

When parts are to be marked with a special customer part number, Motorola must know whether there is a customer print or specification governing the part number, or whether the customer part number is for reference only. Reference-only part numbers are applied according to Motorola specifications. If a customer print or specification for the part number differs from the Motorola specification, a copy of the customer print or specification must be approved by Motorola sales and engineering before the pattern is submitted. A copy of the customer print or specification must be attached to the filed copy of the internal documentation. If the customer part number is for reference only, the customer should sign in the appropriate field on the MCU Ordering Form.

B.12 Technical Information

This is an overview of information needed to complete a custom MCU order. Please refer to the appropriate sections of the user's manual for detailed information.

B.12.1 Clock Reference Option

The standard MC68HC16Y1 is designed to work with a clock synthesizer reference frequency of 4.194 MHz. Operation with a clock synthesizer reference frequency of 32.768 kHz is available as a metal mask option. Please contact your sales representative for more information. Refer to **SECTION 4 SINGLE-CHIP INTEGRATION MODULE** for additional information about the system clock synthesizer.

B.12.2 Masked ROM Module Options

Refer to **SECTION 11 MASKED ROM MODULE** for additional information about the ROM array and the content of the MRM control registers.

B.12.2.1 Base Address

The ROM array can be assigned a default address by mask-programming the reset value of the ROM base address high register (ROMBAH). The array can be mapped to any 64 Kbyte boundary in the memory map. The array occupies addresses from \$x0000 to \$xBFFF, where x = 0 to F. ROMBAH must equal \$000x when $0 \leq x \leq 7$; it must equal \$00Fx when $8 \leq x \leq F$.

For example:

When ROMBAH : ROMBAL = \$00030000, the ROM array is located at addresses \$30000 to \$3BFFF.

When ROMBAH : ROMBAL = \$00FC0000, the ROM array is located at addresses \$C0000 to \$CBFFF.

B.12.2.2 Configuration Options

Default operational characteristics of the ROM can be determined by mask-programming control bits in the masked ROM module configuration register (MRMCR).

BOOT — Boot ROM Control Bit

0 = CPU16 accesses ROM bootstrap word addresses after reset

1 = CPU16 cannot access ROM bootstrap word addresses after reset

Bootstrap function is overridden if STOP = 1.

LOCK — Lock Registers Bit

0 = Write lock disabled; protected registers and fields can be written

1 = Write lock enabled; protected registers and fields cannot be written

LOCK protects the ASPC and WAIT fields, as well as the ROMBAL and ROMBAH registers. ASPC, ROMBAL and ROMBAH are also protected by the STOP bit.

ASPC — ROM Array Space Field

Because the CPU16 operates only in supervisory mode, ASPC determines whether accesses are restricted solely to program space, or whether accesses are made to both program and data space. The following table shows ASPC encoding.

ASPC[1:0]	State Specified
X0	Program and data access
X1	Program access only

WAIT — Wait States Field

WAIT specifies the number of wait states inserted by the MRM during ROM array accesses.

WAIT[1:0]	Cycles per Transfer
00	3
01	4
10	5
11	2

B.12.2.3 Bootstrap Options

When the default value of the MRMCR $\overline{\text{BOOT}}$ bit is zero, the contents of bootstrap words ROMBS[0:3] are used as CPU16 reset vectors. The user must specify the content of these locations. In M68HC16 devices, ROMBS0 to ROMBS3 occupy system addresses \$000000 to \$000006.

Bootstrap Vector Table

Bootstrap Word	Vector Address	Address Space	Vector Content
ROMBS0	0000	P	Initial ZK, SK, and PK
ROMBS1	0002	P	Initial PC
ROMBS2	0004	P	Initial SP
ROMBS3	0006	P	Initial IZ (Direct Page)

B

B.13 Time Processor Unit Options

Although factory-programmed time functions can perform a wide variety of control tasks, they may not be ideal for all applications. The TPU provides emulation capability that allows the user to develop new time functions. Refer to **Section 9 Time Processor Unit** for more information.

To support changing TPU application requirements, Motorola has established a TPU function library. The function library is a collection of TPU functions written for easy assembly in combination with each other or with custom functions. Refer to Motorola Programming Note TPUPN00/D, *Using the TPU Function Library and TPU Emulation Mode* for information about developing custom functions and accessing the TPU function library.

Once new TPU functions have been developed, they can be installed in the TPU micro-ROM as a mask option. Contact a Motorola salesperson to obtain a unique TPU ROM file identification number before submitting code. TPU ROM programming and verification procedures are much the same as for MRM masks. TPU ROM size is 4 Kbytes. Custom TPU microcode must be submitted in a separate file from ROM array code, and must be labeled with the unique TPU ROM file identification number assigned by Motorola.

APPENDIX C DEVELOPMENT SUPPORT

This section serves as a brief reference to Motorola development tools for the MC68HC16Y1 microcontroller. Information provided is complete as of the time of publication, but new systems and software are continually being developed. In addition, there is a growing number of third-party tools available. The Motorola *MCU Tool Box* (MCUTLBX/D Rev. B) provides an up-to-date list of development tools. Contact your Motorola representative for further information.

Table C-1. MC68HC16Y1 Development Tools

Microcontroller Part Number	Modular Development System	Modular Evaluation System
MC68HC16Y1	M68HC16Y1MPB	M68HC16Y1MEVB

C.1 M68HC16Y1MEVB Modular Evaluation Board

The modular evaluation system (MEVB) is a development tool for evaluating M68HC16 and M68300 MCU-based systems. The MEVB consists of the M68HC16MPFB modular platform board, an MCU personality board (MPB), an in-circuit debugger printed circuit board (ICD16 or ICD32), and development software. MEVB features include:

- An economical means of evaluating target systems incorporating M68HC16 and M68300 HCMOS MCU devices.
- Expansion memory sockets for installing RAM, EPROM, or EEPROM.
 - Data RAM: 32K x 16, 128K x 16, or 512K x 16
 - EPROM/EEPROM: 32K x 16, 64K x 16, 128K x 16, 256K x 16, or 512K x 16
 - Fast RAM: 32K x 16 or 128K x 16
- Background-mode operation, for detailed operation from a personal computer platform without an on-board monitor.
- Integrated assembly/editing/evaluation/programming environment for easy development.
- As many as seven software breakpoints.
- Re-usable ICD hardware for your target application debug or control.
- Two RS-232C terminal input/output (I/O) ports for user evaluation of the serial communication interface.

- Logic analyzer pod connectors.
- Port replacement unit (PRU) to rebuild I/O ports lost to address/data/control.
- On-board 5V-to-12V switcher for MCU and flash EEPROM programming.
- On-board wire-wrap area.

C.2 M68HC16Y1MPB Modular Development System

The M68HC16 Motorola modular development system (MMDS16) is a development tool for M68HC16 MCU-based systems. The MMDS16 is an emulator, bus state analyzer, and control station for debugging hardware and software. A separately purchased active probe completes MMDS16 functionality with regard to a particular MCU or MCU family. The many active probes available let your MMDS16 emulate a variety of different MCUs. Contact your Motorola sales representative, who will assist you in selecting and configuring the modular system that fits your needs. A full-featured development system, the MMDS16 provides both in-circuit emulation and bus analysis capabilities, including:

- Real-time in-circuit emulation at maximum speed of 20 MHz (can be upgraded to 33 MHz)
- Built-in emulation memory
 - 1 Mbyte main emulation memory (fast termination, 2 bus cycle)
 - 4 Kbytes dual-port emulation memory
- Real-time bus analysis
 - Instruction disassembly
 - State-machine-controlled triggering
- Four hardware breakpoints, bitwise masking
- Analog/digital emulation
- Synchronized signal output
- Built-in AC power supply, 85–264 V, 50–60 Hz, FCC and EC EMI compliant
- RS-232 connection to host capable of communicating at 1200, 2400, 4800, 9600, 19200, 38400, or 57600 baud

APPENDIX D REGISTER SUMMARY

This appendix contains address maps, register diagrams, and bit/field definitions. Detailed information about register function is located in the corresponding sections of the manual.

Table D–1. MC68HC16Y1 Module Address Map

Module	Size (Bytes)	Address Bus Decoding						Base Address
		23	12	11	10	9	0	
ADC	64	M111	1111	1111	0111	00XX	XXXX	\$YFF700
MRM CTRL	32	M111	1111	1111	1000	0010	0000	\$YFF820
GPT	64	M111	1111	1111	1001	00XX	XXXX	\$YFF900
SCIM	128	M111	1111	1111	1010	0XXX	XXXX	\$YFFA00
TPURAM CTRL	64	M111	1111	1111	1011	0000	0XXX	\$YFFB00
MCCI	64	M111	1111	1111	1100	00XX	XXXX	\$YFFC00
TPU	512	M111	1111	1111	110X	XXXX	XXXX	\$YFFFFF

Control registers for all the modules in the microcontroller are mapped into a 4-Kbyte block. The state of the module mapping (MM) bit in the SCIM module configuration register (SCIMCR) determines where the control registers block is located in the system memory map. When MM = 0, register addresses range from \$7FF000 to \$7FFFFFFF; when MM = 1, register addresses range from \$FFF000 to \$FFFFFFF.

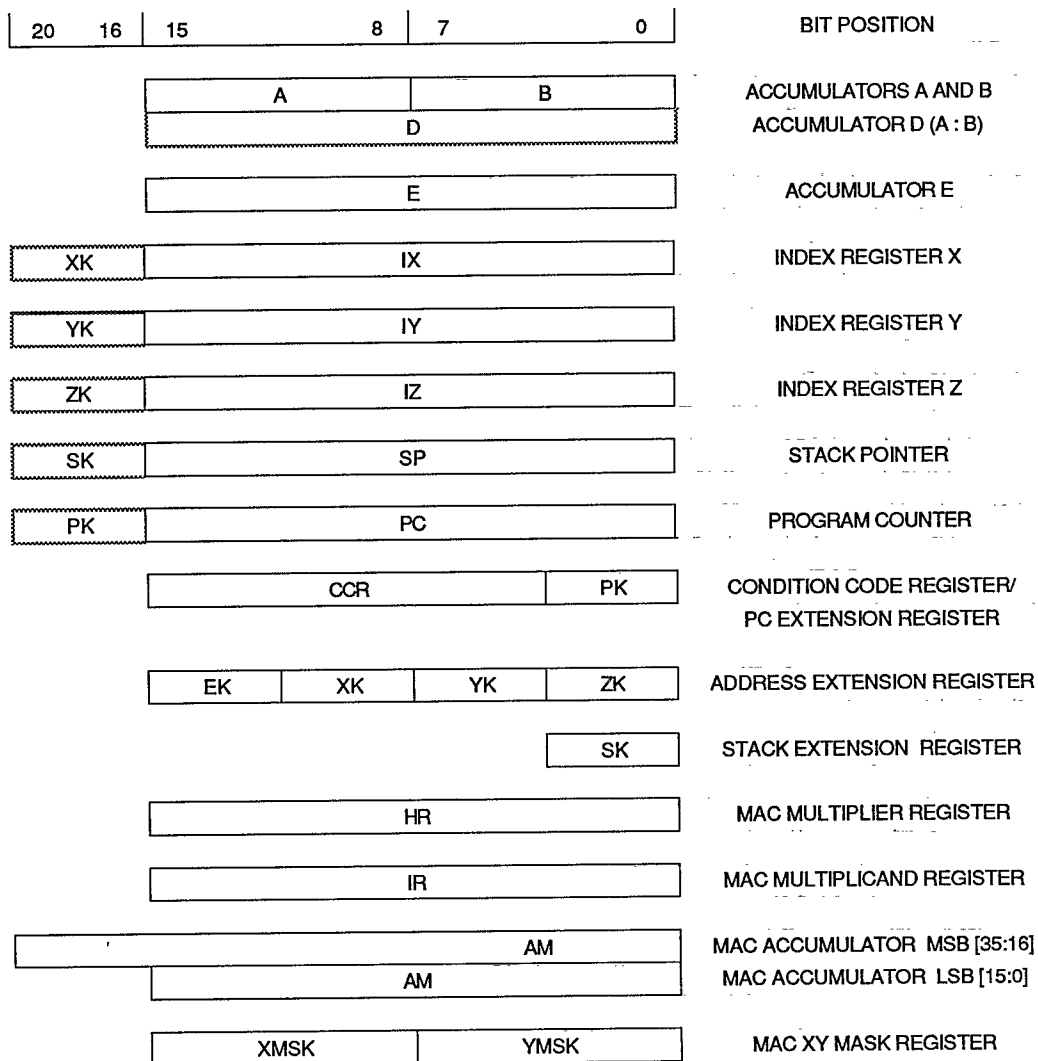
ADDR[23:20] are driven to the same logic state as ADDR19. MM corresponds to IMB ADDR23. If MM is cleared, the SCIM maps IMB modules into address space \$7FF000–\$7FFFFFFF, which is inaccessible to the CPU16. Modules remain inaccessible until reset occurs. The reset state of MM is one, but the bit can be written once. Initialization software should write MM to make certain it remains set.

D.1 Central Processing Unit

CPU16 registers are not part of the module address map. The following diagram is a functional representation of CPU resources.

D

D.1.1 CPU16 Register Model



D.1.2 CCR — 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			SM	PK			

The CCR contains processor status flags, the interrupt priority field, and the program counter address extension field. The CPU16 has a special set of instructions that manipulate the CCR.

S — STOP Enable

0 = Stop clock when LPSTOP instruction is executed.

1 = Perform NOP when LPSTOP instruction is executed.

MV — Accumulator M overflow flag

Set when overflow into AM35 has occurred.

H — Half Carry Flag

Set when a carry from A3 or B3 occurs during BCD addition.

EV — Extension Bit Overflow Flag

Set when an overflow into AM31 has occurred.

N — Negative Flag

Set when the MSB of a result register is set.

Z — Zero Flag

Set when all bits of a result register are zero.

V — Overflow Flag

Set when two's complement overflow occurs as the result of an operation.

C — Carry Flag

Set when carry or borrow occurs during arithmetic operation. 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

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

PK[3:0] — Program Counter Address Extension

This field is concatenated with the program counter to form a 20-bit address.

D.2 Analog-to-Digital Converter Module

Table D–2. ADC Module Address Map

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

Y = M111, where M is the state of the MM bit in SCIMCR. In an M68HC16 system, M must always be set to one.

D.2.1 ADCMCR — ADC Module Configuration Register

\$YFF700

15	14	13	12	8	7	6	0
STOP	FRZ	NOT USED			SUPV	NOT USED	
RESET:							
1	0	0			0		

ADCMCR controls ADC operation during low-power stop and freeze modes.

STOP — STOP Mode

0 = Normal operation

1 = Low-power operation

STOP places the ADC in low-power state. Setting STOP aborts any conversion in progress. STOP is set during reset and must be cleared before ADC operation can begin. Because analog circuitry bias current has been turned off, there must be a period of recovery after STOP is cleared before conversion begins.

FRZ[1:0] — FREEZE Response

The FRZ field determines ADC response to assertion of the FREEZE signal.

Freeze Encoding

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

SUPV — Supervisor/Unrestricted

Because the CPU16 operates only in supervisor mode, this bit has no effect.

D.2.2 ADCTST — ADC Test Register

\$YFF702

ADCTST is used with the SCIM test register for factory test of the ADC.

D.2.3 PORTADA — Port ADA Data Register

\$YFF706

15	8	7	0								
NOT USED				PADA7	PADA6	PADA5	PADA4	PADA3	PADA2	PADA1	PADA0

RESET:

INPUT DATA

A read of PORTADA[7:0] returns the logic levels of port ADA pins. If an input is outside specified logic levels, an indeterminate value is read. Use as a digital input does not preclude use as an analog input.

D.2.4 ADCTL0 — A/D Control Register 0

\$YFF70A

15	8	7	6	5	4	3	2	1	0
NOT USED			RES10	STS		PRS			
RESET:									
			0	0	0	0	0	0	1 1

ADCTL0 is used to select resolution, sample time, and clock/prescaler value. Writing ADCTL0 aborts any conversion in progress. ADC activity halts until ADCTL1 is written.

RES10 — 10-Bit Resolution

- 0 = 8-bit conversion
- 1 = 10-bit conversion

STS[1:0] — Sample Time Selection

The STS field selects one of four sample times.

Sample Time Selection

STS[1:0]	Sample Time
00	4 ADC Clock Periods
01	8 A/D Clock Periods
10	16 A/D Clock Periods
11	32 A/D Clock Periods

PRS[4:0] — Prescaler Rate Selection

ADC clock is generated from system clock using a modulus counter and a divide-by-two circuit. PRS contains the counter modulus.

ADC Clock Selection

PRS[4:0]	ADCCLK
%00000	Reserved
%00001	Sys Clk/4
%00010	Sys Clk/6
...	...
%11101	Sys Clk/60
%11110	Sys Clk/62
%11111	Sys Clk/64

D.2.5 ADCTL1 — ADC Control Register 1

\$YFF70C

15	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 initiates A/D conversion, and selects conversion mode and the analog channel or channels. Writing ADCTL1 aborts any conversion in progress and initiates a new conversion sequence.

SCAN — Scan Mode Selection

- 0 = Single conversion sequence
- 1 = Continuous conversion

MULT — Multichannel Conversion

- 0 = Conversion sequence(s) run on a single channel selected by [CD:CA].
- 1 = Sequential conversion of four or eight channels selected by [CD:CA].

S8CM — Select Eight-Conversion Sequence Mode

- 0 = Four-conversion sequence
- 1 = Eight-conversion sequence

ADC Conversion Modes

SCAN	MULT	S8CM	MODE
0	0	0	Single 4-Conversion Single-Channel Sequence
0	0	1	Single 8-Conversion Single-Channel Sequence
0	1	0	Single 4-Conversion Multichannel Sequence
0	1	1	Single 8-Conversion Multichannel Sequence
1	0	0	Multiple 4-Conversion Single-Channel Sequences
1	0	1	Multiple 8-Conversion Single-Channel Sequences
1	1	0	Multiple 4-Conversion Multichannel Sequences
1	1	1	Multiple 8-Conversion Multichannel Sequences

[CD:CA] — Channel Selection

Bits in this field select input channel or channels for analog-to-digital conversion.

Conversion mode determines which channel or channels are selected for conversion and which result registers are used to store conversion results. The following tables contain a summary of the effects of ADCTL1 bits and fields.

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	0	1	1	0	AN6	RSLT[0:3]
0	0	1	1	1	AN7	RSLT[0:3]
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	RSLT[0:7]
1	0	1	1	1	AN7	RSLT[0:7]
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]

Multichannel Conversions

S8CM	CD	CC	CB	CA	Input	Result Register
0	0	0	X	X	AN[0:3]	RSLT[0:3]
0	0	1	X	X	AN[4:7]	RSLT[0:3]
0	1	0	X	X	Reserved	RSLT[0:3]
0	1	1	X	X	V _{RH} V _{RL} (V _{RH} - V _{RL}) / 2 Test/Reserved	RSLT0 RSLT1 RSLT2 RSLT3
1	0	X	X	X	AN[0:7]	RSLT[0:7]
1	1	X	X	X	Reserved Reserved Reserved Reserved V _{RH} V _{RL} (V _{RH} - V _{RL}) / 2 Test/Reserved	RSLT0 RSLT1 RSLT2 RSLT3 RSLT4 RSLT5 RSLT6 RSLT7

D.2.6 ADSTAT — ADC Status Register

\$YFF70E

15	14	11	10	8	7	0
SCF	NOT USED			CCTR		CCF
RESET:						
0	0 0 0			0 0 0 0 0 0		0

ADSTAT is a read-only register that contains the sequence complete flag, the conversion counter, and a channel-converted flag for each of the input channels.

SCF — Sequence Complete Flag

0 = Sequence not complete

1 = Sequence complete

When SCAN = 0, SCF is set at the end of a conversion sequence. When SCAN = 1, SCF is set at the end of the first conversion sequence. SCF is cleared when converter activity is halted or restarted by a write to ADCTL0.

CCTR[2:0] — Conversion Counter

This field shows the content of the conversion counter pointer during a conversion sequence. The value is the number of the next result register to be written.

CCF[7:0] — Conversion Complete Flags

Each bit in this field corresponds to an ADC result register. A bit is set when conversion of the corresponding input is complete. It remains set until the result register is read. It is cleared when the register is read.

D.2.7 RSLT[0:7] — ADC Result Registers**\$YFF710–\$YFF73E**

Result registers contain conversion results. Data format depends on the address from which it is read. The notation 10 in a diagram indicates that a bit is used only for 10-bit resolution and is cleared during 8-bit conversion. The notation 8/10 indicates a bit is used for both 8-bit and 10-bit resolution. Unused bits return zeros when read.

D.2.7.1 RJRR — Unsigned Right-Justified Result Registers \$YFF710–\$YFF71F

15	10	9	8	7	6	5	4	3	2	1	0
NOT USED				10	10	8/10	8/10	8/10	8/10	8/10	8/10

D.2.7.2 LJSRR — Signed Left-Justified Result Registers**\$YFF720–\$YFF72F**

15	14	13	12	11	10	9	8	7	6	5	0
8/10	8/10	8/10	8/10	8/10	8/10	8/10	8/10	8/10	10	10	NOT USED

This data format assumes that the zero reference point is $(V_{RH} - V_{RL}) / 2$. Bit 15 thus indicates the sign of the result. When bit 15 = 1, the result is positive; when bit 15 = 0, the result is negative. Bits [5:0] return zeros when read.

D.2.7.3 LJRR — Unsigned Left-Justified Result Registers**\$YFF730–\$YFF73F**

15	14	13	12	11	10	9	8	7	6	5	0
8/10	8/10	8/10	8/10	8/10	8/10	8/10	8/10	8/10	10	10	NOT USED

D.3 Masked ROM Module

Table D-3. MRM Control Register Address Map

Address	15	8	7	0
\$YFF820	MASKED ROM MODULE CONFIGURATION REGISTER (MRMCR)			
\$YFF822	NOT IMPLEMENTED			
\$YFF824	ARRAY BASE ADDRESS REGISTER HIGH (ROMBAH)			
\$YFF826	ARRAY BASE ADDRESS REGISTER LOW (ROMBAL)			
\$YFF828	ROM SIGNATURE HIGH REGISTER (RSIGHI)			
\$YFF82A	ROM SIGNATURE LOW REGISTER (RSIGLO)			
\$YFF82C	NOT IMPLEMENTED			
\$YFF82E	NOT IMPLEMENTED			
\$YFF830	ROM BOOTSTRAP WORD 0 (ROMBS0)			
\$YFF832	ROM BOOTSTRAP WORD 1 (ROMBS1)			
\$YFF834	ROM BOOTSTRAP WORD 2 (ROMBS2)			
\$YFF836	ROM BOOTSTRAP WORD 3 (ROMBS3)			
\$YFF838	NOT IMPLEMENTED			
\$YFF83A	NOT IMPLEMENTED			
\$YFF83C	NOT IMPLEMENTED			
\$YFF83E	NOT IMPLEMENTED			

Y = M111, where M is the state of the MM bit in SCIMCR. In an M68HC16 system, M must always be set to one.

D.3.1 Masked ROM Control Registers

The 32-byte control register block contains registers that are used to configure the MRM and to control ROM array function. Configuration information is specified and programmed at the same time as the ROM content.

D.3.1.1 MRMCR — Masked ROM Module Configuration Register \$YFF820

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STOP	0	0	BOOT	LOCK	EMUL	ASPC			WAIT		0	0	0	0	0	0
RESET:		0	0	USER SPEC	USER SPEC		USER SPEC		USER SPEC		0	0	0	0	0	0

*Reset state of STOP = DATA14. Reset state of EMUL = (DATA10 * DATA13).

STOP — Stop Bit

0 = Normal ROM operation

1 = Disable ROM and activate emulator mode if enabled

Reset state of STOP is the complement of DATA14 state during reset. ROM array base address cannot be changed unless STOP is set.

BOOT — Boot ROM Control Bit

- 0 = CPU16 accesses ROM bootstrap word addresses after reset
- 1 = CPU16 cannot access ROM bootstrap word addresses after reset

Reset state of **BOOT** is specified by the user. Bootstrap function is overridden if **STOP** = 1.

LOCK — Lock Registers Bit

- 0 = Write lock disabled; protected registers and fields can be written
- 1 = Write lock enabled; protected registers and fields cannot be written

Reset state of **LOCK** is specified by the user. **LOCK** protects the **ASPC** and **WAIT** fields, as well as the **ROMBAL** and **ROMBAH** registers. **ASPC**, **ROMBAL** and **ROMBAH** are also protected by the **STOP** bit.

EMUL — Emulator Mode Control Bit

- 0 = Normal ROM operation
- 1 = MRM enters emulator mode when **STOP** is set.

Reset state of **EMUL** is the complement of **DATA10** and **DATA13** state during reset. When **EMUL** is set, the MRM responds to accesses by asserting the **CSM** signal.

ASPC — ROM Array Space Field

Because the CPU16 operates only in supervisory mode, **ASPC** determines whether accesses are restricted solely to program space, or whether accesses are made to both program and data space. In systems with restricted access levels, **ASPC** also determines whether accesses are restricted solely to supervisor space. The reset state of **ASPC** is user specified. The following table shows **ASPC** encoding.

ASPC[1:0]	State Specified
X0	Program and data access
X1	Program access only

WAIT — Wait States Field

WAIT specifies the number of wait states inserted by the MRM during ROM array accesses.

WAIT[1:0]	Cycles per Transfer
00	3
01	4
10	5
11	2

D.3.1.2 ROMBAH — Array Base Address Register High

\$YFF824

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:

USER SPECIFIED

D.3.1.3 ROMBAL — Array Base Address Register Low

\$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:

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

ROMBAH and ROMBAL are used to specify ROM array base address. They can only be written when STOP = 1 and LOCK = 0. This prevents accidental remapping of the array. Because ADDR[23:20] are driven to the same logic state as ADDR19, addresses in the range \$080000 to \$F7FFFF cannot be accessed. Because the ROM array must be mapped to a 64 Kbyte boundary, ROMBAL always contains \$0000.

D.3.1.4 RSIGHI — ROM Signature High Register

\$YFF828

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NOT USED												RSP18	RSP17	RSP16	

RESET:

FACTORY SPECIFIED

D.3.1.5 RSIGLO — ROM Signature Low Register

\$YFF82A

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSP15	RSP14	RSP13	RSP12	RSP11	RSP10	RSP9	RSP8	RSP7	RSP6	RSP5	RSP4	RSP3	RSP2	RSP1	RSP0

RESET:

FACTORY SPECIFIED

RSIGHI and RSIGLO are used to specify a ROM signature pattern. A special signature identification algorithm can allow the user to verify the content of the ROM array. The signature is specified by the factory and cannot be changed.

D.3.1.6 ROMBS0 — ROM Bootstrap Word 0

\$YFF830

D.3.1.7 ROMBS1 — ROM Bootstrap Word 1

\$YFF832

D.3.1.8 ROMBS2 — ROM Bootstrap Word 2

\$YFF834

D.3.1.9 ROMBS3 — ROM Bootstrap Word 3

\$YFF836

Typically, reset vectors for the system CPU are contained in nonvolatile memory and are only fetched when the CPU comes out of reset. The user can specify that these four words be used as reset vectors, and can specify the content of these locations. The content of these words cannot be changed. In M68HC16 devices, ROMBS0 to ROMBS3 occupy system addresses \$000000 to \$000006.

D.4 General-Purpose Timer

Table D-4. General-Purpose Timer Address Map

Address	15	8	7	0
\$YFF900	GPT MODULE CONFIGURATION (GPTMCR)			
\$YFF902	(RESERVED FOR TEST)			
\$YFF904	INTERRUPT CONFIGURATION (ICR)			
\$YFFE06	PGP DATA DIRECTION (DDRGP)		PGP DATA (PORTGP)	
\$YFF908	OC1 ACTION MASK (OC1M)		OC1 ACTION DATA (OC1D)	
\$YFF90A	TIMER COUNTER (TCNT)			
\$YFF90C	PA CONTROL (PACTL)		PA COUNTER (PACNT)	
\$YFF90E	INPUT CAPTURE 1 (TIC1)			
\$YFF910	INPUT CAPTURE 2 (TIC2)			
\$YFF912	INPUT CAPTURE 3 (TIC3)			
\$YFF914	OUTPUT COMPARE 1 (TOC1)			
\$YFF916	OUTPUT COMPARE 2 (TOC2)			
\$YFF918	OUTPUT COMPARE 3 (TOC3)			
\$YFF91A	OUTPUT COMPARE 4 (TOC4)			
\$YFF91C	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	PWM CONTROL A (PWMA)		PWM CONTROL B (PWMB)	
\$YFF928	PWM COUNT (PWMCNT)			
\$YFF92A	PWMA BUFFER (PWMBUFA)		PWMB BUFFER (PWMBUFB)	
\$YFF92C	GPT PRESCALER (PRESC)			
\$YFF92E– \$YFF93F	RESERVED			

Y = M111, where M is the state of the MM bit in SCIMCR. In an M68HC16 system, M must always be set to one.

D.4.1 GPTMCR — GPT Module Configuration Register

\$YFF900

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STOP	FRZ	STOPP	INCP	0	0	0	0	SUPV	0	0	0	IARB			
RESET:															
0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

GPTMCR bits control freeze, low-power stop, and single-step modes.

STOP — Stop Clocks

0 = Internal clocks not shut down

1 = Internal clocks shut down

D

FRZ[1:0] — FREEZE Response

FRZ1 is not used; FRZ0 encoding determines response to the IMB FREEZE signal.

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

Because the CPU16 operates in supervisor mode only, this bit has no effect.

IARB[3:0] — Interrupt Arbitration

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. 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.

D.4.2 GPTMTR — GPT Module Test Register (Reserved)

\$YFF902

This address is currently unused and returns zeros when read. It is reserved for GPT factory test.

D.4.3 ICR — GPT Interrupt Configuration Register

\$YFF904

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IPA				0	IRL			IVBA				0	0	0	0
RESET:															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ICR fields determine internal and external interrupt priority, and provide the upper nibble of the interrupt vector number supplied to the CPU when an interrupt is acknowledged.

IPA — Interrupt Priority Adjust

Specifies which of the 11 internal GPT interrupt sources is assigned highest priority.

IPL — Interrupt Priority Level

Specifies the priority level of GPT interrupt requests.



IVBA — Interrupt Vector Base Address

Contains the most significant nibble of interrupt vector numbers supplied by the GPT.

D.4.4 DDRGP — Port GP Data Direction Register

\$YFF906

PORTGP — Port GP Data Register

\$YFF907

15							8	7									0
DDGP7	DDGP6	DDGP5	DDGP4	DDGP3	DDGP2	DDGP1	DDGP0	PGP7	PGP6	PGP5	PGP4	PGP3	PGP2	PGP1	PGP0		
RESET:																	
0	0	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. Clearing a pin designates a pin for input; setting a pin designates it for output. PORTGP latches the port data.

D.4.5 OC1M — OC1 Action Mask Register

\$YFF908

OC1D — OC1 Action Data Register

\$YFF909

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
OC1M						0	0	0	OC1D						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 outputs are affected.

OC1M[5:1] — OC1 Mask

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

0 = If OC1 mask bit is set, clear corresponding output compare pin on OC1 match.

1 = If OC1 mask bit is set, set corresponding output compare pin on OC1 match.

OC1D[5:1] correspond to OC[5:1].

D.4.6 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.

D.4.7 PACTL — Pulse Accumulator Control Register

\$YFF90C

PACNT — Pulse Accumulator Counter

\$YFF90D

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PAIS	PAEN	PAMOD	PEDGE	PCLKS	I4/O5	PACLK			PACNT						
RESET:															
U	0	0	0	U	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 PEDGE and PAMOD are shown in the following table.

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)

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.



D.4.8 TIC[1:3] — Input Capture Registers 1–3**\$YFF90E–\$YFF912**

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

D.4.9 TOC[1:4] — Output Compare Registers 1–4**\$YFF914–\$YFF91A**

The 16-bit read/write output compare registers 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.

D.4.10 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.

D.4.11 TCTL1/TCTL2 — Timer Control Registers 1 and 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	EDGE4	EDGE3	EDGE2	EDGE1				
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.

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

EDGE[4:1] — Input Capture Edge Control

Each pair of bits configures input sensing logic for the corresponding input capture.

EDGE[4:1]	Configuration
00	Capture Disabled
01	Capture on Rising Edge Only
10	Capture on Falling Edge Only
11	Capture on Any (Rising or Falling) Edge

D.4.12 TMSK1/TMSK2 — Timer Interrupt Mask Registers 1 and 2

\$YFF920

15	14	11	10	8	7	6	5	4	3	2	0
I4/O5I	OCI			ICI		TOI	0	PAOVI	PAII	CPROUT	CPR
RESET:											
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 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 is set

PAOVI — Pulse Accumulator Overflow Interrupt Enable

0 = Pulse accumulator overflow interrupt disabled

1 = Interrupt requested when PAOVF is set

PAII — Pulse Accumulator Input Interrupt Enable

0 = Pulse accumulator interrupt disabled

1 = Interrupt requested when PAIF 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.

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

D.4.13 TFLG1/TFLG2 — Timer Interrupt Flag Registers 1 and 2

\$YFF922

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

These registers show condition flags that correspond to 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, it is set at the end of the timed period.

D

D.4.14 CFORC — Compare Force**\$YFF924****PWMC** — PWM Control**\$YFF925**

15	11	10	9	8	7	6	4	3	2	1	0		
FOC				0	FPWMA	FPWMB	PPROUT	PPR		SFA	SFB	F1A	F1B
RESET:													
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 or PWM pins. PWMC sets PWM operating conditions.

FOC[5:1] — Force Output Compare

0 = Has no meaning

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 — Force PWMA Value

0 = Normal PWMA operation

1 = The value of F1A is driven out on the PWMA pin, regardless of the state of PPROUT.

FPWMB — Force PWMB Value

0 = Normal PWMB operation

1 = The value of F1B is driven out on the PWMB pin.

PPROUT — PWM Clock Output Enable

0 = Normal PWM operation on PWMA

1 = TCNT clock 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.

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 — PWMA Slow/Fast Select

0 = PWMA period is 256 PWMCNT increments long.

1 = PWMA period is 32768 PWMCNT increments long.



SFB — PWMB Slow/Fast Select

0 = PWMB period is 256 PWMCNT increments long.

1 = PWMB period is 32768 PWMCNT increments long.

The following table shows the effects of SF settings on PWM frequency (16.78-MHz system clock).

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 — Force Logic Level One on PWMA

0 = Force logic level zero output on PWMA pin.

1 = Force logic level one output on PWMA pin.

F1B — Force Logic Level One on PWMB

0 = Force logic level zero output on PWMB pin.

1 = Force logic level one output on PWMB pin.

D.4.15 PWMA/PWMB — PWM Registers A/B

\$YFF926, \$YFF927

The value in these registers determines pulse width of the corresponding PWM output. A value of \$00 corresponds to continuously low output; a value of \$80 to 50% duty cycle. Maximum value (\$FF) selects an output that is high for 255/256 of the period. Writes to these registers are buffered by PWMBUFA and PWMBUFB.

D.4.16 PWMCNT — PWM Count Register

\$YFF928

PWMCNT is the 16-bit free-running counter used for GPT PWM functions.

D.4.17 PWMBUFA — PWM Buffer Register A

\$YFF92A

PWMBUFB — PWM Buffer Register B

\$YFF92B

To prevent glitches when PWM duty cycle is changed, the contents of PWMA and PWMB are transferred to these read-only registers at the end of each duty cycle. Reset state is \$0000.

D.4.18 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.

D

D.5 Single-Chip Integration Module

Table D–5. SCIM Address Map

Address	15	8	7	0
\$YFFA00	SCIM MODULE CONFIGURATION (SCIMCR)			
\$YFFA02	FACTORY TEST (SCIMTR)			
\$YFFA04	CLOCK SYNTHESIZER CONTROL (SYNCR)			
\$YFFA06	UNUSED		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	UNUSED		PORT E DATA (PORTE0)	
\$YFFA12	UNUSED		PORT E DATA (PORTE1)	
\$YFFA14	PORT A/B DATA DIRECTION (DDRAB)		PORT E DATA DIRECTION (DDRE)	
\$YFFA16	UNUSED		PORT E PIN ASSIGNMENT (PEPAR)	
\$YFFA18	UNUSED		PORT F DATA (PORTF0)	
\$YFFA1A	UNUSED		PORT F DATA (PORTF1)	
\$YFFA1C	UNUSED		PORT F DATA DIRECTION (DDRF)	
\$YFFA1E	UNUSED		PORT F PIN ASSIGNMENT (PFPAR)	
\$YFFA20	UNUSED		SYSTEM PROTECTION CONTROL (SYPCR)	
\$YFFA22	PERIODIC INTERRUPT CONTROL (PICR)			
\$YFFA24	PERIODIC INTERRUPT TIMING (PITR)			
\$YFFA26	UNUSED		SOFTWARE SERVICE (SWSR)	
\$YFFA28	UNUSED		PORT F EDGE-DETECT CONTROL (PORTFE)	
\$YFFA2A	UNUSED		PORT F EDGE-DETECT INTERRUPT VECTOR (PFIVR)	
\$YFFA2C	UNUSED		PORT F EDGE-DETECT INTERRUPT LEVEL (PFLVR)	
\$YFFA2E	UNUSED		UNUSED	
\$YFFA30	TEST MODULE MASTER SHIFT A (TSTMSRA)			
\$YFFA32	TEST MODULE MASTER SHIFT B (TSTMSRB)			
\$YFFA34	TEST MODULE SHIFT COUNT (TSTSC)			
\$YFFA36	TEST MODULE REPETITION COUNTER (TSTRC)			
\$YFFA38	TEST MODULE CONTROL (CREG)			
\$YFFA3A	TEST MODULE DISTRIBUTED REGISTER (DREG)			
\$YFFA3C	UNUSED		UNUSED	
\$YFFA3E	UNUSED		UNUSED	
\$YFFA40	UNUSED		PORT C DATA (PORTC)	
\$YFFA42	UNUSED		UNUSED	
\$YFFA44	CHIP-SELECT PIN ASSIGNMENT (CSPAR0)			

Table D-5. SCIM Address Map (Continued)

Address	15	8	7	0
\$YFFA46	CHIP-SELECT PIN ASSIGNMENT (CSPAR1)			
\$YFFA48	CHIP-SELECT BASE BOOT (CSBARBT)			
\$YFFA4A	CHIP-SELECT OPTION BOOT (CSORBT)			
\$YFFA4C	CHIP-SELECT BASE 0 (CSBAR0)			
\$YFFA4E	CHIP-SELECT OPTION 0 (CSOR0)			
\$YFFA50	UNUSED			
\$YFFA52	UNUSED			
\$YFFA54	UNUSED			
\$YFFA56	UNUSED			
\$YFFA58	CHIP-SELECT BASE 3 (CSBAR3)			
\$YFFA5A	CHIP-SELECT OPTION 3 (CSOR3)			
\$YFFA5C	UNUSED			
\$YFFA5E	UNUSED			
\$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	UNUSED		UNUSED	
\$YFFA7A	UNUSED		UNUSED	
\$YFFA7C	UNUSED		UNUSED	
\$YFFA7E	UNUSED		UNUSED	

Y = M111, where M is the state of the MM bit in SCIMCR. In an M68HC16 system, M must always be set to one.

D.5.1 SCIMCR — SCIM Configuration Register

\$YFFA00

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

SCIMCR controls system configuration. SCIMCR can be read or written at any time, except for the module mapping (MM) bit, which can be written once and then must remain set.

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 Enable

- 0 = When FREEZE is asserted, the software watchdog and periodic interrupt timer counters continue to run.
- 1 = When FREEZE is asserted, the software watchdog and periodic interrupt timer counters are disabled, preventing interrupts during software debug.

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

CPUD is reset to zero when the MCU is in an expanded mode, and to one in single-chip mode.

- 0 = Instruction pipeline signals available on pins IPIPE0 and IPIPE1
- 1 = Pins IPIPE0 and IPIPE1 placed in high-impedance state unless a breakpoint occurs

SLVEN — Factory Test Mode Enabled

- 0 = IMB is not available to an external master.
- 1 = An external bus master has direct access to the IMB.

SHEN[1:0] — Show Cycle Enable

This field determines what the EBI does with the external bus during internal transfer operations.

SUPV — Supervisor/User Data Space

Places SCIM global registers in either supervisor data space or user data space. Because the CPU16 operates only in supervisor mode, this bit has no effect.

MM — Module Mapping

The logic state of MM determines the value of ADDR[23:20] in the IMB module address. Because ADDR[23:20] are driven to the same logic state as ADDR19, MM must remain set. If MM is cleared, the MCU registers become inaccessible.

- 0 = Internal modules are addressed from \$7FF000–\$7FFFFFF.
- 1 = Internal modules are addressed from \$FFF000–\$FFFFFF.

ABD — Address Bus Disable

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

- 0 = Pins ADDR[2:0] operate normally.
- 1 = Pins ADDR[2:0] are disabled.

RWD — Read/Write Disable

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

0 = R/W signal operates normally

1 = R/W signal placed in high-impedance state

IARB[3:0] — Interrupt Arbitration Field

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. 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.

D.5.2 SCIMTR — SCIM Test Register

\$YFFA02

SCIMTR is used for factory test only.

D.5.3 SYNCR — Clock Synthesizer Control Register

\$YFFA04

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
W	X	Y						EDIV	0	0	SLIMP	SLOCK	RSTEN	STSCIM	STEXT
RESET:															
0	0	1	1	1	1	1	1	0	0	0	U	U	0	0	0

SYNCR determines system clock operating frequency and mode of operation. Clock frequency is determined by SYNCR bit settings as follows:

$$F_{\text{SYSTEM}} = \frac{F_{\text{REFERENCE}}}{128} [4(Y + 1)(2^{2W} + X)]$$

W — Frequency Control (VCO)

0 = Base frequency.

1 = Frequency multiplied by four.

X — Frequency Control Bit (Prescaler)

0 = Base system clock frequency (divide by two prescaler enabled).

1 = System clock frequency multiplied by two (divide by two prescaler disabled).

Y[5:0] — Frequency Control (Counter)

The Y field is the initial value for the modulus 64 down counter in the synthesizer feedback loop. Values range from 0 to 63.

EDIV — ECLK Divide Rate

0 = ECLK is system clock divided by 8.

1 = ECLK is system clock divided by 16.

SLIMP — Limp Mode

0 = External crystal is VCO reference.

1 = Loss of crystal reference.

SLOCK — Synthesizer Lock

0 = VCO is enabled, but has not locked.

1 = VCO has locked on the desired frequency or system clock is external.

RSTEN — Reset Enable

0 = Loss of reference causes the MCU to operate in limp mode.

1 = Loss of reference causes system reset.

STSCIM — Stop Mode Single-Chip Integration 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 = CLKOUT held low during low-power stop to conserve power.

1 = CLKOUT driven from SCIM clock during low-power stop, as determined by the state of the STSCIM bit.

D.5.4 RSR — Reset Status Register

\$YFFA07

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NOT USED								EXT	POW	SW	HLT	0	LOC	SYS	TST

RSR contains a status bit for each reset source in the MCU. A bit set to one indicates what type of reset has occurred. When multiple reset sources occur at the same time, more than one bit in RSR can be set. The reset status register is updated by the reset control logic when the MCU comes out of reset. This register can be read at any time. A write has no effect.

EXT — External Reset

Reset caused by an external signal.

POW — Power-Up Reset

Reset caused by the power-up reset circuit.

SW — Software Watchdog Reset

Reset caused by the software watchdog circuit.

HLT — Halt Monitor Reset

Reset caused by the halt monitor.

LOC — Loss of Clock Reset

Reset caused by loss of clock frequency reference.

SYS — System Reset

Reset was caused by a CPU reset instruction. Because the CPU16 has no reset instruction, this bit is not used and always reads zero.

TST — Test Submodule Reset

Reset caused by the test submodule.

D.5.5 SCIMTRE — Module Test Register E

\$YFFA08

Register is used for factory test only.

D.5.6 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

Ports A and B can be read or written at any time. If a pin in either I/O port is configured as an output, the corresponding bit value is driven out on the pin. When a pin is configured for output, a read of the port returns the latched bit value. When a pin is configured for input, a read returns the pin logic level.

D.5.7 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

Ports G and H can be read or written at any time. If a pin in either I/O port is configured as an output, the corresponding bit value is driven out on the pin. When a pin is configured for output, a read of the port returns the latched bit value. When a pin is configured for input, a read returns the pin logic level.

D.5.8 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:															
U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U

The bits in these registers control the direction of the pin drivers when the 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.

D.5.9 PORTE0/PORTE1 — Port E Data Register

\$YFFA11, \$YFFA13

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

PORTE is an internal data latch that can be accessed at two locations. PORTE can be read or written at any time. If a pin in I/O port E is configured as an output, the corresponding bit value is driven out on the pin. When a pin is configured for output, a read of PORTE returns the latched bit value. When a pin is configured for input, a read returns the pin logic level. Reads of PE3 always return one.

D.5.10 DDRAB — Port A/B Data Direction Register

\$YFFA14

DDRE — Port E Data Direction Register

\$YFFA15

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

The bits in these registers control the direction of the pin drivers when the pins are configured as I/O. Setting DDA or DDB configures all pins in the corresponding port as outputs. Clearing DDA or DDB configures all pins in the corresponding port as inputs. Setting a bit in DDRE configures the corresponding pin as an output. Clearing a bit configures the corresponding pin as an input.

D.5.11 PEPAR — Port E Pin Assignment Register

\$YFFA17

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NOT USED							PEPA7	PEPA6	PEPA5	PEPA4	PEPA3	PEPA2	PEPA1	PEPA0	
RESET (Expanded, Single-chip):															
							DATA8	DATA8	DATA8	DATA8	DATA8	DATA8	DATA8	DATA8	
							0	0	0	0	0	0	0	0	

The bits in this register control the function of each port E pin. Any bit set to one defines the corresponding pin as a bus control signal, with the function shown in the following table. Any bit cleared to zero defines the corresponding pin as an I/O pin controlled by PORTE and DDRE. E3 is not connected to a pin. DDE3, PE3, and PEPA3 can be read and written but have no function.



Port E Pin Assignment

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

* When PEPA3 is set, the PE3 pin goes to logic level one. The CPU16 does not support the control function for this pin.

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

D.5.12 PORTF0/PORTF1— Port F Data Register \$YFFA19, \$YFFA1B

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

PORTF is an internal data latch that can be accessed at two locations. It can be read or written at any time. If a pin in I/O port F is configured as an output, the corresponding bit value is driven out on the pin. When a pin is configured for output, a read of PORTF returns the latched bit value; when a pin is configured for input, a read returns the pin logic level.

D.5.13 DDRF — Port F Data Direction Register \$YFFA1D

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

Bits in this register control the direction of the port F pin drivers when pins are configured for I/O. Setting a bit configures the corresponding pin as an output; clearing a bit configures the corresponding pin as an input. This register can be read or written at any time.

D.5.14 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								PFFA3	PFFA2	PFFA1	PFFA0				

RESET (Expanded, Single-chip):

DATA9	DATA9	DATA9	DATA9
0	0	0	0

The fields in this register determine the functions of pairs of port F pins as shown in the following tables.

Port F Pin Assignment

PFPAR Field	Port F Signal	Alternate Signal
PFFA3	PF[7:6]	IRQ[7:6]
PFFA2	PF[5:4]	IRQ[5:4]
PFFA1	PF[3:2]	IRQ[3:2]
PFFA0	PF[1:0]	IRQ1, MODCLK*

*MODCLK signal is only recognized during reset

PFFAx Bits	Port F Signal
00	I/O pin
01	Rising edge detect
10	Falling edge detect
11	Interrupt request

$\overline{\text{BERR}}$ and DATA9 determine the reset state of this register. If $\overline{\text{BERR}}$ and/or DATA9 are low during reset, this register is set to \$00, defining all port F pins to be I/O pins. If $\overline{\text{BERR}}$ and DATA9 are both high during reset, the register is set to \$FF, which defines all port F pins except PF0 to be interrupt signals.

D.5.15 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	HME	BME	BMT		

RESET:

1	MODCLK	0	0	0	0	0	0
---	--------	---	---	---	---	---	---

SYPCR controls system monitor functions, software watchdog clock prescaling, and bus monitor timing. This register can be written once following power-on or reset.

SWE — Software Watchdog Enable

0 = Software watchdog disabled

1 = Software watchdog enabled

SWP — Software Watchdog Prescale

0 = Software watchdog clock not prescaled

1 = Software watchdog clock prescaled by 512



SWT[1:0] — Software Watchdog Timing

This field selects software watchdog timeout period.

SWP	SWT	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 External Enable

0 = Disable bus monitor function for an internal to external bus cycle.

1 = Enable bus monitor function for an internal to external bus cycle.

BMT[1:0] — Bus Monitor Timing

This field selects bus monitor timeout period.

BMT	Bus Monitor Timeout Period
00	64 System Clocks
01	32 System Clocks
10	16 System Clocks
11	8 System Clocks

D.5.16 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			PIV								
RESET:																
0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1

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

PIRQL[2:0] — Periodic Interrupt Request Level

This field determines the priority of periodic interrupt requests. If a PIT interrupt and an external IRQ of the same priority occur simultaneously, the PIT interrupt is serviced first. The periodic timer continues to run when the interrupt is disabled.

PIV[7:0] — Periodic Interrupt Vector

The bits of this field contain the periodic interrupt vector number generated in response to an interrupt from the periodic timer. When the SCIM responds, the periodic interrupt vector is placed on the bus.

D.5.17 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							
RESET:															
0	0	0	0	0	0	0	MODCLK	0	0	0	0	0	0	0	0

PITR contains the count value for 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

PITM[7:0] — Periodic Interrupt Timing Modulus

This 8-bit timing modulus is used to determine periodic interrupt rate. Use the following expression to calculate timer period.

$$\text{PIT Period} = [(\text{PIT Modulus})(\text{Prescaler value})(4)]/\text{System Clock Frequency}$$

D.5.18 SWSR — Software Service Register

\$YFFA27

15	8	7							0
NOT USED								SWSR	
RESET:									
								0	0
								0	0
								0	0
								0	0
								0	0
								0	0
								0	0

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

D.5.19 PORTFE — Port F Edge-Detect Flag Register

\$YFFA29

15	8	7	6	5	4	3	2	1	0						
NOT USED								EF7	EF6	EF5	EF4	EF3	EF2	EF1	EF0
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.

D.5.20 PFIVR — Port F Edge-Detect Interrupt Vector Register

\$YFFA2B

15	8	7	6	5	4	3	2	1	0						
NOT USED								PFIVR7	PFIVR6	PFIVR5	PFIVR4	PFIVR3	PFIVR2	PFIVR1	PFIVR0
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.

D.5.21 PFLVR — Port F Edge-Detect Interrupt Level Register

\$YFFA2D

15	8	7	6	5	4	3	2	1	0						
NOT USED								0	0	0	0	0	PFLV2	PFLV1	PFLV0
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.

D.5.22 TSTMSRA — Master Shift Register A

\$YFFA30

Register is used for factory test only.

D

D.5.23 TSTMSRB — Master Shift Register B **\$YFFA32**

Register is used for factory test only.

D.5.24 TSTSC — Test Module Shift Count **\$YFFA34**

Register is used for factory test only.

D.5.25 TSTRC — Test Module Repetition Count **\$YFFA36**

Register is used for factory test only.

D.5.26 CREG — Test Submodule Control Register **\$YFFA38**

Register is used for factory test only.

D.5.27 DREG — Distributed Register **\$YFFA3A**

Register is used for factory test only.

D.5.28 PORTC — Port C Data Register **\$YFFA41**

15	8	7	6	5	4	3	2	1	0	
NOT USED			0	PC6	PC5	PC4	PC3	PC2	PC1	PC0
RESET:										
			0	1	1	1	1	1	1	1

The state of bits in PORTC determines the state of pins programmed as port C discrete outputs. When a pin is assigned as a discrete output, the value in this register appears at the output. FC[6:0] correspond to pins CS[9:3]. This is a read/write register. Bit 7 is not used. Writing to this bit has no effect and it always reads zero.

D.5.29 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[6]	CSPA0[5]	CSPA0[4]	CSPA0[3]	CSPA0[2]	CSPA0[1]	CSPA0[0]	CSPA0[0]	CSPA0[1]	CSPA0[2]	CSPA0[3]	CSPA0[4]	CSPA0[5]	CSPA0[6]
RESET:															
0	0	0	0	0	0	0	0	0	1	0	1	1	1	1	1
0	0	1	0	0	1	1	0	0	1	0	1	1	0	1	0
0	0	DATA2	1	0	1	DATA2	1	DATA10	1	DATA10	1	DATA2	1	1	DATA0

Contains seven 2-bit fields, CSPAR0[5:1], and CSBOOT that determine the functions of corresponding chip-select pins. CSPAR0[15:14] are not used. These bits always read zero; write has no effect. CSPAR0 bit 1 always reads one; writes to CSPAR0 bit 1 have no effect. The following table shows alternate functions that can be enabled by data bus mode selection during reset.



CSPAR0 Pin Assignments

CSPAR0 Field	CSPAR0 Signal	Alternate Signal
CSPA0[6]	$\overline{CS5}$	FC2
CSPA0[5]	—	FC1
CSPA0[4]	$\overline{CS3}$	FC0
CSPA0[3]	$\overline{CS2}$	\overline{BGACK}
CSPA0[2]	$\overline{CS1}$	\overline{BG}
CSPA0[1]	$\overline{CS0}$	\overline{BR}
\overline{CSBOOT}	\overline{CSBOOT}	—

D.5.30 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[4]	CSPA1[3]	CSPA1[2]	CSPA1[1]	CSPA1[0]					
RESET:															
0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0
0	0	0	0	0	0	1	0	1	0	1	0	1	0	1	0
0	0	0	0	0	0	DATA7	1	DATA6	1	DATA5	1	DATA4	1	DATA3	1

Contains five 2-bit fields, CSPAR1[4:0], that determine the functions of corresponding chip-select pins. CSPAR1[15:10] are not used. These bits always read zero; write has no effect. The following table shows alternate functions that can be enabled by data bus mode selection during reset.

CSPAR1 Pin Assignments

CSPAR1 Field	CSPAR1 Signal	Alternate Signal
CSPA1[4]	$\overline{CS10}$	ADDR23
CSPA1[3]	$\overline{CS9}$	ADDR22
CSPA1[2]	$\overline{CS8}$	ADDR21
CSPA1[1]	$\overline{CS7}$	ADDR20
CSPA1[0]	$\overline{CS6}$	ADDR19

Pin Assignment Field Encoding

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

*Does not apply to the CSBOOT field

D.5.31 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		

RESET:

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

D.5.32 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		

RESET:

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

Each chip-select pin has an associated base address register. A base address is the lowest address in the block of addresses enabled by a chip select. ADDR[23:20] are driven to the same logic level as ADDR19. CSBARBT contains the base address for selection of a bootstrap peripheral memory device. Bit and field definition for CSBARBT and CSBAR[0:10] are the same, but reset block sizes differ.

ADDR[23:11] — Base Address

This field sets the starting address of a particular address space.

BLKSZ — Block Size

This field determines the size of the block above the base address that is enabled by the chip select.

Block Size 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]

ADDR[23:20] are driven to the same logic level as ADDR 19.



D.5.33 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	R/W	STRB	DSACK				SPACE		IPL			AVEC		
RESET:															
0	1	1	1	1	0	1	1	0	1	1	1	0	0	0	0

D.5.34 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	R/W	STRB	DSACK				SPACE		IPL			AVEC		
RESET:															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Contain parameters that support bootstrap operations from peripheral memory devices. Bit and field definitions for CSORBT and CSOR[0:10] are the same.

MODE — Asynchronous/Synchronous Mode

Synchronous mode cannot be used with internally-generated autovectors.

0 = Asynchronous mode selected

1 = Synchronous mode selected

BYTE — Upper/Lower Byte Option

The value in this field determines whether a select signal can be asserted.

R/W — 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.

STRB — Address Strobe/Data Strobe

1 = Data strobe

0 = Address strobe

DSACK — Data Strobe Acknowledge

This field specifies the source of $\overline{\text{DSACK}}$ in asynchronous mode and controls wait state insertion.

SPACE — Address Space Select

This field selects an address space to be used by the chip-select logic.

IPL — Interrupt Priority Level

This field determines interrupt priority level when a chip-select is used for interrupt acknowledge. It does not affect CPU interrupt recognition.

AVEC — Autovector Enable

Do not enable autovector support when in synchronous mode.

1 = Autovector enabled

0 = External interrupt vector enabled

Option Register Function Summary

MODE	BYTE	R/W	STRB	DSACK	SPACE	IPL	AVEC
0 = ASYNC	00 = Disable	00 = Rsvd	0 = AS	0000 = 0 WAIT	00 = CPU SP	000 = All	0 = Off
1 = SYNC	01 = Lower	01 = Read	1 = DS	0001 = 1 WAIT	01 = User SP	001 = Priority 1	1 = On
	10 = Upper	10 = Write		0010 = 2 WAIT	10 = Supv SP	010 = Priority 2	
	11 = Both	11 = Both		0011 = 3 WAIT	11 = S/U SP	011 = Priority 3	
				0100 = 4 WAIT		100 = Priority 4	
				0101 = 5 WAIT		101 = Priority 5	
				0110 = 6 WAIT		110 = Priority 6	
				0111 = 7 WAIT		111 = Priority 7	
				1000 = 8 WAIT	01 = User	000 = Data/Prog	
				1001 = 9 WAIT	10 = Supv	001 = Data Sp	
				1010 = 10 WAIT	11 = S/U	010 = Prog Sp	
				1011 = 11 WAIT		011 = Reserved	
				1100 = 12 WAIT		100 = Reserved	
				1101 = 13 WAIT		101 = Data Sp	
				1110 = F term		100 = Prog Sp	
				1111 = External		111 = Reserved	



D.6 Standby RAM with TPU Emulation

Table D–6. Standby RAM Control Register Address Map

Address	15	8	7	0
\$YFFB00	RAM MODULE CONFIGURATION REGISTER (TRAMMCR)			
\$YFFB02	RAM TEST REGISTER (TRAMTST)			
\$YFFB04	RAM BASE ADDRESS REGISTER (TRAMBAR)			
\$YFFB06– \$YFFB3F	NOT IMPLEMENTED			

Y = M111, where M is the state of the MM bit in SCIMCR. In an M68HC16 system, M must always be set to one.

D.6.1 TRAMMCR — TPU RAM Module Configuration Register \$YFFB00

15	11	9	8	7	6	5	4	3	2	1	0
STOP	0	0	0	PDS	0	0	RASP	NOT USED			

RESET:

U

U

U

TRAMMCR is used to determine whether the TPURAM is in STOP mode or normal mode. It is also used to determine in which space the array resides, and controls access to the base array registers. Reads of unimplemented bits always return zeros. Writes do not affect unimplemented bits.

STOP — Stop Control

0 = TPURAM array operates normally.

1 = TPURAM array enters low-power stop mode.

This bit determines whether the TPURAM array is in low-power stop mode. Reset state is one, leaving the array configured for low-power stop 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.

PDS — Standby Power Status Bit

0 = Loss of standby power.

1 = No loss of standby power.

The RAM array can be powered by a standby power source (V_{STBY}) while V_{DD} to the microcontroller is turned off. PDS indicates when V_{STBY} has fallen below a reference level for a specified period of time. To detect power loss, software must first set PDS, then monitor the state of PDS during normal operation and following reset.

RASP[1:0] — TPURAM Array Space

RASP limits access to the TPURAM array in microcontrollers that support separate user and supervisor operating modes. Because the CPU16 operates in supervisor mode only, RASP1 has no effect.

RASP	Space
X0	Program and Data
X1	Program

D.6.2 TRAMTST — TPURAM Test Register

\$YFFB02

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

D.6.3 TRAMBAR — TPURAM Array Base Address Register High

\$YFFB04

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	NOT USED	RAMDS	
0	0	0	0	0	0	0	0	0	0	0	0	0			1

RESET:

TRAMBAR specifies an array base address in the system memory map, which prevents accidental remapping of the array. TRAMBAR can be written only once after reset.

ADDR[23:11] — TPURAM Array Base Address Field

Specifies bits [23:11] of the array base address. To be accessed, the array must be enabled. Because ADDR[23:20] are driven to the same logic state as ADDR19, addresses in the range from \$080000 to \$F7FFFF cannot be accessed.

RAMDS — RAM Array Disable Status

Indicates whether the array is active or disabled. The array is disabled after reset. Writing a valid base address into TRAMBAR automatically clears RAMDS and enables the array.

0 = TPURAM array enabled

1 = TPURAM array disabled

D.7 Multichannel Communications Interface

Table D–7. MCCI Address Map

Address	15	8	7	0
\$YFFC00	MCCI MODULE CONFIGURATION (MMCR)			
\$YFFC02	MCCI TEST (MTEST)			
\$YFFC04	SCI INTERRUPT (ILSCI)		SCI INTERRUPT VECTOR (MIVR)	
\$YFFC06	SPI INTERRUPT (ILSPI)		RESERVED	
\$YFFC08	RESERVED		MCCI PIN ASSIGNMENT (PMCPAR)	
\$YFFC0A	RESERVED		MCCI DATA DIRECTION (DDRMC)	
\$YFFC0C	RESERVED		MCCI PORT DATA (PORTMC)	
\$YFFC0E	RESERVED		MCCI PORT PIN STATE (PORTMCP)	
\$YFFC10	RESERVED			
\$YFFC12	RESERVED			
\$YFFC14	RESERVED			
\$YFFC16	RESERVED			
\$YFFC18	SCIA CONTROL 0 (SCCR0A)			
\$YFFC1A	SCIA CONTROL 1 (SCCR1A)			
\$YFFC1C	SCIA STATUS (SCSRA)			
\$YFFC1E	SCIA DATA (SCDRA)			
\$YFFC20	RESERVED			
\$YFFC22	RESERVED			
\$YFFC24	RESERVED			
\$YFFC26	RESERVED			
\$YFFC28	SCIB CONTROL 0 (SCCR0B)			
\$YFFC2A	SCIB CONTROL 1 (SCCR1B)			
\$YFFC2C	SCIB STATUS (SCSRB)			
\$YFFC2E	SCIB DATA (SCDRB)			
\$YFFC30	RESERVED			
\$YFFC32	RESERVED			
\$YFFC34	RESERVED			
\$YFFC36	RESERVED			
\$YFFC38	SPI CONTROL (SPCR)			
\$YFFC3A	RESERVED			
\$YFFC3C	SPI STATUS (SPSR)			
\$YFFC3E	SPI DATA (SPDR)			

Y = M111, where M is the state of the MM bit in SCIMCR. In an M68HC16 system, M must always be set to one.

D.7.1 MMCR — MCCI Configuration Register

\$YFFC00

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

STOP — Stop Enable

- 0 = Normal MCCI clock operation
- 1 = MCCI clock operation stopped

STOP places the MCCI into a low power state by disabling the system clock in most parts of the module. MMCR is the only register guaranteed to be readable while STOP is asserted. STOP can be negated by the CPU and by reset.

SUPV — Supervisor/Unrestricted

- 0 = Unrestricted access
- 1 = Supervisor access

In systems with controlled access levels, SUPV places assignable registers in either supervisor-only data space or unrestricted data space. All MCCI registers reside in supervisor-only space. Because the CPU16 operates only in supervisor mode, SUPV has no meaning.

IARB — Interrupt Arbitration Identification Number

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. 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.

D.7.2 MTEST — MCCI Test

\$YFFC02

MTEST is used with SCIM test functions during factory test of the MCCI. Accesses to MTEST must be made while the MCU is in test mode.

D.7.3 ILSCI/MIVR — SCI Interrupt Request Level/MCCI Interrupt Vector **\$YFFC04**

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

ILSCI determines the priority level of interrupts requested by each SCI. Separate fields hold interrupt priority values for SCIA and SCIB. Priority is used to determine which interrupt is serviced first when two or more modules or external peripherals simultaneously request an interrupt.

ILSCIA, ILSCIB — Interrupt Level for SCIA, SCIB

ILSCIA and ILSCIB determine the priority levels of SCIA and SCIB interrupts, respectively. This field must contain a value between \$1 (lowest priority) and \$7 (highest priority) for interrupts to be recognized.

INTV[7:2] — MCCI Interrupt Vector Number (User-defined)

INTV[1:0] — MCCI Interrupt Vector Number (Generated by MCCI)

The value in the INTV fields specifies the interrupt vector number used to service an MCCI interrupt. At reset, MIVR is initialized to \$0F, the uninitialized interrupt vector number. For interrupts to be serviced, INTV[7:2] must be assigned a value corresponding to the six MSB of one of the user-defined vectors in the exception vector table. The values of INTV[1:0] are supplied by the MCCI, depending on the source of an MCCI interrupt request (%00 for SCIA, %01 for SCIB, and %10 for the SPI). Note that this arrangement requires that three adjacent vectors in the vector assignment table be assigned to MCCI service.

D.7.4 ILSPI — SPI Interrupt Level **\$YFFC06**

15	14	13	12	11	10	9	8	7	0							
0	0	ILSPI				0	0	0	RESERVED							
RESET:																
0	0	0	0	0	0	0	0	0								

ILSPI determines the priority of interrupts requested by the SPI. The ILSPI field must contain a value between \$1 (lowest priority) and \$7 (highest priority) for interrupts to be recognized. If ILSPI, ILSCIA, and ILSCIB are the same, simultaneous interrupt requests are recognized in SPI, SCIA, SCIB priority.

D.7.5 PORTMC — MCCI Port Data Register **\$YFFC0C**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								PMC7	PMC6	PMC5	PMC4	PMC3	PMC2	PMC1	PMC0

Writes to PORTMC are stored in an internal data latch. If any bit of PORTMC is configured as discrete output, the latched value is driven onto the corresponding pin. Reads of PORTMC return the value of the pin only if the pin is configured as a discrete input. Otherwise, the value read is the latched value. To avoid driving undefined data, first write a byte to PORTMC, then configure DDRMC.

D

D.7.6 PORTMCP — MCCI Port Pin State Register

\$YFFC0E

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								PMC7	PMC6	PMC5	PMC4	PMC3	PMC2	PMC1	PMC0

Reads of PORTMCP always return the state of the pins regardless of whether the pins are configured as input or output. Writes to PORTMCP have no effect.

D.7.7 PMCPAR — MCCI Pin Assignment Register

\$YFFC08

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								0	0	0	0	PMCPA3	0	PMCPA1	PMCPA0

RESET:

0 0 0 0 0 0 0 0

Setting a bit in PMCPAR assigns SPI pins for use as general-purpose I/O. SPI pins designated by PMCPAR as general-purpose I/O are controlled only by DDRMC and PORTMC; the SPI has no effect on these pins. PMCPAR does not affect the operation of the SCI submodule.

D.7.8 DDRMC — MCCI Data Direction Register

\$YFFC0B

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								DDM7	DDM6	DDM5	DDM4	DDM3	DDM2	DDM1	DDM0

RESET:

0 0 0 0 0 0 0 0

DDRMC determines whether a general-purpose I/O pin is an input or an output. During reset, all MCCI pins are configured as general-purpose inputs. Clearing a bit makes the pin an input; setting a bit makes it an output.

MCCI Pin Control

PMCPAR Bit	DDRMC Bit	Port MC Signal	MCCI Pin
—	DDM7	PMC7	TXDA
—	DDM6	PMC6	FXDA
—	DDM5	PMC5	TXDB
—	DDM4	PMC4	FXDB
PMCPA3	DDM3	PMC3	SS
—	DDM2	PMC2	SCK
PMCPA1	DDM1	PMC1	MOSI
PMCPA0	DDM0	PMC0	MISO

D.7.9 SCCR0A, SCCR0B — SCI Control Register 0**\$YFFC18, \$YFFC28**

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

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

SCBR — Baud Rate

SCI baud rate is programmed by writing a 13-bit value to this field. Writing a value of zero to SCBR disables the baud rate generator.

$$\text{SCI Baud Rate} = \frac{\text{System Clock}}{32\text{SCBR}}$$

where SCBR is in the range {1, 2, 3, ..., 8191}.

D.7.10 SCCR1A, SCCR1B — SCI Control Register 1**\$YFFC1A, \$YFFC2A**

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

Each 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 (receiver status bits) or writing (transmitter status bits) SCDR.

SCCR1A/B15 — 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. The following table lists the available choices.

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 interrupts inhibited

1 = SCI RDRF interrupts 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 can be used as general-purpose 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 in progress when TE is cleared.

RE — Receiver Enable

0 = SCI receiver disabled (status bits inhibited; RXD pin can be used as general-purpose I/O)

1 = SCI receiver enabled (RXD pin dedicated to SCI)

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 transmits 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.

D.7.11 SCSRA, SCSRB — SCI Status Register

\$YFFC1C, \$YFFC2C

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

Each SCSR contains flags that show SCI operational conditions. These flags can be cleared either by hardware or by a special acknowledgement 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, and SCDR must be written or read before the status bit is cleared.

Reading either byte of SCSR causes all 16 bits to be accessed, and any status bit already set in either byte is 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 register TDR.

TDRE is set when the byte in register TDR is transferred to the transmit serial shifter. If TDRE is zero, transfer has not occurred and a write to TDR overwrites the previous value. If TDR is written before TDRE has been cleared, new data is not transmitted.

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 (TDR) of SCDR.

RDRF — Receive Data Register Full Flag

0 = Register RDR is empty or contains previously read data.

1 = Register RDR contains new data.

RDRF is set when the content of the receive serial shifter is transferred to the RDR. 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 is 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 RDR, and RDRF is still set. Data transfer is inhibited until OR is cleared. Previous data in RDR 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 all three samples are not 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

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

0 = No framing error 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 when the stop bit is expected.

PF — Parity Error Flag

1 = Parity error occurred on the received data.

0 = No parity error 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.

D.7.12 SCDRA, SCDRB — SCI Data Register

\$YFFC1E, \$YFFC2E

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
NOT USED								R8/T8	R7/T7	R6/T6	R5/T5	R4/T4	R3/T3	R2/T2	R1/T1	R0/T0
RESET:																
								U	U	U	U	U	U	U	U	U

Each SCDR consists of two data registers at the same address. RDR is a read-only register that contains data received by the SCI serial interface. The data comes into the receive serial shifter and is transferred to RDR. TDR is a write-only register that contains data to be transmitted. The data is first written to TDR, 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 the SCI is configured for 8-bit operation, R8/T8 have no meaning or effect.

D.7.13 SPCR — SPI Control Register

\$YFFC38

15	14	13	12	11	10	9	8	7								0	
SPIE	SPE	WOMP	MSTR	CPOL	CPHA	LSBF	SIZE	SPBR									
RESET:																	
0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0

SPCR contains parameters for configuring the SPI. The CPU has read and write access to all control bits, but the MCCI has read access only to all bits except SPE. Writing a new value to SPCR while the SPI is enabled disrupts operation. Writing the same value into SPCR while the SPI is enabled has no effect on SPI operation.

SPIE — SPI Interrupt Enable

- 0 = SPI interrupts disabled
- 1 = SPI interrupts enabled

SPE — SPI Enable

- 0 = SPI is disabled. SPI pins can be used for general-purpose I/O.
- 1 = SPI is enabled. Pins allocated by PMCPAR are controlled by the SPI.

WOMP — Wired-OR Mode for SPI Pins

- 0 = Outputs have normal MOS drivers.
- 1 = Pins designated for output by DDRMC have open-drain drivers.

WOMP allows SPI pins to be connected for wired-OR operation, regardless of whether they are used for general-purpose output or for SPI output. WOMP affects the pins whether the SPI is enabled or disabled.

MSTR — Master/Slave Mode Select

0 = SPI is a slave device and only responds to externally generated serial data.

1 = SPI is system master and can initiate transmission to external SPI devices.

MSTR configures the SPI for either master or slave mode operation. This bit is cleared on reset and can only be written by the CPU.

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 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.

LSBF — Least Significant Bit First

0 = Serial data transfer starts with MSB

1 = Serial data transfer starts with LSB

SIZE — Transfer Data Size

0 = 8-bit data transfer

1 = 16-bit data transfer

SPBR — Serial Clock Baud Rate

The SPI 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 field. Giving SPBR a value of zero or one disables the baud rate generator. The following equations determine the SCK baud rate:

$$\text{SCK Baud Rate} = \frac{\text{System Clock}}{2\text{SPBR}}$$

or

$$\text{SPBR} = \frac{\text{System Clock}}{2(\text{SCK Baud Rate Desired})}$$

D.7.14 SPSR — SPI Status Register

\$YFFC3C

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SPIF	WCOL	0	MODF	0	0	0	0	0	0	0	0	0	0	0	0
RESET:															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

SPIF — SPI Finished Flag

- 0 = SPI not finished
- 1 = SPI finished

WCOL — Write Collision

- 0 = No write collision occurred
- 1 = Write collision occurred

MODF — Mode Fault Flag

- 0 = Normal operation
- 1 = Another SPI node requested to become the network SPI master while the SPI was enabled in master mode (\overline{SS} input taken low).

D.7.15 SPDR — SPI Data Register

\$YFFC3E

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

A write to SPDR initiates transmission or reception in the master device. At the completion of transmission, the SPIF status bit is set in both master and slave devices. Received data is buffered. SPIF must be cleared before a subsequent transfer of data from the shift register to the buffer or overrun occurs and the byte or word that causes overrun is lost. Transmitted data is not buffered. A write to SPDR places data directly into the shift register for transmission.

UPPB — Upper Byte

In 16-bit transfer mode, UPPB is used to access the most significant eight bits of the data. Bit 15 of the SPDR is the MSB of the 16-bit data.

LOWB — Lower Byte

In 8-bit transfer mode, data is accessed at the address of LOWB. MSB in 8-bit transfer mode is bit 7 of the SPDR. In 16-bit transfer mode, LOWB holds the least significant eight bits of the data.



D.8 Time Processor Unit

Table D-8. TPU Address Map

Address	15	8	7	0
\$YFFE00	TPU MODULE CONFIGURATION (TPUMCR)			
\$YFFE02	TEST CONFIGURATION (TCR)			
\$YFFE04	DEVELOPMENT SUPPORT CONTROL (DSCR)			
\$YFFE06	DEVELOPMENT SUPPORT STATUS (DSSR)			
\$YFFE08	TPU INTERRUPT CONFIGURATION (TICR)			
\$YFFE0A	CHANNEL INTERRUPT ENABLE (CIER)			
\$YFFE0C	CHANNEL FUNCTION SELECTION 0 (CFSR0)			
\$YFFE0E	CHANNEL FUNCTION SELECTION 1 (CFSR1)			
\$YFFE10	CHANNEL FUNCTION SELECTION 2 (CFSR2)			
\$YFFE12	CHANNEL FUNCTION SELECTION 3 (CFSR3)			
\$YFFE14	HOST SEQUENCE 0 (HSQR0)			
\$YFFE16	HOST SEQUENCE 1 (HSQR1)			
\$YFFE18	HOST SERVICE REQUEST 0 (HSRR0)			
\$YFFE1A	HOST SERVICE REQUEST 1 (HSRR1)			
\$YFFE1C	CHANNEL PRIORITY 0 (CPR0)			
\$YFFE1E	CHANNEL PRIORITY 1 (CPR1)			
\$YFFE20	CHANNEL INTERRUPT STATUS (CISR)			
\$YFFE22	LINK (LR)			
\$YFFE24	SERVICE GRANT LATCH (SGLR)			
\$YFFE26	DECODED CHANNEL NUMBER (DCNR)			

Y = M111, where M is the state of the MM bit in SCIMCR. In an M68HC16 system, M must always be set to one.

D.8.1 TPUMCR — TPU Module Configuration Register

\$YFFE00

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STOP	TCR1P	TCR2P	EMU	T2CG	STF	SUPV	PSCK	0	0	IARB					
RESET:															
0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

STOP — Stop Bit

0 = Internal clocks not shut down (reset condition)

1 = Internal clocks shut down

TCR1P — TCR1 Prescaler Control

Writing a value to this field sets a prescaler following the PSCK MUX for the indicated values.

TCR2P — TCR2 Prescaler Control

Writing a value to this field sets a prescaler following the T2CG MUX for the indicated values.

TCRP Value	Divide Ratio
00	1
01	2
10	4
11	8

EMU — Emulation Control

- 0 = TPU and RAM not in emulation mode (reset condition)
- 1 = TPU and RAM in emulation mode

T2CG — TCR2 Clock/Gate Control

- 0 = TCR2 pin used as clock source for TCR2 (reset condition)
- 1 = TCR2 pin used as gate of DIV8 clock for TCR2

STF — Stop Flag

- 0 = TPU operating (reset condition)
- 1 = TPU stopped

SUPV — Supervisor Data Space

- 0 = Assignable registers are unrestricted (FC2 is ignored).
- 1 = Assignable registers are restricted (FC2 is decoded; reset condition).

PSCK — Prescaler Clock

- 0 = DIV32 (system clock/32) is input to TCR1 prescaler.
- 1 = DIV4 (system clock/4) is input to TCR1 prescaler.

IARB — Interrupt Arbitration ID Bits

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. 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.

D.8.2 TCR — Test Configuration Register

\$YFFE02

This register is used for Motorola factory test only.

D.8.3 DSCR — Development Support Control Register

\$YFFE04

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
HOT4	NOT USED				BLC	CLKS	FRZ		CCL	BP	BC	BH	BL	BM	BT	
RESET:																
0					0		0		0		0		0		0	

HOT4 — Hang on T4

- 0 = Exit wait on T4 state caused by assertion of HOT4
- 1 = Enter wait on T4 state

BLC — Branch Latch Control

- 1 = Do not latch conditions into branch condition register before exiting the halted state or during the time-slot transition period.
- 0 = Latch conditions into branch condition register prior to exiting halted state.

CLKS — Stop Clocks (to TCRs)

- 0 = Do not stop TCRs.
- 1 = Stop TCRs during the halted state.

FRZ[1:0] — IMB FREEZE Response

The FRZ bits specify the TPU microengine response to the FREEZE signal.

FRZ[1:0]	TPU Response
00	Ignore Freeze
01	Reserved
10	Freeze at End of Current Microcycle
11	Freeze at Next Time-Slot Boundary

CCL — Channel Conditions Latch

CCL controls the latching of channel conditions (MRL and TDL) when CHAN is written.

- 0 = Only the pin state condition of the new channel is latched as a result of the write CHAN register microinstruction.
- 1 = Pin state, MRL, and TDL conditions of the new channel are latched as a result of a write CHAN register microinstruction.

BP, BC, BH, BL, BM, and BT — Breakpoint Enable Bits

DSCR[5:0] are TPU breakpoint enables. Setting a bit enables a breakpoint condition.

- BP — Break if μ PC equals μ PC breakpoint register.
- BC — Break if CHAN register equals channel breakpoint register at beginning of state or when CHAN is changed through microcode.
- BH — Break if host service latch is asserted at beginning of state.
- BL — Break if link service latch is asserted at beginning of state.
- BM — Break if MRL is asserted at beginning of state.
- BT — Break if TDL is asserted at beginning of state.

D

D.8.4 DSSR — Development Support Status Register

\$YFFE06

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NOT USED							BKPT	PCBK	CHBK	SRBK	TPUF	NOT USED			
RESET:															
							0	0	0	0	0				

BKPT — Breakpoint Asserted Flag

If an internal breakpoint caused the TPU to enter the halted state, the TPU asserts the BKPT signal on the IMB and the BKPT flag. The TPU continues to assert BKPT until it recognizes a breakpoint acknowledge cycle from a host, or until the FREEZE signal on the IMB is asserted.

PCBK — μ PC Breakpoint Flag

PCBK is asserted if a breakpoint occurs because of a μ PC register match with the μ PC breakpoint register. PCBK is negated when the BKPT flag is negated.

CHBK — Channel Register Breakpoint Flag

CHBK is asserted if a breakpoint occurs because of a CHAN register match with the channel register breakpoint register. CHBK is negated when the BKPT flag is negated.

SRBK — Service Request Breakpoint Flag

SRBK is asserted if a breakpoint occurs because of any of the service request latches being asserted along with their corresponding enable flag in the development support control register. SRBK is negated when the BKPT flag is negated.

TPUF — TPU FREEZE Flag

TPUF is asserted whenever the TPU is in a halted state as a result of FREEZE being asserted. This flag is automatically negated when the TPU exits the halted state because of FREEZE being negated.

D.8.5 TICR — TPU Interrupt Configuration Register

\$YFFE08

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

CIRL — Channel Interrupt Request Level

The interrupt request level for all channels is specified by this three-bit encoded field. Level seven for this field indicates a nonmaskable interrupt; level zero indicates that all channel interrupts are disabled.

CIBV — Channel Interrupt Base Vector

This field specifies the most significant nibble of all 16 TPU channel interrupt vector numbers.

D.8.6 CIER — Channel Interrupt Enable Register**\$YFFE0A**

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

CH[15:0] — Interrupt Enable/Disable for Each Channel

0 = Channel interrupts disabled

1 = Channel interrupts enabled

D.8.7 CFSR0–CFSR3 — Channel Function Select Registers**\$YFFE0C–\$YFFE12**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH (15) (11) (7) (3)				CH (14) (10) (6) (2)				CH (3) (9) (5) (1)				CH (12) (8) (4) (0)			
RESET:															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CFSR[15:0] — Encoded One of 16 Time Functions for each Channel

D.8.8 HSQR0 — Host Sequence Register 0**\$YFFE14**

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

D.8.9 HSQR1 — Host Sequence Register 1**\$YFFE16**

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

CH[15:0] — Encoded Host Sequence

D.8.10 HSRRO — Host Service Request Register 0**\$YFFE18**

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

D.8.11 HSRR1 — Host Service Request Register 1**\$YFFE1A**

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

CH[15:0] — Encoded Type of Host Service

CHX[1:0]	Service
00	No Host Service (Reset Condition)
01	Type 1 Host Service
10	Type 2 Host Service
11	Type 3 Host Service

D.8.12 CPR0 — Channel Priority Register 0**\$YFFE1C**

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

D.8.13 CPR1 — Channel Priority Register 1**\$YFFE1E**

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

CH[15:0] — Encoded One of Three Channel Priority Levels

CHX[1:0]	Service
00	Disabled
01	Low
10	Middle
11	High

D.8.14 CISR — Channel Interrupt Status Register**\$YFFE20**

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

CH[15:0] — Interrupt Status Bit

0 = Channel interrupt not asserted

1 = Channel interrupt asserted

Host Service Summary

Function Name	Function Code	Host Service Request Code	Host Sequence Code*
DIO Discrete Input/Output	\$8	1 = Force Output High 2 = Force Output Low 3 = Initialization, Input Specified 3 = Initialization, Periodic Input 3 = Update Pin Status Parameter	0 = Trans Mode — Record Pin on Transition 0 = Trans Mode — Record Pin on Transition 0 = Trans Mode — Record Pin on Transition 1 = Match Mode — Record Pin at Match_Rate 2 = Record Pin State on HSR 11
ITC Input Capture/ Input Transition Counter	\$A	0 = None 1 = Initialization 2 = (Not Implemented) 3 = (Not Implemented)	0 = No Link, Single Mode 1 = No Link, Continuous Mode 2 = Link, Single Mode 3 = Link, Continuous Mode
OC Output Compare	\$E	0 = None 1 = Host-Initiated Pulse Mode 2 = (Not Implemented) 3 = Continuous Pulse Mode	0 = Execute All Functions 1 = Execute All Functions 2 = Only Update TCRn Parameters 3 = Only Update TCRn Parameters
PWM Pulse-Width Modulation	\$9	0 = None 1 = Immediate Update Request 2 = Initialization 3 = (Not Implemented)	(None Implemented)
SPWM Synchronized Pulse- Width Modulation	\$7	0 = None 1 = (Not Implemented) 2 = Initialization 3 = Immediate Update Request	0 = Mode 0 1 = Mode 1 2 = Mode 2 3 = (Not Implemented)
PMA/PMM Period Measurement with Additional/Missing Transition Detect	\$B	0 = None 1 = Initialization 2 = (Not Implemented) 3 = (Not Implemented)	0 = PMA Bank Mode 1 = PMA Count Mode 2 = PMM Bank Mode 3 = PMM Count Mode
PSP Position-Synchronized Pulse Generator	\$C	0 = None 1 = Immediate Update Request 2 = Initialization 3 = Force Change	0 = Pulse Width Set by Angle 1 = Pulse Width Set by Time 2 = Pulse Width Set by Angle 3 = Pulse Width Set by Time
SM Stepper Motor	\$D	0 = None 1 = None 2 = Initialization 3 = Step Request	(None Implemented)
PPWA Period/Pulse-Width Accumulator	\$F	0 = None 1 = (Not Implemented) 2 = Initialization 3 = (Not Implemented)	0 = 24-Bit Period 1 = 16-Bit Period + Link 2 = 24-Bit Pulse Width 3 = 16-Bit Pulse Width + Link

D.8.15 LR — Link Register**\$YFFE22**

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

CH[15:0] — Test Mode Link Service Request Enable Bit

0 = Link bit not asserted

1 = Link bit asserted

D.8.16 SGLR — Service Grant Latch Register**\$YFFE24**

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

CH[15:0] — Service Granted Bits

D.8.17 DCNR — Decoded Channel Number Register**\$YFFE26**

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

CH[15:0] — Service Status Bits

Table D-9. MC68HC16Y1 Module Address Map

(Assumes SCIMCR MM = 1)

A/D Converter

Address	15	8	7	0
\$FFF700	MODULE CONFIGURATION (ADCMCR)			
\$FFF702	FACTORY TEST (ADCTST)			
\$FFF704	(RESERVED)			
\$FFF706	PORT ADA DATA (PORTADA)			
\$FFF708	(RESERVED)			
\$FFF70A	ADC CONTROL 0 (ADCTL0)			
\$FFF70C	ADC CONTROL 1 (ADCTL1)			
\$FFF70E	ADC STATUS (ADSTAT)			
\$FFF710	RIGHT-JUSTIFIED UNSIGNED RESULT 0 (RJURR0)			
\$FFF712	RIGHT-JUSTIFIED UNSIGNED RESULT 1 (RJURR1)			
\$FFF714	RIGHT-JUSTIFIED UNSIGNED RESULT 2 (RJURR2)			
\$FFF716	RIGHT-JUSTIFIED UNSIGNED RESULT 3 (RJURR3)			
\$FFF718	RIGHT-JUSTIFIED UNSIGNED RESULT 4 (RJURR4)			
\$FFF71A	RIGHT-JUSTIFIED UNSIGNED RESULT 5 (RJURR5)			
\$FFF71C	RIGHT-JUSTIFIED UNSIGNED RESULT 6 (RJURR6)			
\$FFF71E	RIGHT-JUSTIFIED UNSIGNED RESULT 7 (RJURR7)			
\$FFF720	LEFT-JUSTIFIED SIGNED RESULT 0 (LJSRR0)			
\$FFF722	LEFT-JUSTIFIED SIGNED RESULT 1 (LJSRR1)			
\$FFF724	LEFT-JUSTIFIED SIGNED RESULT 2 (LJSRR2)			
\$FFF726	LEFT-JUSTIFIED SIGNED RESULT 3 (LJSRR3)			
\$FFF728	LEFT-JUSTIFIED SIGNED RESULT 4 (LJSRR4)			
\$FFF72A	LEFT-JUSTIFIED SIGNED RESULT 5 (LJSRR5)			
\$FFF72C	LEFT-JUSTIFIED SIGNED RESULT 6 (LJSRR6)			
\$FFF72E	LEFT-JUSTIFIED SIGNED RESULT 7 (LJSRR7)			
\$FFF730	LEFT-JUSTIFIED UNSIGNED RESULT 0 (LJURR0)			
\$FFF732	LEFT-JUSTIFIED UNSIGNED RESULT 1 (LJURR1)			
\$FFF734	LEFT-JUSTIFIED UNSIGNED RESULT 2 (LJURR2)			
\$FFF736	LEFT-JUSTIFIED UNSIGNED RESULT 3 (LJURR3)			
\$FFF738	LEFT-JUSTIFIED UNSIGNED RESULT 4 (LJURR4)			
\$FFF73A	LEFT-JUSTIFIED UNSIGNED RESULT 5 (LJURR5)			
\$FFF73C	LEFT-JUSTIFIED UNSIGNED RESULT 6 (LJURR6)			
\$FFF73E	LEFT-JUSTIFIED UNSIGNED RESULT 7 (LJURR7)			

**Table D-9. MC68HC16Y1 Module Address Map
(Continued)**

Masked ROM Module

Address	15	8 7	0
\$FFF820	MASKED ROM MODULE CONFIGURATION (MRMCR)		
\$FFF822	NOT IMPLEMENTED		
\$FFF824	ARRAY BASE ADDRESS HIGH (ROMBAH)		
\$FFF826	ARRAY BASE ADDRESS LOW (ROMBAL)		
\$FFF828	ROM SIGNATURE HIGH (RSIGHI)		
\$FFF82A	ROM SIGNATURE LOW (RSIGLO)		
\$FFF82C	NOT IMPLEMENTED		
\$FFF82E	NOT IMPLEMENTED		
\$FFF830	ROM BOOTSTRAP WORD 0 (ROMBS0)		
\$FFF832	ROM BOOTSTRAP WORD 1 (ROMBS1)		
\$FFF834	ROM BOOTSTRAP WORD 2 (ROMBS2)		
\$FFF836	ROM BOOTSTRAP WORD 3 (ROMBS3)		
\$FFF838	NOT IMPLEMENTED		
\$FFF83A	NOT IMPLEMENTED		
\$FFF83C	NOT IMPLEMENTED		
\$FFF83E	NOT IMPLEMENTED		

**Table D–9. MC68HC16Y1 Module Address Map
(Continued)**

General-Purpose Timer

Address	15 8 7 0	
\$FFF900	GPT MODULE CONFIGURATION (GPTMCR)	
\$FFF902	(RESERVED FOR TEST)	
\$FFF904	INTERRUPT CONFIGURATION (ICR)	
\$FFFE06	PGP DATA DIRECTION (DDRGP)	PGP DATA (PORTGP)
\$FFF908	OC1 ACTION MASK (OC1M)	OC1 ACTION DATA (OC1D)
\$FFF90A	TIMER COUNTER (TCNT)	
\$FFF90C	PA CONTROL (PACTL)	PA COUNTER (PACNT)
\$FFF90E	INPUT CAPTURE 1 (TIC1)	
\$FFF910	INPUT CAPTURE 2 (TIC2)	
\$FFF912	INPUT CAPTURE 3 (TIC3)	
\$FFF914	OUTPUT COMPARE 1 (TOC1)	
\$FFF916	OUTPUT COMPARE 2 (TOC2)	
\$FFF918	OUTPUT COMPARE 3 (TOC3)	
\$FFF91A	OUTPUT COMPARE 4 (TOC4)	
\$FFF91C	INPUT CAPTURE 4/OUTPUT COMPARE 5 (TI4/O5)	
\$FFF91E	TIMER CONTROL 1 (TCTL1)	TIMER CONTROL 2 (TCTL2)
\$FFF920	TIMER MASK 1 (TMSK1)	TIMER MASK 2 (TMSK2)
\$FFF922	TIMER FLAG 1 (TFLG1)	TIMER FLAG 2 (TFLG2)
\$FFF924	FORCE COMPARE (CFORC)	PWM CONTROL C (PWMC)
\$FFF926	PWM CONTROL A (PWMA)	PWM CONTROL B (PWMB)
\$FFF928	PWM COUNT (PWMCNT)	
\$FFF92A	PWMA BUFFER (PWMBUFA)	PWMB BUFFER (PWMBUFB)
\$FFF92C	GPT PRESCALER (PRESCL)	
\$FFF92E– \$FFF93F	RESERVED	

**Table D-9. MC68HC16Y1 Module Address Map
(Continued)**

Single-Chip Integration Module

Address	15	8	7	0
\$FFFA00	SCIM MODULE CONFIGURATION (SCIMCR)			
\$FFFA02	FACTORY TEST (SCIMTR)			
\$FFFA04	CLOCK SYNTHESIZER CONTROL (SYNCR)			
\$FFFA06	UNUSED	RESET STATUS (RSR)		
\$FFFA08	MODULE TEST E (SCIMTRE)			
\$FFFA0A	PORT A DATA (PORTA)		PORT B DATA (PORTB)	
\$FFFA0C	PORT G DATA (PORTG)		PORT H DATA (PORTH)	
\$FFFA0E	PORT G DATA DIRECTION (DDRG)		PORT H DATA DIRECTION (DDRH)	
\$FFFA10	UNUSED		PORT E DATA (PORTE0)	
\$FFFA12	UNUSED		PORT E DATA (PORTE1)	
\$FFFA14	PORT A/B DATA DIRECTION (DDRAB)		PORT E DATA DIRECTION (DDRE)	
\$FFFA16	UNUSED		PORT E PIN ASSIGNMENT (PEPAR)	
\$FFFA18	UNUSED		PORT F DATA (PORTF0)	
\$FFFA1A	UNUSED		PORT F DATA (PORTF1)	
\$FFFA1C	UNUSED		PORT F DATA DIRECTION (DDRF)	
\$FFFA1E	UNUSED		PORT F PIN ASSIGNMENT (PFPAR)	
\$FFFA20	UNUSED		SYSTEM PROTECTION CONTROL (SYPCR)	
\$FFFA22	PERIODIC INTERRUPT CONTROL (PICR)			
\$FFFA24	PERIODIC INTERRUPT TIMING (PITR)			
\$FFFA26	UNUSED		SOFTWARE SERVICE (SWSR)	
\$FFFA28	UNUSED		PORT F EDGE-DETECT CONTROL (PORTFE)	
\$FFFA2A	UNUSED		PORT F EDGE-DETECT INTERRUPT VECTOR (PFIVR)	
\$FFFA2C	UNUSED		PORT F EDGE-DETECT INTERRUPT LEVEL (PFLVR)	
\$FFFA2E	UNUSED		UNUSED	
\$FFFA30	TEST MODULE MASTER SHIFT A (TSTMSRA)			
\$FFFA32	TEST MODULE MASTER SHIFT B (TSTMSRB)			
\$FFFA34	TEST MODULE SHIFT COUNT (TSTSC)			
\$FFFA36	TEST MODULE REPETITION COUNTER (TSTRC)			
\$FFFA38	TEST MODULE CONTROL (CREG)			
\$FFFA3A	TEST MODULE DISTRIBUTED (DREG)			
\$FFFA3C	UNUSED		UNUSED	
\$FFFA3E	UNUSED		UNUSED	
\$FFFA40	UNUSED		PORT C DATA (PORTC)	
\$FFFA42	UNUSED		UNUSED	
\$FFFA44	CHIP-SELECT PIN ASSIGNMENT (CSPAR0)			
\$FFFA46	CHIP-SELECT PIN ASSIGNMENT (CSPAR1)			
\$FFFA48	CHIP-SELECT BASE BOOT (CSBARBT)			



**Table D–9. MC68HC16Y1 Module Address Map
(Continued)**

Single-Chip Integration Module (Continued)

Address	15	8	7	0
\$FFFA4A	CHIP-SELECT OPTION BOOT (CSORBT)			
\$FFFA4C	CHIP-SELECT BASE 0 (CSBAR0)			
\$FFFA4E	CHIP-SELECT OPTION 0 (CSOR0)			
\$FFFA50	UNUSED			
\$FFFA52	UNUSED			
\$FFFA54	UNUSED			
\$FFFA56	UNUSED			
\$FFFA58	CHIP-SELECT BASE 3 (CSBAR3)			
\$FFFA5A	CHIP-SELECT OPTION 3 (CSOR3)			
\$FFFA5C	UNUSED			
\$FFFA5E	UNUSED			
\$FFFA60	CHIP-SELECT BASE 5 (CSBAR5)			
\$FFFA62	CHIP-SELECT OPTION 5 (CSOR5)			
\$FFFA64	CHIP-SELECT BASE 6 (CSBAR6)			
\$FFFA66	CHIP-SELECT OPTION 6 (CSOR6)			
\$FFFA68	CHIP-SELECT BASE 7 (CSBAR7)			
\$FFFA6A	CHIP-SELECT OPTION 7 (CSOR7)			
\$FFFA6C	CHIP-SELECT BASE 8 (CSBAR8)			
\$FFFA6E	CHIP-SELECT OPTION 8 (CSOR8)			
\$FFFA70	CHIP-SELECT BASE 9 (CSBAR9)			
\$FFFA72	CHIP-SELECT OPTION 9 (CSOR9)			
\$FFFA74	CHIP-SELECT BASE 10 (CSBAR10)			
\$FFFA76	CHIP-SELECT OPTION 10 (CSOR10)			
\$FFFA78	UNUSED			UNUSED
\$FFFA7A	UNUSED			UNUSED
\$FFFA7C	UNUSED			UNUSED
\$FFFA7E	UNUSED			UNUSED

Standby RAM with TPU Emulation

Address	15	0
\$FFFB00	RAM MODULE CONFIGURATION (TRAMMCR)	
\$FFFB02	RAM TEST (TRAMTST)	
\$FFFB04	RAM BASE ADDRESS AND STATUS (TRAMBAR)	
\$FFFB06– \$FFFB3F	NOT IMPLEMENTED	

**Table D-9. MC68HC16Y1 Module Address Map
(Continued)**

Multichannel Communications Interface			
Address	15	8	7
\$FFFC00	MCCI MODULE CONFIGURATION (MMCR)		
\$FFFC02	MCCI TEST (MTEST)		
\$FFFC04	SCI INTERRUPT (ILSCI)	SCI INTERRUPT VECTOR (MIVR)	
\$FFFC06	SPI INTERRUPT (ILSPI)	RESERVED	
\$FFFC08	RESERVED	MCCI PIN ASSIGNMENT (PMCPAR)	
\$FFFC0A	RESERVED	MCCI DATA DIRECTION (DDRMC)	
\$FFFC0C	RESERVED	MCCI PORT DATA (PORTMC)	
\$FFFC0E	RESERVED	MCCI PORT PIN STATE (PORTMCP)	
\$FFFC10	RESERVED		
\$FFFC12	RESERVED		
\$FFFC14	RESERVED		
\$FFFC16	RESERVED		
\$FFFC18	SCIA CONTROL 0 (SCCR0A)		
\$FFFC1A	SCIA CONTROL 1 (SCCR1A)		
\$FFFC1C	SCIA STATUS (SCSRA)		
\$FFFC1E	SCIA DATA (SCDRA)		
\$FFFC20	RESERVED		
\$FFFC22	RESERVED		
\$FFFC24	RESERVED		
\$FFFC26	RESERVED		
\$FFFC28	SCIB CONTROL 0 (SCCR0B)		
\$FFFC2A	SCIB CONTROL 1 (SCCR1B)		
\$FFFC2C	SCIB STATUS (SCSRB)		
\$FFFC2E	SCIB DATA (SCDRB)		
\$FFFC30	RESERVED		
\$FFFC32	RESERVED		
\$FFFC34	RESERVED		
\$FFFC36	RESERVED		
\$FFFC38	SPI CONTROL (SPCR)		
\$FFFC3A	RESERVED		
\$FFFC3C	SPI STATUS (SPSR)		
\$FFFC3E	SPI DATA (SPDR)		

**Table D-9. MC68HC16Y1 Module Address Map
(Continued)**

Time Processor Unit

Address	15	8	7	0
\$FFFE00	TPU MODULE CONFIGURATION (TPUMCR)			
\$FFFE02	TEST CONFIGURATION (TCR)			
\$FFFE04	DEVELOPMENT SUPPORT CONTROL (DSCR)			
\$FFFE06	DEVELOPMENT SUPPORT STATUS (DSSR)			
\$FFFE08	TPU INTERRUPT CONFIGURATION (TICR)			
\$FFFE0A	CHANNEL INTERRUPT ENABLE (CIER)			
\$FFFE0C	CHANNEL FUNCTION SELECTION 0 (CFSR0)			
\$FFFE0E	CHANNEL FUNCTION SELECTION 1 (CFSR1)			
\$FFFE10	CHANNEL FUNCTION SELECTION 2 (CFSR2)			
\$FFFE12	CHANNEL FUNCTION SELECTION 3 (CFSR3)			
\$FFFE14	HOST SEQUENCE 0 (HSQR0)			
\$FFFE16	HOST SEQUENCE 1 (HSQR1)			
\$FFFE18	HOST SERVICE REQUEST 0 (HSRR0)			
\$FFFE1A	HOST SERVICE REQUEST 1 (HSRR1)			
\$FFFE1C	CHANNEL PRIORITY 0 (CPR0)			
\$FFFE1E	CHANNEL PRIORITY 1 (CPR1)			
\$FFFE20	CHANNEL INTERRUPT STATUS (CISR)			
\$FFFE22	LINK (LR)			
\$FFFE24	SERVICE GRANT LATCH (SGLR)			
\$FFFE26	DECODED CHANNEL NUMBER (DCNR)			

TPU Parameter RAM

Channel Number	Base Address	Parameter							
		0	1	2	3	4	5	6	7
0	\$FFFF—	00	02	04	06	08	0A	—	—
1	\$FFFF—	10	12	14	16	18	1A	—	—
2	\$FFFF—	20	22	24	26	28	2A	—	—
3	\$FFFF—	30	32	34	36	38	3A	—	—
4	\$FFFF—	40	42	44	46	48	4A	—	—
5	\$FFFF—	50	52	54	56	58	5A	—	—
6	\$FFFF—	60	62	64	66	68	6A	—	—
7	\$FFFF—	70	72	74	76	78	7A	—	—
8	\$FFFF—	80	82	84	86	88	8A	—	—
9	\$FFFF—	90	92	94	96	98	9A	—	—
10	\$FFFF—	A0	A2	A4	A6	A8	AA	—	—
11	\$FFFF—	B0	B2	B4	B6	B8	BA	—	—
12	\$FFFF—	C0	C2	C4	C6	C8	CA	—	—
13	\$FFFF—	D0	D2	D4	D6	D8	DA	—	—
14	\$FFFF—	E0	E2	E4	E6	E8	EA	EC	EE
15	\$FFFF—	F0	F2	F4	F6	F8	FA	FC	FE

D

Table D–10. Register Bit and Field Mnemonics

Mnemonic	Name	Register Location
ABD	Address Bus Disable	SCIMCR
ADDR[23:11]	Chip-Select Base Address	CSBAR[0:10], CSBARBT
ADDR[23:16]	ROM Array Base Address	ROMBAH
ADDR[23:11]	RAM Array Base Address	TRAMBAR
ASPC	ROM Array Space Field	MRMCR
$\overline{\text{AVEC}}$	Autovector Enable	CSOR[0:10], CSORBT
BC	CHAN Breakpoint Enable	DSCR
BH	THSL Breakpoint Enable	DSCR
BKPT	Breakpoint Asserted Flag	DSSR
BL	LS Breakpoint Enable	DSCR
BLC	Branch Latch Control	DSCR
BLKSZ	Block Size	CSBAR[0:10], CSBARBT
BM	MRL Breakpoint Enable	DSCR
BME	Bus Monitor External Enable	SYPCR
BMT[1:0]	Bus Monitor Timing	SYPCR
$\overline{\text{BOOT}}$	Boot ROM Control	MRMCR
BP	PC Breakpoint Enable	DSCR
BT	TDL Breakpoint Enable	DSCR
BYTE	Upper/Lower Byte Option	CSOR[0:10], CSORBT
C	Carry Flag	CCR
CA	Channel Selection	ADCTL1
CB	Channel Selection	ADCTL1
CC	Channel Selection	ADCTL1
CCF[7:0]	Conversion Complete Flags	ADSTAT
CCL	Channel Conditions Latch	DSCR
CCTR[2:0]	Conversion Counter	ADSTAT
CD	Channel Selection	ADCTL1
CHBK	Channel Register Breakpoint Flag	DSSR
CH[15:0]	Channel Interrupt Enable/Disable	CIER
CH[15:0]	Channel Function Select	CFSR[0:3]
CH[15:8]	Host Sequence 0	HSQR0
CH[7:0]	Host Sequence 1	HSQR1
CH[15:8]	Host Service Request 0	HSRR0
CH[7:0]	Host Service Request 1	HSRR1
CH[15:8]	Channel Priority	CPR0
CH[7:0]	Channel Priority	CPR1
CH[15:0]	Channel Interrupt Status	CISR
CH[15:0]	Test Mode Link Request Enable	LR
CH[15:0]	Service Granted	SGLR
CH[15:0]	Service Status	DCNR

Table D–10. Register Bit and Field Mnemonics (Continued)

Mnemonic	Name	Register Location
CIBV	Channel Interrupt Base Vector	TICR
CIRL	Channel Interrupt Request Level	TICR
CLKS	Stop Clocks	DSCR
CPHA	Clock Phase	SPCR
CPOL	Clock Polarity	SPCR
CPROUT	Compare/Capture Unit Clock Output Enable	TMSK1/TMSK2
CPR[2:0]	Timer Prescaler/PCLK Select Field	TMSK1/TMSK2
CPUD	CPU Development Support Disable	SCIMCR
CSBOOT	Boot ROM Chip Select	CSPAR0
CSPA0[6:1]	Chip Select Pin Assignment Fields	CSPAR0
CSPA1[4:0]	Chip Select Pin Assignment Fields	CSPAR1
CSOR[0:10]	Chip Select Option Registers	CSOR[0:10], CSORBT
DBE	Double Bus Fault Enable	SYPCR
DDA	Port A Data Direction	DDRAB
DDB	Port B Data Direction	DDRAB
DDE[7:0]	Port E Data Direction	DDRE
DDF[7:0]	Port F Data Direction	DDRF
DDG[7:0]	Port G Data Direction	DDRG
DDH[7:0]	Port H Data Direction	DDRH
DDGP[7:0]	Port GP Data Direction	DDRGP/PORTGP
DDM[7:0]	Port MC Data Direction	PMCPAR
DSACK	Data Strobe Acknowledge	CSOR[0:10], CSORBT
EDGE[4:1]	Input Capture Edge Control	TCTL1/TCTL2
EF[7:0]	Port FE Edge-Detect Flag	PORTFE
EDIV	ECLK Divide Rate	SYNCR
EMU	Emulation Control	TMCR
EMUL	Emulator Mode Control	MRMCR
EV	Extension Bit Overflow Flag	CCR
EXOFF	External Clock Off	SCIMCR
EXT	External Reset	RSR
F1A	Force Logic Level One On PWMA	CFORC, PWMC
F1B	Force Logic Level One On PWMB	CFORC, PWMC
FC1	Function Code Pin 1	CSPAR0
FE	Framing Error	SCSRA, SCSRB
FOC[5:1]	Force Output Compare	CFORC, PWMC
FPWMA	Force PWMA Value	CFORC, PWMC
FPWMB	Force PWMB Value	CFORC, PWMC

Table D–10. Register Bit and Field Mnemonics (Continued)

Mnemonic	Name	Register Location
FRZ[1:0]	Freeze Response	DSCR
FRZ[1:0]	Freeze Response	ADCMCR
FRZ[1:0]	Freeze Response	GPTMCR
FRZBM	Freeze Bus Monitor Enable	SCIMCR
FRZW	Freeze Software Enable	SCIMCR
H	Half Carry Flag	CCR
HLT	Halt Monitor Reset	RSR
HOT4	Hang On T4	DSCR
I4/O5	Input Capture 4/Output Compare 5	PACTL/PACNT
I4/O5F	Input Capture 4/Output Compare 5 Flag	TFLG1/TFLG2
I4/O5I	I4/O5 Interrupt Enable	TMSK1/TMSK2
IARB[3:0]	Interrupt Arbitration	TMCR
IARB[3:0]	Interrupt Arbitration	SCIMCR
IARB[3:0]	Interrupt Arbitration	MMCR
IARB[3:0]	Interrupt Arbitration	GPTMCR
ICF[10:8]	Input Capture Flags	TFLG1/TFLG2
ICI[3:1]	Input Capture Interrupt Enable	TMSK1/TMSK2
IDLE	Idle-Line Detected	SCSRA, SCSRB
ILIE	Idle-Line Interrupt Enable	SCCR1A, SCCR1B
ILSCIA, ILSCIB	Interrupt Level For SCIA or SCIB	ILSCI/MIVR
ILSPI	SPI Interrupt Level Register	ILSPI
ILT	Idle-Line Detect Type	SCCR1A, SCCR1B
INCP	Increment Prescaler	GPTMCR
INTV[7:0]	Interrupt Vector Number	ILSCI/MIVR
IPA	Interrupt Priority Adjust	ICR
IPL	Interrupt Priority Level	CSOR[0:10], CSORBT
IPL	Interrupt Priority Level	ICR
IP[2:0]	Interrupt Priority Field	CCR
IRL	Interrupt Request Level	ICR
LJSRR	Signed Left-Justified Result	RSLT[0:7]
LJURR	Unsigned Left-Justified Result	RSLT[0:7]
LOC	Loss of Clock Reset	RSR
LOCK	Lock Registers	MRMCR
LOOPS	Loop Mode	SCCR1A, SCCR1B
LOSCD	Loss of Clock Oscillator Disable	SYNCR
LOWB	Lower Byte	SPDR
LSBF	Least Significant Bit First	SPCR
M	Mode Select	SCCR1A, SCCR1B
MIVR	MCCI Interrupt Vector	ILSCI/MIVR

Table D–10. Register Bit and Field Mnemonics (Continued)

Mnemonic	Name	Register Location
MM	Module Mapping	SCIMCR
MODE	Asynchronous/Synchronous Mode	CSOR[0:10], CSORBT
MODF	Mode Fault Flag	SPSR
MSTR	Master/Slave Mode Select	SPCR
MULT	Multichannel Conversion	ADCTL1
MV	Accumulator M Overflow Flag	CCR
N	Negative Flag	CCR
NF	Noise Error	SCSRA, SCSRB
OC1D[5:1]	OC1 Data	OC1M/OC1D
OC1M[5:1]	OC1 Mask	OC1M/OC1D
OCF[4:1]	Output Compare Flags	TFLG1/TFLG2
OCI[4:1]	Output Compare Interrupt Enable	TMSK1/TMSK2
OM/OL[5:2]	Output Compare Mode and Level Bits	TCTL1/TCTL2
OR	Overrun Error	SCSRA, SCSRB
PA[7:0]	Port A Data	PORTA
PA	Priority Adjust	ICR
PACLK[1:0]	Pulse Accumulator Clock Select	PACTL/PACNT
PACNT	Pulse Accumulator Counter	PACTL/PACNT
PADA[7:0]	Port ADA Data	PORTADA
PAEN	Pulse Accumulator Enable	PACTL/PACNT
PAIF	Pulse Accumulator Flag	TFLG1/TFLG2
PAII	Pulse Accumulator Input Interrupt Enable	TMSK1/TMSK2
PAIS	PAI Pin State (Read Only)	PACTL/PACNT
PAMOD	Pulse Accumulator Mode	PACTL/PACNT
PAOVF	Pulse Accumulator Overflow Flag	TFLG1/TFLG2
PAOVI	Pulse Accumulator Overflow Interrupt Enable	TMSK1/TMSK2
PB[7:0]	Port B Data	PORTB
PC[6:0]	Port C Data	PORTC
PCBK	Microprogram Counter Breakpoint Flag	DSSR
PDS	Standby Power Status	TRAMMCR
PCLKS	PCLK Pin State (Read Only)	PACTL/PACNT
PE	Parity Enable	SCCR1A, SCCR1B
PE[7:0]	Port E Data	PORTE
PEDGE	Pulse Accumulator Edge Control	PACTL/PACNT
PEPA[7:0]	Port E Pin Assignment	PEPAR
PF[7:0]	Port F Data	PORTF
PF	Parity Error Flag	SCSRA, SCSRB
PFIV[7:0]	Port F Edge-Detect Interrupt Vector	PFIVR
PFLV[2:0]	Port F Edge-Detect Interrupt	PFLVR

Table D–10. Register Bit and Field Mnemonics (Continued)

Mnemonic	Name	Register Location
PFFPA[7:0]	Port F Pin Assignment	PFFPAR
PG[7:0]	Port G Data	PORTG
PGP[7:0]	Port GP Data	PORTGP
PH[7:0]	Port H Data	PORTH
PIRQL[2:0]	Periodic Interrupt Request Level	PICR
PITM[7:0]	Periodic Interrupt Timing Modulus	PITR
PIV[7:0]	Periodic Interrupt Vector	PICR
PK[3:0]	Program Counter Address Extension Field	CCR
PMC[7:0]	Port MC Data	PORTMC
PMCP[7:0]	MCCI Port Pin State	PORTMCP
POW	Power-Up Reset	RSR
PPROUT	PWM Clock Output Enable	CFORC, PWMC
PPR[2:0]	PWM Prescaler/PCLK Select	CFORC, PWMC
PRS[4:0]	Prescaler Rate Selection	ADCTL0
PSCK	Prescaler Clock	TMCR
PT	Parity Type	SCCR1A, SCCR1B
PTP	Periodic Timer Prescaler Control	PITR
R[8:0]/T[8:0]	SCI Data Receive	SCDRA, SCDRB
RAF	Receiver Active	SCSRA, SCSRB
RAMDS	RAM Array Disable Status	TRAMBAR
RASP	RAM Array Space	TRAMMCR
RDRF	Receive Data Register Full	SCSRA, SCSRB
RE	Receiver Enable	SCCR1A, SCCR1B
RES10	10-Bit Resolution	ADCTL0
RIE	Receiver Interrupt Enable	SCCR1A, SCCR1B
RJURR	Unsigned Right-Justified Result	RSLT[0:7]
RSP[18:16]	ROM Signature High	RSIGHI
RSP[15:5]	ROM Signature Low	RSIGLO
RSTEN	Reset Enable	SYNCR
R \bar{W}	Read/Write	CSOR[0:10], CSORBT
RWD	Read/Write Disable	SCIMCR
RWU	Receiver Wakeup	SCCR1A, SCCR1B
S	Stop Enable	CCR
S8CM	Select 8-Conversion Sequence Mode	ADCTL1
SBK	Send Break	SCCR1A, SCCR1B
SCAN	Scan Mode Selection	ADCTL1



Table D–10. Register Bit and Field Mnemonics (Continued)

Mnemonic	Name	Register Location
SCBR	SCI Baud Rate	SCCR0A, SCCR0B
SCF	Sequence Complete Flag	ADSTAT
SFA	PWMA Slow/Fast Select	CFORC, PWMC
SFB	PWMB Slow/Fast Select	CFORC, PWMC
SHEN[9:8]	Show Cycle Enable	SCIMCR
SIZE	Transfer Data Size	SPCR
SLIMP	Limp Mode	SYNCR
SLOCK	Synthesizer Lock	SYNCR
SLVE	Factory Test Mode Enabled	SCIMCR
SM	Saturate Mode Bit	CCR
SPACE	Address Space Select	CSOR[0:10], CSORBT
SPBR	Serial Clock Baud Rate	SPCR
SPE	SPI Enable	SPCR
SPIE	SPI Finished Interrupt Enable	SPCR
SPIF	SPI Finished Flag	SPSR
SRBK	Service Request Breakpoint Flag	DSSR
STEXT	Stop Mode External Clock	SYNCR
STF	Stop Flag	TMCR
STOP	Stop Control	TMCR
STOP	Stop Control	ADCMCR
STOP	Stop Control	GPTMCR
STOP	Stop Control	MMCR
STOP	Stop Control	TRAMMCR
STOPP	Stop Prescaler	GPTMCR
STRB	Address Strobe/Data Strobe	CSOR[0:10], CSORBT
STSCIM	Stop Mode Single-Chip Integration Clock	SYNCR
STS[6:5]	Sample Time Selection	ADCTL0
SUPV	Supervisor/Unrestricted	ADCMCR
SUPV	Supervisor/Unrestricted	GPTMCR
SUPV	Supervisor/Unrestricted	MMCR
SUPV	Supervisor/Unrestricted	SCIMCR
SUPV	Supervisor/Unrestricted	TMCR
SW	Software Watchdog Reset	RSR
SWE	Software Watchdog Enable	SYPCR
SWP	Software Watchdog Prescale	SYPCR
SWSR	Software Service	SWSR
SWT[5:4]	Software Watchdog Timing	SYPCR
TC	Transmit Complete	SCSRA, SCSRB

Table D–10. Register Bit and Field Mnemonics (Continued)

Mnemonic	Name	Register Location
TCIE	Transmit Complete Interrupt Enable	SCCR1A, SCCR1B
TCR1P	Timer Count 1 Prescaler Control	TMCR
TCR2P	Timer Count 2 Prescaler Control	TMCR
TDRE	Transmit Data Empty	SCSRA, SCSRB
TE	Transmitter Enable	SCCR1A, SCCR1B
TIE	Transmit Interrupt Enable	SCCR1A, SCCR1B
TOF	Timer Overflow Flag	TFLG1/TFLG2
TOI	Timer Overflow Interrupt Enable	TMSK1/TMSK2
TPUF	TPU Freeze Flag	DSSR
TST	Test Submodule Reset	RSR
T2CG	Timer Count Register 2 Clock/Gate Control	TMCR
UPPB	Upper Byte	SPDR
V	Overflow Flag	CCR
VBA	Vector Base Address	ICR
W	Frequency Control (VCO)	SYNCR
WAIT	Wait States	MRMCR
WAKE	Wakeup by Address Mark	SCCR1A, SCCR1B
WCOL	Write Collision	SPSR
WOMP	Wired-OR Mode for SPI Pins	SPCR
WOMS	Wired-OR Mode for SCI Pins	SCCR1A, SCCR1B
X	Frequency Control Bit (Prescaler)	SYNCR
Y[5:0]	Frequency Control (Counter)	SYNCR
Z	Zero Flag	CCR

INDEX

– A –

- Abort ADC conversion sequence and halt 6–7
- Abort ADC conversion sequence and start new sequence 6–7
- Acceptable bus cycle terminations for asynchronous cycles 4–35
- Accumulators 5–3
 - Accumulator A (A) 2–2, 5–3
 - Accumulator B (B) 2–2, 5–3
 - Accumulator D (D) 2–2, 5–3
 - Accumulator E (E) 2–2, 5–3
 - Accumulator M (AM) 2–2, 5–3
- Accumulator M overflow flag (MV) 2–2, 5–4
- Accumulator offset addressing mode 5–12
- ADC and IMB data bus 6–3
- ADC module base address 6–3
- Addition and subtraction instructions 5–14
- ADDR0 indicating byte offset 4–26, 4–27
- Address bus (ADDR[23:0]) 2–3, 3–7, 3–8, 4–21, 4–41, 4–42, 4–45
- Address bus convention 2–8
- Address extension 5–6
- Address extension fields 2–2, 5–5
 - Extended addressing extension field (EK) 2–2, 5–5
 - Index register X extension field (XK) 2–2, 5–5
 - Index register Y extension field (YK) 2–2, 5–5
 - Index register Z extension field (ZK) 2–2, 5–5
 - Program counter extension field (PK) 2–2, 5–5
 - Stack pointer extension field (SK) 2–2, 5–3, 5–6
- Address extension fields and types of access 5–6
- Address extension instructions 5–12
- Address extension register (K) 2–2, 5–5
- Address space 3–11, 4–67, 5–1
- Address space encoding 4–23, 5–3
- Address strobe (\overline{AS}) 2–3, 3–5, 3–7, 3–9, 4–22, 4–39, 4–42, 4–45
- Address-mark wakeup, SCI 7–11
- Addresses 20-bits wide 5–6
- Addressing modes 5–10
 - Accumulator offset addressing mode 5–12
 - Extended addressing modes 5–10
 - Immediate addressing modes 5–10
 - Indexed addressing modes 5–10
 - Inherent addressing mode 5–10
 - Post-modified index addressing mode 5–11
 - Relative addressing modes 5–12
- Aligned word 4–25
- Amount of data transferred by bus cycle 4–25
- Analog input pins, ADC (AN[7:0]) 3–5, 3–7, 3–8, 6–3
- Analog-to-digital converter module (ADC) 3–2, 6–1
 - Analog input signals (AN[7:0]) 2–3, 3–5, 3–7, 6–3
 - Analog reference pins 6–3
 - Analog subsystem 6–5
 - Analog supply pins 6–3
 - Analog-to-digital conversions 6–9
 - Buffer amplifier 6–6
 - Bus interface unit, ADC (ABIU) 6–4
 - Clock and prescaler control 6–7
 - Clock cycles required for sample period 6–8
 - Clock derived from system clock 6–7
 - Clock speed 6–8
 - Comparator 6–7
 - Control and status registers 6–5, 6–7
 - Conversion channel 6–7, 6–9
 - Conversion control logic 6–9
 - Conversion modes 6–9
 - Conversion parameters 6–9
 - Conversion time 6–14
 - Conversions performed in sequences 6–9
 - Differential and buffered data buses 6–3
 - Digital control subsystem 6–7
 - Features 3–1

- Input channel sample capacitor 6–6
- Low-power stop mode 6–4
- Module base address 6–3
- Multiplexer 6–5
- Port ADA 2–3, 6–1
- Power connections (V_{DDA} and V_{SSA}) 2–4, 3–7, 6–3
- Prescaler 6–7
- RC DAC array 6–6
- Reference voltage connections (V_{RH} and V_{RL}) 2–4, 3–6, 6–3
- Resolution 6–8
- Resolution time 6–14
- Result registers (RSLT[0:7]) 6–16
- Status register (ADSTAT) 6–9
- Successive approximation register (SAR) 6–16
- Transfer and resolution 6–14
- Transfer time 6–14
- Analog reference pins, ADC (V_{RH} and V_{RL}) 3–6, 6–3
- Analog supply pins, ADC (V_{DDA} and V_{SSA}) 3–6, 6–3
- Application of system and clock synthesizer power 4–53
- Arbitration between interrupt requests 4–5, 4–57
- Arbitration must always take place 4–74
- Arbitration scheme 4–58
- \overline{AS} 4–29, 4–69
- Asserted, definition 2–8
- Asynchronous bus cycles 4–22
- Asynchronous exceptions 5–39
- Asynchronous input hold times 4–27
- Asynchronous input setup time 4–38
- Automatic interrupt vectors 4–56
- Autovector number 4–79
- Autovector operation timing 4–80
- Autovector signal (\overline{AVEC}) 2–3, 3–5, 3–7, 4–6, 4–24
- Auxiliary timer clock input signal, GPT (PCLK) 3–6, 3–8, 8–8

– B –

- Background debugging mode (BDM) 5–41, 5–67
 - Commands 5–44
 - Serial interface 5–43
 - Sources 5–43
- Bank switching, memory 5–6
- Base address, module control register 3–10
- Basic instruction formats 5–32
- Basic operand size 4–26
- Basic system 4–21
- Baud rate , SCI 7–12, 7–28
- BCD packing 5–7
- BDM commands 5–44
- BDM serial interface 5–43
- BDM sources 5–43
- \overline{BERR} signal assertion and exception processing 4–36, 4–37
- \overline{BERR} , detection/acknowledge timing 4–36, 4–44
- \overline{BG} signal assertion 4–40
- \overline{BGACK} signal assertion 4–40
- Bit cleared, definition 2–8
- Bit set, definition 2–8
- Block diagram, system 3–3
- Block of addresses enabled by chip select 4–64
- Block size, chip select 4–64
- Boolean 0 2–8
- Boolean 1 2–8
- Boot ROM chip-select signal (\overline{CSBOOT}) 2–3, 3–5, 3–7, 4–63
- Boot ROM port size 4–63
- Boot ROM select signal automatically asserted 4–82
- Bootstrap operation 4–65
- Bootstrap operation from reset 4–71
- \overline{BR} signal assertion and bus cycles 4–40
- Breakpoint 4–33, 5–40
 - Breakpoint acknowledge cycle 4–33, 4–34, 4–50
 - Breakpoint acknowledge cycle timing 4–50
 - Breakpoint acknowledge (Type \$0) cycles 4–33
 - Breakpoint exception processing 4–35, 5–40
 - Breakpoint mode selection 4–50
 - Breakpoint number field 4–35
 - Breakpoint operation 4–34, 4–35
- Breakpoint signal (\overline{BKPT}) 2–3, 3–5, 3–7
- Buffer amplifier, ADC 6–6
- Bus allocation for chip-select transfers 4–65
- Bus arbitration 4–39, 4–40
 - Bus arbitration control 4–58
 - Bus arbitration protocols 4–39
 - Bus arbitration requests 4–39
- Bus clock for MC6800 devices (ECLK) 4–19, 4–67

- Bus cycle 4-38, 5-36
 - Not terminated in the expected manner 4-36
 - Regular bus cycles 4-36, 4-39
 - Required for operand accesses 5-36
 - Required to prefetch instruction 5-36
 - Bus error 4-52
 - Bus error condition 4-24
 - Bus error exception processing 4-36
 - Bus error with $\overline{\text{HALT}}$ asserted 4-37
 - Bus error signal ($\overline{\text{BERR}}$) 2-3, 3-5, 3-7, 4-24, 4-39, 4-48, 4-52
 - $\overline{\text{BERR}}$ and immediate exception processing 4-58
 - Bus grant signal ($\overline{\text{BG}}$) 2-3, 3-5, 3-7, 4-55
 - Bus grant acknowledge signal ($\overline{\text{BGACK}}$) 2-3, 3-5, 3-7, 4-55
 - $\overline{\text{BGACK}}$ indicates bus mastership 4-55
 - Bus master, external 4-40
 - Bus monitor 4-24, 4-37
 - Monitor timing (BMT) field, 4-6
 - Monitor period 4-6
 - Bus request 4-58
 - Bus request signal ($\overline{\text{BR}}$) 2-3, 3-5, 3-7, 3-8, 4-52, 4-66
 - Bus signals 4-22
 - Byte data type 5-7
- C -
- Capture/compare, GPT 8-1, 8-11
 - Capture register 8-7
 - Carry flag (C) 2-2, 5-5
 - CCR bits and reset 4-55
 - Central processing unit (CPU16) 5-1
 - Execution model 5-34
 - Execution process 5-35
 - Execution unit 5-35
 - Features 3-1
 - Indexed addressing mode and M68HC11 direct mode 5-11
 - Microsequencer 5-34
 - Register mnemonics 2-2
 - Register model 5-2
 - Supporting one source and type of breakpoint 4-50
 - Changes in program flow 5-35
 - Changing SCI control bits 7-26
 - Channel control, TPU 9-13
 - Channel function aelect, TPU 9-14
 - Channel interrupt enable, TPU 9-13
 - Channel orthogonality, TPU 9-4
 - Channel priority, TPU 9-5, 9-16
 - Channel status, TPU 9-13
 - Chip-select, emulation mode ($\overline{\text{CSE}}$) 4-72
 - Chip-select, emulation memory ($\overline{\text{CSM}}$) 4-72
 - Chip selects 4-1, 4-62
 - Assertion 4-63
 - Block size 4-64
 - Chip-select circuits out of reset 4-65
 - Generates internal $\overline{\text{AVEC}}$ 4-66
 - Logic and external interrupt acknowledge cycles 4-66
 - Operation 4-69
 - Option register function 4-71
 - Chip-select option registers and $\overline{\text{DSACK}}$ 4-67
 - Pin assignment registers (CSPAR) 4-64
 - Pin functions 4-65
 - Registers 4-66
 - Chip-select reset operation 4-71
 - Set up for CPU space access 4-68
 - Signals ($\overline{\text{CS}}[10:5]$) 2-3, 3-5, 3-7, 3-8
 - Clear, defined 2-8
 - Clearing GPT status flags 8-7
 - CLKOUT signal (68000 bus clock) 2-3, 3-5, 3-7, 3-8, 4-12
 - Clock, system 4-12
 - Control 4-15, 4-20
 - Mode select signal (MODCLK) 2-3, 3-5, 3-6, 3-8, 4-7, 4-8, 4-20
 - Mode selection 4-48
 - Clock control multipliers 4-15
 - Operation during low-power stop 4-19
 - Prescaler control 6-7
 - Rate changes 4-12
 - Rate during operation 4-10, 4-12
 - Reference from external source 4-12
 - Reference signal 4-12
 - Signal duty cycle and period 4-13
 - Signal from external source 4-13
 - Sources 4-12

- Synthesizer operation 4-13
- Synthesizer power connection (V_{DDSYN}) 2-4, 3-6, 4-13
- Coherency 4-57, 7-2, 9-5, 10-2
- Combined program and data spaces 3-13
- Combining opcode tracking with other capabilities 5-41
- Comparator, ADC 6-6, 6-7
- Comparison of CPU16 and M68HC11 CPU instruction sets 5-30
- Compatibility with the M68HC11 5-1
- Completion of a bus cycle 4-24
- Condition code register (CCR) 2-2, 5-4
- Conditioning mode select signals 4-64
- Conditions for external bus mastership 4-40
- Configuration options (SCIM) 4-3
- Configuration, TPU 9-11
- Connecting GPT multiplexer to external pins 8-9
- Connection between TPURAM and TPU during emulation 9-5
- Content of ADC result registers 6-5
- Contents of ADC control and status registers 6-5
- Continuous A/D conversion 6-9
- Control registers, location in memory map 3-10
- Conventions 2-7
- Conversion channel, ADC 6-9
- Conversion control logic, ADC 6-9
- Conversion modes, ADC 6-9
- Conversion parameters, ADC 6-9
- CPU (See also Central Processing Unit) 2-2, 3-1
 - Register Model 5-2
 - Setting STOP bits in module configuration registers 4-19
 - Space 4-22, 4-57
 - Space address encoding 4-33, 4-70
 - Space cycles 4-33
 - Space type field 4-33
 - External interrupt requests 4-57
- CPU16 indexed mode replacing M68HC11 direct mode 5-12
- CSBOOT signal automatically asserted out of reset 4-71

- D -

- Data and size acknowledge signals ($\overline{DSACK}[1:0]$) 2-3, 3-5, 3-7, 4-22, 4-6, 4-34 to 4-52, 4-72
- Data bus mode selection 4-62, 4-63
- Data bus signals ($DATA[15:0]$) 2-3, 3-5, 3-7, 4-21, 4-45
- DATA mnemonic 2-8
- Data strobe signal (\overline{DS}) 2-3, 3-5, 3-7, 4-23, 4-42, 4-48, 4-49
- Data types 5-7
 - 4-bit bank address 5-7
 - 4-bit signed integers 5-7
 - 8-bit signed and unsigned integers 5-7
 - 8-bit, 2-digit binary coded decimal numbers 5-7
 - 16-bit byte address 5-7
 - 16-bit fractions 5-7
 - 16-bit signed fractions 5-6
 - 20-bit addresses 5-7
 - 20-bit effective address 5-7
 - 32-bit signed and unsigned integers 5-7
 - 32-bit signed fractions 5-7
 - 36-bit signed fixed-point numbers 5-7
- Deskewing input signals 4-28
- Deterministic opcode tracking 5-41
- Development support, MCU 5-41
 - Development serial clock signal (DSCLK) 2-3, 3-5, 3-7, 5-45
 - Development serial input signal (DSI) 2-3, 3-6, 3-7, 5-45
 - Development serial output signal (DSO) 2-3, 3-6, 3-7, 5-44
- Development support, TPU 9-16
- Digital control subsystem, ADC 6-7
- Digital signal processing 5-47
- Double bus fault 4-38
- Driver types 3-5
- \overline{DSACK} generator shared by chip-select circuits 4-63
- \overline{DSACK} , \overline{BERR} , and \overline{HALT} assertion results 4-37
- \overline{DSACK} option fields 4-32
- Duty cycle of clock signals on OC1 and PWMA 8-10
- Duty cycle ratios of PWM channels 8-17
- Dynamic bus sizing 4-22, 4-25, 5-36

- E -

- EBI data multiplexer 4-26
- ECLK (6800 bus clock) 4-67
- ECLK frequency 4-18
- Edge-detection logic, GPT 8-11, 8-13
- Effect of \overline{DSACK} signals 4-24
- Emulation of microcode ROM 10-4
- Emulation mode 9-13, 10-4
- Enabling BDM 5-42
- Entering BDM 5-42
- Entering TPU emulation mode 9-13
- EV — extension bit overflow flag 5-4
- Event counting mode, GPT 8-16
- Event counting mode, pulse accumulator 8-16
- Event timing, TPU 9-4
- Exception 5-37
 - Handler routine 5-37
 - Priority 5-37
 - Processing 4-38, 4-42, 4-57, 8-6
 - Processing for synchronous exceptions 5-37
 - Processing sequence 5-38
 - Stack frame 5-38
 - Vector 3-11, 5-37, 7-4, 8-6
 - Vector number 7-4, 8-6
 - Vector table 5-38, 8-2, 10-7
- Excessively long bus cycle response times 4-36
- Execution model 5-34
- Execution of LPSTOP instruction 4-36
- Execution process 5-35
- Execution unit, CPU16 5-35
- Extended addressing extension field (EK) 2-2, 5-11
- Extended addressing modes 5-10
- Extension bit overflow flag (EV) 5-4
- Extension fields 5-5
- Extension words, BDM instruction 5-44
- External address spaces 4-21
- External arbitration circuitry 4-40
- External bus 4-1
 - Arbitration 4-22, 4-39
 - Circuitry 4-40
 - Clock signal, 6800 (ECLK) 3-5, 3-7, 4-19
 - Clock signal, 68000 (CLKOUT) 2-3, 3-5, 3-7, 3-8, 4-12
 - Control logic 4-28

Cycles 4-6

- Interface (EBI) 4-1, 4-21
- Monitor 4-6
- External bus master 4-57
- External crystal oscillator connections (EXTAL, XTAL) 2-3, 2-4, 4-13
- External device and \overline{BERR} 4-35
- External device asserts \overline{RESET} 4-71
- External device and IP mask value 4-71
- External exceptions 5-37
- External filter capacitor pin (XFC) 2-4, 4-14
- External interrupt request 4-5, 4-57, 4-58
- External interrupt requests as SCIM interrupts 4-57, 4-59
- External loading and data bus pull-up drivers 4-59
- External logic on input/output or output-only pins 4-54
- External low-leakage capacitor for system clock synthesizer 4-14
- External peripheral power connections (V_{DDE} and V_{SSE}) 2-4
- External peripheral power pins (V_{DDE} , V_{SSE}) 2-4
- External \overline{RESET} signal assertion 4-54
- External system clock reference signal 4-12
- External system clock signal 4-12
- Externally generated \overline{DSACK} 4-29

- F -

- Factory test mode 4-6
- Factory test mode arbitration 4-41
- Fast mode 8-17, 8-19
- Fast termination cycles 4-28, 4-32
 - Encoding 4-70
 - Read cycle 4-32
 - Write cycle 4-32
- Fast termination encoding 4-32
- Fast termination with \overline{DS} asserted 4-32
- First two portions of ADC sample periods 6-8
- Forced output compare, GPT 8-14
- Format of ADC result 6-16
- Freeze mode, ADC 6-4
- Fractional multiplication and division 5-16
- Freeze mode, SCIM 4-11
 - Bus monitor 4-11

Operation 4–11
Signal (FREEZE) 2–3, 4–11, 5–45
Software watchdog 4–11
Function code signals (FC[2:0]) 2–3, 3–5, 3–7, 4–23,
4–33, 4–45

– G –

Gated time accumulation mode 8–16
General-purpose I/O ports 4–45, 6–1, 7–4, 8–8
 Chip-select (port C) pin assignment register
 (CSPAR) 4–64
 Port A and Port B data direction register (DDRAB)
 4–72
 Port A (PA[7:0]) 4–45, 4–48, 4–72
 Port ADA (PADA[7:0]) 6–1
 Port B (PB[7:0]) 4–45, 4–48, 4–72
 Port C (PC[6:0]) 4–48, 4–64, 4–69, 4–72
 Port E data direction register (DDRE) 4–72
 Port E pin assignment register (PEPAR) 4–72
 Port E (PE[7:0]) 4–48, 4–72
 Port F data direction register (DDRF) 4–72
 Port F pin assignment register (PFPAR) 4–46,
 4–49, 4–72
 Port F (PF[7:0]) 4–48, 4–72
 Port G data direction register 4–72
 Port G (PG[7:0]) 4–45, 4–48, 4–72
 Port GP data direction register (DDRGP) 8–8,
 8–15
 Port GP (PGP[7:0]) 4–51, 8–8
 Port H data direction register 4–72
 Port H (PH[7:0]) 4–45, 4–48, 4–72
 Port MC (PMC[7:0]) 7–4, 7–5
General-purpose timer (GPT) 3–2, 8–1
 Activities in progress before FREEZE assertion
 8–4
 Capture/compare functions 8–11
 Counter during normal operation 8–11
 Counter value during read 8–11
 Debugging without BDM 8–4
 Edge-detection logic 8–11, 8–13
 External timer clock input (PCLK) 2–3, 8–1, 8–8,
 8–9, 8–11, 8–16, 8–19
 Features 3–2
 Freeze mode 8–3

Input capture (IC) functions 8–11
Input capture pins (IC[1:3]) 8–7
Input capture/output compare pin (IC4/OC5) 8–8
Input pin synchronizers 8–3
Interrupts 8–5
Low-power stop operation 8–3
Multiplexer outputs connected to external pins
8–10
Output compare (OC) functions 8–8, 8–11, 8–14
Output compare pins (OC[1:4]) 8–8
Pin hysteresis 8–13
Pin synchronizers 8–13
Pins used for general-purpose I/O 8–8
Polled and interrupt-driven operation 8–4
Port GP (PORTGP) 2–3, 4–51, 8–8
Prescaler 8–9
Prescaler driven by system clock 8–9
Prescaler read 8–9
Pulse accumulator 8–8
Pulse-width modulation (PWM) unit 8–17
Single-step mode 8–4
Slow mode 8–17, 8–19
Status flags 8–4, 8–5, 8–7, 8–13
Timer prescaler 8–5

– H –

Half-carry flag (H) 2–2, 5–4
Halt monitor 4–7
Halt monitor reset 4–7
Halt operation 4–39
Halt signal ($\overline{\text{HALT}}$) 2–3, 3–5, 3–7, 4–24, 4–29, 4–35,
4–36, 4–39, 4–40, 4–52
 $\overline{\text{HALT}}$ and external bus cycles 4–36
 $\overline{\text{HALT}}$ assertion, 4–29
 $\overline{\text{HALT}}$ assertion and single bus cycle operation
4–36
 $\overline{\text{HALT}}$ assertion by external device, 4–36
 $\overline{\text{HALT}}$ during internal BERR 4–39
 $\overline{\text{HALT}}$ timing 4–37
Handler routine, exception 5–37
Handshaking between MCU and external peripherals
4–28
Host sequence, TPU 9–14
Host service, TPU 9–14

- Idle frame, SCI 7-10
- Idle-line detection, SCI 7-11
- Idle-line wakeup, SCI 7-11
- Immediate addressing modes 5-11
- Implied radix point 5-7
- Index register 5-3
 - Index register X (IX) 2-2, 5-3
 - Index register Y (IY) 2-2, 5-3
 - Index register Z (IZ) 2-2, 5-3
- Index register X extension field (XK) 2-2, 5-10
- Index register Y extension field (YK) 2-2, 5-10
- Index register Z extension field (ZK) 2-2, 5-10
- Indexed addressing modes 5-10
- Individual module STOP bits 4-19, 4-36
- Inherent addressing mode 5-10
- Initial ADC sample and transfer times 6-14
- Input capture (IC) functions, GPT 8-11, 8-13, 8-14
- Instruction extension words, BDM 5-68
- Instruction fetches 5-4, 5-8
- Instruction format 5-32
- Instruction pipeline 4-34, 5-35, 5-41, 5-35, 5-42
 - State signals 3-5, 3-7, 3-8, 5-65
 - Pipeline states 5-41
- Instruction pipeline signals (IPIPE[1:0]) 2-3
 - Multiplexing 5-41
 - Not usable in BDM 5-43, 5-44
- Instruction set 5-11
- Instruction set abbreviations and symbols 5-29
- Instruction set summary 5-12
- Instruction timing 5-36
- Instruction types 5-33
 - 8-bit opcode with 4-bit index extensions 5-33
 - 8-bit opcode with 8-bit operand 5-33
 - 8-bit opcode with 8-bit prebyte, argument(s) 5-33
 - 8-bit opcode with 8-bit prebyte, no argument 5-33
 - 8-bit opcode with 20-bit argument 5-33
- Intermodule bus (IMB) 3-10
- Internal and external breakpoint signals 4-50
- Internal cycles when external bus is granted 4-53
- Internal handshaking signals generated by chip-select logic 4-28
- Internal module power connections 2-4
- Internal or external frequency source, system clock 4-12, 4-13
- Internal oscillator 4-11
- Internal phase-locked loop 4-11
- Internal pull-up drivers on data bus 4-71
- Internal register addresses 3-11
- Internal source asserts reset signal 4-53
- Interrupts 4-57
 - Acknowledge and arbitration 4-58
 - Acknowledge cycle 4-33, 4-58, 4-69, 4-70, 7-3
 - Acknowledge cycle timing 4-59
 - Acknowledge using chip-select signal 4-59, 4-69
 - Arbitration 4-4, 4-58, 7-3
 - Arbitration during interrupt exception processing 4-58
 - Arbitration field (IARB) 4-4, 4-58
 - Interrupts as a type of asynchronous exception 4-57
 - Exception processing 4-57
 - Instructions 5-23, 5-29
 - Priority (IP) mask 2-2, 4-57, 4-89, 7-4
 - Priority 4-57
 - Interrupt priority and recognition 4-57
 - Processing summary 4-60
 - Recognition 4-57
 - Request 4-5, 4-58, 4-59, 9-13
 - Request acknowledged/no IARB contention 4-36
 - Request circuitry and hysteresis 4-58
 - Request signals ($\overline{\text{IRQ}}[7:1]$) 2-3, 3-6, 3-8, 4-57, 7-3, 8-6
 - Requests 4-57
 - Stack frame 5-12
 - Vector number 4-59, 7-4, 8-6, 8-7
- Interrupt-driven GPT operation 8-4
- IP mask 4-58
- IPIPE[1:0] 5-39, 5-41, 5-42, 5-44
- IPIPE0/IPIPE1 multiplexing 5-41
- IP[2:0] — Interrupt Priority field 5-5
- IPL field 4-68
- $\overline{\text{IRQ}}[7:1]$ signals 4-58
- $\overline{\text{IRQ}}7$ nonmaskable 4-58
- $\overline{\text{IRQ}}[6:1]$ maskable 4-58

– L –

Largest amount of data transferred by bus cycle 4–26
Least significant bit (LSB), definition 2–5
Least significant word (LSW), definition 2–5
Length of ADC conversion sequence 6–9
Loading and data bus pull-up drivers 4–71
Lock time, clock synthesizer 4–13
Logic level one, definition 2–8
Logic level zero, definition 2–8
Long idle-line detection, SCI 7–10
Long word data type 5–7
Low-pass filter, clock synthesizer 4–13
Low-power operation, MRM 11–3
Low-power operation, SCIM 4–18
Low-power operation, TPU 9–13
Low-power operation, TPURAM 10–3
Low-power stop 4–55, 6–4, 7–3, 8–3, 9–2, 9–13
Low-power stop broadcast cycle 4–19, 4–36
Low-power stop used for GPT debugging 8–3
LPSTOP 4–36
 Broadcast cycle 4–36
 CPU space cycle shown externally 4–19, 4–36
 Instruction 4–22
LSB, definition 2–8
LSW, definition 2–8

– M –

M68HC11 compatibility 5–1
M68HC11 direct mode addressing 5–11
M68HC11 instructions 5–13
MAC multiplicand register (IR) 2–2
MAC multiplier register (HR) 2–2
MAC sign latch (SL) 2–2
Manual nomenclature 2–1
 Conventions 2–8
 Mnemonics 2–3
 Operators 2–1
 Symbols 2–1
Mapping of exception vector number to vector table
 5–37
Masked ROM module (MRM) 10–1
 Default reset base address 10–1
 Emulation mode 10–4

 Restricted access levels 10–2
 ROM array 11–1
 Wait states 11–3
Match and capture handled by independent TPU
 channel hardware 9–4
Maximum allowable bus response time 4–6
Maximum monitor period 4–29
Maximum specified system clock frequency 4–13
MC68HC16Y1 features 3–1
MCU power consumption 4–19
MCU and interrupt requests during halt 4–39
Memory management 5–6
Memory maps, system 3–11
Memory organization 5–8
Microcode executed from TPURAM module 9–3
Microengine, TPU 9–3
Microsequencer, CPU 5–34
Misaligned operands 4–26
Misaligned word 5–8
Misaligned word transfers 4–26
Mnemonic, definition 2–5
MODCLK, clock mode selection pin 4–7, 4–8, 4–10
Mode select pin 3–15, 4–62
Modes of operation, MCU 4–13
Module control registers 3–10
Module mapping 3–10, 4–4
Module pin function during reset 3–15, 4–49
Most significant bit (MSB), defined 2–8
Most significant word (MSW), defined 2–8
MRM access during reset 11–3
MRM array address mapping 11–2
Multichannel A/D conversion 6–9, 6–13
Multichannel communications interface (MCCI) 7–1
 Break characters 7–8
 Frame formats 7–7
 Idle-line characters 7–8
 Initialization 7–12
 Operating modes 7–13
 Parity check 7–2, 7–7
 Pin function 7–12
 Registers 7–2
 Serial communication interface (SCI) 7–1, 7–2,
 7–5
 Serial peripheral interface (SPI) 7–1, 7–2, 7–12

Synchronization with incoming data stream 7–8
Serial clock signal (SCK) 2–4, 3–5, 3–7, 3–8, 7–4
Serial mode 7–7
Slave mode 7–13, 7–14
Slave mode operation 7–14
Slave select signal (\overline{SS}) 3–5, 3–7, 3–8, 7–4,
7–11, 7–13

Unable to initiate transfers in slave mode 7–14

Multiple bus errors 4–39
Multiple chip-select assertion 4–69
Multiple exceptions 5–40
Multiplexer, ADC 6–5
Multiplexer outputs (including the PCLK signal) and
external pins 8–10
Multiply and accumulate registers 5–5
Multiply and accumulate (MAC) unit 5–55
MV — Accumulator M overflow flag 5–4

– N –

N — Negative flag 5–4
Negated, definition 2–8
Negating and reasserting \overline{HALT} 4–38
Negative flag (N) 2–2, 5–4
Negative integers 5–7
No IARB assertion with interrupt acknowledge 4–59
Nonmaskable interrupt 4–58
Normal bus cycles 4–6
NRZ (Nonreturn to zero) format, SCI 7–28
Numeric range of branch instruction offset values 5–9

– O –

OC1 and other OC pins, GPT 8–14, 8–10
Occurrence of a bus error with \overline{HALT} asserted 4–39
One source and type of breakpoint 4–34
Opcode map 5–12
Opcode tracking and breakpoints 5–41
Opcodes 5–33
Operand 4–25, 4–31
 Alignment 4–26, 4–27
 Byte order 4–25
 Bytes 4–25
 Coherency 9–5
Operand transfers, 8- and 16-bit ports 4–27

Operand transfer cases 4–26, 4–27
Operating frequency 4–13
Operators 2–1
Order of exception handler execution 5–39
Order of access 4–26

– P –

Page 0 opcode 5–31
Parameter RAM 9–3, 9–4, 9–5
Parameter registers 9–1, 9–12
Parity checking 7–7
Parity error 7–7
Periodic interrupts 4–8
 Modulus counter 4–9
 Priority 4–10
 Request level 4–10
 Timer 4–9
 Vector 4–10
Phase comparator 4–13
Pin and signal mnemonics 2–3, 3–5, 3–7, 3–8
Pin assignment field encoding 4–65
Pin assignment for 160-pin package 3–4
Pin characteristics 3–5
Pin function and pin electrical state 3–5, 3–7, 3–8,
4–49
Pin mnemonics 2–3
Pin state during reset 4–49, 4–50
Pipeline 4–34
Pipeline state signals (IPIPE[1:0]) 2–3, 3–5, 3–8,
5–35, 5–41
Pipeline states 5–39
PK[3:0] — Program counter address extension field
5–5, D–3
Polled GPT operation 8–5
Polling GPT status registers 8–4
PORTC data register 4–69
Port replacement unit (PRU) 4–72
Port size 4–24, 4–65
Port width 4–22, 4–24
Positioning of bytes 4–26
Post-modified index addressing mode 5–11
Power connections 3–7
Power consumption 4–19
Power-on reset 4–53

- Power-on reset timing 4–54
- Prefetch mechanism 5–35
- Prescaler, GPT 8–9
- Prescaler control, TPU 9–2, 9–11
- Priority mask, interrupt 4–57
- Priority of TPU interrupt service requests 9–6
- Processing aborted by reset exception 4–55
- Programmable chip-select circuits 4–69
- Program and data spaces 3–11, 5–6, 5–7, 11–2
- Program counter (PC) 2–2, 5–4
- Program counter extension field (PK) 2–2, 5–5
- Programmable chip-select circuits 4–69
- Programmable interrupt timer 4–9
- Programmable prescaler, ADC 6–7
- Programmable serial transfer length, SPI 7–6
- Programming register model 2–2
- Pseudolinear address space 3–11, 5–1
- Pulse accumulator, GPT 8–16
- Pulse-width modulation (PWM) unit, GPT 8–17

– Q –

- Quotient out signal (QUOT) 2–4

– R –

- Range of mnemonics, definition 2–5
- RC DAC array, ADC 6–6
- Re-enable GPT status flag 8–5
- Read cycle 4–30
- Read/write signal (R/W) 2–4, 3–6, 3–8, 4–23, 4–29, 4–39, 4–52, 4–69
- Reduce MCU power consumption selectively 4–9
- Reduce MCU power consumption to a minimum 4–9
- Reference crystal 4–13
- Register access 4–5
- Regular bus cycles 4–28
- Relative addressing modes 5–12
- Reset 4–42
 - As asynchronous exception 4–42
 - As highest-priority CPU16 exception 4–42
 - At end of bus cycle 4–7, 4–43
 - Control logic 4–43
 - Exception processing 4–42
 - Gated by CLKOUT 4–54

- Mode selection 3–15
- Processing 4–55
- Signal ($\overline{\text{RESET}}$) 2–4, 3–6, 3–8, 4–56
- Source 4–42
- States of other MCU module pins 4–51
- States of SCIM pins 4–51
- Status register (RSR) 4–5, 4–56
- Value of SCIM IARB field 4–5
- Vectors 3–11, 8–5
- Reset signal ($\overline{\text{RESET}}$) 2–4, 3–6, 3–8, 4–42, 4–56
 - $\overline{\text{RESET}}$ assertion period 4–68
 - $\overline{\text{RESET}}$ input 4–42
 - $\overline{\text{RESET}}$ synchronized to system clock 4–42
 - Reset value of IARB for the SCIM 4–5
- Reset during TPURAM access 10–4
- Resets gated by CLKOUT 4–43
- Resolution, ADC 6–8
- Resolution time, ADC 6–14
- Result registers, ADC 6–16
- Returning from BDM 5–45
- RTI instruction 5–40
- ROM array 11–1
- ROM array address space type 11–2
- ROM array verification 11–3
- ROM array base address 11–2

– S –

- S — STOP enable bit 5–4
- Sample capacitor, ADC 6–6
- Sample time, ADC 6–8, 6–14
- Saturation mode control bit (SM) 2–2, 5–5
- Scheduler, TPU 9–3
- SCIA and SCIB, MCCI 7–3, 7–4
- Separated program and data spaces 3–13
- Separate program and data spaces 3–11
- Serial communication interface (SCI) 7–2, 7–5
 - Baud rate 7–8
 - Data frame 7–6
 - Errors 7–9, 7–10
 - Parity checking 7–7
 - Pin function 7–5
 - Receive time clock (RT) 7–8, 7–11
 - Receiver bit processor logic 7–10
 - Receiver idle-line detection 7–10

- Receiver operation 7–9
- Receiver wakeup 7–10
- Registers 7–6
- Serial frame formats 7–7
- Short idle-line detection 7–11
- Transmitter double-buffered 7–8
- Transmitter operation 7–8
- Serial data transferred MSB first 7–6
- Serial frame formats, SCI 7–6
- Serial peripheral interface (SPI) 7–12
 - Clock 7–13
 - Modes 7–13
 - Mode fault 7–14
 - Operation 7–13
 - Phase and polarity 7–13
 - Pin function 7–12
 - Registers 7–12
 - Write collision 7–13
- Set, definition 2–8
- Setting STOP bits in module configuration registers 4–36
- Short and long idle-line detection, SCI 7–11
- Short branch instructions 5–23
- Show cycle 4–5, 4–41
- Show cycle enable bits 4–5
- Signal assertion, definition 2–5
- Signal characteristics 3–7
- Signal function 3–9
- Signal mnemonics 2–8
- Signal negation, definition 2–5
- Signed 8-bit offset 5–11
- Signed 16-bit offset 5–11
- Signed 32-bit fractions 5–7
- Signed 36-bit fixed-point numbers 5–7
- Signed branches 5–23, 5–25
- Signed left-justified conversion format, ADC 6–17
- Signed relative offset 5–27
- Simple branches 5–2, 5–25
- Single buffer amplifier, ADC 6–6
- Single bus cycle operation 4–22
- Single-channel A/D conversion, ADC 6–12
- Single-chip integration module (SCIM) 4–1
 - Asynchronous bus cycles 4–22
 - Automatic interrupt vectors 4–56
 - Boot ROM chip-select signal 4–2
 - Breakpoints 4–33
 - Bus arbitration, external 4–39, 4–40
 - Bus cycle 4–38 to 4–40
 - Bus error 4–52
 - Bus grant 4–38, 4–55
 - Bus monitor 4–5, 4–6, 4–24, 4–37
 - Bus request 4–58
 - Bus signals 4–22
 - Chip-select block 4–2
 - Chip selects 4–1, 4–62
 - Clock, system 4–12
 - Configuration and protection block 4–2
 - CPU space 4–22, 4–57
 - Data bus mode selection 4–62, 4–63
 - Deskews signals 4–28
 - Dynamic bus sizing 4–22, 4–25
 - Emulation chip-select signals 4–2
 - Exception 4–38, 4–42, 4–57
 - External bus interface 4–1, 4–21
 - Features 3–1
 - Freeze mode 4–11
 - Fully expanded mode 4–3, 4–4, 4–45
 - Halt monitor 4–7
 - Interrupts 4–57
 - Low-power operation 4–10, 4–18
 - MCU power consumption 4–19
 - Modes of operation 4–3, 4–44, 4–45
 - Operand transfer cases 4–26
 - Partially expanded mode 4–3, 4–44, 4–45
 - Periodic interrupt generator 4–2
 - Periodic interrupts 4–8
 - Ports 4–45, 4–64
 - Port size 4–24, 4–65
 - Port width 4–22, 4–24
 - Reset 4–42
 - Show cycles 4–5, 4–41
 - Single-chip mode 4–3, 4–44, 4–45
 - Size signal encoding 4–22
 - Software watchdog 4–8
 - Spurious interrupts 4–5, 4–60
 - System configuration and protection 4–3
 - Three-line handshaking interface 4–22

- Using chip-select signals for interrupt acknowledge 4–69
- Single or continuous ADC conversion 6–10
- Single or multiple channel ADC conversion 6–9
- Sixteen-bit fractions 5–7
- Sixteen-bit port 4–25
- Sixteen-bit signed fractions 5–7
- Sixteen-bit (word) signed and unsigned integers 5–7
- Size signals (SIZ[1:0]) 4–22, 4–29
- Slave mode, SPI 7–13, 7–14
- Slave (factory test) mode arbitration 4–41
- Slow mode, GPT 8–17, 8–19
- SM — Saturate mode bit 5–5
- Software and idle-line arbitration, SCI 7–11
- Software watchdog 4–8
- Software watchdog ratio 4–8
- Software watchdog service sequence 4–8
- SPI pin functions 7–4
- Specific mnemonic, definition 2–5
- Spurious interrupts 4–5, 4–60
 - Cycle 4–79
 - Exception 4–60
 - Exception vector 4–7, 4–74
 - Monitor 4–7, 4–11, 4–79
 - Vector number 4–7, 4–79
- SS state between transfers 7–4
- Stack pointer extension field (SK) 2–2, 5–5
- Stack pointer (SP) 2–2, 5–3
- Standard exception processing sequence 4–38
- Standby RAM with TPU emulation (TPURAM) 10–1
 - Array base address 10–1
 - Address registers and low-power stop mode 10–1
 - Array address mapping 10–1
 - Array address space type 10–2
 - Array status during low-power stop 10–2
 - Features 3–2
 - Standby and low-power stop operation 10–2
- Standby RAM power connection 2–4, 3–7, 10–2
- Start bit, SCI 7–11
- State of GPT timer during freeze mode 8–3
- State of MODCLK pin during reset 4–13
- State of MRM following reset 11–3
- State signals 5–41
- Status flags 8–4, 8–7, 8–13
- Stop bit 8–38
- Stop disable control bit (S) 2–2
- Stop enable bit (S) 5–4
- STOP reset state, MRM 11–3
- Subroutines 5–29
- Successive approximation register, ADC 6–16
- Switching between master/slave modes 7–13
- Symbols and operators 2–1
- Synchronization to CLKOUT 4–28
- Synchronizer, GPT 8–13
- Synchronous exceptions 5–39
- Synchronous reset during MRM access 11–4
- System clock 4–2, 4–9
 - Control 4–19
 - Control multipliers 4–15
 - Frequencies 4–15
 - Low-pass filter 4–13
 - Mode select (MODCLK) signal 2–3, 3–6, 3–8, 4–18
 - Mode selection 4–64
 - Operation during low-power stop 4–11, 4–18, 4–36
 - Rate changes 4–19
 - Rate during operation 4–19
 - Reference signal 4–13
 - Signal (CLKOUT) 2–3, 3–5, 3–8
 - Signal input from external source 4–12
 - Signal duty cycle and period 4–13, 4–14
 - Signal generated externally 4–13
 - Sources 4–13
 - States 4–27
 - Synthesizer operation 4–13
 - Synthesizer power connection 2–4, 3–6, 4–14
 - VCO ramp time 4–14
 - Voltage controlled oscillator (VCO) 4–14
- System configuration and protection 4–3
- System frequencies 4–15
- System debugging 5–41
- System memory maps 3–10
- System power connections 2–4, 3–7
- System synchronized to CLKOUT 4–28
- System test 4–1

- Termination of bus cycle for bus error condition 4–35
- Termination of bus error cycles 4–37
- Three-line handshaking interface 4–22
- Three-state control pin 4–55
- Three-state control signal (TSC) 2–4, 3–6, 3–8
- Time bases, TPU 9–2
- Time functions, TPU 9–7
 - Discrete input/output 9–7
 - Input capture/input transition counter 9–7
 - Output compare 9–7
 - Period measurement with additional transition detect 9–8
 - Period measurement with missing transition detect 9–9
 - Period/pulse-width accumulator 9–10
 - Position-synchronized pulse generator 9–9
 - Pulse-width modulation 9–8
 - Stepper motor 9–9
 - Synchronized pulse-width modulation 9–8
- Time processor unit (TPU) 9–1
 - Channel control 9–13
 - Channel function select 9–14
 - Channel interrupt enable 9–13
 - Channel orthogonality 9–4
 - Channel parameters organized as 128 16-bit words 9–3
 - Channel priority 9–5, 9–16
 - Channel status 9–13
 - Channels contain identical hardware 9–4
 - Coherency 9–5
 - Components 9–2
 - Connection between TPURAM and TPU during emulation 9–5
 - Control-store ROM microcode for factory-masked functions 9–3
 - Development support and test 9–16
 - Emulation capability 9–5
 - Event timing 9–4
 - Functions related to 16-bit timebases 9–4
 - Host sequence 9–14
 - Host service 9–14
 - Interrupts 9–6
 - Low-power stop operation 9–13
 - Match and capture handled by independent channel hardware 9–4
 - Memory map contains three groups of registers 9–11
 - Microcode executed from TPURAM module 9–3
 - Microengine 9–3
 - Parameter RAM 9–3
 - Prescaler control 9–2, 9–11
 - Priority of interrupt service requests 9–6
 - Scheduler 9–3
 - System configuration 9–11
 - Time bases 9–2
 - Time functions 9–7
 - Timer Channels 9–2
 - Time required for internal CPU operations 5–36
 - Timing of BERR detection/acknowledge 4–36, 4–44
 - Timing of chip-select assertion in asynchronous mode 4–62, 4–67
 - Total A/D conversion time 6–14
 - Total execution time 5–35
 - TPU channel parameters organized as 128 16-bit words 9–3
 - TPU channels contain identical hardware 9–4
 - TPU components 9–2
 - TPU emulation capability 9–5
 - TPU interrupts 9–6
 - TPU library functions 9–5
 - TPU ROM control store 9–3, 9–5
 - Transfer and resolution, ADC 6–14
 - Transfer length options, SPI 7–14
 - Transfer delay, SPI 7–13
 - Transfer time, ADC 6–14
 - Transfer time, SPI 7–14
 - TSC during power-up reset 4–55
 - Two or more external devices as bus master 4–40
 - Two or more IARB fields with same nonzero value 4–58
 - Two sources of bus exception control cycles 4–35
 - Two-cycle bus access 4–88
 - Two-cycle external bus transfer 4–32
 - Two's complement overflow indicator (V) 2–2, 5–5
 - TXD direction with SCI transmitter disabled 7–5
 - Type of SCI wakeup 7–11
 - Types of exceptions 5–37

– U –

Uninitialized interrupt vector 7–4
Unsigned branches 5–26
Unsigned conditional branches 5–24
Unsigned left-justified conversion format, ADC 6–16
Unsigned right-justified conversion format, ADC 6–16
Use of TSC during power-up reset 4–55
User-defined interrupt vector number 7–4, 7–7, 8–7
Using chip-select for 16-bit memory bank-select 4–69
Using chip-select for interrupt acknowledge 4–69

– V –

V — Overflow flag 5–5
Value of \overline{DSACK} field 4–68
VCO ramp time 4–53
 V_{DDA} , V_{SSA} and analog ADC circuitry 6–3
 V_{DD} ramp time 4–53, 10–3
Vector numbers 5–39
 V_{RH} and V_{RL} analog reference 6–3
Voltage-controlled oscillator, clock synthesizer 4–13

– W –

Wait states 4–28, 8–3, 11–3
Watchdog clock rate 4–7
Watchdog timer 4–7
Word data type 5–7
Write cycle 4–31
Write cycles in progress during reset 4–43
Writing to GPT timer counter in freeze mode 8–11

– Z –

Z — Zero flag 2–2, 5–4, D–3

– NUMERIC –

4-bit bank address 5–6
4-bit signed integers 5–6
8-bit signed and unsigned integers 5–6
8-bit opcode with 8-bit operand 5–33
8-bit opcode with 8-bit prebyte, argument(s) 5–33
8-bit opcode with 8-bit prebyte, no argument 5–33
8-bit opcode with 4-bit index extensions 5–33
8-bit opcode with 20-bit argument 5–33

8-bit port 4–24
8-bit, 2-digit binary coded decimal numbers 5–6
16-bit byte address 5–6
16-bit fractions 5–7
16-bit port 4–24
16-bit signed fractions 5–6
16-bit (word) signed and unsigned integers 5–6
20-bit addresses 5–6
20-bit effective address 5–6
32-bit (long word) signed and unsigned integers 5–6
32-bit signed fractions 5–6
36-bit signed fixed-point numbers 5–6
6800 bus clock signal (ECLK) 2–3
68000 bus clock signal (CLKOUT) 2–3

INTRODUCTION	1
NOMENCLATURE	2
OVERVIEW	3
SINGLE-CHIP INTEGRATION MODULE	4
CENTRAL PROCESSING UNIT	5
ANALOG-TO-DIGITAL CONVERTER	6
MULTICHANNEL COMMUNICATION INTERFACE	7
GENERAL-PURPOSE TIMER	8
TIME PROCESSOR UNIT	9
STANDBY RAM WITH TPU EMULATION	10
MASKED ROM MODULE	11
ELECTRICAL CHARACTERISTICS	A
MECHANICAL DATA AND ORDERING INFORMATION	B
DEVELOPMENT SUPPORT	C
REGISTER SUMMARY	D
INDEX	I

- 1 INTRODUCTION**
- 2 NOMENCLATURE**
- 3 OVERVIEW**
- 4 SINGLE-CHIP INTEGRATION MODULE**
- 5 CENTRAL PROCESSING UNIT**
- 6 ANALOG-TO-DIGITAL CONVERTER**
- 7 MULTICHANNEL COMMUNICATION INTERFACE**
- 8 GENERAL-PURPOSE TIMER**
- 9 TIME PROCESSOR UNIT**
- 10 STANDBY RAM WITH TPU EMULATION**
- 11 MASKED ROM MODULE**
- A ELECTRICAL CHARACTERISTICS**
- B MECHANICAL DATA AND ORDERING INFORMATION**
- C DEVELOPMENT SUPPORT**
- D REGISTER SUMMARY**
- I INDEX**



MOTOROLA

Literature Distribution Centers:

USA: Motorola Literature Distribution; P.O. Box 20912; Phoenix, Arizona 85036

EUROPE: Motorola Ltd.; European Literature Centre; 88 Tanners Drive, Blakelands, Milton Keynes, MK14 5BP, England.

JAPAN: Nippon Motorola Ltd.; 4-32-1, Nishi-Gotanda, Shinagawa-ku, Tokyo 141, Japan.

ASIA PACIFIC: Motorola Semiconductors H.K. Ltd.; Silicon Harbour Center, No. 2 Dai King Street, Tai Po Industrial Estate,

Tai Po, N.T., Hong Kong.

MC68HC16Y1UM/AD

