M68000 FAMILY REFERENCE

Motorola reserves the right to make changes without further notice to any products herein to improve reliability, function or design. Motorola does not assume any liability arising out of the application or use of any product or circuit described herein; neither does it convey any license under its patent rights nor the rights of others. Motorola and Ⓜ are registered trademarks of Motorola, Inc. Motorola, Inc. is an Equal Employment Opportunity/Affirmative Action Employer.

Motorola, Inc. general policy does not recommend the use of its components in life support applications wherein in failure or malfunction of the component may directly threaten life or injury. Per Motorola Terms and Conditions of Sale, the user of Motorola components in life support applications assumes all risks of such use and indemnifies Motorola against all damages.

Printed in U.S.A.

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

MOTOROLA'S M68000 FAMILY 1

SECTION 1
MOTOROLA'S M68000 FAMILY

In this manual, descriptions of the devices in the M68000 Family are grouped in six categories. The processor group includes 8-, 16-, and 32-bit microprocessors, all using 32-bit internal architecture with 17 general-purpose data and address registers. The coprocessor group includes memory management and floating-point coprocessors. The direct memory access (DMA) controller group, the data communication device group, the network device group, and the general-purpose peripheral device group complete the M68000 Family.

The MC68000 is the original microprocessor unit (MPU) of the M68000 Family. It has a 16-bit data bus and many flexible addressing modes. With a 24-bit address bus, it provides a 16 megabyte linear address space.

The MC68HC000 is the HCMOS version of the MC68000, with all its functions and performance. The maximum power dissipation of the 12.5-MHz MC68HC000 is just 0.175 watt, one-tenth of the MC68000's power requirement.

The MC68008 has an 8-bit data bus for designs that need to conserve PC board space. The 48-pin DIP version has a 20-bit address bus providing a one megabyte address space. A 52-pin quad-pack version uses a 22-bit address bus to support four megabytes of linear address space.

The MC68010 uses a 16-bit data bus with the 32-bit internal architecture common to the M68000 Family. Multiple addressing modes provide flexible addressing. The MC68010 uses instruction continuation to permit pre-emption for page swapping in virtual memory systems. Its loop-mode operation allows faster execution of tight software loops. The 24-bit address bus provides 16 megabytes of linear address space.

The MC68020 uses a 32-bit data bus and supports a coprocessor interface for as many as eight coprocessors. It contains a 256-byte, on-chip instruction cache and offers additional addressing modes. The MC68020's 32-bit address bus provides four gigabytes of linear address space.

The newest and most powerful member of the processor group, the MC68030, has on-chip, demand-paged memory management. The increased parallelism of the MC68030 results from two, independent, 32-bit address buses and two, independent, 32-bit data buses, which allow the central processor unit (CPU), data cache, instruction cache, memory management unit (MMU), and bus controller to operate in parallel. While the MMU provides address translation for a variety of page sizes, it also supports transparently addressed windows in memory where direct access is required without address translation. Separate on-chip data and instruction caches, 256 bytes each, provide increased performance. The burst fill mode for these caches further enhances throughput. Also, the asynchronous bus of the M68000 Family is supplemented by synchronous bus capabilities to provide a two-clock physical bus cycle.

The MC68040 microprocessor is previewed in this manual. In addition to significantly improving on the features and capabilities of the MC68030, the MC68040 provides floating-point arithmetic, using a subset of the floating-point instructions supported by the MC68881 floating-point coprocessor.

The coprocessor group includes the MC68851 paged memory managment unit (PMMU), the MC68881 floating-point coprocessor and the MC68882 enhanced floating-point coprocessor. The MC68851 PMMU provides full support for a demand-paged virtual memory environment. It supports a four-gigabyte address space and programmable page sizes from 256 to 32K bytes. An on-chip address translation cache minimizes translation delays. Both the MC68881 and the MC68882 coprocessors conform to a full implementation of the IEEE Standard for Binary Floating-Point Arithmetic. They perform more than 40 types of transcendental and nontranscendental functions in addition to basic add, subtract, multiply, and divide operations. The functions include root values, trigonometric functions, exponentials, hyperbolics, and logarithms. All functions are calculated to 80 bits of precision in hardware. The enhanced MC68882 has dual-ported registers and an advanced pipeline that allows execution of multiple instructions in parallel (more than twice the floating-point performance of the MC68881). Although the MC68881 and MC68882 are intended primarily as coprocessors for the MC68020 and MC68030, they can be used as peripherals with all MPUs of the M68000 Family as well as other MPUs.

The DMA controller group consists of the MC68440 and MC68442 dual DMA controllers and the MC68450 DMA controller. The MC68440 dual DMA controller performs memory-to-memory, peripheral-to-memory, and memory-to-peripheral transfers through each of two, completely independent DMA channels with minimum intervention from the MPU. The MC68442 is an expanded version that supports a four-gigabyte addressing range. The MC68450 DMA controller has four, completely independent DMA channels that use sequential and linked-array chained addressing. The MC68450 maintains hardware and software compatibility with the MC68440.

The data communication group includes the MC2681 dual universal asynchronous receiver/transmitter (DUART), the MC68652/MC2652 multiprotocol communication controller (MPCC), and the MC68681 DUART. The MC2681 DUART contains two, completely independent, full-duplex asynchronous receiver/transmitter channels that interface to non-Motorola buses. Receiver data registers are quadruple buffered, and transmission data registers are double buffered; each channel has an independently selectable baud rate. The maximum transfer rate is 1 Megabyte per second. The MC68652/MC2652 MPCC is a single-channel serial data device that recognizes byte-control and bit-oriented protocols. It transfers data of 8- or 16-bit widths at a maximum two Mbit/second (Mbps) rate with cyclical redundancy check (CRC) error detection. The MC68681 DUART has all the capabilities and features of the MC2681 and interfaces with the M68000 Family processors.

The network device group consists of the MC68184 broadband interface controller (BIC), the MC68194 carrierband modem (CBM), the MC68605 X.25 protocol controller (XPC), the MC68606 multilink access procedure (MLAPD) protocol controller, and the MC68824 token bus controller (TBC). The MC68184 BIC implements the IEEE 802.4 broadband physical layer of the International Standards Organization/Open Systems Interface (ISO/OSI) communication model for standardized multivendor data communications networking. It supports high-speed data rates to 10 Mbps. With the MC68824 TBC, the BIC implements layers one and two of the OSI communication model. The MC68194 CBM is an advanced bipolar LSI device that implements the IEEE 8024 Phase Coherent Physical Layer. It modulates the information from the serial interface and transmits this signal onto the network cable. The CBM also receives signals from the network, demodulates the information, and passes it to the TBC over the serial interface. The MC68605 XPC implements the 1984 CCITT X.25 Recommendation Data Link Procedure (level two) LAPD. It independently performs higher level communications functions such as frame sequencing, retransmission, flow control, retries limit, and timeout conditions for data rates as high as 10 Mbps in addition to the lower-level functions: HDLC framing, CRC generation/checking, and zero insertion/deletion. As a bus master, the XPC uses on-chip DMA capability and two, 22-bit, first in-first out queues (FIFOs) for transferring frames to and from memory. The MC68606 MLAPD protocol controller implements the LAPD protocol for use at the link layer (OSI Layer 2) for both signaling and data transfer

applications in integrated services digital network (ISDN) configurations. An on-chip DMA controller transfers data packets to and from a buffer memory. The MC68824 TBC implements the IEEE 8024 media access control sublayer of the OSI data link layer as specified by General Motors' Manufacturing Automation Protocol (MAP). It supports serial data rates of 1, 5, and 10 Mbps and relieves the host processor of the frame-formatting and token-management functions.

The general-purpose peripheral device group consists of the MC68153 bus interrupter module (BIM), the MC68230 parallel interface/timer (PI/T), the MC68452 bus arbitration module (BAM), and the MC68901 multifunction peripheral. The MC68153 BIM is an interface between the M68000 microcomputer system bus and slave devices that require interrupt capability. It routes four, independent sources of interrupt requests to any of the seven M68000 interrupt levels, and is VMEbus compatible. The MC68230 PI/T provides double-buffered, 8- or 16-bit parallel interfaces and a 24-bit, system-oriented timer with a 5-bit prescaler. The MC68452 BAM arbitrates control of the M68000 bus between as many as eight bus masters. The MC68901 multifunction peripheral provides a full-function, single-channel USART, an eight-source interrupt controller, four eight-bit timers, and eight parallel I/O lines.

VERSAbus is a trademark of Motorola Inc.

SELECTOR GUIDE 2

SECTION 2
SELECTOR GUIDE

2

To guide the designer in selecting the components for a system, Table 2-1 lists the processors, coprocessors, and other devices of the M68000 Family, with a brief description of each. Columns in Table 2-1 indicate available package types and operating frequencies. The devices are listed in numeric order.

Table 2-1. Selector Guide (Sheet 1 of 2)

Device No.	Description	Package Designation						Operating Frequency						
		L	LC	P	R	RC	FN	8	10	12	16	20	25	33
MC68000 MPU	16-Bit Data Bus, 16M-Byte Address Space	X	X	X	X	X	X	X	X	X	X			
M68HC000 MPU	HCMOS Version of MC68000	X	X	X	X	X	X	X	X	X	X			
MC68008 MPU	8-Bit Data Bus. 1M-Byte Address Space		X	X			X	X	X					
MC68010 MPU	16-Bit Data Bus, Supports Virtual Memory, 16M-Byte Address Space	X	X	X	X	X	X	X	X	X				
MC68020 MPU	32-Bit Data Bus, Instruction Cache, Coprocessor Interface, 4G-Byte Address Space				X	X				X	X	X	X	X
MC68030 MPU	32-Bit Data Bus, Data and Instruction Caches, Memory Management Unit, Coprocessor Interface, 4G-Byte Address Space				X	X					X	X	X	X
MC68153 BIM	Four Interrupt Sources, Eight Programmable Read/Write Registers	X		X							X			
MC68184 BIC	IEEE 802.4 Broadband Physical Layer Station Management, 20 Lines, 13 User-Defined Outputs, Up to 10 Mbps			X										
MC68194 CBM	IEEE 802.4 Single-Channel Phase-Coherent FSK Physical Layer, 1 to 10 Mbps						X							
MC68230 PI/T	Bit I/O, Unidirectional and Bidirectional 8- and 16-Bit Modes, 24-Bit Timer		X	X			X	X	X					
MC68440 DMA Controller	Dual Channel, Up to 5M-Bytes/ Second Transfer Rate, 16M-Byte Address Range	X	X	X	X	X	X	X	X					
MC68442 DMA Controller	Same Features as MC68440 but with 4G-Byte Address Range					X	X	X	X					
MC68450 DMA Controller	Four Channel, Up to 5M-Bytes/Second Transfer Rate, 16M-Byte Address Range	X	X		X	X		X	X					
MC68452 BAM	Arbitrates for 8 Bus Users, 52 ns., Maximum Aribtration Time	X		X										
MC68605 XPC	CCITT X.25 Recommendation LAPB Procedure, 16- and 32-Bit CRC, 8- and 16-Bit Data Bus, Up to 10 Mbps Synchronous Serial Data Rate, DMA Transfers				X	X	X		X	X	X			

Table 2-1. Selector Guide (Sheet 2 of 2)

Device No.	Description	Package Designation						Operating Frequency						
		L	LC	P	R	RC	FN	8	10	12	16	20	25	33
MC68606 MLAPD Controller	CCITT Q.920/Q.921 LAPD, Up to 8192 Logical Links, Serial Bit Stream Aggregate in Excess of 2.048 Mbps				X	X	X			X	X			
MC68652/ MC2652 MPCC	BOP or BCP, 8- or 16-Bit Data Bus, Up to 2 Mbps Data Rate, CRC-16 or VRC			X										
MC2681 DUART	Selectable Baud Rate, Non-Motorola Interface			X			X							
MC68681 DUART	Programmable Data Format, 6-Bit Input Port, 8-Bit Output Port	X		X			X							
MC68824 TBC	IEEE 802.4 MAC, 32-Bit Address Bus, DMA Transfers, 8- or 16-Bit Transfers				X	X	X		X	X	X	X		
MC68851 PMMU	32-Bit Logical and Physical Addresses, Coprocessor Interface					X				X	X	X		
MC68881 FPCP	IEEE 754 Standard, 67-Bit Arithmetic Unit, 8-, 16-, or 32-Bit Data Bus					X	X			X	X	X	X	
MC68882 EFCP	All Features of the MC68881 with Enhanced Performance					X	X				X	X	X	X
MC68901 MFP	Eight Programmable I/O Pins, 16-Source Interrupt Controller, Four Timers		X	X			X							

PROCESSORS 3

Technical Summary
16-/32-Bit Microprocessor

This document contains both a summary of the MC68000 as well as a detailed set of parametrics. The purpose is twofold - to provide an introduction to the MC68000 and support for the sophisticated user. For detailed information on the MC68000, refer to the *MC68000 16-/32-Bit Microprossor Advance Information Data Sheet.*

The MC68000 is the first implementation of the M68000 16/32 microprocessor architecture. The MC68000 has a 16-bit data bus and 24-bit address bus while the full architecture provides for 32-bit address and data buses. It is completely code-compatible with the MC68008 8-bit data bus implementation of the M68000 and is upward code compatible to the MC68010/MC68012 virtual extensions and the MC68020 32-bit implementation of the architecture. Any user-mode programs written using the MC68000 instruction set will run unchanged on the MC68008, MC68010, MC68020. This is possible because the user programming model is identical for all five processors and the instruction sets are proper sub-sets of the complete architecture. Resources available to the MC68000 user consists of the following:

- 17 32-Bit Data and Address Registers
- 16 Megabyte Direct Addressing Range
- 56 Powerful Instruction Types
- Operations on Five Main Data Types
- Memory Mapped I/O
- 14 Addressing Modes

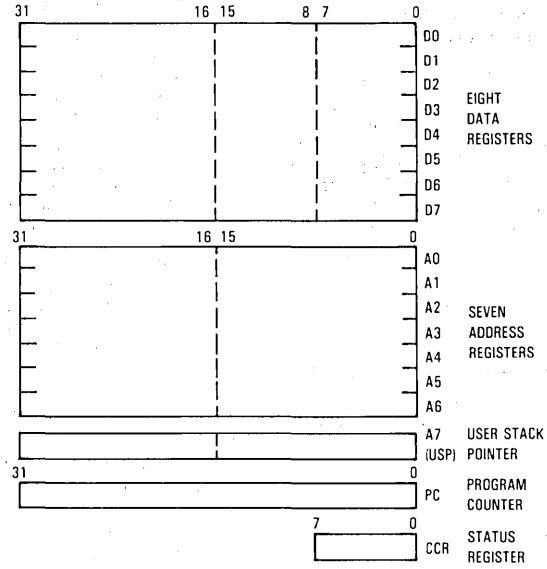


Figure 1. User Programming Model

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

INTRODUCTION

As shown in the user programming model (Figure 1), the MC68000 offers 16 32-bit registers and a 32-bit program counter. The first eight registers (D0-D7) are used as data registers for byte (8-bit), word (16-bit), and long word (32-bit) operations. The second set of seven registers (A0-A6) and the user stack pointer (USP) may be used as software stack pointers and base address registers. In addition, the registers may be used for word and long word operations. All of the 16 registers may be used as index registers.

In supervisor mode, the upper byte of the status register and the supervisor stack pointer (SSP) are also available to the programmer. These registers are shown in Figure 2.

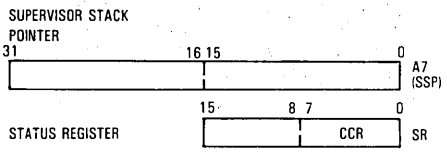


Figure 2. Supervisor Programming Model Supplement

The status register (Figure 3) contains the interrupt mask (eight levels available) as well as the condition codes: extend (X), negative (N), zero (Z), overflow (V), and carry (C). Additional status bits indicate that the processor is in a trace (T) mode and in a supervisor (S) or user state.

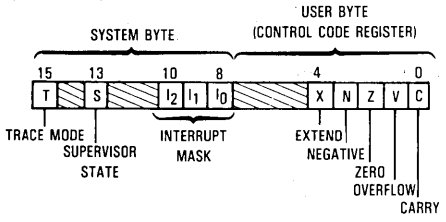


Figure 3. Status Register

DATA TYPES AND ADDRESSING MODES

Five basic data types are supported. These data types are:

- Bits
- BCD Digits (4-Bits)
- Bytes (8 Bits)
- Words (16 Bits)
- Long Words (32 Bits)

In addition, operations on other data types such as memory addresses, status word data, etc. are provided in the instruction set.

The 14 addressing modes, shown in Table 1, include six basic types:

- Register Direct
- Register Indirect
- Absolute
- Program Counter Relative
- Immediate
- Implied

Included in the register indirect addressing modes is a capability to do postincrementing, predecrementing, offsetting, and indexing. The program counter relative mode can also be modified via indexing and offsetting.

Table 1. Addressing Modes

Addressing Modes	Syntax
Register Direct Addressing	
Data Register Direct	Dn
Address Register Direct	An
Absolute Data Addressing	
Absolute Short	xxx.W
Absolute Long	xxx.L
Program Counter Relative Addressing	
Relative with Offset	d_{16}(PC)
Relative with Index Offset	d_8(PC,Xn)
Register Indirect Addressing	
Register Indirect	(An)
Postincrement Register Indirect	(An)+
Predecrement Register Indirect	-(An)
Register Indirect with Offset	d_{16}(An)
Indexed Register Indirect with Offset	d_8(An,Xn)
Immediate Data Addressing	
Immediate	#xxx
Quick Immediate	#1-#8
Implied Addressing	
Implied Register	SR USP SP PC

NOTES:
Dn = Data Register
An = Address Register
Xn = Address of Data Register used as Index Register
SR = Status Register
PC = Program Counter
SP = Stack Pointer
USP = User Stack Pointer
( ) = Effective Address
d_8 = 8-Bit Offset (Displacement)
D_{16} = 16-Bit Offset (Displacement)
#xxx = Immediate Data

INSTRUCTION SET OVERVIEW

The MC68000 instruction set is shown in Table 2. Some additional instructions are variations, or sub-sets, of these and they appear in Table 3. Special emphasis has been given to the instruction set's support of structured high-level languages to facilitate ease of programming. Each instruction, with few exceptions, operates on bytes, words, and long words and most instructions can use any of the 14 addressing modes. Combining instruction types, data types, and addressing modes, over 1000 useful instructions are provided. These instructions include signed and unsigned, multiply and divide, "quick" arithmetic operations, BCD arithmetic, and expanded operations (through traps).

Table 2. Instruction Set Summary

Mnemonic	Description
ABCD	Add Decimal With Extend
ADD	Add
AND	Logical AND
ASL	Arithmetic Shift Left
ASR	Arithmetic Shift Right
Bcc	Branch Conditionally
BCHG	Bit Test and Change
BCLR	Bit Test and Clear
BRA	Branch Always
BSET	Bit Test and Set
BSR	Branch to Subroutine
BTST	Bit Test
CHK	Check Register Against Bounds
CLR	Clear Operand
CMP	Compare
DBcc	Test Condition, Decrement and Branch
DIVS	Signed Divide
DIVU	Unsigned Divide
EOR	Exclusive OR
EXG	Exchange Registers
EXT	Sign Extend
JMP	Jump
JSR	Jump to Subroutine
LEA	Lead Effective Address
LINK	Link Stack
LSL	Logical Shift Left
LSR	Logical Shift Right
MOVE	Move
MULS	Signed Multiply
MULU	Unsigned Multiply
NBCD	Negate Decimal with Extend
NEG	Negate
NOP	No Operation
NOT	One's Complement
OR	Logical OR
PEA	Push Effective Address
RESET	Reset External Devices
ROL	Rotate Left without Extend
ROR	Rotate Right without Extend
ROXL	Rotate Left with Extend
ROXR	Rotate Right with Extend
RTE	Return from Exception
RTR	Return and Restore
RTS	Return from Subroutine
SBCD	Subtract Decimal with Extend
Scc	Set Conditional
STOP	Stop
SUB	Subtract
SWAP	Swap Data Register Halves
TAS	Test and Set Operand
TRAP	Trap
TRAPV	Trap on Overflow
TST	Test
UNLK	Unlink

Table 3. Variations of Instruction Types

Instruction Type	Variation	Description
ADD	ADD	Add
	ADDA	Add Address
	ADDQ	Add Quick
	ADDI	Add Immediate
	ADDX	Add with Extend
AND	AND	Logical AND
	ANDI	AND Immediate
	ANDI to CCR	AND Immediate to Condition Codes
	ANDI to SR	AND Immediate to Status Register
CMP	CMP	Compare
	CMPA	Compare Address
	CMPM	Compare Memory
	CMPI	Compare Immediate
EOR	EOR	Exclusive OR
	EORI	Exclusive OR Immediate
	EORI to CCR	Exclusive OR Immediate to Condition Codes
	EORI to SR	Exclusive OR Immediate to Status Register
MOVE	MOVE	Move
	MOVEA	Move Address
	MOVEM	Move Multiple Registers
	MOVEP	Move Peripheral Data
	MOVEQ	Move Quick
	MOVE from SR	Move from Status Register
	MOVE to SR	Move to Status Register
	MOVE to CCR	Move to Condition Codes
	MOVE USP	Move User Stack Pointer
NEG	NEG	Negate
	NEGX	Negate with Extend
OR	OR	Logical OR
	ORI	OR Immediate
	ORI to CCR	OR Immediate to Condition Codes
	ORI to SR	OR Immediate to Status Register
SUB	SUB	Subtract
	SUBA	Subtract Address
	SUBI	Subtract Immediate
	SUBQ	Subtract Quick
	SUBX	Subtract with Extend

SIGNAL DESCRIPTION

The input and output signals are illustrated functionally in Figure 4 and are described in the following paragraphs.

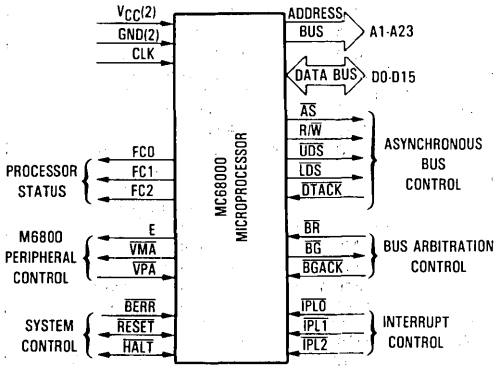


Figure 4. Input and Output Signals

ADDRESS BUS (A1 THROUGH A23)

This 32-bit, unidirectional, three-state bus is capable of addressing 16 megabytes of data. It provides the address for bus operation during all cycles except interrupt cycles. During interrupt cycles, address lines A1, A2, and A3 provide information about what level interrupt is being serviced while address lines A4 through A23 are set to a logic high.

DATA BUS (D0 THROUGH D15)

This 16-bit, bidirectional, three-state bus is the general purpose data path. It can transfer and accept data in either word or byte lenght. During an interrupt acknowledge cycle, the external device supplies the vector number on data lines D0-D7.

ASYNCHRONOUS BUS CONTROL

Asynchronous data transfers are handled using the following control signals: address strobe, read/write, upper and lower data strobes, and data transfer acknowledge. These signals are explained in the following paragraphs.

Address Strobe ($\overline{AS}$)

This signal indicates that there is a valid address on the address bus.

Read/Write (R/$\overline{W}$)

This signal defines the data bus transfer as a read or write cycle. The R/$\overline{W}$ signal also works in conjunction with the data strobes as explained in the following paragraph.

Upper and Lower Data Strobe ($\overline{UDS}$, $\overline{LDS}$)

These signals control the flow of data on the data bus, as shown in Table 4. When the R/$\overline{W}$ line is high, the processor will read from the data bus as indicated. When the R/$\overline{W}$ line is low, the processor will write to the data bus as shown.

Data Transfer Acknowledge ($\overline{DTACK}$)

This input indicates that the data transfer is completed. When the processor recognizes $\overline{DTACK}$ during a read cycle, data is latched and the bus cycle terminated. When $\overline{DTACK}$ is recognized during a write cycle, the bus cycle is terminated.

BUS ARBITRATION CONTROL

The three signals, bus request, bus grant, and bus grant acknowledge, form a bus arbitration circuit to determine which device will be the bus master device.

Bus Request ($\overline{BR}$)

This input is wire-ORed with all other devices that could be bus masters. This input indicates to the processor that some other device desires to become the bus master.

Bus Grant ($\overline{BG}$)

This output indicates to all other potential bus master devices that the processor will release bus control at the end of the current bus cycle.

Bus Grant Acknoweldge ($\overline{BGACK}$)

This input indicates that some other device has become the bus master. This signal should not be asserted until the following four conditions are met:

1. a bus grant has been received,
2. address strobe is inactive which indicates that the microprocessor is not using the bus,
3. data transfer acknowledge is inactive which indicates that neither memory nor peripherals are using the bus, and
4. bus grant acknowledge is inactive which indicates that no other device is still claiming bus mastership.

Table 4. Data Strobe Control of Data Bus

$\overline{UDS}$	$\overline{LDS}$	R/$\overline{W}$	D8-D15	D0-D7
High	High	—	No Valid Data	No Valid Data
Low	Low	High	Valid Data Bits 8-15	Valid Data Bits 0-7
High	Low	High	No Valid Data	Valid Data Bits 0-7
Low	High	High	Valid Data Bits 8-15	No Valid Data
Low	Low	Low	Valid Data Bits 8-15	Valid Data Bits 0-7
High	Low	Low	Valid Data Bits 0-7*	Valid Data Bits 0-7
Low	High	Low	Valid Data Bits 8-15	Valid Data Bits 8-15*

*These conditions are a result of current implementation and may not appear on future devices.

INTERRUPT CONTROL ($\overline{IPL0}$, $\overline{IPL1}$, $\overline{IPL2}$)

These input pins indiacte the encoded priority level of the device requesting an interrupt. Level seven is the highest priority while level zero indicates that no interrupts are requested. Level seven cannot be masked. The least significant bit is given in $\overline{IPL0}$ and the most significant bit is contained in $\overline{IPL2}$. These lines must remain stable until the processor signals interrupt acknowledge (FC0-FC2 are all high) to insure that the interrupt is recognized.

SYSTEM CONTROL

The system control inputs are used to either reset or halt the processor and to indicate to the processor that bus errors have occurred. The three system control inputs are explained in the following paragraphs.

Bus Error ($\overline{BERR}$)

This input informs the processor that there is a problem with the cycle currently being executed. Problems may be a result of:

1. nonresponding devices,
2. interrupt vector number acquisition failure,
3. illegal access request as determined by a memory management unit, or
4. other application dependent errors.

The bus error signal interacts with the halt signal to determine if the current bus cycle should be re-executed or if exception processing should be performed.

Reset ($\overline{RESET}$)

This bidirectional signal line acts to reset (start a system initialization sequence) the processor in response to an external reset signal. An internally generated reset (result of a $\overline{RESET}$ instruction) causes all external devices to be reset and the internal state of the processor is not affected. A total system reset (processor and external devices) is the result of external $\overline{HALT}$ and $\overline{RESET}$ signals applied at the same time.

Halt ($\overline{HALT}$)

When this bidirectional line is driven by an external device, it will cause the processor to stop at the completion of the current bus cycle. When the processor has been halted using this input, all control signals are inactive and all three-state lines are put in their high-impedance state.

When the processor has stopped executing instructions, such as in a double bus fault condition, the $\overline{HALT}$ line is driven by the processor to indicate to external devices that the processor has stopped.

M6800 PERIPHERAL CONTROL

These control signals are used to allow the interfacing of synchronous M6800 peripheral devices with the asynchronous MC68000. These signals are explained in the following paragraphs.

Enable (E)

This signal is the standard enable signal common to all M6800 type peripheral devices. The period for this output is ten MC68000 clock periods (six clocks low, four clocks high). Enable is generated by an internal ring counter which may come up in any state (i.e., at power on, it is impossible to guarantee phase relationship of E to CLK). E is a free-running clock and runs regardless of the state of the bus on the MPU.

Valid Peripheral Address ($\overline{VPA}$)

This input indicates that the device or region addressed is an M68000 Family device or region addressed is an M68000 Family device and that data transfer should be synchronized with the enable (E) signal. This input also indicates that the processor should use automatic vectoring for an interrupt during an IACK cycle.

Valid Memory Address ($\overline{VMA}$)

This output is used to indicate to M68000 peripheral devices that there is a valid address on the address bus and the processor is synchronized to enable. This signal only responds to a valid peripheral address ($\overline{VPA}$) input which indicates that the peripheral is an M68000 Family device.

PROCESSOR STATUS (FC0, FC1, FC2)

These function code outputs indicate the state (user or supervisor) and the cycle type currently being executed, as shown in Table 5. The information indicated by the function code outputs is valid whenever address strobe ($\overline{AS}$) is active.

Table 5. Function Code Outputs

Function Code Output			Cycle Time
FC2	FC1	FC0	
Low	Low	Low	(Undefined, Reserved)
Low	Low	High	User Data
Low	High	Low	User Program
Low	High	High	(Undefined, Reserved)
High	Low	Low	(Undefined, Reserved)
High	Low	High	Supervisor Data
High	High	Low	Supervisor Program
High	High	High	Interrupt Acknowledge

CLOCK (CLK)

The clock input is a TTL-compatible signal that is internally buffered for development of the internal clocks needed by the processor. The clock input should not be gated off at any time and the clock signal must conform to minimum and maximum pulse width times.

DATA TRANSFER OPERATIONS

Transfer of data between devices involves the following leads:

1. address bus A1 through A23,
2. data bus D0 through D15, and
3. control signals.

The address and data buses are separate parallel buses used to transfer data using an asynchronous bus structure. In all cycles, the bus master assumes responsibility for deskewing all signals it issues at both the start and end of a cycle. In addition, the bus master is responsible for deskewing the acknowledge and data signals from the slave device.

The following paragraphs explain the read, write, and read-modify-write cycles. The indivisible read-modify-write cycle is the method used by the MC68000 for interlocked multiprocessor communications.

READ CYCLE

During a read cycle, the processor receives data from the memory of a peripheral device. The processor reads bytes of data in all cases. If the instruction specifies a word (or double word) operation, the processor reads both upper and lower bytes simultaneously by asserting both upper and lower data strobes. When the instruction specifies byte operation, the processor uses an internal A0 bit to determine which byte to read and then issues the data strobe required for that byte. For byte operations, when the A0 bit equals zero, the upper data strobe is issued. When the A0 bit equals one, the lower data strobe is issued. When the data is received, the processor correctly positions it internally.

WRITE CYCLE

During a write cycle, the processor sends data to either the memory or a peripheral device. The processor writes bytes of data in all cases. If the instruction specifies a word operation, the processor writes both bytes. When the instruction specifies a byte operation, the processor uses an internal A0 bit to determine which byte to write and then issues the data strobe required for that byte. For byte operations, when the A0 bit equals zero, the upper data strobe is issued. When the A0 bit equals one, the lower data strobe is issued.

READ-MODIFY-WRITE CYCLE

The read-modify-write cycle performs a read, modifies the data in the arithmetic-logic unit, and writes the data back to the same address. In the MC68000, this cycle is indivisible in that the address strobe is asserted throughout the entire cycle. The test and set (TAS) instruction uses this cycle to provide meaningful communication between processors in a multiple processor environment. This instruction is the only instruction that uses the read-modify-write cycles and since the test and set instruction only operates on bytes, all read-modify-write cycles are byte operations.

PROCESSING STATES

The MC68000 is always in one of three processing states: normal, exception, or halted.

NORMAL PROCESSING

The normal processing state is that associated with instruction execution; the memory references are to fetch instructions and operands, and to store results. A special case of normal state is the stopped state which the processor enters when a stop instruction is executed. In this state, no further references are made.

EXCEPTION PROCESSING

The exception processing state is associated with interrupts, trap instructions, tracing, and other exception conditions. The exception may be internally generated by an instruction or by an unusual condition arising during the execution of an instruction. Externally, exception processing can be forced by an interrupt, by a bus error, or by a reset. Exception processing is designed to provide an efficient context switch so that the processor may handle unusual conditions.

HALTED PROCESSING

The halted processing state is an indication of catastrophic hardware failure. For example, if during the exception processing of a bus error another bus error occurs, the processor assumes that the system is unusable and halts. Only an external reset can restart a halted processor. Note that a processor in the stopped state is not in the halted state, nor vice versa.

INTERFACE WITH M6800 PERIPHERALS

Motorola's extensive line of M6800 peripherals are directly compatible with the MC68000. Some of these devices that are particularly useful are:

MC6821	Peripheral Interface Adapter
MC6840	Programmable Timer Module
MC6843	Floppy Disk Controller
MC6845	CRT Controller
MC6850	Asynchronous Communications Interface Adapter
MC6854	Advanced Data Link Controller

To interface the synchronous M6800 peripherals with the asynchronous MC68000, the processor modifies its bus cycle to meet the M6800 cycle requirements whenever an M6800 device address is detected. This is possible since both the processors use memory mapped I/O.

ELECTRICAL SPECIFICATIONS

MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Supply Voltage	V_{CC}	−0.3 to +7.0	V
Input Voltage	V_{in}	−0.3 to +7.0	V
Operating Temperature Range MC68000 MC68000C	T_A	T_L to T_H 0 to 70 −40 to 85	°C
Storage Temperature	T_{stg}	−55 to 150	°C

The device contains circuitry to protect the inputs against damage due to high static voltages or electric fields; however, normal precautions should be taken to avoid application of voltages higher than maximum-rated voltages to these high-impedance circuits. Tying unused inputs to the appropriate logic voltage level (e.g., either GND or V_{CC}) enhances reliability of operation.

3

THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Symbol	Value	Rating
Thermal Resistance (Still Air) Ceramic, Type L/LC Ceramic, Type R/RC Plastic, Type P Plastic, Type FN	θ_{JA}	 30 33 30 45	θ_{JC}	 15* 15 15* 25*	°C/W

*Estimated

DC ELECTRICAL CHARACTERISTICS (V_{CC} = 5.0 Vdc ± 5%; GND = 0 Vdc; $T_A = T_L$ to T_H)

Characteristic		Symbol	Min	Max	Unit
Input High Voltage		V_{IH}	2.0	V_{CC}	V
Input Low Voltage		V_{IL}	GND −0.3	0.8	V
Input Leakage Current @ 5.25 V	$\overline{BERR}$, $\overline{BGACK}$, $\overline{BR}$, $\overline{DTACK}$, CLK, $\overline{IPL0}$-$\overline{IPL2}$, $\overline{VPA}$ $\overline{HALT}$, $\overline{RESET}$	I_{IN}	— —	2.5 20	μA
Three-State (Off State) Input Current @ 2.4 V/0.4 V	$\overline{AS}$, A1-A23, D0-D15, FC0-FC2, $\overline{LDS}$, R/$\overline{W}$, $\overline{UDS}$, $\overline{VMA}$	I_{TSI}	—	20	μA
Output High Voltage (I_{OH} = −400 μA) (I_{OH} = −400 μA)	E* E, $\overline{AS}$, A1-A23, $\overline{BG}$, D0-D15, FC0-FC2, $\overline{LDS}$, R/$\overline{W}$, $\overline{UDS}$, $\overline{VMA}$	V_{OH}	V_{CC} − 0.75 2.4	— 2.4	V
Output Low Voltage (I_{OL} = 1.6 mA) (I_{OL} = 3.2 mA) (I_{OL} = 5.0 mA) (I_{OL} = 5.3 mA)	 $\overline{HALT}$ A1-A23, $\overline{BG}$, FC0-FC2 $\overline{RESET}$ E, $\overline{AS}$, D0-D15, $\overline{LDS}$, R/$\overline{W}$, $\overline{UDS}$, $\overline{VMA}$	V_{OL}	 — — — —	 0.5 0.5 0.5 0.5	V
Power Dissipation (see **POWER CONSIDERATIONS**)		P_D***	—	—	W
Capacitance (V_{in} = 0 V, T_A = 25°C, Frequency = 1 MHz)**		C_{in}	—	20.0	pF
Load Capacitance	$\overline{HALT}$ All Others	C_L	— —	70 130	pF

*With external pullup resistor of 1.1 Ω.
**Capacitance is periodically sampled rather than 100% tested.
***During normal operation instantaneous V_{CC} current requirements may be as high as 1.5 A.

POWER CONSIDERATIONS

The average die-junction temperature, T_J, in °C can be obtained from:

$$T_J = T_A + (P_D \cdot \theta_{JA}) \quad (1)$$

where:

T_A = Ambient Temperature, °C
θ_{JA} = Package Thermal Resistance, Junction-to-Ambient, °C/W
$P_D = P_{INT} + P_{I/O}$
$P_{INT} = I_{CC} \times V_{CC}$, 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 appropriate relationship between P_D and T_J (if $P_{I/O}$ is neglected) is:

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

Solving equations (1) and (2) for K gives:

$$K = P_D \cdot (T_A + 273°C) + \theta_{JA} \cdot P_D^2 \quad (3)$$

where K is a constant pertaining to the particular part. K can be determined from equation (3) by measuring P_D (at thermal 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.

The curve shown in Figure 11-1 gives the graphic solution to the above equations for the specified power dissipation of 1.5 watts over the ambient temperature range of −55 °C to 125 °C using a maximum θ_{JA} of 45 °C/W. Ambient temperature is that of the still air surrounding the device. Lower values of θ_{JA} cause the curve to shift downward slightly; for instance, for θ_{JA} of 40 °/W, the curve is just below 1.4 watts at 25 °C.

The total thermal resistance of a package (θ_{JA}) can be separated into two components, θ_{JC} and θ_{CA}, representing the barrier to heat flow from the semiconductor junction to the package (case) surface (θ_{JC}) and from the case to the outside ambient air (θ_{CA}). These terms are related by the equation:

$$\theta_{JA} = \theta_{JC} + \theta_{CA} \quad (4)$$

θ_{JC} is device related and cannot be influenced by the user. However, θ_{CA} is user dependent and can be minimized by such thermal management techniques as heat sinks, ambient air cooling, and thermal convection. Thus, good thermal management on the part of the user can significantly reduce θ_{CA} so that θ_{JA} approximately equals θ_{JC}. Substitution of θ_{JC} for θ_{JA} in equation 1 results in a lower semiconductor junction temperature.

Table 6 summarizes maximum power dissipation and average junction temperature for the curve drawn in Figure 5, using the minimum and maximum values of ambient temperature for different packages and substituting θ_{JC} for θ_{JA} (assuming good thermal management). Table 7 provides the maximum power dissipation and average junction temperature assuming that no thermal management is applied (i.e., still air).

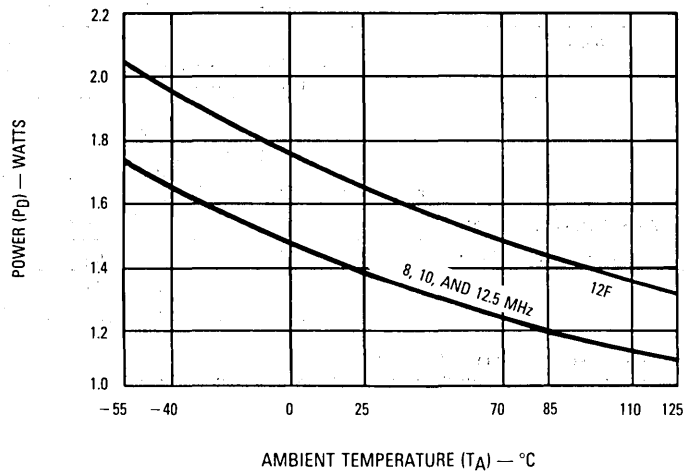


Figure 5. MC68000 Power Dissipation (P_D) vs Ambient Temperature (T_A)

Table 6. Power Dissipation and Junction Temperature vs Temperature ($\theta_{JC} = \theta_{JA}$)

Package	T_A Range	θ_{JC} (°C/W)	P_D (W) @ T_A Min.	T_J (°C) @ T_A Min.	P_D (W) @ T_A Max.	T_J (°C) @ T_A Max.
L/LC	0°C to 70°C	15	1.5	23	1.2	88
	−40°C to 85°C	15	1.7	−14	1.2	103
	0°C to 85°C	15	1.5	23	1.2	103
P	0°C to 70°C	15	1.5	23	1.2	88
R/RC	0°C to 70°C	15	1.5	23	1.2	88
	−40°C to 85°C	15	1.7	−14	1.2	103
	0°C to 85°C	15	1.5	23	1.2	103
FN	0°C to 70°C	25	1.5	38	1.2	101

NOTE: Table does not include values for the MC68000 12F.

3

Table 7. Power Dissipation and Junction Temperature vs Temperature ($\theta_{JC} \triangleq \theta_{JC}$)

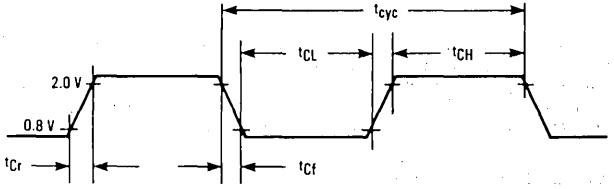
Package	T_A Range	θ_{JA} (°C/W)	P_D (W) @ T_A Min.	T_J (°C) @ T_A Min.	P_D (W) @ T_A Max.	T_J (°C) @ T_A Max.
L/LC	0°C to 70°C	30	1.5	23	1.2	88
	−40°C to 85°C	30	1.7	−14	1.2	103
	0°C to 85°C	30	1.5	23	1.2	103
P	0°C to 70°C	30	1.5	23	1.2	88
R/RC	0°C to 70°C	33	1.5	23	1.2	88
	−40°C to 85°C	33	1.7	−14	1.2	103
	0°C to 85°C	33	1.5	23	1.2	103
FN	0°C to 70°C	40	1.5	38	1.2	101

NOTE: Table does not include values for the MC68000 12F.

AC ELECTRICAL SPECIFICATIONS — CLOCK TIMING (see Figure 6)

Num.	Characteristic	Symbol	8 MHz*		10 MHz*		12.5 MHz*		16.67 MHz '12F'		Unit
			Min	Max	Min	Max	Min	Max	Min	Max	
	Frequency of Operation	f	4.0	8.0	4.0	10.0	4.0	12.5	8.0	16.7	MHz
1	Cycle Time	t_{cyc}	125	250	100	250	80	250	60	125	ns
2,3	Clock Pulse Width (Measured from 1.5 V to 1.5 V for 12F)	t_{CL}	55	125	45	125	35	125	27	62.5	ns
		t_{CH}	55	125	45	125	35	125	27	62.5	
4,5	Clock Rise and Fall Times	t_{Cr}	—	10	—	10	—	5	—	5	ns
		t_{Cf}	—	10	—	10	—	5	—	5	

*These specifications represent an improvement over previously published specifications for the 8-, 10-, and 12.5-MHz MC68000 and are valid only for product bearing date codes of 8827 and later.



NOTE: Timing measurements are referenced to and from a low voltage of 0.8 volt and high a voltage of 2.0 volts, unless otherwise noted. The voltage swing through this range should start outside and pass through the range such that the rise or fall will be linear between 0.8 volt and 2.0 volts.

Figure 6. Clock Input Timing

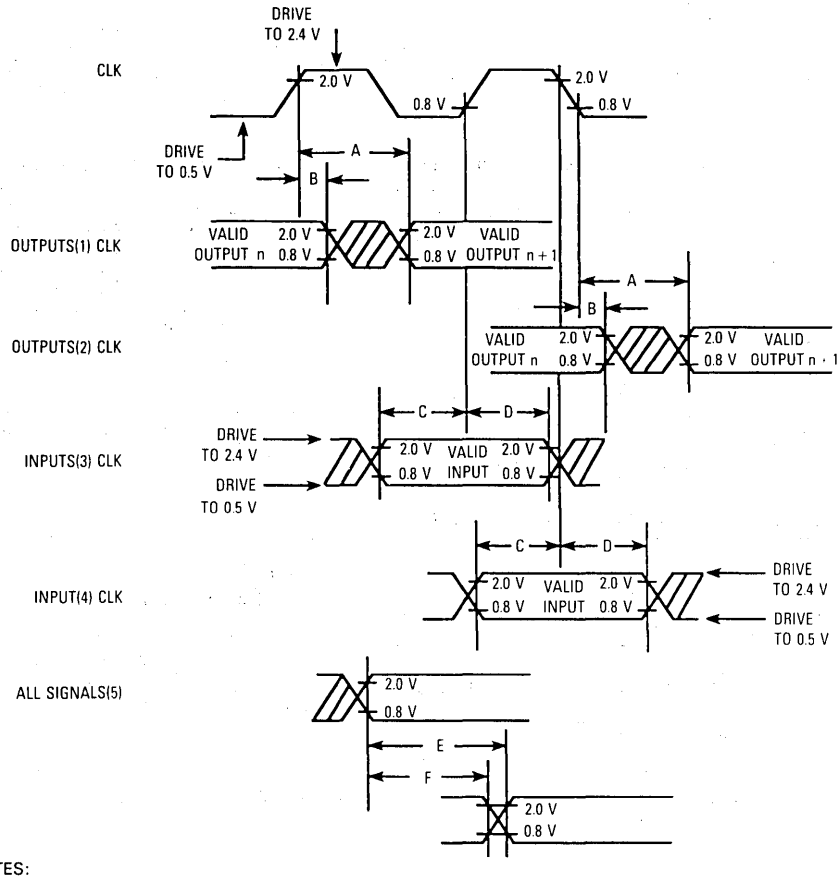
3

AC ELECTRICAL SPECIFICATION DEFINITIONS

The AC specifications presented consist of output delays, input setup and hold times, and signal skew times. All signals are specified relative to an appropriate edge of the clock and possibly to one or more other signals.

The measurement of the AC specifications is defined by the waveforms shown in Figure 7. In order to test the parameters guaranteed by Motorola, inputs must be driven to the voltage levels specified in the figure. Outputs are specified with minimum and/or maximum limits, as appropriate, and are measured as shown in Figure 7. Inputs are specified with minimum setup and hold times, and are measured as shown. Finally, the measurement for signal-to-signal specifications are also shown.

Note that the testing levels used to verify conformance to the AC specifications does not affect the guaranteed DC operation of the device as specified in the DC electrical characteristics.



NOTES:

1. This output timing is applicable to all parameters specified relative to the rising edge of the clock.
2. This output timing is applicable to all parameters specified relative to the falling edge of the clock.
3. This input timing is applicable to all parameters specified relative to the rising edge of the clock.
4. This input timing is applicable to all parameters specified relative to the falling edge of the clock.
5. This timing is applicable to all parameters specified relative to the assertion/negation of another signal.

LEGEND:

A. Maximum output delay specification.
B. Minimum output hold time.
C. Minimum input setup time specification.
D. Minimum input hold time specification.
E. Signal valid to signal valid specification (maximum or minimum).
F. Signal valid to signal invalid specification (maximum or minimum).

Figure 7. Drive Levels and Test Points for AC Specifications

AC ELECTRICAL SPECIFICATIONS — READ AND WRITE CYCLES (V_{CC} = 5.0 Vdc ± 5%; GND = 0 V; $T_A = T_L$ to T_H; see Figures 11 and 12)

Num.	Characteristic	Symbol	8 MHz*		10 MHz*		12.5 MHz*		16.67 MHz '12F'		Unit
			Min	Max	Min	Max	Min	Max	Min	Max	
6	Clock Low to Address Valid	t_{CLAV}	—	62	—	50	—	50	—	50	ns
6A	Clock High to FC Valid	t_{CHFCV}	—	62	—	50	—	45	—	45	ns
7	Clock High to Address, Data Bus High Impedance (Maximum)	t_{CHADZ}	—	80	—	70	—	60	—	50	ns
8	Clock High to Address, FC Invalid (Minimum)	t_{CHAFI}	0	—	0	—	0	—	0	—	ns
9[1]	Clock High to $\overline{AS}$, $\overline{DS}$ Asserted	t_{CHSL}	3	60	3	50	3	40	3	40	ns
11[2]	Address Valid to $\overline{AS}$, $\overline{DS}$ Asserted (Read)/$\overline{AS}$ Asserted (Write)	t_{AVSL}	30	—	20	—	15	—	15	—	ns
11A[2]	FC Valid to $\overline{AS}$, $\overline{DS}$ Asserted (Read)/$\overline{AS}$ Asserted (Write)	t_{FCVSL}	90	—	70	—	60	—	30	—	ns
12[1]	Clock Low to $\overline{AS}$, $\overline{DS}$ Negated	t_{CLSH}	—	62	—	50	—	40	—	40	ns
13[2]	$\overline{AS}$, $\overline{DS}$ Negated to Address, FC Invalid	t_{SHAFI}	40	—	30	—	20	—	10	—	ns
14[2]	$\overline{AS}$ (and $\overline{DS}$ Read) Width Asserted	t_{SL}	270	—	195	—	160	—	120	—	ns
14A	$\overline{DS}$ Width Asserted (Write)	t_{DSL}	140	—	95	—	80	—	60	—	ns
15[2]	$\overline{AS}$, $\overline{DS}$ Width Negated	t_{SH}	150	—	105	—	65	—	60	—	ns
16	Clock High to Control Bus High Impedance	t_{CHCZ}	—	80	—	70	—	60	—	50	ns
17[2]	$\overline{AS}$, $\overline{DS}$ Negated to R/$\overline{W}$ Invalid	t_{SHRH}	40	—	30	—	20	—	10	—	ns
18[1]	Clock High to R/$\overline{W}$ High (Read)	t_{CHRH}	0	55	0	45	0	40	0	40	ns
20[1]	Clock High to R/$\overline{W}$ Low (Write)	t_{CHRL}	0	55	0	45	0	40	0	40	ns
20A[2,6]	$\overline{AS}$ Asserted to R/$\overline{W}$ Valid (Write)	t_{ASRV}	—	10	—	10	—	10	—	10	ns
21[2]	Address Valid to R/$\overline{W}$ Low (Write)	t_{AVRL}	20	—	0	—	0	—	0	—	ns
21A[2]	FC Valid to R/$\overline{W}$ Low (Write)	t_{FCVRL}	60	—	50	—	30	—	20	—	ns
22[2]	R/$\overline{W}$ Low to $\overline{DS}$ Asserted (Write)	t_{RLSL}	80	—	50	—	30	—	20	—	ns
23	Clock Low to Data-Out Valid (Write)	t_{CLDO}	—	62	—	50	—	50	—	50	ns
25[2]	$\overline{AS}$, $\overline{DS}$ Negated to Data-Out Invalid (Write)	t_{SHDOI}	40	—	30	—	20	—	15	—	ns
26[2]	Data-Out Valid to $\overline{DS}$ Asserted (Write)	t_{DOSL}	40	—	30	—	20	—	15	—	ns
27[5]	Data-In Valid to Clock Low (Setup Time on Read)	t_{DICL}	10	—	10	—	10	—	7	—	ns
28[2]	$\overline{AS}$, $\overline{DS}$ Negated to $\overline{DTACK}$ Negated (Asynchronous Hold)	t_{SHDAH}	0	240	0	190	0	150	0	110	ns
29	$\overline{AS}$, $\overline{DS}$ Negated to Data-In Invalid (Hold Time on Read)	t_{SHDII}	0	—	0	—	0	—	0	—	ns
29A	$\overline{AS}$, $\overline{DS}$ Negated to Data-In High Impedance	t_{SHDZ}	—	187	—	150	—	120	—	90	ns
30	$\overline{AS}$, $\overline{DS}$ Negated to $\overline{BERR}$ Negated	t_{SHBEH}	0	—	0	—	0	—	0	—	ns
31[2,5]	$\overline{DTACK}$ Asserted to Data-In Valid (Setup Time)	t_{DALDI}	—	90	—	65	—	50	—	40	ns
32	$\overline{HALT}$ and $\overline{RESET}$ Input Transition Time	$t_{RHr,f}$	0	200	0	200	0	200	0	150	ns
33	Clock High to $\overline{BG}$ Asserted	t_{CHGL}	—	62	—	50	—	40	—	40	ns
34	Clock High to $\overline{BG}$ Negated	t_{CHGH}	—	62	—	50	—	40	—	40	ns
35	$\overline{BR}$ Asserted to $\overline{BG}$ Asserted	t_{BRLGL}	1.5	3.5	1.5	3.5	1.5	3.5	1.5	3.5	Clks
36[7]	$\overline{BR}$ Negated to $\overline{BG}$ Negated	t_{BRHGH}	1.5	3.5	1.5	3.5	1.5	3.5	1.5	3.5	Clks
37	$\overline{BGACK}$ Asserted to $\overline{BG}$ Negated	t_{GALGH}	1.5	3.5	1.5	3.5	1.5	3.5	1.5	3.5	Clks
37A[8]	$\overline{BGACK}$ Asserted to $\overline{BR}$ Negated	t_{GALBRH}	20	1.5 Clks	20	1.5 Clks	20	1.5 Clks	10	1.5 Clks	ns
38	$\overline{BG}$ Asserted to Control, Address, Data Bus High Impedance ($\overline{AS}$ Negated)	t_{GLZ}	—	80	—	70	—	60	—	50	ns

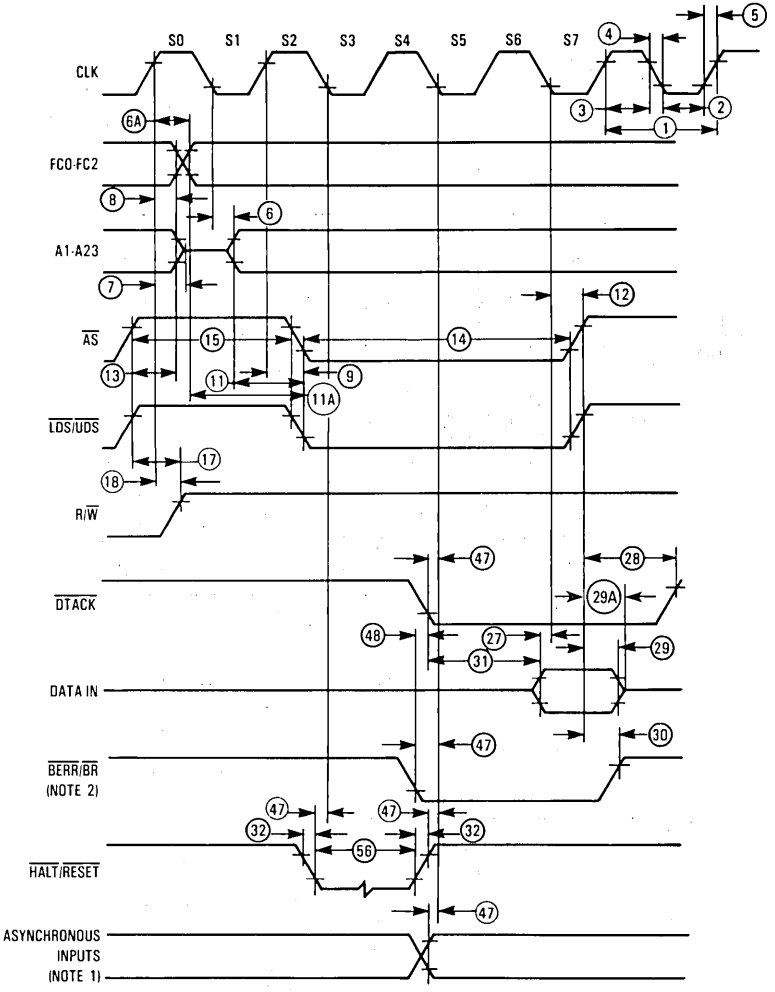
AC ELECTRICAL SPECIFICATIONS — READ AND WRITE CYCLES (Continued)

Num.	Characteristic	Symbol	8 MHz*		10 MHz*		12.5 MHz*		16.67 MHz '12F'		Unit
			Min	Max	Min	Max	Min	Max	Min	Max	
39	$\overline{BG}$ Width Negated	t_{GH}	1.5	—	1.5	—	1.5	—	1.5	—	Clks
40	Clock Low to $\overline{VMA}$ Asserted	t_{CLVML}	—	70	—	70	—	70	—	50	ns
41	Clock Low to E Transition	t_{CLET}	—	55	—	45	—	35	—	35	ns
42	E Output Rise and Fall Time	$t_{Er,f}$	—	15	—	15	—	15	—	15	ns
43	$\overline{VMA}$ Asserted to E High	t_{VMLEH}	200	—	150	—	90	—	80	—	ns
44	$\overline{AS}$, $\overline{DS}$ Negated to $\overline{VPA}$ Negated	t_{SHVPH}	0	120	0	90	0	70	0	50	ns
45	E Low to Control, Address Bus Invalid (Address Hold Time)	t_{ELCAI}	30	—	10	—	10	—	10	—	ns
46	$\overline{BGACK}$ Width Low	t_{GAL}	1.5	—	1.5	—	1.5	—	1.5	—	Clks
47[5]	Asynchronous Input Setup Time	t_{ASI}	10	—	10	—	10	—	10	—	ns
48[2,3]	$\overline{BERR}$ Asserted to $\overline{DTACK}$ Asserted	t_{BELDAL}	20	—	20	—	20	—	10	—	ns
49[9]	$\overline{AS}$, $\overline{DS}$, Negated to E Low	t_{SHEL}	−70	70	−55	55	−45	45	−35	35	ns
50	E Width High	t_{EH}	450	—	350	—	280	—	220	—	ns
51	E Width Low	t_{EL}	700	—	550	—	440	—	340	—	ns
53	Data-Out Hold from Clock High	t_{CHDOI}	0	—	0	—	0	—	0	—	ns
54	E Low to Data-Out Invalid	t_{ELDOI}	30	—	20	—	15	—	10	—	ns
55	R/$\overline{W}$ Asserted to Data Bus Impedance Change	t_{RLDBD}	30	—	20	—	10	—	0	—	ns
56[4]	$\overline{HALT}$/$\overline{RESET}$ Pulse Width	t_{HRPW}	10	—	10	—	10	—	10	—	Clks
57	$\overline{BGACK}$ Negated to $\overline{AS}$, $\overline{DS}$, R/$\overline{W}$ Driven	t_{GASD}	1.5	—	1.5	—	1.5	—	1.5	—	Clks
57A	$\overline{BGACK}$ Negated to FC, $\overline{VMA}$ Driven	t_{GAFD}	1	—	1	—	1	—	1	—	Clks
58[7]	$\overline{BR}$ Negated to $\overline{AS}$, $\overline{DS}$, R/$\overline{W}$ Driven	t_{RHSD}	1.5	—	1.5	—	1.5	—	1.5	—	Clks
58A[7]	$\overline{BR}$ Negated to FC, $\overline{VMA}$ Driven	t_{RHFD}	1	—	1	—	1	—	1	—	Clks

*These specifications represent an improvement over previously published specifications for the 8-, 10-, and 12.5-MHz MC68000 and are valid only for product bearing date codes of 8827 and later.

NOTES:
1. For a loading capacitance of less than or equal to 50 picofarads, subtract 5 nanoseconds from the value given in the maximum columns.
2. Actual value depends on clock period.
3. If #47 is satisfied for both $\overline{DTACK}$ and $\overline{BERR}$, #48 may be ignored. In the absence of $\overline{DTACK}$, $\overline{BERR}$ is an asynchronous input using the asynchronous input setup time (#47).
4. For power-up, the MC68000 must be held in the reset state for 100 milliseconds to allow stabilization of on-chip circuitry. After the system is powered up, #56 refers to the minimum pulse width required to reset the processor.
5. If the asynchronous input setup time (#47) requirement is satisfied for $\overline{DTACK}$, the $\overline{DTACK}$-asserted to data setup time (#31) requirement can be ignored. The data must only satisfy the data-in to clock low setup time (#27) for the following clock cycle.
6. When $\overline{AS}$ and R/$\overline{W}$ are equally loaded (±20%), subtract 5 nanoseconds from the values given in these columns.
7. The processor will negate $\overline{BG}$ and begin driving the bus again if external arbitration logic negates $\overline{BR}$ before asserting $\overline{BGACK}$.
8. The minimum value must be met to guarantee proper operation. If the maximum value is exceeded, $\overline{BG}$ may be re-asserted.
9. The falling edge of S6 triggers both the negation of the strobes ($\overline{AS}$ and x$\overline{DS}$) and the falling edge of E. Either of these events can occur first, depending upon the loading on each signal. Specification #49 indicates the absolute maximum skew that will occur between the rising edge of the strobes and the falling edge of the E clock.

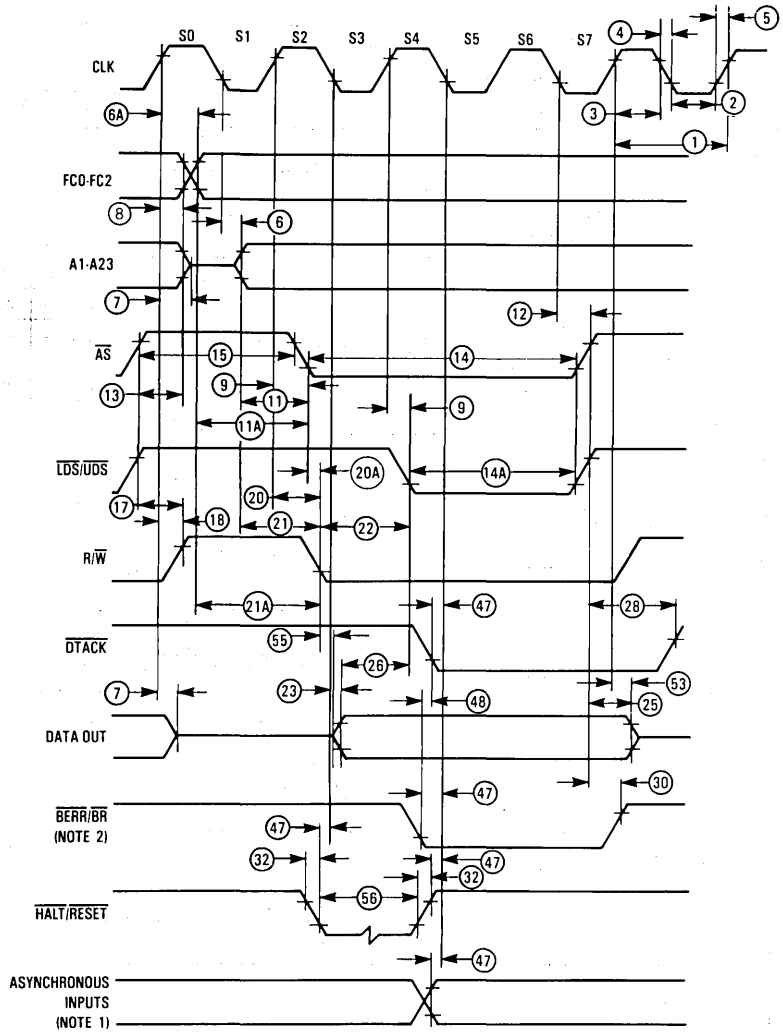
These waveforms should only be referenced in regard to the edge-to-edge measurement of the timing specifications. They are not intended as a functional description of the input and output signals. Refer to other functional descriptions and their related diagrams for device operation.



NOTES:
1. Setup time for the asynchronous inputs IPL0-IPL2 and VPA (#47) guarantees their recognition at the next falling edge of the clock.
2. BR need fall at this time only in order to insure being recognized at the end of the bus cycle.
3. Timing measurements are referenced to and from a low voltage of 0.8 volt and a high voltage of 2.0 volts, unless otherwise noted. The voltage swing through this range should start outside and pass through the range such that the rise or fall is linear between 0.8 volt and 2.0 volts.

Figure 8. MC68000 Read-Cycle Timing Diagram

These waveforms should only be referenced in regard to the edge-to-edge measurement of the timing specifications. They are not intended as a functional description of the input and output signals. Refer to other functional descriptions and their related diagrams for device operation.



NOTES:
1. Timing measurements are referenced to and from a low voltage of 0.8 volt and a high voltage of 2.0 volts, unless otherwise noted. The voltage swing through this range should start outside and pass through the range such that the rise or fall is linear between 0.8 volt and 2.0 volts.
2. Because of loading variations, R/$\overline{W}$ may be valid after $\overline{AS}$ even though both are initiated by the rising edge of S2 (specification #20A).

Figure 9. MC68000 Write-Cycle Timing Diagram

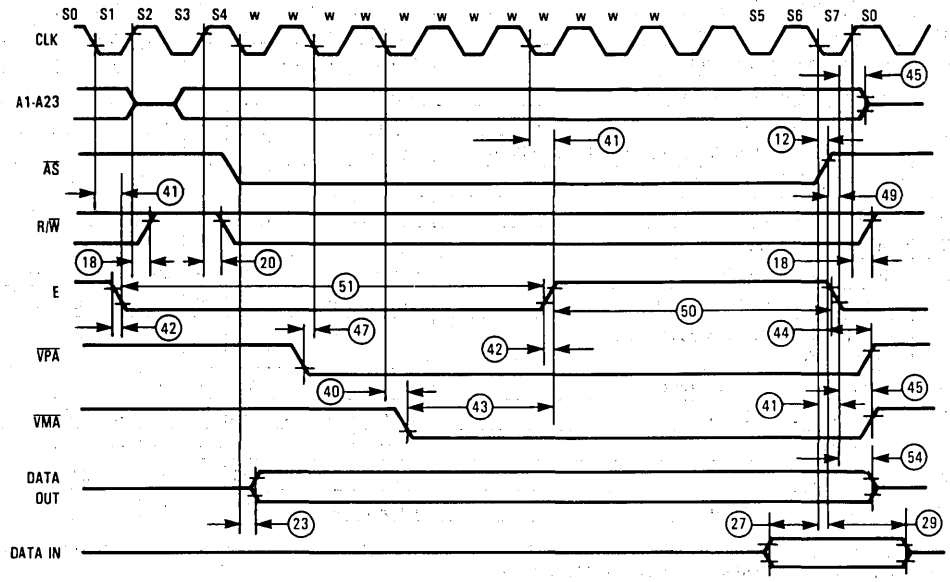
AC ELECTRICAL SPECIFICATIONS — MC68000 TO M6800 PERIPHERAL CYCLES (V_{CC} = 5.0 Vdc ± 5%; GND = 0 Vdc; $T_A = T_L$ to T_H; see Figures 10 and 11)

Num.	Characteristic	Symbol	8 MHz*		10 MHz*		12.5 MHz*		16.67 MHz '12F'		Unit
			Min	Max	Min	Max	Min	Max	Min	Max	
12[1]	Clock Low to $\overline{AS}$, $\overline{DS}$ Negated	t_{CLSH}	—	62	—	50	—	40	—	40	ns
18[1]	Clock High to R/$\overline{W}$ High (Read)	t_{CHRH}	0	55	0	45	0	40	0	40	ns
20[1]	Clock High to R/$\overline{W}$ Low (Write)	t_{CHRL}	0	55	0	45	0	40	0	40	ns
23	Clock Low to Data-Out Valid (Write)	t_{CLDO}	—	62	—	50	—	50	—	50	ns
27	Data-In Valid to Clock Low (Setup Time of Read)	t_{DICL}	10	—	10	—	10	—	7	—	ns
29	$\overline{AS}$, $\overline{DS}$ Negated to Data-In Invalid (Hold Time on Read)	t_{SHDII}	0	—	0	—	0	—	0	—	ns
40	Clock Low to $\overline{VMA}$ Asserted	t_{CLVML}	—	70	—	70	—	70	—	50	ns
41	Clock Low to E Transition	t_{CLET}	—	55	—	45	—	35	—	35	ns
42	E Output Rise and Fall Time	$t_{Er,f}$	—	15	—	15	—	15	—	15	ns
43	$\overline{VMA}$ Asserted to E High	t_{VMLEH}	200	—	150	—	90	—	80	—	ns
44	$\overline{AS}$, $\overline{DS}$ Negated to $\overline{VPA}$ Negated	t_{SHVPH}	0	120	0	90	0	70	0	50	ns
45	E Low to Control, Address Bus Invalid (Address Hold Time)	t_{ELCAI}	30	—	10	—	10	—	10	—	ns
47	Asynchronous Input Setup Time	t_{ASI}	10	—	10	—	10	—	10	—	ns
49[2]	$\overline{AS}$, $\overline{DS}$, Negated to E Low	t_{SHEL}	−70	70	−55	55	−45	45	−35	35	ns
50	E Width High	t_{EH}	450	—	350	—	280	—	220	—	ns
51	E Width Low	t_{EL}	700	—	550	—	440	—	340	—	ns
54	E Low to Data-Out Invalid	t_{ELDOI}	30	—	20	—	15	—	10	—	ns

*These specifications represent an improvement over previously published specifications for the 8-, 10-, and 12.5-MHz MC68000 and are valid only for product bearing date codes of 8827 and later.

NOTES:
1. For a loading capacitance of less than or equal to 50 picofarads, subtract 5 nanoseconds from the value given in the maximum columns.
2. The falling edge of S6 trigger both the negation of the strobes ($\overline{AS}$ and $\overline{xDS}$) and the falling edge of E. Either of these events can occur first, depending upon the loading on each signal. Specification #49 indicates the absolute maximum skew that will occur between the rising edge of the strobes and the falling edge of the E clock.

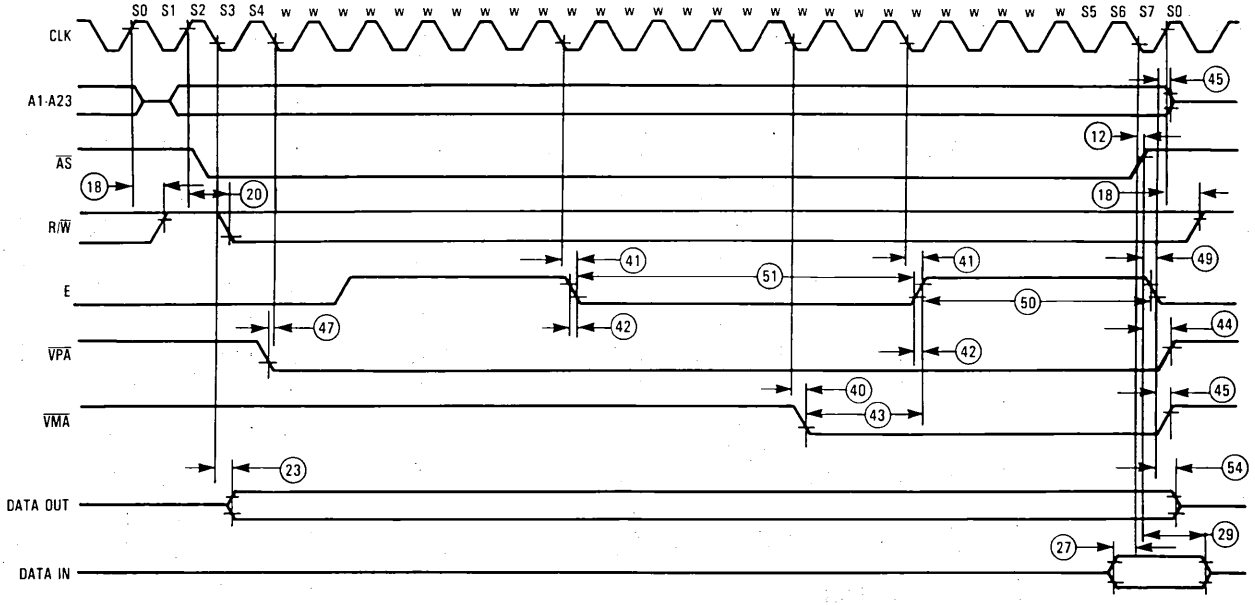
These waveforms should only be referenced in regard to the edge-to-edge measurement of the timing specifications. They are not intended as a functional description of the input and output signals. Refer to other functional descriptions and their related diagrams for device operation.



NOTE: This timing diagram is included for those who wish to design their own circuit to generate $\overline{VMA}$. It shows the best case possibly attainable.

Figure 10. MC68000 to M6800 Peripheral Timing Diagram (Best Case)

These waveforms should only be referenced in regard to the edge-to-edge measurement of the timing specifications. They are not intended as a functional description of the input and output signals. Refer to other functional descriptions and their related diagrams for device operation.



NOTE: This timing diagram is included for those who wish to design their own circuit to generate $\overline{VMA}$. It shows the worst case possibly attainable

Figure 11. MC68000 to M6800 Peripheral Timing Diagram (Worst Case)

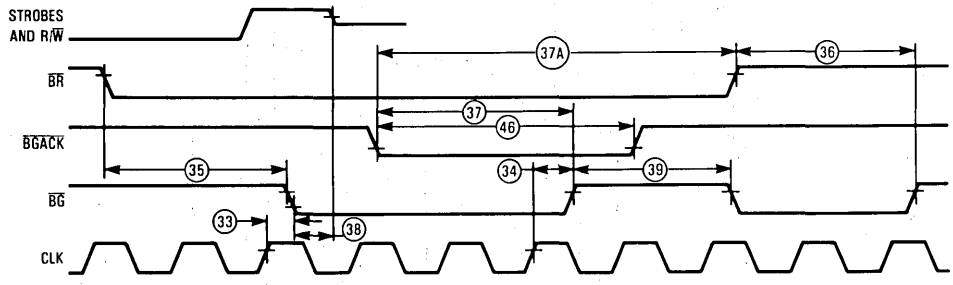
AC ELECTRICAL SPECIFICATIONS — BUS ARBITRATION (V_{CC} = 5.0 Vdc ± 5%; GND = 0 Vdc; $T_A = T_L$ to T_H; see Figure 12)

Num.	Characteristic	Symbol	8 MHz*		10 MHz*		12.5 MHz*		16.67 MHz '12F'		Unit
			Min	Max	Min	Max	Min	Max	Min	Max	
7	Clock High to Address, Data Bus High Impedance (Maximum)	t_{CHADZ}	—	80	—	70	—	60	—	50	ns
16	Clock High to Control Bus High Impedance	t_{CHCZ}	—	80	—	70	—	60	—	50	ns
33	Clock High to $\overline{BG}$ Asserted	t_{CHGL}	—	62	—	50	—	40	—	40	ns
34	Clock High to $\overline{BG}$ Negated	t_{CHGH}	—	62	—	50	—	40	—	40	ns
35	$\overline{BR}$ Asserted to $\overline{BG}$ Asserted	t_{BRLGL}	1.5	3.5	1.5	3.5	1.5	3.5	1.5	3.5	Clks
36[1]	$\overline{BR}$ Negated to $\overline{BG}$ Negated	t_{BRHGH}	1.5	3.5	1.5	3.5	1.5	3.5	1.5	3.5	Clks
37	$\overline{BGACK}$ Asserted to $\overline{BG}$ Negated	t_{GALGH}	1.5	3.5	1.5	3.5	1.5	3.5	1.5	3.5	Clks
37A[2]	$\overline{BGACK}$ Asserted to $\overline{BR}$ Negated	t_{GALBRH}	20	1.5 Clks	20	1.5 Clks	20	1.5 Clks	10	1.5 Clks	ns
38	$\overline{BG}$ Asserted to Control, Address, Data Bus High Impedance ($\overline{AS}$ Negated)	t_{GLZ}	—	80	—	70	—	60	—	50	ns
39	$\overline{BG}$ Width Negated	t_{GH}	1.5	—	1.5	—	1.5	—	1.5	—	Clks
46	$\overline{BGACK}$ Width Low	t_{GAL}	1.5	—	1.5	—	1.5	—	1.5	—	Clks
47	Asynchronous Input Setup Time	t_{ASI}	10	—	10	—	10	—	10	—	ns
57	$\overline{BGACK}$ Negated to $\overline{AS}$, $\overline{DS}$, R/$\overline{W}$ Driven	t_{GASD}	1.5	—	1.5	—	1.5	—	1.5	—	Clks
57A	$\overline{BGACK}$ Negated to FC, $\overline{VMA}$ Driven	t_{GAFD}	1	—	1	—	1	—	1	—	Clks
58[1]	$\overline{BR}$ Negated to $\overline{AS}$, $\overline{DS}$, R/$\overline{W}$ Driven	t_{RHSD}	1.5	—	1.5	—	1.5	—	1.5	—	Clks
58A[1]	$\overline{BR}$ Negated to FC, $\overline{VMA}$ Driven	t_{RHFD}	1	—	1	—	1	—	1	—	Clks

*These specifications represent an improvement over previously published specifications for the 8-, 10-, and 12.5-MHz MC68000 and are valid only for product bearing date codes of 8827 and later.

NOTES:
1. The processor will negate $\overline{BG}$ and begin driving the bus again if external arbitration logic negates $\overline{BR}$ before asserting $\overline{BGACK}$.
2. The minimum value must be met to guarantee proper operation. If the maximum value is exceeded, $\overline{BG}$ may be re-asserted.

These waveforms should only be referenced in regard to the edge-to-edge measurement of the timing specifications. They are not intended as a functional description of the input and output signals. Refer to other functional descriptions and their related diagrams for device operation.

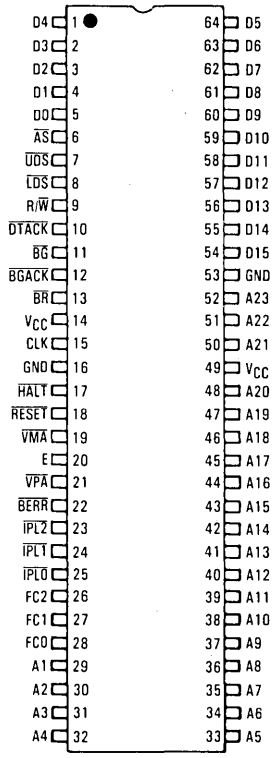


NOTE: Setup time to the clock (#47) for the asynchronous inputs $\overline{BERR}$, $\overline{BGACK}$, $\overline{BR}$, $\overline{DTACK}$, $\overline{IPL0}$-$\overline{IPL2}$, and $\overline{VPA}$ guarantees their recognition at the next falling edge of the clock.

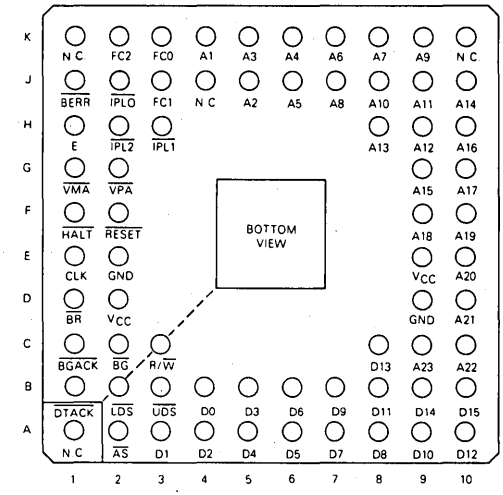
Figure 12. MC68000 Bus Arbitration Timing Diagram

PIN ASSIGNMENTS

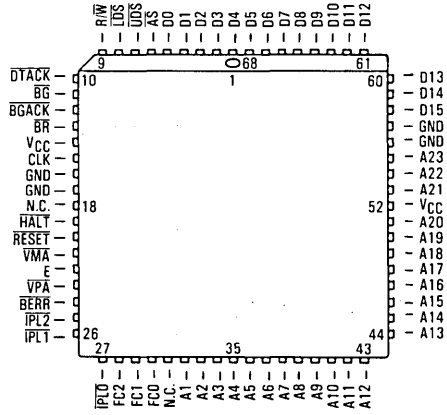
64-PIN DUAL-IN-LINE PACKAGE



68-TERMINAL PIN GRID ARRAY



68-LEAD QUAD PACK



3

MC68HC000

Technical Summary

Low Power HCMOS 16-/32-Bit Microprocessor

3

This document contains both a summary of the MC68HC000 as well as a detailed set of parametrics. The purpose is twofold — to provide an introduction to the MC68HC000 and support for the sophisticated user. For detailed information on the MC68HC000, refer to the *MC68000 16-Bit Micorprocessor User's Manual.*

The primary benefit of the MC68HC000 is its reduced power consumption. The device dissipates an order of magnitude less power than the HMOS MC68000.

The MC68HC000 is an implementation of the M68000 16/32 microprocessor architecture. The MC68HC000 has a 16-bit data bus implementation of the M68000 and is upward code compatible to the MC68010 virtual extension and the MC68020 32-bit implementation of the architecture. Any user-mode programs written using the MC68HC000 instruction set will run unchanged on the MC68000, MC68008, MC68010, and MC68020. This is possible because the user programming model is identical for all five processors and the instruction sets are proper sub-sets of the complete architecture. Resources available to the MC68HC000 user consist of the following:

- 17 32-Bit Data and Address Registers
- 16 Megabyte Direct Addressing Range
- 56 Powerful Instruction Types
- Operations on Five Main Data Types
- Memory Mapped I/O
- 14 Addressing Modes

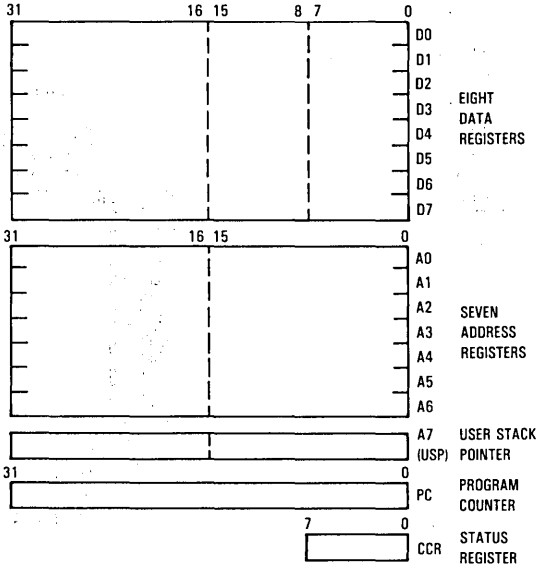


Figure 1. User Programming Model

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

INTRODUCTION

As shown in the user programming model (Figure 1), the MC68HC000 offers 16 32-bit registers and a 32-bit program counter. The first eight registers (D0-D7) are used as data registers for byte (8-bit), word (16-bit), and long word (32-bit) operations. The second set of seven registers (A0-A6) and the user stack pointer (USP) may be used as software stack pointers and base address registers. In addition, the registers may be used for word and long word operations. All of the 16 registers may be used as index registers.

In supervisor mode, the upper byte of the status register and the supervisor stack pointer (SSP) are also available to the programmer. These registers are shown in Figure 2.

The status register (Figure 3) contains the interrupt mask (eight levels available) as well as the condition codes: extend (X), negative (N), zero (Z), overflow (V), and carry (C). Additional status bits indicate that the processor is in a trace (T) mode and in a supervisor (S) or user state.

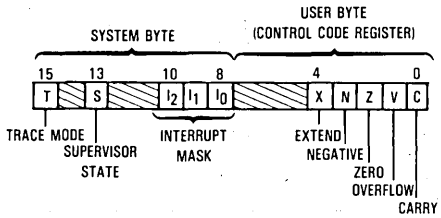


Figure 3. Status Register

DATA TYPES AND ADDRESSING MODES

Five basic data types are supported. These data types are:

- Bits
- BCD Digits (4 Bits)
- Bytes (8 Bits)
- Words (16 Bits)
- Long Words (32 Bits)

In addition, operations on other data types such as memory addresses, status word data, etc. are provided in the instruction set.

The 14 addressing modes, shown in Table 1, include six basic types:

- Register Direct
- Register Indirect
- Absolute
- Program Counter Relative
- Immediate
- Implied

Included in the register indirect addressing modes is the capability to do postincrementing, predecrementing, offsetting, and indexing. The program counter relative mode can also be modified via indexing and offsetting.

Table 1. Addressing Modes

Addressing Modes	Syntax
Register Direct Addressing	
Data Register Direct	Dn
Address Register Direct	An
Absolute Data Addressing	
Absolute Short	xxx.W
Absolute Long	xxx.L
Program Counter Relative Addressing	
Relative with Offset	d_{16}(PC)
Relative with Index Offset	d_8(PC,Xn)
Register Indirect Addressing	
Register Indirect	(An)
Postincrement Register Indirect	(An)+
Predecrement Register Indirect	−(An)
Register Indirect with Offset	d_{16}(An)
Indexed Register Indirect with Offset	d_8(An,Xn)
Immediate Data Addressing	
Immediate	#xxx
Quick Immediate	#1-#8
Implied Addressing	
Implied Register	SR/USP/SP/PC

NOTES:
Dn = Data Register
An = Address Register
Xn = Address of Data Register used as Index Register
SR = Status Register
PC = Program Counter
SP = Stack Pointer
USP = User Stack Pointer
( ) = Effective Address
d_8 = 8-Bit Offset (Displacement)
d_{16} = 16-Bit Offset (Displacement)
#xxx = Immediate Data

INSTRUCTION SET OVERVIEW

The MC68HC000 instruction set is shown in Table 2. Some additional instructions are variations, or sub-sets, of these and they appear in Table 3. Special emphasis has been given to the instruction set's support of structured high-level langugages to facilitate ease of programming. Each instruction, with few exceptions, operates on bytes, words, and long words and most instructions can use any of the 14 addressing modes. Combining instruction types, data types, and addressing modes, over 1000 useful instructions are provided. These instructions include signed and unsigned, multiply and divide, "quick" arithmetic operations, BCD arithmetic, and expanded operations (through traps).

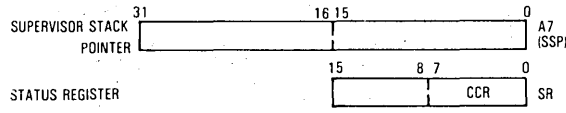


Figure 2. Supervisor Programming Model Supplement

Table 2. Instruction Set Summary

Mnemonic	Description
ABCD	Add Decimal With Extend
ADD	Add
AND	Logical AND
ASL	Arithmetic Shift Left
ASR	Arithmetic Shift Right
Bcc	Branch Conditionally
BCHG	Bit Test and Change
BCLR	Bit Test and Clear
BRA	Branch Always
BSET	Bit Test and Set
BSR	Branch to Subroutine
BTST	Bit Test
CHK	Check Register Against Bounds
CLR	Clear Operand
CMP	Compare
DBcc	Test Condition, Decrement and Branch
DIVS	Signed Divide
DIVU	Unsigned Divide
EOR	Exclusive OR
EXG	Exchange Registers
EXT	Sign Extend
JMP	Jump
JSR	Jump to Subroutine
LEA	Lead Effective Address
LINK	Link Stack
LSL	Logical Shift Left
LSR	Logical Shift Right
MOVE	Move
MULS	Signed Multiply
MULU	Unsigned Multiply
NBCD	Negate Decimal with Extend
NEG	Negate
NOP	No Operation
NOT	One's Complement
OR	Logical OR
PEA	Push Effective Address
RESET	Reset External Devices
ROL	Rotate Left without Extend
ROR	Rotate Right without Extend
ROXL	Rotate Left with Extend
ROXR	Rotate Right with Extend
RTE	Return from Exception
RTR	Return and Restore
RTS	Return from Subroutine
SBCD	Subtract Decimal with Extend
Scc	Set Conditional
STOP	Stop
SUB	Subtract
SWAP	Swap Data Register Halves
TAS	Test and Set Operand
TRAP	Trap
TRAPV	Trap on Overflow
TST	Test
UNLK	Unlink

Table 3. Variations of Instruction Types

Instruction Type	Variation	Description
ADD	ADD	Add
	ADDA	Add Address
	ADDQ	Add Quick
	ADDI	Add Immediate
	ADDX	Add with Extend
AND	AND	Logical AND
	ANDI	And Immediate
	ANDI to CCR	And Immediate to Condition Codes
	ANDI to SR	And Immediate to Status Register
CMP	CMP	Compare
	CMPA	Compare Address
	CMPM	Compare Memory
	CMPI	Compare Immediate
EOR	EOR	Exclusive OR
	EORI	Exclusive OR Immediate
	EORI to CCR	Exclusive OR Immediate to Condition Codes
	EORI to SR	Exclusive OR Immediate to Status Register
MOVE	MOVE	Move
	MOVEA	Move Address
	MOVEM	Move Multiple Registers
	MOVEP	Move Peripheral Data
	MOVEQ	Move Quick
	MOVE from SR	Move from Status Register
	MOVE to SR	Move to Status Register
	MOVE to CCR	Move to Condition Codes
	MOVE USP	Move User Stack Pointer
NEG	NEG	Negate
	NEGX	Negate with Extend
OR	OR	Logical OR
	ORI	OR Immediate
	ORI to CCR	OR Immediate to Condition Codes
	ORI to SR	OR Immediate to Status Register
SUB	SUB	Subtract
	SUBA	Subtract Address
	SUBI	Subtract Immediate
	SUBQ	Subtract Quick
	SUBX	Subtract with Extend

SIGNAL DESCRIPTION

The input and output signals are illustrated functionally in Figure 4 and are described in the following paragraphs.

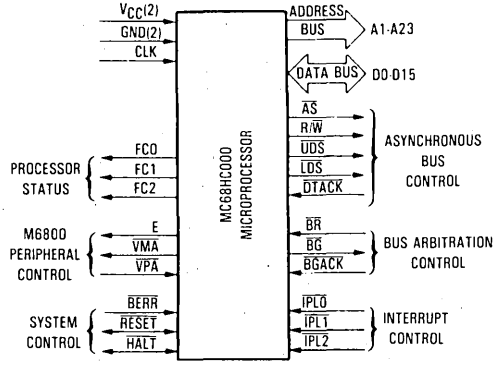


Figure 4. Input and Output Signals

ADDRESS BUS (A1 THROUGH A23)

This 24-bit, unidirectional, three-state bus is capable of addressing 16 megabytes of data. It provides the address for bus operation during all cycles except interrupt cycles. During interrupt cycles, address lines A1, A2, and A3 provide information about what level interrupt is being serviced while address lines A4 through A23 are set to a logic high.

DATA BUS (D0 THROUGH D15)

This 16-bit, bidirectional, three-state bus is the general purpose data path. It can transfer and accept data in either word or byte length. During an interrupt acknowledge cycle, the external device supplies the vector number on data lines D0-D7.

ASYNCHRONOUS BUS CONTROL

Asynchronous data transfers are handled using the following control signals: address strobe, read/write, upper and lower data strobes, and data transfer acknowledge. These signals are explained in the following paragraphs.

Address Strobe ($\overline{AS}$)

This signal indicates that there is a valid address on the address bus.

Read/Write (R/$\overline{W}$)

This signal defines the data bus transfer as a read or write cycle. The R/$\overline{W}$ signal also works in conjunction with the data strobes as explained in the following paragraph.

Upper and Lower Data Strobe ($\overline{UDS}$, $\overline{LDS}$)

These signals control the flow of data on the data bus, as shown in Table 4. When the R/$\overline{W}$ line is high, the processor will read from the data bus as indicated. When the R/$\overline{W}$ line is low, the processor will write to the data bus as shown.

Table 4. Data Strobe Control of Data Bus

$\overline{UDS}$	$\overline{LDS}$	R/$\overline{W}$	D8-D15	D0-D7
High	High	–	No Valid Data	No Valid Data
Low	Low	High	Valid Data Bits 8-15	Valid Data Bits 0-7
High	Low	High	No Valid Data	Valid Data Bits 0-7
Low	High	High	Valid Data Bits 8-15	No Valid Data
Low	Low	Low	Valid Data Bits 8-15	Valid Data Bits 0-7
High	Low	Low	Valid Data Bits 0-7*	Valid Data Bits 0-7
Low	High	Low	Valid Data Bits 8-15	Valid Data Bits 8-15*

*These conditions are a result of current implementation and may not appear on future devices.

Data Transfer Acknowledge ($\overline{DTACK}$)

This input indicates that the data transfer is completed. When the processor recognizes $\overline{DTACK}$ during a read cycle, data is latched and the bus cycle terminated. When $\overline{DTACK}$ is recognized during a write cycle, the bus cycle is terminated.

BUS ARBITRATION CONTROL

The three signals, bus request, bus grant, and bus grant acknowledge, form a bus arbitration circuit to determine which device will be the bus master device.

Bus Request ($\overline{BR}$)

This input is wire ORed with all other devices that could be bus masters. This input indicates to the processor that some other device desires to become the bus master.

Bus Grant ($\overline{BG}$)

This output indicates to all other potential bus master devices that the processor will release bus control at the end of the current bus cycle.

Bus Grant Acknowledge ($\overline{BGACK}$)

This input indicates that some other device has become the bus master. This signal should not be asserted until the following four conditions are met:

1. a bus grant has been received,
2. address strobe is inactive which indicates that the microprocessor is not using the bus,
3. data transfer acknowledge is inactive which indicates that neither memory nor peripherals are using the bus, and
4. bus grant acknowledge is inactive which indicates that no other device is still claiming bus mastership.

INTERRUPT CONTROL ($\overline{IPL0}$, $\overline{IPL1}$, $\overline{IPL2}$)

These input pins indicate the encoded priority level of the device requesting an interrupt. Level seven is the highest priority while level zero indicates that no interrupts are requested. Level seven cannot be masked. The least significant bit is given in $\overline{IPL0}$ and the most significant bit is contained in $\overline{IPL2}$. These

lines must remain stable until the processor signals interrupt acknowledge (FC0-FC2 are all high) to insure that the interrupt is recognized.

SYSTEM CONTROL

The system control inputs are used to either reset or halt the processor and to indicate to the processor that bus errors have occurred. The three system control inputs are explained in the following paragraphs.

Bus Error ($\overline{BERR}$)

This input informs the processor that there is a problem with the cycle currently being executed. Problems may be a result of:

1. nonresponding devices,
2. interrupt vector number acquisition failure,
3. illegal access request as determined by a memory management unit, or
4. other application dependent errors.

The bus error signal interacts with the halt signal to determine if the current bus cycle should be re-executed or if exception processing should be performed.

Reset ($\overline{RESET}$)

This bidirectional signal line acts to reset (start a system initialization sequence) the processor in response to an external reset signal. An internally generated reset (result of a $\overline{RESET}$ instruction) causes all external devices to be reset and the internal state of the processor is not affected. A total system reset (processor and external devices) is the result of external $\overline{HALT}$ and $\overline{RESET}$ signals applied at the same time.

Halt ($\overline{HALT}$)

When this bidirectional line is driven by an external device, it will cause the processor to stop at the completion of the current bus cycle. When the processor has been halted using this input, all control signals are inactive and all three-state lines are put in their high-impedance state.

When the processor has stopped executing instructions, such as in a double bus fault condition, the $\overline{HALT}$ line is driven by the processor to indicate to external devices that the processor has stopped.

M6800 PERIPHERAL CONTROL

These control signals are used to allow the interfacing of synchronous M6800 peripheral devices with the asynchronous MC68HC000. These signals are explained in the following paragraphs.

Enable (E)

This signal is the standard enable signal common to all M6800 type peripheral devices. The period for this output is ten MC68HC000 clock periods (six clocks low, four clocks high). Enable is generated by an internal ring counter which may come up in any state (i.e., at power on, it is impossible to guarantee phase relationship of E to CLK). E is a free-running clock and runs regardless of the state of the bus on the MPU.

Valid Peripheral Address ($\overline{VPA}$)

This input indicates that the device or region addressed is an M68000 Family device and that data transfer should be synchronized with the enable (E) signal. This input also indicates that the processor should use automatic vectoring for an interrupt during an IACK cycle.

Valid Memory Address ($\overline{VMA}$)

This output is used to indicate to M68000 peripheral devices that there is a valid address on the address bus and the processor is synchronized to enable. This signal only responds to a valid peripheral address ($\overline{VPA}$) input which indicates that the peripheral is an M68000 Family device.

PROCESSOR STATUS (FC0, FC1, FC2)

These function code outputs indicate the state (user or supervisor) and the cycle type currently being executed, as shown in Table 5. The information indicated by the function code outputs is valid whenever address strobe ($\overline{AS}$) is active.

Table 5. Function Code Outputs

Function Code Output			
FC2	FC1	FC0	Cycle Time
Low	Low	Low	(Undefined, Reserved)
Low	Low	High	User Data
Low	High	Low	User Program
Low	High	High	(Undefined, Reserved)
High	Low	Low	(Undefined, Reserved)
High	Low	High	Supervisor Data
High	High	Low	Supervisor Program
High	High	High	Interrupt Acknowledge

CLOCK (CLK)

The clock input is a TTL-compatible signal that is internally buffered for development of the internal clocks needed by the processor. The clock input should not be gated off at any time and the clock signal must conform to minimum and maximum pulse width times. The clock is a constant frequency square wave with no stretching or shaping techniques required.

DATA TRANSFER OPERATIONS

Transfer of data between devices involves the following leads:

1. address bus A1 through A23,
2. data bus D0 through D15, and
3. control signals.

The address and data buses are separate parallel buses used to transfer data using an asynchronous bus structure. In all cycles, the bus master assumes responsibility for deskewing all signals it issues at both the start and end of a cycle. In addition, the bus master is responsible for deskewing the acknowledge and data signals from the slave device.

The following paragraphs explain the read, write, and read-modify-write cycles. The indivisible read-modify-write cycle is the method used by the MC68HC000 for interlocked multiprocessor communications.

READ CYCLE

During a read cycle, the processor receives data from the memory of a peripheral device. The processor reads bytes of data in all cases. If the instruction specifies a word (or double word) operation, the processor reads both upper and lower bytes simultaneously by asserting both upper and lower data

strobes. When the instruction specifies byte operation, the processor uses an internal A0 bit to determine which byte to read and then issues the data strobe required for that byte. For byte operations, when the A0 bit equals zero, the upper data strobe is issued. When the A0 bit equals one, the lower data strobe is issued. When the data is received, the processor correctly positions it internally.

WRITE CYCLE

During a write cycle, the processor sends data to either the memory or a peripheral device. The processor writes bytes of data in all cases. If the instruction specifies a word operation, the processor writes both bytes. When the instruction specifies a byte operation, the processor uses an internal A0 bit to determine which byte to write and then issues the data strobe required for that byte. For byte operations, when the A0 bit equals zero, the upper data strobe is issued. When the A0 bit equals one, the lower data strobe is issued.

READ-MODIFY-WRITE CYCLE

The read-modify-write cycle performs a read, modifies the data in the arithmetic-logic unit, and writes the data back to the same address. In the MC68HC000, this cycle is indivisible in that the address strobe is asserted throughout the entire cycle. The test and set (TAS) instruction uses this cycle to provide meaningful communication between processors in a multiple processor environment. This instruction is the only instruction that uses the read-modify-write cycles and since the test and set instruction only operates on bytes, all read-modify-write cycles are byte operations.

PROCESSING STATES

The MC68HC000 is always in one of three processing states: normal, exception, or halted.

NORMAL PROCESSING

The normal processing state is that associated with instruction execution; the memory references are to fetch instructions and operands, and to store results. A special case of normal state is the stopped state which the processor enters when a stop instruction is executed. In this state, no further references are made.

EXCEPTION PROCESSING

The exception processing state is associated with interrupts, trap instructions, tracing, and other exception conditions. The exception may be internally generated by an instruction or by an unusual condition arising during the execution of an instruction. Externally, exception processing can be forced by an interrupt, by a bus error, or by a reset. Exception processing is designed to provide an efficient context switch so that the processor may handle unusual conditions.

HALTED PROCESSING

The halted processing state is an indication of catastrophic hardware failure. For example, if during the exception processing of a bus error another bus error occurs, the processor assumes that the system is unusable and halts. Only an external reset can restart a halted processor. Note that a processor in the stopped state is not in the halted state, nor vice versa.

INTERFACE WITH M6800 PERIPHERALS

Motorola's extensive line of M6800 peripherals are directly compatible with the MC68HC000. Some of these devices that are particularly useful are:

MC6821 Peripheral Interface Adapter
MC6840 Programmable Timer Module
MC6843 Floppy Disk Controller
MC6845 CRT Controller
MC6850 Asynchronous Communications Interface Adapter
MC6854 Advanced Data Link Controller

To interface the synchronous M6800 peripherals with the asynchronous MC68HC000, the processor modifies its bus cycle to meet the M6800 cycle requirements whenever an M6800 device address is detected. This is possible since both the processors use memory mapped I/O.

ELECTRICAL SPECIFICATIONS

MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Supply Voltage	V_{CC}	−0.3 to +6.5	V
Input Voltage	V_{in}	−0.3 to +6.5	V
Operating Temperature Range MC68HC000	T_A	T_L to T_H 0 to 70	°C
Storage Temperature	T_{stg}	−55 to +150	°C

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

3

THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Symbol	Value	Rating
Thermal Resistance (Still Air)	θ_{JA}		θ_{JC}		°C/W
Ceramic, Type L/LC		30		15*	
Ceramic, Type R/RC		33		15	
Plastic, Type P		30		15*	
Plastic, Type FN		45		25*	

*Estimated

CMOS CONSIDERATIONS

The MC68HC000, with its significantly lower power consumption, has other considerations. The CMOS cell is basically composed of two complementary transistors (a P channel and an N channel), and only one transistor is turned on while the cell is in the steady state. The active P-channel transistor sources current when the output is a logic high and presents a high impedance when the output is a logic low. Thus, the overall result is extremely low power consumption because no power is lost through the active P-channel transistor. Also, since only one transistor is turned on during the steady state, power consumption is determined by leakage currents.

Because the basic CMOS cell is composed of two complementary transistors, a virtual semiconductor controlled rectifier (SCR) may be formed when an input exceeds the supply voltage. The SCR that is formed by this high input causes the device to become latched in a mode that may result in excessive current drain and eventual destruction of the device. Although the MC68HC000 is implemented with input protection diodes, care should be exercised to ensure that the maximum input voltage specification is not exceeded. Some systems may require that the CMOS circuitry be isolated from voltage transients; others may require no additional circuitry.

The MC68HC000, implemented in CMOS, is applicable to designs to which the following considerations are relevant:

1. The MC68HC000 completely satisfies the input/output drive requirements of CMOS logic devices.
2. The HCMOS MC68HC000 provides an order of magnitude reduction in power dissipation when compared to the HMOS MC68000. However, the MC68HC000 does not offer a "power down" mode. The minimum operating frequency of the MC68HC000 is 4 MHz.

DC ELECTRICAL CHARACTERISTICS (V_{CC} = 5.0 Vdc ± 5%; GND = 0 Vdc; $T_A = T_L$ to T_H)

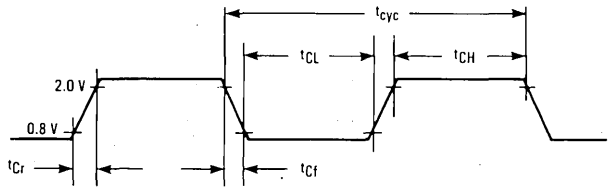
Characteristic		Symbol	Min	Max	Unit
Input High Voltage		V_{IH}	2.0	V_{CC}	V
Input Low Voltage		V_{IL}	GND – 0.3	0.8	V
Input Leakage Current	$\overline{BERR}$, $\overline{BGACK}$, $\overline{BR}$, $\overline{DTACK}$, CLK, $\overline{IPL0}$-$\overline{IPL2}$, $\overline{VPA}$	I_{in}	—	2.5	μA
@ 5.25 V	$\overline{HALT}$, $\overline{RESET}$		—	20	
Three-State (Off State) Input Current	$\overline{AS}$, A1-A23, D0-D15,	I_{TSI}	—	20	μA
@ 2.4 V/0.4 V	FC0-FC2, $\overline{LDS}$, R/$\overline{W}$, $\overline{UDS}$, $\overline{VMA}$				
Output High Voltage	E, $\overline{AS}$, A1-A23, $\overline{BG}$, D0-D15,	V_{OH}	V_{CC} – 0.75	—	V
(I_{OH} = –400 μA)	FC0-FC2, $\overline{LDS}$, R/$\overline{W}$, $\overline{UDS}$, $\overline{VMA}$				
Output Low Voltage		V_{OL}			V
(I_{OL} = 1.6 mA)	$\overline{HALT}$		—	0.5	
(I_{OL} = 3.2 mA)	A1-A23, $\overline{BG}$, FC0-FC2		—	0.5	
(I_{OL} = 5.0 mA)	$\overline{RESET}$		—	0.5	
(I_{OL} = 5.3 mA)	E, $\overline{AS}$, D0-D15, $\overline{LDS}$, R/$\overline{W}$, $\overline{UDS}$, $\overline{VMA}$		—	0.5	
Current Dissipation*	f = 8 MHz	I_D	—	25	mA
	f = 10 MHz		—	30	
	f = 12.5 MHz		—	35	
	f = 16.67 MHz		—	50	
Power Dissipation	f = 8 MHz	P_D	—	0.13	W
	f = 10 MHz		—	0.16	
	f = 12.5 MHz		—	0.19	
	f = 16.67 MHz		—	0.26	
Capacitance (V_{in} = 0 V, T_A = 25°C, Frequency = 1 MHz)**		C_{in}	—	20.0	pF
Load Capacitance	$\overline{HALT}$	C_L	—	70	pF
	All Others		—	130	

*Currents listed are with no loading.
**Capacitance is periodically sampled rather than 100% tested.

AC ELECTRICAL SPECIFICATIONS — CLOCK TIMING (See Figure 5)

Num.	Characteristic	Symbol	8 MHz*		10 MHz*		12.5 MHz*		16.67 MHz '12F'		Unit
			Min	Max	Min	Max	Min	Max	Min	Max	
	Frequency of Operation	f	4.0	8.0	4.0	10.0	4.0	12.5	8.0	16.7	MHz
1	Cycle Time	t_{cyc}	125	250	100	250	80	250	60	125	ns
2,3	Clock Pulse Width (Measured from 1.5 V to 1.5 V for 12F)	t_{CL}	55	125	45	125	35	125	27	62.5	ns
		t_{CH}	55	125	45	125	35	125	27	62.5	
4,5	Clock Rise and Fall Times	t_{Cr}	—	10	—	10	—	5	—	5	ns
		t_{Cf}	—	10	—	10	—	5	—	5	

*These specifications represent an improvement over previously published specifications for the 8-, 10-, and 12.5-MHz MC68HC000 and are valid only for product bearing date codes of 8827 and later.



NOTE: Timing measurements are referenced to and from a low voltage of 0.8 volt and high a voltage of 2.0 volts, unless otherwise noted. The voltage swing through this range should start outside and pass through the range such that the rise or fall will be linear between 0.8 volt and 2.0 volts.

Figure 5. Clock Input Timing

3

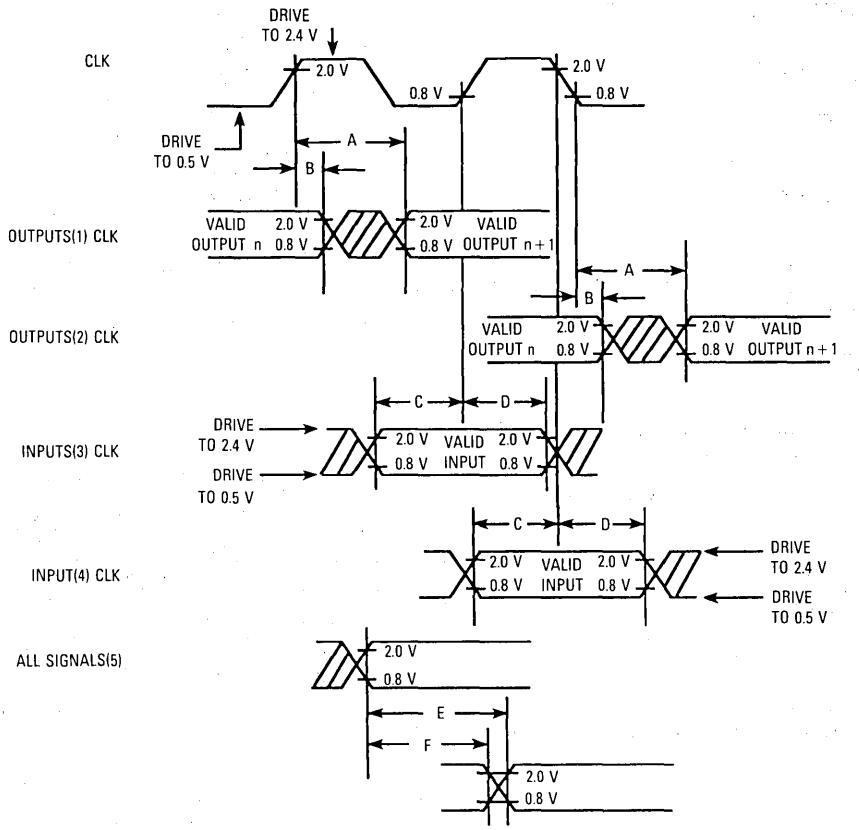
AC ELECTRICAL SPECIFICATION DEFINITIONS

The AC specifications presented consist of output delays, input setup and hold times, and signal skew times. All signals are specified relative to an appropriate edge of the clock and possibly to one or more other signals.

The measurement of the AC specifications is defined by the waveforms shown in Figure 6. In order to test the parameters guaranteed by Motorola, inputs must be driven to the voltage levels specified in that figure. Outputs are specified with minimum and/or maximum limits, as appropriate, and are measured as shown in Figure 6. Inputs are specified with minimum setup and hold times, and are measured as shown. Finally, the measurement for signal-to-signal specifications is also shown.

NOTE

The testing levels used to verify conformance to the AC specifications does not affect the guaranteed DC operation of the device as specified in the DC electrical characteristics.



NOTES:

1. This output timing is applicable to all parameters specified relative to the rising edge of the clock.
2. This output timing is applicable to all parameters specified relative to the falling edge of the clock.
3. This input timing is applicable to all parameters specified relative to the rising edge of the clock.
4. This input timing is applicable to all parameters specified relative to the falling edge of the clock.
5. This timing is applicable to all parameters specified relative to the assertion/negation of another signal.

LEGEND:

A. Maximum output delay specification.
B. Minimum output hold time.
C. Minimum input setup time specification.
D. Minimum input hold time specification.
E. Signal valid to signal valid specification (maximum or minimum).
F. Signal valid to signal invalid specification (maximum or minimum).

Figure 6. Drive Levels and Test Points for AC Specifications

AC ELECTRICAL SPECIFICATIONS — READ AND WRITE CYCLES (V_{CC} = 5.0 Vdc ± 5%; GND = 0 Vdc; $T_A = T_L$ to T_H; see Figures 7 and 8)

Num.	Characteristic	Symbol	8 MHz*		10 MHz*		12.5 MHz*		16.67 MHz '12F'		Unit
			Min	Max	Min	Max	Min	Max	Min	Max	
6	Clock Low to Address Valid	t_{CLAV}	—	62	—	50	—	50	—	50	ns
6A	Clock High to FC Valid	t_{CHFCV}	—	62	—	50	—	45	—	45	ns
7	Clock High to Address, Data Bus High Impedance (Maximum)	t_{CHADZ}	—	80	—	70	—	60	—	50	ns
8	Clock High to Address, FC Invalid (Minimum)	t_{CHAFI}	0	—	0	—	0	—	0	—	ns
9[1]	Clock High to $\overline{AS}$, $\overline{DS}$ Asserted	t_{CHSL}	3	60	3	50	3	40	3	40	ns
11[2]	Address Valid to $\overline{AS}$, $\overline{DS}$ Asserted (Read)/$\overline{AS}$ Asserted (Write)	t_{AVSL}	30	—	20	—	15	—	15	—	ns
11A[2]	FC Valid to $\overline{AS}$, $\overline{DS}$ Asserted (Read)/$\overline{AS}$ Asserted (Write)	t_{FCVSL}	90	—	70	—	60	—	30	—	ns
12[1]	Clock Low to $\overline{AS}$, $\overline{DS}$ Negated	t_{CLSH}	—	62	—	50	—	40	—	40	ns
13[2]	$\overline{AS}$, $\overline{DS}$ Negated to Address, FC Invalid	t_{SHAFI}	40	—	30	—	20	—	10	—	ns
14[2]	$\overline{AS}$ (and $\overline{DS}$ Read) Width Asserted	t_{SL}	270	—	195	—	160	—	120	—	ns
14A	$\overline{DS}$ Width Asserted (Write)	t_{DSL}	140	—	95	—	80	—	60	—	ns
15[2]	$\overline{AS}$, $\overline{DS}$ Width Negated	t_{SH}	150	—	105	—	65	—	60	—	ns
16	Clock High to Control Bus High Impedance	t_{CHCZ}	—	80	—	70	—	60	—	50	ns
17[2]	$\overline{AS}$, $\overline{DS}$ Negated to R/$\overline{W}$ Invalid	t_{SHRH}	40	—	30	—	20	—	10	—	ns
18[1]	Clock High to R/$\overline{W}$ High (Read)	t_{CHRH}	0	55	0	45	0	40	0	40	ns
20[1]	Clock High to R/$\overline{W}$ Low (Write)	t_{CHRL}	0	55	0	45	0	40	0	40	ns
20A[2,6]	$\overline{AS}$ Asserted to R/$\overline{W}$ Valid (Write)	t_{ASRV}	—	10	—	10	—	10	—	10	ns
21[2]	Address Valid to R/$\overline{W}$ Low (Write)	t_{AVRL}	20	—	0	—	0	—	0	—	ns
21A[2]	FC Valid to R/$\overline{W}$ Low (Write)	t_{FCVRL}	60	—	50	—	30	—	20	—	ns
22[2]	R/$\overline{W}$ Low to $\overline{DS}$ Asserted (Write)	t_{RLSL}	80	—	50	—	30	—	20	—	ns
23	Clock Low to Data-Out Valid (Write)	t_{CLDO}	—	62	—	50	—	50	—	50	ns
25[2]	$\overline{AS}$, $\overline{DS}$ Negated to Data-Out Invalid (Write)	t_{SHDOI}	40	—	30	—	20	—	15	—	ns
26[2]	Data-Out Valid to $\overline{DS}$ Asserted (Write)	t_{DOSL}	40	—	30	—	20	—	15	—	ns
27[5]	Data-In Valid to Clock Low (Setup Time on Read)	t_{DICL}	10	—	10	—	10	—	7	—	ns
28[2]	$\overline{AS}$, $\overline{DS}$ Negated to $\overline{DTACK}$ Negated (Asynchronous Hold)	t_{SHDAH}	0	240	0	190	0	150	0	110	ns
29	$\overline{AS}$, $\overline{DS}$ Negated to Data-In Invalid (Hold Time on Read)	t_{SHDII}	0	—	0	—	0	—	0	—	ns
29A	$\overline{AS}$, $\overline{DS}$ Negated to Data-In High Impedance	t_{SHDZ}	—	187	—	150	—	120	—	90	ns
30	$\overline{AS}$, $\overline{DS}$ Negated to $\overline{BERR}$ Negated	t_{SHBEH}	0	—	0	—	0	—	0	—	ns
31[2,5]	$\overline{DTACK}$ Asserted to Data-In Valid (Setup Time)	t_{DALDI}	—	90	—	65	—	50	—	40	ns
32	$\overline{HALT}$ and $\overline{RESET}$ Input Transition Time	$t_{RHr,f}$	0	200	0	200	0	200	0	150	ns
33	Clock High to $\overline{BG}$ Asserted	t_{CHGL}	—	62	—	50	—	40	—	40	ns
34	Clock High to $\overline{BG}$ Negated	t_{CHGH}	—	62	—	50	—	40	—	40	ns
35	$\overline{BR}$ Asserted to $\overline{BG}$ Asserted	t_{BRLGL}	1.5	3.5	1.5	3.5	1.5	3.5	1.5	3.5	Clks
36[7]	$\overline{BR}$ Negated to $\overline{BG}$ Negated	t_{BRHGH}	1.5	3.5	1.5	3.5	1.5	3.5	1.5	3.5	Clks
37	$\overline{BGACK}$ Asserted to $\overline{BG}$ Negated	t_{GALGH}	1.5	3.5	1.5	3.5	1.5	3.5	1.5	3.5	Clks
37A[8]	$\overline{BGACK}$ Asserted to $\overline{BR}$ Negated	t_{GALBRH}	20	1.5 Clks	20	1.5 Clks	20	1.5 Clks	10	1.5 Clks	ns
38	$\overline{BG}$ Asserted to Control, Address, Data Bus High Impedance ($\overline{AS}$ Negated)	t_{GLZ}	—	80	—	70	—	60	—	50	ns

3

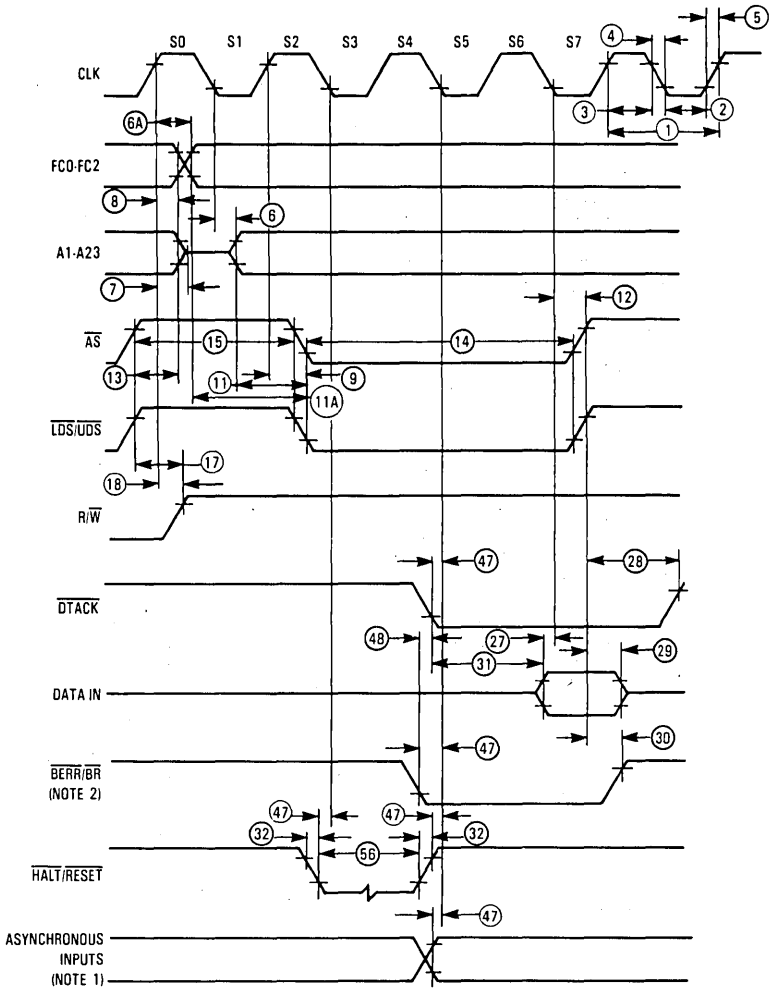
AC ELECTRICAL SPECIFICATIONS — READ AND WRITE CYCLES (Continued)

Num.	Characteristic	Symbol	8 MHz*		10 MHz*		12.5 MHz*		16.67 MHz '12F'		Unit
			Min	Max	Min	Max	Min	Max	Min	Max	
39	$\overline{BG}$ Width Negated	t_{GH}	1.5	—	1.5	—	1.5	—	1.5	—	Clks
40	Clock Low to $\overline{VMA}$ Asserted	t_{CLVML}	—	70	—	70	—	70	—	50	ns
41	Clock Low to E Transition	t_{CLET}	—	55	—	45	—	35	—	35	ns
42	E Output Rise and Fall Time	$t_{Er,f}$	—	15	—	15	—	15	—	15	ns
43	$\overline{VMA}$ Asserted to E High	t_{VMLEH}	200	—	150	—	90	—	80	—	ns
44	$\overline{AS}$, $\overline{DS}$ Negated to $\overline{VPA}$ Negated	t_{SHVPH}	0	120	0	90	0	70	0	50	ns
45	E Low to Control, Address Bus Invalid (Address Hold Time)	t_{ELCAI}	30	—	10	—	10	—	10	—	ns
46	$\overline{BGACK}$ Width Low	t_{GAL}	1.5	—	1.5	—	1.5	—	1.5	—	Clks
47[5]	Asynchronous Input Setup Time	t_{ASI}	10	—	10	—	10	—	10	—	ns
48[2,3]	$\overline{BERR}$ Asserted to $\overline{DTACK}$ Asserted	t_{BELDAL}	20	—	20	—	20	—	10	—	ns
49[9]	$\overline{AS}$, $\overline{DS}$, Negated to E Low	t_{SHEL}	−70	70	−55	55	−45	45	−35	35	ns
50	E Width High	t_{EH}	450	—	350	—	280	—	220	—	ns
51	E Width Low	t_{EL}	700	—	550	—	440	—	340	—	ns
53	Data-Out Hold from Clock High	t_{CHDOI}	0	—	0	—	0	—	0	—	ns
54	E Low to Data-Out Invalid	t_{ELDOI}	30	—	20	—	15	—	10	—	ns
55	R/$\overline{W}$ Asserted to Data Bus Impedance Change	t_{RLDBD}	30	—	20	—	10	—	0	—	ns
56[4]	$\overline{HALT}$/$\overline{RESET}$ Pulse Width	t_{HRPW}	10	—	10	—	10	—	10	—	Clks
57	$\overline{BGACK}$ Negated to $\overline{AS}$, $\overline{DS}$, R/$\overline{W}$ Driven	t_{GASD}	1.5	—	1.5	—	1.5	—	1.5	—	Clks
57A	$\overline{BGACK}$ Negated to FC, $\overline{VMA}$ Driven	t_{GAFD}	1	—	1	—	1	—	1	—	Clks
58[7]	$\overline{BR}$ Negated to $\overline{AS}$, $\overline{DS}$, R/$\overline{W}$ Driven	t_{RHSD}	1.5	—	1.5	—	1.5	—	1.5	—	Clks
58A[7]	$\overline{BR}$ Negated to FC, $\overline{VMA}$ Driven	t_{RHFD}	1	—	1	—	1	—	1	—	Clks

*These specifications represent an improvement over previously published specifications for the 8-, 10-, and 12.5-MHz MC68HC000 and are valid only for product bearing date codes of 8827 and later.

NOTES:
1. For a loading capacitance of less than or equal to 50 picofarads, subtract 5 nanoseconds from the value given in the maximum columns.
2. Actual value depends on clock period.
3. If #47 is satisfied for both $\overline{DTACK}$ and $\overline{BERR}$, #48 may be ignored. In the absence of $\overline{DTACK}$, $\overline{BERR}$ is an asynchronous input using the asynchronous input setup time (#47).
4. For power-up, the MC68000 must be held in the reset state for 100 milliseconds to allow stabilization of on-chip circuitry. After the system is powered up, #56 refers to the minimum pulse width required to reset the processor.
5. If the asynchronous input setup time (#47) requirement is satisfied for $\overline{DTACK}$, the $\overline{DTACK}$-asserted to data setup time (#31) requirement can be ignored. The data must only satisfy the data-in to clock low setup time (#27) for the following clock cycle.
6. When $\overline{AS}$ and R/$\overline{W}$ are equally loaded (±20%), subtract 5 nanoseconds from the values given in these columns.
7. The processor will negate $\overline{BG}$ and begin driving the bus again if external arbitration logic negates $\overline{BR}$ before asserting $\overline{BGACK}$.
8. The minimum value must be met to guarantee proper operation. If the maximum value is exceeded, $\overline{BG}$ may be re-asserted.
9. The falling edge of S6 triggers both the negation of the strobes ($\overline{AS}$ and x$\overline{DS}$) and the falling edge of E. Either of these events can occur first, depending upon the loading on each signal. Specification #49 indicates the absolute maximum skew that will occur between the rising edge of the strobes and the falling edge of the E clock.

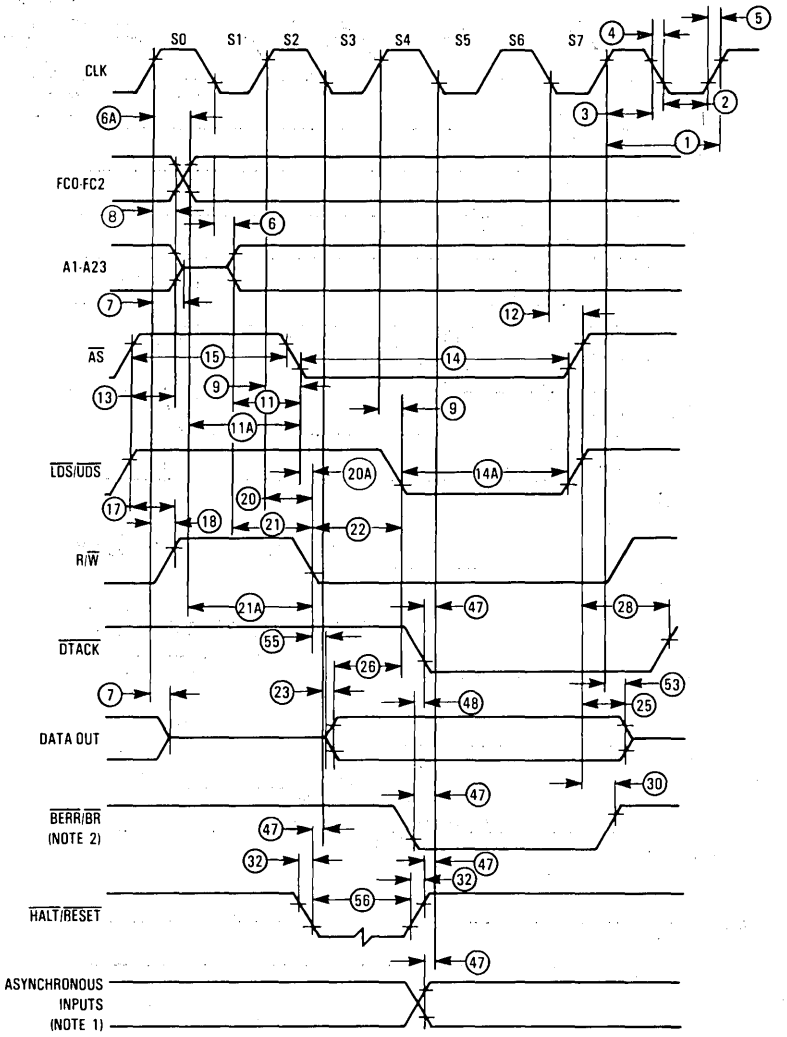
These waveforms should only be referenced in regard to the edge-to-edge measurement of the timing specifications. They are not intended as a functional description of the input and output signals. Refer to other functional descriptions and their related diagrams for device operation.



NOTES:
1. Setup time for the asynchronous inputs IPL0-IPL2 and VPA (#47) guarantees their recognition at the next falling edge of the clock.
2. BR need fall at this time only in order to insure being recognized at the end of the bus cycle.
3. Timing measurements are referenced to and from a low voltage of 0.8 volt and a high voltage of 2.0 volts, unless otherwise noted. The voltage swing through this range should start outside and pass through the range such that the rise or fall is linear between 0.8 volt and 2.0 volts.

Figure 7. MC68HC000 Read-Cycle Timing Diagram

These waveforms should only be referenced in regard to the edge-to-edge measurement of the timing specifications. They are not intended as a functional description of the input and output signals. Refer to other functional descriptions and their related diagrams for device operation.



NOTES:
1. Timing measurements are referenced to and from a low voltage of 0.8 volt and a high voltage of 2.0 volts, unless otherwise noted. The voltage swing through this range should start outside and pass through the range such that the rise or fall is linear between 0.8 volt and 2.0 volts.
2. Because of loading variations, R/$\overline{W}$ may be valid after $\overline{AS}$ even though both are initiated by the rising edge of S2 (specification #20A).

Figure 8. MC68HC000 Write-Cycle Timing Diagram

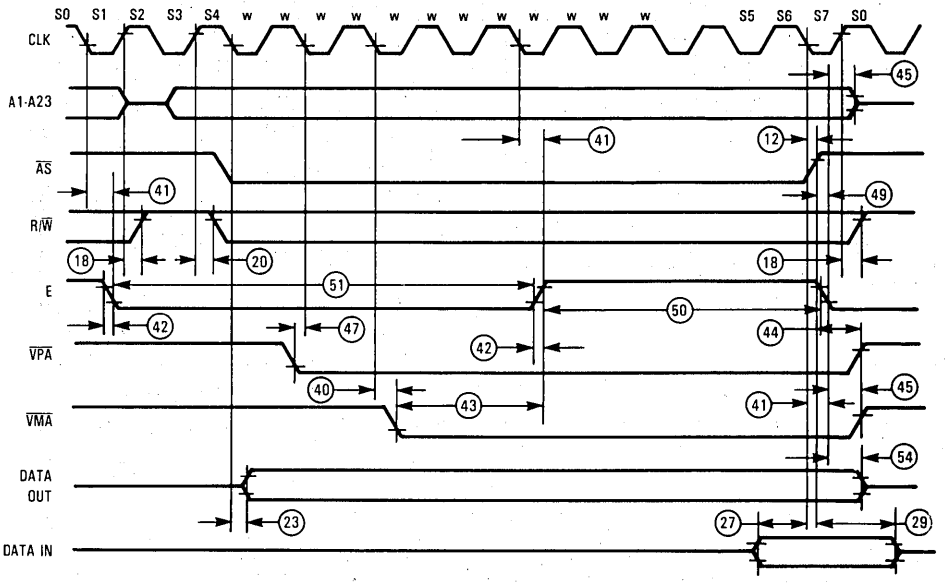
AC ELECTRICAL SPECIFICATIONS — MC68HC000 TO M6800 PERIPHERAL CYCLES (V_{CC} = 5.0 Vdc ± 5%; GND = 0 Vdc; $T_A = T_L$ to T_H; see Figures 9 and 10)

Num.	Characteristic	Symbol	8 MHz*		10 MHz*		12.5 MHz*		16.67 MHz '12F'		Unit
			Min	Max	Min	Max	Min	Max	Min	Max	
12[1]	Clock Low to $\overline{AS}$, $\overline{DS}$ Negated	t_{CLSH}	—	62	—	50	—	40	—	40	ns
18[1]	Clock High to R/$\overline{W}$ High (Read)	t_{CHRH}	0	55	0	45	0	40	0	40	ns
20[1]	Clock High to R/$\overline{W}$ Low (Write)	t_{CHRL}	0	55	0	45	0	40	0	40	ns
23	Clock Low to Data-Out Valid (Write)	t_{CLDO}	—	62	—	50	—	50	—	50	ns
27	Data-In Valid to Clock Low (Setup Time of Read)	t_{DICL}	10	—	10	—	10	—	7	—	ns
29	$\overline{AS}$, $\overline{DS}$ Negated to Data-In Invalid (Hold Time on Read)	t_{SHDII}	0	—	0	—	0	—	0	—	ns
40	Clock Low to $\overline{VMA}$ Asserted	t_{CLVML}	—	70	—	70	—	70	—	50	ns
41	Clock Low to E Transition	t_{CLET}	—	55	—	45	—	35	—	35	ns
42	E Output Rise and Fall Time	$t_{Er,f}$	—	15	—	15	—	15	—	15	ns
43	$\overline{VMA}$ Asserted to E High	t_{VMLEH}	200	—	150	—	90	—	80	—	ns
44	$\overline{AS}$, $\overline{DS}$ Negated to $\overline{VPA}$ Negated	t_{SHVPH}	0	120	0	90	0	70	0	50	ns
45	E Low to Control, Address Bus Invalid (Address Hold Time)	t_{ELCAI}	30	—	10	—	10	—	10	—	ns
47	Asynchronous Input Setup Time	t_{ASI}	10	—	10	—	10	—	10	—	ns
49[2]	$\overline{AS}$, $\overline{DS}$, Negated to E Low	t_{SHEL}	−70	70	−55	55	−45	45	−35	35	ns
50	E Width High	t_{EH}	450	—	350	—	280	—	220	—	ns
51	E Width Low	t_{EL}	700	—	550	—	440	—	340	—	ns
54	E Low to Data-Out Invalid	t_{ELDOI}	30	—	20	—	15	—	10	—	ns

*These specifications represent an improvement over previously published specifications for the 8-, 10-, and 12.5-MHz MC68HC000 and are valid only for product bearing date codes of 8827 and later.

NOTES:
1. For a loading capacitance of less than or equal to 50 picofarads, subtract 5 nanoseconds from the value given in the maximum columns.
2. The falling edge of S6 trigger both the negation of the strobes ($\overline{AS}$ and $\overline{xDS}$) and the falling edge of E. Either of these events can occur first, depending upon the loading on each signal. Specification #49 indicates the absolute maximum skew that will occur between the rising edge of the strobes and the falling edge of the E clock.

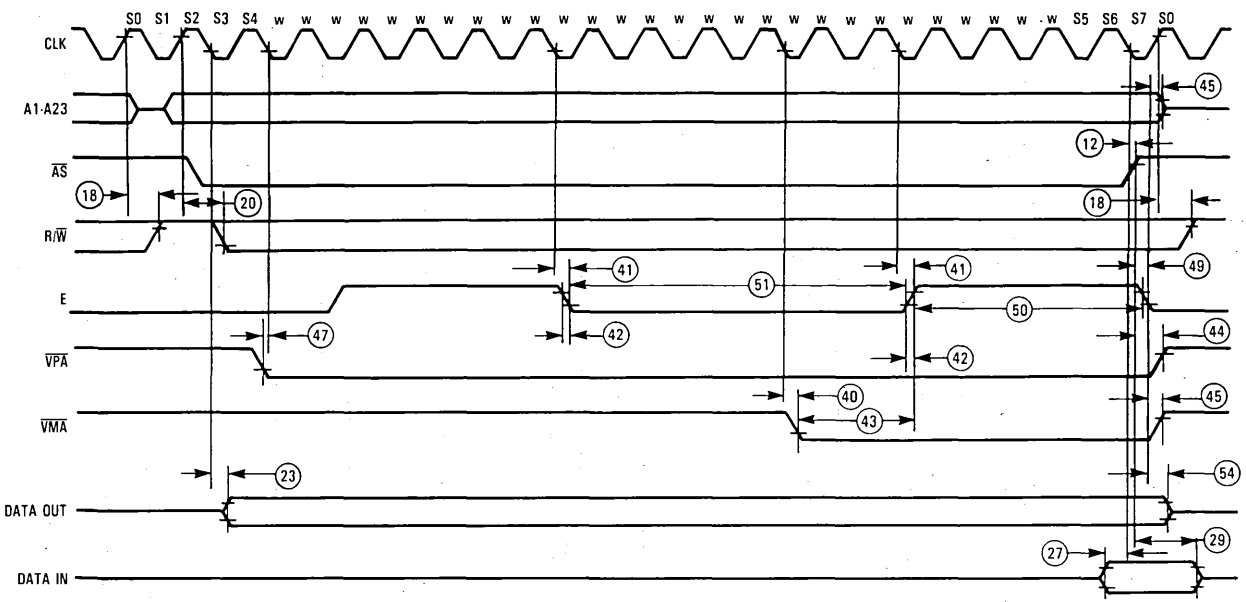
These waveforms should only be referenced in regard to the edge-to-edge measurement of the timing specifications. They are not intended as a functional description of the input and output signals. Refer to other functional descriptions and their related diagrams for device operation.



NOTE: This timing diagram is included for those who wish to design their own circuit to generate $\overline{VMA}$. It shows the best case possibly attainable.

Figure 9. MC68HC000 to M6800 Peripheral Timing Diagram (Best Case)

These waveforms should only be referenced in regard to the edge-to-edge measurement of the timing specifications. They are not intended as a functional description of the input and output signals. Refer to other functional descriptions and their related diagrams for device operation.



NOTE: This timing diagram is included for these who wish to design their own circuit to generate $\overline{VMA}$. It shows the worst case possibly attainable.

Figure 10. MC68HC000 to M6800 Peripheral Timing Diagram (Worst Case)

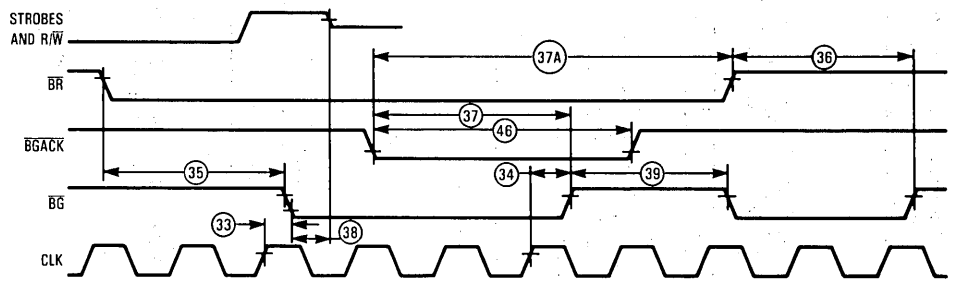
AC ELECTRICAL SPECIFICATIONS — BUS ARBITRATION (V_{CC} = 5.0 Vdc ± 5%; GND = 0 Vdc; $T_A = T_L$ to T_H; see Figure 11)

Num.	Characteristic	Symbol	8 MHz*		10 MHz*		12.5 MHz*		16.67 MHz '12F'		Unit
			Min	Max	Min	Max	Min	Max	Min	Max	
7	Clock High to Address, Data Bus High Impedance (Maximum)	t_{CHADZ}	—	80	—	70	—	60	—	50	ns
16	Clock High to Control Bus High Impedance	t_{CHCZ}	—	80	—	70	—	60	—	50	ns
33	Clock High to $\overline{BG}$ Asserted	t_{CHGL}	—	62	—	50	—	40	—	40	ns
34	Clock High to $\overline{BG}$ Negated	t_{CHGH}	—	62	—	50	—	40	—	40	ns
35	$\overline{BR}$ Asserted to $\overline{BG}$ Asserted	t_{BRLGL}	1.5	3.5	1.5	3.5	1.5	3.5	1.5	3.5	Clks
36[1]	$\overline{BR}$ Negated to $\overline{BG}$ Negated	t_{BRHGH}	1.5	3.5	1.5	3.5	1.5	3.5	1.5	3.5	Clks
37	$\overline{BGACK}$ Asserted to $\overline{BG}$ Negated	t_{GALGH}	1.5	3.5	1.5	3.5	1.5	3.5	1.5	3.5	Clks
37A[2]	$\overline{BGACK}$ Asserted to $\overline{BR}$ Negated	t_{GALBRH}	20	1.5 Clks	20	1.5 Clks	20	1.5 Clks	10	1.5 Clks	ns
38	$\overline{BG}$ Asserted to Control, Address, Data Bus High Impedance ($\overline{AS}$ Negated)	t_{GLZ}	—	80	—	70	—	60	—	50	ns
39	$\overline{BG}$ Width Negated	t_{GH}	1.5	—	1.5	—	1.5	—	1.5	—	Clks
46	$\overline{BGACK}$ Width Low	t_{GAL}	1.5	—	1.5	—	1.5	—	1.5	—	Clks
47	Asynchronous Input Setup Time	t_{ASI}	10	—	10	—	10	—	10	—	ns
57	$\overline{BGACK}$ Negated to $\overline{AS}$, $\overline{DS}$, R/$\overline{W}$ Driven	t_{GASD}	1.5	—	1.5	—	1.5	—	1.5	—	Clks
57A	$\overline{BGACK}$ Negated to FC, $\overline{VMA}$ Driven	t_{GAFD}	1	—	1	—	1	—	1	—	Clks
58[1]	$\overline{BR}$ Negated to $\overline{AS}$, $\overline{DS}$, R/$\overline{W}$ Driven	t_{RHSD}	1.5	—	1.5	—	1.5	—	1.5	—	Clks
58A[1]	$\overline{BR}$ Negated to FC, $\overline{VMA}$ Driven	t_{RHFD}	1	—	1	—	1	—	1	—	Clks

*These specifications represent an improvement over previously published specifications for the 8-, 10-, and 12.5-MHz MC68HC000 and are valid only for product bearing date codes of 8827 and later.

NOTES:
1. The processor will negate $\overline{BG}$ and begin driving the bus again if external arbitration logic negates $\overline{BR}$ before asserting $\overline{BGACK}$.
2. The minimum value must be met to guarantee proper operation. If the maximum value is exceeded, $\overline{BG}$ may be re-asserted.

These waveforms should only be referenced in regard to the edge-to-edge measurement of the timing specifications. They are not intended as a functional description of the input and output signals. Refer to other functional descriptions and their related diagrams for device operation.

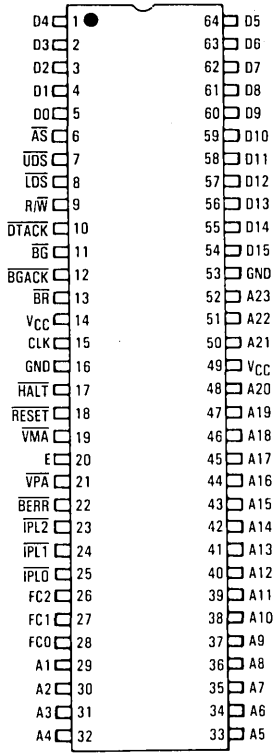


NOTE: Setup time to the clock (#47) for the asynchronous inputs $\overline{BERR}$, $\overline{BGACK}$, $\overline{BR}$, $\overline{DTACK}$, $\overline{IPL0}$-$\overline{IPL2}$ and $\overline{VPA}$ guarantees their recognition at the next falling edge of the clock.

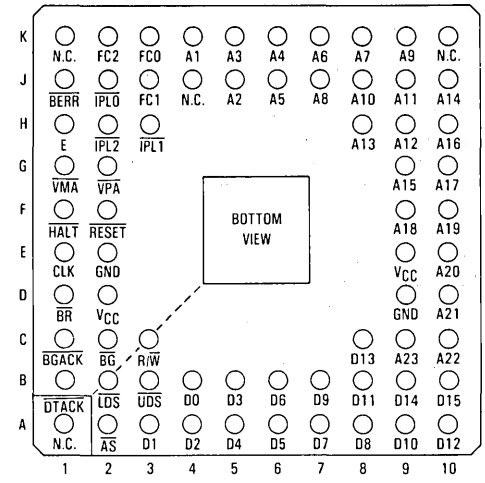
Figure 11. MC68HC000 Bus Arbitration Timing Diagram

PIN ASSIGNMENTS

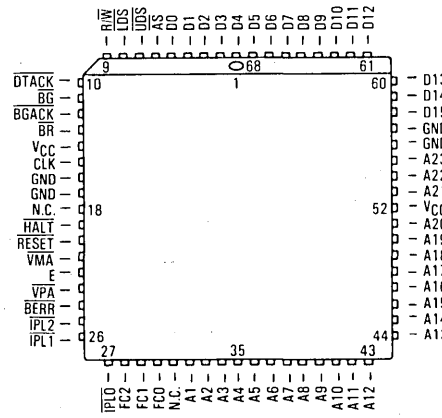
64-PIN DUAL-IN-LINE PACKAGE



68-TERMINAL PIN GRID ARRAY



68-LEAD QUAD PACK



MC68008

Technical Summary

16-Bit Microprocessor With 8-Bit Data Bus

3

This document contains both a summary of the MC68008 as well as a detailed set of parametrics. The purpose is twofold — to provide an introduction to the MC68008 and support for the sophisticated user. For detailed information on the MC68008 refer to the *MC68008 Advance Information Data Sheet.*

The MC68008 is a member of the M68000 Family of advanced microprocessors. This device allows the design of cost effective systems using 8-bit data buses while providing the benefits of a 32-bit microprocessor architecture. The performance of the MC68008 is greater than any 8-bit microprocessor and superior to several 16-bit microprocessors. Resources available to the MC68008 user consist of the following:

- 17 32-Bit Data and Address Registers
- 56 Basic Instruction Types
- Extensive Exception Processing
- Memory Mapped I O
- 14 Addressing Modes
- Complete Code Compatibility with the MC68000

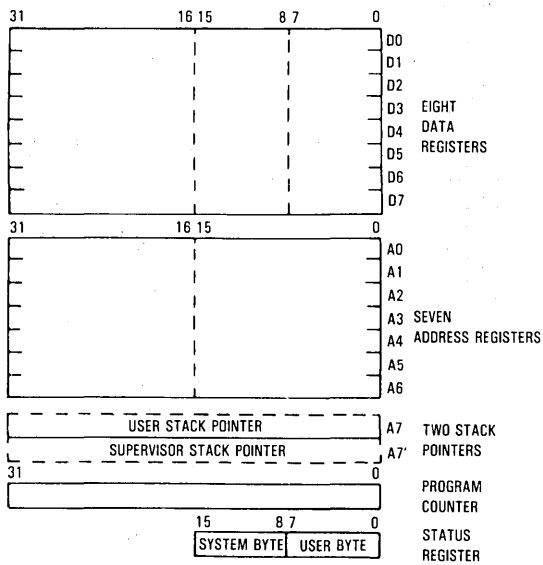


Figure 1. Programming Model

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

INTRODUCTION

The MC68008 allows the design of cost effective systems using 8-bit data buses while providing the benefits of a 32-bit microprocessor architecture. The performance of the MC68008 is greater than any 8-bit microprocessor and superior to several 16-bit microprocessors.

A system implementation based on an 8-bit data bus reduces system cost in comparison to 16-bit systems due to a more effective use of components and the fact that byte-wide memories and peripherals can be used much more effectively. In addition, the non-multiplexed address and data buses eliminate the need for external demultiplexers, thus further simplifying the system.

The MC68008 has full code compatibility (source and object) with the MC68000 which allows programs to be run on either MPU, depending on performance requirements and cost objectives.

The MC68008 is available as a 48-pin dual-in-line package (in plastic or ceramic) and 52-pin quad plastic package. Among the four additional pins of the 52-pin package, two additional address lines are included beyond the 20 address lines of the 48-pin package. The address reach of the MC68008 is 1 of 4 megabytes with the 48- or 52-pin package, respectively.

The large non-segmented linear address space of the MC68008 allows large modular programs to be developed and executed efficiently. A large linear address space allows program segment sizes to be determined by the application rather than forcing the designer to adopt an arbitrary segment size without regard to the applicaton's individual requirements.

The programmer's model is identical to that of the MC68000, as shown in Figure 1, with seventeen 32-bit registers, a 32-bit program counter, and a 16-bit status register. The first eight registers (D0-D7) are used as data registers for byte (8-bit), word (16-bit), and long word (32-bit) operations. The second set of seven registers (A0-A6), the user stack pointer (A7), and the system stack pointer (A7') may be used as software stack pointers and base address registers. In addition, the registers may be used for some simple word and long word data operations. All of the 17 registers may be used as index registers.

The system stack is used by many instructions. The 14 addressing modes allow the creation of user stacks and queues. While all of the address registers can be used to create stacks and queues, the A7 register by convention is used as the system stack pointer. Supplementing this convention is another address register, A7'; also referred to as the system stack pointer. This powerful concept allows the supervisor mode and user mode of the MC68008 to each have their own system stack pointer (consistently reffered to as SP) without needing to move pointers for each context of use when the mode is switched.

The system stack pointer (SP) is either the supervisor stack pointer (A7'≡SSP) or the user stack pointer (A7≡USP), depending on the state of the S bit in the status register. If the S bit is set, indication that the processor is in the supervisor state, then the SSP is the active system stack pointer and the USP is not used. If the S bit is clear, indicating that the processor is in the user state, then the USP is the active system stack pointer and the SSP is protected from user modification.

The status register, shown in Figure 2, may be considered as two bytes, the user byte and the system byte. The user byte contains five bits defining the overflow (V), zero (Z), negative (N), carry (C), and extended (X) condition codes. The system byte contains five defined bits. Three bits are used to define the current interrupt priority; and any interrupt level higher than the current mask level will be recognized. (Note that level 7 interrupts are non-maskable – that is, level 7 interrupts are always processed.) Two additional bits indicate whether the processor is in the trace (T) mode and/or in the supevisor (S) state.

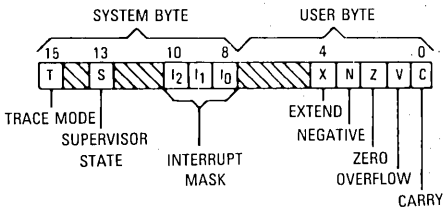


Figure 2. Status Register

DATA TYPES AND ADDRESSING MODES

Five basic data types are supported. These data types are:

- Bits
- BCD Digits (4 bits)
- Bytes (8 bits)
- Words (16 bits)
- Long Words (32 bits)

In addition, operations on other data types such as memory addresses, status word data, etc. are provided in the instruction set.

Most instructions can use any of the 14 addressing modes which are listed in Table 1. These addressing modes consist of six basic types.

- Register Direct
- Register Indirect
- Absoulte
- Program Counter Relative
- Immediate
- Implied

The register indirect addressing modes also have the capability to perform postincrementing, predecrementing, offsetting, and indexing. The program counter relative mode may be used in combination with indexing and offsetting for writing relocatable programs.

INSTRUCTION SET OVERVIEW

The MC68008 is completely code compatible with the MC68000. This means that the programs developed for the MC68000 will run on the MC68008 and visa versa. This applies equally to either source code or object code.

Table 1. Addressing Modes

Addressing Modes	Syntax
Register Direct Addressing	
Data Register Direct	Dn
Address Register Direct	An
Absolute Data Addressing	
Absolute Short	xxx.W
Absolute Long	xxx.L
Program Counter Relative Addressing	
Relative with Offset	d_{16}(PC)
Relative with Index Offset	d_8(PC,Xn)
Register Indirect Addressing	
Register Indirect	(An)
Postincrement Register Indirect	(An)+
Predecrement Register Indirect	–(An)
Register Indirect with Offset	d_{16}(An)
Indexed Register Indirect with Offset	d_8(An,Xn)
Immediate Data Addressing	
Immediate	#xxx
Quick Immediate	#1-#8
Implied Addressing	
Implied Register	SR/USP/SP/PC

NOTES:
Dn = Data Register
An = Address Register
Xn = Address or Data Register used as Index Register
SR = Status Register
PC = Program Counter
SP = Stack Pointer
USP = User Stack Pointer
( ) = Contents of
d_8 = 8-Bit Offset (Displacement)
d_{16} = 16-Bit Offset (Displacement)
#xxx = Immediate Data

This instruction set was designed to minimize the number of mnemonics remembered by the programmer. To further reduce the programmer's burden, the addressing modes are orthogonal.

The instruction set, shown in Table 2, forms a set of programming tools that include all processor functions to perform data movement, integer arithmetic, logical operations, shift and rotate operations, bit manipulation, BCD opertions, and both program and system control. Some additional instructions are variations or subsets of these and appear in Table 3.

SIGNAL DESCRIPTION

The MC68008 is available in two package sizes (48-pin and 52-pin). The additional four pins of the 52-pin quad package allow for additional signals: A20, A21, $\overline{\text{BGACK}}$, and $\overline{\text{IPL2}}$.

Throughout this document, references to the address bus pins (A0-A19) and the interrupt priority level pins ($\overline{\text{IPL0}}$/$\overline{\text{IPL2}}$,$\overline{\text{IPL1}}$) refer to A0-A21, and $\overline{\text{IPL0}}$, $\overline{\text{IPL1}}$, and $\overline{\text{IPL2}}$ for the 52-pin version of the MC68008.

The input and output signals can be functionally organized into the groups shown in Figure 3(a) for the 48-pin version and in Figure 3(b) for the 52-pin version. The following paragraphs provide a brief description of the

Table 2. Instruction Set Summary

Mnemonic	Description
ABCD	Add Decimal With Extend
ADD	Add
AND	Logical AND
ASL	Arithmetic Shift Left
ASR	Arithmetic Shift Right
Bcc	Branch Conditionally
BCHG	Bit Test and Change
BCLR	Bit Test and Clear
BRA	Branch Always
BSET	Bit Test and Set
BSR	Branch to Subroutine
BTST	Bit Test
CHK	Check Register Against Bounds
CLR	Clear Operand
CMP	Compare
DBcc	Test Condition, Decrement and Branch
DIVS	Signed Divide
DIVU	Unsigned Divide
EOR	Exclusive OR
EXG	Exchange Registers
EXT	Sign Extend
JMP	Jump
JSR	Jump to Subroutine
LEA	Lead Effective Address
LINK	Link Stack
LSL	Logical Shift Left
LSR	Logical Shift Right
MOVE	Move
MULS	Signed Multiply
MULU	Unsigned Multiply
NBCD	Negate Decimal with Extend
NEG	Negate
NOP	No Operation
NOT	One's Complement
OR	Logical OR
PEA	Push Effective Address
RESET	Reset External Devices
ROL	Rotate Left without Extend
ROR	Rotate Right without Extend
ROXL	Rotate Left with Extend
ROXR	Rotate Right with Extend
RTE	Return from Exception
RTR	Return and Restore
RTS	Return from Subroutine
SBCD	Subtract Decimal with Extend
Scc	Set Conditional
STOP	Stop
SUB	Subtract
SWAP	Swap Data Register Halves
TAS	Test and Set Operand
TRAP	Trap
TRAPV	Trap on Overflow
TST	Test
UNLK	Unlink

Table 3. Variations of Instruction Types

Instruction Type	Variation	Description
ADD	ADD	Add
	ADDA	Add Address
	ADDQ	Add Quick
	ADDI	Add Immediate
	ADDX	Add with Extend
AND	AND	Logical AND
	ANDI	AND Immediate
	ANDI to CCR	AND Immediate to Condition Codes
	ANDI to SR	AND Immediate to Status Register
CMP	CMP	Compare
	CMPA	Compare Address
	CMPM	Compare Memory
	CMPI	Compare Immediate
EOR	EOR	Exclusive OR
	EORI	Exclusive OR Immediate
	EORI to CCR	Exclusive OR Immediate to Condition Codes
	EORI to SR	Exclusive OR Immediate to Status Register
MOVE	MOVE	Move
	MOVEA	Move Address
	MOVEM	Move Multiple Registers
	MOVEP	Move Peripheral Data
	MOVEQ	Move Quick
	MOVE from SR	Move from Status Register
	MOVE to SR	Move to Status Register
	MOVE to CCR	Move to Condition Codes
	MOVE USP	Move User Stack Pointer
NEG	NEG	Negate
	NEGX	Negate with Extend
OR	OR	Logical OR
	ORI	OR Immediate
	ORI to CCR	OR Immediate to Condition Codes
	ORI to SR	OR Immediate to Status Register
SUB	SUB	Subtract
	SUBA	Subtract Address
	SUBI	Subtract Immediate
	SUBQ	Subtract Quick
	SUBX	Subtract with Extend

signals and a reference (if applicable) to other paragraphs that contain more information about the function being performed.

ADDRESS BUS (48-Pin: A0 through A19, 52-Pin: A0 through A21)

This unidirectional three-state bus provides the address for bus operation during all cycles except interrupt acknowledge cycles. During interrupt acknowledge cycles, address lines A1, A2, and A3 provide information about what level interrupt is being serviced while address lines A0 and A4 through A19 (A21) are all driven high.

DATA BUS (D0 through D7)

This 8-bit, bidirectional, three-state bus is the general purpose data path. During an interrupt acknowledge cycle, the external device supplies the vector number on data lines D0-D7.

ASYNCHROUNOUS BUS CONTROL

Asynchrounous data transfers are handled using the following control signals: address strobe, read/write, data strobe, and data transfer acknowledge. These signals are explained in the following paragraphs.

Address Strobe ($\overline{AS}$)

This three-state signal indicates that there is a valid address on the address bus. It is also used to "lock" the bus during the read-modify-write cycle used by the test and set (TAS) instruction.

Read/Write (R/$\overline{W}$)

This three-state signal defines the data bus transfer as a read or write cycle. The R/$\overline{W}$ signal also works in conjunction with the data strobe as explained in the following paragraph.

Data Strobe ($\overline{DS}$)

This three-state signal control the flow of data on the data bus as shown in Table 4. When the R/$\overline{W}$ line is high, the processor will read from the data bus as indicated. When the R/$\overline{W}$ is low, the processor will write to the data bus as shown.

Table 4. Data Strobe Control of Data Bus

$\overline{DS}$	R/$\overline{W}$	D0-D7
1	—	No Valid Data
0	1	Valid Data Bits 0-7 (Read Cycle)
0	0	Valid Data Bits 0-7 (Write Cycle)

Data Transfer Acknowledge ($\overline{DTACK}$)

This input indicates that the data transfer is completed. Then the processor recognizes $\overline{DTACK}$ during a read cycle, data is latched and the bus cycle is terminated. When $\overline{DTACK}$ is recognized during a write cycle, the bus cycle is terminated.

BUS ARBITRATION CONTROL

The 48-pin MC68008 contains a simple two-wire arbitration circuit and the 52-pin MC68008 contains a full three-wire MC68000 bus arbitration control. Both versions are designed to work with the daisy-chained networks, priority encoded networks, or a combination of these techniques. This circuit is used in determining which device will be the bus master device.

Bus Request ($\overline{BR}$)

This input is wire-ORed with all other devices that could be bus masters. This device indicates to the processor that some other device desires to become the bus master.

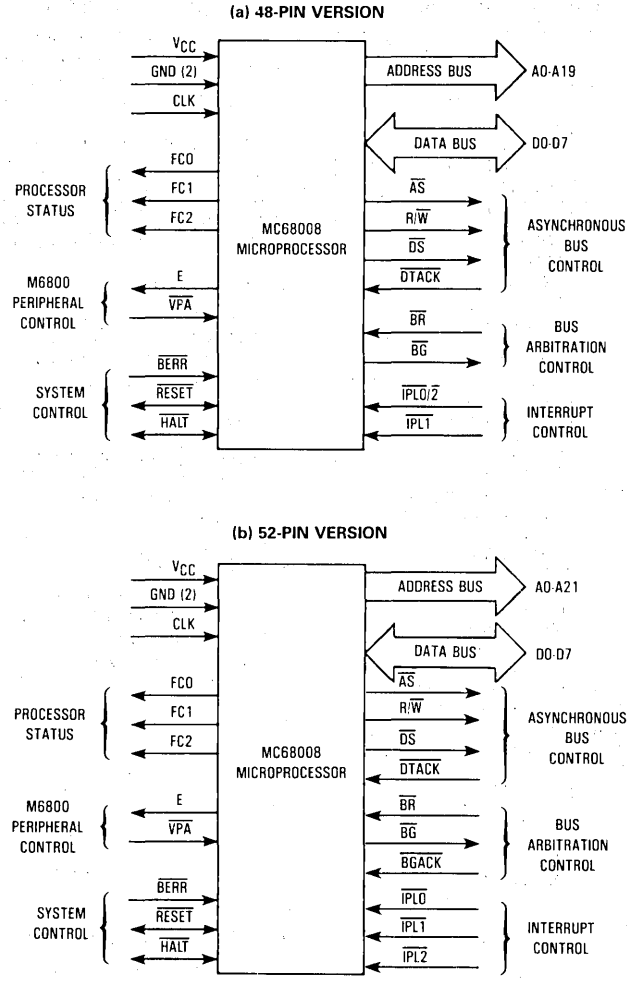


Figure 3. Input and Output Signals

Bus requests may be issued at any time in cycle or even if no cycle is being performed.

Bus Grant ($\overline{BG}$)

This output indicates to all other potential bus master devices that the processor will release bus control at the end of the current bus cycle.

Bus Grant Acknowledge ($\overline{BGACK}$)

This input, available on the 52-pin version only, indicates that some other device has become the bus master. This signal should not be asserted until the following four conditions are met:

1. a bus grant has been recieved,
2. address strobe is inactive which indicates that the microprocessor is not using the bus,
3. data transfer acknowledge is inactive which indicates that neither memory nor peripherals are using the bus, and
4. bus grant acknowledge is inactive which indicates that no other device is still claiming bus mastership.

NOTES

1) There is a two-clock interval straddling the transition of $\overline{AS}$ from the inactive state to the active state during which $\overline{BG}$ can not be issued.
2) If an existing MC68000 system is retrofitted to use the MC68008, 48-pin version (using

$\overline{BR}$ and $\overline{BG}$ only), the existing $\overline{BR}$ and $\overline{BGACK}$ signals should be ANDed and the resultant signal connected to the MC68008's $\overline{BR}$.

INTERRUPT CONTROL (48-Pin: $\overline{IPL0}$/$\overline{IPL2}$, $\overline{IPL1}$ 52-Pin: $\overline{IPL0}$, $\overline{IPL1}$, $\overline{IPL2}$)

These input pins indicate the encoded priority level of the device requesting an interrupt. The MC68000 and the 52-pin MC68008 MPU's use three pins to encode a range of 0-7 but, for the 48-pin MC68008 only two pins are available. By connecting the $\overline{IPL0}$/$\overline{IPL2}$ pin to both the $\overline{IPL0}$ and $\overline{IPL2}$ inputs internally, the 48-pin encodes values of 0, 2, 5, and 7. Level zero is used to indicate that there are no interrupts pending and level seven is a non-maskable edge-triggered interrupt. Except for level seven, the requesting level must be greater than the interrupt mask level contained in the processor status register before the processor will acknowledge the request.

The level presented to these inputs is continually monitored to allow for the case of a requesting level that is less than or equal to the processor status register level to be followed by a request that is greater than the processor status register level. A satisfactory interrupt condition must exist for two successive clocks before triggering an internal interrupt request. An interrupt acknowledge sequence is indicated by the function codes.

SYSTEM CONTROL

The system control inputs are used to either reset or halt the processor and to indicate to the processor that bus errors have occurred. The three system control signals are explained in the following paragraphs.

Bus Error ($\overline{BERR}$)

This input informs the processor that there is a problem with the cycle currently being executed. Problems may be a result of:

1) nonresponting devices,
2) interrupt vector number acquisition failure,
3) illegal access request as determined by a memory management unit, or
4) various other application dependent errors.

The bus error signal interacts with the halt signal to determine if the current bus cycles should be re-executed or if exception processing should be performed. A summarization of the interaction is as follows:

$\overline{BERR}$	$\overline{HALT}$	Resulting Operation
High	High	Normal Operation
High	Low	Single Bus Cycle Operation
Low	High	Bus Error – Exception Processing
Low	Low	Bus Error Re-Run Current Cycle

Reset ($\overline{RESET}$)

This bidirectional signal line acts to reset (start a system initialization sequence) the processor in response to an external $\overline{RESET}$ signal. An internally generated reset (result of a reset instruction) causes all external devices to be reset and the internal state of the processor is not affected. A total system reset (processor and external devices) is the result of external $\overline{HALT}$ and $\overline{RESET}$ signals applied at the same time.

Halt ($\overline{HALT}$)

When this bidirectional line is driven by an external device, it will cause the processor to stop at the completion of the current bus cycle. When the processor has been halted using this input, all control signals are inactive and all three-state lines are put in their high-impedence state.

When the processor has stopped executing instructions, such as in a double-bus fault condition, the halt line is driven by the processor to indicate to external devices that the processor has stopped.

M6800 PERIPHERAL CONTROL

These control signals are used to allow the interfacing of synchronous M6800 peripheral devices with the asynchronous MC68008. These signals are explained in the following paragraphs.

The MC68008 does not supply a valid memory address ($\overline{VMA}$) signal like that of the MC68000. The $\overline{VMA}$ signal indicates to the M6800 peripheral devices that there is a valid addess on the address bus and that the processor is synchronized to the enable clock. This signal can be produced by a TTL circuit (see a sample circuit in Figure 4). The $\overline{VMA}$ signal, in this circuit only responds to a valid peripheral address ($\overline{VPA}$) input which indicates that the peripheral is an M68000 Family device.

The $\overline{VPA}$ decode shown in Figure 4 is an active high decode indicating that address strobe ($\overline{AS}$) has been asserted and the address bus is addressing an M6800 peripheral. The $\overline{VPA}$ output of the circuit is used to indicate to the MC68008 that the data transfer should be synchronized with the enable (E) signal.

Enable (E)

This signal is the standard enable signal common to all M6800 type peripheral devices. The period for this output is ten MC68008 clock periods (six clocks low, four clocks high).

Valid Peripheral Address ($\overline{VPA}$)

This input indicates that the device or region addressed is an M6800 Family device and that data transfer should be synchronized with the enable (E) signal. This input also indicates that the processor should use automatic vectoring for an interrupt.

PROCESSOR STATES (FC0, FC1, and FC2)

These function code outputs indicate the state (user or supervisor) and the cycle type currently being executed, as shown in Table 5. The information indicated by the function code output is valid whenever address strobe ($\overline{AS}$) is active.

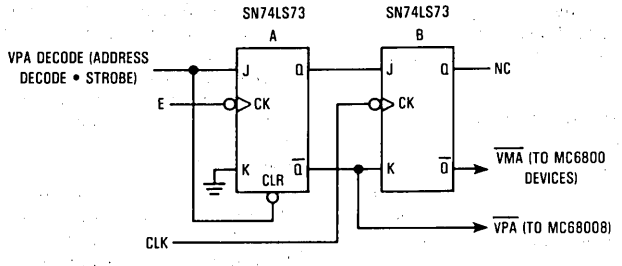


Figure 4. External $\overline{VMA}$ Generation

Table 5. Function Code Outputs

Function Code Output			Cycle Type
FC2	FC1	FC0	
Low	Low	Low	(Undefined, Reserved)
Low	Low	High	User Data
Low	High	Low	User Program
Low	High	High	(Undefined, Reserved)
High	Low	Low	(Undefined, Reserved)
High	Low	High	Supervisor Data
High	High	Low	Supervisor Program
High	High	High	Interrupt Acknowledge

CLOCK (CLK)

The clock input is a TTL-compatible signal that is internally buffered for development of the internal clocks needed by the processor. The clock input shall be a constant frequency.

V_{CC} AND GND

Power is supplied to the processor using these two signals. V_{CC} is power and GND is the ground connection.

SIGNAL SUMMARY

Table 6 is a summary of all the signals discussed in the previous paragraphs.

DATA TRANSFER OPERATIONS

Transfer of data between devices involves the following leads:

1) address bus A0 through A21,
2) data bus D0 through D7, and
3) control signals.

The address and data buses are separated non-multiplexed parallel buses. Data transfer is accomplished with an asynchrounous bus structure that uses handshaes to ensure the corect movement of data. In all cycles, the bus master assumes responsibility for deskewing all signals it issues at both the start and end of a cycle. In addition, the bus master is responsible for deskewing the acknowledge and data signals from the slave device.

The following paragraphs explain the read, write, and read-modify-write cycles. The indivisible read-modify-write cycle is the method used by the MC68008 for interlocked multiprocessor communications.

READ CYCLE

During a read cycle, the processor receives data from the memory or a pripheral device. The processor reads bytes of data in all cases. If the instruction specifies a word (or double word) operation, the processor reads both bytes. When the instruction specifies byte operation, the processor uses A0 to determine which byte to read and then issues data strobe.

WRITE CYCLE

During a write cycle, the processor sends data to either the memory or a peripheral device. The processor writes bytes of data in all cases. If the instruction specifies a word operation, the processor writes both bytes. When the instruction specifies a byte operation, the processor uses A0 to detrmine which byte to write and then issues the data strobe.

READ-MODIFY-WRITE CYCLE

The read-modify-write cycle performs a byte read, modifies the data in the arithmetic-logic unit, and writes the data back to the same address. In the MC68008, this cycle is indivisible in that the address strobe is asserted throughout the entire cycle. The test and set (TAS) instruction uses the cycle to provide meaningful communication between processors in a multiple processor environment. This instruction is the only instruction that uses the read-modify-write cycle and since the test and set instruction only operates on bytes, all read-modify-write cycles are byte operations.

Table 6. Signal Summary

Signal Name	Mnemonic	Input/Output	Active State	Hi-Z	
				on $\overline{HALT}$	on $\overline{BGACK}$
Address Bus	A0-A19 (A21)	Output	High	Yes	Yes
Data Bus	D0-D7	Input/Output	High	Yes	Yes
Address Strobe	$\overline{AS}$	Output	Low	No	Yes
Read/Write	R/$\overline{W}$	Output	Read – High Write – Low	No No	Yes Yes
Data Strobe	$\overline{DS}$	Output	Low	No	Yes
Data Transfer Acknowledge	$\overline{DTACK}$	Input	Low	No	No
Bus Request	$\overline{BR}$	Input	Low	No	No
Bus Grant	$\overline{BG}$	Output	Low	No	No
Bus Grant Acknowledge**	$\overline{BGACK}$	Input	Low	No	No
Interrupt Priority Level	$\overline{IPLx}$	Input	Low	No	No
Bus Error	$\overline{BERR}$	Input	Low	No	No
Reset	$\overline{RESET}$	Input/Output	Low	No*	No*
Halt	$\overline{HALT}$	Input/Output	Low	No*	No*
Enable	E	Output	High	No	No
Valid Peripheral Address	$\overline{VPA}$	Input	Low	No	No
Function Code Output	FC0, FC1, FC2	Output	High	No	Yes
Clock	CLK	Input	High	No	No
Power Input	V_{CC}	Input	—	—	—
Ground	GND	Input	—	—	—

*Open Drain
**52-Pin Version Only

PROCESSING STATES

The MC68008 is always in one of three processing states: normal, exception, or halted.

NORMAL PROCESSING

The normal processing state is that associated with instruction execution; the memory references are to fetch instructions and operands, and to store results. A special case of the normal state is the stopped state which the processor enters when a STOP instruction is executed. In this state, no further memory references are made.

EXCEPTION PROCESSING

The exception processing state is associated with interupts, trap instructions, tracing, and other exceptional conditions,. The exception may be internally generated by an instruction or by an unusual condidion arising during the execution of an instruction. Externally, exception processing can be forced by an interrupt, by a bus error, or by a reset. Exception processing is designed to provide an efficient context switch so that the processor may handle unusual conditions.

HALTED PROCESSING

The halted processing state is an indication of catastrophic hardware failure. For example, if during the exception processing of a bus error another bus error occurs, the processor assumes that the system is unusable and halts. Only an external reset can restart a halted processor. Note that a processor in the stopped state is not in the halted state, nor vise versa.

INTERFACE WITH M6800 PERIPHERALS

Motorola's extensive line of M6800 peripherals are compatible with the MC68008. Some of these devices which are particularly useful are:

MC6821 Peripheral Interface Adapter
MC6840 Programmable Timer Module
MC6845 CRT Controller
MC6850 Asynchronous Communications Interface Adapter
MC6852 Synchronous Serial Data Adapter
MC6854 Advanced Data Link Controller

To interface the synchronous M6800 peripherals with the asynchronous MC68008, the processor modifies its bus cycle to meet the M6800 cycle requirements whenever an M6800 device address is detected. This is possible since both processors use memory mapped I/O.

ELECTRICAL SPECIFICATIONS

3

MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Supply Voltage	V_{CC}	−0.3 to +7.0	V
Input Voltage	V_{in}	−0.3 to +7.0	V
Operating Temperature Range MC68008 MC68008C	T_A	T_L to T_H 0 to 70 −40 to 85	°C
Storage Temperature	T_{stg}	−55 to 150	°C

The device contains circuitry to protect the inputs against damage due to high static voltages or electric fields; however, normal precautions should be taken to avoid application of voltages higher than maximum-rated voltages to these high-impedance circuits. Tying unused inputs to the appropriate logic voltage level (e.g., either GND or V_{CC}) enhances reliability of operation.

THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Symbol	Value	Rating
Thermal Resistance (Still Air) Ceramic, Type LC Plastic, Type P Plastic, Type FN	θ_{JA}	 40 40 50	θ_{JC}	 15* 20* 30*	°C/W

*Estimated

DC ELECTRICAL CHARACTERISTICS (V_{CC} = 5.0 Vdc ± 5%; GND = 0Vdc; $T_A = T_L$ to T_H; see Figures 5, 6, and 7)

Characteristic		Symbol	Min	Max	Unit
Input High Voltage		V_{IH}	2.0	V_{CC}	V
Input Low Voltage		V_{IL}	GND −0.3	0.8	V
Input Leakage Current @ 5.25 V	$\overline{BERR}$, $\overline{BGACK}$, $\overline{BR}$, $\overline{DTACK}$, CLK, $\overline{IPL0}$/$\overline{IPL2}$, $\overline{VPA}$ $\overline{HALT}$, $\overline{RESET}$	I_{IN}	— —	2.5 20	μA
Three-State (Off State) Input Current @ 2.4 V/0.4 V	$\overline{AS}$, A0-A18, A20, A21, D0-D7, FC0-FC2, $\overline{DS}$, R/$\overline{W}$, $\overline{VMA}$	I_{TSI}	—	20	μA
Output High Voltage (I_{OH} = −400 μA) (I_{OH} = −400 μA)	E* E, $\overline{AS}$, A0-A19, A20, A21, $\overline{BG}$, D0-D7, FC0-FC2, $\overline{DS}$, R/$\overline{W}$, $\overline{VMA}$	V_{OH}	V_{CC} − 0.75 2.4	— 2.4	V
Output Low Voltage (I_{OL} = 1.6 mA) (I_{OL} = 3.2 mA) (I_{OL} = 5.0 mA) (I_{OL} = 5.3 mA)	 $\overline{HALT}$ A0-A19, A20, A21, FC0-FC2 $\overline{RESET}$ E, $\overline{AS}$, D0-D7, $\overline{DS}$, R/$\overline{W}$, $\overline{VMA}$	V_{OL}	 — — — —	 0.5 0.5 0.5 0.5	V
Power Dissipation (see **POWER CONSIDERATIONS**)		P_D***	—	—	W
Capacitance (V_{in} = 0 V, T_A = 25°C, Frequency = 1 MHz)**		C_{in}	—	20.0	pF
Load Capacitance	$\overline{HALT}$ All Others	C_L	— —	70 130	pF

*With external pullup resistor of 1.1 Ω.
**Capacitance is periodically sampled rather than 100% tested.
***During normal operation instantaneous V_{CC} current requirements may be as high as 1.5 A.

POWER CONSIDERATIONS

The average die-junction temperature, T_J, in °C can be obtained from:

$$T_J = T_A + (P_D \cdot \theta_{JA}) \quad (1)$$

where:

T_A = Ambient Temperature, °C
θ_{JA} = Package Thermal Resistance, Junction-to-Ambient, °C/W
P_D = $P_{INT} + P_{I/O}$
P_{INT} = $I_{CC} \times V_{CC}$, 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 apropriate relationship between $P_D$and T_J (if $P_{I/O}$ is neglected) is:

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

Solving equations (1) and (2) for K gives:

$$K = P_D \cdot (T_A + 273°C) + \theta_{JA} \cdot P_D^2 \quad (3)$$

where K is a constant pertaining to the particular part. K can be determined from equation (3) by measuring P_D (at thermal 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.

The curve shown in Figure 5 gives the graphic solution to the above equations for the specified power dissipation of 1.5 watts over the ambient temperature range of −55 °C to 125 °C using a maximum θ_{JA} of 45 °C/W. Ambient temperature is that of the still air surrounding the device. Lower values of θ_{JA} cause the curve to shift downward slightly; for instance, for θ_{JA} of 40 °/W, the curve is just below 1.4 watts at 25 °C.

The total thermal resistance of a package (θ_{JA}) can be separated into two components, θ_{JC} and θ_{CA}, representing the barrier to heat flow from the semiconductor junction to the package (case) surface (θ_{JC}) and from the case to the outside ambient air (θ_{CA}). These terms are related by the equation:

$$\theta_{JA} = \theta_{JC} + \theta_{CA} \quad (4)$$

θ_{JC} is device related and cannot be influenced by the user. However, θ_{CA} is user dependent and can be minimized by such thermal management techniques as heat sinks, ambient air cooling, and thermal convection. Thus, good thermal management on the part of the user can significantly reduce θ_{CA} so that θ_{JA} approximately equals θ_{JC}. Substitution of θ_{JC} for θ_{JA} in equation 1 results in a lower semiconductor junction temperature.

Table 7 summarizes maximum power dissipation and average junction temperature for the curve drawn in Figure 5, using the minimum and maximum values of ambient temperature for different packages and substituting θ_{JC} for θ_{JA} (assuming good thermal management). Table 8 provides the maximum power dissipation and average junction temperature for the MC68000 assuming that no thermal management is applied (i.e., still air).

NOTE

Since the power dissipation curve shown in Figure 5 is negatively sloped, power dissipation declines as ambient temperature increases. Therefore, maximum power dissipation occurs at the lowest rated ambient temperature where *power dissipation is lowest.*

Values for thermal resistance presented in this manual, unless estimated, were derived using the procedure described in Motorola Reliability Report 7843, *Thermal Resistance Measurement Method for MC68XXX Microcomponent Devices*, and are provided for design purposes only. Thermal measurements are complex and dependent on procedure and setup. User-derived values for thermal resistance may differ.

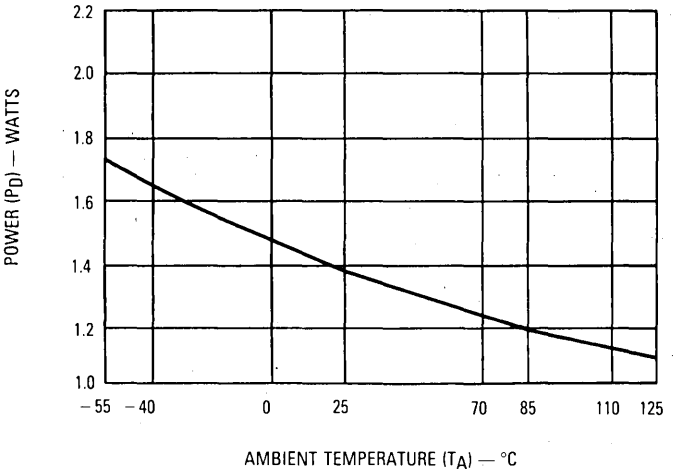


Figure 5. MC68008 Power Dissipation (P_D) vs Ambient Temperature(T_A)

Table 7. MC68008 Power Dissipation and Junction Temperature vs Temperature ($\theta_{JC}=\theta_{JA}$)

Package	T_A Range	θ_{JC} (°C/W)	P_D (W) @ T_A Min.	T_J (°C) @ T_A Min.	P_D (W) @ T_A Max.	T_J (°C) @ T_A Max.
LC	0°C to 70°C	15	1.5	23	1.2	88
	−40°C to 85°C	15	1.7	−14	1.2	103
	0°C to 85°C	15	1.5	23	1.2	103
P	0°C to 70°C	20	1.5	30	1.2	95
FN	0°C to 70°C	30	1.5	45	1.3	108

3

Table 8. MC68008 Power Dissipation and Junction Temperature vs Temperature ($\theta_{JC} \leq \theta_{JA}$)

Package	T_A Range	θ_{JA} (°C/W)	P_D (W) @ T_A Min.	T_J (°C) @ T_A Min.	P_D (W) @ T_A Max.	T_J (°C) @ T_A Max.
LC	0°C to 70°C	40	1.5	60	1.2	121
	−40°C to 85°C	40	1.7	−27	1.2	134
	0°C to 85°C	40	1.5	60	1.2	134
P	0°C to 70°C	40	1.5	60	1.2	121
FN	0°C to 70°C	50	1.5	75	1.3	134

AC ELECTRICAL SPECIFICATIONS — CLOCK TIMING (see Figure 6)

Num.	Characteristic	Symbol	8 MHz*		10 MHz*		Unit
			Min	Max	Min	Max	
	Frequency of Operation	f	2.0	8.0	2.0	10.0	MHz
1	Clock Period	t_{cyc}	125	500	100	500	ns
2,3	Clock Pulse Width	t_{CL}, t_{CH}	55	250	45	250	ns
4,5	Clock Rise and Fall Times	t_{Cr}, t_{Cf}	—	10	—	10	ns

*These specifications represent an improvement over previously published specifications for the 8-, 10-, and 12.5-MHz MC68008 and are valid only for product bearing date codes of 8827 and later.

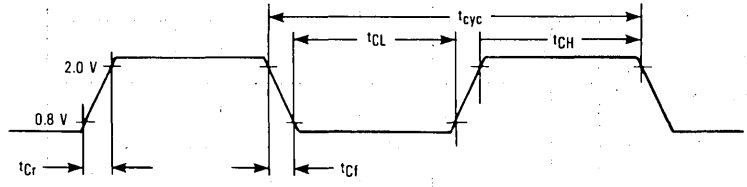


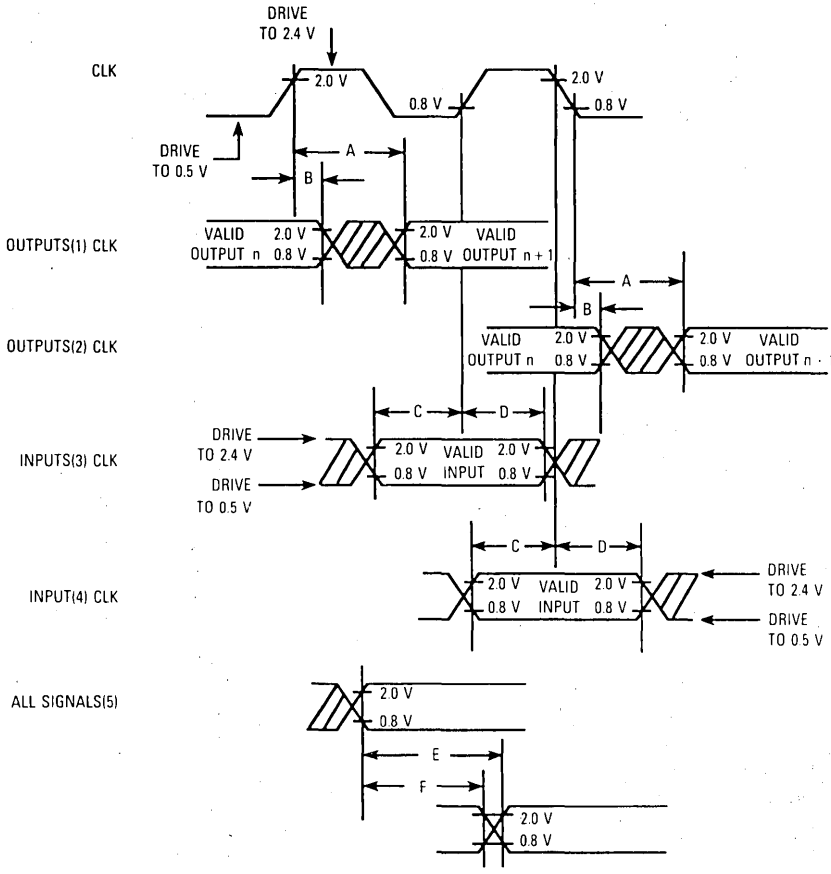
Figure 6. MC68008 Clock Input Timing Diagram

AC ELECTRICAL SPECIFICATION DEFINITIONS

The AC specifications presented consist of output delays, input setup and hold times, and signal skew times. All signals are specified relative to an appropriate edge of the clock and possibly to one or more other signals.

The measurement of the AC specifications is defined by the waveforms shown in Figure 13-3. In order to test the parameters guaranteed by Motorola, inputs must be driven to the voltage levels specified in this figure. Outputs are specified with minimum and/or maximum limits, as appropriate, and are measured as shown in Figure 13-3. Inputs are specified with minimum setup and hold times, and are measured as shown. Finally, the measurement for signal-to-signal specifications are also shown.

Note that the testing levels used to verify conformance to the AC specifications does not affect the guaranteed DC operation of the device as specified in the DC electrical characteristics.



NOTES:

1. This output timing is applicable to all parameters specified relative to the rising edge of the clock.
2. This output timing is applicable to all parameters specified relative to the falling edge of the clock.
3. This input timing is applicable to all parameters specified relative to the rising edge of the clock.
4. This input timing is applicable to all parameters specified relative to the falling edge of the clock.
5. This timing is applicable to all parameters specified relative to the assertion negation of another signal.

LEGEND:

A. Maximum output delay specification.
B. Minimum output hold time.
C. Minimum input setup time specification.
D. Minimum input hold time specification.
E. Signal valid to signal valid specification (maximum or minimum).
F. Signal valid to signal invalid specification (maximum or minimum).

Figure 7. Drive Levels and Test Points for AC Specifications

AC ELECTRICAL SPECIFICATIONS — READ AND WRITE CYCLES (V_{CC} = 5.0 Vdc ± 5%; GND = 0 Vdc; $T_A = T_L$ to T_H; see Figures 8 and 9)

Num.	Characteristic	Symbol	8 MHz*		10 MHz*		Unit
			Min	Max	Min	Max	
6	Clock Low to Address Valid	t_{CLAV}	—	62	—	50	ns
6A	Clock High to FC Valid	t_{CHFCV}	—	62	—	50	ns
7	Clock High to Address, Data Bus High Impedance (Maximum)	t_{CHADZ}	—	80	—	70	ns
8	Clock High to Address, FC Invalid (Minimum)	t_{CHAFI}	0	—	0	—	ns
9[1]	Clock High to $\overline{AS}$, $\overline{DS}$ Asserted	t_{CHSL}	3	60	3	50	ns
11[2]	Address Valid to $\overline{AS}$, $\overline{DS}$ Asserted (Read)/$\overline{AS}$ Asserted (Write)	t_{AVSL}	30	—	20	—	ns
11A[2]	FC Valid to $\overline{AS}$, $\overline{DS}$ Asserted (Read)/$\overline{AS}$ Asserted (Write)	t_{FCVSL}	90	—	70	—	ns
12[1]	Clock Low to $\overline{AS}$, $\overline{DS}$ Negated	t_{CLSH}	—	62	—	50	ns
13[2]	$\overline{AS}$, $\overline{DS}$ Negated to Address, FC Invalid	t_{SHAFI}	40	—	30	—	ns
14[2]	$\overline{AS}$ (and $\overline{DS}$ Read) Width Asserted	t_{SL}	270	—	195	—	ns
14A[2]	$\overline{DS}$ Width Asserted (Write)	t_{DSL}	140	—	95	—	ns
15[2]	$\overline{AS}$, $\overline{DS}$ Width Negated	t_{SH}	150	—	105	—	ns
16	Clock High to Control Bus High Impedance	t_{CHCZ}	—	80	—	70	ns
17[2]	$\overline{AS}$, $\overline{DS}$ Negated to R/$\overline{W}$ Invalid	t_{SHRH}	40	—	30	—	ns
18[1]	Clock High to R/$\overline{W}$ High (Read)	t_{CHRH}	0	55	0	45	ns
20[1]	Clock High to R/$\overline{W}$ Low (Write)	t_{CHRL}	0	55	0	45	ns
20A[2, 6]	$\overline{AS}$ Asserted to R/$\overline{W}$ Valid (Write)	t_{ASRV}	—	10	—	10	ns
21[2]	Address Valid to R/$\overline{W}$ Low (Write)	t_{AVRL}	20	—	0	—	ns
21A[2]	FC Valid to R/$\overline{W}$ Low (Write)	t_{FCVRL}	60	—	50	—	ns
22[2]	R/$\overline{W}$Low to $\overline{DS}$ Asserted (Write)	t_{RLSL}	80	—	50	—	ns
23	Clock Low to Data-Out Valid (Write)	t_{CLDO}	—	62	—	50	ns
25[2]	$\overline{AS}$, $\overline{DS}$ Negated to Data-Out Invalid (Write)	t_{SHDOI}	50	—	30	—	ns
26[2]	Data-Out Valid to $\overline{DS}$ Asserted (Write)	t_{DOSL}	40	—	30	—	ns
27[5]	Data-In Valid to Clock Low (Setup Time of Read)	t_{DICL}	10	—	10	—	ns
28[2]	$\overline{AS}$, $\overline{DS}$ Negated to $\overline{DTACK}$ Negated (Asynchronous Hold)	t_{SHDAH}	0	245	0	190	ns
29	$\overline{AS}$, $\overline{DS}$ Negated to Data-In Invalid (Hold Time on Read)	t_{SHDII}	0	—	0	—	ns
29A	$\overline{AS}$, $\overline{DS}$ Negated to Data In High Impedance	t_{SHDZ}	—	187	—	150	ns
30	$\overline{AS}$, $\overline{DS}$ Negated to $\overline{BERR}$ Negated	t_{SHBEH}	0	—	0	—	ns
31[2, 5]	$\overline{DTACK}$ Asserted to Data In Valid (Setup Time)	t_{DALDI}	—	90	—	65	ns
32	$\overline{HALT}$ and $\overline{RESET}$ Input Transition Time	$t_{RHr,f}$	0	200	0	200	ns
33	Clock High to $\overline{BG}$ Asserted	t_{CHGL}	—	62	—	50	ns
34	Clock High to $\overline{BG}$ Negated	t_{CHGH}	—	62	—	50	ns
35	$\overline{BR}$ Asserted to $\overline{BG}$ Asserted	t_{BRLGL}	1.5	3.5	1.5	3.5	Clks
36[7]	$\overline{BR}$ Negated to $\overline{BG}$ Negated	t_{BRHGH}	1.5	3.5	1.5	3.5	Clks
37	$\overline{BGACK}$ Asserted to $\overline{BG}$ Negated (52-Pin Version Only)	t_{GALGH}	1.5	3.5	1.5	3.5	Clks
37A[8]	$\overline{BGACK}$ Asserted to $\overline{BR}$ Negated (52-Pin Version Only)	t_{GALBRH}	20	1.5 Clks	20	1.5 Clks	ns
38	$\overline{BG}$ Asserted to Control, Address, Data Bus High Impedance (AS Negated)	t_{GLZ}	—	80	—	70	ns
39	$\overline{BG}$ Width Negated	t_{GH}	1.5	—	1.5	—	Clks
41	Clock Low to E Transition	t_{CLET}	—	50	—	45	ns
42	E Output Rise and Fall Time	$t_{Er,f}$	—	15	—	15	ns
44	$\overline{AS}$, $\overline{DS}$ Negated to $\overline{VPA}$ Negated	t_{SHVPH}	0	120	0	90	ns
45	E Low to Control, Address Bus Invalid (Address Hold Time)	t_{ELCAI}	30	—	10	—	ns

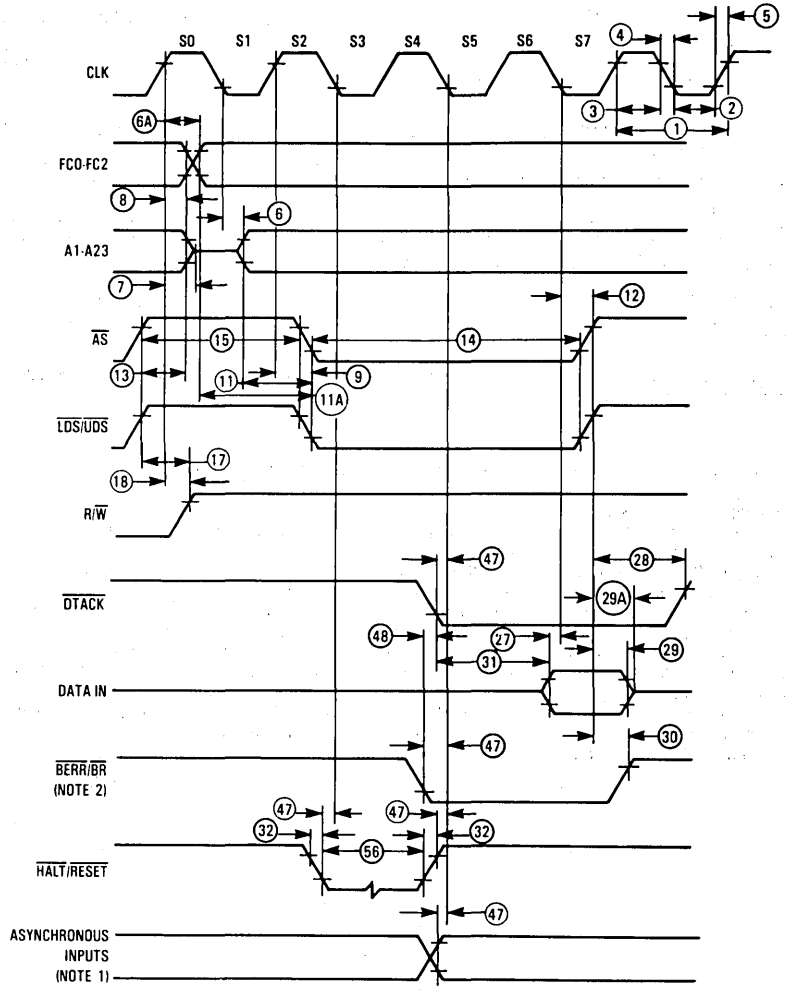
AC ELECTRICAL SPECIFICATIONS — READ AND WRITE CYCLES (Continued)

Num.	Characteristic	Symbol	8 MHz*		10 MHz*		Unit
			Min	Max	Min	Max	
46	$\overline{BGACK}$ Width Low (52-Pin Version Only)	t_{GAL}	1.5	—	1.5	—	Clks
47[5]	Asynchronous Input Setup Time	t_{ASI}	10	—	10	—	ns
48[2, 3]	$\overline{DTACK}$ Asserted to $\overline{BERR}$ Asserted	t_{BELDAL}	20	—	20	—	ns
49[9]	$\overline{AS}$, $\overline{DS}$, Negated to E Low	t_{SHEL}	−70	70	−55	55	ns
50	E Width High	t_{EH}	450	—	350	—	ns
51	E Width Low	t_{EL}	700	—	550	—	ns
53	Data-Out Hold from Clock High	t_{CHDOI}	0	—	0	—	ns
54	E Low to Data-Out Invalid	t_{ELDOI}	30	—	20	—	ns
55	R/$\overline{W}$ Asserted to Data Bus Impedance Change	t_{RLDBD}	30	—	20	—	ns
56[4]	$\overline{HALT}$/$\overline{RESET}$ Pulse Width	t_{HRPW}	10	—	10	—	Clks
57	$\overline{BGACK}$ Negated to $\overline{AS}$, $\overline{DS}$, R/$\overline{W}$ Driven (52-Pin Version Only)	t_{GASD}	1.5	—	1.5	—	Clks
57A	$\overline{BGACK}$ Negated to FC, $\overline{VMA}$ Driven (52-Pin Version Only)	t_{GAFD}	1	—	1	—	Clks
58[7]	$\overline{BR}$ Negated to $\overline{AS}$, $\overline{DS}$, R/$\overline{W}$ Driven	t_{RHSD}	1.5	—	1.5	—	Clks
58A[7]	$\overline{BR}$ Negated to FC, $\overline{VMA}$ Driven	t_{RHFD}	1	—	1	—	Clks

*These specifications represent an improvement over previously published specifications for the 8-, 10-, and 12.5-MHz MC68008 and are valid only for product bearing date codes of 8827 and later.

NOTES:
1. For a loading capacitance of less than or equal to 50 picofarads, subtract 5 nanoseconds from the value given in the maximum columns.
2. Actual value depends on clock period.
3. If #47 is satisfied for both $\overline{DTACK}$ and $\overline{BERR}$, #48 may be ignored. In the absence of $\overline{DTACK}$, $\overline{BERR}$ is an asynchronous input using the asynchronous input setup time (#47).
4. For power-up, the MC68000 must be held in the $\overline{RESET}$ state for 100 milliseconds to allow stabilization of on-chip circuitry. After the system is powered up, #56 refers to the minimum pulse width required to reset the processor.
5. If the asynchronous input setup time (#47) requirement is satisfied for $\overline{DTACK}$, the $\overline{DTACK}$-asserted to data setup time (#31) requirement can be ignored. The data must only satisfy the data-in to clock low setup time (#27) for the following clock cycle.
6. When $\overline{AS}$ and R/$\overline{W}$ are equally loaded (±20%), subtract 5 nanoseconds from the values given in these columns.
7. The processor will negate $\overline{BG}$ and begin driving the bus again if external arbitration logic negates $\overline{BR}$ before asserting $\overline{BGACK}$.
8. The minimum value must be met to guarantee power operation. If the maximum value is exceeded, $\overline{BG}$ may be reasserted.
9. The falling edge of S6 triggers both the negation of the strobes ($\overline{AS}$ and x$\overline{DS}$) and the falling edge of E. Either of these events can occur first, depending upon the loading on each signal. Specification #49 indicates the absolute maximum skew that will occur between the rising edge of the strobes and the falling edge of the E clock.

These waveforms should only be referenced to the edge-to-edge measurement of the timing specifications. They are not intended as a functional description of the input and output signals. Refer to other functional descriptions and their related diagrams for device operation.



NOTES:
1. Setup time for the asynchronous inputs IPL0/IPL2, IPL1, and VPA (#47) guarantees their recognition at the next falling edge of the clock.
2. BR need fall at this time only in order to insure being recognized at the end of this bus cycle.
3. Timing measurements are referenced to and from a low-voltage of 0.8 volts and a high voltage of 2.0 volts, unless otherwise noted.

Figure 8. MC68008 Read-Cycle Timing Diagram

These waveforms should only be referenced to the edge-to-edge measurement of the timing specifications. They are not intended as a functional description of the input and output signals. Refer to other functional descriptions and their related diagrams for device operation.

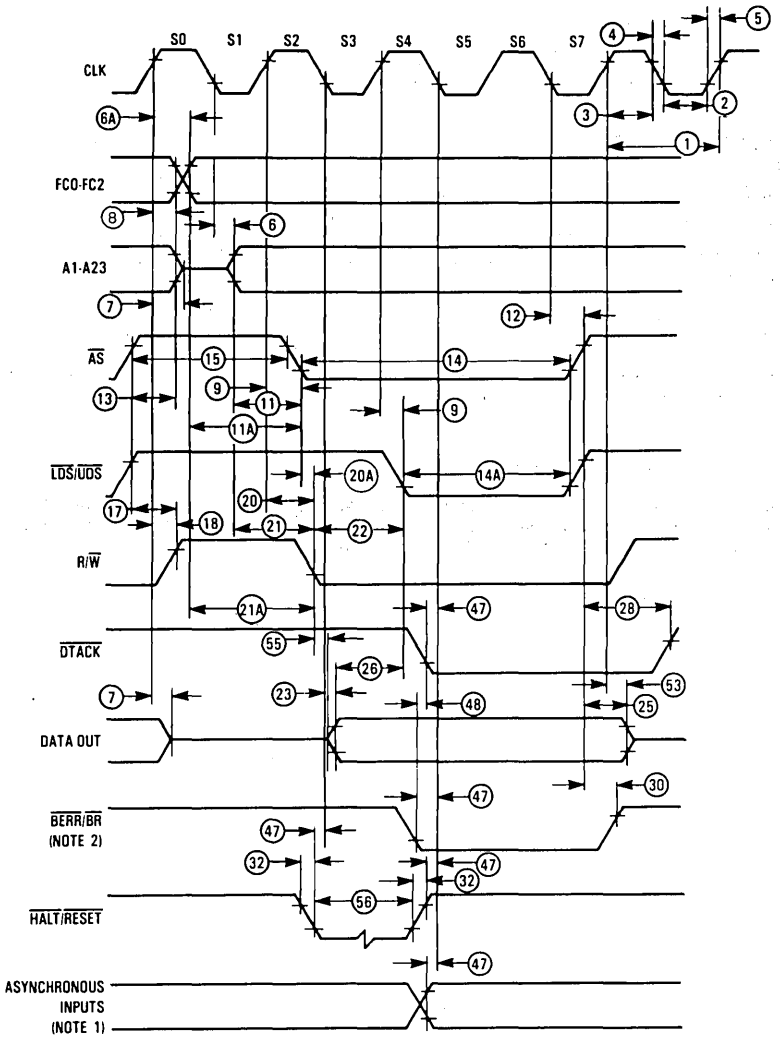


Figure 9. MC68008 Write-Cycle Timing Diagram

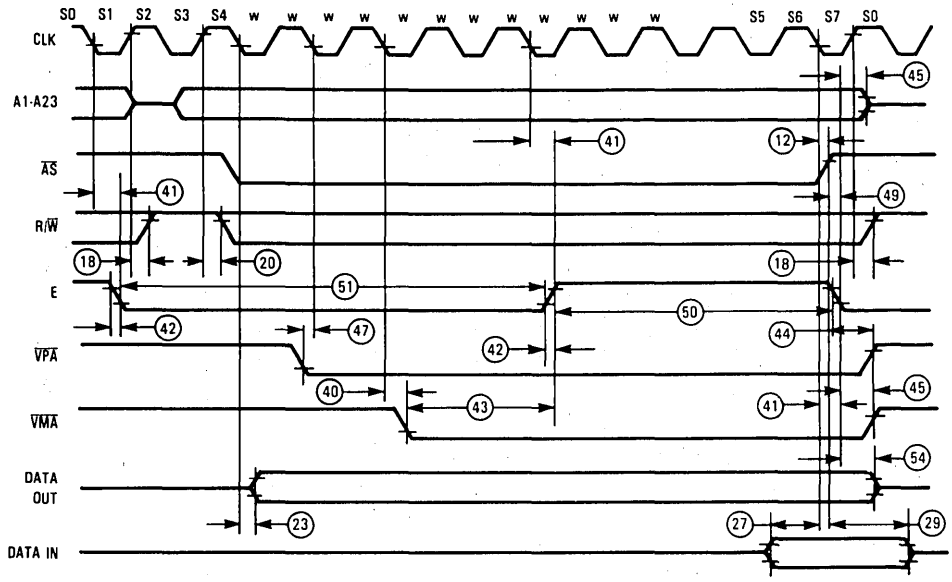
AC ELECTRICAL SPECIFICATIONS — MC68008 TO M6800 PERIPHERAL CYCLES (V_{CC} = 5.0 Vdc ± 5%; GND = 0 Vdc; $T_A = T_L$ to T_H; see Figures 10 and 11)

Num.	Characteristic	Symbol	8 MHz*		10 MHz*		Unit
			Min	Max	Min	Max	
12[1]	Clock Low to $\overline{AS}$, $\overline{DS}$ Negated	t_{CLSH}	—	62	—	50	ns
18[1]	Clock High to R/$\overline{W}$ High (Read)	t_{CHRH}	0	55	0	45	ns
20[1]	Clock High to R/$\overline{W}$ Low (Write)	t_{CHRL}	0	55	0	45	ns
23	Clock Low to Data-Out Valid (Write)	t_{CLDO}	—	62	—	50	ns
27	Data-In Valid to Clock Low (Setup Time of Read)	t_{DICL}	10	—	10	—	ns
29	$\overline{AS}$, $\overline{DS}$ Negated to Data-In Invalid (Hold Time on Read)	t_{SHDII}	0	—	0	—	ns
41	Clock Low to E Transition	t_{CLET}	—	55	—	45	ns
42	E Output Rise and Fall Time	$t_{Er,f}$	—	15	—	15	ns
44	$\overline{AS}$, $\overline{DS}$ Negated to VPA Negated	t_{SHVPH}	0	120	0	90	ns
45	E Low to Control, Address Bus Invalid (Address Hold Time)	t_{ELCAI}	30	—	10	—	ns
47	Asynchronous Input Setup Time	t_{ASI}	10	—	10	—	ns
49[2]	$\overline{AS}$, $\overline{DS}$, Negated to E Low	t_{SHEL}	−70	70	−55	55	ns
50	E Width High	t_{EH}	450	—	350	—	ns
51	E Width Low	t_{EL}	700	—	550	—	ns
54	E Low to Data-Out Invalid	t_{ELDOI}	30	—	20	—	ns

*These specifications represent an improvement over previously published specifications for the 8-, 10-, and 12.5-MHz MC68008 and are valid only for product bearing date codes of 8827 and later.

NOTES:
1. For a loading capacitance of less or equal to 50 picofarads, subtract 5 nanoseconds from the value given in the maximum columns.
2. The falling edge of S6 triggers both the negation of the strobes ($\overline{AS}$ and $\overline{xDS}$) and the falling edge of E. Either of these events can occur first, depending upon the loading on each signal. Specification #49 indicates the absolute maximum skew that will occur between the rising edge of the strobes and the falling edge of the E clock.

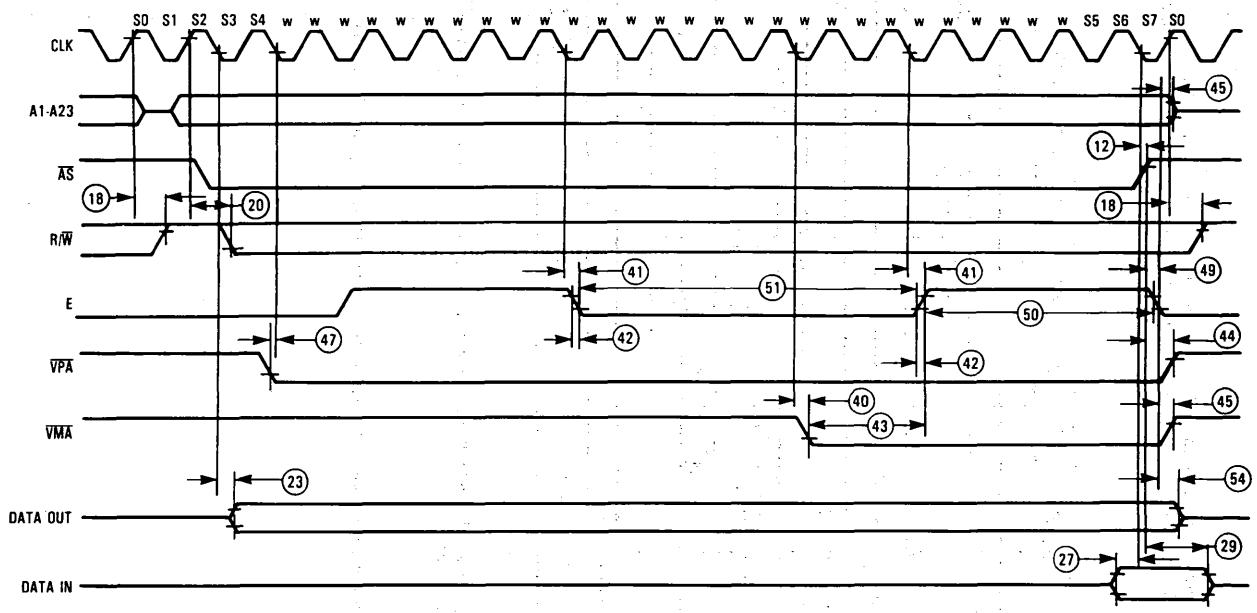
These waveforms should only be referenced to the edge-to-edge measurement of the timing specifications. They are not intended as a functional description of the input and output signals. Refer to other functional descriptions and their related diagrams for device operation.



NOTE: This timing diagram is included for those who wish to design their own circuit to generate $\overline{VMA}$. It shows the best case possibly attainable.

Figure 10. MC68008 to M6800 Peripheral Timing Diagram (Best Case)

These waveforms should only be referenced to the edge-to-edge measurement of the timing specifications. They are not intended as a functional description of the input and output signals. Refer to other functional descriptions and their related diagrams for device operation.



NOTE: This timing diagram is included for those who wish to design their own circuit to generate $\overline{VMA}$. It shows the worst case possibly attainable

Figure 11. MC68008 to M6800 Peripheral Timing Diagram (Worst Case)

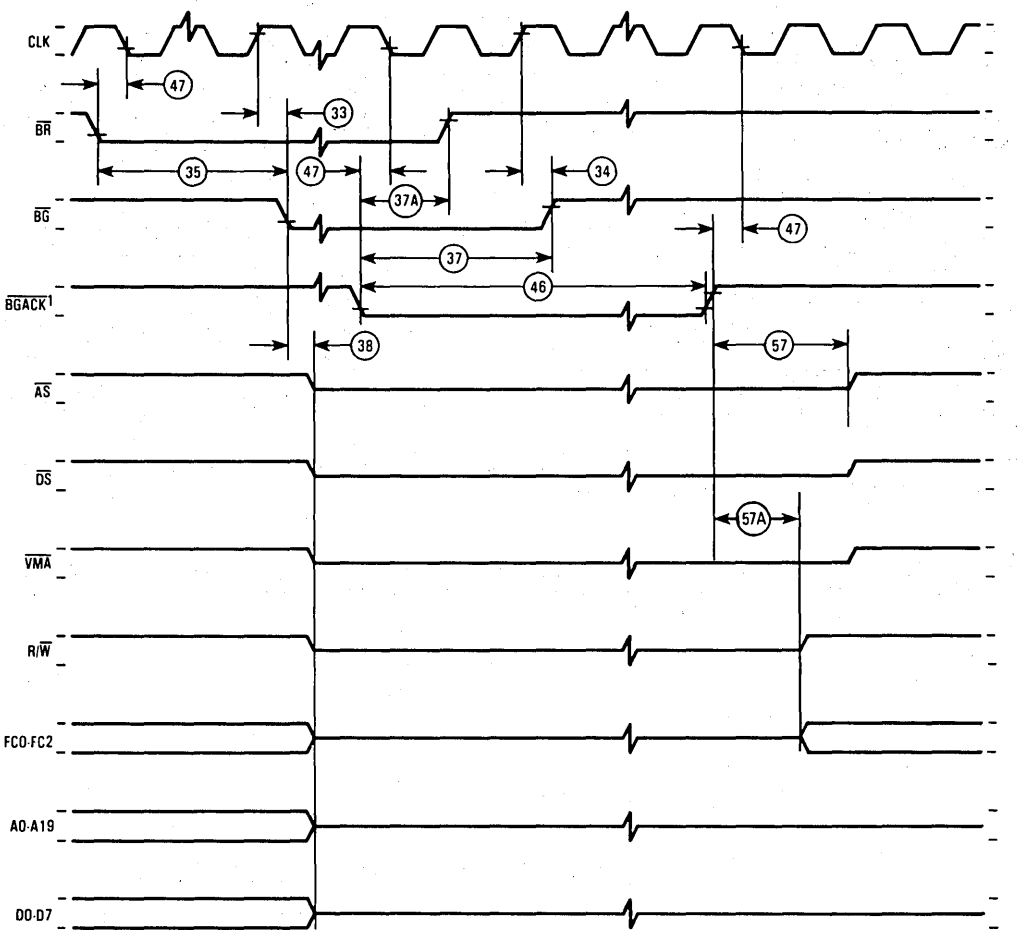
AC ELECTRICAL SPECIFICATIONS — BUS ARBITRATION (V_{CC} = 5.0 Vdc ± 5%; GND = 0 Vdc; $T_A = T_L$ to T_H; see Figures 12, 13, and 14)

Num	Characteristic	Symbol	8 MHz*		10 MHz*		Unit
			Min	Max	Min	Max	
7	Clock High to Address, Data Bus High Impedance (Maximum)	t_{CHADZ}	—	80	—	70	ns
16	Clock High to Control Bus High Impedance	t_{CHCZ}	—	80	—	70	ns
33	Clock High to $\overline{BG}$ Asserted	t_{CHGL}	—	62	—	50	ns
34	Clock High to $\overline{BG}$ Negated	t_{CHGH}	—	62	—	50	ns
35	$\overline{BR}$ Asserted to $\overline{BG}$ Asserted	t_{BRLGL}	1.5	3.5	1.5	3.5	Clks
36[1]	$\overline{BR}$ Negated to $\overline{BG}$ Negated	t_{BRHGH}	1.5	3.5	1.5	3.5	Clks
37	$\overline{BGACK}$ Asserted to $\overline{BG}$ Negated	t_{GALGH}	1.5	3.5	1.5	3.5	Clks
37A[2]	$\overline{BGACK}$ Asserted to $\overline{BR}$ Negated	t_{GALBRH}	20	1.5 Clks	20	1.5 Clks	ns
38	$\overline{BG}$ Asserted to Control, Address, Data Bus High Impedance ($\overline{AS}$ Negated)	t_{GLZ}	—	80	—	70	ns
39	$\overline{BG}$ Width Negated	t_{GH}	1.5	—	1.5	—	Clks
46	$\overline{BGACK}$ Width Low	t_{GAL}	1.5	—	1.5	—	Clks
47	Asynchronous Input Setup Time	t_{ASI}	10	—	10	—	ns
57	$\overline{BGACK}$ Negated to $\overline{AS}$, $\overline{DS}$, R/$\overline{W}$ Driven	t_{GASD}	1.5	—	1.5	—	Clks
57A	$\overline{BGACK}$ Negated to FC, $\overline{VMA}$ Driven	t_{GAFD}	1	—	1	—	Clks
58[1]	$\overline{BR}$ Negated to $\overline{AS}$, $\overline{DS}$, R/$\overline{W}$ Driven	t_{RHSD}	1.5	—	1.5	—	Clks
58A[1]	$\overline{BR}$ Negated to FC, $\overline{VMA}$ Driven	t_{RHFD}	1	—	1	—	Clks

*These specifications represent an improvement over previously published specifications for the 8-, 10-, and 12.5-MHz MC68008 and are valid only for product bearing date codes of 8827 and later.

NOTES:
1. The processor will negate $\overline{BG}$ and begin driving the bus again if external arbitration logic negates $\overline{BR}$ before asserting $\overline{BGACK}$.
2. The minimum value must be met to guarantee proper operation. If the maximum value is exceeded, $\overline{BG}$ may be reasserted.

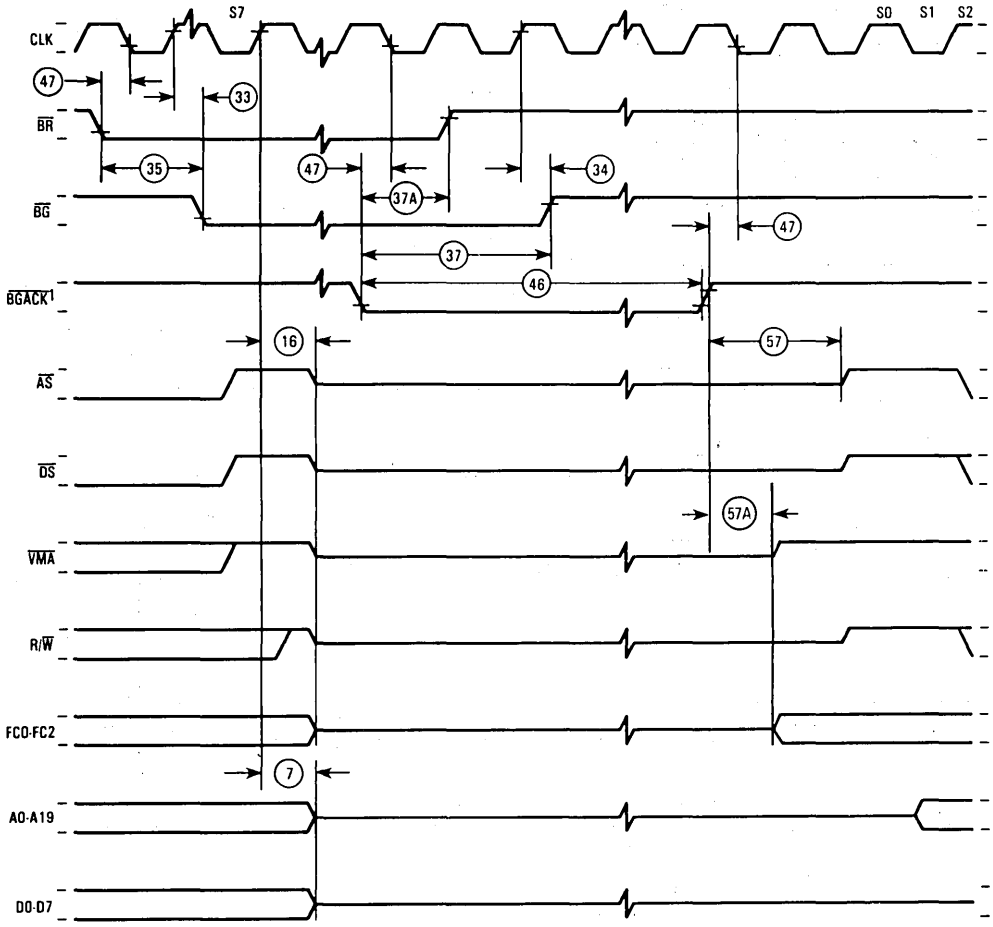
These waveforms should only be referenced to the edge-to-edge measurement of the timing specifications. They are not intended as a functional description of the input and output signals. Refer to other functional descriptions and their related diagrams for device operation.



NOTE:
1. 52-Pin Version of MC68008 Only

Figure 12. MC68008 Bus Arbitration Timing — Idle Bus Case (52-pin Version Only)

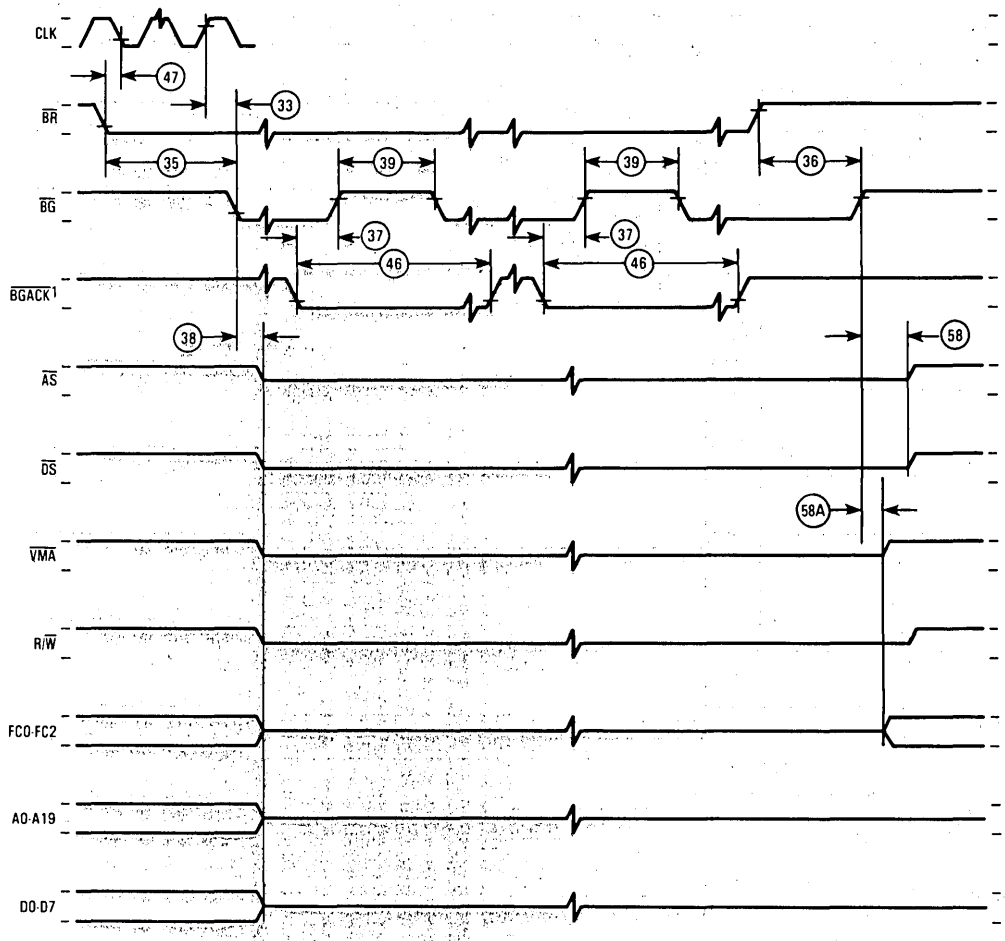
These waveforms should only be referenced to the edge-to-edge measurement of the timing specifications. They are not intended as a functional description of the input and output signals. Refer to other functional descriptions and their related diagrams for device operation.



NOTE:
1. 52-Pin Version of MC68008 Only

Figure 13. MC68008 Bus Arbitration Timing — Active Bus Case (52-pin Version Only)

These waveforms should only be referenced to the edge-to-edge measurement of the timing specifications. They are not intended as a functional description of the input and output signals. Refer to other functional descriptions and their related diagrams for device operation.

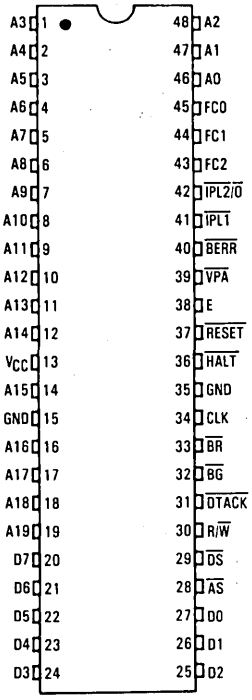


NOTE:
1. 52-Pin Version of MC68008 Only

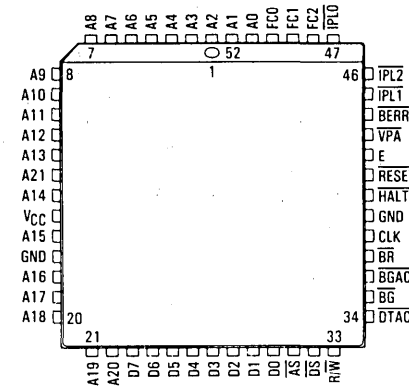
Figure 14. MC68008 Bus Arbitration Timing — Multiple Bus Requests (52-pin Version)

PIN ASSIGNMENTS

48-PIN DUAL-IN-LINE
(Top View)



52-LEAD QUAD PACK
(Top View)



3

MC68010

3

Technical Summary
16-/32-Bit Virtual Memory Microprocessor

This document contains both a summary of the MC68010 as well as a detailed set of parametrics. The purpose is twofold — to provide an introduction to the MC68010 and support for the sophisticated user. For detailed information on the MC68010, refer to the *MC68010/MC68012 16-/32-Bit Virtual Memory Microprocessor Advance Information Data Sheet.*

The MC68010 is a member of the M68000 Family of advanced microprocessors. Utilizing VLSI technology, the MC68010 is a fully-implemented 16-bit microprocessor with 32-bit registers, a rich basic instruction set, and versatile addressing modes.

The resources available to the MC68010 user consist of the following:

- 17 32-Bit Data and Address Registers
- 16 Megabyte Direct Addressing Range
- Virtual Memory/Machine Support
- 57 Powerful Instruction Types
- High-Performance Looping Instructions
- Operations on Five Main Data Types
- Memory Mapped I/O
- 14 Addressing Modes

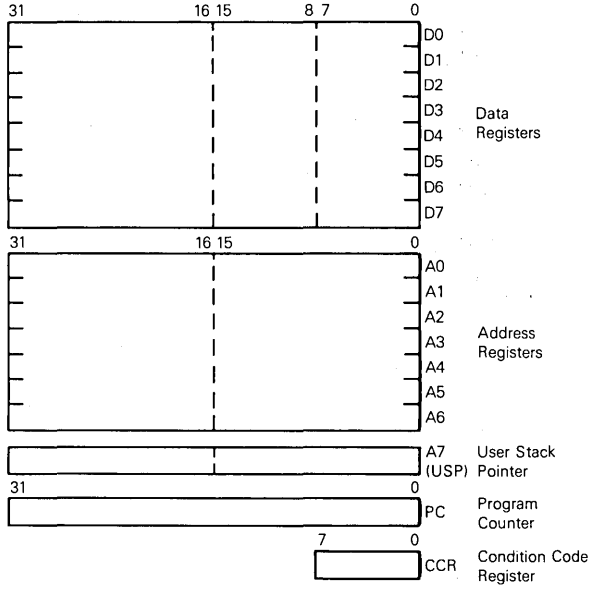


Figure 1. User Programming Model

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

INTRODUCTION

The MC68010 is fully user object code compatible with the earlier members of the M68000 Family and has added features of virtual memory support and enhanced instruction execution timing. The MC68010 is pin-for-pin compatible with the MC68000

The MC68010 possesses an asynchronous bus structure with a 24-bit address bus and a 16-bit data bus.

As shown in the user programming model (Figures 1 and 2), the MC68010 offers 17 32-bit general purpose registers, a 32-bit program counter, a 16-bit status register, a 32-bit vector base register, and two 3-bit alternate function code registers. The first eight registers (D0-D7) are used as data registers for byte (8-bit), word (16-bit), and long word (32-bit) operations. The second set of seven registers (A0-A6) and the stack pointers (SSP and USP) may be used as software stack pointers and base address registers. In addition, the address registers may be used for word and long word operations. All of the 17 registers may be used as index registers.

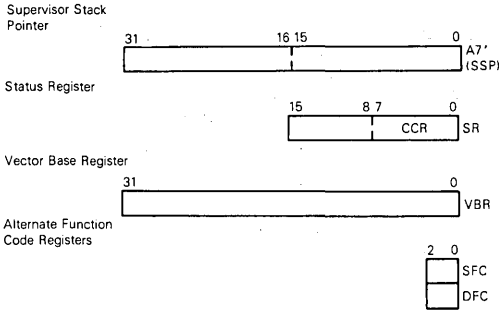


Figure 2. Supervisor Programming Model Supplement

The status register (Figure 3) contains the interrupt mask (eight levels available) as well as the condition codes: extend (X), negative (N), zero (Z), overflow (V), and carry (C). Additional status bits indicate that the processor is in a trace (T) mode and in a supervisor (S) or user state.

The vector base register is used to determine the location of the exception vector table in memory to support multiple vector tables. The alternate function code registers allow the supervisor to access user data space or emulate CPU space cycles.

DATA TYPES AND ADDRESSING MODES

Five basic data types are supported. These data types are:

- Bits
- BCD Digits (4-Bits)
- Bytes (8 Bits)
- Words (16 Bits)
- Long Words (32 Bits)

In addition, operations on other data types such as memory addresses, status word data, etc. are provided in the instruction set.

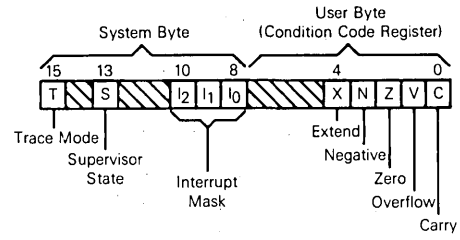


Figure 3. Status Register

Most instructions can use any of the 14 addressing modes which are listed in Table 1. These addressing modes consist of six basic types.

- Register Direct
- Register Indirect
- Absolute
- Program Counter Relative
- Immediate
- Implied

Included in the register indirect addressing modes is a capability to do postincrementing, predecrementing, offsetting, and indexing. The program counter relative mode can also be modified via indexing and offsetting.

INSTRUCTION SET OVERVIEW

The MC68010 instruction set is shown in Table 2. Some additional instructions are variations, or sub-sets, of these and they appear in Table 3. Special emphasis has been given to the instruction set's support of structured high-level languages to facilate ease of programming. Each instruction, with few exceptions, operates on bytes, words, and long words and most instructions can use any of the 14 addressing modes. By combining instruction types, data types, and addressing modes, over 1000 useful instructions are provided. These instructions include signed and unsigned multiply and divide, "quick" arithmetic operations, BCD arithmetic, and expanded operations (through traps). Also, 33 instructions may be used in the loop mode with certain addressing modes and the DBcc instruction to provide 230 high performance string, block manipulation, and extended arithmetic operations.

VIRTUAL MEMORY/MACHINE CONCEPTS

In most systems using the MC68010 as the central processor, only a fraction of the 16-megabyte addressing space will actually contain physical memory. However, by using virtual memory techniques the system can be made to appear to the user to have 16 megabytes of physical memory available to him/her. These techniques have been used for several years in large mainframe

Table 1. Addressing Modes

Addressing Modes	Syntax
Register Direct Addressing	
Data Register Direct	Dn
Address Register Direct	An
Absolute Data Addressing	
Absolute Short	xxx.W
Absolute Long	xxx.L
Program Counter Relative Addressing	
Relative with Offset	d_{16}(PC)
Relative with Index Offset	d_8(PC,Xn)
Register Indirect Addressing	
Register Indirect	(An)
Postincrement Register Indirect	(An)+
Predecrement Register Indirect	–(An)
Register Indirect with Offset	d_{16}(An)
Indexed Register Indirect with Offset	d_8(An,Xn)
Immediate Data Addressing	
Immediate	#xxx
Quick Immediate	#1r#8
Implied Addressing	
Implied Register	SR/USP/SP/PC

NOTES:
- Dn = Data Register
- An = Address Register
- Xn = Address of Data Register used as Index Register
- SR = Status Register
- PC = Program Counter
- SP = Stack Pointer
- USP = User Stack Pointer
- ( ) = Effective Address
- d_8 = 8-Bit Offset (Displacement)
- d_{16} = 16-Bit Offset (Displacement)
- #xxx = Immediate Data

Table 2. Instruction Set Summary

Mnemonic	Description
ABCD*	Add Decimal With Extend
ADD*	Add
AND*	Logical AND
ASL*	Arithmetic Shift Left
ASR*	Arithmetic Shift Right
Bcc	Branch Conditionally
BCHG	Bit Test and Change
BCLR	Bit Test and Clear
BRA	Branch Always
BSET	Bit Test and Set
BSR	Branch to Subroutine
BTST	Bit Test
CHK	Check Register Against Bounds
CLR*	Clear Operand
CMP*	Compare
DBcc	Test Condition, Decrement and Branch
DIVS	Signed Divide
DIVU	Unsigned Divide
EOR*	Exclusive OR
EXG	Exchange Registers
EXT	Signe Extend

Table 2. Instruction Set Summary (Continued)

Mnemonic	Description
JMP	Jump
JSR	Jump to Subroutine
LEA	Lead Effective Address
LINK	Link Stack
LSL*	Logical Shift Left
LSR*	Logical Shift Right
MOVE*	Move
MULS	Signed Multiply
MULU	Unsigned Multiply
NBCD*	Negate Decimal with Extend
NEG*	Negate
NOP	No Operation
NOT*	One's Complement
OR*	Logical OR
PEA	Push Effective Address
RESET	Reset External Devices
ROL*	Rotate Left without Extend
ROR*	Rotate Right without Extend
ROXL*	Rotate Left with Extend
ROXR*	Rotate Right with Extend
RTD	Return and Deallocate
RTE	Return from Exception
RTR	Return and Restore
RTS	Return from Subroutine
SBCD*	Subtract Decimal with Extend
Scc	Set Conditional
STOP	Stop
SUB*	Subtract
SWAP	Swap Data Register Halves
TAS	Test and Set Operand
TRAP	Trap
TRAPV	Trap on Overflow
TST*	Test
UNLK	Unlink

*Loopable Instructions

computers and more recently in minicomputers and now, with the MC68010, can be fully supported in microprocessor-based systems.

In a virtual memory system, a user program can be written as though it has a large amount of memory available to it when only a small amount of memory is physically present in the system. In a similar fashion, a system can be designed in such a manner as to allow user programs to access other types of devices that are not physically present in the system such as tape drives, disk drives, printers, or CRTs. With proper software emulation, a physical system can be made to appear to a user program as any other computer system and the program may be given full access to all of the resources of that emulated system. Such an emulated system is called a virtual machine.

Table 3. Variations of Instruction Types

Instruction Type	Variation	Description
ADD	ADD*	Add
	ADDA*	Add Address
	ADDQ	Add Quick
	ADDI	Add Immediate
	ADDX*	Add with Extend
AND	AND*	Logical AND
	ANDI	AND Immediate
	ANDI to CCR	AND Immediate to Condition Codes
	ANDI to SR	AND Immediate to Status Register
CMP	CMP*	Compare
	CMPA*	Compare Address
	CMPM*	Compare Memory
	CMPI	Compare Immediate
EOR	EOR*	Exclusive OR
	EORI	Exclusive OR Immediate
	EORI to CCR	Exclusive OR Immediate to Condition Codes
	EORI to SR	Exclusive OR Immediate to Status Register
MOVE	MOVE*	Move Source to Destination
	MOVEA*	Move Address
	MOVEC	Move Control Register
	MOVEM	Move Multiple Registers
	MOVEP	Move Peripheral Data
	MOVEQ	Move Quick
	MOVES	Move Alternate Address Space
	MOVE from SR	Move from Status Register
	MOVE to SR	Move to Status Register
	MOVE from CCR	Move from Condition Codes
	MOVE to CCR	Move to Condition Codes
	MOVE USP	Move User Stack Pointer
NEG	NEG*	Negate
	NEGX*	Negate with Extend
OR	OR*	Logical OR
	ORI	OR Immediate
	ORI to CCR	OR Immediate to Condition Codes
	ORI to SR	OR Immediate to Status Register
SUB	SUB*	Subtract
	SUBA*	Subtract Address
	SUBI	Subtract Immediate
	SUBQ	Subtract Quick
	SUBX*	Subtract with Extend

*Loopable Instructions

The MC68010 supports these modes through its instruction continuation mechanism. When an address error or bus error is encountered, the MC68010 will place on the supervisor stack its internal state. The apropriate exception handler is erased and, upon completion, will cause the MC68010 to reload its internal state and resume execution.

Loop mode takes advantage of the fact that the MC68010 can contain three elements of the instruction stream internally. When these elements are 1) a loopable instruction, 2) the DBcc instruction, and 3) a branch displacement to the loopable instruction the MC68010 will enter the "loop mode" where no instruction accesses are made; only data accesses will be performed. This allows extremely fast data transfers as well as providing the bus access made by the MC68010.

SIGNAL DESCRIPTION

The input and output signals are illustrated functionally in Figure 4 and are described in the following paragraphs.

3

ADDRESS BUS (A1 THROUGH A23)

This 23-bit, unidirectional, three-state bus is capable of addressing 8 megabytes of data. It provides the address for bus operation during all cycles except CPU space cycles.

DATA BUS (D0 THROUGH D15)

This 16-bit, bidirectional, three-state bus is the general purpose data path. It can transmit and accept data in either word or byte length.

ASYNCHRONOUS BUS CONTROL

Asynchronous data transfers are handled using the following control signals: address strobe, read write, upper and lower data strobes, and data transfer acknowledge. These signals are explained in the following paragraphs.

Address Strobe ($\overline{AS}$)

This signal indicates that there is a valid address on the address bus.

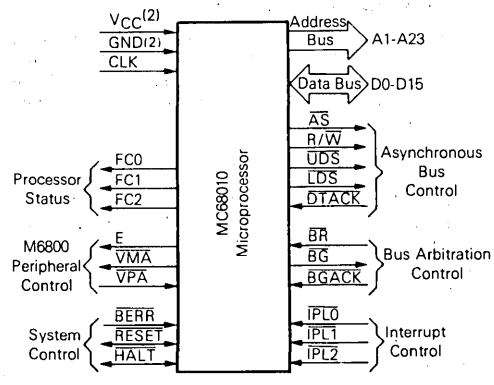


Figure 4. Input and Output Signals

Read/Write (R/$\overline{W}$)

This signal defines the data bus transfer as a read or write cycle. The R/$\overline{W}$ signal also works in conjunction with the data strobe as explained in the following paragraph.

Upper and Lower Data Strobe ($\overline{UDS}$, $\overline{LDS}$)

These signals control the flow of data on the data bus, as shown in Table 4. When the R/$\overline{W}$ line is high, the processor will read from the data bus as indicated. When the R/$\overline{W}$ line is low, the processor will write to the data bus as shown.

Table 4. Data Strobe Control of Data Bus

$\overline{UDS}$	$\overline{LDS}$	R/$\overline{W}$	D8-D15	D0-D7
High	High	—	No Valid Data	No Valid Data
Low	Low	High	Valid Data Bits 8-15	Valid Data Bits 0-7
High	Low	High	No Valid Data	Valid Data Bits 0-7
Low	High	High	Valid Data Bits 8-15	No Valid Data
Low	Low	Low	Valid Data Bits 8-15	Valid Data Bits 0-7
High	Low	Low	Valid Data Bits 0-7*	Valid Data Bits 0-7
Low	High	Low	Valid Data Bits 8-15	Valid Data Bits 8-15*

*These conditions are a result of current implementation and may not appear on future devices.

Read/Write (R/$\overline{W}$)

This signal defines the data bus transfer as a read or write cycle. The R/$\overline{W}$ signal also works in conjunction with the data strobe as explained in the following paragraph.

Upper and Lower Data Strobe ($\overline{UDS}$, $\overline{LDS}$)

These signals control the flow of data on the data bus, as shown in Table 4. When the R/$\overline{W}$ line is high, the processor will read from the data bus as indicated. When the R/$\overline{W}$ line is low, the processor will write to the data bus as shown.

Data Transfer Acknowledge ($\overline{DTACK}$)

This input indicates that the data transfer is completed. When the processor recognizes $\overline{DTACK}$ during a read cycle, data is latched and the bus cycle terminated. When $\overline{DTACK}$ is recognized during a write cycle, the bus cycle is terminated.

BUS ARBITRATION CONTROL

The three signals, bus request, bus grant, and bus grant acknowledge, form a bus arbitration circuit to determine which device will be the bus master device.

Bus Request ($\overline{BR}$)

This input is wire-ORed with all other devices that could be bus masters. This input indicates to the processor that some other device desires to become the bus master.

Bus Grant ($\overline{BG}$)

This output indicates to all other potential bus master devices that the processor will release bus control at the end of the current bus cycle.

Bus Grant Acknoweldge ($\overline{BGACK}$)

This input indicates that some other device has become the bus master. This signal should not be asserted until the following four conditions are met:

1. a bus grant has been received,
2. address strobe is inactive which indicates that the microprocessor is not using the bus,
3. data transfer acknowledge is inactive which indicates that neither memory nor peripherals are using the bus, and
4. bus grant acknowledge is inactive which indicates that no other device is still claiming bus mastership.

INTERRUPT CONTROL ($\overline{IPL0}$, $\overline{IPL1}$, $\overline{IPL2}$)

These input pins indiacte the encoded priority level of the device requesting an interrupt. Level seven is the highest priority while level zero indicates that no interrupts are requested. Level seven cannot be masked. The least significant bit is $\overline{IPL0}$ and the most significant bit is $\overline{IPL2}$. These lines must remain stable until the processor signals interrupt acknowledge (FC0-FC2 are all high, A16-A19 are all high) to insure that the interrupt is recognized.

SYSTEM CONTROL

The system control inputs are used to either reset or halt the processor and to indicate to the processor that bus errors have occurred. The three system control signals are explained in the following paragraphs.

Bus Error ($\overline{BERR}$)

This input informs the processor that there is a problem with the cycle currently being executed. Problems may be a result of:

1. nonresponding devices,
2. interrupt vector number acquisition failure,
3. illegal access request as determined by a memory management unit, or
4. other application dependent errors.

The bus error signal interacts with the halt signal to determine if the current bus cycle should be re-executed or if exception processing should be performed.

Reset ($\overline{RESET}$)

This bidirectional signal line acts to reset (start a system initialization sequence) the processor in response to an external reset signal. An internally generated reset (result of a reset instruction) causes all external devices to be

reset and the internal state of the processor is not affected. A total system reset (processor and external devices) is the result of external $\overline{\text{HALT}}$ and $\overline{\text{RESET}}$ signals applied at the same time.

Halt ($\overline{\text{HALT}}$)

When this bidirectional line is driven by an external device, it will cause the processor to stop at the completion of the current bus cycle. When the processor has been halted using this input, all control signals are inactive and all three-state lines are put in their high-impedance state.

When the processor has stopped executing instructions, such as in a double bus fault condition, the $\overline{\text{HALT}}$ line is driven by the processor to indicate to external devices that the processor has stopped.

M6800 PERIPHERAL CONTROL

These control signals are used to allow the interfacing of synchronous M6800 peripheral devices with the asynchronous MC68010. These signals are explained in the following paragraphs.

Enable (E)

This signal is the standard enable signal common to all M6800 type peripheral devices. The period for this output is ten MC68010 clock periods (six clocks low, four clocks high). Enable is generated by an internal ring counter which may come up in any state (i.e., at power on, it is impossible to guarantee phase relationship of E to CLK). E is a free-running clock and runs regardless of the state of the bus on the MPU.

Valid Peripheral Address ($\overline{\text{VPA}}$)

This input indicates that the device addressed is an M68000 Family device and that data transfer should be synchronized with the enable (E) signal. This input also indicates that the processor should use automatic vectoring for an interrupt.

Valid Memory Address ($\overline{\text{VMA}}$)

This output is used to indicate to M6800 peripheral devices that there is a valid address on the address bus and the processor is synchronized to enable (E). This signal only responds to a valid peripheral address ($\overline{\text{VPA}}$) input which indicates that the peripheral is an M68000 Family device.

PROCESSOR STATUS (FC0, FC1, FC2)

These function code outputs indicate the state (user or supervisor) and the address space of the bus cycle currently being executed, as shown in Table 5. The information indicated by the function code outputs is valid whenever address strobe ($\overline{\text{AS}}$) is active.

CLOCK (CLK)

The clock input is a TTL-compatible signal that is internally buffered for development of the internal clocks needed by the processor. The clock input should not be gated off at any time and the clock signal must conform to minimum and maximum pulse width times.

Table 5. Function Code Assignments

Function Code Output			Address Space
FC2	FC1	FC0	
0	0	0	Undefined, Reserved*
0	0	1	User Data
0	1	0	User Program Space
0	1	1	Undefined, Reserved*
1	0	0	Undefined, Reserved*
1	0	1	Supervisor Data Space
1	1	0	Supervisor Program Space
1	1	1	CPU Space

*Address space 3 is reserved for user definition, while 0 and 4 are reserved for future use by Motorola.

V_{CC} and GND

Power is supplied to the processor using these two signals. V_{CC} is power and GND is the ground connection.

SIGNAL SUMMARY

Table 6 is a summary of all the signals discussed in the previous paragraphs.

DATA TRANSFER OPERATIONS

Transfer of data between devices involves the following leads:

1. address bus A1 through A23,
2. data bus D0 through D15, and
3. control signals.

The address and data buses are separate parallel buses used to transfer data using an asynchronous bus structure. In all cycles, the bus master assumes responsibility for deskewing all signals it issues at both the start and end of a cycle. In addition, the bus master is responsible for deskewing the acknowledge and data signals from the slave device.

The following paragraphs explain the read, write, and read-modify-write cycles. The indivisible read-modify-write cycle is the method used by the MC68010 for interlocked multiprocessor communications.

READ CYCLE

During a read cycle, the processor receives data from the memory of a peripheral device. The processor reads bytes of data in all cases. If the instruction specifies a word (or double word) operation, the processor reads both upper and lower bytes simultaneously by asserting both upper and lower data strobes. When the instruction specifies byte operation, the processor uses an internal A0 bit to determine which byte to read and then issues

Table 6. Signal Summary

Signal Name	Mnemonic	Input/Output	Active State	Hi-Z	
				on $\overline{\text{HALT}}$	ON $\overline{\text{BGACK}}$
Address Bus	A1-A23	Output	High	Yes	Yes
Data Bus	D0-D15	Input/Output	High	Yes	Yes
Address Strobe	$\overline{\text{AS}}$	Output	Low	No	Yes
Read/Write	R/$\overline{\text{W}}$	Output	Read—High Write—Low	No No	Yes Yes
Upper and Lower Data Strobes	$\overline{\text{UDS}}$, $\overline{\text{LDS}}$	Output	Low	No	Yes
Data Transfer Acknowledge	$\overline{\text{DTACK}}$	Input	Low	—	—
Bus Request	$\overline{\text{BR}}$	Input	Low	—	—
Bus Grant	$\overline{\text{BG}}$	Output	Low	No	No
Bus Grant Acknowledge	$\overline{\text{BGACK}}$	Input	Low	—	—
Interrupt Priority Level	$\overline{\text{IPL0}}$, $\overline{\text{IPL1}}$, $\overline{\text{IPL2}}$	Input	Low	—	—
Bus Error	$\overline{\text{BERR}}$	Input	Low	—	—
Reset	$\overline{\text{RESET}}$	Input/Output	Low	No*	No*
Halt	$\overline{\text{HALT}}$	Input/Output	Low	No*	No*
Enable	E	Output	High	No	No
Valid Memory Address	$\overline{\text{VMA}}$	Output	Low	No	Yes
Valid Peripheral Address	$\overline{\text{VPA}}$	Input	Low	—	—
Function Code Output	FC0, FC1, FC2	Output	High	No	Yes
Clock	CLK	Input	High	—	—
Power Input	V_{CC}	Input	—	—	—
Ground	GND	Input	—	—	—

*Open Drain

the data strobe required for that byte. For byte operations, when the A0 bit equals zero, the upper data strobe is issued. When the A0 bit equals one, the lower data strobe is issued. When the data is received, the processor correctly positions it internally. If $\overline{\text{DTACK}}$, $\overline{\text{BERR}}$, or $\overline{\text{VPA}}$ is not asserted for the required setup time before the falling edge of state 4, a wait cycle will be inserted in the bus cycle and $\overline{\text{DTACK}}$ will be sampled again on the falling edge of each wait cycle. The MC68010 will continue to insert wait cycles until $\overline{\text{DTACK}}$ or $\overline{\text{VPA}}$ is recognized.

WRITE CYCLE

During a write cycle, the processor sends data to either the memory or a peripheral device. The processor writes bytes of data in all cases. If the instruction specifies a word operation, the processor writes both bytes. When the instruction specifies a byte operation, the processor uses an internal A0 bit to determine which byte to write and then issues the data strobe required for that byte. For byte operations, when the A0 bit equals zero, the upper data strobe is issued. When the A0 bit equals one, the lower data strobe is issued.

READ-MODIFY-WRITE CYCLE

The read-modify-write cycle performs a read, modifies the data in the arithmetic-logic unit, and writes the data back to the same address. In the MC68010, this cycle is indivisible in that the address strobe is asserted throughout the entire cycle. The test and set (TAS) instruction uses this cycle to provide meaningful communication between processors in a multiple processor environment. This instruction is the only instruction that uses the read-modify-write cycles and since the test and set instruction only operates on bytes, all read-modify-write cycles are byte operations.

CPU SPACE CYCLE

During a CPU space cycle, the MC68010 reads a peripheral device vector number or indicates a breakpoint instruction. If the cycle is to read a vector number it is referred to as an interrupt acknowledge cycle. A CPU space cycle is indicated when the function codes are all high. The address bus then defines what type of CPU space cycle is being executed. The MC68010 defines two types of CPU space cycles, the interrupt acknowledge cycle, and the breakpoint cycle.

The interrupt acknowledge cycle on an M68000 Family compatible processor is defined as a CPU space cycle with the most significant address lines high, on the MC68010 this means that A4-A23 will be high. The level of the interrupt being acknowledged is encoded on address lines A1-A3. An interrupt acknowledge cycle is terminated in the same manner as a normal read cycle. The

processor expects a peripheral device to respond to an interrupt acnkowledge cycle with a vector number that will be used to transfer control to an interrupt handler routine.

The breakpoint read cycle is executed by the MC68010 in response to a breakpoint illegal instruction. A breakpoint cycle on the MC68010 is defined as a CPU space cycle with all of the address lines low. The processor does not accept or send any data during this cycle. The breakpoint cycle may be terminated by DTACK, BERR, or VPA.

PROCESSING STATES

The MC68010 is always in one of three processing states: normal, exception, or halted.

NORMAL PROCESSING

The normal processing state is that associated with instruction execution; the memory references are to fetch instructions and operands, and to store results. A special case of the normal state is the stopped state which the processor enters when a stop instruction is executed. In this state, no further references are made.

EXCEPTION PROCESSING

The exception processing state is associated with interrupts, trap instructions, tracing, and other exception conditions. The exception may be internally generated by an instruction or by an unusual condition arising during the execution of an instruction. Externally, exception processing can be forced by an interrupt, by a bus error, or by a reset. Exception processing is designed to provide an efficient context switch so that the processor may handle unusual conditions.

HALTED PROCESSING

The halted processing state is an indication of catastrophic hardware failure. For example, if during the exception processing of a bus error another bus error occurs, the processor assumes that the system is unusable and halts. Only an external reset can restart a halted processor. Note that a processor in the stopped state is not in the halted state, nor vice versa.

INTERFACE WITH M6800 PERIPHERALS

Motorola's extensive line of M6800 peripherals are directly compatible with the MC68010. Some of these devices that are particularly useful are:

- MC6821 Peripheral Interface Adapter
- MC6840 Programmable Timer Module
- MC6843 Floppy Disk Controller
- MC6845 CRT Controller
- MC6850 Asynchronous Communications Interface Adapter
- MC6854 Advanced Data Link Controller

To interface the synchronous M6800 peripherals with the asynchronous MC68010, the processor modifies its bus cycle to meet the M6800 cycle requirements whenever an M6800 device address is detected. This is possible since both the processors use memory mapped I O.

3

ELECTRICAL SPECIFICATIONS

MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Supply Voltage	V_{CC}	−0.3 to +7.0	V
Input Voltage	V_{in}	−0.3 to +7.0	V
Operating Temperature Range MC68010 MC68010C	T_A	T_L to T_H 0 to 70 −40 to 85	C
Storage Temperature	T_{stg}	−55 to 150	C

The device contains circuitry to protect the inputs against damage due to high static voltages or electric fields; however, normal precautions should be taken to avoid application of voltages higher than maximum-rated voltages to these high-impedance circuits. Tying unused inputs to the appropriate logic voltage level (e.g., either GND or V_{CC}) enhances reliability of operation.

THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Symbol	Value	Rating
Thermal Resistance (Still Air)	θ_{JA}		θ_{JC}		C/W
Ceramic, Type L/LC		30		15*	
Ceramic, Type R/RC		33		15	
Plastic, Type P		30		15*	
Plastic, Type FN		45		25*	

*Estimated

POWER CONSIDERATIONS

The average die-junction temperature, T_J, in °C can be obtained from:

$$T_J = T_A + (P_D \cdot \theta_{JA}) \quad (1)$$

Where:

T_A = Ambient Temperature °C

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

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

$P_{INT} = I_{CC} + V_{CC}$, 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 appropriate relationship between P_D and T_J (if $P_{I/O}$ is neglected) is:

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

Solving equations 1 and 2 for K gives:

$$K = P_D \cdot (T_A + 273°C) + \theta_{JA} \cdot P_D^2 \quad (3)$$

Where K is a constant pertaining to the particular part. K can be determined from equation (3) by measuring P_D (at thermal 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.

The curve shown in Figure 5 gives the graphic solution to the above equations for the specified power dissipation of 1.5 watts over the ambient temperature range of 55 C to 125°C using a maximum θ_{JA} of 45°C/W. Ambient temperature is that of the still air surrounding the device. Lower values of θ_{JA} causes the curve to shift downward slightly; for instance, for θ_{JA} of 40°C/W, the curve is just below 1.4 watts at 25°C.

The total thermal resistance of a package (θ_{JA}) can be seperated into two components, θ_{JA} and θ_{CA}, representing the barrier to heat flow from the semiconductor junction to the package (case) surface (θ_{JC}) and from the case to the outside ambient air (θ_{CA}). These terms are related by the equation:

$$\theta_{JA} = \theta_{JC} + \theta_{CA} \quad (4)$$

θ_{JC} is device related and cannot be influenced by the user. However, θ_{CA} is user dependent and can be minimized by such thermal management techniques as heat sinks, ambient air cooling, and thermal convection. Thus, good thermal management on the part of the user can significantly reduce θ_{CA} so that θ_{JA} approximately equals θ_{JC}. Substitution of θ_{JC} for θ_{JA} in equation (1) will result in a lower semiconductor junction temperature.

Table 7 summarizes maximum power dissipation and average junction temperature for the curve drawn in Figure 5, using the minimum and maximum values of ambient temperature for different packages and substituting θ_{JC} for θ_{JA} (assuming good thermal management). Table 8 provides the maximum power dissipation and average junction temperature assuming that no thermal management is applied (i.e., still air).

NOTE

Since the power dissipation curve shown in Figure 5 is negatively sloped, power dissipation declines as ambient temperature increases. Therefore, maximum power dissipation occurs at the lowest rated ambient temperature, but the highest average junction temperature occurs at the maximum ambient temperature where *power dissipation is lowest.*

Values for thermal resistance presented in this summary, unless estimated, were derived using the procedure described in Motorola Reliability Report 7843, "Thermal Resistance Measurement Method for MC68XXX Microcomponent Devices", and are provided for design purposes only. Thermal measurements are complex and dependent on procedure and setup. User-derived values for thermal resistance may differ.

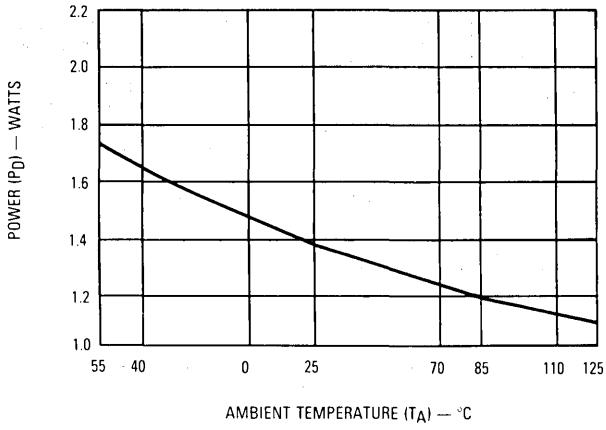


Figure 5. MC68000 Power Dissipation (P_D) vs Ambient Temperature (T_A)

Table 7. MC68010 Power Dissipation and Junction Temperature vs Temperature ($\theta_{JC} = \theta_{JA}$)

Package	T_A Range	θ_{JC} (°C/W)	P_D (W) @ T_A Min	T_J (°C) @ T_A Min	P_D (W) @ T_A Max	T_J (°C) @ T_A Max
L/LC	0°C to 70°C	15	1.5	23	1.2	88
	−40°C to +85°C	15	1.7	−14	1.2	103
	0°C to 85°C	15	1.5	23	1.2	103
P	0°C to 70°C	15	1.5	23	1.2	88
R/RC	0°C to 70°C	15	1.5	23	1.2	88
	−40°C to +85°C	15	1.7	−14	1.2	103
	0°C to 85°C	15	1.5	23	1.2	103
FN	0°C to 70°C	25	1.5	38	1.2	101

Table 8. MC68010 Power Dissipation and Junction Temperature vs Temperature ($\theta_{JC} \leq \theta_{JA}$)

Package	T_A Range	θ_{JA} (°C/W)	P_D (W) @ T_A Min	T_J (°C) @ T_A Min	P_D (W) @ T_A Max	T_J (°C) @ T_A Max
L/LC	0°C to 70°C	30	1.5	23	1.2	88
	−40°C to +85°C	30	1.7	−14	1.2	103
	0°C to 85°C	30	1.5	23	1.2	103
P	0°C to 70°C	30	1.5	23	1.2	88
R/RC	0°C to 70°C	33	1.5	23	1.2	88
	−40°C to +85°C	33	1.7	−14	1.2	103
	0°C to 85°C	33	1.5	23	1.2	103
FN	0°C to 70°C	40	1.5	38	1.2	101

DC ELECTRICAL CHARACTERISTICS ($V_{CC} = 5.0$ Vdc ± 5%; GND = 0 Vdc; $T_A = T_L$ to T_H)

Characteristics		Symbol	Min	Max	Unit
Input High Voltage		V_{IH}	2.0	V_{CC}	V
Input Low Voltage		V_{IL}	GND − 0.3	0.8	V
Input Leakage Current	$\overline{BERR}$, $\overline{BGACK}$, $\overline{BR}$, $\overline{DTACK}$, CLK, $\overline{IPL0}$-$\overline{IPL2}$, $\overline{VPA}$	I_{IN}	—	2.5	μA
@ 5.25 V	$\overline{HALT}$, $\overline{RESET}$		—	20	
Three-State (Off State) Input Current	$\overline{AS}$, A1-A23, D0-D15, FC0-FC2,	I_{TSI}	—	20	μA
@ 2.4V/0.4 V	$\overline{LDS}$, R/$\overline{W}$, $\overline{UDS}$, $\overline{VMA}$				
Output High Voltage (I_{OH} = −400 μA)	E*	V_{OH}	V_{CC} − 0.75	—	V
(I_{OH} = −400 μA)	E, $\overline{AS}$, A1-A23, $\overline{BG}$, D0-D15, FC0-FC2, $\overline{LDS}$, R/$\overline{W}$, $\overline{UDS}$, $\overline{VMA}$		2.4	2.4	
Output Low Voltage		V_{OL}			V
(I_{OL} = 1.6 mA)	$\overline{HALT}$		—	0.5	
(I_{OL} = 3.2 mA)	A1-A23, $\overline{BG}$, FC0-FC2		—	0.5	
(I_{OL} = 5.0 mA)	$\overline{RESET}$		—	0.5	
(I_{OL} = 5.3 mA)	E, $\overline{AS}$, D0-D15, $\overline{LDS}$, R/$\overline{W}$, $\overline{UDS}$, $\overline{VMA}$		—	0.5	
Power Dissipation (see **POWER CONSIDERATIONS**)		P_D***	—	—	W
Capacitance (V_{in} = 0 V, T_A = 25°C, Frequency = 1 MHz)**		C_{in}	—	20.0	pF
Load Capacitance	$\overline{HALT}$	C_L	—	70	pF
	All Others		—	130	

*With external pullup resistor of 1.1 Ω.
**Capacitance is periodically sampled rather than 100% tested.
***During normal operation instantaneous V_{CC} current requirements may be as high as 1.5 A.

AC ELECTRICAL SPECIFICATIONS — CLOCK TIMING (see Figure 6)

Num.	Characteristic	Symbol	8 MHz*		10 MHz*		12.5 MHz*		Unit
			Min	Max	Min	Mat	Min	Max	
	Frequency of Operation	f	4	8	4	10	4	12.5	MHz
1	Clock Period	t_{cyc}	125	250	100	250	80	250	ns
2,3	Clock Pulse Measured from 1.5 V to 1.5 V	t_{CL}, t_{CH}	55	125	45	125	35	125	ns
4,5	Clock Rise and Fall Times	t_{Cr}, t_{Cf}	—	10	—	10	—	5	ns

*These specifications represent an improvement over previously published specifications for the 8-, 10-, and 12.5-MHz MC68010 and are valid only for product bearing date codes of 8827 and later.

3

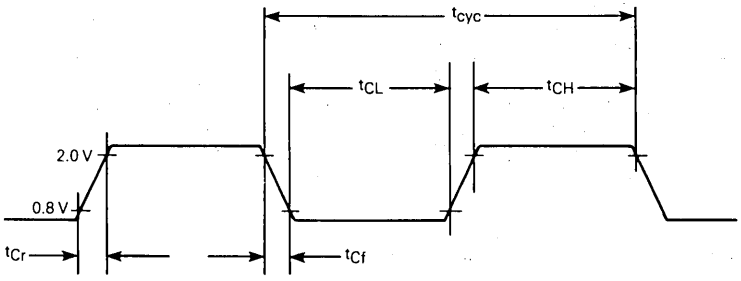


Figure 6. MC68010 Clock Input Timing Diagram

AC ELECTRICAL SPECIFICATIONS — READ AND WRITE CYCLES (V_{CC} = 5.0 Vdc ± 5%; GND = 0 Vdc; $T_A = T_L$ to T_H; see Figures 8 and 9)

Num.	Characteristic	Symbol	8 MHz*		10 MHz*		12.5 MHz*		Unit
			Min	Max	Min	Mat	Min	Max	
6	Clock Low to Address Valid	t_{CLAV}	—	62	—	50	—	50	ns
6A	Clock High to FC Valid	t_{CHFCV}	—	62	—	50	—	45	ns
7	Clock High to Address, Data Bus High Impedance (Maximum)	t_{CHADZ}	—	80	—	70	—	60	ns
8	Clock High to Address, FC Invalid (Minimum)	t_{CHAFI}	0	—	0	—	0	—	ns
9[1]	Clock High to $\overline{AS}$, $\overline{DS}$ Asserted	t_{CHSL}	3	60	3	50	3	40	ns
11[2]	Address Valid to $\overline{AS}$, $\overline{DS}$ Asserted (Read)/$\overline{AS}$ Asserted (Write)	t_{AVSL}	30	—	20	—	15	—	ns
11A[2]	FC Valid to $\overline{AS}$, $\overline{DS}$ Asserted (Read)/$\overline{AS}$ Asserted (Write)	t_{FCVSL}	90	—	70	—	60	—	ns
12[1]	Clock Low to $\overline{AS}$, $\overline{DS}$ Negated	t_{CLSH}	—	62	—	50	—	40	ns
13[2]	$\overline{AS}$, $\overline{DS}$ Negated to Address, FC Invalid	t_{SHAFI}	40	—	30	—	20	—	ns
14[2]	$\overline{AS}$ (and $\overline{DS}$ Read) Width Asserted	t_{SL}	270	—	195	—	160	—	ns
14A[2]	$\overline{DS}$ Width Asserted (Write)	t_{DSL}	140	—	95	—	80	—	ns
15[2]	$\overline{AS}$, $\overline{DS}$ Width Negated	t_{SH}	150	—	105	—	65	—	ns
16	Clock High to Control Bus High Impedance	t_{CHCZ}	—	80	—	70	—	60	ns
17[2]	$\overline{AS}$, $\overline{DS}$ Negated to R/$\overline{W}$ Invalid	t_{SHRH}	40	—	30	—	20	—	ns
18[1]	Clock High to R/$\overline{W}$ High (Read)	t_{CHRH}	0	55	0	45	0	40	ns
20[1]	Clock High to R/$\overline{W}$ Low (Write)	t_{CHRL}	0	55	0	45	0	40	ns
20A[2, 6]	$\overline{AS}$ Asserted to R/$\overline{W}$ Valid (Write)	t_{ASRV}	—	10	—	10	—	10	ns
21[2]	Address Valid to R/$\overline{W}$ Low (Write)	t_{AVRL}	20	—	0	—	0	—	ns
21A[2]	FC Valid to R/$\overline{W}$ Low (Write)	t_{FCVRL}	60	—	50	—	30	—	ns
22[2]	R/$\overline{W}$ Low to $\overline{DS}$ Asserted (Write)	t_{RLSL}	80	—	50	—	30	—	ns
23	Clock Low to Data-Out Valid (Write)	t_{CLDO}	—	62	—	50	—	50	ns
25[2]	$\overline{AS}$, $\overline{DS}$ Negated to Data-Out Invalid (Write)	t_{SHDOI}	40	—	30	—	20	—	ns
26[2]	Data-Out Valid to $\overline{DS}$ Asserted (Write)	t_{DOSL}	40	—	30	—	20	—	ns
27[5]	Data-In Valid to Clock Low (Setup Time of Read)	t_{DICL}	10	—	10	—	10	—	ns
27A[5]	Late $\overline{BERR}$ Asserted to Clock Low (Setup Time)	t_{BELCL}	45	—	45	—	45	—	ns
28[2]	$\overline{AS}$, $\overline{DS}$ Negated to $\overline{DTACK}$ Negated (Asynchronous Hold)	t_{SHDAH}	0	240	0	190	0	150	ns
29	$\overline{AS}$, $\overline{DS}$ Negated to Data-In Invalid (Hold Time on Read)	t_{SHDII}	0	—	0	—	0	—	ns
29A	$\overline{AS}$, $\overline{DS}$ Negated to Data In High Impedance	t_{SHDZ}	—	187	—	150	—	120	ns
30	$\overline{AS}$, $\overline{DS}$ Negated to $\overline{BERR}$ Negated	t_{SHBEH}	0	—	0	—	0	—	ns
31[2, 5]	$\overline{DTACK}$ Asserted to Data-In Valid (Setup Time)	t_{DALDI}	—	90	—	65	—	50	ns
32	$\overline{HALT}$ and $\overline{RESET}$ Input Transition Time	$t_{RHr,f}$	0	200	0	200	0	200	ns
33	Clock High to $\overline{BG}$ Asserted	t_{CHGL}	—	62	—	50	—	40	ns
34	Clock High to $\overline{BG}$ Negated	t_{CHGH}	—	62	—	50	—	40	ns
35	$\overline{BR}$ Asserted to $\overline{BG}$ Asserted	t_{BRLGL}	1.5	3.5	1.5	3.5	1.5	3.5	Clks
36[7]	$\overline{BR}$ Negated to $\overline{BG}$ Negated	t_{BRHGH}	1.5	3.5	1.5	3.5	1.5	3.5	Clks

AC ELECTRICAL SPECIFICATIONS — READ AND WRITE CYCLES (Continued)

Num.	Characteristic	Symbol	8 MHz*		10 MHz*		12.5 MHz*		Unit
			Min	Max	Min	Mat	Min	Max	
37	$\overline{BGACK}$ Asserted to $\overline{BG}$ Negated	t_{GALGH}	1.5	3.5	1.5	3.5	1.5	3.5	Clks
37A[8]	$\overline{BGACK}$ Asserted to $\overline{BR}$ Negated	t_{GALBRH}	20	1.5 Clks	20	1.5 Clks	20	1.5 Clks	ns
38	$\overline{BG}$ Asserted to Control, Address, Data Bus High Impedance ($\overline{AS}$ Negated)	t_{GLZ}	—	80	—	70	—	60	ns
39	$\overline{BG}$ Width Negated	t_{GH}	1.5	—	1.5	—	1.5	—	Clks
40	Clock Low to $\overline{VMA}$ Asserted	t_{CLVML}	—	70	—	70	—	70	ns
41	Clock Low to E Transition	t_{CLET}	—	55	—	45	—	35	ns
42	E Output Rise and Fall Time	$t_{Er,f}$	—	15	—	15	—	15	ns
43	$\overline{VMA}$ Asserted to E High	t_{VMLEH}	200	—	150	—	90	—	ns
44	$\overline{AS}$, $\overline{DS}$ Negated to $\overline{VPA}$ Negated	t_{SHVPH}	0	120	0	90	0	70	ns
45	E Low to Control, Address Bus Invalid (Address Hold Time)	t_{ELCAI}	30	—	10	—	10	—	ns
46	$\overline{BGACK}$ Width Low	t_{GAL}	1.5	—	1.5	—	1.5	—	Clks
47[5]	Asynchronous Input Setup Time	t_{ASI}	10	—	10	—	10	—	ns
48[2,3,5]	$\overline{DTACK}$ Asserted to $\overline{BERR}$ Asserted	t_{DALBEL}	—	80	—	55	—	35	ns
49[9]	$\overline{AS}$, $\overline{DS}$, Negated to E Low	t_{SHEL}	−70	70	−55	55	−45	45	ns
50	E Width High	t_{EH}	450	—	350	—	280	—	ns
51	E Width Low	t_{EL}	700	—	550	—	440	—	ns
53	Data-Out Hold from Clock High	t_{CHDOI}	0	—	0	—	0	—	ns
54	E Low to Data-Out Invalid	t_{ELDOI}	30	—	20	—	15	—	ns
55	R/$\overline{W}$ Asserted to Data Bus Impedance Change	t_{RLDBD}	30	—	20	—	10	—	ns
56[4]	$\overline{HALT}$/$\overline{RESET}$ Pulse Width	t_{HRPW}	10	—	10	—	10	—	Clks
57	$\overline{BGACK}$ Negated to $\overline{AS}$, $\overline{DS}$, R/$\overline{W}$ Driven	t_{GASD}	1.5	—	1.5	—	1.5	—	Clks
57A	$\overline{BGACK}$ Negated to FC, $\overline{VMA}$ Driven	t_{GAFD}	1	—	1	—	1	—	Clks
58[7]	$\overline{BR}$ Negated to $\overline{AS}$, $\overline{DS}$, R/$\overline{W}$ Driven	t_{RHSD}	1.5	—	1.5	—	1.5	—	Clks
58A[7]	$\overline{BR}$ Negated to FC, $\overline{VMA}$ Driven	t_{RHFD}	1	—	1	—	1	—	Clks

*These specifications represent an improvement over previously published specifications for the 8-, 10-, and 12.5-MHz MC68010 and are valid only for product bearing date codes of 8827 and later.

NOTES:

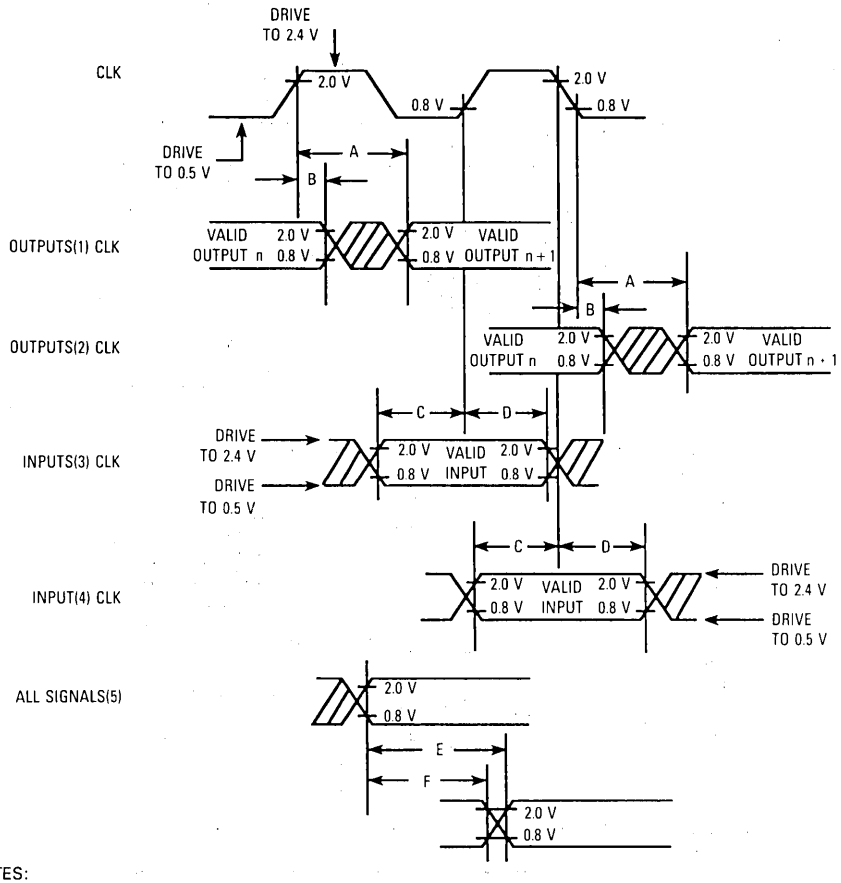
1. For a loading capacitance of less than or equal to 50 picofarads, subtract 5 nanoseconds from the value given in the maximum columns.
2. Actual value depends on clock period.
3. In the absence of $\overline{DTACK}$, $\overline{BERR}$ is an asynchronous input using the asynchronous input setup time (#47).
4. For power up, the MC68000 must be held in the $\overline{RESET}$ state for 100 milliseconds to allow stabilization of on-chip circuitry. After the system is powered up, #56 refers to the minimum pulse width required to reset the processor.
5. If the asynchronous input setup time (#47) requirement is satisfied for $\overline{DTACK}$, the $\overline{DTACK}$-asserted to data setup time (#31) and $\overline{DTACK}$-asserted to $\overline{BERR}$-asserted setup time (#48) requirements can be ignored. The data must only satisfy the data-in to clock low setup time (#27) for the following clock cycle, and $\overline{BERR}$ must only satisfy the late-$\overline{BERR}$-asserted to clock-low setup time (#27A) for the following clock cycle.
6. When $\overline{AS}$ and R/$\overline{W}$ are equally loaded (±20%), subtract 5 nanoseconds from the values given in these columns.
7. The processor will negate $\overline{BG}$ and begin driving the bus again if external arbitration logic negates $\overline{BR}$ before asserting $\overline{BGACK}$.
8. The minimum value must be met to guarantee proper operation. If the maximum value is exceeded, $\overline{BG}$ may be reasserted.
9. The falling edge of S6 triggers both the negation of the strobes ($\overline{AS}$ and $\overline{xDS}$) and the falling edge of E. Either of these events can occur first, depending upon the loading on each signal. Specification #49 indicates the absolute maximum skew that will occur between the rising edge of the strobes and the falling edge of the E clock.

AC ELECTRICAL SPECIFICATION DEFINITIONS

The AC specifications presented consist of output delays, input setup and hold times, and signal skew times. All signals are specified relative to an appropriate edge of the clock input and, possibly, relative to one or more other signals.

The measurement of the AC specifications is defined by the waveforms shown in Figure 7. In order to test the parameters guaranteed by Motorola, inputs must be driven to the voltage levels specified in Figure 7. Outputs are specified with minimum and/or maximum limits, as appropriate, and are measured as shown in Figure 7. Inputs are specified with minimum and, as appropriate, maximum setup and hold times, and are measured as shown. Finally, the measurement for signal-to-signal specifications are also shown.

Note that the testing levels used to verify conformance to the AC specifications does not affect the guaranteed DC operation of the device as specified in the DC electrical characteristics.



NOTES:

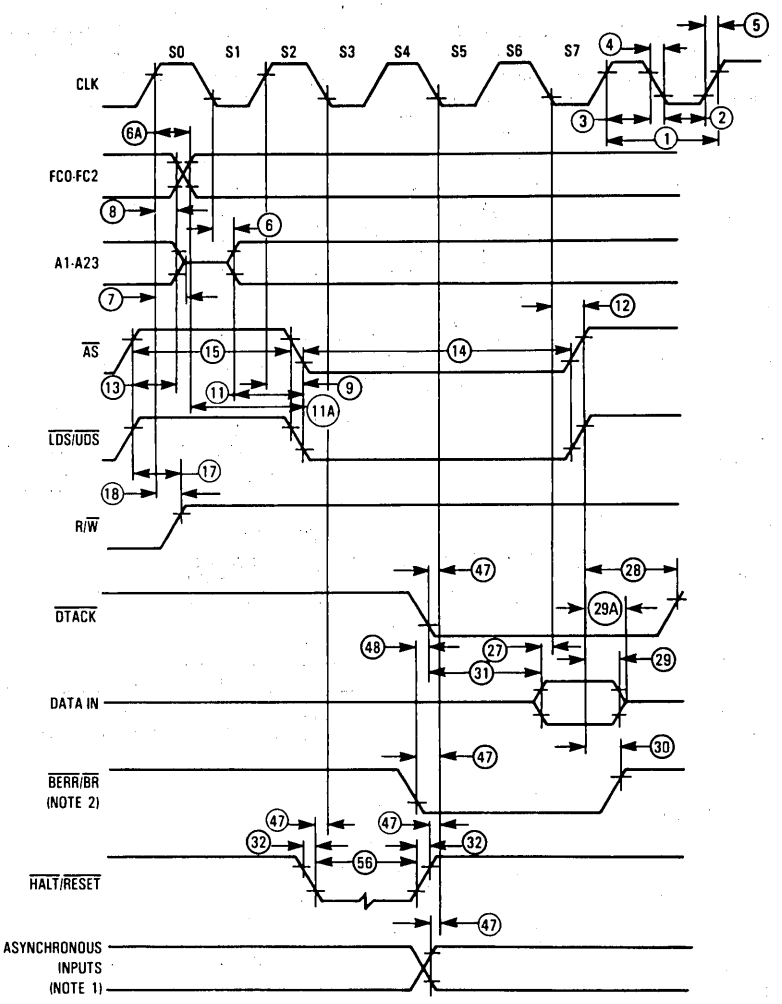
1. This output timing is applicable to all parameters specified relative to the rising edge of the clock.
2. This output timing is applicable to all parameters specified relative to the falling edge of the clock.
3. This input timing is applicable to all parameters specified relative to the rising edge of the clock.
4. This input timing is applicable to all parameters specified relative to the falling edge of the clock.
5. This timing is applicable to all parameters specified relative to the assertion/negation of another signal.

LEGEND:

A. Maximum output delay specification.
B. Minimum output hold time.
C. Minimum input setup time specification.
D. Minimum input hold time specification.
E. Signal valid to signal valid specification (maximum or minimum).
F. Signal valid to signal invalid specification (maximum or minimum).

Figure 7. Drive Levels and Test Points for AC Specifications

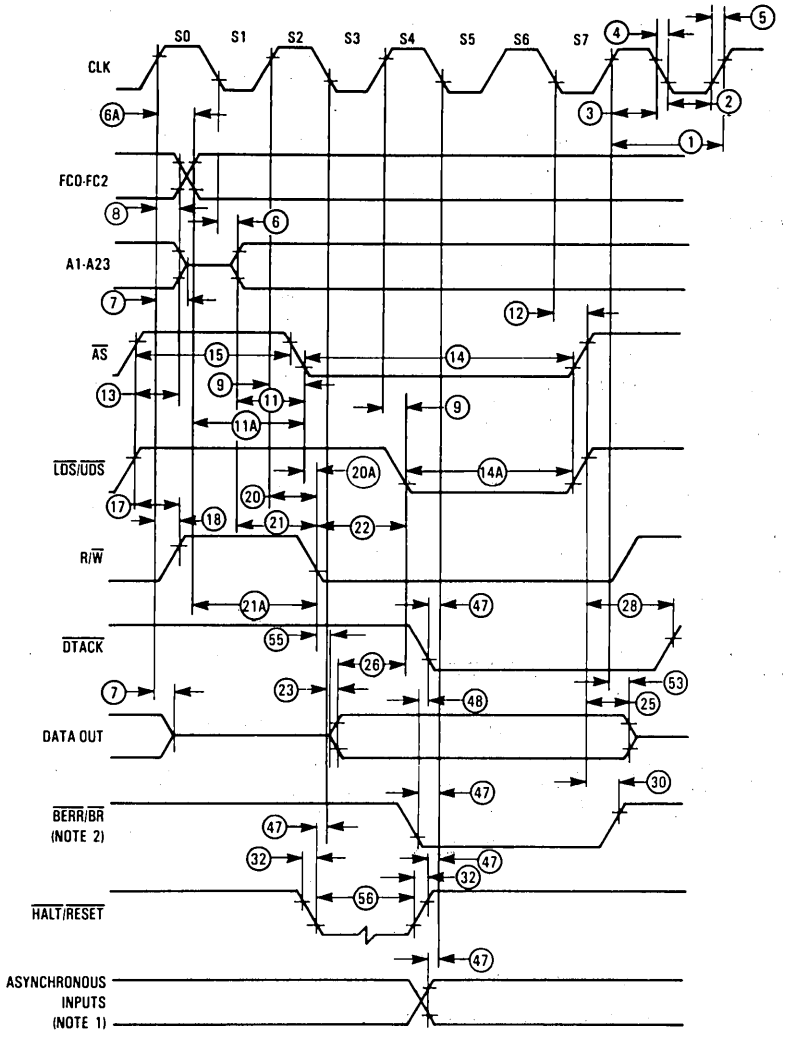
These waveforms should only be referenced in regard to the edge-to-edge measurement of the timing specifications. They are not intended as a functional description of the input and output signals. Refer to other functional descriptions and their related diagrams for device operation.



NOTES:
1. Setup time for the asynchronous inputs IPL0-IPL2 and VPA guarantees their recognition at the next falling edge of the clock.
2. BR need fall at this time only in order to insure being recognized at the end of this bus cycle.
3. Timing measurements are referenced to and from a low voltage of 0.8 volts and a high voltages of 2.0 volts, unless otherwise noted.
4. The timing for the first falling edge (47) of BERR are for BERR without DTACK, the timings for the second falling edge (27A and 48) are for BERR with DTACK.

Figure 8. MC68010 Read-Cycle Timing Diagram

These waveforms should only be referenced in regard to the edge-to-edge measurement of the timing specifications. They are not intended as a functional description of the input and output signals. Refer to other functional descriptions and their related diagrams for device operation.



NOTE:

1. Timing measurements are referenced to and from a low voltage of 0.8 volts and a high voltage of 2.0 volts, unless otherwise noted.
2. Because of loading variations, R/$\overline{W}$ may be valid after $\overline{AS}$ even though both are initiated by the rising edge of S2 (Specification 20A).
3. The timing for the first falling edge (47) of $\overline{BERR}$ are for $\overline{BERR}$ without $\overline{DTACK}$, the timings for the second falling edge (27A and 48) are for $\overline{BERR}$ and $\overline{DTACK}$.

Figure 9. MC68010 Write-Cycle Timing Diagram

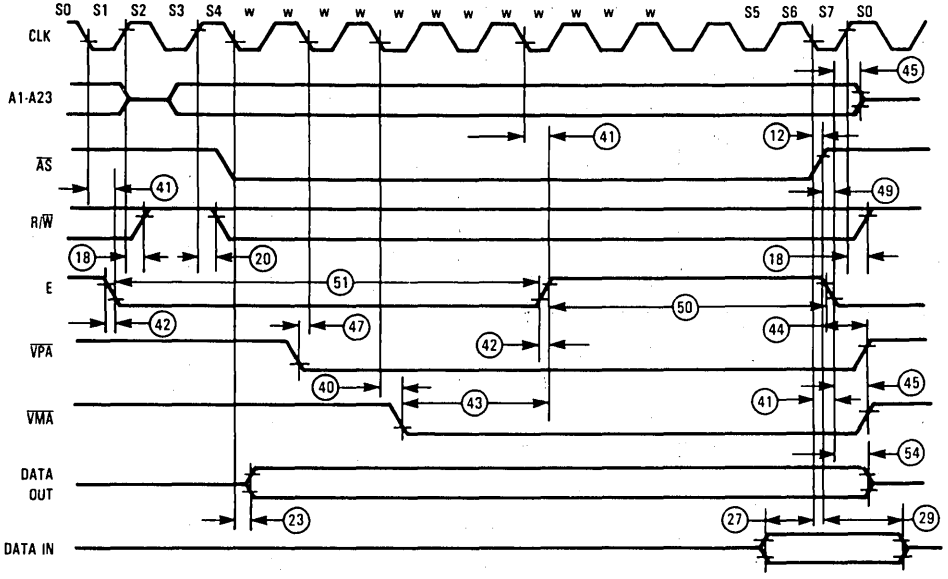
AC ELECTRICAL SPECIFICATIONS — MC68010 TO M6800 PERIPHERAL CYCLES (V_{CC} = 5.0 Vdc ± 5%; GND = 0 Vdc; $T_A = T_L$ to T_H, see Figures 10 and 11)

Num.	Characteristic	Symbol	8 MHz*		10 MHz*		12.5 MHz*		Unit
			Min	Max	Min	Max	Min	Max	
12[1]	Clock Low to $\overline{AS}$, $\overline{DS}$ Negated	t_{CLSH}	—	62	—	50	—	40	ns
18[1]	Clock High to R/$\overline{W}$ High (Read)	t_{CHRH}	0	55	0	45	0	40	ns
20[1]	Clock High to R/$\overline{W}$ Low (Write)	t_{CHRL}	0	55	0	45	0	40	ns
23	Clock Low to Data-Out Valid (Write)	t_{CLDO}	—	62	—	50	—	50	ns
27	Data-In Valid to Clock Low (Setup Time of Read)	t_{DICL}	10	—	10	—	10	—	ns
29	$\overline{AS}$, $\overline{DS}$ Negated to Data-In Invalid (Hold Time on Read)	t_{SHDII}	0	—	0	—	0	—	ns
40	Clock Low to $\overline{VMA}$ Asserted	t_{CLVML}	—	70	—	70	—	70	ns
41	Clock Low to E Transition	t_{CLET}	—	55	—	45	—	35	ns
42	E Output Rise and Fall Time	$t_{Er,f}$	—	15	—	15	—	15	ns
43	$\overline{VMA}$ Asserted to E High	t_{VMLEH}	200	—	150	—	90	—	ns
44	$\overline{AS}$, $\overline{DS}$ Negated to $\overline{VPA}$ Negated	t_{SHVPH}	0	120	0	90	0	70	ns
45	E Low to Control, Address Bus Invalid (Address Hold Time)	t_{ELCAI}	30	—	10	—	10	—	ns
47	Asynchronous Input Setup Time	t_{ASI}	10	—	10	—	10	—	ns
49[2]	$\overline{AS}$, $\overline{DS}$, Negated to E Low	t_{SHEL}	−70	70	−55	55	−45	45	ns
50	E Width High	t_{EH}	450	—	350	—	280	—	ns
51	E Width Low	t_{EL}	700	—	550	—	440	—	ns
54	E Low to Data-Out Invalid	t_{ELDOI}	30	—	20	—	15	—	ns

*These specifications represent an improvement over previously published specifications for the 8-, 10-, and 12.5-MHz MC68010 and are valid only for product bearing date codes of 8827 and later.

NOTES:
1. For a loading capacitance of less than or equal to 50 picofarads, subtract 5 nanoseconds from the value given in the maximum columns.
2. The falling edge of S6 triggers both the negation of the strobes ($\overline{AS}$ and $\overline{xDS}$) and the falling edge of E. Either of these events can occur first, depending upon the loading on each signal. Specification #49 indicates the absolute maximum skew that will occur between the rising edge of the strobes and the falling edge of the E clock.

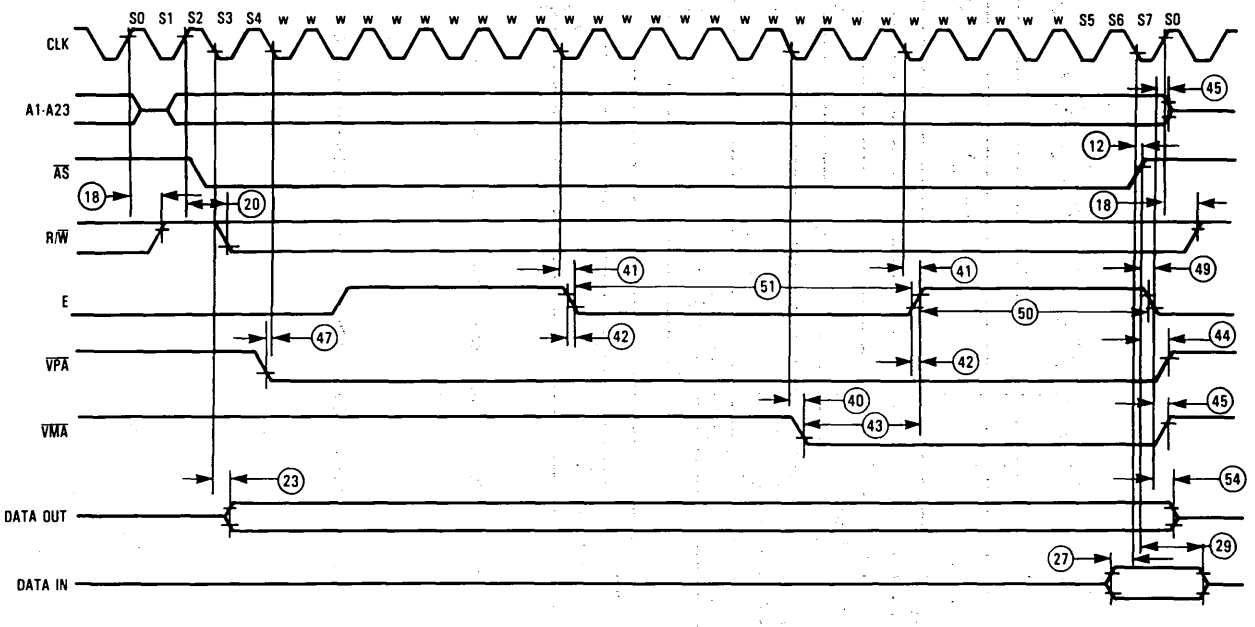
These waveforms should only be referenced in regard to the edge-to-edge measurement of the timing specifications. They are not intended as a functional description of the input and output signals. Refer to other functional descriptions and their related diagrams for device operation.



NOTE: This timing diagram is included for those who wish to design their own circuit to generate $\overline{VMA}$. It shows the best case possibly attainable.

Figure 10. MC68010 to M6800 Peripheral Timing Diagram (Best Case)

These waveforms should only be referenced in regard to the edge-to-edge measurement of the timing specifications. They are not intended as a functional description of the input and output signals. Refer to other functional descriptions and their related diagrams for device operation.



NOTE: This timing diagram is included for those who wish to design their own circuit to generate $\overline{VMA}$. It shows the worst case possibly attainable.

Figure 11. MC68010 to M6800 Peripheral Timing Diagram (Worst Case)

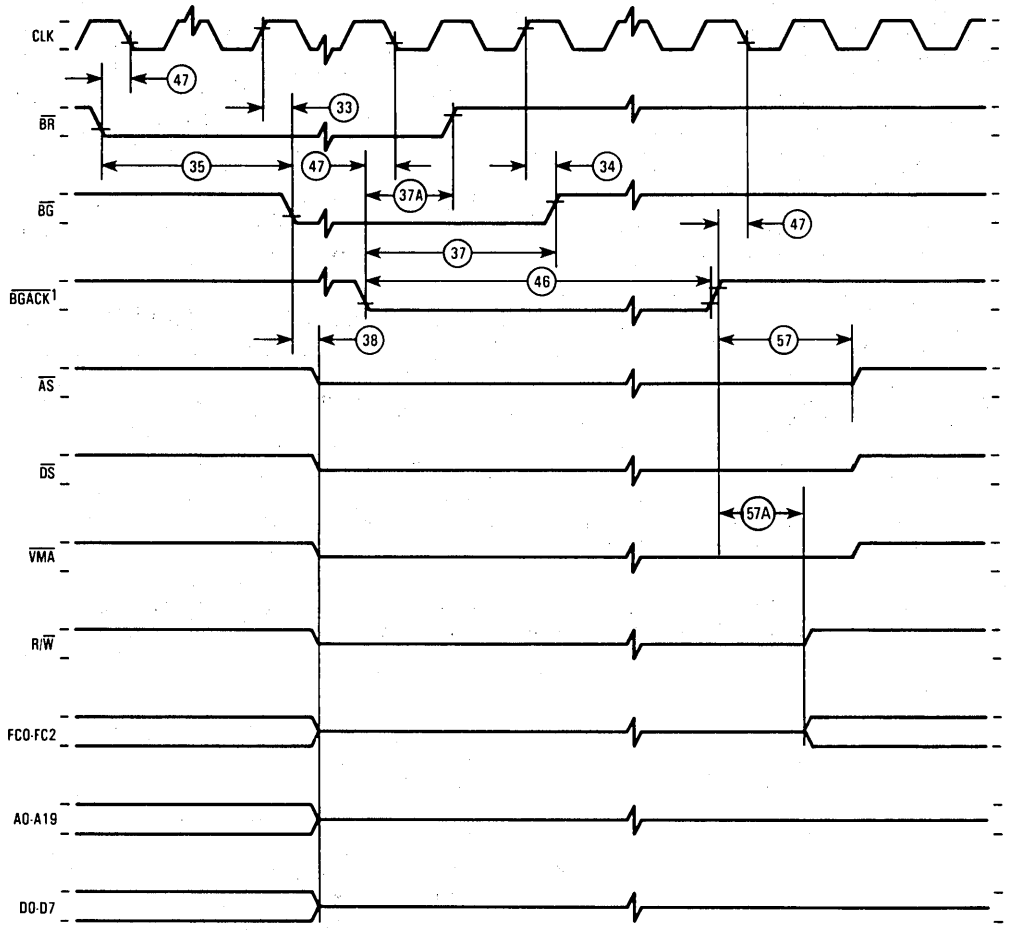
AC ELECTRICAL SPECIFICATIONS — BUS ARBITRATION (V_{CC} = 5.0 Vdc ± 5%; GND = 0 Vdc; $T_A = T_L$ to T_H; see Figures 12,13, and 14)

Num.	Characteristic	Symbol	8 MHz*		10 MHz*		12.5 MHz*		Unit
			Min	Max	Min	Max	Min	Max	
7	Clock High to Address, Data Bus High Impedance (Maximum)	t_{CHADZ}	—	80	—	70	—	60	ns
16	Clock High to Control Bus High Impedance	t_{CHCZ}	—	80	—	70	—	60	ns
33	Clock High to $\overline{BG}$ Asserted	t_{CHGL}	—	62	—	50	—	40	ns
34	Clock High to $\overline{BG}$ Negated	t_{CHGH}	—	62	—	50	—	40	ns
35	$\overline{BR}$ Asserted to $\overline{BG}$ Asserted	t_{BRLGL}	1.5	3.5	1.5	3.5	1.5	3.5	Clks
36[1]	$\overline{BR}$ Negated to $\overline{BG}$ Negated	t_{BRHGH}	1.5	3.5	1.5	3.5	1.5	3.5	Clks
37	$\overline{BGACK}$ Asserted to $\overline{BG}$ Negated	t_{GALGH}	1.5	3.5	1.5	3.5	1.5	3.5	Clks
37A[2]	$\overline{BGACK}$ Asserted to $\overline{BR}$ Negated	t_{GALBRH}	20	1.5 Clks	20	1.5 Clks	20	1.5 Clks	ns
38	$\overline{BG}$ Asserted to Control, Address, Data Bus High Impedance ($\overline{AS}$ Negated)	t_{GLZ}	—	80	—	70	—	60	ns
39	$\overline{BG}$ Width Negated	t_{GH}	1.5	—	1.5	—	1.5	—	Clks
46	$\overline{BGACK}$ Width Low	t_{GAL}	1.5	—	1.5	—	1.5	—	Clks
47	Asynchronous Input Setup Time	t_{ASI}	10	—	10	—	10	—	ns
57	$\overline{BGACK}$ Negated to $\overline{AS}$, $\overline{DS}$, R/$\overline{W}$ Driven	t_{GASD}	1.5	—	1.5	—	1.5	—	Clks
57A	$\overline{BGACK}$ Negated to FC, $\overline{VMA}$ Driven	t_{GAFD}	1	—	1	—	1	—	Clks
58[1]	$\overline{BR}$ Negated to $\overline{AS}$, $\overline{DS}$, R/$\overline{W}$ Driven	t_{RHSD}	1.5	—	1.5	—	1.5	—	Clks
58A[1]	BR Negated to FC, $\overline{VMA}$ Driven	t_{RHFD}	1	—	1	—	1	—	Clks

*These specifications represent an improvement over previously published specifications for the 8-, 10-, and 12.5-MHz MC68010 and are valid only for product bearing date codes of 8827 and later.

NOTES:
1. The processor will negate $\overline{BG}$ and begin driving the bus again if external arbitration logic negates $\overline{BR}$ before asserting $\overline{BGACK}$.
2. The minimum value must be met to guarantee proper operation. If the maximum value is exceeded, $\overline{BG}$ may be reasserted.

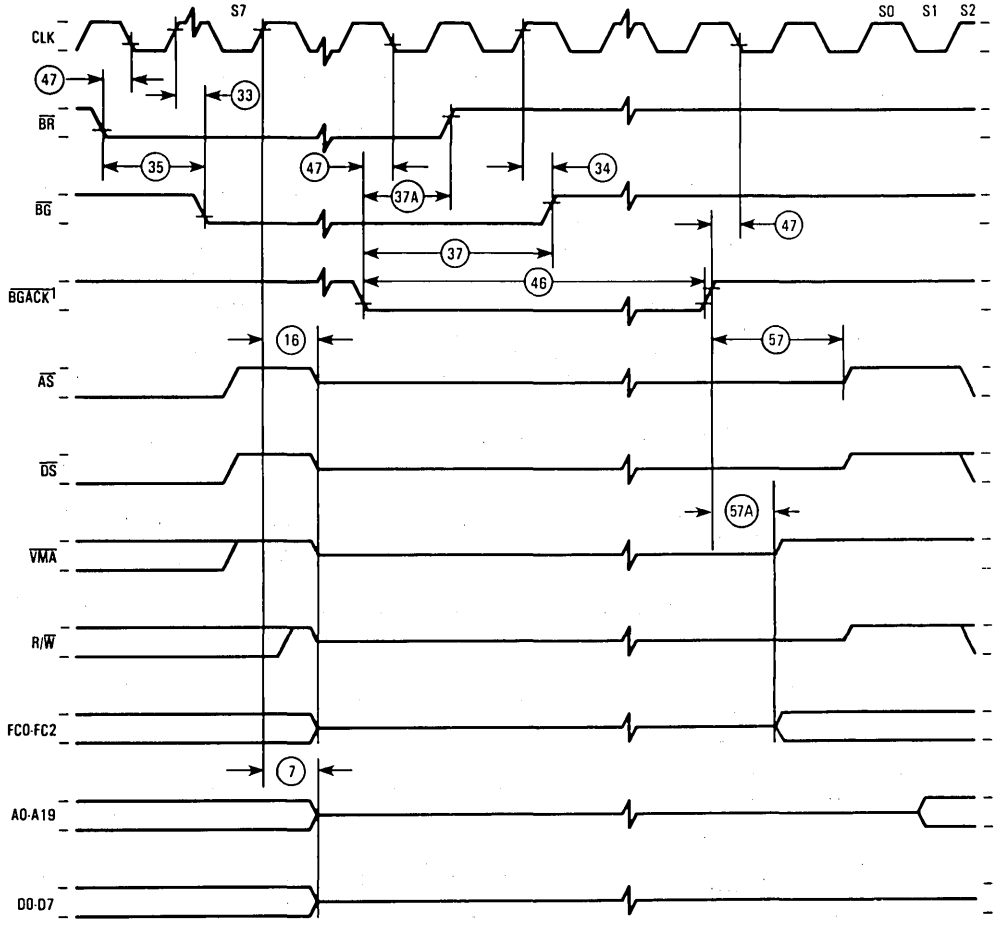
These waveforms should only be referenced in regard to the edge-to-edge measurement of the timing specifications. They are not intended as a functional description of the input and output signals. Refer to other functional descriptions and their related diagrams for device operation.



NOTE:
1. 52-Pin Version of MC68008 Only

Figure 12. MC68010 Bus Arbitration Timing — Idle Bus Case

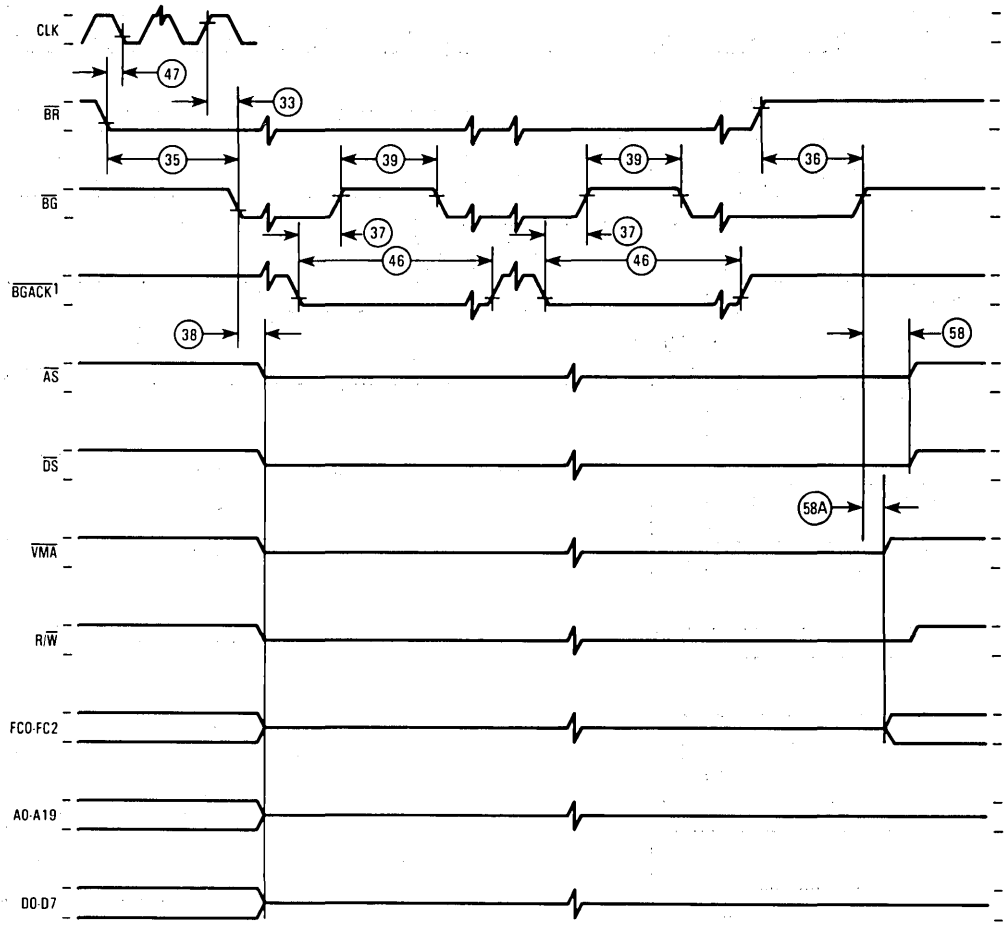
These waveforms should only be referenced in regard to the edge-to-edge measurement of the timing specifications. They are not intended as a functional description of the input and output signals. Refer to other functional descriptions and their related diagrams for device operation.



NOTE:
1. 52-Pin Version of MC68008 Only

Figure 13. MC68010 Bus Arbitration Timing — Active Bus Case

These waveforms should only be referenced in regard to the edge-to-edge measurement of the timing specifications. They are not intended as a functional description of the input and output signals. Refer to other functional descriptions and their related diagrams for device operation.

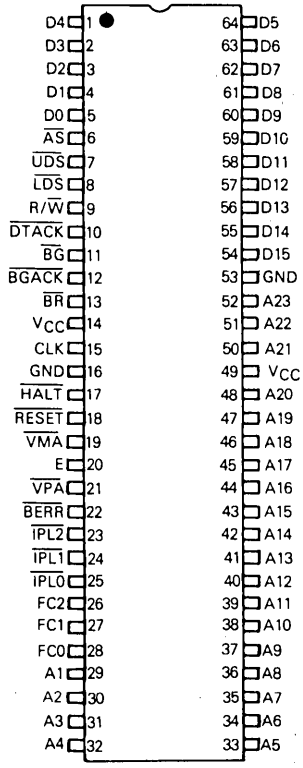


NOTE:
1. 52-Pin Version of MC68008 Only

Figure 14. MC68010 Bus Arbitration Timing — Multiple Bus Requests

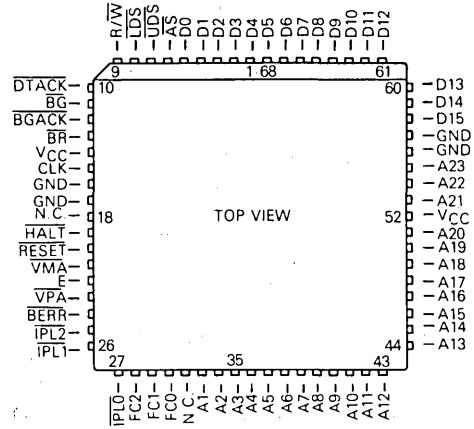
PIN ASSIGNMENTS

64-Pin Dual-In-Line Package

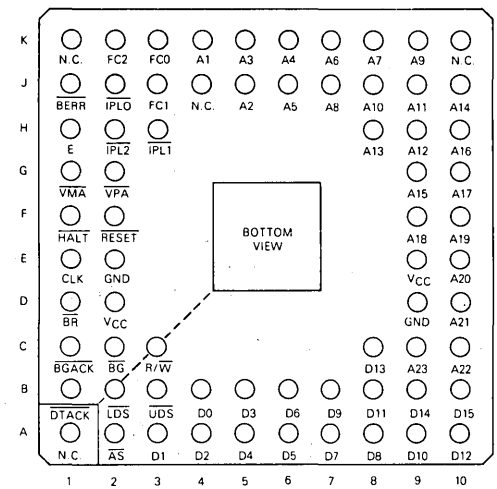


Top View

68-Pin Quad Pack



68-Pin Grid Array



MC68020

Technical Summary

32-Bit Virtual Memory Microprocessor

3

The MC68020 is the first full 32-bit implementation of the M68000 Family of microprocessors from Motorola. Using Motorola's advanced HCMOS technology, the MC68020 is implemented with 32-bit registers and data paths, 32-bit addresses, a rich instruction set, and versatile addressing modes.

The main features of the MC68020 are:

- Object Code Compatible with Earlier M68000 Microprocessors
- Addressing Mode Extensions for Enhanced Support of High Level Languages
- New Bit Field Data Type Accelerates Bit-Oriented Applications, i.e., Video Graphics
- Fast On-Chip Instruction Cache Speeds Instructions and Improves Bus Bandwidth
- Coprocessor Interface to Companion 32-Bit Peripherals — the MC68881 and MC68882 Floating Point Coprocessors and the MC68851 Paged Memory Management Unit
- Pipelined Architecture with High Degree of Internal Parallelism Allowing Multiple Instructions to be Executed Concurrently
- High Performance Asynchronous Bus is Non-Multiplexed and Full 32-Bits
- Dynamic Bus Sizing Efficiently Supports 8-/16-/32-Bit Memories and Peirpherals
- Full Support of Virtual Memory and Virtual Machine
- Sixteen 32-Bit General-Purpose Data and Address Registers
- Two 32-Bit Supervisor Stack Pointers and 5 Special Purpose Control Registers
- 18 Addressing Modes and 7 Data Types
- 4-Gigabyte Direct Addressing Range
- Selection of Processor Speeds: 12.5, 16.67, 20, 25, and 33.33 MHz

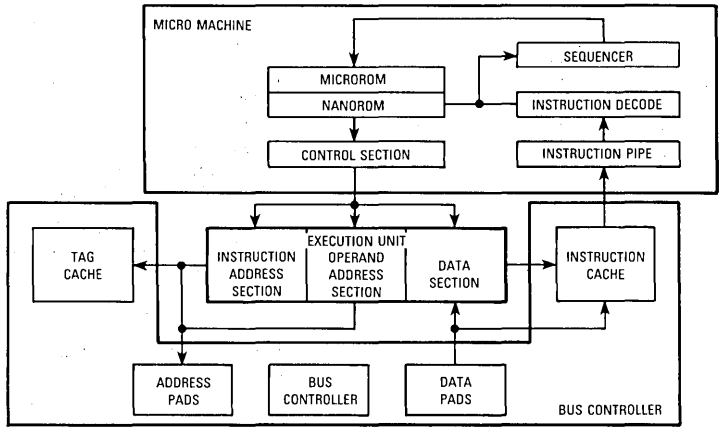


Figure 1. MC68020 Block Diagram

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

INTRODUCTION

The MC68020 is a high-performance 32-bit microprocessor. It is the first microprocessor to have evolved from a 16-bit machine to a full 32-bit machine that provides 32-bit address and data buses as well as 32-bit internal structures. Many techniques were utilized to improve performance and at the same time maintain compatibility with other processors of the M68000 Family. Among the improvements are new addressing modes which better support high-level language structures, an expanded instruction set which provides 32-bit operations for the limited cases not supported by the MC68000 and the MC68010, and several new instructions which support new data types. For special-purpose applications when a general-purpose processor alone is not adequate, a coprocessor interface is provided.

The MC68020 is a high-performance microprocessor implemented in HCMOS, Motorola's low power, small geometry process. This process allows CMOS and HMOS (high density NMOS) gates to be combined on the same device. CMOS structures are used where speed and low power is required, and HMOS structures are used where minimum silicon area is desired. This technology enables the MC68020 to be very fast while consuming less power (less than 1.5 watts), and still have a reasonably small die size.

Figure 1 is a block diagram of the MC68020. The processor can be divided into two main sections: the bus controller and the micromachine. This division reflects the autonomy with which the sections operate.

The bus controller consists of the address and data pads and multiplexers required to support dynamic bus sizing, a macro bus controller which schedules the bus cycles on the basis of priority with two state machines (one to control the bus cycles for operand accesses and the other to control the bus cycles for instruction accesses), and the instruction cache with its associated control.

The micromachine consists of an execution unit, nanorom and microrom storage, an instruction decoder, an instruction pipe, and associated control sections. The execution unit consists of an address section, an operand address section, and a data section. Microcode control is provided by a modified two-level store of microrom and nanorom. Programmed logical arrays (PLAs) are used to provide instruction decode and sequencing information. The instruction pipe and other individual control sections provide the secondary decode of instructions and generate the actual control signals that result in the decoding and interpretation of nanorom and microrom information.

As shown in the programming models (Figures 2 and 3) the MC68020 has sixteen 32-bit general-purposed registers, a 32-bit program counter, two 32-bit supervisor

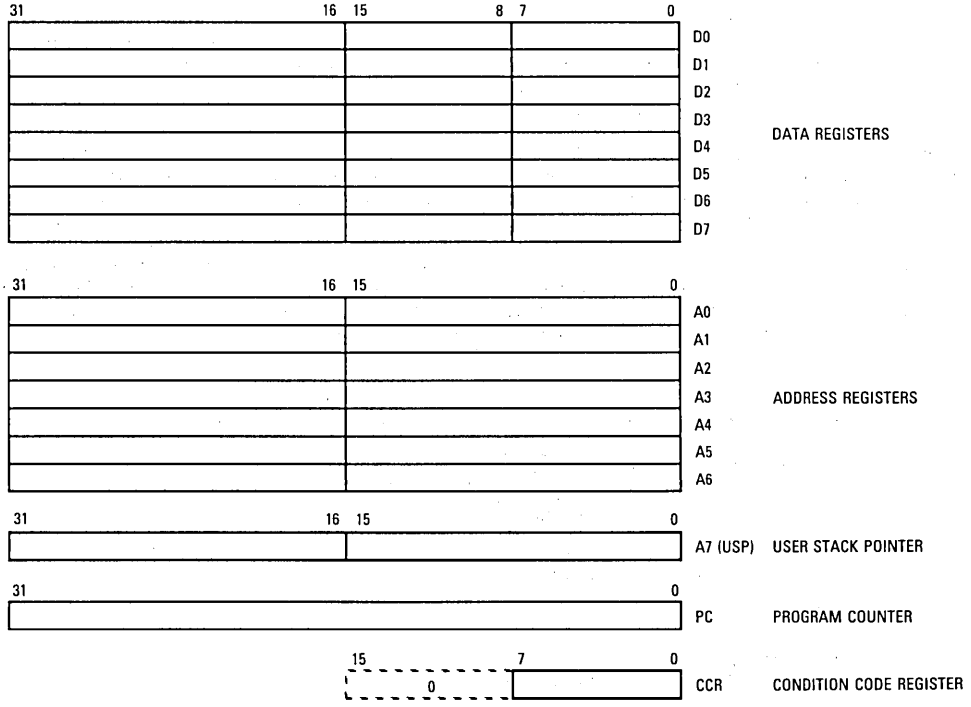


Figure 2. User Programming Model

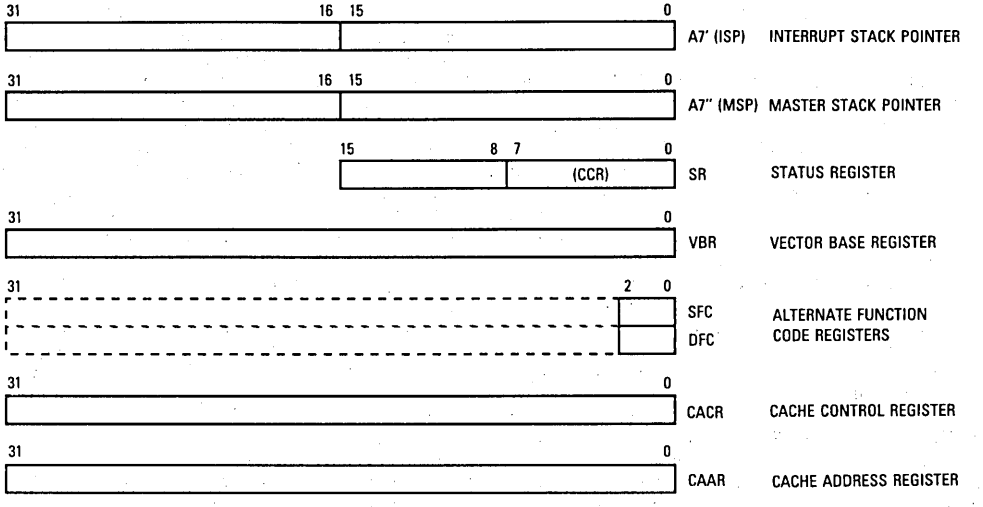


Figure 3. Supervisor Programming Model Supplement

stack pointers, a 16-bit status register, a 32-bit vector base register, two 3-bit alternate function code registers, and two 32-bit cache handling (address and control) registers. Registers D0-D7 are used as data registers for bit and bit field (1 to 32 bit), byte (8 bit), word (16 bit), long word (32 bit), and quad word (64 bit) operations. Registers A0-A6 and the user, interrupt, and master stack pointers are address registers that may be used as software stack pointers or base address registers. In addition, the address registers may be used for word and long word operations. All of the 16 (D0-D7, A0-A7) registers may be used as index registers.

The status register (Figure 4) contains the interrupt priority mask (three bits) as well as the condition codes: extend (X), negate (N), zero (Z), overflow (V), and carry (C). Additional control bits indicate that the processor is in the trace mode (T1 or T0), supervisor/user state (S), and master/interrupt state (M).

All microprocessors of the M68000 Family support instruction tracing (via the T0 status bit in the MC68020) where each instruction executed is followed by a trap to a user-defined trace routine. The MC68020 adds the capability to trace only the change of flow instructions (branch, jump, subroutine call and return, etc.) using the T1 status bit. These features are important for software program development and debug.

The vector base register is used to determine the run-time location of the exception vector table in memory, hence it supports multiple vector tables so each process or task can properly manage exceptions independent of each other.

The M68000 Family processors distinguish address spaces as supervisor/user and program/data. These four combinations are specified by the function code pins (FC0/FC1/FC2) during bus cycles, indicating the particular address space. Using the function codes, the memory subsystem can distinguish between authorized access (supervisor mode is privileged access) and unauthorized access (user mode may not have access to supervisor program or data areas). To support the full privileges of the supervisor, the alternate function code registers allow the supervisor to specify an access to user program or

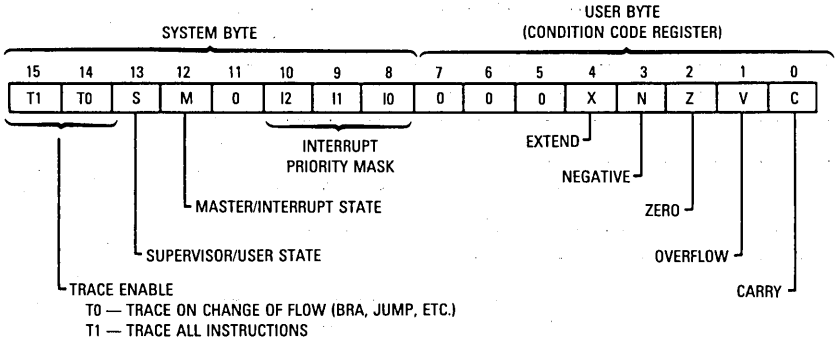


Figure 4. Status Register

data areas by preloading the SFC/DFC registers appropriately.

The cache registers (control — CACR, address — CAAR) allow software manipulation of the on-chip instruction cache. Control and status accesses to the instruction cache are provided by the cache control register (CACR), while the cache address register (CAAR) holds the address for those cache control functions that require an address.

DATA TYPES AND ADDRESSING MODES

Seven basic types are supported. These data types are:

- Bits
- Bit Fields (String of consecutive bits, 1-32 bits long)
- BCD Digits (Packed: 2 digits/byte, Unpacked: 1 digit/byte)
- Byte Integers (8 bits)
- Word Integers (16 bits)
- Long Word Integers (32 bits)
- Quad Word Integers (64 bits)

In addition, operations on other data types, such as memory addresses, status word data, etc., are provided in the instruction set. The coprocessor mechanism allows direct support of floating-point data types with the MC68881 floating-point coprocessor, as well as specialized user-defined data types and functions.

The 18 addressing modes, shown in Table 1, include nine basic types:

- Register Direct
- Register Indirect
- Register Indirect with Index
- Memory Indirect
- Program Counter Indirect with Displacement
- Program Counter Indirect with Index
- Program Counter Memory Indirect
- Absolute
- Immediate

The register indirect addressing modes support postincrement, predecrement, offset, and indexing. Programmers find these capabilities particularly useful for handling advanced data structures common to sophisticated applications and high level languages. The program counter relative mode also has index and offset capabilities; programmers find that this addressing mode is required to support position-independent software. In addition to these addressing modes, the MC68020 provides data operand sizing and scaling; these features provide performance enhancements to the programmer.

INSTRUCTION SET OVERVIEW

The MC68020 instruction set is shown in Table 2. Special emphasis has been given to the instruction set's support of structured high-level languages and sophisticated operating systems. Each instruction, with few exceptions, operates on bytes, words, and long words and most instructions can use any of the 18 addressing modes. Many instruction extensions have been made on the MC68020 to take advantage of the full 32-bit operation where, on the earlier M68000 Family members, only 8- and 16-bit values were used. The MC68020 is upward source- and object-level code compatible with the family because it supports all of the instructions that previous family members offer. Additional instructions are now provided by the MC68020 in support of its advanced features.

BIT FIELD OPERATIONS

The MC68020 supports variable length bit field operations up to 32 bits. A bit field may start in any bit position and span any address boundary for the full length of the bit field, up to the 32 bit maximum. The bit field insert (BFINS) inserts a value into a field. Bit field extract unsigned (BFEXTU) and bit field extract signed (BFEXTS) extract a unsigned or signed value from the field. BFFFO finds the first bit in a bit field that is set. To complement the M68000 bit manipulation instruction, there are bit field change, clear, set, and test instructions (BFCHG, BFCLR, BFSET, BFTST). Using the on-chip barrel shifter, the bit and bit field instructions are very fast and particularly useful in applications using packed bits and bit fields, such as graphics and communications.

BINARY CODED DECIMAL (BCD) SUPPORT

The M68000 Family supports BCD operations including add, subtract, and negation. The MC68020 adds the PACK and UNPACK operations for BCD conversions to and from binary form as well as other conversions, e.g., ASCII and EBCDIC. The PACK instruction reduces two bytes of data into a single byte while UNPACK reverses the operation.

BOUNDS CHECKING

Previous M68000 Family members offer variable bounds checking only on the upper limit of the bound. The underlying assumption is that the lower bound is zero. This is expanded on the MC68020 by providing two new instructions, CHK2 and CMP2. These instructions allow checking and comparing of ***both*** the upper and lower bounds. These instructions may be either signed or unsigned. The CMP2 instruction sets the condition codes upon completion while the CHK2 instruction, in addition to setting the condition codes, will take a system trap if either boundary condition is exceeded.

SYSTEM TRAPS

Three additions have been made to the system trap capabilities of the MC68020. The current TRAPV (trap on overflow) instruction has been expanded to a TRAPcc format where any condition code is allowed to be the trapping condition. And, the TRAPcc instruction is expanded to optionally provide one or two additional words following the trap instruction so user-specified information may be presented to the trap handler. These additional words can be used when needed to provide simple error codes or debug information for interactive runtime debugging or post-mortem program dumps. Compilers

Table 1. MC68020 Addressing Modes

Addressing Modes	Syntax
Register Direct	
Data Register Direct	Dn
Address Register Direct	An
Register Indirect	
Address Register Indirect	(An)
Address Register Indirect with Post Increment	(An)+
Address Register Indirect with Predecrement	−(An)
Address Register Indirect with Displacement	(d_{16},An)
Register Indirect with Index	
Address Register Indirect with Index (8-Bit Displacement)	(d_8,An,Xn)
Address Register Indirect with Index (Base Displacement)	(bd,An,Xn)
Memory Indirect	
Memory Indirect Post-Indexed	([bd,An],Xn,od)
Memory Indrect Pre-Indexed	([bd,An,Xn],od)
Program Counter Indirect with Displacement	(d_{16},PC)
Program Counter Indirect with Index	
PC Indirect with Index (8-Bit Displacement)	(d_8,PC,Xn)
PC Indirect with Index (Base Displacement)	(bd,PC,Xn)
Program Counter Memory Indirect	
PC Memory Indirect Post-Indexed	([bd,PC],Xn,od)
PC Memory Indirect Pre-Indexed	([bd,PC,Xn],od)
Absolute	
Absolute Short	xxx.W
Absolute Long	xxx.L
Immediate	#⟨data⟩

NOTES:
- Dn = Data Register, D0-D7
- An = Address Register, A0-A7
- d_8, d_{16} = A twos-complement, or sign-extended displacement; added as part of the effective address calculation; size is 8 (d_8) or 16 (d_{16}) bits; when omitted, assemblers use a value of zero.
- Xn = Address or data register used as an index register; form is Xn.SIZE*SCALE, where SIZE is .W or .L (indicates index register size) and SCALE is 1, 2, 4, or 8 (index register is multiplied by SCALE); use of SIZE and/or SCALE is optional.
- bd = A twos-complement base displacement; when present, size can be 16 or 32 bits.
- od = Outer displacement, added as part of effective address calculation after any memory indirection; use is optional with a size of 16 or 32 bits.
- PC = Program Counter
- ⟨data⟩ = Immediate value of 8, 16, or 32 bits
- ( ) = Effective Address
- [ ] = Use as indirect address to long word address.

may provide direction to run-time execution routines towards handling of specific conditions.

The breakpoint instruction, BKPT, is used to support the program breakpoint function for debug monitors and real-time in-circuit or hardware emulators, and the operation will be dependent on the actual system implementation. Execution of this instruction causes the MC68020 to run a breakpoint acknowledge bus cycle, with a 3-bit breakpoint identifier placed on address lines A2, A3, and A4. This 3-bit identifier permits up to eight breakpoints to be easily differentiated. The normal response to the MC68020 is an operation word (typically an instruction, originally replaced by the debugger with the breakpoint instruction) placed on the data lines by external debugger hardware and the breakpoint acknowledge cycle properly terminated. The MC68020 then executes this operation word in place of the breakpoint instruction. The debugger hardware can count the number of executions of each breakpoint and halt execution after a pre-determined number of cycles.

MULTI-PROCESSING

To further support multi-processing with the MC68020, a compare and swap instruction, CAS, has been added. This instruction makes use of the read-modify-write cycle to compare two operands and swap a third operand

Table 2. Instruction Set

Mnemonic	Description
ABCD	Add Decimal with Extend
ADD	Add
ADDA	Add Address
ADDI	Add Immediate
ADDQ	Add Quick
ADDX	Add with Extend
AND	Logical AND
ANDI	Logical AND Immediate
ASL, ASR	Arithmetic Shift Left and Right
Bcc	Branch Conditionally
BCHG	Test Bit and Change
BCLR	Test Bit and Clear
BFCHG	Test Bit Field and Change
BFCLR	Test Bit Field and Clear
BFEXTS	Signed Bit Field Extract
BFEXTU	Unsigned Bit Field Extract
BFFFO	Bit Field Find First One
BFINS	Bit Field Insert
BFSET	Test Bit Field and Set
BFTST	Test Bit Field
BKPT	Breakpoint
BRA	Branch
BSET	Test Bit and Set
BSR	Branch to Subroutine
BTST	Test Bit
CALLM	Call Module
CAS	Compare and Swap Operands
CAS2	Compare and Swap Dual Operands
CHK	Check Register Against Bound
CHK2	Check Register Against Upper and Lower Bounds
CLR	Clear
CMP	Compare
CMPA	Compare Address
CMPI	Compare Immediate
CMPM	Compare Memory to Memory
CMP2	Compare Register Against Upper and Lower Bounds
DBcc	Test Condition, Decrement and Branch
DIVS, DIVSL	Signed Divide
DIVU, DIVUL	Unsigned Divide
EOR	Logical Exclusive OR
EORI	Logical Exclusive OR Immediate
EXG	Exchange Registers
EXT, EXTB	Sign Extend
ILLEGAL	Take Illegal Instruction Trap
JMP	Jump
JSR	Jump to Subroutine
LEA	Load Effective Address
LINK	Link and Allocate
LSL,LSR	Logical Shift Left and Right
MOVE	Move
MOVEA	Move Address
MOVE CCR	Move Condition Code Register
MOVE SR	Move Status Register
MOVE USP	Move User Stack Pointer
MOVEC	Move Control Register
MOVEM	Move Multiple Registers
MOVEP	Move Peripheral
MOVEQ	Move Quck
MOVES	Move Alternate Address Sapce
MULS	Signed Multiply
MULU	Unsigned Multiply
NBCD	Negate Decimal with Extend
NEG	Negate
NEGX	Negate with Extend
NOP	No Operation
NOT	Logical Complement
OR	Logical Inclusive OR
ORI	Logical Inclusive OR Immediate
PACK	Pack BCD
PEA	Push Effective Address
RESET	Reset External Devices
ROL, ROR	Rotate Left and Right
ROXL, ROXR	Rotate with Extend Left and Right
RTD	Return and Deallocate
RTE	Return from Exception
RTM	Return from Module
RTR	Return and Restore Codes
RTS	Return from Subroutine
SBCD	Subtract Decimal with Extend
Scc	Set Conditionally
STOP	Stop
SUB	Subtract
SUBA	Subtract Address
SUBI	Subtract Immediate
SUBQ	Subtract Quick
SUBX	Subtract with Extend
SWAP	Swap Register Words
TAS	Test Operand and Set
TRAP	Trap
TRAPcc	Trap Conditionally
TRAPV	Trap on Overflow
TST	Test Operand
UNLK	Unlink
UNPK	Unpack BCD

COPROCESSOR INSTRUCTIONS

cpBCC	Branch Conditionally
cpDBcc	Test Coprocessor Condition, Decrement and Branch
cpGEN	Coprocessor General Instruction
cpRESTORE	Restore Internal State of Coprocessor
cpSAVE	Save Internal State of Coprocessor
cpScc	Set Conditionally
cpTRAPcc	Trap Conditionally

pending the results of the compare. A varient of this instruction, CAS2, performs similarly comparing dual operand pairs, and updating two operands.

These multi-processing operations are useful when using common memory to share or pass data between multiple processing elements. The read-modify-write cycle is an indivisible operand that allows reading and updating a "lock" operand used to control access to the common memory elements. The CAS2 instruction is more powerful since dual operands allow the "lock" to be checked and two values (i.e., both pointers in a doubly-linked list) to be updated according to the lock's status, all in a single operation.

3

MODULE SUPPORT

The MC68020 includes support for modules with the call module (CALLM) and return from module (RTM) instructions. The CALLM instruction references a module descriptor. This descriptor contains control information for entry into the associated module. The CALLM instruction creates a module stack frame and stores the module state in that frame. The RTM instruction recovers the previous module state from the stack frame and returns to the calling module.

The module interface also provides a mechanism for finer resolution of access control by external hardware. Although the MC68020 does not interpret the access control information, it does communicate with external hardware when the access control is to be changed, and relies on the external hardware to verify that the changes are legal.

CALLM and RTM, when used as subroutine calls and returns with proper descriptor formats, cause the MC68020 to perform the necessary actions to verify legitimate access to modules.

VIRTUAL MEMORY/MACHINE CONCEPTS

The full addressing range of the MC68020 is 4 gigabytes (4,294,967,296). However, most MC68020 systems implement a smaller physical memory. Nonetheless, by using virtual memory techniques, the system can be made to appear to have a full 4 gigabytes of physical memory available to each user program. These techniques have been used for many years in large mainframe computers and minicomputers. With the MC68020 (as with the MC68010 and MC68012), virtual memory can be fully supported in microprocessor-based systems.

In a virtual memory system, a user program can be written as though it has a large amount of memory available to it when actually only a smaller amount of memory is physically present in the system. In a similar fashion, a system provides user programs access to other devices that are not physically present in the system such as tape drives, disk drives, printers, or terminals. With proper software emulation, a physical system can be made to appear to a user program as any other M68000 computer system and the program may be given full access to all of the resources of that emulated system. Such an emulated system is called a virtual machine.

VIRTUAL MEMORY

The basic mechanism for supporting virtual memory is to provide a limited amount of high-speed physical memory that can be accessed directly by the processor while maintaining an image of a much larger "virtual" memory on secondary storage devices such as large capacity disk drives. When the processor attempts to access a location in the virtual memory map that is not resident in physical memory (referred to as a page fault), the access to that location is temporarily suspended while the necessary data is fetched from secondary storage and placed in physical memory; the suspended access is then either restarted or continued.

The MC68020 uses instruction continuation to support virtual memory. In order for the MC68020 to use instruction continuation, it stores its internal state on the supervisor stack when a bus cycle is terminated with a bus error signal. It then loads the program counter with the address of the virtual memory bus error handler from the exception vector table (entry number two) and resumes program execution at that new address. When the bus error exception handler routine has completed execution, an RTE instruction is executed which reloads the MC68020 with the internal state stored on the stack, reruns the faulted bus cycle (when required), and continues the suspended instruction.

Instruction continuation is crucial to the support of virtual I/O devices in memory-mapped input/output systems. Since the registers of a virtual device may be simulated in the memory map, an access to such a register will cause a fault and the function of the register can be emulated by software.

VIRTUAL MACHINE

A typical use for a virtual machine system is the development of software, such as an operating system, for a new machine also under development and not yet available for programming use. In such a system, a governing operating system emulates the hardware of the prototype system and allows the new operating system to be executed and debugged as though it were running on the new hardware. Since the new operating system is controlled by the governing operating system, it is executed at a lower privilege level than the governing operating system. Thus, any attempts by the new operating system to use virtual resources that are not physically present (and should be emulated) are trapped to the governing operating system and handled by its software. In the MC68020, a virtual machine is fully supported by running the new operating system in the user mode. The governing operating system executes in the supervisor mode and any attempt by the new operating system to access supervisor resources or execute privileged instructions will cause a trap to the governing operating system.

OPERAND TRANSFER MECHANISM

Though the MC68020 has a full 32-bit data bus, it offers the ability to automatically and dynamically downsize its bus to 8 or 16 bits if peripheral devices are unable to accommodate the entire 32 bits. This feature allows the

programmer the ability to write code that is not bus-width specific. For example, long word (32 bit) accesses to peripherals may be used in the code, yet the MC68020 will transfer only the amount of data that the peripheral can manage. This feature allows the peripheral to define its port size as 8, 16, or 32 bits wide and the MC68020 will dynamically size the data transfer accordingly, using multiple bus cycles when necessary. Hence, programmers are not required to program for each device port size or know the specific port size before coding; hardware designers have flexibility to choose implementations independent of software prejudices.

This is accomplished through the use of the $\overline{\text{DSACK}}$ pins and occurs on a cycle-by-cycle basis. For example, if the processor is executing an instruction that requires the reading of a long word operand, it will attempt to read 32 bits during the first bus cycle to a long word address boundary. If the port responds that it is 32 bits wide, the MC68020 latches all 32 bits of data and continues. If the port responds that it is 16 bits wide, the MC68020 latches the 16 valid bits of data and runs another cycle to obtain the other 16 bits of data. An 8-bit port is handled similarly but with four bus read cycles. Each port is fixed in assignment to particular sections of the data bus.

Justification of data on the bus is handled automatically by dynamic bus sizing. When reading 16-bit data from a 32-bit port; the data may appear on the top or bottom half of the bus, depending on the address of the data. The MC68020 determines which portion of the bus is needed to support the transfer and dynamically adjusts to read or write the data on those data lines.

The MC68020 will always transfer the maximum amount of data on all bus cycles; i.e., it always assumes the port is 32 bits wide when beginning the bus cycle. In addition, the MC68020 has no restrictions concerning alignment of operands in memory; long word operands need not be aligned on long word address boundaries. When misaligned data requires multiple bus cycles, the MC68020 automatically runs the minimum number of bus cycles.

THE COPROCESSOR CONCEPT

The coprocessor interface is a mechanism for extending the instruction set of the M68000 Family. Examples of these extensions are the addition of specialized data operands for the existing data types or, for the case of floating point, the inclusion of new data types and operations for them as implemented by the MC68881 floating-point coprocessor.

The programmer's model for the M68000 Family of microprocessors is based on sequential, non-concurrent instruction execution. This means each instruction is completely executed prior to the beginning of the next instruction. Hence, instructions do not operate concurrently in the programmer's model. Most microprocessors implement the sequential model which greatly simplifies the programmer responsibilities since sequencing control is automatic and discrete.

The M68000 coprocessor interface is designed to extend the programmers model and it provides full support for the sequential, non-concurrent instruction execution model. Hence, instruction execution by the coprocessor is assumed to not overlap with instruction execution with the main microprocessor. Yet, the M68000 coprocessor interface does allow concurrent operation when concurrency can be properly accommodated. For example, the MC68881 floating-point coprocessor will allow the MC68020 to proceed executing instructions while the MC68881 continues a floating-point operation, up to the point that the MC68020 sends another request to the MC68881. Adhering to the sequential execution model, the MC68881 completes each MC68881 instruction before it starts the next, and the MC68020 is allowed to proceed as it can in a concurrent fashion.

Coprocessors are divided into two types by their bus utilization characteristics. A coprocessor is a DMA coprocessor if it can control the bus independent of the main processor. A coprocessor is a non-DMA coprocessor if it does not have the capability of controlling the bus. Both coprocessor types utilize the same protocol and main processor resources. Implementation of a coprocessor as a DMA or non-DMA type is based primarily on bus bandwidth requirements of the coprocessor, performance, and cost issues.

The communication protocol between the main processor and the coprocessor necessary to execute a coprocessor instruction is based on a group of coprocessor interface registers (Table 3) which are defined for the M68000 Family coprocessor interface. The MC68020 hardware uses standard M68000 asynchronous bus cycles to access the registers. Thus, the coprocessor doesn't require a special bus hardware; the bus interface implemented by a coprocessor for its interface register set

Table 3. Coprocessor Interface Registers

Register	Function	R/$\overline{\text{W}}$
Response	Requests Action from CPU	R
Control	CPU Directed Control	$\overline{\text{W}}$
Save	Initiate Save of Internal State	R
Restore	Initiate Restore of Internal State	R/$\overline{\text{W}}$
Operation Word	Current Coprocessor Instruction	$\overline{\text{W}}$
Command Word	Coprocessor Specific Command	$\overline{\text{W}}$
Condition Word	Condition to be Evaluated	$\overline{\text{W}}$
Operand	32-Bit Operand	R/$\overline{\text{W}}$
Register Select	Specifies CPU Register or Mask	R
Instruction Address	Pointer to Coprocessor Instruction	R/$\overline{\text{W}}$
Operand Address	Pointer to Coprocessor Operand	R/$\overline{\text{W}}$

must only satisfy the MC68020 address, data, and control signal timing to guarantee proper communication with the main processor. The MC68020 implements the communication protocol with all coprocessors in hardware (and microcode) and handles all operations automatically so the programmer is only concerned with the instructions and data types provided by the coprocessor as extensions to the MC68020 instruction set and data types.

Other microprocessors in the M68000 Family can operate any M68000 coprocessor even though they may not have the hardware implementation of the coprocessor interface as does the MC68020. Since the coprocessor is operated through the coprocessor interface registers which are accessed via normal asynchronous bus cycles, the coprocessor may be used as a peripheral device. Software easily emulates the communication protocol by addressing the coprocessor interface registers appropriately and passing the necessary commands and operands required by the coprocessor.

The coprocessor interface registers are implemented by the coprocessor in addition to those registers implemented as extensions to the MC68020 programmer's model. For example, the MC68881 implements the coprocessor interface registers shown in Table 3 and the registers in the programming model, including eight 80-bit floating-point data registers and three 32-bit control/status registers used by the MC68881 programmer.

Up to eight coprocessors are supported in a single system with a system-unique coprocessor identifier encoded in the coprocessor instruction. When accessing a coprocessor, the MC68020 executes standard read and write bus cycles in CPU address space, as encoded by the function codes, and places the coprocessor identifier on the address bus to be used by chip-select logic to select the particular coprocessor. Since standard bus cycles are used to access the coprocessor, the coprocessor may be located according to system design requirements, whether it be located on the microprocessor local bus, on another board on the system bus, or any other place where the chip-select and coprocessor protocol using standard M68000 bus cycles can be supported.

COPROCESSOR PROTOCOL

Interprocessor transfers are all initiated by the main processor during coprocessor instruction execution. During the processing of a coprocessor instruction, the main processor transfers instruction information and data to the associated coprocessor, and receives data, requests, and status information from the coprocessor. These transfers are all based on the M68000 bus cycles.

The typical coprocessor protocol which the main processor follows is:

a) The main processor initiates the communication by writing command information to a location in the coprocessor interface.
b) The main processor reads the coprocessor response to that information.
   1) The response may indicate that the coprocessor is busy, and the main processor should again query the coprocessor. This allows the main processor and coprocessor to synchronize their concurrent operations.
   2) The response may indicate some exception condition; the main processor acknowledges the exception and begins exception processing.
   3) The response may indicate that the coprocessor needs the main processor to perform some service such as transferring data to or from the coprocessor. The coprocessor may also request that the main processor query the coprocessor again after the service is complete.
   4) The response may indicate that the main processor is not needed for further processing of the instruction. The communication is terminated, and the main processor is free to begin execution of the next instruction. At this point in the coprocessor protocol, as the main processor continues to execute the instruction stream, the main processor may operate concurrently with the coprocessor.

When the main processor encounters the next coprocessor instruction, the main processor queries the coprocessor until the coprocessor is ready; meanwhile, the main processor can go on to service interrupts and do a context switch to execute other tasks, for example.

Each coprocessor instruction type has specific requirements based on this simplified protocol. The coprocessor interface may use as many extension words as required to implement a coprocessor instruction.

PRIMITIVES/RESPONSE

The response register is the means by which the coprocessor communicates service requests to the main processor. The content of the coprocessor response register is a primitive instruction to the main processor which is read during coprocessor communication by the main processor. The main processor "executes" this primitive, thereby providing the services required by the coprocessor. Table 4 summarizes the coprocessor primitives that the MC68020 accepts.

EXCEPTIONS

KINDS OF EXCEPTIONS

Exceptions can be generated by either internal or external causes. The externally generated exceptions are the interrupts, the bus error, and reset requests. The interrupts are requests from peripheral devices for processor action while the bus error and reset pins are used for access control and processor restart. The internally generated exceptions come from instructions, address errors, tracing, or breakpoints. The TRAP, TRAPcc, TRAPV, cpTRAPcc, CHK, CHK2, and DIV instructions can all generate exceptions as part of their instruction execution. Tracing behaves like a very high priority, internally generated interrupt whenever it is processed. The other internally generated exceptions are caused by illegal instructions, instruction fetches from odd addresses, and privilege violations.

Table 4. Coprocessor Primitives

Processor Synchronization
Busy with Current Instruction
Proceed with Next Instruction, If No Trace
Service Interrupts and Re-query, If Trace Enabled
Proceed with Execution, Condition True/False
Instruction Manipulation
Transfer Operation Word
Transfer Words from Instruction Stream
Exception Handling
Take Privilege Violation if S Bit Not Set
Take Pre-Instruction Exception
Take Mid-Instruction Exception
Take Post-Instruction Exception
General Operand Transfer
Evaluate and Pass ⟨ea⟩
Evaluate ⟨ea⟩ and Transfer Data
Write to Previously Evaluated ⟨ea⟩
Take Address and Transfer Data
Transfer to/from Top of Stack
Register Transfer
Transfer CPU Register
Transfer CPU Control Register
Transfer Multiple CPU Registers
Transfer Multiple Coprocessor Registers
Transfer CPU SR and/or ScanPC

EXCEPTION PROCESSING SEQUENCE

Exception processing occurs in four steps. During the first step, an internal copy is made of the status register. After the copy is made, the special processor state bits in the status register are changed. The S bit is set, putting the processor into supervisor privilege state. Also, the T1 and T0 bits are negated, allowing the exception handler to execute unhindered by tracing. For the reset and interrupt exceptions, the interrupt priority mask is also updated.

In the second step, the vector number of the exception is determined. For interrupts, the vector number is obtained by a processor read that is classified as an interrupt acknowledge cycle. For coprocessor detected exceptions, the vector number is included in the coprocessor exception primitive response. For all other exceptions, internal logic provides the vector number. This vector number is then used to generate the address of the exception vector.

The third step is to save the current processor status. The exception stack frame is created and filled on the supervisor stack. In order to minimize the amount of machine state that is saved, various stack frame sizes are used to contain the processor state depending on the type of exception and where it occurred during instruction execution. If the exception is an interrupt and the M bit is on, the M bit is forced off, and a short four word exception stack frame is saved on the master stack which indicates that the exception is saved on the interrupt stack. If the exception is a reset, the M bit is simply forced off, and the reset vector is accessed.

The MC68020 provides an extension to the exception stacking process. If the M bit in the status register is set, the master stack pointer (MSP) is used for all task related exceptions. When a non-task related exception occurs (i.e., an interrupt), the M bit is cleared and the interrupt stack pointer (ISP) is used. This feature allows all the task's stack area to be carried within a single processor control block and new tasks may be initiated by simply reloading the master stack pointer and setting the M bit.

The fourth and last step of exception processing is the same for all exceptions. The exception vector offset is determined by multiplying the vector number by four. This offset is then added to the contents of the vector base register (VBR) to determine the memory address of the exception vector. The new program counter value is fetched from the exception vector. The instruction at the address given in the exception vector is fetched, and normal instruction decoding and execution is started.

3

ON-CHIP INSTRUCTION CACHE

Studies have shown that typical programs spend most of their execution time in a few main routines or tight loops. This phenomenon is known as locality of reference, and has an impact on performance of the program. The MC68010 takes limited advantage of this phenomenon in the form of its loop mode operation which allows certain instructions, when coupled with the DBcc instruction, to execute without the overhead of instruction fetches. In effect this is a three word cache. Although the cache hardware has been supplied in a full range of computer systems for many years, technology now allows this feature to be integrated into the microprocessor.

MC68020 CACHE GOALS

There were two primary goals for the MC68020 microprocessor cache. The first design goal was to reduce the processor external bus activity. In a given M68000 system, the MC68000 processor will use approximately 80 to 90 percent (or greater) of the available bus bandwidth. This is due to its extremely efficient prefetching algorithm and the overall speed of its internal architecture design. Thus, in an M68000 system with more than one bus master (such as a processor and DMA device) or in a multiprocessor system, performance degradation can occur due to lack of available bus bandwidth. Therefore, an important goal for an MC68020 on-chip cache was to provide a substantial increase in the total available bus bandwidth.

The second primary design goal was to increase effective CPU throughput as larger memory sizes or slower memories increased average access time. By placing a high speed cache between the processor and the rest of the memory system, the effective access time now becomes:

$$t_{acc} = h*t_{cache} + (1-h)*t_{ext}$$

where t_{acc} is the effective system access time, t_{cache} is the cache access time, t_{ext} is the access time of the rest of the system, and h is the hit ratio or the percentage of time that the data is found in the cache. Thus, for a given system design, an MC68020 on-chip cache provides a substantial CPU performance increase, or allows much slower and less expensive memories to be used for the same processor performance.

The throughput increase in the MC68020 is gained in two ways. First, the MC68020 cache is accessed in two clock cycles versus the three cycles (minimum) required for an external access. Any instruction fetch that is currently resident in the cache will provide a 33% improvement over the corresponding external access.

Second, and probably the most important benefit of the cache, is that it allows instruction stream fetches and operand accesses to proceed in parallel. For example, if the MC68020 requires both an instruction stream access and an operand access, and the instruction is resident in the cache, the operand access will proceed unimpeded rather than being queued behind the instruction fetch. Similarly, the MC68020 is fully capable of executing several internal instructions (instructions that do not require the bus) while completing an operand access for another instruction.

The MC68020 instruction cache is a 256-byte direct mapped cache organized as 64 long word entries. Each cache entry consists of a tag field made up of the upper 24 address bits, the FC2 (user/supervisor) value, one valid bit, and 32 bits of instruction data (Figure 5).

The MC68020 employs a 32-bit data bus and fetches instructions on long word address boundaries. Hence, each 32-bit instruction fetch brings in two 16-bit instruction words which are then written into the on-chip cache. When the cache is enabled, the subsequent prefetch will find the next 16-bit instruction word is already present in the cache and the related bus cycle is saved. If the cache were not enabled, the subsequent prefetch will find the bus controller still holds the full 32 bits and can satisfy the prefetch and again save the related bus cycle. So, even when the on-chip instruction cache is not enabled, the bus controller provides an instruction "cache hit" rate up to 50%.

SIGNAL DESCRIPTION

The MC68020 is offered in a 114 lead pin-grid array package (PGA). Figure 6 illustrates the functional signal groups and Table 5 lists the signals and their function.

The V_{CC} and GND pins are separated into four groups to provide individual power supply connections for the address bus buffers, data bus buffers, and all other output buffers and internal logic.

Group	V_{CC}	GND
Address Bus	A9, D3	A10, B9, C3, F12
Data Bus	M8, N8, N13	L7, L11, N7, K3
Logic	D1, D2, E3, G11, G13	G12, H13, J3, K1
Clock	—	B1

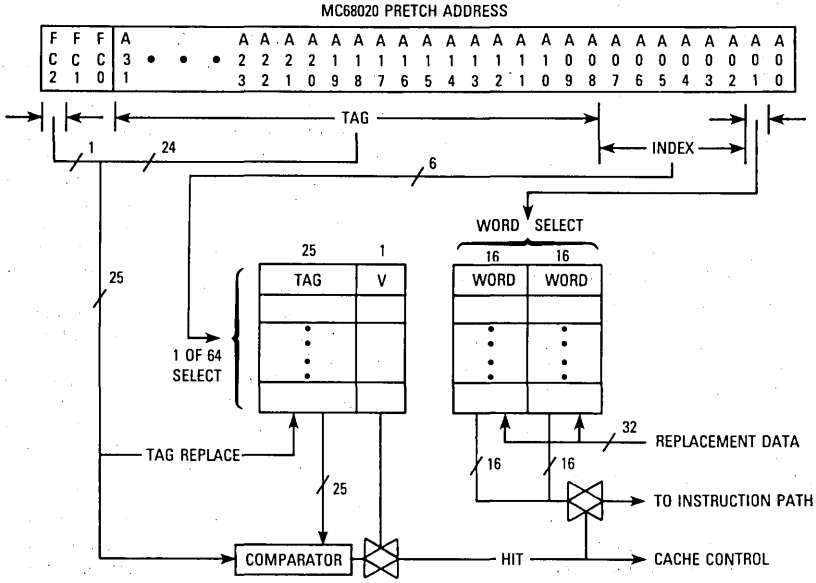


Figure 5. MC68020 On-Chip Cache Organization

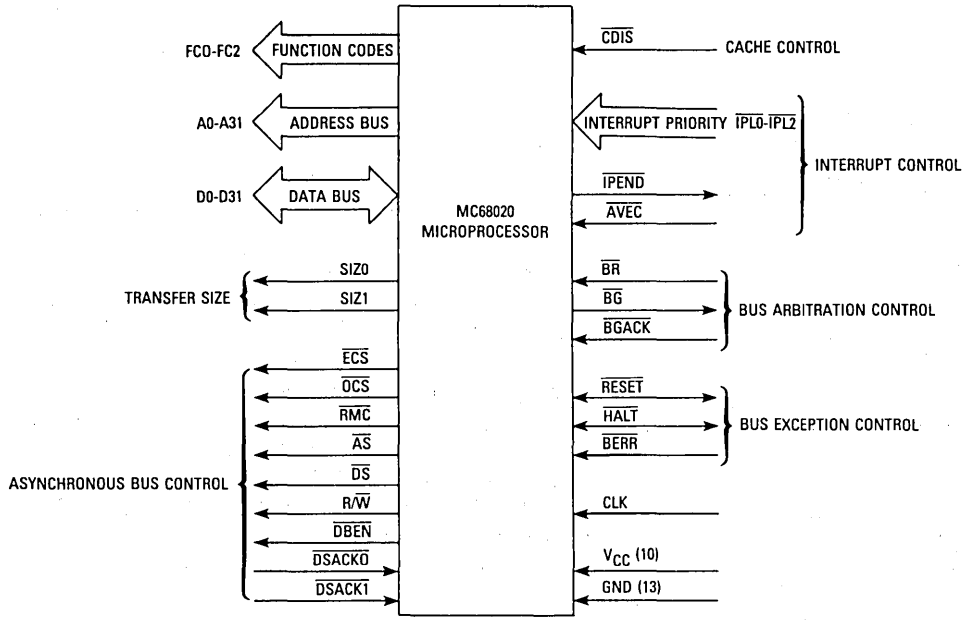


Figure 6. Functional Signal Groups

Table 5. Signal Index

Signal Name	Mnemonic	Function
Address Bus	A0-A32	32-bit address bus used to address any of 4,294,967,296 bytes.
Data Bus	D0-D31	32-bit data bus used to transfer 8, 16, 24, or 32 bits of data per bus cycle.
Function Codes	FC0-FC2	3-bit function code used to identify the address space of each bus cycle.
Size	SIZ0/SIZ1	Indicates the number of bytes remaining to be transferred for this cycle. These signals, together with A0 and A1, define the active sections of the data bus.
Read-Modify-Write Cycle	$\overline{RMC}$	Provides an indicator that the current bus cycle is part of an indivisible read-modify-write operation.
External Cycle Start	$\overline{ECS}$	Provides an indication that a bus cycle is beginning.
Operand Cycle Start	$\overline{OCS}$	Identical operation to that of $\overline{ECS}$ except that $\overline{OCS}$ is asserted only during the first bus cycle of an operand transfer.
Address Strobe	$\overline{AS}$	Indicates that a valid address is on the bus.
Data Strobe	$\overline{DS}$	Indicates that valid data is to be placed on the data bus by an external device or has been placed on the data bus by the MC68020.
Read/Write	R/$\overline{W}$	Defines the bus transfer as an MPU read or write.
Data Buffer Enable	$\overline{DBEN}$	Provides an enable signal for external data buffers.
Data Transfer and Size Acknowledge	$\overline{DSACK0}$/$\overline{DSACK1}$	Bus response signals that indicate the requested data transfer operation is completed. In addition, these two lines indicate the size of the external bus port on a cycle-by-cycle basis.
Cache Disable	$\overline{CDIS}$	Dynamically disables the on-chip cache to assist emulator support.
Interrupt Priority Level	$\overline{IPL0}$-$\overline{IPL2}$	Provides an encoded interrupt level to the processor.
Autovector	$\overline{AVEC}$	Requests an autovector during an interrupt acknowledge cycle.
Interrupt Pending	$\overline{IPEND}$	Indicates that an interrupt is pending.
Bus Request	$\overline{BR}$	Indicates that an external device requires bus mastership.
Bus Grant	$\overline{BG}$	Indicates that an external device may assume bus mastership.
Bus Grant Acknowledge	$\overline{BGACK}$	Indicates that an external device has assumed bus mastership.
Reset	$\overline{RESET}$	System reset.
Halt	$\overline{HALT}$	Indicates that the processor should suspend bus activity.
Bus Error	$\overline{BERR}$	Indicates an invalid or illegal bus operation is being attempted.
Clock	CLK	Clock input to the processor.
Power Supply	V_{CC}	+5 volt ± 5% power supply.
Ground	GND	Ground connection.

ELECTRICAL CHARACTERISTICS

MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Supply Voltage	V_{CC}	−0.3 to +7.0	V
Input Voltage	V_{in}	−0.3 to +7.0	V
Operating Temperature Range	T_A	0 to 70	°C
Storage Temperature Range	T_{stg}	−55 to 150	°C

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

THERMAL CHARACTERISTICS — PGA PACKAGE

Characteristic	Symbol	Value	Rating
Thermal Resistance — Ceramic			°C/W
Junction to Ambient	θ_{JA}	30	
Junction to Case	θ_{JC}	11	

3

POWER CONSIDERATIONS

The average chip-junction temperature, T_J, in °C can be obtained from:

$$T_J = T_A + (P_D \cdot \theta_{JA}) \quad (1)$$

where:

T_A = Ambient Temperature, °C
θ_{JA} = Package Thermal Resistance, Junction-to-Ambient, °C/W
$P_D = P_{INT} + P_{I/O}$
$P_{INT} = I_{CC} \times V_{CC}$, Watts — Chip Internal Power
$P_{I/O}$ = Power Dissipation on Input and Output Pins — User Determined

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

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

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

Solving equations (1) and (2) for K gives:

$$K = P_D \cdot (T_A + 273°C) + \theta_{JA} \cdot P_D^2 \quad (3)$$

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.

The total thermal resistance of a package (θ_{JA}) can be separated into two components, θ_{JC} and θ_{CA}, representing the barrier to heat flow from the semiconductor junction to the package (case) surface (θ_{JC}) and from the case to the outside ambient (θ_{CA}). These terms are related by the equation:

$$\theta_{JA} = \theta_{JC} + \theta_{CA} \quad (4)$$

θ_{JC} is device related and cannot be influenced by the user. However, θ_{CA} is user dependent and can be minimized by such thermal management techniques as heat sinks, ambient air cooling and thermal convection. Thus, good thermal management on the part of the user can significantly reduce θ_{CA} so that θ_{JA} approximately equals θ_{JC}. Substitution of θ_{JC} for θ_{JA} in equation (1) will result in a lower semiconductor junction temperature.

Values for thermal resistance presented in this document, unless estimated, were derived using the procedure described in Motorola Reliability Report 7843, "Thermal Resistance Measurement Method for MC68XX Microcomponent Devices," and are provided for design purposes only. Thermal measurements are complex and dependent on procedure and setup. User derived values for thermal resistance may differ.

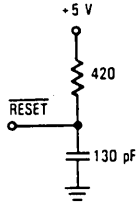


Figure 7. $\overline{\text{RESET}}$ Test Load

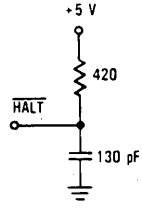


Figure 8. $\overline{\text{HALT}}$ Test Load

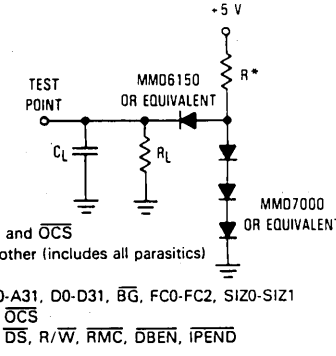


Figure 9. Test Load

DC ELECTRICAL CHARACTERISTICS (V_{CC} = 5.0 Vdc ± 5%; GND = 0 Vdc; T_A = 0 to 70°C; see Figures 7, 8, and 9)

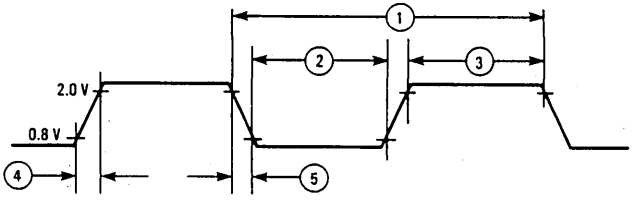
Characteristic		Symbol	Min	Max	Unit
Input High Voltage		V_{IH}	2.0	V_{CC}	V
Input Low Voltage		V_{IL}	GND −0.5	0.8	V
Input Leakage Current GND ≤ V_{in} ≤ V_{CC}	$\overline{BERR}$, $\overline{BR}$, $\overline{BGACK}$, CLK, $\overline{IPL0}$-$\overline{IPL2}$, $\overline{AVEC}$, $\overline{CDIS}$, $\overline{DSACK0}$, $\overline{DSACK1}$ $\overline{HALT}$, $\overline{RESET}$	I_{in}	−1.0 −20	1.0 20	μA
Hi-Z (Off-State) Leakage Current @ 2.4 V/0.5 V	A0-A31, $\overline{AS}$, $\overline{DBEN}$, $\overline{DS}$, D0-D31, FC0-FC2, R/$\overline{W}$, $\overline{RMC}$, SIZ0-SIZ1	I_{TSI}	−20	20	μA
Output High Voltage I_{OH} = 400 μA	A0-A31, $\overline{AS}$, $\overline{BG}$, D0-D31, $\overline{DBEN}$, $\overline{DS}$, $\overline{ECS}$, R/$\overline{W}$, $\overline{IPEND}$, $\overline{OCS}$, $\overline{RMC}$, SIZ0-SIZ1, FC0-FC2	V_{OH}	2.4	—	V
Output Low Voltage I_{OL} = 3.2 mA I_{OL} = 5.3 mA I_{OL} = 2.0 mA I_{OL} = 10.7 mA	 A0-A31, FC0-FC2, SIZ0-SIZ1, $\overline{BG}$, D0-D31 $\overline{AS}$, $\overline{DS}$, R/$\overline{W}$, $\overline{RMC}$, $\overline{DBEN}$, $\overline{IPEND}$ $\overline{ECS}$, $\overline{OCS}$ $\overline{HALT}$, $\overline{RESET}$	V_{OL}	 — — — —	 0.5 0.5 0.5 0.5	V
Power Dissipation (T_A = 0°C)	f = 25 MHz f = 33.33 MHz	P_D	—	2.0	W
Capacitance (see Note 1) V_{in} = 0 V, T_A = 25°C, f = 1 MHz		C_{in}	—	20	pF

NOTE 1. Capacitance is periodically sampled rather than 100% tested.

AC ELECTRICAL SPECIFICATIONS — CLOCK INPUT (see Figure 10)

Num.	Characteristic	12.5 MHz		16.67 MHz		20 MHz		25 MHz*		33.33 MHz		Unit
		Min	Max	Min	Max	Min	Max	Min	Max	Min	Max	
	Frequency of Operation	8	12.5	8	16.67	12.5	20	12.5	25	12.5	33.33	MHz
1	Cycle Time	80	125	60	125	50	80	40	80	30	80	ns
2, 3	Clock Pulse Width (Measured from 1.5 V to 1.5 V for 25 and 33.33 MHz)	32	87	24	95	20	54	19	61	14	66	ns
4, 5	Rise and Fall Times	—	5	—	5	—	5	—	4	—	3	ns

*These specifications represent an improvement over previously published specifications for the 25 MHz MC68020 and are valid only for product bearing date codes of 8827 and later.



NOTE:
Timing measurements are referenced to and from a low voltage of 0.8 volts and a high voltage of 2.0 volts, unless otherwise noted. The voltage swing through this range should start outside, and pass through, the range such that the rise or fall will be linear between 0.8 volts and 2.0 volts.

Figure 10. Clock Input Timing Diagram

AC ELECTRICAL SPECIFICATIONS — READ AND WRITE CYCLES (V_{CC} = 5.0 Vdc ±5%; GND = 0 Vdc; T_A = 0 to 70°C; see Figures 12, 13, and 14)

Num.	Characteristic	12.5 MHz		16.67 MHz		20 MHz		25 MHz*		33.33 MHz		Unit
		Min	Max	Min	Max	Min	Max	Min	Max	Min	Max	
6	Clock High to Address, FC, Size, $\overline{RMC}$ Valid	0	40	0	30	0	25	0	25	0	21	ns
6A	Clock High to $\overline{ECS}$, $\overline{OCS}$ Asserted	0	30	0	20	0	15	0	12	0	10	ns
7	Clock High to Address, Data, FC, Size, $\overline{RMC}$, High Impedance	0	80	0	60	0	50	0	40	0	30	ns
8	Clock High to Address, FC, Size, $\overline{RMC}$ Invalid	0	—	0	—	0	—	0	—	0	—	ns
9	Clock Low to $\overline{AS}$, $\overline{DS}$ Asserted	3	40	3	30	3	25	3	18	3	15	ns
9A[1]	$\overline{AS}$ to $\overline{DS}$ Assertion (Read) (Skew)	−20	20	−15	15	−10	10	−10	10	−10	10	ns
9B[11]	$\overline{AS}$ Asserted to $\overline{DS}$ Asserted (Write)	47	—	37	—	32	—	27	—	22	—	ns
10	$\overline{ECS}$ Width Asserted	25	—	20	—	15	—	15	—	10	—	ns
10A	$\overline{OCS}$ Width Asserted	25	—	20	—	15	—	15	—	10	—	ns
10B[7]	$\overline{ECS}$, $\overline{OCS}$ Width Negated	20	—	15	—	10	—	5	—	5	—	ns
11	Address, FC, Size, $\overline{RMC}$ Valid to $\overline{AS}$ (and $\overline{DS}$ Asserted Read)	20	—	15	—	10	—	6	—	5	—	ns
12	Clock Low to $\overline{AS}$, $\overline{DS}$ Negated	0	40	0	30	0	25	0	15	0	15	ns
12A	Clock Low to $\overline{ECS}$, $\overline{OCS}$ Negated	0	40	0	30	0	25	0	15	0	15	ns
13	$\overline{AS}$, $\overline{DS}$ Negated to Address, FC, Size, $\overline{RMC}$ Invalid	20	—	15	—	10	—	10	—	5	—	ns
14	$\overline{AS}$ (and $\overline{DS}$ Read) Width Asserted	120	—	100	—	85	—	70	—	50	—	ns
14A	$\overline{DS}$ Width Asserted Write	50	—	40	—	38	—	30	—	25	—	ns
15	$\overline{AS}$, $\overline{DS}$ Width Negated	50	—	40	—	38	—	30	—	23	—	ns
15A[8]	$\overline{DS}$ Negated to $\overline{AS}$ Asserted	45	—	35	—	30	—	25	—	18	—	ns
16	Clock High to $\overline{AS}$, $\overline{DS}$, R/$\overline{W}$, $\overline{DBEN}$ High Impedance	—	80	—	60	—	50	—	40	—	30	ns
17	$\overline{AS}$, $\overline{DS}$ Negated to R/$\overline{W}$ Invalid	20	—	15	—	10	—	10	—	5	—	ns
18	Clock High to R/$\overline{W}$ High	0	40	0	30	0	25	0	20	0	15	ns
20	Clock High to R/$\overline{W}$ Low	0	40	0	30	0	25	0	20	0	15	ns
21	R/$\overline{W}$ High to $\overline{AS}$ Asserted	20	—	15	—	10	—	5	—	5	—	ns
22	R/$\overline{W}$ Low to $\overline{DS}$ Asserted (Write)	90	—	75	—	60	—	50	—	35	—	ns
23	Clock High to Data Out Valid	—	40	—	30	—	25	—	25	—	18	ns
25	$\overline{DS}$ Negated to Data Out Invalid	20	—	15	—	10	—	5	—	5	—	ns
25A[9]	$\overline{DS}$ Negated to $\overline{DBEN}$ Negated (Write)	20	—	15	—	10	—	5	—	5	—	ns
26	Data Out Valid to $\overline{DS}$ Asserted (Write)	20	—	15	—	10	—	5	—	5	—	ns
27	Data-In Valid to Clock Low (Data Setup)	10	—	5	—	5	—	5	—	5	—	ns
27A	Late $\overline{BERR}$/$\overline{HALT}$ Asserted to Clock Low Setup Time	25	—	20	—	15	—	10	—	5	—	ns
28	$\overline{AS}$, $\overline{DS}$ Negated to $\overline{DSACKx}$, $\overline{BERR}$, $\overline{HALT}$, $\overline{AVEC}$ Negated	0	110	0	80	0	65	0	50	0	40	ns
29	$\overline{DS}$ Negated to Data-In Invalid (Data-In Hold Time)	0	—	0	—	0	—	0	—	0	—	ns
29A	$\overline{DS}$ Negated to Data-In (High Impedance)	—	80	—	60	—	50	—	40	—	30	ns
31[2]	$\overline{DSACKx}$ Asserted to Data-In Valid	—	60	—	50	—	43	—	32	—	17	ns
31A[3]	$\overline{DSACKx}$ Asserted to $\overline{DSACKx}$ Valid ($\overline{DSACK}$ Asserted Skew)	—	20	—	15	—	10	—	10	—	10	ns

AC ELECTRICAL SPECIFICATIONS — READ AND WRITE CYCLES (Continued)

Num.	Characteristic	12.5 MHz		16.67 MHz		20 MHz		25 MHz*		33.33 MHz		Unit
		Min	Max	Min	Max	Min	Max	Min	Max	Min	Max	
32	$\overline{\text{RESET}}$ Input Transition Time	—	1.5	—	1.5	—	1.5	—	1.5	—	1.5	Clks
33	Clock Low to $\overline{\text{BG}}$ Asserted	0	40	0	30	0	25	0	20	0	20	ns
34	Clock Low to $\overline{\text{BG}}$ Negated	0	40	0	30	0	25	0	20	0	20	ns
35	$\overline{\text{BR}}$ Asserted to $\overline{\text{BG}}$ Asserted ($\overline{\text{RMC}}$ Not Asserted)	1.5	3.5	1.5	3.5	1.5	3.5	1.5	3.5	1.5	3.5	Clks
37	$\overline{\text{BGACK}}$ Asserted to $\overline{\text{BG}}$ Negated	1.5	3.5	1.5	3.5	1.5	3.5	1.5	3.5	1.5	3.5	Clks
37A[6]	$\overline{\text{BGACK}}$ Asserted to $\overline{\text{BR}}$ Negated	0	1.5	0	1.5	0	1.5	0	1.5	0	1.5	Clks
39	$\overline{\text{BG}}$ Width Negated	120	—	90	—	75	—	60	—	50	—	ns
39A	$\overline{\text{BG}}$ Width Asserted	120	—	90	—	75	—	60	—	50	—	ns
40	Clock High to $\overline{\text{DBEN}}$ Asserted (Read)	0	40	0	30	0	25	0	20	0	15	ns
41	Clock Low to $\overline{\text{DBEN}}$ Negated (Read)	0	40	0	30	0	25	0	20	0	15	ns
42	Clock Low to $\overline{\text{DBEN}}$ Asserted (Write)	0	40	0	30	0	25	0	20	0	15	ns
43	Clock High to $\overline{\text{DBEN}}$ Negated (Write)	0	40	0	30	0	25	0	20	0	15	ns
44	R/$\overline{\text{W}}$ Low to $\overline{\text{DBEN}}$ Asserted (Write)	20	—	15	—	10	—	10	—	5	—	ns
45[5]	$\overline{\text{DBEN}}$ Width Asserted Read Write	80 160	— —	60 120	— —	50 100	— —	40 80	— —	30 60	— —	ns
46	R/$\overline{\text{W}}$ Width Valid (Write or Read)	180	—	150	—	125	—	100	—	75	—	ns
47A	Asynchronous Input Setup Time	10	—	5	—	5	—	5	—	5	—	ns
47B	Asynchronous Input Hold Time	20	—	15	—	15	—	10	—	10	—	ns
48[4]	$\overline{\text{DSACK}}$x Asserted to $\overline{\text{BERR}}$, $\overline{\text{HALT}}$ Asserted	—	35	—	30	—	20	—	18	—	15	ns
53	Data Out Hold from Clock High	0	—	0	—	0	—	0	—	0	—	ns
55	R/$\overline{\text{W}}$ Valid to Data Bus Impedance Change	40	—	30	—	25	—	20	—	20	—	ns
56	$\overline{\text{RESET}}$ Pulse Width (Reset Instruction)	512	—	512	—	512	—	512	—	512	—	Clks
57	$\overline{\text{BERR}}$ Negated to $\overline{\text{HALT}}$ Negated (Rerun)	0	—	0	—	0	—	0	—	0	—	ns
58[10]	$\overline{\text{BGACK}}$ Negated to Bus Driven	1	—	1	—	1	—	1	—	1	—	Clks
59[10]	$\overline{\text{BG}}$ Negated to Bus Driven	1	—	1	—	1	—	1	—	1	—	Clks

*These specifications represent an improvement over previously published specifications for the 25 MHz MC68020 and are valid only for product bearing date codes of 8827 and later.

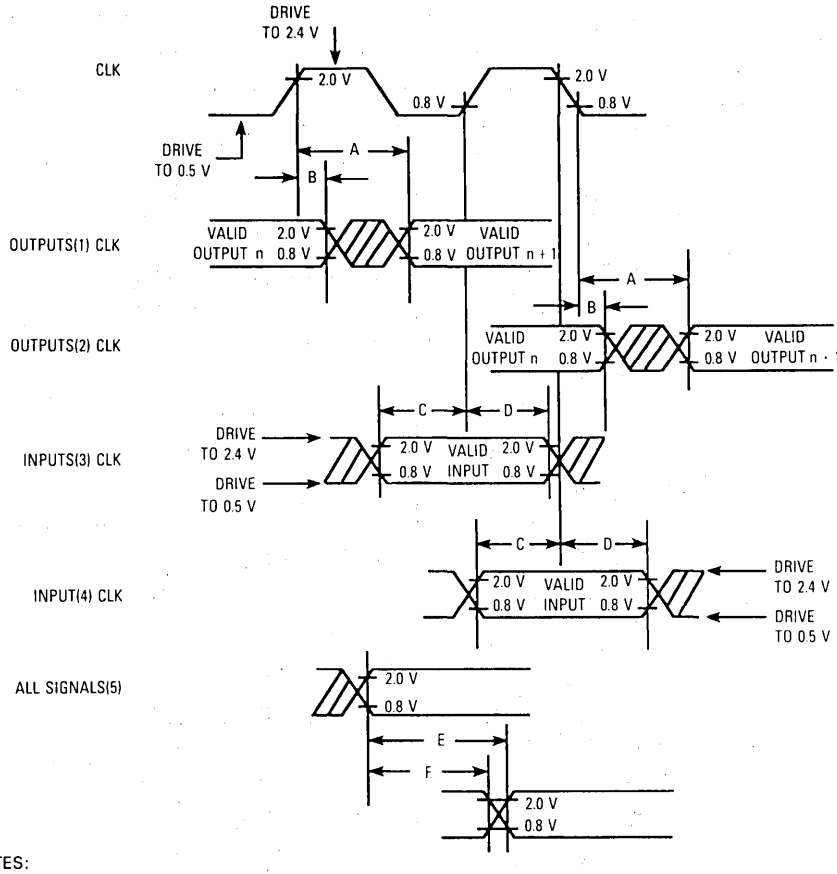
NOTES:
1. This number can be reduced to 5 nanoseconds if strobes have equal loads.
2. If the asynchronous setup time (#47A) requirements are satisfied, the $\overline{\text{DSACK}}$x low to data setup time (#31) and $\overline{\text{DSACK}}$x low to $\overline{\text{BERR}}$ low setup time (#48) can be ignored. The data must only satisfy the data-in to clock low setup time (#27) for the following clock cycle and $\overline{\text{BERR}}$ must only satisfy the late $\overline{\text{BERR}}$ low to clock low setup time (#27A) for the following clock cycle.
3. This parameter specifies the maximum allowable skew between $\overline{\text{DSACK0}}$ to $\overline{\text{DSACK1}}$ asserted or $\overline{\text{DSACK1}}$ to $\overline{\text{DSACK0}}$ asserted; specification #47A must be met by $\overline{\text{DSACK0}}$ or $\overline{\text{DSACK1}}$.
4. This specification applies to the first ($\overline{\text{DSACK0}}$ or $\overline{\text{DSACK1}}$) $\overline{\text{DSACK}}$x signal asserted. In the absence of $\overline{\text{DSACK}}$x, $\overline{\text{BERR}}$ is an asynchronous input using the asynchronous input setup time (#47A).
5. $\overline{\text{DBEN}}$ may stay asserted on consecutive write cycles.
6. The minimum values must be met to guarantee proper operation. If this maximum value is exceeded, $\overline{\text{BG}}$ may be reasserted.
7. This specification indicates the minimum high time for $\overline{\text{ECS}}$ and $\overline{\text{OCS}}$ in the event of an internal cache hit followed immediately by a cache miss or operand cycle.
8. This specification guarantees operation with the MC68881/MC68882, which specifies a minimum time for $\overline{\text{DS}}$ negated to $\overline{\text{AS}}$ asserted (specification #13A in the *MC68881/MC68882 User's Manual*). Without this specification, incorrect interpretation of specifications #9A and #15 would indicate that the MC68020 does not meet the MC68881/MC68882 requirements.
9. This specification allows a system designer to guarantee data hold times on the output side of data buffers that have output enable signals generated with $\overline{\text{DBEN}}$.
10. These specifications allow system designers to guarantee that an alternate bus master has stopped driving the bus when the MC68020 regains control of the bus after an arbitration sequence.
11. This specification allows system designers to qualify the $\overline{\text{CS}}$ signal of an MC68881/MC68882 with $\overline{\text{AS}}$ (allowing 7ns for a gate delay and still meet the $\overline{\text{CS}}$ to $\overline{\text{DS}}$ setup time requirement (specification 8B) of the MC68881/MC68882.

AC ELECTRICAL SPECIFICATIONS DEFINITIONS

The AC specifications presented consist of output delays, input setup and hold times, and signal skew times. All signals are specified relative to an appropriate edge of the MC68020 clock input and, possibly, relative to one or more other signals.

The measurement of the AC specifications is defined by the waveforms in Figure 11. In order to test the parameters guaranteed by Motorola, inputs must be driven to the voltage levels specified in Figure 11. Outputs of the MC68020 are specified with minimum and/or maximum limits, as appropriate, and are measured as shown. Inputs to the MC68020 are specified with minimum and, as appropriate, maximum setup and hold times, and are measured as shown. Finally, the measurements for signal-to-signal specifications are also shown.

Note that the testing levels used to verify conformance of the MC68020 to the AC specifications does not affect the guaranteed DC operation of the device as specified in the DC electrical characteristics.



NOTES:

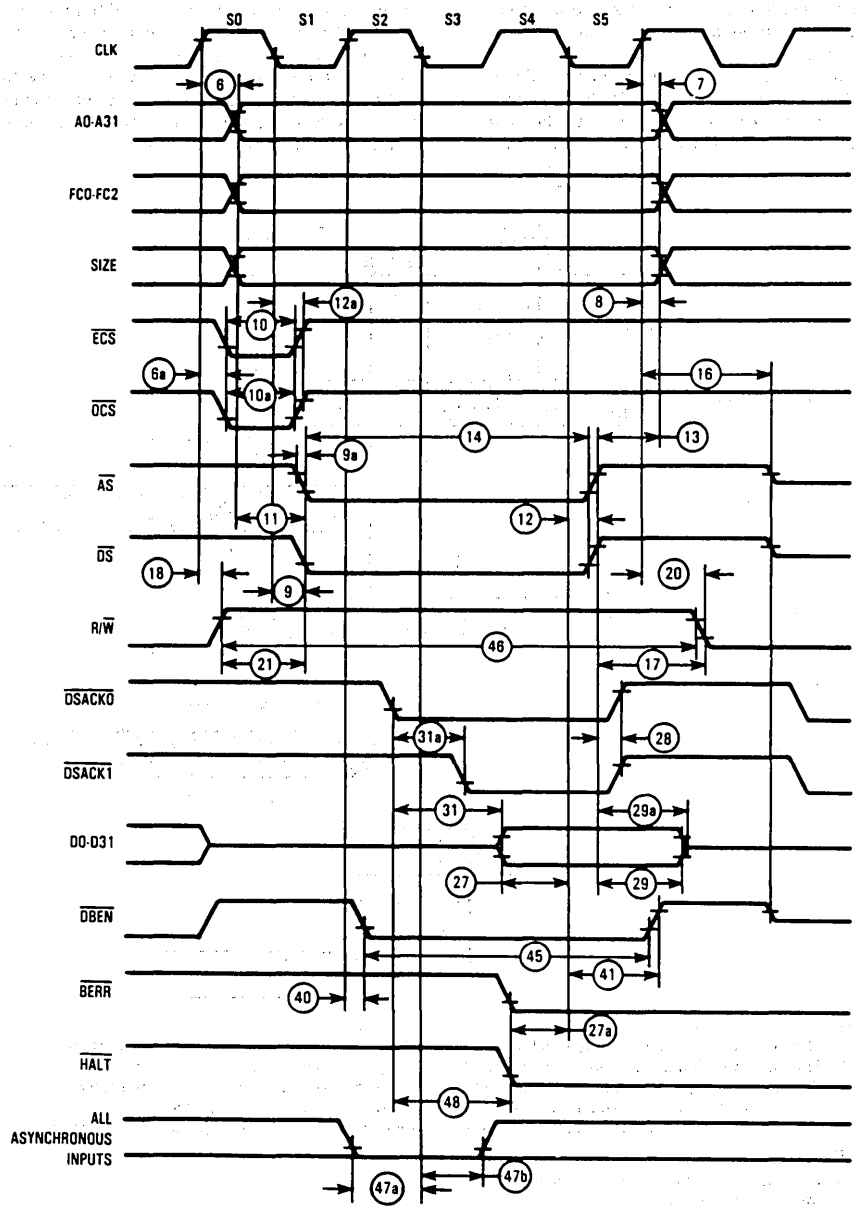
1. This output timing is applicable to all parameters specified relative to the rising edge of the clock.
2. This output timing is applicable to all parameters specified relative to the falling edge of the clock.
3. This input timing is applicable to all parameters specified relative to the rising edge of the clock.
4. This input timing is applicable to all parameters specified relative to the falling edge of the clock.
5. This timing is applicable to all parameters specified relative to the assertion/negation of another signal.

LEGEND:

A. Maximum output delay specification.
B. Minimum output hold time.
C. Minimum input setup time specification.
D. Minimum input hold time specification.
E. Signal valid to signal valid specification (maximum or minimum).
F. Signal valid to signal invalid specification (maximum or minimum).

Figure 11. Drive Levels and Test Points for AC Specifications

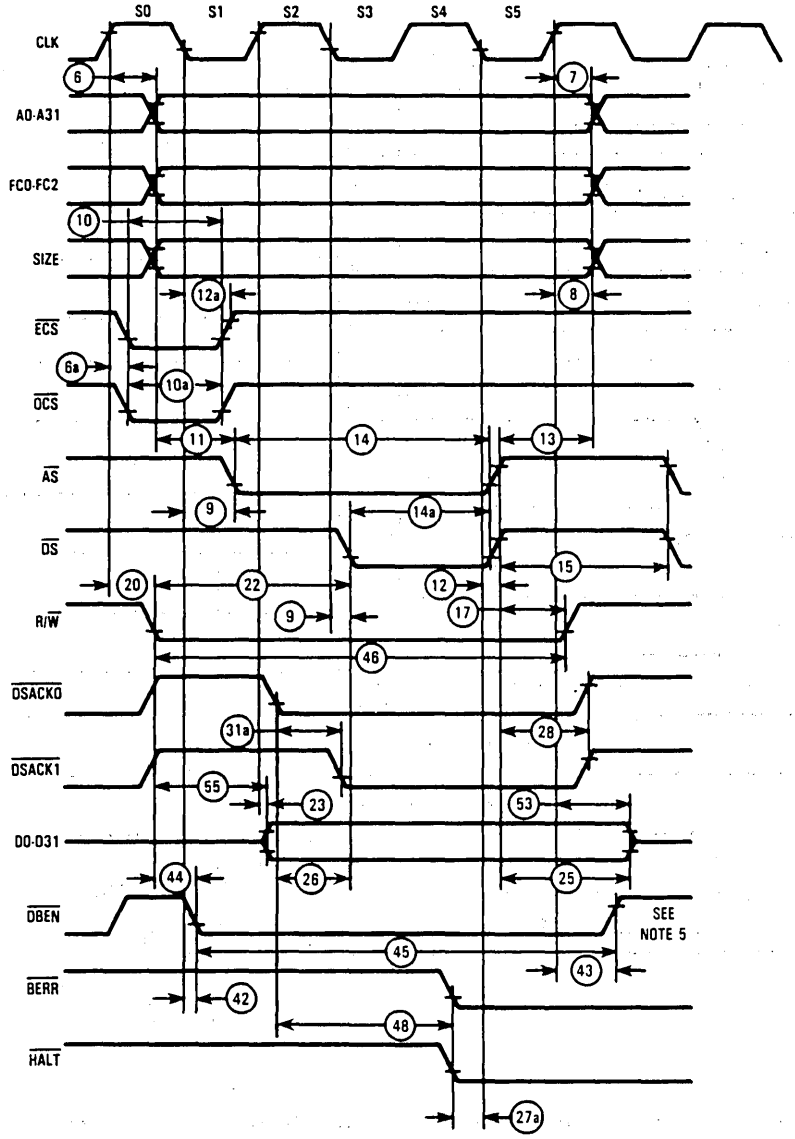
These waveforms should only be referenced in regard to the edge-to-edge measurement of the timing specifications. They are not intended as a functional description of the input and output signals. Refer to other functional descriptions and their related diagrams for device operation.



NOTE:
Timing measurements are referenced to and from a low voltage of 0.8 volts and a high voltage of 2.0 volts, unless otherwise noted. The voltage swing through this range should start outside and pass through the range such that the rise or fall will be linear between 0.8 volts and 2.0 volts.

Figure 12. Read-Cycle Timing Diagram

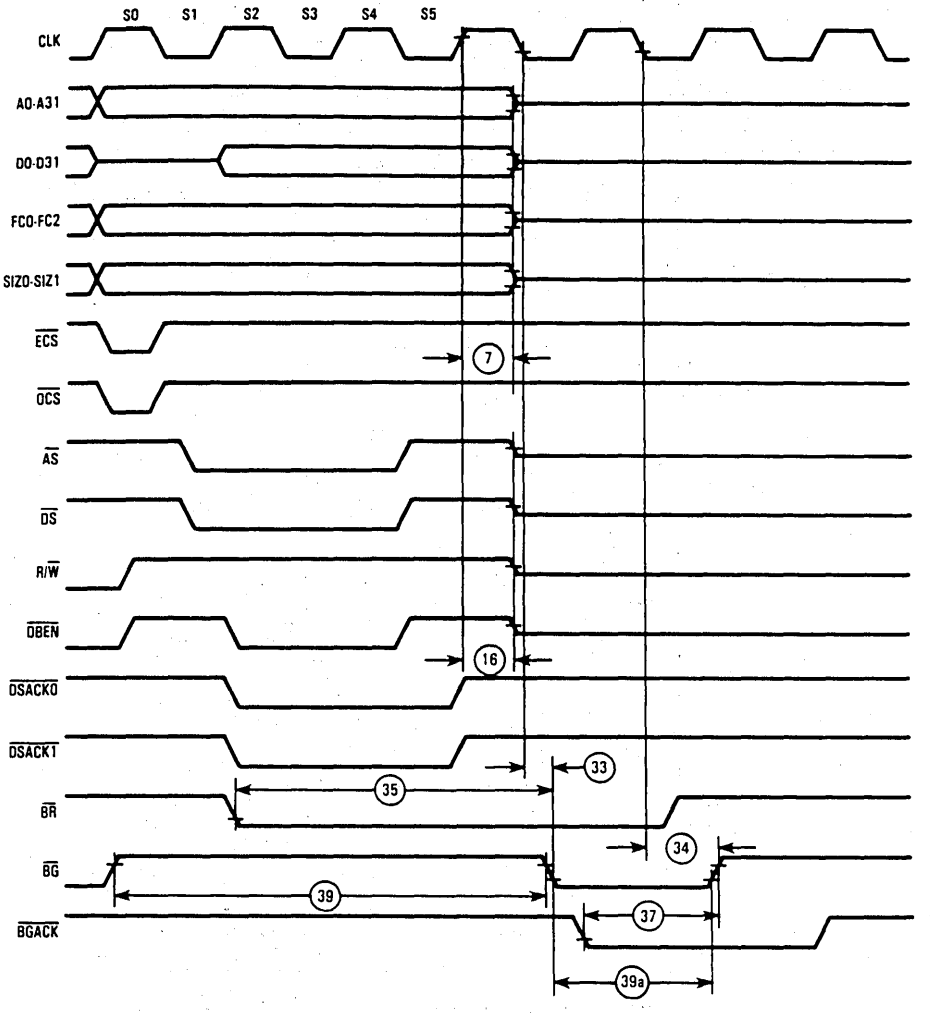
These waveforms should only be referenced in regard to the edge-to-edge measurement of the timing specifications. They are not intended as a functional description of the input and output signals. Refer to other functional descriptions and their related diagrams for device operation.



NOTE:
Timing measurements are referenced to and from a low voltage of 0.8 volts and a high voltage of 2.0 volts, unless otherwise noted. The voltage swing through this range should start outside and pass through the range such that the rise or fall will be linear between 0.8 volts and 2.0 volts.

Figure 13. Write-Cycle Timing Diagram

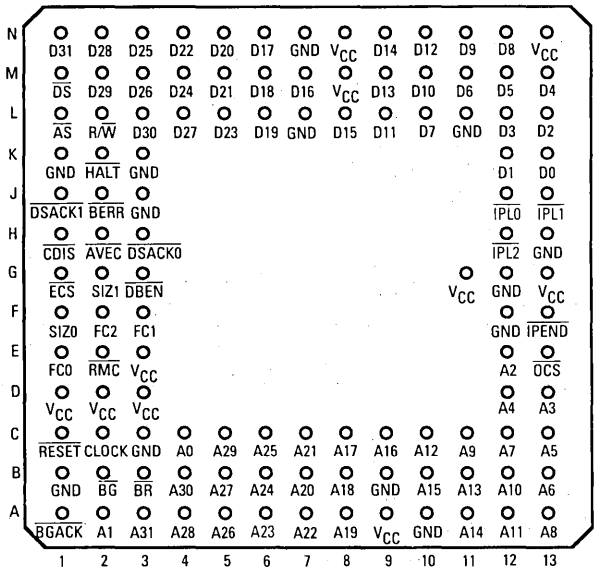
These waveforms should only be referenced in regard to the edge-to-edge measurement of the timing specifications. They are not intended as a functional description of the input and output signals. Refer to other functional descriptions and their related diagrams for device operation.



NOTE:
Timing measurements are referenced to and from a low voltage of 0.8 volts and a high voltage of 2.0 volts, unless otherwise noted. The voltage swing through this range should start outside and pass through the range such that the rise or fall will be linear between 0.8 volts and 2.0 volts.

Figure 14. Bus Arbitration Timing Diagram

PIN ASSIGNMENT



MC68030

Technical Summary

Second Generation 32-Bit Enhanced Microprocessor

3

The MC68030 is the industry's second generation 32-bit enhanced microprocessor. The MC68030 is a virtual memory microprocessor based on an MC68020 core with additional enhanced performance features. Increased internal parallelism is provided by multiple internal data buses and address buses and a versatile bus controller that supports two-clock cycle bus accesses and one-clock cycle burst accesses in order to maximize performance with paged mode, nibble mode, and static column DRAM technology. A 256-byte on-chip instruction cache in addition to a 256-byte data cache improves data flow to the execution unit and further boosts performance. On-chip paged memory management reduces the minimum physical bus cycle time to two clocks, and provides zero translation time to any bus cycle. The paged memory management structure can be enabled/disabled by software for applications not requiring the memory management feature. The rich instruction set and addressing modes of the MC68020 have been maintained allowing a clear migration path for M68000 systems.

The main features of the MC68030 are:

- Object Code Compatible with the MC68020 and Earlier M68000 Microprocessors
- Complete 32-Bit Non-Multiplexed Address and Data Buses
- Sixteen 32-Bit General Purpose Data and Address Registers
- Two 32-Bit Supervisor Stack Pointers and 10 Special Purpose Control Registers
- 256-Byte Instruction Cache and 256-Byte Data Cache that can be Accessed Simultaneously
- Paged Memory Management Unit that Translates Addresses in Parallel with Instruction Execution
- Two Transparent Segments Allow Untranslated Blocks to be Defined for Systems that Transfer Large Blocks of Data to Predefined Addresses, e.g., Graphics Applications
- Pipelined Architecture with Increased Parallelism Allows Accesses from Internal Caches to Occur in Parallel with Bus Transfers and Multiple Instructions to be Executing Concurrently
- Enhanced Bus Controller Supports Asynchronous Bus Cycles, Synchronous Bus Cycles that can Operate in Two Clocks, and Burst Data Transfers that can Operate in One Clock, all with Physical Addresses
- Dynamic Bus Sizing Supports 8-/16-/32-Bit Memories and Peripherals
- Complete Support for Coprocessors with the M68000 Coprocessor Interface
- 4-Gigabyte Direct Addressing Range
- Implemented in Motorola's HCMOS Technology that Allows CMOS and HMOS (High Density NMOS) Gates to be Combined for Maximum Speed, Low Power, and Small Die Size
- Selection of Processors Speeds: 16.67 and 20 MHz

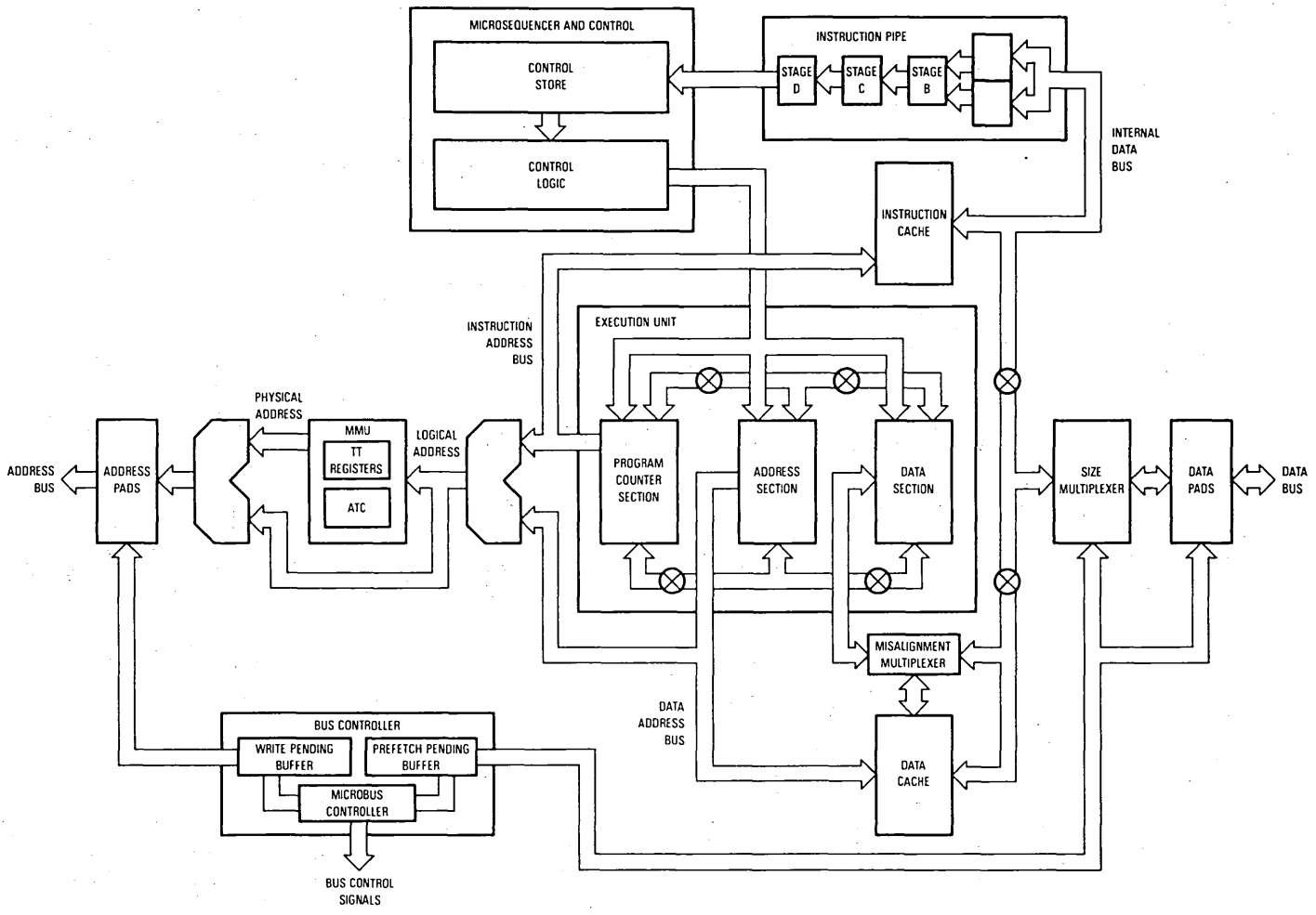


Figure 1. MC68030 Block Diagram

INTRODUCTION

The MC68030 is an enhanced 32-bit HCMOS microprocessor that incorporates the capabilities of the MC68020 MPU, an on-chip data cache, an on-chip instruction cache, an improved bus controller, multiple internal data buses, multiple internal instruction buses, and an on-chip paged memory management structure defined by the MC68851 Paged Memory Management Unit. The MC68030 maintains the 32-bit registers available with the entire M68000 Family as well as the 32-bit address and data paths, rich instruction set, versatile addressing modes, and flexible coprocessor interface provided with the MC68020. In addition, the internal operations of this integrated processor are designed to operate in parallel, allowing multiple instructions to be executed concurrently. It allows allows instruction execution to proceed in parallel with accesses to the internal caches, the on-chip memory management unit, and the bus controller.

The MC68030 fully supports the non-multiplexed asynchronous bus of the MC68020 as well as a dynamic bus sizing mechanism that allows the processor to transfer operands to or from external devices while automatically determining device port size on a cycle-by-cycle basis. In addition to the asynchronous bus, the MC68030 also supports a fast synchronous bus for off-chip caches and fast memories. Further, the MC68030 bus is capable of fetching up to four long words of data in a burst mode compatible with DRAM chips that have burst capability. Burst mode can reduce by up to 50% the time necessary to fetch the four long words. The four long words are used to prefill the on-chip instruction and data caches so that the hit ratio of the caches improves and the average access time for operand fetches is minimized.

The block diagram shown in Figure 1 depicts the major sections of the MC68030 and illustrates the autonomous nature of these blocks. The bus controller consists of the address and data pads, the multiplexors required to support dynamic bus sizing, and a macro bus controller which schedules the bus cycles on the basis of priority. The micromachine contains the execution unit and all related control logic. Microcode control is provided by a modified two-level store of microrom and nanorom contained in the micromachine. Programmed logic arrays (PLAs) are used to provide instruction decode and sequencing information. The instruction pipe and other individual control sections provide the secondary decode of instructions and generate the actual control signals that result in the decoding and interpretation of nanorom and microrom information.

The instruction and data cache blocks operate independently from the rest of the machine, storing information read by the bus controller for future use with very fast access time. Each cache resides on its own address and data buses, allowing simultaneous access to both. Both the caches are organized as 64 long word entries (256 bytes) with a block size of four long words. The data cache uses a write-through policy with no write allocation on cache misses.

Finally, the memory management unit controls the mapping of addresses for page sizes ranging from 256 bytes to 32K bytes. Mapping information stored in descriptors resides in translation tables in memory that are automatically searched by the MC68030 on demand. Recently-used descriptors are maintained in a 22-entry fully associative cache called the Address Translation Cache (ATC) allowing address translation and other MC68030 functions to occur simultaneously. Additionally, the MC68030 contains two transparent translation registers that can be used to define a one-to-one mapping for two segments ranging in size from 16M bytes to 4G bytes each.

PROGRAMMING MODEL

As shown in the programming models (Figures 2 and 3), the MC68030 has sixteen 32-bit general purpose registers, a 32-bit program counter, two 32-bit supervisor stack pointers, a 16-bit status register, a 32-bit vector base register, two 3-bit alternate function code registers, two 32-bit cache handling (address and control) registers, two 64-bit root pointer registers used by the MMU, a 32-bit translation control register, two 32-bit transparent translation registers, and a 16-bit MMU status register. Registers D0-D7 are used as data registers for bit and bit field (1 to 32 bit), byte (8 bit), word (16 bit), long word (32 bit), and quad word (64 bit) operations. Registers A0-A6 and the user, interrupt, and master stack pointers are address registers that may be used as software stack pointers or base address registers. In addition, the address registers may be used for word and long word operations. All of the 16 (D0-D7, A0-A7) registers may be used as index registers.

The status register (Figure 4) contains the interrupt priority mask (three bits) as well as the condition codes: extend (X), negate (N), zero (Z), overflow (V), and carry (C). Additional control bits indicate that the processor is in the trace mode (T1 or T0), supervisor/user state (S), and master/interrupt state (M).

All microprocessors of the M68000 Family support instruction tracing (via the T0 status bit in the MC68030) where each instruction executed is followed by a trap to a user-defined trace routine. The MC68030 also has the capability to trace only on the change of flow instructions (branch, jump, subroutine call and return, etc.) using the T1 status bit. These features are important for software program development and debug.

The vector base register is used to determine the run-time location of the exception vector table in memory, hence it supports multiple vector tables so each process or task can properly manage exceptions independent of each other.

The M68000 Family processors distinguish address spaces as supervisor/user, program/data, and CPU space. These five combinations are specified by the function code pins, FC0-FC2, during bus cycles, indicating the particular address space. Using the function codes, the memory subsystem (hardware) can distinguish between supervisor mode accesses and user accesses as well as program accesses, data accesses, and CPU space accesses. Additionally, the system software can configure the on-chip MMU so that supervisor/user privilege checking is performed by the address translation mechanism and the look-up of translation descriptors can be differentiated on the basis of function code. To support the full

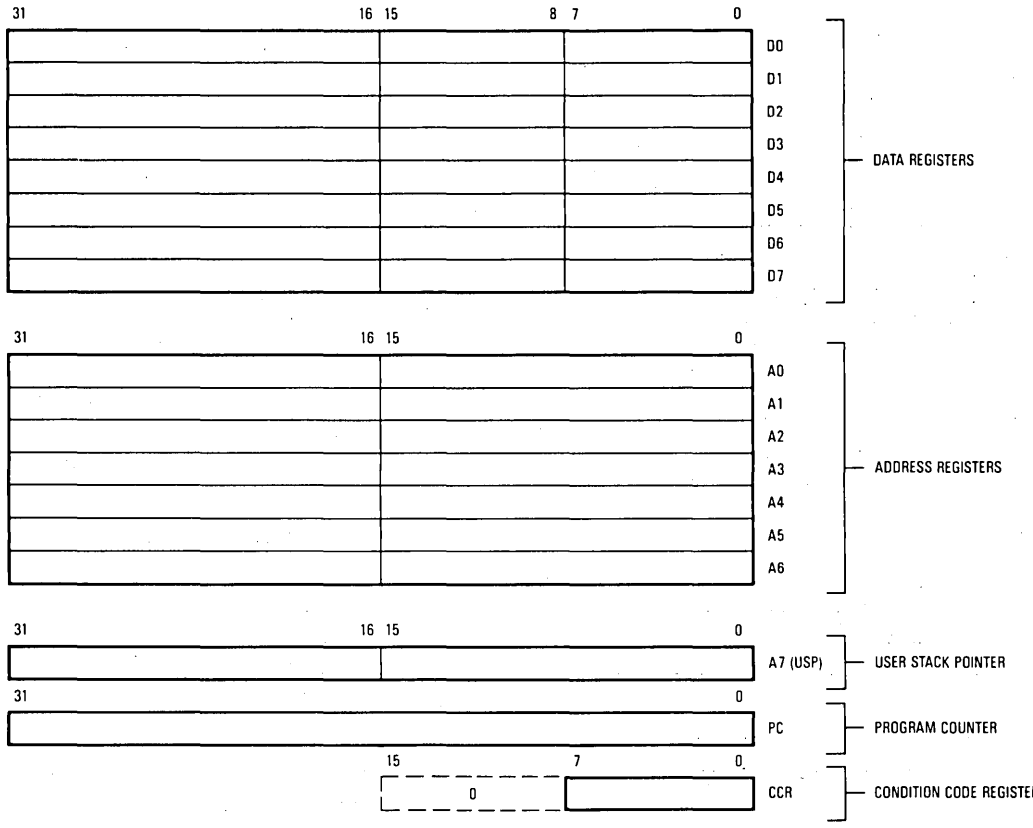


Figure 2. User Programming Model

privileges of the supervisor, the alternate function code registers allow the supervisor to specify the function code for an access by preloading the SFC/DFC registers appropriately.

The cache registers (control – CACR, address – CAAR) allow supervisor software manipulation of the on-chip instruction and data caches. Control and status accesses to the caches are provided by the cache control register (CACR), while the cache address register (CAAR) specifies the address for those cache control functions that require an address.

All of the MMU registers (CRP, SRP, TC, TT0, TT1, and PSR) are accessible by the supervisor only. The CPU root pointer contains a descriptor for the first pointer to be used in the translation table search for page descriptors pertaining to the current task. If the SRE (Supervisor Root pointer Enable) bit of the translation control register is set, the supervisor root pointer is used as a pointer to the translation tables for all supervisor accesses. If the SRE bit is clear, this register is unused and the CPU root pointer is used for both supervisor and user translations. The translation control register configures the table look-up mechanism to be used for all table searches as well as the page size and any initial shift of logical address required by the operating system. In addition, this register has an enable bit that enables the MMU. The transparent translation registers can be used to define two transparent windows for transferring large blocks of data with untranslated addresses. Finally, the MMU status register (PSR) contains status information related to a specific address translation and the results generated by the PTEST instruction. This information can be useful in locating the cause of an MMU fault.

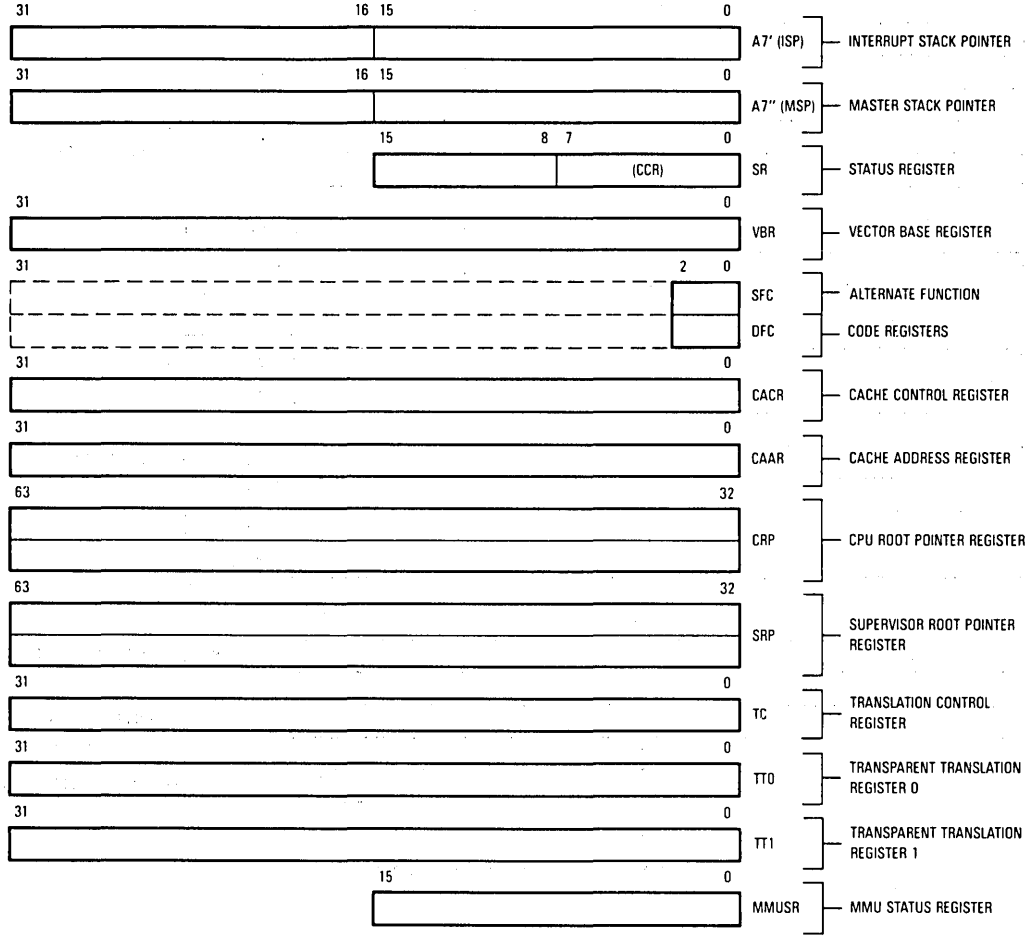


Figure 3. Supervisor Programming Model Supplement

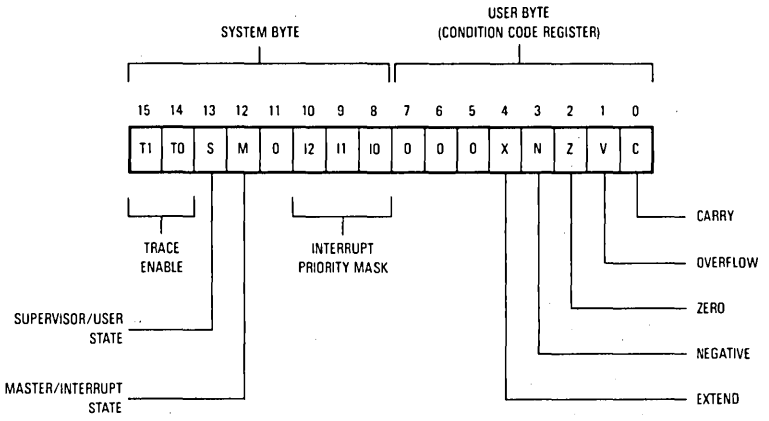


Figure 4. Status Register

DATA TYPES AND ADDRESSING MODES

Seven basic data types are supported on the MC68030. These are:

- Bits
- Bit Fields (String of consecutive bits, 1-32 bits long)
- BCD Digits (Packed: 2 digits/byte, Unpacked: 1 digit/byte)
- Byte Integers (8 bits)
- Word Integers (16 bits)
- Long Word Integers (32 bits)
- Quad Word Integers (64 bits)

In addition, operations on other data types, such as memory addresses, status word data, etc., are provided in the instruction set. The coprocessor mechanism allows direct support of floating-point data types with the MC68881 floating-point coprocessor, as well as specialized user-defined data types and functions.

The 18 addressing modes, shown in Table 1, include nine basic types:

- Register Direct
- Register Indirect
- Register Indirect with Index
- Memory Indirect
- Program Counter Indirect with Displacement
- Program Counter Indirect with Index
- Program Counter Memory Indirect
- Absolute
- Immediate

The register indirect addressing modes support postincrement, predecrement, offset, and indexing. These capabilities are particularly useful for handling advanced data structures common to sophisticated applications and high level languages. The program counter relative mode also has index and offset capabilities; programmers find that this addressing mode is required to support position-independent software. In addition to these addressing modes, the MC68030 provides index sizing and scaling; these features provide performance enhancements to the programmer.

INSTRUCTION SET OVERVIEW

The MC68030 instruction set is shown in Table 2. Each instruction, with few exceptions, operates on bytes, words, and long words, and most instructions can use any of the 18 addressing modes. The MC68030 is upward source- and object-level code compatible with the M68000 Family because it supports all of the instructions that previous family members offer. Included in this set are the bit field operations, binary coded decimal support, bounds checking, additional trap conditions, and additional mutli-processing support (CAS and CAS2 instructions) offered by the MC68020. The new instructions supported by the MC68030 are a subset of the instructions introduced by the MC68851 paged memory management unit and are used to communicate with the MMU.

INSTRUCTION AND DATA CACHES

Studies have shown that typical programs spend most of their execution time in a few main routines or tight loops. This phenomenon is known as locality of reference, and has an impact on the performance of the program. The MC68010 takes limited advantage of this phenomenon with the loop mode of operation that can be used with the DBcc instruction. The MC68020 takes much more advantage of locality with its 256-byte on-chip instruction cache. The MC68030 takes further advantage of cache technology to provide the system with two on-chip caches, one for instructions and one for data.

MC68030 CACHE GOALS

Similar to the MC68020, there were two primary goals for the MC68030 microprocessor caches. The first design goal was to reduce the processor external bus activity even more than what was accomplished with the MC68020. The second design goal was to increase effective CPU throughput as larger memory sizes or slower

Table 1. MC68030 Addressing Modes

Addressing Modes	Syntax
Register Direct	
Data Register Direct	Dn
Address Register Direct	An
Register Indirect	
Address Register Indirect	(An)
Address Register Indirect with Post Increment	(An)+
Address Register Indirect with Predecrement	−(An)
Address Register Indirect with Displacement	(d_{16},An)
Register Indirect with Index	
Address Register Indirect with Index (8-Bit Displacement)	(d_8,An,Xn)
Address Register Indirect with Index (Base Displacement)	(bd,An,Xn)
Memory Indirect	
Memory Indirect Post-Indexed	([bd,An],Xn,od)
Memory Indrect Pre-Indexed	([bd,An,Xn],od)
Program Counter Indirect with Displacement	(d_{16},PC)
Program Counter Indirect with Index	
PC Indirect with Index (8-Bit Displacement)	(d_8,PC,Xn)
PC Indirect with Index (Base Displacement)	(bd,PC,Xn)
Program Counter Memory Indirect	
PC Memory Indirect Post-Indexed	([bd,PC],Xn,od)
PC Memory Indirect Pre-Indexed	([bd,PC,Xn],od)
Absolute	
Absolute Short	xxx.W
Absolute Long	xxx.L
Immediate	#⟨data⟩

NOTES:
- Dn = Data Register, D0-D7
- An = Address Register, A0-A7
- d_8, d_{16} = A twos-complement, or sign-extended displacement; added as part of the effective address calculation; size is 8 (d_8) or 16 (d_{16}) bits; when omitted, assemblers use a value of zero.
- Xn = Address or data register used as an index register; form is Xn.SIZE*SCALE, where SIZE is .W or .L (indicates index register size) and SCALE is 1, 2, 4, or 8 (index register is multiplied by SCALE); use of SIZE and/or SCALE is optional.
- bd = A twos-complement base displacement; when present, size can be 16 or 32 bits.
- od = Outer displacement, added as part of effective address calculation after any memory indirection; use is optional with a size of 16 or 32 bits.
- PC = Program Counter
- ⟨data⟩ = Immediate value of 8, 16, or 32 bits
- ( ) = Effective Address
- [ ] = Use as indirect address to long word address.

memories increased average access time. By placing a high speed cache between the processor and the rest of the memory system, the effective memory access time becomes:

$$t_{acc} = h*t_{cache} + (1-h)*t_{ext}$$

where t_{acc} is the effective system access time, t_{cache} is the cache access time, t_{ext} is the access time of the rest of the system, and h is the hit ratio or the percentage of time that the data is found in the cache. Thus, for a given system design, two MC68030 on-chip caches provide an even more substantial CPU performance increase over that obtainable with the MC68020 with its instruction cache. Alternately, slower and less expensive memories can be used for the same processor performance.

The throughput increase in the MC68030 is gained in three ways. First, the MC68030 caches are accessed in less time than is required for external accesses, providing improvement in the access time for items residing in the cache. Second, the burst filling of the caches allows instruction and data words to be found in the on-chip caches the first time they are accessed by the micromachine, with the time required to bring those items into the cache minimized. This has the capability of lowering the average access time for items found in the caches even further.

Third, and perhaps most importantly, the autonomous nature of the caches allows instruction stream fetches, data fetches, and a third external access to all occur si-

Table 2. Instruction Set

Mnemonic	Description
ABCD	Add Decimal with Extend
ADD	Add
ADDA	Add Address
ADDI	Add Immediate
ADDQ	Add Quick
ADDX	Add with Extend
AND	Logical AND
ANDI	Logical AND Immediate
ASL, ASR	Arithmetic Shift Left and Right
Bcc	Branch Conditionally
BCHG	Test Bit and Change
BCLR	Test Bit and Clear
BFCHG	Test Bit Field and Change
BFCLR	Test Bit Field and Clear
BFEXTS	Signed Bit Field Extract
BFEXTU	Unsigned Bit Field Extract
BFFFO	Bit Field Find First One
BFINS	Bit Field Insert
BFSET	Test Bit Field and Set
BFTST	Test Bit Field
BKPT	Breakpoint
BRA	Branch
BSET	Test Bit and Set
BSR	Branch to Subroutine
BTST	Test Bit
CAS	Compare and Swap Operands
CAS2	Compare and Swap Dual Operands
CHK	Check Register Against Bound
CHK2	Check Register Against Upper and Lower Bounds
CLR	Clear
CMP	Compare
CMPA	Compare Address
CMPI	Compare Immediate
CMPM	Compare Memory to Memory
CMP2	Compare Register Against Upper and Lower Bounds
DBcc	Test Condition, Decrement and Branch
DIVS, DIVSL	Signed Divide
DIVU, DIVUL	Unsigned Divide
EOR	Logical Exclusive OR
EORI	Logical Exclusive OR Immediate
EXG	Exchange Registers
EXT, EXTB	Sign Extend
ILLEGAL	Take Illegal Instruction Trap
JMP	Jump
JSR	Jump to Subroutine
LEA	Load Effective Address
LINK	Link and Allocate
LSL, LSR	Logical Shift Left and Right
MOVE	Move
MOVEA	Move Address
MOVE CCR	Move Condition Code Register
MOVE SR	Move Status Register
MOVE USP	Move User Stack Pointer
MOVEC	Move Control Register
MOVEM	Move Multiple Registers
MOVEP	Move Peripheral
MOVEQ	Move Quck
MOVES	Move Alternate Address Sapce
MULS	Signed Multiply
MULU	Unsigned Multiply
NBCD	Negate Decimal with Extend
NEG	Negate
NEGX	Negate with Extend
NOP	No Operation
NOT	Logical Complement
OR	Logical Inclusive OR
ORI	Logical Inclusive OR Immediate
PACK	Pack BCD
PEA	Push Effective Address
RESET	Reset External Devices
ROL, ROR	Rotate Left and Right
ROXL, ROXR	Rotate with Extend Left and Right
RTD	Return and Deallocate
RTE	Return from Exception
RTM	Return from Module
RTR	Return and Restore Codes
RTS	Return from Subroutine
SBCD	Subtract Decimal with Extend
Scc	Set Conditionally
STOP	Stop
SUB	Subtract
SUBA	Subtract Address
SUBI	Subtract Immediate
SUBQ	Subtract Quick
SUBX	Subtract with Extend
SWAP	Swap Register Words
TAS	Test Operand and Set
TRAP	Trap
TRAPcc	Trap Conditionally
TRAPV	Trap on Overflow
TST	Test Operand
UNLK	Unlink
UNPK	Unpack BCD

MMU INSTRUCTIONS

PMOVE	Move to or from MMU Registers
PLOAD	Load Page Descriptor into ATC
PTEST	Test Translation
PFLUSH	Flush Selected ATC Entries
PFLUSHA	Flush Entire ATC

COPROCESSOR INSTRUCTIONS

cpBCC	Branch Conditionally
cpDBcc	Test Coprocessor Condition, Decrement and Branch
cpGEN	Coprocessor General Instruction
cpRESTORE	Restore Internal State of Coprocessor
cpSAVE	Save Internal State of Coprocessor
cpScc	Set Conditionally
cpTRAPcc	Trap Conditionally

multaneously with instruction execution. For example, if the MC68030 requires both an instruction stream access and an external peripheral access, and the instruction is resident in the on-chip cache, the peripheral access will proceed unimpeded rather than being queued behind the instruction fetch. Additionally, if a data operand was also required, and it was resident in the data cache, it ccould also be accessed without hindering either the instruction access from its cache or the peripheral access external to the chip. The parallelism designed into the MC68030 also allows multiple instructions to execute concurrently so that several insternal instructions (those that do not require any external accesses) could execute while the processor is performing an external access for a previous instruction.

INSTRUCTION CACHE

The instruction cache resident on the MC68030 is a 256-byte direct mapped cache organized as 16 blocks consisting of four long words per block. Each long word is independently accessible yielding 64 possible entries, with A1 selecting the correct word during an access. Thus each block has a tag field made up of the upper 24 address bits, the FC2 (supervisor/user) value, four valid bits (one for each long word entry) and the four long word entries (see Figure 5). The instruction cache is automatically filled by the MC68030 whenever a cache miss occurs and using the burst transfer capability, up to four long words can be filled in one burst. Neither the instruction or data caches can be manipulated directly by the programmer except by the use of the CACR register which provides cache clearing and cache entry clearing facilities. The caches can also be enabled/disabled through the use of this register. Finally, the system hardware can disable the on-chip caches at any time by the assertion of the $\overline{\text{CDIS}}$ signal.

DATA CACHE

The organization of the data cache is similar to that of the instruction cache as shown in Figure 6. However, the tag is composed of the upper 24 address bits, the four valid bits, and all three function code bits, explicitely specifying the address space associated with each block. The data cache employs a write-through policy with no write allocation of data writes. In other words, if a cache hit occurs on a write cycle, both the data cache and the external device are updated with the new data. If a write cycle generates a miss in the data cache, only the external device is updated and no data cache entry is replaced or allocated for that address.

OPERAND TRANSFER MECHANISMS

The MC68030 offers three different mechanisms by which data can be transferred into and out of the chip. Asynchronous bus cycles, compatible with the asynchronous bus on the MC68020, can transfer data in a minimum of three clock cycles and the amount of data transferred on each cycle is determined by the dynamic bus sizing mechanism on a cycle-by-cycle basis with the $\overline{\text{DSACKx}}$ signals. Synchronous bus cycles are terminated with the $\overline{\text{STERM}}$ (Synchronous Termination) signal and always transfer 32-bits of data in a minimum of two clock cycles, increasing the bus bandwidth available for other bus masters, therefore increasing possible performance. Burst mode transfers can be used to fill blocks of the instruction and data caches when the MC68030 asserts $\overline{\text{CBREQ}}$ (Cache Burst Request). After completing the first cycle with $\overline{\text{STERM}}$, subsequent cycles may accept data on every clock cycle where $\overline{\text{STERM}}$ is asserted until the burst is completed. Use of this mode can further increase the available bus bandwidth in systems that use DRAMs with page, nibble, or static column mode operation.

ASYNCHRONOUS TRANSFERS

Though the MC68030 has a full 32-bit data bus, it offers the ability to automatically and dynamically downsize its bus to 8 or 16 bits if peripheral devices are unable to accommodate the entire 32 bits. This feature allows the programmer the ability to write code that is not bus-width specific. For example, long word (32 bit) accesses to peripherals may be used in the code, yet the MC68030 will transfer only the amount of data that the peripheral can manage at one time. This feature allows the peripheral to define its port size as 8, 16, or 32 bits wide and the MC68030 will dynamically size the data transfer accordingly, using mutliple bus cycles when necessary. Hence, programmers are not required to program for each device port size or know the specific port size before coding; hardware designers have flexibility to choose implementations independent of software prejudices.

The dynamic bus sizing is invoked with the use of the $\overline{\text{DSACKx}}$ pins and occurs on a cycle-by-cycle basis. For example, if the processor is executing an instruction that requires the reading of a long word operand, it will attempt to read 32 bits during the first bus cycle to a long word address boundary. If the port responds that it is 32 bits wide, the MC68030 latches all 32 bits of data and continues. If the port responds that it is 16 bits wide, the MC68030 latches the 16 valid bits of data and runs another cycle to obtain the other 16 bits of data. An 8-bit port is handled similarly but with four bus read cycles. Each port is fixed in the assignment to particular sections of the data bus. However, the MC68030 has no restrictions concerning the alignment of operands in memory; long word operands need not be aligned to long word address boundaries. When misaligned data requires multiple bus cycles, the MC68030 automatically runs the minimum number of bus cycles. Instructions must still be aligned to word boundaries.

The timing of asynchronous bus cycles is also determined by the assertion of the $\overline{\text{DSACKx}}$ signals on a cycle-by-cycle basis. If the $\overline{\text{DSACKx}}$ signals are valid 1.5 clocks after the beginning of the bus cycle (with the appropriate setup time), the cycle terminates in its minimum amount of time corresponding to three clock cycles total. The cycle can be lengthened by delaying $\overline{\text{DSACKx}}$ (effectively inserting wait states in one clock increments) until the device being accessed is able to terminate the cycle. This flexibility gives the processor the ability to communicate with devices of varying speeds while operating at the fastest rate possible for each device.

Use of the asynchronous transfer mechanism allows external errors to abort cycles upon the assertion of $\overline{\text{BERR}}$ (Bus Error), or individual bus cycles to be retried with the

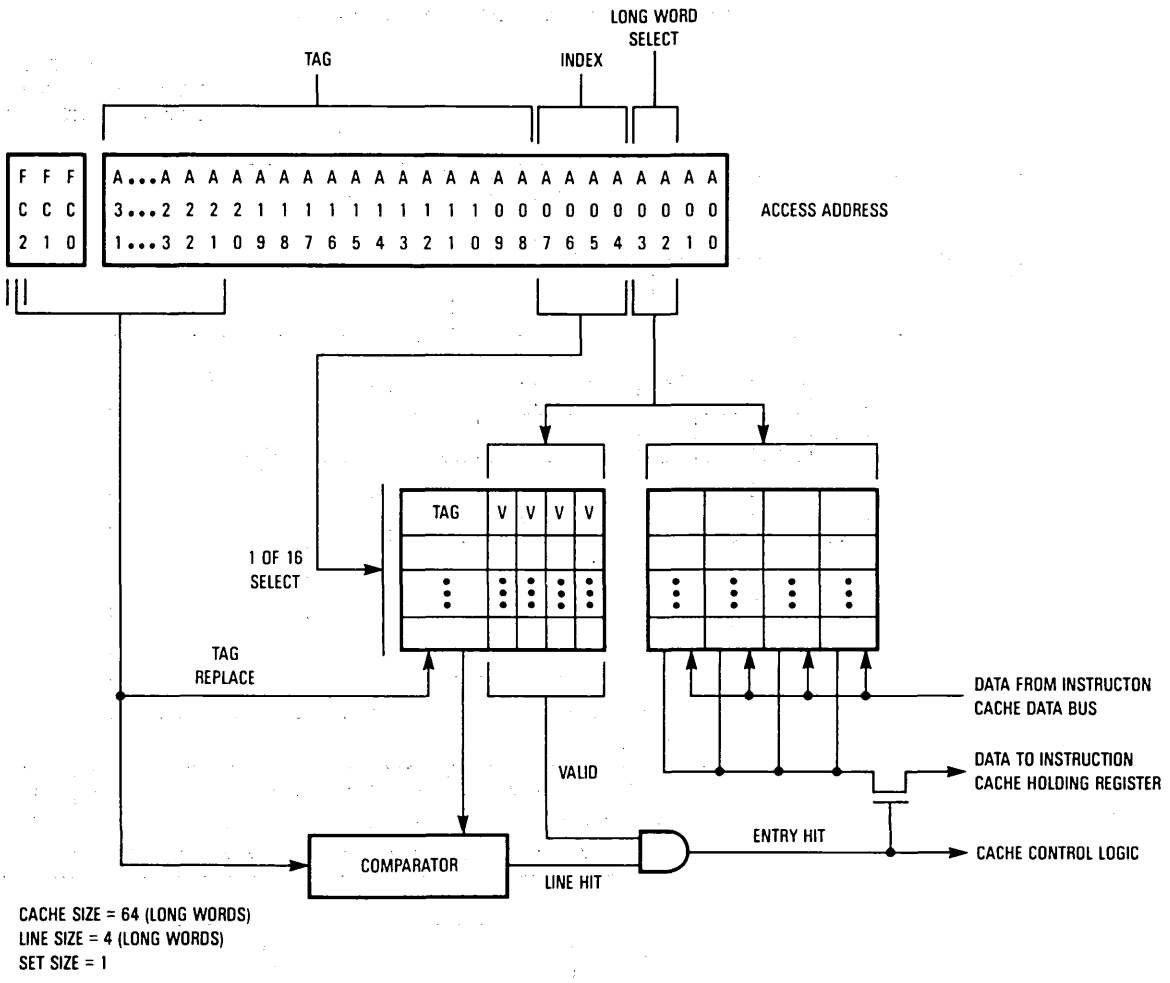


Figure 5. MC68030 On-Chip Instruction Cache Organization

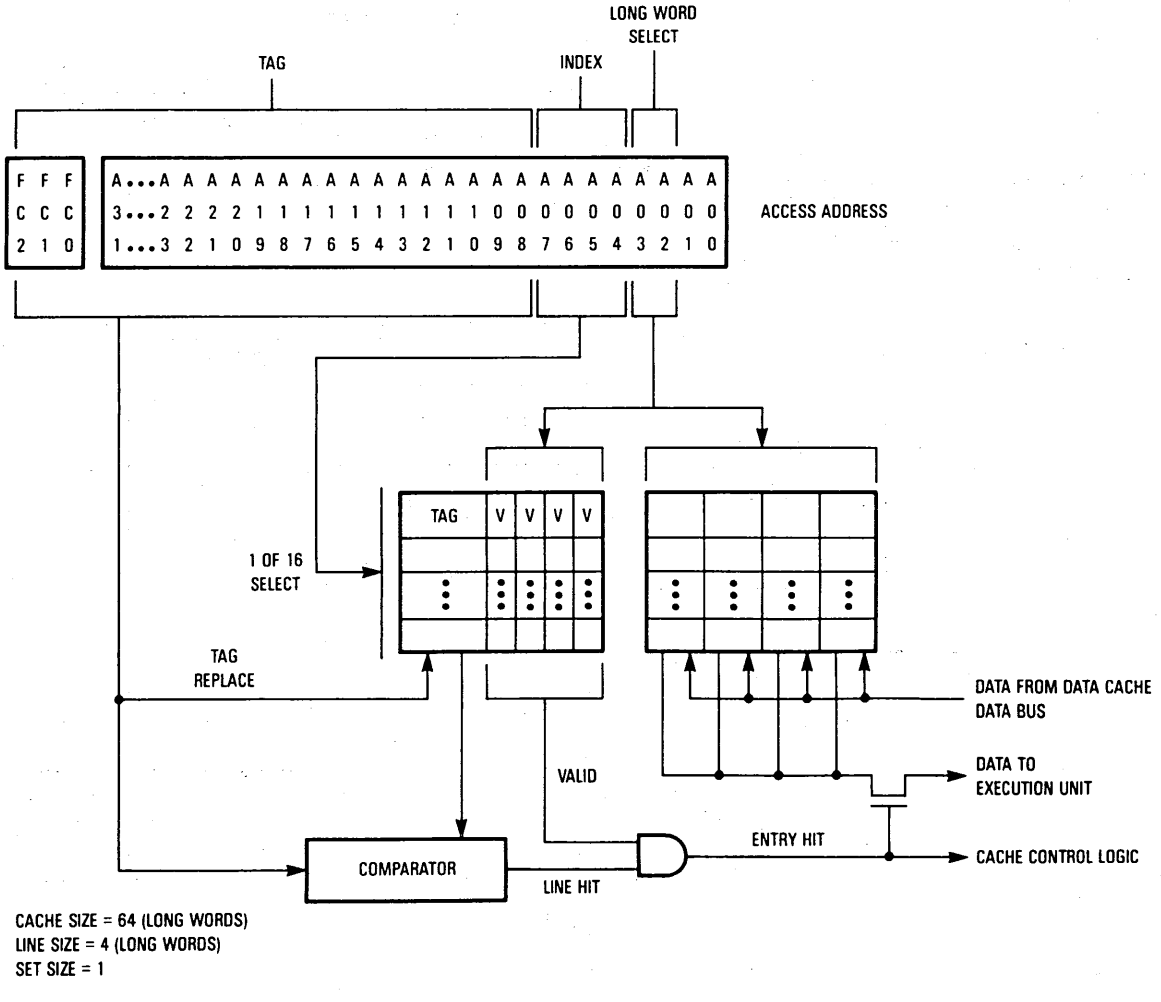


Figure 6. MC68030 On-Chip Data Cache Organization

simultaneous assertion of $\overline{BERR}$ and $\overline{HALT}$, after the $\overline{DSACKx}$ signals have been asserted.

SYNCHRONOUS TRANSFERS

Synchronous bus cycles are terminated with the assertion of the $\overline{STERM}$ signal which automatically indicates that the bus transfer is for 32 bits. This input is not synchronized internally thereby allowing two clock cycle bus accesses to be performed if the signal is valid one clock after the beginning of the bus cycle with the appropriate setup time. However, the bus cycle may be lengthened by delaying $\overline{STERM}$ (inserting wait states in one clock increments) until the device being accessed is able to terminate the cycle as in the case of asynchronous transfers. Additionally, these cycles may be aborted upon the asserting of $\overline{BERR}$, or they may be retried with the simultaneous assertion of $\overline{BERR}$ and $\overline{HALT}$, after the assertion of $\overline{STERM}$.

BURST READ CYCLES

The MC68030 provides support for burst filling of its on-chip instruction and data caches, adding to the overall system performance. The on-chip caches are organized with a block size of four long words, so that there is only one tag for the four long words in a block. Since locality of reference is present to some degree in most programs, filling of all four entries when a single entry misses can be advantageous, especially if the time spent filling the additional entries is minimal. When the caches are burst-filled, data can be latched by the processor in as little as one clock for each 32 bits.

Burst read cycles can be performed only when the MC68030 requests them (with the assertion of $\overline{CBREQ}$) and only when the bus cycles are terminated with $\overline{STERM}$ as described above. If the $\overline{CBACK}$ (Cache Burst Acknowledge) input is valid at the appropriate time in the synchronous bus cycle, the processor will keep the original $\overline{AS}$, $\overline{DS}$, R/$\overline{W}$, address, function code, and size outputs asserted and will latch 32 bits from the data bus at the end of each subsequent clock cycle that has $\overline{STERM}$ asserted. This procedure continues until the burst is complete (the entire block has been transferred), $\overline{BERR}$ is asserted in lieu of $\overline{STERM}$, or the $\overline{CBACK}$ input is negated.

EXCEPTIONS

KINDS OF EXCEPTIONS

Exceptions can be generated by either internal or external causes. The externally generated exceptions are the interrupts, the bus error, and reset requests. The interrupts are requests from peripheral devices for processor action while the bus error and reset pins are used for access control and processor restart. The internally generated exceptions come from instructions, address errors, tracing, or breakpoints. The TRAP, TRAPcc, TRAPVcc, cpTRAPcc, CKH, CKH2, and DIV instructions can all generate exceptions as part of their instruction execution. Tracing behaves like a very high priority, internally generated interrupt whenever it is processed. The other internally generated exceptions are caused by illegal instructions, instruction fetches from odd addresses, and privilege violations. Finally, the MMU can generate exceptions when it detects an invalid translation in the ATC (Address Translation Cache) and an access to the corresponding address is attempted, or when it is unable to locate a valid translation for an address in the translation tables.

EXCEPTION PROCESSING SEQUENCE

Exception processing occurs in four steps. During the first step, an internal copy is made of the status register. After the copy is made, the special processor state bits in the status register are changed. The S bit is set, putting the processor into supervisor state. Also, the T1 and T0 bits are negated, allowing the exception handler to execute unhindered by tracing. For the reset and interrupt exceptions, the interrupt priority mask is also updated.

In the second step, the vector number of the exception is determined. For interrupts, the vector number is obtrained by a processor read that is classified as an interrupt acknowledge cycle. For coprocessor detected exceptions, the vector number is included in the coprocessor exception primitive response. For all other exceptions, internal logic provides the vector number. This vector number is then used to generate the address of the exception vector.

The third step is to save the current processor status. The exception stack frame is created and filled on the supervisor stack. In order to minimize the amount of machine state that is saved, various stack frame sizes are used to contain the processor state, depending on the type of exception and where it occurred during instruction execution. If the exception is an interrupt and the M bit is on, the M bit is forced off, and the short four word exception stack frame is saved on the master stack which indicates that the exception is saved on the interrupt stack. If the exception is a reset, the M bit is simply forced off and the reset vector is accessed.

The MC68030 provides the same extensions to the exception stacking process as the MC68020. If the M bit in the status register is set, the master stack pointer (MSP) is used for all task related exceptions. When a non-task related exception occurs (i.e., an interrupt), the M bit is cleared and the interrupt stack pointer (ISP) is used. This feature allows all the task's stack area to be carried within a single processor control block and new tasks may be initiated by simply reloading the master stack pointer and setting the M bit.

The fourth and last step of exception processing is the same for all exceptions. The exception vector offset is determined by mutliplying the vector number by four. This offset is then added to the contents of the vector base register (VBR) to determine the memory address of the exception vector. The new program counter is fetched from the exception vector. The instruction at the address given in the exception vector is fetched and normal instruction decoding and execution is started.

MC68030 ON-CHIP PAGED MEMORY MANAGEMENT

The full addressing range of the MC68030 is four gigabytes (4,294,967,296 bytes). However, most MC68030 systems implement a smaller physical memory. Nonetheless, by using virtual memory techniques, the system can be made to appear to have a full four gigabytes of physical memory available to each user program. In a similar fashion, a virtual system provides user programs access to other devices that are not physically present in the system such as tape drives, disk drives, printers, or terminals. The paged Memory Management Unit (MMU) on the MC68030 provides the capability to easily support a virtual system and virtual memory. In addition, it provides protection of supervisor areas from accesses by user programs and also provides write protection on a page basis. All this capability is provided along with maximum performance as address translations occur in parallel with other processor activities. For applications not requiring the paged Memory Management Unit a register bit is used to enable/disable this feature.

DEMAND PAGED IMPLEMENTATION

A typical system with a large addressing range such as one with the MC68030 provides a limited amount of high-speed physical memory that can be accessed directly by the processor while maintaining an image of a much larger "virtual" memory on secondary storage devices such as large capacity disk drives. When the processor attempts to access a location in the virtual memory map that is not resident in physical memory, the access to that location is temporarily suspended while the necessary data is fetched from secondary storage and placed in physical memory; the suspended access is then either restarted or continued.

A paged system is one in which the physical memory is subdivided into equal sized blocks called page frames and the logical (untranslated) address space of a task is divided into pages which have the same size as the page frames. The operating system controls the allocation of pages to page frames so that when data is needed from the secondary storage device, it is brought in on a page basis. The memory management scheme employed by the MC68030 is called a "demand" implementation because a process does not need to specify in advance what areas of its logical address space it requires. An access to a logical address is interpreted by the system as a request for the corresponding page.

The MMU on the MC68030 employs the same address translation mechanism introduced by the MC68851 Paged Memory Management Unit with possible page sizes ranging from 256 bytes to 32K bytes.

TRANSLATION MECHANISM

Logical-to-physical address translation is the most frequently executed operation of the MC68030 MMU, so this task has been optimized and can function autonomously. The MMU initiates address translation by searching for a descriptor with the address translation information (a page descriptor) in the on-chip address translation cache (ATC). The ATC is a very fast fully-assocative cache memory that stores recently used page descriptors. If the descriptor does not reside in the ATC then the MMU requests external bus cycles of the bus controller to search the translation tables in physical memory. After being located, the page descriptor is loaded into the ATC and the address is correctly translated for the access, provided no exception conditions are encountered.

The status of the page in question is easily maintained in the translation tables. When a page must be brought in from a secondary storage device, the table entry can signal that this descriptor is invalid so that the table search results in an invalid descriptor being loaded into the ATC. In this way, the access to the page is aborted and the processor initiates bus error exception processing for this address. The operating system can then control the allocation of a new page in physical memory and can load the page all within the bus error handling routine.

ADDRESS TRANSLATION CACHE

An integral part of the translation function described above is the cache memory that stores recently used logical-to-physical address translation information, or page descriptors. This cache consists of 22 entries and is fully-associative. The ATC compares the logical address and function code of the incoming access against its entries. If one of the entries matches, there is a hit and the ATC sends the physical address to the bus controller, which then starts the external bus cycle (provided there was no hit in the instruction or data caches for the access).

The ATC is composed of three major components: the content-addressable memory (CAM) containing the logical address and function code information to be compared against incoming logical addresses, the physical address store that contains the physical address associated with a particular CAM entry, and the control section containing the entry replacement circuitry that implements the replacement algorithm (a variation of the least-recently-used algorithm).

TRANSLATION TABLES

The translation tables supported by the MC68030 have a tree structure, minimizing the amount of memory necessary to set up the tables for most programs, since only a portion of the complete tree needs to exist at any one time. The root of a translation table tree is pointed to by one or two root pointer registers that are part of the MC68030 programmer's model; the CPU and supervisor. Table entries at the higher levels of the tree (pointer tables) contain pointers to other tables. Entries at the leaf level (page tables) contain page descriptors. The mechanism for performing table searches uses portions of the logical address as indices for each level of the lookup. All addresses contained in the translation table entries are physical addresses.

Figure 7 illustrates the structure of the MC68030 translation tables. Several determinants of the detailed table structure are software selectable. The first level of lookup in the table normally uses the function codes as an index but this may be suppressed if desired. In addition, up to 15 of the logical address lines can be ignored for the purposes of the table searching. The number of levels in

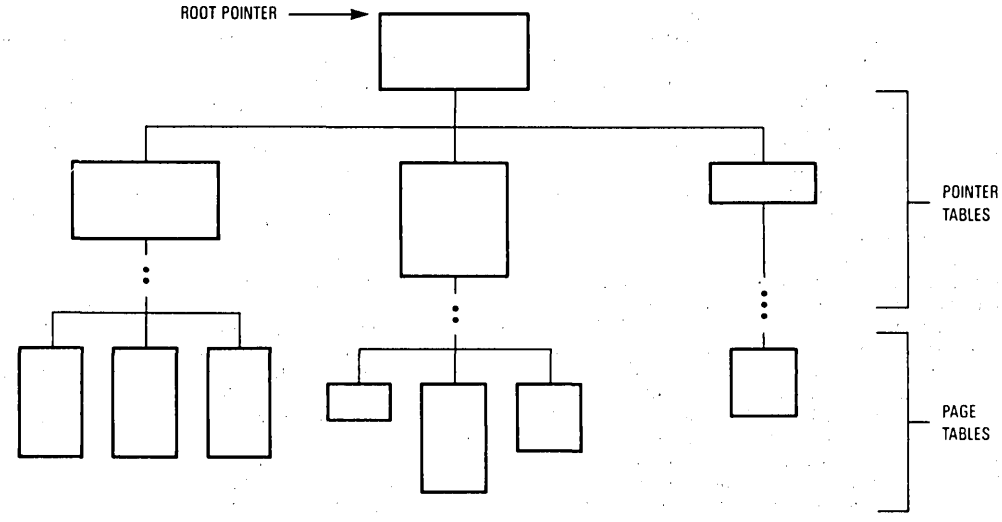


Figure 7. MMU Translation Table Tree Structure

the table indexed by the logical address can be set from one to four, and up to 15 logical address bits can be used as an index at each level. A major advantage to using this tree structure for the translation tables is the ability to deallocate large portions of the logical address space with a single entry at the higher levels of the tree. Additionally, portions of the tree itself may reside on a secondary storage device or may not exist at all until they are required by the system.

The entries in the translation tables contain status information pertaining to the pointers for the next level of lookup or the page themselves. These bits can be used to designate certain pages or blocks of pages as supervisor-only, write-protected, or non-cacheable. If a page is marked as non-cacheable, accesses within the page will not be cached by the MC68030 instruction or data caches and the $\overline{\text{CIOUT}}$ (cache inhibit out) signal is asserted for those accesses. In addition, the MMU automatically maintains history information for the pointers and pages in the descriptors via the Used (U) and Modified (M) bits.

MMU INSTRUCTIONS

The MMU instructions supported by the MC68030 are the PMOVE, PTEST, PLOAD, PFLUSH, and PFLUSHA instructions and they are completely compatible with the corresponding instructions introduced by the MC68851 PMMU. Whereas the MC68851 required the coprocessor interface to execute its instructions, the MC68030 MMU instructions execute just like all other CPU instructions. All of the MMU instructions are privileged (can be executed by the supervisor only) and are summarized below:

PMOVE Used to move data to or from MMU registers.

PTEST Takes an address and function code and searches the ATC or the translation tables for the corresponding entry. The results of the search are available in the MMU status register (PSR) and is often useful in determining the cause of a fault.

PLOAD Takes an address and function code and searches the translation tables for the corresponding page descriptor. It then loads the ATC with the appropriate information.

PFLUSH Flushes the ATC by function code or function code and logical address.

PFLUSHA Flushes all of the ATC entries.

TRANSPARENT TRANSLATION

Two transparent translation registers have been provided on the MC68030 MMU to allow portions of the logical address space to be transparently mapped and accessed without corresponding entries resident in the ATC. Each register can be used to define a range of logical addresses from 16M bytes to 4G bytes with a base address and a mask. All addresses within these ranges will not be mapped and protection is provided only on a basis of read/write and function code.

COPROCESSOR INTERFACE

The coprocessor interface is a mechanism for extending the instruction set of the M68000 Family. The interface provided on the MC68030 is the same as that on the

MC68020. Examples of these extensions are the addition of specialized data operands for the existing data types or, for the case of floating point, the inclusion of new data types and operations for them as implemented by the MC68881 and MC68882 floating-point coprocessors.

Coprocessors are divided into two types by their bus utilization characteristics. A coprocessor is a DMA coprocessor if it can control the bus independent of the main processor. A coprocessor is a non-DMA coprocessor if it does not have the capability of controlling the bus. Both coprocessor types utilize the same protocol and main processor resources. Implementation of a coprocessor as a DMA or non-DMA is based primarily on bus bandwidth requirements of the coprocessor, performance, and cost issues.

The communication protocol between the main processor and the coprocessor necessary to execute a coprocessor instruction is based on a group of coprocessor interface registers (CIRs) which have been defined for the M68000 Family (see Table 3) and are implemented on the coprocessor. The MC68030 hardware uses standard read and write cycle to access the registers. Thus the coprocessor interface doesn't require any special bus hardware; the bus interface implemented by a coprocessor for its interface register set must only satisfy the MC68030 address, data, and control signal timing to guarantee proper communication with the CPU. The MC68030 implements the communication protocol with all coprocessors in hardware (and microcode) and handles all operations automatically so the programmer is only concerned with the instructions and data types provided by the coprocessor as extensions to the MC68030 instruction set and data types.

Since the CIRs are accessed via normal read and write cycles, coprocessors can be used as peripheral devices by other M68000 Family members that do not support the coprocessor interface. The communcation protocol can be easily emulated by addressing the CIRs appropriately and passing the necessary commands and operands required by the coprocessor. In addition to the CIRs, the coprocessor contains those registers added to the MC68030 programmer's model for specific coprocessor operations. For example, the Motorola floating-point coprocessors contain the CIRs as well as eight 80-bit floating-point data registers and three 32-bit control/status registers.

Table 3. Coprocessor Interface Registers

Register	Function	R/$\overline{W}$
Response	Requests Action from CPU	R
Control	CPU Directed Control	$\overline{W}$
Save	Initiate Save of Internal State	R
Restore	Initiate Restore of Internal State	R/$\overline{W}$
Operation Word	Current Coprocessor Instruction	$\overline{W}$
Command Word	Coprocessor Specific Command	$\overline{W}$
Condition Word	Condition to be Evaluated	$\overline{W}$
Operand	32-Bit Operand	R/$\overline{W}$
Register Select	Specifies CPU Register or Mask	R
Instruction Address	Pointer to Coprocessor Instruction	R/$\overline{W}$
Operand Address	Pointer to Coprocessor Operand	R/$\overline{W}$

Up to eight coprocessors are supported in a single MC68030 system with a system-unique coprocessor identifier encoded in the coprocessor instruction. When accessing a coprocessor, the MC68030 executes standard bus cycles in CPU address space, as encoded by the function codes, and places the coprocessor identifier on the address bus to be used by chip-select logic to select the particular coprocessor. Since standard bus cycles are used, the coprocessor may be located according to system design requirements, whether it be located on the microprocessor local bus, on another board on the system bus, or any other place where the chip-select and coprocessor protocol using standard bus cycles can be supported.

COPROCESSOR PROTOCOL

Interprocessor transfers are all initiated by the main processor during coprocessor instruction execution. During the processing of a coprocessor instruction, the main processor transfers instruction information and data to the associated coprocessor, and receives data, requests, and status information from the coprocessor. These transfers are all based on standard read and write bus cycles.

The typical coprocessor protocol which the main processor follows is:

a) The main processor initiates the communication by writing command information to a location in the coprocessor interface.

b) The main processor reads the coporcessor response to that information.
   1) The response may indicate that the coprocessor is busy, and main processor should again query the coprocessor. This allows the main processor and coprocessor to synchronize their concurrent operations.
   2) The response may indicate some exception condition; the main processor acknowledges the exception and begins exception processing.
   3) The response may indicate that the coprocessor needs the main processor to perform some service such as transferring data to or from the coprocessor. The coprocessor may also request that the main processor query the coprocessor again after the service is complete.
   4) The response may indicate that the main processor is not needed for further processing of the instruction. The communication is terminated, and the main processor is free to begin execution of the next instruction. At this

point in the coprocessor protocol, as the main processor continues to execute the instruction stream, the main processor may operate concurrently with the coprocessor.

When the main processor encounters the next coporcessor instruction, the main processor queries the coprocessor until the coprocessor is ready; meanwhile, the main processor can go on to service interrupts and do a context switch to execute other tasks, for example.

Each coprocessor instruction type has specific requirements based on this simplified protocol. The coprocessor interface may use as many extension words as required to implement a coprocessor instruction.

PRIMITIVE/RESPONSE

The response register is the means by which the coprocessor communicates service requests to the main processor. The content of the coprocessor response register is a primitive instruction to the main processor which is read during coprocessor communication by the main processor. The main processor "executes" this primitive, thereby providing the services required by the coprocessor. Table 4 summarizes the coprocessor primitives that the MC68030 accepts.

Table 4. Coprocessor Primitives

Processor Synchronization Busy with Current Instruction Proceed with Next Instruction, If No Trace Service Interrupts and Re-query, If Trace Enabled Proceed with Execution, Condition True/False
Instruction Manipulation Transfer Operation Word Transfer Words from Instruction Stream
Exception Handling Take Privilege Violation if S Bit Not Set Take Pre-Instruction Exception Take Mid-Instruction Exception Take Post-Instruction Exception
General Operand Transfer Evaluate and Pass ⟨ea⟩ Evaluate ⟨ea⟩ and Transfer Data Write to Previously Evaluated ⟨ea⟩ Take Address and Transfer Data Transfer to/from Top of Stack
Register Transfer Transfer CPU Register Transfer CPU Control Register Transfer Multiple CPU Registers Transfer Multiple Coprocessor Registers Transfer CPU SR and/or ScanPC

SIGNAL DESCRIPTION

Figure 8 and Table 5 describe the signals on the MC68030 and provide an indication of their function.

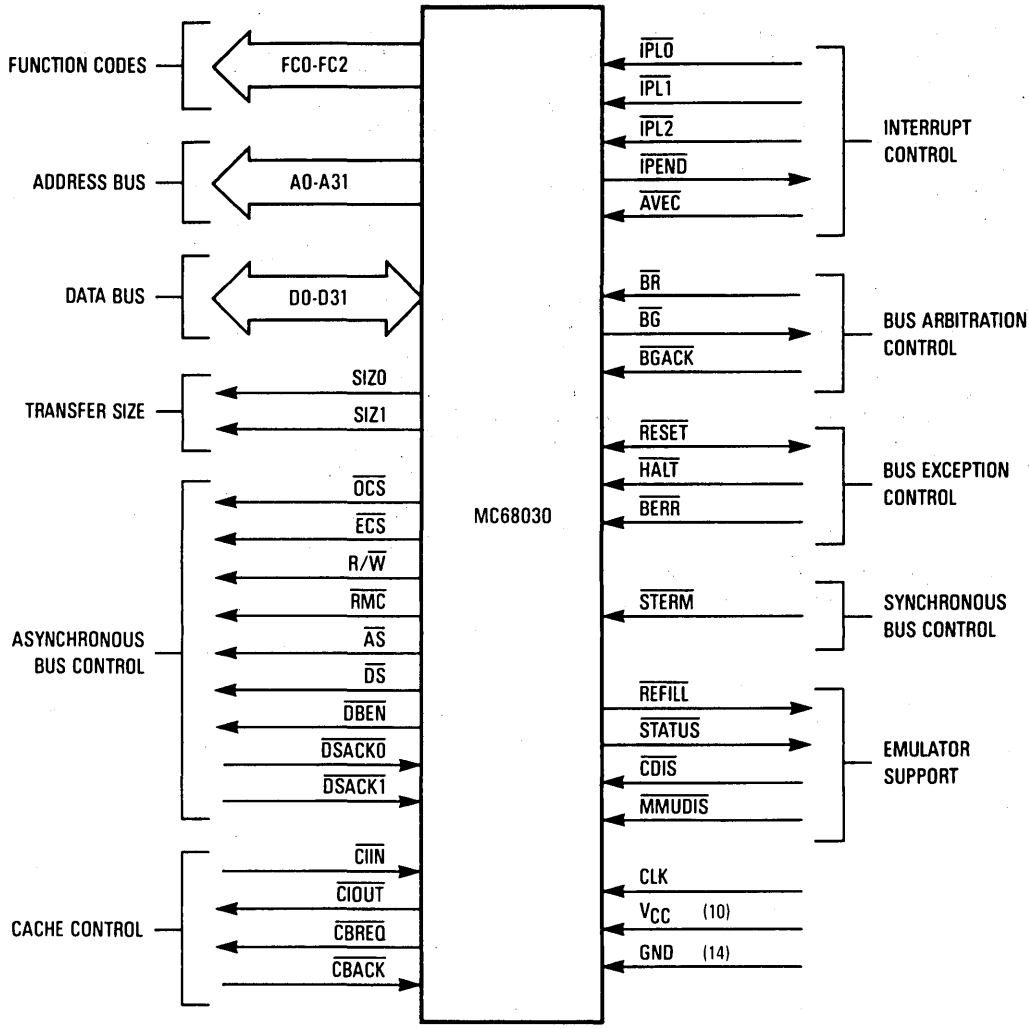


Figure 8. MC68030 Functional Signal Groups

Table 5. Signal Index

Signal Name	Mnemonic	Function
Function Codes	FC0-FC2	3-bit function code used to identify the address space of each bus cycle.
Address Bus	A0-A31	32-bit address bus used to address any of 4,294,967,296 bytes.
Data Bus	D0-D31	32-bit data bus used to transfer 8, 16, 24, or 32 bits of data per bus cycle.
Size	SIZ0/SIZ1	Indicates the number of bytes remaining to be transferred for this cycle. These signals, together with A0 and A1, define the active sections of the data bus.
Operand Cycle Start	$\overline{OCS}$	Identical operation to that of $\overline{ECS}$ except that $\overline{OCS}$ is asserted only during the first bus cycle of an operand transfer.
External Cycle Start	$\overline{ECS}$	Provides an indication that a bus cycle is beginning.
Read/Write	R/$\overline{W}$	Defines the bus transfer as an MPU read or write.
Read-Modify-Write Cycle	$\overline{RMC}$	Provides an indicator that the current bus cycle is part of an indivisible read-modify-write operation.
Address Strobe	$\overline{AS}$	Indicates that a valid address is on the bus.
Data Strobe	$\overline{DS}$	Indicates that valid data is to be placed on the data bus by an external device or has been placed on the data bus by the MC68030.
Data Buffer Enable	$\overline{DBEN}$	Provides an enable signal for external data buffers.
Data Transfer and Size Acknowledge	$\overline{DSACK0}$/$\overline{DSACK1}$	Bus response signals that indicate the requested data transfer operation is completed. In addition, these two lines indicate the size of the external bus port on a cycle-by-cycle basis.
Cache Inhibit In	$\overline{CIIN}$	Prevents data from being loaded into the MC68030 instruction and data caches.
Cache Inhibit Out	$\overline{CIOUT}$	Reflects the CI bit in ATC entries or a transparent translation register; indicates that external caches should ignore these accesses.
Cache Burst Request	$\overline{CBREQ}$	Indicates a miss in either the instruction or data cache.
Cache Burst Acknowledge	$\overline{CBACK}$	Indicates that accessed device can operate in burst mode.
Interrupt Priority Level	$\overline{IPL0}$-$\overline{IPL2}$	Provides an encoded interrupt level to the processor.
Interrupt Pending	$\overline{IPEND}$	Indicates that an interrupt is pending.
Autovector	$\overline{AVEC}$	Requests an autovector during an interrupt acknowledge cycle.
Bus Request	$\overline{BR}$	Indicates that an external device requires bus mastership.
Bus Grant	$\overline{BG}$	Indicates that an external device may assume bus mastership.
Bus Grant Acknowledge	$\overline{BGACK}$	Indicates that an external device has assumed bus mastership.
Reset	$\overline{RESET}$	System reset.
Halt	$\overline{HALT}$	Indicates that the processor should suspend bus activity.
Bus Error	$\overline{BERR}$	Indicates an invalid or illegal bus operation is being attempted.
Synchronous Termination	$\overline{STERM}$	Bus response signal that indicates a port size of 32 bits and that data may be latched on the next falling clock edge.
Cache Disable	$\overline{CDIS}$	Dynamically disables the on-chip cache to assist emulator support.
MMU Disable	$\overline{MMUDIS}$	Dynamically disables the translation mechanism of the MMU.
Microsequencer Status	$\overline{STATUS}$	Status indications for debug purposes.
Pipe Refill	$\overline{REFILL}$	Indicates when the instruction pipe is beginning to refill
Clock	CLK	Clock input to the processor.
Power Supply	V_{CC}	+5 volt ± 5% power supply.
Ground	GND	Ground connection.

ELECTRICAL CHARACTERISTICS

MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Supply Voltage	V_{CC}	−0.3 to +7.0	V
Input Voltage	V_{in}	−0.5 to +7.0	V
Operating Temperature Range	T_A	0 to 70	°C
Storage Temperature Range	T_{stg}	−55 to 150	°C

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

3

THERMAL CHARACTERISTICS — PGA PACKAGE

Characteristic	Symbol	Value	Rating
Thermal Resistance — Ceramic			°C/W
Junction to Ambient	θ_{JA}	30*	
Junction to Case	θ_{JC}	15*	

*Estimated

POWER CONSIDERATIONS

The average chip-junction temperature, T_J, in °C can be obtained from:

$$T_J = T_A + (P_D \cdot \theta_{JA}) \quad (1)$$

where:

T_A = Ambient Temperature, °C
θ_{JA} = Package Thermal Resistance, Junction-to-Ambient, °C/W
P_D = $P_{INT} + P_{I/O}$
P_{INT} = $I_{CC} \times V_{CC}$, 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.

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

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

Solving equations (1) and (2) for K gives:

$$K = P_D \cdot (T_A + 273°C) + \theta_{JA} \cdot P_D^2 \quad (3)$$

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.

The total thermal resistance of a package (θ_{JA}) can be separated into two components, θ_{JC} and θ_{CA}, representing the barrier to heat flow from the semiconductor junction to the package (case) surface (θ_{JC}) and from the case to the outside ambient (θ_{CA}). These terms are related by the equation:

$$\theta_{JA} = \theta_{JC} + \theta_{CA} \quad (4)$$

θ_{JC} is device related and cannot be influenced by the user. However, θ_{CA} is user dependent and can be minimized by such thermal management techniques as heat sinks, ambient air cooling, and thermal convention. Thus, good thermal management on the part of the user can significantly reduce θ_{CA} so that θ_{JA} approximately equals θ_{JC}. Substitution of θ_{JC} for θ_{JA} in equation (1) will result in a lower semiconductor junction temperature.

Values for thermal resistance presented in this document, unless estimated, were derived using the procedure described in Motorola Reliability Report 7843, "Thermal Resistance Measurement Method for MC68XX Microcomponent Devices," and are provided for design purposes only. Thermal measurements are complex and dependent on procedure and setup. User derived values for thermal resistance may differ.

DC ELECTRICAL CHARACTERISTICS (V_{CC} = 5.0 Vdc ± 5%; GND = 0 Vdc; T_A = 0 to 70°C; see Figures 10 and 11)

Characteristic	Symbol	Min	Max	Unit
Input High Voltage	V_{IH}	2.0	V_{CC}	V
Input Low Voltage	V_{IL}	GND −0.5	0.8	V
Input Leakage Current $GND \leq V_{in} \leq V_{CC}$ — $\overline{BERR}$, $\overline{BR}$, $\overline{BGACK}$, CLK, $\overline{IPL0}$-$\overline{IPL2}$, $\overline{AVEC}$, $\overline{CDIS}$, $\overline{DSACK0}$, $\overline{DSACK1}$, $\overline{HALT}$, $\overline{RESET}$	I_{in}	−2.5 −20	2.5 20	μA
Hi-Z (Off-State) Leakage Current @ 2.4 V/0.5 V — A0-A31, $\overline{AS}$, $\overline{DBEN}$, $\overline{DS}$, D0-D31, FC0-FC2, R/$\overline{W}$, $\overline{RMC}$, SIZ0-SIZ1	I_{TSI}	−20	20	μA
Output High Voltage I_{OH} = 400 μA — A0-A31, $\overline{AS}$, $\overline{BG}$, D0-D31, $\overline{DBEN}$, $\overline{DS}$, $\overline{ECS}$, R/$\overline{W}$, $\overline{IPEND}$, $\overline{OCS}$, $\overline{RMC}$, SIZ0-SIZ1, FC0-FC2, $\overline{CBREQ}$, $\overline{CIOUT}$, $\overline{STATUS}$, $\overline{REFILL}$	V_{OH}	2.4	—	V
Output Low Voltage	V_{OL}			V
I_{OL} = 3.2 mA — A0-A31, FC0-FC2, SIZ0-SIZ1, $\overline{BG}$, D0-D31		—	0.5	
I_{OL} = 5.3 mA — $\overline{CBREQ}$, $\overline{AS}$, $\overline{DS}$, R/$\overline{W}$, $\overline{RMC}$, $\overline{DBEN}$, $\overline{IPEND}$		—	0.5	
I_{OL} = 2.0 mA — $\overline{STATUS}$, $\overline{REFILL}$, $\overline{CIOUT}$, $\overline{ECS}$, $\overline{OCS}$		—	0.5	
I_{OL} = 10.7 mA — $\overline{RESET}$		—	0.5	
Power Dissipation (T_A = 0°C)	P_D	—	2.6	W
Capacitance (see Note) V_{in} = 0 V, T_A = 25°C, f = 1 MHz	C_{in}	—	20	pF
Load Capacitance — $\overline{EC5}$, $\overline{OCS}$	C_L	—	50	pF
$\overline{CIOUT}$, $\overline{STATUS}$, $\overline{REFILL}$			70	
All Other			130	

NOTE: Capacitance is periodically sampled rather than 100% tested.

AC ELECTRICAL SPECIFICATIONS — CLOCK INPUT (see Figure 12)

Num.	Characteristic	16.67 MHz		20 MHz		25 MHz		Unit
		Min	Max	Min	Max	Min	Max	
	Frequency of Operation	12.5	16.67	12.5	20	12.5	25	MHz
1	Cycle Time Clock	60	125	50	80	40	80	ns
2, 3	Clock Pulse Width Measured from 1.5 V to 1.5 V	25	55	23	57	19	61	ns
4, 5	Clock Rise and Fall Times	—	5	—	5	—	4	ns

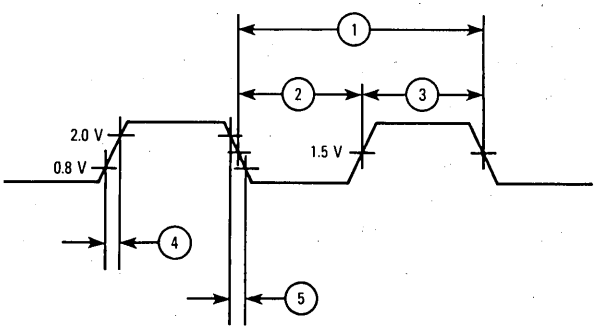


Figure 9. Clock Input Timing Diagram

PRELIMINARY AC ELECTRICAL SPECIFICATIONS — READ AND WRITE CYCLES (V_{CC} = 5.0 Vdc ± 5%; GND = 0 Vdc; T_A = 0 to 70°C; see Figures 11 through 16)

Num.	Characteristic	16.67 MHz		20 MHz		25 MHz		Unit
		Min	Max	Min	Max	Min	Max	
6	Clock High to Function Code, Size, $\overline{RMC}$, $\overline{IPEND}$, $\overline{CIOUT}$, Address Valid	0	30	0	25	0	20	ns
6A	Clock High to $\overline{ECS}$, $\overline{OCS}$ Asserted	0	20	0	15	0	15	ns
6B	Address, Function Code Valid to Negating Edge of $\overline{ECS}$	5	—	4	—	3	—	ns
7	Clock High to Function Code, Size, $\overline{RMC}$, $\overline{CIOUT}$, Address, Data High Impedance	0	60	0	50	0	40	ns
8	Clock High to Function Code, Size, $\overline{RMC}$, $\overline{IPEND}$, $\overline{CIOUT}$, Address Invalid	0	—	0	—	0	—	ns
9	Clock Low to $\overline{AS}$, $\overline{DS}$ Asserted, $\overline{CBREQ}$ Valid	3	30	3	20	3	18	ns
9A[1]	$\overline{AS}$ to $\overline{DS}$ Assertion Skew (Read)	−15	15	−10	10	−10	10	ns
9B[14]	$\overline{AS}$ Asserted to $\overline{DS}$ Asserted (Write)	37	—	32	—	27	—	ns
10	$\overline{ECS}$ Width Asserted	20	—	15	—	10	—	ns
10A	$\overline{OCS}$ Width Asserted	20	—	15	—	10	—	ns
10B[7]	$\overline{ECS}$, $\overline{OCS}$ Width Negated	15	—	10	—	5	—	ns
11[6]	Function Code, Size, $\overline{RMC}$, $\overline{CIOUT}$, Address Valid to $\overline{AS}$ Asserted (and $\overline{DS}$ Asserted, Read)	15	—	10	—	7	—	ns
12	Clock Low to $\overline{AS}$, $\overline{DS}$ Negated	0	30	0	20	0	18	ns
12A	Clock Low to $\overline{ECS}$/$\overline{OCS}$ Negated	0	30	0	20	0	18	ns
13	$\overline{AS}$, $\overline{DS}$ Negated to Function Code, Size, $\overline{RMC}$ $\overline{CIOUT}$, Address Invalid	15	—	10	—	7	—	ns
14	$\overline{AS}$ (and $\overline{DS}$ Read) Width Asserted (Asynchronous Cycle)	100	—	85	—	70	—	ns
14A[11]	$\overline{DS}$ Width Asserted Write	40	—	38	—	30	—	ns
14B	$\overline{AS}$ (and $\overline{DS}$ Read) Width Asserted (Synchronous Cycle)	40	—	35	—	30	—	ns
15	$\overline{AS}$, $\overline{DS}$ Width Negated	40	—	38	—	30	—	ns
15A[8]	$\overline{DS}$ Negated to $\overline{AS}$ Asserted	35	—	30	—	25	—	ns
16	Clock High to $\overline{AS}$, $\overline{DS}$, R/$\overline{W}$, $\overline{DBEN}$, $\overline{CBREQ}$ High Impedance	—	60	—	50	—	40	ns
17[6]	$\overline{AS}$, $\overline{DS}$ Negated to R/$\overline{W}$ Invalid	15	—	10	—	7	—	ns
18	Clock High to R/$\overline{W}$ High	0	30	0	25	0	20	ns
20	Clock High to R/$\overline{W}$ Low	0	30	0	25	0	20	ns
21[6]	R/$\overline{W}$ High to $\overline{AS}$ Asserted	15	—	10	—	7	—	ns
22[6]	R/$\overline{W}$ Low to $\overline{DS}$ Asserted (Write)	75	—	60	—	47	—	ns
23	Clock High to Data Out Valid	—	30	—	25	—	20	ns
24	Data-Out Valid to Negating Edge of $\overline{AS}$	12	—	8	—	5	—	ns
25[6,11]	$\overline{AS}$, $\overline{DS}$ Negated to Data Out Invalid	15	—	10	—	7	—	ns
25A[9,11]	$\overline{DS}$ Negated to $\overline{DBEN}$ Negated (Write)	15	—	10	—	7	—	ns
26[6,11]	Data Out Valid to $\overline{DS}$ Asserted (Write)	15	—	10	—	7	—	ns
27	Data-In Valid to Clock Low (Synchronous Setup)	5	—	4	—	2	—	ns
27A	Late $\overline{BERR}$/$\overline{HALT}$ Asserted to Clock Low (Setup)	15	—	10	—	5	—	ns

PRELIMINARY AC ELECTRICAL SPECIFICATIONS (Continued)

Num.	Characteristic	16.67 MHz		20 MHz		25 MHz		Unit
		Min	Max	Min	Max	Min	Max	
28[12]	$\overline{AS}$, $\overline{DS}$ Negated to $\overline{DSACKx}$, $\overline{BERR}$, $\overline{HALT}$, $\overline{AVEC}$ Negated (Asynchronous Hold)	0	60	0	50	0	40	ns
28A[12]	Clock Low to $\overline{DSACKx}$, $\overline{BERR}$, $\overline{HALT}$, $\overline{AVEC}$ Negated (Synchronous Hold)	15	100	12	85	8	70	ns
29[12]	$\overline{DS}$ Negated to Data-In Invalid (Asynchronous Hold)	0	—	0	—	0	—	ns
29A[12]	$\overline{DS}$ Negated to Data-In High Impedance	—	60	—	50	—	40	ns
30[12]	Clock Low to Data-In Invalid (Synchronous Hold)	15	—	12	—	8	—	ns
30A[12]	Clock Low to Data-In High Impedance (Read followed by Write)	—	90	—	75	—	60	ns
31[2]	$\overline{DSACKx}$ Asserted to Data-In Valid (Asynchronous Data Setup)	—	50	—	43	—	28	ns
31A[3]	$\overline{DSACKx}$ Asserted to $\overline{DSACKx}$ Valid (Skew)	—	15	—	10	—	7	ns
32	$\overline{RESET}$ Input Transition Time	—	1.5	—	1.5	—	1.5	Clks
33	Clock Low to $\overline{BG}$ Asserted	0	30	0	25	0	20	ns
34	Clock Low to $\overline{BG}$ Negated	0	30	0	25	0	20	ns
35	$\overline{BR}$ Asserted to $\overline{BG}$ Asserted ($\overline{RMC}$ Not Asserted)	1.5	3.5	1.5	3.5	1.5	3.5	Clks
37	$\overline{BGACK}$ Asserted to $\overline{BG}$ Negated	1.5	3.5	1.5	3.5	1.5	3.5	Clks
37A	$\overline{BGACK}$ Asserted to $\overline{BR}$ Negated	0	1.5	0	1.5	0	1.5	Clks
39[14]	$\overline{BG}$ Width Negated	90	—	75	—	60	—	ns
39A	$\overline{BG}$ Width Asserted	90	—	75	—	60	—	ns
40	Clock High to $\overline{DBEN}$ Asserted (Read)	0	30	0	25	0	20	ns
41	Clock Low to $\overline{DBEN}$ Negated (Read)	0	30	0	25	0	20	ns
42	Clock Low to $\overline{DBEN}$ Asserted (Write)	0	30	0	25	0	20	ns
43	Clock High to $\overline{DBEN}$ Negated (Write)	0	30	0	25	0	20	ns
44[6]	R/$\overline{W}$ Low to $\overline{DBEN}$ Asserted (Write)	15	—	10	—	7	—	ns
45[5]	$\overline{DBEN}$ Width Asserted — Asynchronous Read	60	—	50	—	40	—	ns
	$\overline{DBEN}$ Width Asserted — Asynchronous Write	120	—	100	—	80	—	
45A[9]	$\overline{DBEN}$ Width Asserted — Synchronous Read	10	—	10	—	5	—	ns
	$\overline{DBEN}$ Width Asserted — Synchronous Write	60	—	50	—	40	—	
46	R/$\overline{W}$ Width Asserted (Asynchronous Write or Read)	150	—	125	—	100	—	ns
46A	R/$\overline{W}$ Width Asserted (Synchronous Write or Read)	90	—	75	—	60	—	ns
47A	Asynchronous Input Setup Time to Clock Low	5	—	4	—	2	—	ns
47B	Asynchronous Input Hold Time from Clock Low	15	—	12	—	8	—	ns
48[4]	$\overline{DSACKx}$ Asserted to $\overline{BERR}$, $\overline{HALT}$ Asserted	—	30	—	20	—	25	ns
53	Data Out Hold from Clock High	3	—	3	—	3	—	ns
55	R/$\overline{W}$ Asserted to Data Bus Impedance Change	30	—	25	—	20	—	ns
56	$\overline{RESET}$ Pulse Width (Reset Instruction)	512	—	512	—	512	—	Clks
58[10]	$\overline{BGACK}$ Negated to Bus Driven	1	—	1	—	1	—	Clks
59[10]	$\overline{BG}$ Negated to Bus Driven	1	—	1	—	1	—	Clks
60[13]	Synchronous Input Valid to Clock High (Setup Time)	5	—	4	—	2	—	ns
61[13]	Clock High to Synchronous Input Invalid (Hold Time)	15	—	12	—	8	—	ns

PRELIMINARY AC ELECTRICAL SPECIFICATIONS (Concluded)

Num.	Characteristic	16.67 MHz		20 MHz		25 MHz		Unit
		Min	Max	Min	Max	Min	Max	
62	Clock Low to $\overline{\text{STATUS}}$, $\overline{\text{REFILL}}$ Asserted	0	30	0	25	0	20	ns
63	Clock Low to $\overline{\text{STATUS}}$, $\overline{\text{REFILL}}$ Negated	0	30	0	25	0	20	ns

NOTES:
1. This number can be reduced to 5 nanoseconds if strobes have equal loads.
2. If the asynchronous setup time (#47A) requirements are satisfied, the $\overline{\text{DSACKx}}$ low to data setup time (#31) and $\overline{\text{DSACKx}}$ low to $\overline{\text{BERR}}$ low setup time (#48) can be ignored. The data must only satisfy the data-in clock low setup time (#27) for the following clock cycle and $\overline{\text{BERR}}$ must only satisfy the late $\overline{\text{BERR}}$ low to clock low setup time (#27A) for the following clock cycle.
3. This parameter specifies the maximum allowable skew between $\overline{\text{DSACK0}}$ to $\overline{\text{DSACK1}}$ asserted or $\overline{\text{DSACK1}}$ to $\overline{\text{DSACK0}}$ asserted; specification #47A must be met by $\overline{\text{DSACK0}}$ or $\overline{\text{DSACK1}}$.
4. This specification applies to the first ($\overline{\text{DSACK0}}$ or $\overline{\text{DSACK1}}$) $\overline{\text{DSACKx}}$ signal asserted. In the absence of $\overline{\text{DSACKx}}$, $\overline{\text{BERR}}$ is an asynchronous input using the asynchronous input setup time (#47A).
5. $\overline{\text{DBEN}}$ may stay asserted on consecutive write cycles.
6. The minimum values must be met to guarantee proper operation. If this maximum value is exceeded, $\overline{\text{BG}}$ may be reasserted.
7. This specification indicates the minimum high time for $\overline{\text{ECS}}$ and $\overline{\text{OCS}}$ in the event of an internal cache hit followed immediately by another cache hit, a cache miss, or an operand cycle.
8. This specification guarantees operation with the MC68881/MC68882, which specifies a minimum time for $\overline{\text{DS}}$ negated to $\overline{\text{AS}}$ asserted (specification #13A in the *MC68881/MC68882 User's Manual*). Without this specification, incorrect interpretation of specifications #9A and #15 would indicate that the MC68030 does not meet the MC68881/MC68882 requirements.
9. This specification allows a system designer to guarantee data hold times on the output side of data buffers that have output enable signals generated with $\overline{\text{DBEN}}$. The timing on $\overline{\text{DBEN}}$ precludes its use for synchronous READ cycles with no wait states.
10. These specifications allow system designers to guarantee that an alternate bus master has stopped driving the bus when the MC68030 regains control of the bus after an arbitration sequence.
11. $\overline{\text{DS}}$ will not be asserted for synchronous write cycles with no wait states.
12. These hold times are specified with respect to strobes (asynchronous) and with respect to the clock (synchronous). The designer is free to use either time.
13. Synchronous inputs must meet specifications #60 and #61 with stable logic levels for ***all*** rising edges of the clock while $\overline{\text{AS}}$ is asserted. These values are specified relative to the high level of the rising clock edge. The values originally published were specified relative to the low level of the rising clock edge.
14. This specification allows system designers to qualify the $\overline{\text{CS}}$ signal of an MC68881/MC68882 with $\overline{\text{AS}}$ (allowing 7 ns for a gate delay) and still meet the $\overline{\text{CS}}$ to $\overline{\text{DS}}$ setup time requirement (spec 8B) of the MC68881/MC68882.

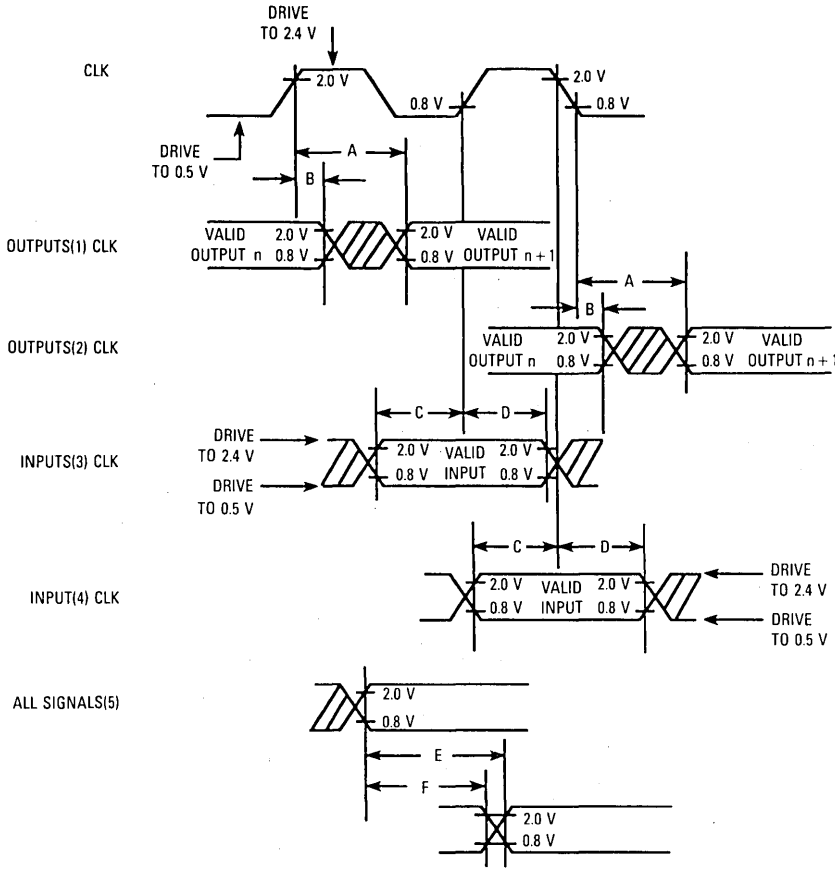
3

AC ELECTRICAL SPECIFICATIONS DEFINITIONS

The AC specifications presented consist of output delays, input setup and hold times, and signal skew times. All signals are specified relative to an appropriate edge of the MC68030 clock input and, possibly, relative to one or more other signals.

The measurement of the AC specifications is defined by the waveforms in Figure 10. In order to test the parameters guaranteed by Motorola, inputs must be driven to the voltage levels specified in Figure 10. Outputs of the MC68030 are specified with minimum and/or maximum limits, as appropriate, and are measured as shown. Inputs to the MC68030 are specified with minimum and, as appropriate, maximum setup and hold times, and are measured as shown. Finally, the measurements for signal-to-signal specifications are also shown.

Note that the testing levels used to verify conformance of the MC68030 to the AC specifications does not affect the guaranteed DC operation of the device as specified in the DC electrical characteristics.



NOTES:

1. This output timing is applicable to all parameters specified relative to the rising edge of the clock.
2. This output timing is applicable to all parameters specified relative to the falling edge of the clock.
3. This input timing is applicable to all parameters specified relative to the rising edge of the clock.
4. This input timing is applicable to all parameters specified relative to the falling edge of the clock.
5. This timing is applicable to all parameters specified relative to the assertion/negation of another signal.

LEGEND:

A. Maximum output delay specification.
B. Minimum output hold time.
C. Minimum input setup time specification.
D. Minimum input hold time specification.
E. Signal valid to signal valid specification (maximum or minimum).
F. Signal valid to signal invalid specification (maximum or minimum).

Figure 10. Drive Levels and Test Points for AC Specifications

These waveforms should only be referenced in regard to the edge-to-edge measurement of the timing specifications. They are not intended as a functional description of the input and output signals. Refer to other functional descriptions and their related diagrams for device operation.

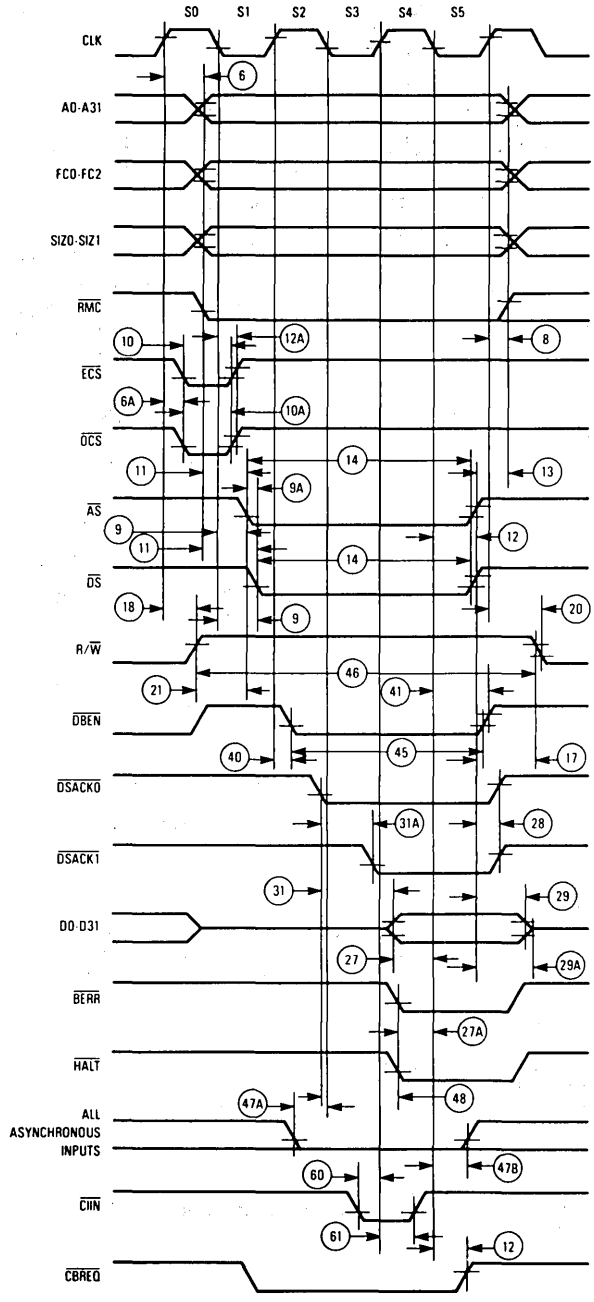


Figure 11. Asynchronous Read Cycle Timing Diagram

These waveforms should only be referenced in regard to the edge-to-edge measurement of the timing specifications. They are not intended as a functional description of the input and output signals. Refer to other functional descriptions and their related diagrams for device operation.

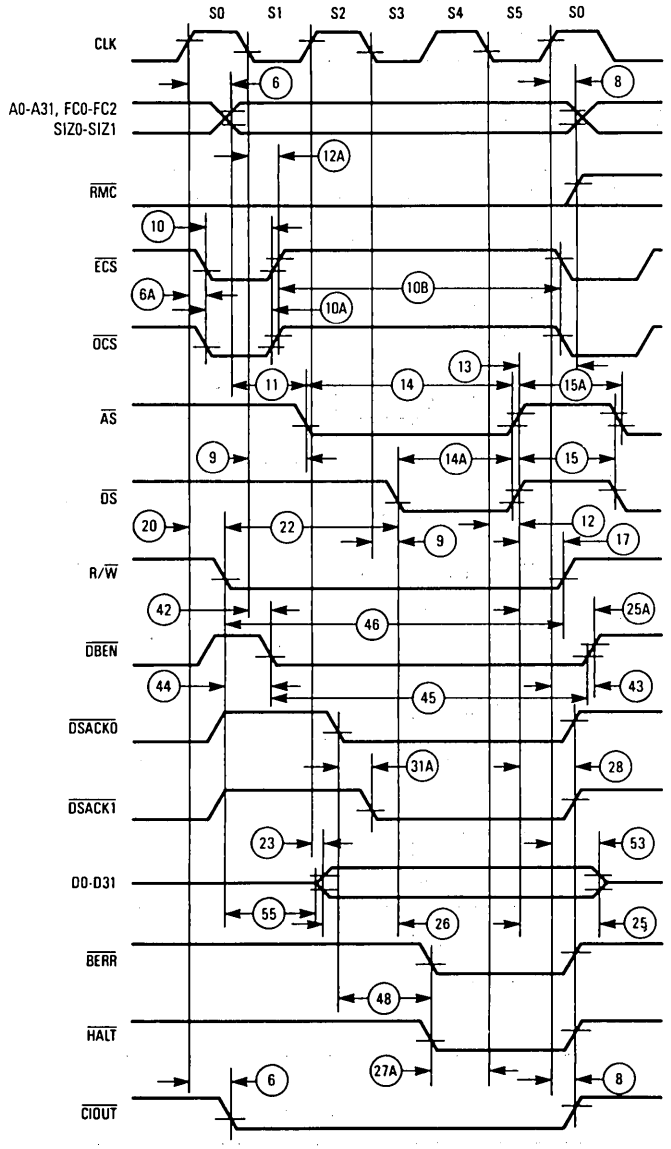


Figure 12. Asynchronous Write Cycle Timing Diagram

These waveforms should only be referenced in regard to the edge-to-edge measurement of the timing specifications. They are not intended as a functional description of the input and output signals. Refer to other functional descriptions and their related diagrams for device operation.

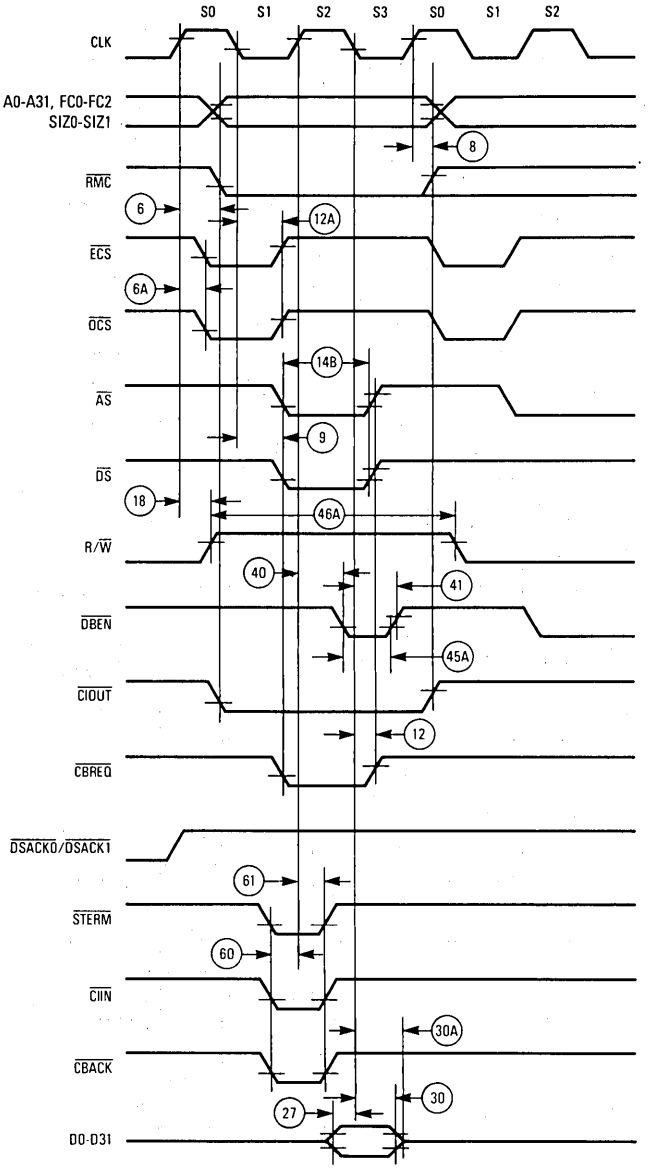


Figure 13. Synchronous Read Cycle Timing Diagram

These waveforms should only be referenced in regard to the edge-to-edge measurement of the timing specifications. They are not intended as a functional description of the input and output signals. Refer to other functional descriptions and their related diagrams for device operation.

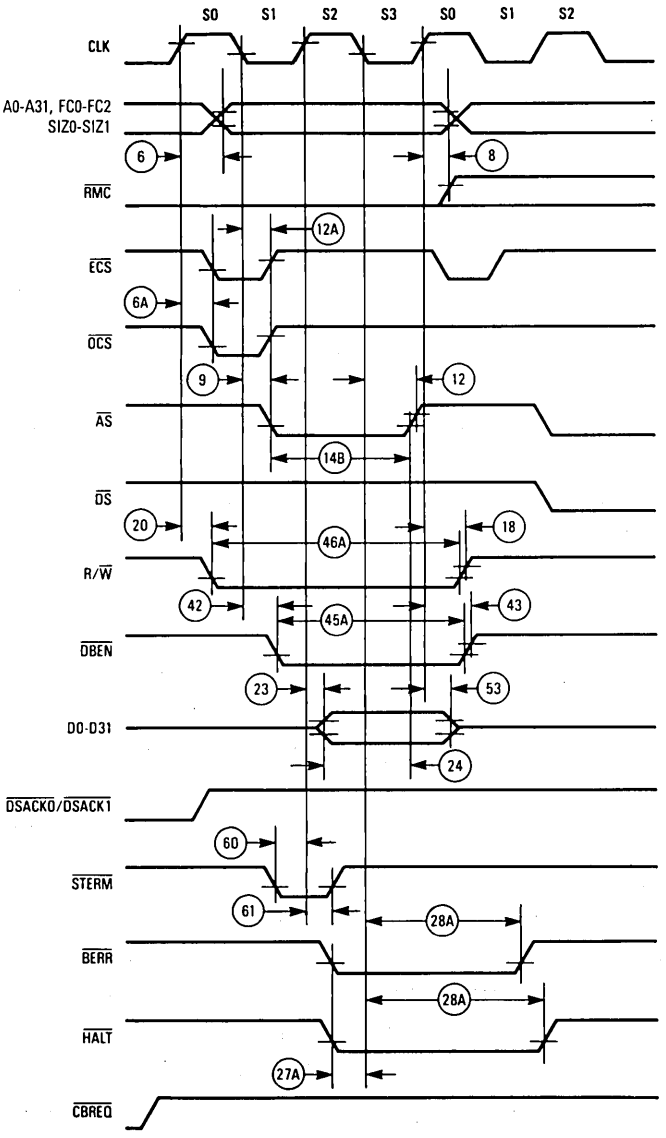


Figure 14. Synchronous Write Cycle Timing Diagram

These waveforms should only be referenced in regard to the edge-to-edge measurement of the timing specifications. They are not intended as a functional description of the input and output signals. Refer to other functional descriptions and their related diagrams for device operation.

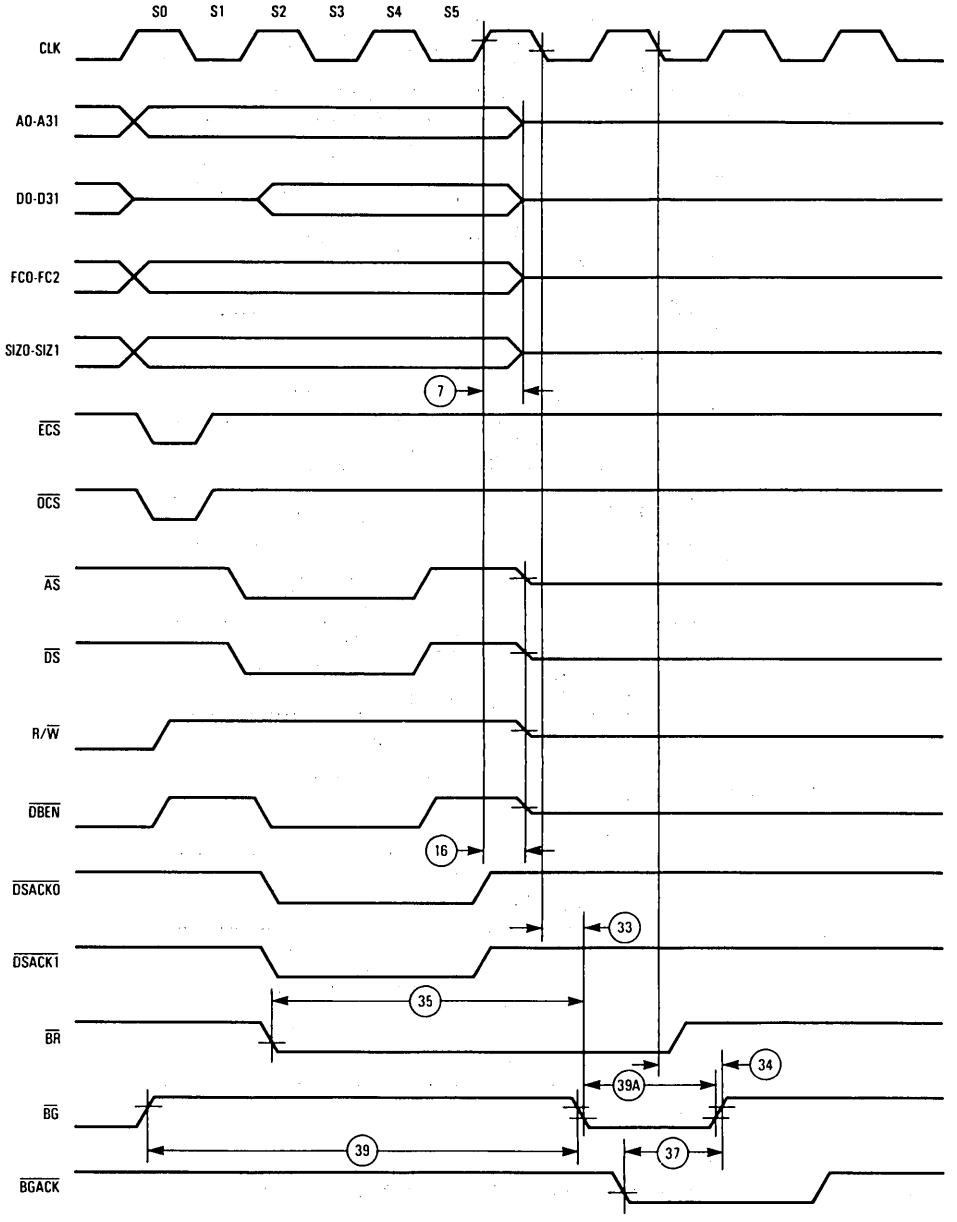
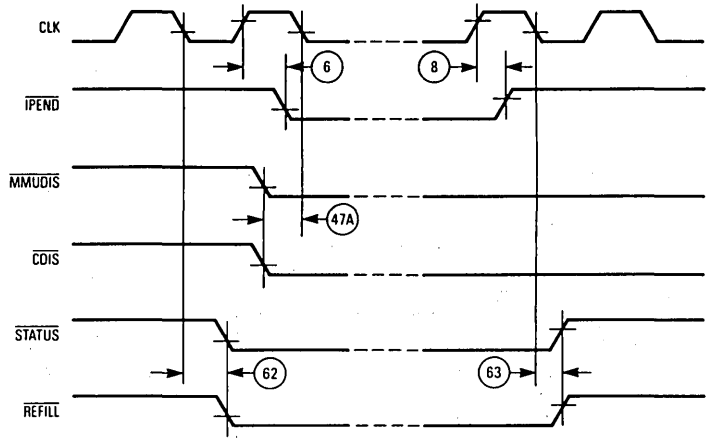


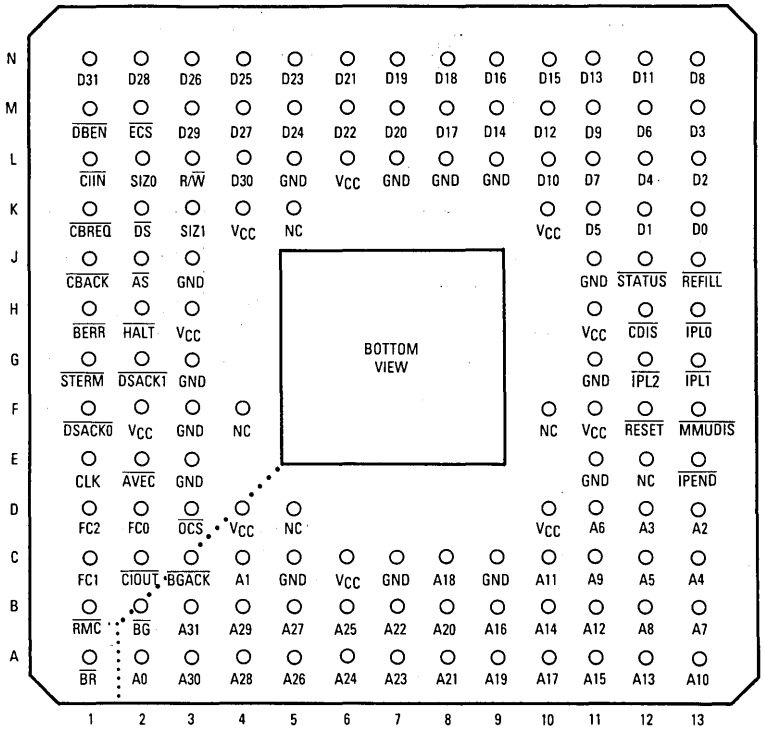
Figure 15. Bus Arbitration Timing Diagram

These waveforms should only be referenced in regard to the edge-to-edge measurement of the timing specifications. They are not intended as a functional description of the input and output signals. Refer to other functional descriptions and their related diagrams for device operation.



PIN ASSIGNMENT

The V_{CC} and GND pins are separated into three groups to provide individual power supply connections for the address bus buffers, data bus buffers, and all other output buffers and internal logic.



V_{CC} and GND Pin Assignments

Pin Group	V_{CC}	GND
Address Bus	C6, D10	C5, C7, C9, E11
Data Bus	L6, K10	J11, L9, L7, L5
$\overline{ECS}$, SIZx, $\overline{DS}$, $\overline{AS}$, $\overline{DBEN}$, $\overline{CBREQ}$, R/$\overline{W}$	K4	J3
FC0–FC2, $\overline{RMC}$, $\overline{OCS}$, $\overline{CIOUT}$, $\overline{BG}$	D4	E3
Internal Logic, $\overline{RESET}$, STATUS, REFILL, Misc.	H3, F2, F11, H11	L8, G3, F3, G11

COPROCESSORS 4

MC68851

Technical Summary

32-Bit Paged Memory Management Unit (PMMU)

The MC68851 is a high-performance paged memory management unit (PMMU) designed to efficiently support a demand paged virtual memory environment with the MC68020 32-bit microprocessor. Implemented using VLSI technology and Motorola's advanced HCMOS fabrication process, the MC68851 is optimized to perform very fast logical-to-physical address translations, to provide a comprehensive access control and protection mechanism, and to provide extensive support for paged virtual systems. Features of the MC68851 include:

- Fast Logical-to-Physical Address Translation
- Hierarchical Protection Mechanism with up to Eight Levels of Protection
- Full 32-Bit Logical and Physical Addresses with 4-Bit Function Code
- Wide Selection of Page Sizes from 256 to 32K Bytes
- Fully Associative 64-Entry On-Chip Address Translation Cache
- Automatic Update of the On-Chip Translation Cache from External Translation Tables
- Multiple Tasks Supported Simultaneously for Fast Task Switching
- M68000 Family Coprocessor Interface
- MC68020 Instruction Set Extensions
- Instruction Breakpoints for Software Debug and Program Control
- Support for Logical and/or Physical Data Cache
- Support for Multiple Logical and/or Physical Bus Masters

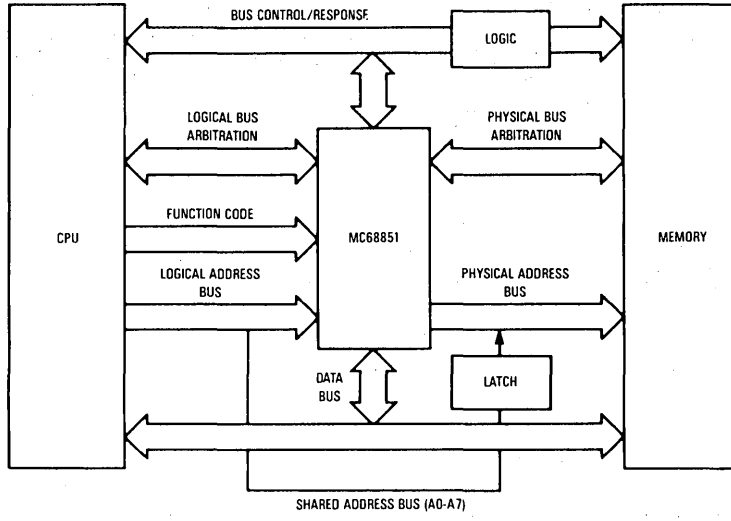


Figure 1. System Block Diagram

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

MC68851 OVERVIEW

The primary system functions of the MC68851 are to provide logical-to-physical address translation, to monitor and enforce the protection/privilege mechanism, and to support the breakpoint operations used for system debug. The MC68851 also supports the M68000 Family coprocessor interface in order to simplify processor/coprocessor communication.

SYSTEM CONFIGURATION

In a simple microprocessor-based system, the CPU is connected directly to memory. No memory mapping or protection functions are provided and the addresses generated by the CPU directly identify the physical locations to be accessed. Any location in the address space that does not contain a memory device cannot be used by the CPU. This type of system is unsuitable for execution of multiple concurrent tasks since there is no mechanism to protect the memory of one task from corruption by another task. It is unsuitable for hosting virtual systems that allow uniform use of a virtual address space that is larger than the physical address space represented by the memory devices, and it is also unsuitable to provide a separate unique address space for each task in the system.

The MC68851 is designed to provide the mapping and protection facilities needed to construct a multi-tasking, demand-paged virtual system. In order to build such a system, the address bus is divided into two sections separated by the MC68851, as shown in Figure 1. The 'logical' address is output by the processor and is monitored by the MC68851 on its logical address inputs. The MC68851 performs translation and privilege checking on the logical address and, if valid, outputs the translated 'physical' value on the physical address bus where it is used to access memory or other physical devices. Using this configuration, all accesses to physical devices are controlled by the MC68851; tasks can be prevented from accessing the resources owned by other tasks. Also, under control of an operating system with virtual capabilities, the logical-to-physical mapping functions of the MC68851 allow tasks to utilize the entire address space of the CPU without knowledge of the physical attributes of the system.

ADDRESS TRANSLATION

The address translation facility of the MC68851 is a comprehensive mechanism that provides logical-to-physical mapping of up to a 4-gigabyte logical address space with no software assistance from the CPU. The address translation mechanism is fully implemented in hardware in order to minimize any penalty in system performance for the mapping functions. The address translation mechanism provides full logical-to-physical mapping in less than one clock cycle for a very high percentage of all bus cycles. The functional timing for these translations is shown in Figure 2. Physical Address Strobe is asserted one clock after the assertion of Logical Address Strobe with translation completed and access rights checked for the valid physical address.

In order to perform the translation functions as shown in Figure 2, the MC68851 contains a high-speed memory that stores recently used logical-to-physical address translations. This memory, the address translation cache (ATC), is a 64-entry, fully-associative array containing logical addresses and their corresponding physical translations (see Figure 3). When a bus cycle is initiated by a logical master, the logical address and function code is input to the ATC where it is simultaneously compared against all current entries. If one of the ATC entries matches (there is a 'hit'), the ATC drives the stored physical address out onto the physical address bus. If the

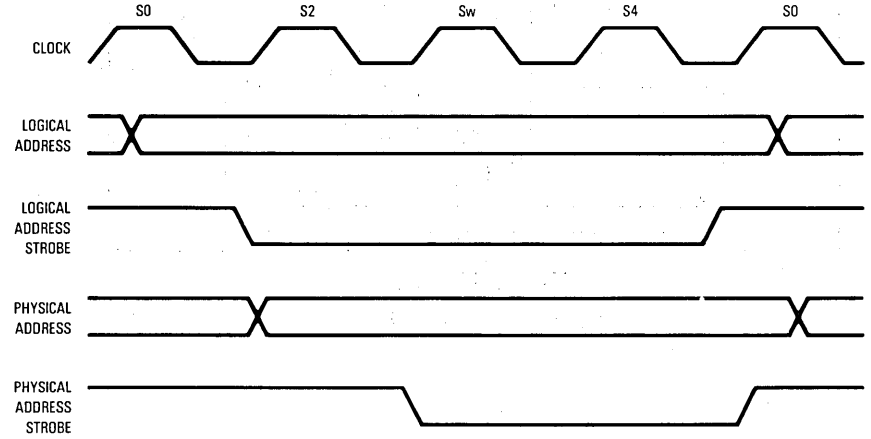
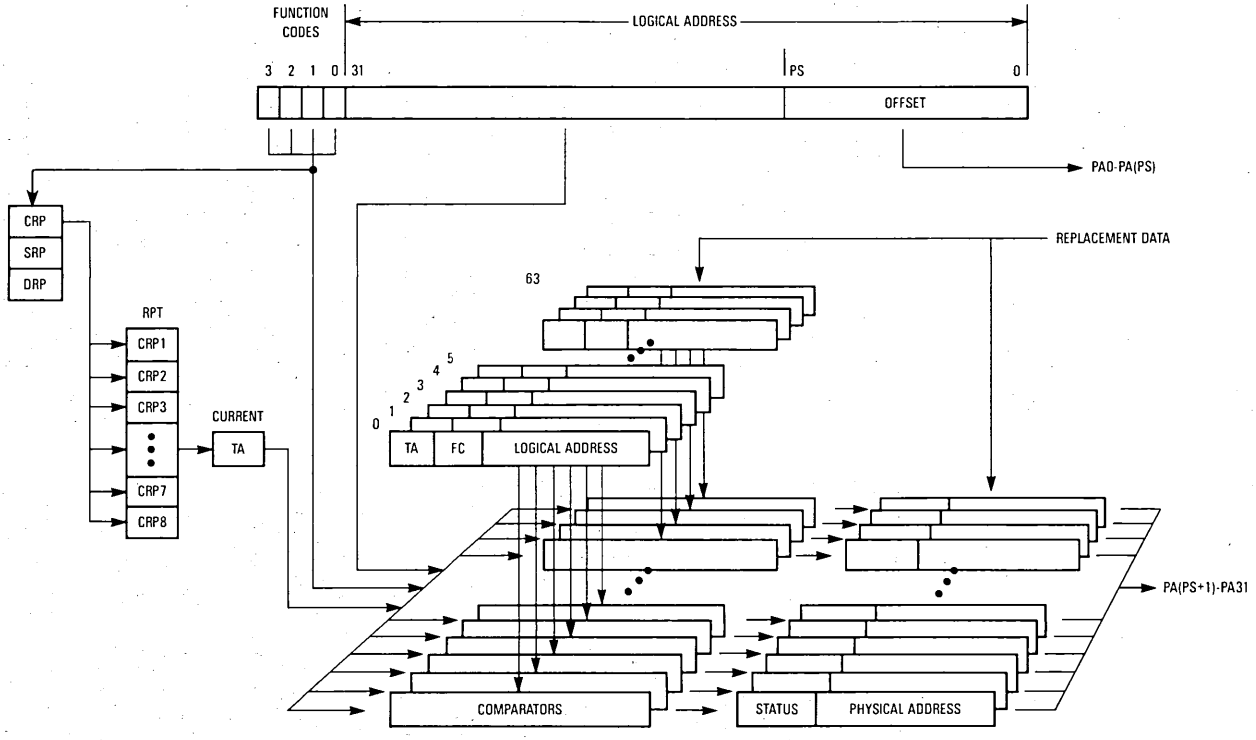


Figure 2. MC68851 Address Translation Functional Timing



PAGE SIZE = 2^{PS} = 256/512/1K/2K/4K/8K/16K/32K BYTES

STATUS INCLUDES: ATC ENTRY LOCKED OR BUS ERRORED, CACHE INHIBIT, MODULE DESCRIPTOR GATE, MODIFIED, AND WRITE PROTECT BITS

Figure 3. ATC with Task Alias, RPT, and Caches

MC68851 detects no exceptional conditions (e.g., write violation), it then asserts the physical address strobe (PAS).

In addition to the address mappings, each entry in the ATC also contains bits that describe the protection information for that mapping (e.g., read-only), a data cache inhibit indicator (used, for instance, when memory-mapped I/O registers must ***not*** be cached), a lock-entry flag (used to freeze individual ATC translation entries), as well as history information used by the MC68851 (used by the ATC replacement algorithm to keep active translations in the ATC).

In order to improve utilization of the MC68851 address translation cache in a multi-tasking environment, translation descriptors for multiple tasks can reside in the ATC simultaneously. To control this, the logical portion of each ATC entry has three additional bits, a 'task alias', that are included in the compare operation to determine if a cache hit has occurred. The task alias identifies one of eight tasks that may have translation descriptors resident in the ATC simultaneously and is used as an extension to the logical address during the cache comparison operation.

The task alias mechanism works in conjunction with the MC68851 root pointers and the root pointer caching function of the MC68851 Root Pointer Table (RPT). When the MC68020 starts a new task, the operating system loads the MC68851 CPU Root Pointer register with the pointer needed for translations of the new task. The CPU Root Pointer register of the MC68851 then contains the address, in physical memory, of the root of the translation table for the currently executing task. The RPT is a table of eight recently-used CPU Root Pointers maintained on-chip by the MC68851. Each of the eight entries in the RPT has a unique 3-bit task alias associated with it. These three bits are included in each ATC entry, so eight tasks can be distinguished among all ATC entries.

When the operating system initiates a new task, or restarts a suspended one, it writes a value to the MC68851 CPU Root Pointer register identifying the location of the translation table for that task. When this value is written, it is compared by the MC68851 against all entries currently in the RPT. This is fast because the RPT is a cache also. If no RPT match is found, then a new entry is made in the RPT and the task alias associated with that entry is assigned to the current task. When the RPT is full, the new entry overwrites the oldest RPT entry, and this task alias becomes associated with the new task. In this case, the MC68851 automatically flushes any entries in the ATC that are currently identified with this task alias, since they are associated with the old task.

If the value loaded into the CPU Root Pointer register matches an entry in the RPT, then the MC68851 already has a task alias assigned to identify those ATC entries that belong to the new task. So, none of the ATC entries are flushed as they can be reused for address translations of the new task. The eight entries of the RPT and the 64 entries of the ATC are well balanced. When a task is restarted, it is very likely the ATC holds some of its entries so the task begins execution immediately. Without task aliasing, the new task would be delayed while the needed working set of translation descriptors are loaded into the ATC from the translation table in memory.

In addition to the CRP, the MC68851 maintains exclusive root pointers for the supervisor (SRP) and for DMA-type devices and coprocessors (DRP) which also require logical-to-physical address translations.

ADDRESS TRANSLATION TABLES

When a logical bus master (e.g., the processor) initiates a cycle that does not have a corresponding translation resident in the ATC, the MC68851 performs bus operations to load the mapping for that cycle from the translation tables in memory pointed to by the relevant root pointer. To perform this search operation, the MC68851 simultaneoulsy aborts the logical bus cycle, signals the master to retry the operation, and requests mastership of the logical bus. Upon receiving indication that the logical bus is free, the MC68851 completes the bus arbitration sequence, assumes mastership of the bus, and begins to search the translation tables pointed to by the relevant root pointer to locate the needed translation descriptor that describes the page accessed by this logical address. After loading the required translation descriptor, the MC68851 returns control of the bus to the logical master to retry the previous bus cycle which can now be checked for access rights and properly translated by the MC68851.

The operation of searching the translation tables and reloading the ATC is called a 'table search'. The MC68851 automatically searches the translation tables when a translation misses in the ATC and does so completely in hardware, without any software assist from the operating system. Using hardware, the MC68851 has significantly minimized the table search overhead when compared to previous memory management schemes that required software assistance.

The translation tables supported by the MC68851 have a tree structure. The root of a translation table tree is pointed to by one of the three Root Pointer registers: CPU, Supervisor, or DMA. Table entries at the higher levels of the tree (pointer tables) contain pointers to other tables. Entries at the leaf level (page tables) contain page descriptors. All addresses contained in the translation table entries are physical addresses. The three root pointers allow the MC68851 to manage three simultaneous activities: the CPU root pointer points at the translation table tree for the currently executing task, the Supervisor root pointer points to the operating system's translation table, and the DMA root pointer manages the space of an alternate bus master which could be a DMA controller.

Figure 4 illustrates the structure of the MC68851 translation tables. Several determinants of the detailed table structure are software selectable. The first level of lookup in the table normally uses the function codes as an index, but this may be suppressed. The function codes are control signals output by the MC68020 that distinguish memory accesses as Supervisor/User and Program/Data space accesses. The function codes can be used by the MC68851 to separate the address map into address spaces that protect any Supervisor space from being corrupted by a faulty User program and/or to prevent a Data access from corrupting a Program space. The logical address can be limited between 17 and 32 bits (inclusive) to control the

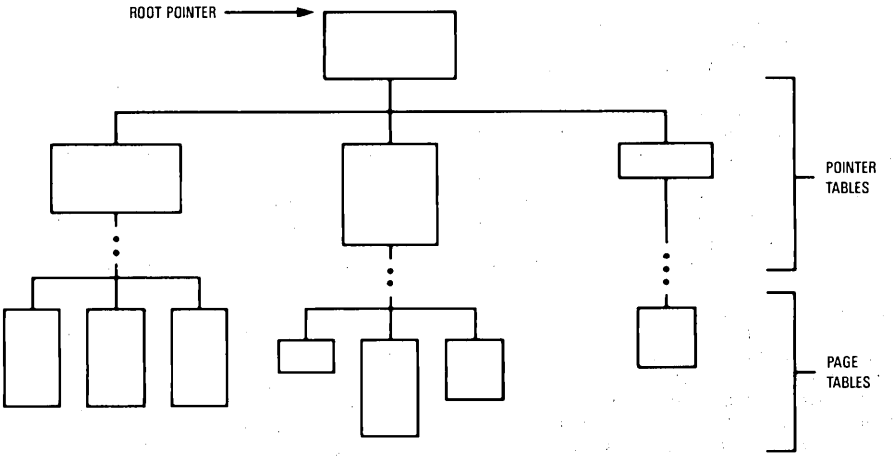


Figure 4. MC68851 Translation Table Tree Structure

necessary size of the corresponding translation table. The number of levels in the table indexed by the logical address can be set from one to four, and up to 15 logical address bits can be used as an index at each level.

PROTECTION MECHANISM

The MC68851 supports a comprehensive protection mechanism that facilitates implementation of fully protected systems. In addition to the option of enforcing the distinction of User and Supervisor modes normally found in an M68000 system, the MC68851 also supports the Access Level mechanism that provides finer granularity of protection within the User address spaces.

The Access Level mechanism subdivides the logical address spaces of User mode operations into one, two, four, or eight level(s) of privilege. Routines operating at different access levels can have different privileges to memory and a facility is provided to closely control changes in Access level.

The Access Level for a bus cycle is encoded in the highest order (zero, one, two, or three) bits of the **logical** address generated by the CPU. The Access Level mechanism, when enabled, compares this value against the current Access Level as specified in the current access level (CAL) register. The current Access Level specifies the highest privilege level that a task may assume at that time. If the privilege level presented by the bus cycle is more privileged then the current level allowed, then the cycle is requesting a privilege in excess of its rights and is aborted by the MC68851.

In the MC68851 protection scheme, the privilege associated with a task is specified by its Access Level. Smaller values for access levels represent higher privilege levels. In a system using eight access levels, level zero is the highest privilege in the hierarchy and level seven is the lowest.

In order to access code and/or data that require a higher level of privilege than is possessed by the current task, the MC68851 supports the MC68020 module call (CALLM) and return (RTM) instructions that allow a routine to transfer execution control to a module operating at the same or higher level of privilege and to return from that module after completion of the module function. When the MC68020 executes a CALLM instruction that requests an increase in Access Level, the MC68020 automatically communicates with the MC68851 Access Level mechanism via Access Level Control CPU Space cycles, to determine if the requested change is valid. The MC68851 checks the request against a module descriptor for that operation and indicates the validity of that request to the MC68020. The RTM instruction operates similarly except that control is always passed from a task to a task of the same or lesser privilege.

BREAKPOINTS

The MC68851 provides a breakpoint acknowledge facility to support the MC68020 and other processors with an on-chip cache. When the MC68020 encounters a breakpoint instruction it executes a breakpoint acknowledge bus cycle by reading from a predetermined address in the CPU address space. The MC68851 decodes this address and responds by either providing a replacement opcode for the breakpoint opcode and completing the bus cycle normally (by asserting the data transfer and size acknowledge outputs) or terminating the bus cycle with an exception (by asserting bus error to initiate illegal instruction exception processing). The MC68851 can be programmed to signal the illegal instruction exception

on every breakpoint or to provide the replacement opcode *n* times ($1 \leq n \leq 255$) before signaling the exception. With eight sets of breakpoint registers, the MC68851 supports eight breakpoints simultaneously.

Debugging a MC68020-based system which uses an on-chip cache is simplified with the MC68851 breakpoint support. Programmers typically use a debug monitor when debugging programs, which can place up to eight breakpoints in the program's code when using the breakpoint support of the MC68851. The debug monitor replaces each target instruction with a breakpoint instruction, passing the target opcode to the MC68851 with a skip count. During execution of the program, the MC68020 encounters the various breakpoint instructions. The MC68851 automatically provides the correct target opcode to the MC68020 for each execution of the corresponding breakpoint instruction and count down the skip count to zero. The MC68020 executes the substituted instruction as if it were still in program memory and the program continues with only a small overhead to retrieve the substituted opcode. Once the skip count is exhausted during breakpoint processing, the debug monitor regains control through the illegal instruction exception handler.

COPROCESSOR CONCEPT

The M68000 Family coprocessor interface is an integral part of the design of the MC68020 advanced microprocessor, the MC68881 floating-point coprocessor, and the MC68851 paged memory management unit. The coprocessor interface allows the execution of special purpose instructions that are logical extensions to the microprocessor. Each coprocessor (e.g., MC68851 or MC68881) has an instruction set that reflects its special function. These instructions may be executed merely by placing the instruction opcode and parameters in the MC68020 instruction stream. The MC68020 decodes the coprocessor instruction and performs bus communication with the coprocessor registers specifying the nature of the action to be taken. Both the MC68020 and the coprocessor execute parts of the instruction, depending on which processor is best suited to handle a particular task.

The interchange of information and the division of responsibility between the processor and the coprocessor are controlled by the coprocessor interface and this activity is transparent to the user/programmer. The addition of a coprocessing unit to an MC68020 system is a logical extension that simply complements the instruction set executable by the processor.

The coprocessor interface is designed to be flexible, functional, and expandable. The interface is intended to support the current M68000 Family of devices and future extensions to the Motorola coprocessor family, as well as user-defined coprocessors for single or multiple coprocessor systems.

MC68851 INSTRUCTIONS

The MC68851 implements an extension to the M68000 Family instruction set using the coprocessor interface. These instructions provide control functions for:

- Loading and storing of PMMU registers,
- Testing access rights, and conditionals based on the results of this test, and
- PMMU control functions.

The MC68851 instruction set extension to the MC68020 instruction set provides the software programmer a programming model that allows the registers and functions implemented by the MC68851 to appear to the programmer as if actually available on-chip with the MC68020. The programming model is shown in Figure 5.

The instruction set extensions for the MC68851 are as follows:

PMOVE
Moves data to/from MC68851 register.

PVALID
Compares access rights of a logical address against the current access level and traps if address requires a higher privilege than allowed. This instruction can be used by a routine to verify that an address passed to it by a calling routine is a valid address.

PTESTR
Searches the translation tables and loads the status and access rights information of a logical address used for a read cycle into the MC68851 status register. This instruction allows the operating system to quickly determine the cause of faults generated by a read cycle from a particular logical address.

PTESTW
Searches the translation tables and loads the status and access rights information of a logical address used for a write access into the MC68851 status register. This instruction allows the operating system to quickly determine the cause of faults generated by a write cycle to a particular logical address.

PLOADR
Searches translation tables and loads the ATC with a translation for the specified logical address. The history information in the external translation tables is updated to reflect that the physical page corresponding to the logical address has been used.

PLOADW
Searches translation tables and loads the ATC with a translation for the specified logical address. The history information in the external translation tables is updated to reflect that the physical page corresponding to the logical address has been modified.

PFLUSH
Flushes translation cache entries by logical address, function code, or function code and effective address. The PFLUSH instructions allow the operating system to easily remove entries from the ATC after making modifications to the external translation tables.

PFLUSHA
Flushes all entries from the translation cache.

PFLUSHR
Flushes root pointer table and translation cache entries by root pointer.

PFLUSHS
Flushes entries from the ATC by logical address and/or function code including globally shared entries.

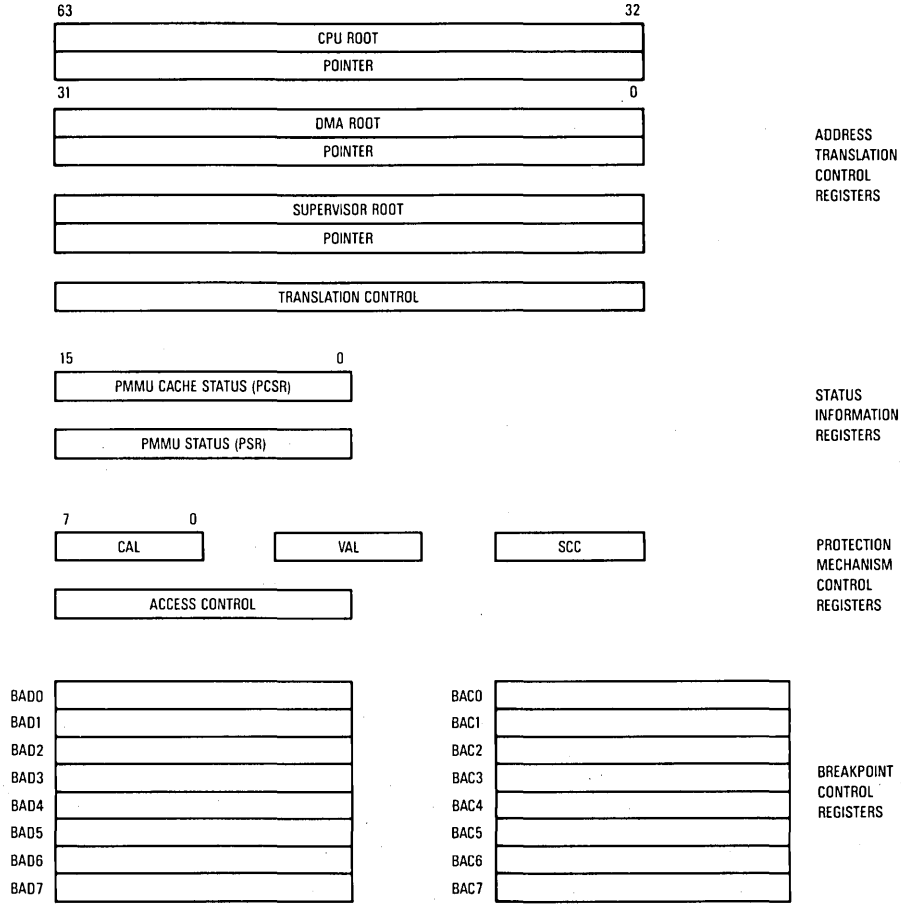


Figure 5. MC68851 Programming Model

PSAVE
Saves the internal state of the MC68851 in order to support fast context switching and MC68020 virtual memory/virtual machine capabilities.

PRESTORE
Restores the internal state of the MC68851 stored by the PSAVE instruction.

PBcc
Branches conditionally on MC68851 condition. The conditional instructions provide the operating system with a means by which program flow can be controlled by MC68851 conditions.

PDBcc
Tests MC68851 condition, decrements a CPU register, and branches.

PScc
Sets operand according to MC68851 condition.

TRAPcc
Traps on MC68851 condition.

SIGNAL DESCRIPTION

This following paragraphs provide a brief description of the input and output signals of the MC68851 paged memory management unit. The signals are functionally grouped as shown in Figure 6.

NOTE
The terms assertion and negation are used extensively. This is done to avoid confusion when dealing with a mixture of 'active low' and 'active high' signals. The term **assert** and **assertion** is used to indicate that a signal is

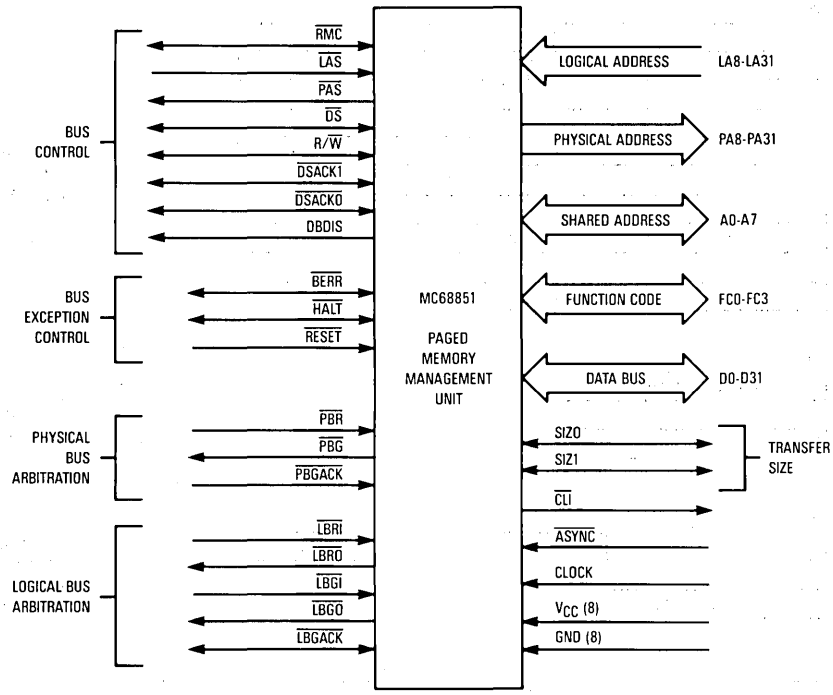


Figure 6. Functional Signal Groups

active or **true**, independent of whether that level is represented by a high or low voltage. The term **negate** or **negation** is used to indicate that a signal is inactive or **false**.

LOGICAL ADDRESS BUS (LA8 through LA31)

These inputs are the lines on which the MC68851 accepts a logical address for translation or for internal operations. The logical address bus should be connected to the address outputs of all logical bus masters.

If the logical address is less than 32 bits (logical address space $<2^{32}$ bytes) as determined by the translation control register, the unused bits are ignored and should be tied to a constant voltage level (either V_{CC} or ground).

PHYSICAL ADDRESS BUS (PA8 through PA31)

These three-state outputs provide the physical address for both address translations and MC68851-initiated bus operations.

SHARED ADDRESS BUS (A0 through A7)

The use of these three-state, bidirectional lines is shared between the functions of the logical and physical buses. When the MC68851 is performing address translations, these signals are input in order that the MC68851 be able to monitor the entire logical address in the event that a CPU space cycle accesses one of its registers. When the MC68851 is the bus master, these pins output the low order eight bits of the physical address. With the inclusion of A0 through A7, both the logical and physical buses have a 32-bit (4 gigabyte) linear addressing range.

FUNCTION CODE (FC0 through FC3)

These three-state, bidirectional signals indicate the address space of the current bus cycle. When the MC68851 is performing translations, these signals provide the address space being accessed by the current logical bus master. The MC68851 uses the function code associated with a bus cycle as an extension to the logical address when creating entries in the address translation cache. The function code may also be used as an index from the root pointer in the first level of a translation table search.

The 4-bit function code consists of the three function code outputs of the M68000 Family processor and a fourth bit that indicates that a DMA access is in progress.

When the MC68851 is bus master it drives the function code pins as outputs with a constant value of FC3–FC0 = $5, indicating the supervisor data space.

DATA BUS (D0 through D31)

These three-state bidirectional signals provide the general purpose data path between the MC68851 and other devices. This bus may be dynamically sized through use

of the DSACKx signals, transferring 8, 16, 24, or 32 bits of information during a bus cycle. The most significant byte of the data bus is D24 through D31.

In systems that do not use the MC68020 (or any other 32-bit CPU) as the main processor, the width of the data bus used to communicate between the processor and the MC68851 may be fixed at 16, or 8 bits. In such systems, the dynamic bus sizing mechanism still functions but the maximum amount of data transferred in a single cycle is limited to the bus size. In either case, the processor data bus is aligned towards the high order portion of the MC68851 data bus — that is, an 8-bit master is connected to D24 through D31 and a 16-bit master is connected to D16 through D31.

When the RESET signal is asserted, the MC68851 inputs configuration information from the least significant byte of the data bus (D0–D7). This information determines the bus size for coprocessor operations, sets the 'decision time' for determining whether or not an ATC hit has occurred, determines whether the CLI signal is asserted for all MC68851-initiated bus operations, and sets the timing for PAS and DS assertion during table searches.

TRANSFER SIZE (SIZ0, SIZ1)

These three-state, bidirectional signals are used in conjunction with the dynamic bus sizing capabilities of the MC68851. When the MC68851 is the bus master, the SIZE signals are driven as outputs and when accessed as a slave, these signals are inputs. Otherwise, the size signals are ignored. Regardless of the state (input or output) of these signals, they indicate the number of bytes remaining to be transferred during the current operand cycle.

An operand cycle is a bus cycle or sequence of bus cycles required to transfer a complete operand.

BUS CONTROL SIGNALS

The logical and physical bus control signals are described in the following paragraphs.

Read-Modify-Write (RMC)

This three-state, bidirectional signal is used to indicate that the bus cycle in progress is an indivisible read-modify-write cycle. This signal is asserted for the duration of the read-modify-write sequence and should be used as a bus lock to ensure integrity of operation of these cycles.

When the MC68851 is translating addresses, the assertion of RMC by the logical bus master indicates that the master is performing a read-modify-write cycle and that a write operation to the same operand is likely to follow. When RMC is asserted during a read cycle, the MC68851 performs access and privilege checking for that cycle as if it were a write cycle in order that the operation not be aborted after having partially completed the write portion of the cycle. In addition, physical bus arbitration is suspended once the physical bus cycle for the address translation is initiated.

When the MC68851 is bus master, RMC may be asserted to indicate that the operation in progress should not be interrupted by other bus traffic and, hence, all arbitration for the physical bus is suspended by the MC68851 when this signal is asserted.

Logical Address Strobe (LAS)

The assertion of this input indicates that the logical bus master has driven the logical address bus, function code, and R/W valid. When the MC68851 is being accessed as a slave, the assertion of LAS also indicates that the SIZE signals are driven valid.

Physical Address Strobe (PAS)

This three-state output is asserted when the MC68851 has driven a valid address on the physical address bus. When the MC68851 is master of the logical bus, the assertion of PAS also indicates that the function code, R/W, and SIZE signals are valid.

Data Strobe (DS)

This bidirectional, three-state signal is used to control the flow of information on the data bus.

When the MC68851 is selected by the CPU, DS is an input that indicates that the MC68851 should drive the data bus on a read cycle, or that the CPU has placed valid data on the bus during a write cycle.

When the MC68851 is the bus master, DS indicates that the slave device should drive the data bus in the case of a read cycle, or that the MC68851 has placed valid data on the bus in the case of a write cycle.

The data strobe is ignored for the purposes of address translation.

Read/Write (R/W)

This bidirectional, three-state signal is used to indicate the direction of transfer for a bus cycle.

When the MC68851 is translating addresses, the state of the R/W signal is input in order to support write-protection checking.

When the MC68851 register set is accessed by the CPU for an operation, the R/W output by the CPU determines the direction of data transfer. If this signal is asserted (low) the MC68851 latches data from the data bus at the termation of the cycle. If the signal is negated (high), the MC68851 outputs data on the data bus and signals that the transfer is complete.

When the MC68851 is bus master, the R/W signal is driven as an output. A high level indicates a read from an external device, a low indicates a write to an external device.

Data Transfer and Size Acknowledge (DSACK0, DSACK1)

These bidirectional, three-state signals, whether used as inputs or outputs, are used to normally terminate a bus cycle and to indicate the **port** size of the responding device.

When the MC68851 register set is accessed by the CPU, the DSACKx signals are output to indicate that valid data has been or will be (see below) placed on the data bus for a read cycle, or that data has been accepted from the data bus for a write cycle. Note that the relationship between DSACKx and data is dependent on the operating mode of the MC68851. When operating in the synchronous mode, the MC68851 drives the data bus on the same clock edge that DSACKx is asserted. Otherwise, the MC68851 drives the data bus a minimum of one clock period before asserting the DSACKx signals.

The DSACKx signals are monitored as inputs when the MC68851 arbitrates for the logical bus. After receiving a bus grant from the CPU, the MC68851 waits until LBGACK, LAS, and both DSACKx signals are negated before asserting logical bus grant acknowledge in order to ensure that the previous slave device has released connection from the bus.

When the MC68851 is executing bus cycles as the physical bus master, the DSACKx signals are inputs to indicate that a data transfer is complete and the port size of the external device being accessed. During a read cycle, when the MC68851 recognizes DSACKx, it latches the data and then terminates the bus cycle; during a write cycle, when the MC68851 recognizes DSACKx, the bus cycle is terminated.

When operating as bus master, the MC68851 synchronizes the DSACKx inputs and allows skew between the two inputs of up to one quarter of a clock.

4

Data Buffer Disable (DBDIS)

This active-high output provides an enable to external data buffers connected to the MC68851 data bus.

When the logical bus master reads the contents of one of the MC68851 registers, the MC68851 drives the data bus with the required operand. Typical systems directly connect the MC68851 data bus with that of the main processor and the combined bus is buffered before being routed to a large number of physical address space devices. In order to avoid contention, the buffers between the MC68851/CPU bus and the bus driving the physical memory must be disabled when the MC68851 drives the bus. The MC68851 provides the control necessary to perform this function with the DBDIS signal.

In addition, DBDIS performs a function similar to the function of the MC68020 DBEN signal. DBDIS can be used to control data bus transceivers in order to avoid contention between the transceivers and the MC68851 data bus drivers during table search operations.

Finally, DBDIS is driven during reset in order to isolate the MC68851 data bus while configuration information is being input.

BUS EXCEPTION CONTROL SIGNALS

The following paragraphs describe the bus exception control signals for the MC68851.

Reset (RESET)

Assertion of this input signals the MC68851 to disable the address translation mechanism, clear all breakpoints, set the internal state to idle, and input configuration information from the data bus.

Halt (HALT)

HALT is a bidirectional, three-state signal.

When the MC68851 is the logical bus master, HALT is an input and assertion of HALT stops all MC68851 bus activity at the completion of the current bus cycle. When the MC68851 has been halted using this input, all control signals, with the exception of bus arbitration outputs, are placed in their inactive states and the physical address bus remains driven with the value used during the previous bus cycle. Bus arbitration functions normally when the MC68851 is halted.

When the MC68851 is translating addresses, HALT is used as an output in conjunction with BERR and/or LBRO to signal the current logical bus master to perform either a 'relinquish and retry' or a 'relinquish' operation.

During address translation, the assertion of HALT by an external device does not effect translation operations of the MC68851.

Bus Error (BERR)

This bidirectional, three-state signal is used to indicate that a bus cycle should be terminated due to abnormal conditions.

When the MC68851 is bus master, BERR is an input and assertion of BERR by an external device signals that there has been some problem with the bus cycle currently being executed. These problems may be the result of:

1) Non-responding devices, or
2) Various other application-dependent errors (for example, parity errors).

When the MC68851 is translating addresses, bus error is used as an output to the logical bus master. Bus error is asserted by the MC68851 for the following conditions:

1) The BERR bit is set in the matched ATC entry,
2) A write or read-modify-write cycle is attempted to a write-protected page,
3) An instruction breakpoint is detected and the associated count register is zero or it is disabled,
4) As a portion of the relinquish and retry operation if:
   a) the required address mapping is not resident in the ATC,
   b) a write operation occurs to a previously unmodified page,
   c) a read from the response CIR causes a suspended PLOAD or PTEST instruction to be restarted,
   d) a module call operation references a descriptor that does not have a corresponding entry in the ATC.
5) An RMC cycle is attempted and a corresponding descriptor with appropriate status is not resident in the ATC,
6) The access level protection mechanism detects an access violation.

The bus error signal interacts with the HALT signal to determine if the current bus cycle should be retried or aborted.

CACHE LOAD INHIBIT (CLI)

During address translation this three-state output is asserted by the MC68851 if the matched address translation cache entry has its CI (cache inhibit) bit set. Assertion of this output signals to external caches that the data associated with the current bus cycle is non-cacheable.

In order to maintain the distinction between CPU space and other address spaces (for example, supervisor program, . . ., etc.) the MC68851 does not assert PAS for CPU space cycles. Cache load inhibit is used to generate a CPU space address strobe during CPU space cycles that do not access the MC68851. CLI is asserted on the falling

edge of the clock and external qualification of $\overline{CLI}$ with $\overline{LAS}$ and a CPU space indicator provides a CPU space address strobe. CPU space cycles that access the MC68851 registers are decoded internally and generate no physical bus activity. Note that if the MC68851 is not master of the physical bus, $\overline{CLI}$ is not asserted until ownership of the physical bus is returned to the MC68851.

When the MC68851 is performing table search operations, it continuously asserts $\overline{CLI}$ in order to prevent caching of translation table information. This function may be suppressed during reset configuration if desired.

ASYNCHRONOUS CONTROL ($\overline{ASYNC}$)

When a logical bus master does not present logical bus control signals with the exact timing specifications of the MC68020, this input must be driven, with appropriate setup and hold times, to inform the MC68851 that input synchronization must take place.

Operating in a synchronous mode, the MC68851 utilizes known signal relationships in order to perform faster translations. If the logical bus master does not present signals conforming to these relationships (different control strobe timings and/or different operating frequency), it must assert $\overline{ASYNC}$ prior to initiating bus activity.

CLOCK (CLK)

The MC68851 clock input is a TTL-compatible signal that is internally buffered to develop internal clocks for the memory management unit. The clock must conform to minimum and maximum period and pulse width specifications and must be of a constant frequency.

Note that the MC68851 and the logical bus master may operate at different clock frequencies.

PHYSICAL BUS ARBITRATION

This section describes the three-wire physical bus arbitration circuitry of the MC68851 used to determine which device in a system is the master of the physical bus.

The MC68851 is the default master of the physical bus and any other devices requiring access to the bus must arbitrate for mastership.

Physical Bus Request ($\overline{PBR}$)

This input is the wire-OR of the bus request signals from all potential physical bus masters and indicates that some device other than the MC68851 requires mastership of the physical bus.

Physical Bus Grant ($\overline{PBG}$)

This output signal indicates to potential bus masters that the MC68851 will release ownership of the physical bus when the current bus cycle is completed.

Physical Bus Grant Acknowledge ($\overline{PBGACK}$)

This input indicates that some other device has become master of the physical bus. This signal should not be asserted until the following conditions have been met:

1) A physical bus grant ($\overline{PBG}$) has been received through the arbitration process,
2) $\overline{PAS}$ is negated, indicating that neither the MC68851 nor the logical bus master is using the physical bus,
3) $\overline{DSACKx}$ are negated, indicating that no external device is still driving the data bus, and
4) $\overline{PBGACK}$ is negated, indicating that no other device is still claiming bus mastership.

$\overline{PBGACK}$ must remain asserted as long as any device other than the MC68851 is bus master.

LOGICAL BUS ARBITRATION

The following paragraphs describe the five-wire bus arbitration pins used to determine which device in the system is the master of the logical bus.

Logical Bus Request In ($\overline{LBRI}$)

The $\overline{LBRI}$ input indicates that a device with higher priority than the MC68851 or the current logical bus master requires ownership of the logical bus.

Logical Bus Request Out ($\overline{LBRO}$)

This output is asserted to inform the processor that the MC68851 requires ownership of the logical bus and is used as a portion of the relinquish operation and the relinquish and retry operation.

The request input to the logical bus arbiter (usually the main processor) should consist of the wire-OR of requests input to $\overline{LBRI}$ logically ORed with the $\overline{LBRO}$ output of the MC68851.

Logical Bus Grant In ($\overline{LBGI}$)

This input, generated by the MC68020, indicates that the MC68020 will release ownership of the bus at the completion of the current bus cycle, or, if an alternate master is currently the owner of the bus, that the MC68020 will not claim the bus after the alternate master has released it.

Logical Bus Grant Out ($\overline{LBGO}$)

This output indicates that the MC68851 has recognized and synchronized the assertion of $\overline{LBGI}$ by the MC68020, has detected the assertion of $\overline{LBRI}$, and is passing the bus grant to an alternate logical bus master or to arbitration prioritization circuitry.

Logical Bus Grant Acknowledge ($\overline{LBGACK}$)

This bidirectional, three-state signal indicates that a logical bus master, other than the CPU, has taken control of the logical bus.

This signal is asserted by the MC68851 to indicate when it is the current logical bus master. $\overline{LBGACK}$ is also monitored as an input to determine when the MC68851 can become bus master.

SIGNAL SUMMARY

Table 1 provides a summary of the electrical characteristics of the signals discussed in the previous paragraphs.

Table 1. Signal Summary

Signal Function	Signal Name	Input/Output	Active State	Three-State	Driven by MC68851 When
Logical Address Bus	LA8rLA31	Input	High	—	—
Physical Address Bus	PA8rPA31	Output	High	Yes	MC68851 Owns Physical Bus
Shared Address Bus	A0rA7	Input/Output	High	Yes	MC68851 Owns Logical and Physical Buses
Function Codes	FC0rFC3	Input/Output	High	Yes	MC68851 Owns Logical and Physical Buses
Data Bus	D0rD31	Input/Output	High	Yes	Read from MC68851 Registers or MC68851 Write Cycle
Size	SIZ0rSIZ1	Input/Output	High	Yes	MC68851 Owns Logical and Physical Buses
Cache Load Inhibit	$\overline{CLI}$	Output	Low	No	Always
Asynchronous Control	$\overline{ASYNC}$	Input	Low	—	—
Read-Modify-Write Cycle	$\overline{RMC}$	Input/Output	Low	Yes	MC68851 Owns Logical and Physical Buses
Logical Address Strobe	$\overline{LAS}$	Input	Low	—	—
Physical Address Strobe	$\overline{PAS}$	Output	Low	Yes	MC68851 Owns Physical Bus
Data Strobe	$\overline{DS}$	Input/Output	Low	Yes	MC68851 Owns Logical and Physical Buses
Read/Write	R/$\overline{W}$	Input/Output	High/Low	Yes	MC68851 Owns Logical and Physical Buses
Data Transfer and Size Acknowledge	$\overline{DSACK0}$r$\overline{DSACK1}$	Input/Output	Low	Yes	Access to Address Map Occupied by MC68851 Interface Register Set
Data Bus Disable	DBDIS	Output	High	No	Always
Bus Error	$\overline{BERR}$	Input/Output	Low	Yes	Exceptional Condition is Generated by Address Translation
Halt	$\overline{HALT}$	Input/Output	Low	Yes	Exceptional Condition is Generated by Address Translation
Reset	$\overline{RESET}$	Input	Low	—	—
Physical Bus Request	$\overline{PBR}$	Input	Low	—	—
Physical Bus Grant	$\overline{PBG}$	Output	Low	No	Always
Physical Bus Grant Acknowledge	$\overline{PBGACK}$	Input	Low	—	—
Logical Bus Request In	$\overline{LBRI}$	Input	Low	—	—
Logical Bus Request Out	$\overline{LBRO}$	Output	Low	No	Always
Logical Bus Grant In	$\overline{LBGI}$	Input	Low	—	—
Logical Bus Grant Out	$\overline{LBGO}$	Output	Low	No	Always
Logical Bus Grant Acknowledge	$\overline{LBGACK}$	Input/Output	Low	Yes	MC68851 Has Assumed Mastership of the Logical Bus
Clock	CLK	Input	—	—	—
Power Supply	V_{CC}	Input	—	—	—
Ground	GND	Input	—	—	—

ELECTRICAL SPECIFICATIONS

MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Supply Voltage	V_{CC}	−0.3 to +7.0	V
Input Voltage	V_{in}	−0.5 to +7.0	V
Operating Temperature	T_A	0 to 70	°C
Storage Temperature	T_{stg}	−55 to +150	°C

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

THERMAL CHARACTERISTICS — PGA PACKAGE

Characteristic	Symbol	Value	Rating
Thermal Resistance — Ceramic			
Junction to Ambient	θ_{JA}	30*	°C/W
Junction to Case	θ_{JC}	15*	°C/W

*Estimated

4

POWER CONSIDERATIONS

The average chip-junction temperature, T_J, in °C can be obtained from:

$$T_J = T_A + (P_D \cdot \theta_{JA}) \quad (1)$$

where:

T_A = Ambient Temperature, °C
θ_{JA} = Package Thermal Resistance, Junction-to-Ambient, °C/W
$P_D = P_{INT} + P_{I/O}$
$P_{INT} = I_{CC} \times V_{CC}$, Watts — Chip Internal Power
$P_{I/O}$ = Power Dissipation on Input and Output Pins — User Determined

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

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

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

Solving equations (1) and (2) for K gives:

$$K = P_D \cdot (T_A + 273°C) + \theta_{JA} \cdot P_D^2 \quad (3)$$

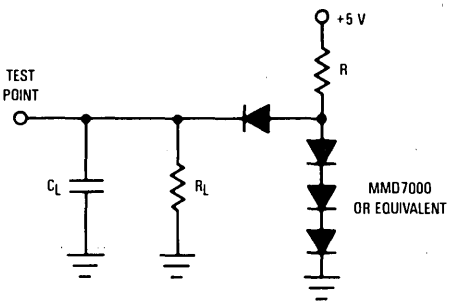
where K is a constant pertaining to the particular part. K can be determined from equation (3) by measuring P_D (at equilibrium) for a known T_A. Using this value of K, the values of P_D and T_J can be obtained by solving equations (1) and (2) iteratively for any value of T_A.

The total thermal resistance of a package (θ_{JA}) can be separated into two components, θ_{JC} and θ_{CA}, representing the barrier to heat flow from the semiconductor junction to the package (case) surface (θ_{JC}) and from the case to the outside ambient (θ_{CA}). These terms are related by the equation:

$$\theta_{JA} = \theta_{JC} + \theta_{CA} \quad (4)$$

θ_{JC} is device related and cannot be influenced by the user. However, θ_{CA} is user dependent and can be minimized by such thermal management techniques as heat sinks, ambient air cooling and thermal convection. Thus, good thermal management on the part of the user can significantly reduce θ_{CA} so that θ_{JA} approximately equals θ_{JC}. Substitution of θ_{JC} for θ_{JA} in equation (1) will result in a lower semiconductor junction temperature.

Values for thermal resistance presented in this document, unless estimated, were derived using the procedure described in Motorola Reliability Report 7843, "Thermal Resistance Measurement Method for MC68XX Microcomponent Devices," and are provided for design purposes only. Thermal measurements are complex and dependent on procedure and setup. User derived values for thermal resistance may differ.



C_L = 130 pF (includes all parasitics)
R_L = 6.0 kΩ
R = 740 Ω for $\overline{DSACK0}$, $\overline{DSACK1}$, D0-D31, $\overline{PAS}$, $\overline{DS}$, R/$\overline{W}$, $\overline{RMC}$, $\overline{BERR}$, $\overline{HALT}$, $\overline{LBGACK}$, DBDIS, $\overline{LBRO}$, $\overline{LBGO}$, $\overline{PBG}$, $\overline{CLI}$
R = 1.22 kΩ for A0-A7, FC0-FC3, SIZ0/SIZ1, PA8-PA31

Figure 7. Test Loads

DC ELECTRICAL CHARACTERISTICS (V_{CC} = 5.0 Vdc ± 5%; GND = 0 Vdc; T_A = 0 to 70°C; see Figure 7)

Characteristic	Symbol	Min	Max	Unit
Input High Voltage	V_{IH}	2.0	V_{CC}	V
Input Low Voltage	V_{IL}	GND r0.5	0.8	V
Input Leakage Current @ 5.25 V — CLK, RESET, LA8rLA31, LAS, LBRI, LBGI, PBR, PBGACK, ASYNC	I_{in}	—	10	μA
Hi-Z (Off-State) Input Current @ 2.4 V/0.4 V — DSACK0, DSACK1, D0rD31, FC0rFC3, SIZ0rSIZ1, PAS, DS, R/W, RMC, BERR, HALT, LBGACK	I_{TSI}	—	20	μA
Output High Voltage (I_{OH} = −400 μA) — A0rA7, DSACK0, DSACK1, D0rD31, FC0rFC3, SIZ0rSIZ1, PAS, DS, R/W, RMC, BERR, HALT, LBGACK, PA8rPA31, DBDIS, LBRO, LBGO, PBG, CLI	V_{OH}	2.4	—	V
Output Low Voltage (I_{OL} = 5.3 mA) — DSACK0, DSACK1, HALT, PAS, DS, R/W, RMC, BERR, LBGACK, DBDIS, LBRO, LBGO, PGB, CLI	V_{OL}	—	0.5	V
Output Low Voltage (I_{OL} = 3.2 mA) — D0rD31, A0rA7, FC0rFC3, SIZ0rSIZ1, PA8rPA31	V_{OL}	—	0.5	V
Power Dissipation	P_D	—	1.50	W
Capacitance* (V_{in} = 0, T_A = 25°C, f = 1 MHz)	C_{in}	—	20	pF

*Capacitance is periodically sampled rather than 100% tested.

AC ELECTRICAL SPECIFICATIONS — CLOCK INPUT (V_{CC} = 5.0 Vdc ± 5%; GND = 0 Vdc; T_A = 0 to 70 °C; see Figure 8)

No.	Characteristic	Symbol	MC68851RC12		MC68851RC16		MC68851RC20		Unit
			Min	Max	Min	Max	Min	Max	
	Frequency of Operation	f	8.0	12.5	8.0	16.67	10	20	MHz
1	Cycle Time	t_{cyc}	80	125	60	125	50	100	ns
2, 3	Clock Pulse Width	t_{CL}, t_{CH}	32	87	24	95	19	81	ns
4, 5	Clock Rise and Fall Time	t_{Cr}, t_{Cf}	—	5	—	5	—	5	ns

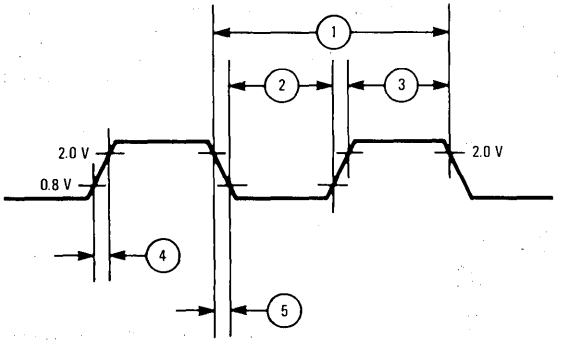


Figure 8. Clock Input Timing Diagram

NOTE

Notes referenced to in the following table have been placed after the final entry.

The timing diagrams (Figures 9 through 17) are intended to provide parametric timing information for the MC68851. Effort has been made to ensure that the diagrams provide correct functional signal relationships. However, not all relationships depicted are valid operations for the MC68851 (e.g., during a CPU space access as shown in Figure 14, accesses to the MC68851 will not cause assertion of $\overline{CLI}$).

AC ELECTRICAL SPECIFICATIONS — ALL BUS OPERATIONS (V_{CC} = 5.0 Vdc ± 5%; GND = 0 Vdc; T_A = 0 to 70°C; see Figures 9 through 17)

No.	Characteristic	Mode	MC68851RC12		MC68851RC16		MC68851RC20		Unit
			Min	Max	Min	Max	Min	Max	
6	Clock High to FC, Size, $\overline{RMC}$, Physical Address, Shared Address Valid (see Note a) (see Figures 9 and 10)	T[b]	0	40	0	30	0	25	ns
7	Clock High to FC, Size, $\overline{RMC}$, Data-Out, Physical Address, Shared Address High Impedance (see Figure 14)	T	0	40	0	30	0	25	ns
8	Clock High to FC, Size, $\overline{RMC}$, Physical Address, Shared Address Invalid (see Figures 9 and 10)	T	0	—	0	—	0	—	ns
9	Clock Transition to $\overline{PAS}$ Asserted (see Figures 9 and 10)	T	0	35	0	25	0	20	ns
9A[k]	$\overline{PAS}$ to $\overline{DS}$ Assertion (Read) (Skew) (see Figure 9)	T	−20	20	−15	15	−10	10	ns
9B	Clock Transition to $\overline{DS}$ Asserted (see Figure 10)	T	0	40	0	30	0	25	ns
11[p]	FC, Size, $\overline{RMC}$, Physical Address, Shared Address Valid to $\overline{PAS}$, $\overline{DS}$ Asserted (see Figures 9 and 10)	T	20	—	15	—	10	—	ns
12	Clock Low to $\overline{PAS}$ Negated (see Figures 9 and 10)	T	0	35	0	25	0	20	ns
12A	Clock Low to $\overline{DS}$ Negated (see Figures 9 and 10)	T	0	40	0	30	0	25	ns
13	$\overline{PAS}$, $\overline{DS}$ Negated to FC, Size, $\overline{RMC}$, Physical Address, Shared Address Invalid (see Note u) (see Figures 9 and 10)	T	20	—	15	—	10	—	ns
14	$\overline{PAS}$, $\overline{DS}$ (Read) Width Asserted (see Figures 9 and 10)	T	120	—	100	—	85	—	ns
14A	$\overline{DS}$ Width Asserted (Write) (see Figure 10)	T	50	—	40	—	35	—	ns
15	$\overline{PAS}$, $\overline{DS}$ Width Negated	T	50	—	40	—	35	—	ns
16	Clock High to $\overline{PAS}$, $\overline{DS}$, R/$\overline{W}$, DBDIS High Impedance (see Figure 9)	T	0	40	0	30	0	25	ns
17	$\overline{PAS}$, $\overline{DS}$ Negated to R/$\overline{W}$ Invalid (Read or Write) (see Figures 9 and 10)	T	20	—	15	—	10	—	ns
18	Clock High to R/$\overline{W}$ High (Read) (see Figure 9)	T	0	40	0	30	0	25	ns
20	Clock High to R/$\overline{W}$ Low (Write) (see Figure 10)	T	0	40	0	30	0	25	ns
21	R/$\overline{W}$ High to $\overline{PAS}$ Asserted (see Figure 9)	T	20	—	15	—	10	0	ns
22	R/$\overline{W}$ Low to $\overline{DS}$ Asserted (Write) (see Figure 10)	T	90	—	75	—	60	—	ns
23	Clock High to Data-Out Valid (see Figures 10 and 15)	T/O	0	40	0	30	0	25	ns

AC ELECTRICAL SPECIFICATIONS — ALL BUS OPERATIONS (Continued)

No.	Characteristic	Mode	MC68851RC12		MC68851RC16		MC6851RC20		Unit
			Min	Max	Min	Max	Min	Max	
25	$\overline{DS}$ Negated to Data-Out Invalid (see Figures 10 and 15)	T/O	20	—	15	—	10	—	ns
26	Data-Out Valid to $\overline{DS}$ Asserted (see Figure 10)	T	20	—	15	—	10	—	ns
27	Data-In Valid to Clock Low (Data Setup) (see Figure 9)	T	10	—	5	—	5	—	ns
27A	$\overline{BERR}$-i/$\overline{HALT}$-i Asserted to Clock Low (Late $\overline{BERR}$/$\overline{HALT}$ Setup Time) (see Note c) (see Figure 9)	T	25	60[d]	20	45[d]	15	35	ns
29	$\overline{DS}$ Negated to Data-In Invalid (Data-In Hold Time) (see Figure 9)	T	0	—	0	—	0	—	ns
29A	$\overline{DS}$ Negated to Data-In High Impedance (see Figure 9)	T	0	80	0	60	0	50	ns
31[l]	$\overline{DSACKx}$ Asserted to Data-In Valid (see Figure 9)	T	—	60	—	50	—	40	ns
31A[m]	$\overline{DSACKx}$ Asserted to $\overline{DSACKx}$ Valid (Assertion Skew) (see Figures 9 and 10)	T	—	20	—	15	—	10	ns
32	$\overline{RESET}$ Input Transition Time (see Figure 17)	X	—	2	—	2	—	2	Clk Per
33	Clock Low to $\overline{PBG}$ Asserted (see Figure 14)	X	0	40	0	30	0	25	ns
34	Clock Low to $\overline{PBG}$ Negated (see Figure 14)	X	0	40	0	30	0	25	ns
35A	$\overline{PBR}$ Asserted to $\overline{PBG}$ Asserted ($\overline{RMC}$ Not Asserted) (see Figure 14)	T	1.5	3.5	1.5	3.5	1.5	3.5	Clk Per
35B	$\overline{PBR}$ Asserted to $\overline{PBG}$ Asserted ($\overline{RMC}$ Not Asserted) (see Figure 14)	M	1.5	5.5	1.5	5.5	1.5	5.5	Clk Per
36	$\overline{PBR}$ Negated to $\overline{PBG}$ Negated (Transient or Spurious Request) (see Figure 14)	X	1.5	3.5	1.5	3.5	1.5	3.5	Clk Per
37	$\overline{PBGACK}$ Asserted to $\overline{PBG}$ Negated (see Figure 14)	X	1.5	3.5	1.5	3.5	1.5	3.5	Clk Per
39	$\overline{PBG}$ Width Negated (see Figure 14)	X	1.5	—	1.5	—	1.5	—	Clk Per
39A	$\overline{PBG}$ Width Asserted (see Figure 14)	X	1.5	—	1.5	—	1.5	—	Clk Per
40A	Clock High to DBDIS Negated (Read) (see Figure 9)	T	0	40	0	30	0	25	ns
40B	Clock Low to DBDIS Negated (Write (T)) (Read (O)) (see Figure 10)	T/O	0	40	0	30	0	25	ns
41A	Clock Low to DBDIS Asserted (Read (T)) (Write (O)) (see Figures 9 and 15)	T/O	0	40	0	30	0	25	ns
41B	Clock High to DBDIS Asserted (Write) (see Figures 10 and 15)	T	0	40	0	30	0	25	ns
43	$\overline{PBGACK}$ Negated to $\overline{PAS}$, Physical Address Impedance Change (see Note j) (see Figure 14)	X	0.5	2.5	0.5	2.5	0.5	2.5	Clk Per
44	R/$\overline{W}$ Asserted to DBDIS Negated (Read or Write) (see Figures 9 and 10)	T	20	—	15	—	10	—	ns
45A	DBDIS Width Negated (Read) (see Figure 9)	T	80	—	60	—	50	—	ns
45B[o]	DBDIS Width Negated (Write) (see Figure 10)	T	160	—	120	—	100	—	ns
46	R/$\overline{W}$ Width Asserted (Read or Write) (see Figures 9 and 10)	T	180	—	150	—	125	—	ns
47A	Asynchronous Input Setup Time to Sampling Clock Edge (see Figures 9, 10, and 14)	T	10	60[d]	5	45[d]	5	35	ns

AC ELECTRICAL SPECIFICATIONS — ALL BUS OPERATIONS (Continued)

No.	Characteristic	Mode	MC68851RC12		MC68851RC16		MC68851RC20		Unit
			Min	Max	Min	Max	Min	Max	
47B	$\overline{PAS}$, $\overline{DS}$ Negated to Asynchronous Input Negated (see Figures 9 and 10)	T	0	100	0	80	0	65	ns
48[n]	$\overline{DSACK}$x-i Asserted to $\overline{BERR}$-i/$\overline{HALT}$-i Asserted (Late Bus Error or Retry) (see Figures 9 and 10)	T	—	40	—	30	—	25	ns
53	Data-Out Hold from Clock High (see Figure 10)	T/O	0	—	0	—	0	—	ns
53A	Data-Out Hold from $\overline{LAS}$ Negated (see Figure 15)	O	0	—	0	—	0	—	ns
55	R/$\overline{W}$ Low to Data Bus Impedance Change (see Figure 10)	T	40	—	30	—	25	—	ns
56	DBDIS Asserted to Data Bus Impedance Change (see Figure 15)	O	15	—	15	—	15	—	ns
56A	Data Bus Impedance Change to DBDIS Negated (see Figure 15)	O	0	—	0	—	0	—	ns
59	DBDIS High to R/$\overline{W}$ Low (see Figure 9)	T	20	—	15	—	10	—	ns
60	$\overline{BERR}$-i Negated to $\overline{HALT}$-i Invalid (Hold Time for Retry) (see Figure 9)	T	0	—	0	—	0	—	ns
63	$\overline{DS}$ Negated to DBDIS Asserted (Write) (see Figure 10)	T	20	—	15	—	10	—	ns
64	DBDIS Negated to Data Bus Impedance Change (Write) (see Figure 10)	T	20	—	15	—	10	—	ns
65	Clock Low to $\overline{LBGACK}$-o, $\overline{LBGO}$ Asserted (see Figures 12 and 13)	X	0	40	0	30	0	25	ns
66	Clock Low to $\overline{LBGACK}$-o, $\overline{LBGO}$ Negated (see Figures 12 and 13)	X	0	40	0	30	0	25	ns
67	$\overline{LBGACK}$-o Asserted to $\overline{LBRO}$ Negated (see Figure 12)	T	20	60	15	45	10	35	ns
68	$\overline{LBGACK}$-o Asserted to $\overline{CLI}$ Asserted (see Figure 12)	T	0	80	0	60	0	50	ns
69	$\overline{CLI}$ Negated to $\overline{LBGACK}$-o Negated (see Figure 12)	T	0	80	0	60	0	50	ns
70	$\overline{LBGACK}$-o Asserted to DBDIS Asserted (see Figure 12)	T	−20	20	−15	15	−10	10	ns
71	$\overline{LBGI}$ Asserted to $\overline{LBGO}$ Asserted (see Figure 13)	X	1.5	12.5[h]	1.5	12.5[h]	1.5	12.5[h]	Clk Per
72	$\overline{LBRI}$ Negated to $\overline{LBGO}$ Negated (see Figure 13)	X	1.5	3.5	1.5	3.5	1.5	3.5	Clk Per
73	$\overline{LBGO}$ Width Asserted (see Figure 13)	X	40	—	30	—	25	—	ns
74	$\overline{LBGO}$ Width Negated (see Figure 13)	X	40	—	30	—	25	—	ns
75	$\overline{LBGI}$ Negated to $\overline{LBGO}$ Negated (see Figure 13)	X	1.5	3.5	1.5	3.5	1.5	3.5	Clk Per
77	$\overline{LBGI}$ Asserted to $\overline{LBGACK}$-o Asserted (see Figure 12)	T	1.5	3.5	1.5	3.5	1.5	3.5	Clk Per
79	$\overline{RESET}$ Width Asserted (V_{CC} Active and Stable) (see Figure 17)	X	512	—	512	—	512	—	Clk Per
79A	$\overline{RESET}$ Width Asserted (V_{CC} Stable > 512 Clocks) (see Figure 17)	X	10	—	10	—	10	—	Clk Per

4

AC ELECTRICAL SPECIFICATIONS — ALL BUS OPERATIONS (Continued)

No.	Characteristic	Mode	MC68851RC12		MC68851RC16		MC68851RC20		Unit
			Min	Max	Min	Max	Min	Max	
80	$\overline{RESET}$ Asserted to Bus Control Signals Negated (V_{CC} Active and Stable) (see Figure 17)	X	0	4	0	4	0	4	Clk Per
81	$\overline{RESET}$ Negated to $\overline{LAS}$ Asserted (see Figure 17)	X	4	—	4	—	4	—	Clk Per
82	$\overline{RESET}$ Negated to Mode Select Data Invalid (Hold) (see Figure 17)	X	0	—	0	—	0	—	ns
83	Mode Select Data Valid to $\overline{RESET}$ Negated (Setup) (see Figure 17)	X	2	—	2	—	2	0	Clk Per
84	Clock Transition to $\overline{HALT}$/$\overline{BERR}$/$\overline{LBRO}$ Asserted (Logical Master Relinquish and Retry) (see Figure 11)	M	0	40	0	30	0	25	ns
86A	$\overline{LAS}$ Asserted to $\overline{HALT}$/$\overline{BERR}$/$\overline{LBRO}$ Asserted (Logical Master Relinquish and Retry) (see Figure 12)	MS	0.5	1.5[g]	0.5	1.5[g]	0.5	1.5[g]	Clk Per
86B	$\overline{LAS}$ Asserted to $\overline{HALT}$/$\overline{BERR}$/$\overline{LBRO}$ Asserted (Logical Master Relinquish and Retry) (see Figure 12)	MA	.5	3.0[f]	.5	3.0[f]	.5	3.0[f]	Clk Per
89	$\overline{HALT}$ Negated to $\overline{LBGACK}$-o Asserted (see Figure 12)	M	20	60	15	45	10	35	ns
90	$\overline{LAS}$ Negated to $\overline{BERR}$-o Negated (Termination of Relinquish and Retry) (see Figure 12)	M	0	40	0	30	0	25	ns
91	Logical Address, FC, $\overline{RMC}$, R/$\overline{W}$ Valid to Clock High (Setup) (see Figure 11)	MS/OS	40	—	30	—	25	—	ns
92A	Logical Address, FC, $\overline{RMC}$, R/$\overline{W}$, Valid to $\overline{LAS}$ Asserted (see Figures 11, 15, and 16)	MS/OS	20	—	15	—	10	—	ns
92B	Logical Address, FC, $\overline{RMC}$, R/$\overline{W}$ Valid to $\overline{LAS}$ Asserted (see Figures 13, 15, and 16)	MA/OA	0	—	0	—	0	—	ns
93A	$\overline{LAS}$ Negated to Logical Address, FC, $\overline{RMC}$, R/$\overline{W}$ Invalid (Synch Mode) (see Figures 11, 15, and 16)	MS/OS	20	—	15	—	10	—	ns
93B	$\overline{LAS}$ Negated to Logical Address, FC, $\overline{RMC}$, R/$\overline{W}$ Invalid (Asynch Mode) (see Figures 13, 15, and 16)	MA/OA	0	—	0	—	0	—	ns
95	Logical Address Valid to Physical Address Valid (Translation Cache Hit or CPU Space Cycle) (see Figures 11, 13, 15, and 16)	M	0	50[s]	0	45[s]	0	38	ns
96	Size, Shared Address Valid to $\overline{LAS}$ Asserted (Access to MC68851 Register) (see Figures 15 and 16)	OS	20	—	15	—	10	—	ns
97	$\overline{LAS}$ Negated to Size, Shared Address Invalid (Access to MC68851 Register) (see Figures 15 and 16)	OS	20	—	15	—	10	—	ns
100	$\overline{LAS}$ Asserted to Clock Low (Setup Time) (see Figure 11)	MS	40	—	30	—	25	—	ns
103	$\overline{LAS}$ Width Asserted (see Figures 11 and 13)	M	1.5	—	1.5	—	1.5	—	Clk Per
104	$\overline{LAS}$, $\overline{DS}$ Width Negated (see Figure 11)	MS	0.5	—	0.5	—	0.5	—	Clk Per
104A	$\overline{LAS}$, $\overline{DS}$ Width Negated (see Figure 13)	MA	30	—	20	—	10	—	ns
105	$\overline{ASYNCH}$ Asserted to $\overline{LAS}$, $\overline{DS}$ Asserted (see Figure 13)	M	1.5	—	1.5	—	1.5	—	Clk Per

AC ELECTRICAL SPECIFICATIONS — ALL BUS OPERATIONS (Continued)

No.	Characteristic	Mode	MC68851RC12		MC68851RC16		MC68851RC20		Unit
			Min	Max	Min	Max	Min	Max	
106	$\overline{LAS}$, $\overline{DS}$ Negated to $\overline{ASYNCH}$ Negated (see Figures 11 and 13)	M	0	—	0	—	0	—	ns
107	$\overline{ASYNCH}$ Negated to $\overline{LAS}$, $\overline{DS}$ Asserted (For Synchronous Next Cycle) (see Figure 11)	M	1.5	—	1.5	—	1.5	—	Clk Per
108	Data Valid to $\overline{DS}$ Asserted (Write Setup Time to MC68851) (see Figure 16)	O	0	—	0	—	0	—	ns
109A	$\overline{DS}$ Negated to Data Invalid (Write Hold Time to MC68851) (see Figure 16)	O	0	—	0	—	0	—	ns
109B	$\overline{LAS}$ Negated to Data High Impedance (see Figure 16)	O	0	80	0	60	0	50	ns
110	$\overline{DSACKx}$-o Asserted to $\overline{DSACKy}$-o Valid (see Figures 15 and 16)	O	0	20[r]	0	15[r]	0	10[r]	ns
111	Clock High to $\overline{DSACKx}$-o Asserted (see Figures 15 and 16)	O	0	40	0	30	0	25	ns
112A	$\overline{LAS}$ Asserted to $\overline{DSACKx}$-o Asserted (see Figures 15 and 16)	OS	2.0	23	2.0	23	2.0	23	Clk Per
112B	$\overline{LAS}$ Asserted to $\overline{DSACKx}$-o Asserted (see Figures 15 and 16)	OA	2.0	26	2.0	26	2.0	26	Clk Per
113	$\overline{LAS}$ Negated to $\overline{DSACKx}$-o, $\overline{BERR}$-o Negated (see Figures 15 and 16)	O	0	40	0	30	0	25	ns
114	$\overline{LAS}$ Negated to $\overline{DSACKx}$-o, $\overline{BERR}$-o High Impedance (see Figure 15)	O	0	60	0	40	0	30	ns
115	Clock Low to $\overline{PAS}$ Asserted (see Figure 11)	MS	0[e]	35[e]	0[e]	25[e]	0[e]	20[e]	ns
116	Clock Transition (Rising or Falling Edge) to $\overline{PAS}$ Asserted (see Figures 11 and 13)	M	0[e]	35[e]	0[e]	25[e]	0[e]	20[e]	ns
116A	Clock Low to $\overline{CLI}$ Asserted (see Figures 11 and 15)	M	0[e]	40[e]	0[e]	30[e]	0[e]	25[e]	ns
117	$\overline{LAS}$ Asserted to $\overline{PAS}$ Asserted (Synchronous Translation with ATC Hit) (see Figure 11)	MS	0.5[g]	1.5[g]	0.5[g]	1.5[g]	0.5[g]	1.5[g]	Clk Per
118	$\overline{LAS}$ Negated to $\overline{PAS}$ Negated (see Figures 11 and 13)	M	0	20	0	15	0	10	ns
119	Physical Address Valid to $\overline{PAS}$ Asserted (see Figure 11)	M	20	—	15	—	10	—	ns
120A	$\overline{PAS}$ Negated to Physical Address Invalid (see Figure 13)	MS	15	—	10	—	10	—	ns
120B	$\overline{PAS}$ Negated to Physical Address Invalid (see Figure 13)	MA	0	—	0	—	0	—	ns
121	$\overline{LAS}$ Asserted to $\overline{PAS}$ Asserted (Asynchronous Operation Only) (see Figure 13)	M	0.5[f]	3.0[i]	0.5[f]	3.0[i]	0.5[f]	3.0[i]	Clk Per
122	$\overline{LAS}$ Negated to $\overline{PAS}$ High Impedance ($\overline{PBR}$ Asserted by Alternate Physical Master) (see Figures 11 and 14)	M	—	80	—	60	—	50	ns
123	Physical Address Valid to $\overline{CLI}$ Asserted (CPU Space Cycle Not Accessing MC68851) (see Figures 15 and 16)	M	20	—	15	—	10	—	ns
124A	$\overline{LAS}$ Asserted to $\overline{CLI}$ Asserted (CPU Space Cycle Not Accessing MC68851) (see Figures 15 and 16)	MA	0.5[v]	3.0[v]	0.5[v]	3.0[v]	0.5[v]	3.0[v]	Clk Per

4

AC ELECTRICAL SPECIFICATIONS — ALL BUS OPERATIONS (Concluded)

No.	Characteristic	Mode	MC68851RC12		MC68851RC16		MC68851RC20		Unit
			Min	Max	Min	Max	Min	Max	
124B	$\overline{LAS}$ Asserted to $\overline{CLI}$ Asserted (CPU Space Cycle Not Accessing MC68851) (see Figures 15 and 16)	MS	0.5[V]	1.5[V]	0.5[V]	1.5[V]	0.5[V]	1.5[V]	Clk Per
126	$\overline{LAS}$ Negated to $\overline{CLI}$ Negated (CPU Space Cycle Not Accessing MC68851) (see Figures 15 and 16)	M	0	40	0	30	0	25	ns
127	$\overline{CLI}$ Negated to Physical Address Invalid (CPU Space Cycle Not Accessing MC68851) (see Figures 15 and 16)	MS	10	—	5	—	5	—	ns
128	$\overline{PAS}$ Asserted to $\overline{CLI}$ Asserted (Not CPU Space Access) (see Figure 11)	M	—	20	—	15	—	10	ns
129	$\overline{PAS}$ Negated to $\overline{CLI}$ Negated (Not CPU Space Access) (see Figure 11)	M	5	40	5	30	5	25	ns

4

NOTES:

a) In this specification the terms 'high', 'low', 'asserted', 'negated', 'valid', and 'invalid' are used frequently to describe a signal state. For inputs to the MC68851, 'high' indicates that the signal conforms to the V_{IH} voltage specification while 'low' indicates that the V_{IL} specification is satisfied. Similarly, a MC68851 output is 'high' if it conforms to the V_{OH} specification and 'low' if it conforms to the V_{OL} parameter. An active-low input (output) is asserted if it satisfies the respective V_{IL} (V_{OL}) requirements and negated if it satisfies the V_{IH} (V_{OH}) specification. A signal is 'valid' if it conforms to either the voltage high or the voltage low specifications and is an appropriate value for the current operation (for example, R/$\overline{W}$ should output a valid low during an MC68851 initiated write cycle). A signal is 'invalid' if it either does not conform to the V_H or V_L specifications or is an inappropriate value for the current operation as above.

b) In order to better understand the parameters given, a 'mode' identification is included with each specification: **X** indicates that this specification is valid in any operating mode whatever; **T** indicates that the MC68851 is the current bus master and is performing a table search operation; **M** indicates that the MC68851 is mapping translations for the current bus master with the designation **MS** indicating that the master is operating synchronously with the MC68851, **MA** indicating an asynchronous master, and **MX** indicating that the parameter is valid for any type of logical master; **O** indicates that the parameter is valid for operations which access the internal registers of the MC68851.

c) Due to the numerous MC68851 signals that are used as inputs in one operating mode and as outputs in another, some attempt has been made to clarify whether a particular signal is acting as an input or as an output in cases where ambiguity is possible. The suffix "-o" indicates that the signal is an output of the MC68851 while the suffix "-i" indicates that this signal is acting as an input to the MC68851.

d) The maximum value for parameter #47A is specified in order that the system designer may deterministically identify the clock edge on which an asynchronous input to the MC68851 will be recognized. Any signal that meets the minimum specified setup time to an appropriate clock edge (rising/falling) for that signal to be recognized on, and does not exceed the maximum time, is guaranteed to be recognized as asserted on that edge. Signals that do not meet the minimum setup time may or may not be recognized; signals that exceed the maximum specified setup time may be recognized on the previous rising/falling clock edge.

e) The actual assertion delay from the low-going clock edge that causes the strobe(s) to assert includes the time specified in the parameter plus any additional delay specified on D3/D4 during MC68851 configuration at RESET.

f) The actual assertion delay from the assertion of $\overline{LAS}$ when mapping in the asynchronous mode is the time specified in the parameter plus any additional delay specified on D3/D4 during MC68851 configuration at RESET.

g) The actual assertion delay from the assertion of $\overline{LAS}$ is the time specified in the parameter plus any additional delay specified on D3/D4 during MC68851 configuration at RESET. This specification has a range of one clock period in order to allow for cases in which the CPU exhibits a best-case (minimum) assertion delay for the $\overline{LAS}$ signal relative to the clock while the MC68851 $\overline{PAS}$ or $\overline{CLI}$ outputs exhibit worst-case (maximum) assertion delays. When operating in the synchronous translation mode, the MC68851 asserts $\overline{PAS}$ ($\overline{CLI}$) on the falling edge of the clock (plus additional specified delay) one clock period after the CPU drives $\overline{LAS}$.

h) The worst case assertion delay for this specification can be reduced to 5.5 clock periods if the early processing startup mode of operation is disabled.

i) This maximum can be reduced to 2.5 clock periods if the logical address strobe ($\overline{LAS}$) high time (negated period) is one clock period or greater.

j) This specification also applies to the signals A0rA7, FC0rFC3, SIZ0rSIZ1, and $\overline{RMC}$ if the MC68851 is awaiting the negation of $\overline{PBGACK}$ to initiate or complete a table search operation.

k) This number can be reduced to ±5 nanoseconds if the strobes have equal load.

NOTES:

l) If the asynchronous setup time (#47) requirements are satisfied, the $\overline{\text{DSACKx}}$ low to data setup (#31) and $\overline{\text{DSACKx}}$ low to $\overline{\text{BERR}}$ low setup time (#48) can be ignored. The data must only satisfy the data-in to clock low setup time (#27) for the following clock cycle. $\overline{\text{BERR}}$ must only satisfy the late $\overline{\text{BERR}}$ low to clock low setup time (#27A) for the following clock cycle.

m) This parameter specifies the maximum allowable skew between $\overline{\text{DSACK0}}$ to $\overline{\text{DSACK1}}$ asserted or $\overline{\text{DSACK1}}$ to $\overline{\text{DSACK0}}$ asserted. Specification #47 must be met by either $\overline{\text{DSACK0}}$ or $\overline{\text{DSACK1}}$.

n) In the absence of $\overline{\text{DSACKx}}$, $\overline{\text{BERR}}$ is an asynchronous input using the asynchronous input setup time (#47).

o) DBDIS may stay asserted on consecutive write cycles (e.g., a retry of an MC68851 write operation).

p) Actual value depends on the clock input waveform.

q) This number can be reduced to 5 nanoseconds if $\overline{\text{CLI}}$ and $\overline{\text{PAS}}$ have equal loading.

r) This specification is valid only if the loading of the $\overline{\text{DSACKx}}$ outputs are equal (±50 pF).

s) This specification can be reduced to 35 ns or 50 ns at 16.67 and 12.5 MHz, respectively for those bits of the logical address that are not translated by the MC68851. This includes all bits of the logical address if the MC68851 translation mechanism is disabled, and all bits, LA*n*, of the logical address (page size 2^m) such that $n \leq m$.

u) This specification also applies to the signals A0rA7, FC0rFC3, and SIZ0rSIZ1 if the MC68851 is granting physical bus mastership to an alternate device during a table search operation.

v) The actual assertion delay from the assertion of $\overline{\text{LAS}}$ is the time specified in the parameter plus a delay derived from the reset configuration. Although the reset configuration allows additional strobe delay in half clock increments, $\overline{\text{CLI}}$ is always asserted relative to the falling edge of the clock and so can only be delayed by full clock increments. Therefore, if a 1 or 2 half clock delay is specified in the reset configuration, then $\overline{\text{CLI}}$ assertion is delayed by 1 full clock.

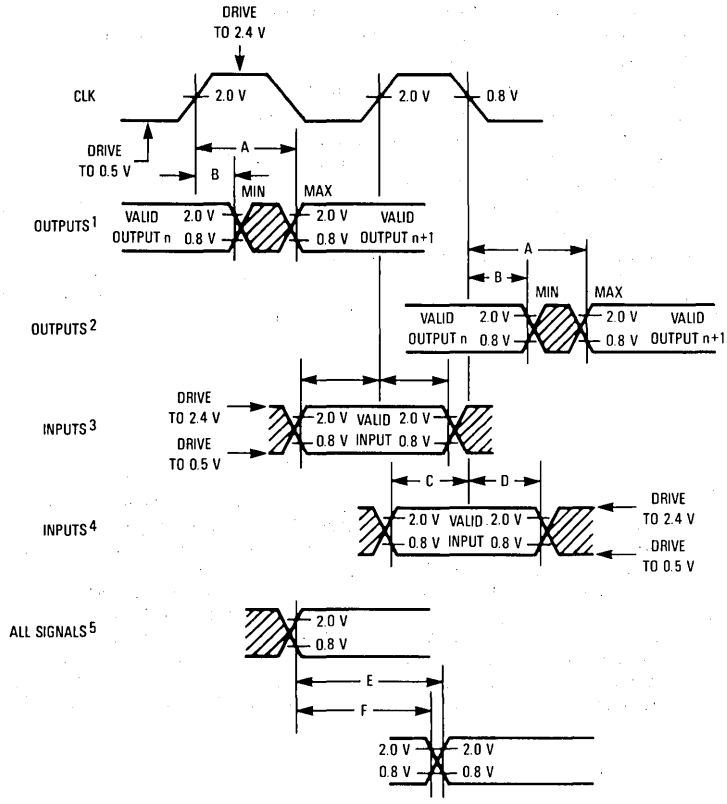
4

AC ELECTRICAL SPECIFICATION DEFINITIONS

The AC specifications presented consist of output delays, input setup and hold times, and signal skew times. All signals are specified relative to an appropriate edge of the MC68851 clock input and, possibly, relative to one or more other signals.

The measurement of the AC specifications is defined by the waveforms below. In order to test the parameters guaranteed by Motorola, inputs must be driven to the voltage levels specified. Outputs of the MC68851 are specified with minimum and/or maximum limits, as appropriate, and are measured as shown. Inputs to the MC68851 are specified with minimum and, as appropriate maximum setup and hold times, and are measured as shown. Finally, the measurement for signal-to-signal specifications are also shown.

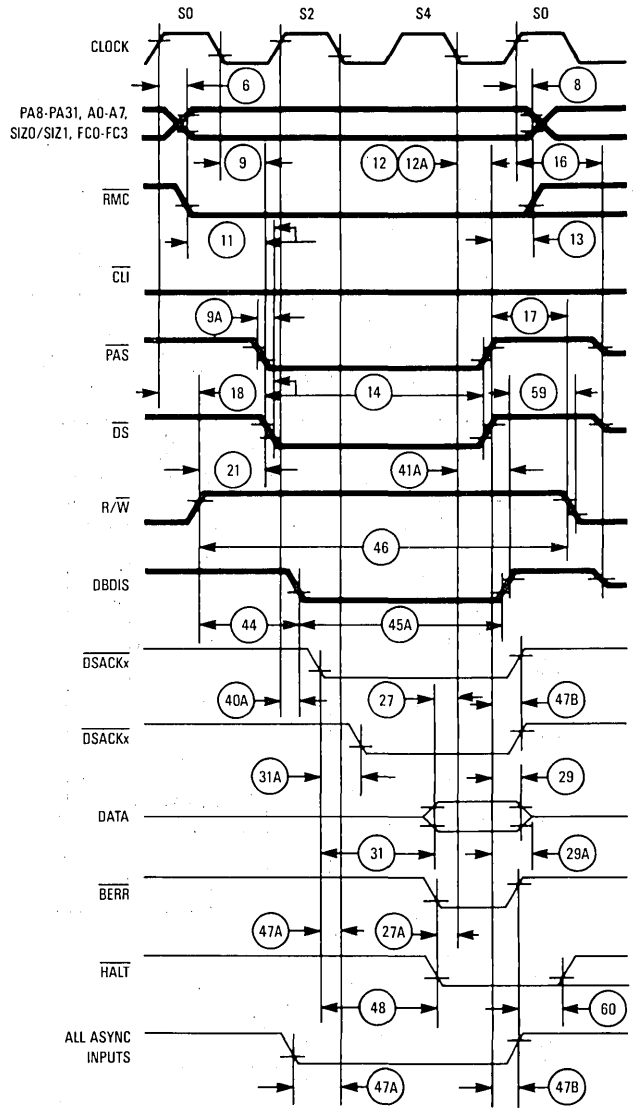
Note that the testing levels used to verify conformance of the MC68851 to the AC specifications does not affect the guaranteed DC operation of the device as specified in the DC electrical characteristics.



Notes:
1 - This output timing is applicable to all parameters specified relative to the rising edge of the clock
2 - This output timing is applicable to all parameters specified relative to the falling edge of the clock
3 - This input timing is applicable to all parameters specified relative to the rising edge of the clock
4 - This input timing is applicable to all parameters specified relative to the falling edge of the clock
5 - This timing is applicable to all parameters specified relative to the assertion/negation of another signal

Legend:
A - Maximum output delay specification
B - Minimum output delay specification
C - Minimum input setup time specification
D - Minimum input hold specification
E - Signal valid to signal valid specification (maximum or minimum)
F - Signal valid to signal invalid specification (maximum or minimum)

Drive Levels and Test Points for AC Specifications



Key: ━━ Indicates that the signal is driven by the MC68851
── Indicates that the signal is driven by the Main Processor
── Indicates that the signal is driven by an external device or alternate bus master

Note: The Clock Signal is always depicted with a normal width line

Figure 9. MC68851 Initiated Read Cycle

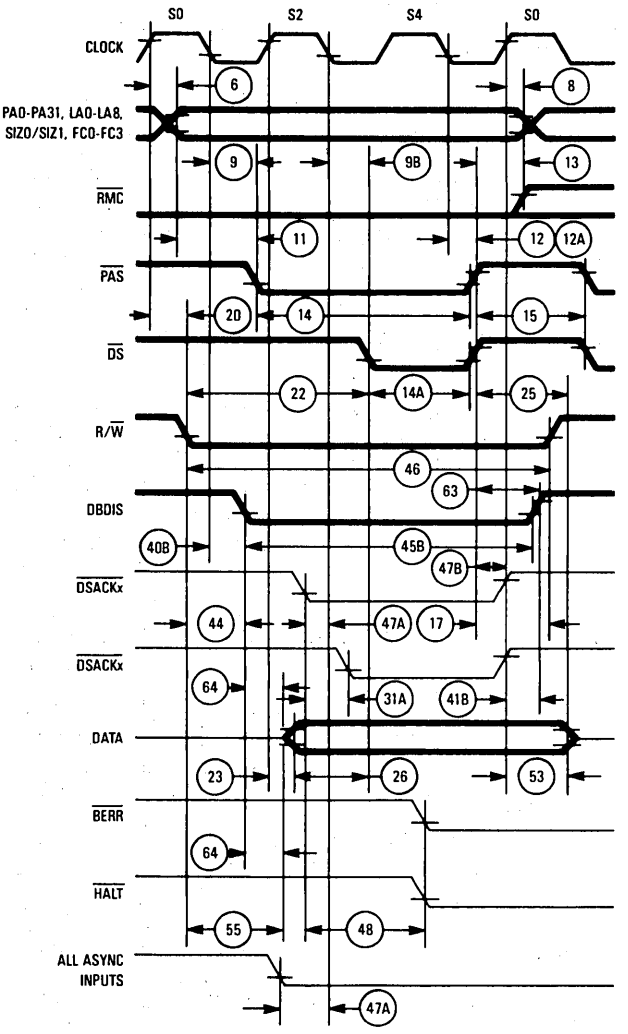


Figure 10. MC68851 Initiated Write Cycle

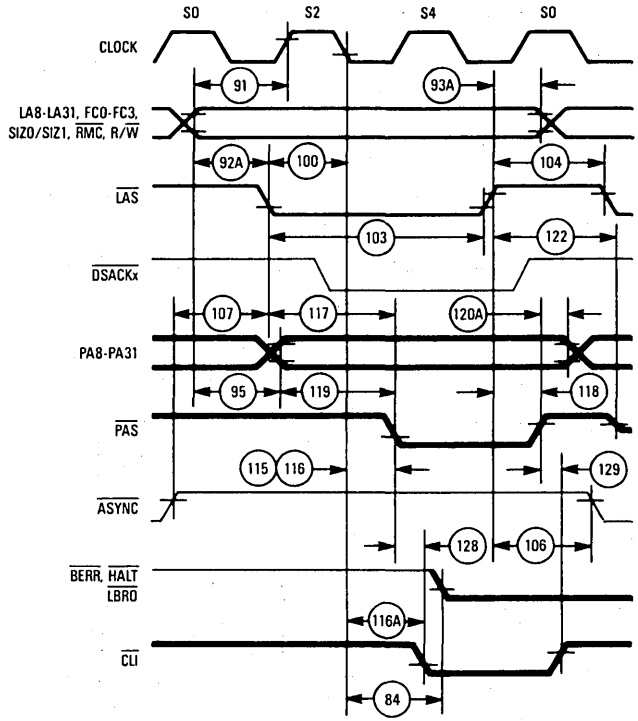


Figure 11. Synchronous Mode Translation

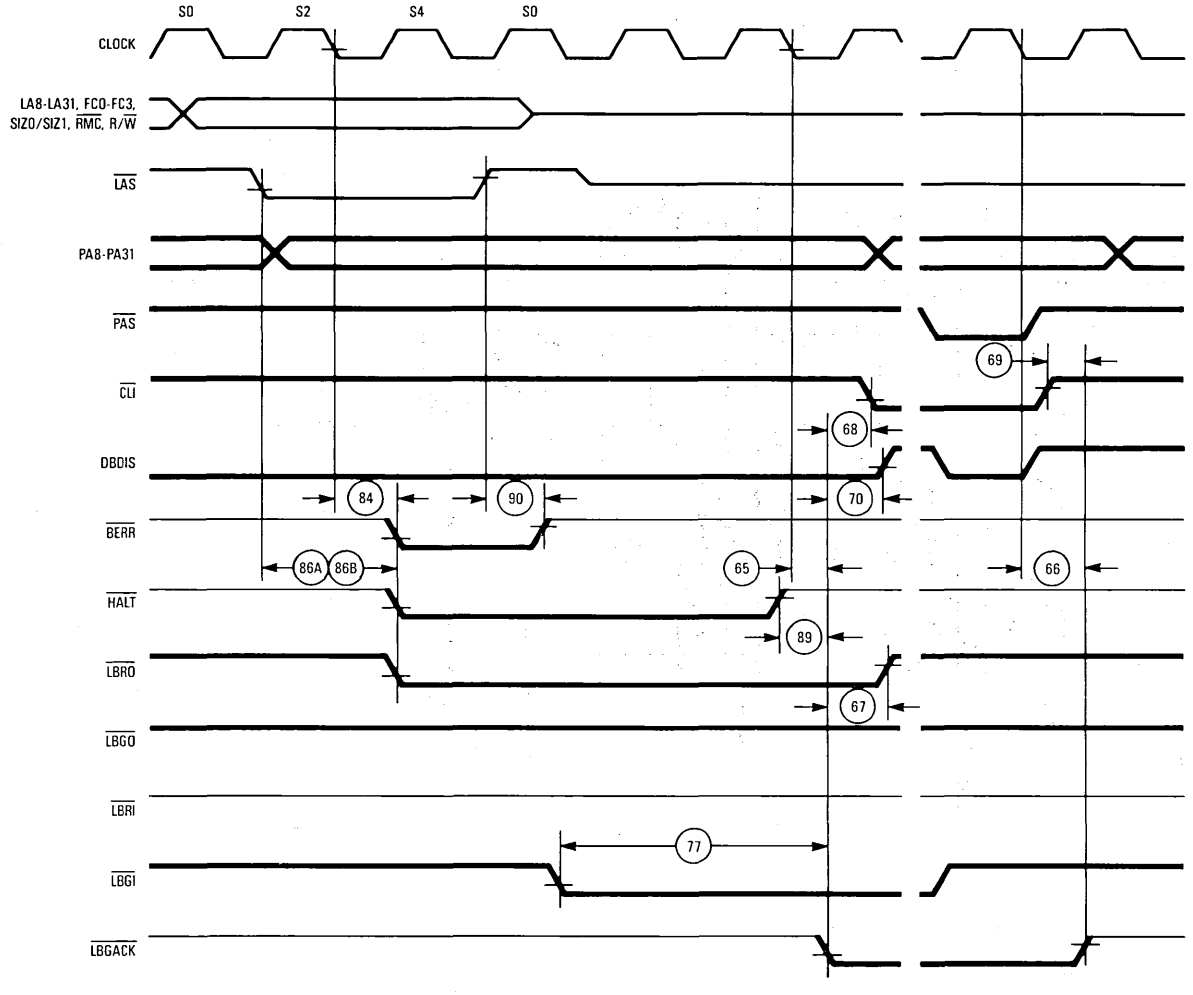


Figure 12. Logical Master Relinquish and Retry Timing Diagram

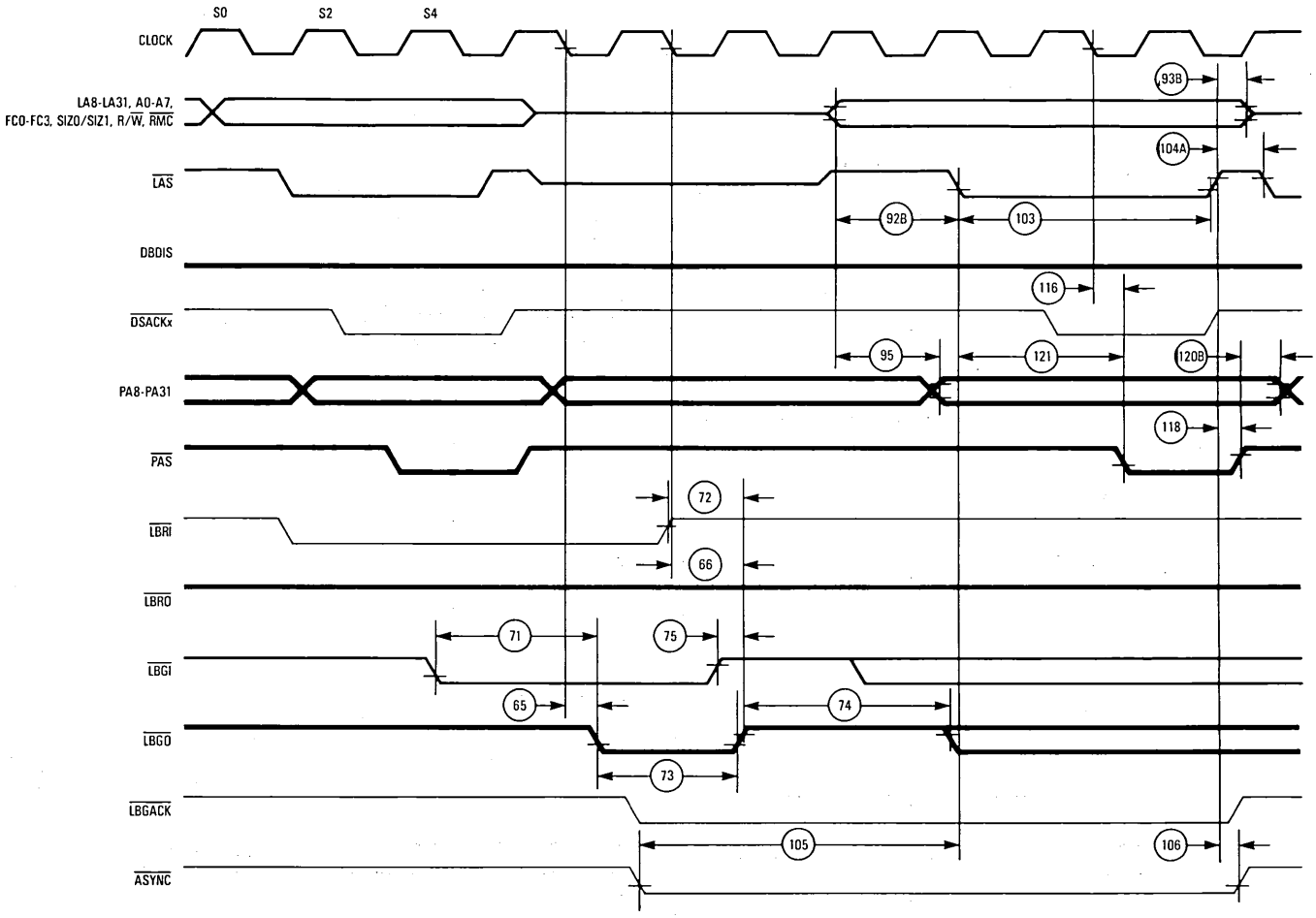


Figure 13. Logical Bus Arbitration by Asynchronous Master Timing Diagram

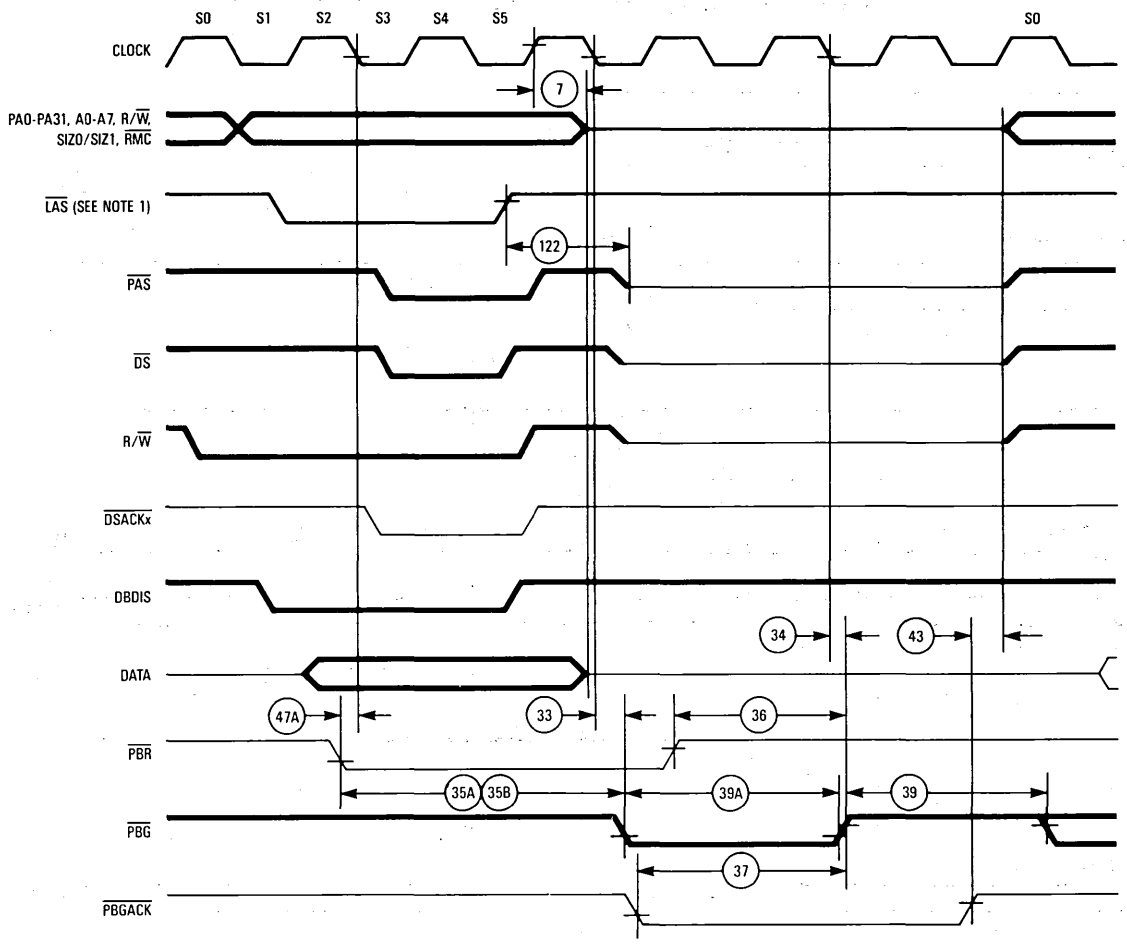


Figure 14. Physical Bus Arbitration Timing Diagram

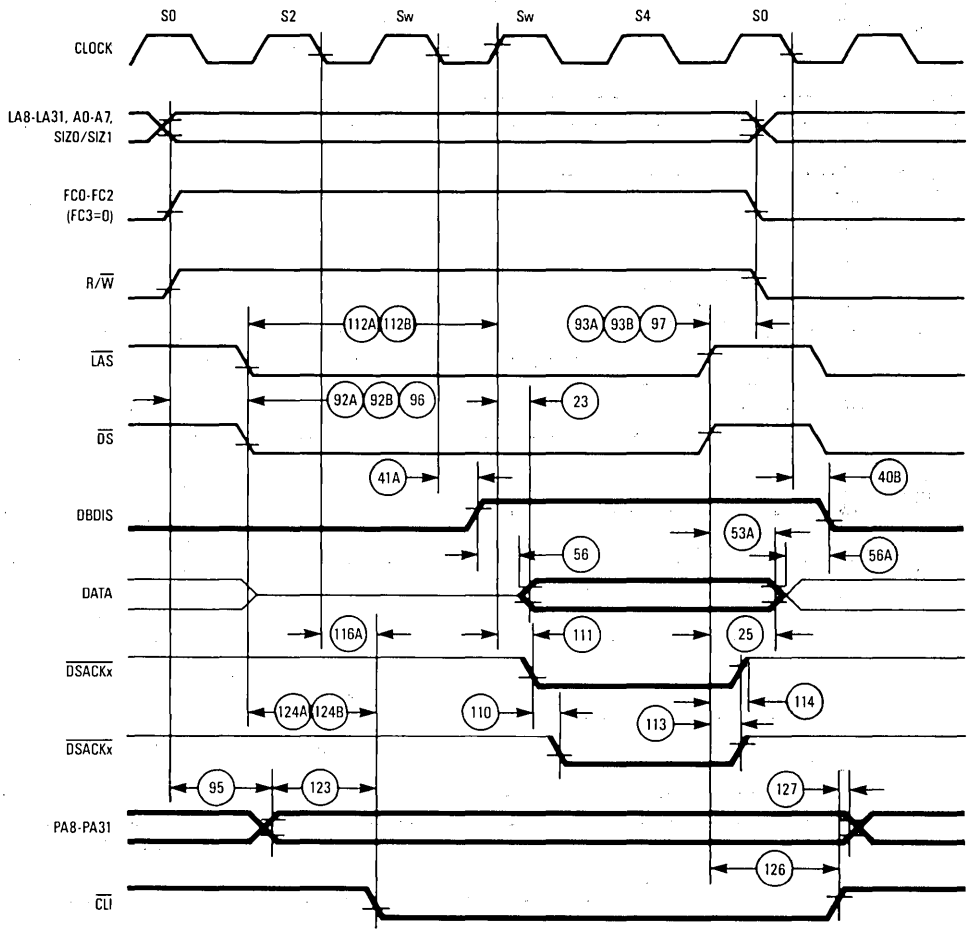


Figure 15. CPU Space Read From MC68851 or From Other Coprocessor (CLI Asserted by MC68851) Timing Diagram

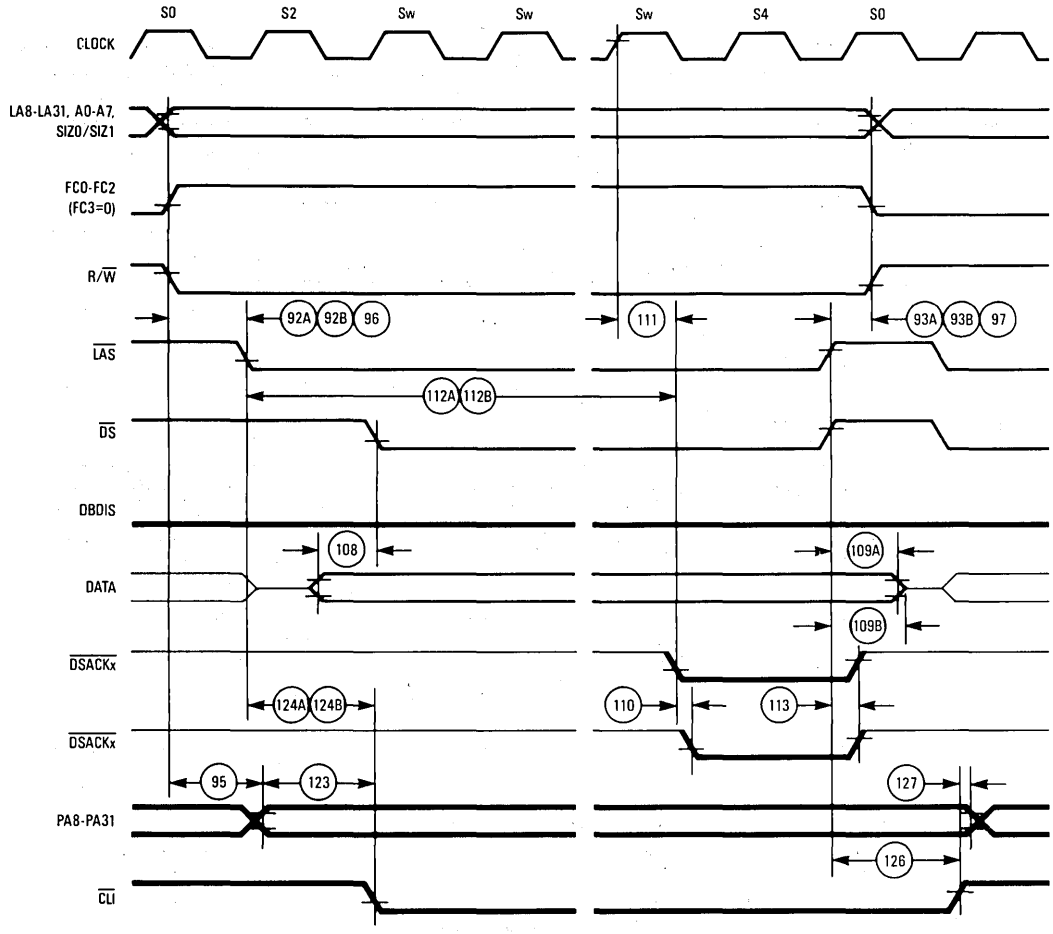


Figure 16. CPU Space Write To MC68851 or To Other Coprocessor (CLI Asserted by MC68851) Timing Diagram

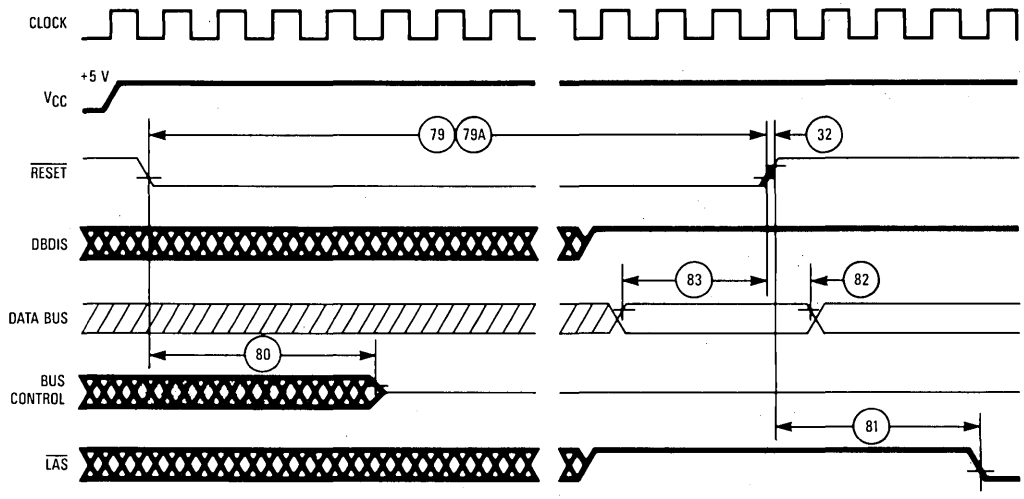
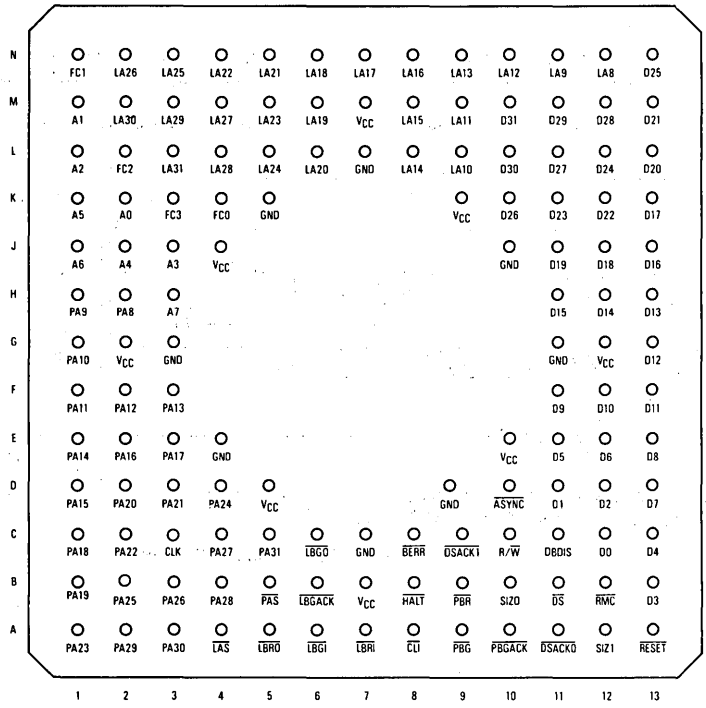


Figure 17. Reset and Mode Select Timing Diagram

PIN ASSIGNMENTS



The V_{CC} and GND pins are separated into four groups to provide individual power supply connections for the address bus buffers, data bus buffers, and all other output buffers and internal logic.

Pin Group	V_{CC}	GND
Physical Address	D5, G2, J4	E4, G3, K5
Logical Address, Internal Logic	M7	L7
D0–D31	E10,0G12, K9	D9, G11, J10
Internal Logic, Clocks	B7	C7

MC68881

Technical Summary

HCMOS Floating-Point Coprocessor

The MC68881 floating-point coprocessor is a full implementation of the *IEEE Standard for Binary Floating-Point Arithmetic* (754) for use with the Motorola M68000 Family of microprocessors. It is implemented using VLSI technology to give systems designers the highest possible functionality in a physically small device.

Intended primarily for use as a coprocessor to the MC68020 32-bit microprocessor unit (MPU), the MC68881 provides a logical extension to the main MPU integer data processing capabilities. It does this by providing a very high performance floating-point arithmetic unit and a set of floating-point data registers that are utilized in a manner that is analogous to the use of the integer data registers. The MC68881 instruction set is a natural extension of all earlier members of the M68000 Family, and supports all of the addressing modes of the host MPU. Due to the flexible bus interface of the M68000 Family, the MC68881 can be used with any of the MPU devices of the M68000 Family, and it may also be used as a peripheral to non-M68000 processors.

The major features of the MC68881 are:

- Eight general purpose floating-point data registers, each supporting a full 80-bit extended precision real data format (a 64-bit mantissa plus a sign bit, and a 15-bit signed exponent).
- A 67-bit arithmetic unit to allow very fast calculations, with intermediate precision greater than the extended precision format.
- A 67-bit barrel shifter for high-speed shifting operations (for normalizing etc.).
- Forty-six instructions, including 35 arithmetic operations.
- Full conformation to the IEEE 754 standard, including all requirements and suggestions.
- Support of functions not defined by the IEEE standard, including a full set of trigonometric and transcendental functions.
- Seven data types: byte, word and long integers; single, double, and extended precision real numbers; and packed binary coded decimal string real numbers.
- Twenty-two constants available in the on-chip ROM, including π, e, and powers of 10.
- Virtual memory/machine operations.
- Efficient mechanisms for procedure calls, context switches, and interrupt handling.
- Fully concurrent instruction execution with the main processor.
- Use with any host processor, on an 8-, 16-, or 32-bit data bus.

THE COPROCESSOR CONCEPT

The MC68881 functions as a coprocessor in systems where the MC68020 or MC68030 is the main processor via the M68000 coprocessor interface.* It functions as a peripheral processor in systems where the main processor is the MC68000, MC68008, or MC68010.

The MC68881 utilizes the M68000 Family coprocessor interface to provide a logical extension of the MC68020 registers and instruction set in a manner which is transparent to the programmer. The programmer perceives the MC68020/MC68881 execution model as if both devices are implemented on one chip.

A fundamental goal of the M68000 Family coprocessor interface is to provide the programmer with an execution model based upon sequential instruction execution by the MC68020 and the MC68881. For optimum performance, however, the coprocessor interface allows concurrent operations in the MC68881 with respect to the MC68020 whenever possible. In order to simplify the programmer's model, the coprocessor interface is designed to emulate, as closely as possible, non-concurrent operation between the MC68020 and the MC68881.

The MC68881 is a non-DMA type coprocessor which uses a subset of the general purpose coprocessor interface supported by the MC68020. Features of the interface implemented in the MC68881 are as follows:

- The main processor(s) and MC68881 communicate via standard M68000 bus cycles.
- The main processor(s) and MC68881 communications are not dependent upon the instruction sets or internal details of the individual devices (e.g., instruction pipes or caches, addressing modes).
- The main processor(s) and MC68881 may operate at different clock speeds.
- MC68881 instructions utilize all addressing modes provided by the main processor; all effective addresses are calculated by the main processor at the request of the coprocessor.
- All data transfers are performed by the main processor at the request of the MC68881; thus memory management, bus errors, address errors, and bus arbitration function as if the MC68881 instructions are executed by the main processor.
- Overlapped (concurrent) instruction execution enhances throughput while maintaining the programmer's model of sequential instruction execution.
- Coprocessor detection of exceptions which require a trap to be taken are serviced by the main processor at the request of the MC68881; thus exception processing functions as if the MC68881 instructions were executed by the main processor.
- Support of virtual memory/virtual machine systems is provided via the FSAVE and FRESTORE instructions.
- Up to eight coprocessors may reside in a system simultaneously; multiple coprocessors of the same type are also allowed.
- Systems may use software emulation of the MC68881 without reassembling or relinking user software.

HARDWARE OVERVIEW

The MC68881 is a high performance floating-point device designed to interface with the MC68020 as a coprocessor. This device fully supports the MC68020 virtual machine architecture, and is implemented in HCMOS, Motorola's low power, small geometry process. This process allows CMOS and HMOS (high-density NMOS) gates to be combined on the same device. CMOS structures are used where speed and low power is required, and HMOS structures are used where minimum silicon area is desired. The HCMOS technology enables the MC68881 to be very fast while consuming less power than comparable HMOS, and still have a reasonably small die size.

With some performance degradation, the MC68881 can also be used as a peripheral processor in systems where the MC68020 is not the main processor (e.g., MC68000, MC68008, MC68010). The configuration of the MC68881 as a peripheral processor or coprocessor may be completely transparent to user software (i.e., the same object code may be executed in either configuration).

The architecture of the MC68881 appears to the user as a logical extension of the M68000 Family architecture. Coupling of the coprocessor interface, allows the MC68020 programmer to view the MC68881 registers as though the registers are resident in the MC68020. Thus, a MC68020/MC68881 device pair appears to be one processor that supports seven floating-point and integer data types, and has eight integer data registers, eight address registers, and eight floating-point data registers.

The MC68881 programming model is shown in Figures 1 through 6, and consists of the following:

- Eight 80-bit floating-point data registers (FP0-FP7). These registers are analogous to the integer data registers (D0-D7) and are completely general purpose (i.e., any instruction may use any register).
- A 32-bit control register that contains enable bits for each class of exception trap, and mode bits to set the user-selectable rounding and precision modes.
- A 32-bit status register that contains floating-point condition codes, quotient bits, and exception status information.
- A 32-bit instruction address register that contains the main processor memory address of the last floating-point instruction that was executed. This address is used in exception handling to locate the instruction that caused the exception.

The connection between the MC68020 and the MC68881 is a simple extension of the M68000 bus interface. The MC68881 is connected as a coprocoessor to the MC68020, and the selection of the MC68881 is based upon a chip select ($\overline{CS}$), which is decoded from the MC68020 function codes and address bus. Figure 7 illustrates the MC68881/MC68020 configuration.

As shown in Figure 8, the MC68881 is internally divided into three processing elements; the bus interface unit (BIU), the execution control unit (ECU), and the microcode control unit (MCU). The BIU communicates with the MC68020, and the ECU and MCU execute all MC68881 instructions.

*All references to the MC68020, throughout this technical summary, also apply to the MC68030.

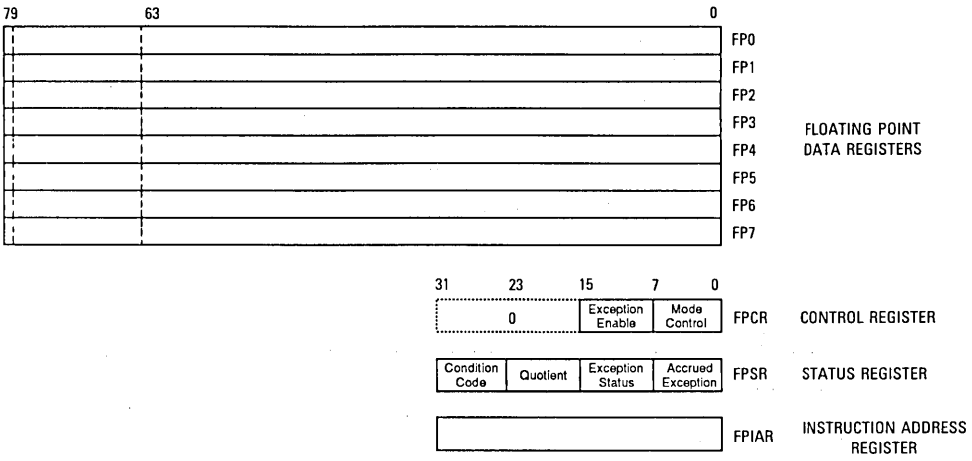


Figure 1. MC68881 Programming Model

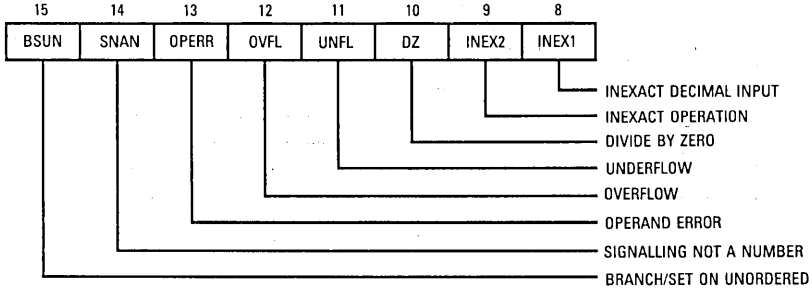


Figure 2. Exception Status/Enable Byte

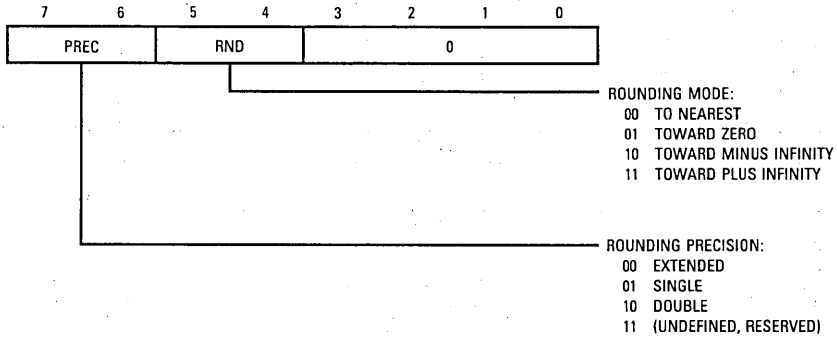


Figure 3. Mode Control Byte

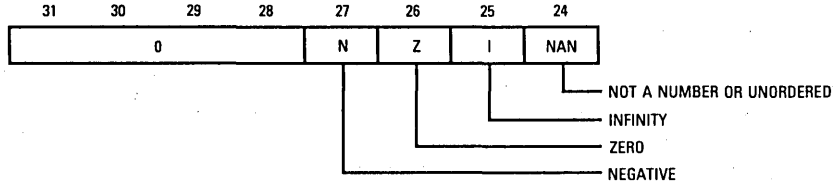


Figure 4. Condition Code Byte

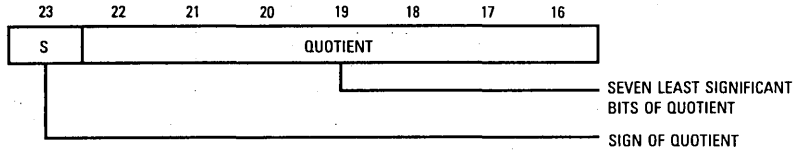


Figure 5. Quotient Byte

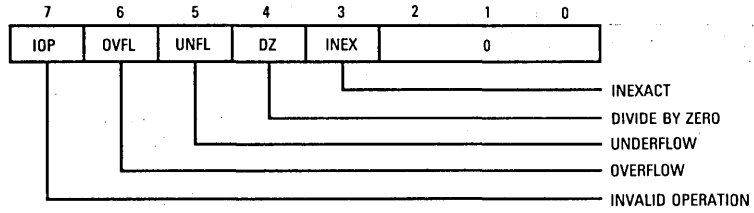


Figure 6. Accrued Exception Byte

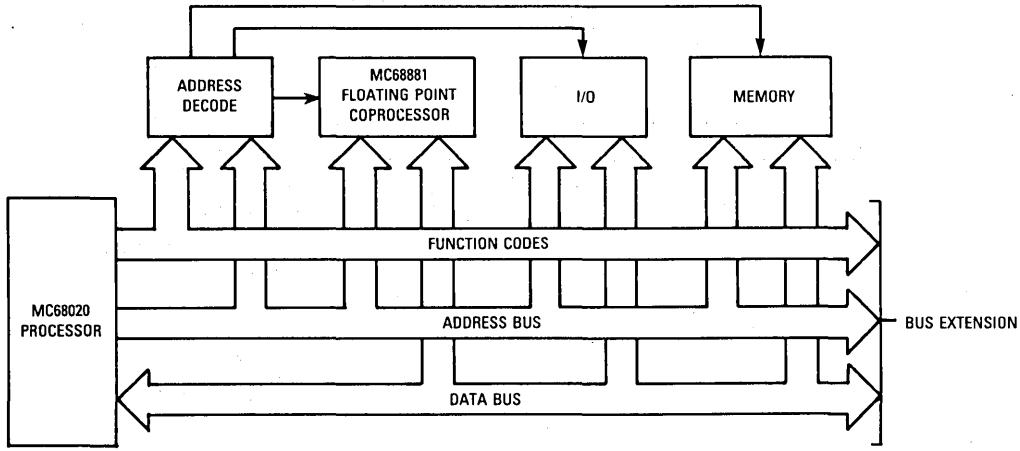


Figure 7. Typical Coprocessor Configuration

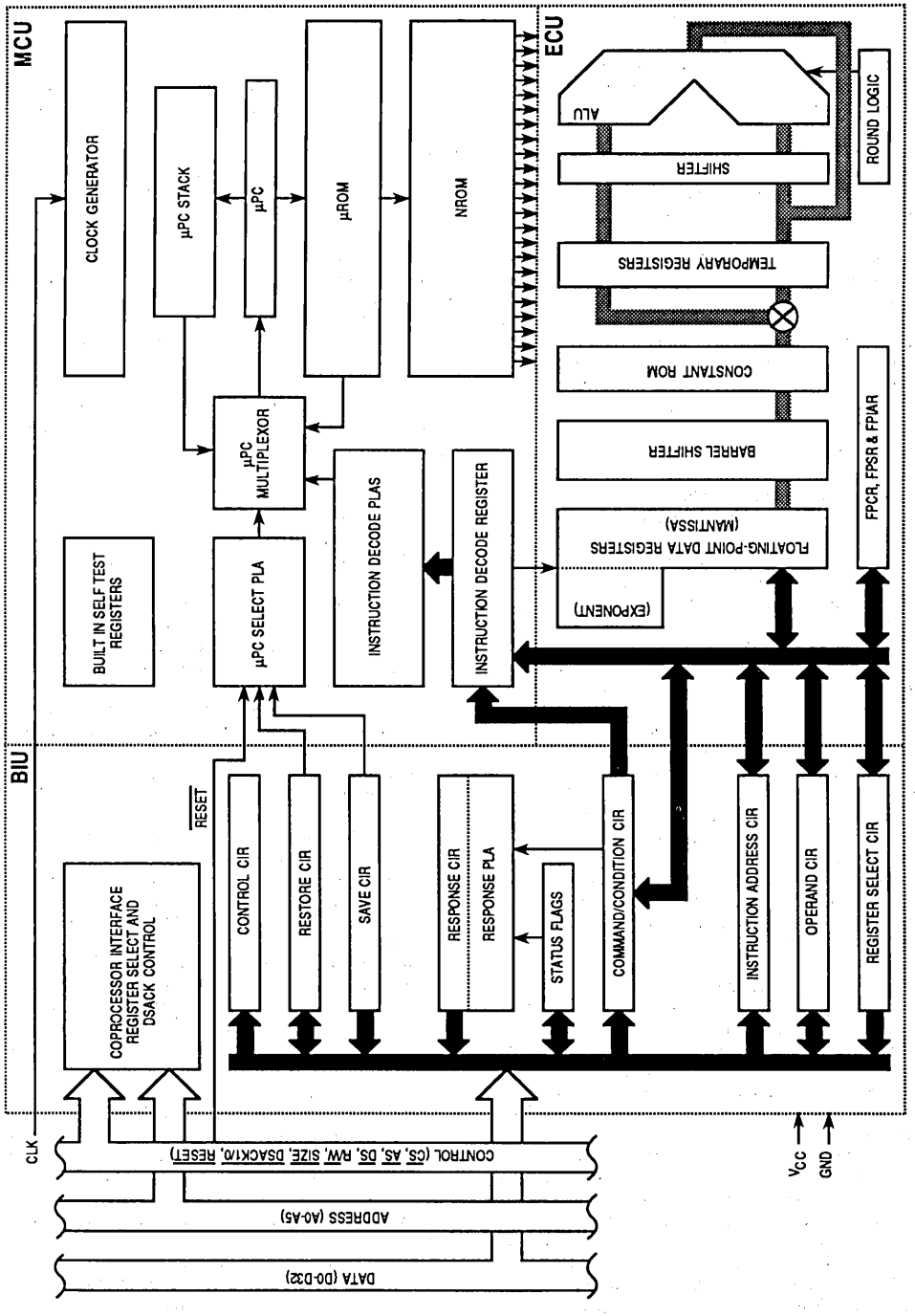


Figure 8. MC68881 Simplified Block Diagram

4

The BIU contains the coprocessor interface registers, and the 32-bit control, status, and instruction address registers. In addition to these registers, the register select and DSACK timing control logic is contained in the BIU. Finally, the status flags used to monitor the status of communications with the main processor are contained in the BIU.

The eight 80-bit floating-point data registers (FP0-FP7) are located in the ECU. In addition to these registers, the ECU contains a high-speed 67-bit arithmetic unit used for both mantissa and exponent calculations, a barrel shifter that can shift from 1 bit to 67 bits in one machine cycle, and ROM constants (for use by the internal algorithms or user programs).

The MCU contains the clock generator, a two-level microcoded sequencer that controls the ECU, the microcode ROM, and self-test circuitry. The built-in self-test capabilities of the MC68881 enhance reliability and ease manufacturing requirements; however, these diagnostic functions are not available to the user.

BUS INTERFACE UNIT

All communications between the MC68020 and the MC68881 occur via standard M68000 Family bus transfers. The MC68881 is designed to operate on 8-, 16-, or 32-bit data buses.

The MC68881 contains a number of coprocessor interface registers (CIRs) which are addressed in the same manner as memory by the main processor. The M68000 Family coprocessor interface is implemented via a protocol of reading and writing to these registers by the main processor. The MC68020 implements this general purpose coprocessor interface protocol in hardware and microcode.

When the MC68020 detects a typical MC68881 instruction, the MC68020 writes the instruction to the memory-mapped command CIR, and reads the response CIR. In this response, the BIU encodes requests for any additional action required of the MC68020 on behalf of the MC68881. For example, the response may request that the MC68020 fetch an operand from the evaluated effective address and transfer the operand to the operand CIR. Once the MC68020 fulfills the coprocessor request(s), the MC68020 is free to fetch and execute subsequent insructions.

A key concern in a coprocessor interface that allows concurrent instrucion execution is synchronization during main processor and coprocessor communication. If a subsequent instruction is written to the MC68881 before the ECU has completed execution of the previous instruction, the response instructs the MC68020 to wait. Thus, the choice of concurrent or nonconcurrent instruction execution is determined on an instruction-by-instruction basis by the coprocessor.

The only difference between a coprocessor bus transfer and any other bus transfer is that the MC68020 issues a function code to indicate the CPU address space during the cycle (the function codes are generated by the M68000 Family processors to identify eight separate address spaces). Thus, the memory-mapped coprocessor interface registers do not infringe upon instruction or data address spaces. The MC68020 places a coprocessor ID field from the coprocessor instruction onto three of the upper address lines during coprocessor accesses. This ID, along with the CPU address space function code, is decoded to select one of eight coprocessors in the system.

Since the coprocessor interface protocol is based solely on bus transfers, the protocol is easily emulated by software when the MC68881 is used as a peripheral with any processor capable of memory-mapped I/O over an M68000 style bus. When used as a peripheral processor with the 8-bit MC68008 or the 16-bit MC68000, or MC68010, all MC68881 instructions are trapped by the main processor to an exception handler at execution time. Thus, the software emulation of the coprocessor interface protocol can be totally transparent to the user. The system can be quickly upgraded by replacing the main processor with an MC68020 without changes to the user software.

Since the bus is asynchronous, the MC68881 need not run at the same clock speed as the main processor. Total system performance may therefore be customized. For example, a system requiring very fast floating-point arithmetic with relatively slow integer arithmetic can be designed with an inexpensive main processor and a fast MC68881.

COPROCESSOR INTERFACE

The M68000 Family coprocessor interface is an integral part of the MC68881 and MC68020 design, with the interface tasks shared between the two. The interface is fully compatible with all present and future M68000 Family products. Tasks are partitioned such that the MC68020 does not have to decode coprocessor instructions, and the MC68881 does not have to duplicate main processor functions such as effective address evaluation.

This partitioning provides an orthogonal extension of the instruction set by permitting MC68881 instructions to utilize all MC68020 addressing modes and to generate execution time exception traps. Thus, from the programmer's view, the CPU and coprocessor appear to be integrated onto a single chip. While the execution of the majority of MC68881 instructions may be overlapped with the execution of MC68020 instructions, concurrency is completely transparent to the programmer. The MC68020 single-step and program flow (trace) modes are fully supported by the MC68881 and the M68000 Family coprocessor interface.

While the M68000 Family coprocessor interface permits coprocessors to be bus masters, the MC68881 is never a bus master. The MC68881 requests that the MC68020 fetch all operands and store all results. In this manner, the MC68020 32-bit data bus provides high speed transfer of floating-point operands and results while simplifying the design of the MC68881.

Since the coprocessor interface is based solely upon bus cycles and the MC68881 is never a bus master, the MC68881 can be placed on either the logical or physical side of the system memory management unit. This provides a great deal of flexibility in the system design.

The virtual machine architecture of the MC68020 is supported by the coprocessor interface and the MC68881 through the FSAVE and FRESTORE instructions. If the MC68020 detects a page fault and/or task time out, the MC68020 can force the MC68881 to stop whatever operation is in process at any time (even in the middle of

the execution of an instruction) and save the MC68881 internal state in memory.

The size of the saved internal state of the MC68881 is dependent upon what the ECU is doing at the time that the FSAVE is executed. If the MC68881 is in the reset state when the FSAVE instruction is received, only one word of state is transferred to memory, which may be examined by the operating system to determine that the coprocessor programmer's model is empty. If the coprocessor is idle when the save instruction is received, only a few words of internal state are transferred to memory. If the MC68881 is in the middle of executing an instruction, it may be necessary to save the entire internal state of the machine. Instructions that can complete execution in less time than it would take to save the larger state in mid-instruction are allowed to complete execution and then save the idle state. Thus the size of the saved internal state is kept to a minimum. The ability to utilize several internal state sizes greatly reduces the average context switching time.

The FRESTORE instruction permits reloading of an internal state that was saved earlier, and continues any operation that was previously suspended. Restoring of the reset internal state functions just like a hardware reset to the MC68881 in that defaults are re-established.

OPERAND DATA FORMATS

The MC68881 supports the following data formats:

Byte Integer (B)
Word Integer (W)
Long Word Integer(L)
Single Precision Real (S)
Double Precision Real(D)
Extended Precision Real (X)
Packed Decimal String Real(P)

The capital letters contained in parenthesis denote suffixes added to instructions in the assembly language source to specify the data format to be used.

INTEGER DATA FORMATS

The three integer data formats (byte, word, and long word) are the standard data formats supported in the M68000 Family architecture. Whenever an integer is used in a floating-point operation, the integer is automatically converted by the MC68881 to an extended precision floating-point operation, the integer is automatically converted by the MC68881 to an extended precision floating-point number before being used. For example, to add an integer constant of five to the number contained in floating-point data register 3 (FP3), the following instruction can be used:

FADD.W #5,FP3

(The Motorola assembler syntax "#" is used to is used to denote immediate addressing.)

The ability to effectively use integers in floating-point operaions saves user memory since an integer representation of a number, if respresentable, is usually smaller than the equivalent floating-point representation.

FLOATING-POINT DATA FORAMTS

The floating-point data formats single precision (32-bits) and double precision (64-bits) are as defined by the IEEE standard. These are the main floating-point formats and should be used for most calculations involving real numbers. Table 1 lists the exponent and mantissa size for single, double, and extended precision. The exponent is biased, and the mantissa is in sign and magnitude form. Since single and double precision require normalized numbers, the most significant bit of the mantissa is implied as one and is not included, thus giving one extra bit of precision.

Table 1. Exponent and Mantissa Sizes

Data Format	Exponent Bits	Mantissa Bits	Bias
Single	8	23(+1)	127
Double	11	52(+1)	1023
Extended	15	64	16383

The extended precision data format is also in conformance with the IEEE standard, but the standard does not specify this format to the bit level as it does for single and double precision. The memory format on the MC68881 consists of 96 bits (three long words). Only 80 bits are actually used, the other 16 bits are for future expandability and for long-word alignment of floating-point data structures. Extended format has a 15-bit exponent, a 64-bit mantissa, and a 1-bit mantissa sign.

Extended precision numbers are intended for use as temporary variables, intermediate values, or in places where extra precision is needed. For example, a compiler might select extended precision arithmetic for evaluation of the right side of an equation with mixed sized data and then convert the answer to the data type on the left side of the equation. It is anticipated that extended precision data will not be stored in large arrays, due to the amount of memory required by each number.

PACKED DECIMAL STRING REAL DATA FORMAT

The packed decimal data format allows packed BCD strings to be input to and output from the MC68881. The strings consist of a 3-digit base 10 exponent and a 17-digit base 10 mantissa. Both the exponent and mantissa have a separate sign bit. All digits are packed BCD, such that an entire string fits in 96 bits (three long words). As is the case with all data formats, when packed BCD strings are input to the MC68881, the strings are automatically converted to extended precision real values. This allows packed BCD numbers to be used as inputs to any operation. For example:

FADD.P #-6.023E+24,FP5

BCD numbers can be output from the MC68881 in a format readily used for printing by a program generated by a high-level language compiler. For example:

FMOVE.P FP3,BUFFER{#-5}

instructs the MC68881 to convert the floating-point data register 3 (FP3) contents into a packed BCD string with five digits to the right of the decimal point (FORTRAN F format).

DATA FORMAT SUMMARY

All data formats described above are supported orthogonally by all arithmetic and transcendental operations, and by all appropriate MC68020 addressing modes. For example, all of the following are legal instructions:

```
FADD.B    #3,FP0
FADD.W    D2,FP3
FADD.L    BIGINT,FP7
FADD.S    #3.14159,FP5
FADD.D    (SP)+,FP6
FADD.X    [(TEMP__PTR,A7)],FP3
FADD.P    #1.23E25,FP0
```

On-chip calculations are performed to extended precision format, and the eight floating point data registers always contain exetended precision values. All data used in an operation is converted to extended precision by the MC68881 before the specific operation is performed, and all results are in extended precision. This ensures maximum accuracy without sacrificing performance.

Refer to Figure 9 for a summary of the memory formats for the seven data formats supported by the MC68881.

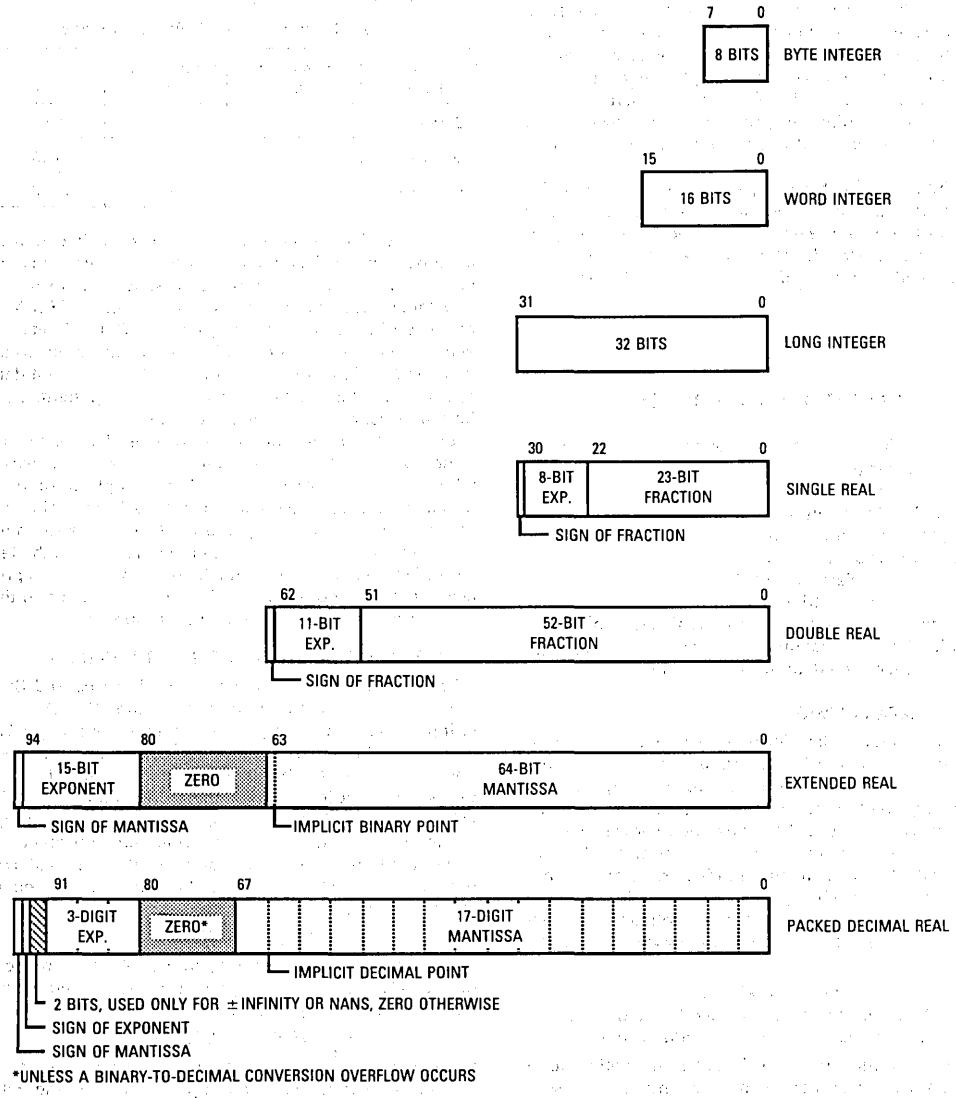


Figure 9. MC68881 Data Format Summary

INSTRUCTION SET

The MC68881 instruction set is organized into six major classes:

1. Moves between the MC68881 and memory or the MC68020 (in and out),
2. Move multiple registers (in and out),
3. Monadic operations,
4. Dyadic operations,
5. Branch, set, or trap conditionally, and
6. Miscellaneous.

MOVES

All moves from memory (or from an MC68020 data register) to the MC68881, cause data conversion from the source data format to the internal extended precision format.

All moves from the MC68881 to memory (or to an MC68020 data register), cause data conversion from the internal extended precison format to the destination data format.

Note that data movement instructions perform arithmeic operations, since the result is always rounded to the precision selected in the FPCR mode contol byte. The result is rounded using the selected rounding mode, and is checked for overflow and underflow.

The syntax for the move is:

FMOVE.⟨fmt⟩	⟨ea⟩,FPn	Move to MC68881
FMOVE.⟨fmt⟩	FPm,⟨ea⟩	Move from MC68881
FMOVE.X	FPm,FPn	Move within MC68881

where:

⟨ea⟩ is an MC68020 effective address operand and ⟨fmt⟩ is the data format size. FPm and FPn are floating-point data registers.

MOVE MULTIPLES

The floating-point move multiple instructions on the MC68881 are much like the integer counterparts on the M68000 Family processors. Any set of the floating-point registers FP0 through FP7 can be moved to or from memory with one instruction. These registers are always moved as 96-bit extended data with no conversion (hence no possibility of conversion errors). Some move multiple examples are as follows:

FMOVEM	⟨ea⟩,FP0-FP3/FP7
FMOVEM	FP2/FP4/FP6,⟨ea⟩

Move multiples are useful during context switches and interrupts to save or restore the state of a program. These moves are also useful at the start and end of a procedure to save and restore the calling routine's register set. In order to reduce procedure call overhead, the list of registers to be saved or restored can be contained in a data register. This allows run-time optimization by allowing a called routine to save as few registers as possible. Note that no rounding or overflow/underflow checking is performed by these operations.

MONADIC OPERATIONS

Monadic operations have one operand. This operand may be in a floating-point data register, memory, or in an MC68020 data register. The result is always stored in a floating-point data register. For example, the syntax for square root is:

FSQRT.⟨fmt⟩	⟨ea⟩,FPn or,
FSQRT.X	FPm,FPn or,
FSQRT.X	FPn

The MC68881 monadic operations available are as follows:

FABS	Absolute Value
FACOS	Arc Cosine
FASIN	Arc Sine
FATAN	Arc Tangent
FATANH	Hyperbolic Arc Tangent
FCOS	Cosine
FCOSH	Hyperbolic Cosine
FETOX	e to the x Power
FETOXM1	e to the x Power −1
FGETEXP	Get Exponent
FGETMAN	Get Mantissa
FINT	Integer Part
FINTRZ	Integer Part (Truncated)
FLOG10	Log Base 10
FLOG2	Log Base 2
FLOGN	Log Base e
FLOGNP1	Log Base e of(x+1)
FNEG	Negate
FSIN	Sine
FSINCOS	Simultaneous Sine and Cosine
FSINH	Hyperbolic Sine
FSQRT	Square Root
FTAN	Tangent
FTANH	Hyperbolic Tangent
FTENTOX	10 to the x Power
FTST	Test
FTWOTOX	2 to the x Power

DYADIC OPERATIONS

Dyadic operations have two input operands. The first input operand comes from a floating-point data register, memory, or an MC68020 data register. The second input operand comes from a floating-point data register. The destination is the same floating-point data register used for the second input. For example, the syntax for add is:

FADD.⟨fmt⟩	⟨ea⟩,FPn or,
FADD.X	FPm,FPn

The MC68881 dyadic operations available are as follows:

FADD	Add
FCMP	Compare
FDIV	Divide
FMOD	Modulo Remainder
FMUL	Multiply
FREM	IEEE Remainder
FSCALE	Scale Exponent
FSGLDIV	Single Precision Divide
FSGLMUL	Single Precision Multiply
FSUB	Subtract

BRANCH, SET, AND TRAP-ON CONDITION

The floating-point branch, set, and trap-on condition instructions implemented by the MC68881 are similar to

the equivalent integer instructions of the M68000 Family processors, except that more conditions exist due to the special values in IEEE floating-point arithmetic. When a conditional instruction is executed, the MC68881 performs the necessary condition checking and tells the MC68020 whether the condition is true or false; the MC68020 then takes the appropriate action. Since the MC68881 and MC68020 are closely coupled, the floating-point branch operations executed by the pair are very fast.

The MC68881 conditional operations are:

FBcc	Branch
FDBcc	Decrement and Branch
FScc	Set Byte According to Condition
FTRAPcc	Trap-on Condition (with an Optional Parameter)

where:

cc is one of the 32 floating-point conditional test specifiers as shown in Table 2.

4

Table 2. Floating-Point Conditional Test Specifiers

Mnemonic	Definition
	NOTE The following conditional tests do not set the BSUN bit in the status register exception byte under any circumstances.
F	False
EQ	Equal
OGT	Ordered Greater Than
OGE	Ordered Greater Than or Equal
OLT	Ordered Less Than
OLE	Ordered Less Than or Equal
OGL	Ordered Greater or Less Than
OR	Ordered
UN	Unordered
UEQ	Unordered or Equal
UGT	Unordered or Greater Than
UGE	Unordered or Greater or Equal
ULT	Unordered or Less Than
ULE	Unordered or Less or Equal
NE	Not Equal
T	True
	NOTE The following conditional tests set the BSUN bit in the status register exception byte if the NAN condition code bit is set when a conditional instruction is executed.
SF	Signaling False
SEQ	Signaling Equal
GT	Greater Than
GE	Greater Than or Equal
LT	Less Than
LE	Less Than or Equal
GL	Greater or Less Than
GLE	Greater Less or Equal
NGLE	Not (Greater, Less or Equal)
NGL	Not (Greater or Less)
NLE	Not (Less or Equal)
NLT	Not (Less Than)
NGE	Not (Greater or Equal)
NGT	Not (Greater Than)
SNE	Signaling Not Equal
ST	Signaling True

MISCELLANEOUS INSTRUCTIONS

Miscellaneous instructions include moves to and from the status, control, and instruction address registers. Also included are the virtual memory/machine FSAVE and FRESTORE instructions that save and restore the internal state of the MC68881.

FMOVE	⟨ea⟩,FPcr	Move to Control Register(s)
FMOVE	FPcr,⟨ea⟩	Move from Control Register(s)
FSAVE	⟨ea⟩	Virtual Machine State Save
FRESTORE	⟨ea⟩	Virtual Machine State Restore

ADDRESSING MODES

The MC68881 does not perform address calculations. This satifies the criterion that an M68000 Family coprocessor must not depend on certain features or capabilities that may or may not be implemented by a given main processor. Thus, when the MC68881 instructs the MC68020 to transfer an operand via the coprocessor interface, the MC68020 performs the addessing mode calculations requested in the instruction. In this case, the instruction is encoded specifically for the MC68020, and the execution of the MC68881 is not dependent on that encoding, but only on the value of the command word written to the MC68881 by the main processor.

This interface is quite flexible and allows any addressing mode to be used with floating-point instructions. For the M68000 Family, these addressing modes include immediate, postincrement, predecrement, data or address register direct, and the indexed/indirect addressing modes of the MC68020. Some addressing modes are restricted for some instructions in keeping with the M68000 Family architectural definitions (e.g. PC relative addressing is not allowed for a destination operand).

The orthogonal instruction set of the MC68881, along with the flexible branches and addressing modes, allows a programmer writing assembly language code, or a compiler writer generating object or source code for the MC68020/MC68881 device pair, to think of the MC68881 as though the MC68881 is part of the MC68020. There are no special restrictions imposed by the coprocessor interface, and floating-point arithmetic is coded exactly like integer arithmetic.

TIMING TABLES FOR TYPICAL EXECUTION

This set of tables allows a quick determination of the typical execution time for any MC68881 instruction when the MC68020 is used as the main processor. The first table presented is for effective address caculations performed by the MC68020. Entries from this table are added to entries in the other tables, if necessary, to obtain the total number of clock cycles for an operation. The assumptions for the following tables are:

- The main processor is an MC68020 and operates on the same clock as the MC68881. Instruction prefetches do not hit in the MC68020 cache (or it is disabled) and the instruction is aligned such that a prefetch occurs before the command CIR is written by the MC68020.

- A 32-bit memory interface is used, and memory accesses occur with zero wait states. All memory operands, as well as the stack pointers, are long-word aligned.
- Accesses to the MC68881 require 3 clock cycles, with the exception of read accesses to the response and save CIRs, which require 5 clock cycles.
- No instruction overlap is utilized, so the coprocessor interface overhead is 11 clocks. This can be reduced to 2 clock cycles if optimized code sequences are used, or may be 11 clock cycles if overlap is attempted and a synchronization delay is required.
- Typical operand conversion and calculation times are used (i.e. input operands are assumed to be normalized numbers in the legal range for a given fuction).
- No exceptions are enabled or occur, and the default rounding mode and precision of round-to-nearest, extended precision, is used.

EFFECTIVE ADDRESS CALCULATIONS

For any instruction that requires an operand external to the MC68881, an evaluate effective address and transfer data response primitive is issued by the MC68881 during the dialog for that instruction. The amount of time that is required for the MC68020 to calculate the effective address while processing this primiive for each addressing mode, excluding the transfer of the data to the MC68881, is shown in Table 3.

ARITHMETIC OPERATIONS

The Table 4 gives the typical instruction execution time for each arithmetic instruction. This group of instructions includes the majority of the MC68881 operations such as FADD, FSUB, etc. In addition to the instructions that perform arithmetic calculations as part of their function, the FCMP, FMOVE and FTST instructions are also included,

4

Table 3. Effective Address Calculations Execution Timing

Addressing Mode	Best Case	Cache Case	Worst Case
Dn or An	0	0	0
(An)	0	2	2
(An)+	3	6	6
−(An)	3	6	6
(d_{16},An) or (d_{16},PC)	0	2	3
(xxx.W)	0	2	3
(xxx).L	1	4	5
#⟨data⟩	0	0	0
(d_8,An,Xn) or (d_8,PC,Xn)	1	4	5
(d_{16},An,Xn) or (d_{16},PC,Xn)	3	6	7
(B)	3	6	7
(d_{16},B)	5	8	9
(d_{32},B)	11	14	16
([B],I)	8	11	12
([B],I,d_{16})	8	11	12
([B],I,d_{32})	10	13	15
([d_{16},B],I)	10	13	14
([d_{16},B],I,d_{16})	10	13	15
([d_{16},B],I,d_{32})	12	15	17
([d_{32},B],I)	16	19	21
([d_{32},B],I,d_{16})	16	19	21
([d_{32},B],I,d_{32})	18	21	24

B = Base address; 0, An, PC, Xn, An + Xn, PC + Xn. Form does not affect timing.
I = Index; 0 or Xn.
Note that Xn cannot be in B and I at the same time. Scaling and size of Sn does not affect timing.

Table 4. Arithmetic Operations Execution Timing

Instruction	FPm Source	Memory Source or Destination Operand Format*				
		Integer**	Single**	Double	Extended	Packed
FABS	35	62	54	60	58	872
FACOS	652	625	644	650	648	1462
FADD	51	80	72	78	76	888
FASIN	581	608	600	606	604	1418
FATAN	403	430	422	428	426	1240
FATANH	693	720	712	718	716	1530
FCMP	33	62	54	60	58	870
FCOS	391	418	410	416	414	1228
FCOSH	607	634	626	632	630	1444
FDIV	103	132	124	130	128	940
FETOX	497	524	516	522	520	1334
FETOXM1	545	572	564	570	568	1382
FGETEXP	45	72	64	70	68	882
FGETMAN	31	58	50	56	54	868
FINT	55	82	74	80	78	892
FINTRZ	55	82	78	80	74	892
FLOGN	525	552	544	550	548	1362
FLOGNP1	571	598	590	596	594	1408
FLOG10	581	608	600	606	604	1418
FLOG2	581	608	600	606	604	1418
FMOD	67	94	86	92	90	902
FMOVE to FPn	33	60	52	58	56	870
FMOVE to Memory***	—	100	80	86	72	1996
FMOVECR****	29	—	—	—	—	—
FMUL	71	100	92	98	96	908
FNEG	35	62	54	60	58	872
FREM	67	94	86	92	90	902
FSCALE	41	70	62	68	66	878
FSGLDIV	69	98	90	96	94	906
FSGLMUL	59	88	80	86	84	896
FSIN	391	418	410	416	414	1228
FSINCOS	451	478	470	476	474	1288
FSINH	687	714	706	712	710	1524
FSQRT	107	134	126	132	130	944
FSUB	51	80	72	78	76	888
FTAN	473	500	492	498	496	1310
FTANH	661	688	680	686	684	1498
FTENTOX	567	594	586	592	590	1404
FTST	33	60	52	58	56	870
FTWOTOX	567	594	586	592	590	1404

*Add the appropriate effective address calculation time.
**If the source or destination is an MC68020 data register, subtract 5 or 2 clock cycles, respectively.
***Assume a static K-factor is used if the destination data format is packed decimal. Add 14 clock cycles if a dynamic K-factor is used.
****The source operand is from the constant ROM rather than a floating-point data register.

since an implicit conversion is performed by those operations. For memory operands, the timing for the appropriate effective addressing mode must be added to the numbers in this table to determine the overall instruction execution times.

MOVE CONTROL REGISTER AND MOVE MULTIPLE OPERATIONS

The Table 5 gives the execution times for the FMOVE FPcr and FMOVEM instructions. The timing for the appropriate effective addressing mode must be added to the numbers in this table to determine the overall instruction execution times.

CONDITIONAL OPERATIONS

The Table 6 gives the execution times for the MC68881 conditional instructions. Each entry in this table, except those for the FScc instruction, is complete and does not require the addition of values from any other table. For the FScc instruction, the only additional factor that must be included is the calculate effective address time for the operand to be modified.

FSAVE AND FRESTORE INSTRUCTIONS

The time requred for a context save or restore operation is given in Table 7. The appropriate calculate effective address times must be added to the values in this table to obtain the total execution time for these operations. For the FSAVE instruction, the MC68881 may use the not ready format code to force the MC68020 to wait while internal operations are completed in order to reduce the size of the saved state frame or reach a point where a save operation can be performed. The idle (minimum) time occurs if the MC68881 is in the idle phase when the save CIR is written. A time between the idle (minimum) and the idle (maximum) occurs if an instruction is in the end phase when the save CIR is read. The busy (minimum) time occurs if the MC68881 is in the initial phase, or at a save boundary in the middle phase, when the save CIR is read. Finally, the busy (maximum) time occurs if the MC68881 has just passed a save boundary in the middle phase when the save CIR is read.

Table 5. Move Control Register and FMOVEM Operations Execution Timing

Operation*		Best Case	Cache Case	Worst Case
FMOVE	FPcr,Rn	29	31	34
	FPcr,⟨ea⟩	31	33	36
	Rn,FPcr	26	28	31
	⟨ea⟩,FPcr	31	33	36
	#⟨data⟩,FPcr	30	30	31
FMOVEM	FPcr__list,⟨ea⟩	25+6n	27+6n	30+6n
	⟨ea⟩,FPcr__list	25+6n	27+6n	30+6n
	#⟨data⟩,FPcr__list	24+6n	25+6n	29+6n
FMOVEM	FPdr__list,⟨ea⟩	35+25n	37+25n	40+25n
	⟨ea⟩,FPdr__list	33+23n	35+23n	38+23n
	Dn,⟨ea⟩	49+25n	51+25n	54+25n
	⟨ea⟩,Dn	47+23n	49+23n	52+23n

*Add the appropriate effective address calculation time.
n is the number of registers transferred.

Table 6. Conditional Instructions Execution Timing

Operation	Comments	Best Case	Cache Case	Worst Case
FBcc.W	Branch Taken	18	20	23
	Branch Not Taken	16	18	19
FBcc.L	Branch Taken	18	20	23
	Branch Not Taken	16	18	21
FDBcc	True, Not Taken	18	20	24
	False, Not Taken	22	24	32
	False, Taken	18	20	26
FNOP	No Operation	16	18	19
FScc	Dn	16	18	21
	(An)+ or −(An)*	18	22	25
	Memory**	16	20	23
FTRAPcc	Trap Taken	36	39	47
	Trap Not Taken	16	18	22
FTRAPcc.W	Trap Taken	38	41	45
	Trap Not Taken	18	20	23
FTRAPcc.L	Trap Taken	40	43	52
	Trap Not Taken	20	22	27

*For condition true; subtract one clock for condition false.
**Add the appropriate effective address calculation time.

Table 7. FSAVE and FRESTORE Instructions Execution Timing

Operation*	State Frame	Best Case	Cache Case	Worst Case
FRESTORE	Null	19	21	22
	Idle	55	57	58
	Busy	312	314	315
FSAVE	Null	14	16	18
	Idle (Minimum)	50	52	54
	Idle (Maximum)	286	218	290
	Busy (Minimum)	316	318	320
	Busy (Maximum)	552	554	556

*Add the appropriate effective address calculation time.

FUNCTIONAL SIGNAL DESCRIPTIONS

This section contains a brief description of the input and output signals for the MC68881 floating-point coprocessor. The signals are functionally organized into groups as shown in Figure 10.

NOTE

The terms **assertion** and **negation** are used extensively. This is done to avoid confusion when describing "active-low" and "active-high" signals. The term **assert** or **assertion** is used to indicate that a signal is active or true, independent of whether that level is represented by a high or low voltage. The term **negate** or **negation** is used to indicate that a signal is inactive or false.

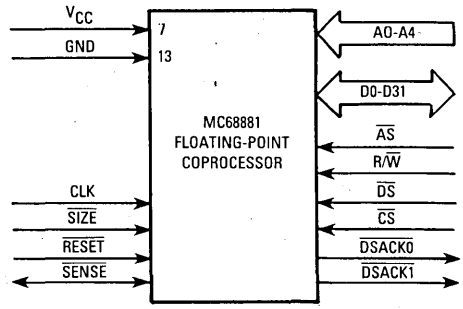


Figure 10. MC68881 Input/Output Signals

Table 8. Coprocessor Interface Register Selection

A4-A0	Offset	Width	Type	Register
0000x	$00	16	Read	Response
0001x	$02	16	Write	Control
0010x	$04	16	Read	Save
0011x	$06	16	R/$\overline{W}$	Restore
0100x	$08	16	—	(Reserved)
0101x	$0A	16	Write	Command
0110x	$0C	16	—	(Reserved)
0111x	$0E	16	Write	Condition
100xx	$10	32	R/$\overline{W}$	Operand
1010x	$14	16	Read	Register Select
1011x	$16	16	—	(Reserved)
110xx	$18	32	Read	Instruction Address
111xx	$1C	32	R/$\overline{W}$	Operand Address

ADDRESS BUS (A0 through A4)

These active-high address line inputs are used by the main processor to select the coprocessor interface register locations located in the CPU address space. These lines control the register selection as listed in Table 8.

When the MC68881 is configured to operate over an 8-bit data bus, the A0 pin is used as an address signal for byte acceses of the coprocessor interface registers. When the MC68881 is configured to operate over a 16- or 32-bit system data bus, both the A0 and $\overline{SIZE}$ pins are strapped high and/or low as listed in Table 9.

Table 9. System Data Bus Size Configuration

A0	SIZE	Data Bus
—	Low	8-Bit
Low	High	16-Bit
High	High	32-Bit

DATA BUS (D0 through D31)

This 32-bit, bidirectional, three-state bus serves as the general purpose data path between the MC68020 and the MC68881. Regardless of whether the MC68881 is operated as a coprocessor or a peripheral processor, all interprocessor transfers of instruction information, operand data, status information, and requests for service occur as standard M68000 bus cycles.

The MC68881 will operate over an 8-, 16-, or 32-bit system data bus. Depending upon the system data bus configuration, both the A0 and $\overline{SIZE}$ pins are configured specifically for the applicable bus configuration. (Refer to **ADDRESS BUS (A0 through A4)** and **SIZE ($\overline{SIZE}$)** for further details.)

SIZE ($\overline{SIZE}$)

This active-low input signal is used in conjunction with the A0 pin to configure the MC68881 for operation over an 8-, 16-, or 32-bit system data bus. When the MC68881 is configured to operate over a 16- or 32-bit system data bus, both the $\overline{SIZE}$ and A0 pins are strapped high and/or low as listed in Table 9.

ADDRESS STROBE ($\overline{AS}$)

This active-low input signal indicates that there is a valid address on the address bus, and both the chip select ($\overline{CS}$) and read/write (R/$\overline{W}$) signal lines are valid.

CHIP SELECT ($\overline{CS}$)

This active-low input signal enables the main processor access to the MC68881 coprocessor interface registers. When operating the MC68881 as a peripheral processor, the chip select decode is system dependent (i.e., like the chip select on any peripheral). The $\overline{CS}$ signal must be valid (either asserted or negated) when $\overline{AS}$ is asserted. Refer to **CHIP SELECT TIMING** for further discussion of timing restrictions for this signal.

READ/WRITE (R/$\overline{W}$)

This input signal indicates the direction of a bus transaction (read/write) by the main processor. A logic high (1) indicates a read from the MC68881, and a logic low (0) indicates a write to the MC68881. The R/$\overline{W}$ signal must be valid when $\overline{AS}$ is asserted.

DATA STROBE ($\overline{DS}$)

This active-low input signal indicates that there is valid data on the data bus during a write bus cycle.

DATA TRANSFER AND SIZE ACKNOWLEDGE (DSACK0, DSACK1)

These active-low, three-state output signals indicate the completion of a bus cycle to the main processor. The MC68881 asserts both the DSACK0 and DSACK1 signals upon assertion of CS.

If the bus cycle is a main processor read, the MC68881 asserts DSACK0 and DSACK1 signals to indicate that the information on the data bus is valid. (Both DSACK signals may be asserted in advance of the valid data being placed on the bus.) If the bus cycle is a main processor write to the MC68881, DSACK0 and DSACK1 are used to acknowledge acceptance of the data by the MC68881.

The MC68881 also uses DSACK0 and DSACK1 signals to dynamically indicate to the MC68020 the "port" size (system data bus width) on a cycle-by-cycle basis. Depending upon which of the two DSACK pins are asserted in a given bus cycle, the MC68020 assumes data has been transferred to/from an 8-, 16-, or 32-bit wide data port. Table 10 lists the DSACK assertions that are used by the MC68881 for the various bus cycles over the various bus cycles over the various system data bus configurations.

Table 13 indicates that all accesses over a 32-bit bus where A4 equals zero are to 16-bit registers. The MC68881 implements all 16-bit coprocessor interface registers on data lines D16-D31 (to eliminate the need for on-chip multiplexers); however, the MC68020 expects 16-bit registers that are located in a 32-bit port at odd word addresses (A1 = 1) to be implemented on data lines D0-D15. For accesses to these registers when configured for 32-bit bus operation, the MC68881 generates DSACK signals as listed in Table 10 to inform the MC68020 of valid data on D16-D31 instead of D0-D15.

An external holding resistor is required to maintain both DSACK0 and DSACK1 high between bus cycles. In order to reduce the signal rise time, the DSACK0 and DSACK1 lines are actively pulled up (negated) by the MC68881 following the rising edge of AS or DS, and both DSACK lines are then three-stated (placed in the high-impedance state) to avoid interference with the next bus cycle.

RESET (RESET)

This active-low input signal causes the MC68881 to initialize the floating-point data registers to non-signaling not-a-numbers (NANs) and clears the floating-point control, status, and instruction address registers.

When performing a power-up reset, external circuitry should keep the RESET line asserted for a minimum of four clock cycles after V_{CC} is within tolerance. This assures correct initialization of the MC68881 when power is applied. For compatibility with all M68000 Family devices, 100 milliseconds should be used as the minimum.

When performing a reset of the MC68881 after V_{CC} has been within tolerance for more than the initial power-up time, the RESET line must have an asserted pulse width which is greater than two clock cycles. For compatability with all M68000 Family devices, 10 clock cycles should be used as the minimum.

CLOCK (CLK)

The MC68881 clock input is a TTL-compatable signal that is internally buffered for development of the internal clock signals. The clock input should be a constant frequency square wave with no stretching or shaping techniques required. The clock should not be gated off at any time and must conform to minimum and maximum period and pulse width times.

SENSE DEVICE (SENSE)

This pin may be used optionally as an additional GND pin, or as an indicator to external hardware that the MC68881 is present in the system. This signal is internally connected to the GND of the die, but it is not necessary to connect it to the external ground for correct device operation. If a pullup resistor (which should be larger than 10 kohm) is connected to this pin location, external hardware may sense the presence of the MC68881 in a system.

POWER (V_{CC} and GND)

These pins provide the supply voltage and system reference level for the internal circuitry of the MC68881. Care should be taken to reduce the noise level on these pins with appropriate capacitive decoupling.

NO CONNECT (NC)

One pin of the MC68881 package is designated as a no connect (NC). This pin position is reserved for future use by Motorola, and should not be used for signal routing or connected to V_{CC} or GND.

SIGNAL SUMMARY

Table 11 provides a summary of all the MC68881 signals described in this section.

Table 10. DSACK Assertions

Data Bus	A4	DSACK1	DSACK0	Comments
32-Bit	1	L	L	Valid Data on D31-D0
32-Bit	0	L	H	Valid Data on D31-D16
16-Bit	x	L	H	Valid Data on D31-D16 or D15-D0
8-Bit	x	H	L	Valid Data on D31-D24, D23-D16, D15-D8, or D7-D0
All	x	H	H	Insert Wait States in Current Bus Cycle

Table 11. Signal Summary

Signal Name	Mnemonic	Input/Output	Active State	Three State
Address Bus	A0-A4	Input	High	—
Data Bus	D0-D13	Input/Output	High	Yes
Size	$\overline{SIZE}$	Input	Low	—
Address Strobe	$\overline{AS}$	Input	Low	—
Chip Select	$\overline{CS}$	Input	Low	—
Read/Write	R/$\overline{W}$	Input	High/Low	—
Data Strobe	$\overline{DS}$	Input	Low	—
Data Transfer and Size Acknowledge	$\overline{DSACK0}$, $\overline{DSACK1}$	Output	Low	Yes
Reset	$\overline{RESET}$	Input	Low	—
Clock	CLK	Input	—	—
Sense Device	$\overline{SENSE}$	Input/Output	Low	No
Power Input	V_{CC}	Input	—	—
Ground	GND	Input	—	—

4

INTERFACING METHODS

MC68881/MC68020 INTERFACING

The following paragraphs describe how to connect the MC68881 to an MC68020 for coprocessor operation via an 8-, 16-, or 32-bit data bus.

32-Bit Data Bus Coprocessor Connection

Figure 11 illustrates the coprocessor interface connection of an MC68881 to an MC68020 via a 32-bit data bus. The MC68881 is configured to operate over a 32-bit data bus when both the A0 and $\overline{SIZE}$ pins are connected to V_{CC}.

16-Bit Data Bus Coprocessor Connection

Figure 12 illustrates the coprocessor interface connection of an MC68881 to an MC68020 via a 16-bit data bus. The MC68881 is configured to operate over a 16-bit data bus when the $\overline{SIZE}$ pin is connected to V_{CC}, and the A0 pin is connected to GND. The sixteen least significant data pins (D0-D15) must be connected to the sixteen most significant data pins (D16-D31) when the MC68881 is configured to operate over a 16-bit data bus (i.e., connect D0 to D16, D1 to D17, ... and D15 to D31). The DSACK pins of the two devices are directly connected, although it is not necessary to connect the $\overline{DSACK0}$ pin since the MC68881 never asserts it in this configuration.

8-Bit Data Bus Coprocessor Connection

Figure 13 illustrates the connection of an MC68881 to an MC68020 as a coprocessor over an 8-bit data bus. The MC68881 is configured to operate over an 8-bit data bus when the $\overline{SIZE}$ pin is connected to GND. The twenty four least significant data pins (D0-D23) must be connected to the eight most significant data pins (D24-D31) when the MC68881 is configured to operate over an 8-bit data bus (i.e., connect D0 to D8, D16 and D24; D1 to D9, D17, and D25; ... and D7 to D15, D23 and D31). The DSACK pins of the two devices are directy connected, although it is not necessary to connect the $\overline{DSACK1}$ pin since the MC68881 never asserts it in this configuration.

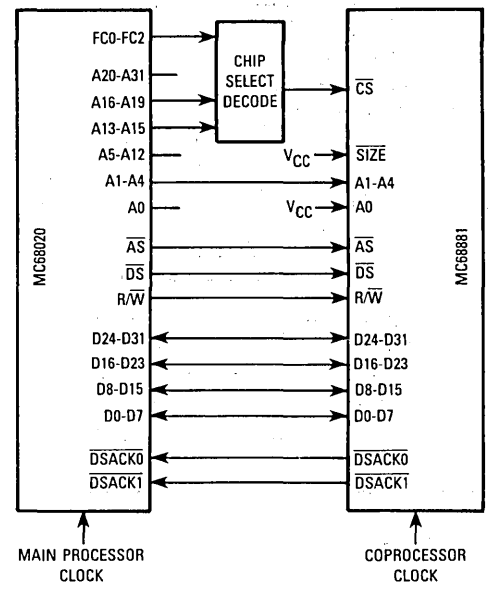


Figure 11. 32-Bit Data Bus Coprocessor Connection

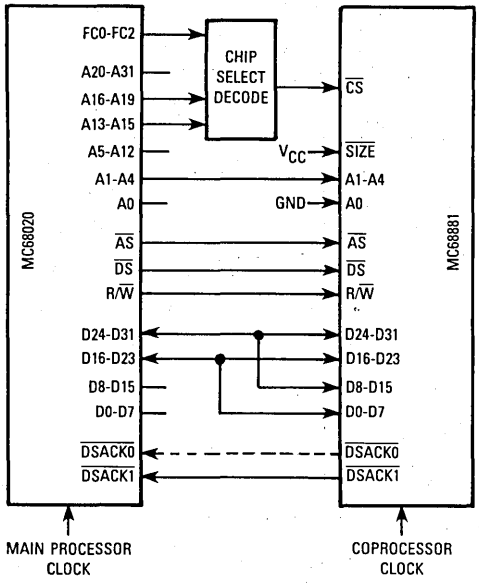


Figure 12. 16-Bit Data Bus Coprocessor Connection

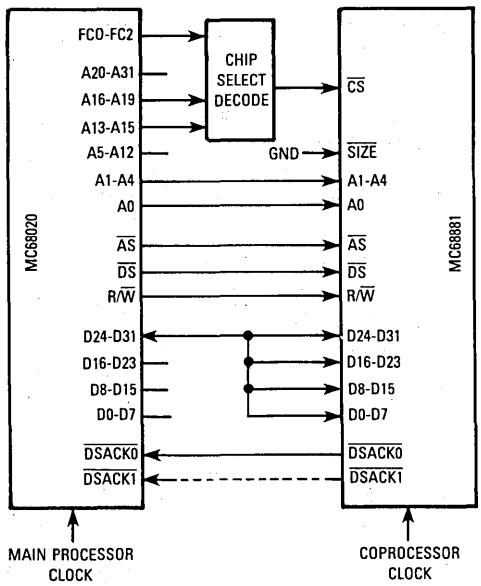


Figure 13. 8-Bit Data Bus Coprocessor Connection

MC68881-MC68000/MC68008/MC68010 INTERFACING

The following paragraphs describe how to connect the MC68881 to an MC68000, MC68008, or MC68010 processor for operation as a peripheral via an 8- or 16-bit data bus.

16-Bit Data Bus Peripheral Processor Connection

Figure 14 illustrates the connection of an MC68881 to an MC68000 or MC68010 as a peripheral processor over a 16-bit data bus. The MC68881 is configured to operate over a 16-bit data bus when the $\overline{SIZE}$ pin is connected to V_{CC}, and the A0 pin is connected to GND. The sixteen least significant data pins (D0-D15) must be connected to the sixteen most significant data pins (D16-D31) when the MC68881 is configured to operate over a 16-bit data bus (i.e., connect D0 to D16, D1 to D17, ... and D15 to D31). The $\overline{DSACK1}$ pin of the MC68881 is connected to the $\overline{DTACK}$ pin of the main processor, and the DSACK0 pin is not used.

When connected as a peripheral processor, the MC68881 chip select ($\overline{CS}$) decode is system dependent. If the MC68000 is used as the main processor, the MC68881 $\overline{CS}$ must be decoded in the supervisor or user data spaces. However, if the MC68010 is used for the main processor, the MOVES instruction may be used to emulate any CPU space access that the MC68020 generates for coprocessor communications. Thus, the $\overline{CS}$ decode logic for such systems may be the same as in an MC68020 system, such that the MC68881 will not use any part of the data address spaces.

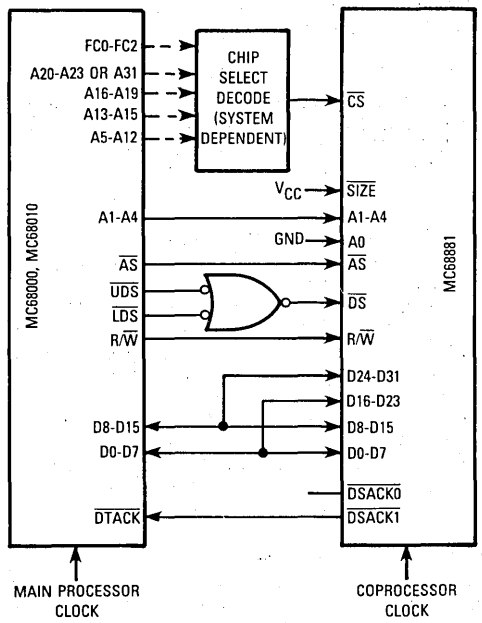


Figure 14. 16-Bit Data Bus Peripheral Processor Connection

8-Bit Data Bus Peripheral Processor Connection

Figure 15 illustrates the connection of an MC68881 to an MC68008 as a peripheral processor over an 8-bit data bus. The MC68881 is configured to operate over an 8-bit data bus when the $\overline{SIZE}$ pin is connected to GND. The eight least significant data pins (D0-D7) must be connected to the twenty four most significant data pins (D8-D31) when the MC68881 is configured to operate over an 8-bit data bus (i.e. connect D0 to D8, D16, and D24; D1 to D9, D17, and D25; ... and D7 to D15, D23, and D31). The $\overline{DSACK0}$ pin of the MC68881 is connected to the $\overline{DTACK}$ pin of the MC68008, and the $\overline{DSACK1}$ pin is not used.

When connected as a peripheral processor, the MC68881 chip select ($\overline{CS}$) decode is system dependent, and the $\overline{CS}$ must be decoded in the supervisor or user data spaces.

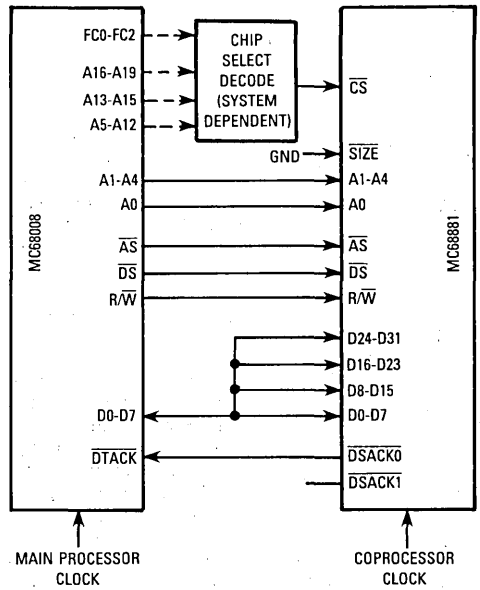


Figure 15. 8-Bit Data Bus Peripheral Processor Connection

ELECTRICAL SPECIFICATIONS

MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Supply Voltage	V_{CC}	−0.3 to +7.0	V
Input Voltage	V_{in}	−0.3 to +7.0	V
Operating Temperature	T_A	0 to 70	°C
Storage Temperature	T_{stg}	−55 to +150	°C

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

THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Rating
Thermal Resistance — Ceramic			°C/W
Junction to Ambient	θ_{JA}	33	
Junction to Case	θ_{JC}	15	

4

POWER CONSIDERATIONS

The average chip-junction temperature, T_J, in °C can be obtained from:

$$T_J = T_A + (P_D \cdot \theta_{JA}) \quad (1)$$

where:

T_A = Ambient Temperature, °C
θ_{JA} = Package Thermal Resistance, Junction-to-Ambient, °C/W
P_D = $P_{INT} + P_{I/O}$
P_{INT} = $I_{CC} \times V_{CC}$, 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.

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

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

Solving equations (1) and (2) for K gives:

$$K = P_D \cdot (T_A + 273°C) + \theta_{JA} \cdot P_D^2 \quad (3)$$

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.

The total thermal resistance of a package (θ_{JA}) can be separated into two components, θ_{JC} and θ_{CA}, representing the barrier to heat flow from the semiconductor junction to the package (case) surface (θ_{JC}) and from the case to the outside ambient (θ_{CA}). These terms are related by the equation:

$$\theta_{JA} = \theta_{JC} + \theta_{CA} \quad (4)$$

θ_{JC} is device related and cannot be influenced by the user. However, θ_{CA} is user dependent and can be minimized by such thermal management techniques as heat sinks, ambient air cooling, and thermal convention. Thus, good thermal management on the part of the user can significantly reduce θ_{CA} so that θ_{JA} approximately equals θ_{JC}. Substitution of θ_{JC} for θ_{JA} in equation (1) will result in a lower semiconductor junction temperature.

Values for thermal resistance presented in this document, unless estimated, were derived using the procedure described in Motorola Reliability Report 7843, "Thermal Resistance Measurement Method for MC68XX Microcomponent Devices," and are provided for design purposes only. Thermal measurements are complex and dependent on procedure and setup. User derived values for thermal resistance may differ.

DC ELECTRICAL CHARACTERISTICS (V_{CC} = 5.0 Vdc ± 5%; GND = 0 Vdc; T_A = 0°C to 70°C)

Characteristic		Symbol	Min	Max	Unit
Input High Voltage		V_{IH}	2.0	V_{CC}	V
Input Low Voltage		V_{IL}	GND – 0.3	0.8	V
Input Leakage Current @ 5.25 V	CLK, $\overline{RESET}$, R/$\overline{W}$, A0-A4, $\overline{CS}$, $\overline{DS}$, $\overline{AS}$, $\overline{SIZE}$	I_{in}	—	10	μA
Hi-Z (Off State) Input Current @ 2.4 V/0.4 V	$\overline{DSACK0}$, $\overline{DSACK1}$, D0-D31	I_{TSI}	—	20	μA
Output High Voltage (I_{OH} = –400 μA)	$\overline{DSACK0}$, $\overline{DSACK1}$, D0-D31	V_{OH}	2.4	—	V
Output Low Voltage (I_{OL} = 5.3 mA)	$\overline{DSACK0}$, $\overline{DSACK1}$, D0-D31	V_{OL}	—	0.5	V
Power Dissipation		P_D	—	0.75	W
Input Capacitance* (V_{in} = 0, T_A = 25°C, f = 1 MHz)		C_{in}	—	20	pF
Output Load Capacitance		C_L	—	130	pF

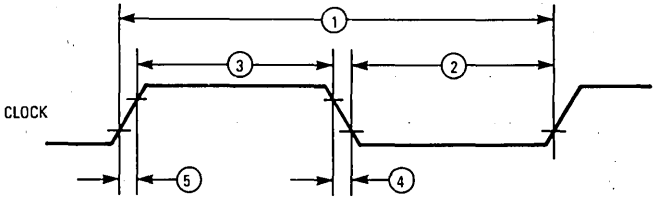
*Capacitance is periodically sampled rather than 100% tested.

4

AC ELECTRICAL CHARACTERISTICS — CLOCK INPUT

(V_{CC} = 5.0 Vdc ± 5%; GND = 0 Vdc; T_A = 0 to 70°C; refer to Figure 16)

Num	Characteristic	12 MHz		16.67 MHz		20 MHz		25 MHz		Unit
		Min	Max	Min	Max	Min	Max	Min	Max	
	Frequency of Operation	8	12.5	8	16.67	12.5	20	12.5	25	MHz
1	Cycle Time	80	125	60	125	50	80	40	80	ns
2,3	Clock Pulse Width	32	87	24	95	20	54	15	59	ns
4,5	Rise and Fall Times	—	5	—	5	—	5	—	4	ns



NOTE:
1. Timing measurements are referenced to and from a low voltage of 0.8 volts and a high voltage of 2.0 volts, unless otherwise noted. The voltage swing through this range should start outside, and pass through, the range such that the rise or fall will be linear between 0.8 volts and 2.0 volts.

Figure 16. Clock Input Timing Diagram

AC ELECTRICAL SPECIFICATIONS — READ AND WRITE CYCLES

(V_{CC} = 5.0 Vdc ± 5%; GND = 0 Vdc; T_A = 0 to 70°C; refer to Figures 17, 18, and 19)

Num	Characteristic	12 MHz		16.67 HMz		20 MHz		25 MHz		Unit
		Min	Max	Min	Max	Min	Max	Min	Max	
6	Address Valid to $\overline{AS}$ Asserted (see Note 5)	20	—	15	—	10	—	5	—	ns
6A	Address Valid to $\overline{DS}$ Asserted (Read) (see Note 5)	20	—	15	—	10	—	5	—	ns
6B	Address Valid to $\overline{DS}$ Asserted (Write) (see Note 5)	65	—	50	—	50	—	35	—	ns
7	$\overline{AS}$ Negated to Address Invalid (see Note 6)	15	—	10	—	10	—	5	—	ns
7A	$\overline{DS}$ Negated to Address Invalid (see Note 6)	15	—	10	—	10	—	5	—	ns
8	$\overline{CS}$ Negated to $\overline{AS}$ Asserted (see Note 9)	0	—	0	—	0	—	0	—	ns
8A	$\overline{CS}$ Negated to $\overline{DS}$ Asserted (Read) (see Note 9)	0	—	0	—	0	—	0	—	ns
8B	$\overline{CS}$ Asserted to $\overline{DS}$ Asserted (Write)	40	—	30	—	25	—	20	—	ns
9	$\overline{AS}$ Negated to $\overline{CS}$ Negated	10	—	10	—	10	—	5	—	ns
9A	$\overline{DS}$ Negated to $\overline{CS}$ Negated	10	—	10	—	10	—	5	—	ns
10	R/$\overline{W}$ High to $\overline{AS}$ Asserted (Read)	20	—	15	—	10	—	5	—	ns
10A	R/$\overline{W}$ High to $\overline{DS}$ Asserted (Read)	20	—	15	—	10	—	5	—	ns
10B	R/$\overline{W}$ Low to $\overline{DS}$ Asserted (Write)	45	—	35	—	30	—	25	—	ns
11	$\overline{AS}$ Negated to R/$\overline{W}$ Low (Read) or $\overline{AS}$ Negated to R/$\overline{W}$ High (Write)	15	—	10	—	10	—	5	—	ns
11A	$\overline{DS}$ Negated to R/$\overline{W}$ Low (Read) or $\overline{DS}$ Negated to R/$\overline{W}$ High (Write)	15	—	10	—	10	—	5	—	ns
12	$\overline{DS}$ Width Asserted (Write)	50	—	40	—	38	—	30	—	ns
13	$\overline{DS}$ Width Negated	50	—	40	—	38	—	30	—	ns
13A	$\overline{DS}$ Negated to $\overline{AS}$ Asserted (see Note 4)	40	—	30	—	30	—	25	—	ns
14	$\overline{CS}$, $\overline{DS}$ Asserted to Data-Out Valid) (Read) (see Note 2)	—	110	—	80	—	45	—	45	ns
15	$\overline{DS}$ Negated to Data-Out Invalid (Read)	0	—	0	—	0	—	0	—	ns
16	$\overline{DS}$ Negated to Data-Out High Impedance (Read)	—	70	—	50	—	35	—	35	ns
17	Data-In Valid to $\overline{DS}$ Asserted (Write)	20	—	15	—	10	—	5	—	ns
18	$\overline{DS}$ Negated to Data-In Invalid (Write)	20	—	15	—	10	—	5	—	ns
19	$\overline{START}$ True to $\overline{DSACK0}$ and $\overline{DSACK1}$ Asserted (see Notes 2 and 10)	—	70	—	50	—	35	—	25	ns
19A	$\overline{DSACK0}$ Asserted to $\overline{DSACK1}$ Asserted (Skew) (see Note 7)	−20	20	−15	15	−10	10	−10	10	ns
20	$\overline{DSACK0}$ or $\overline{DSACK1}$ Asserted to Data-Out Valid	—	60	—	50	—	43	—	32	ns
21	$\overline{START}$ False to $\overline{DSACK0}$ and $\overline{DSACK1}$ Negated (see Note 8)	—	70	—	50	—	40	—	40	ns
22	$\overline{START}$ False to $\overline{DSACK0}$ and $\overline{DSACK1}$ High Impedance (see Note 8)	—	90	—	70	—	55	—	55	ns
23	$\overline{START}$ True to Clock High (Synchronous Read) (see Notes 3 and 8)	0	—	0	—	0	—	0	—	ns
24	Clock Low to Data-Out Valid (Synchronous Read) (see Note 3)	—	140	—	105	—	80	—	60	ns
25	$\overline{START}$ True to Data-Out Valid (Synchronous Read) (see Notes 3 and 8)	— 1.5	140 + 2.5	— 1.5	105 + 2.5	— 1.5	80 + 2.5	— 1.5	60 + 2.5	ns Clks
26	Clock Low to $\overline{DSACK0}$ and $\overline{DSACK1}$ Asserted (Synchronous Read) (see Note 3)	—	100	—	75	—	55	—	45	ns
27	$\overline{START}$ True to $\overline{DSACK0}$ and $\overline{DSACK1}$ Asserted (Synchronous Read) (see Notes 3 and 8)	— 1.5	100 + 2.5	— 1.5	75 + 2.5	— 1.5	55 + 2.5	— 1.5	45 + 2.5	ns Clks

NOTES:

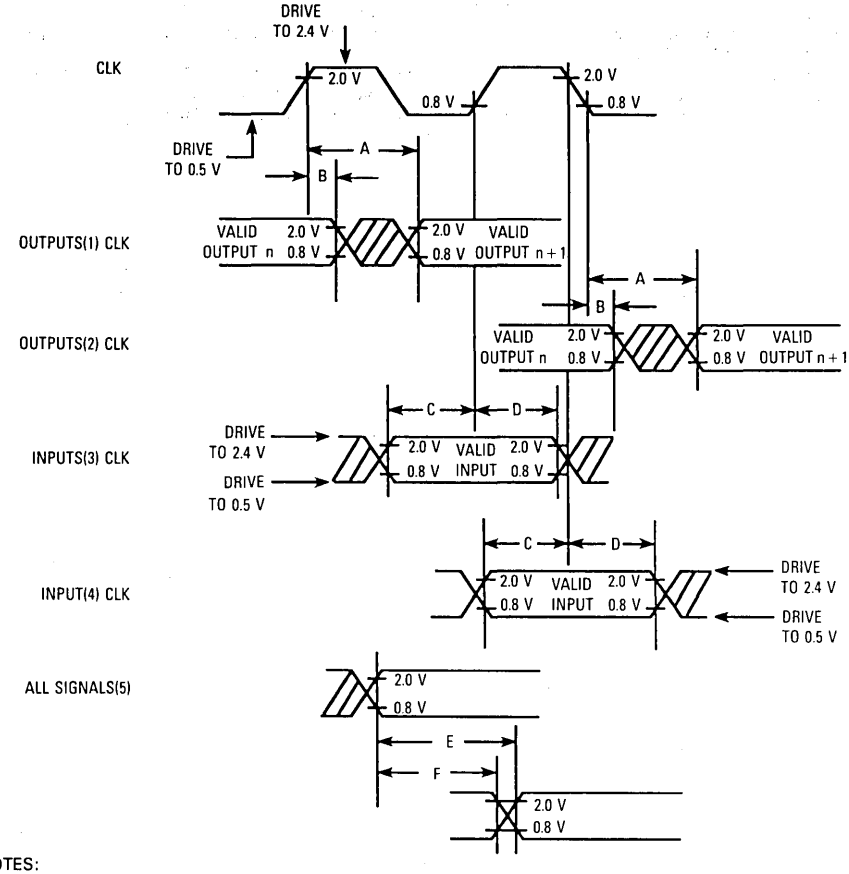
1. Timing measurements are referenced to and from a low voltage of 0.8 volts and a high voltage of 20 volts, unless otherwise noted. The voltage swing through this range should start outside, and pass through, the range such that the rise or fall will be linear between 0.8 volts and 2.0 volts.
2. These specifications only apply if the MC68882 has completed all internal operations initiated by the termination of the previous bus cycle when $\overline{DS}$ was negated.
3. Synchronous read cycles occur ***only*** when the save or response CIR locations are read.
4. This specification only applies to systems in which back-to-back accesses (read-write or write-write) of the operand CIR can occur. When the MC68882 is used as a coprocessor to the MC68020/MC68030, this can occur when the addressing mode is Immediate.
5. If the $\overline{SIZE}$ pin is ***not*** strapped to either V_{CC} or GND, it must have the same setup times as do addresses.
6. If the $\overline{SIZE}$ pin is ***not*** strapped to either V_{CC} or GND, it must have the same hold times as do addresses.
7. This number is reduced to 5 nanoseconds if $\overline{DSACK0}$ and $\overline{DSACK1}$ have equal loads.
8. START is not an external signal; rather, it is the logical condition that indicates the start of an access. The logical equation for this condition is $\overline{START} = \overline{CS} + \overline{AS} + R/\overline{W} \cdot \overline{DS}$.
9. If a subsequent access is not a FPCP access, $\overline{CS}$ must be negated before the assertion of $\overline{AS}$ and/or $\overline{DS}$ on the non-FPCP access. These specifications replace the old specifications 8 and 8A (the old specifications implied that in all cases, transitions in $\overline{CS}$ must not occur simultaneously with transitions of $\overline{AS}$ or $\overline{AS}$. This is not a requirement of the MC68881).

AC ELECTRICAL SPECIFICATION DEFINITIONS

The AC specifications presented consist of output delays, input setup and hold times, and signal skew times. All signals are specified relative to an appropriate edge of the clock input and, possibly, relative to one or more other signals.

The measurement of the AC specifications is defined by the waveforms shown below. In order to test the parameters guaranteed by Motorola, inputs must be driven to the voltage levels specified in the figure. Outputs are specified with minimum and/or maximum limits, as appropriate, and are measured as shown. Inputs are specified with minimum and, as appropriate maximum setup and hold times, and are measured as shown. Finally, the measurement for signal-to-signal specifications are also shown.

Note that the testing levels used to verify conformance to the AC specifications does not affect the guaranteed DC operation of the device as specified in the DC electrical characteristics.



NOTES:

1. This output timing is applicable to all parameters specified relative to the rising edge of the clock.
2. This output timing is applicable to all parameters specified relative to the falling edge of the clock.
3. This input timing is applicable to all parameters specified relative to the rising edge of the clock.
4. This input timing is applicable to all parameters specified relative to the falling edge of the clock.
5. This timing is applicable to all parameters specified relative to the assertion/negation of another signal.

LEGEND:

A. Maximum output delay specification.
B. Minimum output hold time.
C. Minimum input setup time specification.
D. Minimum input hold time specification.
E. Signal valid to signal valid specification (maximum or minimum).
F. Signal valid to signal invalid specification (maximum or minimum).

Figure 17. Drive Levels and Test Points for AC Specifications

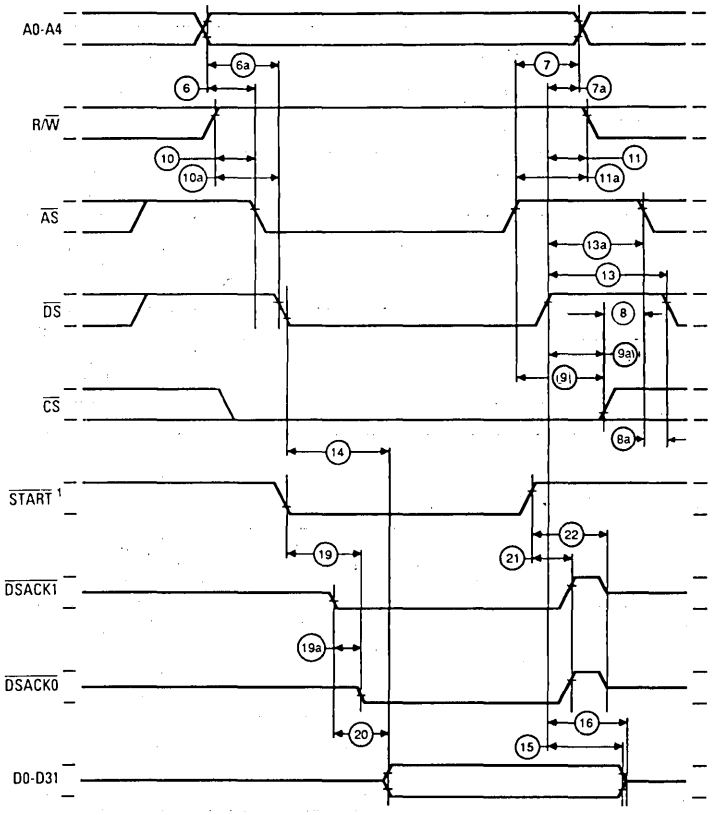


Figure 18. Asynchronous Read Cycle Timing Diagram

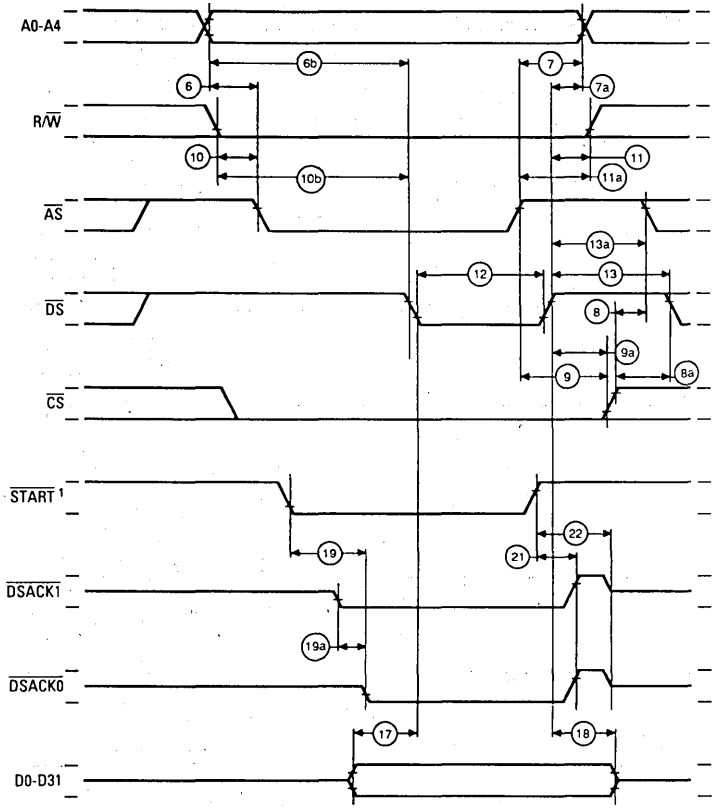


Figure 19. Asynchronus Write-Cycle Timing Diagram

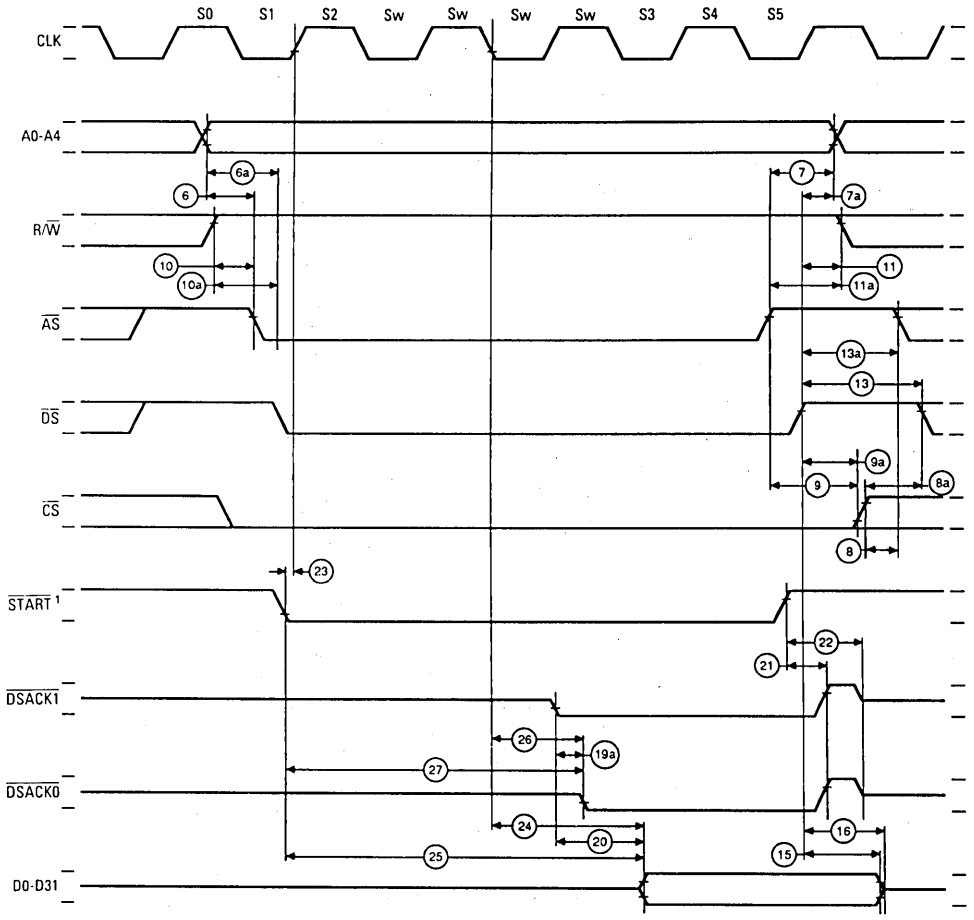
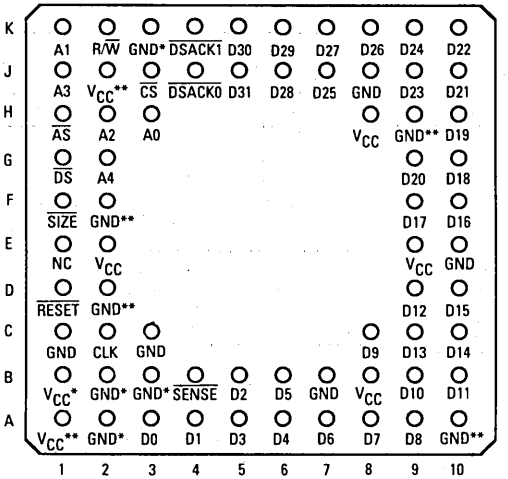


Figure 20. Synchronous Read-Cycle Timing Diagram

4

PIN ASSIGNMENTS



Pin Group	V_{CC}	GND
D31-D16	H8	J8
D15-D00	B8	B7
Internal Logic, $\overline{DSACK1}$, $\overline{DSACK0}$	E2, E9	A2, B2, B3, B4***, C3, E10, K3
Separate	—	C1
Extra	A1, B1, J2	A10, D2, F2, H9

*New assignment for the A93N mask.
**Reserved for future Motorola use.
***SENSE pin, may be used as an additional GND pin.

MC68882

Technical Summary

HCMOS Enhanced Floating-Point Coprocessor

The MC68882 floating-point coprocessor fully implements the IEEE Standard for Binary Floating-Point Arithmetic (ANSI-IEEE Standard 754-1985) for use with the Motorola M68000 Family of microprocessors. An upgrade of the MC68881, it is pin and software compatible with an optimized MPU interface providing in excess of 1.5 times the performance of the MC68881. It is implemented using VLSI technology to give systems designers the highest possible functionality in a physically small device.

Intended primarily for use as a coprocessor to the MC68020 or MC68030 32-bit microprocessor unit (MPU), the MC68882 provides a logical extension to the main MPU integer data processing capabilities. This extension is achieved by providing a very high performance floating-point arithmetic unit and a set of floating-point data registers which are analogous to the use of the integer data registers. The MC68882 instruction set is a natural extension of all earlier members of the M68000 Family, and it supports all of the addressing modes of the host MPU. Due to the flexible bus interface of the M68000 Family, the MC68882 can be used with any of the MPU devices of the M68000 Family and as a peripheral to non-M68000 processors.

The major features of the MC68882 are:

- Eight general purpose floating-point data registers, each supporting a full 80-bit extended precision real data format (a 64-bit mantissa plus a sign bit, and a 15-bit signed exponent).
- A 67-bit arithmetic unit to allow very fast calculations, with intermediate precision greater than the extended precision format.
- A 67-bit barrel shifter for high-speed shifting operations (for normalizing etc.).
- Special purpose hardware for high-speed conversion of binary real memory operands to and from the internal extended format.
- Reduced coprocessor interface overhead to increase throughput.
- Forty-six instructions, including 35 arithmetic operations.
- Full conformation to the ANSI-IEEE 754 standard, including all requirements and suggestions.
- Support of functions not defined by the IEEE standard, including a full set of trigonometric and transcendental functions.
- Seven data types: byte, word and long word integers; single, double, and extended precision real numbers; and packed binary coded decimal string real numbers.
- Twenty-two constants available in the on-chip ROM, including π, e, and powers of 10.
- Virtual memory/machine operations.
- Efficient mechanisms for procedure calls, context switches, and interrupt handling.
- Concurrent instruction execution with the main processor.
- Concurrent instruction execution of multiple floating-point instructions.
- Use with any host processor, on an 8-, 16-, or 32-bit data bus.

THE COPROCESSOR CONCEPT

The MC68882 functions as a coprocessor in systems where the MC68020 or MC68030 is the main processor via the M68000 coprocessor interface. It functions as a peripheral processor in systems where the main processor is the MC68000, MC68008, or MC68010.

The MC68882 utilizes the M68000 Family coprocessor interface to provide a logical extension of the MC68020 or MC68030 registers and instruction set in a manner which is transparent to the programmer. The programmer perceives the MPU/FPCP execution model as if both devices are implemented on one chip. A fundamental goal of the M68000 Family coprocessor interface is to provide the programmer with an execution model based upon sequential instruction execution by the MC68020 or MC68030 and the MC68882. For optimum performance, however, the coprocessor interface allows concurrent operations in the MC68882 with respect to the MC68020 or MC68030 whenever possible. In order to simplify the programmer's model, the coprocessor interface is designed to emulate, as closely as possible, non-concurrent operation between the MC68020 or MC68030 and the MC68882.

The MC68882 is a non-DMA type coprocessor which uses a subset of the general purpose coprocessor interface supported by the MC68020 or MC68030. Features of the interface implemented in the MC68882 are as follows:

- The main processor(s) and MC68882 communicate via standard M68000 bus cycles.
- The main processor(s) and MC68882 communications are not dependent upon the architecture of the individual devices (e.g., instruction pipes or caches, addressing modes).
- The main processor(s) and MC68882 may operate at different clock speeds.
- MC68882 instructions utilize all addressing modes provided by the main processor.
- All effective addresses are calculated by the main processor at the request of the coprocessor.
- All data transfers are performed by the main processor at the request of the MC68882.
- Overlapped (concurrent) instruction execution enhances throughput while maintaining the programmer's model of sequential instruction execution.
- Coprocessor detection of exceptions which require a trap to be taken are serviced by the main processor at the request of the MC68882.
- Support of virtual memory/virtual machine systems is provided via the FSAVE and FRESTORE instructions.
- Up to eight coprocessors may reside in a system simultaneously. Multiple coprocessors of the same type are allowed.
- Systems may use software emulation of the MC68882 without reassembling or relinking user software.

HARDWARE OVERVIEW

The MC68882 is a high performance floating-point device designed to interface with the MC68020 or MC68030 as a coprocessor. This device fully supports the MC68020 or MC68030 virtual machine architecture and is implemented in HCMOS, Motorola's low power, small geometry process. This process allows CMOS and HMOS (high density NMOS) gates to be combined on the same device. CMOS structures are used where speed and low power is required, and HMOS structures are used where minimum silicon area is desired. Using this technology increases speed performance while using low power consumption, yet still confines the MC68881 to a reasonably small die size.

The MC68882 can also be used as a peripheral processor in systems where the MC68020 or MC68030 is not the main processor (e.g., MC68000, MC68008, MC68010). The configuration of the MC68882 as a peripheral processor or coprocessor may be completely transparent to user software (i.e., the same object code may be executed in either configuration).

The architecture of the MC68882 appears to the user as a logical extension of the M68000 Family architecture. Because of the coupling of the coprocessor interface, the MC68020 or MC68030 programmer can view the MC68882 registers as though the registers are resident in the MC68020 or MC68030. Thus, a MC68020 or MC68030 and an MC68882 device pair functions as one processor with eight integer data registers, eight address registers, and eight floating-point data registers supporting seven floating-point and integer data types.

The MC68882 programming model is shown in Figures 1 through 6 and consists of the following:

- Eight 80-bit floating-point data registers (FP0-FP7). These registers are analogous to the integer data registers (D0-D7) and are completely general purpose (i.e., any instruction can use any register).
- A 32-bit control register that contains enable bits for each class of exception trap, and mode bits to set the user-selectable rounding and precision modes.
- A 32-bit status register that contains floating-point condition codes, quotient bits, and exception status information.
- A 32-bit instruction address register that contains the main processor memory address of the last floating-point instruction that was executed. This address is used in exception handling to locate the instruction that caused the exception.

The connection between the MC68020 or MC68030 and the MC68882 is a simple extension of the M68000 bus interface. The MC68882 is connected as a coprocessor to the MC68020 or MC68030, and the selection of the MC68882 is based on a chip select which is decoded from the MC68020 or MC68030 function codes and address bus. Figure 7 illustrates the MPU/coprocessor configuration.

As shown in Figure 8, the MC68882 is internally divided into three processing elements: the bus interface unit

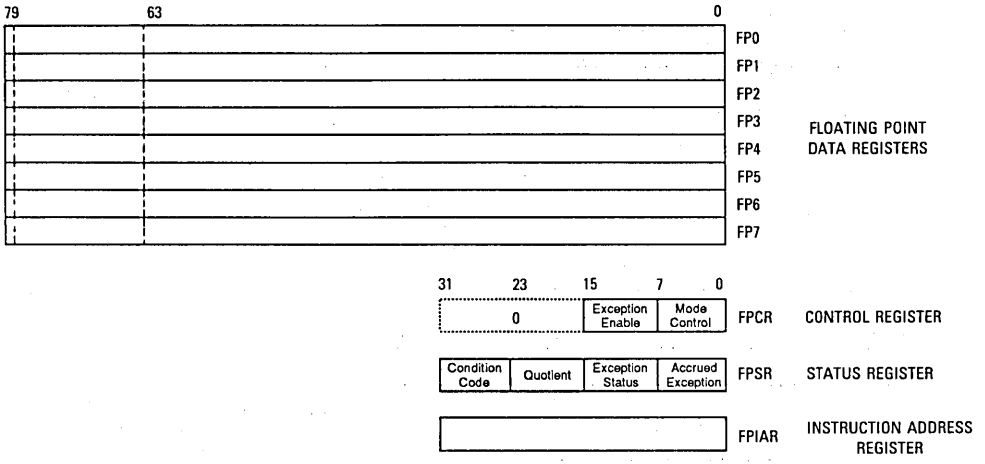


Figure 1. MC68882 Programming Model

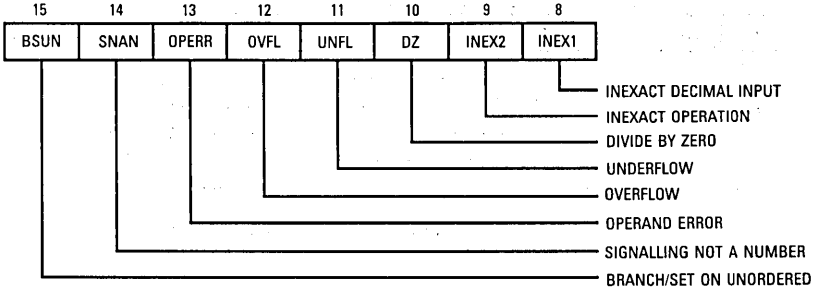


Figure 2. Exception Status/Enable Byte

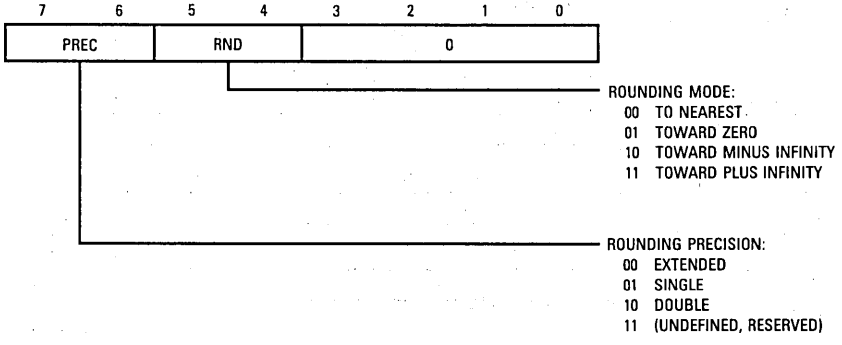


Figure 3. Mode Control Byte

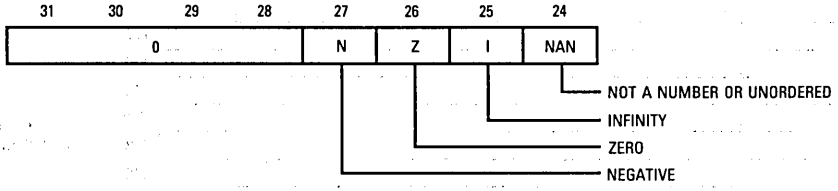


Figure 4. Condition Code Byte

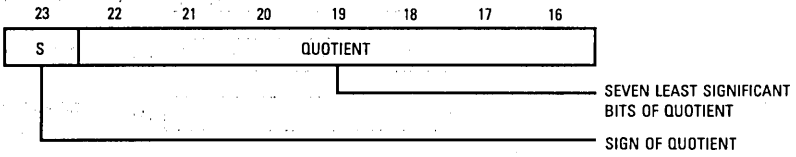


Figure 5. Quotient Byte

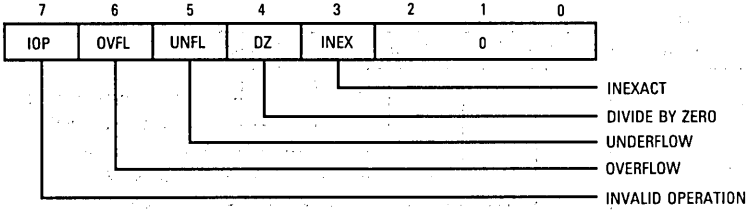


Figure 6. Accrued Exception Byte

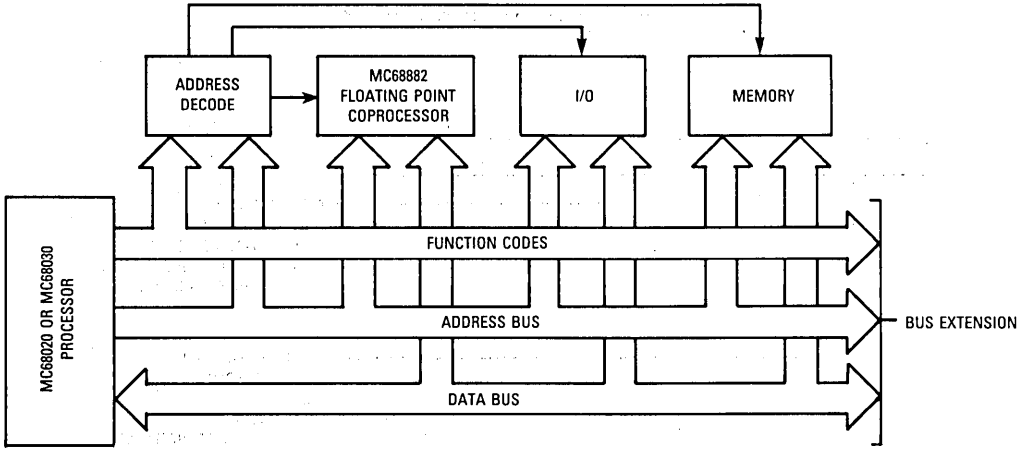


Figure 7. Typical Coprocessor Configuration

(BIU), the conversion unit (CU), and the arithmetic processing unit (APU). The BIU communicates with the MC68020 or MC68030, the CU performs data conversion for binary real data formats, and the APU executes all MC68882 instructions.

The BIU contains the coprocessor interface registers (CIRs). In addition to these registers, the register select and DSACK timing control logic is contained in the BIU. Finally, the status flags used to monitor the status of communications with the main processor are contained in the BIU.

The CU contains special purpose hardware that performs data format conversions between binary real data formats to and from the internal extended format. The CU relieves the APU of a significant work load and allows the MC68882 to execute data movement and preparation functions concurrently with arithmetic and transcendental calculations.

The eight 80-bit floating-point data registers (FP0-FP7) and the 32-bit control, status, and instruction address registers (FPCR, FPSR and FPIAR) are located in the APU. In addition to these registers, the APU contains a high-speed 67-bit arithmetic unit used for both mantissa and exponent calculations, a barrel shifter that can shift from 1 bit to 67 bits in one machine cycle, and ROM constants (for use by the internal algorithms or user programs).

The control section of the APU contains the clock generator, a two-level microcode sequencer, the microcode ROM, and self-test circuitry. The built-in self-test capabilities of the MC68882 enhance reliability and ease manufacturing requirements; however, these diagnostic functions are not accessible outside of the special test environment supported by VLSI test equipment.

BUS INTERFACE UNIT

All communications between the MC68020 or MC68030 and the MC68882 occur via standard M68000 Family bus transfers. The MC68882 is designed to operate on 8-, 16-, or 32-bit data buses.

The MC68882 contains a number of coprocessor interface registers (CIRs) that are addressed in the same manner as memory by the main processor. The M68000 Family coprocessor interface is implemented via a protocol of reading and writing to these registers by the main processor. The MC68020 and MC68030 implement this general purpose coprocessor interface protocol in hardware and microcode.

When the MC68020 or MC68030 detects a general type MC68882 instruction, the MC68020 or MC68030 writes the instruction to the memory-mapped command CIR and reads the response CIR. In this response, the BIU encodes requests for any additional action required of the MC68020 or MC68030 on behalf of the MC68882. For example, the response may request that the MC68020 or MC68030 fetch an operand from the evaluated effective address and transfer the operand to the operand CIR. Once the MC68020 or MC68030 fulfills the coprocessor request(s), the MC68020 or MC68030 is free to fetch and execute subsequent instructions.

The only difference between a coprocessor bus transfer and any other bus transfer is that the MC68020 or MC68030 issues a CPU address space function code during the cycle. (The function codes are generated by the M68000 Family processors to identify eight separate address spaces.) Thus, the memory-mapped coprocessor interface registers do not infringe upon instruction or data address spaces. The MC68020 or MC68030 places a coprocessor ID field from the coprocessor instruction onto three of the upper address lines during coprocessor accesses. This ID, along with the CPU address space function code, is decoded to select one of eight coprocessors in the system.

Since the coprocessor interface protocol is based solely on bus transfers, the protocol is easily emulated by software when the MC68882 is used as a peripheral with any processor capable of memory-mapped I/O over an M68000 style bus. When used as a peripheral processor with the 8-bit MC68008, the 16-bit MC68000, or the MC68010, all MC68882 instructions are trapped by the main processor to an exception handler at execution time. Thus, the software emulation of the coprocessor interface protocol can be totally transparent to the user. The MC68882 can provide a performance option for MC68000-based designs by changing the main processors to the MC68020 or MC68030. The software migrates without change to the next generation equipment using the MC68020 or MC68030.

Since the bus is asynchronous, the MC68882 need not run at the same clock speed as the main processor. Total system performance may therefore be customized. For a given CPU performance requirement, the floating-point performance can be selected to meet particular price/performance specifications, running the MC68882 at slower (or faster) clock speeds than the MPU clock.

COPROCESSOR INTERFACE

The M68000 Family coprocessor interface is an integral part of the MC68882 and MC68020 or MC68030 designs. The interface partitions MPU and coprocessor operations so that the MC68020 or MC68030 does not have to completely decode coprocessor instructions, and the MC68882 does not have to duplicate main processor functions (such as effective address evaluation). This partitioning provides an orthogonal extension of the instruction set by permitting MC68882 instructions to utilize all MC68020 or MC68030 addressing modes and to generate execution time exception traps. Thus, from the programmer's view, the MPU and coprocessor appear to be integrated onto a single chip.

While the execution of the great majority of MC68882 instructions may be overlapped with the execution of MC68020 or MC68030 instructions, concurrency is completely transparent to the programmer. The MC68020 and MC68030 single-step and program flow (trace) modes are fully supported by the MC68882 and the M68000 Family coprocessor interface.

While the M68000 Family coprocessor interface permits coprocessors to be bus masters, the MC68882 is never a bus master. The MC68882 requests that the MC68020 or MC68030 fetch all operands and store all results. In this manner, the MC68020 and MC68030 32-bit data bus provides high speed transfer of floating-point

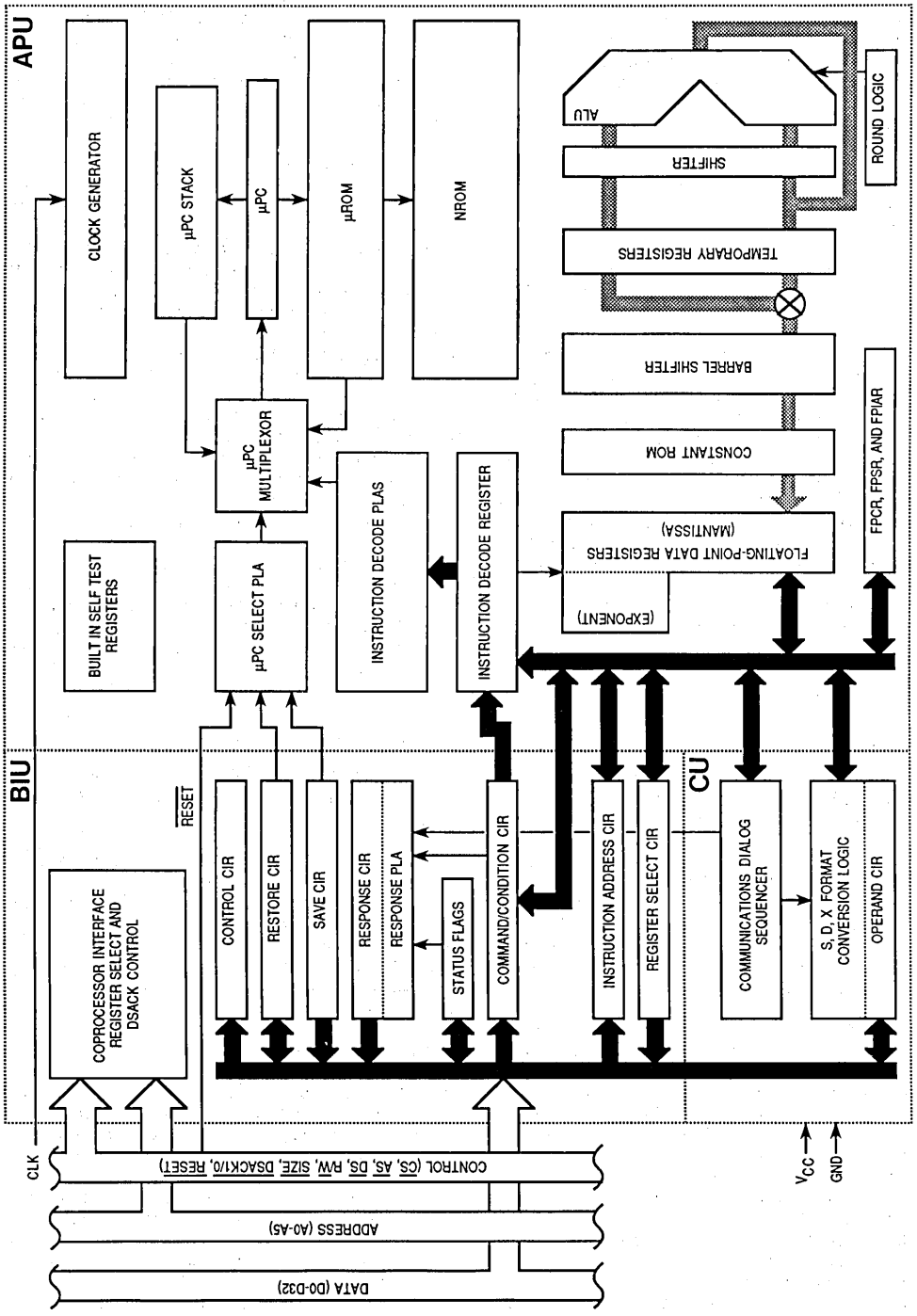


Figure 8. MC68882 Simplified Block Diagram

operands and results while simplifying the design of the MC68882.

Since the coprocessor interface is based solely upon bus cycles (to and from CPU space) and the MC68882 is never a bus master, the MC68882 can be placed on either the logical or physical side of the system memory management unit in an MC68020-based system. Since the memory management unit of the MC68030 is on-chip, the MC68882 is always on the physical side of the memory management unit in an MC68030 system.

The virtual machine architecture of the MC68020 or MC68030 is supported by the coprocessor interface and the MC68882 through the FSAVE and FRESTORE instructions. If the MC68020 or MC68030 detects a page fault and/or a task time out, the MC68020 or MC68030 can force the MC68882 to stop whatever operation is in progress at any time and save the MC68882 internal state in memory. During the execution of a floating-point instruction, the MC68882 can stop at predetermined points as well as at the completion of the instruction.

The size of the saved internal state of the MC68882 is dependent upon the state of the APU at the time the FSAVE is executed. If the MC68882 is in the reset state when the FSAVE instruction is received, only one word of state is transferred to memory, which may be examined by the operating system to determine that the coprocessor programmer's model is empty. If the coprocessor is in the idle state when the save instruction is received, only a few words of internal state are transferred to memory. If executing an instruction in the busy state, it may be necessary to save the entire internal state of the machine. Instructions completing execution in less time than it takes to save the larger state in mid-instruction are allowed to complete execution and then save the idle state. Thus, the size of the saved internal state is kept to a minimum. The ability to utilize several internal state sizes greatly reduces the average context switching time.

The FRESTORE instruction permits reloading of an internal state that was saved earlier and continues any operation that was previously suspended. An FRESTORE of the null state frame re-establishes default register values, a function identical to the MC68882 hardware reset.

MC68882 PERFORMANCE ENHANCEMENTS

The high performance of the MC68882 is the result of the MC68882's abilty to execute multiple floating-point instructions concurrently. The direct result of concurrency is to utilize the Arithmetic Processing Unit (APU) more efficiently by decreasing its idle time.

When the MC68882 receives an instruction, the BIU, along with the CU, can initiate the instruction, fetch the necessary operands, and convert them to the internal extended format even though the APU is busy completing execution of a previous instruction. Although the MC68881 can only instruct the main processor to wait if the APU is busy, the MC68882 CU can proceed with the next instruction. When the APU is finally ready to perform the calculation, it can do so immediately without incurring delay due to data movement and preparation functions.

Another factor in obtaining increased performance in the MC68882 is the optimized FMOVE instructions for binary real data formats. These FMOVE instructions execute twice as fast as the corresponding FMOVE instructions of the MC68881. The FMOVE instructions are also potentially fully concurrent and, therefore, can be completely executed during the execution of a previous instruction.

The MC68882 also has a more optimized coprocessor interface than the MC68881. If an arithmetic instruction has data formats of Single, Double or Extended, the dialogs are designed to increase the potential overlap with subsequent instructions. This overlap can significantly decrease the effective instruction execution time.

OPERAND DATA FORMATS

The MC68882 supports the following data formats:

- Byte Integer (B)
- Word Integer (W)
- Long Word Integer (L)
- Single Precision Real (S)
- Double Precision Real (D)
- Extended Precision Real (X)
- Packed Decimal String Real (P)

The capital letters contained in parentheses denote suffixes added to instructions in the assembly language source specifying the data format to be used.

INTEGER DATA FORMATS

The three integer data formats (byte, word, and long word) are the standard twos complement data formats supported in the M68000 Family architecture. Whenever an integer is used in a floating-point operation, the integer is automatically converted by the MC68882 to an extended precision floating-point number before being used. For example, to add an integer constant of five to the number contained in floating-point data register 3 (FP3), the following instruction can be used:

```
FADD.W #5,FP3
```

(The Motorola assembler syntax "#" is used to denote immediate addressing.)

The ability to effectively use integers in floating-point operations saves user memory since an integer representation of a number, if representable, is usually smaller than the equivalent floating-point representation.

FLOATING-POINT DATA FORMATS

The floating-point data formats, single precision (32-bits) and double precision (64-bits), are defined by the IEEE standard. These data formats are the main floating-point formats and should be used for most calculations involving real numbers. Table 1 lists the exponent and mantissa size for single, double, and extended precision. The exponent is biased, and the mantissa is in sign and magnitude form. Since single and double precision require normalized numbers, the most-significant bit of the mantissa is implied as a one and is not included, thus giving one extra bit of precision.

The extended precision data format is also in conformance with the IEEE standard, but the standard does not specify this format to the bit level whereas it does for

Table 1. Exponent and Mantissa Sizes

Data Format	Exponent Bits	Mantissa Bits	Bias
Single	8	23(+1)	127
Double	11	52(+1)	1023
Extended	15	64	16383

single and double precision. The memory format on the MC68882 consists of 96 bits (three long words). Only 80 bits are actually used; the other 16 bits are for future expandability and for long-word alignment of floating-point data structures. Extended format has a 15-bit exponent, a 64-bit mantissa, and a 1-bit mantissa sign.

Extended precision numbers are intended for use as temporary variables, intermediate values, or in areas where extra precision is needed. For example, a compiler might select extended precision arithmetic for evaluation of the right side of an equation with mixed sized data and then convert the answer to the data type on the left side of the equation. It is anticipated that extended precision data will not be stored in large arrays due to the amount of memory required by each value.

PACKED DECIMAL STRING REAL DATA FORMAT

The packed decimal data format allows packed BCD strings to be transferred to and from the MC68882. The strings consist of a 3-digit base 10 exponent and a 17-digit base 10 mantissa. Both the exponent and mantissa have a separate sign bit. All digits are packed BCD; an entire string fits in 96 bits (three long words). As is the case with all data formats when packed BCD strings are supplied to the MC68882, the strings are automatically converted to extended precision real values. This conversion allows packed BCD numbers to be used as inputs to any operation. For example:

```
FADD.P   #-6.023E+24,FP5
```

BCD numbers can be output from the MC68882 in a format readily used for printing by a program generated by a high-level language compiler. For example:

```
FMOVE.P  FP3,BUFFER{#-5}
```

This instruction converts the floating-point data register 3 (FP3) contents into a packed BCD string with five digits to the right of the decimal point (FORTRAN F format).

DATA FORMAT SUMMARY

All data formats described above are supported orthogonally by all arithmetic and transcendental operations and by all appropriate MC68020 or MC68030 addressing modes. For example, all of the following are legal instructions:

```
FADD.B    #0,FP0
FADD.W    D2,FP3
FADD.L    BIGINT,FP7
FADD.S    #3.14159,FP5
FADD.D    (SP)+,FP6
FADD.X    [(TEMP_PTR,A7)],FP3
FADD.P    #1.23E25,FP0
```

Most on-chip calculations are performed in the extended precision format, and the eight floating-point data registers always contain extended precision values. All operands used are converted to extended precision by the MC68882 before a specific operation is performed, and all results are in extended precision. The use of extended precision ensures maximum accuracy without sacrificing performance. Refer to Figure 9 for a summary of the memory formats for the seven data formats supported by the MC68882.

INSTRUCTION SET

The MC68882 instruction set is organized into six major classes:

1. Moves between the MC68882 and memory or the MC68020 or MC68030 (in and out),
2. Move multiple registers (in and out),
3. Monadic operations,
4. Dyadic operations,
5. Branch, set, or trap conditionally, and
6. Miscellaneous.

MOVES

On all moves from memory (or from an MC68020 or MC68030 data register) to the MC68882, data is converted from the source data format to the internal extended precision format. On all moves from the MC68882 to memory (or to an MC68020 or MC68030 data register), data is converted from the internal extended precision format to the destination data format. Note that data movement instructions perform arithmetic operations, since the result is always rounded to the precision selected in the FPCR mode control byte. The result is rounded using the selected rounding mode and is checked for overflow and underflow.

The syntax for the move is:

```
FMOVE.<fmt> <ea>,FPn     Move to MC68882
FMOVE.<fmt> FPm,<ea>     Move from MC68882
FMOVE.X     FPm,FPn      Move within MC68882
```

where:

<ea> is an MC68020 or MC68030 effective address operand.
.<fmt> is the data format size.
FPm and FPn are floating-point data registers.

MOVE MULTIPLE REGISTERS

The floating-point move multiple instructions on the MC68882 are much like the integer counterparts on the M68000 Family processors. Any set of the floating-point registers FP0 through FP7 can be moved to or from memory with one instruction. These registers are always moved as 96-bit extended data with no conversion (hence no possibility of conversion errors). Some examples of the move multiple instruction are as follows:

```
FMOVEM    ⟨ea⟩,FP0-FP3/FP7
FMOVEM    FP2/FP4/FP6,<ea>
```

The move multiple instructions are useful during context switches and interrupts to save or restore the state

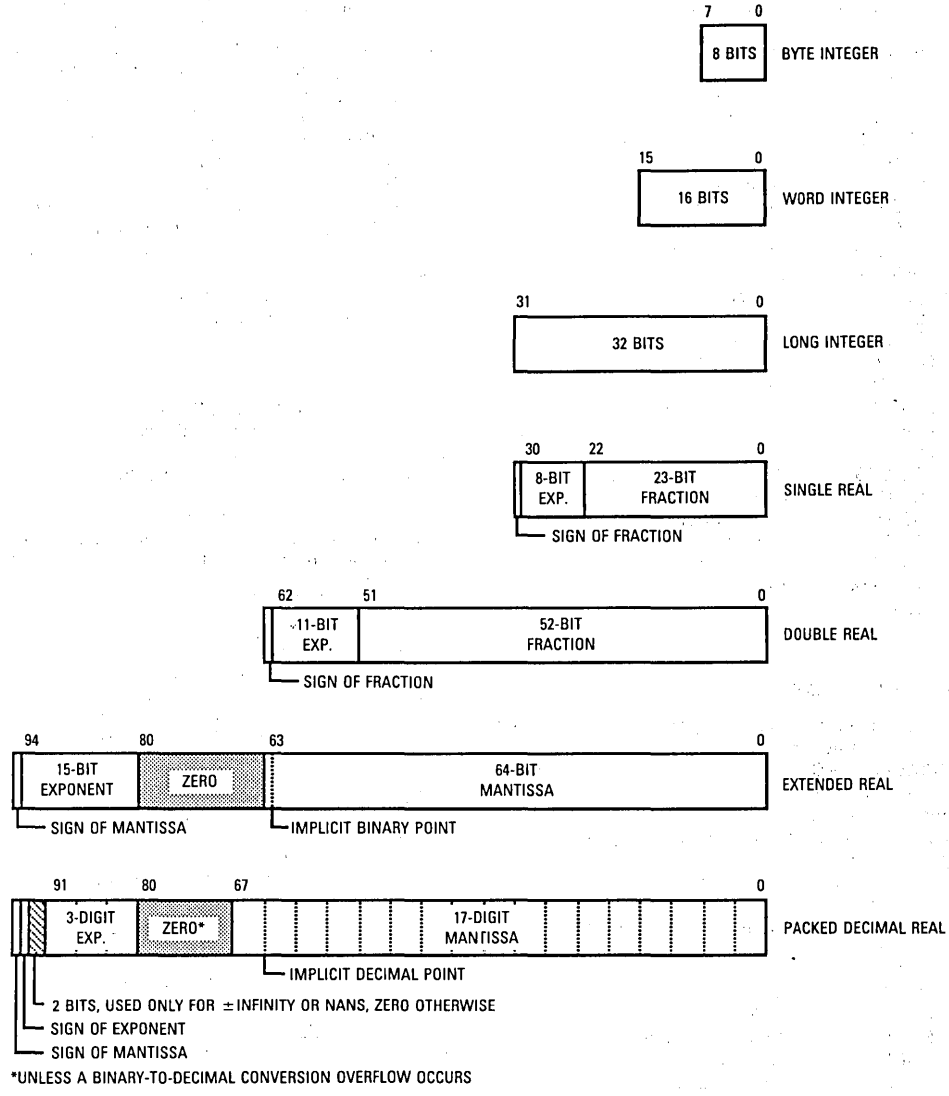


Figure 9. MC68882 Data Format Summary

of a program. These moves are also useful at the start and end of a procedure to save and restore the register set of the calling routine. In order to reduce procedure call overhead, the list of registers to be saved or restored can be contained in a data register thus enabling run-time optimization by allowing a called routine to save as few registers as possible. Note that no rounding or overflow/underflow checking is performed by these operations.

MONADIC OPERATIONS

Monadic operations have one operand. This operand may be in a floating-point data register, memory, or in an MC68020 or MC68030 data register. The result is always stored in a floating-point data register. For example, the syntax for square root is:

```
FSQRT.<fmt>   <ea>,FPn or,
FSQRT.X       FPm,FPn  or,
FSQRT.X       FPn
```

The MC68882 monadic operations available are as follows:

FABS	Absolute Value
FACOS	Arc Cosine
FASIN	Arc Sine
FATAN	Arc Tangent
FATANH	Hyperbolic Arc Tangent
FCOS	Cosine
FCOSH	Hyperbolic Cosine
FETOX	e to the x Power
FETOXM1	e to the X Power −1
FGETEXP	Get Exponent
FGETMAN	Get Mantissa
FINT	Integer Part
FLINTRZ	Integer Part (Truncated)
FLOG10	Log Base 10
FLOG2	Log Base 2
FLOGN	Log Base e
FLOGNP1	Log Base e of(x+1)
FNEG	Negate
FSIN	Sine
FSINCOS	Simultaneous Sine and Cosine
FSINH	Hyperbolic Sine
FSQRT	Square Root
FTAN	Tangent
FTANH	Hyperbolic Tangent
FTENTOX	10 to the x Power
FTST	Test
FTWOTOX	2 to the x Power

4

DYADIC OPERATIONS

Dyadic operations have two operands each. The first operand is in a floating-point data register, memory, or an MC68020 or MC68030 data register. The second operand is the contents of a floating-point data register. The destination is the same floating-point data register used for the second operand. For example, the syntax for floating-point add is:

FADD.<fmt> <ea>,FPn
FADD.X FPm,FPn

The dyadic operations available with the MC68882 are as follows:

FADD	Add
FCMP	Compare
FDIV	Divide
FMOD	Modulo Remainder
FMUL	Multiply
FREM	IEEE Remainder
FSCALE	Scale Exponent
FSGLDIV	Single Precision Divide
FSGLMUL	Single Precision Multiply
FSUB	Subtract

BRANCH, SET, AND TRAP-ON CONDITION

The floating-point branch, set, and trap-on condition instructions implemented by the MC68882 are similar to the equivalent integer instructions of the M68000 Family processors, except more conditions exist due to the special values in IEEE floating-point arithmetic. When a conditional instruction is executed, the MC68882 performs the necessary condition checking and reports to the MC68020 or MC68030 whether the condition is true or false. The MC68020 or MC68030 then takes the appropriate action. Since the MC68882 and MC68020 or MC68030 are closely coupled, the floating-point branch operations execute very quickly.

The MC68882 conditional operations are:

FBcc	Branch
FDBcc	Decrement and Branch
FScc	Set According to Condition
FTRAPcc	Trap-on Condition (with an Optional Parameter)

where:

cc is one of the 32 floating-point conditional test specifiers as given in Table 2.

Table 2. Floating-Point Conditional Test Specifiers

Mnemonic	Definition
NOTE The following conditional tests do not set the BSUN bit in the status register exception byte under any circumstances.	
F	False
EQ	Equal
OGT	Ordered Greater Than
OGE	Ordered Greater Than or Equal
OLT	Ordered Less Than
OLE	Ordered Less Than or Equal
OGL	Ordered Greater or Less Than
OR	Ordered
UN	Unordered
UEQ	Unordered or Equal
UGT	Unordered or Greater Than
UGE	Unordered or Greater or Equal
ULT	Unordered or Less Than
ULE	Unordered or Less or Equal
NE	Not Equal
T	True
NOTE All the conditional tests below set the BSUN bit in the status register exception byte if the NAN condition code bit is set when a conditional instruction is executed.	
SF	Signaling False
SEQ	Signaling Equal
GT	Greater Than
GE	Greater Than or Equal
LT	Less Than
LE	Less Than or Equal
GL	Greater or Less Than
GLE	Greater Less or Equal
NGLE	Not (Greater, Less or Equal)
NGL	Not (Greater or Less)
NLE	Not (Less or Equal)
NLT	Not (Less Than)
NGE	Not (Greater or Equal)
NGT	Not (Greater Than)
SNE	Signaling Not Equal
ST	Signaling True

MISCELLANEOUS INSTRUCTIONS

Miscellaneous instructions include moves to and from the status, control, and instruction address registers. Also included are the virtual memory/machine FSAVE and FRESTORE instructions that save and restore the internal state of the MC68882.

FMOVE	<ea>,FPcr	Move to Control Register(s)
FMOVE	FPcr,<ea>	Move from Control Register(s)
FSAVE	<ea>	Virtual Machine State Save
FRESTORE	<ea>	Virtual Machine State Restore

ADDRESSING MODES

The MC68882 does not perform address calculations. Thus, if the MC68882 instructs the MC68020 or MC68030 to transfer an operand via the coprocessor interface, the MC68020 or MC68030 performs the addressing mode calculations requested in the instruction. In this case, the instruction is encoded specifically for the MC68020 or MC68030, and the execution of the MC68882 is dependent only on the value of the command word written to the MC68882 by the main processor.

This interface is flexible and allows any addressing mode to be used with floating-point instructions. For the M68000 Family, these addressing modes include immediate, postincrement, predecrement, data or address register direct, and the indexed/indirect addressing modes of the MC68020 and MC68030. Some addressing modes are restricted for instructions consistent with the M68000 Family architectural definitions (e.g., program counter relative addressing is not allowed for a destination operand).

The orthogonal instruction set of the MC68882 and the flexible branches and addressing modes of the MC68020/MC68030 allow a programmer or a compiler writer to think of the MC68882 as though it is part of the MC68020 or MC68030. There are no special restrictions imposed by the coprocessor interface, and floating-point arithmetic is coded exactly like integer arithmetic.

MC68881 COMPATIBILITY

Using the MC68882 in an existing MC68881 socket does not require hardware changes nor user-software modifications. Implementation of multiple floating-point instruction execution concurrency gives the MC68882 a performance advantage over the MC68881. However, to guarantee that the floating-point exception model maintains the precepts of a sequential execution model, some systems-level software modifications are needed to upgrade the system to operate properly with an MC68882.

First, note that the idle and busy state frames (generated by the FSAVE instruction) are both 32 bytes larger with the MC68882 than the MC68881. The offsets for the exceptional operand, the operand register word, and the BIU flag word from the top of the saved idle state frame are 32 bytes more than that of the MC68881. However, a unique format word is generated by the MC68882 enabling the system software to detect this difference. The unique format word prevents a saved MC68881 context from being restored into an MC68882 and vice versa.

Second, the BSUN (Branch or Set on Unordered), SNAN (Signaling Not-A-Number), OPERR (Operand Error), OVFL (Overflow), DZ (Divide by Zero) and INEX (Inexact result) floating-point exception handlers must have these minimum requirements:

1. An FSAVE must be executed before any other floating-point instruction.
2. A BSET or similar instruction that sets bit 27 of the BIU flag word (located in the saved idle state frame).
3. An FRESTORE instruction must be executed before the RTE instruction.

The above requirements are not applicable to interrupt handlers that do not contain any floating-point instructions. For interrupt handlers that have floating-point instructions, only requirements #1 and #3 must be implemented.

FUNCTION SIGNAL DESCRIPTIONS

The following paragraphs contain a brief description of the input and output signals for the MC68882 floating-point coprocessor. The signals are functionally organized into groups as shown in Figure 10.

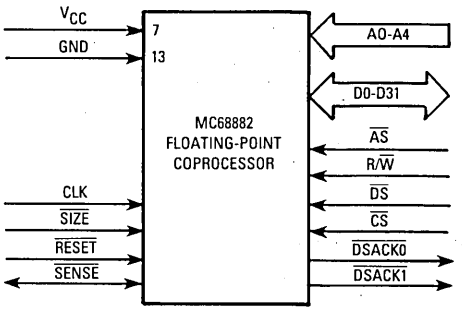


Figure 10. MC68882 Input/Output Signals

NOTE

The terms **assertion** and **negation** are used extensively to avoid confusion when describing "active-low" and "active-high" signals. The term **assert** or **assertion** is used to indicate that a signal is active or true, independent of whether that level is represented by a high or low voltage. The term **negate** or **negation** is used to indicate that a signal is inactive or false.

ADDRESS BUS (A0 through A4)

These active-high address line inputs are used by the main processor to select the coprocessor interface register locations located in the CPU address space. These lines control the register selection as listed in Table 3.

When the MC68882 is configured to operate over an 8-bit data bus, the A0 pin is used as an address signal for byte accesses of the coprocessor interface registers. When the MC68882 is configured to operate over a 16- or 32-bit system data bus, both the A0 and the SIZE pins are strapped high and/or low as listed in Table 4.

Table 3. Coprocessor Interface Register Selection

A4-A0	Offset	Width	Type	Register
0000x	$00	16	Read	Response
0001x	$02	16	Write	Control
0010x	$04	16	Read	Save
0011x	$06	16	R/$\overline{W}$	Restore
0100x	$08	16	—	(Reserved)
0101x	$0A	16	Write	Command
0110x	$0C	16	—	(Reserved)
0111x	$0E	16	Write	Condition
100xx	$10	32	R/$\overline{W}$	Operand
1010x	$14	16	Read	Register Select
1011x	$16	16	—	(Reserved)
110xx	$18	32	Read	Instruction Address
111xx	$1C	32	R/$\overline{W}$	Operand Address

Table 4. System Data Bus Size Configuration

A0	$\overline{SIZE}$	Data Bus
—	Low	8-Bit
Low	High	16-Bit
High	High	32-Bit

DATA BUS (D0 through D31)

This 32-bit, bidirectional, three-state bus serves as the general purpose data path between the MC68020/MC68030 and the MC68882. Regardless of whether the MC68882 is operated as a coprocessor or a peripheral processor, all inter-processor transfers of instruction information, operand data, status information, and requests for service occur as standard M68000 bus cycles.

The MC68882 will operate over an 8-, 16-, or 32-bit system data bus. Depending upon the system data bus configuration, both the A0 and $\overline{SIZE}$ pins are configured specifically for the applicable bus configuration. (Refer to **ADDRESS BUS (A0 through A4)** and **SIZE ($\overline{SIZE}$)** for further details).

SIZE ($\overline{SIZE}$)

This active-low input signal is used in conjunction with the A0 pin to configure the MC68882 for operation over an 8-, 16-, or 32-bit system data bus. When the MC68882 is configured to operate over a 16- or 32-bit system data bus, both the $\overline{SIZE}$ and A0 pins are strapped high and/or low as listed in Table 4.

ADDRESS STROBE ($\overline{AS}$)

This active-low input signal indicates that there is a valid address on the address bus, and both the chip select ($\overline{CS}$) and read/write (R/$\overline{W}$) signal lines are valid.

CHIP SELECT ($\overline{CS}$)

This active-low input signal enables the main processor access to the MC68882 coprocessor interface registers. When operating the MC68882 as a peripheral processor, the chip select decode is system dependent (i.e., like the chip select on any peripheral).

READ/WRITE (R/$\overline{W}$)

This input signal indicates the direction of a bus transaction (read/write) by the main processor. A logic high (1) indicates a read from the MC68882, and a logic low (0) indicates a write to the MC68882. The R/$\overline{W}$ signal must be valid when $\overline{AS}$ is asserted.

DATA STROBE ($\overline{DS}$)

This active-low input signal indicates that there is valid data on the data bus during a write bus cycle.

DATA TRANSFER AND SIZE ACKNOWLEDGE ($\overline{DSACK0}$, $\overline{DSACK1}$)

These active-low, three-state output signals indicate the completion of a bus cycle to the main processor. The MC68882 asserts both the $\overline{DSACK0}$ and $\overline{DSACK1}$ signals upon assertion of $\overline{CS}$.

If the bus cycle is a main processor read, the MC68882 asserts $\overline{DSACK0}$ and $\overline{DSACK1}$ signals to indicate that the informatin on the data bus is valid. (Both DSACK signals may be asserted in advance of the valid data being placed on the bus.) If the bus cycle is a main processor write to the MC68882, $\overline{DSACK0}$ and $\overline{DSACK1}$ are used to acknowledge acceptance of the data by the MC68882.

The MC68882 also uses $\overline{DSACK0}$ and $\overline{DSACK1}$ signals to dynamically indicate to the MC68020/MC68030 the "port" size (system data bus width) on a cycle-by-cycle basis. Depending upon which of the two DSACK pins are asserted in a given bus cycle, the MC68020/MC68030 assumes data has been transferred to/from an 8-, 16-, or 32-bit wide data port. Table 5 lists the DSACK assertions that are used by the MC68882 for the various bus cycles over the various system data bus configurations.

Table 5 indicates that all accesses over a 32-bit bus where A4 equals zero are to 16-bit registers. The MC68882 implements all 16-bit coprocessor interface registers on data lines D16-D31 (to eliminate the need for on-chip multiplexers); however, the MC68020/MC68030 expects 16-bit registers that are located in a 32-bit port at odd word addresses (A1 = 1) to be implemented on data lines D0-D15. For accesses to these registers when configured for 32-bit bus operation, the MC68882 generates DSACK signals as listed in Table 5 to inform the MC68020/MC68030 of valid data on D16-D31 instead of D0-D15.

An external holding resistor is required to maintain both $\overline{DSACK0}$ and $\overline{DSACK1}$ high between bus cycles. In order to reduce the signal rise time, the $\overline{DSACK0}$ and $\overline{DSACK1}$ lines are actively pulled up (negated) by the MC68882 following the rising edge of $\overline{AS}$ or $\overline{DS}$, and both DSACK lines are then three-stated (placed in the high-impedance state) to avoid interference with the next bus cycle.

Table 5. DSACK Assertions

Data bus	A4	$\overline{DSACK1}$	$\overline{DSACK0}$	Comments
32-Bit	1	Low	Low	Valid Data on D31-D0
32-Bit	0	Low	High	Valid Data on D31-D16
16-Bit	x	Low	High	Valid Data on D31-D16 or D15-D0
8-Bit	x	High	Low	Valid Data on D31-D24, D23-D16, D15-D8, or D7-D0
All	x	High	High	Insert Wait States in Current Bus Cycle

RESET ($\overline{RESET}$)

This active-low input signal causes the MC68882 to initialize the floating-point data registers to non-signaling not-a-numbers (NANs) and clears the floating-point control, status, and instruction address registers.

When performing a power-up reset, external circuitry should keep the $\overline{RESET}$ line asserted for a minimum of four clock cycles after V_{CC} is within tolerance. This assures correct initialization of the MC68882 when power is applied. For compatibility with all M68000 Family devices, 100 milliseconds should be used as the minimum.

When performing a reset of the MC68882 after V_{CC} has been within tolerance for more than the initial power-up time, the $\overline{RESET}$ line must have an asserted pulse width which is greater than two clock cycles. For compatibility with all M68000 Family devices, 10 clock cycles should be used as the minimum.

CLOCK (CLK)

The MC68882 clock input is a TTL-compatible signal that is internally buffered for development of the internal clock signals. The clock input should be a constant frequency square wave with no stretching or shaping techniques required. The clock should not be gated off at any time and must conform to minimum and maximum period and pulse width times.

SENSE DEVICE ($\overline{SENSE}$)

This pin may be used optionally as an additional GND pin or as as indicator to external hardware that the MC68882 is present in the system. This signal is internally connected to the GND of the die, but it is not necessary to connect it to the external ground for correct device operation. If a pullup resistor (which should be larger than 10 kohm) is connected to this pin location, external hardware may sense the presence of the MC68882 in a system.

POWER (V_{CC} and GND)

These pins provide the supply voltage and system reference level for the internal circuitry of the MC68882. Care should be taken to reduce the noise level on these pins with apropriate capacitive decoupling.

NO CONNECT (NC)

One pin of the MC68882 package is designated as a no connect (NC). This pin position is reserved for future use by Motorola, and should not be used for signal routing or connected to V_{CC} or GND.

SIGNAL SUMMARY

Table 6 provides a summary of all the MC68882 signals described in the above paragraphs.

Table 6. Signal Summary

Signal Name	Mnemonic	Input/Output	Active State	Three State
Address Bus	A0-A4	Input	High	—
Data Bus	D0-D13	Input/Output	High	Yes
Size	$\overline{SIZE}$	Input	Low	—
Address Strobe	$\overline{AS}$	Input	Low	—
Chip Select	$\overline{CS}$	Input	Low	—
Read/Write	R/$\overline{W}$	Input	High/Low	—
Data Strobe	$\overline{DS}$	Input	Low	—
Data Transfer and Size Acknowledge	$\overline{DSACK0}$, $\overline{DSACK1}$	Output	Low	Yes
Reset	$\overline{RESET}$	Input	Low	—
Clock	CLK	Input	—	—
Sense Device	$\overline{SENSE}$	Input/Output	Low	No
Power Input	V_{CC}	Input	—	—
Ground	GND	Input	—	—

INTERFACING METHODS

MC68882/MC68010 OR MC68030 INTERFACING

The following paragraphs describe how to connect the MC68882 to an MC68020 or MC68030 for coprocessor operation via an 8-, 16-, or 32-bit data bus.

32-Bit Data Bus Coprocessor Connection

Figure 11 illustrates the coprocessor interface connection of an MC68882 to an MC68020/MC68030 via a 32-bit data bus. The MC68882 is configured to operate over a 32-bit data bus when both the A0 and $\overline{SIZE}$ pins are connected to V_{CC}.

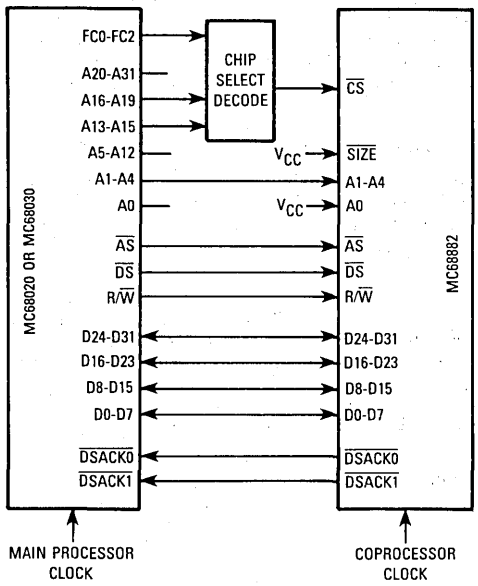


Figure 11. 32-Bit Data Bus Coprocessor Connection

16-Bit Data Bus Coprocessor Connection

Figure 12 illustrates the coprocessor interface connection of an MC68882 to an MC68020/MC68030 via a 16-bit data bus. The MC68882 is configured to operate over a 16-bit 5 data bus when the $\overline{SIZE}$ pin is connected to V_{CC}, and the A0 pin is connected to GND. The sixteen least-significant data pins (D0-D15) must be connected to the sixteen most-significant data pins (D16-D31) when the MC68882 is configured to operate over a 16-bit data bus (i.e., connect D0 to D16, D1 to D17, ... and D15 to D31). The DSACK pins of the two devices are directly connected, although it is not necessary to connect the $\overline{DSACK0}$ pin since the MC68882 never asserts it in this configuration.

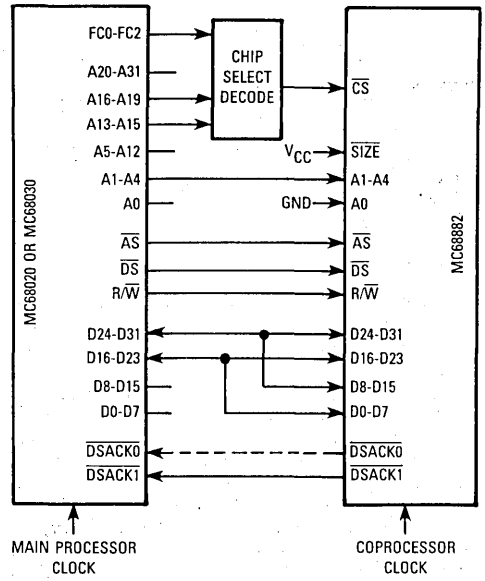


Figure 12. 16-Bit Data Bus Coprocessor Connection

8-Bit Data Bus Coprocessor Connection

Figure 13 illustrates the connect of an MC68882 to an MC68020/MC68030 as a coprocessor over an 8-bit data bus. The MC68882 is configured to operate over an 8-bit data bus when the $\overline{SIZE}$ pin is connected to GND. The twenty-four least-significant data pins (D0-D23) must be connected to the eight most-significant data pins (D24-D31) when the MC68882 is configured to operate over an 8-bit data bus (i.e., connect D0 to D8, D16 and D24; D1 to D9, D17, and D25; ... and D7 to D15, D23 and D31). The DSACK pins of the two devices are directly connected, although it is not necessary to connect the $\overline{DSACK1}$ pin since the MC68882 never asserts it in this configuration.

MC68882-MC68000/MC68008/MC68010 INTERFACING

The following paragraphs describe how to connect the MC68882 to an MC68000, MC68008, or MC68010 processor for opertion as a peripheral via an 8- or 16-bit data bus.

16-Bit Data Bus Peripheral Processor Connection

Figure 14 illustrates the connection of an MC68882 to an MC68000 or MC68010 as a peripheral processor over a 16-bit data bus. The MC68882 is configured to operate over a 16-bit data bus when the $\overline{SIZE}$ pin is connected to V_{CC}, and the A0 pin is connected to GND. The sixteen least-significant data pins (D0-D15) must be connected to the sixteen most-significant data pins (D16-D31) when the MC68882 is configured to operate over a 16-bit data bus (i.e., connect D0 to D16, D1 to D17, ... and D15 to D31).

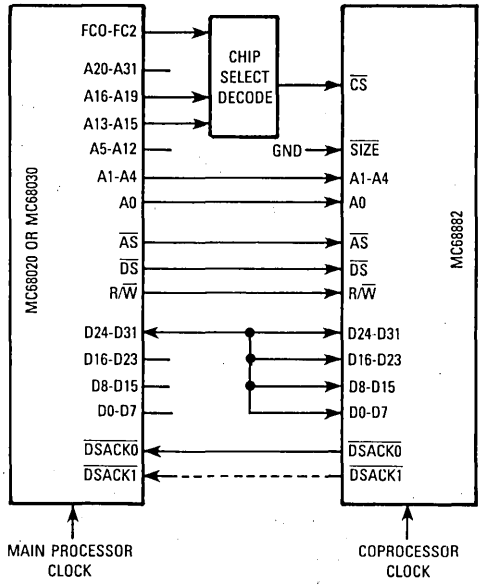


Figure 13. 8-Bit Data Bus Coprocessor Connection

The DSACK1 pin of the MC68882 is connected to the DTACK pin of the main processor, and the DSACK0 pin is not used.

When connected as a peripheral processor, the MC68882 chip select (CS) decode is system dependent. If the MC68000 is used as the main processor, the MC68882 CS must be decoded in the supervisor or user data spaces. However, if the MC68010 is used for the main processor, the MOVES instruction may be used to emulate any CPU space access that the MC68020/MC68030 generates for coprocessor communications. Thus, the CS decode logic for such systems may be the same as in an MC68020/MC68030 system, such that the MC68882 will not use any part of the data address spaces.

8-Bit Data Bus Peripheral Processor Connection

Figure 15 illustrates the connection of an MC68882 to an MC68008 as a peripheral processor over an 8-bit data bus. The MC68882 is configured to operate over an 8-bit data bus when the SIZE pin is connected to GND. The eight least-significant data pins (D0-D7) must be connected to the twenty-four most-significant pins (D8-D31) when the MC68882 is configured to operate over an 8-bit data bus (i.e., connect D0 to D8, D16, and D24; D1 to D9, D17, and D25; . . . and D7 to D15, D23, and D31). The DSACK0 pin of the MC68882 is connected to the DTACK pin of the MC68008, and the DSACK1 pin is not used.

When connected as a peripheral processor, the MC68882 chip select (CS) decode is system dependent, and the CS must be decoded in the supervisor or user data spaces.

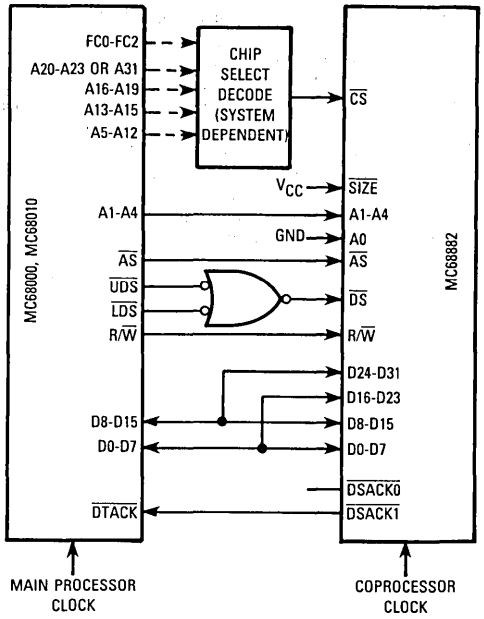


Figure 14. 16-Bit Data Bus Peripheral Processor Connection

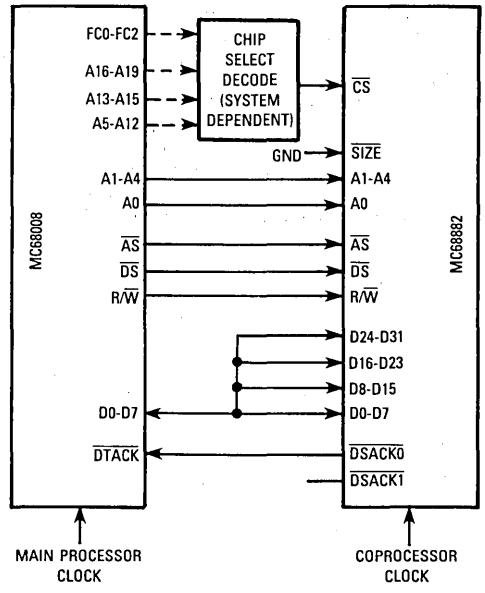


Figure 15. 8-Bit Data Bus Peripheral Processor Connection

ELECTRICAL SPECIFICATIONS

MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Supply Voltage	V_{CC}	−0.3 to +7.0	V
Input Voltage	V_{in}	−0.3 to +7.0	V
Operating Temperature	T_A	0 to 70	°C
Storage Temperature	T_{stg}	−55 to +150	°C

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

THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Rating
Thermal Resistance — Ceramic			°C/W
Junction to Ambient	θ_{JA}	33	
Junction to Case	θ_{JC}	15	

4

POWER CONSIDERATIONS

The average chip-junction temperature, T_J, in °C can be obtained from:

$$T_J = T_A + (P_D \cdot \theta_{JA}) \quad (1)$$

where:

T_A = Ambient Temperature, °C
θ_{JA} = Package Thermal Resistance, Junction-to-Ambient, °C/W
$P_D = P_{INT} + P_{I/O}$
$P_{INT} = I_{CC} \times V_{CC}$, 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.

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

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

Solving equations (1) and (2) for K gives:

$$K = P_D \cdot (T_A + 273°C) + \theta_{JA} \cdot P_D^2 \quad (3)$$

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.

The total thermal resistance of a package (θ_{JA}) can be separated into two components, θ_{JC} and θ_{CA}, representing the barrier to heat flow from the semiconductor junction to the package (case) surface (θ_{JC}) and from the case to the outside ambient (θ_{CA}). These terms are related by the equation:

$$\theta_{JA} = \theta_{JC} + \theta_{CA} \quad (4)$$

θ_{JC} is device related and cannot be influenced by the user. However, θ_{CA} is user dependent and can be minimized by such thermal management techniques as heat sinks, ambient air cooling, and thermal convention. Thus, good thermal management on the part of the user can significantly reduce θ_{CA} so that θ_{JA} approximately equals θ_{JC}. Substitution of θ_{JC} for θ_{JA} in equation (1) will result in a lower semiconductor junction temperature.

Values for thermal resistance presented in this document, unless estimated, were derived using the procedure described in Motorola Reliability Report 7843, "Thermal Resistance Measurement Method for MC68XX Microcomponent Devices," and are provided for design purposes only. Thermal measurements are complex and dependent on procedure and setup. User derived values for thermal resistance may differ.

DC ELECTRICAL CHARACTERISTICS (V_{CC} = 5.0 Vdc ± 5%; GND = 0 Vdc; T_A = 0°C to 70°C) (see Figure 16)

Characteristic		Symbol	Min	Max	Unit
Input High Voltage		V_{IH}	2.0	V_{CC}	V
Input Low Voltage		V_{IL}	GND −0.5	0.8	V
Input Leakage Current @ 5.25 V	CLK, $\overline{RESET}$, R/$\overline{W}$, A0-A4, $\overline{CS}$, $\overline{DS}$, $\overline{AS}$, SIZE	I_{in}	—	10	μA
Hi-Z (Off State) Input Current @ 2.4 V/0.4 V	$\overline{DSACK0}$, $\overline{DSACK1}$, D0-D31	I_{TSI}	—	20	μA
Output High Voltage (I_{OH} = −400 μA)	$\overline{DSACK0}$, $\overline{DSACK1}$, D0-D31	V_{OH}	2.4	—	V
Output Low Voltage (I_{OL} = 5.3 mA)	$\overline{DSACK0}$, $\overline{DSACK1}$, D0-D31	V_{OL}	—	0.5	V
Output Low Current (V_{OL} = GND)	$\overline{SENSE}$	I_{OL}	—	500	μA
Power Dissipation		P_D	—	0.75	W
Capacitance* (V_{in} = 0, T_A = 25°C, f = 1 MHz)		C_{in}	—	20	pF
Output Load Capacitance		C_L	—	130	pF

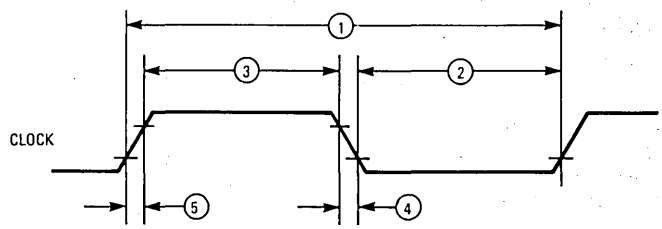
*Capacitance is periodically sampled rather than 100% tested.

4

AC ELECTRICAL CHARACTERISTICS — CLOCK INPUT

(V_{CC} = 5.0 Vdc ± 5%; GND = 0 Vdc; T_A = 0 to 70°C; refer to Figure 16)

Num	Characteristic	16.67 MHz		20 MHz		25 MHz		Unit
		Min	Max	Min	Max	Min	Max	
	Frequency of Operation	8	16.67	12.5	20	12.5	25	MHz
1	Cycle Time	60	125	50	80	40	80	ns
2,3	Clock Pulse Width	24	95	20	54	15	59	ns
4,5	Rise and Fall Times	—	5	—	5	—	4	ns



NOTE:

1. Timing measurements are referenced to and from a low voltage of 0.8 volts and a high voltage of 2.0 volts, unless otherwise noted. The voltage swing through this range should start outside, and pass through, the range such that the rise or fall will be linear between 0.8 volts and 2.0 volts.

Figure 16. Clock Input Timing Diagram

AC ELECTRICAL SPECIFICATIONS — READ AND WRITE CYCLES

(V_{CC} = 5.0 Vdc ± 5%; GND = 0 Vdc; T_A = 0 to 70°C; refer to Figures 17, 18, and 19)

Num	Characteristic	16.67 HMz		20 MHz		25 MHz		Unit
		Min	Max	Min	Max	Min	Max	
6	Address Valid to $\overline{AS}$ Asserted (see Note 5)	15	—	10	—	5	—	ns
6A	Address Valid to $\overline{DS}$ Asserted (Read) (see Note 5)	15	—	10	—	5	—	ns
6B	Address Valid to $\overline{DS}$ Asserted (Write) (see Note 5)	50	—	50	—	35	—	ns
7	$\overline{AS}$ Negated to Address Invalid (see Note 6)	10	—	10	—	5	—	ns
7A	$\overline{DS}$ Negated to Address Invalid (see Note 6)	10	—	10	—	5	—	ns
8	$\overline{CS}$ Negated to $\overline{AS}$ Asserted (see Note 9)	0	—	0	—	0	—	ns
8A	$\overline{CS}$ Negated to $\overline{DS}$ Asserted (Read) (see Note 9)	0	—	0	—	0	—	ns
8B	$\overline{CS}$ Asserted to $\overline{DS}$ Asserted (Write)	30	—	25	—	20	—	ns
9	$\overline{AS}$ Negated to $\overline{CS}$ Negated	10	—	10	—	5	—	ns
9A	$\overline{DS}$ Negated to $\overline{CS}$ Negated	10	—	10	—	5	—	ns
10	R/$\overline{W}$ High to $\overline{AS}$ Asserted (Read)	15	—	10	—	5	—	ns
10A	R/$\overline{W}$ High to $\overline{DS}$ Asserted (Read)	15	—	10	—	5	—	ns
10B	R/$\overline{W}$ Low to $\overline{DS}$ Asserted (Write)	35	—	30	—	25	—	ns
11	$\overline{AS}$ Negated to R/$\overline{W}$ Low (Read) or $\overline{AS}$ Negated to R/$\overline{W}$ High (Write)	10	—	10	—	5	—	ns
11A	$\overline{DS}$ Negated to R/$\overline{W}$ Low (Read) or $\overline{DS}$ Negated to R/$\overline{W}$ High (Write)	10	—	10	—	5	—	ns
12	$\overline{DS}$ Width Asserted (Write)	40	—	38	—	30	—	ns
13	$\overline{DS}$ Width Negated	40	—	38	—	30	—	ns
13A	$\overline{DS}$ Negated to $\overline{AS}$ Asserted (see Note 4)	30	—	30	—	25	—	ns
14	$\overline{CS}$, $\overline{DS}$ Asserted to Data-Out Valid) (Read) (see Note 2)	—	80	—	45	—	45	ns
15	$\overline{DS}$ Negated to Data-Out Invalid (Read)	0	—	0	—	0	—	ns
16	$\overline{DS}$ Negated to Data-Out High Impedance (Read)	—	50	—	35	—	35	ns
17	Data-In Valid to $\overline{DS}$ Asserted (Write)	15	—	10	—	5	—	ns
18	$\overline{DS}$ Negated to Data-In Invalid (Write)	15	—	10	—	5	—	ns
19	$\overline{START}$ True to $\overline{DSACK0}$ and $\overline{DSACK1}$ Asserted (see Notes 2 and 10)	—	50	—	35	—	25	ns
19A	$\overline{DSACK0}$ Asserted to $\overline{DSACK1}$ Asserted (Skew) (see Note 7)	−15	15	−10	10	−10	10	ns
20	$\overline{DSACK0}$ or $\overline{DSACK1}$ Asserted to Data-Out Valid	—	50	—	43	—	32	ns
21	$\overline{START}$ False to $\overline{DSACK0}$ and $\overline{DSACK1}$ Negated (see Note 8)	—	50	—	40	—	40	ns
22	$\overline{START}$ False to $\overline{DSACK0}$ and $\overline{DSACK1}$ High Impedance (see Note 8)	—	70	—	55	—	55	ns
23	$\overline{START}$ True to Clock High (Synchronous Read) (see Notes 3 and 8)	0	—	0	—	0	—	ns
24	Clock Low to Data-Out Valid (Synchronous Read) (see Note 3)	—	105	—	80	—	60	ns
25	$\overline{START}$ True to Data-Out Valid (Synchronous Read) (see Notes 3 and 8)	— 1.5	105 + 2.5	— 1.5	80 + 2.5	— 1.5	60 + 2.5	ns Clks
26	Clock Low to $\overline{DSACK0}$ and $\overline{DSACK1}$ Asserted (Synchronous Read) (see Note 3)	—	75	—	55	—	45	ns
27	$\overline{START}$ True to $\overline{DSACK0}$ and $\overline{DSACK1}$ Asserted (Synchronous Read) (see Notes 3 and 8)	— 1.5	75 + 2.5	— 1.5	55 + 2.5	— 1.5	45 + 2.5	ns Clks

NOTES:
1. Timing measurements are referenced to and from a low voltage of 0.8 volts and a high voltage of 20 volts, unless otherwise noted. The voltage swing through this range should start outside, and pass through, the range such that the rise or fall will be linear between 0.8 volts and 2.0 volts.
2. These specifications only apply if the MC68882 has completed all internal operations initiated by the termination of the previous bus cycle when $\overline{DS}$ was negated.
3. Synchronous read cycles occur ***only*** when the save or response CIR locations are read.

NOTES (Continued)

4. This specification only applies to systems in which back-to-back accesses (read-write or write-write) of the operand CIR can occur. When the MC68882 is used as a coprocessor to the MC68020/MC68030, this can occur when the addressing mode is Immediate.
5. If the $\overline{SIZE}$ pin is ***not*** strapped to either V_{CC} or GND, it must have the same setup times as do addresses.
6. If the $\overline{SIZE}$ pin is ***not*** strapped to either V_{CC} or GND, it must have the same hold times as do addresses.
7. This number is reduced to 5 nanoseconds if $\overline{DSACK0}$ and $\overline{DSACK1}$ have equal loads.
8. START is not an external signal; rather, it is the logical condition that indicates the start of an access. The logical equation for this condition is $\overline{START} = \overline{CS} + \overline{AS} + R/\overline{W} \cdot \overline{DS}$.
9. If a subsequent access is not a FPCP access, $\overline{CS}$ must be negated before the assertion of $\overline{AS}$ and/or $\overline{DS}$ on the non-FPCP access. These specifications replace the old specifications 8 and 8A (the old specifications implied that in all cases, transitions in $\overline{CS}$ must not occur simultaneously with transitions of $\overline{AS}$ or $\overline{AS}$. This is not a requirement of the MC68882).

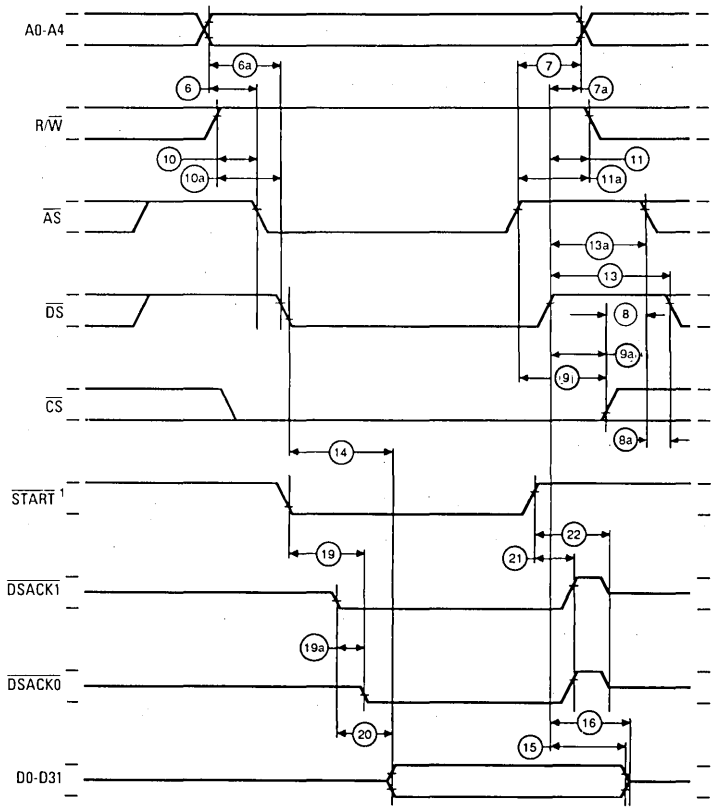


Figure 17. Asynchronous Read-Cycle Timing Diagram

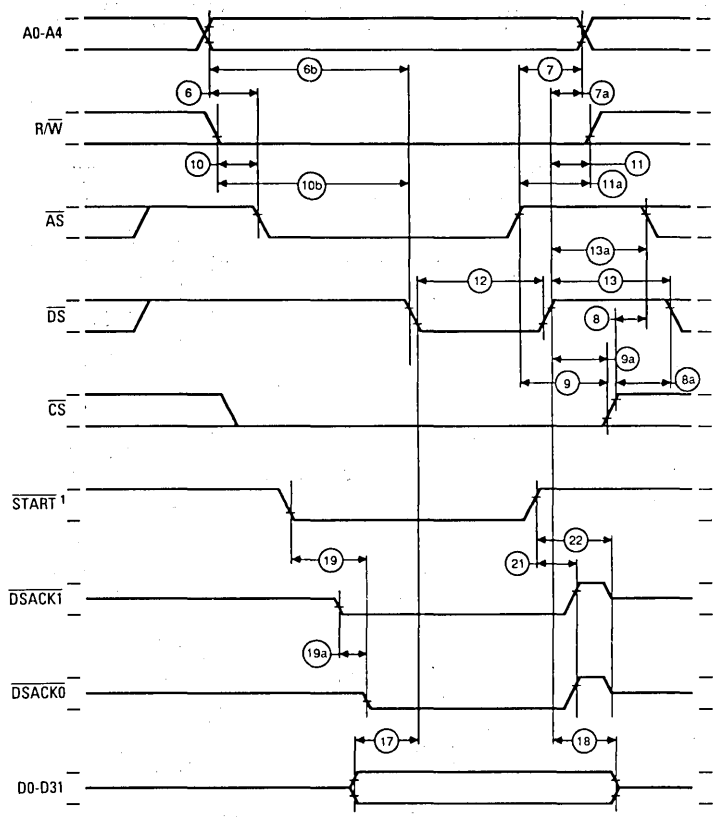


Figure 18. Asynchronous Write-Cycle Timing Diagram

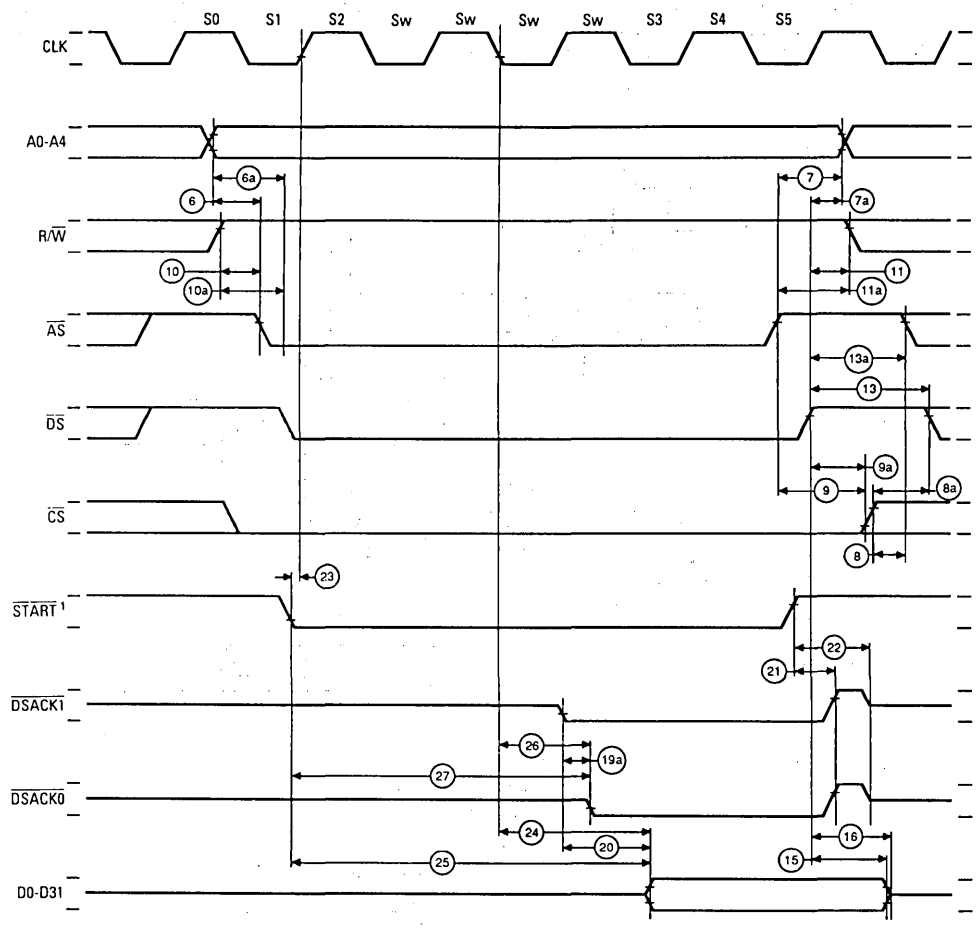


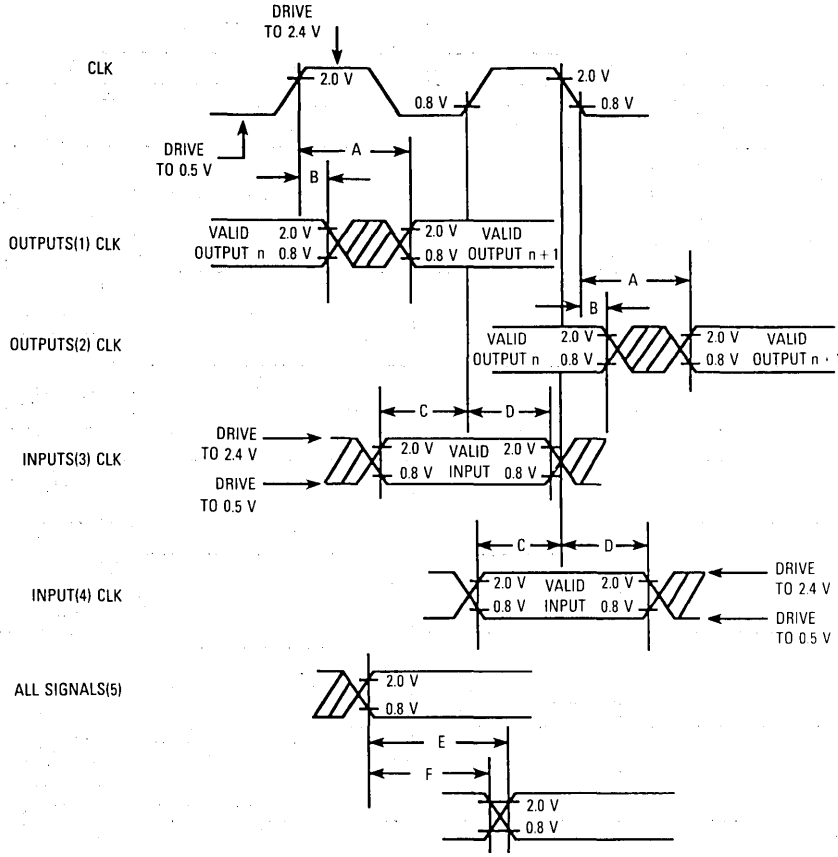
Figure 19. Synchronous Read-Cycle Timing Diagram

AC ELECTRICAL SPECIFICATION DEFINITIONS

The AC specifications presented consist of output delays, input setup and hold times, and signal skew times. All signals are specified relative to an appropriate edge of the clock input and, possibly, relative to one or more other signals.

The measurement of the AC specifications is defined by the waveforms shown in Figure 20. In order to test the parameters guaranteed by Motorola, inputs must be driven to the voltage levels specified in Figure 20. Outputs are specified with minimum and/or maximum limits, as appropriate, and are measured as shown. Inputs are specified with minimum and, as appropriate, maximum setup and hold times, and are measured as shown. Finally, the measurement for signal-to-signal specifications are also shown.

Note that the testing levels used to verify conformance to the AC specifications does not affect the guaranteed DC operation of the device as specified in the DC electrical characteristics.



NOTES:

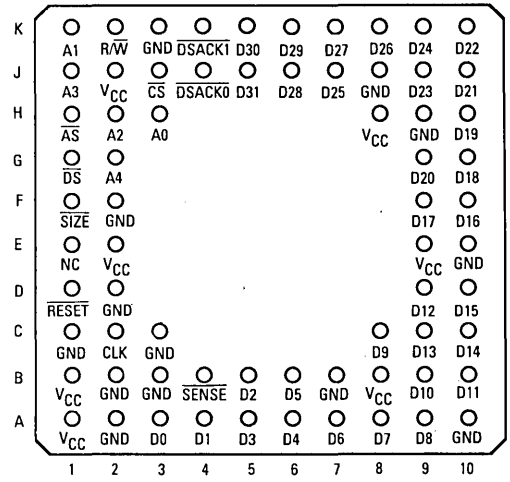
1. This output timing is applicable to all parameters specified relative to the rising edge of the clock.
2. This output timing is applicable to all parameters specified relative to the falling edge of the clock.
3. This input timing is applicable to all parameters specified relative to the rising edge of the clock.
4. This input timing is applicable to all parameters specified relative to the falling edge of the clock.
5. This timing is applicable to all parameters specified relative to the assertion/negation of another signal.

LEGEND:

A. Maximum output delay specification.
B. Minimum output hold time.
C. Minimum input setup time specification.
D. Minimum input hold time specification.
E. Signal valid to signal valid specification (maximum or minimum).
F. Signal valid to signal invalid specification (maximum or minimum).

Figure 20. Drive Levels and Test Points for AC Specifications

PIN ASSIGNMENTS



DMA CONTROLLERS 5

5

MC68440
MC68442

Technical Summary

Dual-Channel Direct Memory Access Controllers

M68000 microprocessors utilize state-of-the-art MOS technology to maximize performance and throughput. The MC68440 dual-channel direct memory access controller (DDMA) is designed to complement the performance and architectural capabilities of M68000 Family microprocessors by moving blocks of data in a quick, efficient manner with minimum intervention from a processor.

The MC68442 extended dual-channel direct memory access controller (EDMA) implements an expanded addressing range over that of the MC68440 with the addition of eight address lines and one function code line (A24-A31 and FC3). This allows the EDMA to linearly access four giagbytes in any of 16 address spaces.

NOTE

Information contained in this document applies to both the DDMA and the EDMA. Reference is primarily given to the DDMA where the descriptions are identical. When applicable, difference data for the EDMA will be highlighted.

The DDMA and EDMA perform memory-to-memory, peripheral-to-memory, and memory-to-peripheral data transfers by utilizing the following features:

- Two Independent DMA Channels with Programmable Priority
- Asynchronous M68000 Bus Structure with:
  - DDMA — 24-Bit Address and a 16-Bit Data Bus
  - EDMA — 32-Bit Address and a 16-Bit Data Bus
- Fast Transfer Rates: Up to 5 Megabytes per Second at 10 MHz, No Wait States
- Fully Supports all M68000 Bus Options such as Halt, Bus Error, and Retry
- FIFO Locked Step Support with Device Transfer Complete Signal
- Can Operate on an 8-Bit Data Bus with the MC68008
- Flexible Request Generation:
  - Internal, Maximum Rate
  - Internal, Limited Rate
  - External, Cycle Steal
  - External, Burst
- Programmable 8-Bit or 16-Bit Peripheral Device Types:
  - Explicitly Addressed, M68000 Type
  - Implicitly Addressed:
    - Device with Request and Acknowledge
    - Device with Request, Acknowledge, and Ready
- Non-Contiguous Block Transfer Operation (Continue Mode)
- Block Transfer Restart Operation (Reload Mode)
- Pin and Register Compatible Functional Subset of the MC68450 DMAC

INTRODUCTION

The main purpose of a direct memory access (DMA) controller in any system is to transfer data at very high rates, usually much faster than a microprocessor under software control can handle. The term DMA is used to refer to the ability of a peripheral device to access memory in a system in the same manner as a microprocessor does. DMA operations can occur concurrently with other operations that the system processor needs to perform, thus greatly boosting overall system performance. Figure 1 illustrates a typical system configuration using a DMA interface to a high speed disk storage device. In a system such as this, the DDMA will move blocks of data between the disk and memory at rates approaching the limits of the memory bus since the simple function of data movement is implemented in high speed MOS hardware. A block of data consists of a sequence of byte or word operands starting at a specific address in memory with the length of the block determined by a transfer count as shown in Figure 2. A single channel operation may involve the transfer of several blocks of data between the memory and a device.

Any operation involving the DDMA will follow the same basic steps: channel initialization by the system processor, data transfer, and block termination. In the initialization phase, the host processor loads the registers of the DDMA with control information, address pointers, and transfer counts and then starts the channel. During the transfer phase, the DDMA accepts requests for operand transfers and provides addressing and bus control for the transfers. The termination phase occurs after the operation is complete, when the DDMA indicates the status of the operation in a status register. During all phases of a data transfer operation, the DDMA will be in one of three operating modes:

IDLE This is the state that the DDMA assumes when it is reset by an external device and waiting for initialization by the system processor or an operand transfer request from a peripheral.

MPU This is the state that the DDMA enters when it is chip selected by another bus master in the system (usually the main system processor). In this mode, the DDMA internal registers are written or read, to control channel operation or check the status of a block transfer.

DMA This is the state that the DDMA enters when it is acting as a bus master to perform an operand transfer.

TRANSFER MODES

The DDMA can perform implicit address or explicit address data transfers. Implicitly addressed devices do not require the generation of a device data register address

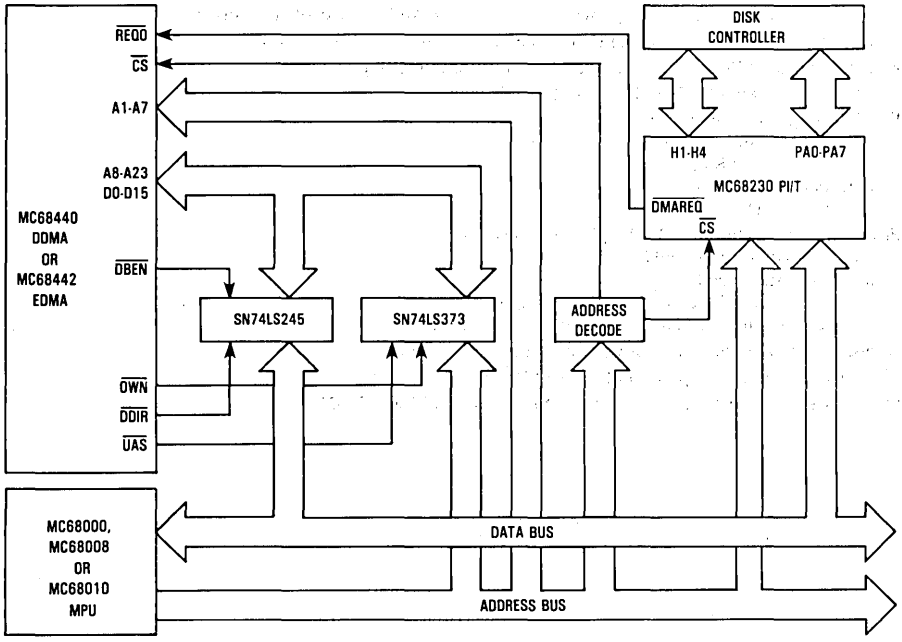


Figure 1. Typical System Configuration

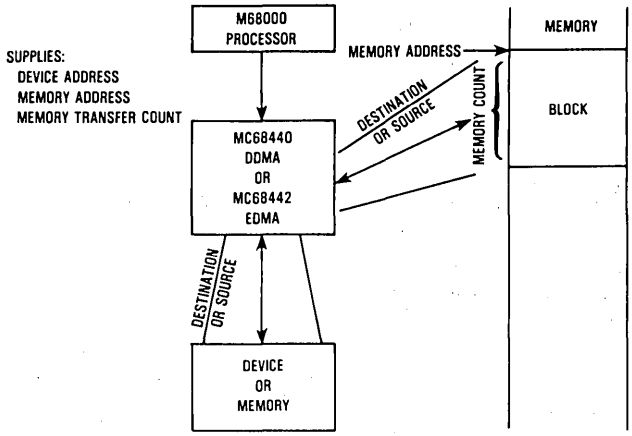


Figure 2. Data Block Format

for a data transfer. Such a device is controlled by a five signal device control interface on the DDMA during implicit address transfers as shown in Figure 3. Since only memory is addressed during such a data transfer, this method is called the single-address method.

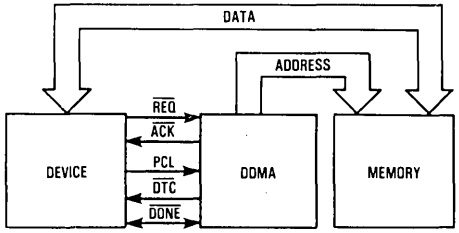


Figure 3. Implicitly Addressed Device Interface

Explicitly addressed devices require that a data register within the peripheral device be addressed. No signals other than the M68000 asynchronous bus control signals are needed to interface with such a device, although any of the five device control signals may also be used. Because the address bus is used to access the peripheral, the data cannot be directly transferred to/from memory since memory also requires addressing. Therefore, data is transferred from the source to an internal holding register in the DDMA and then transferred to the destination during a second bus transfer as shown in Figure 4. Since both memory and the device are addressed during such a data transfer, this method is called the dual-address method.

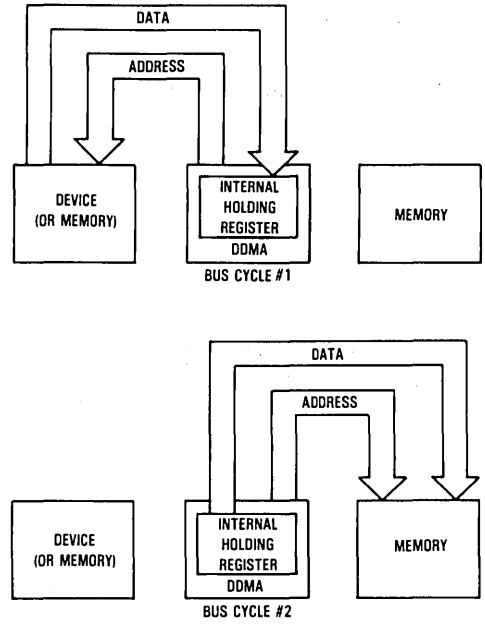


Figure 4. Dual-Address Transfer Sequence

REQUEST MODES

Requests may be externally generated by a device or internally generated by the auto-request mechanism of the DDMA. Auto-requests may be generated either at the maximum rate, where the channel always has a request pending, or at a limited rate determined by selecting a portion of the bus bandwidth to be available for DMA activity. External requests can be either burst requests or cycle steal requests that are generated by the request signal associated with each channel.

REGISTERS

The DDMA contains 17 on-chip registers for each of the two channels plus one general control register, all of which are under complete software control. The user programmer's model of the registers is shown in Figure 5.

The DDMA registers contain information about the data transfer such as the source and destination address and function codes, transfer count, operand size, device port size, channel priority, continuation address and transfer count, and the function of the peripheral control line. One register also provides status and error information on channel activity, peripheral inputs, and various events which may have occurred during a DMA transfer. A general control register selects the bus utilization factor to be used in limited rate auto-request DMA operations.

SIGNAL DESCRIPTION

The following paragraphs contain a brief description of the input and output signals. Included at the end of the functional description of the signals is a table describing the electrical characteristics of each pin (i.e., the type of driver used).

5

NOTE

The terms **assertion** and **negation** will be used extensively. This is done to avoid confusion when dealing with a mixture of "active-low" and "active-high" signals. The term assert or assertion is used to indicate that a signal is active or true, independent of whether that level is represented by a high or low voltage. The term negate or negation is used to indicate that a signal is inactive or false.

SIGNAL ORGANIZATION

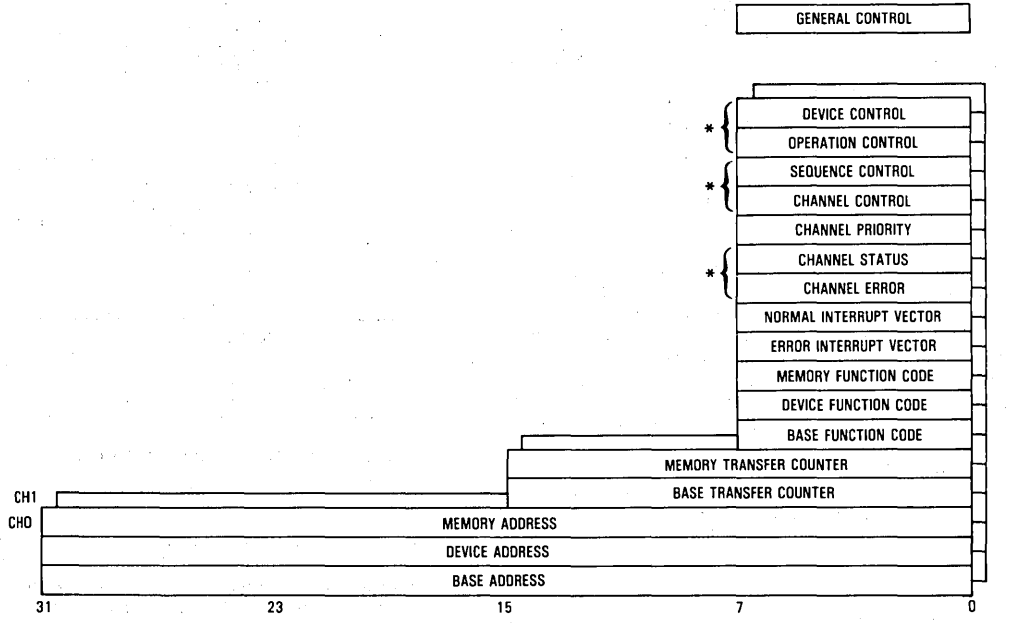
The input and output signals can be functionally organized into the groups shown in Figures 6 and 7. The function of each signal or group of signals is discussed in the following paragraphs.

Address/Data Bus (A8/D0 through A23/D15)

This 16-bit bus is time multiplexed to provide address outputs during the DMA mode of operation and is used as a bidirectional data bus to input data from an external device (during an MPU write or DMA read) or to output data to an external device (during an MPU read or a DMA write). This is a three-state bus and is demultiplexed using external latches and buffers controlled by the multiplex control lines.

Lower Address Bus (A1 through A7)

These bidirectional three-state lines are used to address the DDMA internal registers in the MPU mode and



*Word aligned register pairs.

Figure 5. DDMA Programmer's Model

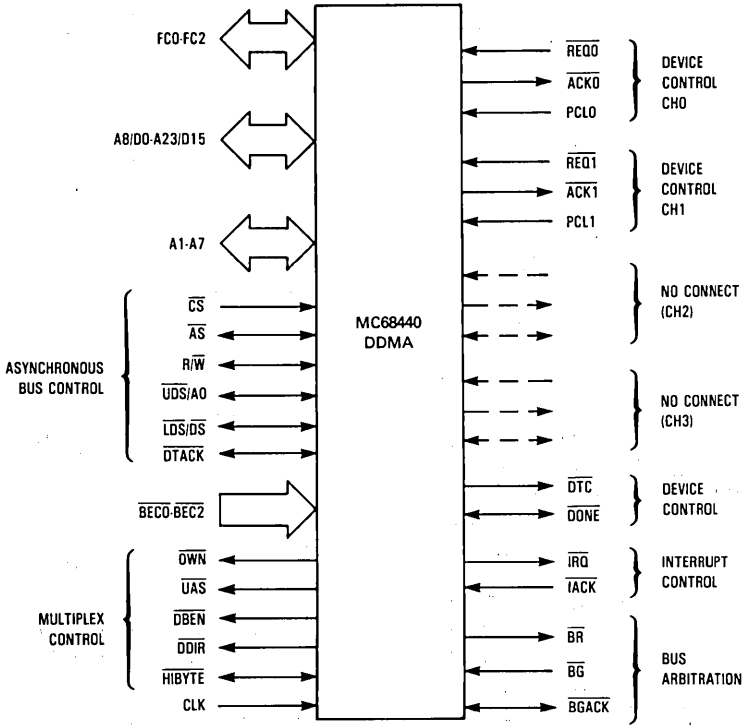


Figure 6. MC68440 Signal Organization

to provide the lower seven address outputs in the DMA mode.

EDMA Upper Address Bus (A24 through A31)

The EDMA implements an expanded addressing range over that of the DDMA with the addition of eight address lines. This allows the EDMA to linearly access four gigabytes in any of 16 address spaces. A24-A31 are non-multiplexed, with timing characteristics identical to address lines A1-A7.

Function Codes (DDMA — FC0 through FC2, EDMA — FC0 through FC3)

These three-state output lines are used in the DMA mode to further qualify the value on the address bus to provide eight separate address spaces that may be defined by the user. The value placed on these lines is taken from one of the internal function code registers, depending on the source register for the address used during a DMA bus cycle.

Asynchronous Bus Control

Asynchronous data transfers are handled using the following control signals: chip select, address strobe, read/write, upper and lower data strobes (or A0 and data strobe when using an 8-bit bus), and data transfer acknowledge. These signals are described in the following paragraphs.

CHIP SELECT ($\overline{CS}$). This input signal is used to select the DDMA for an MPU bus cycle. When $\overline{CS}$ is asserted, the address on A1-A7 and the data strobes (or A0 when using an 8-bit bus) select the internal DDMA register that will be involed in the transfer. $\overline{CS}$ should be generated by qualifying an address decode signal with the address and data strobes.

ADDRESS STROBE ($\overline{AS}$). This bidirectional signal is used as an output in the DMA mode to indicate that a valid address is present on the address bus. In the MPU or IDLE modes, it is used as an input to determine when the DDMA can take control of the bus (if the DDMA has requested and been granted use of the bus).

READ/WRITE (R/$\overline{W}$). This bidirectional signal is used to indicate the direction of a data transfer during a bus cycle. In the MPU mode, a high level indicates that a transfer is from the DDMA to the data bus and a low level indicates a transfer from the data bus to the DDMA. In the DMA mode, a high level indicates a transfer from the addressed memory or device to the data bus, and a low level indicates a transfer from the data bus to the addressed memory or device.

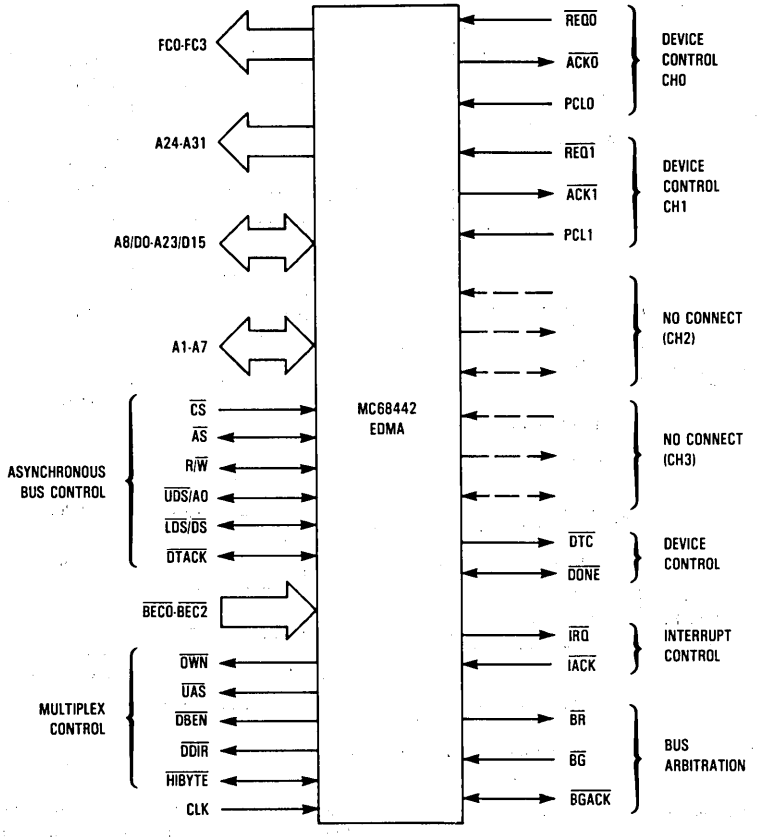


Figure 7. MC68442 Signal Organization

UPPER AND LOWER DATA STROBES (UDS/A0 AND LDS/DS). These bidirectional lines are used for different purposes depending on whether the DDMA is operating on an 8-bit or a 16-bit bus.

When using a 16-bit bus, these pins function as UDS and LDS. During any bus cycle, UDS is asserted if data is to be transferred over data lines D8-D15 and LDS is asserted if data is to be transferred over data lines D0-D7. UDS/LDS are asserted by the DDMA when operating in the DMA mode and by another bus master when in the MPU mode.

When using an 8-bit bus, these pins function as A0 and DS. A0 is an extension to the lower address lines to provide the address of a byte in the 16 megabyte address map and is valid when A1-A7 are valid. DS is used as a data strobe to enable external data buffers and to indicate that valid data is on the bus during a write cycle.

DATA TRANSFER ACKNOWLEDGE (DTACK). This bidirectional line is used to signal that an asynchronous bus cycle may be terminated. In the MPU mode, this output indicates that the DDMA has accepted data from the MPU or placed data on the bus for the MPU. In the DMA mode, this input is monitored by the DDMA to determine when to terminate a bus cycle. As long as DTACK remains negated, the DDMA will insert wait cycles into a bus cycle and when DTACK is asserted, the bus cycle will be terminated (except when PCL is used as a ready signal, in which case both signals must be asserted before the cycle is terminated).

BUS EXCEPTION CONTROL (BEC0 THROUGH BEC2). These input lines provide an encoded signal that indicates an abnormal bus condition such as a bus error or reset. Refer to **4.4.2 Bus Exception Control Functions** for a detailed description of the function of these pins.

Multiplex Control

These signals are used to control external multiplex/demultiplex devices to separate the address and data

information on the A8/D0-A23/D15 lines and to transfer data between the upper and lower halves of the data bus during certain DMA bus cycles.

Figure 8 shows the five external devices needed to demultiplex the address/data pins and the interconnection of the multiplex control signals. The SN74LS245 that may connect the upper and lower halves of the data bus is needed only if an 8-bit device is used to transfer data to or from a 16-bit system data bus during single address transfers. When the DDMA is used on an 8-bit system data bus with 8-bit devices, only the SN74LS245 buffer for D0-D7 is needed.

OWN ($\overline{OWN}$). This output indicates that the DDMA is controlling the bus. It is used as the enable signal to turn on the external address latch drivers and control signal buffers.

UPPER ADDRESS STROBE ($\overline{UAS}$). This output is used as the gate signal to the transparent latches that capture the value of A8-A23 on the multiplexed address/data bus.

DATA BUFFER ENABLE ($\overline{DBEN}$). This output is used as the enable signal to the external bidirectional data buffers.

DATA DIRECTION ($\overline{DDIR}$). This output controls the direction of the external bidirectional data buffers. If $\overline{DDIR}$ is high, the data transfer is from the DDMA to the data bus. If $\overline{DDIR}$ is low, the data transfer is from the data bus to the DDMA.

HIGH BYTE ($\overline{HIBYTE}$). This bidirectional signal determines the size of the bus used by the DDMA during a reset operation. If this signal is asserted (tied to ground) during reset, the data bus size is eight bits and $\overline{HIBYTE}$

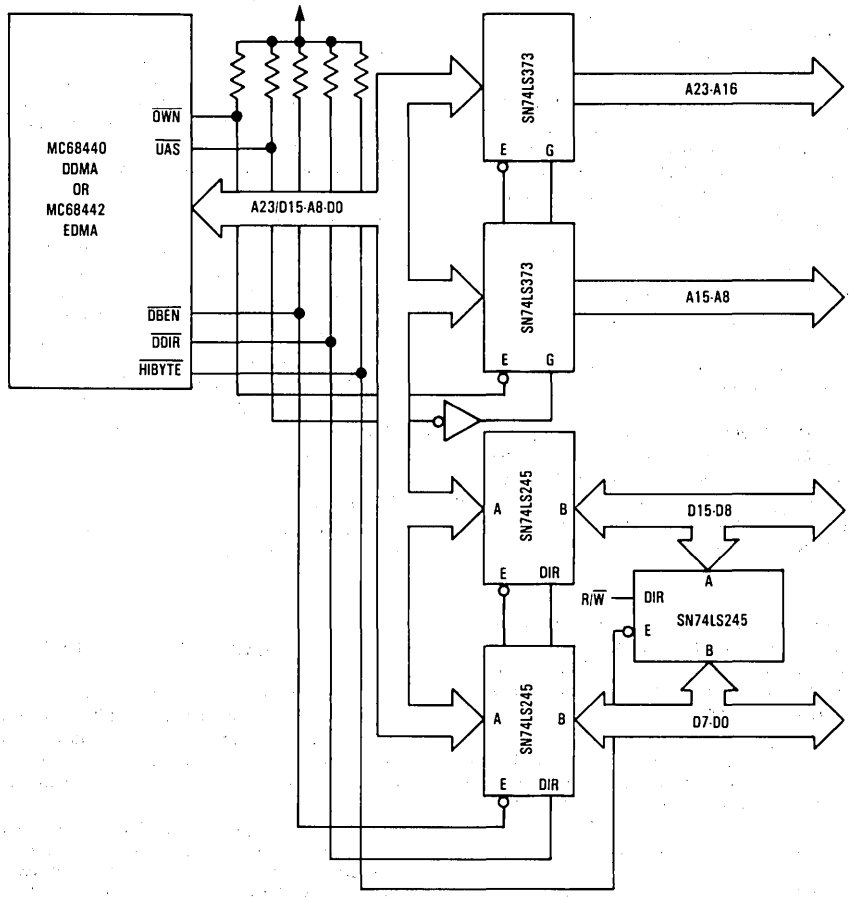


Figure 8. Demultiplex Logic

will not be used as an output. If it is negated (pulled high by a resistor) during reset, the data bus size is assumed to be 16 bits and HIBYTE will be used as an output during single-address DMA transfers between an 8-bit device and a 16-bit memory. As an output, HIBYTE indicates that data will be present on data lines D8-D15 that must be transferred to data lines D0-D7 or vice versa through an external buffer during a single address transfer between an 8-bit device and a 16-bit memory.

Bus Arbitration Control

These three signals form a bus arbitration circuit used to determine which device in a system will be the current bus master.

BUS REQUEST (BR). This output is asserted by the DDMA to request control of the bus.

BUS GRANT (BG). This input is asserted by an external bus arbiter to inform the DDMA that it may assume bus mastership as soon as the current bus cycle is completed. The DDMA will not take control of the bus until CS, IACK, AS, and BGACK are all negated.

5

BUS GRANT ACKNOWLEDGE (BGACK). This bidirectional signal is asserted by the DDMA to indicate that it is the current bus master. BGACK is monitored as an input to determine when the DDMA can become bus master and if a bus master other than the system MPU is a master during limited rate auto-request operation.

Interrupt Control

These two signals form an interrupt request/acknowledge handshake circuit with an MPU.

INTERRUPT REQUEST (IRQ). This output is asserted by the DDMA to request service from the MPU.

INTERRUPT ACKNOWLEDGE (IACK). This input is asserted by the MPU to acknowledge that it has received an interrupt from the DDMA. In response to the assertion of IACK, the DDMA will place a vector on D0-D7 that will be used by the MPU to fetch the address of the proper DDMA interrupt handler routine.

Device Control

These eight lines perform the interface between the DDMA and two peripheral devices. Two sets of three lines are dedicated to a single DDMA channel and its associated peripheral; the remaining two lines are global signals shared by both channels.

REQUEST (REQ0, REQ1). These inputs are asserted by a peripheral device to request an operand transfer between that peripheral device and memory. In the cycle steal request generation mode, these inputs are edge sensitive; in burst mode, they are level sensitive.

ACKNOWLEDGE (ACK0, ACK1). These outputs are asserted by the DDMA to signal to a peripheral that an operand is being transferred in response to a previous transfer request.

PERIPHERAL CONTROL LINE (PCL0, PCL1). These inputs are multi-purpose signals that may be programmed to function as ready, abort, reload, status, or interrupt inputs.

DATA TRANSFER COMPLETE (DTC). This output is asserted by the DDMA during any DDMA bus cycle to indicate that the data has been successfully transferred (i.e., the bus cycle was not terminated abnormally).

DONE (DONE). This bidirectional signal is asserted by the DDMA or a peripheral device during any DMA bus cycle to indicate that the data being transferred is the last item in a block. The DDMA will assert this signal during a bus cycle when the memory transfer count register is decremented to zero and the continue bit in the channel control register is not set.

Clock (CLK)

The clock input is a TTL-compatible signal that is internally buffered for development of the internal clocks needed by the DDMA. The clock input should not be gated off at any time and the clock signal must conform to minimum and maximum pulse width times.

No Connection (NC)

Six pins are not connected in order to maintain pin compatibility with the MC68450 DMAC; these pins are in the positions of the device control signals for channels two and three (REQ2, ACK2, PCL2, REQ3, ACK3, and PCL3) on the DMAC. It is suggested that these pins be left unconnected to allow future expansion; however, if a DDMA is placed into a socket designed to also accommodate a DMAC, the four input signals will be ignored and the two output signals will be allowed to float.

SIGNAL SUMMARY

Table 1 is a summary of all the signals discussed in the previous paragraphs.

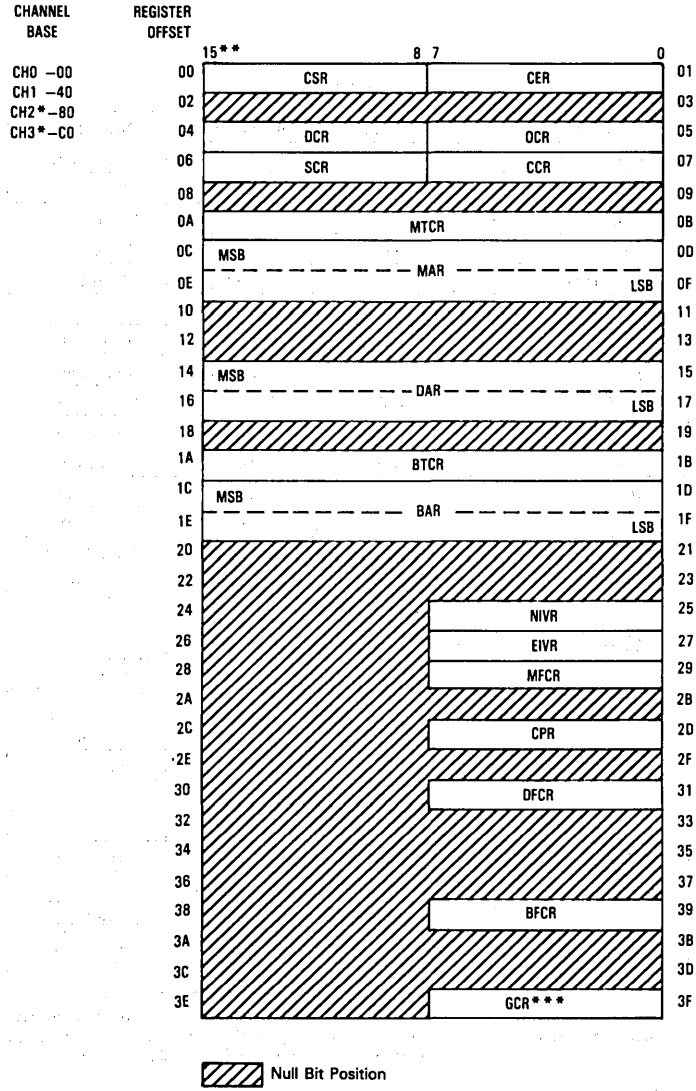
REGISTER DESCRIPTION

Figure 9 shows the memory mapped locations of the registers for each channel on a 16-bit bus. Figure 10 shows the register summary and may be used for a quick reference to the bit definitions within each register. The register locations defined as 'Null Register' may be read or written; however, a write access will not affect any DDMA channel operation and a read access will always return all ones. In the descriptions of each register, some bits are defined as 'not used', in which case writes to those bits will have no affect and they will always read as zeros.

Table 1. Signal Summary

Signal Name	Direction	Active State	Driver Type
A8/D0-A23/D15	In/Out	High	Three State
A1-A7	In/Out	High	Three State
A24-A31 — EDMA Only	In/Out	High	Three State
FC0-FC2	Out	High	Three State
FC3 — EDMA Only	Out	High	Three State
$\overline{CS}$	In	Low	—
$\overline{AS}$	In/Out	Low	Three State*
R/$\overline{W}$	In/Out	High/Low	Three State*
$\overline{UDS}$/A0	In/Out	Low/High	Three State*
$\overline{LDS}$/$\overline{DS}$	In/Out	Low	Three State*
$\overline{DTACK}$	In/Out	Low	Open Drain*
$\overline{OWN}$	Out	Low	Open Drain*
$\overline{UAS}$	Out	Low	Three State*
$\overline{DBEN}$	Out	Low	Three State*
$\overline{DDIR}$	Out	High/Low	Three State*
$\overline{HIBYTE}$	In/Out	Low	Three State*
$\overline{BEC0}$-$\overline{BEC2}$	In	Low	—
$\overline{BR}$	Out	Low	Open Drain
$\overline{BG}$	In	Low	—
$\overline{BGACK}$	In/Out	Low	Open Drain*
$\overline{IRQ}$	Out	Low	Open Drain
$\overline{IACK}$	In	Low	—
$\overline{REQ0}$, $\overline{REQ1}$	In	Low	—
$\overline{ACK0}$, $\overline{ACK1}$	Out	Low	Three State*
PCL0, PCL1	In	Programmed	—
$\overline{DTC}$	Out	Low	Three State*
$\overline{DONE}$	In/Out	Low	Open Drain
CLK	In	—	—

*These signals require a pull resistor to maintain a high voltage when in the high-impedance or negated state. However, when these signals go to the high-impedance or negated state, they will first drive the pin high momentarily to reduce the signal rise time.



*All accesses to channel 2 and 3 registers will be treated as null accesses.

**When operated on an 8-bit bus, all register data is transferred over D0-D7, the word and long word registers are then accessed as a contiguous set of bytes.

***The GCR is located at FF only.

Figure 9. Register Memory Map

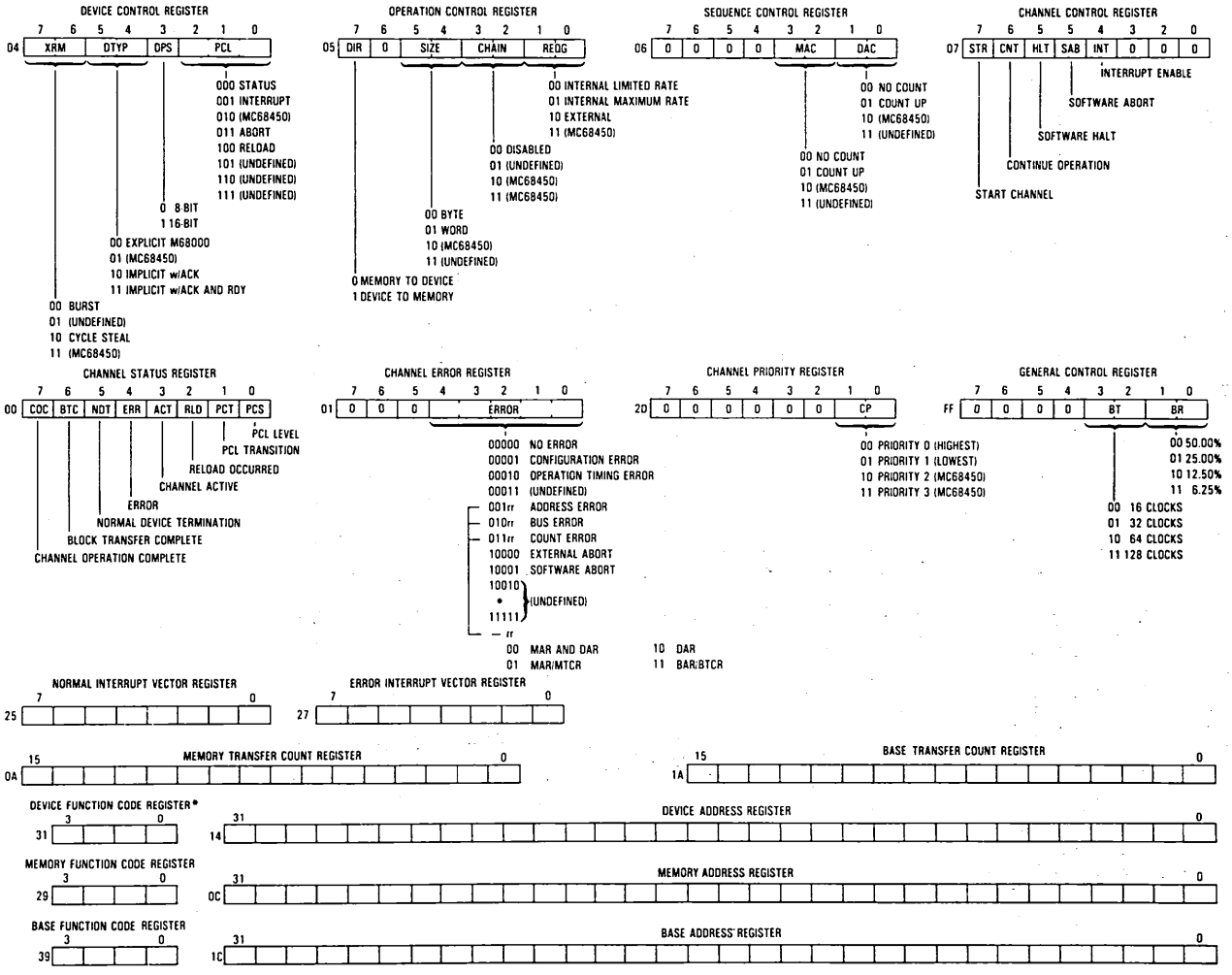


Figure 10. Register Summary

The register memory map is identical to the register memory map for the MC68450 DMAC, including the individual bit assignments within the registers. However, not all functional options available on the DMAC are available on the DDMA and vice versa, and the channel 2 and 3 registers on the DMAC are treated as null registers on the DDMA. If any programmable options labeled 'MC68450 Reserved' or 'Undefined, Reserved' are programmed into a DDMA channel, a configuration error will occur when the MPU attempts to start that channel.

All registers within the DDMA are always accessible as bytes or words by the MPU (assuming that the MPU can gain control of the DMA bus); however, some registers may not or should not be modified while a channel is actively transferring data. If a register may not be modified during operation and an attempt is made to write to it, an operation timing error will be signaled and the channel operation aborted.

RESET OPERATION RESULTS

When the DDMA is reset, either during a system power-up sequence or to re-initialize the DDMA, many of the registers will be affected and will be set to known values. Table 2 shows the hexadecimal value that will be placed in each register by a reset operation.

Table 2. Reset Operation Results

Register	Value	Comments
MARc	XXXXXXXX	
DARc	XXXXXXXX	
BARc	XXXXXXXX	
MFCRc	X	
DFCRc	X	
BFCRc	X	
MTCRc	XXXX	
BTCRc	XXXX	
NIVRc	0F	Uninitialized Vector
EIVRc	0F	Uninitialized Vector
CPRc	00	
DCRc	00	
OCRc	00	
SCRc	00	
CCRc	00	Channel Not Active, Interrupts Disabled
CSRc	00 or 01	(Depending on the Level of PCLc)
CERc	00	No Errors
GCR	00	

X r Indicates an unknown value or the previous value of the register.
c r Is the channel number (i.e., 0 or 1).

ELECTRICAL SPECIFICATIONS

MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Supply Voltage	V_{CC}	−0.3 to +7.0	V
Input Voltage	V_{in}	−0.3 to +7.0	V
Operating Temperature Range	T_A	0 to 70	°C
Storage Temperature	T_{stg}	−55 to 150	°C

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

THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Symbol	Value	Unit
Thermal Resistance	θ_{JA}		θ_{JC}		°C/W
Ceramic (L Suffix)		30		15*	
Plastic (P Suffix)		30		15*	
Pin Grid Array (R/RC Suffix)		33		15	

*Estimated

POWER CONSIDERATIONS

The average chip-junction temperature, T_J, in °C can be obtained from:

$$T_J = T_A + (P_D \cdot \theta_{JA}) \quad (1)$$

where:

T_A = Ambient Temperature, °C
θ_{JA} = Package Thermal Resistance, Junction-to-Ambient, °C/W
$P_D = P_{INT} + P_{I/O}$
$P_{INT} = I_{CC} \times V_{CC}$, 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.

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

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

Solving equations (1) and (2) for K gives:

$$K = P_D \cdot (T_A + 273°C) + \theta_{JA} \cdot P_D^2 \quad (3)$$

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.

The total thermal resistance of a package (θ_{JA} can be separated into two components, θ_{JA} and θ_{CA}, representing the barrier to heat flow from the semiconductor junction to the package (case) surface (θ_{JC}) and from the case to the outside ambient (θ_{CA}). These terms are related by the equation:

$$\theta_{JA} = \theta_{JC} + \theta_{CA} \quad (4)$$

θ_{JC} is device related and cannot be influenced by the user. However, θ_{CA} is user dependent and can be minimized by such thermal management techniques as heat sinks, ambient air cooling and thermal convection. Thus, good thermal management on the part of the user can significantly reduce θ_{CA} so that θ_{JA} aproximately equals θ_{JC}. Substitution of θ_{JC} for θ_{JA} in equation (1) will result in a lower semiconductor junction temperature.

Values for thermal resistance presented in this document, unless estimated, were derived using the procedure described in Motorola Reliability Report 7843, "Thermal Resistance Measurement Method for MC68XX Microcomponent Devices," and are provided for design purposes only. Thermal measurements are complex and dependent on procedure and setup. User derived values for thermal resistance may differ.

Figure 11 illustrates the graphic solution to the equations, given above, for the specification power dissipation of 1.50 watts over the ambient temperature range of −55°C to 125°C using an average θ_{JA} of 40°C/watt to represent the various MC68440/MC68442 packages. However, actual θ_{JA}'s in the range of 30°C to 50°C/watt only change the curve slightly.

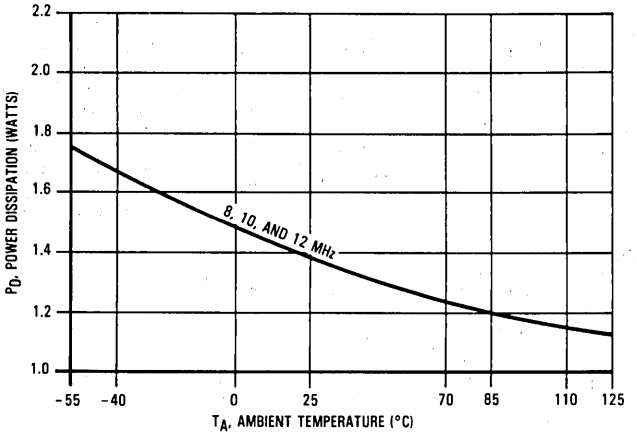


Figure 11. MC68440/MC68442 Power Dissipation vs Ambient Temperature

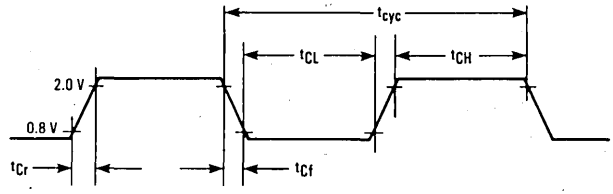
5

DC ELECTRICAL CHARACTERISTICS (V_{CC} = 4.75 V to 5.25 V, GND = 0 V, T_A = 0°C to 70°C, unless otherwise noted)

Characteristic		Symbol	Min	Max	Unit
Input High Voltage	All Inputs	V_{IH}	2.0	V_{CC}	V
Input Low Voltage	All Inputs	V_{IL}	GND −0.3	0.8	V
Input Leakage Current @ 5.25 V	All Inputs	I_{in}	—	10	μA
Input Capacitance (V_{in} = 0 V, T_A = 25°C, Frequency = 1 MHz)	All Inputs	C_{in}	—	13	pF
Three-State (Off-State) Input Current @ 2.4V/0.4V	$\overline{AS}$, A1-A7, $\overline{BGACK}$, $\overline{DTACK}$, A8/D0-A23/D15, HIBYTE, LDS/DS, UDS/A0, R/W	I_{TSI}	—	20	μA
Open-Drain (Off-State) Input Current @ 5.25 V	$\overline{IRQ}$, $\overline{DONE}$	I_{DD}	—	20	μA
Output High Voltage (I_{OH} = −400 μA Minimum)	$\overline{AS}$, A1-A7, A8/D0-A23/D15, $\overline{ACK0}$, $\overline{ACK1}$, $\overline{BR}$, $\overline{BGACK}$, DBEN, DDIR, DTACK, OWN, LDS/DS, UDS/A0, R/W, UAS, DTC, FC0-FC2, IRQ, DONE, HIBYTE	V_{OH}	2.4	—	V
Output Low Voltage (I_{OL} = 3.2 mA Minimum)	A1-A7, FC0-FC2	V_{OL}	—	0.5	V
(I_{OL} = 5.3 mA Minimum)	A8/D0-A23/D15, ACK0, ACK1, AS, BGACK, BR, DBEN, DDIR, DTACK, DTC, HIBYTE, LDS/DS, UDS/A0, OWN, R/W, UAS				
(I_{OL} = 8.9 mA Minimum)	IRQ, DONE				
Power Dissipation at 0°C (Frequency = 8 MHz)		P_D	—	1.5	W
Output Load Capacitance		C_L	—	130	pF

AC ELECTRICAL SPECIFICATIONS — CLOCK TIMING (see Figure 12)

Characteristic	Symbol	8 MHz		10 MHz		12 MHz		Unit
		Min	Max	Min	Max	Min	Max	
Frequency of Operation	F	2.0	8.0	2.0	10.0	2.0	12.5	MHz
Cycle Period	t_{cyc}	125	500	100	500	80	500	ns
Clock Width Low	t_{CL}	55	250	45	250	35	250	ns
Clock Width High	t_{CH}	55	250	45	250	35	250	ns
Clock Fall Time	t_{Cf}	—	10	—	10	—	5	ns
Clock Rise Time	t_{Cr}	—	10	—	10	—	5	ns



NOTE: Timing measurements are referenced to and from a low voltage of 0.8 volts and a high voltage of 2.0 volts, unless otherwise noted. The voltage swing through this range should start outside and pass through the range such that the rise or fall will be linear between 0.8 volts and 2.0 volts.

Figure 12. Clock Input Timing Diagram

5

AC ELECTRICAL SPECIFICATIONS — READ AND WRITE CYCLES (V_{CC} = 4.75 V to 5.25 V, GND = 0 V, T_A = 0°C to 70°C, unless otherwise noted) (see Figures 13 through 19)

Num.	Characteristic	Symbol	8 MHz		10 MHz		12 MHz		Unit
			Min	Max	Min	Max	Min	Max	
1	Clock Period	t_{cyc}	125	500	100	500	80	500	ns
2	Clock Width Low	t_{CL}	55	250	45	250	35	250	ns
3	Clock Width High	t_{CH}	55	250	45	250	35		ns
4	Clock Fall Time	t_{Cf}	—	10	—	10	—	5	ns
5	Clock Rise Time	t_{Cr}	—	10	—	10	—	5	ns
6	MPU Address Valid to $\overline{CS}$ Low	t_{ADVCSL}	0	—	0	—	0	—	ns
7	MPU $\overline{AS}$ High to Address Invalid	t_{ASHADI}	0	—	0	—	0	—	ns
8	Asynchronous Input Setup Time	t_{ASI}	20	—	20	—	20	—	ns
9	Data Strobe(s) Low to $\overline{CS}$ Low	t_{DSLCSL}	0	—	0	—	0	—	ns
10[1]	$\overline{CS}$ Low to $\overline{DDIR}$ High (MPU Read)	$t_{CSLDDHR}$	2 Clks +20	3 Clks +80	2 Clks +20	3 Clks +70	2 Clks +20	3 Clks +60	ns
11[1]	$\overline{CS}$ Low to $\overline{DBEN}$ Low (MPU Read)	$t_{CSLENLR}$	2.5 Clks +20	3.5 Clks +80	2.5 Clks +20	3.5 Clks +70	2.5 Clks +20	3.5 Clks +60	ns
12[1]	$\overline{CS}$ Low to Data Out Valid (MPU Read)	t_{CSLDOV}	2 Clks +20	3 Clks +110	2 Clks +20	3 Clks +95	2 Clks +20	3 Clks +80	ns
13[1]	$\overline{CS}$ Low to $\overline{DTACK}$ Low (MPU Read)	$t_{CSLDTLR}$	3.5 Clks +20	4.5 Clks +80	3.5 Clks +20	4.5 Clks +70	3.5 Clks +20	4.5 Clks +60	ns
14	Clock High to Data Out Valid	t_{CHDOV}	—	90	—	75	—	65	ns
15	$\overline{CS}$ High to $\overline{DDIR}$ High Impedance	t_{CSHDOZ}	—	60	—	50	—	40	ns
16	$\overline{CS}$ High to $\overline{DBEN}$ High Impedance	t_{CSHENZ}	—	60	—	50	—	40	ns
17	$\overline{CS}$ High to Data High Impedance	t_{CSHDZ}	—	60	—	50	—	40	ns
18	Clock Low to $\overline{DTACK}$ Low	t_{CLDTL}	—	60	—	50	—	40	ns
19	$\overline{DTACK}$ Low to $\overline{CS}$ High	t_{DTLSCH}	0	—	0	—	0	—	ns
20	$\overline{CS}$ High to $\overline{DTACK}$ High	t_{CSHDTH}	—	50	—	45	—	35	ns
21	$\overline{CS}$ High to $\overline{DTACK}$ High Impedance	t_{CSHDTZ}	—	60	—	50	—	40	ns
22	$\overline{CS}$ Width High	t_{CSWH}	1	—	1	—	1	—	Clk Per
23	R/$\overline{W}$ Low to $\overline{CS}$ Low	t_{RWLCSL}	0	—	0	—	0	—	ns
24[1]	$\overline{CS}$ Low to $\overline{DDIR}$ Low (MPU Write)	$t_{CSLDDLW}$	1 Clk +20	2 Clks +80	1 Clk +20	2 Clks +70	1 Clk +20	2 Clks +60	ns
25[1]	$\overline{CS}$ Low to $\overline{DBEN}$ Low (MPU Write)	$t_{CSLENLW}$	1.5 Clks +20	2.5 Clks +80	1.5 Clks +20	2.5 Clks +70	1.5 Clks +20	2.5 Clks +60	ns
26	$\overline{CS}$ Low to Data In Valid (MPU Write)	t_{CSLDIV}	—	3	—	3	—	3	Clk Per
27[1]	$\overline{CS}$ Low to $\overline{DTACK}$ Low (MPU Write)	$t_{CSLDTLW}$	2.5 Clks +20	3.5 Clks +80	2.5 Clks +20	3.5 Clks +70	2.5 Clks +20	3.5 Clks +60	ns
28[8]	$\overline{DDIR}$ Low to $\overline{DBEN}$ Low	t_{DDLENL}	30	—	20	—	20	—	ns
29	$\overline{DBEN}$ Low to Data In Valid	t_{ENLDIV}	—	105	—	80	—	60	ns
30	Data In Valid to Clock High (Setup Time)	t_{DIVCH}	15	—	15	—	10	—	ns
31[8]	$\overline{DTACK}$ Low to $\overline{DDIR}$ High	t_{DTLDDH}	125	—	100	—	80	—	ns
32[8]	$\overline{DTACK}$ Low to $\overline{DBEN}$ High	t_{DTLENH}	65	—	50	—	40	—	ns

AC ELECTRICAL SPECIFICATIONS — READ AND WRITE CYCLES (Continued)

Num.	Characteristic	Symbol	8 MHz		10 MHz		12 MHz		Unit
			Min	Max	Min	Max	Min	Max	
33[8]	$\overline{DBEN}$ High to $\overline{DDIR}$ High (Read)	t_{ENHDDH}	30	—	20	—	20	—	ns
34	$\overline{CS}$ High to Data Not Valid	t_{CSHDNV}	0	—	0	—	0	—	ns
34a	Clock Low to Data Not Valid	t_{CLDNV}	0	—	0	—	0	—	ns
35	$\overline{REQ}$ Width Low	t_{REQL}	2	—	2	—	2	—	Clk Per
36	$\overline{REQ}$ Low to $\overline{BR}$ Low	$t_{REQLBRL}$	2 Clks +20	3 Clks +80	2 Clks +20	3 Clks +70	2 Clks +70	3 Clks +60	ns
37[2]	$\overline{REQ}$ Low to $\overline{BGACK}$ Low	$t_{REQLBKL}$	4	—	4	—	4	—	Clk Per
38	Clock High to $\overline{BR}$ Low	t_{CHBRL}	—	60	—	50	—	40	ns
39[3]	$\overline{BR}$ Low to $\overline{BG}$ Low	t_{BRLBGL}	−1	—	−1	—	−1	—	Clk Per
40[3]	$\overline{BR}$ Low to $\overline{AS}$ In High	t_{BRLASH}	−1	—	−1	—	−1	—	Clk Per
41	Clock High to $\overline{BR}$ High Impedance	t_{CHBRZ}	—	60	—	50	—	40	ns
42	Clock Low to $\overline{OWN}$ Low	t_{CLOL}	—	60	—	50	—	40	ns
43	Clock High to $\overline{BGACK}$ Low	t_{CHBKL}	—	60	—	50	—	40	ns
44[8]	$\overline{BGACK}$ Low to $\overline{BR}$ High Impedance	t_{BKLBRZ}	60	1 Clk +60	50	1 Clk +50	40	1 Clk +40	ns
45	$\overline{BGACK}$ to $\overline{BG}$ High	t_{BKLBGH}	0	—	0	—	0	—	ns
46[2]	$\overline{AS}$, $\overline{CS}$ In High to $\overline{BGACK}$ Low	t_{ASHBKL}	2	—	2	—	2	—	Clk Per
47[2]	Clock Low on which $\overline{OWN}$ Asserted to Clock High on which $\overline{AS}$ Asserted	t_{OLASL}	—	1.5	—	1.5	—	1.5	Clk Per
48	Clock High to $\overline{BGACK}$ High	t_{CHBKH}	—	60	—	50	—	40	ns
49	Clock High to $\overline{BGACK}$ High Impedance	t_{CHBKZ}	—	65	—	55	—	45	ns
50	Clock Low to $\overline{OWN}$ High	t_{CLOH}	—	60	—	50	—	40	ns
51	Clock Low to $\overline{OWN}$ High Impedance	t_{CHOZ}	—	65	—	55	—	45	ns
52[4,8]	$\overline{BGACK}$ High to $\overline{OWN}$ High	t_{BKHOH}	30	—	20	—	20	—	ns
53[8]	$\overline{DTC}$ High Impedance to $\overline{BGACK}$ High	t_{TCZBKH}	—	1 Clk +60	—	1 Clk +50	—	1 Clk +40	ns
54	Clock High to Address/FC Valid	t_{CHAV}	—	90	—	75	—	65	ns
55	Clock High to Control and Non-Muxed Bus Lines High Impedance	t_{CHNXZ}	—	60	—	50	—	40	ns
56	CLK Low to Muxed Address Bus High Impedance	t_{CLMXAZ}	—	60	—	50	—	40	ns
57	Data In Valid to Clock High (Setup Time)	t_{DIVCH}	15	—	15	—	10	—	ns
58	Clock High to $\overline{UAS}$ Low	t_{CHUL}	—	90	—	75	—	54	ns
59	Clock High to $\overline{UAS}$ High	t_{CHUH}	—	60	—	50	—	40	ns
60[8]	$\overline{UAS}$ High to Address Invalid	t_{UHAI}	20	—	20	—	20	—	ns
61[8]	Address/FC Valid to $\overline{AS}$/$\overline{DS}$ (Read), $\overline{AS}$ (Write) Low	t_{AVSL}	60	—	50	—	40	—	ns
62	$\overline{AS}$, $\overline{DS}$ Width Low (Read)	t_{ASLR}	125	—	100	—	80	—	ns
63	Clock Low to $\overline{AS}$, $\overline{DS}$ High	t_{CLSH}	—	60	—	50	—	40	ns
64[8]	$\overline{AS}$, $\overline{DS}$ High to Address/FC/Data Invalid	t_{SHAI}	40	—	20	—	20	—	ns

5

AC ELECTRICAL SPECIFICATIONS — READ AND WRITE CYCLES (Continued)

Num.	Characteristic	Symbol	8 MHz		10 MHz		12 MHz		Unit
			Min	Max	Min	Max	Min	Max	
65[8]	$\overline{AS}$ High to $\overline{UAS}$ Low	t_{ASHUL}	20	—	20	—	20	—	ns
66[5]	Clock High to $\overline{AS}$ Low	t_{CHASL}	—	50	—	40	—	40	ns
67[8]	$\overline{AS}$ Low to $\overline{DBEN}$ Low	t_{ASLENL}	—	120	—	100	—	80	ns
68	Clock High to $\overline{DS}$ Low (Read)	t_{CHDSLR}	—	60	—	50	—	40	ns
69	Clock High to $\overline{DDIR}$ Low	t_{CHDDL}	—	60	—	50	—	40	ns
70	Clock High to $\overline{DDIR}$ High	t_{CHDDH}	—	60	—	50	—	40	ns
71	Clock Low to $\overline{DBEN}$ Low	t_{CLENL}	—	60	—	50	—	50	ns
72	Clock Low to $\overline{DBEN}$ High (Read)	t_{CLENH}	—	60	—	50	—	40	ns
73	$\overline{DTACK}$ Low to Data In Valid	t_{DTLDIV}	—	105	—	80	—	60	ns
74	$\overline{DS}$ High to $\overline{DTACK}$ High	t_{DSHDTH}	0	100	0	80	0	80	ns
75	$\overline{BEC}$ Valid to $\overline{DTACK}$ Low	$t_{BECVDTL}$	0	—	0	—	0	—	ns
76	$\overline{AS}$ High to $\overline{BEC}$ Negated	$t_{ASHBECN}$	10	—	10	—	—	0	ns
77	$\overline{BEC}$ Width Low	t_{BECL}	2	—	2	—	2	—	Clk Per
78	Clock High to $\overline{ACK}$ Low (Read)	t_{CHAKLR}	—	60	—	50	—	40	ns
79	Clock High to $\overline{ACK}$ High	t_{CHAKH}	—	60	—	50	—	40	ns
80	Clock High to $\overline{DTC}$ Low	t_{CHTCL}	—	50	—	40	—	35	ns
81[4,8]	$\overline{DTC}$ Low to $\overline{DS}$ High	t_{TCLDSH}	30	—	20	—	20	—	ns
82	Clock High to $\overline{DTC}$ High	t_{CHTCH}	—	60	—	50	—	40	ns
83	$\overline{DONE}$ Input Low to Clock on which $\overline{DTC}$ Asserted	t_{DNLTCL}	20	—	20	—	20	—	ns
84	$\overline{DTC}$ Width Low	t_{DTCL}	1	—	1	—	1	—	Clk Per
85	Clock High to $\overline{DONE}$ Low (Read)	t_{CHDNL}	—	60	—	50	—	40	ns
86	Clock High to $\overline{DONE}$ High Impedance	t_{CHDNZ}	—	60	—	50	—	40	ns
87	Clock High to $\overline{IRQ}$ Low	t_{CHIRL}	—	60	—	50	—	40	ns
88	Clock High to $\overline{IRQ}$ High Impedance	t_{CHIRZ}	—	60	—	50	—	40	ns
89	Data Out Valid to $\overline{DS}$ Low	t_{DOVDSL}	70	—	50	—	45	—	ns
90	Clock High to Muxed Data Bus High Impedance	t_{CHMXDZ}	—	60	—	50	—	40	ns
91[8]	$\overline{UAS}$ Low to $\overline{AS}$ Low	t_{ULASL}	60	—	50	—	40	—	ns
92	$\overline{AS}$ Width Low (Write)	t_{ASLW}	250	—	200	—	160	—	ns
93	Clock Low to $\overline{DS}$ Low (Write)	t_{CLDSLW}	—	60	—	50	—	40	ns
94[8]	$\overline{DBEN}$ Low to $\overline{DS}$ Low (Write)	t_{ENLDSL}	60	—	50	—	40	—	ns
95	$\overline{DS}$ Width Low (Write)	t_{DSLW}	70	—	55	—	50	—	ns
96[8]	Address/FC Valid to R/$\overline{W}$ Low	t_{AVRL}	60	—	50	—	40	—	ns
97[8]	R/$\overline{W}$ Low to $\overline{DS}$ Low (Write)	t_{RLDSL}	125	—	100	—	80	—	ns
98[8]	$\overline{DS}$ High to R/$\overline{W}$ High	t_{DSHRH}	20	—	20	—	20	—	ns
99	Clock High to R/$\overline{W}$ High	t_{CHRH}	—	60	—	50	—	40	ns
100[5]	Clock High to R/$\overline{W}$ Low	t_{CHRL}	—	60	—	50	—	40	ns

5

AC ELECTRICAL SPECIFICATIONS — READ AND WRITE CYCLES (Concluded)

Num.	Characteristic	Symbol	8 MHz		10 MHz		12 MHz		Unit
			Min	Max	Min	Max	Min	Max	
101	Clock High to $\overline{DBEN}$ High (Write)	t_{CHENH}	—	60	—	50	—	40	ns
102	$\overline{DTACK}$ Width High	t_{DTWH}	0	—	0	—	0	—	ns
103	Clock Low to $\overline{ACK}$ Low (Write)	t_{CLAKLW}	—	60	—	50	—	40	ns
104	Clock Low to $\overline{DONE}$ Low (Write)	t_{CLDNL}	—	60	—	50	—	40	ns
105	Clock High to $\overline{HIBYTE}$ Low (Read)	t_{CHHBLR}	—	60	—	50	—	40	ns
106	Clock High to $\overline{HIBYTE}$ High	t_{CHHBH}	—	60	—	50	—	40	ns
107	$\overline{DTACK}$ and PCL Low to $\overline{AS}$ High (Single Address Read)	t_{DTLASH}	190	—	150	—	120	—	ns
108	Clock High to $\overline{HIBYTE}$ Low (Write)	t_{CLHBLW}	—	60	—	50	—	40	ns
109[8]	$\overline{ACK}$ Low to $\overline{DS}$ Low (Single Address Write)	t_{AKLDSL}	80	—	60	—	45	—	ns
110	PCL Low to $\overline{DS}$ Low ($\overline{ACK}$ with Ready Write)	t_{PCLDSL}	190	—	150	—	120	—	ns

NOTES:
1. These specifications assume that the input setup time for $\overline{CS}$ is zero, which violates #8, but it is still recognized as asserted.
2. With both channels active these numbers increase by one clock.
3. $\overline{AS}$ and $\overline{BG}$ from the MPU are first sampled on the rising clock edge on which $\overline{BR}$ is asserted. Therefore, if $\overline{AS}$ is negated and $\overline{BG}$ is asserted for at least one asynchronous input setup time prior to that clock edge, then the minimum arbitration times will be achieved.
4. These minimum times assume that the two signals have equal resistive and capacitive loading (±20%).
5. When $\overline{AS}$ and R/$\overline{W}$ are equally loaded (±10%) $\overline{AS}$ will be asserted no more than 20 ns before R/$\overline{W}$.
6. Minimum timing for single address write cycles occurs with $\overline{ACK}$ only or with $\overline{ACK}$ and PCL as READY when PCL is asserted for more than one synchronization delay before the clock edge on which $\overline{ACK}$ is asserted.
7. Specifications that include a number of clock periods refer to the actual input clock used and not the specified clock minimum or maximum values.
8. These specifications refer to the skew between two output signals that change following different edges of the clock; therefore, the actual value depends on the clock signal that is used. The minimum times are guaranteed for a minimum clock timing (high or low and period).

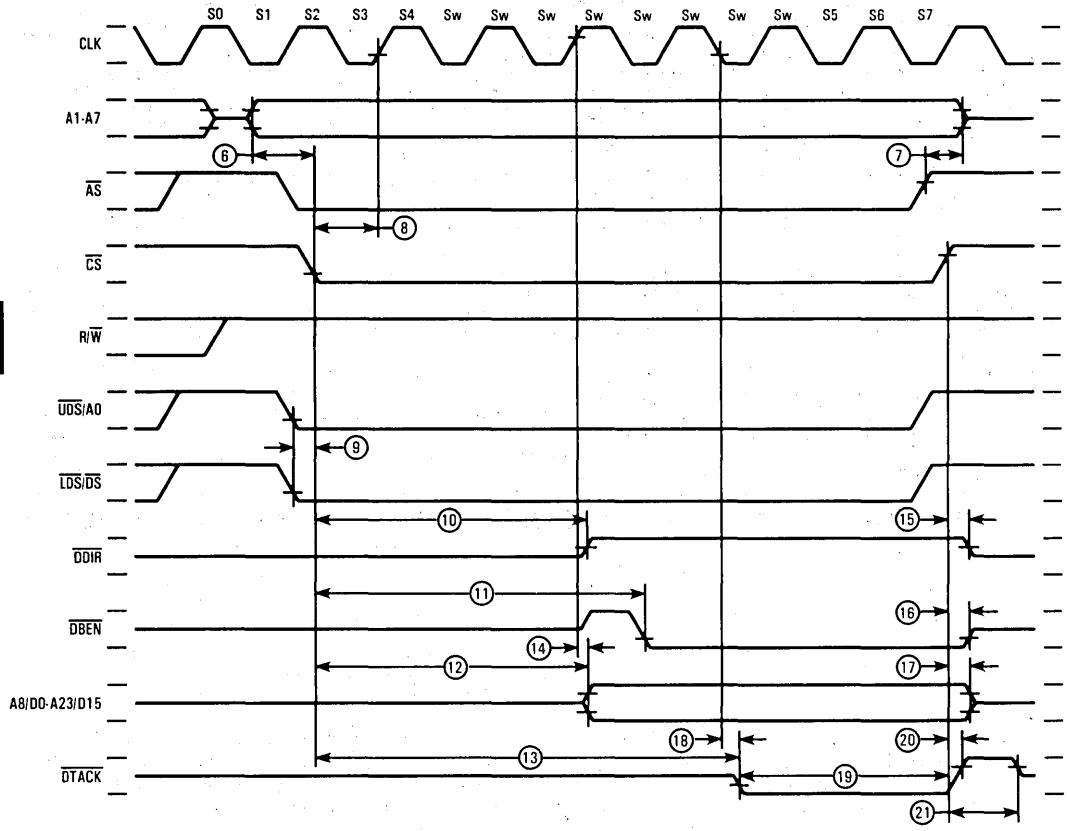
5

NOTE

For clarity, specification numbers are shown only once in Figures 16 through 19. However, many specifications apply equally to all four diagrams. For example, specification numbers 54 and 56 are shown only in Figure 16, but apply to Figures 17 through 19 as well. As a guideline, Figure 16 includes all necessary specifications for a dual-address read cycle and Figure 18 includes additional specifications for a single-address read cycle, the same relationship is true for Figures 17 and 19. Thus, the specifications shown in Figure 17 through 19 can be considered to be additions to or substitutions for the specifications shown in Figure 16.

When referring to the timing specifications shown in Figure 13 through 19, it is helpful to remember that all output signals will change states only in response to a specific transition on the CLK input and that all input signals are latched and synchronized on rising edges of the CLK input.

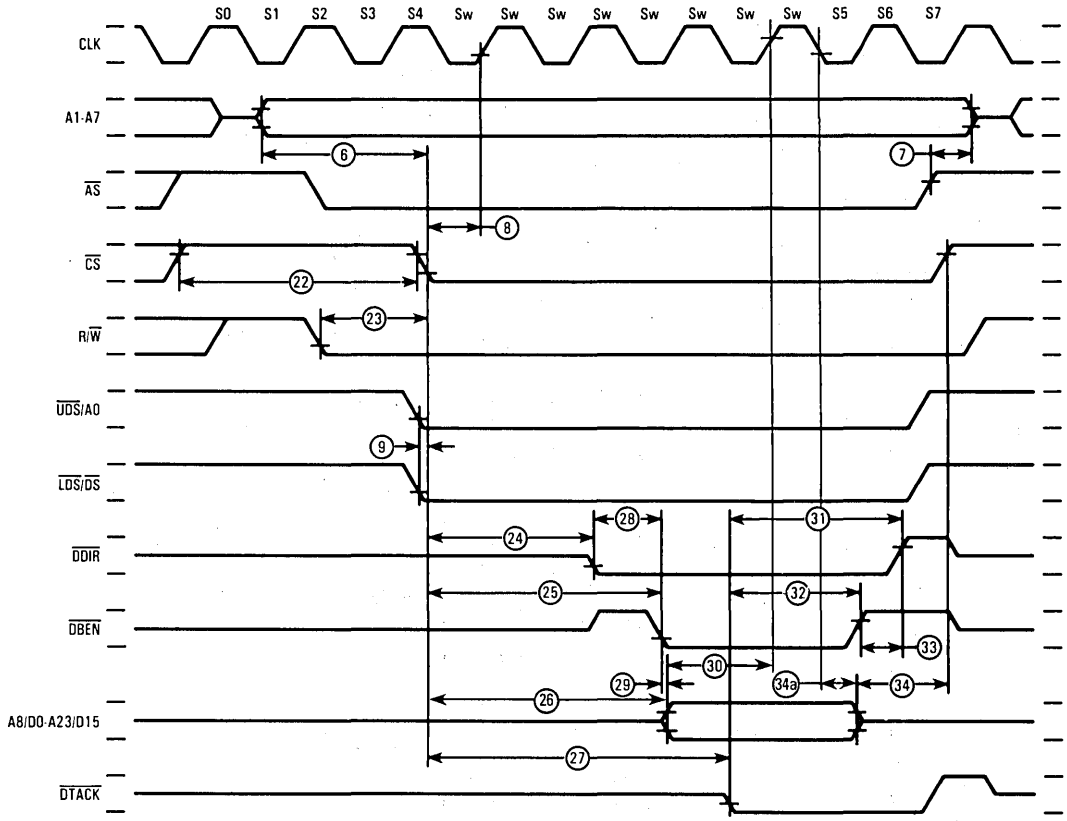
These waveforms should only be referenced in regard to the edge-to-edge measurement of the timing specifications. They are not intended as a functional description of the input and output signals. Refer to other functional descriptions and their related diagrams for device operation.



NOTE: Timing measurements are referenced to and from a low voltage of 0.8 volt and a high voltage of 2.0 volts, unless otherwise noted. The voltage swing through this range should start outside and pass through the range such that the rise or fall will be linear between 0.8 volt and 2.0 volts.

Figure 13. MPU Read Cycle Timing Diagram

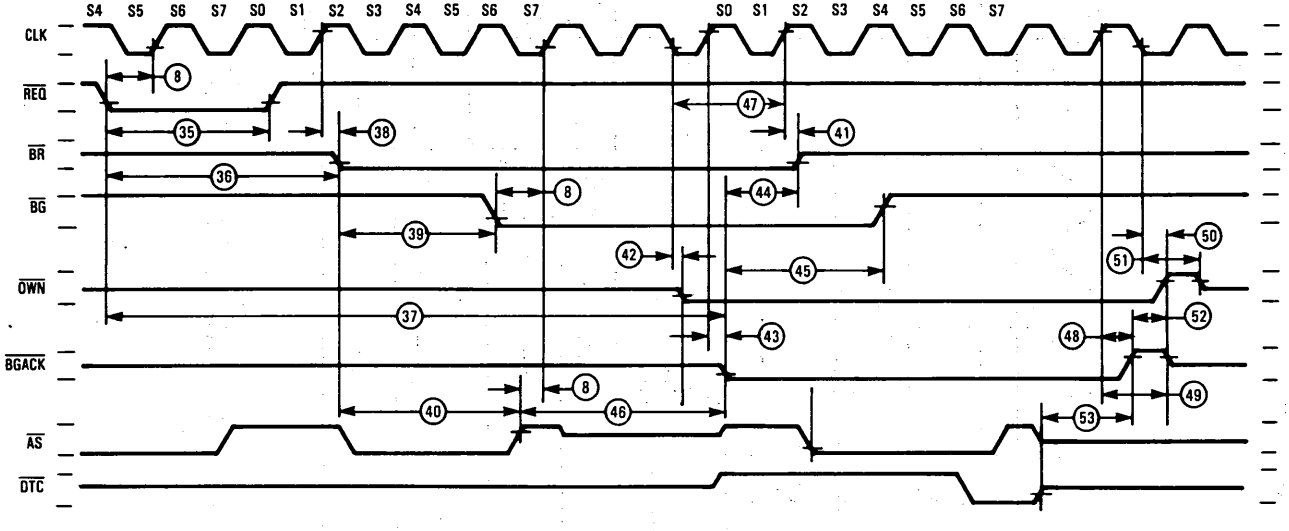
These waveforms should only be referenced in regard to the edge-to-edge measurement of the timing specifications. They are not intended as a functional description of the input and output signals. Refer to other functional descriptions and their related diagrams for device operation.



NOTE: Timing measurements are referenced to and from a low voltage of 0.8 volt and a high voltage of 2.0 volts, unless otherwise noted. The voltage swing through this range should start outside and pass through the range such that the rise or fall will be linear between 0.8 volt and 2.0 volts.

Figure 14. MPU Write Cycle Timing Diagram

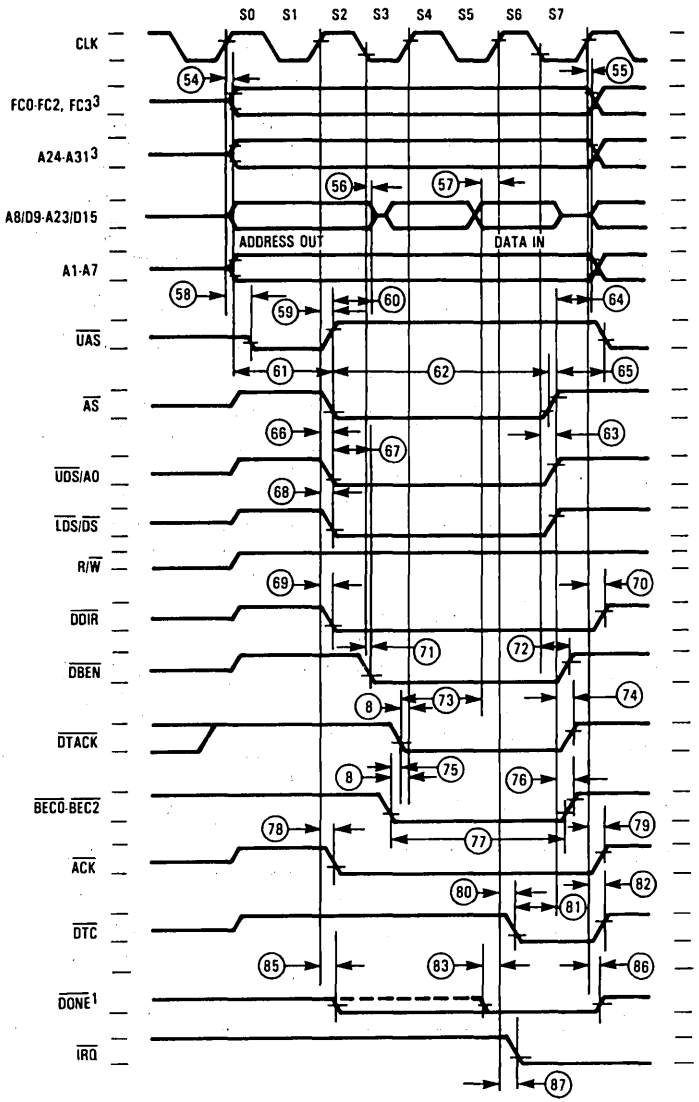
These waveforms should only be referenced in regard to the edge-to-edge measurement of the timing specifications. They are not intended as a functional description of the input and output signals. Refer to other functional descriptions and their related diagrams for device operation.



NOTE: Timing measurements are referenced to and from a low voltage of 0.8 volt and a high voltage of 2.0 volts, unless otherwise noted. The voltage swing through this range should start outside and pass through the range such that the rise or fall will be linear between 0.8 volt and 2.0 volts.

Figure 15. Bus Arbitration Timing Diagram

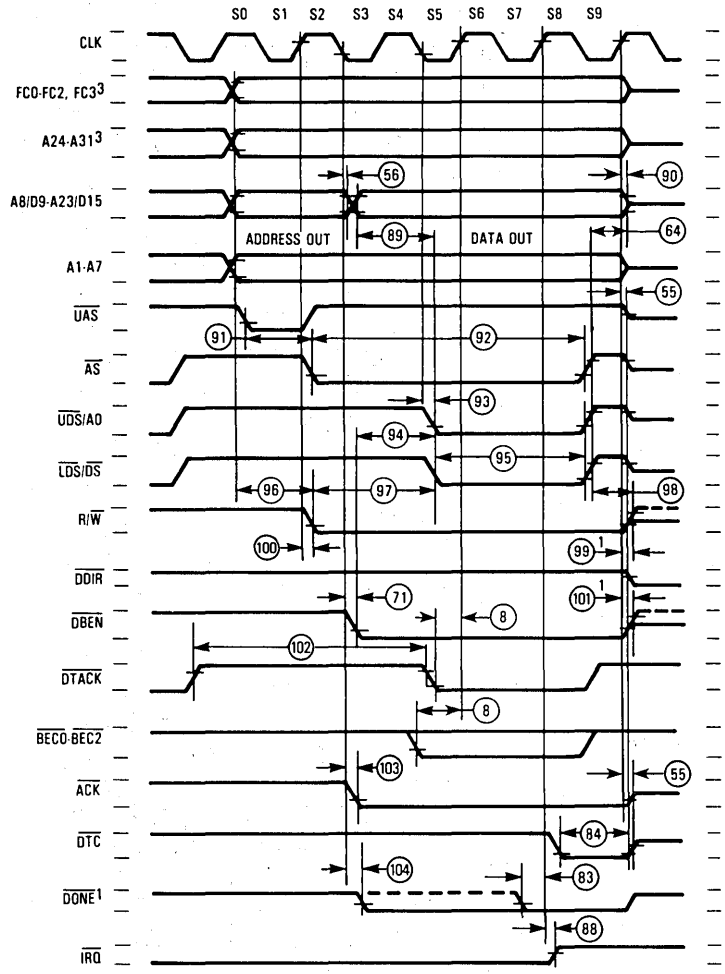
These waveforms should only be referenced in regard to the edge-to-edge measurement of the timing specifications. They are not intended as a functional description of the input and output signals. Refer to other functional descriptions and their related diagrams for device operation.



NOTES:
1. The solid line illustrates $\overline{\text{DONE}}$ as an output, and the dotted line illustrates $\overline{\text{DONE}}$ as an input.
2. Timing measurements are referenced to and from a low voltage of 0.8 volt and a high voltage of 2.0 volts, unless otherwise noted. The voltage swing through this range should start outside and pass through the range such that the rise or fall will be linear between 0.8 volt and 2.0 volts.
3. MC68442 only.

Figure 16. Dual Address Read Cycle Timing Diagram

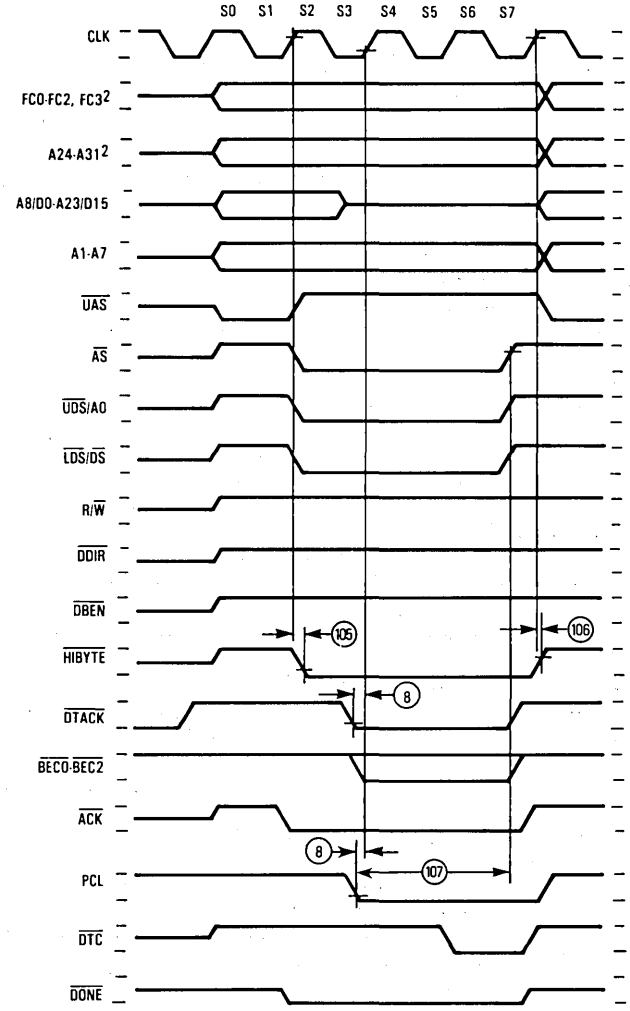
These waveforms should only be referenced in regard to the edge-to-edge measurement of the timing specifications. They are not intended as a functional description of the input and output signals. Refer to other functional descriptions and their related diagrams for device operation.



NOTES:
1. Timing specifications #99 and #101 are only applicable when another DDMA bus cycle immediatley follows this one.
2. Timing measurements are referenced to and from a low voltage of 0.8 volt and a high voltage of 2.0 volts, unless otherwise noted. The voltage swing through this range should start outside and pass through the range such that the rise or fall will be linear between 0.8 volt and 2.0 volts.
3. MC68442 only.

Figure 17. Dual Address Write Cycle Timing Diagram

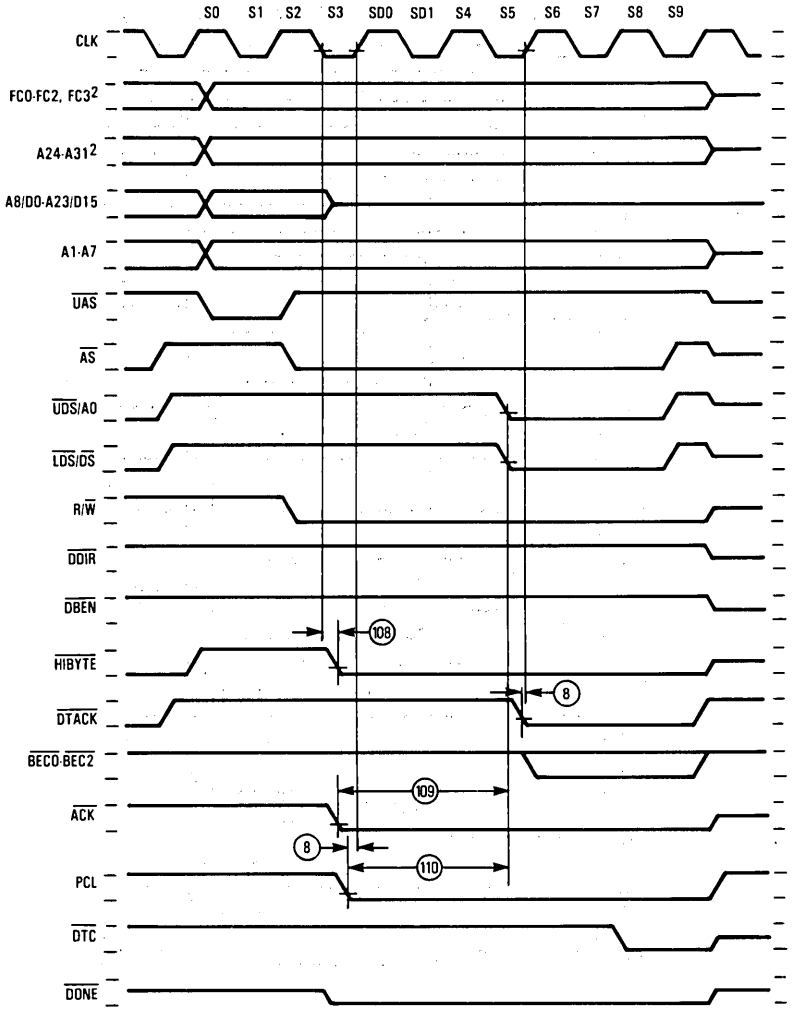
These waveforms should only be referenced in regard to the edge-to-edge measurement of the timing specifications. They are not intended as a functional description of the input and output signals. Refer to other functional descriptions and their related diagrams for device operation.



NOTES:
1. Timing measurements are referenced to and from a low voltage of 0.8 volt and a high voltage of 2.0 volts, unless otherwise noted. The voltage swing through this range should start outside and pass through the range such that the rise or fall will be linear between 0.8 volt and 2.0 volts.
2. MC68442 only.

Figure 18. Single Address Read Cycle Timing Diagram

These waveforms should only be referenced in regard to the edge-to-edge measurement of the timing specifications. They are not intended as a functional description of the input and output signals. Refer to other functional descriptions and their related diagrams for device operation.

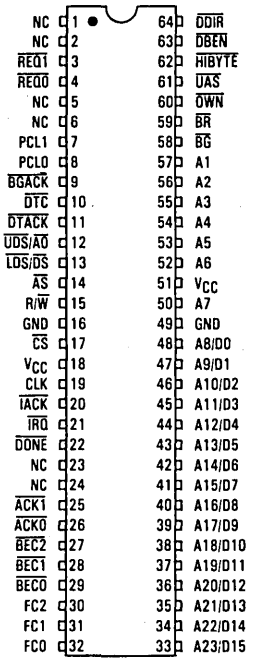


NOTES:
1. Timing measurements are referenced to and from a low voltage of 0.8 volt and a high voltage of 2.0 volts, unless otherwise noted. The voltage swing through this range should start outside and pass through the range such that the rise or fall will be linear between 0.8 volt and 2.0 volts.
2. MC68442 only.

Figure 19. Single Address Write Cycle Timing Diagram

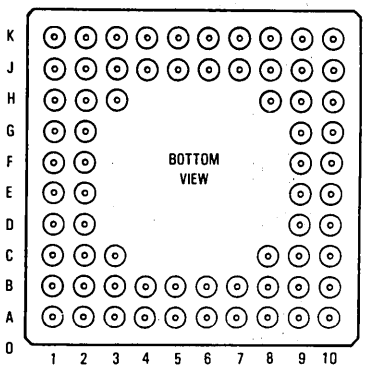
PIN ASSIGNMENTS

64-Pin Dual-in-Line Package
MC68440 Only



5

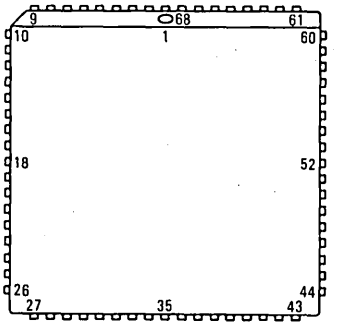
68-Terminal Pin Grid Array
MC68440 and MC68442



Pin Number	Function	
	MC68440	MC68442
A1	NC	FC3
B1	A13/D5	
C1	A11/D3	
D1	A10/D2	
E1	A8/D0	
F1	A7	
G1	A6	
H1	A5	
J1	A3	
K1	NC	A25
K2	$\overline{BR}$	
K3	$\overline{UAS}$	
K4	$\overline{DBEN}$	
K5	NC	A24
K6	NC	A26
K7	$\overline{REQ0}$	
K8	NC	A28
K9	PCL1	
K10	$\overline{DTACK}$	
J10	$\overline{UDS}$/A0	
H10	$\overline{AS}$	
G10	R/$\overline{W}$	
F10	NC	
E10	$\overline{CS}$	
D10	$\overline{CLK}$	
C10	$\overline{IACK}$	
B10	NC	A30
A10	$\overline{ACK0}$	
A9	$\overline{BEC1}$	
A8	FC2	
A7	FC1	
A6	A23/D15	
A5	A22/D14	
A4	A20/D12	

Pin Number	Function	
	MC68440	MC68442
A3	A19/D11	
A2	A17/D9	
B2	A15/D7	
C2	A12/D4	
D2	A9/D1	
E2	GND	
F2	V_{CC}	
G2	A4	
H2	A2	
J2	$\overline{BG}$	
J3	$\overline{OWN}$	
J4	$\overline{HIBYTE}$	
J5	$\overline{DDIR}$	
J6	$\overline{REQ1}$	
J7	NC	A29
J8	PCL0	
J9	NC	A27
H9	$\overline{BGACK}$	
G9	$\overline{LDS}$/$\overline{DS}$	
F9	GND	
E9	V_{CC}	
D9	$\overline{DONE}$	
C9	$\overline{IRQ}$	
B9	NC	A31
B8	$\overline{BEC2}$	
B7	$\overline{BEC0}$	
B6	FC0	
B5	A21/D13	
B4	A18/D10	
B3	A16/D8	
C3	A14/D6	
H3	A1	
H8	$\overline{DTC}$	
C8	$\overline{ACK1}$	

68-Lead Pastic Leaded
Chip Carrier
MC68440 and MC68442



Pin Number	Function	
	MC68440	MC68442
1	$\overline{\text{DDIR}}$	
2	NC	A25
3	NC	A26
4	NC	A27
5	$\overline{\text{REQ1}}$	
6	$\overline{\text{REQ0}}$	
7	NC	A28
8	NC	A29
9	$\overline{\text{PCL1}}$	
10	$\overline{\text{PCL0}}$	
11	$\overline{\text{BGACK}}$	
12	$\overline{\text{DTC}}$	
13	$\overline{\text{DTACK}}$	
14	$\overline{\text{UDS}}$/A0	
15	$\overline{\text{LDS}}$/$\overline{\text{DS}}$	
16	$\overline{\text{AS}}$	
17	R/$\overline{\text{W}}$	
18	GND	
19	$\overline{\text{CS}}$	
20	V_{CC}	
21	CLK	
22	$\overline{\text{IACK}}$	
23	$\overline{\text{IRQ}}$	
24	$\overline{\text{DONE}}$	
25	NC	A30
26	NC	A31
27	$\overline{\text{ACK1}}$	
28	$\overline{\text{ACK0}}$	
29	$\overline{\text{BEC2}}$	
30	$\overline{\text{BEC1}}$	
31	$\overline{\text{BEC0}}$	
32	NC	FC3
33	FC2	
34	FC1	

Pin Number	Function	
	MC68440	MC68442
35	FC0	
36	A23/D15	
37	A22/D14	
38	A21/D13	
39	A20/D12	
40	A19/D11	
41	A18/D10	
42	A17/D9	
43	A16/D8	
44	A15/D7	
45	A14/D6	
46	A13/D5	
47	A12/D4	
48	A11/D3	
49	A10/D2	
50	A9/D1	
51	A8/D0	
52	NC	
53	GND	
54	A7	
55	V_{CC}	
56	A6	
57	A5	
58	A4	
59	A3	
60	A2	
61	A1	
62	$\overline{\text{BG}}$	
63	$\overline{\text{BR}}$	
64	$\overline{\text{OWN}}$	
65	$\overline{\text{UAS}}$	
66	$\overline{\text{HIBYTE}}$	
67	$\overline{\text{DBEN}}$	
68	NC	A24

5

MC68450

Technical Summary

Direct Memory Access Controller (DMAC)

M68000 microprocessors utilize state-of-the-art MOS technology to maximize performance and throughput. The MC68450 direct memory access controller (DMAC) is designed to complement the performance and architectural capabilities of M68000 Family microprocessors by moving blocks of data in a quick, efficient manner with minimum intervention from a processor. The DMAC performs memory-to-memory, memory-to-device, and device-to-memory data transfers by utilizing the following features:

- Four Independent DMA Channels with Programmable Priority
- Asynchronous M68000 Bus Structure with a 24-Bit Address and a 16-Bit Data Bus
- Fast Transfer Rates: Up to 5 Megabytes per Second at 10 MHz, No Wait States
- Fully Supports all M68000 Bus Options such as Halt, Bus Error, and Retry
- FIFO Locked Step Support with Device Transfer Complete Signal
- Flexible Request Generation:
  Internal, Maximum Rate
  Internal, Limited Rate
  External, Cycle Steal (With or Without Hold)
  External, Burst
  Mixed Internal and External
- Programmable 8-Bit or 16-Bit Peripheral Device Types:
  Explicitly Addressed, M68000 Type
  Explicitly Addressed, M6800 Type
  Implicitly Address:
    Device with Request and Acknowledge
    Device with Request, Acknowledge, and Ready
- Pin and Register Compatible Functional Superset of the MC68440 DDMA and MC68442 EDMA

5

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

INTRODUCTION

The main purpose of a direct memory access (DMA) controller in any system is to transfer data at very high rates, usually much faster than a microprocessor under software control can handle. The term DMA is used to refer to the ability of a peripheral device to access memory in a system in the same manner as a microprocessor does. DMA operation can occur concurrently with other operations that the system processor needs to perform, thus greatly boosting overall system performance.

Figure 1 illustrates a typical system configuration using a DMA interface to a high speed disk storage device. In a system such as this, the DMAC will move blocks of data between the peripheral controllers and memory at rates approaching the limits of the memory bus, since the simple function of data movement is implemented in high speed MOS hardware. A block of data consists of a sequence of byte, word, or long-word operands starting at

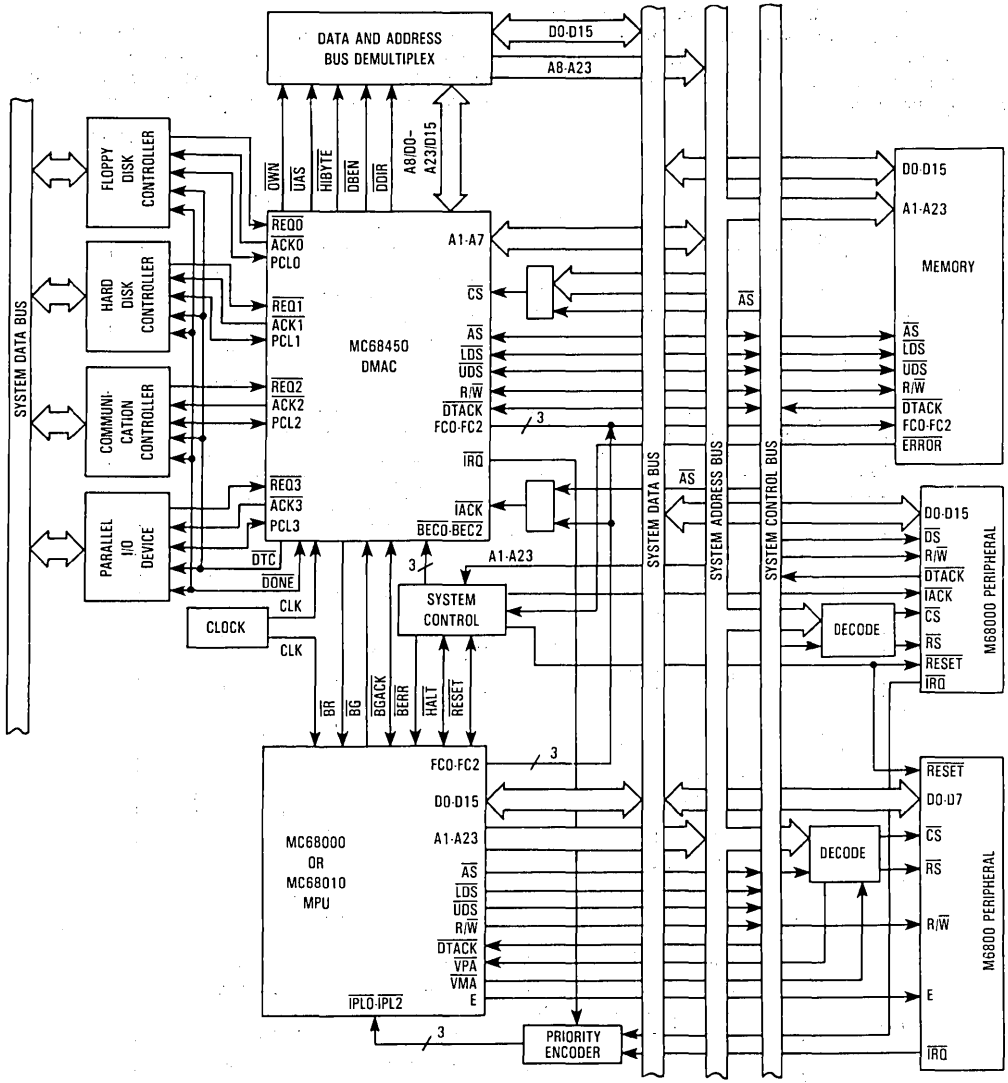


Figure 1. Typical M68000-Based System Configuration

a specific address in memory with the length of the block determined by a transfer count. A single channel operation may involve the transfer of several blocks of data between the memory and a device.

Any operation involving the DMAC will follow the same basic steps: channel initialization by the system processor, data transfer, and block termination. In the initialization phase, the host processor loads the registers of the DMAC with control information, address pointers, and transfer counts and then starts the channel. During the transfer phase, the DMA accepts requests for operand transfers and provides addressing and bus control for the transfers. The termination phase occurs after the operation is complete, when the DMAC indicates the status of the operation in the status register. During all phases of a data transfer operation, the DMAC will be in one of three operating modes:

IDLE This is the state that the DMAC assumes when it is reset by an external device and waiting for initialization by the system processor or an operand transfer request from a peripheral.

MPU This is the state that the DMAC enters when it is chip selected by another bus master in the system (usually the main system processor). In this mode, the DMAC internal registers are written or read, to control channel operation or check the status of a block transfer.

DMA This is the state that the DMAC enters when it is acting as a bus master to perform an operand transfer.

In addition to fully supporting the M68000 Family bus, the DMAC also supports transfers to or from M6800 Family peripheral devices. Figure 2 illustrates a typical system configuration utilizing an M6800-type peripheral device.

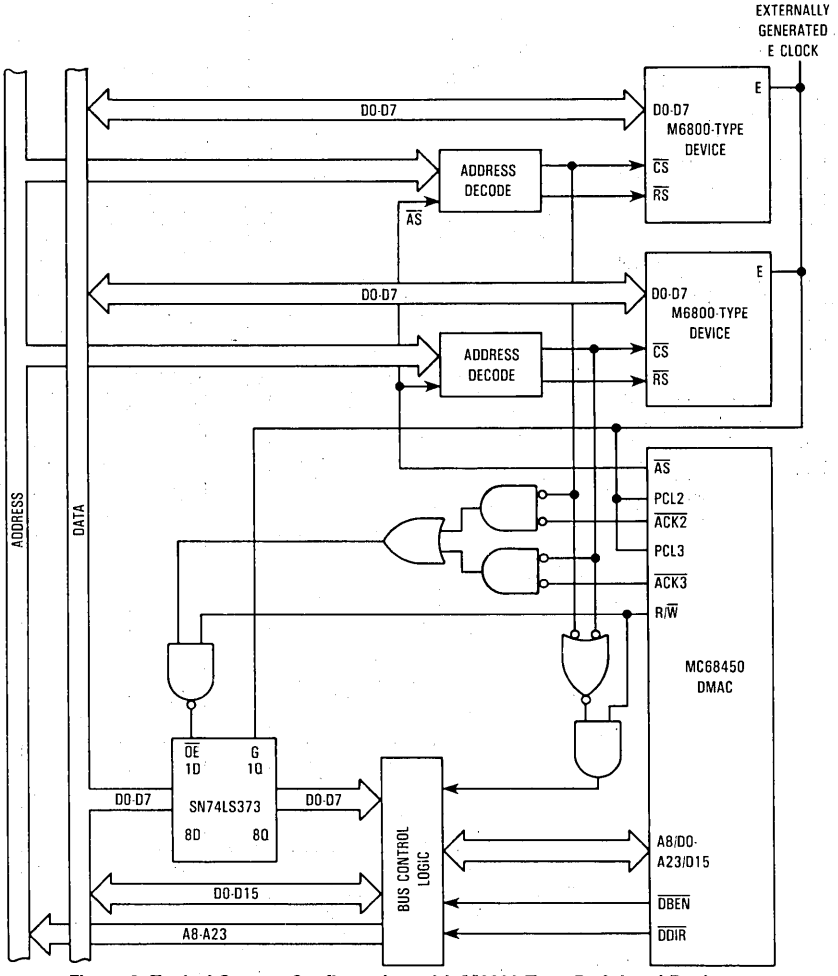


Figure 2. Typical System Configuration with M6800-Type Peripheral Devices

OPERAND TRANSFER MODES

The DMAC can perform implicit address of explicit address data transfers using any of the following protocols:

1. explicitly addressed, MC68000 compatible device,
2. explicitly addressed, MC6800 compatible devices,
3. implicitly addressed, device with acknowledge, and
4. implicitly addressed, device with acknowledge and ready.

In the first two protocols, data is transferred from the source to an internal DMAC holding register, and then on the next bus cycle moved from the holding register to the destination. Protocols 3 and 4 require only one bus cycle for data transfer, since only one device needs to be addressed. With these protocols, communication is performed using a two-signal and three-signal handshake, respectively.

Implicitly addressed devices do not require the generation of a device data register address for a data transfer. Such a device is controlled by a five signal device control interface on the DMAC during implicit address transfers as shown in Figure 3. Since only memory is addressed during such a data transfer, this method is called the single-address method.

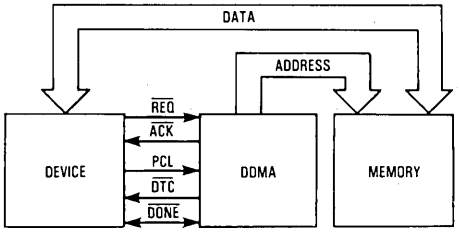


Figure 3. Implicitly Addressed Device Interface

Explicitly addressed devices require that a data register within the peripheral device be addressed. No signals other than the M68000 asynchronous bus control signals are needed to interface with such a device, although any of the five device control signals may also be used. Because the address bus is used to access the peripheral, the data cannot be directly transferred to/from memory since memory also requires addressing. Therefore, data is transferred from the source to an internal holding register in the DMAC and then transferred to the destination during a second bus transfer as shown in Figure 4. Since both memory and the device are addressed during such a data transfer, this method is called the dual-address method.

CHANNEL OPERATING MODES

There are three types of channel operations: 1) single block transfers, 2) continued operation, and 3) chained operations. The first two modes utilize on-chip registers while the last mode uses an on-chip address register to

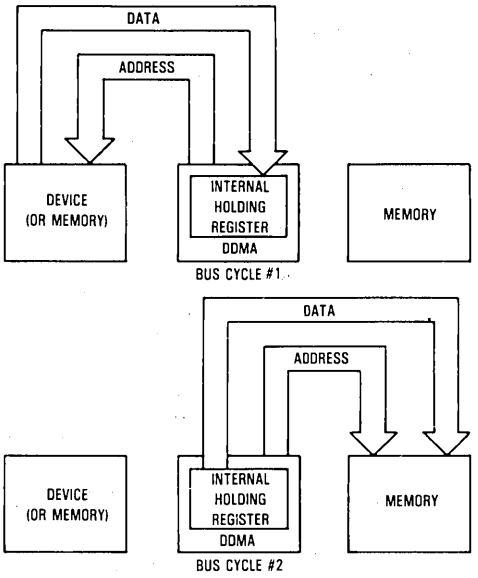


Figure 4. Dual-Address Transfer Sequence

point to address and count parameters stored in system memory.

When transferring single blocks of data, the memory address and device address registers are initialized by the user to specify the source and destination of the transfer. Also initialized is the memory transfer count register to count the number of operands transferred in a block. Repeated transfers are possible with the continue mode of operation, where the memory address and transfer count registers are automatically loaded from internal registers upon completion of a block transfer. See Figure 5.

The two chaining modes are array chaining and linked array chaining. The array chaining mode operates from a contiguous array in memory consisting of memory addresses and transfer counts. The base address register and base transfer count register are initialized to point to the beginning address of the array and the number of array entries, respectively. As each block transfer is completed, the next entry is fetched from the array, the base transfer count is decremented, and the base address is incremented to point to the next array entry. When the base transfer count reaches zero, the entry just fetched is the last block transfer defined in the array. See Figure 6.

The linked array chaining mode is similar to the array chaining mode, except that each entry in the memory array also contains a link address which points to the next entry in the array. This allows a non-contiguous memory array. The last entry contains a link address set to zero. No base transfer count register is needed in this mode. The base address register is initialized to the address of the first entry in the array. The link address is

5

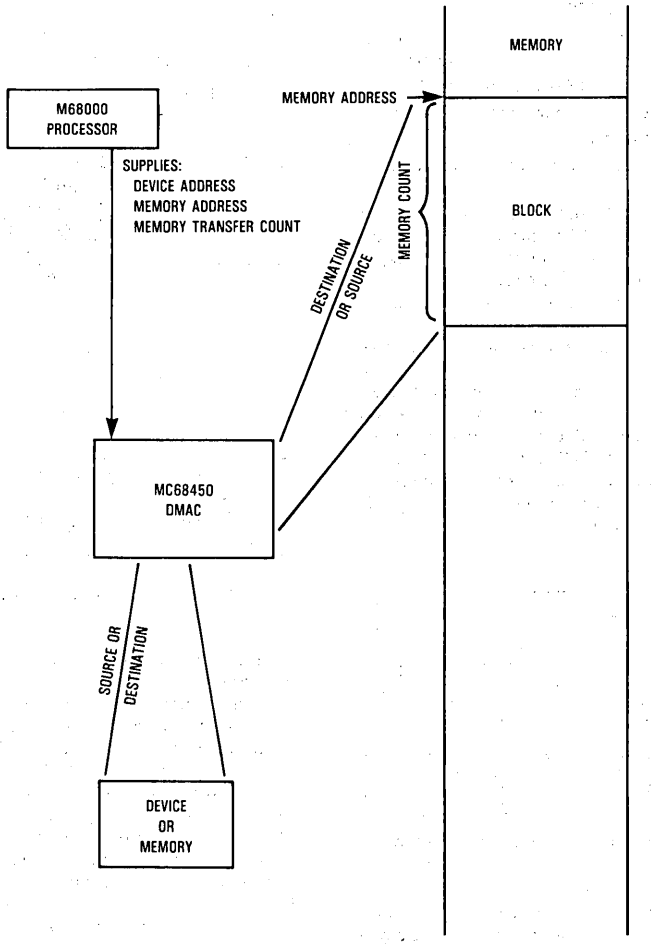


Figure 5. Single Block Transfer

used to update the base address register at the beginning of each block transfer. This chaining mode allows array entries to be easily moved or inserted without having to reorganize the array into sequential order. Also, the number of entries in the array need not be specified to the DMAC. See Figure 7.

INTERRUPT OPERATION

The DMAC will interrupt the MPU for a number of event occurrences such as the completion of a DMA operation, or at the request of a peripheral device using a PCL line. The user must write interrupt vectors into an on-chip vector register, for use in the M68000 vectored interrupt structure. Two vector registers are available for each channel.

CHANNEL PRIORITY

Each channel may be given a priority level of 0, 1, 2, or 3. If several channel requests occur at the same priority level, a round-robin is entered automatically.

REQUEST MODES

Requests may be externally generated by a device or internally generated by the auto-request mechanism of

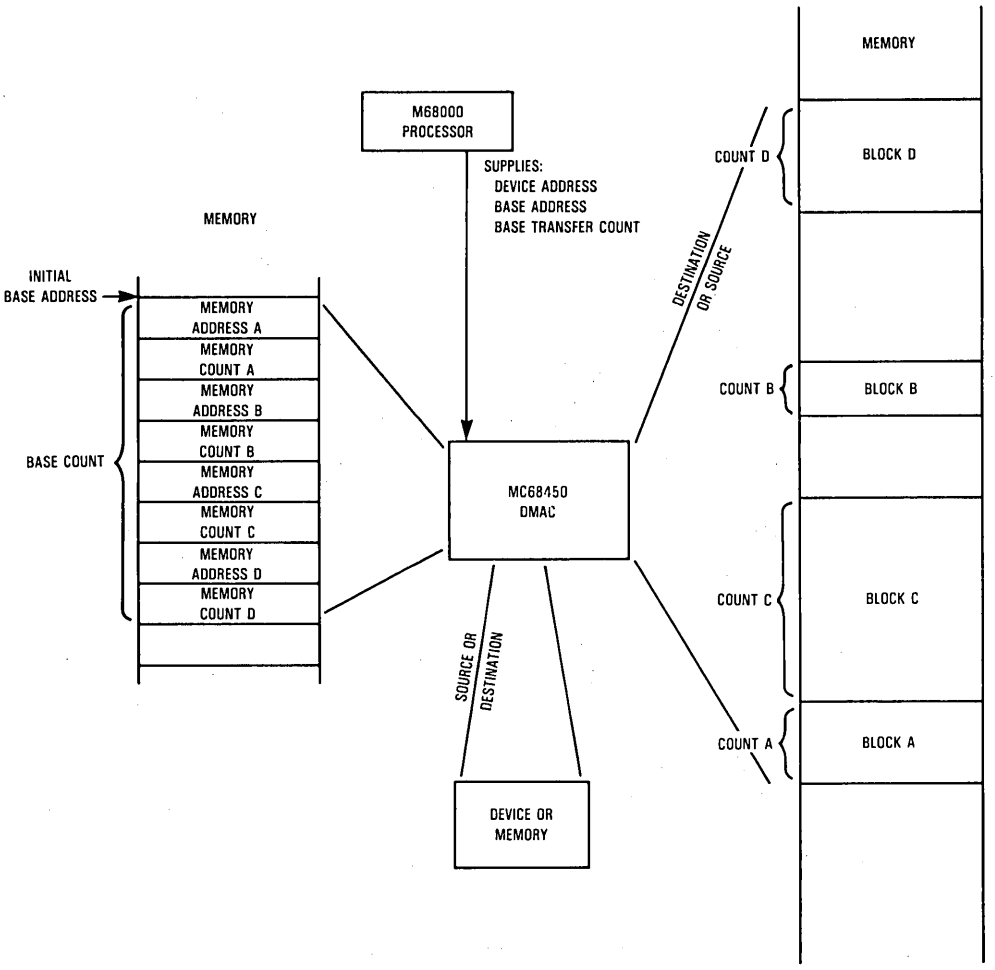


Figure 6. Array Chain Transfer

the DMAC. Auto-requests may be generated either at the maximum rate, where the channel always has a request pending, or at a limited rate determined by selecting a portion of the bus bandwidth to be available for DMA activity. External requests can be either burst requests or cycle steal requests that are generated by the request signal associated with each channel.

REGISTERS

The DMAC contains 17 registers for each of four channels plus one general control register, all of which are under complete software control. The user programmer's model of the registers is shown in Figure 8.

The DMAC registers contain information about the data transfers such as the source and destination address and function codes, transfer count, operand size, device port size, channel priority, continuation address and transfer count, and the function of the peripheral control line. One register also provides status and error information on channel activity, peripheral inputs, and various events which may have occurred during a DMA transfer. A general control register selects the bus utilization factor to be used in limited rate auto-request DMA operations.

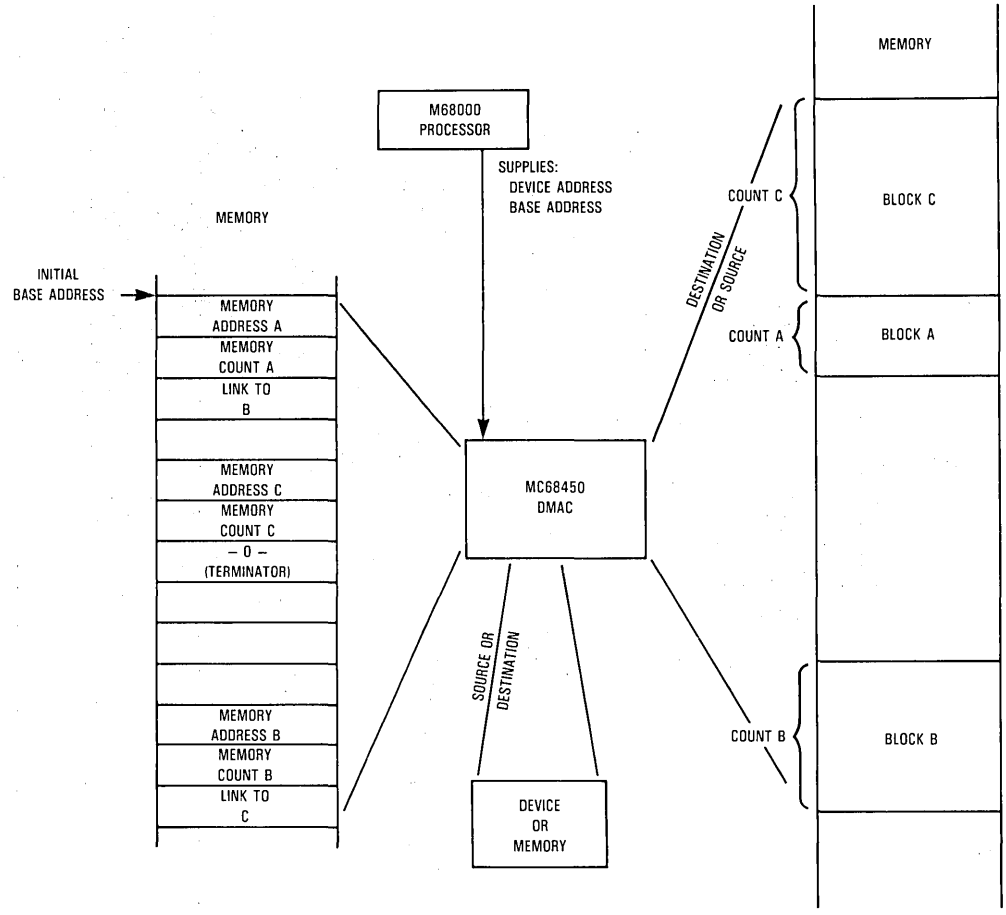


Figure 7. Linked Array Chain Transfer

SIGNAL DESCRIPTION

The following paragraphs contain a brief description of the DMAC input and output signals. Included at the end of the functional description of the signals is a table describing the electrical characteristics of each pin (i.e., the type of driver used).

NOTE

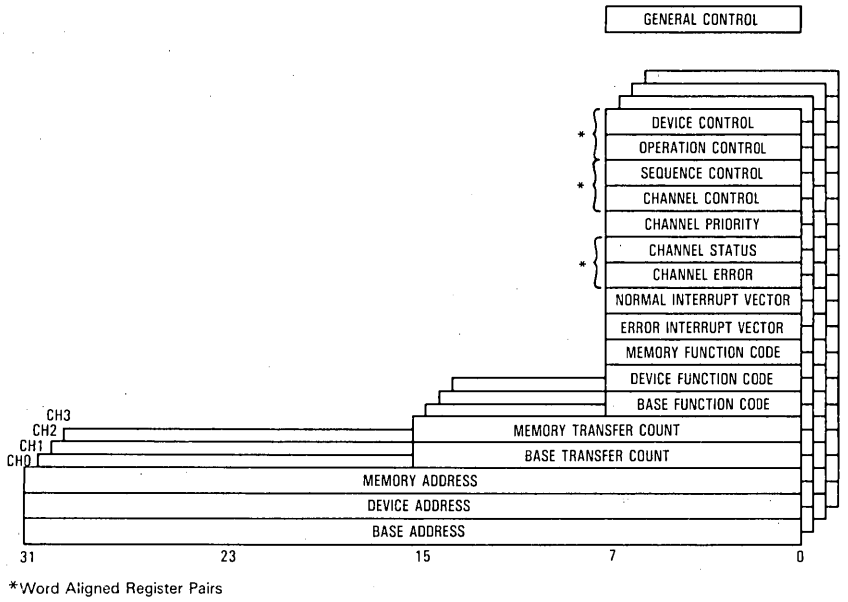
The terms **assertion** and **negation** will be used extensively. This is done to avoid confusion when dealing with a mixture of "active-low" and "active-high" signals. The term assert or assertion is used to indicate that a signal is active or true, independent of whether that level is represented by a high or low voltage. The term negate or negation is used to indicate that a signal is inactive or false.

SIGNAL ORGANIZATION

The input and output signals can be functionally organized into the groups shown in Figure 9. The function of each signal or group of signals is discussed in the following paragraphs.

Address/Data Bus (A8/D0 through A23/D15)

This 16-bit bus is time multiplexed to provide address outputs during the DMA mode of operation and is used as a bidirectional data bus to input data from an external device (during an MPU write or DMA read) or to output



*Word Aligned Register Pairs

Figure 8. DMAC Programmer's Model

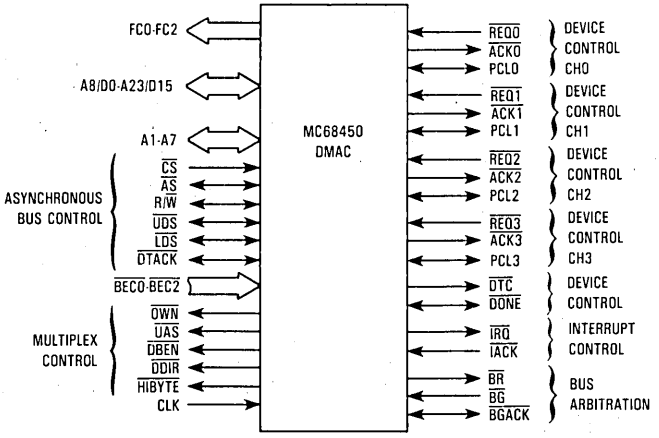


Figure 9. Signal Organization

data to an external device (during an MPU read or a DMA write). This is a three-state bus and is demultiplexed using external latches and buffers controlled by the multiplex control lines.

Lower Address Bus (A1 through A7)

These bidirectional three-state lines are used to address the DMAC internal registers in the MPU mode and to provide the lower seven address outputs in the DMA mode.

Function Codes (FC0 through FC2)

These three-state output lines are used in the DMA mode to further qualify the value on the address bus to provide eight seperate address spaces that may be defined by the user. The value placed on these lines is taken from one of the internal function code registers, depending on the register that provides the address used during a DMA bus cycle.

Asynchronous Bus Control

Asynchronous data transfers are handled using the following control signals: chip select, address strobe, read/write, upper and lower data strobes, and data transfer acknowledge. These signals are described in the following paragraphs.

5

CHIP SELECT ($\overline{CS}$). This input signal is used to select the DMAC for an MPU bus cycle. When $\overline{CS}$ is asserted, the address on A1-A7 and the data strobes (or A when using an 8-bit bus) select the internal DMAC register that will be involved in the transfer. $\overline{CS}$ should be generated by qualifying an address decode signal with the address and data strobes.

ADDRESS STROBE ($\overline{AS}$). This bidirectional signal is used as an output in the DMA mode to indicate that a valid address is present on the address bus. In the MPU or IDLE modes, it is used as an input to determine when the DMAC can take control of the bus (if the DMAC has requested and been granted use of the bus).

READ/WRITE (R/$\overline{W}$). This bidirectional signal is used to indicate the direction of a data transfer during a bus cycle. In the MPU mode, a high level indicates that a transfer is from the DMAC to the data bus and a low level indiates a transfer from the data bus to the DMAC. In the DMA mode, a high level indicates a transfer from the addressed memory or device to the data bus, and a low level indicates a transfer from the data bus to the addressed memory or device.

UPPER AND LOWER DATA STROBE ($\overline{UDS}$) AND ($\overline{LDS}$). These bidirectional lines indicate when data is valid on the bus and what portions of the bus should be involved in the transfer.

DATA TRANSFER ACKNOWLEDGE ($\overline{DTACK}$). This bidirectional line is used to signal that an asynchronous bus cycle may be terminated. In the MPU mode, this output indicates that the DMC has accepted data from the MPU or placed data on the bus for the MPU. In the DMA mode, this input is monitored by the DMAC to determine when to terminate a bus cycle. As long as $\overline{DTACK}$ remains negated, the DMAC will insert wait cycles into a bus cycle and when $\overline{DTACK}$ is asserted, the bus cycle will be terminated (except when PCL is used as a ready signal, in which case both signals must be asserted before the cycle is terminated).

BUS EXCEPTION CONTROL ($\overline{BEC0}$ THROUGH $\overline{BEC2}$). These input lines provide an encoded signal that indicates an abnormal bus condition such as a bus error or reset.

Multiplex Control

These signals are used to control external multiplex/demultiplex devices to separate the address and data information on the A8/D0-A23/D15 lines and to transfer data between the upper and lower halves of the data bus during certain DMA bus cycles.

Figure 10 shows the five external devices needed to demultiplex the address/data pins and the interconnection of the multiplex control signals. The SN74LS245 that may connect the upper and lower halves of the data bus is needed only if an 8-bit device is used during single address transfers.

OWN ($\overline{OWN}$). This three-state output indicates that the DMAC is controlling the bus. It is used as the enable signal to turn on the external address drivers and control signal buffers.

UPPER ADDRESS STROBE ($\overline{UAS}$). This three-state output is used as the gate signal to the transparent latches that capture the value of A8-A23 on the multiplexed address/data bus.

DATA BUFFER ENABLE ($\overline{DBEN}$). This three-state output is used as the enable signal to the external bidirectional data buffers.

DATA DIRECTION ($\overline{DDIR}$). This three-state output controls the direction of the external bidirectional data buffers. If $\overline{DDIR}$ is high, the data transfer is from the DMAC to the data bus. If $\overline{DDIR}$ is low, the data transfer is from the data bus to the DMAC.

HIGH BYTE ($\overline{HIBYTE}$). This three-state output indicate that data will be present on data lines D8-D15 that must be transferred to data lines D0-D7, or vice versa, through an external buffer during a single address transfer between an 8-bit device and memory.

Bus Arbitration Control

These three signals form a bus arbitration circuit used to determine which device in a system will be the current bus master.

BUS REQUEST ($\overline{BR}$). This output is asserted by the DMAC to request control of the bus.

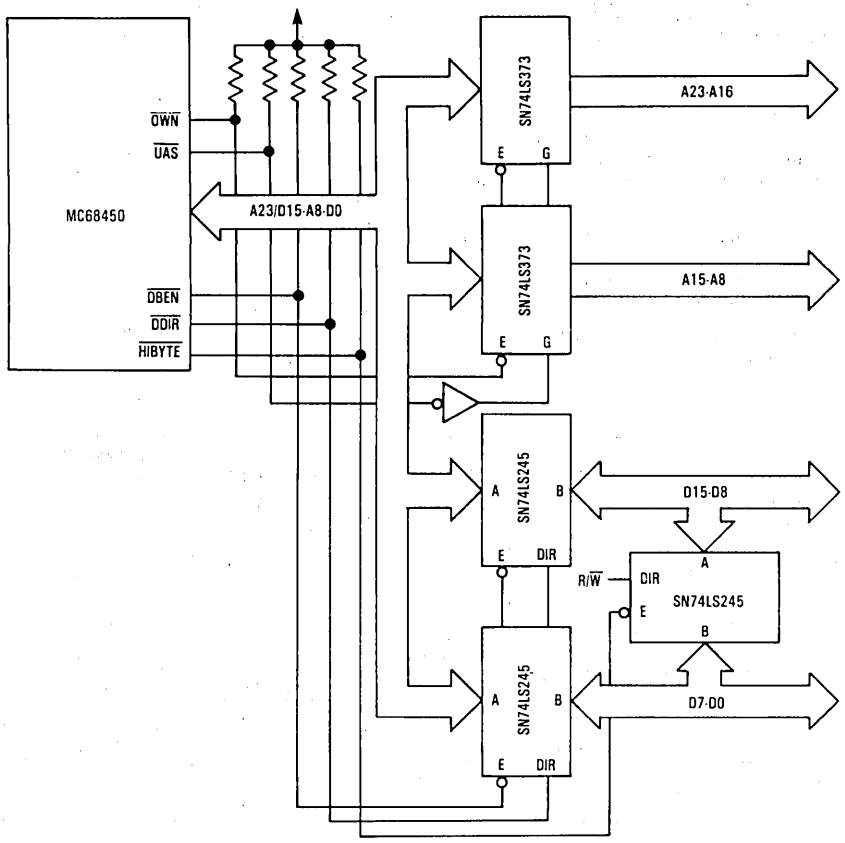


Figure 10. Demultiplex Logic

BUS GRANT ($\overline{BG}$). This input is asserted by an external bus arbiter to inform the DMAC that it may assume bus mastership as soon as the current bus cycle is completed. The DMAC will not take control of the bus until $\overline{CS}$, $\overline{IACK}$, $\overline{AS}$, and $\overline{BGACK}$ are all negated.

BUS GRANT ACKNOWLEDGE ($\overline{BGACK}$). This bidirectional signal is asserted by the DMAC to indicate that it is the current bus master. $\overline{BGACK}$ is monitored as an input to determine when the DMAC can become bus master and if a bus master other than the system MPU is a master during limited rate auto-request operation.

Interrupt Control

These two signals form an interrupt request/acknowledge handshake circuit with an MPU.

INTERRUPT REQUEST ($\overline{IRQ}$). This output is asserted by the DMAC to request service from the MPU.

INTERRUPT ACKNOWLEDGE ($\overline{IACK}$). This input is asserted by the MPU to acknowledge that it has received an interrupt from the DMAC. In response to the assertion of $\overline{IACK}$, the DMAC will place a vector on D0-D7 that will be used by the MPU to fetch the address of the proper DMAC interrupt handler routine.

Device Control

These eight lines perform the interface between the DMAC and four peripheral devices. Four sets of three lines are dedicated to a single DMAC channel and its associated peripheral; the remaining two lines are global signals shared by all channels.

REQUEST ($\overline{REQ0}$ THROUGH $\overline{REQ3}$). These inputs are asserted by a peripheral device to request an operand transfer between that peripheral device and memory. In the cycle steal request generation mode, these inputs are edge sensitive; in burst mode, they are level sensitive.

ACKNOWLEDGE ($\overline{ACK0}$ THROUGH $\overline{ACK3}$). These outputs are asserted by the DMAC to signal to a peripheral that an operand is being transferred in response to a previous transfer request.

PERIPHERAL CONTROL LINE (PCL0 THROUGH PCL3). These bidirectional lines are multi-purpose signals that may be programmed to function as ready, abort, reload, status, interrupt, or enable clock inputs or as start pulse outputs.

DATA TRANSFER COMPLETE ($\overline{DTC}$). This output is asserted by the DMAC during any DMAC bus cycle to indicate that data has been successfully transferred (i.e., the bus cycle was not terminated abnormally).

DONE ($\overline{DONE}$). This bidirectional signal is asserted by the DMAC or a peripheral device during DMA bus cycle to indicate that the data being transferred is the last item in a block. The DMAC will assert this signal during a bus cycle when the memory transfer count register is decremented to zero.

Clock (CLK)

The clock input is a TTL-compatible signal that is internally buffered for development of the internal clocks needed by the DMAC. The clock input should not be gated off at any time and the clock signal must conform to minimum and maximum pulse width times.

SIGNAL SUMMARY

Table 1 is a summary of all the signals discussed in the previous paragraphs.

Table 1. Signal Summary

Signal Name	Direction	Active State	Driver Type	Synchronizing Clock Edge
A8/D0-A23/D15	In/Out	High	Three State	Falling**
A1-A7	In/Out	High	Three State	Falling**
FC0-FC2	Out	High	Three State	
$\overline{CS}$	In	Low		Falling
$\overline{AS}$	In/Out	Low	Three State*	Falling
R/$\overline{W}$	In/Out	High/Low	Three State*	Falling
$\overline{UDS}$	In/Out	Low/High	Three State*	Falling
$\overline{LDS}$	In/Out	Low	Three State*	Falling
$\overline{DTACK}$	In/Out	Low	Three State*	Rising
$\overline{OWN}$	Out	Low	Open Drain*	
$\overline{UAS}$	Out	Low	Three State*	
$\overline{DBEN}$	Out	Low	Three State*	
$\overline{DDIR}$	Out	High/Low	Three State*	
$\overline{HIBYTE}$	Out	Low	Three State*	
$\overline{BEC0}$-$\overline{BEC2}$	In	Low		Rising
$\overline{BR}$	Out	Low	Open Drain	
$\overline{BG}$	In	Low		Rising
$\overline{BGACK}$	In/Out	Low	Open Drain*	
$\overline{IRQ}$	In	Low		
$\overline{IACK}$	In	Low		Falling
$\overline{REQ0}$-$\overline{REQ3}$	In	Low		Rising
$\overline{ACK0}$-$\overline{ACK3}$	Out	Low	Always Driven	
PCL0-PCL3	In/Out	Programmed	Three State	Rising
$\overline{DTC}$	Out	Low	Three State*	
$\overline{DONE}$	In/Out	Low	Open Drain	Rising
CLK	In			

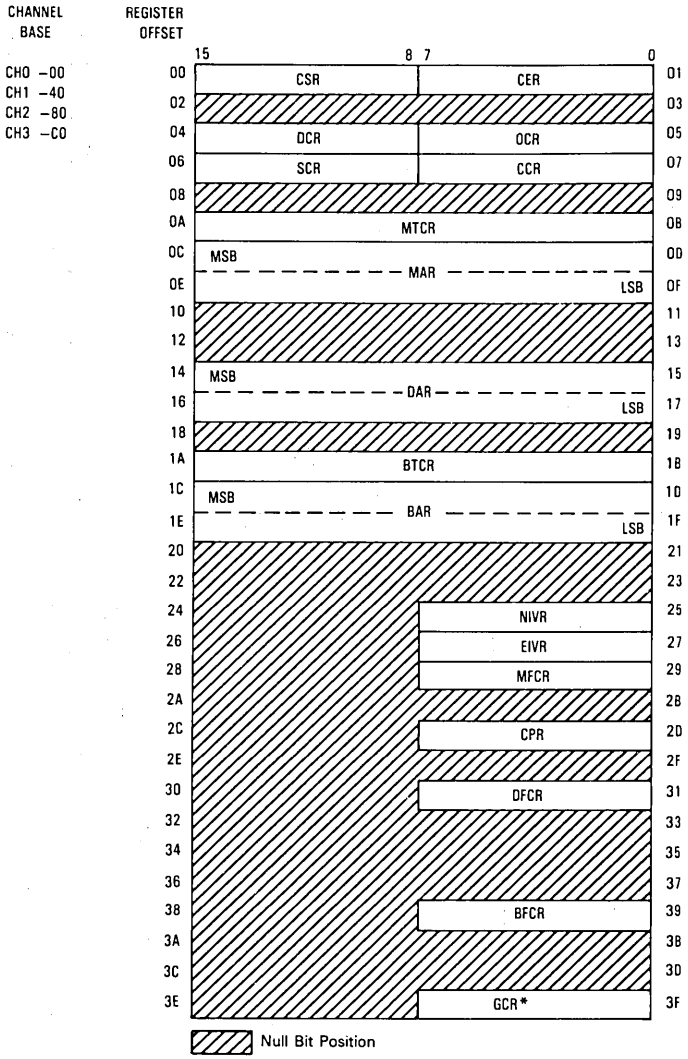
*These signals require a pullup resistor to maintain a high voltage when in the high-impedance or negated state. When these signals go to the high-impedance or negated state, they will first drive the pin high momentarily to reduce the signal rise time.

**These signals are latched on a clock edge, but are not synchronized (i.e., the latched value is used immediately, rather than delayed by one clock).

REGISTER DESCRIPTION

Figure 11 shows the memory mapped locations of the registers for each channel. Figure 12 shows the register summary and may be used as a quick reference to the bit definitions within each register.

The register memory map for the MC68450 DMAC is identical to the register memory map for the MC68440 Dual Channel DMA Controller (DDMA), including the individual bit assignments within the registers. However, not all functional options available on the DMAC are available on the DDMA and vice versa. If any programmable options labeled "MC68440 Reserved" or "Undefined, Reserved" are programmed into a DMAC channel, a configuration error will occur when the MPU attempts to start that channel.



*The GCR is located at FF only.

Figure 11. Register Memory Map

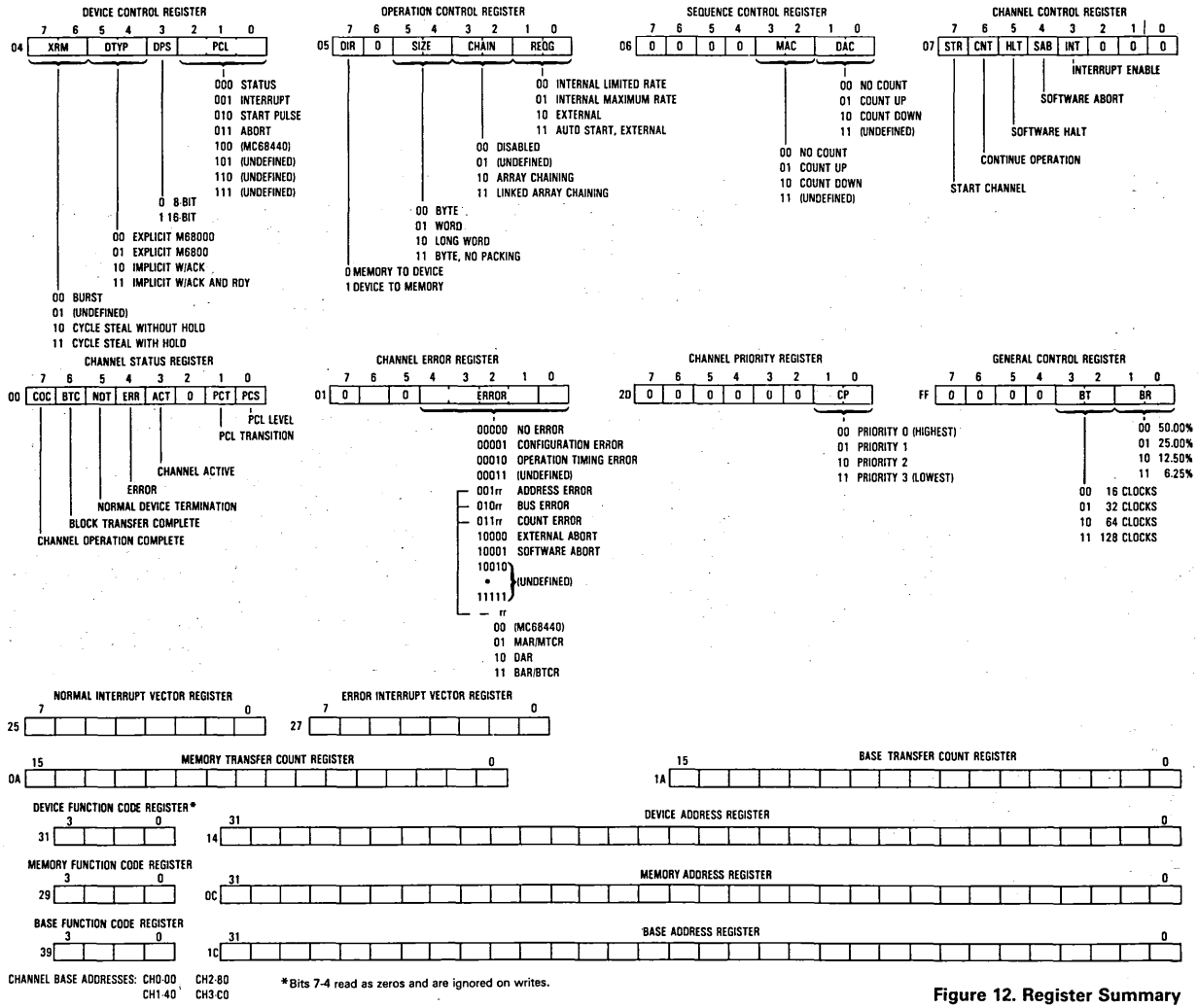


Figure 12. Register Summary

All registers within the DMAC are always accessible as bytes or words by the MPU (assuming that the MPU can gain control of the DMA bus); however, some registers may not or should not be modified while a channel is actively transferring data. If a register may not be modified during operation and an attempt is made to write to it, an operation timing error will be signaled and the channel operation aborted.

RESET OPERATION RESULTS

When the DMAC is reset, either during a system power-up sequence or to re-initialize the DMAC, many of the registers will be affected and will be set to known values. Table 2 shows the hexadecimal value that will be placed in each register by a reset operation.

Table 2. Reset Operation Results

Register	Value	Comments
MARc	XXXXXXXX	Not Affected
DARc	XXXXXXXX	Not Affected
BARc	XXXXXXXX	Not Affected
MFCRc	X	Not Affected
DFCRc	X	Not Affected
BFCRc	X	Not Affected
MTCRc	XXXX	Not Affected
BRCRc	XXXX	Not Affected
NIVRc	OF	Uninitialized Vector
EIVRc	OF	Uninitialized Vector
CPRc	00	
DCRc	00	
OCRc	00	
SCRc	00	
CCRc	00	Channel Not Active, Interrupts Disabled
CSRc	00 or 01	(Depending on the Level of PCLc)
CERc	00	No Errors
GCR	00	

X—Indicates an unknown value or the previous value of the register.
c—is the channel number (i.e., 0, 1, 2, or 3)

5

ELECTRICAL SPECIFICATIONS

ABSOLUTE MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Supply Voltage	V_{CC}	−0.3 to +7.0	V
Input Voltage	V_{in}	−0.3 to +7.0	V
Operating Temperature Range	T_A	0 to +70	°C
Storage Temperature	T_{stg}	−55 to +150	°C

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

THERMAL CHARACTERISTICS

Characteristic	Value		Rating
	θ_{JA}	θ_{JC}	
Thermal Resistance (Still Air)			°C/W
Ceramic, (L/LC)	30	15*	
Plastic, (P)	30	15*	
Pin Grid Array (R/RC)	30	15	

POWER CONSIDERATIONS

The average chip-junction temperature, T_J, in °C can be obtained from:

$$T_J = T_A + (P_D \cdot \theta_{JA}) \quad (1)$$

where:

T_A = Ambient Temperature, °C
θ_{JA} = Package Thermal Resistance, Junction-to-Ambient, °C/W
$P_D = P_{INT} + P_{I/O}$
$P_{INT} = I_{CC} \times V_{CC}$, 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.

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

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

Solving equations (1) and (2) for K gives:

$$K = P_D \cdot (T_A + 273°C) + \theta_{JA} \cdot P_D^2 \quad (3)$$

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.

The total thermal resistance of a package (θ_{JA} can be separated into two components, θ_{JA} and θ_{CA}, representing the barrier to heat flow from the semiconductor junction to the package (case) surface (θ_{JC}) and from the case to the outside ambient (θ_{CA}). These terms are related by the equation:

$$\theta_{JA} = \theta_{JC} + \theta_{CA} \quad (4)$$

θ_{JC} is device related and cannot be influenced by the user. However, θ_{CA} is user dependent and can be minimized by such thermal management techniques as heat sinks, ambient air cooling and thermal convection. Thus, good thermal management on the part of the user can significantly reduce θ_{CA} so that θ_{JA} approximately equals θ_{JC}. Substitution of θ_{JC} for θ_{JA} in equation (1) will result in a lower semiconductor junction temperature.

Values for thermal resistance presented in this document, unless estimated, were derived using the procedure described in Motorola Reliability Report 7843, "Thermal Resistance Measurement Method for MC68XX Microcomponent Devices," and are provided for design purposes only. Thermal measurements are complex and dependent on procedure and setup. User derived values for thermal resistance may differ.

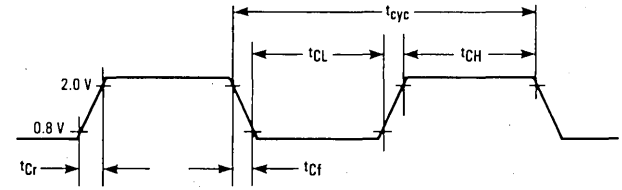
5

DC ELECTRICAL CHARACTERISTICS (T_A = 0°C to 70°C; V_{CC} = 5.0 V ±5%)

Characteristic		Symbol	Min	Max	Unit
Input High Voltage		V_{IH}	2.0	V_{CC}	V
Input Low Voltage		V_{IL}	GND −0.3	0.8	V
Input Leakage Current	CS, IACK, BG, CLK, BEC0-BEC2, REQ0-REQ3	I_{in}	—	10	μA
Three-State (Off-State) Input Current	A1-A7, D0/A8-D15/A23, AS, UDS, LDS, R/W, UAS, DTACK, BGACK, OWN, DTC, HIBYTE, DDIR, DBEN, FC0-FC2, PCL0-PCL3	I_{TSI}	—	20	μA
Input Capacitance (V_{in} = 0 V, T_A = 25°C, f = 1 MHz)		C_{in}	—	15	pF
Open-Drain (Off-State) Input Current	IRQ, DONE	I_{DD}	—	20	μA
Output High Voltage I_{OH} = −400 μA	A1-A7, D0/A8-D15/A23, AS, UDS, LDS, R/W, DTACK, BGACK, BR, OWN, DTC, HIBYTE, DDIR, DBEN, ACK0-ACK3, PCL0-PCL3, FC0-FC2	V_{OH}	2.4	—	V
Output Low Voltage I_{OL} = 3.2 mA I_{OL} = 5.3 mA I_{OL} = 8.9 mA	 A1-A7, FC0-FC2 D0/A8-D15/A23, AS, UDS, LDS, R/W, DTACK, BR, OWN, DTC, HIBYTE, DDIR, DBEN, ACK0-ACK3, UAS, PCL0-PCL3, BGACK IRQ, DONE	V_{OL}	—	0.5	V
Power Dissipation at 25°C (Frequency = 8 MHz)		P_D	—	2.0	W
Output Load Capacitance		C_L	—	130	pF

AC ELECTRICAL SPECIFICATIONS—CLOCK TIMING (see Figure 13)

Parameter	Symbol	8 MHz		10 MHz		Unit
		Min	Max	Min	Max	
Frequency of Operation	f	2	8	2	10	MHz
Clock Period	t_{cyc}	125	500	100	500	ns
Clock Width Low	t_{CL}	55	250	45	250	ns
Clock Width High	t_{CH}	55	250	45	250	ns
Clock Fall Time	t_{Cf}	—	10	—	10	ns
Clock Rise Time	t_{Cr}	—	10	—	10	ns



NOTE:
Timing measurements are referenced to and from a low voltage of 0.8 volt and a high voltage of 2.0 volts, unless otherwise noted. The voltage swing through this range should start outside and pass through the range such that the rise or fall will be linear between 0.8 volt and 2.0 volts.

Figure 13. Clock Input Timing Diagram

AC ELECTRICAL SPECIFICATIONS—READ AND WRITE CYCLES

(V_{CC} = 15 V ± 5%, GND = 0 V, T_A = 0°C to 70°C, unless otherwise noted) (see Figures 14 through 19)

No.	Parameter	Symbol	8 MHz		10 MHz		Unit
			Min	Max	Min	Max	
1	Clock Period	t_{cyc}	125	500	100	500	ns
2	Clock Width Low	t_{CL}	55	250	45	250	ns
3	Clock Width High	t_{CH}	55	257	45	250	ns
4	Clock Fall Time	t_{Cf}	—	10	—	10	ns
5	Clock Rise Time	t_{Cr}	—	10	—	10	ns
6	Asynchronous Input Setup Time	t_{ASI}	20	—	15	—	ns
7	Data In to $\overline{DBEN}$ Low	t_{DIDBL}	0	—	0	—	ns
8	$\overline{DTACK}$ Low to Data In Invalid	t_{DTLDI}	0	—	0	—	ns
9	Address In to $\overline{AS}$ In Low	t_{AIASL}	0	—	0	—	ns
10	$\overline{AS}$ In High to Address In Invalid	t_{SIHAIV}	0	—	0	—	ns
11	Clock High to $\overline{DDIR}$ Low	t_{CHDRL}	—	70	—	60	ns
12	Clock High to $\overline{DDIR}$ High	t_{CHDRH}	—	70	—	60	ns
13	$\overline{DS}$ In High to $\overline{DDIR}$ High Impedance	t_{DSHDRZ}	—	120	—	110	ns
14	Clock Low to $\overline{DBEN}$ Low	t_{CLDBL}	—	70	—	60	ns
15	Clock Low to $\overline{DBEN}$ High	t_{CLDBH}	—	70	—	60	ns
16	$\overline{DS}$ In High to $\overline{DBEN}$ High Impedance	t_{DSHDBZ}	—	120	—	110	ns
17	Clock High to Data Out Valid (MPU Read)	t_{CHDVM}	—	180	—	160	ns
18	$\overline{DS}$ In High to Data Out Invalid	t_{DSHDZn}	0	—	0	—	ns
19	$\overline{DS}$ In High to Data High Impedance	t_{DSHDZ}	—	120	—	110	ns
20	Clock Low to $\overline{DTACK}$ Low	t_{CLDTL}	—	70	—	60	ns
21	$\overline{DS}$ In High to $\overline{DTACK}$ High	t_{DSHDTH}	—	110	—	110	ns
22	$\overline{DTACK}$ Width High	t_{DTH}	10	—	10	—	ns
23	$\overline{DS}$ In High to $\overline{DTACK}$ High Impedance	t_{DSHDTZ}	—	180	—	160	ns
24	$\overline{DTACK}$ Low to $\overline{DS}$ In High	t_{DTLDSH}	0	—	0	—	ns
25	$\overline{REQ}$ Width Low	t_{REQL}	2.0	—	2.0	—	Clk. Per.
26	$\overline{REQ}$ Low to $\overline{BR}$ Low	t_{RELBRL}	250	—	200	—	ns
27	Clock High to $\overline{BR}$ Low	t_{CHBRL}	—	70	—	60	ns
28	Clock High to $\overline{BR}$ High	t_{CHBRH}	—	70	—	60	ns
29	$\overline{BG}$ Low to $\overline{BGACK}$ Low	t_{BGLBL}	4.5	—	4.5	—	Clk. Per.
31	MPU Cycle End ($\overline{AS}$ In High) to $\overline{BGACK}$ Low	t_{ASHBL}	4.5	5.5	4.5	5.5	Clk. Per.
32	$\overline{REQ}$ Low to $\overline{BGACK}$ Low	t_{REQLBL}	12	—	12	—	Clk. Per.
33	Clock High to $\overline{BGACK}$ Low	t_{CHBL}	—	70	—	60	ns
34	Clock High to $\overline{BGACK}$ High	t_{CHBH}	—	70	—	60	ns
35	Clock Low to $\overline{BGACK}$ High Impedance	t_{CLBZ}	—	80	—	70	ns
36	Clock High to FC Valid	t_{CHFCV}	—	100	—	90	ns
37	Clock High to Address Valid	t_{CHAV}	—	120	—	110	ns
38	Clock High to Address/FC/Data High Impedance	t_{CHAZx}	—	100	—	100	ns
39	Clock High to Address/FC/Data Invalid	t_{CHAZn}	0	—	0	—	ns
40	Clock Low to Address High Impedance (Read)	t_{CLAZ}	—	100	—	90	ns

AC ELECTRICAL SPECIFICATIONS — READ AND WRITE CYCLES (Continued)

No.	Parameter	Symbol	8 MHz		10 MHz		Unit
			Min	Max	Min	Max	
41	Clock High to $\overline{UAS}$ Low	t_{CHUL}	—	70	—	60	ns
42	Clock High to $\overline{UAS}$ High	t_{CHUH}	—	70	—	60	ns
43	Clock Low to $\overline{UAS}$ High Impedance	t_{CLUZ}	—	80	—	70	ns
44	$\overline{UAS}$ High to Address Invalidance	t_{UHAI}	30	—	20	—	ns
45	Clock High to $\overline{AS}$, $\overline{DS}$ Low	t_{CHSL}	—	60	—	55	ns
46	Clock Low to $\overline{DS}$ Low (Write)	t_{CLDSL}	—	60	—	55	ns
47	Clock Low to $\overline{AS}$, $\overline{DS}$ High	t_{CLSH}	—	70	—	60	ns
48	Clock Low to $\overline{AS}$, $\overline{DS}$ High Impedance	t_{CLSZ}	—	80	—	70	ns
49	$\overline{AS}$ Width Low	t_{ASL}	255	—	195	—	ns
50	$\overline{DS}$ Width Low	t_{DSL}	255	—	190	—	ns
51	$\overline{AS}$, $\overline{DS}$ Width High	t_{SH}	150	—	105	—	ns
52	Address/FC Valid to $\overline{AS}$, $\overline{DS}$ Low (Read)	t_{AVSL}	30	—	20	—	ns
53	$\overline{AS}$, $\overline{DS}$ High to Address/FC/Data Invalid	t_{SHAZ}	30	—	20	—	ns
54	Clock High to R/$\overline{W}$ Low	t_{CHRL}	—	70	—	60	ns
55	Clock High to R/$\overline{W}$ High	t_{CHRH}	—	70	—	60	ns
56	Clock Low to R/$\overline{W}$ High Impedance	t_{CLRZ}	—	80	—	70	ns
57	Address/FC Valid to R/$\overline{W}$ Low	t_{AVRL}	20	—	10	—	ns
58	R/$\overline{W}$ Low to $\overline{DS}$ Low (Write)	t_{RLSL}	120	—	90	—	ns
59	$\overline{DS}$ High to R/$\overline{W}$ High	t_{SHRH}	40	—	20	—	ns
60	Clock Low to $\overline{OWN}$ Low	t_{CLOL}	—	70	—	60	ns
61	Clock Low to $\overline{OWN}$ High	t_{CLOH}	—	70	—	60	ns
62	Clock High to $\overline{OWN}$ High Impedance	t_{CHOZ}	—	80	—	70	ns
63	$\overline{OWN}$ Low to $\overline{BGACK}$ Low	t_{OLBL}	30	—	20	—	ns
64	$\overline{BGACK}$ High to $\overline{OWN}$ High	t_{BHOH}	30	—	20	—	ns
65	$\overline{OWN}$ Low to $\overline{UAS}$ Low	t_{OLUL}	30	—	20	—	ns
66	Clock High to $\overline{ACK}$ Low (Read)	t_{CHACL}	—	70	—	60	ns
67	Clock Low to $\overline{ACK}$ Low (Write)	t_{CLACL}	—	70	—	60	ns
68	Clock High to $\overline{ACK}$ High	t_{CHACH}	—	70	—	60	ns
69	$\overline{ACK}$ Low to $\overline{DS}$ Low	t_{ACLDSL}	100	—	80	—	ns
70	$\overline{DS}$ High to $\overline{ACK}$ High	t_{DSHACH}	30	—	20	—	ns
71	Clock High to $\overline{HIBYTE}$ Low	t_{CHHIL}	—	70	—	60	ns
72	Clock Low to $\overline{HIBYTE}$ Low	t_{CLHIL}	—	70	—	60	ns
73	Clock High to $\overline{HIBYTE}$ High	t_{CHHIH}	—	70	—	60	ns
74	Clock Low to $\overline{HIBYTE}$ High Impedance	t_{CLHIZ}	—	80	—	70	ns
75	Clock High to $\overline{DTC}$ Low	t_{CHDTL}	—	70	—	60	ns
76	Clock High to $\overline{DTC}$ High	t_{CHDTH}	—	70	—	60	ns
77	Clock Low to $\overline{DTC}$ High Impedance	t_{CLDTZ}	—	80	—	70	ns
78	$\overline{DTC}$ Width Low	t_{DTCL}	105	—	80	—	ns
79	$\overline{DTC}$ Low to $\overline{DS}$ High	t_{DTLDH}	30	—	20	—	ns

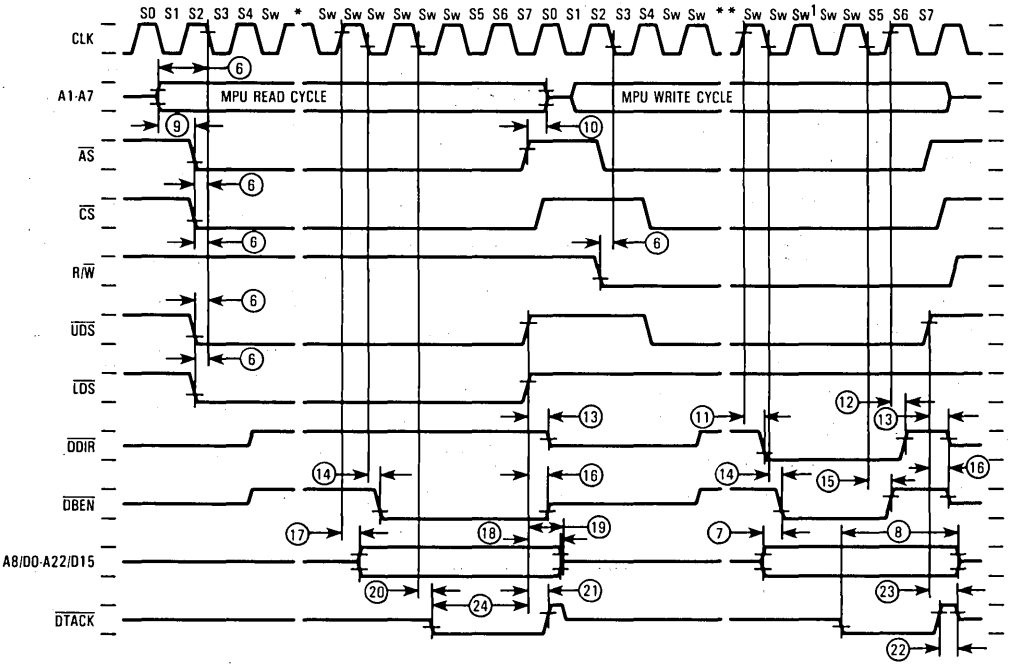
5

AC ELECTRICAL SPECIFICATIONS — READ AND WRITE CYCLES (Concluded)

No.	Parameter	Symbol	8 MHz		10 MHz		Unit
			Min	Max	Min	Max	
80	Clock High to $\overline{DONE}$ Low	t_{CHDOL}	—	70	—	60	ns
81	Clock Low to $\overline{DONE}$ Low	t_{CLDOL}	—	70	—	60	ns
82	Clock High to $\overline{DONE}$ High	t_{CHDOH}	—	130	—	120	ns
83	Clock High to $\overline{DDIR}$ High Impedance	t_{CLDRZ}	—	80	—	70	ns
84	Clock Low to $\overline{DBEN}$ High Impedance	t_{CLDBZ}	—	80	—	70	ns
85	$\overline{DDIR}$ Low to $\overline{DBEN}$ Low	t_{DRLDBL}	30	—	20	—	ns
86	$\overline{DBEN}$ High to $\overline{DDIR}$ High	t_{DBHDRH}	30	—	20	—	ns
87	$\overline{DBEN}$ Low to Address/Data High Impedance	t_{DBLAZ}	—	17	—	17	ns
88	Clock Low to PCL Low (1/8 Clock)	t_{CLPL}	—	70	—	60	ns
89	Clock Low to PCL High (1/8 Clock)	t_{CLPH}	—	70	—	60	ns
90	PCL Width Low (1/8 Clock)	t_{PCLL}	4.0	—	4.0	—	Clk. Per.
91	$\overline{DTACK}$ Low to Data In (Setup Time)	t_{DALDI}	—	150	—	115	ns
92	$\overline{DS}$ High to Data Invalid (Hold Time)	t_{SHDI}	0	—	0	—	ns
93	$\overline{DS}$ High to $\overline{DTACK}$ High	t_{SHDAH}	0	120	0	90	ns
94	Data Out Valid to $\overline{DS}$ Low	t_{DOSL}	0	—	0	—	ns
95	Data In to Clock Low (Setup Time)	t_{DICL}	15	—	15	—	ns
96	$\overline{BEC}$ Low to $\overline{DTACK}$ Low	t_{BECDAL}	50	—	50	—	ns
97	$\overline{BEC}$ Width Low	t_{BECL}	2.0	—	2.0	—	Clk. Per.
98	Clock High to $\overline{IRQ}$ Low	t_{CHIRL}	—	70	—	60	ns
99	Clock High to $\overline{IRQ}$ High Impedance	t_{CHIRH}	—	130	—	120	ns
100	PCL (as $\overline{READY}$) In to $\overline{DTC}$ Low (Read)	t_{RALDTL}	145	—	120	—	ns
101	PCL (as $\overline{READY}$) In to $\overline{DS}$ Low (Write)	t_{RALDSL}	205	—	170	—	ns
102	$\overline{DS}$ High to PCL (as $\overline{READY}$) High	t_{DSHRAH}	0	120	0	90	ns
103	$\overline{DONE}$ In Low to $\overline{DTACK}$ Low	t_{DOLDAL}	50	—	50	—	ns
104	$\overline{DS}$ High to $\overline{DONE}$ In High	t_{DSHDOH}	0	120	0	90	ns

5

These waveforms should only be referenced in regard to the edge-to-edge measurement of the timing specifications. They are not intended as a functional description of the input and output signals. Refer to other functional descriptions and their related diagrams for device operation.



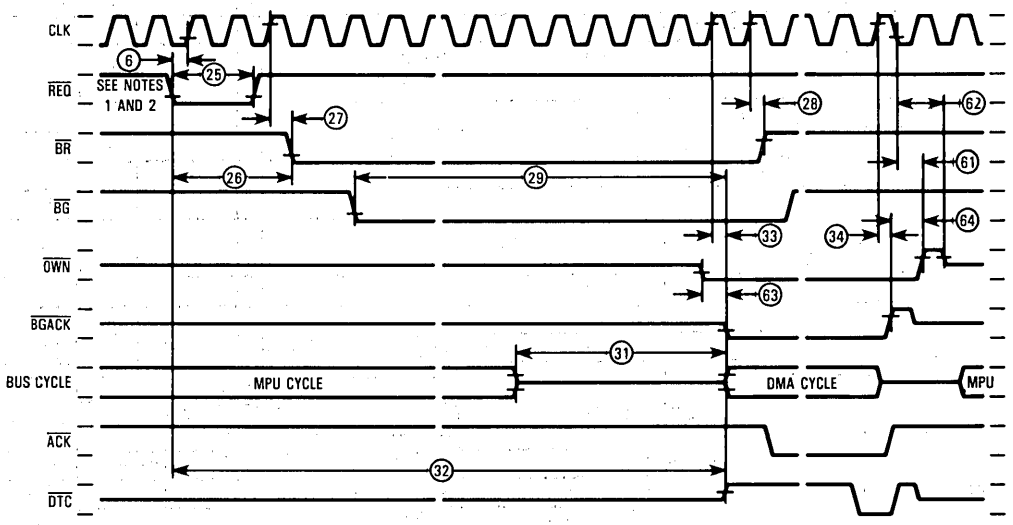
*10 Additional Wait Cycles
**8 Additional Wait Cycles

NOTES:
1. Data is latched at the end of this clock.
2. Timing measurements are referenced to and from a low voltage of 0.8 volt and a high voltage of 2.0 volts, unless otherwise noted.

Figure 14. MPU Read/Write Timing Diagram

5

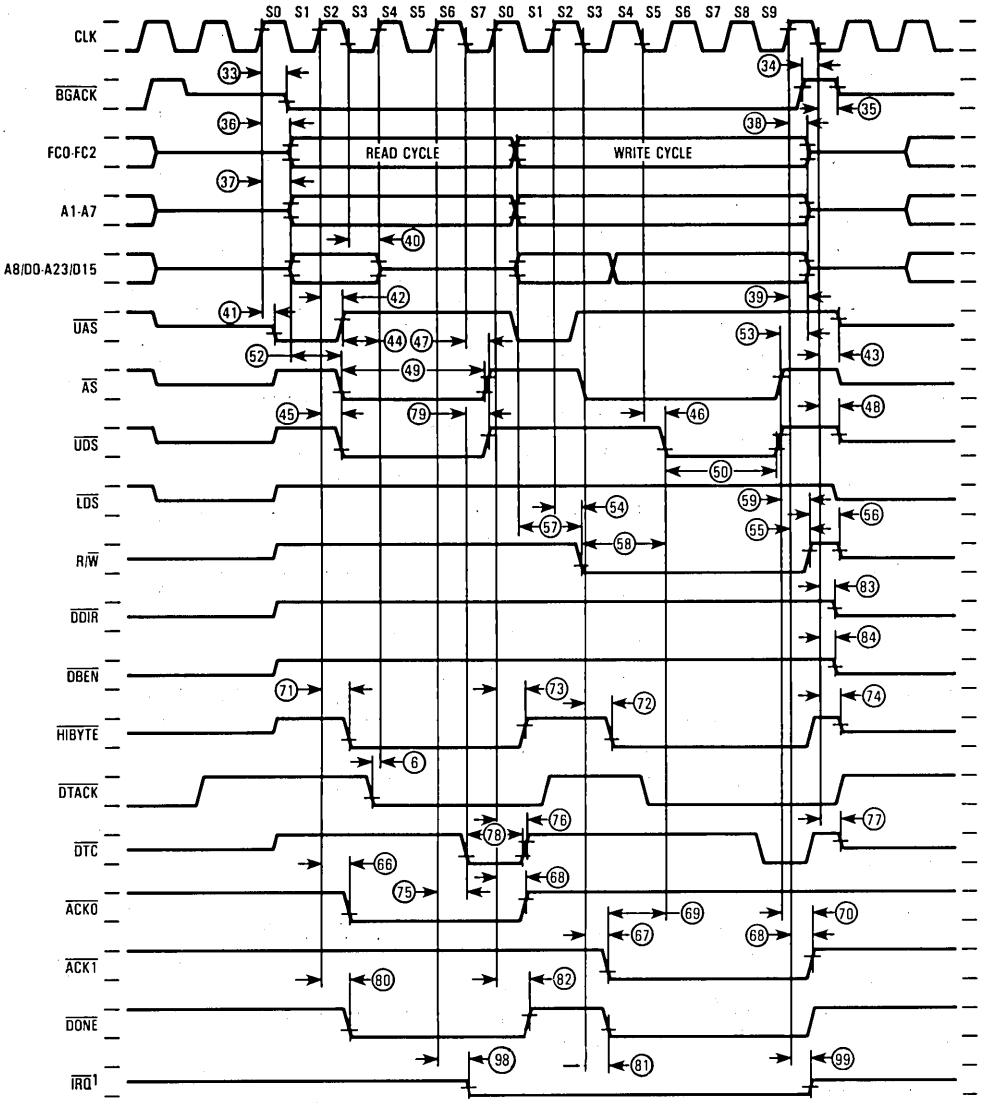
These waveforms should only be referenced in regard to the edge-to-edge measurement of the timing specifications. They are not intended as a functional description of the input and output signals. Refer to other functional descriptions and their related diagrams for device operation.



NOTES:
1. $\overline{REQ}$ is sampled on the rising edge of CLK in cycle steal and burst modes.
2. $\overline{BR}$ will not be asserted while any $\overline{BEC}$ exception condition exists, or when $\overline{CS}$ or $\overline{IACK}$ is asserted.
3. Timing measurements are referenced to and from a low voltage of 0.8 volt and a high voltage of 2.0 volts, unless otherwise noted.

Figure 15. Bus Arbitration Timing Diagram

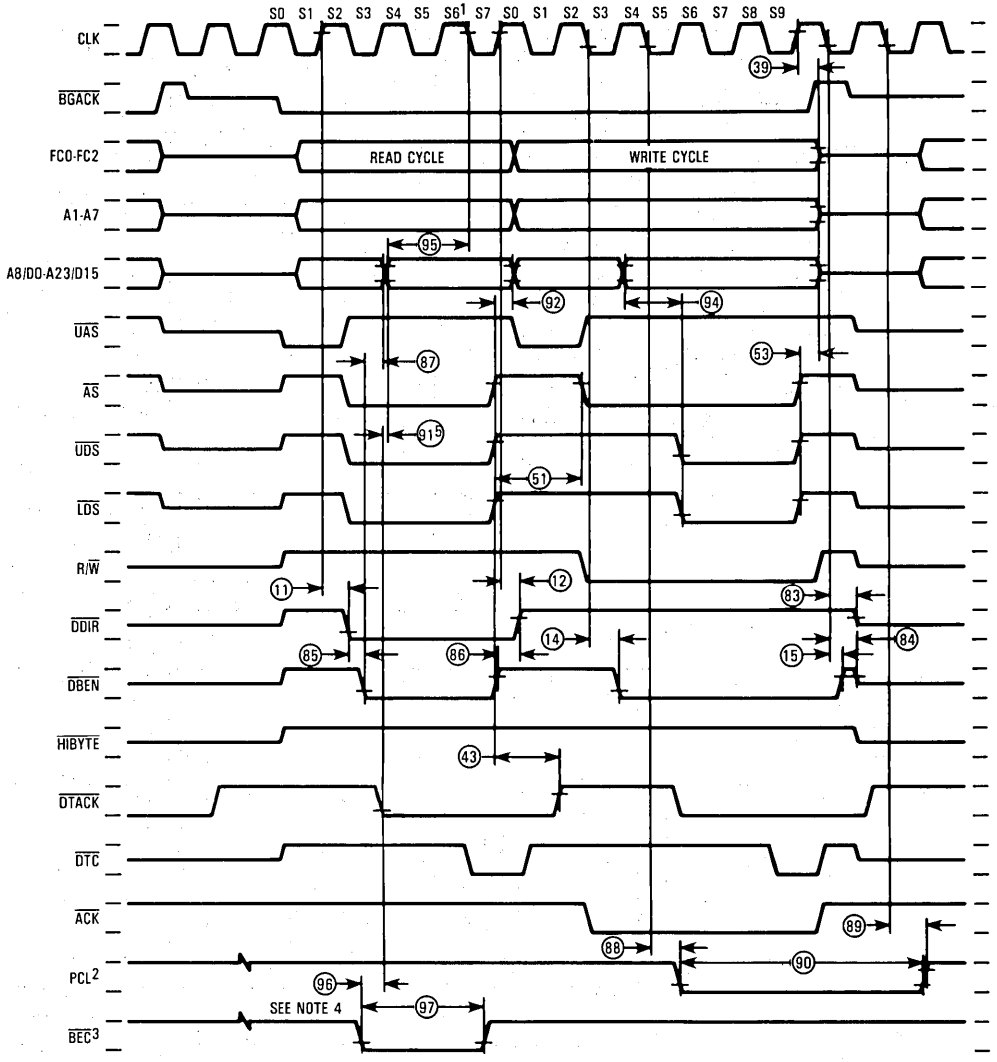
These waveforms should only be referenced in regard to the edge-to-edge measurement of the timing specifications. They are not intended as a functional description of the input and output signals. Refer to other functional descriptions and their related diagrams for device operation.



NOTES:
1. This timing is not directly related to the DMA read/write (single cycle) sequence.
2. Timing measurements are referenced to and from a low voltage of 0.8 volt and a high voltage of 2.0 volts, unless otherwise noted.

Figure 16. DMA Read/Write Timing Diagram (Single Cycle)

These waveforms should only be referenced in regard to the edge-to-edge measurement of the timing specifications. They are not intended as a functional description of the input and output signals. Refer to other functional descriptions and their related diagrams for device operation.

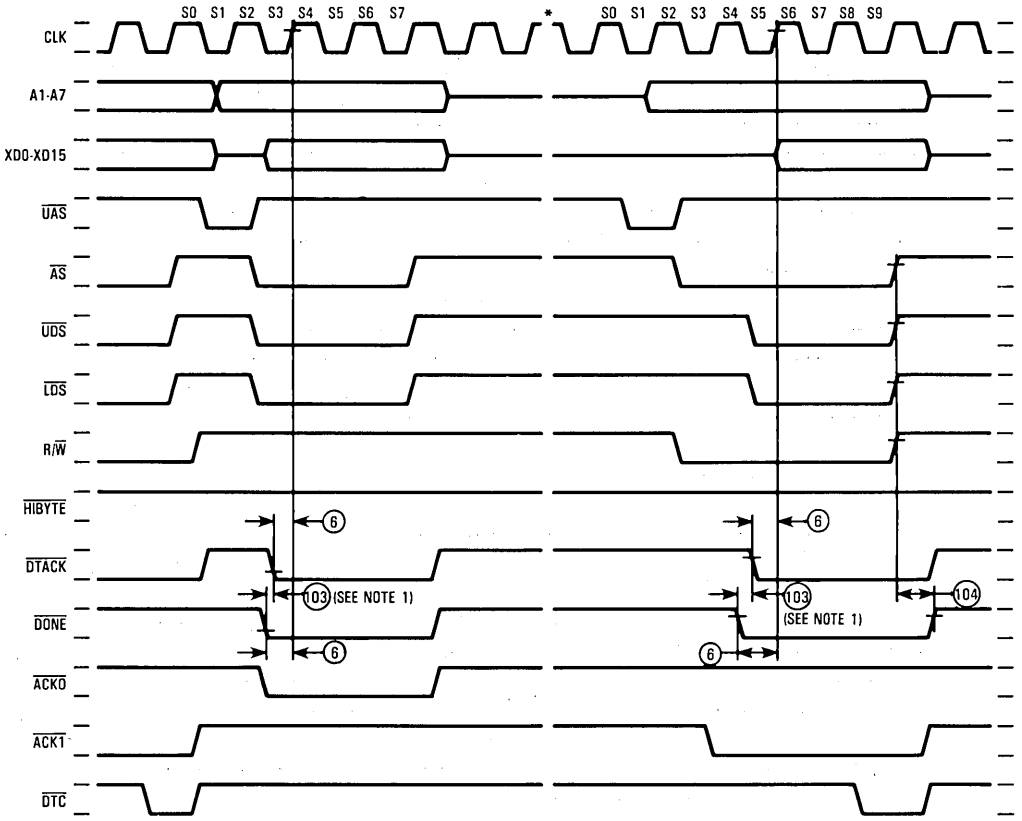


NOTES:

1. Data is latched at the end of clock S6.
2. Timing is not directly related to the DMA read/write (dual cycle) sequence and is only applicable when the start pulse mode is selected.
3. Timing is applicable when a bus exception occurs.
4. If specification number 6 is satisfied for both $\overline{DTACK}$ and $\overline{BEC}$, specification number 96 may be 0 nanoseconds.
5. If the propagation delay of the external bidirectional buffer is less than 17 nanoseconds, a conflict may occur between the address output of the DMAC and the system data bus. In this case, the output of $\overline{DBEN}$ must be delayed externally.
6. Timing measurements are referenced to and from a low voltage of 0.8 volt and a high voltage of 2.0 volts, unless otherwise noted.

Figure 17. DMA Read/Write Timing Diagram (Dual Cycle)

These waveforms should only be referenced in regard to the edge-to-edge measurement of the timing specifications. They are not intended as a functional description of the input and output signals. Refer to other functional descriptions and their related diagrams for device operation.

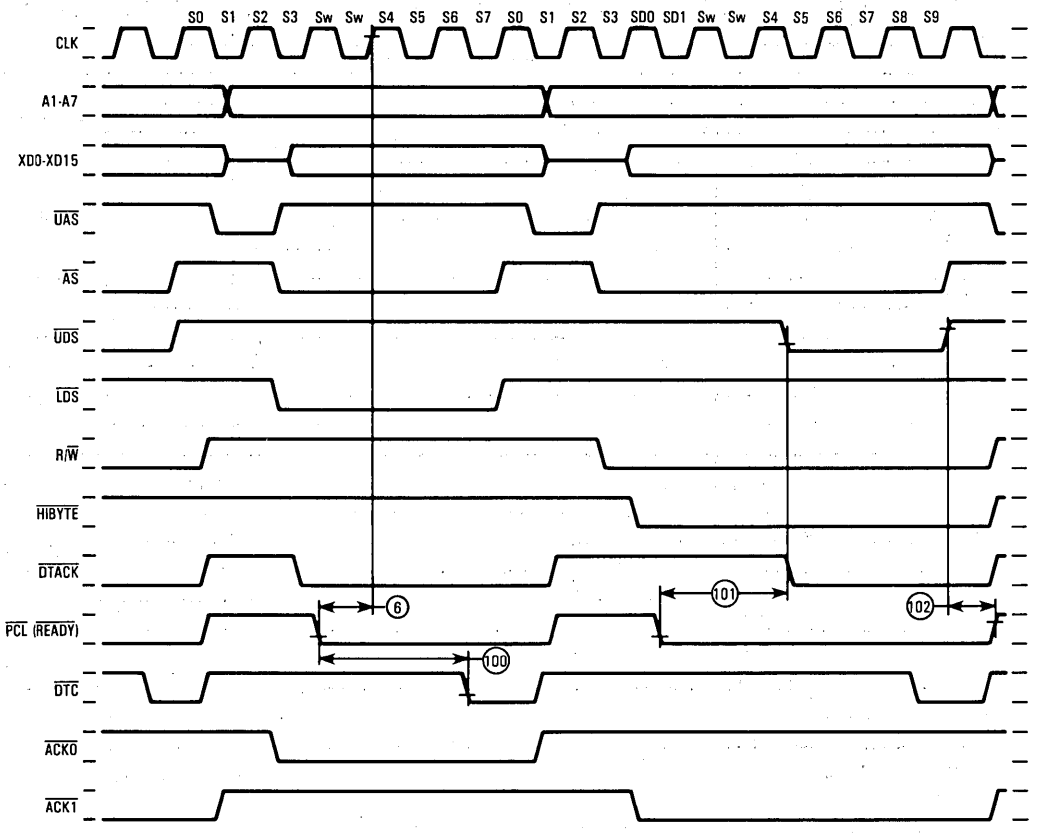


*3 to 9 Additional Clocks

NOTES:
1. If specification number 6 is satisfied for both DTACK and DONE, specification number 103 may be 0 nanoseconds.
2. The setup time for asynchronous inputs BG, BGACK, CS, IACK, AS, UDS, LDS, and R/W guarantees their recognition at the next falling edge of the clock. The setup time for BEC0-BEC2, REQ0-REQ3, PCL0-PCL3, DTACK, and DONE guarantees their recognition at the next rising edge of the clock.
3. Timing measurements are referenced to and from a low voltage of 0.8 volt and a high voltage of 2.0 volts, unless otherwise noted.

Figure 18. DONE Input Timing Diagram

These waveforms should only be referenced in regard to the edge-to-edge measurement of the timing specifications. They are not intended as a functional description of the input and output signals. Refer to other functional descriptions and their related diagrams for device operation.

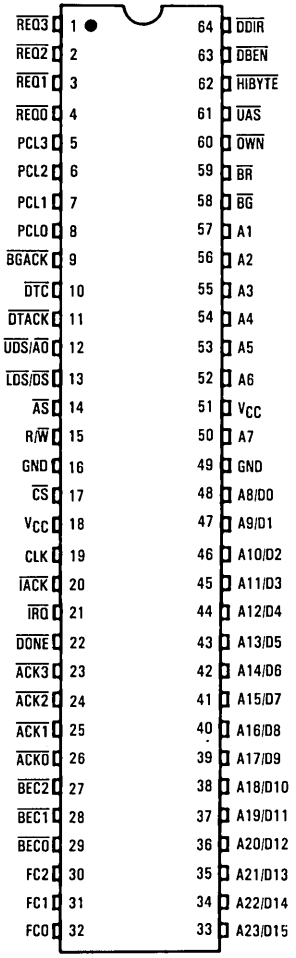


NOTE:
1. Timing measurements are referenced to and from a low voltage of 0.8 volt and a high voltage of 2.0 volts, unless otherwise noted.

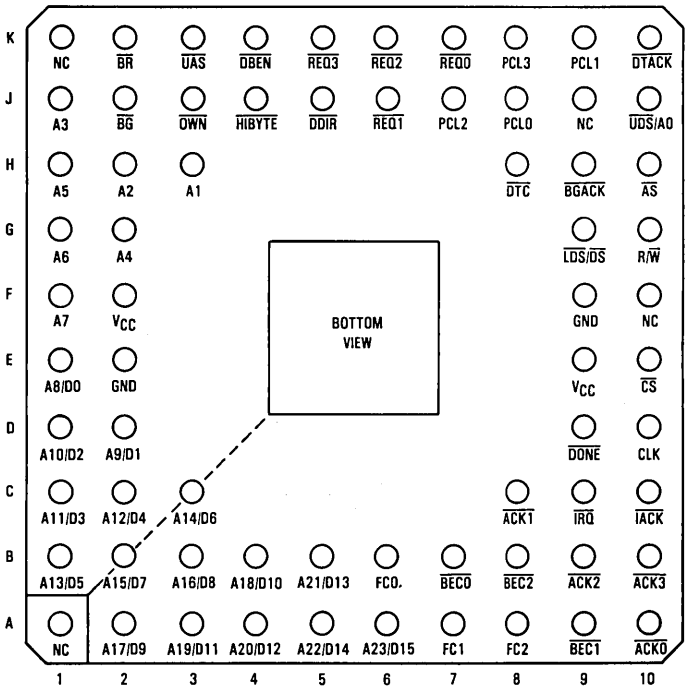
Figure 19. DMA Read/Write Timing Diagram (Single Cycle with PCL as $\overline{\text{READY}}$)

PIN ASSIGNMENTS

64-Pin Dual-In-Line Package



68-Terminal Pid-Grid Array



5

DATA COMMUNICATION DEVICES 6

MC2681

Technical Summary

Dual Asynchronous Receiver/Transmitter (DUART)

The MC2681 dual universal asynchronous receiver/transmitter (DUART) is part of the M68000 Family of peripherals and directly interfaces to the MC68000 processor via a general purpose interface which may be used with both synchronous and asynchronous microprocessors. This device has a multipurpose 7-bit input port and a multipurpose 8-bit output port. These ports can be used as general purpose I/O ports or can be assigned specific functions (such as clock inputs or status/interrupt outputs) under program control.

Figure 1 illustrates the basic block diagram of the MC2681 and should be referred to during the discussion of its features which include the following:

- Two Independent Full-Duplex Asynchronous Receiver/Transmitter Channels (A and B)
- Maximum Data Transfer
  - 1X – 1 MB/second
  - 16X – 125 kB/second
- Quadruple-Buffered Receiver Data Registers
- Double-Buffered Transmitter Data Registers
- Independently Programmable Baud Rate for Each Receiver and Transmitter Selectable From:
  - 18 Fixed Rates: 50 to 38.4k Baud
  - One User Defined Rate Derived from a Programmable Timer/Counter
  - External 1X Clock or 16X Clock
- Programmable Data Format
  - Five to Eight Data Bits plus Parity
  - Odd, Even, No Parity, or Force Parity (Low or High)
  - One, One and One-Half, or Two Stop Bits Programmable in One-Sixteenth Bit Increments
- Programmable Channel Modes
  - Normal (Full Duplex)
  - Automatic Echo
  - Local Loopback
  - Remote Loopback
- Automatic Wake-up Mode for Multidrop Applications
- Multi-Function 6-Bit Input Port
  - Can Serve as Clock or Control Inputs
  - Change-of-State Detection on Four Inputs
- Multi-Function 8-Bit Output Port
  - Individual Bit Set/Reset Capability
  - Outputs Can be Programmed to be Status/Interrupt Signals
- Multi-Function 16-Bit Programmable Counter/Timer

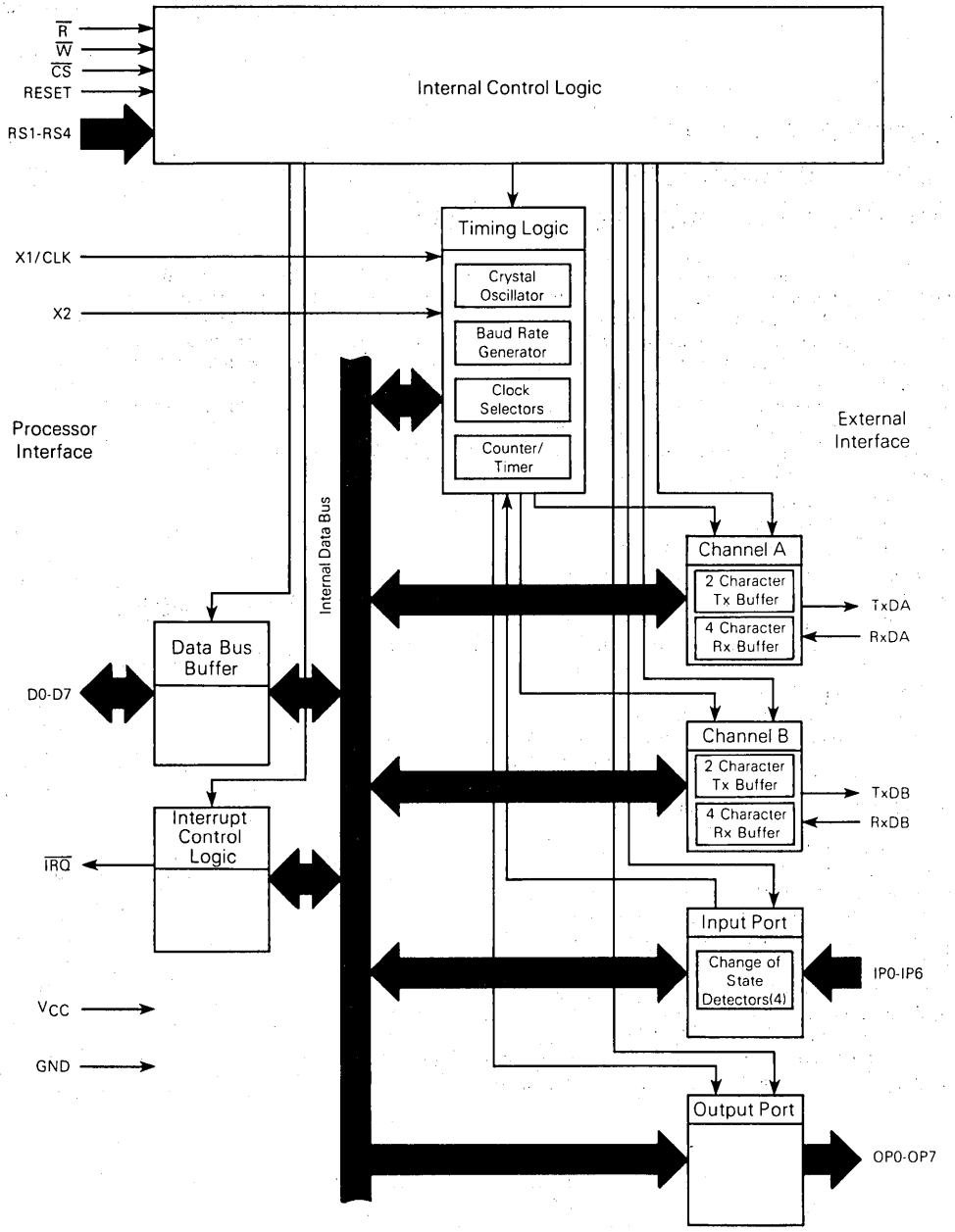


Figure 1. Block Diagram

Features (Continued)

- Versatile Interrupt System
  - — Single Interrupt Output with Eight Maskable Interrupting Conditions
  - — Interrupt Vector Output on Interrupt Acknowledge
  - — Output Port Can be Configured to Provide a Total of Up to Six Separate Wire-ORable Interrupt Outputs
- Parity, Framing, and Overrun Error Detection
- False-Start Bit Detection
- Line-Break Detection and Generation
- Detects Break Which Originates in the Middle of a Character
- Start-End Break Interrupt/Status
- On-Chip Crystal Oscillator
- TTL Comaptible
- Single +5 V Power Supply

INTERNAL CONTROL LOGIC

The internal control logic receives operation commands from the central processing unit (CPU) and generates appropriate signals to the internal sections to control device operation. It contains address decoding and read and write circuits to permit communications with the microprocessor via the data bus buffer.

TIMING LOGIC

The timing logic consists of a crystal oscillator, a baud-rate generator (BRG), a programmable 16-bit counter/timer (C/T), and four clock selectors. The crystal oscillator operates directly from a 3.6864 MHz crystal connected across the X1/CLK and X2 inputs or from an external clock of the appropriate frequency connected to X1/CLK. The clock serves as the basic timing reference for the baud-rate generator, the counter/timer, and other internal circuits. A clock signal, within the limits given in **ELECTRICAL SPECIFICATIONS**, must always be supplied to the DUART.

The baud-rate generator operates from the oscillator or external clock input and is capable of generating 18 commonly used data communication baud rates ranging from 50 to 38.4k by producing internal clock outputs at 16 times the actual baud rate. The counter/timer can be used in the timer mode to produce a 16X clock for any other baud rate by counting down the crystal clock or external clock. Other baud rates may also be derived by connecting 16X or 1X clocks to certain input port pins which have alternate functions as receiver or transmitter clock inputs. The four clock selectors allow the independent selection, for each receiver and transmitter, or any of these baud rates.

The 16-bit counter/timer (C/T) included within the DUART and timing logic can be programmed to use one of several timing sources as its input. The output of the counter/timer is available to the internal clock selectors and can also be programmed to be a parallel output at OP3. In the timer mode, the counter/timer acts as a programmable divider and can be used to generate a square-wave output at OP3. In the counter mode, the contents of the counter/timer can be read by the CPU and it can be stopped and started under program control. The counter counts down the number of pulses stored in the concatenation of the counter/timer upper register and counter/timer lower register and produces an interrupt. This is a system oriented feature which may be used to keep track of timeouts when implementing various application protocols.

INTERRUPT CONTROL LOGIC

A single active-low interrupt output ($\overline{IRQ}$) is provided which is activated upon the occurrence of any of eight internal events. Associated with the interrupt system are the interrupt mask register (IMR) and the interrupt status register (ISR). The IMR may be programmed to select only certain conditions to cause $\overline{IRQ}$ to be asserted. The ISR can be read by the CPU to determine all currently active interrupting conditions.

In addition, the DUART offers the ability to program the parallel outputs OP3 through OP7 to provide discrete interrupt outputs for the transmitters, the receivers, and the counter/timer.

DATA BUS BUFFER

The data bus buffer provides the interface between the external and internal data buses. It is controlled by the internal control logic to allow read and write data transfer operations to take place between the controlling CPU and DUART by way of the eight parallel data lines (D0 through D7).

COMMUNICATION CHANNELS A AND B

Each communication channel comprises a full-duplex asynchronous receiver/transmitter (UART). The operating frequency for each receiver and each transmitter can be selected independently from the baud-rate generator, the counter/timer, or from an external clock.

The transmitter accepts parallel data from the CPU, converts it to a serial bit stream, inserts the appropriate start, stop, and optional parity bits, and outputs a composite serial stream of data on the TxD output pin. The receiver accepts serial data on the RxD pin, converts this serial input to parallel format, checks for a start bit, stop bit, parity bit (if any), or break condition, and transfers an assembled character to the CPU during read operations.

INPUT PORT

The inputs to this unlatched 7-bit port (IP0 through IP6) can be read by the CPU by performing a read operation.

A high input results in a logic one while a low input results in a logic zero. D7 will always be read as a logic one. The pins of this port can also serve as auxiliary inputs to certain portions of the DUART logic.

Four change-of-state detectors, also provided within the input port, are associated with inputs IP0, IP1, IP2, and IP3. A high-to-low or low-to-high transition of these inputs lasting longer than 25 to 50 microseconds (best-to-worst case times) will set the corresponding bit in the input port change register (IPCR). The bits are cleared when the register is read by the CPU. Also, the DUART can be programmed so any particular change of state can generate an interrupt to the CPU.

OUTPUT PORT

This 8-bit multi-purpose output port can be used as a general-purpose output port. Associated with the output port is an output port register (OPR).

All bits of the output port register can be individually set and reset. A bit is set by performing a write operation at the appropriate address with the accompanying data specifying the bits to be set (one equals set and zero equals no change). Similarly, a bit is reset by performing a write operation at another address with the accompanying data specifying the bits to be reset (one equals reset and zero equals no change).

The output port register stores data that is to be output at the output port pins. Unlike the input port, if a particular bit of the output port register is set to a logic one or logic zero the output pin will be at a low or high level, respectively. Thus, a **logic inversion** takes place internal to the DUART with respect to this register. The outputs are complements of the data contained in the output port register.

Besides general-purpose outputs, the outputs can be individually assigned specific auxiliary functions serving the communication channels. The assignment is accomplished by appropriately programming the channel A and B mode registers (MR1A, MR1B, MR2A, and MR2B) and the output port configuration register (OPCR).

SIGNAL DESCRIPTION

The following paragraphs contain a brief description of the input and output signals.

NOTE

The terms **assertion** and **negation** will be used extensively. This is done to avoid confusion when dealing with a mixture of "active low" and "active high" signals. The term assert or assertion is used to indicate that a signal is active or true, independent of whether that level is represented by a high or low voltage. The term negate or negation is used to indicate that a signal is inactive or false.

V_{CC} AND GND

Power is supplied to the DUART using these two signals. V_{CC} is power (+5 volts) and GND is the ground connection.

CRYSTAL INPUT OR EXTERNAL CLOCK (X1/CLK)

This input is one of two connections to a crystal or a connection to an external clock. A crystal or a clock, within the specified limits, must be supplied at all times. If a crystal is used, a capacitor of approximately 10 to 15 picofarads should be connected from this pin to ground.

CRYSTAL INPUT (X2)

This input is an additional connection to a crystal. If an external TTL-level clock is used, this pin should be tied to ground. If a crystal is used, a capacitor of approximately 0 to 5 picofarads should be connected from this pin to ground.

RESET (RESET)

When active high, this input clears internal registers (SRA, SRB, IMR, ISR, OPR, OPCR), puts OP0-OP7 in the high state, stops the counter/timer, and puts channel A and B in the inactive state with the TxDA and TxDB outputs in the mark (high) state.

CHIP SELECT ($\overline{CS}$)

This active low signal, when low, enables data transfers between the CPU and DUART on the data lines (D0 through D7). These data transfers are controlled by write ($\overline{W}$), read ($\overline{R}$), and the register-select inputs (RS1 through RS4). When chip enable is high, the D0 through D7 data lines are placed in the high-impedance state.

WRITE STROBE ($\overline{W}$)

When this signal and chip enable are low, the contents of the data bus are loaded into the address register. The transfer occurs on the rising edge of the signal.

READ STROBE ($\overline{R}$)

When this signal and chip enable are low, the contents of the addressed register are presented on the data bus. The read cycle begins on the falling edge of the signal.

REGISTER-SELECT BUS (RS1 THROUGH RS4)

The register-select bus lines during read/write operations select the DUART internal registers, ports, or commands.

DATA BUS (D0 THROUGH D7)

These bidirectional three-state data lines are used to transfer commands, data, and status between the CPU and DUART. D0 is the least-significant bit.

INTERRUPT REQUEST ($\overline{IRQ}$)

This active low, open-drain output signals the CPU that one or more of the eight maskable interrupting conditions are true.

CHANNEL A TRANSMITTER SERIAL-DATA OUTPUT (TxDA)

This signal is the transmitter serial-data output for channel A. The least-significant bit is transmitted first. This output is held high (mark condition) when the transmitter is disabled, idle, or operating in the local loopback mode. (Mark is high and space is low.) Data is shifted out this pin on the falling edge of the programmed clock source.

CHANNEL A RECEIVER SERIAL-DATA INPUT (RxDA)

This signal is the receiver serial-data input for channel A. The least-significant bit is received first. Data on this pin is sampled on the rising edge of the programmed clock source.

CHANNEL B TRANSMITTER SERIAL-DATA OUTPUT (TxDB)

This signal is the transmitter serial-data output for channel B. The least-significant bit is transmitted first. This output is held high (mark condition) when the transmitter is disabled, idle, or operating in the local loopback mode. Data is shifted out this pin on the falling edge of the programmed clock source.

CHANNEL B RECEIVER SERIAL-DATA INPUT (RxDB)

This signal is the receiver serial-data input for channel B. The least-significant bit is received first. Data on this pin is sampled on the rising edge of the programmed clock source.

PARALLEL INPUTS (IP0 THROUGH IP6)

Each of the parallel inputs (IP0 through IP6) can be used as general-purpose inputs. However, each one has an alternate function(s) which is described in the following paragraphs.

IP0 This input can be used as the channel A clear-to-send active low input ($\overline{\text{CTSA}}$). A change-of-state detector is also associated with this input.

IP1 This input can be used as the channel B clear-to-send active low input ($\overline{\text{CTSB}}$). A change-of-state detector is also associated with this input.

IP2 This signal can be used as a general-purpose input or a counter/timer external clock input.

IP3 This input can be used as the channel A transmitter external clock input (TxCA). When this input is used as the external clock by the transmitter, the transmitted data is clocked on the falling edge of the clock. A change-of-state detector is also associated with this input.

IP4 This input can be used as the channel A receiver external clock input (RxCA). When this input is used as the external clock by the receiver, the received data is sampled on the rising edge of the clock.

IP5 This input can be used as the channel B transmitter external clock (TxCB). When this input is used as the external clock by the transmitter, the transmitted data is clocked on the falling edge of the clock.

IP6 This signal can be used as a general-purpose input or a channel B receiver external clock input (RxCB). When the external clock is used by the receiver, the received data is sampled on the rising edge of the clock.

PARALLEL OUTPUTS (OP0 THROUGH OP7)

Each of the parallel outputs can be used as general-purpose outputs. However, each one has an alternate function(s) which is described in the following paragraphs.

OP0 This output can be used as the channel A active low request-to-send ($\overline{\text{RTSA}}$) output. When used for this function, it is automatically negated and reasserted by either the receiver or transmitter.

OP1 This output can be used as the channel B active low request-to-send ($\overline{\text{RTSB}}$) output. When used for this function, it is negated and reasserted automatically by either the receiver or transmitter.

OP2 This output can be used as the channel A transmitter 1X-clock or 16X-clock output, or the channel A receiver 1X-clock output.

OP3 This output can be used as the open-drain active low counter-ready output, the open-drain timer output, the channel B transmitter 1X-clock output, or the channel B receiver 1X-clock output.

OP4 This output can be used as the channel A open-drain active-low receiver-ready or buffer-full interrupt outputs ($\overline{\text{RxRDYA/FFULLA}}$) by appropriately programming bit 6 of mode register 1A.

OP5 This output can be used as the channel B open-drain active-low receiver-ready or buffer-full interrupt outputs ($\overline{\text{RxRDYB/FFULLB}}$) by appropriately programming bit 6 of mode register 1B.

OP6 This output can be used as the channel A open-drain active-low transmitter-ready interrupt output ($\overline{\text{TxRDYA}}$) by appropriately programming bit 6 of the output port configuration register.

OP7 This output can be used as the channel B open-drain active-low transmitter-ready interrupt output ($\overline{\text{TxRDYB}}$) by appropriately programming bit 7 of the output port configuration register.

SIGNAL SUMMARY

Table 1 provides a summary of all the MC2681 signals described above.

6

Table 1. Signal Summary

Signal Name	Mnemonic	Pin No.	Input/Output	Active State
Power Supply (+5 V)	V_{CC}	40	Input	High
Ground	GND	20	Input	Low
Crystal Input or External Clock	X1/CLK	32	Input	—
Crystal Input	X2	33	Input	—
Reset	RESET	34	Input	High
Chip Select	$\overline{CS}$	35	Input	Low
Write Strobe	$\overline{W}$	8	Input	Low
Read Strobe	$\overline{R}$	9	Input	Low
Register-Select Bus Bit 4	RS4	6	Input	High
Register-Select Bus Bit 3	RS3	5	Input	High
Register-Select Bus Bit 2	RS2	3	Input	High
Register-Select Bus Bit 1	RS1	1	Input	High
Bidirectional-Data Bus Bit 7	D7	19	Input/Output	High
Bidirectional-Data Bus Bit 6	D6	22	Input/Output	High
Bidirectional-Data Bus Bit 5	D5	18	Input/Output	High
Bidirectional-Data Bus Bit 4	D4	23	Input/Output	High
Bidirectional-Data Bus Bit 3	D3	17	Input/Output	High
Bidirectional-Data Bus Bit 2	D2	24	Input/Output	High
Bidirectional-Data Bus Bit 1	D1	16	Input/Output	High
Bidirectional-Data Bus Bit 0	D0	25	Input/Output	High
Interrupt Request	$\overline{IRQ}$	21	Output*	Low
Channel A Transmitter Serial Data	TxDA	30	Output	—
Channel A Receiver Serial Data	RxDA	31	Input	—
Channel B Transmitter Serial Data	TxDB	11	Output	—
Channel B Receiver Serial Data	RxDB	10	Input	—
Parallel Input 6	IP6	37	Input	—
Parallel Input 5	IP5	38	Input	—
Parallel Input 4	IP4	39	Input	—
Parallel Input 3	IP3	2	Input	—
Parallel Input 2	IP2	36	Input	—
Parallel Input 1	IP1	4	Input	—
Parallel Input 0	IP0	7	Input	—
Parallel Output 7	OP7	15	Output**	—
Parallel Output 6	OP6	26	Output**	—
Parallel Output 5	OP5	14	Output**	—
Parallel Output 4	OP4	27	Output**	—
Parallel Output 3	OP3	13	Output**	—
Parallel Output 2	OP2	28	Output	—
Parallel Output 1	OP1	12	Output	—
Parallel Output 0	OP0	29	Output	—

*Requires a pullup resistor.
**May require a pullup resistor depending upon its programmed function.

PROGRAMMING AND REGISTER DESCRIPTION

The operation of the DUART is programmed by writing control words into the appropriate registers. Operational feedback is provided by way of the status registers which can be read by the CPU. The register address and address-triggered commands are described in Table 2.

Figure 2 illustrates a block diagram of the DUART from a programming standpoint and details the register configuration for each block. The locations marked "do not access" should never be read during normal operation. They are used by the factory for testing purposes.

Tables 3 and 4 are provided to illustrate the various input port pin functions and output port pin functions respectively.

Table 5 is provided to illustrate the various clock sources which may be selected for the counter and timer. More detailed information can be obtained from Table 6.

Care should be exercised if the contents of a register is changed during receiver/transmitter operation since certain changes may cause undesired results. For example, changing the number of bits-per-character while the transmitter is active may cause the transmission of an incorrect character. The contents of the mode registers (MR), the clock-select register (CSR), the output port configuration register (OPCR), and bit 7 of the auxiliary control register (ACR[7]) should only be changed after the receiver(s) and transmitter(s) have been issued software Rx and Tx reset commands. Similarly, certain changes to the auxiliary control register (ACR bits six through four) should only be made while the counter/timer (C/T) is not used (i.e., stopped if in counter mode, output and/or interrupt masked in timer mode).

Mode registers one and two of each channel are accessed via independent auxiliary pointers. The pointer is set to channel A mode register one (MR1A) and channel B mode register one (MR1B) by RESET or by issuing a "reset pointer" command via the corresponding command register. Any read or write of the mode register while the pointer is at MR1A or MR1B switches the pointer to channel A mode register two (MR2A) or channel B mode register two (MR2B). The pointer then remains at MR2A or MR2B. So, subsequent accesses will address MR2A or MR2B, unless the pointer is reset to MR1A or MR1B as described above.

Mode, command, clock-select, and status registers are duplicated for each channel to provide total independent operation and control. Refer to Table 6 for descriptions of the register and input and output port bits.

6

Table 2. Register Addressing and Address-Triggered Commands

RS4	RS3	RS2	RS1	Read ($\overline{R}$=0)	Write ($\overline{W}$=0)
0	0	0	0	Mode Register A (MR1A, MR2A)	Mode Register A (MR1A, MR2A)
0	0	0	1	Status Register A (SRA)	Clock-Select Register A (CSRA)
0	0	1	0	Do Not Access*	Command Register A (CRA)
0	0	1	1	Receiver Buffer A (RBA)	Transmitter Buffer A (TBA)
0	1	0	0	Input Port Change Register (IPCR)	Auxiliary Control Register (ACR)
0	1	0	1	Interrupt Status Register (ISR)	Interrupt Mask Register (IMR)
0	1	1	0	Counter Mode: Current MSB of Counter (CUR)	Counter/Timer Upper Register (CTUR)
0	1	1	1	Counter Mode/ Current LSB of Counter (CLR)	Counter/Timer Lower Register (CTLR)
1	0	0	0	Mode Register B (MR1B, MR2B)	Mode Register B (MR1B, MR2B)
1	0	0	1	Status Register B (SRB)	Clock-Select Register B (CSRB)
1	0	1	0	Do Not Access*	Comand Register B (CRB)
1	0	1	1	Receiver Buffer B (RBB)	Transmitter Buffer B (TBB)
1	1	0	0	Do Not Access*	Do Not Access*
1	1	0	1	Input Port (Unlatched)	Output Port Configuration Register (OPCR)
1	1	1	0	Start-Counter Command**	Output Port Register (OPR) Bit Set Command**
1	1	1	1	Stop-Counter Command**	Output Port Register (OPR) Bit Set Command**

*This address location is used for factory testing of the DUART and should not be read. Reading this location will result in undesired effects and possible incorrect transmission or reception of characters. Register contents may also be changed.
**Address triggered commands.

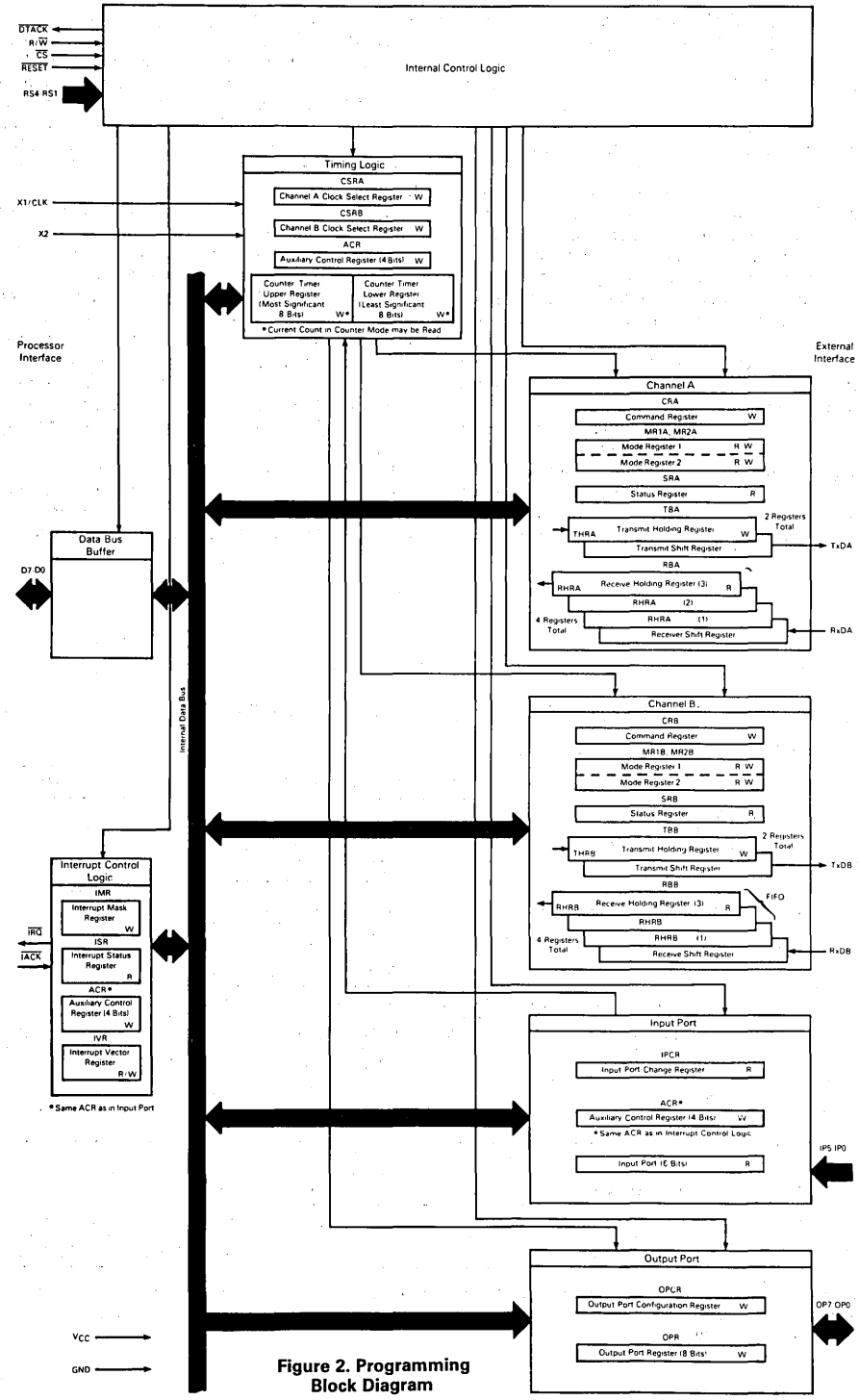


Figure 2. Programming Block Diagram

Table 3. Programming of Input Port Functions

Function	Input Port Pin					
	IP5	IP4	IP3	IP2	IP1	IP0
General Purpose	Default	Default	Default	Default	Default	Default
Change-of-State Detector			Default	Default	Default	Default
External Counter 1X Clock Input				ACR[6:4]* = 000		
External Timer 16X Clock Input				ACR[6:4]* = 100		
External Timer 1X Clock Input				ACR[6:4]* = 101		
RxCA 16X		CSRA[7:4] = 1110				
RxCA 1X		CSRA[7:4] = 1111				
TxCA 16X			CSRA[3:0] = 1110			
TxCA 1X			CSRA[3:0] = 1111			
RxCB 16X				CSRB[7:4] = 1110		
RxCB 1X				CSRB[7:4] = 1111		
TxCB 16X	CSRB[3:0] = 1110					
TxCB 1X	CSRB[3:0] = 1111					
TxCTSA						MR2A[4] = 1
TxCTSB					MR2B[4] = 1	

NOTE: Default refers to the function the input port pins perform when not used in one of the other modes. Only those functions which show the register programming are available for use.

*In these modes, because IP2 is used for the counter/timer-clock input, it is not available for use as the channel B receiver-clock input.

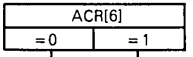
6

Table 4. Programming of Output Port Functions

Function	Output Port Pin							
	OP7	OP6	OP5	OP4	OP3	OP2	OP1	OP0
General Purpose	OPCR[7] = 0	OPCR[6] = 0	OPCR[5] = 0	OPCR[4] = 0	OPCR[3:2] = 00	OPCR[1:0] = 00	MR1B[7] = 0 MR2B[5] = 0	MR1A[7] = 0 MR2A[5] = 0
$\overline{\text{CTRDY}}$					OPCR[3:2] = 01, ACR[6] = 0*			
Timer Output					OPCR[3:2] = 01, ACR[6] = 1*			
TxCB 1X					OPCR[3:2] = 10			
RxCB 1X					OPCR[3:2] = 11			
TxCA 16X						OPCR[1:0] = 01		
TxCA 1X						OPCR[1:0] = 10		
RxCA 1X						OPCR[1:0] = 11		
$\overline{\text{TxRDYA}}$		OPCR[6] = 1*						
$\overline{\text{TxRDYB}}$	OPCR[7] = 1*							
$\overline{\text{RxRDYA}}$				OPCR[4] = 1, MR1A[6] = 0*				
$\overline{\text{RxRDYB}}$			OPCR[5] = 1, MR1B[6] = 0*					
$\overline{\text{FFULLA}}$				OPCR[4] = 1, MR1A[6] = *				
$\overline{\text{FFULLB}}$			OPCR[5] = 1, MR1B[6] = 1*					
$\overline{\text{RxRTSA}}$								MR1A[7] = 1
$\overline{\text{TxRTSA}}$								MR2A[5] = 1
RxRTSB							MR1B[7] = 1	
$\overline{\text{TxRTSB}}$							MR2B[5] = 1	

Note: Only those functions which show the register programming are available for use.
*Pin requires a pullup resistor if used for this function.

Table 5. Selection of Clock Sources for the Counter and Timer Modes



Counter Mode Clock Sources	ACR[5:4] =
External Input via Input Port Pin 2 (IP2)	00
Channel A 1X Transmitter Clock TxCA	01
Channel B 1X Transmitter Clock TxCB	10
Crystal Oscillator Divide by 16 via X1/CLK and X2 Inputs	11
External Input Divide by 16 via X1/CLK Input Pin	11

Timer Mode Clock Sources	ACR[5:4] =
External Input via Input Port Pin 2 (IP2)	00
External Input Divide by 16 via Input Port Pin 2 (IP2)	01
Crystal Oscillator via X1/CLK and X2 Inputs	10
Crystal Oscillator Divide by 16 via X1/CLK and X2 Inputs	11
External Input via X1/CLK Input Pin	10
External Input Divide by 16 via X1/CLK Input Pin	11

NOTE: Only those functions which show the register programming are available for use.

Table 6. Register Bit Formats (Sheet 1 of 5)

CHANNEL A MODE REGISTER 1 (MR1A) AND CHANNEL B MODE REGISTER 1 (MR1B)

Rx RTS Control	Rx $\overline{IRQ}$ Select	Error Mode	Parity Mode	Parity Type	Bits-per-Character
Bit 7	Bit 6	Bit 5	Bit 4 Bit 3	Bit 2 With Parity 0 = Even 1 = Odd	Bit 1 Bit 0
0 = Disabled 1 = Enabled	0 = RxRDY 1 = FFULL	0 = Char 1 = Block	0 0 = With Parity 0 1 = Force Parity 1 0 = No Parity 1 1 = Multidrop Mode*	Force Parity 0 = Low 1 = High Multidrop Mode 0 = Data 1 = Address	0 0 = 5 0 1 = 6 1 0 = 7 1 1 = 8

*The parity bit is used as the address/data bit in multidrop mode.

CHANNEL A MODE REGISTER 2 (MR2A) AND CHANNEL B MODE REGISTER 2 (MR2B)

Channel Mode	Tx RTS Control	CTS Enable Transmitter
Bit 7 Bit 6	Bit 5	Bit 4
0 0 = Normal 0 1 = Automatic Echo 1 0 = Local Loopback 1 1 = Remote Loopback	0 = Disabled 1 = Enabled	0 = Disabled 1 = Enabled

NOTE:
If an external 1X clock is used for the transmitter, MR2 bit 3 = 0 selects one stop bit and MR2 bit 3 = 1 selects two stop bits to be transmitted.

Stop Bit Length					6-8 Bits/ Character	5-Bits/ Character
	Bit 3	Bit 2	Bit 1	Bit 0		
(0)	0	0	0	0 =	0.563	1.063
(1)	0	0	0	1 =	0.625	1.125
(2)	0	0	1	0 =	0.688	1.188
(3)	0	0	1	1 =	0.750	1.250
(4)	0	1	0	0 =	0.813	1.313
(5)	0	1	0	1 =	0.875	1.375
(6)	0	1	1	0 =	0.938	1.438
(7)	0	1	1	1 =	1.000	1.500
(8)	1	0	0	0 =	1.563	1.563
(9)	1	0	0	1 =	1.625	1.625
(A)	1	0	1	0 =	1.688	1.688
(B)	1	0	1	1 =	1.750	1.750
(C)	1	1	0	0 =	1.813	1.813
(D)	1	1	0	1 =	1.875	1.875
(E)	1	1	1	0 =	1.938	1.938
(F)	1	1	1	1 =	2.000	2.000

Table 6. Register Bit Formats (Sheet 2 of 5)

CLOCK-SELECT REGISTER A (CSRA)

Receiver-Clock Select			Transmitter-Clock Select		
Bit 7 Bit 6 Bit 5 Bit 4	Baud Rate Set 1 ACR Bit 7=0	Baud Rate Set 2 ACR Bit 7=1	Bit 3 Bit 2 Bit 1 Bit 0	Baud Rate Set 1 ACR Bit 7=0	Baud Rate Set 2 ACR Bit 7=1
0 0 0 0	50	75	0 0 0 0	50	75
0 0 0 1	110	110	0 0 0 1	110	110
0 0 1 0	134.5	134.5	0 0 1 0	134.5	134.5
0 0 1 1	200	150	0 0 1 1	200	150
0 1 0 0	300	300	0 1 0 0	300	300
0 1 0 1	600	600	0 1 0 1	600	600
0 1 1 0	1200	1200	0 1 1 0	1200	1200
0 1 1 1	1050	2000	0 1 1 1	1050	2000
1 0 0 0	2400	2400	1 0 0 0	2400	2400
1 0 0 1	4800	4800	1 0 0 1	4800	4800
1 0 1 0	7200	1800	1 0 1 0	7200	1800
1 0 1 1	9600	9600	1 0 1 1	9600	9600
1 1 0 0	38.4k	19.2k	1 1 0 0	38.4k	19.2k
1 1 0 1	Timer	Timer	1 1 0 1	Timer	Timer
1 1 1 0	IP4-16X	IP4-16X	1 1 1 0	IP3-16X	IP3-16X
1 1 1 1	IP4-1X	IP4-1X	1 1 1 1	IP3-1X	IP3-1X

NOTE: Receiver clock is always a 16X clock except when CSRA bits seven through four equal 1111.

NOTE: Transmitter clock is always a 16X clock except when CSRA bits three through zero equal 1111.

6

CLOCK-SELECT REGISTER B (CSRB)

Receiver-Clock Select			Transmitter-Clock Select		
Bit 7 Bit 6 Bit 5 Bit 4	Baud Rate Set 1 ACR Bit 7=0	Baud Rate Set 2 ACR Bit 7=1	Bit 3 Bit 2 Bit 1 Bit 0	Baud Rate Set 1 ACR Bit 7=0	Baud Rate Set 2 ACR Bit 7=1
0 0 0 0	50	75	0 0 0 0	50	75
0 0 0 1	110	110	0 0 0 1	110	110
0 0 1 0	134.5	134.5	0 0 1 0	134.5	134.5
0 0 1 1	200	150	0 0 1 1	200	150
0 1 0 0	300	300	0 1 0 0	300	300
0 1 0 1	600	600	0 1 0 1	600	600
0 1 1 0	1200	1200	0 1 1 0	1200	1200
0 1 1 1	1050	2000	0 1 1 1	1050	2000
1 0 0 0	2400	2400	1 0 0 0	2400	2400
1 0 0 1	4800	4800	1 0 0 1	4800	4800
1 0 1 0	7200	1800	1 0 1 0	7200	1800
1 0 1 1	9600	9600	1 0 1 1	9600	9600
1 1 0 0	38.4k	19.2k	1 1 0 0	38.4k	19.2k
1 1 0 1	Timer	Timer	1 1 0 1	Timer	Timer
1 1 1 0	IP2-16X	IP2-16X	1 1 1 0	IP5-16X	IP5-16X
1 1 1 1	IP2-1X	IP2-1X	1 1 1 1	IP5-1X	IP5-1X

NOTE: Receiver clock is always a 16X clock except when CSRB bits seven through four equal 1111.

NOTE: Transmitter clock is always a 16X clock except when CSRB bits three through zero equal 1111.

Table 6. Register Bit Formats (Sheet 3 of 5)

CHANNEL A COMMAND REGISTER (CRA) AND CHANNEL B COMMAND REGISTER (CRB)

Not Used*	Miscellaneous Commands	Transmitter Commands	Receiver Commands
Bit 7	Bit 6 Bit 5 Bit 4	Bit 3 Bit 2	Bit 1 Bit 0
X	0 0 0 No Command 0 0 1 Reset MR Pointer to MR1 0 1 0 Reset Receiver 0 1 1 Reset Transmitter 1 0 0 Reset Error Status 1 0 1 Reset Channel's Break-Change Interrupt 1 1 0 Start Break 1 1 1 Stop Break	0 0 No Action, Stays in Present Mode 0 1 Transmitter Enabled 1 0 Transmitter Disabled 1 1 Don't Use, Indeterminate	0 0 No Action, Stays in Present Mode 0 1 Receiver Enabled 1 0 Receiver Disabled 1 1 Don't Use, Indeterminate

* Bit seven is not used and may be set to either zero or one.

CHANNEL A STATUS REGISTER (SRA) AND CHANNEL B STATUS REGISTER (SRB)

Received Break	Framing Error	Parity Error	Overrun Error	TxEMT	TxRDY	FFULL	RxRDY
Bit 7*	Bit 6*	Bit 5*	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0 = No 1 = Yes	0 = No 1 = Yes	0 = No 1 = Yes	0 = No 1 = Yes	0 = No 1 = Yes	0 = No 1 = Yes	0 = No 1 = Yes	0 = No 1 = Yes

* These status bits are appended to the corresponding data character in the receive FIFO and are valid only when the RxRDY bit is set. A read of the status register provides these bits (seven through five) from the top of the FIFO together with bits four through zero. These bits are cleared by a reset error status command. In character mode, they are discarded when the corresponding data character is read from the FIFO.

OUTPUT PORT CONFIGURATION REGISTER (OPCR)

OP7	OP6	OP5	OP4	OP3	OP2
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3 Bit 2	Bit 1 Bit 0
0 = OPR Bit 7 1 = TxRDYB	0 = OPR Bit 6 1 = TxRDYA	0 = OPR Bit 5 1 = RxRDYB / FFULLB	0 = OPR Bit 4 1 = RxRDYA/ FFULLA	0 0 = OPR Bit 3 0 1 = C/T Output * 1 0 = TxCB (1X) 1 1 = RxCB (1X)	0 0 = OPR Bit 2 0 1 = TxCA (16X) 1 0 = TXCA (1X) 1 1 = RxCA (1X)

* If OP3 is to be used for the timer output, the counter/timer should be programmed for timer mode (ACR[6] = 1), the counter/timer preload registers (CTUR and CTLR) initialized, and the start counter command issued before setting OPCR[3:2] = 01.

NOTE: OP1 and OP0 can be used as transmitter and receiver RTS control lines by appropriately programming the mode registers (MR1[7] for the receiver RxRTS, and MR2[5] for the transmitter TxRTS). OP1 is used for channel B's RTS control line and OP0 for channel A's RTS control line. When OP1 and OP0 are not used for RTS control, they may be used as general-purpose outputs. (See Table 4-3.)

OUTPUT PORT REGISTER (OPR)

OPR7	OPR6	OPR5	OPR4	OPR3	OPR2	OPR1	OPR0
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Table 6. Register Bit Formats (Sheet 4 of 5)

AUXILIARY CONTROL REGISTER (ACR)

BRG SET Select*	Counter/Timer Mode and Source**			Delta*** IP3 IRQ	Delta*** IP2 IRQ	Delta*** IP1 IRQ	Delta*** IP0 IRQ
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	Mode		Clock Source				
0 = Set 1				0 = Disabled	0 = Disabled	0 = Disabled	0 = Disabled
1 = Set 2	0	0	0 Counter	External (IP2)****	1 = Enabled	1 = Enabled	1 = Enabled
	0	0	1 Counter	TxCA – 1X Clock of Channel A Transmitter			
	0	1	0 Counter	TxCB – 1X Clock of Channel B Transmitter			
	0	1	1 Counter	Crystal or External Clock (X1/CLK) Divided by 16			
	1	0	0 Timer	External (IP2)****			
	1	0	1 Timer	External (IP2) Divided by 16****			
	1	1	0 Timer	Crystal or External Clock (X1/CLK)			
	1	1	1 Timer	Crystal or External Clock (X1/CLK) Divided by 16			

*Should only be changed after both channels have been reset and are disabled.

**Should only be altered while the counter/timer is not in use (i.e., stopped if in counter mode, output and/or interrupt masked if in timer mode).

***Delta is equivalent to change-of-state.

****In these modes, because IP2 is used for the counter/timer clock input, it is not available for use as the channel B receiver-clock input.

INPUT PORT CHANGE REGISTER (IPCR)

Delta* Detected IP3	Delta* Detected IP2	Delta* Detected IP1	Delta* Detected IP0	Level IP3	Level IP2	Level IP1	Level IP0
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0 = No	0 = No	0 = No	0 = No	0 = Low	0 = Low	0 = Low	0 = Low
1 = Yes	1 = Yes	1 = Yes	1 = Yes	1 = High	1 = High	1 = High	1 = High

*Delta is equivalent to change-of-state.

INTERRUPT STATUS REGISTER (ISR)

Input Port Change	Delta Break B	RxRDYB/ FFULLB	TxRDYB	Counter/ Timer Ready	Delta Break A	RxRDYA/ FFULLA	TxRDYA
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0 = No	0 = No	0 = No	0 = No	0 = No	0 = No	0 = No	0 = No
1 = Yes	1 = Yes	1 = Yes	1 = Yes	1 = Yes	1 = Yes	1 = Yes	1 = Yes

INTERRUPT MASK REGISTER (IMR)

Input Port Change IRQ	Delta Break B IRQ	RxRDYB/ FFULLB IRQ	TxRDYB IRQ	Counter/ Timer Ready IRQ	Delta Break A IRQ	RxRDYA/ FFULLA IRQ	TxRDYA IRQ
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0 = Masked	0 = Masked	0 = Masked	0 = Masked	0 = Masked	0 = Masked	0 = Masked	0 = Masked
1 = Pass	1 = Pass	1 = Pass	1 = Pass	1 = Pass	1 = Pass	1 = Pass	1 = Pass

Table 6. Register Bit Formats (Sheet 5 of 5)

COUNTER/TIMER UPPER REGISTER (CTUR)

C/T[15]	C/T[14]	C/T[13]	C/T[12]	C/T[11]	C/T[10]	C/T[9]	C/T[8]
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

COUNTER/TIMER LOWER REGISTER (CTLR)

C/T[7]	C/T[6]	C/T[5]	C/T[4]	C/T[3]	C/T[2]	C/T[1]	C/T[0]
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

INTERRUPT VECTOR REGISTER (IVR)

IVR[7]	IVR[6]	IVR[5]	IVR[4]	IVR[3]	IVR[2]	IVR[1]	IVR[0]
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

INPUT PORT

*	**	IP5	IP4	IP3	IP2	IP1	IP0
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

*Bit seven has no external pin. Upon reading the input port, bit seven will always be read as a one.

**Bit six has no external pin. Upon reading the input port, bit six will reflect the current logic level of $\overline{IACK}$.

OUTPUT PORT

OP7	OP6	OP5	OP4	OP3	OP2	OP1	OP0
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
$\overline{OPR[7]}$	$\overline{OPR[6]}$	$\overline{OPR[5]}$	$\overline{OPR[4]}$	$\overline{OPR[3]}$	$\overline{OPR[2]}$	$\overline{OPR[1]}$	$\overline{OPR[0]}$

ELECTRICAL SPECIFICATIONS

ABSOLUTE MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Supply Voltage	V_{CC}	−0.5 to +6.0	V
Input Voltage	V_{in}	−0.5 to +6.0	V
Operating Temperature Range	T_A	0 to +70	°C
Storage Temperature	T_{stg}	−65 to +150	°C

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

STILL AIR THERMAL CHARACTERISTICS

Characteristic	Value		Rating
	θ_{JA}	θ_{JC}	
Thermal Resistance			°C/W
Ceramic, Type L	50	25*	
Plastic, P			
Cu Lead Frame	50	25*	
A42 Lead Frame	100	50*	

*Estimated

POWER CONSIDERATIONS

The average chip-junction temperature, T_J, in °C can be obtained from:

$$T_J = T_A + (P_D \cdot \theta_{JA}) \quad (1)$$

where:

T_A = Ambient Temperature, °C
θ_{JA} = Package Thermal Resistance, Junction-to-Ambient, °C/W
$P_D = P_{INT} + P_{I/O}$
$P_{INT} = I_{CC} \times V_{CC}$, 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.

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

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

Solving equations (1) and (2) for K gives:

$$K = P_D \cdot (T_A + 273°C) + \theta_{JA} \cdot P_D^2 \quad (3)$$

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) interatively for any value of T_A.

The total thermal resistance of a package (θ_{JA}) can be separated into two components, θ_{JC} and θ_{CA}, representing the barrier to heat flow from the semiconductor junction to the package (case) surface (θ_{JC}) and from the case to the outside ambient (θ_{CA}). These terms are related by the equation:

$$\theta_{JA} = \theta_{JC} + \theta_{CA} \quad (4)$$

θ_{JC} is device related and cannot be influenced by the user. However, θ_{CA} is user dependent and can be minimized by such thermal management techniques as heat sinks, ambient air cooling, and thermal convention. Thus, good thermal management on the part of the user can significantly reduce θ_{CA} so that θ_{JA} approximately equals θ_{JC}. Substitution of θ_{JC} for θ_{JA} in equation (1) will result in a lower semiconductor junction temperature.

Values for thermal resistance presented in this document, unless estimated, were derived using the procedure described in Motorola Reliability Report 7843, "Thermal Resistance Measurement Method for MC68XX Microcomponent Devices," and are provided for design purposes only. Thermal measurements are complex and dependent on procedure and setup. User derived values for thermal resistance may differ.

DC ELECTRICAL CHARACTERISTICS (T_A = 0°C to 70°C; V_{CC} = 5.0 V ±5%)

Characteristic	Symbol	Min	Typ	Max	Unit
Input High Voltage, Except X1/CLK	V_{IH}	2.0	—	—	V
Input High Voltage, X1/CLK	V_{IH}	4.0	—	—	V
Input Low Voltage	V_{IL}	—	—	0.8	V
Output High Voltage, Except Open-Collector Outputs (I_{OH} = −400 µA)	V_{OH}	2.4	—	—	V
Output Low Voltage (I_{OL} = 2.4 mA)	V_{OL}	—	—	0.4	V
Input Leakage Current (V_{in} = 0 to V_{CC})	I_{IL}	−10	—	10	µA
Data Bus Hi-Z Leakage Current (V_{out} = 0 to V_{CC})	I_{LL}	−10	—	10	µA
Open-Collector Output Leakage Current (V_{out} = 0 to V_{CC})	I_{OC}	−10	—	10	µA
Power Supply Current	I_{CC}	—	—	150	mA
Capacitance (V_{in} = 5 V, T_A = 25°C, f = 1 MHz)	C_{in}	—	—	15	pF
X1/CLK Low Input Current V_{in} = 0, X2 Grounded V_{in} = 0, X2 Floated	I_{X1L}	 −4.0 −3.0	 −2.0 −1.5	 0 0	mA
X1/CLK High Input Current V_{in} = V_{CC}, X2 Grounded V_{in} = V_{CC}, X2 Floated	I_{X1H}	 −1.0 0	 0.2 3.5	 1.0 10.0	mA
X2 Low Input Current V_{in} = 0, X1/CLK Floated	I_{X2L}	−100	−30	0	µA
X2 High Input Current V_{in} = V_{CC}, X1/CLK Floated	I_{X2H}	0	30	100	µA

AC ELECTRICAL CHARACTERISTICS (T_A = 0°C to 70°C, V_{CC} = 5.0 V ±5%)

Characteristic	Symbol	Min	Max	Unit
X1/CLK Frequency*	f_{CLK}	2.0	4.0	MHz
CTCLK (IP2) Frequency	f_{CTC}	0	4.0	MHz
Receiver Clock Frequency (RxC) 16X Clock 1X Clock	f_{Rx}	 0 0	 2.0 1.0	MHz
Transmitter Clock Frequency (TxC) 16X Clock 1X Clock	f_{Tx}	 0 0	 2.0 1.0	MHz

*To use the standard baud rates selected by the clock-select register given in Table 6, the X1/CLK frequency should be set at 3.6864 MHz.

AC ELECTRICAL CHARACTERISTICS — RESET TIMING (see Figure 3)

Characteristic	Symbol	Min	Max	Unit
RESET Pulse Width	t_{RES}	1.0	—	µs

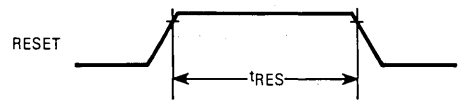


Figure 3. RESET Timing

AC ELECTRICAL CHARACTERISTICS — BUS TIMING (see Figure 4)

Characteristic	Symbol	Min	Max	Unit
RS1-RS4 Setup Time to $\overline{R}$, $\overline{W}$ Asserted	t_{RSS}	10	—	ns
RS1-RS4 Hold Time from $\overline{R}$, $\overline{W}$ Negated	t_{RSH}	0	—	ns
$\overline{CS}$ Setup Time to $\overline{R}$, $\overline{W}$ Asserted	t_{CS}	0	—	ns
$\overline{CS}$ Hold Time from $\overline{R}$, $\overline{W}$ Negated	t_{CH}	0	—	ns
$\overline{R}$, $\overline{W}$ Pulse Width Asserted	t_{RW}	205	—	ns
Data Valid after $\overline{R}$ Asserted	t_{DD}	—	175	ns
Data Bus Floating after $\overline{R}$ Negated	t_{DF}	—	100	ns
Data Setup Time before $\overline{W}$ Negated	t_{DS}	100	—	ns
Data Hold Time after $\overline{R}$ or $\overline{W}$ Negated	t_{DH}	10	—	ns
High Time Between Reads and/or Writes*	t_{RWD}	200	—	ns

*If $\overline{CS}$ is used as the "strobing" input, this parameter defines the minimum high time between one $\overline{CS}$ and the next. $\overline{CS}$ and $\overline{R}(\overline{W})$ are ANDed internally. Subsequently, the signal asserted last initiates the cycle and the signal negated first terminates the cycle. $\overline{R}$ must be negated for t_{RWD} to guarantee that any status register changes are valid. Consecutive write operations to the same command register require at least three edges of the X1 clock between writes. Typically, a processor is incapable of accessing the same command register a second time prior to three transitions on the X1/CLK pin.

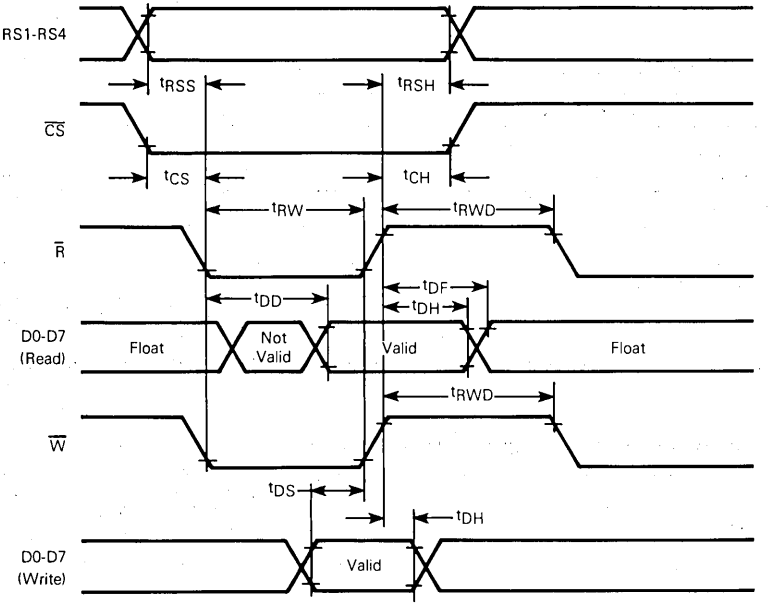


Figure 4. Bus Timing

AC ELECTRICAL CHARACTERISTICS — PORT TIMING (see Figure 5)

Characteristic	Symbol	Min	Max	Unit
Port Input Setup Time to $\overline{R}$ Asserted	t_{PS}	0	—	ns
Port Input Hold Time from $\overline{R}$ Negated	t_{PH}	0	—	ns
Port Output Valid from $\overline{W}$ Negated	t_{PD}	—	400	ns

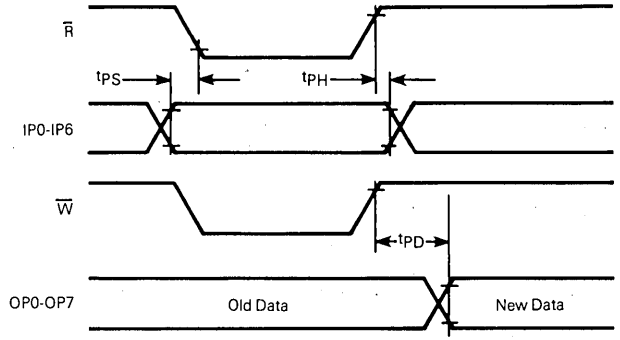


Figure 5. Port Timing

AC ELECTRICAL CHARACTERISTICS — INTERRUPT TIMING (see Figure 6)

Characteristic	Symbol	Min	Max	Unit
$\overline{IRQ}$ Negated or OP3-OP7 High for $\overline{R}$ or $\overline{W}$ Negated when used as Interrupts from:	t_{IR}			ns
Read RHR (RxRDY/FFULL Interrupt)		—	300	
Write THR (TxRDY Interrupt)		—	300	
Reset Command (Delta Break Interrupt)		—	300	
Stop C/T Command (Counter Interrupt)		—	300	
Read IPCR (Input Port Change Interrupt)		—	300	
Write IMR (Clear-of-Interrupt Mask Bit)		—	300	

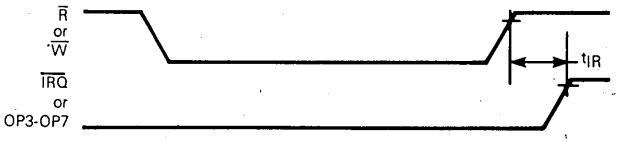
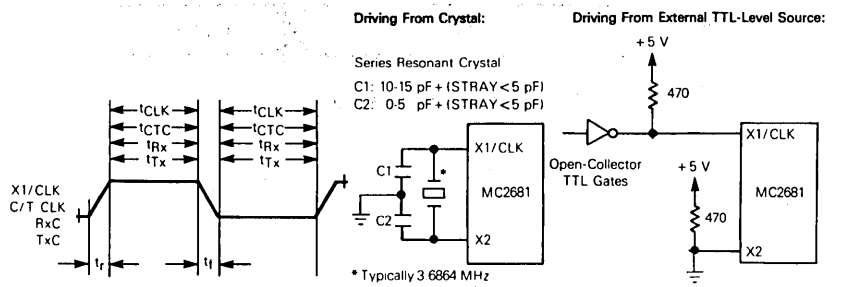


Figure 6. Interrupt Timing

AC ELECTRICAL CHARACTERISTICS — CLOCK TIMING (see Figure 7 and Note 1)

Characteristic	Symbol	Min	Max	Unit
X1/CLK High or Low Time	t_{CLK}	100	—	ns
Counter/Timer Clock High or Low Time	t_{CTC}	100	—	ns
Receive Clock (RxC) High or Low Time	t_{Rx}	220	—	ns
Transmit Clock (TxC) High or Low Time	t_{Tx}	220	—	ns
Clock Rise Time	t_r	—	20	ns
Clock Fall Time	t_f	—	20	ns

NOTE:
1. All voltage measurements are referenced to ground (GND). For testing, all input signals except X1/CLK swing between 0.4 volt and 2.4 volts with a maximum transition time of 20 nanoseconds. For X1/CLK, this swing is between 0.4 and 4.4 volts. All time measurements are referenced at input and output voltages of 0.8 volt and 2.0 volts as appropriate. Test conditions for non-interrupt outputs: $C_L = 150$ picofarads, $R_L = 750$ ohms to V_{CC}. Test conditions for interrupt outputs: $C_L = 50$ picofarads, $R_L = 27$ kilohms to V_{CC}.



NOTE: Board layout should be such that the crystal and capacitor(s) are as close as possible to the pins of the DUART to minimize stray capacitance. Also, crystal series resistance should be less than 180 ohms.

Figure 7. Clock Timing

AC ELECTRICAL CHARACTERISTICS — TRANSMITTER TIMING (see Figure 8 and Note 1)

Characteristic	Symbol	Min	Max	Unit
TxD Output Valid from TxC Low	t_{TxD}	—	350	ns
TxC Low to TxD Output Valid	t_{TCS}	—	150	ns

NOTE:
1. All voltage measurements are referenced to ground (GND). For testing, all input signals except X1/CLK swing between 0.4 volt and 2.4 volts with a maximum transition time of 20 nanoseconds. For X1/CLK, this swing is between 0.4 and 4.4 volts. All time measurements are referenced at input and output voltages of 0.8 volt and 2.0 volts as appropriate. Test conditions for non-interrupt outputs: $C_L = 150$ picofarads, $R_L = 750$ ohms to V_{CC}. Test conditions for interrupt outputs: $C_L = 50$ picofarads, $R_L = 27$ kilohms to V_{CC}.

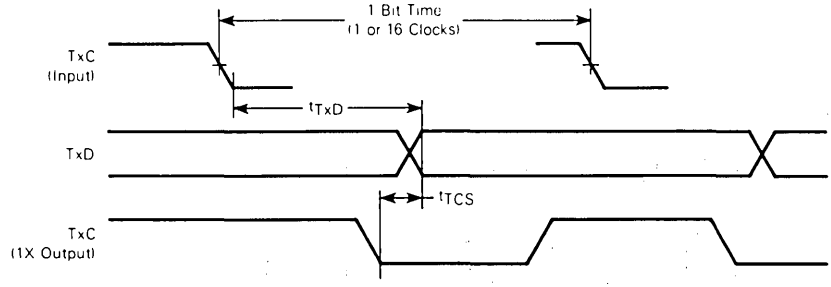


Figure 8. Transmit Timing

AC ELECTRICAL CHARACTERISTICS — RECEIVER TIMING (see Figure 9)

Characteristic	Symbol	Min	Max	Unit
RxD Data Setup Time to RxC High	t_{RxS}	240	—	ns
RxD Data Hold Time from RxC High	t_{RxH}	200	—	ns

6

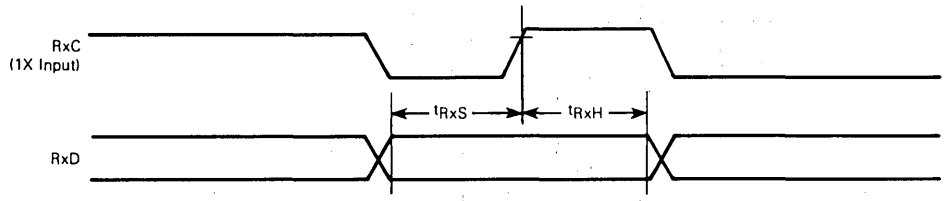
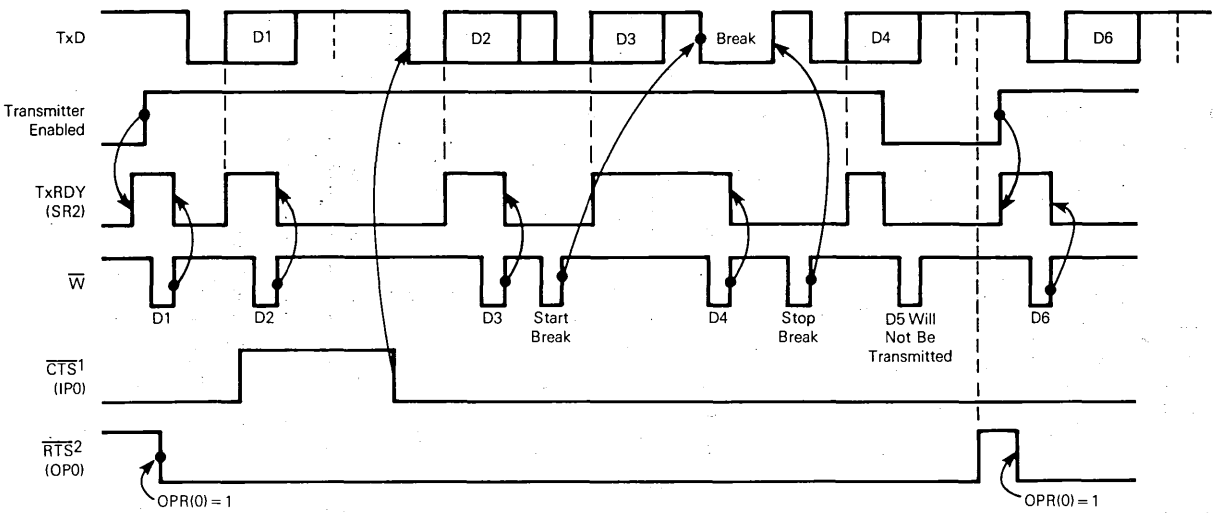
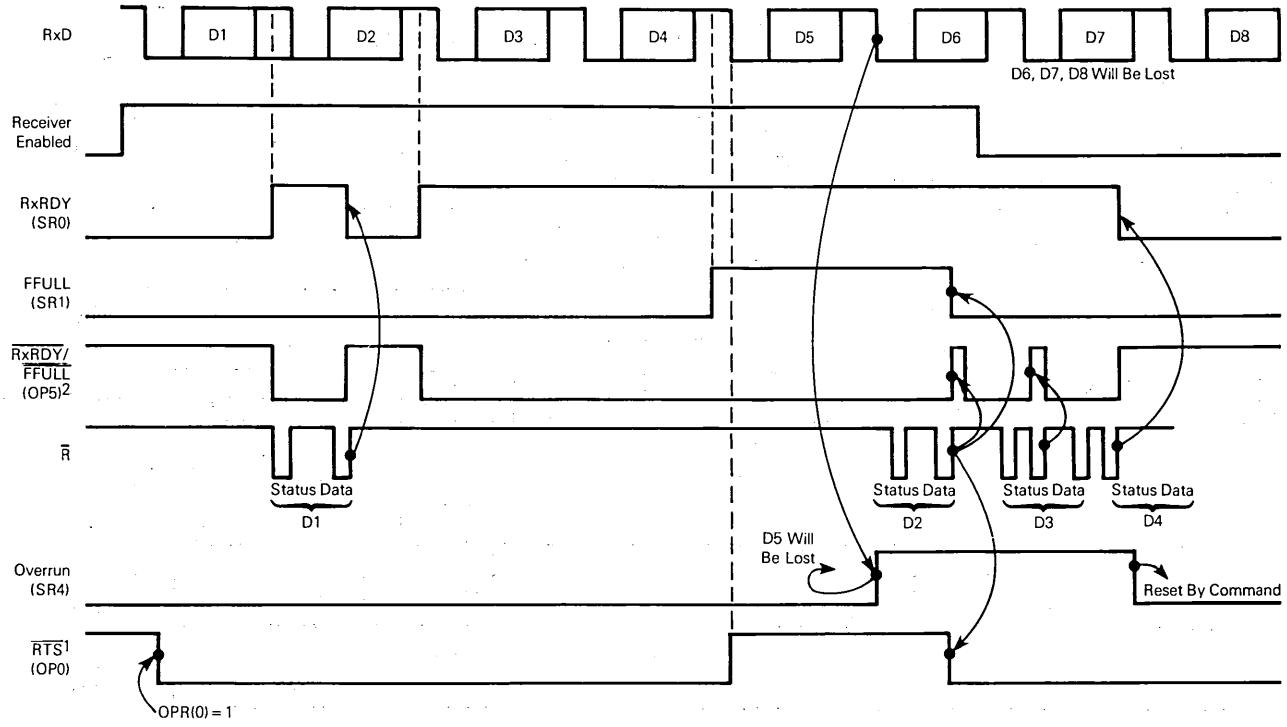


Figure 9. Receive Timing



OTES:
1. Timing shown for MR2(4) = 1.
2. Timing shown for MR2(5) = 1.

Figure 10. Transmitter Timing



NOTES
1. Timing shown for MR1(7) = 1.
2. Shown for OPCR(4) = 1 and MR(6) = 0.

Figure 11. Receiver Timing

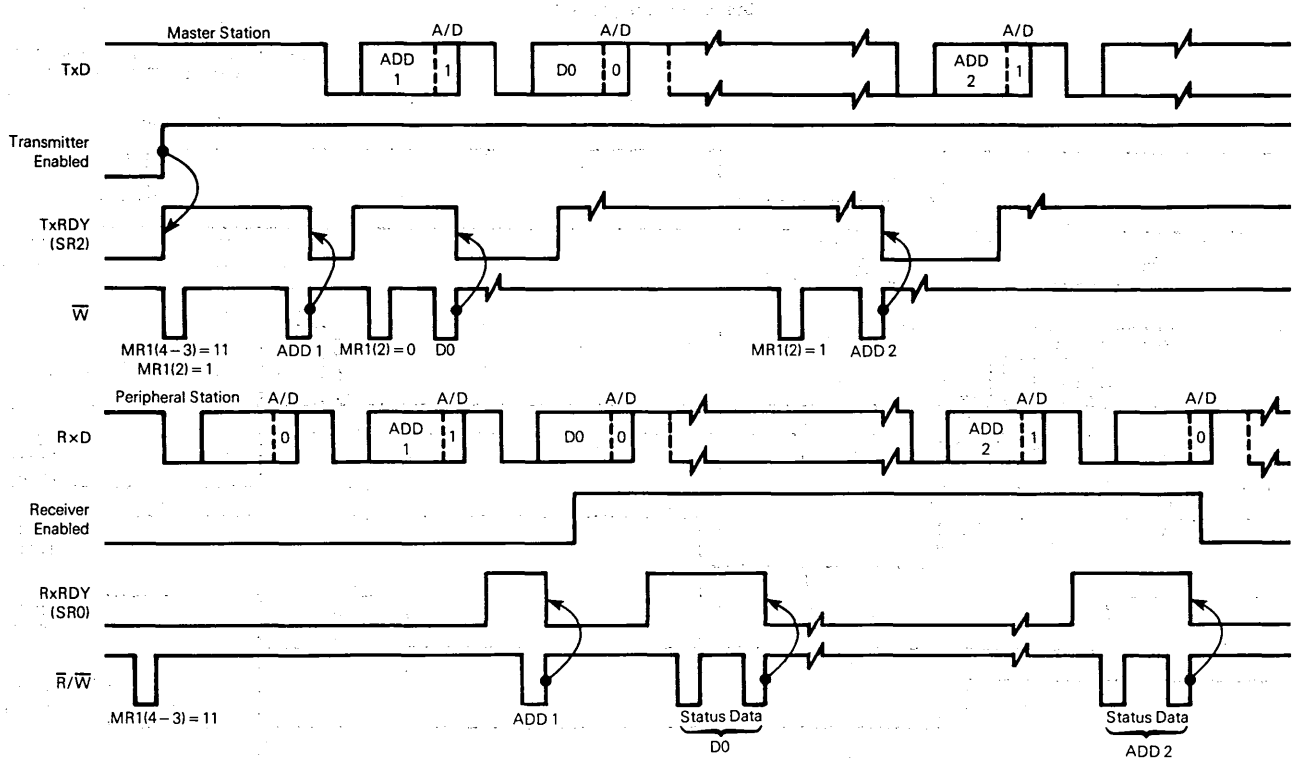
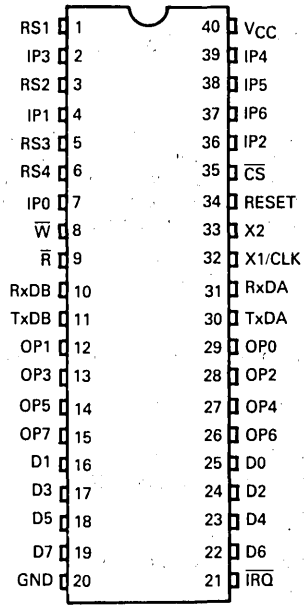


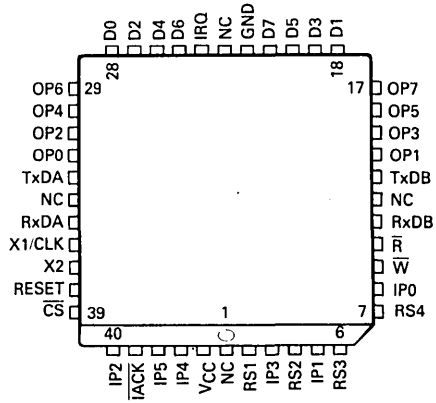
Figure 12. Wake-Up Mode Timing

PIN ASSIGNMENTS

DUAL-IN-LINE



PLASTIC LEADED CHIP CARRIER



6

MC68652
MC2652

Advance Information

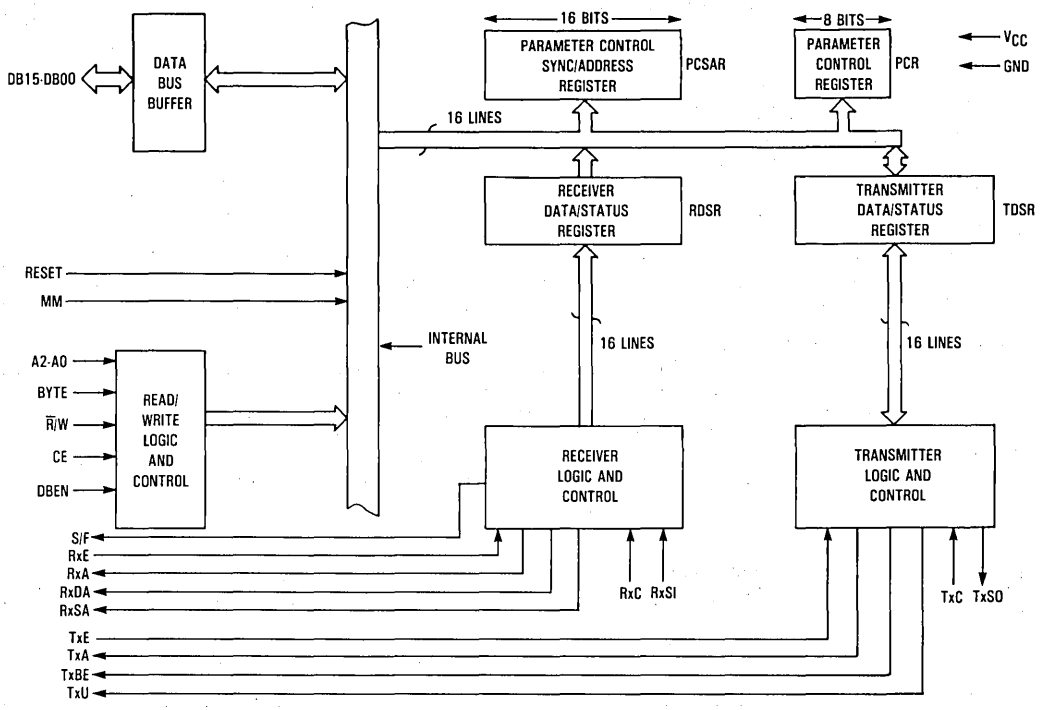
Multi-Protocol Communications Controller (MPCC)

The MC68652/MC2652 MPCC formats, transmits, and receives synchronous serial data while supporting bit-oriented protocols (BOP) or byte-control protocols (BCP). The parallel bus of the MPCC readily interfaces with the M6800 and M68000 Microprocessor Families as well as many other 8- or 16-bit processors. Typical applications include intelligent terminals, front-end communications, remote-data concentrators, communication test equipment, and computer-to-computer links.

- DC to 2 Mbps Data Rate
- Bit-Oriented Protocols (BOP): SCLC, ADCP, HDLC, X.25
  - — Character Length — 1-to-8 Bits
  - — Address Comparison
  - — Automatic Detection and Generation of Special Control Characters, i.e., FLAG ABORT, GA
  - — Automatic Zero Insertion and Deletion
  - — Short Last Character
  - — Idle Transmission of FLAG or ABORT Characters
- Byte-Control Protocols (BCP), DDCMP, BISYNC (External CRC)
  - — Character Length — 5-to-8 Bits
  - — SYNC Generation Detection and Stripping
  - — Automatic Generation and Checking of CRC-16 or VRC
- Maintenance Mode for Self-Checking
- Bidirectional, Three-State, 8- or 16-Bit Data Bus
- TTL Compatible
- Compatible with MC68653/MC2653 Polynomial Generator Checker

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

BLOCK DIAGRAM



6

MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Supply Voltage	V_{CC}	−0.3 to +7.0	V
Input Voltage	V_{in}	−0.3 to +15.0	V
Output Voltage	V_{out}	−0.3 to +15.0	V
Operating Temperature Range	T_A	0 to 70	°C
Storage Temperature Range	T_{stg}	−65 to 150	°C

This device contains circuitry to protect the inputs against damage due to high static voltages or electric fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum rated voltages to this high-impedance circuit. For proper operation it is recommended that V_{in} and V_{out} be constrained to the range GND ≤ (V_{in} or V_{out}) ≤ V_{CC}. Reliability of operation is enhanced if unused inputs are tied to an appropriate logic voltage level (e.g., either GND or V_{CC}).

THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Symbol	Value	Unit
Thermal Resistance (Still Air)	θ_{JA}		θ_{JC}		°C/W
Plastic, Type P					
Copper Lead Frame		50		25*	
Alloy 42 Lead Frame		100		50*	

*Estimated

POWER CONSIDERATIONS

The average chip-junction temperature, T_J, in °C can be obtained from:

$$T_J = T_A + (P_D \cdot \theta_{JA}) \quad (1)$$

where:

T_A ≡ Ambient Temperature, °C

θ_{JA} ≡ Package Thermal Resistance, Junction-to-Ambient, °C/W

P_D ≡ $P_{INT} + P_{PORT}$

P_{INT} ≡ $I_{CC} \times V_{CC}$, Watts — Chip Internal Power

P_{PORT} ≡ Power Dissipation, Watts — User Determined

For most applications $P_{PORT} \ll P_{INT}$ and can be neglected. P_{PORT} may become significant if the device is configured to drive Darlington bases or sink LED loads.

An approximate relationship between P_D and T_J (if P_{PORT} is neglected) is:

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

Solving equations (1) and (2) for K gives:

$$K = P_D \cdot (T_A + 273°C) + \theta_{JA} \cdot P_D^2 \quad (3)$$

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.

The total thermal resistance of a package (θ_{JA}) can be separated into two components, θ_{JC} and θ_{CA}, representing the barrier to heat flow from the semiconductor junction to the package (case) surface (θ_{JC}) and from the case to the outside ambient (θ_{CA}). These terms are related by the equation:

$$\theta_{JA} = \theta_{JC} + \theta_{CA} \quad (4)$$

θ_{JC} is device related and cannot be influenced by the user. However, θ_{CA} is user dependent and can be minimized by such thermal management techniques as heat sinks, ambient air cooling and thermal convection. Thus, good thermal management on the part of the user can significantly reduce θ_{CA} so that θ_{JA} approximately equals θ_{JC}. Substitution of θ_{JC} for θ_{JA} in equation (1) will result in a lower semiconductor junction temperature.

Values for thermal resistance presented in this document, unless estimated, were derived using the procedure described in Motorola Reliability Report 7843, "Thermal Resistance Measurement Method for MC68XX Microcomponent Devices," and are provided for design purposes only. Thermal measurements are complex and dependent on procedure and setup. User derived values for thermal resistance may differ.

DC ELECTRICAL CHARACTERISTICS (V_{CC} = 5.0 Vdc ±5%, GND = 0 Vdc, C_L = 100 pF, T_A = 0°C to 70°C, unless otherwise noted)

Characteristic	Symbol	Min	Max	Unit
Input High Voltage	V_{IH}	2.0	—	V
Input Low Voltage	V_{IL}	—	0.8	V
Input Leakage Current (V_{in} = 0 to 5.25 V)	I_{IL}	—	10	μA
Output High Voltage (I_{Load} = −100 μA)	V_{OH}	2.4	—	V
Output Low Voltage (I_{Load} = −1.6 mA)	V_{OL}	—	0.4	V
Output Leakage Current (Off-State) (V_{out} = 0 to 5.25 V)	I_{OL}	—	10	μA
Internal Power Dissipation (V_{CC} = 5.25 V, T_A = 0°C)	P_{INT}	—	750	mW
Input Capacitance (V_{in} = 0 V, f = 1.0 MHz)	C_{in}	—	20	pF
Output Capacitance (V_{out} = 0 V, f = 1.0 MHz)	C_{out}	—	20	pF

AC ELECTRICAL CHARACTERISTICS (V_{CC} = 5.0 Vdc ±5%, GND = 0 Vdc, C_L = 100 pF, T_A = 0°C to 70°C, unless otherwise noted)

Characteristic		Symbol	MC68652 MC2652 Min	MC68652 MC2652 Max	MC68652–2 MC2652–2 Min	MC68652–2 MC2652–2 Max	Unit
Pulse Width, Clock Low		PW_{CL}	490	–	240	–	ns
Pulse Width, Clock High	MM = 0	PW_{CH}	340	–	165	–	ns
	MM = 1		490	–	240	–	ns
Pulse Width, RESET		PW_{RES}	250	–	250	–	ns
Pulse Width, DBEN		PW_{DBEN}	250	m*	200	m*	ns
Receiver Serial Data Setup Time		t_{RxS}	150	–	150	–	ns
Receiver Serial Data Hold Time		t_{RxH}	150	–	150	–	ns
Address/Control Setup Time		t_{ACS}	50	–	50	–	ns
Address/Control Hold Time		t_{ACH}	0	–	0	–	ns
Data Bus Setup Time (Write)		t_{DS}	50	–	50	–	ns
Data Bus Hold Time (Write)		t_{DH}	0	–	0	–	ns
Data Bus Delay Time (Read)		t_{DD}	–	200	–	170	ns
Transmit Serial Data Delay Time		t_{TxD}	–	325	–	250	ns
DBEN to DBEN Delay Time		t_{DBEND}	200	–	200	–	ns
Data Bus Float Time (Read)		t_{DF}	–	150	–	150	ns
Serial Data Clock Frequency (TxC, RxC)		f_c	–	1.0	–	2.0	MHz

*m = TxC negated and applies to writing to $TDSR_H$ only.

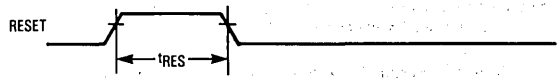
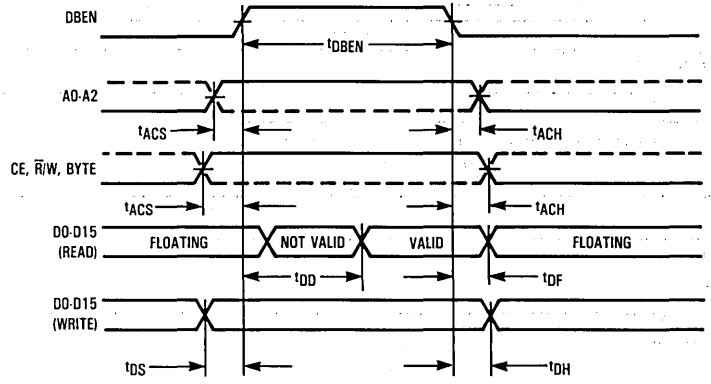
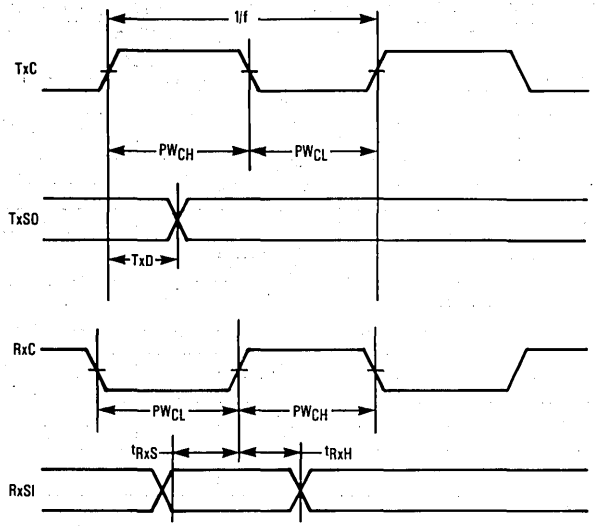


Figure 1. Reset Timing Diagram



NOTE:
Timing measurements are referenced to and from a low voltage of 0.8 volt and a high voltage of 2.0 volts, unless otherwise noted. The voltage swing through this range should start outside, and pass through, the range such that the rise or fall will be linear between 0.8 volt and 2.0 volts.

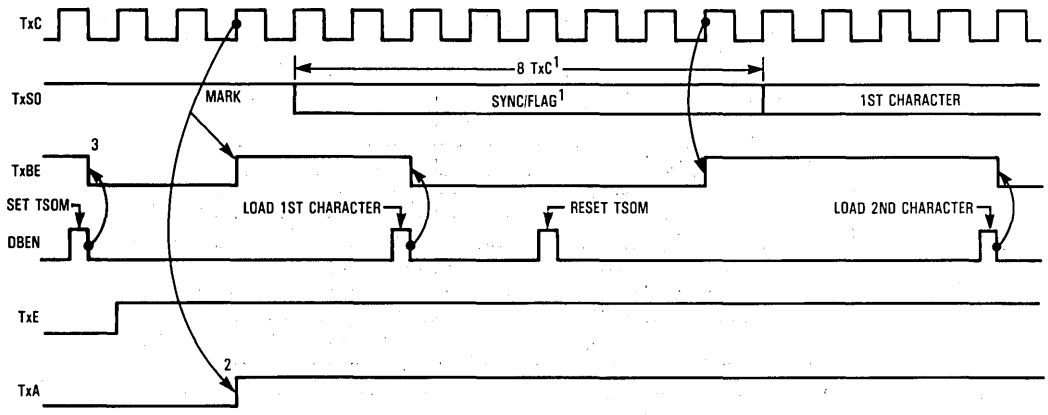
Figure 2. Read/Write Bus Timing Diagram



NOTE:
Timing measurements are referenced to and from a low voltage of 0.8 volt and a high voltage of 2.0 volts, unless otherwise noted. The voltage swing through this range should start outside, and pass through, the range such that the rise or fall will be linear between 0.8 volt and 2.0 volts.

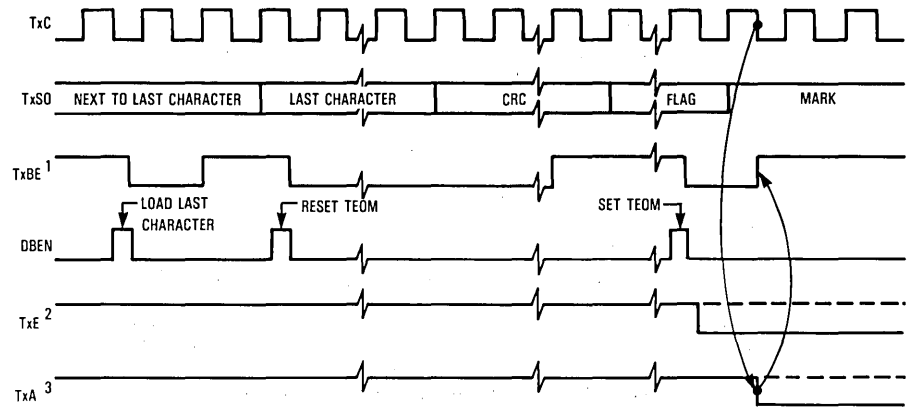
Figure 3. Serial Clock and Data Timing Diagram

6



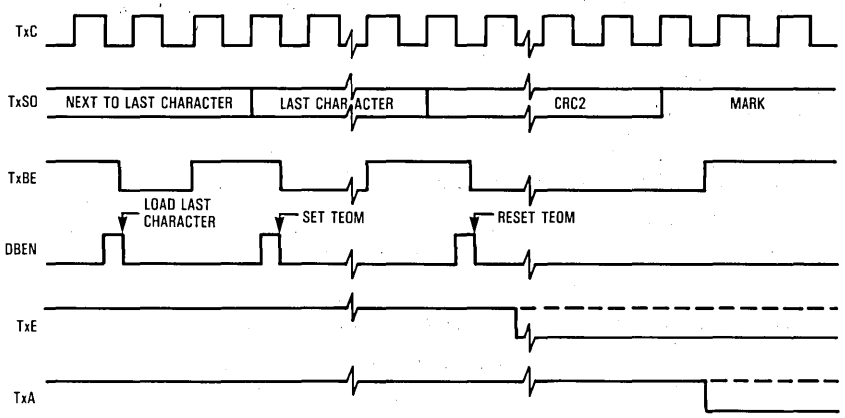
NOTES:
1. SYNC may be 5 to 8 bits and will contain parity bit as specified.
2. TxA is asserted relative to TxC rising edge after TSOM has been set and TxE has been asserted.
3. TxBE is negated relative to DBEN falling edge on the first write transfer into TDSR. It is reasserted one TxC time before the first bit of the transmitted SYNC/FLAG. TxBE is then negated relative to DBEN falling edge when writing into $TDSR_H$ and/or $TDSR_L$. It is reasserted on the rising edge of the TxC that corresponds to the transmission of the last bit of each character except in BOP mode when the CRC is sent as the next character (see Figure 5).

Figure 4. Transmit—Start of Message Timing Diagram



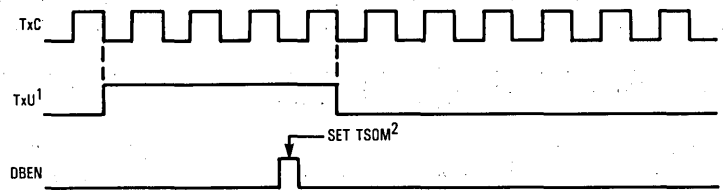
NOTES:
1. TxBE is negated relative to the falling edge of DBEN corresponding to loading $TDSR_{H/L}$. It is asserted one TxC cycle before character transmission begins and also when TxA is negated.
2. TxE can be negated before clearing TEOM if TxBE (corresponding to the closing FLAG) is asserted. Alternatively TxE can remain asserted and a new message initiated.
3. TxA is negated after TxE is negated and 1 1/2 TxC cycles after the last bit of the closing FLAG has been transmitted.

Figure 5. Transmit—End of BOP Message Timing Diagram



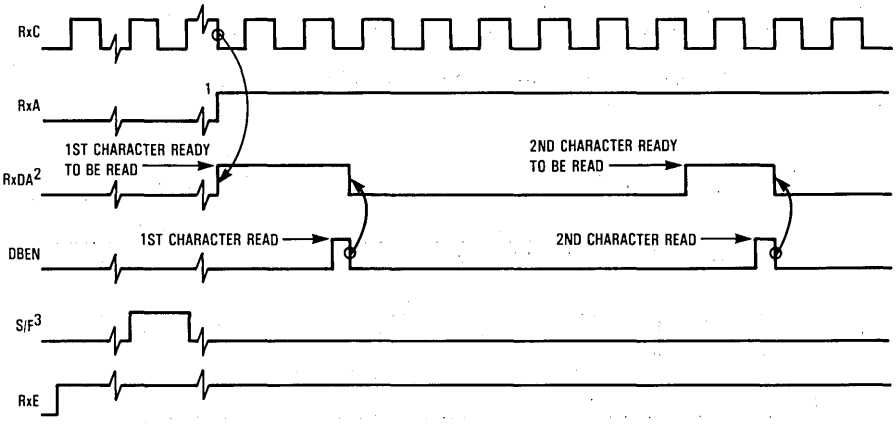
NOTE:
1. When MC68652/MC2652-generated CRC is not required, TEOM should only be set if SYNCs are to follow the message block. In that case, TxE should be negated in response to TxBE (which corresponds to the start of transmission of the last character). When CRC is required, TxE must be negated before CRC transmission is complete. Otherwise, the contents of TxDB will be shifted out on TxSO. This facilitates transmission of contiguous messages.

Figure 6. Transmit—End of BCP Message Timing Diagram



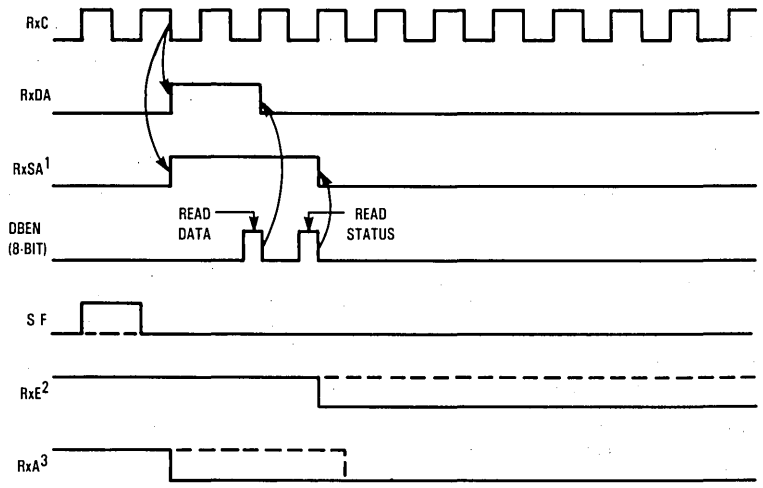
NOTES:
1. TxU is asserted relative to TxC falling edge if TxBE has not be serviced after n-1/2 TxC times (where n equals transmit character length). TxU is negated on the TxC falling edge following setting of the TSOM command.
2. An underrun will occur at the next character boundary if TEOM is cleared and the transmitter remains enabled, unless the TSOM command is set or a character is loaded into the TxDB.

Figure 7. Transmit—Underrun Timing Diagram



NOTES:
1. RxA is asserted relative to the falling edge of RxC when RxE is asserted and:
   a. A data character following two SYNC's is in RxDB (BCP mode).
   b. The character following FLAG is in RxDB (BOP primary station mode).
   c. The character following FLAG is in RxDB and matches the secondary station address or all parties address (BOP secondary station mode).
2. TxDA is asserted on RxC falling edge when a character in RxDB is ready to be read. It is asserted before RxSA is negated on the falling edge of DBEN when RxDB is read.
3. S/F is asserted relative to the rising edge of RxC anytime a SYNC (BCP) or FLAG (BOP) is detected.

Figure 8. Receive—Start of Message Timing Diagram



NOTES:
1. At the end of a BOP message, RxSA is asserted when FLAG detection (S/F = 1) forces the setting of REOM. The processor should read the last data character ($RDSR_L$) and status ($RDSR_L$) which negates RxDA and RxSA, respectively. For BCP end of message, RxSA must not be asserted and S/F must equal zero. The processor should read the last data character and status.
2. RxE must be negated for BCP with non-contiguous messages. It may be left asserted at the end of a BOP message (see **BOP Mode**).
3. RxA is negated relative to the falling edge of RxC after the closing FLAG of a BOP message (REOM = 1 and RxSA is asserted) or when RxE is negated.

Figure 9. Receive—End of Message Timing Diagram

SIGNAL DESCRIPTION

The following paragraphs provide a brief description of the input and output signals.

NOTE

The terms **assertion** and **negation** are used extensively. This is done to avoid confusion when dealing with a mixture of "active-low" and "active-high" signals. The term assert or assertion is used to indicate that a signal is active or true, independent of whether that level is represented by a high or low voltage. The term negate or negation is used to indicate that a signal is inactive or false.

DATA BUS (DB0-DB15)

These bidirectional three-state data lines transfer data, control, and status information between the CPU and the MPCC. The DB0-DB7 signal lines contain data while the DB8-DB15 signal lines contain control and status information. Corresponding bits of the high and low order bytes can be wire-ORed onto an 8-bit bus. These lines are three-stated if either chip enable (CE) or data bus enable (DBEN) are negated.

ADDRESS BUS (A0-A2)

These inputs select the internal register being addressed. The four 16-bit registers can be addressed on a word or byte basis. Accesses of the four addressable registers can be either word accesses (16-bit data transfers) or byte accesses (8-bit data transfers) depending upon the state of BYTE.

BYTE

Single byte (8 bits) data bus transfers are specified when this input is asserted. When negated, word (16 bits) data bus transfers are specified.

CHIP ENABLE (CE)

This active-high input permits a data bus operation when asserted.

READ/WRITE ($\overline{R}$/W)

This input signal controls the direction of data bus transfer. A high level on this pin indicates the transfer of data from the data bus to the addressed register. A low level on the pin indicates the transfer of data from the addressed register to the data bus.

DATA BUS ENABLE (DBEN)

The active-high input signal should be asserted after A0-A2, CE, BYTE, and $\overline{R}/W$ are valid. During a read, the three-stated data bus (DB) is enabled with information from the addressed register. During a write, the data appearing on the data bus is loaded into the addressed register and transmitter buffer empty (TxBE) will be negated if the transmit data/status register (TDSR) was addressed.

RESET

This active-high input, when asserted, initializes all internal registers (to zero) and timing.

MAINTENANCE MODE (MM)

This active-high input signal, when asserted, internally gates the transmitter serial output (TxSO) to the receiver serial input (RxSI) and the transmitter clock input (TxC) to the receiver clock input (RxC), thus placing the MPCC in a maintenance mode (for off-line diagnostic purposes). When in this mode, the RxC and RxSI inputs are ignored and the TxSO output is held in the mark (high) state.

RECEIVER ENABLE (RxE)

This active-high input, when asserted, permits the processing of RxSI data. When negated, receiver logic is disabled and all receiver registers and timing are initialized.

RECEIVER ACTIVE (RxA)

This active-high output signal is asserted when the first data character of a message is ready for the processor. In the BOP mode, this character is the address. The received address must match the secondary station address if the MPCC is a secondary station. In BCP mode, if strip-SYNC (parameter control sync/address register (PCSAR bit 13) is set, the first non-SYNC character is the first data character; if strip-SYNC is zero, the character following the second SYNC is the first data character. In the BOP mode, the closing FLAG negates RxA. In the BCP mode, RxA is negated by the negation of RxE.

RECEIVER DATA AVAILABLE (RxDA)

This active-high output signal is asserted when an assembled character is in the receive data/status register ($RDSR_L$) and is ready to be presented to the processor. RxDA is negated when $RDSR_L$ is read. Note that RxDA is a possible interrupt signal.

RECEIVER CLOCK (RxC)

The RxC(1X) input provides timing for the receiver logic. The positive going edge shifts serial data into the receiver shift register (RxSR) from the receiver serial input (RxSI).

SYNC/FLAG (S/F)

This active-high output signal is asserted when a SYNC or FLAG character has been detected. This signal is negated one RxC clock cycle later.

RECEIVER STATUS AVAILABLE (RxSA)

This active-high output signal is asserted when there is a zero-to-one transition of any bit in $RDSR_H$ except for the receiver start of message (RSOM) bit. It is negated when $RDSR_H$ is read. Note that RxSA is a possible interrupt signal.

RECEIVER SERIAL INPUT (RxSI)

This input signal is the received serial data (mark = 1, space = 0).

TRANSMITTER ENABLE (TxE)

This active-high input signal, when asserted, enables the transmitter data path between the transmit data/status register (TDSR) and the transmitter serial output pin (TxSO). When negated, TxSO is driven to the mark (high) state and the transmitter active output pin (TxA) is negated after the closing FLAG (BOP) or last character (TCP) is sent out the TxSO pin.

TRANSMITTER ACTIVE (TxA)

This active-high output signal is asserted after TSOM (TSDR bit 8) is set and TxE is asserted. TxA is negated when TxE is negated and the closing FLAG (BOP) or last character (BCP) has been output on TxSO.

TRANSMITTER BUFFER EMPTY (TxBE)

This active-high output signal is asserted when TDSR is ready to be loaded with new control information or data. TxBE is negated when new control information or data is loaded into TDSR. Note that TxBE is a possible interrupt signal.

TRANSMITTER UNDERRUN (TxU)

This active-high output signal is asserted during a transmit sequence when the service of TxBE has been delayed for one character time. This indicates the processor is not keeping up with the transmitter. Line fill depends on bit 11 of PCSAR. TxU is negated by asserting RESET or setting TSOM (TDSR bit 8), followed by the falling edge of TxC. Note that TxU is a possible interrupt signal.

TRANSMITTER CLOCK (TxC)

TxC(1X) provides timing for the transmitter logic. The positive going edge of this input shifts data out of the TxSR to TxSO.

TRANSMITTER SERIAL OUTPUT (TxSO)

This output is the transmitted serial data (mark = 1, space = 0).

V_{CC} AND GND

Power is supplied to the MPCC using these pins. V_{CC} is the +5 volt power supply and GND is the 0 volt reference ground.

FUNCTIONAL DESCRIPTION

The MPCC can be functionally partitioned into receiver logic, transmitter logic, registers that can be read or loaded by the processor, and data bus control circuitry. Table 1 provides a description of the registers, Table 2 outlines the error control, and Table 3 highlights the special characters found in the MPCC.

The register bit formats are shown in Figure 10 while the receiver and transmitter data paths are depicted in Figures 11 and 12.

Table 1. Register Description

Registers		No. of Bits	Description*
Addressable			
PCSAR	Parameter Control Sync/Address Register	16	$PCSAR_H$ and PCR contain parameters common to the receiver and transmitter. $PCSAR_L$ contains a programmable SYNC character (BCP) or secondary station address (BOP).
PCR	Parameter Control Register	8	
RDSR	Receive Data/Status Register	16	$RDSR_H$ contains receiver status information. $RDSR_L$ = RxDB contains the received assembled character.
TDSR	Transmit Data/Status Register	16	$TDSR_H$ contains transmitter command and status information. TDSRL = TxDB contains the character to be transmitted.
Internal			
CCSR	Control Character Shift Register	8	These registers are used for character assembly (CCSR, HSR, RxSR), disassembly (TxSR), and CRC accumulation/generation (RxCRC, TxCRC).
HSR	Holding Shift Register	16	
RxSR	Receiver Shift Register	8	
TxSR	Transmitter Shift Register	8	
RxCRC	Receiver CRC Accumulation Register	16	
TxCRC	Transmitter CRC Generation Register	16	

*H = High Byte (Bits 15-8)
L = Low Byte (Bits 7-0)

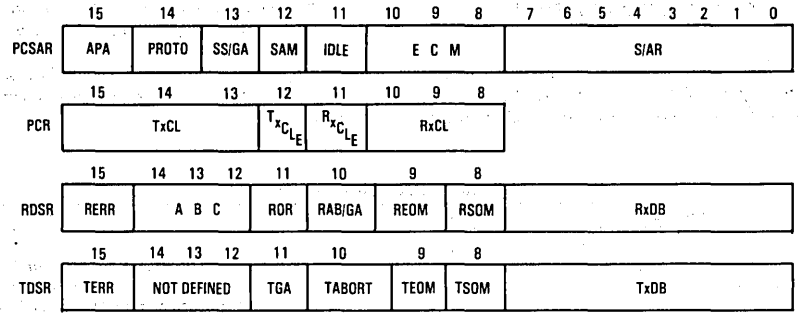
Table 2. Error Control

Character	Description
FCS	Frame Check Sequence is transmitted/received as 16 bits following the last data character of a BOP message. The divisor is usually CRC-CCITT ($X^{16}+X^{12}+X^5+1$) with dividend preset to 1s but can be otherwise determined by ECM. The inverted remainder is transmitted as the FCS.
BCC	Block Check Character is transmitted/received as two successive characters following the last data character of a BCP message. The polynomial is CRC-16 ($X^{16}+X^{15}+X^2+1$) or CRC-CCITT with dividend preset to 0s (as specified by ECM). The true remainder is transmitted as the BCC.

Table 3. Special Characters

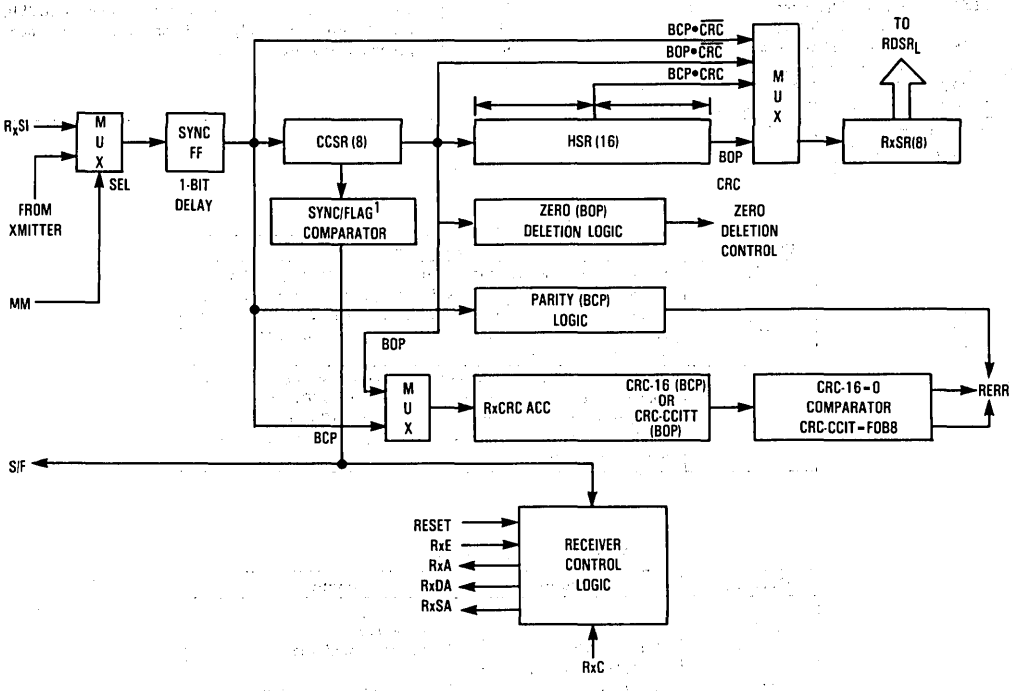
Operation	Bit Pattern	Function
BOP		
FLAG	01111110	Frame Message
ABORT	1111111 Generation 01111111 Detection	Terminate Communication
GA	01111111	Terminate Loop Mode Repeater Function
Address	$(PCSAR_L)$[1]	Secondary Station Address
BCP		
SYNC	$(PCSAR_L)$ or (TxDB)[2] Generation	Character Synchronization

NOTES:
1. (∂) refers to contents of ∂.
2. For IDLE = 0 or 1 respectively.



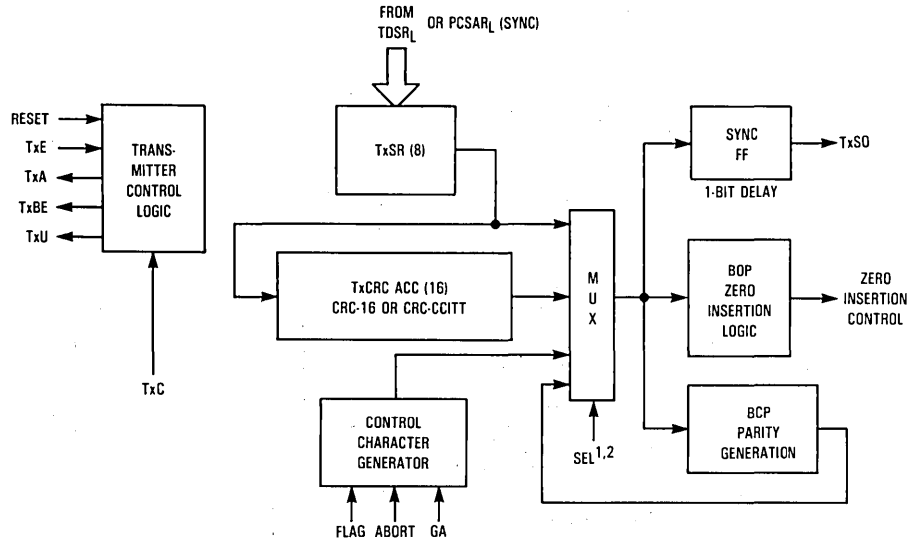
NOTE:
Refer to Table 1 for mnemonics and description.

Figure 10. Short Form Register Bit Formats



NOTES:
1. Detected in SYNC FF and 7 MS bits of CCSR.
2. In BOP mode, a minimum of two data characters must be received to activate the receiver.

Figure 11. MPCC Receiver Data Path



NOTES:
1. TxCRC is selected if TEOM = 1 and the last data character has been shifted out of TxSR.
2. In BCP, parity selected will be generated after each character is shifted out of TxSR.

Figure 12. MPCC Transmitter Data Path

RECEIVER OPERATION

After initializing the parameter control registers (PCSAR and PCR), the RxE input must be asserted to enable the receiver data path. The serial data on the RxSI is synchronized and shifted into an 8-bit control character shift register (CCSR) on the rising edge of RxC. A comparison between CCSR contents and the FLAG (BOP) or SYNC (BCP) characters is made until a match is found. At that time, the S/F output is asserted for one RxC time and the 16-bit holding shift register (HSR) is enabled. The receiver then operates as described in the following paragraphs.

BOP Mode

A flowchart of receiver operation in BOP mode appears in Figure 13. Zero deletion (after five ones are received) is implemented on the received serial data so that a data character will not be interpreted as a FLAG, ABORT, or GA. Bits following the FLAG are shifted through CCSR, HSR, and into RxSR. A character will be assembled in the RxSR and transferred to the RDSR$_L$ for presentation to the processor. At that time the RxDA output will be asserted and the processor must take the character no later than one RxC time after the next character is assembled in the RxSR. If not, an overrun (RSDR bit 11 = 1) will occur and succeeding characters will be lost.

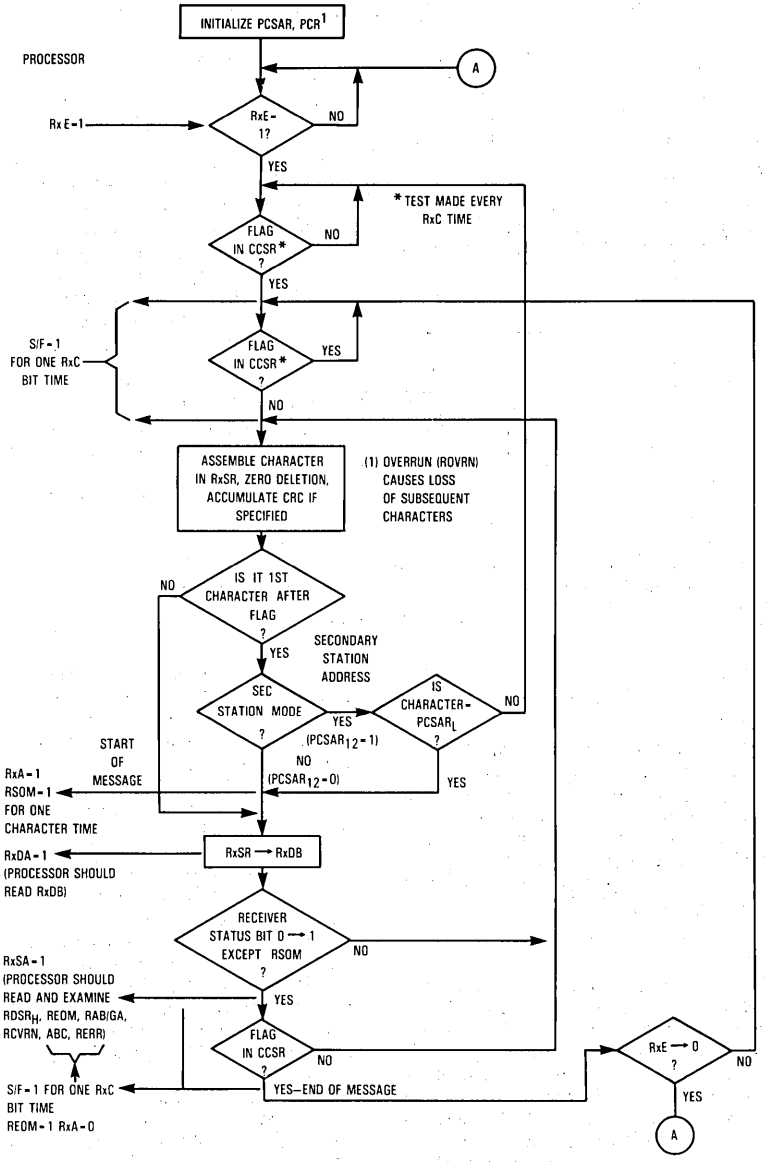
The first character following the FLAG is the secondary station address. If the MPCC is a secondary station (PCSAR bit 12 = 1), the contents of RxSR are compared with the address stored in PCSAR$_L$. A match indicates the forthcoming message is intended for the station; the RxA output is asserted, the character is loaded into RDSR$_L$, RxDA is asserted, and the receive start of message bit (RSOM) is set. No match indicates that another station is being addressed and the receiver searches for the next FLAG.

If the MPCC is a primary station (PCSAR bit 12 = 0), no secondary address check is made; RxA is asserted and RSOM is set once the first non-FLAG character has been loaded into RDSR$_L$ and RxDA has been asserted. Extended address field can be supported by software if PCSAR$_{12}$ = 0.

When the eight bits following the address character have been loaded into RDSR$_L$ and RxDA has been asserted, RSOM will be cleared. The processor should read this 8-bit character and interpret it as the control field.

Received serial data that follows is read and interpreted as the information field by the processor. It will be assembled into character lengths as specified by PCR bits 8-10. As before, RxDA is asserted each time a character has been transferred into RDSR$_L$ and is negated when RDSR$_L$ is read by the processor. RDSR$_H$ should only be read when RxSA is asserted. This occurs on a zero-to-one transition of any bit in RDSR$_H$ except for RSOM. RxSA and all bits in RDSR$_H$ except RSOM are cleared when RDSR$_H$ is read. The processor should check RDSR bits 9-15 each time RxSA is asserted. If RDSR bit 9 is set, then RDSR bits 12-15 should be examined.

Receiver character length may be changed dynamically in response to RxDA; read the character in RxDB and write the new character length into RxCL. The character length will be changed on the next receiver character boundary. A received residual (short) character will be transferred into RxDB after the previous character in RxDB has been read, i.e., there will not be an overrun. In general the last two characters are protected from underrun.



NOTE:
RxE must be negated during initialization of PCSAR and PCR.

Figure 13. BOP Receiver Flowchart

The CRC-CCITT, if specified by PCSAR bits 8-10 is accumulated in RxCRC on each character following the FLAG. When the closing FLAG is detected in the CCSR, the received CRC is the 16-bit HSR. At that time, the receive end of message bit (REOM) will be set; RxSA and RxDA will be asserted. The processor should read the last data character in $RDSR_L$ and the receiver status in RDSR bits 9-15. If RSDR bit 15 equals one, there has been a transmission error; the accumulated CRC-CCITT is incorrect. If RDSR bits 12-15 equal zero, the last data character is not of prescribed length. Neither the received CRC nor closing FLAG are presented to the processor. The processor may negate RxE or leave it asserted at the end of the received message.

BCP Mode

The operation of the receiver in BCP mode is shown in Figure 14. The receiver initially searches for two successive SYNC characters, of length specified by PCR bits 8-10, that match the contents of $PCSAR_L$. The next non-SYNC character or next SYNC character, if stripping is not specified (PCSAR bit 13 = 0), causes RxA to be asserted, and enables the receiver data path. Once enabled, all characters are assembled in RxSR and loaded in $RDSR_L$. RxDA is asserted when a character is available in $RDSR_L$. RxSA is asserted on a zero-to-one transition of any bit in $RDSR_H$. The signals are negated when $RDSR_L$ or $RDSR_H$ are read respectively.

If CRC-16 error control is specified by PCSAR bits 8-10, the processor must determine the last character received prior to the CRC field. When that character is loaded into $RDSR_L$ and RxDA is asserted, the received CRC will be in CCSR and HSR_L. To check for a transmission error, the processor must read the receiver status ($RDSR_H$) and examine RDSR bit 15. This bit will be set for one character time if an error-free message has been received. If RDSR bit 15 equals zero, the CRC-16 is in error. The state of RDSR bit 15 in BCP CRC mode does not assert RxSA. Note that this bit should be examined only at the end of a message. The accumulated CRC will include all characters starting with the first non-SYNC character if PCSAR bit 13 equals one, or the character after the opening two SYNC's if PCSAR bit 13 equals zero. This necessitates external CRC generation/checking when supporting IBM's BISYNC. This can be accomplished using the MC68653 polynomial generator/checker.

If VRC had been selected for error control, parity (odd or even) is regenerated on each character and checked when the parity bit is received. A discrepancy causes RDSR bit 15 to be set and RxSA to be asserted. This must be sensed by the processor. The received parity bit is stripped before the character is presented to the processor.

When the processor has read the last character of the message, it should negate RxE which disables the receiver logic and initializes all receiver registers and timing.

TRANSMITTER OPERATION

After the parameter control registers (PCSAR and PCR) have been initialized, TxSO is held at mark until transmit start of message (TSOM) (TDSR bit 8) is set and TxE is asserted. Then, transmitter operation depends on protocol mode.

BOP Mode

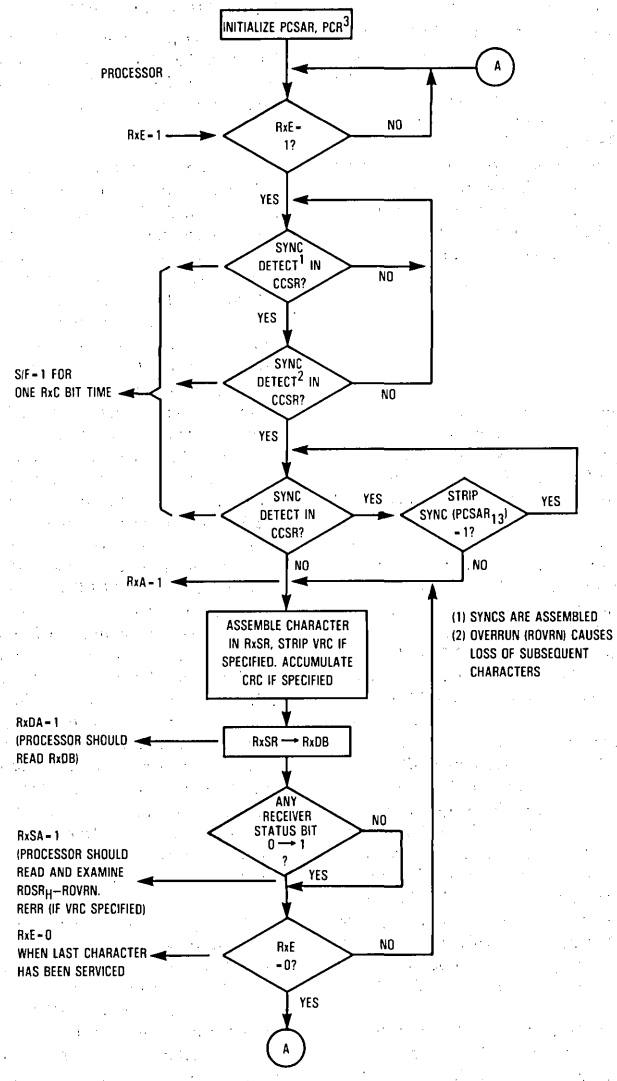
Transmitter operation for BOP mode is shown in Figure 15. A FLAG is sent after the processor sets the transmit start of message bit (TSOM) and asserts TxE. The FLAG is used to synchronize the message that follows. TxA will also be asserted. When TxBE is asserted by the MPCC, the processor should load $TDSR_L$ with the first character of the message. TSOM should be cleared at the same time $TDSR_L$ is loaded (16-bit data bus) or immediately thereafter (8-bit data bus), FLAGs are sent as long as TSOM equals one. For counting the number of FLAGs, the processor should set TSOM in response to the assertion of TxBE.

All succeeding characters are loaded into $TDSR_L$ by the processor when TxBE equals one. Each character is serialized in TxSR and transmitted on TxSO. Internal zero insertion logic stuffs a zero into the serial bit stream after five successive ones are sent. This ensures a data character will not match a FLAG, ABORT, or GA reserved control character. As each character is terminated, the frame check sequence (FCS) is generated as specified by error control mode (PCSAR bits 8-10). The FCS should be the CRC-CCITT polynomial ($X^{16} + X^{12} + X^5 + 1$) preset to ones. If an underrun occurs (processor is not keeping up with the transmitter), TxU and transmitter error (TERR) (TDSR bit 15) will be asserted with ABORT or FLAG used as the TxSO line fill depending on the state of IDLE (PCSAR bit 11). The processor must set TSOM to reset the underrun condition. To retransmit the message, the processor should proceed with the normal start of message sequence.

A residual character of one to seven bits may be transmitted at the end of the information field. In response to TxBE, write the residual character length into TxCL and load TxDB with the residual character. Dynamic alteration of character length should be done in exactly the same sequence. The character length will be changed on the next transmit character boundary.

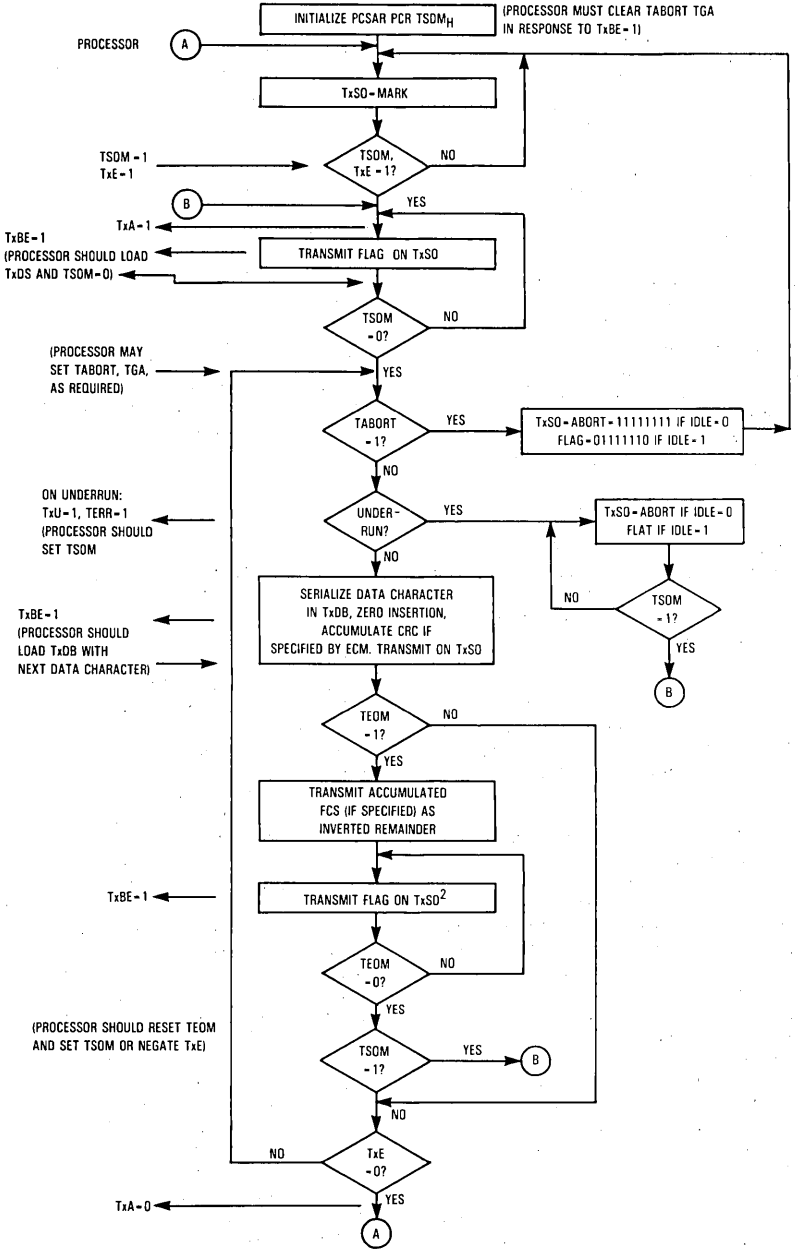
After the last data character has been loaded into $TDSR_L$ and sent to TxSR (TxBE = 1), the processor should set TEOM (TDSR bit 9). The MPCC will finish transmitting the last character followed by the FCS and the closing FLAG. The processor should clear TEOM and negate TxE when the next TxBE has been negated, TxA will be negated one and one-half bit times after the first bit of the closing FLAG has been transmitted. TxSO will be marked after the closing FLAG has been transmitted.

If TxE is asserted and TEOM is high, the transmitter continues to send FLAGs. The processor may initiate the next message by resetting TEOM and setting TSOM, or by loading $TDSR_L$ with a data character and then simply resetting TEOM (without setting TSOM).



NOTES:
1. Test made every RxC time.
2. Test made on Rx character boundary.
3. RxE must be negated during initialization of PCSAR and PCR.

Figure 14. BCP Receiver Flowchart



NOTES:
1. TxE must be negated during initialization of PCSAR and PCR.
2. GA will be transmitted if TGA is set together with TEOM.

Figure 15. BOP Transmitter Flowchart

BCP Operation

Transmitter operation for BCP mode is shown in Figure 16. TxA will be asserted after TSOM = 1 and TxE is asserted. At that time SYNC characters are sent from $PCSAR_L$ or $TDSR_L$ (IDLE = 0 or 1) as long as TSOM = 1. TxBE is asserted at the start of transmission of the first SYNC character. For counting the number of SYNC's, the processor should reassert TSOM in response to the assertion of TxBE. When TSOM = 0 transmission is from $TDSR_L$, which must be loaded with characters from the processor each time TxBE is asserted. If this loading is delayed for more than one character time, an underrun results: TxU and TERR are asserted and the TxSO line fill depends on IDLE (PCSAR bit 11). The processor must set TSOM and retransmit the message to recover. This is not compatible with IBM's BISYNC, so that the user must not underrun when supporting that protocol.

CRC-16, if specified by PCSAR bits 8-10, is generated on each character transmitted from $TDSR_L$ when TSOM = 0. The processor must set TEOM = 1 after the last data character has been set to TxSR (TxBE = 1). The MPCC will finish transmitting the last data character and the CRC-16 transmission, the processor should clear TEOM and negate TxE when the TxBE corresponding to the start of CRC-16 transmission is asserted. When TEOM = 0, the line is marked and a new message may be initiated by setting TSOM and asserting TxE.

If VRC is specified, it is generated on each data character and the data character length must not exceed seven bits. For software LRC or CRC, TEOM should be set only if SYNC's are required at the end of the message block.

Special Case

The capability to transmit 16 spaces is provided for line turnaround in half-duplex mode or for a control recovery situation. This is achieved by setting TSOM and TEOM, clearing TEOM when TxBE is asserted, and proceeding as required.

PROGRAMMING

Prior to initiating data transmission or reception, PCSAR and PCR must be loaded with control information from the processor. The contents of these registers will configure the MPCC for the user's specific data communication environment. These registers should be loaded during power-on initialization and after a reset operation. They can be changed at any time that the respective transmitter or receiver is disabled.

The default value for all registers is zero. This corresponds to BOP, primary station mode, 8-bit character length, FCS = CRC-CCITT preset to ones.

For BOP mode the character length register (PCR) may be set to the desired values during system initialization. The address and control fields will automatically be eight bits. If a residual character is to be transmitted, TxCL should be changed to the residual character length prior to transmission of that character.

DATA BUS CONTROL

The processor must set up the MPCC register address (A2-A0), chip enable (CE), byte select (BYTE), and read/write ($\overline{R}/W$) inputs before each data bus transfer operation.

During a read operation ($\overline{R}/W = 0$), the asserting edge of DBEN will initiate an MPCC read cycle. The addressed register will place its contents on the data bus. If BYTE is asserted, the 8-bit byte is placed on DB8-DB15 or DB0-DB7 depending on the H/L status of the register addressed. Unused bits in $RDSR_L$ are zero. If BYTE is negated, all 16 bits (DB0-DB15) contain MPCC information. The negating edge of DBEN will negate RxDA and/or RxSA if $RDSR_L$ or $RDSR_H$ is addressed respectively. Refer to Table 4.

DBEN acts as the enable and strobe so that the MPCC will not begin its internal read cycle until DBEN is asserted.

During a write operation ($\overline{R}/W = 1$), the data must be stable on DB15-DB8 and/or DB7-DB0 prior to the asserting edge of DBEN. The stable data is strobed into the addressed register by DBEN. TxBE will be negated if the addressed register was $TDSR_H$ or $TDSR_L$.

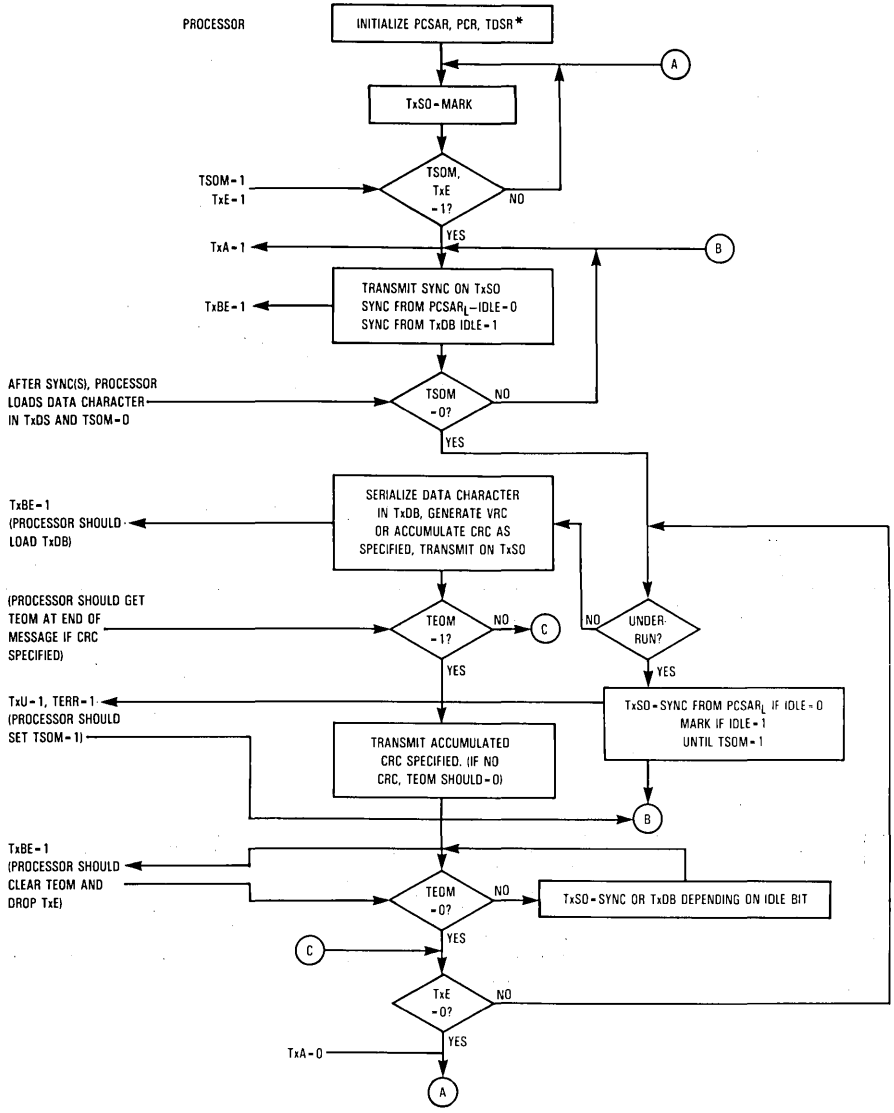
Tables 5, 6, 7, and 8 describe the functions of the parameter control register, the parameter control SYNC/address register, the transmit data/status register, and the receiver data/status register, respectively.

MC68000-TO-MC2652/MC68652 INTERFACE CIRCUIT

The MC68000-to-MC2652/MC68652 interface circuit is shown in Figure 21. It will generate the DBEN select signal required by the MC2652/MC68652 multiple protocol communications controller (MPCC) and the $\overline{DTACK}$ signal required by the MC68000. This allows the MPCC to interface with the MC68000's asynchronous bus.

The MPCC has a 16-bit data bus (D0-D15), three register select signals (A0, A1, and A2), a chip enable (CE), and a read/write signal ($\overline{R}/W$). The data bus is connected directly to the MPU data bus and the register selects are connected to the A1 and A2 address lines of the processor. A0 of the MPCC is generated from $\overline{LDS}$. Together $\overline{UDS}$ and $\overline{LDS}$ determine the width of the transfer on the data bus. If $\overline{UDS}$ and $\overline{LDS}$ are both asserted, the BYTE input is negated and a word is transferred; otherwise the byte input is asserted, indicating byte transfers. The MPU's $\overline{R}/W$ line is inverted and connected to the $\overline{R}/W$ pin of the MPCC. Since DBEN controls all data transfers and operations internal to the MPCC, CE is held asserted and DBEN is used as the chip select.

An access begins when $\overline{AS}$ is asserted by the MPU, indicating a valid address on the address bus. $\overline{UDS}$ and/or $\overline{LDS}$ are then asserted, enabling flip-flop U2 to assert DBEN on the next rising edge of the system clock. The one clock cycle delay between $\overline{SELECT}$ and DBEN allows the MPCC setup time to be met. $\overline{DTACK}$ is then asserted on the next rising edge, forcing the processor to insert four wait states. This guarantees that the MPCC access time is met. At the end of the cycle, $\overline{AS}$ and the data strobes are negated, causing DBEN to be negated. The timing for a MPU write cycle is shown in Figure 22.



*TxE must be asserted during initialization of PCSAR and PCR.

Figure 16. BCP Transmit Flowchart

6

Table 4. MPCC Register Addressing

	A2	A1	A0	Register
BYTE = 0	16-BIT DATA BUS = $DB_{15} - DB_{00}$			
	0	0	X	RDSR
	0	1	X	TDSR
	1	0	X	PCSAR
	1	1	X	PCR*
BYTE = 1	8-BIT DATA BUS = DB_{7-0} or DB_{15-8}**			
	0	0	0	$RDSR_L$
	0	0	1	$RDSR_H$
	0	1	0	$TDSR_L$
	0	1	1	$TDSR_H$
	1	0	0	$PCSAR_L$
	1	0	1	$PCSAR_H$
	1	1	0	PCR_L*
	1	1	1	PCR_H

*PCR lower byte does not exist. It will be all "0"s when read.
**Corresponding high and low order pins must be tied together.

Table 5. Parameter Control Register (PCR)-($\overline{R}$/W)

Bit	Name	Mode	Function
00-07	Not Defined		
08-10	RxCL	BOP/BCP	Receiver character length is loaded by the processor when RxCLE = 0. The character length is valid after transmission of single byte address and control fields have been received.
11	RxCLE	BOP/BCP	Receiver character length enable should be zero when the processor loads RxCL. The remaining bits of PCR are not affected during loading. Always 0 when read.
12	TxCLE	BOP/BCP	Transmitter character length enable should be zero when the processor loads TxCL. The remaining bits of PCR are not affected during loading. Always 0 when read.
13-15	TxCL	BOP/BCP	Transmitter character length is loaded by the processor when TxCLE = 0. Character bit length specification format is identical to RxCL. It is valid after transmission of single byte address and control fields.

10	9	8	Character Length (Bits)
0	0	0	8
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

Table 6. Parameter Control SYNC/Address Register (PCSAR)($\overline{R}$/W)

Bit	Name	Mode	Function
00-07	S/AR	BOP	SYNC Register. Contains the secondary station address if the MPCC is a secondary station. The contents of this register is compared with the first received non-FLAG character to determine if the message is meant for this station.
		BCP	SYNC character is loaded into this register by the processor. It is used for receive and transmit bit synchronization with bit length specified by RxCL and TxCL.
08-10	ECM	BOP/BCP	**Error Control Mode** — **10** **9** **8** — **Suggested Mode** — **Character Length** CRC-CCITT present to 1's — 0 0 0 — BOP — 1-8 CRC-CCITT present to 0's — 0 0 1 — BCP — 8 Not Used — 0 1 0 — — — CRC-16 preset to 0's — 0 1 1 — BCP — 8 VRC odd — 1 0 0 — BCP — 5-7 VRC even — 1 0 1 — BCP — 5-7 Not used — 1 1 0 — — — No error control — 1 1 1 — BCP/BOP — 5-8 ECM should be loaded by the processor during initialization or when both data paths are idle.
11	IDLE		Determines line fill character to be used if transmitter underrun occurs (TxU asserted and TERR set) and transmission of special characters for BOP/BCP.
		BOP	IDLE = 0, transmit ABORT characters during underrun and when TABORT = 1. IDLE = 1, transmit FLAG characters during underrun and when TABORT = 1.
		BCP	IDLE = 0, transmit initial SYNC characters and underrun line fill characters from the S/AR. IDLE = 1, transmit initial SYNC characters from TxDB and marks TxSO during underrun.
12	SAM	BOP	Secondary Address Mode = 1 if the MPCC is a secondary station. This facilitates automatic recognition of the received secondary station address. When transmitting the processor must load the secondary address into TxDB. SAM = 0 inhibits the received secondary address comparison which serves to activate the receiver after the first non-FLAG character has been received.
13	SS/GA		Strip SYNC/Go Ahead. Operation depends on mode.
		BOP	SS/GA = 1 is used for loop mode only and enables GA detection. When a GA is detected as a closing character, REOM and RAB/GA will be set and the processor should terminate the repeater function. SS/GA = 0 is the normal mode which enables ABORT detection. It causes the receiver to terminate the frame upon detection of an ABORT or FLAG.
		BCP	SS/GA = 1, causes the receiver to strip SYNC's immediately following the first two SYNC's detected. SYNC's in the middle of a message will not be stripped. SS/GA = 0, presents any SYNC's after the initial two SYNC's to the processor.
14	PROTO		Determines MPCC Protocol mode
		BOP	PROTO = 0
		BCP	PROTO = 1
15	APA	BOP	All Parties Address. If this bit is set, the receiver data path is enabled by an address field of '11111111' as well as the normal secondary station address.

6

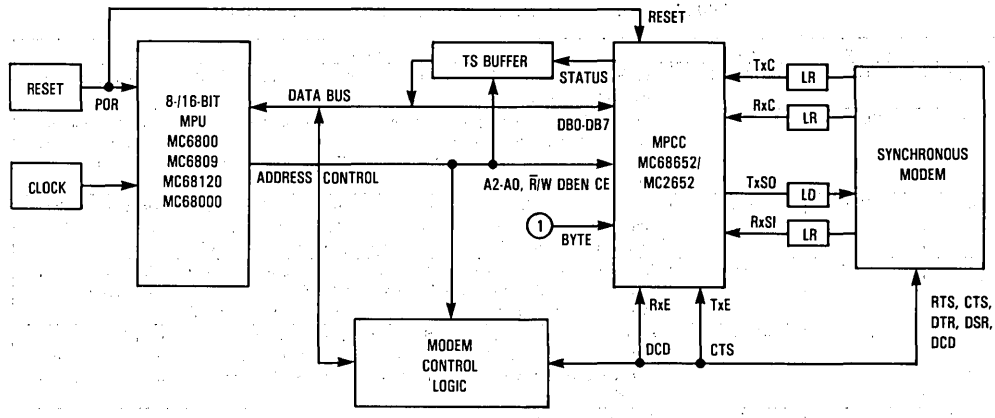
Table 7. Transmit Data/Status Register (TDSR) ($\overline{R}$/W except TDSR Bit 15)

Bit	Name	Mode	Function
00-07	TxDB	BOP/BCP	Transmit Data Buffer. Contains processor loaded characters to be serialized in TxSR and transmitted on TxSO.
08	TSOM		Transmitter Start of Message. Set by the processor to initiate message transmission provided TxE = 1.
		BOP	TSOM = 1 generates FLAGs. When TSOM = 0 transmission is from TxDB and FCS generation (if specified) begins. FCS, as specified by $PCSAR_{8-10}$, should be CRC-CCITT preset to 1's.
		BCP	TSOM = 1 generates SYNCs from $PCSAR_L$ or transmits from TxDB for IDLE = 0 or 1 respectively. When TSOM = 0 transmission is from TxDB and CRC generation (if specified) begins.
09	TEOM		Transmit End of Message. Used to terminate a transmitted message.
		BOP	TEOM = 1 causes the FCS and the closing FLAG to be transmitted following the transmission of the data character in TxSR. FLAGs are transmitted until TEOM = 0 ABORT or GA are transmitted if TABORT or TGA are set when TEOM = 1.
		BCP	TEOM = 1 causes CRC-16 to be transmitted (if selected) followed by SYNCs from $PCSAR_L$ or TxDB (IDLE = 0 or 1). Clearing TEOM prior to the end of CRC-16 transmission (when TxBE = 1) causes TxSO to be marked following the CRC-16. TxE must be dropped before a new message can be initiated. If CRC is not selected, TEOM should not be set.
10	TABORT	BOP	Transmitter Abort = 1 will cause ABORT or FLAG to be sent (IDLE = 0 or 1) after the current character is transmitted. (ABORT = 11111111)
11	TGA	BOP	Transmit Go Ahead (GA) instead of FLAG when TEOM = 1. This facilitates repeater termination in loop mode. (GA = 01111111)
12-14	Not Defined		
15	TERR	(Read Only)	Transmitter Error = 1 indicates the TxDB has not been loaded in time (one character time — ½ TxC period after TxBE is asserted) to maintain continuous transmission. TxU will be asserted to inform the processor of this condition. TERR is cleared by setting TSOM. See timing diagram.
		BOP	ABORT's or FLAG's are sent as fill characters (IDLE = 0 or 1)
		BCP	SYNC's or MARK's are sent as fill characters (IDLE = 0 or 1). For IDLE = 1 the last character before underrun is not valid.

6

Table 8. Receiver Data/Status Register (RDSR) (Read Only)

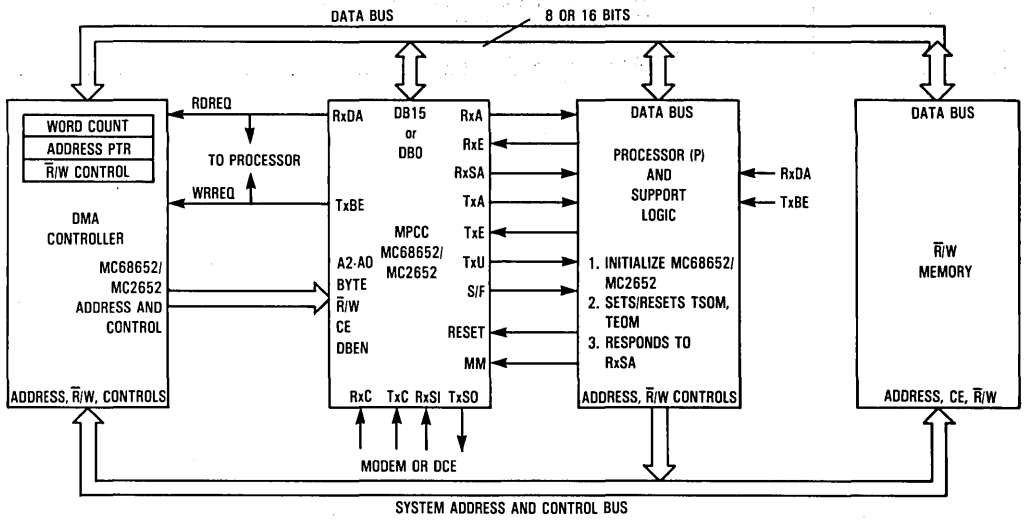
Bit	Name	Mode	Function
00-07	RxDB	BOP/BCP	Receiver Data Buffer. Contains assembled characters from the RxSR. If VRC is specified, the parity bit is stripped.
08	RSOM	BOP	Receiver Start of Message = 1 when a FLAG followed by a non-FLAG has been received and the latter character matches the secondary station address if SAM = 1 RxA will be asserted when RSOM = 1. RSOM resets itself after one character time and has no effect on RxSA.
09	REOM	BOP	Receiver End of Message = 1 when the closing FLAG is detected and the last data character is loaded into RxDB or when an ABORT/GA character is received REOM is cleared on reading $RDSR_H$, reset operation, or dropping of RxE.
10	RAB/GA	BOP	Received ABORT or GA character = 1 when the receiver senses an ABORT character if SS/GA = 0 or a GA character if SS/GA = 1. RAB/GA is cleared on reading $RDSR_H$, reset operation, or dropping of RxE. A received ABORT does not set RxDA.
11	ROR	BOP/BCP	Receiver Overrun = 1 indicates the processor has not read last character in the RxDB within one character time + ½ RxC period after RxDA is asserted. Subsequent characters will be lost. ROR is cleared on reading $RDSR_H$, reset operation, on dropping of RxE.
12-14	ABC	BOP	Assembled Bit Count. Specifies the number of bits in the last received data character of a message and should be examined by the processor when REOM = 1 (RxDA and RxSA asserted). ABC = 0 indicates the message was terminated (by a FLAG or GA) on a character boundary as specified by PCR_{8-10}. Otherwise, ABC = number of bits in the last data character. ABC is cleared when $RDSR_H$ is read, reset operation, or dropping RxE. The residual character is right justified in $RDSR_L$.
15	RERR	BOP/BCP	Receiver Error indicator should be examined by the processor when REOM = 1 in BOP, or when the processor determines the last data character of the message in BCP with CRC or when RxSA is set in BCP with VRC. CRC-CCITT preset to 1's/0's as specified by $PCSAR_{8-10}$: RERR = 1 indicates FCS error (CRC ≠ FOB8 ≠ 0) RERR = 0 indicates FCS received correctly (CRC = FOB8/ = 0) CRC-16 preset to 0's on 8-bit data characters specified by $PSCAR_{8-10}$ RERR = 1 indicates CRC-16 received correctly (CRC = 0). RERR = 0 indicates CRC-16 error (CRC ≠ 0). VRC specified by $PCSAR_{8-10}$: RERR = 1 indicates VCR error RERR = 0 indicates VRC is correct.



NOTES:
1. Possible MPU interrupt requests are: RxDA, RxSA, TxBE, and TxU.
2. Other MC68652/MC2652 status signals and possible uses are:
   S/F line: idle indicator, frame delimiter.
   RxA handshake on RxE, line turn around control.
   TxA handshake on TxE, line turn around control.
3. Line drivers/receivers (LD/LR) convert EIA to TTL voltages and vice versa.
4. RTS should be cleared after the CRC (BCP) or FLAG (BOP) has been transmitted. This forces CTS low and TxE low.
5. Corresponding high and low order bits of DB must be OR tied.

Figure 17. MC68652/MC2652 MPCC Microprocessor Interface

6



NOTE:
For non-DMA operation, TxBE and RxDA are sent to the processor which then loads or reads data characters as required.

Figure 18. DMA/Processor Interface

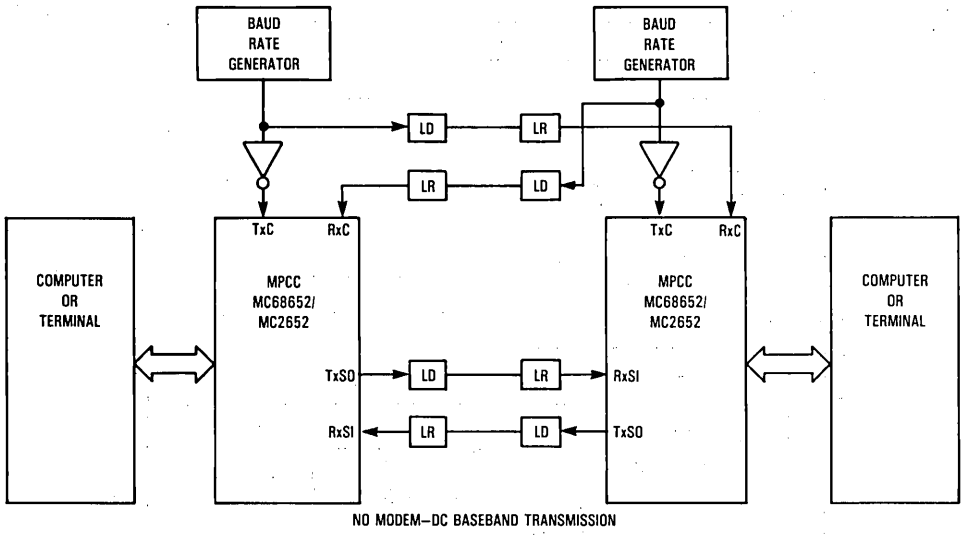


Figure 19. Channel Interface

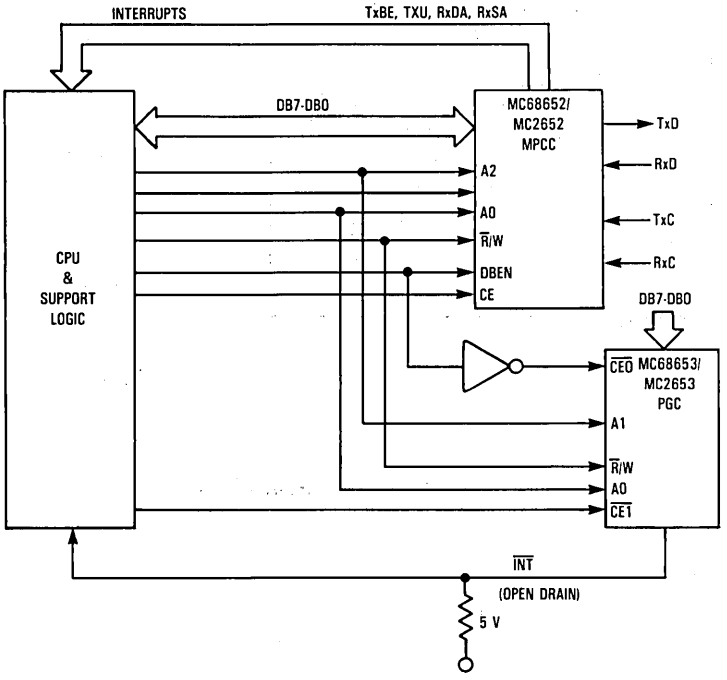


Figure 20. MC68652/MC68653 Interface
Typical Protocols: BISYNC, DDCMP, SDLC, HDLC

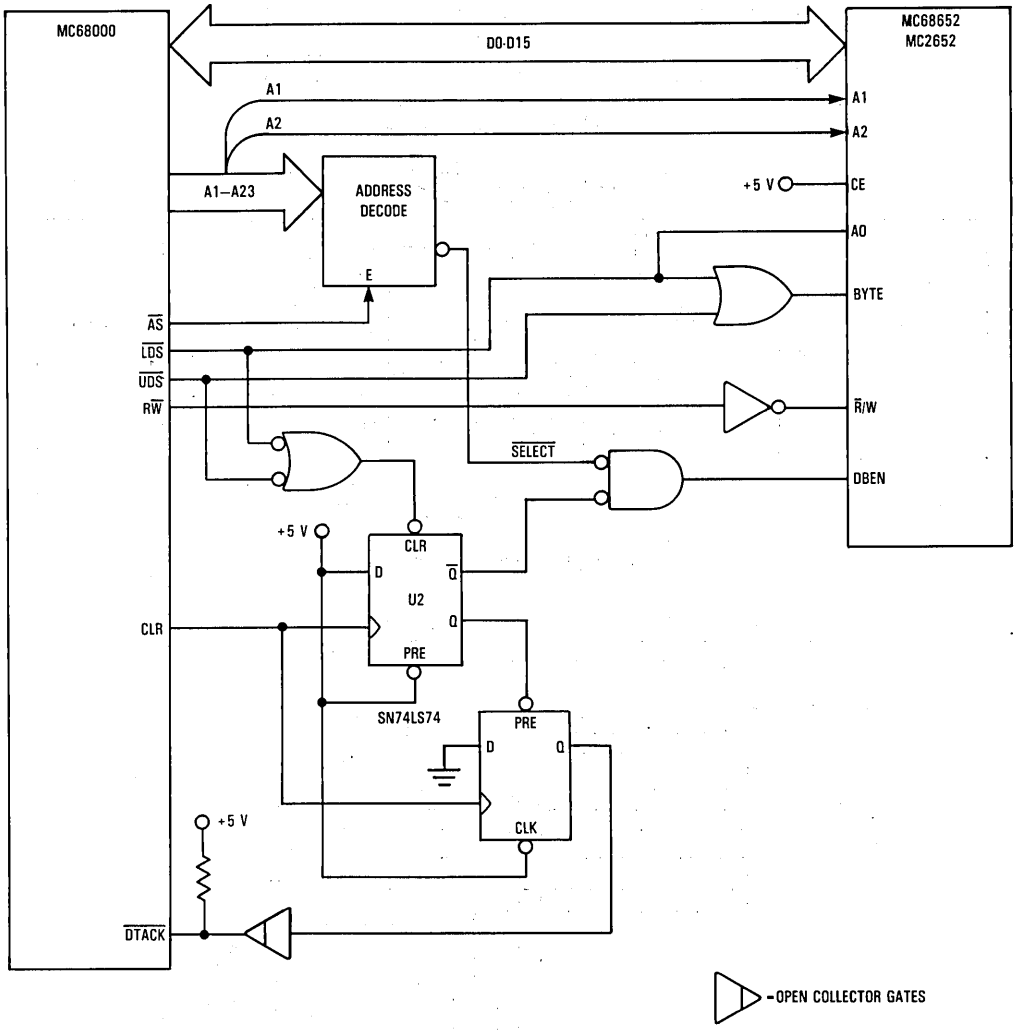


Figure 21. MC68000/MC68652 Interface Circuit

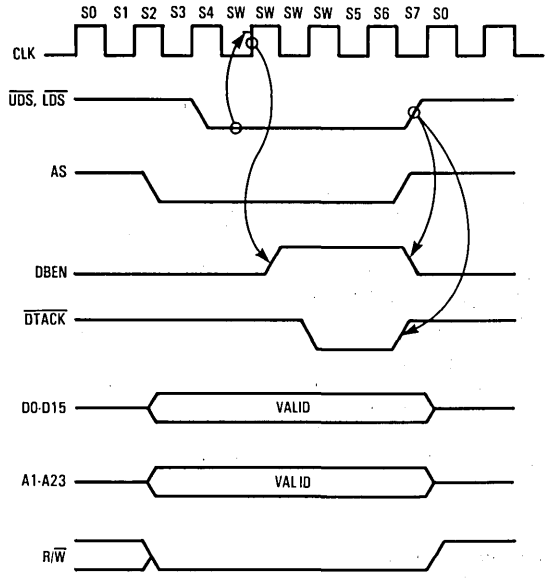
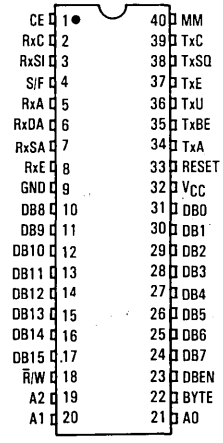


Figure 22. MC68000 Write Cycle Timing

6

PIN ASSIGNMENT



MC68681

TechnicalSummary

Dual Asynchronous Receiver/Transmitter (DUART)

The MC68681 dual universal asynchronous receiver/transmitter (DUART) is part of the M68000 Family of peripherals and directly interfaces to the MC68000 processor via an asynchronous bus structure. The MC68681 consists of eight major sections: internal control logic, timing logic, interrupt control logic, a bidirectional 8-bit data bus buffer, two independent communication channels (A and B), a 6-bit parallel input port, and an 8-bit parallel output port.

Figure 1 illustrates the basic block diagram of the MC68681 and should be referred to during the discussion of its features which include the following:

- M68000 Bus Compatible
- Two Independent Full-Duplex Asynchronous Receiver/Transmitter Channels
- Maximum Data Transfer
  - 1X—1 MB/second
  - 16X—125 kB/second
- Quadruple-Buffered Receiver Data Registers
- Double-Buffered Transmitter Data Registers
- Independently Programmable Baud Rate for Each Receiver and Transmitter Selectable From:
  - 18 Fixed Rates: 50 to 38.4k Baud
  - One User Defined Rate Derived from a Programmable Timer/Counter
  - External 1X Clock or 16X Clock
- Programmable Data Format
  - Five to Eight Data Bits plus Parity
  - Odd, Even, No Parity, or Force Parity
  - One, One and One-Half, or Two Stop Bits Programmable in One-Sixteenth Bit Increments
- Programmable Channel Modes
  - Normal (Full Duplex)
  - Automatic Echo
  - Local Loopback
  - Remote Loopback
- Automatic Wake-up Mode for Multidrop Applications
- Multi-Function 6-Bit Input Port
  - Can Serve as Clock or Control Inputs
  - Change-of-State Detection on Four Inputs
- Multi-Function 8-Bit Output Port
  - Individual Bit Set/Reset Capability
  - Outputs Can be Programmed to be Status/Interrupt Signals
- Multi-Function 16-Bit Programmable Counter/Timer
- Versatile Interrupt System
  - Single Interrupt Output with Eight Maskable Interrupting Conditions
  - Interrupt Vector Output on Interrupt Acknowledge
  - Output Port Can be Configured to Provide a Total of Up to Six Separate Wire-ORable Interrupt Outputs

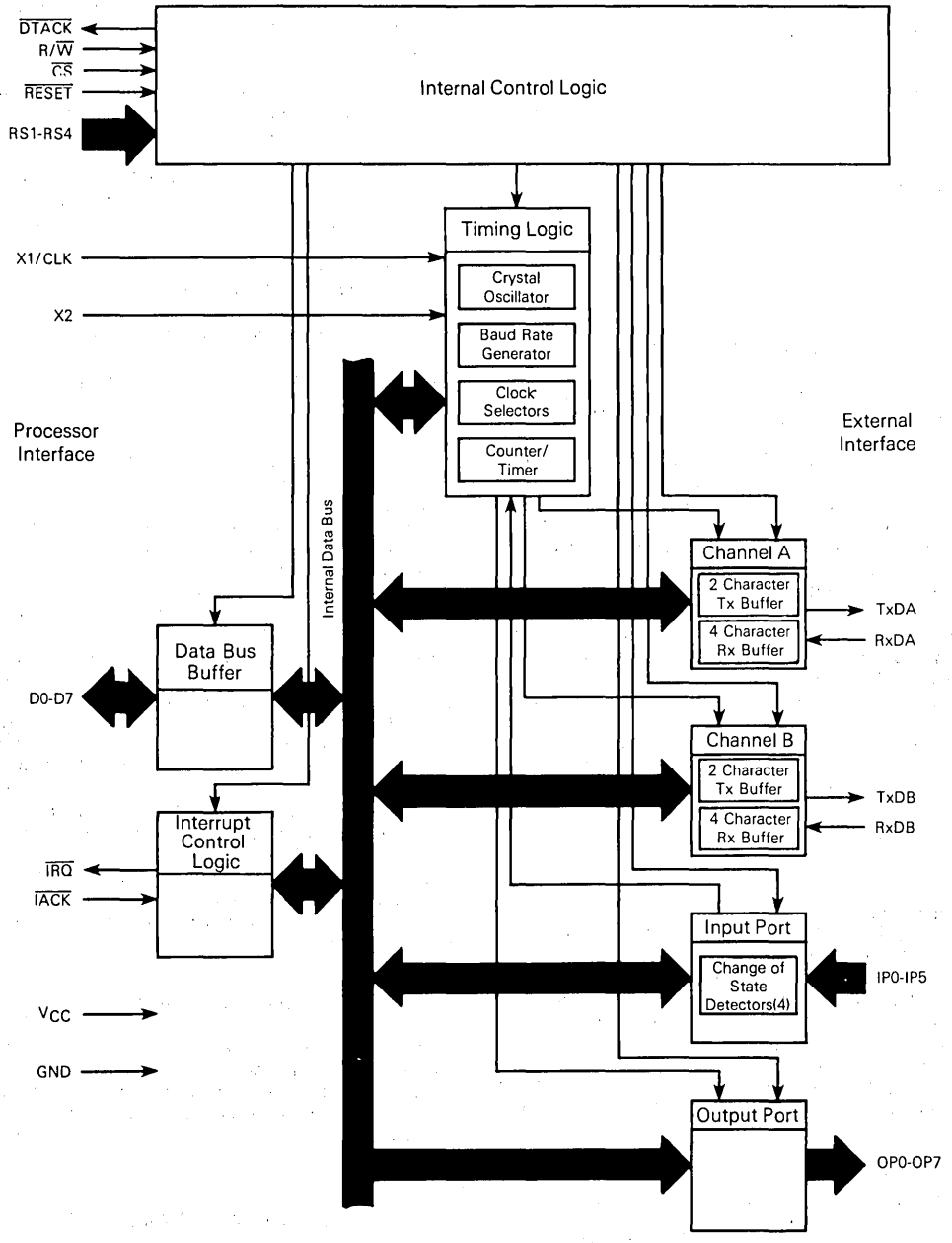


Figure 1. Block Diagram

6

FEATURES (Continued)

- Parity, Framing, and Overrun Error Detection
- False-Start Bit Detection
- Line-Break Detection and Generation
- Detects Break Which Originates in the Middle of a Character
- Start-End Break Interrupt/Status
- On-Chip Crystal Oscillator
- TTL Comaptible
- Single +5 V Power Supply

INTERNAL CONTROL LOGIC

The internal control logic receives operation commands from the central processing unit (CPU) and generates appropriate signals to the internal sections to control device operation. It allows the registers within the DUART to be accessed and various commands to be performed by decoding the four register-select lines (RS1 through RS4). Besides the four register-select lines, there are three other inputs to the internal control logic from the CPU: read/write (R/$\overline{W}$), which allows read and write transfers between the CPU and DUART via the data bus buffer; chip select ($\overline{CS}$), which is the DUART chip select; and reset ($\overline{RESET}$),which is used to initialize or reset the DUART. Output from the internal control logic is the data transfer acknowledge ($\overline{DTACK}$) signal which is asserted during read, write, or interrupt acknowledge cycles. $\overline{DTACK}$ indicates to the CPU that data has been latched on a CPU write cycle or that valid data is present on the data bus during a CPU read cycle or interrupt acknowledge ($\overline{IACK}$) cycle.

TIMING LOGIC

The timing logic consists of a crystal oscillator, a baud-rate generator (BRG), a programmable 16-bit counter/timer (C/T), and four clock selectors. The crystal oscillator operates directly from a 3.6864 MHz crystal connected across the X1/CLK and X2 inputs or from an external clock of the appropriate frequency connected to X1/CLK. The clock serves as the basic timing reference for the baud-rate generator, the counter/timer, and other internal circuits. A clock signal, within the limits given in **ELECTRICAL SPECIFICATIONS**, must always be supplied to the DUART.

The baud-rate generator operates from the oscillator or external clock input and is capable of generating 18 commonly used data communication baud rates ranging from 50 to 38.4k by producing internal clock outputs at 16 times the actual baud rate. The counter/timer can be used in the timer mode to produce a 16X clock for any other baud rate by counting down the crystal clock or external clock. Other baud rates may also be derived by connecting 16X or 1X clocks to certain input port pins which have alternate functions as receiver or transmitter clock inputs. The four clock selectors allow the independent selection, for each receiver and transmitter, of any of these baud rates.

The 16-bit counter/timer (C/T) included within the DUART and timing logic can be programmed to use one of several timing sources as its input. The output of the counter/timer is available to the internal clock selectors and can also be programmed to be a parallel output at OP3. In the timer mode, the counter/timer acts as a programmable divider and can be used to generate a square-wave output at OP3. In the counter mode, the contents of the counter/timer can be read by the CPU and it can be stopped and started under program control. The counter counts down the number of pulses stored in the concatenation of the counter/timer upper register and counter/timer lower register and produces an interrupt. This is a system oriented feature which may be used to keep track of timeouts when implementing various application protocols.

INTERRUPT CONTROL LOGIC

The following registers are associated with the interrupt control logic: interrupt mask register (IMR), interrupt status register (ISR), auxiliary control register (ACR), and interrupt vector register (IVR).

A single active-low interrupt output ($\overline{IRQ}$) is provided which can be used to notify the processor that any of eight internal events has occurred. The interrupt mask register (IMR) can be programmed to select only certain conditions which cause $\overline{IRQ}$ to be asserted while the interrupt status register (ISR) can be read by the CPU to determine all currently active interrupting conditions. When an active-low interrupt acknowledge signal ($\overline{IACK}$) from the processor is assserted while the DUART has an interrupt pending, the DUART will place the contents of the interrupt vector register (IVR) (i.e., the interrupt vector) on the data bus and assert the data transfer acknowledge signal ($\overline{DTACK}$).

In addition, the DUART offers the ability to program the parallel outputs OP3 through OP7 to provide discrete interrupt outputs for the transmitters, the receivers, and the counter/timer.

DATA BUS BUFFER

The data bus buffer provides the interface between the external and internal data buses. It is controlled by the internal control logic to allow read and write data transfer operations to take place between the controlling CPU and DUART by way of the eight parallel data lines (D0 through D7).

COMMUNICATION CHANNELS A AND B

Each communication channel comprises a full-duplex asynchronous receiver/transmitter (UART). The operating frequency for each receiver and each transmitter can be selected independently from the baud-rate generator, the counter/timer, or from an external clock.

The transmitter accepts parallel data from the CPU, converts it to a serial bit stream, inserts the appropriate start, stop, and optional parity bits, and outputs a composite serial stream of data on the TxD output pin. The receiver accepts serial data on the RxD pin, converts this serial input to parallel format, checks for a start bit, stop bit, parity bit (if any), or break condition, and transfers an assembled character to the CPU during read operations.

INPUT PORT

The inputs to this unlatched 6-bit port (IP0 through IP5) can be read by the CPU by performing a read operation. High or low inputs to the input port result in the CPU reading a logic one or logic zero, respectively; that is there is no inversion of the logic level. Since the input port is a 6-bit port, performing a read operation will result in D7 being read as a logic one and D6 reflecting the logic level of $\overline{IACK}$. Besides general-purpose inputs, the inputs to this port can be individually assigned specific auxiliary functions serving the communication channels.

Four change-of-state detectors, also provided within the input port, are associated with inputs IP0, IP1, IP2, and IP3. A high-to-low or low-to-high transition of these inputs lasting longer than 25 to 30 microseconds (best-to-worst case times) will set the corresponding bit in the input port change register (IPCR). The bits are cleared when the register is read by the CPU. Also, the DUART can be programmed so any particular change of state can generate an interrupt to the CPU. The DUART recognizes a level change on an input pin internally only after it has sampled the new level on the pin for two successive pulses of the sampling clock. The sampling clock is 38.4 kHz and is derived from one of the baud-rate generator taps. The resulting sampling period is slightly more than 25 microseconds (this assumes that the clock input is 3.6864 MHz). Subsequently, if the level change occurs on or just before a sampling pulse, it will be recognized internally after 25 microseconds. However, if the level change occurs just after a sampling pulse, it will be sampled the first time after 25 microseconds. Thus, in this case the level change will not be recognized internally until 50 microseconds after the level change took place on the pin.

OUTPUT PORT

This 8-bit multi-purpose output port can be used as a general-purpose output port. Associated with the output port is an output port register (OPR).

All bits of the output port register can be individually set and reset. A bit is set by performing a write operation at the appropriate address with the accompanying data specifying the bits to be set (one equals set and zero equals no change). Similarly, a bit is reset by performing a write operation at another address with the accompanying data specifying the bits to be reset (one equals reset and zero equals no change).

The output port register stores data that is to be output at the output port pins. Unlike the input port, if a particular bit of the output port register is set to a logic one or logic zero the output pin will be at a low or high level, respectively. Thus, a **logic inversion** takes place internal to the DUART with respect to this register. The outputs are complements of the data contained in the output port register.

Besides general-purpose outputs, the outputs can be individually assigned specific auxiliary functions serving the communication channels. The assignment is accomplished by appropriately programming the channel A and B mode registers (MR1A, MR1B, MR2A, and MR2B) and the output port configuration register (OPCR).

SIGNAL DESCRIPTION

The following paragraphs contain a brief description of the input and output signals.

NOTE

The terms **assertion** and **negation** will be used extensively. This is done to avoid confusion when dealing with a mixture of "active low" and "active high" signals. The term assert or assertion is used to indicate that a signal is active or true, independent of whether that level is represented by a high or low voltage. The term negate or negation is used to indicate that a signal is inactive or false.

V_{CC} AND GND

Power is supplied to the DUART using these two signals. V_{CC} is power (+5 volts) and GND is the ground connection.

CRYSTAL INPUT OR EXTERNAL CLOCK (X1/CLK)

This input is one of two connections to a crystal or a connection to an external clock. A crystal or a clock, within the specified limits, must be supplied at all times. If a crystal is used, a capacitor of approximately 10 to 15 picofarads should be connected from this pin to ground.

CRYSTAL INPUT (X2)

This input is an additional connection to a crystal. If an external TTL-level clock is used, this pin should be tied to ground. If a crystal is used, a capacitor of approximately 0 to 5 picofarads should be connected from this pin to ground.

RESET ($\overline{RESET}$)

The DUART can be reset by asserting the $\overline{RESET}$ signal or by programming the appropriate command register. A hardware reset, assertion of $\overline{RESET}$, clears status registers A and B (SRA and SRB), the interrupt mask register (IMR), the interrupt status register (ISR), the output port register (OPR), and the output port configuration register (OPCR). $\overline{RESET}$ initializes the interrupt vector register (IVR) to $0F_{16}$, places parallel outputs OP0 through OP3 in the high state, places the counter/timer in timer mode, and

places channels A and B in the inactive state with the channel A transmitter serial-data output (TxDA) and channel B transmitter serial-data output (TxDB) in the mark (high) state.

Software resets are not as encompassing and are achieved by appropriately programming the channel A and/or B command register. Reset commands can be programmed through the command register to reset the receiver, transmitter, error status, or break-change interrupts for each channel.

CHIP SELECT ($\overline{CS}$)

This active low input signal, when low, enables data transfers between the CPU and DUART on the data lines (D0 through D7). These data transfers are controlled by read/write (R/$\overline{W}$) and the register-select inputs (RS1 through RS4). When chip select is high the D0 through D7 data lines are placed in the high-impedance state.

READ/WRITE (R/$\overline{W}$)

When high, this input indicates a read cycle, and when low, it indicates a write cycle. A cycle is initiated by assertion of the chip-select input.

DATA TRANSFER ACKNOWLEDGE ($\overline{DTACK}$)

This three-state active low open-drain output is asserted in read, write, or interrupt acknowledge ($\overline{IACK}$) cycles to indicate the proper transfer of data between the CPU and DUART.

6

REGISTER-SELECT BUS (RS1 THROUGH RS4)

The register-select bus lines during read write operations select the DUART internal registers, ports, or commands.

DATA BUS (D0 THROUGH D7)

These bidirectional three-state data lines are used to transfer commands, data, and status between the CPU and DUART. D0 is the least-significant bit.

INTERRUPT REQUEST ($\overline{IRQ}$)

This active low, open-drain output signals the CPU that one or more of the eight maskable interrupting conditions are true.

INTERRUPT ACKNOWLEDGE ($\overline{IACK}$)

This active low input indicates an interrupt acknowledge cycle. If there is an interrupt pending ($\overline{IRQ}$ asserted) and this pin asserted, the DUART responds by placing the interrupt vector on the data bus and then asserting $\overline{DTACK}$. If there is not an interrupt pending ($\overline{IRQ}$ negated), the DUART ignores the status of this pin.

CHANNEL A TRANSMITTER SERIAL-DATA OUTPUT (TxDA)

This signal is the transmitter serial-data output for channel A. The least-significant bit is transmitted first. This output is held high (mark condition) when the transmitter is disabled, idle, or operating in the local loopback mode. (Mark is high and space is low.) Data is shifted out this pin on the falling edge of the programmed clock source.

CHANNEL A RECEIVER SERIAL-DATA INPUT (RxDA)

This signal is the receiver serial-data input for channel A. The least-significant bit is received first. Data on this pin is sampled on the rising edge of the programmed clock source.

CHANNEL B TRANSMITTER SERIAL-DATA OUTPUT (TxDB)

This signal is the transmitter serial-data output for channel B. The least-significant bit is transmitted first. The output is held high (mark condition) when the transmitter is disabled, idle, or operating in the local loopback mode. Data is shifted out this pin on the falling edge of the programmed clock source.

CHANNEL B RECEIVER SERIAL-DATA INPUT (RxDB)

This signal is the receiver serial-data input for channel B. The least-significant bit is received first. Data on this pin is sampled on the rising edge of the programmed clock source.

PARALLEL INPUTS (IP0 THROUGH IP5)

Each of the parallel inputs (IP0 through IP5) can be used as general-purpose inputs. However, each one has an alternate function(s) which is described in the following paragraphs.

IP0 This input can be used as the channel A clear-to-send active low input ($\overline{CTSA}$). A change-of-state detector is also sassociated with this input.

IP1 This input can be used as the channel B clear-to-send active low input ($\overline{CTSB}$). A change-of-state detector is also associated with this input.

IP2 This input can be used as the channel B receiver external clock input (RxCB), or the counter/timer external clock input. When this input is used as the external clock by the receiver, the received data is sampled on the rising edge of the clock. A change-of-state detector is also associated with this input.

IP3 This input can be used as the channel A transmitter external clock input (TxCA). When this input is used as the external clock by the transmitter, the transmitted data is clocked on the falling edge of the clock. A change-of-state detector is also associated with this input.

IP4 This input can be used as the channel A receiver external clock input (RxCA). When this input is used as the external clock by the receiver, the received data is sampled on the rising edge of the clock.

IP5 This input can be used as the channel B transmitter external clock (TxCB). When this input is used as the external clock by the transmitter, the transmitted data is clocked on the falling edge of the clock.

PARALLEL OUTPUTS (OP0 THROUGH OP7)

Each of the parallel outputs can be used as general-purpose outputs. However, each one has an alternate function(s) which is described in the following paragraphs.

OP0 This output can be used as the channel A active low request-to-send ($\overline{RTSA}$) output. When used for this function, it is automatically negated and reasserted by either the receiver or transmitter.

OP1 This output can be used as the channel B active low request-to-send ($\overline{RTSB}$) output. When used for this function, it is negated and reasserted automatically by either the receiver or transmitter.

OP2 This output can be used as the channel A transmitter 1X-clock or 16X-clock output, or the channel A receiver 1X-clock output.

OP3 This output can be used as the open-drain active low counter-ready output, the open-drain timer output, the channel B transmitter 1X-clock output, or the channel B receiver 1X-clock output.

OP4 This output can be used as the channel A open-drain active-low receiver-ready or buffer-full interrupt outputs ($\overline{RxRDYA/FFULLA}$) by appropriately programming bit 6 of mode register 1A.

OP5 This output can be used as the channel B open-drain active-low receiver-ready or buffer-full interrupt outputs ($\overline{RxRDYB/FFULLB}$) by appropriately programming bit 6 of mode register 1B.

OP6 This output can be used as the channel A open-drain active-low transmitter-ready interrupt output ($\overline{TxRDYA}$) by appropriately programming bit 6 of the output port configuration regiser.

OP7 This output can be used as the channel B open-drain active-low transmitter-ready interrupt output ($\overline{TxRDYB}$) by appropriately programming bit 7 of the output port configuration register.

SIGNAL SUMMARY

Table 1 provides a summary of all the MC68681 signals described above.

Table 1. Signal Summary (Sheet 1 of 2)

Signal Name	Mnemonic	Pin No.	Input/Output	Active State
Power Supply (+5 V)	V_{CC}	40	Input	High
Ground	GND	20	Input	Low
Crystal Input or External Clock	X1/CLK	32	Input	—
Crystal Input	X2	33	Input	—
Reset	$\overline{RESET}$	34	Input	Low
Chip Select	$\overline{CS}$	35	Input	Low
Read/Write	$R/\overline{W}$	8	Input	High/Low
Data Transfer Acknowledge	$\overline{DTACK}$	9	Output*	Low
Register-Select Bus Bit 4	RS4	6	Input	High
Register-Select Bus Bit 3	RS3	5	Input	High
Register-Select Bus Bit 2	RS2	3	Input	High
Register-Select Bus Bit 1	RS1	1	Input	High
Bidirectional-Data Bus Bit 7	D7	19	Input/Output	High
Bidirectional-Data Bus Bit 6	D6	22	Input/Output	High
Bidirectional-Data Bus Bit 5	D5	18	Input/Output	High
Bidirectional-Data Bus Bit 4	D4	23	Input/Output	High
Bidirectional-Data Bus Bit 3	D3	17	Input/Output	High
Bidirectional-Data Bus Bit 2	D2	24	Input/Output	High
Bidirectional-Data Bus Bit 1	D1	16	Input/Output	High
Bidirectional-Data Bus Bit 0 (Least-Significant Bit)	D0	25	Input/Output	High
Interrupt Request	$\overline{IRQ}$	21	Output*	Low
Interrupt Acknowledge	$\overline{IACK}$	37	Input	Low
Channel A Transmitter Serial Data	TxDA	30	Output	—
Channel A Receiver Serial Data	RxDA	31	Input	—

Table 1. Signal Summary (Sheet 2 of 2)

Signal Name	Mnemonic	Pin No.	Input/Output	Active State
Channel B Transmitter Serial Data	TxDB	11	Output	—
Channel B Receiver Serial Data	RxDB	10	Input	—
Parallel Input 5	IP5	38	Input	—
Parallel Input 4	IP4	39	Input	—
Parallel Input 3	IP3	2	Input	—
Parallel Input 2	IP2	36	Input	—
Parallel Input 1	IP1	4	Input	—
Parallel Input 0	IP0	7	Input	—
Parallel Output 7	OP7	15	Output**	—
Parallel Output 6	OP6	26	Output**	—
Parallel Output 5	OP5	14	Output**	—
Parallel Output 4	OP4	27	Output**	—
Parallel Output 3	OP3	13	Output**	—
Parallel Output 2	OP2	28	Output	—
Parallel Output 1	OP1	12	Output	—
Parallel Output 0	OP0	29	Output	—

*Requires a pullup resistor.
**May require a pullup resistor, depending upon its programmed function.

6

PROGRAMMING AND REGISTER DESCRIPTION

The operation of the DUART is programmed by writing control words into the appropriate registers. Operational feedback is provided by way of the status registers which can be read by the CPU. The DUART register address and address-triggered commands are described in Table 2.

Figure 2 illustrates a block diagram of the DUART from a programming standpoint and details the register configuration for each block. The locations marked "do not access" should never be read during normal operation. They are used by the factory for testing purposes.

Tables 3 and 4 are provided to illustrate the various input port pin functions and output port pin functions respectively.

Table 5 is provided to illustrate the various clock sources which may be selected for the counter and timer. More detailed information can be obtained from Table 6.

Care should be exercised if the contents of a register is changed during receiver/transmitter operation since certain changes may cause undesired results. For example, changing the number of bits-per-character while the transmitter is active may cause the transmission of an incorrect character. The contents of the mode registers (MR), the clock-select register (CSR), the output port configuration register (OPCR), and bit 7 of the auxiliary control register (ACR[7]) should only be changed after the receiver(s) and transmitter(s) have been issued software Rx and Tx reset commands. Similarly, certain changes to the auxiliary control register (ACR bits six through four) should only be made while the counter/timer (C/T) is not used (i.e., stopped if in counter mode, output and/or interrupt masked in timer mode).

Mode registers one and two of each channel are accessed via independent auxiliary pointers. The pointer is set to channel A mode register one (MR1A) and channel B mode register one (MR1B) by $\overline{\text{RESET}}$ or by issuing a "reset pointer" command via the corresponding command register. Any read of write of the mode register while the pointer is at MR1A or MR1B switches the pointer to channel A mode register two (MR2A) or channel B mode register 2 (MR2B). The pointer then remains at MR2A or MR2B. So, subsequent accesses will address MR2A or MR2B, unless the pointer is reset to MR1A or MR1B as described above.

Mode, command, clock-select, and status register are duplicated for each channel to provide total independent operation and control. Refer to Table 6 for descriptions of the register and input and output port bits.

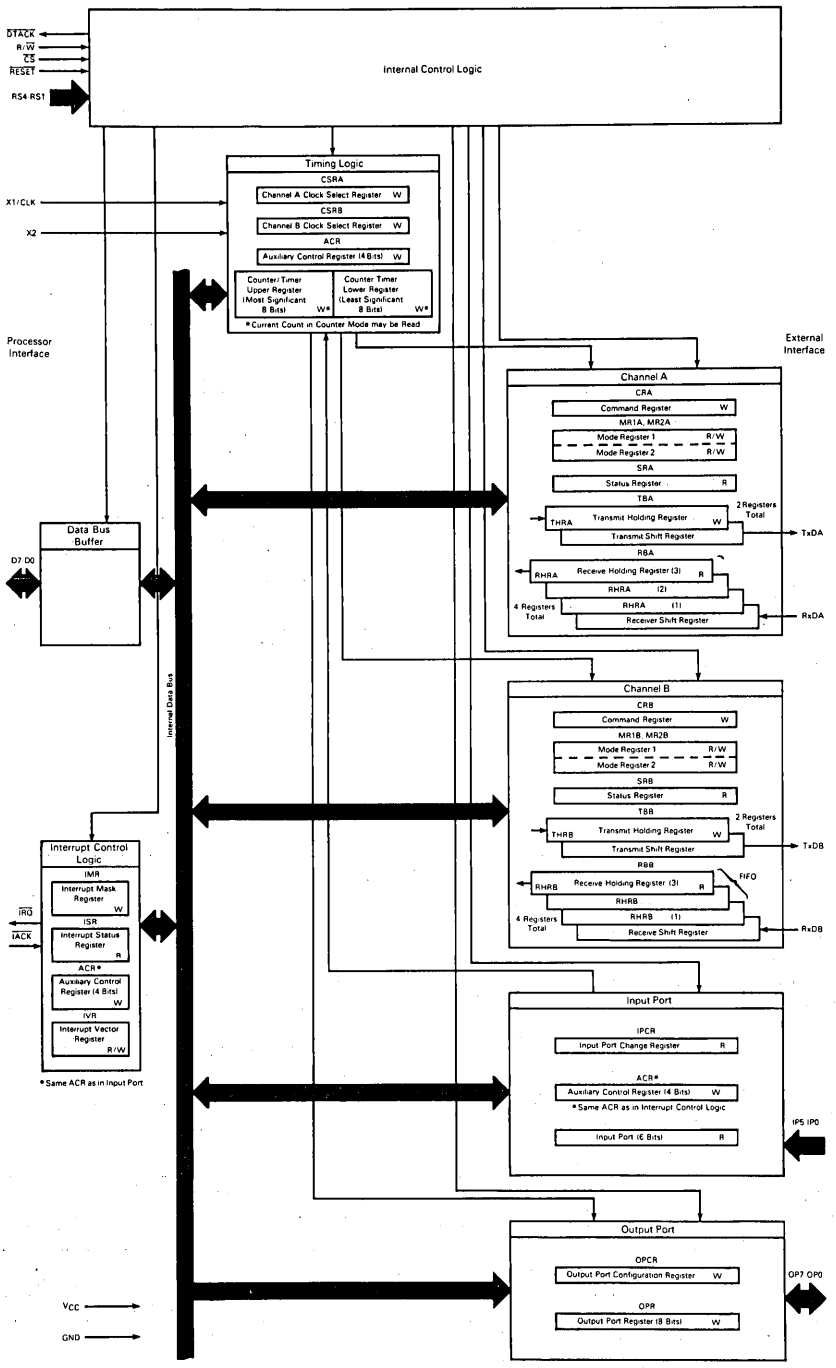


Figure 2. Programming Block Diagram

Table 2. Register Addressing and Address-Triggered Commands

RS4	RS3	RS2	RS1	Read (R/$\overline{W}$ = 1)	Write (R/$\overline{W}$ = 0)
0	0	0	0	Mode Register A (MR1A, MR2A)	Mode Register A (MR1A, MR2A)
0	0	0	1	Status Register A (SRA)	Clock-Select Register A (CSRA)
0	0	1	0	Do Not Access*	Command Register A CRA)
0	0	1	1	Receiver Buffer A (RBA)	Transmitter Buffer A (TBA)
0	1	0	0	Input Port Change Register (IPCR)	Auxiliary Control Register (ACR)
0	1	0	1	Interrupt Status Register (ISR)	Interrupt Mask Register (IMR)
0	1	1	0	Counter Mode: Current MSB of Counter (CUR)	Counter/Timer Upper Register (CTUR)
0	1	1	1	Counter Mode/ Current LSB of Counter (CLR)	Counter/Timer Lower Register (CTLR)
1	0	0	0	Mode Register B (MR1B, MR2B)	Mode Register B (MR1B, MR2B)
1	0	0	1	Status Register B (SRB)	Clock-Select Register B (CSRB)
1	0	1	0	Do Not Access*	Comand Register B (CRB)
1	0	1	1	Receiver Buffer B (RBB)	Transmitter Buffer B (TBB)
1	1	0	0	Interrupt-Vector Register (IVR)	Interrupt-Vector Register (IVR)
1	1	0	1	Input Port (Unlatched)	Output Port Configuration Register (OPCR)
1	1	1	0	Start-Counter Command**	Output Port Register (OPR) Bit Set Command**
1	1	1	1	Stop-Counter Command**	Output Port Register (OPR) Bit Reset Command**

*This address location is used for factory testing of the DUART and should not be read. Reading this location will result in undesired effects and possible incorrect transmission or reception of characters. Register contents may also be changed.
**Address triggered commands.

6

Table 3. Programming of Input Port Functions

Function	Input Port Pin					
	IP5	IP4	IP3	IP2	IP1	IP0
General Purpose	Default	Default	Default	Default	Default	Default
Change-of-State Detector			Default	Default	Default	Default
External Counter 1X Clock Input				ACR[6:4]* = 000		
External Timer 16X Clock Input				ACR[6:4]* = 100		
External Timer 1X Clock Input				ACR[6:4]* = 101		
RxCA 16X		CSRA[7:4] = 1110				
RxCA 1X		CSRA[7:4] = 1111				
TxCA 16X			CSRA[3:0] = 1110			
TxCA 1X			CSRA[3:0] = 1111			
RxCB 16X				CSRB[7:4] = 1110		
RxCB 1X				CSRB[7:4] = 1111		
TxCB 16X	CSRB[3:0] = 1110					
TxCB 1X	CSRB[3:0] = 1111					
TxCTSA						MR2A[4] = 1
TxCTSB					MR2B[4] = 1	

NOTE: Default refers to the function the input port pins perform when not used in one of the other modes. Only those functions which show the register programming are available for use.

*In these modes, because IP2 is used for the counter/timer-clock input, it is not available for use as the channel B receiver-clock input.

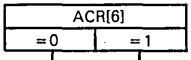
Table 4. Programming of Output Port Functions

Function	Output Port Pin							
	OP7	OP6	OP5	OP4	OP3	OP2	OP1	OP0
General Purpose	OPCR[7] = 0	OPCR[6] = 0	OPCR[5] = 0	OPCR[4] = 0	OPCR[3:2] = 00	OPCR[1:0] = 00	MR1B[7] = 0 MR2B[5] = 0	MR1A[7] = 0 MR2A[5] = 0
$\overline{CTRDY}$					OPCR[3:2] = 01, ACR[6] = 0*			
Timer Output					OPCR[3:2] = 01, ACR[6] = 1*			
TxCB 1X					OPCR[3:2] = 10			
RxCB 1X					OPCR[3:2] = 11			
TxCA 16X						OPCR[1:0] = 01		
TxCA 1X						OPCR[1:0] = 10		
RxCA 1X						OPCR[1:0] = 11		
$\overline{TxRDYA}$		OPCR[6] = 1*						
$\overline{TxRDYB}$	OPCR[7] = 1*							
$\overline{RxRDYA}$				OPCR[4] = 1, MR1A[6] = 0*				
$\overline{RxRDYB}$			OPCR[5] = 1, MR1B[6] = 0*					
$\overline{FFULLA}$				OPCR[4] = 1, MR1A[6] = *				
$\overline{FFULLB}$			OPCR[5] = 1, MR1B[6] = 1*					
$\overline{RxRTSA}$								MR1A[7] = 1
$\overline{TxRTSA}$								MR2A[5] = 1
RxRTSB							MR1B[7] = 1	
$\overline{TxRTSB}$							MR2B[5] = 1	

Note: Only those functions which show the register programming are available for use.
*Pin requires a pullup resistor if used for this function.

6

Table 5. Selection of Clock Sources for the Counter and Timer Modes



Counter Mode Clock Sources	ACR[5:4] =
External Input via Input Port Pin 2 (IP2)	00
Channel A 1X Transmitter Clock TxCA	01
Channel B 1X Transmitter Clock TxCB	10
Crystal Oscillator Divide by 16 via X1/CLK and X2 Inputs	11
External Input Divide by 16 via X1/CLK Input Pin	11

Timer Mode Clock Sources	ACR[5:4] =
External Input via Input Port Pin 2 (IP2)	00
External Input Divide by 16 via Input Port Pin 2 (IP2)	01
Crystal Oscillator via X1/CLK and X2 Inputs	10
Crystal Oscillator Divide by 16 via X1/CLK and X2 Inputs	11
External Input via X1/CLK Input Pin	10
External Input Divide by 16 via X1/CLK Input Pin	11

NOTE: Only those functions which show the register programming are available for use.

Table 6. Register Bit Formats (Sheet 1 of 5)

CHANNEL A MODE REGISTER 1 (MR1A) AND CHANNEL B MODE REGISTER 1 (MR1B)

Rx RTS Control	Rx IRQ Select	Error Mode	Parity Mode	Parity Type	Bits-per-Character
Bit 7	Bit 6	Bit 5	Bit 4 Bit 3	Bit 2	Bit 1 Bit 0
0 = Disabled 1 = Enabled	0 = RxRDY 1 = FFULL	0 = Char 1 = Block	0 0 = With Parity 0 1 = Force Parity 1 0 = No Parity 1 1 = Multidrop Mode*	With Parity 0 = Even 1 = Odd Force Parity 0 = Low 1 = High Multidrop Mode 0 = Data 1 = Address	0 0 = 5 0 1 = 6 1 0 = 7 1 1 = 8

*The parity bit is used as the address/data bit in multidrop mode.

CHANNEL A MODE REGISTER 2 (MR2A) AND CHANNEL B MODE REGISTER 2 (MR2B)

Channel Mode	Tx RTS Control	CTS Enable Transmitter	Stop Bit Length
Bit 7 Bit 6	Bit 5	Bit 4	Bit 3 Bit 2 Bit 1 Bit 0
0 0 = Normal 0 1 = Automatic Echo 1 0 = Local Loopback 1 1 = Remote Loopback	0 = Disabled 1 = Enabled	0 = Disabled 1 = Enabled	

NOTE:
If an external 1X clock is used for the transmitter, MR2 bit 3 = 0 selects one stop bit and MR2 bit 3 = 1 selects two stop bits to be transmitted.

	6-8 Bits/ Character	5-Bits/ Character
(0) 0 0 0 0 =	0.563	1.063
(1) 0 0 0 1 =	0.625	1.125
(2) 0 0 1 0 =	0.688	1.188
(3) 0 0 1 1 =	0.750	1.250
(4) 0 1 0 0 =	0.813	1.313
(5) 0 1 0 1 =	0.875	1.375
(6) 0 1 1 0 =	0.938	1.438
(7) 0 1 1 1 =	1.000	1.500
(8) 1 0 0 0 =	1.563	1.563
(9) 1 0 0 1 =	1.625	1.625
(A) 1 0 1 0 =	1.688	1.688
(B) 1 0 1 1 =	1.750	1.750
(C) 1 1 0 0 =	1.813	1.813
(D) 1 1 0 1 =	1.875	1.875
(E) 1 1 1 0 =	1.938	1.938
(F) 1 1 1 1 =	2.000	2.000

Table 6. Register Bit Formats (Sheet 2 of 5)

CLOCK-SELECT REGISTER A (CSRA)

Receiver-Clock Select						Transmitter-Clock Select					
Bit 7	Bit 6	Bit 5	Bit 4	Baud Rate Set 1 ACR Bit 7=0	Baud Rate Set 2 ACR Bit 7=1	Bit 3	Bit 2	Bit 1	Bit 0	Baud Rate Set 1 ACR Bit 7=0	Baud Rate Set 2 ACR Bit 7=1
0	0	0	0	50	75	0	0	0	0	50	75
0	0	0	1	110	110	0	0	0	1	110	110
0	0	1	0	134.5	134.5	0	0	1	0	134.5	134.5
0	0	1	1	200	150	0	0	1	1	200	150
0	1	0	0	300	300	0	1	0	0	300	300
0	1	0	1	600	600	0	1	0	1	600	600
0	1	1	0	1200	1200	0	1	1	0	1200	1200
0	1	1	1	1050	2000	0	1	1	1	1050	2000
1	0	0	0	2400	2400	1	0	0	0	2400	2400
1	0	0	1	4800	4800	1	0	0	1	4800	4800
1	0	1	0	7200	1800	1	0	1	0	7200	1800
1	0	1	1	9600	9600	1	0	1	1	9600	9600
1	1	0	0	38.4k	19.2k	1	1	0	0	38.4k	19.2k
1	1	0	1	Timer	Timer	1	1	0	1	Timer	Timer
1	1	1	0	IP4-16X	IP4-16X	1	1	1	0	IP3-16X	IP3-16X
1	1	1	1	IP4-1X	IP4-1X	1	1	1	1	IP3-1X	IP3-1X

NOTE: Receiver clock is always a 16X clock except when CSRA bits seven through four equal 1111.

NOTE: Transmitter clock is always a 16X clock except when CSRA bits three through zero equal 1111.

CLOCK-SELECT REGISTER B (CSRB)

Receiver-Clock Select						Transmitter-Clock Select					
Bit 7	Bit 6	Bit 5	Bit 4	Baud Rate Set 1 ACR Bit 7=0	Baud Rate Set 2 ACR Bit 7=1	Bit 3	Bit 2	Bit 1	Bit 0	Baud Rate Set 1 ACR Bit 7=0	Baud Rate Set 2 ACR Bit 7=1
0	0	0	0	50	75	0	0	0	0	50	75
0	0	0	1	110	110	0	0	0	1	110	110
0	0	1	0	134.5	134.5	0	0	1	0	134.5	134.5
0	0	1	1	200	150	0	0	1	1	200	150
0	1	0	0	300	300	0	1	0	0	300	300
0	1	0	1	600	600	0	1	0	1	600	600
0	1	1	0	1200	1200	0	1	1	0	1200	1200
0	1	1	1	1050	2000	0	1	1	1	1050	2000
1	0	0	0	2400	2400	1	0	0	0	2400	2400
1	0	0	1	4800	4800	1	0	0	1	4800	4800
1	0	1	0	7200	1800	1	0	1	0	7200	1800
1	0	1	1	9600	9600	1	0	1	1	9600	9600
1	1	0	0	38.4k	19.2k	1	1	0	0	38.4k	19.2k
1	1	0	1	Timer	Timer	1	1	0	1	Timer	Timer
1	1	1	0	IP2-16X	IP2-16X	1	1	1	0	IP5-16X	IP5-16X
1	1	1	1	IP2-1X	IP2-1X	1	1	1	1	IP5-1X	IP5-1X

NOTE: Receiver clock is always a 16X clock except when CSRB bits seven through four equal 1111.

NOTE: Transmitter clock is always a 16X clock except when CSRB bits three through zero equal 1111.

6

Table 6. Register Bit Formats (Sheet 3 of 5)

CHANNEL A COMMAND REGISTER (CRA) AND CHANNEL B COMMAND REGISTER (CRB)

Not Used*	Miscellaneous Commands	Transmitter Commands	Receiver Commands
Bit 7	Bit 6 Bit 5 Bit 4	Bit 3 Bit 2	Bit 1 Bit 0
X	0 0 0 No Command 0 0 1 Reset MR Pointer to MR1 0 1 0 Reset Receiver 0 1 1 Reset Transmitter 1 0 0 Reset Error Status 1 0 1 Reset Channel's Break-Change Interrupt 1 1 0 Start Break 1 1 1 Stop Break	0 0 No Action, Stays in Present Mode 0 1 Transmitter Enabled 1 0 Transmitter Disabled 1 1 Don't Use, Indeterminate	0 0 No Action, Stays in Present Mode 0 1 Receiver Enabled 1 0 Receiver Disabled 1 1 Don't Use, Indeterminate

*Bit seven is not used and may be set to either zero or one.

CHANNEL A STATUS REGISTER (SRA) AND CHANNEL B STATUS REGISTER (SRB)

Received Break	Framing Error	Parity Error	Overrun Error	TxEMT	TxRDY	FFULL	RxRDY
Bit 7*	Bit 6*	Bit 5*	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0 = No 1 = Yes	0 = No 1 = Yes	0 = No 1 = Yes	0 = No 1 = Yes	0 = No 1 = Yes	0 = No 1 = Yes	0 = No 1 = Yes	0 = No 1 = Yes

*These status bits are appended to the corresponding data character in the receive FIFO and are valid only when the RxRDY bit is set. A read of the status register provides these bits (seven through five) from the top of the FIFO together with bits four through zero. These bits are cleared by a reset error status command. In character mode, they are discarded when the corresponding data character is read from the FIFO.

OUTPUT PORT CONFIGURATION REGISTER (OPCR)

OP7	OP6	OP5	OP4	OP3	OP2
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3 Bit 2	Bit 1 Bit 0
0 = OPR Bit 7 1 = TxRDYB	0 = OPR Bit 6 1 = TxRDYA	0 = OPR Bit 5 1 = RxRDYB / FFULLB	0 = OPR Bit 4 1 = RxRDYA/ FFULLA	0 0 = OPR Bit 3 0 1 = C/T Output * 1 0 = TxCB (1X) 1 1 = RxCB (1X)	0 0 = OPR Bit 2 0 1 = TxCA (16X) 1 0 = TXCA (1X) 1 1 = RxCA (1X)

*If OP3 is to be used for the timer output, the counter/timer should be programmed for timer mode (ACR[6] = 1), the counter/timer preload registers (CTUR and CTLR) initialized, and the start counter command issued before setting OPCR[3:2] = 01.

NOTE: OP1 and OP0 can be used as transmitter and receiver RTS control lines by appropriately programming the mode registers (MR1[7] for the receiver RxRTS, and MR2[5] for the transmitter TxRTS). OP1 is used for channel B's RTS control line and OP0 for channel A's RTS control line. When OP1 and OP0 are not used for RTS control, they may be used as general-purpose outputs. (See Table 4-3.)

OUTPUT PORT REGISTER (OPR)

OPR7	OPR6	OPR5	OPR4	OPR3	OPR2	OPR1	OPR0
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Table 6. Register Bit Formats (Sheet 4 of 5)

AUXILIARY CONTROL REGISTER (ACR)

BRG SET Select*	Counter/Timer Mode and Source**			Delta*** IP3 IRQ	Delta*** IP2 IRQ	Delta*** IP1 IRQ	Delta*** IP0 IRQ
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	Mode		Clock Source				
0 = Set 1				0 = Disabled	0 = Disabled	0 = Disabled	0 = Disabled
1 = Set 2	0	0	0 Counter	External (IP2)****	1 = Enabled	1 = Enabled	1 = Enabled	1 = Enabled
	0	0	1 Counter	TxCA—1X Clock of Channel A Transmitter			
	0	1	0 Counter	TxCB—1X Clock of Channel B Transmitter			
	0	1	1 Counter	Crystal or External Clock (X1/CLK) Divided by 16			
	1	0	0 Timer	External (IP2)****			
	1	0	1 Timer	External (IP2) Divided by 16****			
	1	1	0 Timer	Crystal or External Clock (X1/CLK)			
	1	1	1 Timer	Crystal or External Clock (X1/CLK) Divided by 16			

*Should only be changed after both channels have been reset and are disabled.

**Should only be altered while the counter/timer is not in use (i.e., stopped if in counter mode, output and/or interrupt masked if in timer mode).

***Delta is equivalent to change-of-state.

****In these modes, because IP2 is used for the counter/timer clock input, it is not available for use as the channel B receiver-clock input.

INPUT PORT CHANGE REGISTER (IPCR)

Delta* Detected IP3	Delta* Detected IP2	Delta* Detected IP1	Delta* Detected IP0	Level IP3	Level IP2	Level IP1	Level IP0
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0 = No	0 = No	0 = No	0 = No	0 = Low	0 = Low	0 = Low	0 = Low
1 = Yes	1 = Yes	1 = Yes	1 = Yes	1 = High	1 = High	1 = High	1 = High

*Delta is equivalent to change-of-state.

INTERRUPT STATUS REGISTER (ISR)

Input Port Change	Delta Break B	RxRDYB/ FFULLB	TxRDYB	Counter/ Timer Ready	Delta Break A	RxRDYA/ FFULLA	TxRDYA
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0 = No	0 = No	0 = No	0 = No	0 = No	0 = No	0 = No	0 = No
1 = Yes	1 = Yes	1 = Yes	1 = Yes	1 = Yes	1 = Yes	1 = Yes	1 = Yes

INTERRUPT MASK REGISTER (IMR)

Input Port Change IRQ	Delta Break B IRQ	RxRDYB/ FFULLB IRQ	TxRDYB IRQ	Counter/ Timer Ready IRQ	Delta Break A IRQ	RxRDYA/ FFULLA IRQ	TxRDYA IRQ
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0 = Masked	0 = Masked	0 = Masked	0 = Masked	0 = Masked	0 = Masked	0 = Masked	0 = Masked
1 = Pass	1 = Pass	1 = Pass	1 = Pass	1 = Pass	1 = Pass	1 = Pass	1 = Pass

6

Table 6. Register Bit Formats (Sheet 5 of 5)

COUNTER/TIMER UPPER REGISTER (CTUR)

C/T[15]	C/T[14]	C/T[13]	C/T[12]	C/T[11]	C/T[10]	C/T[9]	C/T[8]
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

COUNTER/TIMER LOWER REGISTER (CTLR)

C/T[7]	C/T[6]	C/T[5]	C/T[4]	C/T[3]	C/T[2]	C/T[1]	C/T[0]
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

INTERRUPT VECTOR REGISTER (IVR)

IVR[7]	IVR[6]	IVR[5]	IVR[4]	IVR[3]	IVR[2]	IVR[1]	IVR[0]
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

INPUT PORT

*	**	IP5	IP4	IP3	IP2	IP1	IP0
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

* Bit seven has no external pin. Upon reading the input port, bit seven will always be read as a one.
** Bit six has no external pin. Upon reading the input port, bit six will reflect the current logic level of $\overline{IACK}$.

OUTPUT PORT

OP7	OP6	OP5	OP4	OP3	OP2	OP1	OP0
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
$\overline{OPR[7]}$	$\overline{OPR[6]}$	$\overline{OPR[5]}$	$\overline{OPR[4]}$	$\overline{OPR[3]}$	$\overline{OPR[2]}$	$\overline{OPR[1]}$	$\overline{OPR[0]}$

ELECTRICAL SPECIFICATIONS

ABSOLUTE MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Supply Voltage	V_{CC}	−0.5 to +6.0	V
Input Voltage	V_{in}	−0.5 to +6.0	V
Operating Temperature Range	T_A	0 to +70	°C
Storage Temperature	T_{stg}	−65 to +150	°C

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

THERMAL CHARACTERISTICS

Characteristic	Value		Rating
	θ_{JA}	θ_{JC}	
Thermal Resistance (Still Air)			°C/W
Ceramic, Type L	50	25*	
Plastic, Type P			
Cu Lead Frame	50	25*	
A42 Lead Frame	100	50*	

POWER CONSIDERATIONS

The average chip-junction temperature, T_J, in °C can be obtained from:

$$T_J = T_A + (P_D \cdot \theta_{JA}) \quad (1)$$

where:

T_A = Ambient Temperature, °C
θ_{JA} = Package Thermal Resistance, Junction-to-Ambient, °C/W
P_D = $P_{INT} + P_{I/O}$
P_{INT} = $I_{CC} \times V_{CC}$, 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.

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

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

Solving equations (1) and (2) for K gives:

$$K = P_D \cdot (T_A + 273°C) + \theta_{JA} \cdot P_D^2 \quad (3)$$

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.

The total thermal resistance of a package (θ_{JA} can be separated into two components, θ_{JA} and θ_{CA}, representing the barrier to heat flow from the semiconductor junction to the package (case) surface (θ_{JC}) and from the case to the outside ambient (θ_{CA}). These terms are related by the equation:

$$\theta_{JA} = \theta_{JC} + \theta_{CA} \quad (4)$$

θ_{JC} is device related and cannot be influenced by the user. However, θ_{CA} is user dependent and can be minimized by such thermal management techniques as heat sinks, ambient air cooling and thermal convection. Thus, good thermal management on the part of the user can significantly reduce θ_{CA} so that θ_{JA} aproximately equals θ_{JC}. Substitution of θ_{JC} for θ_{JA} in equation (1) will result in a lower semiconductor junction temperature.

Values for thermal resistance presented in this document, unless estimated, were derived using the procedure described in Motorola Reliability Report 7843, "Thermal Resistance Measurement Method for MC68XX Microcomponent Devices," and are provided for design purposes only. Thermal measurements are complex and dependent on procedure and setup. User derived values for thermal resistance may differ.

6

DC ELECTRICAL CHARACTERISTICS (T_A = 0°C to 70°C; V_{CC} = 5.0 V ±5%)

Characteristic	Symbol	Min	Typ	Max	Unit
Input High Voltage, Except X1/CLK	V_{IH}	2.0	—	—	V
Input High Voltage, X1/CLK	V_{IH}	4.0	—	—	V
Input Low Voltage	V_{IL}	—	—	0.8	V
Output High Voltage, Except Open-Collector Outputs (I_{OH} = −400 µA)	V_{OH}	2.4	—	—	V
Output Low Voltage (I_{OL} = 2.4 mA)	V_{OL}	—	—	0.4	V
Input Leakage Current (V_{in} = 0 to V_{CC})	I_{IL}	−10	—	10	µA
Data Bus Hi-Z Leakage Current (V_{out} = 0 to V_{CC})	I_{LL}	−10	—	10	µA
Open-Collector Output Leakage Current (V_{out} = 0 to V_{CC})	I_{OC}	−10	—	10	µA
Power Supply Current	I_{CC}	—	—	150	mA
Capacitance (V_{in} = 5 V, T_A = 25°C, f = 1 MHz)	C_{in}	—	—	15	pF
X1/CLK Low Input Current V_{in} = 0, X2 Grounded V_{in} = 0, X2 Floated	I_{X1L}	 −4.0 −3.0	 −2.0 −1.5	 0 0	mA
X1/CLK High Input Current V_{in} = V_{CC}, X2 Grounded V_{in} = V_{CC}, X2 Floated	I_{X1H}	 −1.0 0	 0.2 3.5	 1.0 10.0	mA
X2 Low Input Current V_{in} = 0, X1/CLK Floated	I_{X2L}	−100	−30	0	µA
X2 High Input Current V_{in} = V_{CC}, X1/CLK Floated	I_{X2H}	0	30	100	µA

6

AC ELECTRICAL CHARACTERISTICS (T_A = 0°C to 70°C, V_{CC} = 5.0 V ± 5%) (see Note 1)

Characteristic	Symbol	Min	Max	Unit
X1/CLK Frequency (see Note 2)	f_{CLK}	2.0	4.0	MHz
Counter/Timer Clock Frequency	f_{CTC}	0	4.0	MHz
Receiver Clock Frequency (RxC) 16X Clock 1X Clock	f_{Rx}	 0 0	 2.0 1.0	MHz
Transmitter Clock Frequency (TxC) 16X Clock 1X Clock	f_{Tx}	 0 0	 2.0 1.0	MHz

NOTES:
1. All voltage measurements are referenced to ground (GND). For testing, all input signals except X1/CLK swing between 0.4 volt and 2.4 volts with a maximum transition time of 20 nanoseconds. For X1/CLK, this swing is between 0.4 volt and 4.4 volts. All time measurements are referenced at input and output voltages of 0.8 volt and 2.0 volts as appropriate. Test conditions for outputs C_L = 150 picofarads, R_L = 750 ohm to V_{CC}.
2. To use the standard baud rates selected by the clock-select register given in Table 6, the X1/CLK frequency should be set to 3.6864 MHz or a 3.6864 MHz crystal should be connected across pins X1/CLK and X2.

AC ELECTRICAL CHARACTERISTICS—$\overline{\text{RESET}}$ TIMING (see Figure 3 and Note 1)

Characteristic	Symbol	Min	Max	Unit
$\overline{\text{RESET}}$ Pulse Width	t_{RES}	1.0	—	μs

NOTE:
1. All voltage measurements are referenced to ground (GND). For testing, all input signals swing between 0.4 volt and 2.4 volts with a maximum transition time of 20 nanoseconds. All time measurements are referenced at input and output voltages of 0.8 volt and 2.0 volts as appropriate. Test conditions for non-interrupt outputs: $C_L = 150$ picofarads, $R_L = 750$ ohms to V_{CC}. Test conditions for interrupt outputs: $C_L = 50$ picofarads, $R_L = 27$ kilohms to V_{CC}.

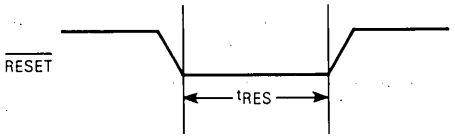


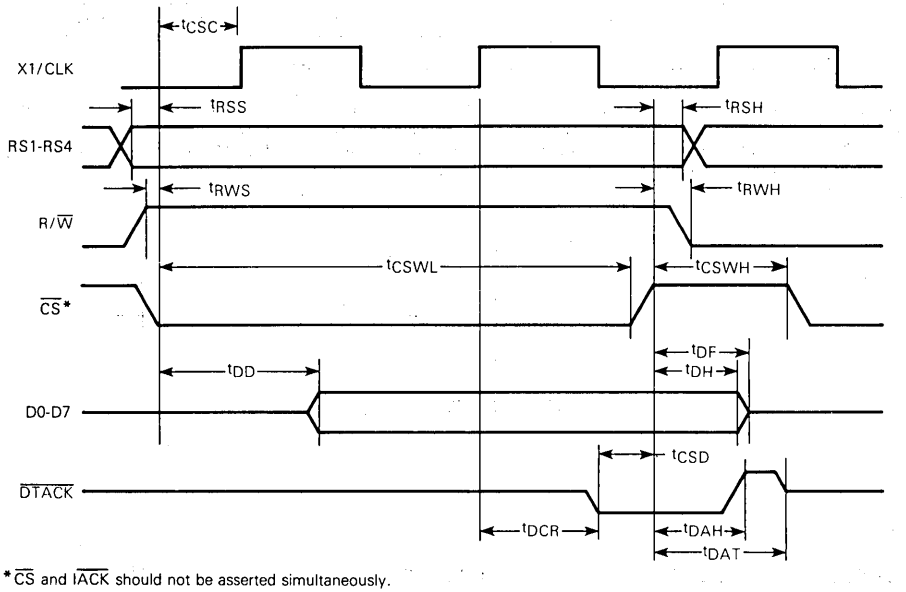
Figure 3. $\overline{\text{RESET}}$ Timing

AC ELECTRICAL CHARACTERISTICS—READ CYCLE BUS TIMING
(see Figure 4 and Note 1)

Characteristic	Symbol	Min	Max	Unit
$\overline{\text{CS}}$ Setup Time to X1/CLK High (see Note 2)	t_{CSC}	90	—	ns
RS1-RS4 Setup Time to $\overline{\text{CS}}$ Asserted	t_{RSS}	10	—	ns
R/$\overline{\text{W}}$ Setup Time to $\overline{\text{CS}}$ Asserted	t_{RWS}	0	—	
$\overline{\text{CS}}$ Pulse Width Asserted (see Note 3)	t_{CSWL}	205	—	ns
Data Valid from $\overline{\text{CS}}$ Asserted	t_{DD}	—	175	ns
$\overline{\text{DTACK}}$ Asserted from X1/CLK High	t_{DCR}	—	125	ns
$\overline{\text{CS}}$ Negated from $\overline{\text{DTACK}}$ Asserted (see Note 3)	t_{CSD}	20	—	ns
RS1-RS4 Hold Time from $\overline{\text{CS}}$ Negated	t_{RSH}	0	—	ns
R/$\overline{\text{W}}$ Hold Time from $\overline{\text{CS}}$ Negated	t_{RWH}	0	—	ns
Data Hold Time from $\overline{\text{CS}}$ Negated	t_{DH}	0	—	ns
Data Bus Floating from $\overline{\text{CS}}$ Negated	t_{DF}	—	100	ns
$\overline{\text{DTACK}}$ Negated from $\overline{\text{CS}}$ Negated	t_{DAH}	—	100	ns
$\overline{\text{DTACK}}$ Hi-Z from $\overline{\text{CS}}$ Negated	t_{DAT}	—	125	ns
$\overline{\text{CS}}$ Pulse Width Negated	t_{CSWH}	90	—	ns

NOTES:
1. All voltage measurements are referenced to GND. For testing, all input signals except X1/CLK swing between 0.4 volt and 2.4 volts with a maximum transition time of 20 nanoseconds. For X1/CLK, this swing is between 0.4 volt and 4.4 volts. All time measurements are referenced at input and output voltages of 0.8 volt and 2.0 volts as appropriate. Test conditions for non-interrupt outputs: $C_L = 150$ picofarads, $R_L = 750$ ohms to V_{CC}. Test conditions for interrupt outputs: $C_L = 50$ picofarads, $R_L = 27$ kilohms to V_{CC}.
2. This specification is made only to insure $\overline{\text{DTACK}}$ is asserted with respect to the rising edge of X1/CLK as shown in Figure 4, not to guarantee operation of the part. If the setup time is violated, $\overline{\text{DTACK}}$ may be asserted as shown, or may be asserted one clock cycle later.
3. The t_{CSD} specification is made only to insure that $\overline{\text{DTACK}}$ will be asserted. If $\overline{\text{CS}}$ is negated before $\overline{\text{DTACK}}$ is asserted, $\overline{\text{DTACK}}$ may not be asserted.

6



*CS and IACK should not be asserted simultaneously.

Figure 4. Read Cycle Bus Timing

6

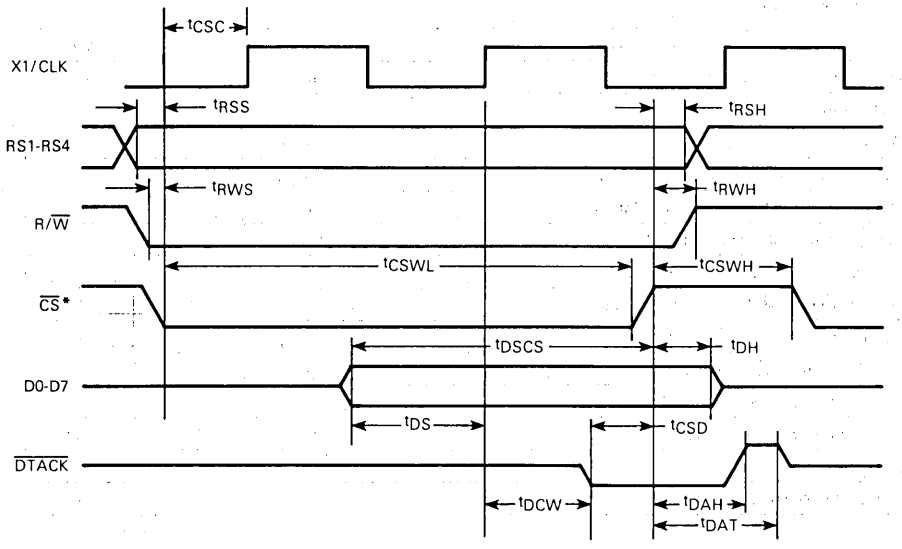
AC ELECTRICAL CHARACTERISTICS—WRITE CYCLE BUS TIMING

(see Figure 5 and Note 1)

Characteristic	Symbol	Min	Max	Unit
$\overline{CS}$ Setup Time to X1/CLK High (see Note 2)	t_{CSC}	90	—	ns
RS1-RS4 Setup Time to $\overline{CS}$ Asserted	t_{RSS}	10	—	ns
R/$\overline{W}$ Setup Time to $\overline{CS}$ Asserted	t_{RWS}	0	—	
$\overline{CS}$ Pulse Width Asserted (see Notes 3 and 4)	t_{CSWL}	205	—	ns
Data Setup Time to X1/CLK High (see Note 4)	t_{DS}	100	—	ns
Data Setup Time to $\overline{CS}$ Negated (see Note 4)	t_{DSCS}	100	—	ns
$\overline{DTACK}$ Asserted from X1/CLK High	t_{DCW}	—	125	ns
$\overline{CS}$ Negated from $\overline{DTACK}$ Asserted (see Note 3)	t_{CSD}	20	—	ns
RS1-RS4 Hold Time from $\overline{CS}$ Negated	t_{RSH}	0	—	ns
R/$\overline{W}$ Hold Time from $\overline{CS}$ Negated	t_{RWH}	0	—	ns
Data Hold Time from $\overline{CS}$ Negated	t_{DH}	0	—	ns
$\overline{DTACK}$ Negated from $\overline{CS}$ Negated	t_{DAH}	—	100	ns
$\overline{DTACK}$ Hi-Z from $\overline{CS}$ Negated	t_{DAT}	—	125	ns
$\overline{CS}$ Pulse Width Negated (see Note 5)	t_{CSWH}	90	—	ns

NOTES:

1. All voltage measurements are referenced to ground (GND). For testing, all input signals except X1/CLK swing between 0.4 volt and 2.4 volts with a maximum transition time of 20 nanoseconds. For X1/CLK, this swing is between 0.4 volt and 4.4 volts. All time measurements are referenced at input and output voltages of 0.8 volt and 2.0 volts as appropriate. Test conditions for non-interrupt outputs: $C_L = 150$ picofarads, $R_L = 750$ ohms to V_{CC}. Test conditions for interrupt outputs: $C_L = 50$ picofarads, $R_L = 27$ kilohms to V_{CC}.
2. This specificationis made only to insure $\overline{DTACK}$ is asserted with respect to the rising edge of X1/CLK as shown in Figure 4, not to guarantee operation of the part. If the setup time is violated, $\overline{DTACK}$ may be asserted as shown, or maybe asserted one clock cycle later.
3. The t_{CSD} specification is made only to insure that $\overline{DTACK}$ will be asserted. If $\overline{CS}$ is negated before $\overline{DTACK}$ is asserted, $\overline{DTACK}$ may not be asserted.
4. During write cycles, data is latched on either the asserting edge of $\overline{DTACK}$ or the negating edge of $\overline{CS}$, whichever occurs first. If $\overline{CS}$ is negated within one clock cycle after $\overline{CS}$ has been recognized (i.e., first rising edge of X1/CLK where $\overline{CS}$ is asserted), then $\overline{DTACK}$ may not be generated. In this case, data will be latched on the negating edge of $\overline{CS}$. Thus, t_{DS} can be ignored but t_{DSCS} must be observed.
5. Consecutive write operations to the same command register (CRA or CRB) require at least three transitions of X1/CLK between write cycles. Typically, a processor is incapable of accessing the same command register a second time prior to three transitions on the X1/CLK pin.



*CS and IACK should not be asserted simultaneously.

Figure 5. Write Cycle Bus Timing

6

AC ELECTRICAL CHARACTERISTICS—INTERRUPT CYCLE BUS TIMING*

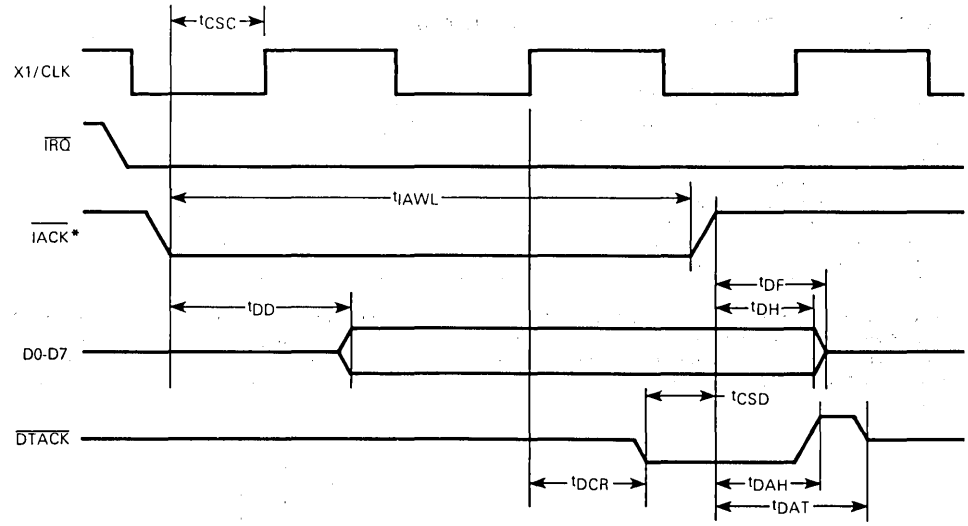
(see Figure 6 and Note 1)

Characteristic	Symbol	Min	Max	Unit
$\overline{IACK}$ Setup Time to X1/CLK High (see Note 2)	t_{CSC}	90	—	ns
$\overline{IACK}$ Pulse Width Asserted (see Note 3)	t_{IAWL}	205	—	ns
Data Valid from $\overline{IACK}$ Asserted	t_{DD}	—	175	ns
$\overline{DTACK}$ Asserted from X1/CLK High	t_{DCR}	—	125	ns
$\overline{IACK}$ Negated from $\overline{DTACK}$ Asserted (see Note 3)	t_{CSD}	0	—	ns
Data Hold Time from $\overline{IACK}$ Negated	t_{DH}	0	—	ns
Data Bus Floating from $\overline{IACK}$ Negated	t_{DF}	—	100	ns
$\overline{DTACK}$ Negated from $\overline{IACK}$ Negated	t_{DAH}	—	100	ns
$\overline{DTACK}$ Hi-Z from $\overline{IACK}$ Negated	t_{DAT}	—	125	ns

*During $\overline{IACK}$ cycles, the status of the R/$\overline{W}$ line is ignored.

NOTES:

1. All voltage measurements are referenced to ground (GND). For testing, all input signals except X1/CLK swing between 0.4 volt and 2.4 volts with a maximum transition time of 20 nanoseconds. For X1/CLK, this swing is between 0.4 volt and 4.4 volts. All time measurements are referenced at input and output voltages of 0.8 volt and 2.0 volts as appropriate. Test conditions for non-interrupt outputs: $C_L = 150$ picofarads, $R_L = 750$ ohms to V_{CC}. Test conditions for interrupt outputs: $C_L = 50$ picofarads, $R_L = 27$ kilohms to V_{CC}.
2. This specification is made only to insure $\overline{DTACK}$ is asserted with respect to the rising edge of X1/CLK as shown in Figure 4, not to guarantee operation of the part. If the setup time is violated, $\overline{DTACK}$ may be asserted as shown, or may be asserted one clock cycle later.
3. The t_{CSD} specification is made only to insure that $\overline{DTACK}$ will be asserted. If $\overline{CS}$ is negated before $\overline{DTACK}$ is asserted, $\overline{DTACK}$ may not be asserted.



* $\overline{CS}$ and $\overline{IACK}$ should not be asserted simultaneously.

Figure 6. Interrupt Cycle Bus Timing

AC ELECTRICAL CHARACTERISTICS—PORT TIMING

(see Figure 7 and Note 1)

Characteristic	Symbol	Min	Max	Unit
Port Input Setup Time to $\overline{CS}$ Asserted	t_{PS}	0	—	ns
Port Input Hold Time from $\overline{CS}$ Negated	t_{PH}	0	—	ns
Port Output Valid from $\overline{CS}$ Negated	t_{PD}	—	400	ns

NOTE:

1. All voltage measurements are referenced to ground (GND). For testing, all signals except X1/CLK swing between 0.4 volt and 2.4 volts with a maximum transition time of 20 nanoseconds. For X1/CLK, this swing is between 0.4 and 4.4 volts. All time measurements are referenced at input and output voltages of 0.8 volt and 2.0 volts as appropriate. Test conditions for non-interrupt outputs: $C_L = 150$ picofarads, $R_L = 750$ ohms to V_{CC}. Test conditions for interrupt outputs: $C_L = 50$ picofarads, $R_L = 27$ kilohms to V_{CC}.

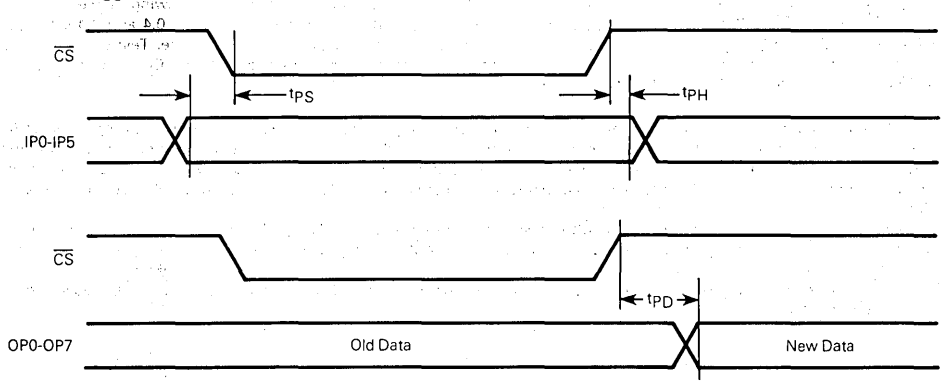


Figure 7. Port Timing

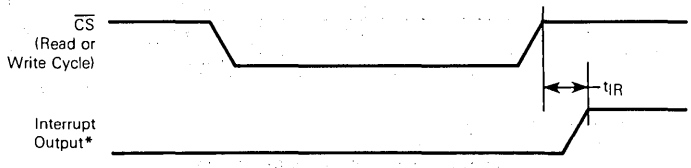
AC ELECTRICAL CHARACTERISTICS—INTERRUPT RESET TIMING

(see Figure 8 and Note 1)

Characteristic	Symbol	Min	Max	Unit
$\overline{IRQ}$ Negated or OP3-OP7 High from $\overline{CS}$ Negated when used as Interrupts from:	t_{IR}			ns
Read RB (RxRDY/FFULL Interrupt)		—	300	
Write TB (TxRDY Interrupt)		—	300	
Reset Command (Delta Break Interrupt)		—	300	
Stop C/T Command (Counter Interrupt)		—	300	
Read IPCR (Input Port Change Interrupt)		—	300	
Write IMR (Clear-of-Interrupt Mask Bit)		—	300	

NOTE:

1. All voltage measurements are referenced to ground (GND). For testing, all input signals except X1/CLK swing between 0.4 volt and 2.4 volts with a maximum transition time of 20 nanoseconds. For X1/CLK, this swing is between 0.4 and 4.4 volts. All time measurements are referenced at input and output voltages of 0.8 volt and 2.0 volts as appropriate. Test conditions for non-interrupt outputs: $C_L = 150$ picofarads, $R_L = 750$ ohms to V_{CC}. Test conditions for interrupt outputs: $C_L = 50$ picofarads, $R_L = 27$ kilohms to V_{CC}.



*$\overline{IRQ}$ or OP3-OP7 when used as interrupt outputs.

Figure 8. Interrupt Reset Timing

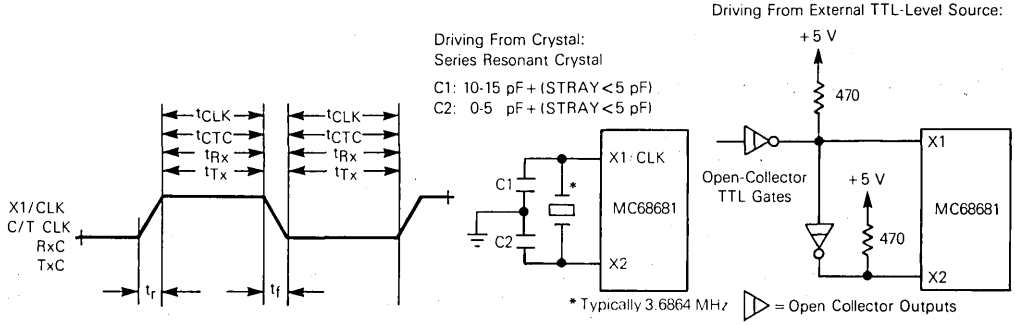
AC ELECTRICAL CHARACTERISTICS—CLOCK TIMING

(see Figures 9 and Note 1)

Characteristic	Symbol	Min	Max	Unit
X1/CLK High or Low Time	t_{CLK}	100	—	ns
Counter/Timer Clock High or Low Time	t_{CTC}	100	—	ns
Receive clock (RxC) High or Low Time	t_{Rx}	220	—	ns
Transmit Clock (TxC) High or Low Time	t_{Tx}	220	—	ns
Clock Rise Time	t_r	—	20	ns
Clock Fall Time	t_f	—	20	ns

NOTE:

1. All voltage measurements are referenced to ground (GND). For testing, all signals except X1/CLK swing between 0.4 volt and 2.4 volts with a maximum transition time of 20 nanoseconds. For X1/CLK, this swing is between 0.4 and 4.4 volts. All time measurements are referenced at input and output voltages of 0.8 volt and 2.0 volts as appropriate. Test conditions for non-interrupt outputs: $C_L = 150$ picofarads, $R_L = 750$ ohms to V_{CC}. Test conditions for interrupt outputs: $C_L = 50$ picofarads, $R_L = 27$ kilohms to V_{CC}.



NOTE: Board layout should be such that the crystal and capacitor(s) are as close as possible to the pins of the DUART to minimize stray capacitance. Also, crystal series resistance should be less than 180 ohms.

Figure 9. Clock Timing

AC ELECTRICAL CHARACTERISTIC—TRANSMITTER TIMING

(see Figure 10 and Note 1)

Characteristic	Symbol	Min	Max	Unit
TxD Output Valid from TxC Low	t_{TxD}	—	350	ns
TxC Low to TxD Output Valid	t_{TCS}	—	150	ns

NOTE:

1. All voltage measurements are referenced to ground (GND). For testing, all signals except X1/CLK swing between 0.4 volt and 2.4 volts with a maximum transition time of 20 nanoseconds. For X1/CLK, this swing is between 0.4 and 4.4 volts. All time measurements are referenced at input and output voltages of 0.8 volt and 2.0 volts as appropriate. Test conditions for non-interrupt outputs: $C_L = 150$ picofarads, $R_L = 750$ ohms to V_{CC}. Test conditions for interrupt outputs: $C_L = 50$ picofarads, $R_L = 27$ kilohms to V_{CC}.

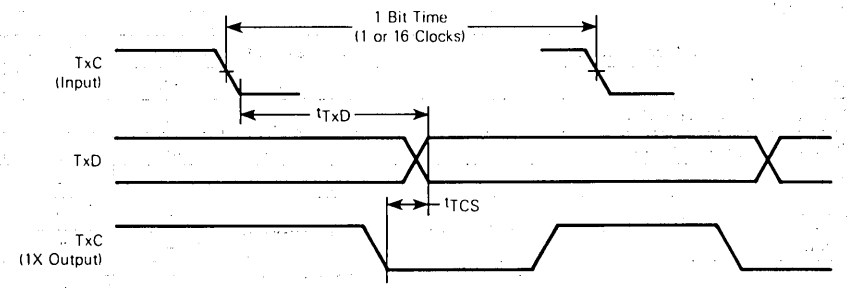


Figure 10. Transmit Timing

AC ELECTRICAL CHARACTERISTICS—RECEIVER TIMING

(see Figure 11 and Note 1)

Characteristics	Symbol	Min	Max	Unit
RxD Data Setup Time to RxC High	t_{RxS}	240	—	ns
RxD Data Hold Time from RxC High	t_{RxH}	200	—	ns

6

NOTE:

1. All voltage measurements are referenced to ground (GND). For testing, all signals except X1/CLK swing between 0.4 volt and 2.4 volts with a maximum transition time of 20 nanoseconds. For X1/CLK, this swing is between 0.4 and 4.4 volts. All time measurements are referenced at input and output voltages of 0.8 volt and 2.0 volts as appropriate. Test conditions for non-interrupt outputs: $C_L = 150$ picofarads, $R_L = 750$ ohms to V_{CC}. Test conditions for interrupt outputs: $C_L = 50$ picofarads, $R_L = 27$ kilohms to V_{CC}.

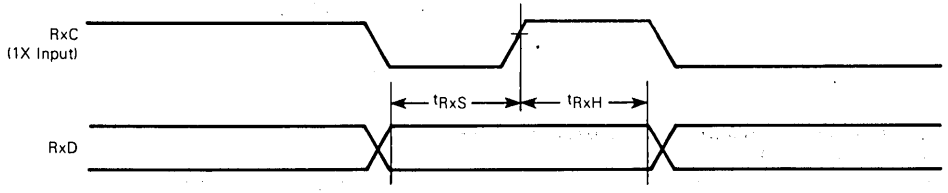
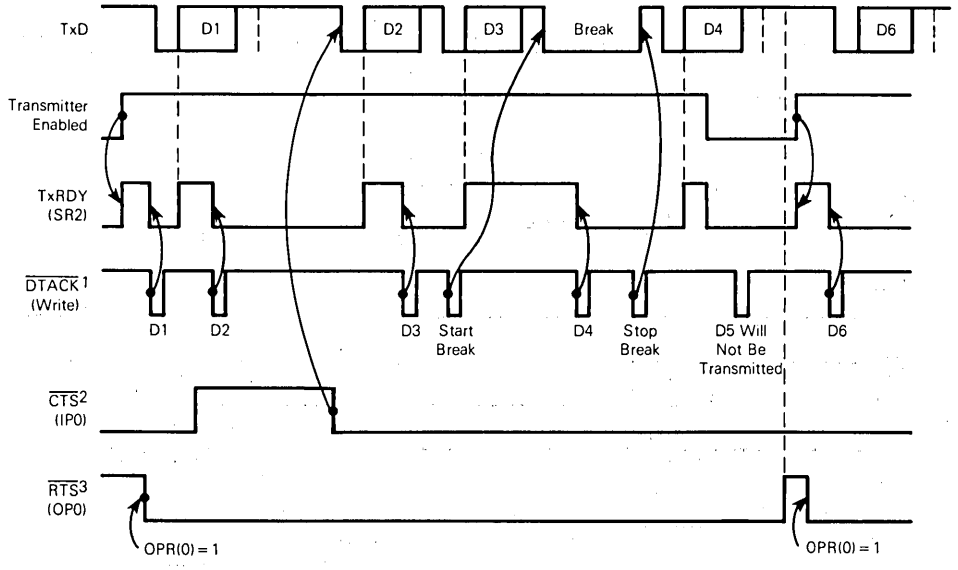
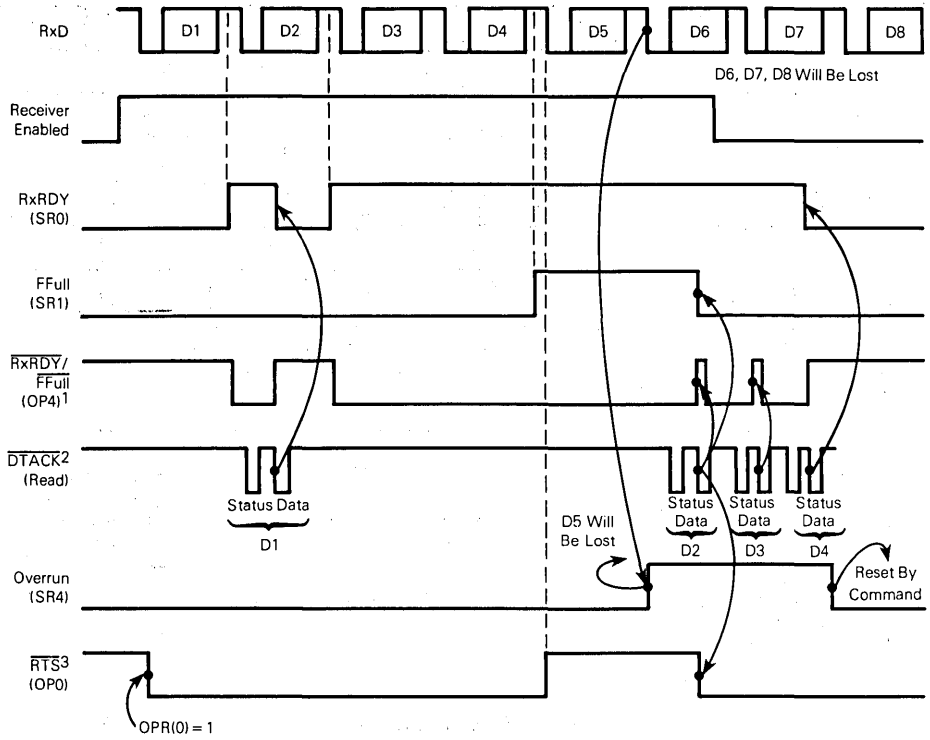


Figure 11. Receive Timing



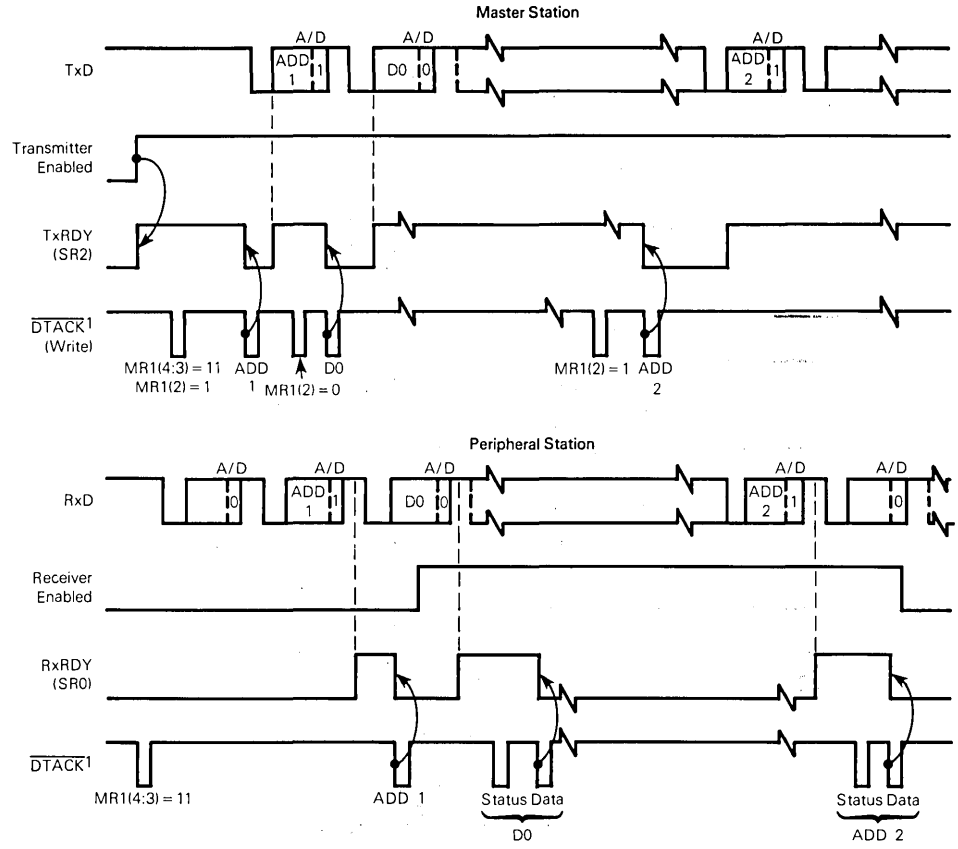
NOTES:
1. If $\overline{CS}$ is negated within one clock cycle after $\overline{CS}$ is recognized, $\overline{DTACK}$ will not be asserted. In this case the negation of $\overline{CS}$ will cause the transitions.
2. Timing shown for MR2(4) = 1.
3. Timing shown for MR2(5) = 1.

Figure 12. Transmitter Timing



NOTES:
1. Shown for OPCR(4) = 1 and MR1(6) = 0.
2. If $\overline{CS}$ is negated within one clock cycle after $\overline{CS}$ is recognized, $\overline{DTACK}$ will not be asserted. In this case the negation of $\overline{CS}$ will cause the transitions.
3. Timing shown for MR1(7) = 1.

Figure 13. Receiver Timing



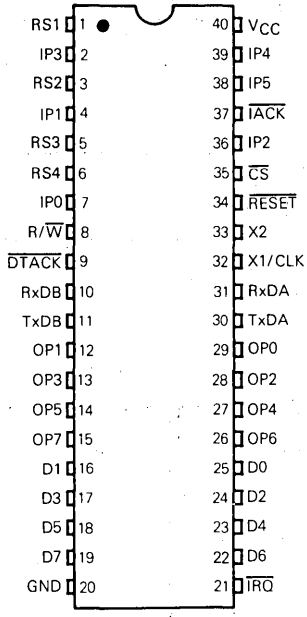
NOTE 1: If $\overline{CS}$ is negated within one clock cycle after $\overline{CS}$ is recognized, $\overline{DTACK}$ will not be asserted. In this case the negation of $\overline{CS}$ will cause the transitions.

Figure 14. Wake-Up Mode Timing

6

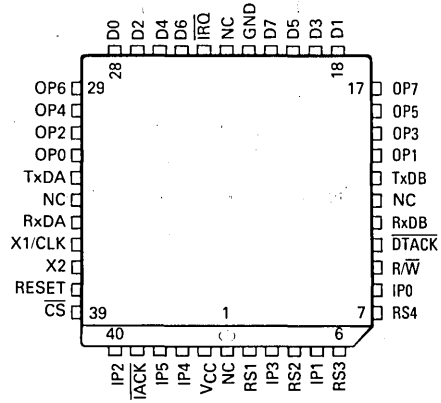
PIN ASSIGNMENTS

DUAL-IN-LINE



6

PLASTIC LEADED CHIP CARRIER



NETWORK DEVICES 7

MC68184

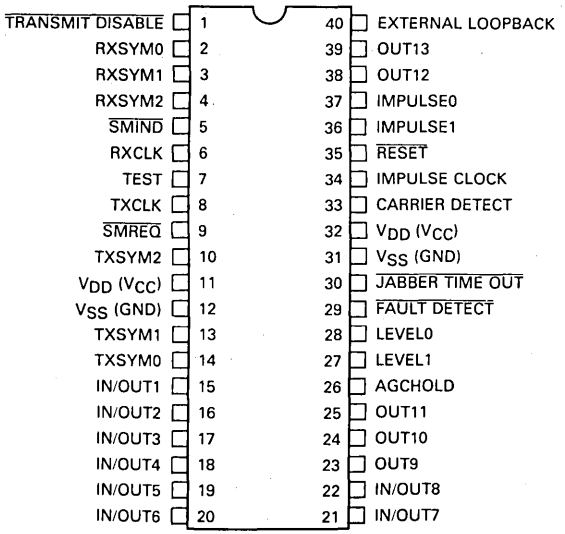
Advance Information

Broadband Interface Controller (BIC)

The CMOS LSI MC68184 Broadband Interface Controller (BIC) is used in conjunction with a protocol handler such as the MC68824 Token Bus Controller to implement a broadband IEEE 802.4 Token Bus node. The BIC handles both data manipulation and management control for R.F. transmitter and receiver circuitry.

- Implements Digital Portion of IEEE 802.4 Broadband Physical Layer
- Provides Station Management for Physical Layer
- Supports Serial Data Rates Up to 10 Mbps
- Twenty Lines for Receiver/Transmitter Control Including Thirteen User-Defined
- Includes Ability to Scramble and Unscramble Data
- Kicker Insertion and Deletion
- Post-Error Correction Capability
- Pseudo-Silence Deletion
- Two Loopback Modes
- Interfaces Via a Standard Serial Interface to a Protocol Handler Either Directly or Through An Arbitrary Length of Cable

PIN ASSIGNMENTS



Motorola Order Number: MC68184P

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

SECTION 1
INTRODUCTION

1.1 BROADBAND LOCAL AREA NETWORK OVERVIEW

A broadband network consists of a coaxial cable, a headend remodulator, and several nodes or stations as shown in Figure 1-1. The purpose of the network is to permit information to be exchanged between any two nodes in an orderly and efficient manner. In a broadband system, data is transmitted in frequency channels similar to CATV. All nodes receive on one channel called the forward channel and transmit on another channel called the reverse channel. The headend does the translation between the two channels. For a 10 Mbps data rate, the two channels are each 12 MHz wide. For example, if Node A wants to send data to Node B, Node A transmits the data on the reverse channel to the headend. The headend receives the data from the reverse channel and retransmits it on the forward channel. All the nodes receive the data on the forward channel but Node B recognizes that the data is addressed to itself and therefore copies the data.

In addition to translating the data, the headend also provides the clock source for the network on the forward channel. The clock is used to recover data from the forward channel and also to transmit data on the reverse channel. Therefore, data is always transmitted at the rate determined by the headend. Because the data received by the headend is guaranteed to be at the same data rate as the data transmitted, the headend does not have to do a store and forward of the message. So that all nodes can recover the clock at all times, the headend must provide the clock signal at all times. This has two influences on the network.

1) Because clock edges are derived from data edges, a node transmitting a very long string of ones or zeros will therefore not be providing a clock signal. The transmitting node must therefore scramble the data to eliminate any long string of ones and zeros. The receiving nodes automatically unscramble the data. See ANSI/IEEE Std 802.4-1985 and the scrambler and kicker inserter block descriptions (Subsections 3.2 and 3.3) for more details.
2) When no stations are transmitting, if the headend faithfully replicated this silence, all the nodes would lose clock synchronization. To prevent this, the headend transmits a special sequence called pseudo-silence. The receiving nodes use this sequence to recover the clock but report it as silence.

1.2 BROADBAND NODE

A typical Token Bus Local Area Network (LAN) node consists of a host processor, memory, Token Bus Controller (TBC), BIC and R.F. circuitry as shown in Figure 1-2. The Token Bus Controller fetches messages out of memory, serializes them, and passes the data to the physical layer (BIC and R.F. circuitry) to be transmitted over the coaxial cable. When data is received from the coaxial cable, the physical layer decodes the data, and passes it up to the Token Bus Controller which deserializes it and places the data in memory. The BIC performs the digital functions of an IEEE 802.4 physical layer when implementing a broadband token bus node. The BIC provides data and control for the R.F. transmitter/receiver circuitry as well as a standard serial interface to connect the BIC to the Token Bus Controller (MC68824).

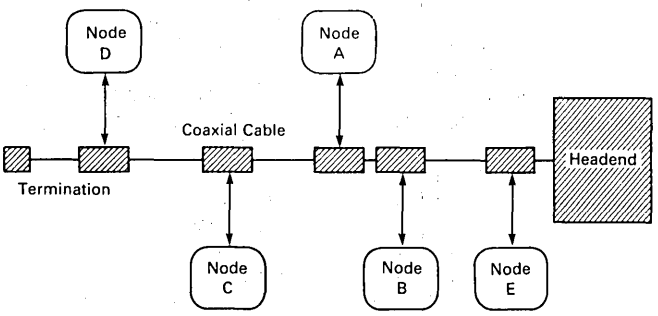


FIGURE 1-1 — BROADBAND NETWORK

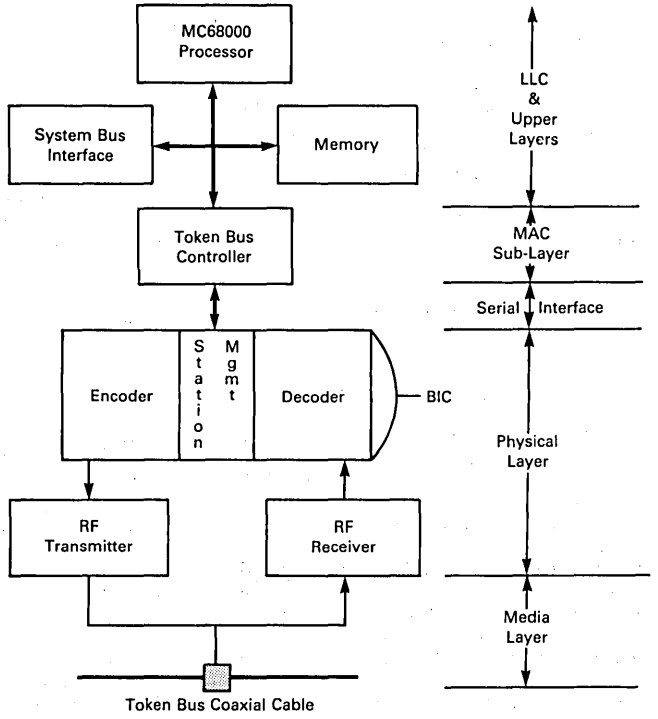


FIGURE 1-2 — TOKEN BUS BROADBAND NODE

1.3 BROADBAND SIGNALS

When the processor wants to put a message onto the cable, it puts the message on the transmit queue which is located in the memory shared with the TBC. The TBC pulls the message out of the memory, serializes it, and outputs it to the BIC. Each bit time, the TBC can request one of five things to be transmitted:

1) Silence. Requesting silence instructs the modem not to output any energy onto the cable. In the network, only one station is normally transmitting on the reverse channel so the rest of the stations are "transmitting" silence. There is a minimum requirement of at least two silence bits between messages to allow the transmitter time to ramp down and up gracefully.
2) Pad-idle. Before each transmission some integer number of octets (eight bits) of pad-idle are transmitted. Pad-idle is used to provide a training signal for the headend to lock to before the actual data is transmitted. Between consecutive messages, pad-idle is used to separate the messages to allow the receiving TBC time to process the current message before the next message comes in. Before transmitting, the BIC translates pad-idle into a repeating one-zero sequence which is the proper training signal. At the receiver, ones and zeros are reported instead of pad-idles.
3) Non-data. Non-datas are used to determine the start and end of messages. The start of a message is called the start delimiter and consists of the following eight bits: non-data, non-data, zero, non-data, non-data, zero, zero, zero. An end delimiter signals the end of the message and consists of the following eight bits: non-data, non-data, one, non-data, non-data, one, one or zero, one or zero. The last two bits of the end delimiter are used as status bits by the TBC. The start and end delimiters are the only time that the TBC will request a non-data. Therefore, non-datas from the TBC will always come in pairs and there will never be more than two non-datas in a row. Non-datas are also used by the BIC to create "kickers" and "pseudo-silence." See the block descriptions for the kicker inserter and the psuedo-silence inserter for more details.
4) One. Ones are transmitted as part of the "information" or data part of the message (between the start and end delimiter), as part of the end delimiter, and as part of the pad-idle training signal. The ones in the pad-idle can be used to lock up the AGC of the receiver.

7

5) Zero. Zeros are transmitted as part of the "information" part of the message (between the start and end delimiter), as part of the start delimiter, and as part of the pad-idle training signal. Zero's can also be present in the last two bits of the end delimiter.

A message from the TBC to the BIC consists of: some number of octets of pad-idle, a start delimiter, some number of octets of ones and zeros — the "information," an end delimiter, followed by either pad-idle for the next message or at least two bits of silence. With P representing pad-idle, S silence, N non-data, a typical message would look like:

SILENCE	PAD-IDLE				START DELIMITER	DATA	
SSSSSSSS	SSSSSSSS	PPPPPPPP	PPPPPPPP	PPPPPPPP	PPPPPPPP	NN0NN000	01010100	00001110

DATA CONTINUED						END DELIMITER	SILENCE
11111000	11001011	11111001	00000100	10101000	11101001	01000011	NN1NN100	SSSSSSSS

Before transmitting, the R.F. transmitter performs a duo-binary AM-PSK translation on the data. Duo-binary is a bandwidth compression technique which converts the data into an analog waveform. At the output of the duo-binary encoder, ones have a full amplitude, either positive or negative, zeros have almost no amplitude, and non-datas are at half amplitude, either positive or negative. This analog waveform is then impressed onto a carrier using AM-PSK modulation. With AM-PSK, the amplitude of the carrier represents the amplitude of the duo-binary output (Amplitude Modulation). The phase of the carrier shows whether the amplitude is positive or negative (Phase Shift Keying). AM-PSK puts the spectrum of the output into a frequency "channel" similar to a TV channel. Twelve megahertz of bandwidth is required to send data at a rate of 10 megabits per second. Two twelve megahertz channels are used, one for the nodes to transmit to the headend (the reverse channel), and the other for the headend to transmit to all the nodes (the forward channel). This method is similar to cable TV (CATV) and in fact IEEE 802.4 was patterned after CATV systems. The advantage of having the data in a specific channel is that the rest of the cable bandwidth is free for other uses such as video information, point-to-point modems, and other networks.

At the receiver, the data can be recovered by comparing the amplitude in the middle of the bit time against the maximum amplitude. A maximum amplitude indicates a one, a half-amplitude is a non-data, and very little amplitude is a zero. Silence at the node receiver is an error condition since the headend is supposed to be transmitting at all times. The node also recovers the clock by sensing the transitions between no amplitude and maximum amplitude or visa-versa.

1.4 BIC OVERVIEW

The functional block diagram of the BIC is shown in Figure 1-3. The Encoder accepts data from the serial interface and outputs to the R.F. transmitter using outputs IMPULSE0 and IMPULSE1 encoded as shown in Table 1 (Section 2.1.1). In the encoder, the data from the TBC is translated into a form which is acceptable for the broadband network. This includes inserting any necessary clock information.

The Decoder section accepts the encoded receiver levels from the R.F. receiver circuitry and outputs data to the TBC using the serial interface. LEVEL0 and LEVEL1 signals are inputs from the R.F. receiver which are encoded as shown in Table 2. The receiver must perform the opposite function as the encoder in order to make the broadband network transparent to the TBC's.

The Station Management Interface controls the encoder, decoder and the R.F. transmitter/receiver. This interface accepts commands from the TBC and inputs from JABBER TIME OUT, FAULT DETECT and CARRIER DETECT to determine the proper mode of operation. RESET is a bidirectional signal which is normally an input from external circuitry and becomes an output when the BIC receives a reset command from the TBC. The R.F. transmitter/receiver is controlled using RESET, TRANSMIT DISABLE, EXTERNAL LOOPBACK, AGC HOLD, OUT13 to OUT9, and IN/OUT8 to IN/OUT1.

1.5 BROADBAND MODEM

The BIC provides the digital portion of the broadband physical layer. To complete the broadband modem, the R.F. and clock functions shown to the right of the BIC in Figure 1-4 are required.

The duo-binary encoder uses the BIC outputs IMPULSE0 and IMPULSE1 to produce the correct analog waveform as described in ANSI/IEEE 802.4 Std-1985, Section 14. This analog waveform is applied to the carrier by the AM-PSK R.F. TRANSLATER. This modulated carrier is placed on the coax cable by the power stage. The power stage can be disabled by the BIC output TRANSMIT DISABLE. The R.F. receiver detects the amplitude modulated carrier and converts it to an analog signal. The receiver reports this signal to the clock recovery circuit which uses the transitions to recover the clock. The data then can be recovered and put into the BIC on LEVEL0 and LEVEL1. The automatic gain control of the receiver maintains the current gain when the BIC output AGC HOLD is asserted. In addition, a power-up detect circuit is required to reset the BIC at power-up and hold the BIC in reset until ten clocks after power becomes stable. The jabber timer is required by ANSI/IEEE 802.4 Std-1985 to detect if the modem transmits for more than 1/2 second. If so, the jabber timer asserts JABBER TIME OUT which will stop the BIC from transmitting. The modem failure detect is optional and asserts FAULT DETECT in case of a modem error. The jabber timer and the modem failure detect should be put to the inactive state by reset.

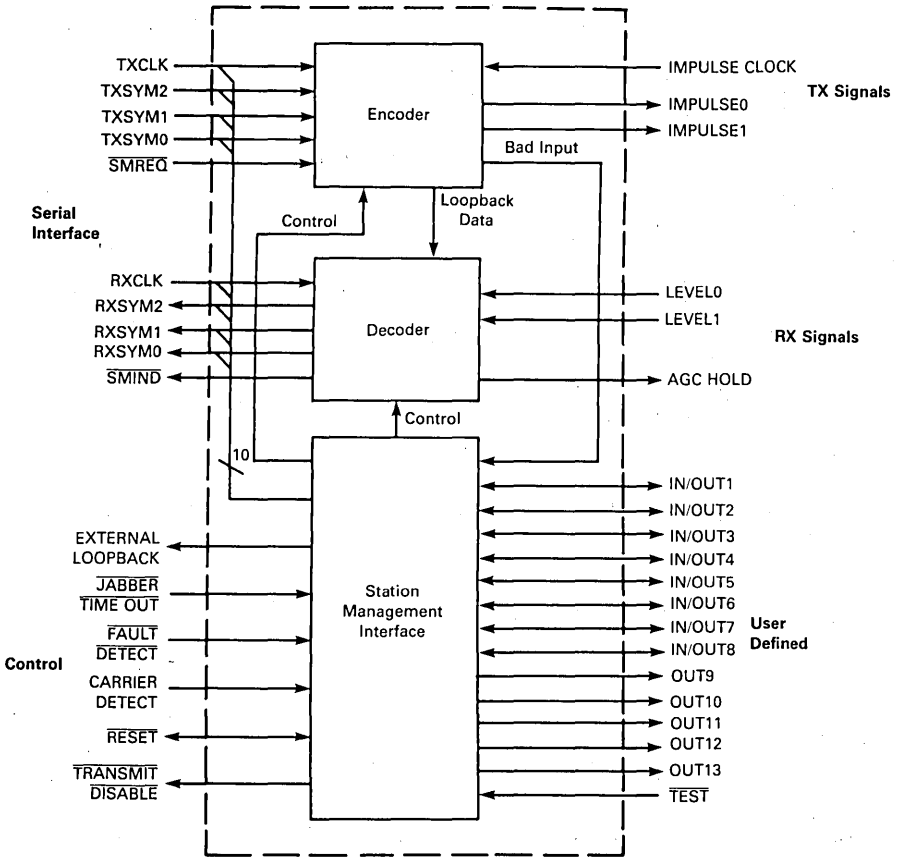


FIGURE 1-3 — FUNCTIONAL BLOCK DIAGRAM FOR BROADBAND INTERFACE CONTROLLER

7

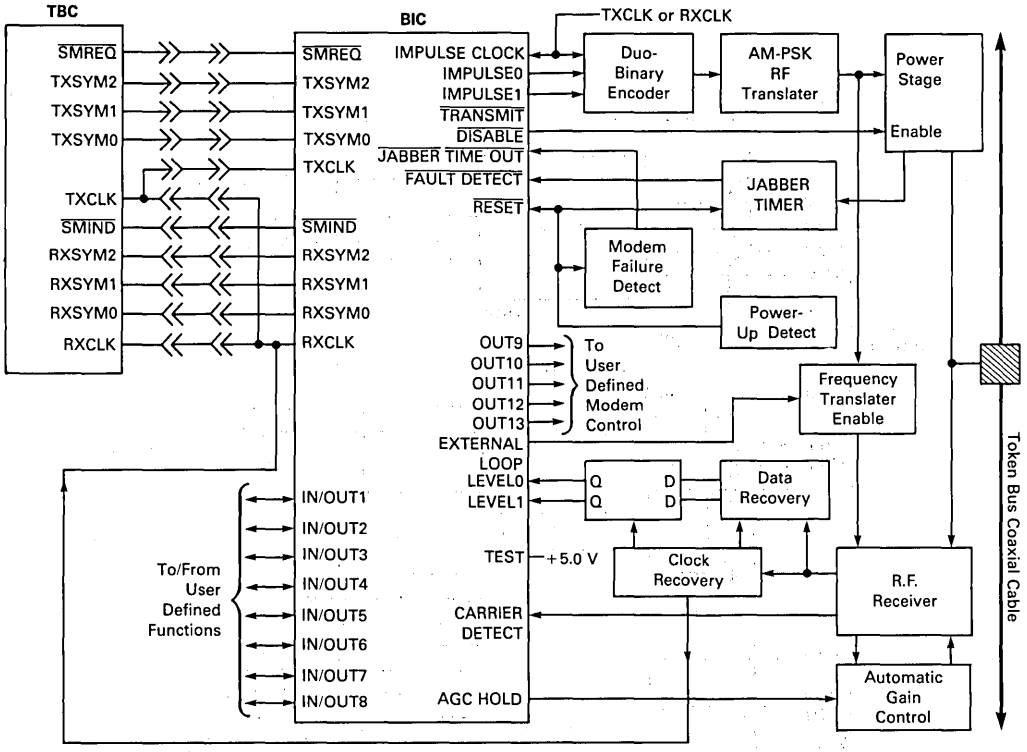


FIGURE 1-4 — PHYSICAL LAYER BLOCK DIAGRAM

7

SECTION 2
SIGNAL DESCRIPTION

The BIC signals can be broken into several groups including the serial interface, R.F. transmitter signals, R.F. receiver signals, and control signals. The twenty control signals have thirteen that are user-definable.

The terms **assertion** and **negation** will be used extensively. This is done to avoid confusion when dealing with a mixture of "active-low" and "active-high" signals. The term assert or assertion is used to indicate that a signal is active or true, independent of whether that level is represented by a high or low voltage. The term negate or negation is used to indicate that a signal is inactive or false. Positive logic is used throughout so a high logic level is equivalent to a one state.

Internal pullup resistors (>100 k) are provided on some inputs. These pullups are intended to drive the input to a known state if the input is disconnected. Because of the very small external drive capability (4.0 μA), it is not recommended to use the internal pullup resistors to drive external loads.

2.1 R.F. TRANSMITTER SIGNALS

The R.F. transmitter signals consist of IMPULSE0, IMPULSE1, and IMPULSE CLOCK.

2.1.1 IMPULSE0 AND IMPULSE1

IMPULSE0 and IMPULSE1 are output signals from the BIC encoder and pass duo-binary signals to the R.F. transmitter circuitry with the meanings as shown in Table 1. IMPULSE0 and IMPULSE1 can have one of two possible meanings depending on whether the duo-binary precoder is enabled or not. See Section 3.5 on the duo-binary precoder for more details.

TABLE 1 — ENCODINGS OF IMPULSE0 AND IMPULSE1
(Duo-binary encoding defined by ANSI/IEEE Std 802.4-1985 Table 14-1)

		Duo-Binary Precoder	
IMPULSE0	**IMPULSE1**	**Enabled**	**Disabled**
0	0	0	Zero
0	1	−2	One
1	0	+2	Non-Data
1	1	Silence	Silence

2.1.2 IMPULSE CLOCK

IMPULSE0 and IMPULSE1 are clocked out on the rising edge of the IMPULSE CLOCK. IMPULSE CLOCK must be at the same frequency as TXCLK and is normally either connected to TXCLK or RXCLK. As long as the synchronizer (see Subsection 3.6 on the synchronizer) is enabled, TXCLK, RXCLK, and IMPULSE CLOCK may have any phase relationship. All three must be at the exact same frequency, however.

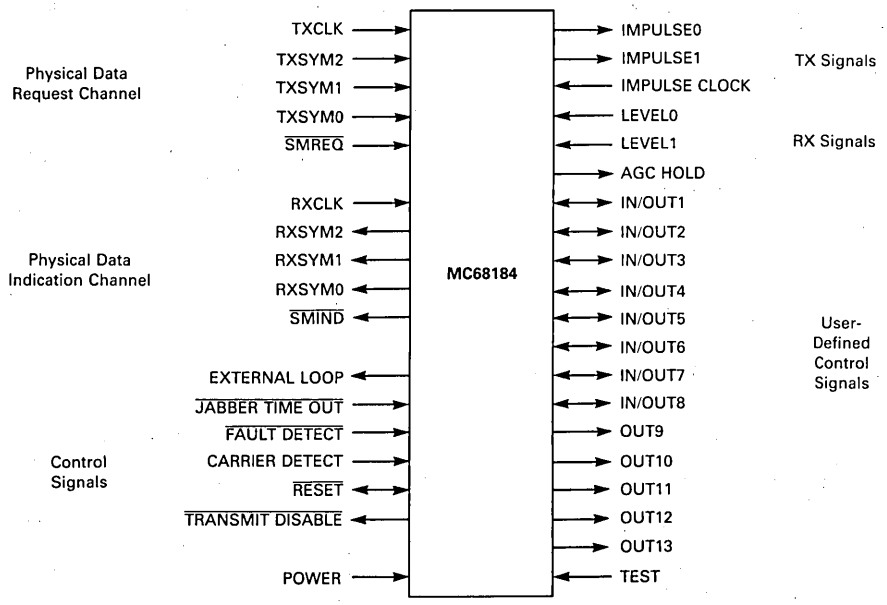


FIGURE 2-1 — MC68184 FUNCTIONAL PINOUT

7

2.2 R.F. RECEIVER SIGNALS

The R.F. receiver signals consist of LEVEL0, LEVEL1, and AGC HOLD.

2.2.1 LEVEL0 AND LEVEL1

LEVEL0 and LEVEL1 are input signals to the BIC decoder from the R.F. receiver circuitry. These signals are decoded by the BIC decoder as shown below and are clocked in on the falling edge of RXCLK, LEVEL0 and LEVEL1 are created in the R.F. receiver by comparing the amplitude of the received signal against preset levels. See subsection 3.10 on the error corrector for more details.

TABLE 2 — LEVEL0 AND LEVEL1 ENCODINGS

LEVEL0	LEVEL1	Encoding
0	0	Data Zero
0	1	Data One
1	0	2-0 Non-Data
1	1	2-4 Non-Data

2.2.2 AGC HOLD

The Automatic Gain Control (AGC HOLD) pin is negated whenever the CARRIER DETECT control pin is negated. If the CARRIER DETECT signal is asserted, the AGC HOLD pin will be high one and one-half RXCLK's after a start delimiter is presented on LEVEL0 and LEVEL1 pins. A start delimiter consists of the following sequence of eight bits: 2-0 or 2-4, 2-0 or 2-4, data zero, 2-0 or 2-4, 2-0 or 2-4, data zero, data zero, data zero.

When the CARRIER DETECT signal is asserted, the AGC HOLD will go low after one of the following occurs:

1) Twenty four of the same inputs are received on LEVEL0 and LEVEL1 (the encodings 2-0 and 2-4 are considered different inputs)
2) If two, three, or four Non-Datas (2-0 or 2-4) are contained in bits positions 1, 2, 4, and 5 relative to the start delimiter
3) If the error correcter is disabled (Bit 7 of Register 1 is set) and a Non-Data in bit positions 3, 6, 7, or 8 not belonging to a kicker is present
4) If the descrambler disable bit is set (Register 1 Bit 4), the error corrector is disabled, and a kicker is present
5) During reset.

The following shows a start delimiter and the bit positions of a message:

NN0NN000 12345678 12345678 12345678 12345678 . . .

In general, AGC HOLD will go high at the start of receiving a message and go low at the end of a message or if an uncorrectable error occurs in the message. The no transition detector (Subsection 3.8) performs condition 1 and essentially, conditions 2, 3, and 4 happen whenever a non-data is reported to the TBC. With the error corrector (Subsection 3.10) and the descrambler (Subsection 3.12) enabled, the only non-datas that can be reported are those in bit positions 1, 2, 4, 5. A single non-data in these positions would be considered an error and not be reported to the TBC. With the error corrector disabled and noise on the cable, a non-data could occur in any bit position and would in this case be reported. The non-datas belonging to "kickers" would be eliminated as long as the descrambler is enabled.

2.3 PHYSICAL DATA REQUEST CHANNEL

Five signals comprise the physical data request channel: TXCLK, $\overline{\text{SMREQ}}$, TXSYM2, TXSYM1, and TXSYM0. Three of these signals (TXSYM2, TXSYM1 and TXSYM0) are multiplexed and have different meanings depending on the mode selected by the state of $\overline{\text{SMREQ}}$. Internal pullup resistors (>100 kΩ) are also provided to drive $\overline{\text{SMREQ}}$, TXSYM2, TXSYM1 and TXSYM0 to the high state if disconnected.

2.3.1 TXCLK

The transmit clock is an input which can be up to 10.5 MHz. TXSYM2, TXSYM1, TXSYM0 and $\overline{\text{SMREQ}}$ are synchronized by the TBC to the positive edge of TXCLK. The IEEE 802.4 standard for broadband allows 1.0, 5.0, or 10 MHz clocks but requires TXCLK to be at exactly the same frequency as RXCLK.

2.3.2 $\overline{\text{SMREQ}}$

$\overline{\text{SMREQ}}$ indicates if the physical layer is in the MAC mode ($\overline{\text{SMREQ}}$ = 1) or in the Station Management mode ($\overline{\text{SMREQ}}$ = 0) of operation. In MAC mode, the node is on-line and data requests and data indication signals are passed over the serial interface. In Station Management Mode, the node is offline and accepts station management commands from the TBC.

2.3.3 TXSYM2, TXSYM1, TXSYM0

In Management Mode TXSYM2, TXSYM1 and TXSYM0 have the meanings shown in Table 3:

TABLE 3 — REQUEST CHANNEL MANAGEMENT MODE ENCODING ($\overline{\text{SMREQ}}$ = 0)

State	TXSYM2	TXSYM1	TXSYM0
Reset	1	1	1
Loopback Disable	1	0	1
Enable Transmitter	0	1	1
Serial SM Data/Idle	0	0	0/1

In Management mode, the Token Bus Controller (MC68824) can pass commands to the BIC. In addition to the three standard commands of reset, loopback disable and enable transmitter, user generated commands can be passed to the BIC to control or monitor the R.F. circuitry. See Section 5 for more details on management mode. Encodings not included in Table 3 are illegal commands. The BIC will NAK all illegal commands but take no other actions.

In MAC mode, the encodings for TXSYM2, TXSYM1, and TXSYM0 are shown in Table 4:

TABLE 4 — REQUEST CHANNEL DATA MODE ENCODING ($\overline{SMREQ}$ = 1)

Symbol	TXSYM2	TXSYM1	TXSYM0
ZERO	0	0	0
ONE	0	0	1
NON-DATA	1	0	*
PAD-IDLE	0	1	*
SILENCE	1	1	*

Where
ZERO is a data zero.
ONE is a data one.
NON-DATA is a delimiter flag and is always requested in pairs.
PAD-IDLE is one symbol of preamble/interframe idle.
SILENCE is silence or pseudo-silence.
*Don't care.

2.4 PHYSICAL DATA INDICATION CHANNEL

Five signals comprise the physical data indication channel: RXCLK, $\overline{SMIND}$, RXSYM2, RXSYM1, and RXSYM0. Three of these signals (RXSYM2, RXSYM1 and RXSYM0) are multiplexed and have different meanings depending on the state of $\overline{SMIND}$.

2.4.1 RXCLK

The receive clock can be up to 10.5 MHz. RXSYM2, RXSYM1, RXSYM0, and $\overline{SMIND}$ are synchronized to rising edge of RXCLK. The IEEE 802.4 standard for broadband networks allows 1.0, 5.0, or 10 MHz clocks.

2.4.2 $\overline{SMIND}$

$\overline{SMIND}$ signal indicates whether the BIC is in MAC mode ($\overline{SMIND}$ = 1) or Station Management mode ($\overline{SMIND}$ = 0) of operation. When in MAC mode of operation, RXSYM2, RXSYM1, and RXSYM0 are encoded indications of data reception. When in Station Management mode of operation, RXSYM2, RXSYM1 and RXSYM0 are encoded to confirm response to management commands.

2.4.3 RXSYM2, RXSYM1, RXSYM0

In management mode, the encoding for RXSYM2, RXSYM1, and RXSYM0 are shown in Table 5:

TABLE 5 — INDICATION CHANNEL MANAGEMENT MODE ENCODING ($\overline{SMIND}$ = 0)

State	RXSYM2	RXSYM1	RXSYM0
NACK (non-acknowledgement)	1	0	+
ACK (acknowledgement)	0	1	+
Idle Response	0	0	1
Physical Layer Error	1	1	1

+ Indicates RXSYM0 contains the SM RXdata when responding to a serial data command.

The encoding of RXSYM2, RXSYM1, and RXSYM0 in MAC mode are shown in Table 6:

TABLE 6 — INDICATION CHANNEL DATA MODE ENCODING ($\overline{SMIND}$ = 1)

Symbol	RXSYM2	RXSYM1	RXSYM0
ZERO	0	0	0
ONE	0	0	1
BAD-SIGNAL	0	1	1
NON-DATA	1	0	0
SILENCE	1	1	1

Where
ZERO is a data zero.
ONE is a data one.
BAD-SIGNAL indicates lack of carrier
NON-DATA is a delimiter flag. In the absence of errors, NON-DATAs will always be present in pairs.
SILENCE is silence or pseudo-silence.

2.5 CONTROL SIGNALS

The BIC has twenty control signals with thirteen of these being user-defined.

2.5.1 $\overline{RESET}$

The reset pin is a bidirectional signal which is open drain as an output. Whenever $\overline{RESET}$ is driven low, the BIC will reset. The reset pin must be driven low at power-up and held low for at least ten clock cycles after V_{DD} reaches recommended operation conditions. A pullup resistor (>100 kΩ) is provided to drive this pin to the inactive state if disconnected. An external pullup resistor of between 15 kΩ and 1.0 kΩ on the reset pin is recommended in order to have reasonable rise times (less than 4 clocks). In addition, when the BIC receives a reset command from the TBC, it will drive $\overline{RESET}$ low for as long as the reset command is asserted.

2.5.2 $\overline{JABBER\ TIME\ OUT}$

The $\overline{JABBER\ TIME\ OUT}$ pin is an input to the BIC from the external Jabber Timer circuitry as shown in Figure 1-4. When the $\overline{JABBER\ TIME\ OUT}$ pin is asserted, the BIC will drive IMPULSE0 and IMPULSE1 high (silence state), drive $\overline{TRANSMIT\ DISABLE}$ and $\overline{SMIND}$ low, and drive RXSYM2, RXSYM1, and RXSYM0 high. These pins will continue in these states until a reset is received by the BIC. The reset can occur by either driving the $\overline{RESET}$ pin low or sending a reset command to the BIC. A pullup resistor (>100 kΩ) is provided to drive the $\overline{JABBER\ TIME\ OUT}$ pin to the inactive state if disconnected.

2.5.3 $\overline{FAULT\ DETECT}$

The $\overline{FAULT\ DETECT}$ pin operates in the same manner as the $\overline{JABBER\ TIME\ OUT}$ pin. When the $\overline{FAULT\ DETECT}$ pin is driven low, the BIC will drive IMPULSE0 and IMPULSE1 high, drive $\overline{TRANSMIT\ DISABLE}$ and $\overline{SMIND}$ low, and drive RXSYM2, RXSYM1, and RXSYM0 high. These pins will continue in these states until a reset is

7

received by the BIC. The reset can occur by either driving the RESET pin low or sending a reset command to the BIC. An internal pullup resistor (>100 kΩ) is provided to drive the FAULT DETECT pin to the inactive state if disconnected.

2.5.4 CARRIER DETECT

When the CARRIER DETECT pin is not asserted, it indicates no carrier is present. This pin is driven by external circuitry which senses when the carrier is lost on the broadband cable. The AGC HOLD signal is driven low when CARRIER DETECT is negated. When CARRIER DETECT is negated and the BIC is not in internal loopback mode or in Management mode, then RXSYM2, RXSYM1, and RXSYM0 are driven to the LOW, HIGH, HIGH states, respectively. This is the bad-signal indication. When CARRIER DETECT is negated and the BIC is not in external loopback, then TRANSMIT DISABLE is asserted and IMPULSE0 and IMPULSE1 are driven high (silence). An internal pullup resistor (>100 kΩ) is provided to drive the CARRIER DETECT pin to the active state if disconnected.

2.5.5 EXTERNAL LOOPBACK

The EXTERNAL LOOPBACK pin is asserted if, since the last reset, Bits 1 and 2 of Register 1 have been set to 0 and 1 respectively and a loopback disable command has not been received by the BIC from the TBC. In this condition, the BIC is in external loopback mode.

2.5.6 TRANSMIT DISABLE

TRANSMIT DISABLE is asserted (low) by any one of the following conditions:

1) During a reset
2) When the transmitter is disabled as indicated by Bit 4 of Register 0
3) When SMREQ is low
4) When in internal loopback
5) When CARRIER DETECT *AND* EXTERNAL LOOPBACK are negated
6) When TXCLK is at zero frequency
7) When IMPULSE0 AND IMPULSE1 are high
8) When a physical error is being reported (RXSYM2 = 1, RXSYM1 = 1, RXSYM0 = 1 and SMIND = 0).

Whenever TRANSMIT DISABLE is asserted, IMPULSE0 and IMPULSE1 will both be forced high (silence condition).

TRANSMIT DISABLE allows smooth transmitter turn-on and turn-off by being negated a little before data starts coming out of IMPULSE0 and IMPULSE1 and by being asserted a little after data stops coming out of IMPULSE0 and IMPULSE1. In particular, ten IMPULSE CLOCKS after both IMPULSE0 and IMPULSE1 go high, TRANSMIT DISABLE will be asserted. Between four and ten IMPULSE CLOCKS before either IMPULSE0 or IMPULSE1 go low (from the high-high or silence state), TRANSMIT DISABLE will be negated unless some other condition is overriding. Note that TRANSMIT DISABLE will be asynchronous to IMPULSE CLOCK. The exact number of clocks depends on the mode of operation (such as whether the synchronizer is enabled) and the phase relationship between TXCLK and IMPULSE CLOCK.

If TXCLK is at a zero frequency, after approximately 16 RXCLKs, TRANSMIT DISABLE will be asserted. This is intended as a failsafe for remote systems. If the BIC is remote from the TBC, then in order to maintain proper timing, TXCLK is provided by the TBC as shown in Figure 1-4. If the cable between the BIC and the TBC is disconnected, the TXCLK frequency will go to zero while the RXCLK frequency, which is generated on the modem, will remain the same. The BIC will recognize this difference in frequency and assert TRANSMIT DISABLE. Note that TXCLK into the BIC must be at a valid input level, that is, it must not "float".

2.6 TEST SIGNAL

TEST is used during product testing and must be held high for normal operations.

2.7 USER DEFINED CONTROL SIGNALS

2.7.1 IN/OUT1–IN/OUT8

These eight independent pins are bidirectional. Register 3E contains the state of these eight pins. Register 3F determines if these pins are inputs or outputs. Each pin can be individually programmed as an output to control the R.F. section as needed. As inputs, these pins can be used to monitor the R.F. section. An internal pullup resistor (>100 kΩ) is provided for each IN/OUT pin.

2.7.2 OUT9–OUT13

These five output signals are user-defined. OUT9, OUT10, OUT11 are driven low at reset while OUT12 and OUT13 are driven high at reset. These outputs are controlled by Register 3D and can be used to control the R.F. section as required.

SECTION 3
BIC BLOCK DESCRIPTION

3.0 DETAILED BLOCK DESCRIPTION

Figure 3-1 shows a detailed block diagram of the BIC. The encoder includes the scrambler, kicker inserter, pseudo-silence adder, duo-binary precoder, and the impulse clock synchronizer. The decoder includes the start delimiter detector, no transition detector, kicker deleter, error corrector, pseudo-silence deleter, and the descrambler. The station management interface includes the command/data decoder and encoder, bad input detector, error monitor, TXCLK alive monitor, register control, and five registers.

3.1 COMMAND/DATA DECODER

The command/data decoder interprets the commands to the BIC and the data to be transmitted coming from the TBC. When $\overline{\text{SMREQ}}$ is low, the BIC is receiving commands as shown in Table 3 and when $\overline{\text{SMREQ}}$ is high, the BIC is receiving data as shown in Table 4.

3.1.1 DATA

Silence on $\overline{\text{SMREQ}}$, TXSYM2, TXSYM1, TXSYM0 indicates that no energy is to be transmitted. Since the minimum number of silence bits is two, a single silence bit is illegal. After silence must always come at least four octets (32 bits) of pad-idle. Pad-idle is generated in order to provide a signal for the headend to lock onto. The data decoder translates pad-idles into a repeating one-zero pattern. Note that the scrambler is disabled during the ones and zeros of pad-idle but not during the ones and zeros of a message. Following the pad-idle is a start delimiter. The start delimiter denotes the start of a frame and consists of the following eight bits: non-data, non-data, zero, non-data, non-data, zero, zero, zero. Following the start delimiter are up to 8191 octets of zeros or ones. This is the "information" part of the message. Following this is the end delimiter which indicates the end of the frame and consists of the following eight bits: non-data, non-data, one, non-data, non-data, one, one or zero, one or zero. Non-datas are only present in the start and end delimiters, must always come in pairs, and must have at least one data bit (zero or one) separating each non-data pair. Following the end delimiter is silence if this was the last frame to be transmitted, or pad-idle if another frame is to follow.

3.1.2 COMMANDS

In order to enter management mode and give a command, $\overline{\text{SMREQ}}$ is brought low. This will cause the BIC to bring $\overline{\text{SMIND}}$ low within eight RXCLKs. $\overline{\text{SMIND}}$ will also go high within eight RXCLKs of $\overline{\text{SMREQ}}$ going high (unless a physical error indication is present). In management mode, the TBC can control and interrogate the BIC by sending commands and by addressing registers internal to the BIC.

3.1.2.1 RESET COMMAND

When the BIC receives a reset command from the TBC it:

1) Stores whether a physical error indication is being caused by a bad input, $\overline{\text{JABBER TIME OUT}}$, and/or $\overline{\text{FAULT DETECT}}$.
2) Attempts to pull the $\overline{\text{RESET}}$ pin low.
3) Initializes the register control so that the next byte of serial station management data received will be considered the first byte of two.
4) If a physical error indication ($\overline{\text{SMIND}}$ = 0, RXSYM2 = RXSYM1 = RXSYM0 = 1) is not present, the BIC acknowledges the reset command ($\overline{\text{SMIND}}$ = RXSYM2 = 0, RXSYM1 = RXSYM0 = 1) within eight RXCLKs after the command is present. If a physical error indication is present, the RESET pin must go low before the reset command will be acknowledged.

While the $\overline{\text{RESET}}$ pin is low, either pulled low by the BIC after receiving a reset command or by external circuitry (such as the power up detector), a reset occurs. A reset has the following effects:

1) If a physical error indication is present, it is cleared.
2) All bits in Register 3F are forced low, which makes IN/OUT1 to IN/OUT8 inputs.
3) All bits in Register 3E are forced low. Since, at reset, IN/OUT1 to IN/OUT8 are inputs, this will have no effect until Register 3F is changed.
4) Bits 6, 7 of Register 3D are forced high, bits 1, 2, 3, 4, 5 are forced low. This will cause OUT12, OUT13 high, force OUT9, OUT10, OUT11 low and enable the duo-binary precoder, the bad input detector, and the impulse clock synchronizer.
5) Bits 0, 2, 3, 4, 6, 7 of Register 1 are forced low and bits 1, 5 are forced high. This selects internal loopback, enables the scrambler, kicker inserter, descrambler, kicker deleter, pseudo-silence subtracter, error corrector and disables the pseudo-silence adder.
6) Bits 4, 5, 7 of Register 0 are forced high and Bit 6 is forced low indicating internal loopback mode.
7) The loopback mode is enabled. Since Bits 1, 2 of Register 1 are forced to a high, low respectively, a reset places the BIC in the internal loopback mode.
8) The transmitter is disabled. This forces IMPULSE0, IMPULSE1 high and $\overline{\text{TRANSMIT DISABLE}}$ low.
9) AGC HOLD is forced low (after one RXCLK low to high transition).
10) If $\overline{\text{SMREQ}}$ is high (reset was caused by an external source pulling the $\overline{\text{RESET}}$ pin low), silence is reported ($\overline{\text{SMIND}}$ = RXSYM2 = RXSYM1 = RXSYM0 = 1).

The $\overline{\text{RESET}}$ pin must be driven low at power-up and held low for at least ten clock cycles (of TXCLK and RXCLK) after power becomes valid. During normal operation, the $\overline{\text{RESET}}$ pin must be held low for a minimum of two clock cycles of TXCLK and RXCLK for a valid reset to occur.

7

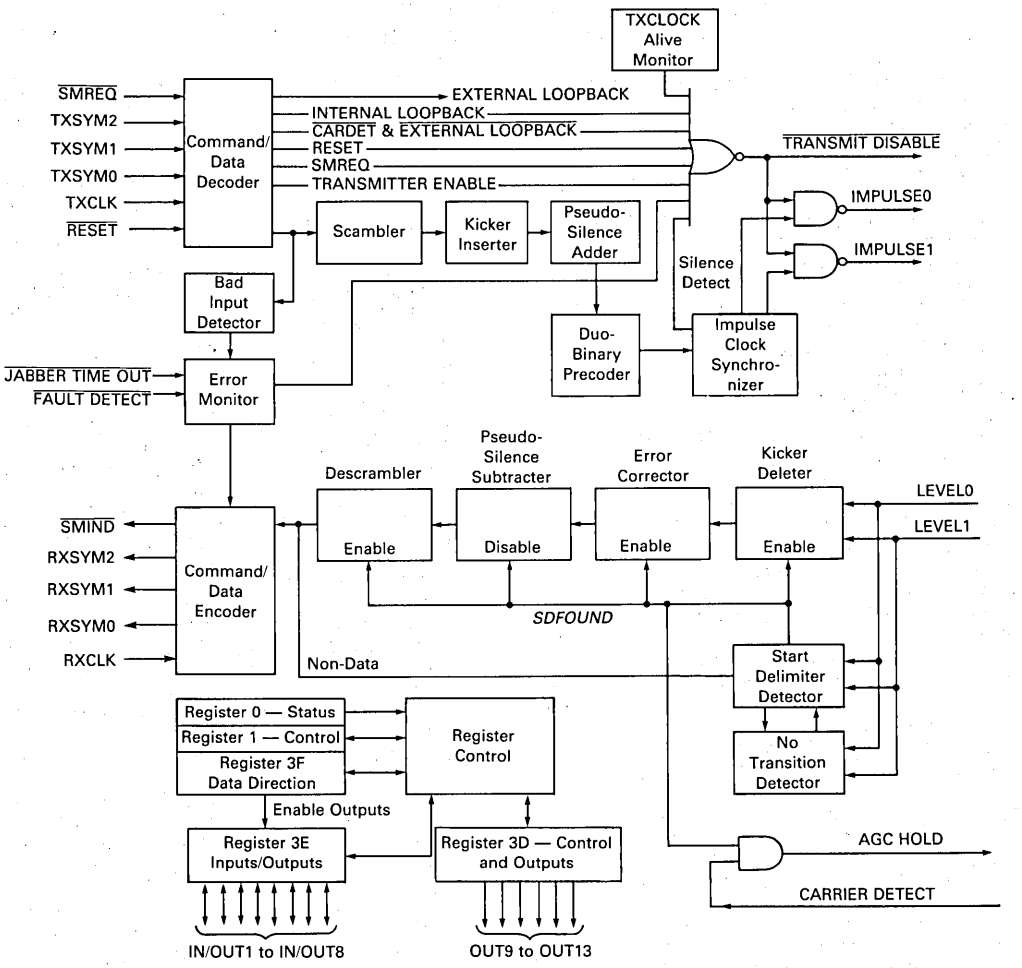


FIGURE 3-1 — BIC BLOCK DIAGRAM

3.1.2.2 LOOPBACK DISABLE COMMAND

When the BIC receives the loopback disable command ($\overline{SMREQ}$ = TXSYM1 = 0, TXSYM2 = TXSYM0 = 1), it goes out of loopback mode and forces Bits 5, 6, 7 of Register 0 low. A reset is required to bring Bit 7 of Register 0 back high. The register control is also initialized so that the next byte of serial station management data received will be considered the first byte of two. If a physical error indication is not present, the BIC acknowledges the loopback disable command — $\overline{SMIND}$ = RXSYM2 = 0, RXSYM1 = RXSYM0 = 1 within eight RXCLKs after the command is present.

3.1.2.3 ENABLE TRANSMITTER COMMAND

When the BIC receives the enable transmitter command, it forces Bit 4 of Register 0 low. Bit 4 of Register

0 must be low before the $\overline{\text{TRANSMIT DISABLE}}$ pin can go high and before IMPULSE0 or IMPULSE1 can go low. (See signal descriptions for additional restrictions.) A reset is required to bring Bit 4 of Register 0 back high. The register control is initialized so that the next byte of serial station management data received will be considered the first byte of two. If a physical error indication is not present, the BIC acknowledges the enable transmitter command — $\overline{\text{SMIND}}$ = RXSYM2 = 0, RXSYM1 = RXSYM0 = 1 within eight RXCLKs after the command is present.

3.1.2.4 SERIAL SM DATA/IDLE COMMANDS

These commands are passed to the register control which handles the reading and writing of the five registers as explained in sub-section 5.

3.2 SCRAMBLER

The scrambler algorithm is described in 14.9.2.3 of ANSI/IEEE Std 802.4-1985. The basic function is to take the data (consisting of only ones and zeros) in a message, divide it by the polynomial $1+X^{-6}+X^{-7}$, and output this result. This has two stated purposes: first, it helps randomize the spectral components of the transmitted modulation and second, it reduces the chance of transmitting a long string of zeros or a long string of ones. The receiver uses one to zero or zero to one transitions to recover the clock. If a long enough string of ones (or zeros) were allowed to be transmitted, the receiver would lose clock and start reporting erroneous data. Therefore, the scrambler is used to transform the data so that most long strings are eliminated. Even after being scrambled, however, there is still a small possibility of encountering just the right input pattern to produce a long output string of ones (or zeros). These are handled by the kicker inserter. The scrambler is disabled when Bit 3 of Register 1 is high (which also disables the kicker inserter) or when transmitting pad-idles.

3.3 KICKER INSERTER

The kicker inserter compares the current octet (eight bits) of data with the previous octet. If all the bits are the same, that is, there are sixteen ones or sixteen zeros, the last three bits of the current octet is replaced with a "kicker." Notice that since a kicker consists of three bits, the total number of bits transmitted is unchanged. The two possibilities are:

```
                 11111111 11111111      or
is replaced with 11111111 111110NN
                                | |
                              kicker

                 00000000 00000000
is replaced with 00000000 000001NN
                                | |
                              kicker
```

where N indicates a non-data bit.

The end result is that after kickers are inserted, the maximum number of consecutive bits that can be the same is 22. Any longer string will result in a kicker being inserted which contains a one to zero (or a zero to one) transition. The following shows the longest possible output string of consecutive zeros. Notice that the maximum number of consecutive zeros with a kicker would be 21.

```
10000000  00000000  00000001
```

The octet boundary is established at the start of the frame by the start delimiter and is maintained throughout the frame. The kicker inserter is disabled when Bit 3 of Register 1 is high (which also disables the scrambler) or when transmitting pad-idles.

3.4 PSEUDO-SILENCE ADDER

The pseudo-silence adder, if enabled, will replace silence with pseudo-silence. Pseudo-silence is the repeating sequence: non-data, non-data, zero, one. The pseudo-silence adder is used in loopback to simulate one of the headend functions. Normally, when the headend detects that no station is transmitting, it transmits pseudo-silence. This guarantees that the receivers will always have a clock signal (either valid signals or pseudo-silence) and can therefore have very long lock-up times. When the BIC is in internal loopback, the pseudo-silence adder can be enabled in order to test the pseudo-silence subtracter. In external loopback, the pseudo-silence adder acts like the headend so that a receiver with long lock-up times can be tested under "normal" conditions. The pseudo-silence adder is disabled when Bit 5 of Register 1 is high. It is also disabled when in reset, when not in external or internal loopback, and when $\overline{\text{SMREQ}}$ is low if in internal loopback. When in internal loopback and the pseudo-silence adder is disabled, such as after a reset, any silence sent to the BIC will be returned as non-data.

3.5 DUO-BINARY PRECODER

The duo-binary precoder does the translation described in Table 14-1 of ANSI/IEEE Std 802.4-1985. Its inputs are zero, one, non-data, silence and its outputs are 0, +2, −2, silence. A +2 output tells the duo-binary encoder (external to the BIC) to output a positive impulse. A −2 output indicates a negative impulse, a 0 output indicates no impulse, and silence indicates that the transmitter is to be turned off.

The duo-binary precoder formats the data for the external duo-binary encoder. In order to create a waveform of the correct amplitude, the duo-binary encoder adds the current amplitude to the amplitude of the previous bit. Therefore, the precoder must change the data to a form where this addition will produce the correct output. The terminology used is that a one, which is a maximum amplitude either positive or negative, is a +4

or a −4. A non-data, which is half amplitude either positive or negative, is a +2 or a −2. A zero, which is almost no energy, is a 0. In order to create a one, if the previous output of the precoder was a +2, the current output should be a +2. These add to produce the +4. If the previous output was a −2, the current output should be a −2 which makes a −4. In order to create a zero, if the previous output was a +2, the current output should be a −2 to add up to 0. If the previous output was a −2, the current output should be a +2. Creating non-datas is a little more difficult. Non-datas can only be output in pairs. The first non-data is created by outputting a 0. Since the previous output was either a +2 or a −2, this makes either a +2 or a −2, respectively. The second non-data is created by outputting a +2 or a −2. Either will add with the 0 to create a +2 or −2 respectively. The choice of either a +2 or a −2 in this case is not random. The second non-data must be the same output as the bit prior to the first non-data. The following example shows the input to the duo-binary precoder and the corresponding output:

```
input — S S S S  1  0  1  0  1  0  1  0 N N  0 N N  0  0  0  1  1  1  1 S S S
output— S S S +2 +2 −2 −2 +2 +2 −2 −2 +2 0 +2 −2 0 −2 +2 −2 +2 +2 +2 +2 +2 0 S S
```

As shown above, input silence is changed to output silence except for the first and last silence in a string of silences. The last silence (which comes right before transmitting) is changed to an output +2. The first silence is changed to an output 0. These are required to "ramp" the transmitter up and down by starting and stopping all transmissions with a non-data. The duo-binary precoder will give an incorrect output if silence is immediately followed by non-data. Silence should always be followed by pad-idle (or equivalent data).

The precoding functions (described above) will not be performed when the precoder is disabled, which is when Bit 0 of Register 3D is set or when in internal loopback. Instead, the duo-binary precoder outputs will be the same as the inputs with one exception. The last silence will be changed to a data one which provides a ramp-up function.

3.6 IMPULSE CLOCK SYNCHRONIZER

The impulse clock synchronizer is used if IMPULSE CLOCK is not externally connected to TXCLK. IMPULSE CLOCK must always be at the exact same frequency as TXCLK, (except when in internal loopback) but the impulse clock synchronizer permits IMPULSE CLOCK to have an arbitrary phase relation to TXCLK. An example of where the impulse clock synchronizer would be used is if RXCLK is externally connected to IMPULSE CLOCK. Since RXCLK is used to generate TXCLK, they are always at the same frequency. However, because of buffer and signal path delays, the phase relationship of TXCLK and RXCLK may not be known.

The impulse clock synchronizer is a FIFO with TXCLK being used to write data to the FIFO and IMPULSE CLOCK being used to read the data from the FIFO. Because the FIFO is of very limited length, it is important that TXCLK and IMPULSE CLOCK be at the same frequency to prevent an underrun or overrun. The impulse clock synchronizer is reset on the low to high transition of $\overline{\text{SMREQ}}$. Therefore, in order to initialize correctly, the management state should be entered after power-up and before the first data transmission.

The impulse clock synchronizer is used when in internal loopback to synchronize data from TXCLK to RXCLK and so should not be disabled. When not in internal loopback, the impulse clock synchronizer should only be disabled if TXCLK is connected to IMPULSE CLOCK. The impulse clock synchronizer is disabled when Bit 2 of Register 3D is high.

3.7 START DELIMITER DETECTOR

The start delimiter detector looks at LEVEL0 and LEVEL1 to determine if a start delimiter has been received. The output of the start delimiter detector, *SDFOUND*, is ANDed with CARRIER DETECT to produce AGC HOLD. *SDFOUND* is forced low after any one of the following occurs:

1) 24 of the same input are received on LEVEL0 and LEVEL1 (see no transition detector 3.8)
2) If a non-data is present at the descrambler output
3) If not in internal loopback and CARRIER DETECT goes low
4) During reset

Typically, *SDFOUND* will be high after a start delimiter is found and low when the non-data's in the following end delimiter are encountered. When low, *SDFOUND* disables those functions that are only to be performed on the data section of a message — the kicker deleter, error corrector, and the descrambler. When high, *SDFOUND* disables the pseudo-silence subtracter.

In internal loopback *SDFOUND* is not affected by CARRIER DETECT. Therefore, if *SDFOUND* is high and the BIC is in internal loopback, AGC HOLD will follow the state of CARRIER DETECT. If not in internal loopback, AGC HOLD will go low when CARRIER DETECT goes low and will not go high until CARRIER DETECT goes high and another start delimiter is found (in that order).

3.8 NO TRANSITION DETECTOR

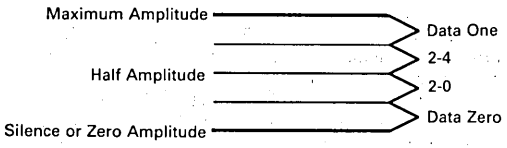
The no transition detector looks at LEVEL0 and LEVEL1 to determine if the same input has been present for 24 clocks. (The encodings 2-0 and 2-4 are considered different inputs.) If so, a reset command is given to the start delimiter detector which forces AGC HOLD low. Forcing the AGC HOLD output low will correct the most probable cause of error, the external automatic gain control in the receiver being held at the wrong level. Note: because of the kicker inserter on the transmitter, the receiver is guaranteed that in the absence of errors there will never be more than 22-bits of the same input.

3.9 KICKER DELETER

The kicker deleter looks at LEVEL0 and LEVEL1 and removes any kickers that are present. This is the inverse of the function of the kicker inserter. The kicker deleter is disabled when *SDFOUND* is low or if Bit 4 of Register 1 is high (which also disables the descrambler). If the kicker deleter is disabled and a kicker is present in the message, *SDFOUND* will go low once the start delimiter detector sees the kicker. This will enable the pseudo-silence subtracter which can cause the corruption of the end delimiter. For this reason, it is not recommended to disable the pseudo-silence subtracter except for testing purposes.

3.10 ERROR CORRECTOR

The error corrector attempts to correct errors in transmission to improve the bit error rate of the receiver. This is done by knowing what patterns of data's and non-data's are valid in a message. The error corrector can only be enabled when *SDFOUND* is high. Because *SDFOUND* is only high after a start delimiter, only data and the end delimiter should be present when the error corrector is enabled. Therefore, any non-datas that are not part of the end delimiter are errors that should be converted to datas. As shown below, a bit that is reported as a 2-4 (non-data) that should really be a data is much more likely to have been a one than a zero. Therefore, the error corrector will change invalid 2-4s to data ones and invalid 2-0s to data zeros.



This technique assumes that the error was caused by a small amount of noise which pushed the data out of its normal range into the nearest non-data range. In order to determine if a non-data is part of an end delimiter or not, four rules are used: (See 2.2.2 AGC HOLD for bit positions.)

1) Non-datas in bit positions 3, 6, 7, 8 can never be part of an end delimiter so always convert them to datas.
2) There must be four non-datas in bit positions 1, 2, 4, 5 to make an end delimiter so if there is only one non-data in these bit positions, make that non-data into a data.
3) If there are three non-datas in bit positions 1, 2, 4, 5 then an end delimiter probably had one of its non-datas converted by noise to a data so the error corrector converts the data into a non-data.
4) If two or four non-datas are present in bit positions 1, 2, 4, 5 the error corrector does nothing. Note that in cases 3 and 4, two or four non-datas will pass by the error corrector; which will reset the start delimiter detector; which will disable the error corrector (until the next start delimiter is found).

If the error corrector is not needed, it can be disabled by setting Bit 7 of Register 1 high. If the error corrector is disabled, only the no transition detector makes any distinction between a non-data 2-0 and non-data 2-4 on LEVEL0, LEVEL1.

3.11 PSEUDO-SILENCE SUBTRACTER

When the headend detects that no station is transmitting, instead of repeating the silence, the headend transmits pseudo-silence. Pseudo-silence consists of the repeating sequence: non-data (2-0 or 2-4), non-data (2-0 or 2-4), data zero, data one. This sequence can be terminated at any time and followed by data. The pseudo-silence subtracter replaces pseudo-silence with silence to make the pseudo-silence insertion (by the headend) transparent to the TBC. The pseudo-silence subtracter is disabled when a frame is being received on LEVEL0 and LEVEL1, that is, when *SDFOUND* is high. The pseudo-silence subtracter is also disabled when Bit 6 of Register 1 is high. When enabled, the pseudo-silence subtracter will:

change	NN01	to	SSSS;
change	NN00	to	SSSS;
change	NN10	to	SS10;
change	N101	to	S101;
change	N010	to	S010;

where N represents non-data (2-0 or 2-4) and S represents silence.

3.12 DESCRAMBLER

The descrambler performs the opposite function of the scrambler. Therefore, data that is scrambled and unscrambled is identical to the original data. This means that, in the absence of errors, the scrambling and descrambling of the data is transparent to the user (TBC). The descrambler is disabled when *SDFOUND* is low or if Bit 4 of Register 1 is set (which also disables the kicker deleter).

3.13 BAD INPUT DETECTOR

The bad input detector looks for an invalid sequence of data from the TBC (on $\overline{\text{SMREQ}}$, TXSYM2, TXSYM1, TXSYM0). The possible invalid sequences are:

1) A single non-data. That is, a non-data preceded and followed by data or silence or preamble. Non-datas are always supposed to come in pairs.
2) Three consecutive non-datas. There must always be at least one data between non-data pairs. Note: The output of the BIC to the RF transmitter could have four consecutive non-datas, if a kicker is inserted immediately prior to the end delimiter.
3) A single silence. In broadband systems, a minimum of two silences are required in order to correctly ramp down and ramp up. See the duo-binary precoder (3.5) description for more details on first and last silence.

When an invalid sequence is detected, it is reported to the error monitor. The bad input detector is disabled when Bit 1 of Register 3D is high. When the bad input

7

detector is disabled, Bit 2 of Register 0 will be low after a reset command.

3.14 ERROR MONITOR

Three sources of errors are possible, a fault in the modem, a jabber time out, and a bad input sequence from the TBC. The error monitor controls the reporting of these errors. FAULT DETECT is an input that is generated by an external modem failure detect circuitry and when activated, indicates that a failure has occurred and transmission should stop. JABBER TIME OUT is generated by an external jabber timer and when activated, indicates that the transmitter has been continuously putting out energy (transmitting) for approximately 1/2 second. The jabber timer is required by the ANSI/IEEE Std 802.4-1985 so that a modem will automatically shut itself off if its transmitter is "stuck on." Lastly, the bad input indication is generated internally as described previously.

When FAULT DETECT goes low, the error monitor forces a physical error indication, that is, SMIND is forced low and RXSYM2, RXSYM1, RXSYM0 are forced high. In addition, in order to stop transmission, TRANSMIT DISABLE is forced low and IMPULSE0, IMPULSE1 are forced high. The error monitor will maintain this state until a reset occurs, even if FAULT DETECT goes high. Typically, the physical error indication will cause the TBC to interrupt the host processor. The host processor can then execute an error recovery program. The error recovery program would give a reset command to the BIC, read Register 0 to determine what the error was, and then decide whether to continue or to stop all transmissions. When the BIC receives the reset command from the TBC; it pulls the RESET pin low. This resets the BIC and should also reset the external modem failure detector (and the jabber timer), forcing FAULT DETECT (and JABBER TIME OUT) high.

When a reset command is received by the BIC from the TBC, the error monitor will save whether the FAULT DETECT pin has gone low (in Bit 1 of Register 0) since the last reset. After the reset, Bit 1 of Register 0 will be set if the FAULT DETECT pin was the cause of the physical error indication. Figure 3-2 shows how the information is stored. Since this bit is unaffected by the reset pin, before the first reset command is received by the BIC this bit has no meaning. If the FAULT DETECT pin is permanently held low, the BIC will hold the physical error indication until a reset occurs which will override FAULT DETECT. Therefore, if the FAULT DETECT pin is held low and a reset command is given to the BIC, the BIC will pull the RESET pin low, overriding the FAULT DETECT, and then give an acknowledgement to the reset command (SMIND = RXSYM2 = 0, RXSYM1 = RXSYM0 = 1) while holding IMPULSE0, IMPULSE1 high and TRANSMIT DISABLE low. When the reset command goes away, the BIC will release the RESET pin. When

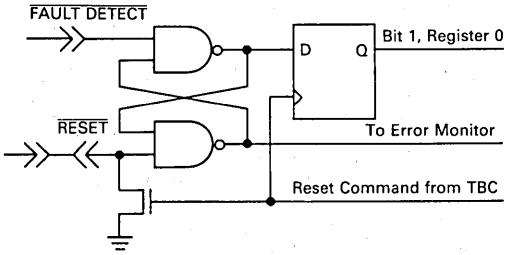


FIGURE 3-2 — FAULT DETECT INPUT

the RESET pin goes inactive, a physical error will again be generated by the error monitor. Note that, in this case, (FAULT DETECT is always low) it is impossible for the TBC to read any of the registers on the BIC.

The JABBER TIME OUT pin acts identically to the FAULT DETECT pin except that Bit 0 of Register 0 is used instead of Bit 1 to indicate that the external jabber timer was the cause of the error.

When the bad input detector indicates a bad input, the error monitor forces a physical error indication, forces TRANSMIT DISABLE low, and forces IMPULSE0, IMPULSE1 high. When a reset command is received, the error monitor saves (in Bit 2 of Register 0) whether a bad input was present. Reset clears the bad input detector but does not affect Bit 2 of Register 0. Therefore, Bit 2 of Register 0 has no meaning before the first reset is received by the BIC.

The FAULT DETECT, JABBER TIME OUT, and the bad input are independent errors. Therefore, if a physical error indication is present, then at least one of the errors has occurred but possibly two or even all three could have happened. This will be indicated in bits 0, 1, 2 of Register 0 if a reset command is used to clear the physical error indication. External circuitry can be used to clear a physical error indication by asserting the RESET pin. In this case, however, bits 0, 1, 2 of Register 0 will not reflect the current cause of the physical error indication.

3.15 REGISTER CONTROL

The register control is used to read and write the five registers (Register 0 is read only). In order to write (or read) a register, two octets of serial station management data are sent to TXSYM0 while SMREQ, TXSYM2, TXSYM1 are held low. The BIC responds to each octet on the TXSYM pins with an octet on the RXSYM pins. The first octet to the BIC (on TXSYM0) defines which register is to be written (or read) and whether a write or a read is to be performed. The BIC responds to this

by repeating the same octet on RXSYM0. The format of the first octet is:

start bit | six bit address (LSB ... MSB) | don't care bit | read/$\overline{\text{write}}$ bit | stop bit

The start bit is a zero and the stop bit is a one. Following the register address is a don't care bit. The BIC does not use this bit but the current IEEE 802.4 recommendation defines this bit as low. Following the don't care bit is the read/$\overline{\text{write}}$ bit. The read/$\overline{\text{write}}$ bit is high if a read is to be performed and low if a write is to be performed. The second octet to the BIC (on TXSYM0) contains the data to be written to the selected register. The format of the second octet is:

start bit | bit position 0, bit position 1, . . . , bit position 7 | stop bit

If a read function is being performed, the second octet contains no useful information but is a place holder to tell the BIC when to respond with the second octet. The BIC responds to receiving the second octet by transmitting the contents of the selected register on RXSYM0. In the case of a write, the register is first written and then the contents of the selected register are transmitted. A write to Register 0 is interpretted the same as a read to Register 0. If the selected register is not one of the five supported by the BIC, the second octet from the BIC (on RXSYM0) will contain the contents of Register 0.

In order to determine the start bit and stop bit of the octets, the following ordering is used. Before an octet is presented to the BIC (on TXSYM0), the TBC can generate any number of "idle requests" ($\overline{\text{SMREQ}}$ = TXSYM2 = TXSYM1 = 0, TXSYM0 = 1). The TBC then generates a start bit ($\overline{\text{SMREQ}}$ = TXSYM2 = TXSYM1 = TXSYM0 = 0), eight data bits ($\overline{\text{SMREQ}}$ = TXSYM2 = TXSYM1 = 0, TXSYM0 = data), and a stop bit ($\overline{\text{SMREQ}}$ = TXSYM2 = TXSYM1 = 0, TXSYM0 = 1). This must be followed by idle requests until the BIC has put the stop bit of the response octet on the RXSYM pins.

When the BIC receives an idle request, it generates an "idle response" ($\overline{\text{SMIND}}$ = RXSYM2 = RXSYM1 = 0, RXSYM0 = 1). The idle response is overridden by a response octet as described below.

In response to an octet (first or second) from the TBC, the BIC will:

1) Approximately six clocks after the stop bit (of the octet from the TBC), the BIC will bring RXSYM1 high and RXSYM2 low if the selected register is valid (acknowledgement) or bring RXSYM2 high and RXSYM1 low if the selected register is invalid (non-acknowledgement). RXSYM2, RXSYM1 are held in one of these states until after the stop bit has been sent.
2) Five RXCLKs later, the BIC will bring RXSYM0 low for one bit to create a start bit.
3) The eight data bits of the response octet are then presented on RXSYM0, LSB first.
4) RXSYM0 is then held high for one bit to create a stop bit.
5) The idle response is then given as long as the idle request is present.

In order to determine which octet of two that the BIC is receiving, it uses the following algorithm: The BIC will always expect the first octet after receiving a reset command, a loopback disable command, an enable transmitter command, or after $\overline{\text{SMREQ}}$ has been high. After receiving the first octet, the BIC will expect the second unless one of the above occurs. If the first octet selects an invalid register, the BIC will give a non-acknowledgement while transmitting the first octet to the TBC. The BIC will then be expecting the second octet (unless one of the above occurs first). When the BIC receives the second octet, it will again give a non-acknowledgement and will transmit the contents of Register 0 with the non-acknowledgement.

Note that when writing to a register, some bits may not read the same as written. Except for the following cases, this would be an error. All of Register 0 and Bit 0 of Register 1 are read only and are not affected by an attempt to write to them. (Bit 0 of Register 1 is not used and is always read as low.) Also, any bits in Register 3E that are inputs (as defined by Register 3F) will show the state of the respective IN/OUT pin, not what is written. When in internal loopback the duo-binary precoder is always disabled. Therefore, Bit 0 of Register 3D will always read high when in internal loopback. Since errors due to noise are possible, it is recommended that after writing to a register, several reads be performed to verify the contents.

3.16 LOOPBACK MODES OF OEPRATION

Two loopback modes, internal loopback and external loopback, provide a method to test the BIC and the modem. Internal loopback is used to test the connection to the BIC and most of the encoder and decoder of the BIC.

3.16.1 INTERNAL LOOPBACK

Internal loopback is entered if a loopback disable command has not been received since the last reset and Bits 1, 2 of Register 1 are high, low respectively. After a reset, the BIC goes to internal loopback mode.

When in internal loopback, the BIC:

1) Forces high Bits 5 and 7 of Register 0 and Bit 6 of Register 0 low.
2) Forces $\overline{\text{TRANSMIT DISABLE}}$ low. This, in turn, will force IMPULSE0, IMPULSE1 high.

7

3) Disables the duo-binary precoder which forces Bit 0 of Register 3D low. The decoder can not interpret the output of the precoder so it must be turned off. This means the duo-binary precoder is not tested in internal loopback mode.
4) Disables the pseudo-silence adder while $\overline{\text{SMREQ}}$ is low.
5) Disconnects IMPULSE CLOCK from the impulse clock synchronizer and connects RXCLK instead. This means that RXCLK can have an arbitrary (but constant) phase relationship with TXCLK but the synchronizer must not be disabled in internal loopback.
6) Disables the no transition detector.
7) Allows *SDFOUND* to go high, even if CARRIER DETECT is low. This permits the testing of the descrambler, error corrector, and the kicker deleter. Note that AGC HOLD will still go low if CARRIER DETECT is low.

Note that in internal loopback, if the pseudo-silence adder is disabled, silence on the TXSYM pins is translated to non-data on the RXSYM pins (after the encoder and decoder delays).

3.16.2 EXTERNAL LOOPBACK

External loopback is used to test the modem circuitry external to the BIC. External loopback is entered if a loopback disable command has not been received since the last reset and Bits 1, 2 of Register 1 are low, high respectively.

When in external loopback, the BIC:
1) Asserts the EXTERNAL LOOPBACK pin.
2) Forces Bits 6 and 7 of Register 0 high and Bit 5 of Register 0 low.
3) CARRIER DETECT negated does not assert $\overline{\text{TRANSMIT DISABLE}}$ or bring IMPULSE0, IMPULSE1 high when in external loopback.

3.16.3 INVALID LOOPBACK

If Bits 1, 2 of Register 1 are high-high or low-low and a loopback disable command has not been received since the last reset command, then the BIC is in neither internal loopback or external loopback. In this case, the effect is the same as if loopback was disabled.

SECTION 4
BIC REGISTERS

Five registers are present in the BIC. All registers are read/write except Register 0 which is a read only register. Bit 0 is the LSB for all registers and therefore is the first bit transmitted on the serial SM commands.

7

4.1 REGISTER 0

Register 0 is an 8-bit, read-only register with the following format:

7	6	5	4	3	2	1	0
LE	EL	IL	TD	CF	IF	MF	JTO

JTO JABBER TIME OUT. This bit indicates the $\overline{\text{JABBER TIME OUT}}$ pin was active (low) prior to the last reset command from the TBC when high. Before the first reset command is given (after the BIC is powered up), this bit has no meaning.

MF MODEM FAILURE. This bit indicates the $\overline{\text{FAULT DETECT}}$ pin was active (low) prior to the last reset command from the TBC when high. Before the first reset command is given, (after the BIC is powered up), this bit has no meaning.

IF INPUT FAILURE. This bit indicates the bad-input detector (from the encoder) received an invalid state prior to the last reset command from the TBC when high. If the bad-input detector is disabled, this bit will be low after a reset command. Before the first reset command is given, (after the BIC is powered up), this bit has no meaning.

CF CABLE FAILURE. This bit indicates the inverse of the state of the Carrier Detect pin.

TD TRANSMIT DISABLE. This bit indicates that the transmitter has been disabled (by reset) when high.

IL INTERNAL LOOPBACK. This bit indicates that the BIC is currently in internal loopback state when high.

EL EXTERNAL LOOPBACK. This bit indicates that the BIC is currently in external loopback state when high.

LE LOOPBACK ENABLE. This bit indicates that loopback has been enabled by a reset when high. If this bit is high and neither internal loopback or external loopback is high, the BIC will take no action.

4.2 REGISTER 1

Register 1 is an 8-bit read/write register with the following format:

7	6	5	4	3	2	1	0
ECD	PSD	PSA	DD	SD	LS2	LS1	NU

NU NOT USED. Read always as zero. Should always be written as zero.

LS1& LS2 LOOPBACK SELECT 1 AND LOOPBACK SELECT 2. These bits determine which loopback state to go into when loopback is enabled as shown in the table below:

LOOPBACK SELECT 1	LOOPBACK SELECT 2	Mode
0	0	None
0	1	External Loopback
1	0	Internal Loopback
1	1	None

SD SCRAMBLER DISABLE. This bit disables the encoder scrambler and the kicker inserter when high.

DD DESCRAMBLER DISABLE. This bit disables the descrambler and kicker deleter when high.

PSA PSEUDO-SILENCE ADDER DISABLE. This bit disables the pseudo-silence adder in the encoder when high. The pseudo-silence adder only operates when the BIC is in either internal loopback or external loopback mode.

PSD PSEUDO-SILENCE SUBTRACTER DISABLE. This bit disables the pseudo-silence subtracter in the BIC decoder when high.

ECD ERROR CORRECTOR DISABLE. This bit disables the BIC decoder error corrector when high.

4.3 REGISTER 3D

Register 3D is an 8-bit read/write register which can disable functions of the BIC as well as control the state of the five user-defined output pins. Register 3D has the following format:

7	6	5	4	3	2	1	0
O13	O12	O11	O10	O9	SYD	BID	DBD

DBD DUO-BINARY DISABLE. This bit disables the duo-binary precoder when high. The duo-binary precoder is automatically disabled when in internal loopback. In internal loopback, this bit is always low.

BID BAD INPUT DISABLE. This bit disables the bad input detector in the BIC encoder when high.

SYD SYNCHRONIZER DISABLE. This bit disables the IMPULSE CLOCK synchronizer when high.

O9 OUT9. This bit controls the state of OUT9. When this bit is read, it indicates the actual TTL state of OUT9 and not the requested state. A reset forces OUT9 low.

O10 OUT10. This bit controls the state of OUT10. When this bit is read it indicates the actual TTL state of OUT10 and not the requested state. A reset forces OUT10 low.

O11 OUT11. This bit controls the state of OUT11. When this bit is read it indicates the actual TTL state of OUT11 and not the requested state. A reset forces OUT11 low.

O12 OUT12. This bit controls the state of OUT12. When this bit is read it indicates the actual TTL state of OUT12 and not the requested state. A reset forces OUT12 high.

O13 OUT13. This bit controls the state of OUT13. When this bit is read it indicates the actual TTL state of OUT13 and not the requested state. A reset forces OUT13 high.

4.4 REGISTER 3E

Register 3E is an 8-bit read/write register which allows the control bits to be read or written. When read, this register always indicates the actual TTL state of the IN/OUT1 to IN/OUT8 pins. Writing to a bit of Register 3E that has been defined as an input by the respective IN/OUT DIRECTION bit in Register 3F does not affect what is read. However, it will change the pin state when changed to an output. IN/OUT1 is the LSB of Register 3D and therefore is the first bit transmitted on serial SM commands to/from the TBC.

4.5 REGISTER 3F

Register 3F is an 8-bit read/write register which determines if the IN/OUT pins are being read from or written to. Reset forces IN/OUT DIRECTION1 to IN/OUT DIRECTION8 bits low defining IN/OUT1 to IN/OUT8 as inputs.

4.6 REGISTER RESET CONDITIONS

A reset has the following effects on the BIC registers:

1) All eight bits of Register 3F are forced low which makes IN/OUT1–IN/OUT8 inputs.
2) All eight bits of Register 3E are forced low.
3) Bits 6 and 7 of Register 3D are forced high and Bits 5 to 0 are forced low. This forces OUT13 and OUT12 high and OUT11, OUT10, and OUT9 low at reset and enables the duo-binary precoder, the bad input detector and the synchronizer.
4) Bits 5 and 1 of Register 1 are forced high and Bits 7, 6, 4, 3, and 2 are forced low. This forces the internal loopback mode with the pseudo-silence adder disabled.
5) Bits 7, 5, and 4 of Register 0 will be forced high and Bits 6 will be forced low which enables loopback and disables the transmitter.

4.7 PROGRAMMING CONSIDERATIONS

To have the BIC operate as expected, some care must be taken when writing the software that provides management of the BIC.

1) After power-up, and before transmitting on the network, the TBC must enter management mode at least once. This is required in order to reset the synchronizer. This is normally not a problem because man-

agement mode must be entered to disable loopback mode, enable the transmitter, etc. (Note that at power-up, the BIC must receive a reset in order to put it in a known condition.)

2) When a physical error occurs, the BIC will indicate it to the TBC on the serial interface. The TBC will then (if enabled) interrupt the processor. The interrupt routine must provide some sort of error recovery. There are three types of errors: jabber time out, fault detect, and bad input. The interrupt routine should first give a reset command to the BIC so the type of error can be determined. Bits 0, 1, 2 of Register 0 will then indicate whether the error was a jabber time out, fault detect, and/or bad input, respectively. A jabber time out error happens when the transmitter has been on longer than the protocol will allow (1/2 second). This can happen because of a hardware fault in the modem, a hardware fault in the TBC, or a software problem in the TBC. The last can occur, for example, if one of the data buffers on the transmit queue has been linked back to itself. A reset of the TBC might be required to reinitialize the queues. As with all of these errors, the error recovery routine could keep track of the error as to cause and how many so that corrective action can take place. If a jabber time out keeps reoccurring, the system should stop transmitting and report the failure. The fault detect failure is a hardware failure in the modem. Depending on what type of failure is indicated, the modem may or may not be able to recover by being reset. The bad input error indicates that the TBC gave an incorrect input to the BIC. This could be caused by a hardware error or by noise on the serial interface between the TBC and the BIC. Since noise is much more likely, resetting the BIC and reprogramming it should be sufficient.
3) BITS 0, 1, 2 of Register 0 are different from the rest of the bits in the BIC because they do not indicate the current condition of the BIC, but the prior history. These bits are not valid until a reset command is given to the BIC. As stated above, after a reset command, these bits indicate the source(s) of an error, if any, and are used to track down the error problem. More than one bit can be high at a time, indicating that more than one error condition occurred since the last reset. It is not possible in this case to determine which error occurred first.
4) When giving a reset command to the BIC, some guidelines are necessary. The BIC will drive it's $\overline{\text{RESET}}$ pin low when it receives the reset command. The BIC will not completely reset itself until the $\overline{\text{RESET}}$ pin goes low (for two clocks). The loading on the $\overline{\text{RESET}}$ pin should be kept to less than 50 pF and 12 mA so that the normal reset command used by the TBC will be sufficient. Secondly, after the reset command, the BIC is not ready for the next command until the $\overline{\text{RESET}}$ pin has gone high. If a 15 kΩ or less resistor is used as a pullup on the $\overline{\text{RESET}}$ pin, a waiting period is not required after the reset command. If a large external load is present on the $\overline{\text{RESET}}$ pin, or a larger value resistor is used, it becomes necessary to wait before giving the next command after every reset command to the BIC.
5) Register 3D is used to program the outputs OUT9, OUT10, OUT11, OUT12, and OUT13. When writing to Register 3D, the state requested is first put on the pins. Then the actual state of the pins is read and returned to the TBC. If there is a discrepancy between what is written and what is read, it could be caused by either noise on the serial interface or a hardware failure in the modem. To eliminate the noise problem, the register should be read several times and the results compared. If this does not clear up the problem, several writes (with several reads per write) should be attempted. If incorrect data is still read back, then a hardware error is likely.
6) When programming the BIC, in several cases what is read will not correspond to what is written. For example, Register 0 is read only. Therefore, a write to Register 0 is seen the same as a read by the BIC. Register 3E, when read, will always indicate the true (TTL) state of the IN/OUT1 to IN/OUT8 pins. What is written is the state the output pins will go to. Register 3F controls which pins are inputs and which pins are outputs. Therefore, when writing to Register 3E, you should compare what is read back but mask out the bits that ARE inputs, which are the bits that are low in Register 3F. After a reset, IN/OUT1–IN/OUT8 are all inputs. If some or all of IN/OUT1–IN/OUT8 are to be outputs, it is recommended to first write to Register 3E the state that the output is to go to. What is read back will not correspond to what is written and should be ignored. Then Register 3F should be written to make outputs where needed. At this point, Register 3E should be read to make sure the outputs are in the correct state. Bit 0 of Register 1 is currently not used and will read as a zero. It is recommended that writes to Register 1 always contain zero in Bit position 0. Bit 0 of Register 3D (DBD) will read as a zero when in internal loopback. If a one is written to DBD and then internal loopback mode is disabled, DBD will then indicate a one when read.

SECTION 5
SERIAL INTERFACE

The BIC serial interface consists of ten signals (TXCLK, TXSYM2, TXSYM1, TXSYM0, $\overline{\text{SMREQ}}$, RXCLK, RXSYM2, RXSYM1, RXSYM0, and $\overline{\text{SMIND}}$) as shown in Figure 5-1. The serial interface has two modes of operation, MAC mode and Station Management Mode. In MAC mode of operation, this interface provides a means for the Token Bus Controller to transfer requests for data transmission and for the BIC to indicate data reception. In Station Management Mode of operation, the serial interface allows the TBC to reset the BIC, enable the transmitter, disable loopback, and read and write the BIC internal registers.

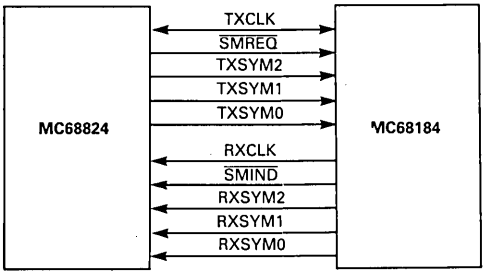


FIGURE 5-1 — SERIAL INTERFACE

5.1 PHYSICAL DATA REQUEST CHANNEL

In MAC operation mode, the physical data request channel provides encoded requests for data unit transmission. Figure 5-2 shows the signals on the physical data request channel during data transmission. $\overline{\text{SMREQ}}$ goes high indicating MAC mode and TXSYM2, TXSYM1 and TXSYM0 indicate the Physical symbol encodings shown in Table 4.

5.2 PHYSICAL DATA INDICATION CHANNEL

In MAC operation mode, the physical data indication channel provides encoded indications of data unit reception. Figure 5-3 shows the signals on the physical data indication channel during reception of data. $\overline{\text{SMIND}}$ goes high indicating MAC mode and RXSYM2, RXSYM1 and RXSYM0 indicate the Physical symbols encodings shown in Table 6.

5.3 PHYSICAL LAYER MANAGEMENT

In Station Management mode, commands are sent to the BIC from the Token Bus Controller and confirmations of those commands are sent back to the Token Bus Controller. Valid commands for the request channel are RESET, LOOPBACK DISABLE, ENABLE TRANSMITTER, and SERIAL SM DATA/IDLE as shown in Table 3.

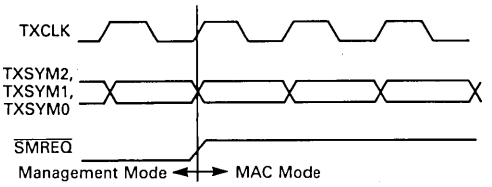


FIGURE 5-2 — TRANSMISSION ON SERIAL INTERFACE

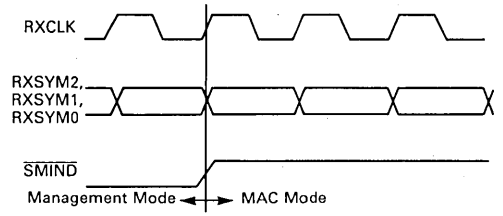


FIGURE 5-3 — RECEPTION ON SERIAL INTERFACE

The following is a typical station management sequence which could be used to place the BIC in external loopback mode.

1) First a reset command is issued on the physical data request channel.
2) Within eight RXCLKs, the BIC will respond to the reset command with an ACK on the physical data indication channel. As shown in Table 5, an ACK has $\overline{\text{SMIND}}$ low, RXSYM2 low, and RXSYM1 high. Since the reset command is not a serial data command, RXSYM0 is high.
3) Next, an idle request ($\overline{\text{SMREQ}}$ asserted, TXSYM2 negated, TXSYM1 negated and TXSYM0 asserted) is given on the physical request channel. Since there is never a start bit (TXSYM0 low), no serial SM data is transferred. See the next section for more details on serial SM data.
4) Within eight RXCLKs, the BIC will respond to the idle request with an idle response ($\overline{\text{SMIND}}$ asserted, RXSYM2 negated, RXSYM1 negated and RXSYM0 asserted) on the physical indication channel.
5) An ENABLE TRANSMITTER COMMAND is given on the request channel. This is necesary to enable IMPULSE0 and IMPULSE1 and before $\overline{\text{TRANSMIT DISABLE}}$ can be negated.
6) Within eight RXCLKs, the BIC gives an ACK response. Once again, RXSYM0 is high because ENABLE LOOPBACK is not a serial SM command.

7

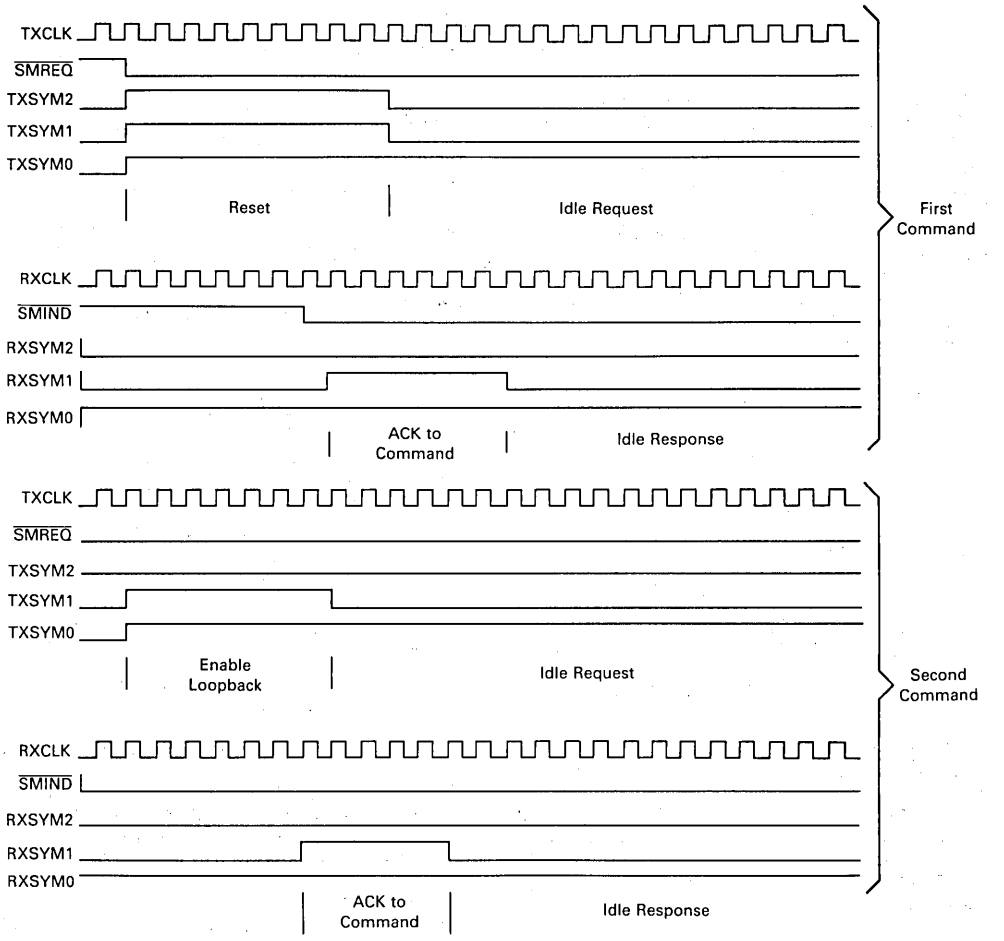


FIGURE 5-4 — STATION MANAGEMENT (SM) RESET AND LOOPBACK SEQUENCE

7) An idle request is again given on the request channel.
8) Within eight RXCLKs, the idle response is given on the physical data indication channel.
9) Register 1, is then written to, so that Bits 1 and 2 are low, high respectively. This is done using serial SM commands described below.

5.4 STATION MANAGEMENT COMMANDS AND RESPONSES

When the serial interface is in station management mode, the serial SM data command (TXSYM2 = 0, TXSYM1 = 0) is used to send data to the Physical Layer. Data is sent via TXSYM0. TXSYM0 = 1 is used as a data "one" and as a stop bit. TXSYM0 = 0 is used as a start bit and as a data "zero."

When TXSYM2 and TXSYM1 are negated, TXSYM0 is typically asserted first. This is an idle request until TXSYM0 is negated. A start bit is formed when TXSYM0 is negated for the first time. The next eight bits on TXSYM0, following the start bit, is the first octet of a two octet "command." TXSYM0 is then asserted for one bit to produce a stop bit. TXSYM0 should remain asserted until the response is complete on the physical data indication channel. The BIC will respond to the

original idle request with an idle response as decribed previously. After the BIC receives the start bit, eight data bits, and the stop bit, it will continue to give the idle response for approximately six RXCLKs, and then will either assert RXSYM2, and negate RXSYM1 (NACK) or will negate RXSYM2 and assert RXSYM1 (ACK). Five RXCLKs later, RXSYM0 will be negated for one bit to provide a start bit, following the start bit will be eight data bits on RXSYM0; RXSYM0 will then be asserted for one bit to produce the stop bit, RXSYM2 and RXSYM1 will be negated after the stop bit producing an idle response.

After the first octet, a second octet is transferred to complete the command. See 3.15 Register Control for more details on the format of the first and second octets. After receiving the first octet, the BIC will be expecting the second octet of data. If, before receiving the second octet, $\overline{\text{SMREQ}}$ is negated or a valid command (reset, loopback disable, transmitter enable) is present, then the BIC will go back to expecting the first octet.

Figure 5-5 shows the case of a serial SM data command, a write to Register 3D, being issued on the request channel to the BIC. This figure does not show the 5 to 6 clock delay between the end of the command and the start of the response. The 6 bit address for Register 3D is 111101. Since the least significant bit of the address is sent first, the address field is 101111. The format of the SM data command octet is: start bit = 0, 6 bit address = 101111, don't care bit = 0, read/write bit equal 0 and stop bit = 1. The first octet plus the start and stop bits is 0101111001. Within 5 to 6 clocks, the BIC will respond to the command with either an ACK or NACK. After the ACK to the SM data command, the data indication channel repeats the data sent to the BIC.

In this example, the second octet sent will control OUT9, OUT10, OUT11 and OUT12 to set these outputs to 1101, respectively. The data word 00011010 is passed with the least significant bit first. Adding the start and stop bits to the data gives 0000110101.

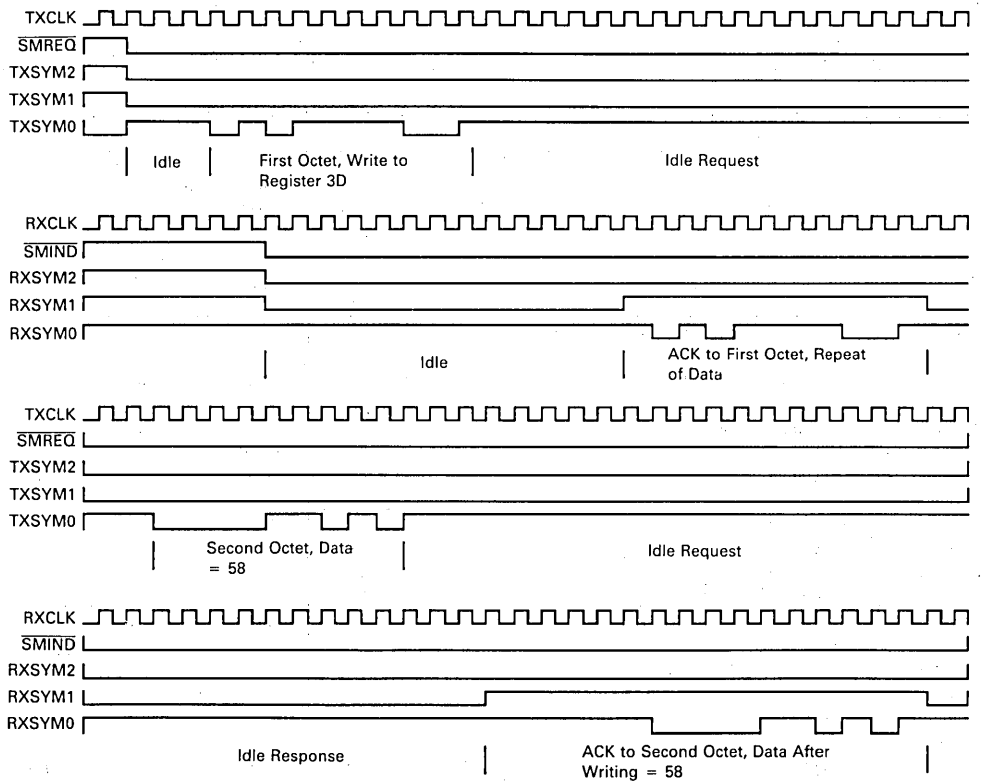


FIGURE 5-5 — STATION MANAGEMENT SEQUENCE

SECTION 6
ELECTRICAL SPECIFICATIONS

This section contains electrical specifications and associated timing information. All voltages are referenced to V_{SS}. This device contains circuitry to protect the inputs against damage due to high static voltages or electric fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum rated voltages to this high impedance circuit. For proper operation it is recommended that V_{in} and V_{out} be constrained to the range $V_{SS} \leq (V_{in} \text{ or } V_{out}) \leq V_{DD}$.

6.1 ABSOLUTE MAXIMUM RATINGS

Parameter	Symbol	Value	Unit
DC Supply Voltage	V_{DD}	−0.5 to 7.0	V
DC Input, Output Voltage	V_{in}, V_{out}	−0.5 to V_{DD} + 0.5	V
DC Current Drain Per Pin, Any Input or Output	I	25	mA
Storage Temperature	T_{stg}	−65 to +150	°C
Lead Temperature (10 second soldering)	T_L	300	°C

6.2 RECOMMENDED OPERATING CONDITIONS (to guarantee functionality)

Parameter	Symbol	Min	Max	Unit
DC Supply Voltage	V_{DD}	3.0	6.0	V
Input Voltage, Output Voltage	V_{in}, V_{out}	0	V_{DD}	V
Operating Temperature	T_A	−40	+85	°C

6.3 DC ELECTRICAL CHARACTERISTICS (Note: $\overline{\text{CARRIER DETECT}}$, $\overline{\text{JABBER TIME OUT}}$, $\overline{\text{RESET}}$, $\overline{\text{FAULT DETECT}}$ are CMOS level inputs and all others are TTL.

Parameter	Symbol	V_{DD}	25°C Typical	−40 to +85°C Guaranteed Limit	Unit
Minimum High-Level Input Voltage, **CMOS** input.	V_{IH}	4.5 5.5	2.4 2.9	3.15 3.85	V
Maximum Low-Level Input Voltage, **CMOS** input.	V_{IL}	4.5 5.5	1.8 2.2	1.35 1.65	V
Minimum High-Level Input Voltage, **TTL** input.	V_{IH}	4.5 5.5	1.6 1.6	2.0 2.0	V
Maximum Low-Level Input Voltage, **TTL** input.	V_{IL}	4.5 5.5	1.2 1.2	0.8 0.8	V
Minimum High-Level Output Voltage (V_{in} = V_{IH} or V_{IL}, I_{out} = −20 μA)	V_{OH}	4.5 5.5	4.499 5.499	4.4 5.4	V
Minimum High-Level Output Voltage (V_{in} = V_{IH} or V_{IL}, I_{out} = −4.0 mA)[1]	V_{OH}	4.5	4.0	3.7	V
Maximum Low-Level Output Voltage (V_{in} = V_{IH} or V_{IL}, I_{out} = 20 μA)	V_{OL}	4.5 5.5	0.001 0.001	0.1 0.1	V
Maximum Low-Level Output Voltage (V_{in} = V_{IH} or V_{IL}, I_{out} = 4.0 mA)[1]	V_{OL}	4.5 5.5	0.2 0.2	0.4 0.4	V
Maximum Input Leakage Current, No Pull-Up Resistor (V_{in} = V_{DD} or V_{SS})	I_{in}	5.5	±0.0001	±1.0	μA
Maximum Input Current, with Pull-Up Resistor (V_{in} = V_{SS})	I_{in}	5.5	—	−45	μA
Minimum Input Current, with Pull-Up Resistor (V_{in} = 2.0 V)	I_{in}	4.5	—	−4.0	μA
Maximum Output Leakage Current, Three-State Output = High Impedance (V_{out} = V_{DD} or V_{SS})	I_{OZ}	5.5	±0.05	±5.0	μA

(continued)

6.3 DC ELECTRICAL CHARACTERISTICS — (continued)

Parameter	Symbol	V_{DD}	25°C Typical	−40 to +85°C Guaranteed Limit	Unit
Maximum Quiescent Supply Current (V_{in} = V_{DD} or V_{SS}, I_{out} = 0 μA, Clocks = 0 MHz)	I_{QDD}	5.5	0.025	0.65	mA
Maximum Supply Current (V_{in} = V_{DD} or V_{SS}, I_{out} = 0 μA, Clocks = 10 MHz)	I_{DD}	5.5	25	50	mA
Maximum Input Capacitance[2]	C_{in}	—	—	10	pF
Maximum Output Capacitance[3]	C_{out}	—	—	15	pF
Maximum I/O Capacitance[4]	$C_{I/O}$	—	—	15	pF
Power Dissipation, Clocks = 10 MHz	P_D	5.5	0.14	0.3	W

1. $|I_{out}|$ equals 20 mA for TXDISP and AGC HOLD, 12 mA for OUT12 and OUT13 and $\overline{\text{RESET}}$, 8.0 mA for EXTERNAL LOOPBACK
2. RXSYM0, RXSYM1, RXSYM2, $\overline{\text{SMIND}}$, RXCLK, TEST, TXCLK, $\overline{\text{SMREQ}}$, TXSYM2, TXSYM1, TXSYM0, IMPULSE CLOCK, CARRIER DETECT, $\overline{\text{JABBER TIME OUT}}$, $\overline{\text{FAULT DETECT}}$, LEVEL0, LEVEL1
3. $\overline{\text{TRANSMIT DISABLE}}$, EXTERNAL LOOPBACK, IMPULSE0, IMPULSE1, AGC HOLD
4. IN/OUT1, IN/OUT2, IN/OUT3, IN/OUT4, IN/OUT5, IN/OUT6, IN/OUT7, IN/OUT8, OUT9, OUT10, OUT11, OUT12, OUT13, $\overline{\text{RESET}}$

6.4 AC ELECTRICAL CHARACTERISTICS (Output Load equals 0 pF to 50 pF)

Number (See Figures 6-2, 6-3)	Characteristic	Symbol	Min	Max	Unit
1	RXCLK, TXCLK, IMPLUSE CLOCK frequency	f_S	—	10.5	MHz
2	RXCLK, TXCLK, IMPULSE CLOCK period	t_{CP}	95	—	ns
3	TXCLK, RXCLK, IMPULSE CLOCK width high	t_{CWH}	40	—	ns
4	TXCLK, RXCLK, IMPULSE CLOCK width low	t_{CWL}	40	—	ns
5	TXCLK, RXCLK, IMPULSE CLOCK rise/fall time	t_{CRF}	—	50	ns
6	RXSYM2, RXSYM1, RXSYM0, $\overline{\text{SMIND}}$ delay time	t_{RXD}	10	50	ns
7	$\overline{\text{SMREQ}}$, TXSYM2, TXSYM1, TXSYM0 setup time	t_{TXSU}	20	—	ns
8	$\overline{\text{SMREQ}}$, TXSYM2, TXSYM1, TXSYM0 hold time	t_{TXHT}	5.0	—	ns
9	LEVEL0 and LEVEL1 setup time	t_{LSUT}	10	—	ns
10	LEVEL0 and LEVEL1 hold time	t_{LHT}	20	—	ns
11	USER I/O delay from serial interface	$t_{I/OD}$	2.0	2.0 + 70 ns	TXCLKs
12	IMPULSE0, IMPULSE1 delay time	t_{IMD}	5.0	35	ns
13	$\overline{\text{JABBER TIME OUT}}$, $\overline{\text{FAULT DETECT}}$ width low	t_{JFWL}	25	—	ns
14	$\overline{\text{RESET}}$ minimum width low	t_{RWL}	2.0	—	CLKs

6.5 THERMAL INFORMATION

$R_{\theta JA}$ = 45°C/W (Max)

$T_J = T_A + (P_D \times R_{\theta JA})$

$P_D = (V_{DD} \times I_{DD}) + PIO$

Where PIO is the P_D on pins (user determined) which can be neglected in most cases, e.g., when T_A = 85°C and P_D = 0.3 Watts; T_J is calculated to be 99°C.

7

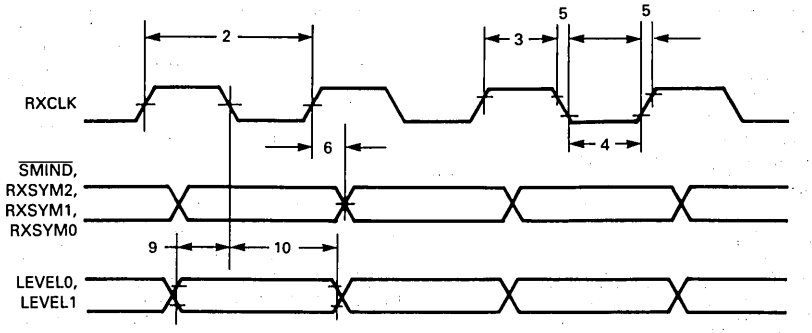


FIGURE 6-1 — SERIAL INTERFACE RECEIVER TIMING

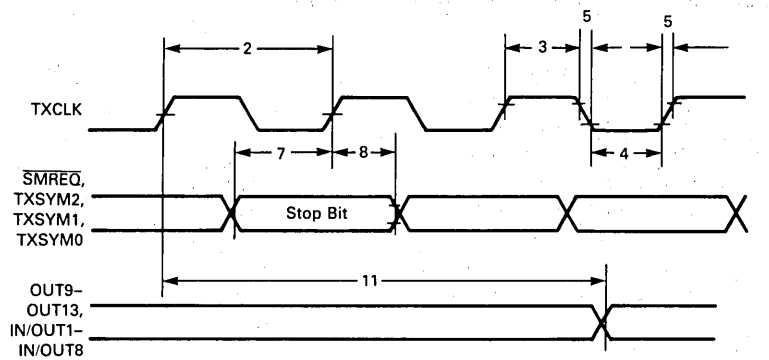


FIGURE 6-2 — SERIAL INTERFACE TRANSMITTER TIMING

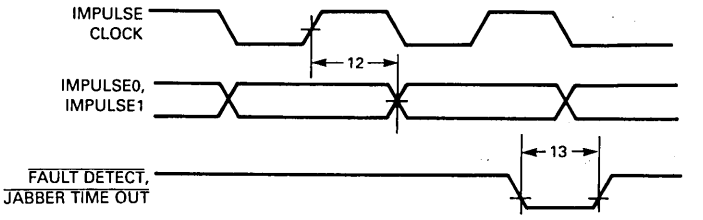


FIGURE 6-3 — IMPULSE OUTPUTS AND FAULT INPUTS

MC68185

Product Preview

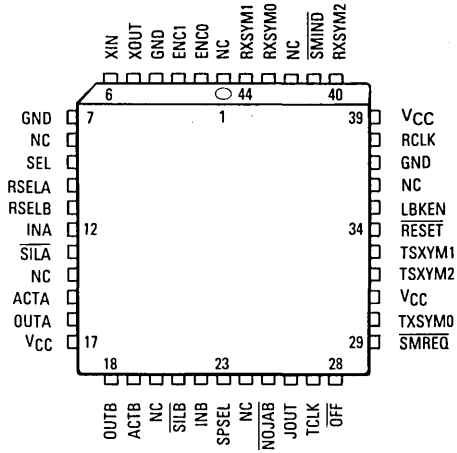
Twisted-Pair Modem

The MC68185 twisted-pair modem (TPM) chip is used in conjunction with the MC68824 token bus controller (TBC), an RS485 transceiver, and twisted-pair media to implement a very low-cost token bus node. The TPM interfaces to the TBC via the IEEE 802.4 recommended exposed DTE-DCE interface, thus providing physical layer management, including media access control (MAC) symbol encoding/decoding at data rates up to 2 Mbps.

The MC68185 provides the following features:

- Implements Manchester, Differential Manchester, or Level Encoding
- On-Chip Clock Recovery without External Components
- Interfaces to the MC68824 TBC via IEEE 802.4 Recommended Exposed DTE-DCE Interface
- Physical Layer Management Includes Local Loopback Mode, Transmitter Enable, and Reset
- Supports Data Rates up to 2 Mbps
- Supports Dual Media for Redundant Applications
- On-Chip Jabber-Inhibit Timer for 1 or 2 Mbps Data Rates
- Noise Reduction for Received Data
- External Clock Rate from dc to 2 MHz
- Crystal Oscillator to Generate a Transmit Clock (3 kHz to 2 MHz)
- Low-Power CMOS
- 44-Lead PLCC Package

PIN ASSIGNMENT



7

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

GENERAL DESCRIPTION

As a part of Motorola's token bus node, shown in Figure 1, the TPM provides a very low-cost local area network solution. The twisted-pair modem chip, with the RS485 drivers and receivers, performs the functions of the physical layer of the seven-layer open system interconnect (OSI) network model, using twisted-pair wires for the media. The TPM modulates the data transmission to the twisted-pair bus, receives and demodulates data from the bus, and manages the physical layer upon request by the TBC. The RS485 drivers and receivers translate the TPM TTL levels to and from RS485 levels.

Station management commands are passed from the TBC over the IEEE 802.4 recommended exposed DTE-DCE interface to the TPM. The management commands provide the ability to reset the TPM, to select the transmitters and receivers for redundant configuration, and to enable a local loopback mode for testing. An on-chip timer provides a jabber-inhibit function to turn off the transmitter and report an error condition if the transmitter is on beyond a user-specified time. A digital filter mode aids the bit integrity of the incoming data.

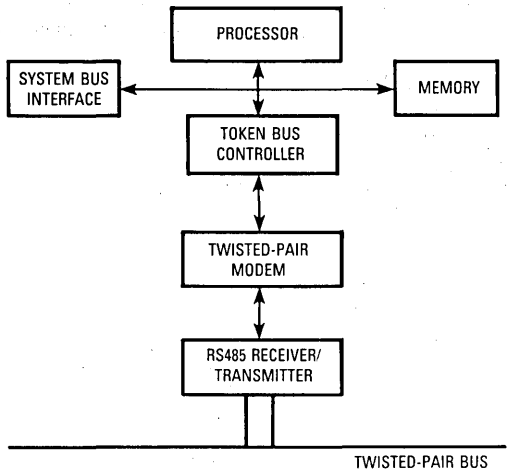


Figure 1. Token Bus Node

MC68194

Technical Summary

Carrierband Modem (CBM)

The bipolar LSI MC68194 Carrierband Modem (CBM) when combined with the MC68824 Token-Bus Controller provides an IEEE 802.4 single-channel, phase-coherent carrierband LAN connection. The CBM performs the Physical Layer function including symbol encoding/decoding, signal transmission and reception, and physical management.

Features include:

- Implements IEEE 802.4 Single-Channel, Phase-Coherent FSK Physical Layer Including End-of-Transmission Receiver Blanking
- Provides Physical Layer Management
- Supports Data Rates from 1 to 10 Mbps — IEEE 802.4 Standards use 5 or 10 Mbps
- Interfaces via Standard Serial Interface to MC68824 Token-Bus Controller
- Crystal Controlled Transmit Clock
- Local Loopback Mode for Testing
- Recovery of Clocked Data through Phase-Locked Loop
- Adjustable Signal-Detection Threshold
- RC Controlled Jabber-Inhibit Timer
- Single +5.0-Volt Power Supply

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

GENERAL DESCRIPTION

The MC68194 Carrierband Modem (CBM) is part of Motorola's solution for an IEEE 802.4 token bus carrierband LAN node. The CBM integrates the function of the single-channel phase-coherent FSK physical layer. Figure 1 illustrates the architecture of a token bus LAN node as commonly used in MAP industrial communications. Based on the ISO-OSI model shown in Figure 2, the LLC Sublayer and additional upper layers are typically supported by a local MPU subsystem, while the IEEE 802.4 token bus MAC Sublayer and Physical Layer are implemented by the MC68824 Token-Bus Controller (TBC) and MC68194 CBM respectively.

The MC68194 provides the three basic functions of the physical layer including data transmission to the coax cable, data reception from the cable, and management of the physical layer. For standard data mode (also called MAC mode), the carrierband modem receives a serial transmit data stream from the MC68824 TBC (called symbols or atomic symbols), encodes, modulates the carrier, and transmits the signal to the coaxial cable. Also in the data mode, the CBM receives a signal from the cable, demodulates the signal, recovers the data, and sends the received data symbols to the TBC. End-of-transmission receiver blanking as required by IEEE 802.4 is supported. Communication between the TBC and CBM is through a standardized serial interface consistent with the IEEE 802.4 DTE-DCE interface.

The physical layer management provides the ability to reset the CBM, control the transmitter, and do loopback testing. Also, an on-board RC timer provides a "jabber" inhibit function to turn off the transmitter and report an error condition if the transmitter has been continuously on for too long. Similar to the data mode, the CBM management mode makes use of the TBC serial interface.

The CBM uses phase-coherent shift keying (FSK) modulation on a single channel system. In this modulation technique, the two signaling frequencies are integrally related to the data rate, and transitions between the two

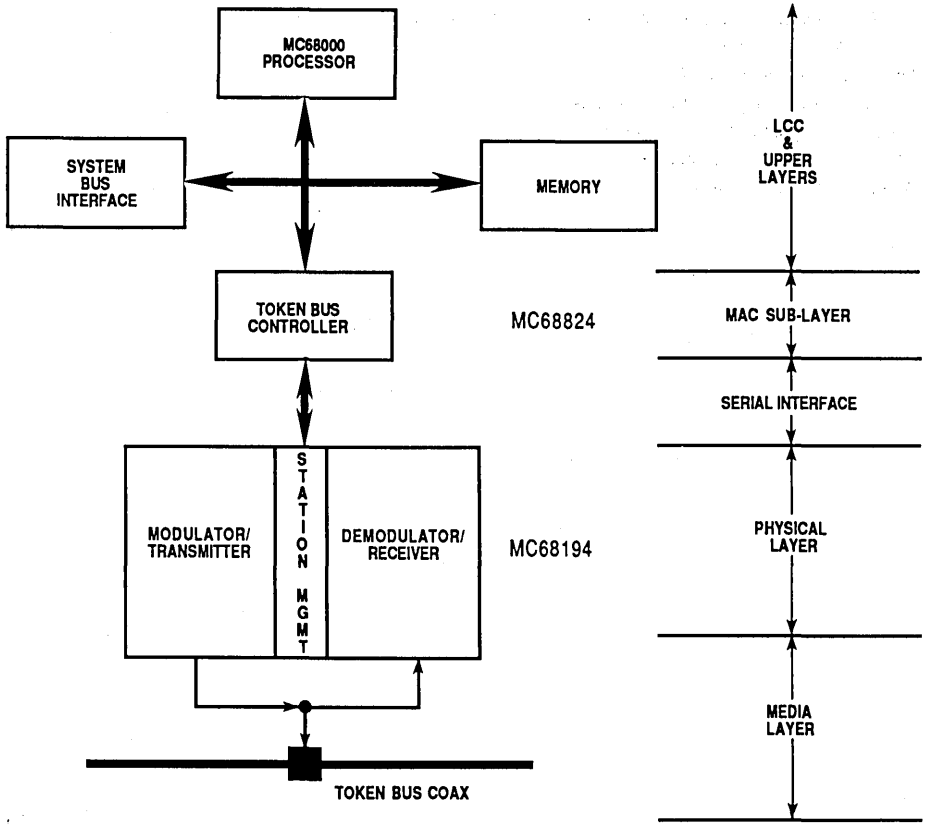


Figure 1. Token Bus Node

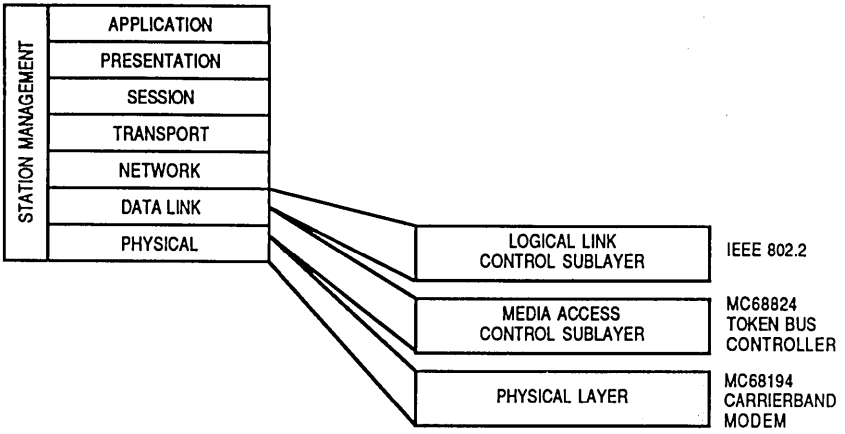


Figure 2. OSI Model

signaling frequencies are made at zero crossings of the carrier waveform. Table 1 shows the data rate and signaling frequencies. An {L} is represented as one-half cycle of a signal, starting and ending with a nominal zero amplitude, whose period is equal to the period of the data rate, with the phase of one-half cycle changing at each successive {L}. An {H} is represented as one full cycle of a signal, starting and ending with a nominal zero amplitude whose period is equal to half the period of the data rate. In a 5 Mbps implementation, the frequency of {L} is 5 MHz and for {H} is 10 MHz. For a 10 Mbps implementation, the frequency of {L} is 10 MHz and for {H} is 20 MHz. The other possible physical symbol is when no signal occurs for a period equal to one half of the period of the data rate. This condition is represented by {off}.

Table 1. Data Rate vs Signaling Frequencies

Data Rate Mbps	Frequency of Lower Tone MHz {L}	Frequency of Higher Tone MHz {H}
5	5.0	10.0
10	10.0	20.0

The specified physical symbols ({L}, {H}, and {off}) are combined into pairs which are called MAC-symbols. The MAC-symbols are transferred across the serial link. The encodings for the five MAC-symbols are shown in Table 2. Figure 3 shows the phase coherent FSK modulation scheme for ONE, ZERO, and NON-DATA. The IEEE 802.4 document does not specify the polarity used to transmit data on the physical cable. The receiver must operate without respect to polarity.

Figure 4 illustrates the functional blocks of the CBM and peripheral circuitry required for an IEEE 802.4 carrierband 5 Mbps or 10 Mbps data rate phase-coherent

Table 2. MAC Symbol Encodings

MAC-Symbol	Encoding
SILENCE	{OFF OFF}
PAD-IDLE PAIRS	{L L}{H H}
ZERO	{H H}
ONE	{L L}
NON-DATA ND1 ND2	 {H L} {L H}

FSK physical layer. A number of passive components directly support CBM operation to set the jabber inhibit timer and data recovery timing. In addition, an external crystal or clock source is required (20 MHz for 5 Mbps data rate or 40 MHz for 10 Mbps data rate). The receive clock recovery is based on a phased-locked loop which uses an active filter with an external op amp.

For the coaxial cable interface, the CBM can directly receive the filtered signal from the cable, meeting the IEEE 802.4 requirement of a 4 dB to 10 dB (1mV, 75 Ω) [dBmV] threshold window. The receive threshold is trimmable if desired by the user. For signal transmission, the CBM provides a set of differential transmit outputs (ECL signals referenced to V_{CC}, i.e., logic high = 4.1 V and logic low = 3.3 V) and a TX disable signal. The IEEE 802.4 requires a +63 dBmV to +66 dBmV transmit level and as a result an amplifier with waveshaping is required. Typically, an RF transformer is used for connection to the cable.

Although primarily intended for the IEEE 802.4 carrierband, the CBM is also an excellent device for point-to-point data links, fiberoptic modems, and proprietary LANs. The MC68194 can be used over a wide range of frequencies and interfaces easily into different kinds of media.

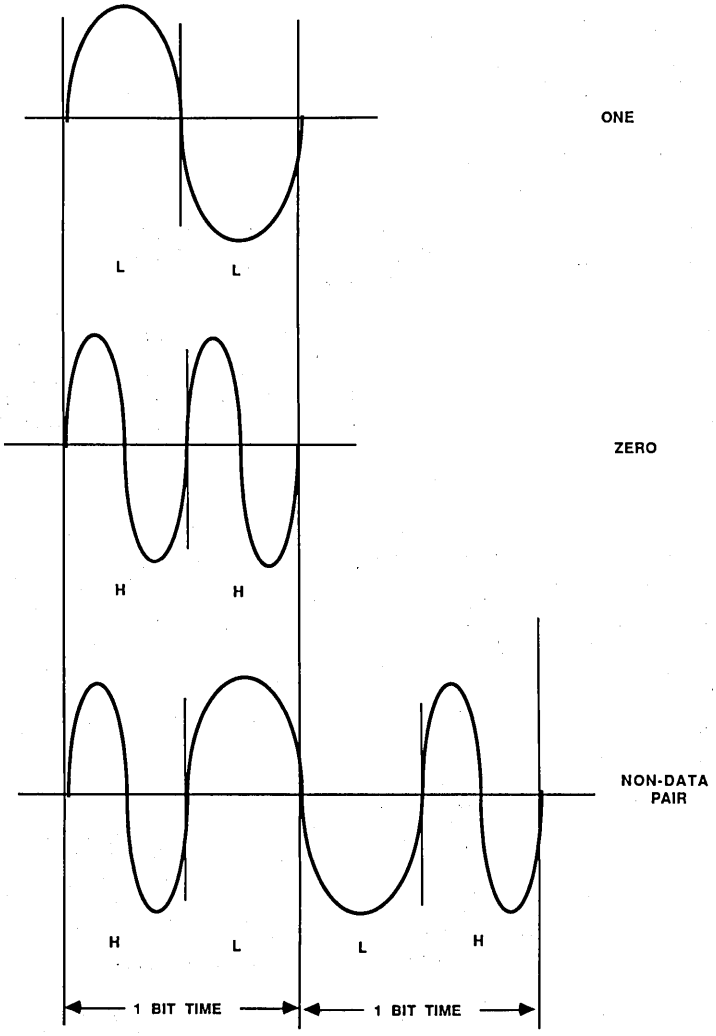


Figure 3. Phase-Coherent FSK Modulation Scheme

SERIAL INTERFACE

The serial interface is composed of the physical data request channel and the physical data indication channel. Five signals comprise the physical data request channel including TXSYM0, TXSYM1, TXSYM2, TXCLK, and $\overline{\text{SMREQ}}$. The physical data indication channel is composed of RXSYM0, RXSYM1, RXSYM2, RXCLK, and $\overline{\text{SMIND}}$. The serial interface is used to pass commands and data frames between the TBC and the CBM. This interface is based on the IEEE 802.4 DTE-DCE interface.

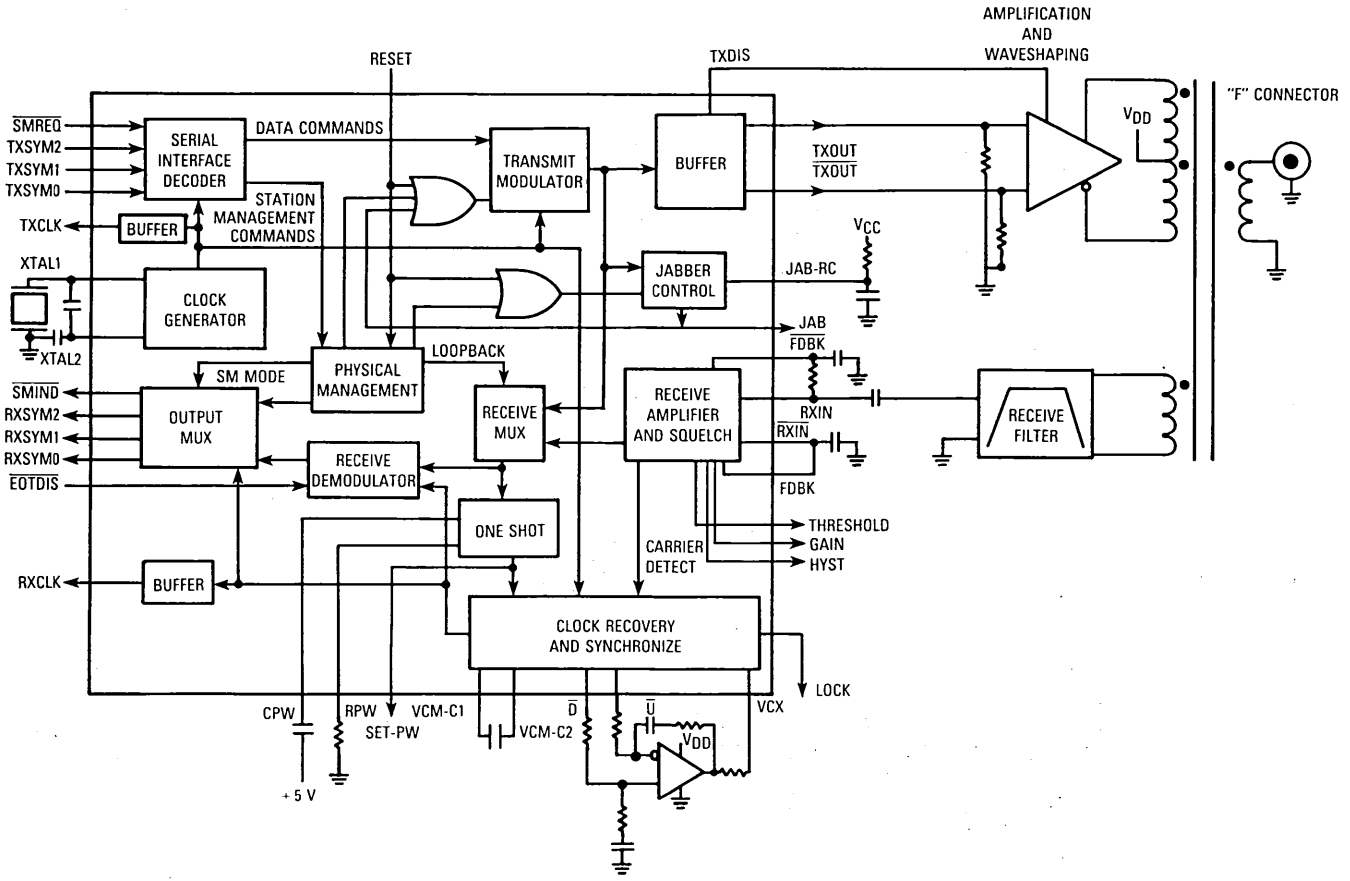
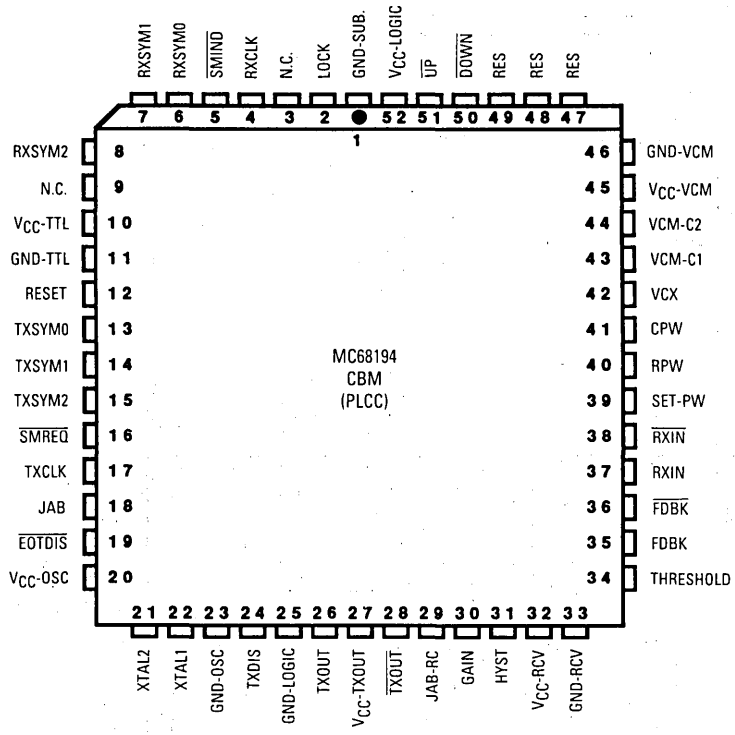


Figure 4. Functional Block Diagram

PIN ASSIGNMENTS



Motorola Order Number: MC68194 FN

Package Designation

Device

MC68605

Technical Summary

X.25 Protocol Controller (XPC)

The MC68605 X.25 Protocol Controller (XPC) is an intelligent HCMOS communications protocol controller that implements the 1984 International Telegraph and Telephone Consultative Committee (CCITT) X.25 Recommendation, data link access procedure (LAPB). It supports full duplex point-to-point serial communication at up to 10 megabits per second (MBPS) and relieves the host processor of managing the communications link by providing sequencing using HDLC framing, error control, retransmission based upon a cyclic redundancy check (CRC), and flow control using the receive not ready supervisory frame. The XPC directly supports the physical level interfaces (Recommendation X.21 physical level, X.21 *bis*, and V-series) and also provides an efficient interface to the packet level for information and control exchange. Key features of the XPC include:

- Fully Implements X.25 Recommendation LAPB Procedure by Independently Generating Link Level Commands and Responses
- Option to Implement X.75 Recommendation
- Optional Transparent Operation (Monitor Mode) where XPC Provides HDLC/SDLC Framing Functions for User Generated Frames
- Performs DMA Transfer of Information Frames To and From Memory Using Two On-Chip 22-Byte FIFOs
- Primary Communication Through Shared Memory Structures with a Powerful Command Set to Off-Load Data Link Management
- Flexible Rx/Tx Linked Memory Structures Minimize Host Intervention and Simplify Memory Management
- Basic (Modulo 8) and Extended (Modulo 128) Operation
- Automatic Comparison of the Programmable Local and Remote Addresses
- Detection of Programmable Timeout and Retries Limit Conditions
- 16- or 32-Bit CRC Generation and Checking
- Standard Modem Interface
- NRZ or NRZI Encoding/Decoding
- Vectored Interrupts and Status Reporting
- Built-In Diagnostics Provide Local Loopback and External Loopback Testing
- Up to 10 Mbps Synchronous Serial Data Rate
- 12.5 and 10 MHz System Clock Versions
- 8- and 16-Bit Data Bus Support
- 32-Bit Address Bus with Virtual Address Capability
- M68000 Family Asynchronous Bus Structure
- Programmable Byte Ordering of Data for Alternate Memory Organization Schemes

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

GENERAL DESCRIPTION

The XPC supports high-speed X.25 communications between host computers, between host computers and remote units, and between remote units. The XPC also supports a transparent operation mode which does not apply the LAPB procedure. Data is passed between the XPC and the host processor through shared memory structures. This permits a minimum command set for host processor/XPC communication. Additionally the XPC is a full M68000 bus master, providing on-chip DMA capability for management of memory tables and frame buffers. Since the XPC data bus interface is configurable, the XPC can handle both 8-bit and 16-bit data transfers.

When the X.25 mode is selected by the user, the XPC is configured as a combined station for full duplex point-to-point communication. The XPC supports a non-operational mode and two operational modes as defined by the LAPB procedure. The non-operational mode is asynchronous disconnect mode (ADM). In this balanced data link mode, the combined station is logically disconnected from the data link and is not permitted to transmit or accept information. Operational modes include asynchronous balanced mode (ABM) and asynchronous mode extended (ABME). A balanced data link allows a combined station to send a command or initiate a response frame transmission without receiving explicit permission from the other station. In ABM/ABME the XPC performs the following operations:

1) Transmission of a chain of information (I) frames when instructed by the host,
2) Transmission of supervisory (S) frames as defined by the X.25 LAPB Recommendation,
3) Transmission of unnumbered (U) commands as required or when instructed by the host, and
4) Transmission of unnumbered (U) responses as defined by the X.25 LAPB Recommendation.

When the transparent mode is selected, the XPC can be configured as a master, a slave or a combined station for full duplex operation. The XPC can support any HDLC/SDLC defined operational mode. All frames are user-generated and are transmitted only when instructed by the host.

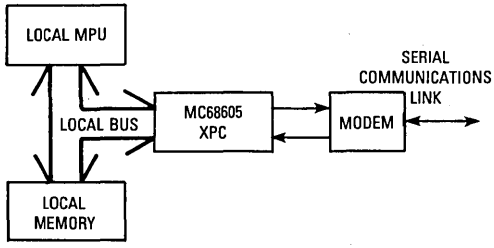


Figure 1. XPC System Configuration

INTERNAL REGISTERS

The XPC has four functional blocks including: serial, DMA, microcode controller, and register-file/ALU. Each of these sections contain user visible and nonvisible registers that define and control the operation of the XPC. A block diagram of the MC68605 is shown in Figure 2.

Because the XPC communicates with the host primarily through shared memory, a minimum number of host processor accessible registers are required. Registers in the XPC fall into two groups. One group is directly accessible by the user and the other group is indirectly accessible through the station table. The directly accessible registers include the command register, semaphore register, interrupt vector register, and data register. The complete register set is shown in Table 1.

SHARED MEMORY STRUCTURES

The host processor communicates with the XPC using three tables located in shared memory (Figure 3). The station table allows the host processor to initialize and update the XPC operating parameters and table pointers, and to receive status and error information. The transmit frame specificaton table queues frames to be transmitted by the XPC, and the receive frame specification table queues available receive buffers for the XPC to store received information frames. The XPC is given a pointer to the station table during initialization. The transmit frame specification table and receive frame specification table pointers are contained in the station table.

STATION TABLE

The station table format is shown in Table 2. The first 19 words of the station table are written by the host processor and are read by the XPC. This portion of the table contains the XPC operating information. The XPC accesses this table area as the result of a host processor command. The next 22 words of the table are written by the XPC and read by the host processor. Some of these entries are written by the XPC as the result of a command while other entries are updated by the XPC when a change occurs. When the XPC accesses the table as the result of a host processor command, it sets the semaphore register to hex 'FF' upon completion of the access. While the XPC is processing a command, the semaphore register is hex 'FE'.

TRANSMIT FRAME SPECIFICATION TABLE

The transmit frame specification table queues transmit frames for the XPC. These frames are stored in memory buffers located throughout memory. The transmit frame specification table contains a sequential list of transmit frame specification blocks. The transmit frame specification blocks describe the location of transmit buffers and provide information about the transmit queue. The transmit table pointer location in the station table points to the first transmit frame specification block. See Figure 4.

When the host processor instructs the XPC to load transmit table pointer, the XPC loads the transmit table

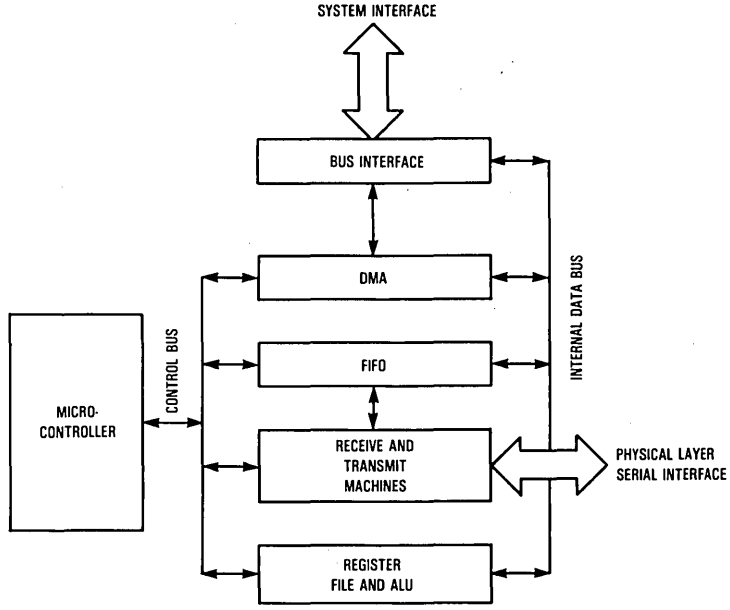


Figure 2. MC68605 Block Diagram

Table 1. XPC Register Set

Register	Description
Directly Accessible Registers	
Command	8-Bit Write Only
Semaphore	8-Bit Read Only
Interrupt Vector	8-Bit Write Only, Read on Host Processor Interrupt Acknowledge Cycle
Data	32-Bit Write Only

Indirectly Accessible Registers	
Register	**Mnemonic**
Station Table Pointer	STP
Station Table Function Code	STFC
Local Address	LA
Remote Address	RA
Hardware Configuration	HC
Station Configuration	SC
Option Bits	OB
Mode Descriptor	MD
Frame Reject Descriptor	FRD
Rx/Host Status	RHS
Tx/Link Status	TLS

Register	Mnemonic
V(S)	V(S)
V(R)	V(R)
Time Scale Divider	TSD
Retries Count	RC
Transmit Table Pointer	TTP
Transmit Table Function Code	TTFC
Transmit Buffer Pointer	TBP
Transmit Buffer Function Code	TBFC
Transmit Buffer Count	TBC
Receive Table Pointer	RTP
Receive Table Function Code	RTFC
Receive Buffer Pointer	RBP
Receive Buffer Function Code	RBFC
Receive Buffer Count	RBC
Time-Out Preset	TOP
Retries Limit	RL
Outstanding Frames Limit	OFL
Pad Time Select	PTS
Last Received N(R)	LRN

7

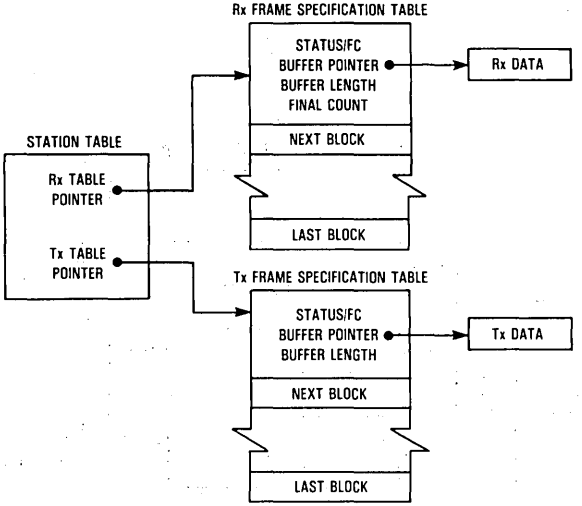


Figure 3. Shared Memory Tables

pointer and transmit table function code registers from the corresponding station table entries. The transmit table pointer register then has the address of the first transmit frame specification block. Before the transmission of each frame, the XPC accesses the current transmit frame specification block to load the transmit buffer function code, transmit buffer address, and the transmit buffer length into the corresponding internal registers. The XPC presents a transmit buffer address and function code to the system to load the information contained in the transmit buffer.

During transparent operation, the XPC accesses the next transmit frame specification block and transmits the corresponding frame buffer until the end of the transmit frame specification table is reached. The XPC updates its internal V(S) register after the transmission of each frame. When all frames have been transmitted, the XPC sets the IFAK bit (information frames acknowledged) in the Tx/link status register.

During X.25 operation, the XPC accesses the next transmit frame specification block and transmits the corresponding frame buffer according to the X.25 Recommendation until either the outstanding frames limit or the end of the transmit frame specification table is reached. The XPC updates its internal V(S) registers after the transmission of an information frame. The XPC monitors the N(R) of incoming frames until all transmitted frames have been acknowledged. After all frames have been acknowledged, the XPC sets the IFAK bit in the Tx/link status register.

RECEIVE FRAME SPECIFICATION TABLE

The receive frame specification table queues receive buffers for the XPC. These buffers are stored throughout memory. The receive frame specification table contains a sequential list of receive frame specification blocks. The receive frame specification blocks describe the location of the receive buffers and provide information about the queue. The receive table pointer in the station table points to the first receive frame specification block. See Figure 5.

When the host processor instructs the XPC to load receive table pointer, the XPC loads the receive table function code and receive table pointer registers from the corresponding station table entries. The receive table pointer register then contains the address of the first receive frame specification block. The XPC accesses the receive frame specification block to load the receive buffer function code, receive buffer address, and the receive buffer length into its internal registers. The XPC then presents the receive buffer address and function code to the system to store the received information field in the memory buffer. After reception of a frame, the XPC writes the number of unused bytes in the final count entry in the current receive frame specification block and updates its internal V(R) register. Next the XPC sets the RXI bit (receive information frame) in the Rx/host status register. The XPC accesses the next receive frame specification block to store incoming frames, until the end of the receive frame specification table is reached.

To decrease the possibility of a receive not ready condition due to a lack of available receive buffers, a method is provided for linking receive frame specification tables. When the EOT (end of table) bit is set in a receive frame specification block, the XPC inspects the link bit value. If the link bit is set, then the XPC loads the receive table pointer and FC registers from the corresponding station table locations. The XPC then sets the RTE (receive table ended) bit in the Rx/host status register and issues an

Table 2. Station Table Structure

Word	15 12	11 8	7 4	3 0	
0	Option Bits				
1	Time Out Preset				
2	Time Scale Divider		Pad Time Select		
3	Outstanding Frames Limit		Retries Limit		
4	Rx/Host Mask Bits				
5	Tx/Link Mask Bits				
6	Rx/Host Status Clear Bits				
7	Tx/Link Status Clear Bits				
8	0 0 0 0	0 0 0 0	Local Address		Host Processor Area Read by the XPC Written by Host Processor
9	0 0 0 0	0 0 0 0	Remote Address		
10	0 0 0 0	0 0 0 0	0 0 0 0	RTFC	
11	Receive Table Pointer — High Word				
12	Receive Table Pointer — Low Word				
13	0 0 0 0	0 0 0 0	0 0 0 0	TTFC	
14	Transmit Table Pointer — High Word				
15	Transmit Table Pointer — Low Word				
16	0 0 0 0	0 0 0 0	0 0 0 0	DAFC	
17	Dump Area Pointer — High Word				
18	Dump Area Pointer — Low Word				
19	Rx/Host Status				
20	Tx/Link Status				
21	Mode Descriptor		Frame Reject Descriptor		
22	VS		VR		
23	0 0 0 0	0 0 0 0	0 0 0 0	FUFC	
24	First Unacknowledged Pointer — High Word				
25	First Unacknowledged Pointer — Low Word				
26	0 0 0 0	0 0 0 0	0 0 0 0	TFC	
27	Transmit Pointer — High Word				
28	Transmit Pointer — Low Word				XPC Area Read by the Host Processor Written by the XPC
29	0 0 0 0	0 0 0 0	0 0 0 0	RFC	
30	Receive Pointer — High Word				
31	Receive Pointer — Low Word				
32	0 0 0 0	0 0 0 0	0 0 0 0	REFC	
33	Receive Bus/Address Error Pointer — High Word				
34	Receive Bus/Address Error Pointer — Low Word				
35	0 0 0 0	0 0 0 0	0 0 0 0	TEFC	
36	Transmit Bus/Address Error Pointer — High Word				
37	Transmit Bus/Address Error Pointer — Low Word				
38	Received FRMR Information Field — Word 1				
39	Received FRMR Information Field — Word 2				
40	Received FRMR Information Field — Word 3				

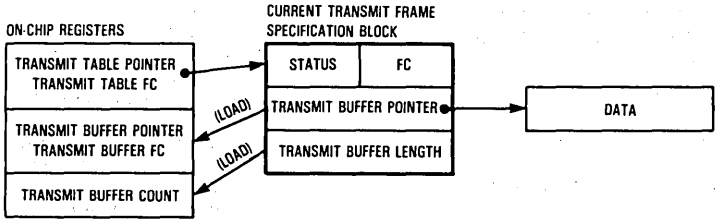


Figure 4. Transmit Frame Specification Table

7

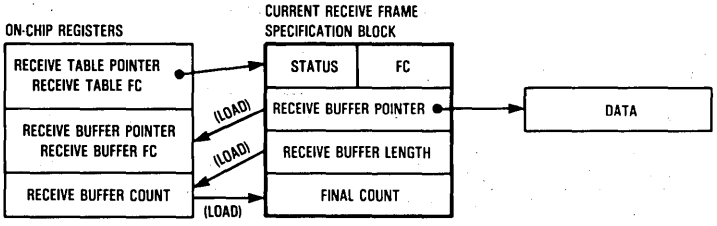


Figure 5. Receive Frame Sepcification Table

interrupt if enabled. The link operation can be used to implement a cyclical queue by using the original RTP and FC values in the station table. However, the user must read filled receive buffers expediently to ensure that the XPC does not overwrite the buffers with incoming frames.

COMMAND SET

The host processor issues commands to the XPC to perform various functions by writing to the XPC command register. There are 23 commands that fall in the following four categories:

1) Initialization
2) Table Handling
3) Link Handling
4) Test/Diagnostics

INITIALIZATION COMMANDS

Initialization commands configure the XPC for operation after a hardware or software reset. The four initialization commands specify various system attributes, communication protocol options, and the location of the station table in memory.

7

Reset

The reset command and hardware reset causes the following actions:

- Reset the Receive Channel and Isolate RxD
- Reset the Transmit Channel, Negate $\overline{\text{RTS}}$, and Transmit Ones
- Immediately Relinquish the System Bus
- Set the Interrupt Vector Register to '0F' Hex
- Disable Transmit and Receive Memory Buffers
- Clear all Rx/Host and Tx/Link Status Bits
- Clear all Hardware and Station Configuration Bits
- Clear all Option Bits
- Clear all Mode Descriptor and Frame Reject Descriptor Bits
- Zero Station Table Pointer and Station Table FC Registers
- Zero Transmit Table Pointer and Transmit Table FC Registers
- Zero Receive Table Pointer and Receive Table FC Registers
- Zero Remote Address and Local Address Registers
- Zero V(R), V(S), and Last Received N(R) Registers
- Zero Preset Values and Retries Count Register

Set Station Configuration

The set station configuration command specifies protocol parameters. The command has the following format.

7	6	5	4	3	2	1	0
1	0	1	0	ECRC	0	ECNT	0

ECRC — Extended CRC
- 0 16-Bit CRC
  CRC CCITT ($X^{16}+X^{12}+X^5+1$)
- 1 32-Bit CRC
  ($X^{32}+X^{26}+X^{23}+X^{22}+X^{16}+X^{12}+11$
  $+X^{10}+X^8+X^7+X^5+X^4+X^2+X^1+1$)

ECNT — Extended Control
- 0 Basic Control Field Format (Modulo 8)
- 1 Extended Control Field Format (Modulo 128)

Set Hardware Configuration

The set hardware configuration command defines the data decoding/encoding scheme, DMA burst control, data organization in memory and data bus size. The format of the command is shown below.

7	6	5	4	3	2	1	0
1	1	0	NRZI	BRSC	0	DORGM	BUSW

NRZI — Non-Returned to Zero Invert
- 0 NRZ Decoding/Encoding
- 1 NRZI Decoding/Encoding

BRSC — Burst Control
- 0 DMA Burst is Unlimited
- 1 DMA Burst is Limited to Eight Successive Memory Cycles

DORGM — Data Organization in Memory for a 16-Bit Data Bus System
- 0 Data in Memory is Organized with High-Order Byte in Lower Memory Address (Motorola and IBM Convention)
- 1 Data in Memory is Organized with Low-Order Byte in Lower Memory Address (DEC and Intel Convention)

(This capability is available only for I frame buffers and not for parameters or tables.)

BUSW — Bus Width
- 0 8-Bit Data Bus
- 1 16-Bit Data Bus

Load Function Code

The load FC command writes the function code value in the data register into the station table FC register. This command is issued after the host processor has written the function code to the data register.

Load Station Table Pointer

The load station table pointer command writes the station table address from the data register into the station table pointer register. This command is issued after the host processor has written the station table pointer to the data register.

TABLE HANDLING COMMANDS

The 11 table handling commands cause the XPC to access the station table, transmit frame specification table, or receive frame specification table.

Load Option Bits

The load option bits command loads the option set from the station table into the option bits register.

F	E	D	C	B	A	9	8
0	0	0	0	0	0	0	X.75

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	CRCNOA

X.75 — X.75 Option
- 0 X.25 Operation
- 1 X.75 Operation

CRCNOA — CRC Bypass Option
- 0 Non-octet aligned frames or frames with a CRC error are not accepted
- 1 Non-octet aligned frames or frames with a CRC error are accepted

Load Addresses

The load addresses command loads the local and remote addresses from the station table into the internal XPC registers. After these registers are loaded, the XPC is ready to establish the link. The XPC monitors the receive line and transmits continuous flags.

Load Preset Values

The load preset values command loads the time out preset value, time scale divider, pad time select, outstanding frames limit, and retries limit from the station table into the respective XPC internal registers.

Load Transmit Table Pointer

The load transmit table pointer command loads the transmit table pointer and the transmit table FC from the station table into the corresponding XPC registers and enables the transmission of a chain of information frames.

Continue Transmit

The continue transmit command (hex '95') is used to extend the transmit queue after adding entries to the transmit frame specification table. The user should set the EOT bit in the transmit status location of the last added entry and then clear the EOT bit at the previous end of table. Finally, the user should instruct the XPC to "continue transmit". This command is useful in the case where the XPC has already detected the previous EOT and will not read a new table entry. Instead, it is waiting for all transmitted frames to be acknowledged, and during this period, it will not accept a new load transmit table pointer command.

Load Receive Table Pointer

The load receive table pointer command loads the receive table pointer and the receive table FC from the station table into the corresponding XPC registers and enables the reception of information frames.

Load Station Parameters

This command combines the load option bits, load preset values, and load addresses commands.

Update Status

The update status command allows the host to request current XPC status information.

Clear Tx/Link Status

The clear Tx/link status command clears the status bits in the Tx/link status register as specified by the Tx/link status clear bits in the station table.

Clear Rx/Host Status

The clear Rx/link status command clears the Rx/host status register as specified by the Rx/host status clear bits in the station table.

Clear Status

The clear status command clears both the Tx/link and Rx/host status bits in the respective XPC registers as specified by the Tx/link status clear bits and the Rx/host status clear bits in the station table.

Dump Parameters

The dump parameters command writes the following XPC parameters into the corresponding status table locations in the order given: Rx/host status, Tx/link status, mode descriptor, frame reject descriptor, V(R), V(S), first unacknowledged transmit block FC and pointer, next transmit block FC and pointer, and next receive block FC and pointer.

LINK HANDLING COMMANDS

The two link handling commands cause the XPC to set the link to a new operation mode and to automatically

handle communication on both channels according to the predefined configuration and option bits.

Start Link

The start link command initiates the link setup procedure.

Stop Link

The stop link command initiates the link disconnect procedure.

TEST/DIAGNOSTICS

The five commands in the text/diagnostics category test the XPC circuit and run diagnostics on the link.

Dump Registers

The dump registers command writes the XPC registers to a user specified dump area in external memory.

DMA Transfer

The DMA transfer command tests the handling of parallel data. The XPC reads the data from a transmit memory buffer and writes it to a receive memory buffer. The XPC transfers data from the transmit buffer to the receive buffer via the data register without using the internal transmit or receive FIFOs. The serial link is not affected by this operation.

Serial Loopback

The serial loopback command tests the handling of parallel and serial data. The XPC reads data from the transmit memory buffer into the transmit FIFO. The data is then serialized and shifted internally into the receive FIFO and onto the TxD line. Finally the data is stored in the receive memory buffer. $\overline{RTS}$ is not active during serial loopback.

Monitor

The monitor command allows the XPC to check the communication channel by reading/writing the entire frame from/to memory. The monitor command may be used to perform an external loopback test of the system or to implement any HDLC/SDLC operation mode where all frames are user generated. The XPC transmits and/or receives multiple information frames using the transmit and receive frame specification tables until an end monitor command is received.

In each transmit buffer the user places the address, control, and data (if any) fields. The XPC only provides framing, zero insertion, and CRC for each frame. On the receive side, the XPC strips off flags, handles zero deletion, and writes the address control and data fields into the receive buffer. The received CRC is also appended to the end of each memory buffer and is verified by the XPC.

End Monitor

The end monitor command terminates the monitor command.

XPC IMPLEMENTATION OF LAPB PROTOCOL

INITIALIZATION PROCEDURE

The XPC enters the initialization procedure as the result of a hardware or software reset. During this initialization, the station table address and function code (FC), system configuration information, and the XPC interrupt vector are loaded by the XPC under the direction of the host, as shown in the sample program below. Internal XPC registers directly accessed during the initialization procedure are the command register (CR), data register (DR), interrupt vector register (IV), and semaphore register (SR).

RESET

```
  Repeat: Read Semaphore Register Until it is 'FF'
Write CR: Set Hardware Configuration
  Repeat: Read Semaphore Register Until it is 'FF'
Write CR: Set Station Configuration
  Repeat: Read Semaphore Register Until it is 'FF'
Write DR: 4-Bit Function Code Value for Station Table
            Access
Write CR: Load Function Code
  Repeat: Read Semaphore Register Until it is 'FF'
Write DR: 32-Bit Address of Station Table
Write CR: Load Station Table Pointer (STP)
  Repeat: Read Semaphore Register Until it is 'FF'
 Write IV: Load Interrupt Vector
Write CR: Load Station Table Parameters
  Repeat: Read Semaphore Register Until it is 'FF'
```

NOTE

The XPC will not come out of hardware or software reset without the system clock and the transmit clock. The transmit clock is used to initialize the serial section of the chip.

INFORMATION FRAME TRANSMISSION

After the XPC enters asynchronous balanced mode (ABM) or asynchronous balanced mode extended (ABME), the host processor can instruct the XPC to transmit a chain of information frames by issuing the load transmit table pointer command. In response the XPC loads the transmit table pointer and the transmit table function code from the station table into its internal registers. Next the XPC loads the first transmit buffer pointer, transmit buffer function code, and transmit buffer count from the transmit frame specification table into the corresponding XPC registers. Now the XPC is ready to build the first frame.

The remote address is copied from the remote address register into the XPC transmit FIFO. Next the control field is generated and placed in the FIFO. The information field pointed to by the transmit buffer pointer register is then read from the memory buffer into the transmit FIFO until the transmit buffer count is satisfied. A frame check sequence is attached to complete the frame. Zero insertion is performed throughout the transmission. After frame transmission the send state variable, V(S), is updated and timer T1 is started (if it is not already running) to determine when the programmed time period permitted for a reply to be received has elapsed.

This transmission sequence repeats for each frame until the end of the transmit chain is reached, or until the outstanding frames limit is reached. The XPC continues to transmit any available information frames even when the XPC receiver is in the busy condition. The XPC prematurely terminates frame transmission if a link command interrupts the information frame transmission or an error condition arises.

Transmission begins when six bytes are present in the transmit FIFO. Transmission can begin when less than six bytes are present in the FIFO if the entire frame is less than six bytes in length. Between frames the XPC transmits the user selected number of pad flags. Additional flags are transmitted if the required number of bytes are not present in the transmit FIFO. While transmitting an information frame, the XPC requests the bus when there are at least six empty bytes in the transmit FIFO.

INFORMATION FRAME RECEPTION

The host processor enables information reception by instructing the XPC to load receive table pointer. The XPC will load the receive table pointer and function code into its internal registers. Next the receive buffer pointer and function code, and the receive buffer count are loaded into the corresponding XPC registers. The XPC is now ready to receive information frames.

The address field of an incoming I frame is compared to the local address register and the remote address register. If the address does not match the local or remote address, the frame is ignored. If the address field matches the remote address, a frame reject (FRMR) is transmitted and the W (invalid or unimplemented control field) bit of the frame reject descriptor register (FRD) is set. If the address field matches the local address, the frame is accepted by the XPC and the received N(R) acknowledges previously transmitted I frames.

Next, the send sequence number N(S) of the incoming frame is compared to the XPC internal receive state variable V(R). If the frame is in sequence, then the information field is transferred through the receive FIFO to the receive memory buffer. Out-of-sequence frames are rejected.

Lastly, the XPC performs a CRC check on the incoming information frame. If an error-free frame is received, the XPC acknowledges the frame reception with a supervisory frame (receive ready RR or receive not ready RNR) or with an updated receive sequence number N(R) in the next information frame.

Zero deletion is performed throughout the reception process. The XPC requests the bus when there are six bytes in the receive FIFO. Only a single frame can reside in the receive FIFO. Frames are received in sequence as long as memory buffers are available.

XPC STATE DIAGRAM

The XPC state diagram (Figure 6 which is located on the last page of this document) is a detailed description of the XPC implementation of the LAPB procedure. The state diagram defines the various XPC states based on command frames received (no errors), response frames received (no errors), and miscellaneous inputs received. For example, referring to Figure 6, if the command received was an RR with the poll bit set to one while in the remote station busy condition (state 9), then the XPC responds with an RR with the final bit set to a one and changes to information transfer (state 5).

XPC TRANSPARENT MODE OF OPERATION

The XPC transparent mode of operation can be used to implement a variety of bit oriented protocols. The following paragraphs describe the XPC transparent mode of operation.

INITIALIZATION PROCEDURE

The XPC enters the initialization procedure as the result of a hardware or software reset. During initialization, the station table address and function code, system configuration information, and the XPC interrupt vector should be loaded by the XPC under the direction of the host, as shown in the sample program below. Note that the XPC will not come out of hardware or software reset without the system clock and the transmit clock. The transmit clock is used to initialize the serial section of the chip.

ENTERING TRANSPARENT OPERATION

Transparent operation is entered when the host issues the monitor command. After the monitor command, the XPC asserts $\overline{RTS}$, transmits flags, and monitors RxD. Since handshaking between nodes is not possible before the monitor command is executed, the host processor at each node must issue the monitor command in order for the two nodes to communicate.

FRAME TRANSMISSION

After the monitor command is issued, the XPC begins transmission of frames only after receiving a load transmit table pointer command from the host. All frames are user-generated and may contain user-provided address, control, and/or data fields. After the host issues the load transmit table pointer command, the XPC loads the transmit table pointer and the transmit table funtion code from the station table into its internal registers. Next, the XPC loads the first trasmit buffer pointer, transmit buffer function code, and transmit buffer count from the transmit frame specification table into the corresponding XPC registers. Now the XPC is ready to transmit the first frame.

The frame pointed to by the transmit buffer pointer register is read from the memory buffer into the transmit FIFO until the transmit buffer count is satisfied. An XPC-generated frame check sequence is then attached to complete the frame. After each frame transmission, the internal V(S) register is incremented without regard to the frame type. This transmission sequence repeats for each frame until the end of the transmit chain is reached. Zero insertion is performed throughout the transmission process.

In transparent operation, the XPC transmits frames until the end of the transmit specification table is reached. After the last frame is transmitted, the XPC sets the IFAK

(information frames acknowledged) bit in the Tx/link status register to indicate the end of the transmit table. The XPC does not analyze any incoming frames for acknowledgements or link control information during transparent operation. The only errors reported in the Tx/link status register are address error, bus error, clear-to-send lost, and underrun.

Transmission begins when six bytes are present in the transmit FIFO. Transmission can begin when less than six bytes are present in the FIFO, if the entire frame is less than six bytes in length. Between frames, the XPC transmits the user-selected number of pad flags. Additional pad flags are transmitted if the required number of bytes are not present in the transmit FIFO for transmission to begin. While transmitting a frame, the XPC requests the bus when there are at least six empty bytes in the transmit FIFO.

FRAME RECEPTION

The host processor enables frame reception by instructing the XPC to load receive table pointer. The XPC then loads the receive table pointer and function code into its internal registers. Next, the receive buffer pointer and function code, and the receive buffer count are loaded into the corresponding XPC registers. The XPC is now ready to receive frames.

The XPC does not analyze the address and control fields of incoming frames, but does perform a CRC check on incoming frames. After the flags are stripped off, the entire frame including CRC is written into the current receive buffer and the RXI (received information frame) bit is set in the Rx/host status register. If a frame is received with a CRC error, the XPC sets the E bit in that frame's receive specification block. After a frame is received, the XPC increments V(R) without regard to frame type. Zero deletion is performed throughout the reception process.

In transparent operation, the XPC continues to receive frames until the end of the receive specification table is reached. The XPC then sets the RTE (receive table ended) bit in the Rx/host status register.

The XPC requests the bus when there are six bytes in the receive FIFO. Only a single frame can reside in the receive FIFO. Frames are received in sequence as long as memory buffers are available.

SIGNAL DESCRIPTION

The input and output signals can be functionally organized into the groups shown in Figure 7.

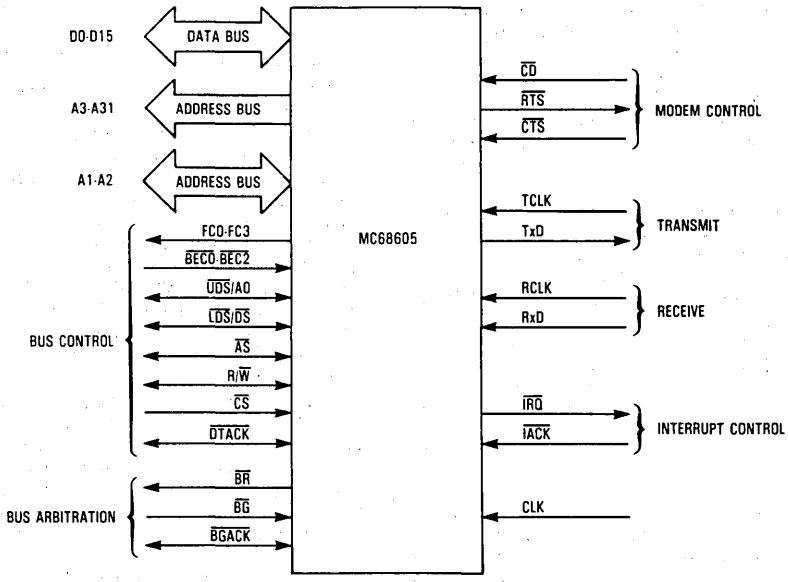


Figure 7. MC68605 Signals

ELECTRICAL SPECIFICATIONS

MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Supply Voltage	V_{DD}	−0.3 to +7.0	V
Input Voltage	V_{in}	−0.3 to +7.0	V
Operating Temperature Range MC68605 MC68605I	T_A	 0 to 70 0 to 85	°C
Storage Temperature Range	T_{stg}	−55 to +150	°C

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

THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Unit
Thermal Resistance for PGA	θ_{JA}	33	°C/W

$T_J = T_A + (P_D \bullet \theta_{JA})$
$P_D = (V_{DD} \bullet I_{DD}) + P_{I/O}$
where:
$P_{I/O}$ is the power dissipation on pins (user determined) which can be neglected in most cases.
For $T_A = 70°C$ and $P_D = 0.55$ W @ 12.5 MHz
$T_J = 88°C$

POWER CONSIDERATIONS

The average chip-junction temperature, T_J, in °C can be obtained from:

$$T_J = T_A + (P_D \cdot \theta_{JA}) \quad (1)$$

where:

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

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

If $P_{I/O}$ is neglected, an approximate relationship between P_D and T_J is:

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

Solving equations (1) and (2) for K gives:

$$K = P_D \cdot (T_A + 273°C) + \theta_{JA} \cdot P_D^2 \quad (3)$$

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.

7

DC ELECTRICAL CHARACTERISTICS

All specifications are valid under the following conditions: V_{DD} = 4.75 V to 5.25 V, V_{SS} = 0 V, $T_A = T_L$ to T_H and 130 pF total capacitance on output pins.

Characteristic	Symbol	Min	Max	Unit
Input High Voltage (Except System Clock)	V_{IH}	2.0	V_{DD}	V
Input Low Voltge (Except System Clock)	V_{IL}	$V_{SS}-0.3$	0.8	V
Input High Voltage (System Clock)	V_{CIH}	2.4	V_{DD}	V
Input Low Voltage (System Clock)	V_{CIL}	$V_{SS}-0.3$	0.5	V
Input Leakage Current	I_{in}	—	20	μA
Input Capacitance	C_{in}	—	13	pF
Three-State Leakage Current (2.4/0.5 V)	I_{TSI}	—	20	μA
Open-Drain Leakage Current (2.4 V)	I_{OD}	—	20	μA
Output High Voltage (I_{OH} = 400 μA)	V_{OH}	2.4	—	V
Output Low Voltage	V_{OL}			V
(I_{OL} = 3.2 mA) A1-A31, FC0-FC3, $\overline{RTS}$, $\overline{TXD}$, $\overline{UDS}$/A0 as A0		—	0.5	
(I_{OL} = 5.3 mA) D0-D15, $\overline{AS}$, $\overline{LDS}$, $\overline{UDS}$/A0 as $\overline{UDS}$, $\overline{DTACK}$, $\overline{BGACK}$, R/$\overline{W}$		—	0.5	
(I_{OL} = 8.9 mA) $\overline{BR}$, $\overline{IRQ}$		—	0.5	
Power Dissipation @ 10 MHz, 0°C	P_D	—	0.50	W
@ 12.5 MHz, 0°C		—	0.55	
@ 16.67 MHz, 0°C		—	0.65	

AC ELECTRICAL CHARACTERISTICS

High and low outputs are measured at 2.0 V minimum and 0.8 V maximum respectively. High and low inputs are driven to 2.4 V and 0.5 V respectively for AC test purposes. However, input specifications are still measured from 2.0 V to 0.8 V. All specifications are valid under the following conditions: V_{DD} = 4.75 V to 5.25 V, V_{SS} = 0 V, $T_A = T_L$ to T_H, output load = 130 pF, and output current as specified in **DC ELECTRICAL CHARACTERISTICS**. See Figures 8-19.

Num.	Characteristic	10 MHz		12.5 MHz		16.67		Unit
		Min	Max	Min	Max	Min	Max	
1	Asynchronous Input Setup Time	20	—	20	—	10	—	ns
2	$\overline{UDS}$, $\overline{LDS}$ Inactive to $\overline{CS}$, $\overline{IACK}$ Inactive	—	100	—	80	—	60	ns
3	CLK Low (On Which $\overline{UDS}$ or $\overline{LDS}$ and $\overline{CS}$ or $\overline{IACK}$ are Recognized) to Data-Out Valid (see Note 5)	— —	1/2 +150	— —	1/2 +120	— —	1/2 +90	Clk. Per. ns
4	$\overline{CS}$ or $\overline{IACK}$ High to Data-Out High-Impedance	—	60	—	50	—	35	ns
5	$\overline{LDS}$/$\overline{DS}$ High to Data-Out Hold Time (see Note 6)	0	—	0	—	0	—	ns
6	$\overline{IACK}$ or $\overline{CS}$ Low to $\overline{DTACK}$ High (Driving Three-State $\overline{DTACK}$ High)	—	80	—	70	—	60	ns
7	CLK Low (On Which $\overline{UDS}$ or $\overline{LDS}$ and $\overline{CS}$ or $\overline{IACK}$ are Recognized) to $\overline{DTACK}$ Low (see Note 5)	— —	2 +90	— —	2 +80	— —	2 +50	Clk. Per. ns
8	CLK Low to $\overline{DTACK}$ Low	—	90	—	80	—	50	ns
9	Data-Out Valid to $\overline{DTACK}$ Low	20	—	20	—	20	—	ns
10	$\overline{DTACK}$ Low to $\overline{UDS}$, $\overline{LDS}$, $\overline{CS}$, $\overline{IACK}$ High (Earliest)	100	—	80	—	60	—	ns
11	$\overline{CS}$ or $\overline{IACK}$ or Data Strobes (The Earliest) High to $\overline{DTACK}$ High (see Note 7)	—	60	—	50	—	40	ns
12	$\overline{DTACK}$ High to $\overline{DTACK}$ High Impedance (At End of Bus Cycle)	—	50	—	50	—	40	ns
13	$\overline{UDS}$, $\overline{LDS}$ Inactive Time	1	—	1	—	1	—	Clk. Per.
14	$\overline{CS}$, $\overline{IACK}$ Inactive Time	0	—	0	—	0	—	ns
15	A1-A2 Valid to $\overline{UDS}$, $\overline{LDS}$, $\overline{CS}$ (The Latest One) Low (Write)	30	—	20	—	20	—	ns
16	$\overline{DTACK}$ Low to Data and A1-A2 Hold Time	100	—	80	—	60	—	ns

AC ELECTRICAL CHARACTERISTICS (Continued)

Num.	Characteristic	10 MHz		12.5 MHz		16.67 MHz		Unit
		Min	Max	Min	Max	Min	Max	
17	$\overline{UDS}$ or $\overline{LDS}$, $\overline{CS}$ or $\overline{IACK}$ (The Latest One) Low to Data-In Valid	—	80	—	70	—	60	ns
18	R/$\overline{W}$ Valid to $\overline{UDS}$ or $\overline{LDS}$, $\overline{CS}$ or $\overline{IACK}$ (The Latest One) Low	20	—	20	—	10	—	ns
19	$\overline{UDS}$, $\overline{LDS}$ High to R/$\overline{W}$ High	0	—	0	—	0	—	ns
20	CLK High to $\overline{IRQ}$ Low	—	100	—	80	—	60	ns
21	Reserved							
22	Reserved							
23	CLK High to $\overline{BR}$ Low	—	60	—	55	—	40	ns
24	CLK High to $\overline{BR}$ High Impedance	—	55	—	50	—	40	ns
25	$\overline{BGACK}$ Low to $\overline{BR}$ High Impedance	20	—	20	—	10	—	ns
26	$\overline{BG}$ Active/Inactive to CLK Low Setup Time	20	—	20	—	10	—	ns
27	CLK Low to $\overline{BGACK}$ Low	—	60	—	55	—	40	ns
28	CLK High to $\overline{BGACK}$ High Impedance	—	45	—	40	—	40	ns
29	$\overline{AS}$ and $\overline{BGACK}$ High (The Latest One) to $\overline{BGACK}$ Low (When $\overline{BG}$ is Previously Asserted)	2 +20	3 +80	2 +20	3 +70	2 +10	3 +50	Clk. Per. ns
30	$\overline{BG}$ Low to $\overline{BGACK}$ Low (No Other Bus Master)	2 +20	3 +80	2 +20	3 +70	2 +10	3 +50	Clk. Per. ns
31	$\overline{BR}$ High Impedance to $\overline{BG}$ High	0	—	0	—	0	—	ns
32	Clock on which $\overline{BGACK}$ Low to Clock on which $\overline{AS}$ Low	1.5	1.5	1.5	1.5	1.5	1.5	Clk. Per.
33	Clock Low to $\overline{BGACK}$ High	—	55	—	50	—	40	ns
34	CLK on which $\overline{BR}$ Low to CLK on which $\overline{BGACK}$ Low (Assuming that $\overline{BG}$ is Active and $\overline{BGACK}$ and $\overline{AS}$ are Inactive for at Least 2 CLK Periods)	1.5	1.5	1.5	1.5	1.5	1.5	Clk. Per.
35	CLK on which $\overline{AS}$ is High to CLK on which $\overline{BGACK}$ is High	—	1	—	1	—	1	Clk. Per.
36	CLK High to Address Valid	—	100	—	80	—	60	ns
37	CLK High to Address/FC High Impedance	—	70	—	60	—	50	ns
38	CLK High to FC Valid	—	60	—	55	—	50	ns
39	Address Valid to $\overline{AS}$ Valid	20	—	15	—	10	—	ns
40	CLK High to $\overline{AS}$, $\overline{UDS}$, $\overline{LDS}$ Low	—	50	—	40	—	40	ns
41	CLK to $\overline{AS}$, $\overline{UDS}$, $\overline{LDS}$ High	—	55	—	50	—	40	ns
42	$\overline{AS}$ High to Address/FC Invalid	20	—	10	—	0	—	ns
43	CLK High to $\overline{AS}$, $\overline{UDS}$, $\overline{LDS}$ High Impedance	—	70	—	60	—	45	ns
44	CLK to R/$\overline{W}$ High (see Note 4)	—	55	—	50	—	45	ns
45	CLK Low to R/$\overline{W}$ High Impedance	—	70	—	60	—	45	ns
46	$\overline{UDS}$, $\overline{LDS}$ High to Data-In Invalid	0	—	0	—	0	—	ns
47	$\overline{AS}$, $\overline{UDS}$, $\overline{LDS}$ High to $\overline{DTACK}$ High (Earliest of $\overline{AS}$, $\overline{UDS}$, or $\overline{LDS}$)	0	100	0	90	0	60	ns
48	Data-In to CLK Low Setup Time Required when $\overline{DTACK}$ Satisfies (1) (see Note 1)	10	—	10	—	5	—	ns
49	$\overline{DTACK}$ Low to Data-In Valid Required when $\overline{DTACK}$ does not Satisfy (1) (see Note 2)	—	65	—	50	—	40	ns
50	CLK High to R/$\overline{W}$ Low	—	60	—	55	—	40	ns
51	$\overline{AS}$ Low to Data-Out Valid (Write)	—	90	—	80	—	60	ns

7

AC ELECTRICAL CHARACTERISTICS (Concluded)

Num.	Characteristic	10 MHz		12.5 MHz		16.67 MHz		Unit
		Min	Max	Min	Max	Min	Max	
52	CLK Low to Data-Out Valid	—	55	—	55	—	40	ns
53	Data-Out Valid to $\overline{UDS}$, $\overline{LDS}$ Low	20	—	15	—	10	—	ns
54	$\overline{UDS}$, $\overline{LDS}$ High to Data-Out Invalid	20	—	15	—	0	—	ns
55	CLK High to Data-Out Hold Time	0	100	0	100	0	60	ns
56	No Exception to $\overline{BR}$ ($\overline{DTACK}$ Active)	1.5 +20	2.5 +80	1.5 +20	2.5 +70	1.5 +10	2.5 +50	Clk. Per. ns
57	$\overline{DTACK}$ Low to Asynchronous Exception Active Required when $\overline{DTACK}$ Does Not Satisfy (1) (see Note 2)	—	55	—	35	—	30	ns
58	Exception Active to CLK Low Setup Time Synchronous Input ("Late Exception") Required when $\overline{DTACK}$ Satisfies (1) (see Note 1)	45	—	45	—	20	—	ns
59	Exception Active to CLK Low Setup Time Asynchronous Input (Required when $\overline{DTACK}$ is Absent) (see Note 3)	20	—	20	—	10	—	ns
60	$\overline{AS}$, $\overline{UDS}$, $\overline{LDS}$ High to Exception Inactive	0	—	0	—	0	—	ns
61	Exception Inactive to CLK Low Setup Time (for Identification of No Exception)	20	—	20	—	—	10	ns
62	No Exception to $\overline{BR}$ ($\overline{DTACK}$ Inactive)	2.5 +20	3.5 +80	2.5 +20	3.5 +70	2.5 +10	3.5 +50	Clk. Per. ns
63	$\overline{RESET}$ (on $\overline{BEC0}$-$\overline{BEC2}$) Width	10	—	10	—	10	—	Clk. Per.
64	CLK Frequency	4	10	4	12.5	4	16.67	MHz
65	CLK Period	100	250	80	250	60	250	ns
66	CLK Width High (see Note 8)	45	125	35	125	25	125	ns
67	CLK Rise/Fall Time (see Note 8)	—	10	—	5	—	5	ns
68	CLK Width Low (see Note 8)	45	125	35	125	25	125	ns
69	RCLK, TCLK Frequency	0	10	0	12.5	0	16.67	MHz
70	RxD to RCLK High Setup Time	35	—	35	—	25	—	ns
71	RCLK High to RxD Hold Time	5	—	5	—	5	—	ns
72	RCLK, TCLK Rise/Fall Time	—	10	—	10	—	5	ns
73	RCLK, TCLK Width Low	45	—	35	—	25	—	ns
74	RCLK, TCLK Width High	45	—	35	—	25	—	ns
75	RCLK, TCLK Period	100	—	80	—	60	—	ns
76	TCLK Low to TxD Valid	10	80	10	60	10	45	ns
77	Reserved							
78	CD Low to RCLK Low Setup Time	25	—	25	—	25	—	ns
79	CTS Low to TCLK High Setup Time	25	—	25	—	25	—	ns

NOTES:

1. If $\overline{DTACK}$ satisfies the asynchronous setup time (1), then (48) is required for the data-in setup time and (58) for the synchronous exception setup time. Erroneous behavior may occur if (58) is not satisfied.
2. If $\overline{DTACK}$ does not satisfy (1), then (49) is required for data-in and (57) for the exception. Erroneous behaviour may occur if (57) is not satisfied.
3. Active exception when $\overline{DTACK}$ is absent must satisfy the asynchronous setup time (59). Erroneous behavior may occur if (59) is not satisfied.
4. R/$\overline{W}$ rises on the end of a write cycle (i.e., on the phase following S7). If the XPC relinquishes the bus, then R/$\overline{W}$ is three-stated one phase later. When the XPC takes the bus, R/$\overline{W}$ is three-stated until S1 and changes on that phase.
5. Data (3) and $\overline{DTACK}$ (7) will be timed from the earliest clock on which $\overline{CS}$ and either data strobe are recognized during an MPU cycle. Data (3) and $\overline{DTACK}$ will be timed from the earliest clock on which $\overline{IACK}$ and either data strobe are recognized during an $\overline{IACK}$ cycle.
6. If $\overline{CS}$ or $\overline{IACK}$ is negated before $\overline{UDS}$/$\overline{LDS}$, the data bus will be three-stated (4), possibly before $\overline{UDS}$/$\overline{LDS}$ negation.

AC ELECTRICAL CHARACTERISTIC (Concluded)

NOTES:

7. If an 8-bit bus is used, only $\overline{LDS}$ need be considered. If a 16-bit bus is used, both $\overline{UDS}$ and $\overline{LDS}$ must negate to apply to this specification.
8. The clock signal during test has 5 ns of rise time and 5 ns of fall time. For system implementations that have less clock rise and fall time, the clock pulse minimum should be commensurately wider such that:
   1. System (TCL + (TCR + TCF) ÷ 2) ≥ (minimum t_{cyc}) ÷ 2
   2. System (TCH + (TCR + TCF) ÷ 2) ≥ (minimum t_{cyc}) ÷ 2

   where
   TCL is CLK width low (see electrical specification #68)
   TCH is CLK width high (see electrical specification #66)
   TCF is CLK rise time
   t_{cyc} is CLK period (see electrical specification #65)
   TCR is CLK fall time

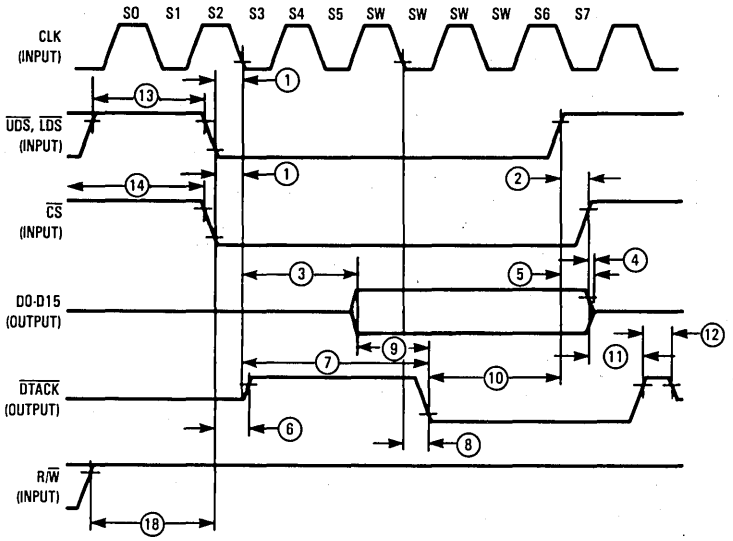


Figure 8. Host Processor Read Cycle Timing Diagram

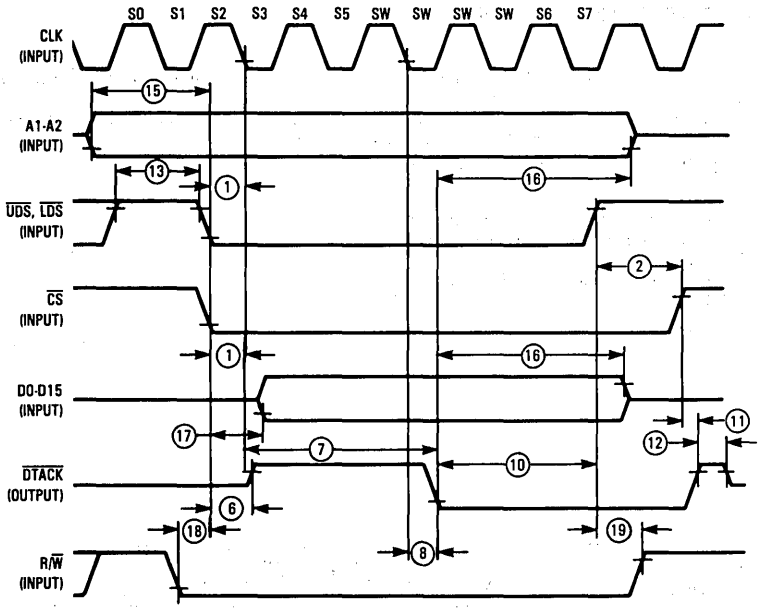


Figure 9. Host Processor Write Cycle Timing Diagram

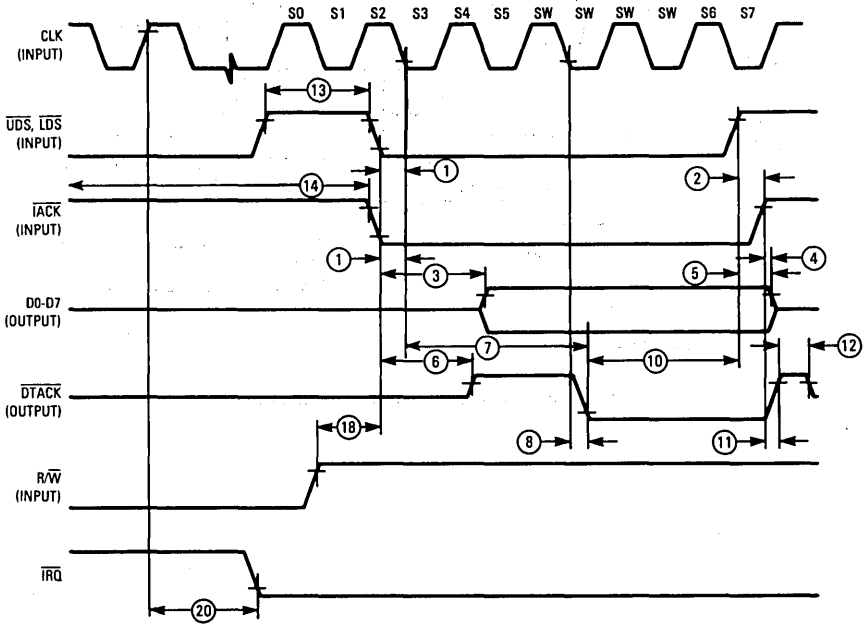


Figure 10. Interrupt Acknowledge Cycle Timing Diagram

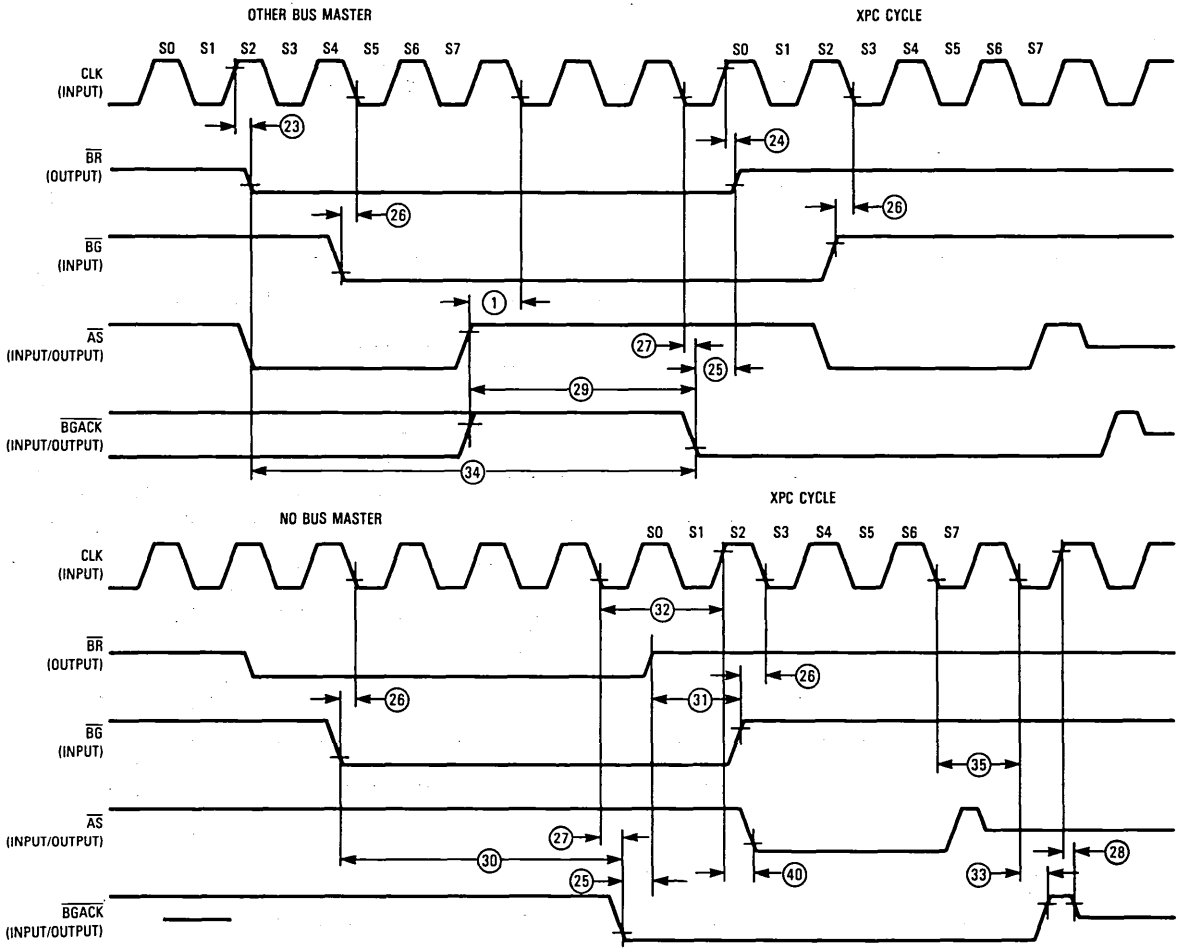
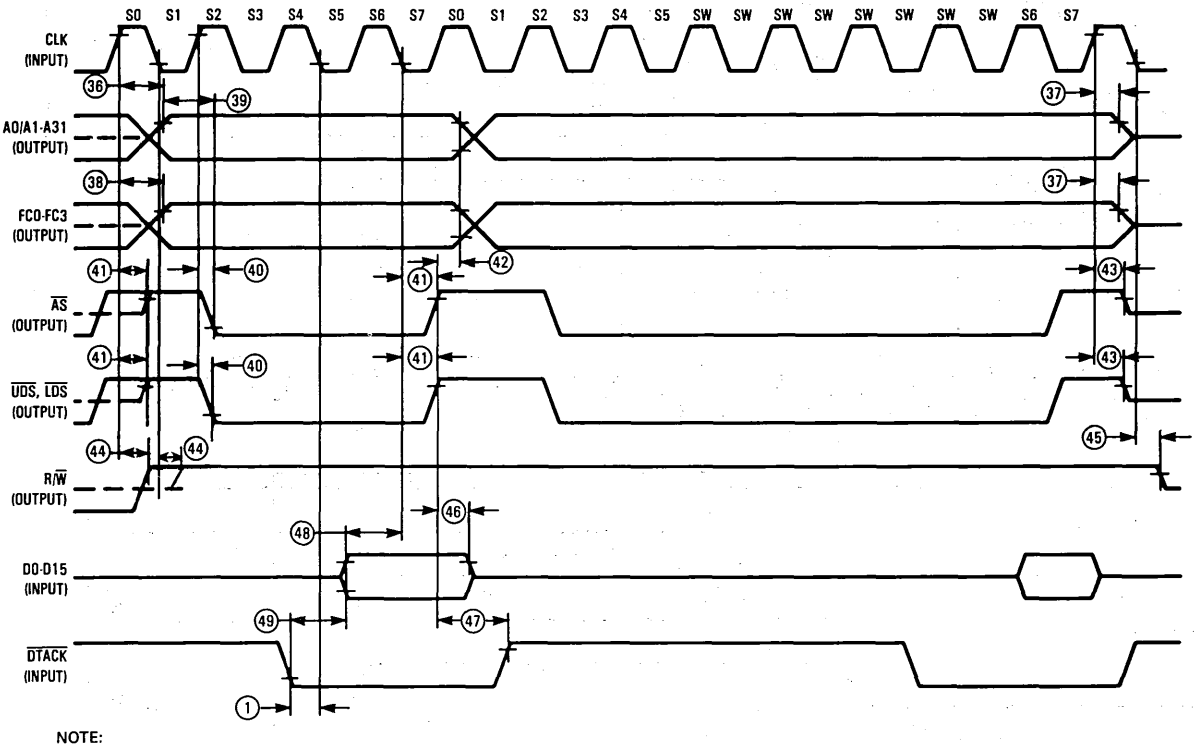


Figure 11. Bus Arbitration Timing Diagram



NOTE:
The solid lines assume that the communication controller was bus master on the last cycle. The dotted lines assume that there was a different bus master.

Figure 12. Read Cycle and Slow Read Cycle Timing Diagram

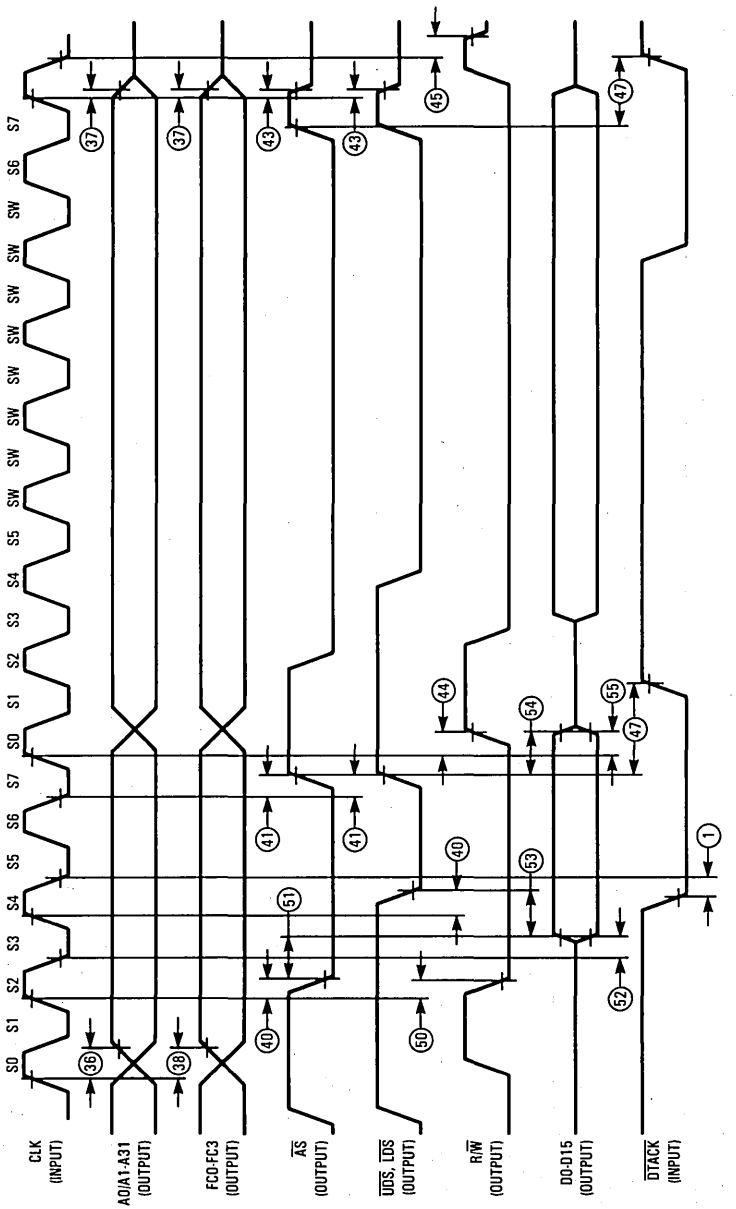
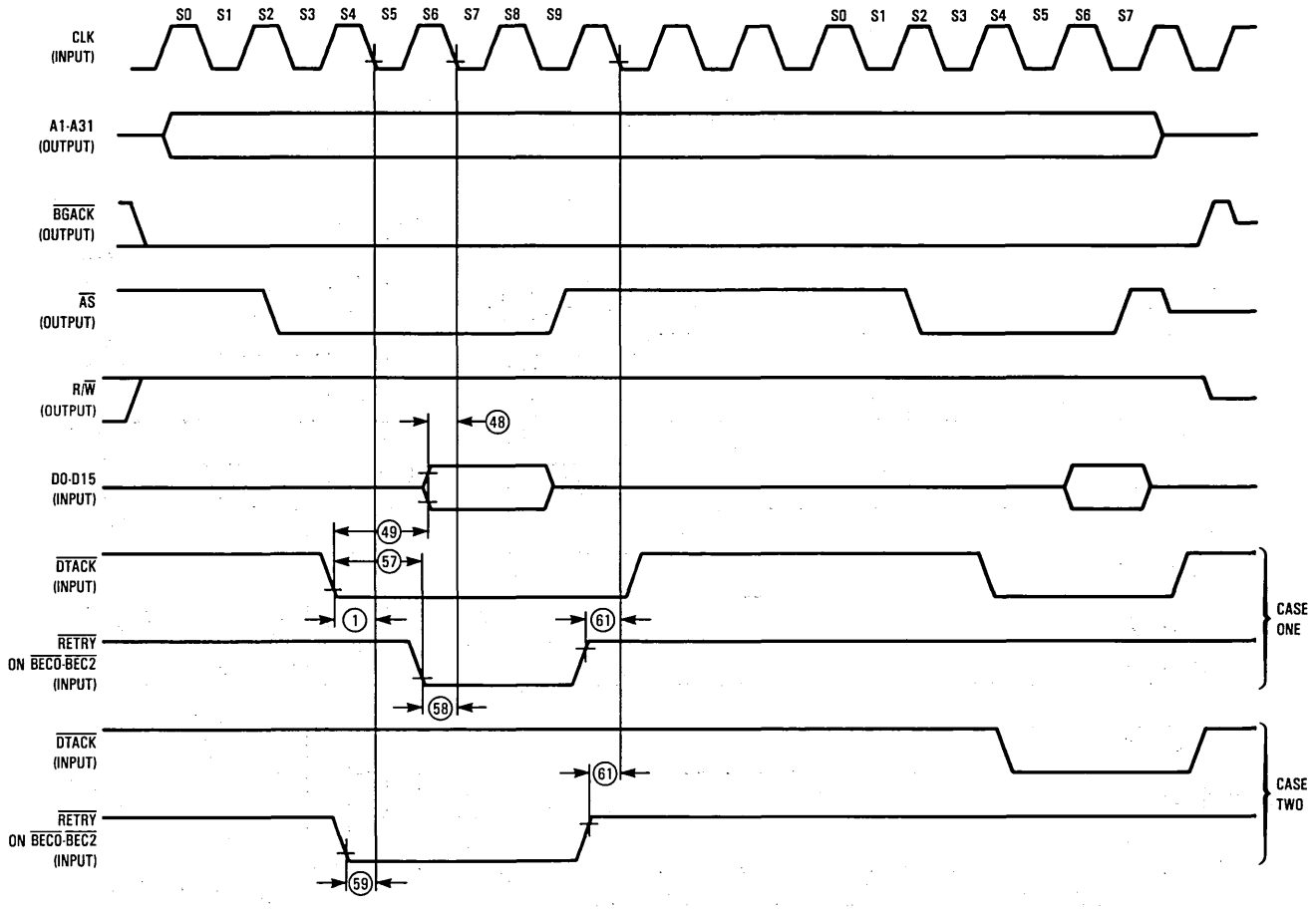


Figure 13. Write Cycle Timing Diagram

7



CASE 1: If DTACK satisfies (1), then (48) and (58) are required; if DTACK is active but does not satisfy (1), then (49) and (57) are required.
CASE 2: If DTACK is not active, then (59) is required for the exception active setup time. Parameter (61) is always required for the exception inactive setup time.

Figure 14. XPC Read Cycle with Retry Timing Diagram

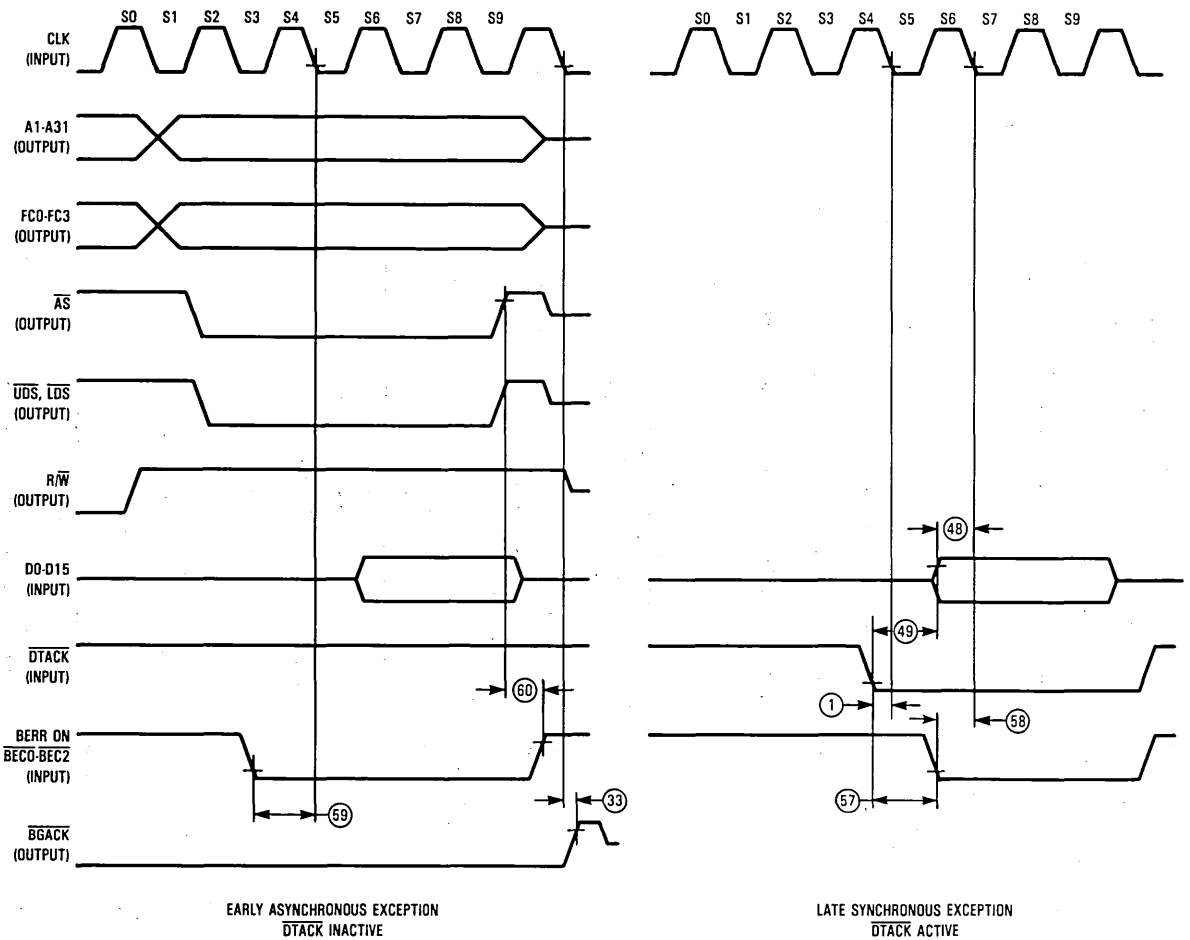
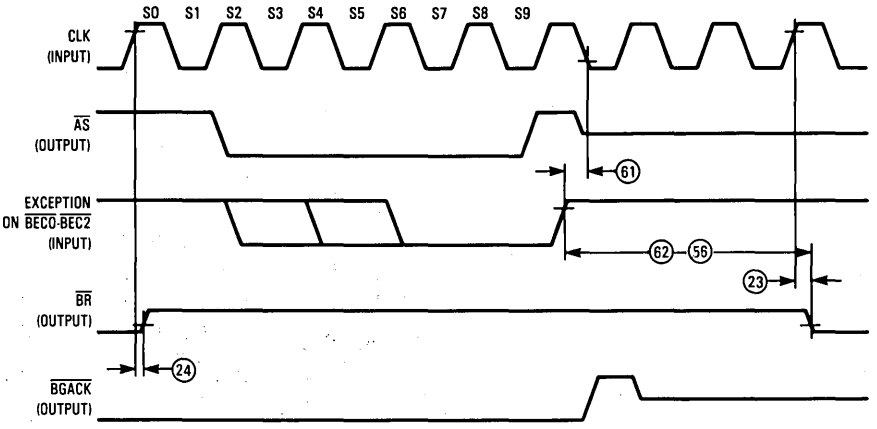


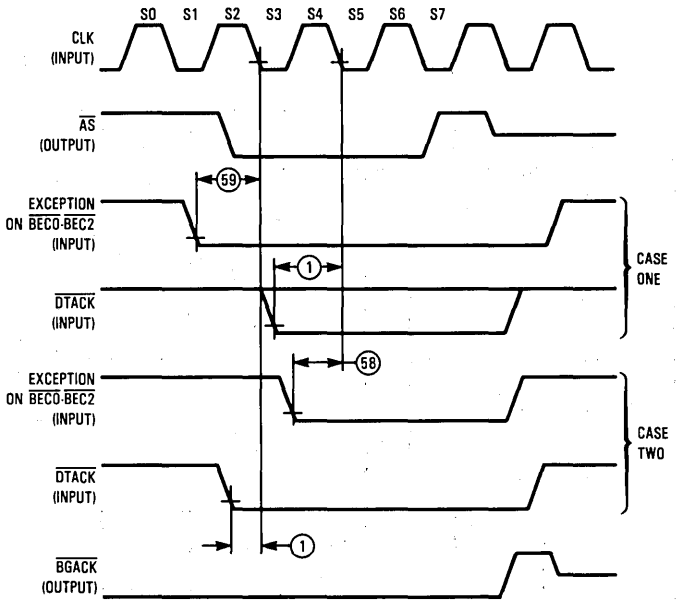
Figure 15. Read Cycle with Bus Error Timing Diagram

7



The above occurs when the XPC requires the bus cycle after a previous exception.

Figure 16. BR After Previous Exception Timing Diagram



Two alternatives of DTACK and exception. Case one has DTACK occur after exception and case two has exception occur after DTACK. Note that a HALT cycle can be terminated only by DTACK.

Figure 17. Short Exception Cycle Timing Diagram

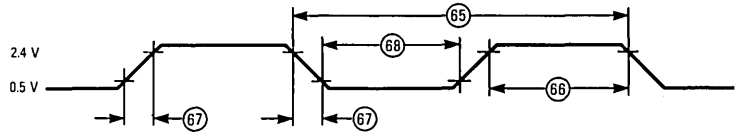


Figure 18. Clock (CLK) Timing Diagram

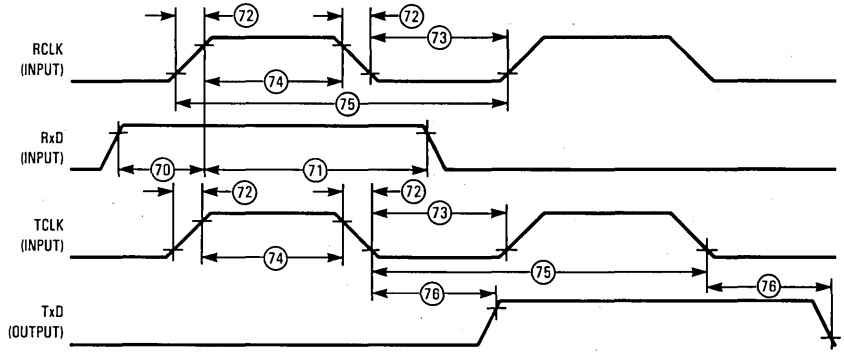


Figure 19. XPC Serial Data RxD, TxD, and Serial Clocks (RCLK, TCLK) Timing Diagram

7

STATE	I FRAME WITH POLL	I FRAME W/O POLL	RR WITH POLL	RR W/O POLL	REJ WITH POLL	REJ W/O POLL	RNR WITH POLL	RNR W/O POLL	SABM WITH OR W/O POLL	DISC. WITH OR W/O POLL	RR WITH FINAL	RR W/O FINAL	REJ WITH FINAL	REJ W/O FINAL	RNR WITH FINAL
S1 DISCONNECTED	DM, F=1	—	DM, F=1	—	DM, F=1	—	DM, F=1	—	UA, F=P TO S5	DM, F=P	—	—	—	—	—
S2 LINK SETUP	—	—	—	—	—	—	—	—	UA, F=P	DM, F=P TO S1	—	—	—	—	—
S3 FRAME REJECT	FRMR, F=1	FRMR, F=0	FRMR, F=1	FRMR, F=0	FRMR, F=1	FRMR, F=0	FRMR, F=1	FRMR, F=0	UA, F=P TO S5	UA, F=P TO S1	—	—	—	—	—
S4 DISCONNECT REQUEST	—	—	—	—	—	—	—	—	DM, F=P TO S1	UA, F=P	—	—	—	—	—
S5 INFORMATION TRANSFER	RR, F=1	**	RR, F=1	**	RR, F=1	**	RR, F=1 TO S9	RR, F=0 TO S9	UA, F=P	UA, F=P TO S1	(UNXF) SABM TO S2	***	(UNXF) SABM TO S2	***	(UNXF) SABM TO S2
S6 REJ FRAME SENT	RR, F=1 TO S5	** TO S5	RR, F=1	**	RR, F=1	**	RR, F=1 TO S14	RR, F=0 TO S14	UA, F=P TO S5	UA, F=P TO S1	(UNXF) SABM TO S2	***	(UNXF) SABM TO S2	***	(UNXF) SABM TO S2
S7 WAITING ACKNOWLEDGE-MENT	RR, F=1	RR, F=0	RR, F=1	RR, F=0	RR, F=1	RR, F=0	RR, F=1 TO S12	RR, F=0 TO S12	UA, F=P TO S5	UA, F=P TO S1	*** TO S5	—	*** TO S5	—	TO S9
S8 STATION BUSY	RNR, F=1	RNR, F=0	RNR, F=1	*N	RNR, F=1	*N	RNR, F=1 TO S10	RNR, F=0 TO S10	UA, F=P	UA, F=P TO S1	(UNXF) SABM TO S2	***	(UNXF) SABM TO S2	***	(UNXF) SABM TO S2
S9 REMOTE STATION BUSY	RR, F=1	RR, F=0	RR, F=1 TO S5	** TO S5	RR, F=1 TO S5	** TO S5	RR, F=1	RR, F=0	UA, F=P TO S5	UA, F=P TO S1	(UNXF) SABM TO S2	*** TO S5	(UNXF) SABM TO S2	*** TO S5	(UNXF) SABM TO S2
S10 BOTH STATIONS BUSY	RNR, F=1	RNR, F=0	RNR, F=1 TO S8	*N TO S8	RNR, F=1 TO S8	*N TO S8	RNR, F=1	RNR, F=0	UA, F=P TO S8	UA, F=P TO S1	(UNXF) SABM TO S2	*** TO S8	(UNXF) SABM TO S2	*** TO S8	(UNXF) SABM TO S2
S11 WAITING ACK AND STATION BUSY	RNR, F=1	RNR, F=0	RNR, F=1	RNR, F=0	RNR, F=1	RNR, F=0	RNR, F=1 TO S13	RNR, F=0 TO S13	UA, F=P TO S8	UA, F=P TO S1	*** TO S8	—	*** TO S8	—	TO S10
S12 WAITING ACK AND REMOTE STATION BUSY	RR, F=1	RR, F=0	RR, F=1 TO S7	RR, F=0 TO S7	RR, F=1 TO S7	RR, F=0 TO S7	RR, F=1	RR, F=0	UA, F=P TO S5	UA, F=P TO S1	*** TO S5	— TO S7	*** TO S5	— TO S7	— TO S9
S13 WAITING ACK AND BOTH STATIONS BUSY	RNR, F=1	RNR, F=0	RNR, F=1 TO S11	RNR, F=0 TO S11	RNR, F=1 TO S11	RNR, F=0 TO S11	RNR, F=1	RNR, F=0	UA, F=P TO S8	UA, F=P TO S1	*** TO S8	— TO S11	*** TO S8	— TO S11	— TO S10
S14 REJ FRAME SENT AND REMOTE STATION BUSY	RR, F=1 TO S9	RR, F=0 TO S9	RR, F=1 TO S6	** TO S6	RR, F=1 TO S6	** TO S6	RR, F=1	RR, F=0	UA, F=P TO S5	UA, F=P TO S1	(UNXF) SABM TO S2	*** TO S6	(UNXF) SABM TO S2	*** TO S6	(UNXF) SABM TO S2

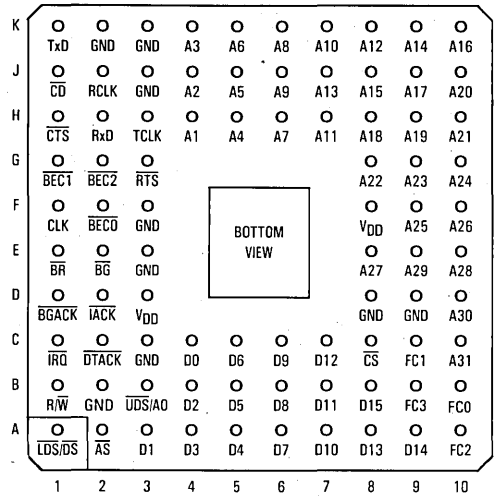
** If I available then Tx I frame else Tx RR, F=0
*** If I available then Tx I frame else do nothing
*N If I available then Tx I frame else Tx RNR, F=0
**N If P=1 then Tx RNR, F=1 else if I available then Tx I frame else Tx RNR, F=0
*FR If P=1 then Tx FRMR, F=1 else if P=0 then Tx FRMR, F=0 else do nothing
*DM If P=1 then Tx DM, F=1 else do nothing
*J If no REJ FRAME is outstanding then transmit REJ, F=P else if P=1 then Tx RR, F=1 else do nothing
**J If the I field of a correctly received frame has been discarded (due to the busy condition) then Tx REJ, F=0 else Tx RR, F=0
— Do nothing
X This event never occurs in this state
UNXF Unexpected final bit

Figure 6. XPC State Diagram

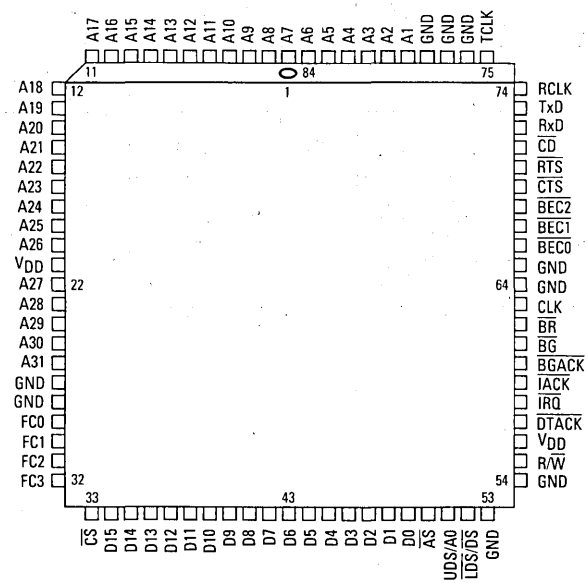
RNR W/O FINAL	UA WITH OR W/O FINAL	DM WITH FINAL	DM W/O FINAL	FRMR WITH OR W/O FINAL	LOCAL START COMMAND	LOCAL STOP COMMAND	STATION BECOMES BUSY	BUSY CONDITION CLEAR	T1 EXPIRES	N2=T1 IS EXCEEDED	NS SEQUENCE ERROR	INVALID NR RECEIVED	UNRECOGNIZED FRAME RECEIVED
—	—	SABM TO S2	SABM TO S2	—	SABM TO S2	DISC TO S4	X	—	X	X	*DM	*DM	*DM
—	TO S5	TO S1	—	—	X	X	X	—	SABM	TO S1	—	—	—
—	—	SABM TO S2	SABM TO S2	SABM TO S2	SABM TO S2	DISC TO S4	X	—	FRMR	SABM TO S2	*FR	*FR	*FR
—	TO S1	TO S1	—	—	X	X	X	—	DISC	TO S1	—	—	—
TO S9	SABM TO S2	SABM TO S2	SABM TO S2	SABM TO S2	SABM TO S2	DISC TO S4	RNR, F=P TO S8	X	RR, P=1 TO S7	SABM TO S2	*J TO S6	FRMR(Z) TO S3	FRMR(W) TO S3
TO S14	SABM TO S2	SABM TO S2	SABM TO S2	SABM TO S2	SABM TO S2	DISC TO S4	RNR, F=P TO S8	X	RR, P=1 TO S7	SABM TO S2	IF P=1 Tx RR, F=1	FRMR(Z) TO S3	FRMR(W) TO S3
TO S12	SABM TO S2	SABM TO S2	SABM TO S2	SABM TO S2	SABM TO S2	DISC TO S4	RNR, F=P TO S11	X	RR, P=1	SABM TO S2	*J	FRMR(Z) TO S3	FRMR(W) TO S3
TO S10	SABM TO S2	SABM TO S2	SABM TO S2	SABM TO S2	SABM TO S2	DISC TO S4	X	**J TO S5	RNR, P=1 TO S11	SABM TO S2	RNR, F=P	FRMR(Z) TO S3	FRMR(W) TO S3
—	SABM TO S2	SABM TO S2	SABM TO S2	SABM TO S2	SABM TO S2	DISC TO S4	RNR, F=P TO S10	X	RR, P=1 TO S12	SABM TO S2	*J TO S14	FRMR(Z) TO S3	FRMR(W) TO S3
—	SABM TO S2	SABM TO S2	SABM TO S2	SABM TO S2	SABM TO S2	DISC TO S4	X	**J TO S9	RNR, P=1 TO S13	SABM TO S2	RNR, F=P	FRMR(Z) TO S3	FRMR(W) TO S3
TO S13	SABM TO S2	SABM TO S2	SABM TO S2	SABM TO S2	SABM TO S2	DISC TO S4	X	**J TO S7	RNR, P=1	SABM TO S2	RNR, F=P	FRMR(Z) TO S3	FRMR(W) TO S3
—	SABM TO S2	SABM TO S2	SABM TO S2	SABM TO S2	SABM TO S2	DISC TO S4	RNR, F=P TO S13	X	RR, P=1	SABM TO S2	*J	FRMR(Z) TO S3	FRMR(W) TO S3
—	SABM TO S2	SABM TO S2	SABM TO S2	SABM TO S2	SABM TO S2	DISC TO S4	X	**J TO S12	RNR, P=1	SABM TO S2	RNR, F=P	FRMR(Z) TO S3	FRMR(W) TO S3
—	SABM TO S2	SABM TO S2	SABM TO S2	SABM TO S2	SABM TO S2	DISC TO S4	RNR, F=P TO S10	X	RR, P=1 TO S12	SABM TO S2	IF P=1 Tx RR, F=1	FRMR(Z) TO S3	FRMR(W) TO S3

PIN ASSIGNMENT

Pin Grid Array



Plastic-Leaded Chip Carrier



7

MC68606

Technical Summary

Multi-Link LAPD (MLAPD) Protocol Controller

The Motorola MC68606 Multi-Link LAPD (MLAPD) protocol controler is an integrated circuit implementing the link access procedure (LAPD) protocol. LAPD is the proposed protocol for use at the link layer (ISO-Layer 2) for both signalling and data transfer applications in Integrated Services Digital Network (ISDN) configurations. The LAPD protocol is specified in CCITT Recommendation Q.920/Q.921.

Current product implementations of this link level protocol are accomplished with firmware. This significantly loads the local processor, and presents limitations to both the maximum potential throughput of data and to the number of logical links which may be supported by such packet data interfaces. A generic view of where the MLAPD device can be used to interconnect a diversity of data endpoints in a high speed packet switch network is shown in Figure 1. The data links illustrated could differ functionally and provide data rates in the range of 64 kbps to 2.048 Mbps.

The MLAPD functions as an intelligent peripheral device to a central processing unit (CPU) in a microcomputer system. An on-chip DMA controller transfers data packets to and from a buffer memory. A microcoded buffer management scheme queues packets during transmission and reception. All link management duties are handled by the MLAPD device to maximize the bandwidth available for CPU operation and to increase the throughput for packet data transfer. This VLSI implementation provides a cost effective solution, while encouraging a universal implementation of the LAPD protocol. Key features of the MLAPD device include:

- Full Implementation of CCITT Recommendation Q.920/Q.921 Link Access Procedure (LAPD) with Independent Generation of Commands and Responses for Each Logical Link
- Control of up to 8192 Logical Links Using a Memory-Based Architecture, Wherein the Protocol Controller and the Supervising Microprocessor Communicate Through Shared Memory
- Reliable, Interleaved Data Transfers for Multiple Logical Links with the Following Protocol Actions:
  - — HDLC framing with zero-bit insertion/deletion for a serial bit stream; or optional parallel assist mode where zero insertion/deletion is disabled and frame delineation is provided by external pins for supporting a parallel interface to the physical level.
  - — Error control using a 16-bit CRC.
  - — Flow control to prevent data from accumulating at the receiving end faster than the data can be processed.
- Termination of a Non-Channelized Serial Bit Stream with an Aggregate Rate in Excess of 2.048 Mbps or Optional Memory-to-Memory Operation Allowing the MLAPD to Act as a LAPD Controller Independent of the System's Physical Level Characteristics

7

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

- Supports User Prioritization of I Frame and XID/UI Frame Transmission
- Supports Optional Receive and Transmit of a User-Defined, Non-Standard LAPD Unnumbered (U) Frame
- On-Chip, Content-Addressable Memory (CAM) Provides Address Translation for Up to 16 Logical Links. When Supporting More Than 16 Logical Links, Translation is Provided Via a Translation Table in Shared Memory
- Provides Error/Statistical Counters and Maskable Interrupts to the Level 3 Process
- Supports Optional Non-Protocol Mode on a Per-Logical Link Basis, Which Allows the Host to Receive and Transmit Frames Without Application of the LAPD Procedures by the MLAPD
- Supports Promiscuous Receive Mode in Which the MLAPD Receives All Frames From the Line and Transfers the Entire Frame to Memory
- System Interface Tailored for Different Microprocessor System Implementations:
  - Motorola M68000 and Intel iAPX86 Family Bus Interface Options
  - 8- and 16-Bit Data Bus Support
  - Direct Addressing of 16 Mbytes of System Memory
- Available in 12.5 MHz and 16.67 MHz System Clock Versions
- 84 Lead PGA and PLCC J-Lead Surface Mount Packages
- 1.5 Micron HCMOS Technology

GENERAL DESCRIPTION

SUBSYSTEM ENVIRONMENT

The MLAPD protocol controller provides simultaneous control of a maximum of 8192 logical links, while under the overall supervision of a microprocessor. Refer to Figure 2. The host can be any 8- or 16-bit microprocessor that supports general multimaster bus capability, operating with either the Motorola M68000 family or the Intel iAPX86 family bus interface definition. The desired MLAPD bus operation mode is determined by the level on the Motorola/Intel Mode pin.

The MLAPD serial interface provides HDLC-type framing functions for the bit stream entering/exiting the full duplex serial interface. NRZ data encoding/decoding is implemented. The MLAPD may also be optionally configured to interface to a physical level which implements a parallel interface, such as a backplane in a switching controller or host computer system. In this mode zero insertion/deletion is disabled and flag sequences are not generated. Instead, the $\overline{RTS}$ and $\overline{CTS}$ signals function as $\overline{TSTART}$ and $\overline{RSTART}$, respectively, to delineate a frame for the transmit and receive data lines.

Shared Memory Control Components

The communication between the microprocessor and the protocol controller is established through command and data block structures stored in shared memory. The shared memory

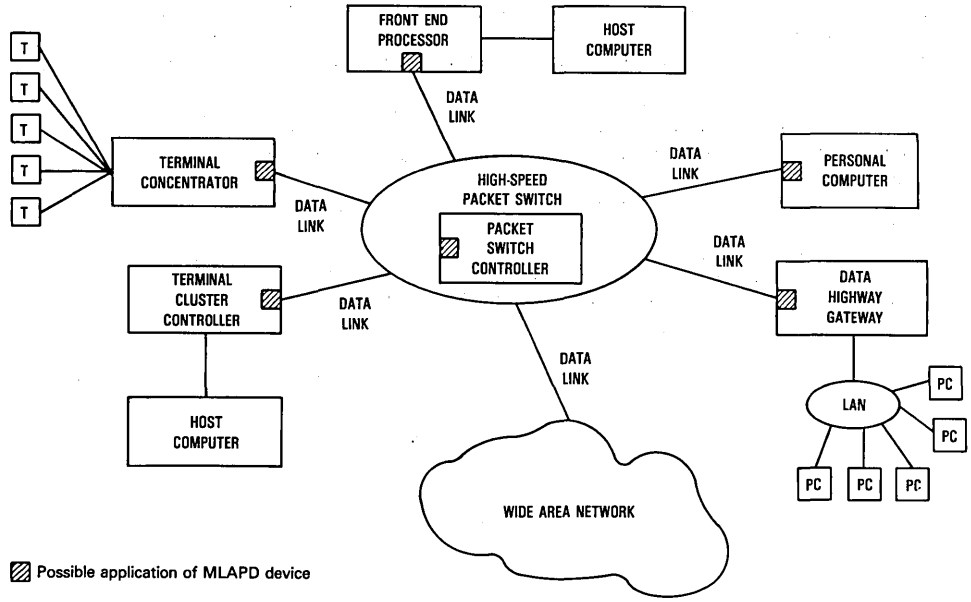


Figure 1. High Speed Data Switching

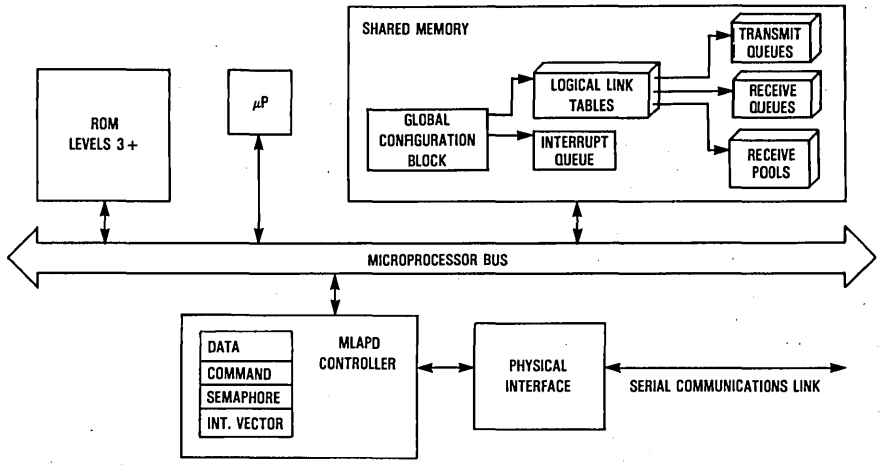


Figure 2. MLAPD System Environment

structures include the Global Configuration Block (GCB), Logical Link Tables (LLTs), receive and transmit data structures, and an Interrupt Queue. In addition to these structures, there are also four memory tables required for the implementation of the LAPD procedure. These tables include the Match Table, LLID-LLT Table, Timer Table, and the Level 2 Queue. Figure 3 shows a concise summary of the functions performed by these blocks.

BLOCK TYPE	PURPOSE
GLOBAL CONFIGURATION BLOCK	CONTAINS POINTERS TO OTHER MEMORY TABLES. CONTAINS GLOBAL INFORMATION FOR ALL LOGICAL LINKS. CONTAINS USER SYSTEM AND LINK OPTIONS.
LOGICAL LINK TABLES	CONTAINS SPECIFIC LINK CONTROL PARAMETERS FOR THE GIVEN LINK. ALSO CONTAINS POINTERS TO THE RECEIVE/TRANSMIT QUEUES. CONTAINS WORKING PROTOCOL STATUS FOR THE LINK.
RECEIVE AND TRANSMIT QUEUES	CONTAINS LINKED LIST OF RECEIVE AND TRANSMIT FRAME DESCRIPTORS FOR I, UI, AND XID FRAMES.
RECEIVE POOLS	LINKED LIST OF AVAILABLE RECEIVE FRAME DESCRIPTORS WITH ASSOCIATED DATA BUFFERS.
INTERRUPT QUEUE	CONTAINS TIME SEQUENTIAL LIST OF INTERRUPT STATUS FOR EACH INTERRUPT EVENT.
MATCH TABLE	CONTAINS TRANSLATION FROM DATA LINK CONNECTION IDENTIFIER (DLCI) TO LOGICAL LINK IDENTIFIER (LLID) FOR INCOMING FRAMES (EXPANDED OPERATION MODE).
LLID TO LLT TABLE	TRANSLATES LLID INTO THE ADDRESS OF THE CORRESPONDING LOGICAL LINK TABLE.
LEVEL 2 TRANSMIT QUEUE	CONTAINS LEVEL 2 SUPERVISORY AND UNNUMBERED FRAMES GENERATED BY THE MLAPD.
TIMER TABLE	IMPLEMENTS T200/T203 FOR ALL LOGICAL LINKS THAT ARE IN MULTIPLE FRAME ESTABLISHED MODE OF OPERATION.

Figure 3. Shared Memory Guide

During initialization, the host processor specifies protocol and system parameters and the addresses of the various memory tables via the Global Configuration Block. Then the host programs the MLAPD controller with the address of the Global Configuration Block and commands the MLAPD to initialize itself by loading its internal registers from the shared memory tables. After this point, the host and the MLAPD communicate primarily via software flags and an Interrupt Queue in shared memory.

Command Structure Overview

The host issues a command to the MLAPD by first writing the command arguments (if any) into the command parameter fields in the Global Configuration Block in shared memory. Next, the host performs a write operation to enter the command into the on-chip command register. Upon reception of a command, the MLAPD sets its on-chip semaphore register to the value 'FE' hex. After either command completion or command acceptance, depending on the specific command, the MLAPD sets the value of the semaphore register to 'FF' hex. The host must always read the semaphore register before writing a new command.

There are 32 commands which belong to one of the following five categories:

Initialization—These commands configure the MLAPD for operation after a hardware or software reset by specifying the system data bus width and the location of the Global Configuration Block in memory.

Host/MLAPD Interface—These commands instruct the MLAPD to perform operations that are not explicitly named in the LAPD protocol, but which are required by the host to interface with the MLAPD.

Protocol Handling—These commands are the primitives explicitly defined in the LAPD protocol. The protocol commands allow the host to set-up and resume link operation, as well as to control the flow of frames on the link.

Protocol Extension Handling—These commands are not defined in the LAPD protocol. These commands allow the host to intercede in the normal flow of the MLAPD protocol processing.

Test/Diagnostics—These commands allow the host to access all internal MLAPD registers or to test the DMA interface.

Figure 4 summarizes the MLAPD command set.

Exception Processing

An Interrupt Queue is located in shared memory to support the speed requirement of reporting many interrupting events occurring in rapid succession across the various active logical links. Each entry in this Interrupt Queue consists of a logical link identification number (LLID) if applicable, and a cause indication. Each of the 31 potential interrupt sources, with the exception of bus error/address error and interrupt queue overflow, may be individually masked by the host to prevent the reporting of a specific event.

After the MLAPD has written an entry into the Interrupt Queue, a hardware interrupt request may be activated. If polled operation is desired, this external interrupt indication is disabled by the host during MLAPD initialization. If an interrupt acknowledgement cycle is to be implemented in an interrupt driven system, the interrupt_vector register must be programmed by the host during the initialization procedure. The interrupt_vector register is located on-chip and allows the MLAPD to return an indication of the interrupt cause as part of an interrupt acknowledgement cycle on the system bus. The two possible vectors reported across the bus are:

The two possible vectors reported across the bus are:

- Normal Interrupt
- Severe Interrupt (bus error/address error)

Interrupt information is written into the Interrupt Queue to fully identify the specific cause of the interrupt in either a polled or interrupt driven system. Interrupts are the principle mechanism to report protocol events and errors (including error thresholds exceeded). Since the Interrupt Queue is circular and of limited size, the host is responsible for "reading" the interrupt information and maintaining free entries to record new interrupt events.

Initialization Overview

As a part of initialization, the host must:

- Prepare the Global Configuration Block (GCB). The host specifies the addresses of the various memory tables in the GCB. The GCB is also programmed with several global parameters that are required to process the LAPD protocol for all links.
- Clear these tables: Match Table, Interrupt Queue, and Timer Table.
- Prepare the Logical Link Table(s) (LLT) with the local parameters for the link(s) in service.
- Construct receive pool(s) of frame descriptors.

COMMAND GROUP	COMMAND	FUNCTIONAL DESCRIPTION
INITIALIZATION	RESET	SOFTWARE RESET.
	SET_BUS_WIDTH_8	DATA BUS WIDTH IS 8 BITS.
	SET_BUS_WIDTH_16	DATA BUS WIDTH IS 16 BITS.
	INIT	MLAPD LOADS REGISTERS FROM THE GCB.
HOST/MLAPD INTERFACE	OFF-LINE	MLAPD ONLY EXECUTES HOST COMMANDS.
	ON-LINE	MLAPD EXECUTES COMMANDS, RECEIVES FRAMES, AND TRANSMITS FRAMES.
	RELOAD	MLAPD LOADS REGISTERS FROM RELOADABLE AREA OF GCB.
	DUMP_STATISTICS	MLAPD WRITES GLOBAL COUNTERS TO SPECIFIED MEMORY AREA.
	PRESET_STATISTICS	MLAPD LOADS GLOBAL COUNTERS FROM STATISTICS THRESHOLD AREA OF GCB.
	ENABLE_IRQ	ALL PENDING INTERRUPT ENTRIES HANDLED. ENABLE EXTERNAL INTERRUPT SIGNAL.
	ASSIGN_POOL_POINTER	HOST HAS CREATED A NEW RECEIVE POOL.
PROTOCOL	MDL_ASSIGN_REQUEST	PLACE SPECIFIED LOGICAL LINK IN THE TEI_ASSIGN STATE.
	DL_ESTABLISH_REQUEST	SET-UP LINK. LINK ENTERS MULTIPLE FRAME ESTABLISHED MODE.
	DL_DATA_REQUEST	REQUEST TRANSMISSION OF AN I FRAME QUEUE FOR THE SPECIFIED LOGICAL LINK.
	RELINK_REQUEST	FRAMES WERE ADDED TO A QUEUE IN WHICH ALL FRAMES WERE 'Tx AWAITING ACK'.
	GLOBAL_XID/UI_REQUEST	REQUEST TRANSMISSION OF XID/UI FRAMES IN THE GLOBAL XID/UI QUEUE.
	XID/UI_QUEUE_0_REQUEST	REQUEST TRANSMISSION OF XID/UI FRAMES IN XID/UI_QUEUE_0.
	XID/UI_QUEUE_1_REQUEST	REQUEST TRANSMISSION OF XID/UI FRAMES IN XID/UI_QUEUE_1.
	MDL_ERROR_RESPONSE	INDICATION THAT NO DLCI ASSIGNED. THE LOGICAL LINK IS IN TEI_UNASSIGN STATE.
	DL_RELEASE_REQUEST	RELEASE LINK FROM THE MULTIPLE FRAME ESTABLISHED MODE.
	MDL_REMOVE_REQUEST	REMOVE SPECIFIED LOGICAL LINK FROM LAPD SERVICE.
PROTOCOL EXTENSION	ACTIVATE_LL	PLACE SPECIFIED LOGICAL LINK IN THE TEI_UNASSIGN STATE.
	DEACTIVATE_LL	REMOVE DLCI-LLID PAIR. PLACE LOGICAL LINK IN TEI_UNASSIGN STATE.
	REMOTE_STATUS_REQUEST	SEND FRAME WITH P BIT SET TO CHECK REMOTE STATION STATUS FOR SPECIFIED LINK.
	SET_LOCAL_BUSY	SPECIFIED LOGICAL LINK ENTERS THE LOCAL BUSY CONDITION.
	CLEAR_LOCAL_BUSY	SPECIFIED LOGICAL LINK EXITS THE LOCAL BUSY CONDITION.
	STOP_TX_I	STOP TRANSMISSION OF THE SPECIFIED LOGICAL LINK'S I TRANSMIT QUEUE.
	STOP_GLOBAL_XID/UI	STOP TRANSMISSION OF FRAMES IN THE GLOBAL XID/UI QUEUE.
	STOP_XID/UI_QUEUE_0	STOP TRANSMISSION OF FRAMES IN THE XID/UI_QUEUE_0.
	STOP_XID/UI_QUEUE_1	STOP TRANSMISSION OF FRAMES IN THE XID/UI_QUEUE_1.
TEST/ DIAGNOSTIC	DMA TEST	TEST DMA OPERATION.
	DUMP	DUMP ALL INTERNAL MLAPD REGISTERS AND CAM.

Figure 4. Command Set Summary

- Construct the necessary entry(s) in the LLID-LLT Table.
- Issue a RESET command to the MLAPD.
- Issue a SET_BUS_WIDTH_16,(8) command to the MLAPD, which specifies an 8- or 16-bit data bus configuration.
- Load the interrupt_vector register in the MLAPD device.
- Load the address of the Global Configuration Block into the data register in the MLAPD device.
- Issue the INIT command to the MLAPD.

The directly addressible MLAPD registers are shown in Figure 5.

A1	A2	15 — 8	7 — 0
0	0	UNDEFINED/RESERVED	COMMAND/SEMAPHORE
0	1	UNDEFINED/RESERVED	INTERRUPT_VECTOR
1	0	DATA	
1	1		

Figure 5. Directly Accessible Registers

Frame transmission and reception is enabled for a specific logical link after initialization by performing a Data Link Connection Identifier (DLCI) assignment procedure. (The DLCI is contained in the address field of LAPD frames. The DLCI is formed by the concatenation of the 6-bit service access point identifier, SAPI, and the 7-bit terminal endpoint identifier, TEI.) This logical link then enters a protocol-defined state in which it can receive and transmit unnumbered information frames. After the exchange of SABME and UA frames, the link set-up procedure is complete and the data connection is established. The MLAPD device is now ready to receive and transmit numbered information (I) frames that contain the assigned DLCI.

RECEIVE PROCESS

RECEIVE DATA STRUCTURES

The host maintains receive pools that contain free receive frame descriptors with associated data buffers. The host can create up to 8192 receive pools and can assign any number of logical links to the same receive pool. The data buffers in a pool must be of a length at least equal to the largest N201_value specified for any logical link assigned to the pool. The receive data structure is shown in Figure 6. The format of a receive frame descriptor is shown in Figure 7.

A receive pool must always contain at least one frame descriptor. The last frame descriptor in the pool is the pool_dummy frame descriptor and the last_in_pool bit in this frame descriptor's control_bits entry is set. When the receive_pool_pointer points to this pool_dummy frame descriptor, the receive pool is defined to be empty.

The host can add frame descriptors to a receive pool dynamically. The host maintains a user_Rx_pool tail_pointer for each receive pool, which points to the pool_dummy frame descriptor, for this operation. The host may also request a red_line interrupt when the MLAPD uses a frame descriptor near the end of the receive pool, allowing the host to add frame descriptors to avoid a receive busy condition.

After creating a receive pool, the host specifies a receive_pool_pointer, which is the address of the first frame descriptor in the receive pool, and issues an ASSIGN_POOL_POINTER command to cause the MLAPD to load the receive_pool_pointer into the Receive Pool Pointers Table. The MLAPD stores the first sixteen receive_pool_pointers (0 to 15) on-chip and stores any additional receive_pool_pointers (16 to 8191) in the Receive Pool Pointers Table in shared memory. The address of the memory table is specified by the host in the Global Configuration Block.

The host assigns each logical link to a receive pool via the receive_pool_number entry in the link's LLT. During frame reception, the MLAPD uses the receive_pool_number entry for the addressed link as an offset into the Receive Pool Pointers Table to locate the first receive frame descriptor in the assigned pool. Refer to Figure 8. When the frame is successfully received, the MLAPD indicates the frame type and stores the associated LLID in the receive frame descriptor. Then the MLAPD reads the next_rx_frame descriptor entry in this frame descriptor and updates the receive_pool_pointer for this pool in the Receive Pool Pointers Table. Although the linkage to the receive pool is not broken, the frame descriptor can now be considered part of a receive queue. All logical links that share the same receive pool will also share a common receive queue. It is the responsibility of higher level software to remove valid frames from the receive queue and eventually return the frame descriptors to the receive pool.

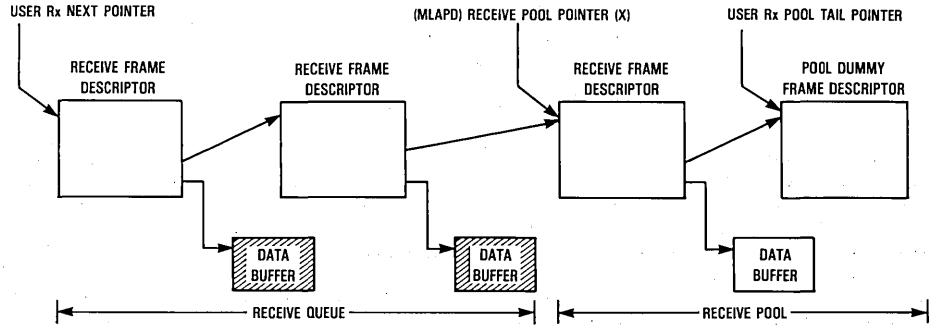


Figure 6. Receive Structures

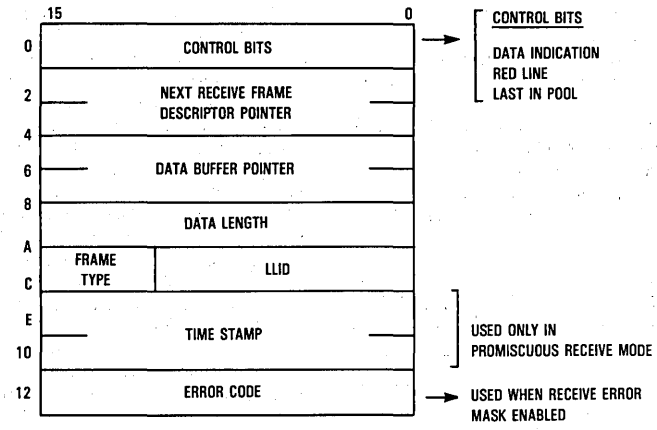


Figure 7. Receive Frame Descriptor

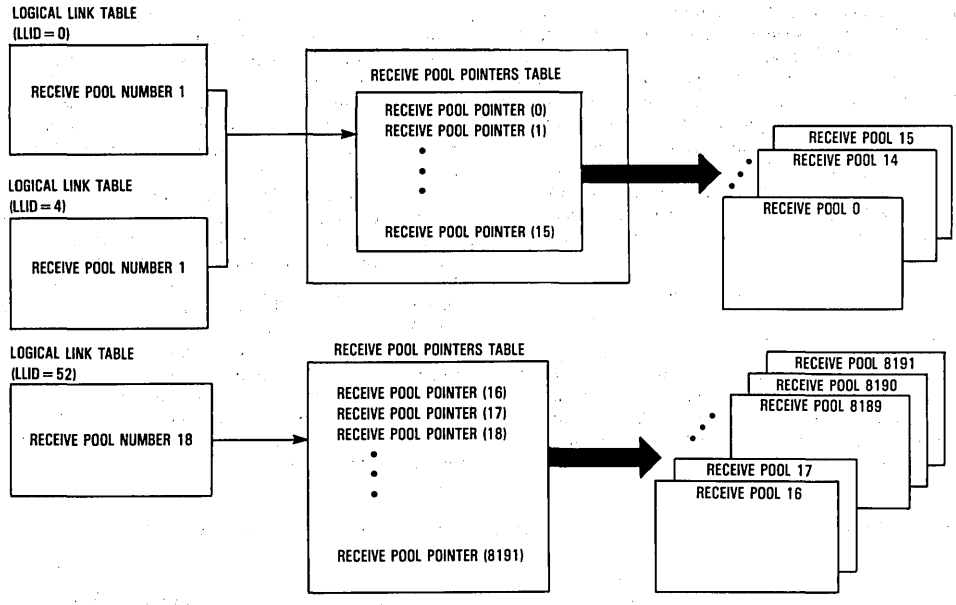


Figure 8. Receive Pool Lookup

Information Frame Reception

As the first step in the receive process, the MLAPD must determine whether the DLCI in an incoming frame has been assigned to an active logical link by Level 3. When the expanded system operation mode is selected by the host, the MLAPD uses the incoming DLCI as an offset into the external Match Table. If this entry is marked as invalid, then the incoming DLCI has not been assigned to a logical link and the frame is ignored. An error counter is also incremented. The receive process for expanded system operation mode is shown in Figure 9.

If the DLCI entry is marked as valid, then the entry also contains an associated logical link identification number (LLID). (Level 3 defines a 13-bit logical link identification number corresponding to each active DLCI. The LLID is used during link level processing and is only of local significance. The LLID serves to reduce the external memory requirements for LAPD processing.) The MLAPD uses the LLID as an offset into the LLID-LLT Table to locate this link's Logical Link Table (LLT). The MLAPD accesses the link's LLT to obtain the specific protocol parameters for this link which are required for subsequent protocol processing of the received frame. In the case

of an information bearing frame, the MLAPD uses the receive_pool_number entry in the LLT as an offset into the Receive Pool Pointers Table to locate the first available receive frame descriptor for this logical link. (The first 16 pool pointers are stored on-chip to reduce the number of memory accesses for the receive operation.) If the receive pool associated with this logical link is empty or the link's local_busy flag is set, then a Level 2 frame (RNR) is queued for transmission and the data is ignored. Otherwise the MLAPD continues to process the I frame by comparing the N(S) to the logical link's state variable V(R), also contained in the LLT. If the frame is in sequence, then the information field is transferred through the receive FIFO into the receive buffer. An out-of-sequence frame causes a Level 2 frame (REJ) to be queued for transmission and the out-of-sequence frame's information field is discarded. A frame received with an invalid N(R) causes a Level 2 frame (FRMR) to be queued for transmission (when the Tx_FRMR_select option is enabled) and the information field of the frame with an invalid N(R) is discarded.

Lastly the MLAPD checks the CRC of the incoming frame. If no CRC error is detected, the MLAPD updates the LLT state variable V(R) and acknowledges the frame reception by queuing a RR or RNR frame for transmission.

Zero deletion is performed throughout the reception process. The MLAPD requests the system bus when there are at least four words in the receive FIFO. Frames are received in sequence as long as memory buffers are available and adequate bandwidth is provided for DMA on the system bus.

When the application is limited to supporting 16 logical links or less, the host selects on-chip system operation mode. In the on-chip operation mode, the DLCI match operation is performed by an internal CAM circuit, reducing external memory requirements. Figure 10 shows how the MLAPD uses the received DLCI field to search the CAM as part of the receive process.

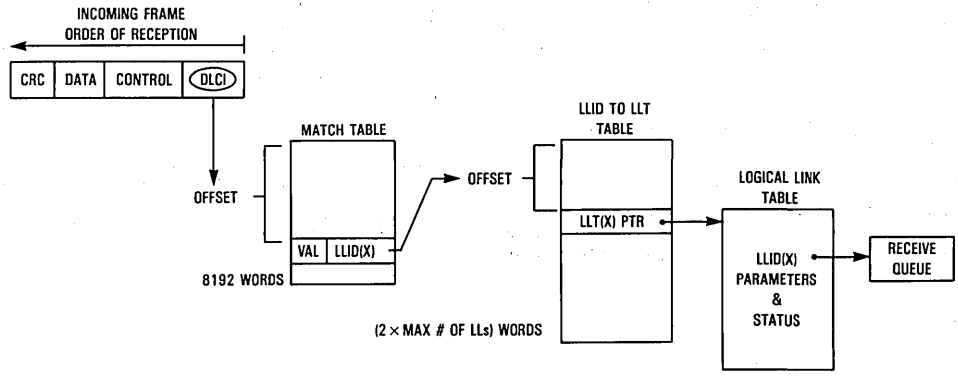


Figure 9. Receive Process for Expanded Mode

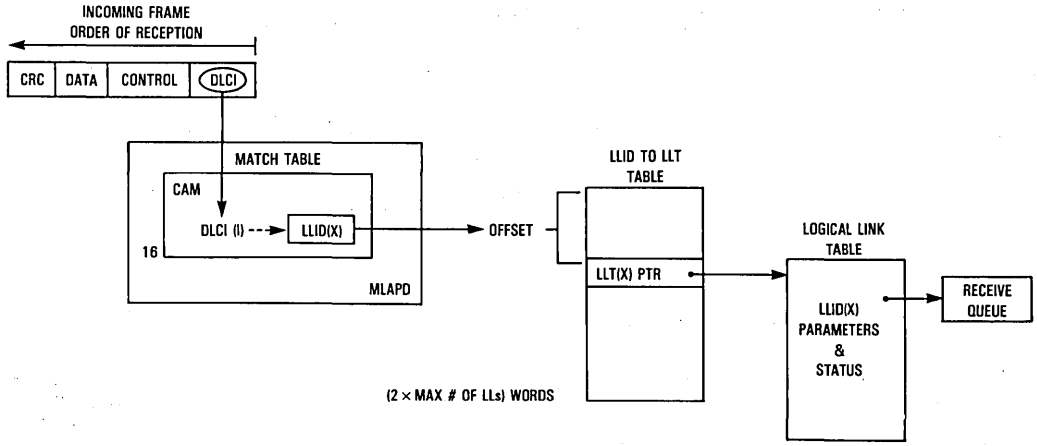


Figure 10. Receive Process for On-Chip Mode

XID/UI Frame Reception

The MLAPD handles reception of exchange identification (XID) frames, unnumbered information (UI) frames, and frames containing a user-defined non-standard control field in a similar manner as I frame reception. Received XID, UI, and non-standard_control frames are placed in the logical link's receive queue. To differentiate between types of received frames in the receive queue, the MLAPD encodes the frame_type bits in each receive frame descriptor.

Collecting Received Frames

The user must maintain a user_Rx_next_pointer for each queue that points to the first receive frame to be collected by the host. The host collects received frames by traversing the linked list of frame descriptors until a frame descriptor is reached with frame_type bits indicating that this frame descriptor still belongs to the receive pool and has not been used for frame reception. The host may set the data_indication bit in this frame descriptor to request an interrupt on the removal of this frame descriptor from the receive pool. After collecting receive frames, the host should update the user_Rx_next_pointer.

FRAME TRANSMISSION OVERVIEW

TRANSMIT SERVICING SCHEME

The MLAPD services several transmit queues. The frames in any single queue can be categorized as: Level 2 generated frames, XID/UI frames (including non-standard_control frames), or numbered information (I) frames. A fixed internal servicing scheme determines when each transmit queue is handled.

The highest priority transmit queue is the Level 2 Transmit Queue. The Level 2 Queue contains unnumbered (U) and supervisory (S) frames generated by the MLAPD in response to received frames and host commands. These frames are referred to as "Level 2 frames".

The next lower priority transmit queue contains XID and UI frames for transmission on the logical link specified in the frame descriptor. It is recommended that system implementations use this Global XID/UI Queue for frames generated by the Level 3 management entity.

The lowest priority transmit queues contain either I frames or XID/UI frames. The MLAPD supports up to four transmit I frame queues (I frame Queues_0-3) and up to two XID/UI queues (XID/UI Queues_0,1). The user defines the relative servicing for these six queues by the MLAPD transmit task by programming a scan length for each queue. The scan length defines the maximum number of frames to be transmitted from the queue before switching to service the next queue. When a scan length of zero is programmed, the corresponding queue is never serviced. In this way the user can implement zero, one, two, three, or four queues for I frame transmission, and zero, one, or two queues for XID/UI frame transmission.

Before each frame from the lowest priority queues is transmitted, the MLAPD transmits all frames in the Level 2 Queue and all frames in the Global XID/UI Queue. The MLAPD services the lowest priority queues in round robin fashion. The MLAPD transmit servicing scheme is shown in flowchart form in Figure 11.

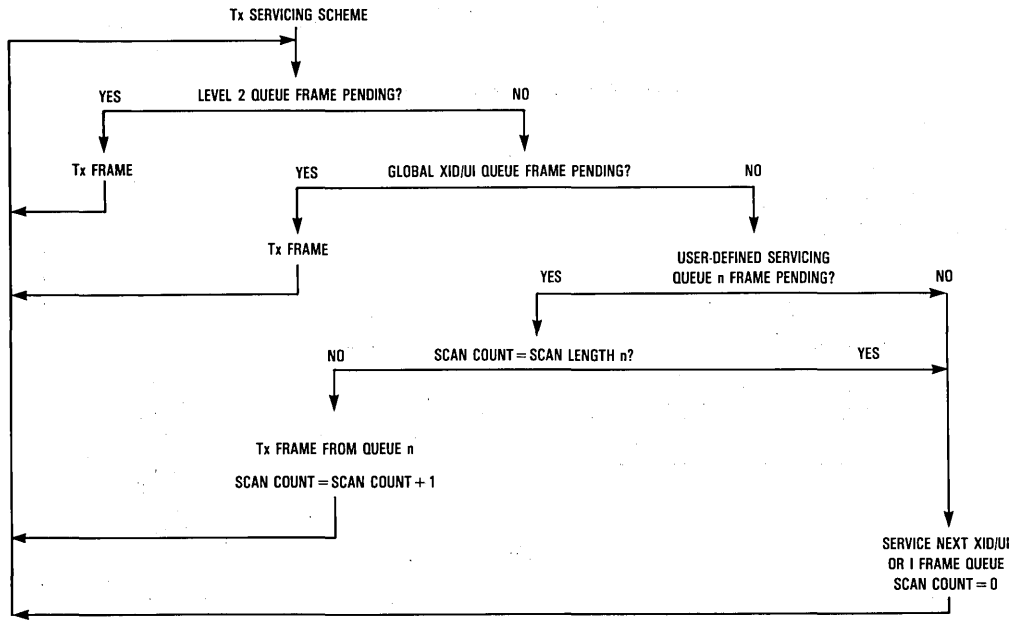


Figure 11. Transmit Servicing Scheme

I FRAME TRANSMISSION PROCESS

I Frame Queues and Logical Link Transmit Queues

Each I frame Queue is organized as a linked list, where the linkage mechanism is implemented by each queued link's Tx_next_LLT entry in its associated Logical Link Table (LLT). To minimize context switching, each link has its own transmit queue for I frames. The Tx_next_pointer entry in each LLT points to the first frame descriptor in the link's transmit queue. An I frame Queue is shown in Figure 12.

The host assigns each logical link to one of the four I frame Queues for all its numbered information frame transmissions. When the logical link's transmit queue is linked to its assigned I frame Queue, the pending I frames can be transmitted as part of the MLAPD transmit servicing scheme.

I Frame Transmission Queueing

When the host has data ready for transmission on a logical link, the host prepares one or more frame descriptors which contain information required by the MLAPD to transmit the associated frame(s). The format of a transmit I frame descriptor is shown in Figure 13. The host then links the frame descriptor(s) to the desired logical link's transmit queue. The transmit queue structure for numbered I frames is shown in Figure 14. If the logical link's transmit queue is empty when the new frame descriptor(s) is (are) linked to the queue, the host must issue a DL_DATA_REQUEST command for this link and specify the head of the transmit queue. This command causes the MLAPD to place this logical link into its assigned I frame Queue for transmit servicing. However, if frames are waiting for transmission in the link's transmit queue when the new frame descriptor(s) is (are) added, the linking of the descriptor(s) is sufficient for the associated frame(s) to be transmitted. If all the frames for this logical link were already transmitted but there are still some frames awaiting acknowledgement, the host must issue a RELINK_REQUEST command to enable transmission.

MLAPD I Frame Queue Processing

When an information frame is to be transmitted from an I frame Queue, the MLAPD first fetches the link's context from its associated Logical Link Table (LLT). This context information is used to build the address and control fields for the frame, which are then placed in the transmit FIFO. The Tx_next_pointer entry in the LLT identifies the address of the next I frame descriptor to be handled. The MLAPD uses information contained in this frame descriptor to locate the data to be sent, and then transfers this data into its transmit FIFO. As the MLAPD sends the frame, zeros are inserted when necessary for transparency and the MLAPD attaches a frame check sequence to complete the frame. After frame transmission, V(S) is updated and T200 is started if it is not already running. The MLAPD sets the transmit bit in the frame descriptor and updates the Tx_next_pointer entry.

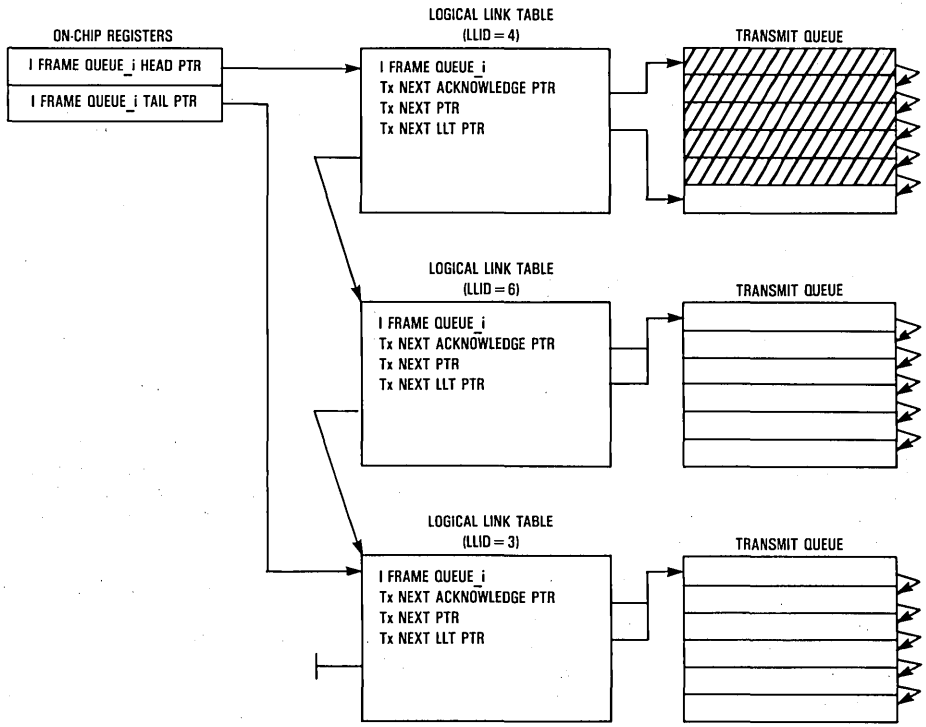
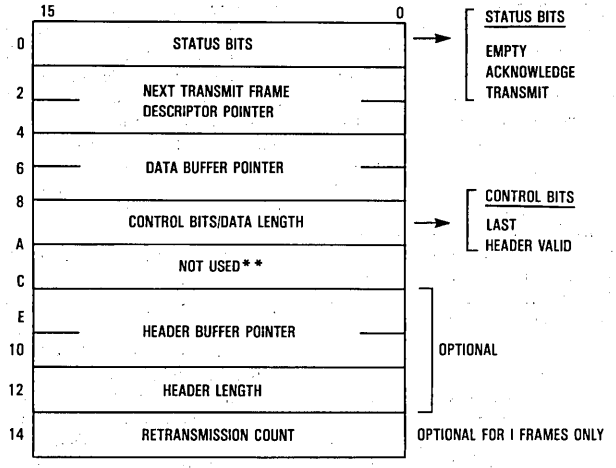


Figure 12. I Frame Queue Structure

7



**Contains frame type and LLID for XID/UI frames.
Contains CRC enable and address enable bits for non-protocol links.

Figure 13. Transmit Frame Descriptor

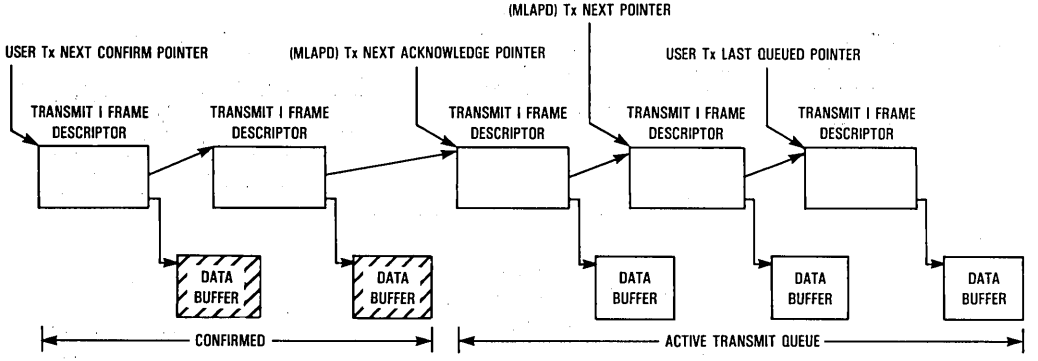


Figure 14. Logical Link Transmit Queue Structure

Transmission begins when ten bytes are present in the transmit FIFO, in addition to the address and control fields, or when the entire frame is present in the transmit FIFO. Between frames the MLAPD transmits the user selected number of pad flags. Additional flags may be transmitted, until the requirements for start of transmission are met. While transmitting a frame, the MLAPD requests the system bus when there are six to eight empty bytes in the transmit FIFO.

The MLAPD continues transmitting frames for the same logical link until one of the following events occur:

- The logical link's transmit queue is exhausted;
- The MLAPD transmitter switches to service another queue according to the transmit servicing scheme.

The MLAPD may also temporarily remove this link's transmit queue from its I frame Queue as the result of link conditions and host commands. The link conditions that cause removal are:

- Remote busy condition;
- Outstanding remote status request (command frame sent with poll bit set to one);
- Outstanding frames limit (K);
- Link reset;
- Timer recovery condition.

Stopping I Frame Transmission

At any time, the host may instruct the MLAPD to immediately suspend all transmit activity for a logical link's transmit queue by issuing a STOP_TX_I command. Upon receiving this command, the MLAPD aborts any current frame transmission for this link and removes the specified link from its assigned I frame Queue. The host is later notified when all outstanding acknowledgments have been received. At this point, the host is free to manipulate the transmit queue as desired, since the MLAPD is deactivated for that queue. This stop mechanism may be used to provide faster servicing of certain I frames or this mechanism may be necessary for various types of error recovery.

Collecting Acknowledged I Frames

To collect acknowledged frames, the host reads the status_bits in each frame descriptor, beginning with the frame descriptor indicated by the user_Tx_next_confirm_pointer. When the acknowledge bit is set, the associated data buffer has been transmitted and acknowledged. Acknowledged frames may be collected by the host dynamically (while other frames await acknowledgement) or after being notified by the MLAPD that all frames in the queue are acknowledged via a DL_data_confirmation interrupt. The host should update the user_Tx_next_confirm_pointer after each frame is collected.

XID/UI FRAME TRANSMISSION PROCESS

XID/UI Transmit Servicing

An XID/UI frame may be an unnumbered information (UI) frame, an exchange identification (XID) frame, or a frame with a user-defined non-standard_control field. In most ISDN applications it is anticipated that these frames will be used infrequently by Level 3 entities. However in some applications, such as bridges which interface to connectionless-oriented networks, UI frames may be used heavily for data transfer. To address the varying usage of XID, UI, and non-standard_control frames, the MLAPD provides two methods of servicing XID/UI queues. When frames are queued to the Global XID/UI Queue, the MLAPD will transmit all pending frames each time this queue is serviced. When frames are queued to either XID/UI Queue_0 or XID/UI Queue_1, the MLAPD will transmit only the number of frames specified by the corresponding user-defined scan length. It is recommended that the Global XID/UI Queue be reserved for management information exchange.

All XID/UI queues are linked lists of frame descriptors, where the frame descriptor specifies the logical link associated with the queued frame and the frame type. The structure of the XID/UI queues is shown in Figure 15.

XID/UI Frame Transmission Queueing

When XID/UI (or non-standard_control) information is ready for transmission on a logical link, the host prepares one or more frame descriptors which contain the information required by the MLAPD to transmit the frame(s). Then the host links the frame descriptor(s) to the appropriate XID/UI queue, depending on the servicing desired by the host for these frames. If the queue is empty when the new frame descriptor(s) is (are) linked, the host must issue the appropriate command (GLOBAL_XID/UI_REQUEST, XID/UI_QUEUE_0_REQUEST, or XID/UI_QUEUE_1_REQUEST) and specify the head of this queue. When this command is issued, the MLAPD places the corresponding XID/UI queue into the transmit servicing scheme. If frames are awaiting transmission when the new frame descriptor(s) is (are) added, the linking of the frame descriptor(s) is sufficient for the associated frame(s) to be transmitted.

MLAPD XID/UI Queue Processing

When an XID/UI or non-standard_control frame is to be transmitted, the MLAPD uses the LLID entry in the frame descriptor to index into the LLID-LLT Table to obtain the appropriate DLCI for the frame. The MLAPD places this DLCI and the correct control field into its transmit FIFO. Next, the address of the frame data is read from the frame descriptor and the MLAPD transfers this data into the transmit FIFO. As the frame is transmitted, zeros are inserted when necessary for transparency. CRC is calculated and appended to complete the frame.

7

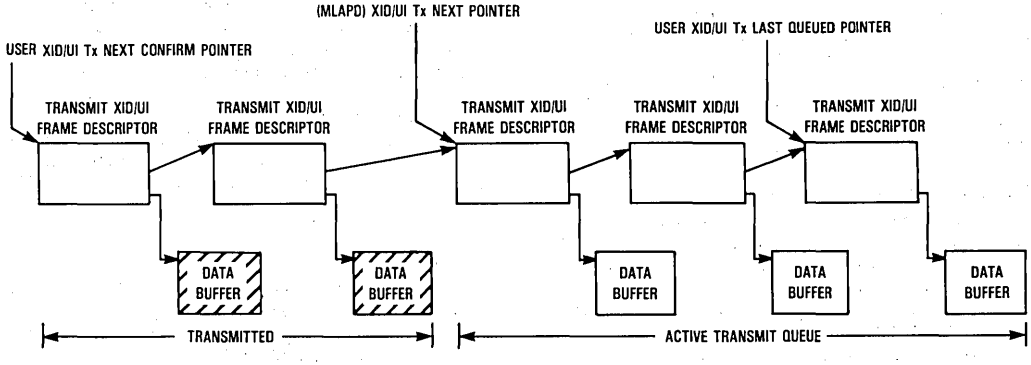


Figure 15. XID/UI Queue Structure

Stopping XID/UI Frame Transmission

At any time the host may instruct the MLAPD to immediately suspend all transmit activity for the Global XID/UI Queue, for XID/UI Queue_0 or for XID/UI Queue_1. This result is produced by the corresponding STOP_TRANSMIT command (STOP_GLOBAL_XID/UI, STOP_XID/UI_QUEUE_0, or STOP_XID/UI_QUEUE_1). Upon receiving one of these commands, the MLAPD aborts any current XID/UI (or non-standard-_control) frame transmission and does not service additional XID/UI frame descriptors in the specified queue. At this point, the host is free to manipulate the queue as desired, since the MLAPD is deactivated for that queue. This stop mechanism may be used to provide faster servicing of certain XID, UI, or non-standard_control frames for a particular logical link, or this mechanism may be necessary for various types of error recovery.

Collecting Transmitted XID/UI Frames

The collection of transmitted XID/UI frames is the same for all three XID/UI Queues. To collect transmitted frames, the host reads the status_bits in each frame descriptor, beginning with the frame descriptor indicated by the user_XID/UI_Tx_next_confirm_pointer associated with the queue. When either the positive_confirmation bit or the negative_confirmation bit is set, the associated data buffer has been handled by the MLAPD.

The positive_confirmation bit indicates that the associated data buffer has been transmitted, while the negative_confirmation bit indicates that the associated data buffer cannot be transmitted. A negative indication is generated when a frame transmission is requested in conflict with the services available in the current link state (TEI_UNASSIGN or EST_WAIT_TEI). The host should later resubmit the DL_UI frame for transmission.

Transmitted frames may be collected by the host dynamically (while other frames await transmission) or after being notified by the MLAPD via an XID/UI_confirmation interrupt that all frames in the queue have been sent. After each frame is collected from an XID/UI queue, the host should update the associated user XID/UI_Tx_next_confirm_pointer.

7

MLAPD IMPLEMENTATION OF SPECIAL MODES

NON-PROTOCOL LINKS

The MLAPD can simultaneously support links which operate according to the LAPD protocol and links which do not operate according to the LAPD protocol. The host specifies that the link will operate in non-protocol mode by programming the non-protocol_select bit in the link's Logical Link Table. When supporting a non-protocol link, the MLAPD functions as a HDLC framer with DMA capability.

Queueing Frames for Transmission

To queue frames for transmission on a non-protocol link, the host first creates a transmit queue. The structure of the transmit queue is identical to a transmit queue which is used for I frames for LAPD links. To enable transmission of the transmit queue for a non-protocol link, the host issues a DL_DATA_REQUEST command which causes the MLAPD to place this logical link into its assigned I frame Queue for transmit servicing. If the host adds frames to a non-empty transmit queue, the linking of the frame descriptor(s) is sufficient for the associated frames to be transmitted. Note that for non-protocol links, the MLAPD does not recognize any incoming acknowledgements and the RELINK_REQUEST command is not meaningful.

MLAPD Frame Transmission

When a frame is to be transmitted for a non-protocol link, the MLAPD first fetches the link's context from its LLT. Then, based upon the transmit_address_enable bit in the frame descriptor, the MLAPD may or may not append the DLCI associated with this logical link to the beginning of the transmit frame. Next the MLAPD uses the information contained in the frame descriptor to locate the transmit memory buffer, and then transfers this information into its transmit FIFO. The MLAPD transmits the frame inserting zeros when necessary for transparency. After the transmit buffer is transmitted, the MLAPD may or may not append a CRC to the frame, based upon the transmit_CRC_enable bit in the frame descriptor.

Transmission begins when ten bytes are present in the transmit FIFO or when the entire frame is present in the transmit FIFO. Between frames the MLAPD transmits the user selected number of pad flags. Additional flags may be transmitted, until the requirements for start of transmission are met. While transmitting a frame, the MLAPD requests the system bus when there are six to eight empty bytes in the transmit FIFO.

Once the MLAPD begins handling a non-protocol link's transmit queue, the MLAPD will continue servicing the queue until all frames are transmitted or until the scan length associated with the I frame Queue is exhausted.

Stopping Frame Transmission

At any time, the host may instruct the MLAPD to immediately suspend all transmit activity on a non-protocol link by issuing a STOP_TX_I command. Upon receiving this command, the MLAPD aborts any current frame transmission for this link and does not service additional frame descriptors. At this point, the host is free to manipulate the transmit queue as desired, since the MLAPD is deactivated for this link. This stop mechanism may be used to provide faster servicing of certain frames for this link or this mechanism may be necessary for various types of error recovery.

Collecting Transmitted Frames

To collect transmitted frames, the host reads the status_bits in each frame descriptor, beginning with the frame descriptor indicated by the user_Tx_next_confirm_pointer. When the transmit bit is set, the associated data buffer has been transmitted. Transmitted frames may be collected by the host dynamically (while other frames await transmission) or after being notified by the MLAPD that all frames in the queue are transmitted via a DL_data_confirmation interrupt. The host should update the user_Tx_next_confirm_pointer after each frame is collected.

Receive Structures

The receive structures for a non-protocol link are identical to the structures for a LAPD link. A non-protocol link is assigned to a receive pool by the host using the receive_pool_number entry in the link's LLT. Any number of links may share a receive pool, although it is anticipated that most system

implementations will not assign LAPD links and non-protocol links to the same receive pool. Note that the N201_value specified for a non-protocol link defines the maximum length for a received frame.

MLAPD Frame Reception

As an incoming frame is received in on-chip system operation mode, the DLCI field in the frame is compared to the on-chip CAM to identify the corresponding logical link, if any. In expanded system operation mode, the MLAPD accesses the external Match Table to determine the associated logical link. If no DLCI match is found, then the incoming DLCI has not been assigned to a logical link, and the frame is ignored.

If a DLCI match is found, the corresponding entry in the Match Table or CAM specifies both the LLID and whether the link is assigned to LAPD protocol operation or non-protocol operation. If the frame is for a non-protocol link, the MLAPD proceeds to locate a free buffer to store the information between the opening and closing flags of the frame, minus the 16-bit address field. (The associated LLID is written into the frame descriptor.) The MLAPD locates the link's receive pool by using the link's receive_pool_number LLT entry to index into the Receive Pool Pointers Table. If the receive pool associated with this logical link is empty, then the frame is ignored and a local_busy interrupt is generated.

Otherwise the MLAPD begins to transfer the frame into the memory buffer. During frame reception, the MLAPD performs zero deletion, checks for receive errors, and calculates the CRC. The host may save erroneous frames for inspection based upon a non-protocol_error_mask. For non-protocol links, the host may also choose to have the MLAPD access multiple receive buffers as needed to store a received frame.

PROMISCUOUS RECEIVE MODE

Promiscuous receive mode is a special case of non-protocol link operation. This mode only affects the MLAPD receiver operation; the MLAPD transmitter operation is not affected. When promiscuous receive mode is selected, the MLAPD does not perform LAPD frame processing on the incoming serial bit stream. The MLAPD strips flags, performs zero deletion, and checks the frame CRC. No address matching or control field analysis is performed. The entire frame, as delineated by the HDLC flag characters, is stored in memory. The MLAPD uses the receive pool assigned by the host to the logical link associated with DLCI = 0.

The host may choose to accept erroneous frames based upon the non-protocol_error_mask. The host may also choose to have the MLAPD access multiple receive buffers as needed to store a received frame. As each receive buffer is filled, the MLAPD writes a time stamp into the associated frame descriptor. This time stamp indicates the "time", with respect to an internal 32-bit timer, when the MLAPD completed transferring the incoming frame to the receive buffer. When the MLAPD is in promiscuous receive mode, the host may enable a filtering option which allows the host to selectively receive frames from the serial link based upon the first 32 bits of each frame. This provides a built-in logic analyzer trigger-type diagnostic capability.

LINE MONITOR

This mode allows the user to monitor all the bits on the data link. When the $\overline{\text{RSTART}}$ pin is asserted, the receive operation begins. As long as $\overline{\text{RSTART}}$ remains asserted, the chip will pack all received bits to words and write them into receive buffers from a single receive pool. This mode may be very useful in analyzing line problems, bit errors, zero insertion errors and special bit patterns such as flags, abort, and idle sequences, that are not normally dumped into memory.

SYSTEM LOOPBACK TESTING

To allow the MLAPD to perform the LAPD procedures during a system loopback test, the MLAPD provides a "flip" option. When enabled, the MLAPD will invert the least significant bit of the DLCI field for each received frame. To implement this mode, the host activates pairs of logical links with DLCIs that differ only in the least significant bit position. For each pair, one link is assigned as network and the other link is assigned as user. Then when a frame is transmitted for one logical link, it is received for the other logical link. Thus enabling the MLAPD to implement the LAPD procedures while in loopback mode.

The system loopback can be internal or external. When internal loopback is selected, the host may also specify that any bits received on RxD should be echoed back to the network on TxD. Note that when the flip option is not enabled, the host may only operate non-protocol links under the internal or external loopback configuration.

MEMORY-TO-MEMORY OPERATION

The MLAPD can be configured for operation in systems which do not implement a non-channelized serial interface, such as channelized T1 networks or LANs. It is assumed that a device in the system, labeled "Device A" in Figure 16, directly interfaces to the physical level. Received frames are placed into memory and transmit frames are placed into Device A using either DMA or host I/O capabilities. In memory-to-memory operation, the MLAPD performs the LAPD procedures on these memory resident receive and transmit frames. This mode allows the system designer to select the MLAPD to implement the LAPD procedures independent of the physical level characteristics of the system.

To enable memory-to-memory operation, the host sets an option_bit in the Global Configuration Block. In this configuration, the MLAPD transmitter and receiver are internally connected. The RxD and TxD pins are not used. The MLAPD on-chip DMA controller feeds the receive and transmit machines via their associated FIFOs. The host activates a logical link with DLCI = 0 as a non-protocol link and assigns this link to I frame Queue_0. Note that I frame Queue_0 should be reserved for use only by this logical link associated with DLCI = 0.

As frames are received from the physical level, the host is responsible for placing these received frames into the memory format defined for a MLAPD transmit queue. Each transmit buffer contains an entire received frame. The host then issues a DL_DATA_REQUEST with the LLID associated with DLCI = 0 as command argument 1 and the address of the first transmit frame descriptor as command argument 2. In response, the MLAPD begins transmitting frames from I frame Queue_0 according to the transmit servicing scheme described in **Transmit Servicing Scheme**. Since the logical link associated with DLCI = 0 is assigned to non-protocol operation, the MLAPD acts as a simple HDLC framer. The MLAPD may optionally

add the frame CRC as described in **MLAPD Frame Transmission**. The frame address should be contained in the transmit data buffer (passed through from the physical level). When the frame is received via the internal loopback path, the MLAPD analyzes the address field, determines whether the addressed logical link is assigned to LAPD or non-protocol operation and then handles the frame as described in **Information Frame Reception, XID/UI Frame Reception,** and **MLAPD Frame Reception**. The MLAPD continues to transmit frames from I frame Queue_0 until the scan length specified for I frame Queue_0 is exhausted or until all queued frames are transmitted. A DL_data_confirmation interrupt is issued to the host when all frames are transmitted. This interrupt indicates that all frames received from the physical level have been handled by the MLAPD. Level 3 collects the received I, XID, UI, and non-standard_control frames for the various logical links as described in **Collecting Received Frames.**

The memory-to-memory operation mode does not affect the procedure for passing transmit frames from Level 3 to the MLAPD for Level 2 handling. Level 3 queues I, XID, UI, and non-standard_control frames to the Global XID/UI Queue, I frame Queues_1-3, and XID/UI Queues_0,1. The MLAPD independently generates U and S frames and places them on the Level 2 Queue. The transmit servicing scheme for these transmit queues and the protocol actions provided by the MLAPD are unchanged. During the transmit process, the MLAPD appends the appropriate DLCI, control field and CRC to queued transmit frames for LAPD links, and the MLAPD may optionally append the DLCI and CRC for non-protocol links. As each transmitted frame is received via the internal loopback path, the MLAPD receiver handles the frame as for promiscuous receive mode. The entire frame is stored in memory. The MLAPD uses the receive pool assigned by the host to the logical link associated with DLCI = 0. It is the responsibility of the host to enable transmission of these frames by the physical level.

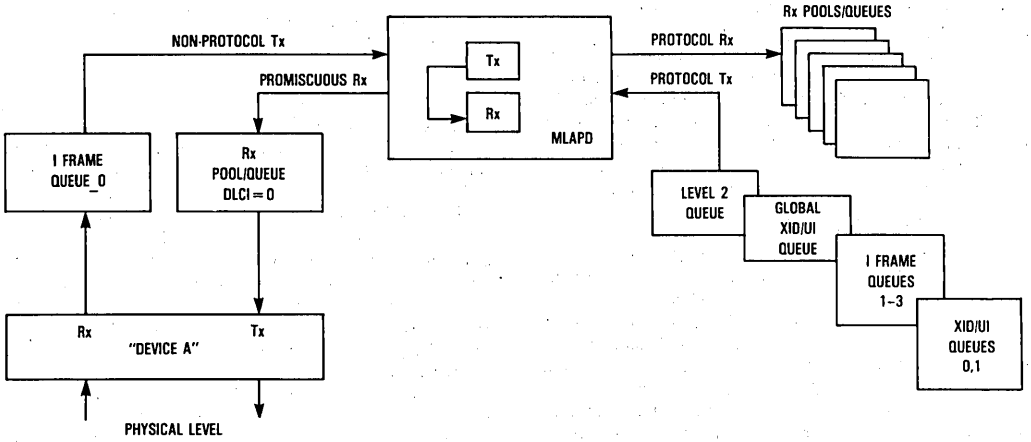
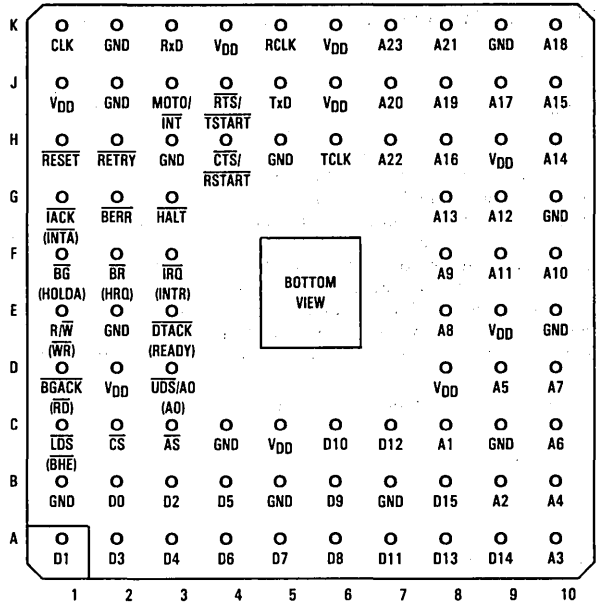


Figure 16. Memory-to-Memory Operation

7

PIN ASSIGNMENT

PIN GRID ARRAY



7

M68606ESP

Product Preview

Evaluation and Support Package (ESP) for the MC68606

The M68606ESP is a tool intended for evaluating the MC68606 multi-link access procedure (MLAPD) protocol controller as well as developing and debugging MC68606-based products. The Motorola MC68606 MLAPD is an integrated circuit that implements the link access procedure (LAPD) protocol. The M68606ESP provides the user with both hardware and software platforms for developing MLAPD applications as indicated in Figures 1 and 2.

The ESP hardware consists of a single board, which may be run as a stand-alone unit or connected to other devices, such as a host or printer, by serial or parallel I/O interfaces. The MC68606 serial interface signals are TTL buffered and made available to the user through a connector. Through this connector, the user can run the MLAPD serial clocks at a variable speed (e.g., dc to 2.048 Mbps) and interface to other devices, such as another ESP or a protocol analyzer. As shown in Figure 1, this package requires a power supply, an external clock generator for the MC68606 receive and transmit clocks, and a VT100-compatible CRT. The hardware features of the M68606ESP are as follows:

- Stand-Alone Operations (i.e., No Host Computer Necessary)
- MC68606 Running at 16.67 MHz
- MC68020 Running at 16.67 MHz
- 1/2-Mbyte Dynamic RAM
- 256-Kbytes EPROM
- 2-Kbytes EEPROM
- Two RS-232 Serial Ports (Terminal and Host)
- Parallel I/O Interface (Host or Printer)

7

To ease integration of user prototype software products with the MC68606, the ESP software provides powerful windows to all of the MC68606 memory structures and high-level software interfaces. The ESP software consists of two main modules provided in EPROM. The first module fully implements the layer-3 protocol defined for ISDN primary rate and DMI mode-3 data-transfer applications, namely CCITT Recommendation X.25 data phase. Shown in Figure 2, the source code for this module, called M68606SW3D, is also available. The layer-3 software accesses the MC68606 by a set of chip drivers, which comprise the layer 2/3 interface. The second module provides a menu-driven interface to the MC68606. This module includes the MC68606 chip drivers shown in Figure 2. The menus provided enable the user to issue single commands to the MC68606, to read and write all MC68606 memory structures, and to single-step execute protocol primitives at either layer 2 or 3.

The main software features are as follows:

- X.25 Data-Phase Layer-3 Protocol
- User-Software Interface from Either Layer 2 or 3
- MC68606 Chip Drivers
- Menu-Driven Software Allows User to Perform:
  - — Single Command Loading of MC68606
  - — Symbolic Read/Write Access to All MC68606 Memory Structures
  - — Execution of Single Protocol Primitives at Layer 2 or 3
- Board-Level Monitor

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

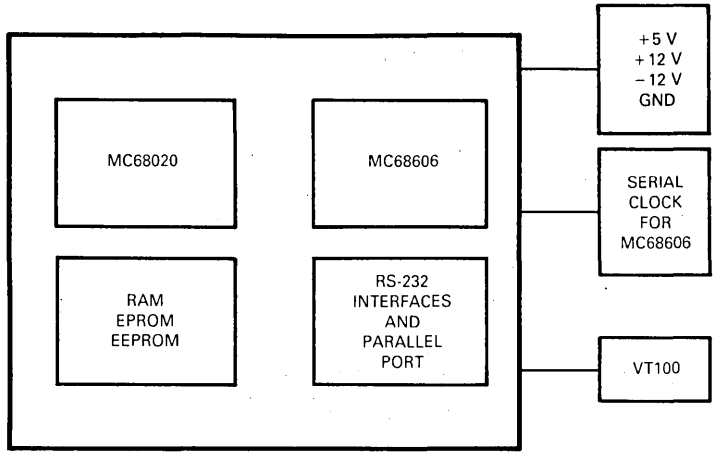


Figure 1. M68606ESP Minimum-Hardware Support

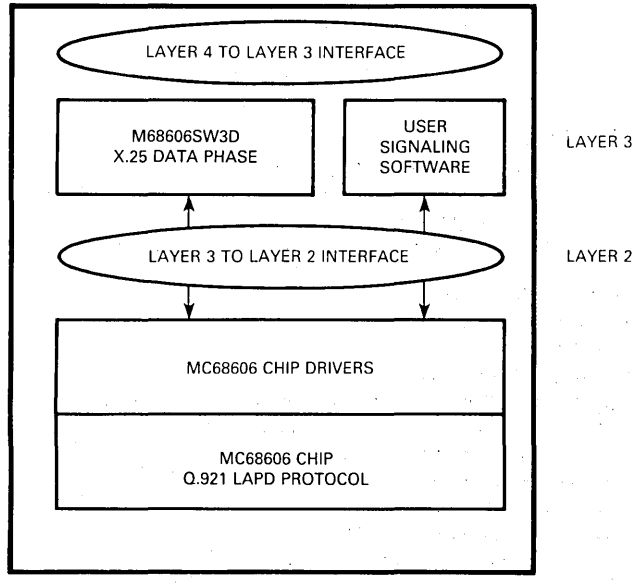


Figure 2. M68606ESP Software Support

7

MC68824

Technical Summary

Token-Passing Bus Controller (TBC)

The Motorola MC68824 Token Bus Controller (TBC) is a silicon integrated circuit which implements the Media Access Control (MAC) function for an IEEE 802.4 LAN station and the receiver portion for IEEE 802.2 Logical Link Control (LLC) type 3 as well as providing support for LLC type 1 and type 2 (see Figure 1). IEEE 802.4 defines the physical and MAC portion of the data link layer of the Manufacturing Automation Protocol (MAP) specification. The LLC functions implemented on chip are those associated with real time applications, namely "Acknowledged Connectionless Service" (type 3) as required in the Enhanced Performance Architecture (EPA) specified in Manufacturing Automation Protocol (MAP) 3.0.

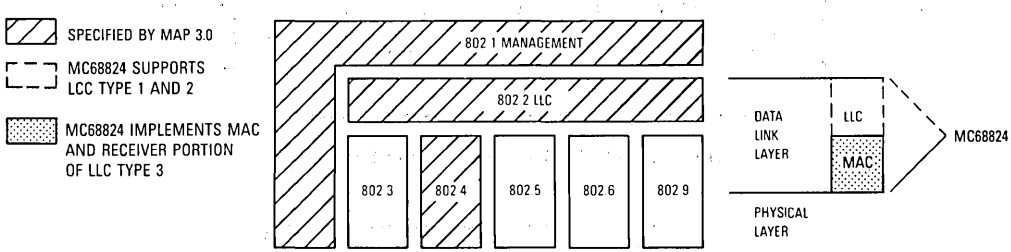


Figure 1. IEEE Standard Model

The major features of the MC68824 are:

- Implementation of the MAC Portion of the IEEE 802.4 Standard
- Implementation of the Receiver Portion of IEEE 802.2 LLC Acknowledged Connectionless Service (type 3)
- Support of IEEE 802.2 LLC Type 1 and Type 2
- Support of ANSI/ISA-72.01 PROWAY PLC Send Data with Acknowledge (SDA) and Request Data with Reply (RDR)
- MAC Options Suitable for Real Time Environments
  - — Four Receive and Four Transmit Queues Supporting Four Priority Levels
  - — Immediate Response Mechanism
- On-Chip Network Monitoring and Diagnostics
- Simple Interface to Higher Level Software by means of Powerful, Fully Linked Data Structure
- Options for Bridging Include: Hierarchical and IBM Defined Source Routing as well as Support for Flat Bridges
- Powerful Addressing; Group Address Recognition and Multidrop Capability
- Contains Several Modes to Increase Reliability and Flexibility Including:
  - — Reduced Data Structure Mode for Increased Performance
  - — Control Frame Preference for Increased Reliability
  - — Address Comparison Options for Increased Performance
  - — Bus Analyzer Mode to Enable Running the TBC as a Powerful Protocol Analyzer

— Continued —

Features (Continued)

- System Clock Rate up to 16.67 MHz
- Serial Data Rates from 10 Kbits/Second to 12.5 Mbits/Second
- IEEE 802.4 Recommended Serial Interface Supporting Various Physical Layers
- Highly Integrated M68000 Family Bus Master/Slave Interface
  - — Four Channel DMA for Transfer of Data Frames to and from Memory
    - – 40-Byte FIFO to Efficiently Support High Data Rate
    - – 32-Bit Address Bus with Virtual Address Capabilities
- Simplified Interface to Other Processor Environments
  - — Byte Swapping Capability for Alternate Memory Structures
  - — 8- or 16-Bit Data Bus
- Low Power Consumption through 1.5 Micron HCMOS Fabrication

GENERAL DESCRIPTION

The TBC functions as an intelligent peripheral device to a microprocessor. An on-chip DMA transfers data frames to and from a buffer memory with minimal microprocessor interference required. A microcoded fully linked buffer management scheme queues frames during transmission and reception, and optimizes memory use. The TBC simplifies interfacing a microcomputer to a token bus network by providing the link layer services including: managing ordered access to the token bus medium, providing a means for admission and deletion of stations, and handling fault recovery. This allows the host to operate almost totally isolated from the task of ensuring error free transmission and reception of data.

The TBC can be used in a variety of machines in the factory from programmable controllers to large computers. Although token bus is especially well suited to the factory because of its deterministic characteristics, the TBC can be used in other networking applications such as office automation. The TBC provides the capability to swap the byte ordering of data to support alternate memory organizations. Additionally, the TBC is a full M68000 bus master, providing on-chip DMA capability for management of memory tables and frame buffers. Since the TBC bus interface is configurable, the TBC can handle both 8-bit and 16-bit data transfers. Figure 2 shows the TBC in a typical intelligent I/O processor system environment.

The following sections summarize the MC68824's operational modes, data structures, and commands. Refer to Table 1 for an explanation of the abbreviations used throughout this document.

INTERNAL ARCHITECTURE

The TBC has four functional blocks including: serial, DMA, microcoded controller, and register file/ALU. Each of these sections contain user visible and non-visible registers that define control and operation of the TBC. A block diagram of the MC68824 is shown in Figure 3.

Because the TBC communicates with the host primarily through shared memory, a minimum number of host processor accessible registers are required. These directly accessible registers include the command register, semaphore register, interrupt vector register, and data register. Internal registers not directly accessible to the host processor can be accessed through the initialization table in shared memory.

OPERATIONAL MODES

The MC68824 has three main operational modes, TBC mode, EPA mode, and Bus Analyzer mode. In the TBC mode, which is the default mode, the MC68824 provides a full MAC implementation; it only supports and does

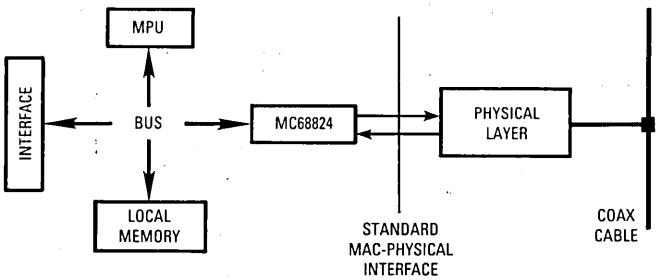


Figure 2. Token Bus LAN Node

Table 1. Abbreviations

Abbreviation	Explanation
ACM	Access Control Machine
BD	Buffer Descriptor
CRC	Cyclic Redundancy Check
DA	Destination Address
DB	Data Buffer
DMA	Direct Memory Access
DSAP	Destination Service Access Point
EOQ	End of Queue
EPA	Enhanced Performance Architecture
FC	Frame Control
FD	Frame Descriptor
HOQ	Head of Queue
IA	Individual Address

Abbreviation	Explanation
LLC	Logical Link Control
LSAP	Link Service Access Point
LSDU	Link Service Data Unit
MAC	Media Access Control
RDS	Reduced Data Structure
RWR	Request with Response
RX	Receive
SA	Source Address
SSAP	Source Service Access Point
TBC	Token Bus Controller
TS	This Station Address
TX	Transmit

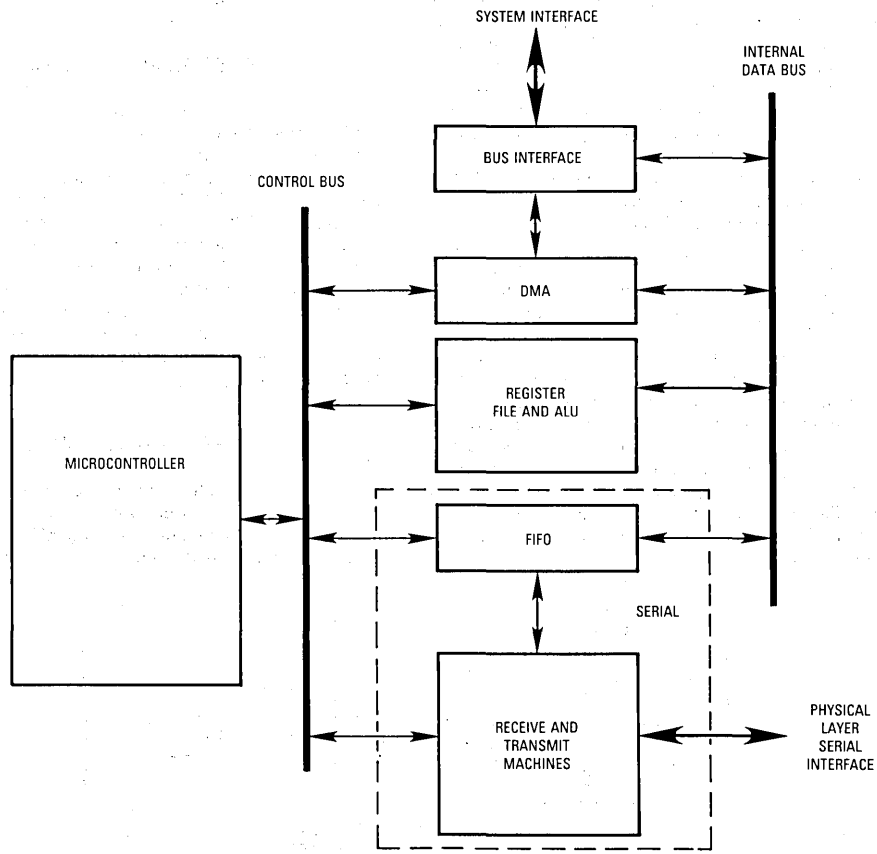


Figure 3. MC68824 Block Diagram

not implement LLC type 3. Running in the default mode allows the user to perform the LLC functions or equivalent in software. In the EPA mode, the MC68824 performs the receiver portion of the 'Acknowledged Connectionless Service' of the LLC type 3 sublayer, as well as operating normally when a non-request with response frame is received. In the Bus Analyzer mode, the MC68824 is not part of the logical ring. In this mode, the MC68824 receives all frames which makes the TBC an ideal chip to be used in a protocol analyzer application. The MC68824 also offers several additional modes and options which can be used to tailor the TBC to a specific application. The TBC/EPA mode is selected via the Mode Selector Word located in the initialization table. Several other options are available to the user by using the SET MODE commands. Table 2 summarizes all the modes available on the MC68824 while running in either the TBC mode or the EPA mode except where noted. Refer to Table 1 for the explanation of abbreviations.

Interrupt Status Words Definitions

Interrupt Status Word 1	Interrupt Status Word 2
TBC Command Complete	Duplicate MAC Address Detected
Frame Descriptor Pool Empty	Faulty Transmitter
Buffer Descriptor Pool Empty	Successor Changed
Transmit Queue Empty	No Successor/No Successor 1
Token Skipped	Unexpected Frame 6
Token Passed	Solicit Any Arc of the ACM Performed
Bus/Address Error	Unexpected Frame 10
Frame Descriptor Pool Low	Receive Claim Token
Buffer Descriptor Pool Low	No Response Received (ACM in the Await Response State)
Overrun	Win Address Sort
Underrun	Bus Idle Timer Expired
Transmitted RWR Frame	Threshold Counter Exceeded
Transmitted Response Frame	Lose Address Sort
Transmitted Data Frame	Modem Error
Received RWR Frame	
Received Data Frame	

SHARED MEMORY STRUCTURES

The TBC and host processor communicate through a memory structure which consists of two tables and a fully linked buffer structure. The initialization table allows the host processor to set and update the TBC operating parameters and table pointers, and to receive status and error information. The TBC is given a pointer to the initialization table during initialization. The private area is the other table which is used by the TBC to store MAC parameters. The fully linked buffer structure consists of frame descriptors, buffer descriptors, and data buffers. There is one frame descriptor for each frame. Each frame descriptor contains pointers to a buffer descriptor which in turn points to a data buffer where the message data is stored. If more than one data buffer is needed, the buffer descriptor will point to another buffer descriptor which in turn points to another data buffer.

INITIALIZATION TABLE

The initialization table format is shown in Table 3. The first 124 bytes (0-7C hex) initialize the TBC and the TBC's private area in memory. The command parameter area (CPA) is most frequently used by the host to set and read internal TBC parameters. The CPA is divided into the command area and the command return area. The commands dedicated to reading and setting internal TBC parameters are SET ONE WORD, SET TWO WORDS, and READ VALUE.

The TBC continuously updates the two Interrupt Status Words as status changes. The interrupt status word, combined with the interrupt status mask, determines whether an interrupt will be generated. If an interrupt condition occurs and the corresponding bit in the interrupt status mask is set the MC68824 will assert $\overline{IRQ}$. The host must, after determining the interrupting event, issue a CLEAR INTERRUPT STATUS command which will negate $\overline{IRQ}$ and clear the interrupt bit. The following lists the events which are updated in the interrupt status words by the TBC.

The initialization table also contains statistics about the operation of the network. The statistics are 16-bit wrap around counters. Every counter has a threshold variable which the TBC checks against. If the threshold is reached, the host may be notified through the interrupt status bit. The host may select to collect all statistics or may disable the collection of the two most frequent statistics which are number of tokens passed and number of tokens heard.

When the MC68824 is the EPA mode, the initialization table must be extended by 1024 words to accomodate the LSAP table. The LSAP table consists of 2×128 entries, one for each possible LSAP address, individual and group. Each of the first 128 entries represents an individual LSAP while each of the following 128 entries represents group LSAPs. Figure 4 shows the format of each entry in the LSAP table, while Table 4 shows the entries in the initialization table which are different when the MC68824 is in the EPA mode.

Note that if the MC68824 is running in the EPA mode and the user wishes to enable the flow control mode, the initialization table must be extended beyond the LSAP table by 512 words to accomodate the message counter table.

PRIVATE AREA

The MC68824 private area is a 128-byte area of RAM reserved for use by the TBC to store internal variables and statistical information associated with the MAC operation. During initialization, the host specifies the appropriate initial values of the private area parameters in the initialization table (see displacement 08 hex through 76 hex in the initialization table shown in Table 3). The private area should never be directly accessed by the host as this would not guarantee IEEE 802.4 operation. The parameters in the private area must only be set and read by the SET/READ VALUE commands. When the MC68824 is operating in the EPA mode, the private area must be doubled to 256 bytes to provide for extra storage to save protocol variables.

Table 2. MC68824 Modes and Options

Mode	Description	Selected By
EPA	Enables the TBC to implement LLC type 3 functions. The TBC mode is the default	Mode Selector Word
Reduced Data Structure	Optimizes the number of back to back frames the TBC can receive by modifying the frame structures in memory	Mode Selector Word
Copy all Data Frames	Determines whether the TBC copies to memory all non RWR data frames that are heard. Station may still be part of logical ring. Useful mode for bridge implementations. Valid only in Reduced Data Structure Mode.	Mode Selector Word
Control Frames Preference	Ensures that the TBC will not lose control frames if adequate system performance is provided	Mode Selector Word
No Mask Mode	Indicates to the TBC that only individually addressed frames with DA = TS exactly or all group addressed frames regardless of the mask will be received	Mode Selector Word
Receive Erroneous Frames in RDS	Allows the TBC to store erroneous frames while in the reduced data structure mode	Mode Selector Word
MAC Address Length	Allows the user to select 16- or 48-bit MAC addresses	Mode Selector Word
Bus Analyzer	Enables the bus analyzer mode	SET MODE 1
Copy Frames with Undefined FC	Allows user to store frames with undefined frame control.	SET MODE 1
Copy all Control Frames	May be used to set the TBC in promiscuous mode to monitor network traffic	SET MODE 1
Limited Statistics Tracking	TBC can keep track of all statistics or of a subset (see Initialization Table).	SET MODE 1
Response SA Filtering	Allows the TBC to selectively store response frames according to a mask on the SA while running in the EPA mode only	SET MODE 1
Lower Bridge Mode	Puts the TBC in lower bridge mode which is used in hierarchical bridging	SET MODE 2
In_ring Desired	Determines whether or not the TBC is a member of the logical ring while in the steady state condition	SET MODE 2
Source Routing Limited Broadcast	If source routing is enabled then broadcast frames can be recognized	SET MODE 2
Bridge Delay Mode	To be used in bridges when SA can be unequal to TS for long distance broadband networks.	SET MODE 2
Recognize Source Routing	Allows the TBC to act as part of a source routing bridge between interconnected networks	SET MODE 2
Copy CRC to Memory	Causes the TBC to copy the 4 byte CRC of data frames to memory as part of the data unit	SET MODE 3
Suppress CRC Generation (All Frames)	Disables CRC generation for all data frames transmitted by the TBC	SET MODE 3
Halt Generator Enable	Controls the maximum number of DMA transfers that the TBC performs in one DMA burst	SET MODE 3
Data Byte Swap Mode	Allows the user to select either the Motorola/IBM or the Intel/DEC data organization for data buffers	SET MODE 3
Prescaler	Determines whether a prescaler of 3 or 6 should be used for octet timers	SET MODE3
Flow Control	Enables the flow control algorithm to allow the user to selectively copy RWR frames to memory while in the EPA mode only	LSAP Only
Suppress CRC Generation (Per Frame)	Disables transmission of CRC on a frame by frame basis	Frame Only

Table 3. Initialization Table Format

Displacement (In Bytes)	Description of Field
00	Private Area Function Code
02	Private Area Pointer High
03	Private Area Pointer Low
05	Zero
08	Initial Hi_Priority_Token_Hold_Time
0A	Zero
0C	Zero
0E	Initial Target_Rotation_Time for Access Class 4
10	Zero
12	Initial Target_Rotation_Time for Access Class 2
14	Zero
16	Initial Target_Rotation_Time for Access Class 0
18	Zero
1A	Initial Target_Rotation_Time for Ring Maintenance
1C	Initial Ring Maintenance Time Initial Value
1E	Initial Source Segment/Bridge ID (SID)
20	Initial Target Segment/Bridge ID (TID)
22	Initial Segment Number Mask for Source Routing
24	Initial Max_Inter_Solicit_Count
26	Initial RX Frame Status Error Mask
28	Initial TX Queue Access Class 6 Status
2A	Initial TX Queue Access Class 6 HOQ Pointer
2E	Zero
30	Initial TX Queue Access Class 4 Status
32	Initial TX Queue Access Class 4 HOQ Pointer
36	Zero
38	Initial TX Queue Access Class 2 Status
3A	Initial TX Queue Access Class 2 HOQ Pointer
3E	Zero
40	Initial TX Queue Access Class 0 Status
42	Initial TX Queue Access Class 0 HOQ Pointer
46	Zero
48	Initial RX Queue Access Class 6 EOQ Pointer
4C	Initial RX Queue Access Class 4 EOQ Pointer
50	Initial RX Queue Access Class 2 EOQ Pointer
54	Initial RX Queue Access Class 0 EOQ Pointer
58	Initial Free Frame Descriptor Pool Pointer to First FD
5C	Initial Free Buffer Descriptor Pool Pointer to First BD
60	Initial Group Address Mask — Low
62	Initial Group Address Mask — Medium
64	Initial Group Address Mask — High
66	Initial Individual Address Mask — Low
68	Initial Individual Address Mask — Medium
6A	Initial Individual Address Mask — High
6C	Initial Non-RWR Maximum Retry Limit
6E	Initial RWR Maximum Retries Limit
70	Initial Slot Time
72	Initial This Station Address — Low
74	Initial This Station Address — Medium
76	Initial This Station Address — High
78	Initial Pad Timer Preset (PTP)
7A	Mode Selector Word
7C▲	Response SA Mask — Low
7E▲	Response SA Mask — Medium
80▲	Response SA Mask — High
82	Zero
84*	Response Destination Address Pointer
88*	Response Pointer
8C*	RWR Pointer

Displacement (In Bytes)	Description of Field
	COMMAND PARAMETER AREA
90	Command Parameter Area VAL0
92	Command Parameter Area VAL1
94	Command Parameter Area VAL2
96	Command Return Area RET0
98	Command Return Area RET1
9A	Command Return Area RET2
9C	Command Status and Done Bit
9E	Zero
A0	Zero
A2	Zero
A4	Zero
A6	Zero
	INTERRUPT STATUS AREA
A8	Interrupt Status Word 0
AA	Interrupt Mask 0
AC	Interrupt Status Word 1
AE	Interrupt Mask 1
	STATISTICS
B0	Number of Tokens Passed (Threshold)
B2	Number of Tokens Passed
B4	Number of Tokens Heard (Threshold)
B6	Number of Tokens Heard
B8	Number of Tokens Passed Through No_Successor_8 Arcs (Threshold)
BA	Number of Tokens Passed Through No_Successor_8 Arcs
BC	Number of Who_Follows Transmitted (Threshold)
BE	Number of Who_Follows Transmitted
C0	Number of Token Passes That Failed (Threshold)
C2	Number of Token Passes That Failed
C4	Number of Non-Silence (Threshold)
C6	Number of Non-Silence
C8	Number of FCS Errors (Threshold)
CA	Number of FCS Errors
CC	Number of E-Bit Errors (Threshold)
CE	Number of E-Bit Errors
D0	Number of Frame Fragments (Threshold)
D2	Number of Frame Fragments
D4	Number of Frames Too Long (>8K Bytes) (Threshold)
D6	Number of Frames Too Long (>8K Bytes)
D8	Number of No FD/BD Errors (Threshold)
DA	Number of No FD/BD Errors
DC	Number of Overruns (Threshold)
DE	Number of Overruns
	DMA DUMP AREA
E0	FC1
E2	DPTR1
E6	FC2
E8	DPTR2
EC	FC3
EE	DPTR3
F2	FC4
F4	DPTR4
F8-FE	Zero

*See Table 4.
▲In Response SA Filtering Mode Only

7

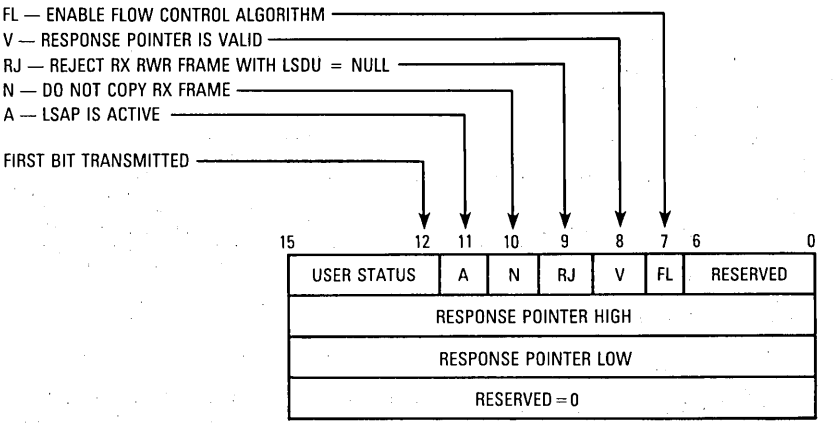


Figure 4. LSAP Table Entry

Table 4. Initialization Table Entries in EPA Mode

Displacement in Hex Bytes	Description of Field
84	Not Used
86	Not Used
88	RX Retry RWR Frame – Threshold
8A	RX Retry RWR Frame – Counter
8C	RX RWR with LSDU = Null – Threshold
8E	RX RWR with LSDU = Null – Counter

7

LINKED BUFFER STRUCTURES

The fully linked buffer structures include frame descriptors (FD), buffer descriptors (BD), and data buffers (DB). One frame descriptor is required per received or transmitted frame. Each frame descriptor contains information pertaining both to the frame sent or received plus a pointer to the next FD as well as a pointer to its first BD. Buffer descriptors contain the pointer to a data buffer as well as that buffer's attributes, and a pointer to the next buffer descriptor in the frame if used. The data buffers are used to store message data; one data buffer is associated with each buffer descriptor. Figure 5 illustrates the linked buffer structure.

To fully support the IEEE 802.4 message priorities, the TBC provides four transmit queues and four receive queues. Before transmission of a message, the host processor creates frame descriptors, buffer descriptors, and data buffers for that message and then links the frame descriptors to the appropriate transmit queue. Transmission queues may be enabled or disabled using the SET ONE WORD command. The TBC confirms transmission of the frames in each frame descriptor as they are sent out. During reception, the TBC reverses the process to use frame descriptors and buffer descriptors from the pre-linked free frame descriptors pool and free buffer descriptors pool as frames are received, assigning these frames to the proper reception queue. Receive queues may not be disabled. The free frame descriptor and buffer descriptor pool pointers, which are located in the private area, must be valid at initialization time and may be changed thereafter using the SET TWO WORDS command while the TBC is OFFLINE. Finally, the host processor removes the frame data from these reception queues as programmed. Figure 6 illustrates the linking between the queues, FDs, BDs, and DBs.

If the MC68824 is programmed to run in the Reduced Data Structure mode, a separate data structure is used for storing received frames while the TX frame queue structure remains the same as previously explained. In the Reduced Data Structure, there is only one receive queue; frames of all priority classes are copied to the same receive queue.

COMMAND SET

The host processor issues commands to the TBC to perform various functions by writing to the TBC command register. The commands fall into seven categories and are listed in Table 5.

INITIALIZATION COMMANDS

Initialization commands configure the TBC for operation after a hardware or software reset. The five initialization commands specify various system attributes and the location of the initialization table in memory.

RESET Command

Both the RESET command and hardware reset cause the TBC to perform a reset operation.

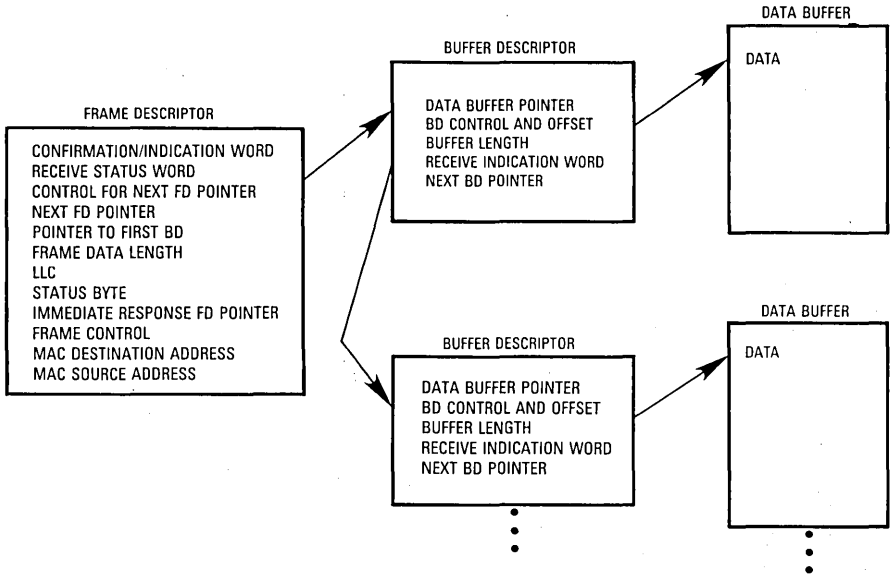


Figure 5. Linked Buffer Structures

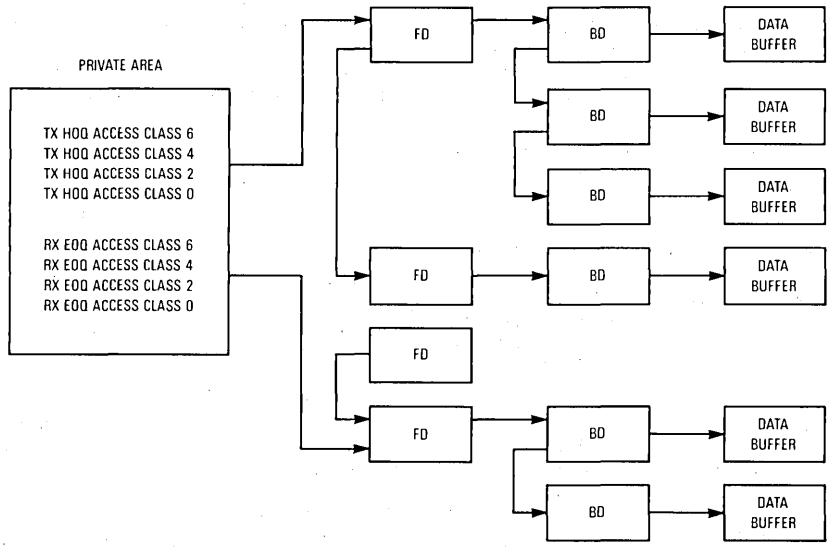


Figure 6. MC68824 Queues

Table 5. MC68824 Commands by Categories

Initialization LOAD INITIAL TABLE FUNCTION CODE INITIALIZE OFFLINE IDLE RESET	**TEST** INTERNAL/EXTERNAL LOOPBACK MODE RECEIVER TEST TRANSMITTER TEST HOST INTERFACE TEST FULL DUPLEX LOOPBACK TEST SELF TEST MEASURE SLOT TIME
SET OPERATION MODE SET MODE 0 SET MODE 1 SET MODE 2 SET MODE 3 SET/CLEAR IN_RING_DESIRED	**NOTIFY TBC** CLEAR INTERRUPT STATUS RESPONSE READY SET RESPONSE
TX DATA FRAMES STOP RESTART START	**MODEM CONTROL** PHYSICAL END PHYSICAL
SET/READ VALUE READ VALUE SET FUNCTION CODE OF BUFFER DESCRIPTORS SET FUNCTION CODE OF FRAME DESCRIPTORS SET FUNCTION CODE OF RX AND TX DATA BUFFERS SET PAD TIMER PRESET (PTP) REGISTER SET ONE WORD SET TWO WORDS	UPDATE LSAP STATUS UPDATE LSAP COUNTERS RESET TRT

OFFLINE Command

The OFFLINE command causes the TBC to transition to the offline state. The OFFLINE command is typically used to end an internal diagnostic test on the TBC.

7

IDLE Command

The IDLE command causes the TBC to transition from the offline state to the idle state. The IDLE command is used after initialization is complete.

LOAD INITIALIZATION TABLE FUNCTION CODE Command

The LOAD INITIALIZATION TABLE FUNCTION CODE writes the initialization table function code value in the data register into the TBC and sets the bus width to 8 or 16 bits.

INITIALIZE Command

The INITIALIZE command loads the initialization table pointer from the data register, loads initial values from the initialization table into the TBC and the TBC private area. This command should only be given while in the offline state after reset.

SET OPERATION MODE

The five commands in the set operation mode category are used to set various operation modes and options in the TBC. Defaults are off except for the Halt Generator Enable bit which comes up set after a reset.

SET MODE 0 Command

The SET MODE 0 command allows the user to dynamically change the No Mask mode, the Control Frames Preference mode, and the ability to receive erroneous frames while in the Reduced Data Structure mode. See Table 2 for a description of these modes.

SET MODE 1, 2, AND 3 Commands

The SET MODE 1, 2, and 3 are commands used at initialization time to set up various options of the MC68824. See Table 2 for a description of these modes.

SET/CLEAR IN_RING_DESIRED Command

The SET/CLEAR IN_RING DESIRED command is used to change the value of the IEEE 802.4 boolean in_ring_desired during normal operation.

TRANSMIT DATA FRAMES

Transmit data frames commands are used to stop/restart transmission of data frames or to start an empty transmission queue.

STOP Command

The STOP command suspends transmission of data frames by the TBC.

RESTART Command

The RESTART command restarts transmission of data frames by the TBC after transmission has been stopped via the STOP command.

START Command

The START command is used by the host processor whenever new frame(s) is (are) added to an empty transmit queue. Before issuing this command, the command parameter area must contain the code of the appropriate transmit queue status, and the next two words must contain the pointer to the new frame(s).

SET/READ VALUE

SET/READ VALUE commands are used by the host processor to set and read TBC parameters. The parameters which may be modified include function codes, pad timer preset register, and some of the parameters which reside in the private area or initialization table.

READ VALUE Command

The read value command reads internal TBC parameters and statistical information. The opcode of the parameter to be read is placed into the first word of the command parameter area, the read value command is issued, and the TBC returns the value of the parameter in the command return area in the initialization table.

SET ONE WORD/TWO WORDS Command

The SET ONE WORD/TWO WORDS commands allow the user to set a number of MAC parameters and TBC pointers. The opcode of the parameter to be set must be placed in the first word of the command parameter area with the value in the next three words.

SET PTP Command

This command allows the user to set the pad timer preset register which is used by the MC68824 to set the length and pattern of the preamble and the minimum number of preamble octets transmitted between frames.

SET FUNCTION CODES Command

There are three commands available to the user to set the function codes. The MC68824 utilizes function codes to access the buffer descriptors, frame descriptors, and RX and TX data buffers. If function codes are not used, these commands may be ignored.

UPDATE LSAP STATUS Command

The UPDATE LSAP STATUS command can be used to activate, deactivate, or update user status when the TBC is online. This command can only be issued when the MC68824 is in the EPA mode.

UPDATE LSAP COUNTERS Command

The UPDATE LSAP COUNTERS command allows the user to set the flow control counters to new values without disabling the specific LSAP. This command can only be issued when the MC68824 is in the EPA mode and with the Flow Control mode enabled.

RESET TRT Command

The RESET TRT command when issued causes the MC68824 to reset the internal token rotation timer to zero. Since the last value of the token rotation timer is also returned to the host, this command may be used to make time calculations. This command must only be issued when the MC68824 is in the Bus Analyzer mode.

TEST

Test commands are used to test the interface from the host to the TBC, the transmitter, the receiver, and the serial section as well as the internal sections of the TBC.

SET INTERNAL/EXTERNAL LOOPBACK MODE Command

This command determines whether the TBC is in internal or external loopback mode while running the receiver, transmitter, and full duplex loopback tests.

Tests

There are five available tests which can be run on the TBC while in the offline state. The five tests are HOST INTERFACE TEST, RECEIVER TEST, TRANSMITTER TEST, FULL DUPLEX LOOPBACK TEST, and SELF TEST.

MEASURE SLOT TIME Command

The MEASURE SLOT TIME command is used when the MC68824 is offline to enable the user to monitor the length of a response window opened by a station already in the ring. Upon issuance of this command, the MC68824 waits until the first solicit_successor_1 frame is heard. Then, the time is measured until a successive token is heard.

NOTIFY TBC

The seven commands in this category are used to notify the TBC to perform an action dynamically.

CLEAR INTERRUPT STATUS Command

The CLEAR INTERRUPT STATUS command resets the interrupt request signal and clears specific status bits in the status words.

RESPONSE READY Command

The RESPONSE READY command notifies the TBC that the host has completed preparing a response frame for the TBC in answer to a request with response frame. This command is only valid when not in predefined response mode.

SET RESPONSE Command

The SET RESPONSE command is used to notify the MC68824 that an updated response frame is to be sent out at a later time when requested by another station. The SET RESPONSE command performs a function equivalent to L_REPLY_UPDATE defined in IEEE 802.2. This command can only be issued when the MC68824 is in the EPA mode.

MODEM CONTROL

Two commands are provided that allow the TBC to provide management services to the physical layer of the node.

PHYSICAL Command

The PHYSICAL command is used by the host processor to control the physical layer. This command allows the TBC to either pass commands or send serial data to the physical layer while in station management. Status or data is returned in the last word in the command return area.

END PHYSICAL Command

The END PHYSICAL command removes the TBC from the station management mode. The TBC negates $\overline{\text{SMREQ}}$ and waits for the modem to negate $\overline{\text{SMIND}}$ before setting the command confirmation bit.

TRANSMISSION OF A FRAME

Data frames are transmitted from one of four transmission priority queues. The host processor passes the pointer from the start of the queue to be transmitted to the TBC using the start command. Upon reception of the token by the station, the TBC checks the transmission queues for frames to be sent. If frames are present in one or more of the queues, the TBC will send frames based on their priorities until its allotted transmission time expires. At the end of the frame transmission, the TBC writes into the frame descriptor the status of the frame. The TBC may then generate an interrupt request according to the frame type (non-RWR, RWR, response), interrupt status, and the interrupt mask. The host processor services the interrupt and checks for the status of the frame, i.e., if transmission was successful or not. The TBC can transmit frames with lengths of up to 64K bytes. The TBC does not check if the frame is longer than 8K bytes. (The IEEE standard specifies frame lengths of up to 8K bytes.) The TBC does not check if the frame control, destination address, and source address are correct but takes them from the frame descriptor as they are. This means that the host must be sure to write them correctly.

RECEPTION OF A FRAME

If the incoming frame's destination address matches the individual station's address, the individual station's group address, or the broadcast address, then the TBC will accept the frame. The TBC can accept frames with undefined frame control field and data frames, can treat control frames as data frames, and can write the CRC of a received frame into the data buffer as defined by the SET MODE commands. The CRC is always calculated and compared with the CRC of the frame. When a frame is accepted, the frame is placed into the appropriate RX priority queue in memory, and its frame descriptor is linked to the last frame descriptor in the queue. If an error occurred on frame reception, the RX status error mask (set by the user initialization) in the TBC private area determines whether to accept or reject the frame. The TBC may generate interrupts upon detecting error conditions. A normal interrupt may also be generated according to the frame's type (RWR or request with no response), the interrupt status, and the interrupt mask.

LLC TYPE 3 PROTOCOL IMPLEMENTATION

While running in the EPA mode the MC68824 provides the user with the IEEE 802.2 Acknowledged Connectionless Service primitives described in Table 6. These services provide the means at the data link level to exchange link service data units point to point which have been acknowledged at the LLC sublayer without establishing a data link connection. Upon receiving an RWR frame, the MC68824, if in the EPA mode will first check if data is requested in the response. If data is not requested, the MC68824 sends an ACK as a response to acknowledge the RWR frame's reception. If data is requested in the response, then the MC68824 checks for a response associated with the specific DSAP of the request. If the LSAP is active and the response is ready, the response is sent back to the requester with the appropriate status; otherwise, a NACK is sent.

Table 6. LLC Type 3 Primitives Provided by the MC68824

Acknowledged Connectionless Data Unit Transmission Service	
Primitive	**Description**
L_DATA_ACK.request	Used to send a data unit with acknowledgement to another node
L_DATA_ACK.indication	Used to indicate the reception of a non-null, non-duplicate LSDU from a remote data link node
L_DATA_ACK_STATUS.indication	Used to indicate whether the previous LSDU transmission request was successful
Acknowledged Connectionless Data Unit Exchange Service	
Primitive	**Description**
L_REPLY.request	Used to request an acknowledged connectionless data unit exchange
L_REPLY.indication	Used to indicate reception of an acknowledged connectionless data frame
L_REPLY_STATUS.indication	Used to confirm acknowledged connectionless data frames
Reply Data Unit Preparation	
Primitive	**Description**
L_REPLY_UPDATE.request	Used to indicate the need to update a response
L_REPLY_UPDATE_STATUS.indication	Used to confirm that a response was updated

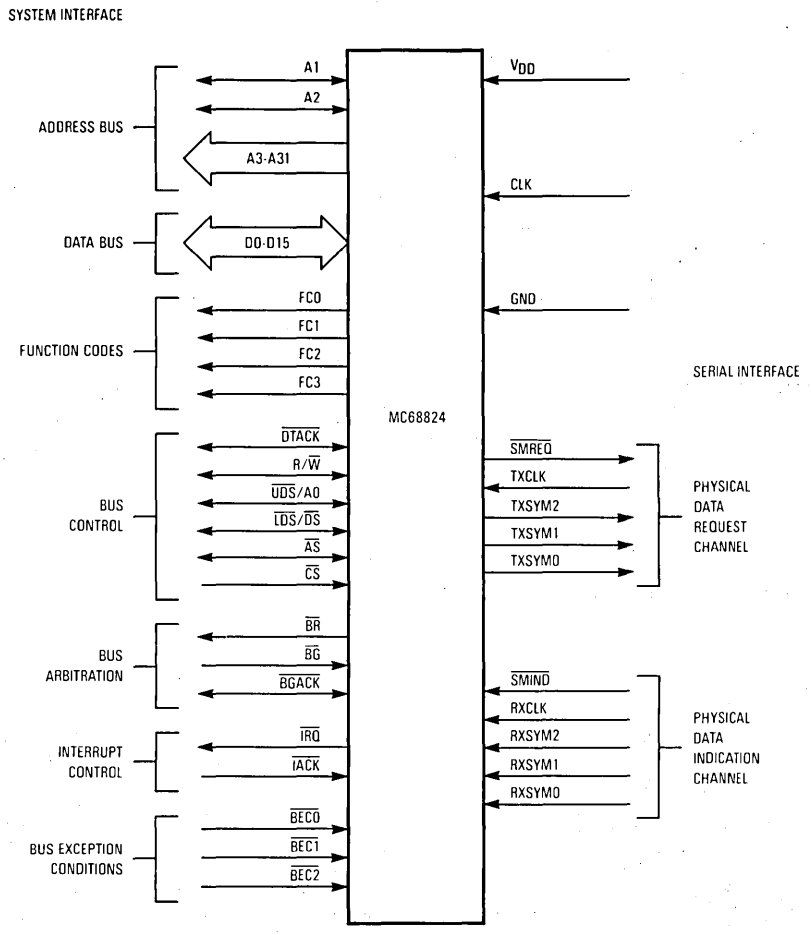


Figure 7. MC68824 Signals

7

ELECTRICAL SPECIFICATIONS

This sections contains the electrical specifications and associated timing information for the MC68824. See Figure 7 for a diagram of the MC68824 signals.

MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Supply Voltage	V_{DD}	−0.3 to +7.0	V
Input Voltage	V_{in}	−0.3 to +7.0	V
Operating Temperature Range MC68824 MC68824I	T_A	 0 to 70 0 to 85	°C
Storage Temperature Range	T_{stg}	−55 to +150	°C

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

THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Unit
Thermal Resistance for PGA	θ_{JA}	33	°C/W

$T_J = T_A + (P_D \bullet \theta_{JA})$
$P_D = (V_{DD} \bullet I_{DD}) + P_{I/O}$
where:
$P_{I/O}$ is the power dissipation on pins (user determined) which can be neglected in most cases.
For $T_A = 70°C$ and $P_D = 0.55$ W @ 12.5 MHz
$T_J = 88°C$

POWER CONSIDERATIONS

The average chip-junction temperature, T_J, in °C can be obtained from:

$$T_J = T_A + (P_D \bullet \theta_{JA}) \quad (1)$$

where:
T_A = Ambient Temperature, °C
θ_{JA} = Package Thermal Resistance, Junction-to-Ambient, °C/W
P_D = $P_{INT} + P_{PORT}$
P_{INT} = $I_{DD} \times V_{DD}$, Watts — Chip Internal Power
$P_{I/O}$ = Power Dissipation on Input and Output Pins, Watts — User Determined

For most applications $P_{I/O} < P_{INT}$ and can be neglected.
If $P_{I/O}$ is neglected, an approximate relationship between P_D and T_J is:

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

Solving equations (1) and (2) for K gives:

$$K = P_D \bullet (T_A + 273°C) + \theta_{JA} \bullet P_D^2 \quad (3)$$

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.

DC ELECTRICAL CHARACTERISTICS

All specifications are valid under the following conditions: V_{DD} = 4.75 V to 5.25 V, V_{SS} = 0 V, $T_A = T_L$ to T_H and 130 pF total capacitance on output pins.

Characteristic		Symbol	Min	Max	Unit
Input High Voltage (Except System Clock)		V_{IH}	2.0	V_{DD}	V
Input Low Voltage (Except System Clock)		V_{IL}	$V_{SS}-0.3$	0.8	V
Input High Voltage (System Clock)		V_{CIH}	2.4	V_{DD}	V
Input Low Voltage (System Clock)		V_{CIL}	$V_{SS}-0.3$	0.5	V
Input Leakage Current @5.25 V		I_{in}	—	20	μA
Input Capacitance (V_{in} = 0 V, T_A = 25°C, F = 1 MHz)		C_{in}	—	13	pF
Three-State Leakage Current @2.4/0.5 V		I_{TSI}	—	20	μA
Open-Drain Leakage Current @2.4 V		I_{OD}	—	20	μA
Output High Voltage		V_{OH}			V
(I_{OH} = 1 mA)	$\overline{SMREQ}$, TXSYM2, TXSYM1, TXSYM0		2.5	—	
(I_{OH} = 400 μA)	All Others		2.4	—	
Output Low Voltage		V_{OL}			V
(I_{OL} = 8.0 mA)	$\overline{SMREQ}$, TXSYM2, TXSYM1, TXSYM0		—	0.5	
(I_{OL} = 3.2 mA)	A1-A31, FC0-FC3, and $\overline{UDS}$/A0 as A0		—	0.5	
(I_{OL} = 5.3 mA)	D0-D15, $\overline{AS}$, $\overline{LDS}$/$\overline{DS}$, $\overline{UDS}$/A0 as $\overline{UDS}$, $\overline{DTACK}$, $\overline{BGACK}$, R/$\overline{W}$		—	0.5	
(I_{OL} = 8.9 mA)	$\overline{IRQ}$, $\overline{BR}$		—	0.5	
Power Dissipation	@ 10 MHz, 0°C	P_D	—	0.50	W
	@ 12.5 MHz, 0°C			0.55	
	@ 16.67 MHz, 0°C			0.70	

AC ELECTRICAL CHARACTERISTICS

High and low outputs are measured at 2.0 V minimum and 0.8 V maximum respectively, except $\overline{SMREQ}$ and TXSYM0-2 which are measured from 2.5 V and 0.5 V. High and low inputs are driven to 2.4 V and 0.5 V respectively for AC test purposes. However, input specifications are still measured from 2.0 V to 0.8 V. All specifications are valid under the following conditions: (V_{DD} = 4.75 V to 5.25 V, V_{SS} = 0 V, $T_A = T_L$ to T_H, output load = 130 pF, and output current as specified in **8.4 DC ELECTRICAL CHARACTERISTICS**)

Num.	Characteristic	10 MHz		12.5 MHz		16.67 MHz		Unit
		Min	Max	Min	Max	Min	Max	
1	Asynchronous Input Setup Time	20	—	20	—	10	—	ns
2	$\overline{UDS}$, $\overline{LDS}$ Inactive to $\overline{CS}$, $\overline{IACK}$ Inactive	—	100	—	80	—	60	ns
3	CLK Low (On Which $\overline{UDS}$ or $\overline{LDS}$ and $\overline{CS}$ or $\overline{IACK}$ are Recognized) to Data-Out Valid (see Note 5)	— —	1/2 +150	— —	1/2 +120	— —	1/2 +90	Clk. Per. ns
4	$\overline{CS}$ or $\overline{IACK}$ High to Data-Out High-Impedance	—	60	—	50	—	35	ns
5	$\overline{LDS}$/$\overline{DS}$ High to Data-Out Hold Time (see Note 6)	0	—	0	—	0	—	ns
6	$\overline{IACK}$ or $\overline{CS}$ Low to $\overline{DTACK}$ High (Driving Three-State $\overline{DTACK}$ High)	—	80	—	70	—	60	ns
7	CLK Low (On Which $\overline{UDS}$ or $\overline{LDS}$ and $\overline{CS}$ or $\overline{IACK}$ are Recognized) to $\overline{DTACK}$ Low (see Note 5)	— —	2 +90	— —	2 +80	— —	2 +60	Clk. Per. ns
8	CLK Low to $\overline{DTACK}$ Low	—	90	—	80	—	50	ns
9	Data-Out Valid to $\overline{DTACK}$ Low	20	—	20	—	20	—	ns
10	$\overline{DTACK}$ Low to $\overline{UDS}$, $\overline{LDS}$, $\overline{CS}$, $\overline{IACK}$ High (Earliest)	100	—	80	—	60	—	ns
11	$\overline{CS}$ or $\overline{IACK}$ or Data Strobes (The Earliest) High to $\overline{DTACK}$ High (see Note 7)	—	60	—	50	—	40	ns
12	$\overline{DTACK}$ High to $\overline{DTACK}$ High Impedance (At End of Bus Cycle)	—	50	—	50	—	40	ns

— CONTINUED —

AC ELECTRICAL CHARACTERISTICS (Continued)

Num.	Characteristic	10 MHz		12.5 MHz		16.67 MHz		Unit
		Min	Max	Min	Max	Min	Max	
13	$\overline{UDS}$, $\overline{LDS}$ Inactive Time	1	—	1	—	1	—	Clk. Per.
14	$\overline{CS}$, $\overline{IACK}$ Inactive Time	0	—	0	—	0	—	ns
15	A1-A2 Valid to $\overline{UDS}$, $\overline{LDS}$, $\overline{CS}$ (The Latest One) Low (Write)	30	—	20	—	20	—	ns
16	$\overline{DTACK}$ Low to Data and A1-A2 Hold Time	100	—	80	—	60	—	ns
17	$\overline{UDS}$ or $\overline{LDS}$, $\overline{CS}$ or $\overline{IACK}$ (The Latest One) Low to Data-In Valid	—	80	—	70	—	60	ns
18	R/$\overline{W}$ Valid to $\overline{UDS}$, or $\overline{LDS}$, $\overline{CS}$ or $\overline{IACK}$ (The Latest One) Low	20	—	20	—	10	—	ns
19	$\overline{UDS}$, $\overline{LDS}$ High to R/$\overline{W}$ High	0	—	0	—	0	—	ns
20	CLK High to $\overline{IRQ}$ Low	—	100	—	80	—	60	ns
21	Reserved							
22	Reserved							
23	CLK High to $\overline{BR}$ Low	—	60	—	55	—	40	ns
24	CLK High to $\overline{BR}$ High Impedance	—	55	—	50	—	40	ns
25	$\overline{BGACK}$ Low to $\overline{BR}$ High Impedance	20	—	20	—	10	—	ns
26	$\overline{BG}$ Active/Inactive to CLK Low Setup Time	20	—	20	—	10	—	ns
27	CLK Low to $\overline{BGACK}$ Low	—	60	—	55	—	40	ns
28	CLK High to $\overline{BGACK}$ High Impedance	—	45	—	40	—	30	ns
29	$\overline{AS}$ and $\overline{BGACK}$ High, the Latest One, to $\overline{BGACK}$ Low (when $\overline{BG}$ is Previously Asserted)	2 +20	3 +80	2 +20	3 +70	2 +10	3 +50	Clk. Per. ns
30	$\overline{BG}$ Low to $\overline{BGACK}$ Low (No Other Bus Master)	2 +20	3 +80	2 +20	3 +70	2 +10	3 +50	Clk. Per. ns
31	$\overline{BR}$ High Impedance to $\overline{BG}$ High	0	—	0	—	0	—	ns
32	Clock on which $\overline{BGACK}$ Low to Clock on which $\overline{AS}$ Low	1.5	1.5	1.5	1.5	1.5	1.5	Clk. Per.
33	Clock Low to $\overline{BGACK}$ High	—	55	—	50	—	40	ns
34	CLK on which $\overline{BR}$ Low to CLK on which $\overline{BGACK}$ Low (Assuming that $\overline{BG}$ is Active and $\overline{BGACK}$ and $\overline{AS}$ are Inactive for at Least 2 CLK Periods)	1.5	1.5	1.5	1.5	1.5	1.5	Clk. Per.
35	CLK on which $\overline{AS}$ is High to CLK on which $\overline{BGACK}$ is High	—	1	—	1	—	1	Clk. Per.
36	CLK High to Address Valid	—	100	—	80	—	60	ns
37	CLK High to Address/FC High Impedance	—	70	—	60	—	50	ns
38	CLK High to FC Valid	—	60	—	55	—	50	ns
39	Address Valid to $\overline{AS}$ Valid	20	—	15	—	10	—	ns
40	CLK High to $\overline{AS}$, $\overline{UDS}$, $\overline{LDS}$ Low	—	50	—	40	—	30	ns
41	CLK to $\overline{AS}$, $\overline{UDS}$, $\overline{LDS}$ High	—	55	—	50	—	45	ns
42	$\overline{AS}$ High to Address/FC Invalid	20	—	10	—	10	—	ns
43	CLK High to $\overline{AS}$, $\overline{UDS}$, $\overline{LDS}$ High Impedance	—	70	—	60	—	45	ns
44	CLK to R/$\overline{W}$ High (see Note 4)	—	55	—	50	—	45	ns
45	CLK Low to R/$\overline{W}$ High Impedance	—	70	—	60	—	45	ns
46	$\overline{UDS}$, $\overline{LDS}$ High to Data-In Invalid	0	—	0	—	0	—	ns
47	$\overline{AS}$, $\overline{UDS}$, $\overline{LDS}$ High to $\overline{DTACK}$ High (Earliest of $\overline{AS}$, $\overline{UDS}$, or $\overline{LDS}$)	0	100	0	90	0	60	ns
48	Data-In to CLK Low Setup Time Required when $\overline{DTACK}$ Satisfies (1) (see Note 1)	10	—	10	—	5	—	ns

— CONTINUED —

AC ELECTRICAL CHARACTERISTICS (Continued)

Num.	Characteristic	10 MHz		12.5 MHz		16.67 MHz		Unit
		Min	Max	Min	Max	Min	Max	
49	$\overline{DTACK}$ Low to Data-In Valid Required when $\overline{DTACK}$ does not Satisfy (1) (see Note 2)	—	65	—	50	—	40	ns
50	CLK High to R/$\overline{W}$ Low	—	60	—	55	—	40	ns
51	$\overline{AS}$ Low to Data-Out Valid (Write)	—	90	—	80	—	60	ns
52	CLK Low to Data-Out Valid	—	55	—	55	—	40	ns
53	Data-Out Valid to $\overline{UDS}$, $\overline{LDS}$ Low	20	—	15	—	10	—	ns
54	$\overline{UDS}$, $\overline{LDS}$ High to Data-Out Invalid	20	—	15	—	10	—	ns
55	CLK High to Data-Out Hold Time	0	100	0	80	0	60	ns
56	No Exception to $\overline{BR}$ ($\overline{DTACK}$ Active)	1.5 +20	2.5 +80	1.5 +20	2.5 +70	1.5 +10	2.5 +50	Clk. Per. ns
57	$\overline{DTACK}$ Low to Asynchronous Exception Active Required when $\overline{DTACK}$ Does Not Satisfy (1) (see Note 2)	—	55	—	35	—	30	ns
58	Exception Active to CLK Low Setup Time Synchronous Input ("Late Exception") Required when $\overline{DTACK}$ Satisfies (1) (see Note 1)	45	—	45	—	20	—	ns
59	Exception Active to CLK Low Setup Time Asynchronous Input (Required when $\overline{DTACK}$ is Absent) (see Note 3)	20	—	20	—	10	—	ns
60	$\overline{AS}$, $\overline{UDS}$, $\overline{LDS}$ High to Exception Inactive	0	—	0	—	0	—	ns
61	Exception Inactive to CLK Low Setup Time (for Identification of No Exception)	20	—	20	—	10	—	ns
62	No Exception to $\overline{BR}$ ($\overline{DTACK}$ Inactive)	2.5 +20	3.5 +80	2.5 +20	3.5 +70	2.5 +10	3.5 +50	Clk. Per. ns
63	$\overline{RESET}$ (on $\overline{BEC0}$-$\overline{BEC2}$) Width	10	—	10	—	10	—	Clk. Per.
64	CLK Frequency	4	10	4	12.5	8	16.67	MHz
65	CLK Period	100	250	80	250	60	125	ns
66	CLK Width High (see Note 8)	45	125	35	125	25	62.5	ns
67	CLK Rise/Fall Time (see Note 8)	—	10	—	5	—	5	ns
68	CLK Width Low (see Note 8)	45	125	35	125	25	62.5	ns
69*	RXCLK, TXCLK Frequency (see Note 9)	1	10	1	12.5	5	16.67	MHz
70	RXD Signals Setup Time	35	—	35	—	25	—	ns
71	RXD Signals Hold Time	5	—	5	—	5	—	ns
72	RXCLK, TXCLK Rise/Fall Time	—	10	—	10	—	5	ns
73*	RXCLK, TXCLK Width Low	40	525	30	525	25	100	ns
74*	RXCLK, TXCLK Width High	40	525	30	525	25	100	ns
75*	RXCLK, TXCLK Period	95	1050	80	1050	60	200	ns
76	TXCLK High to TXD Signals Output Delay	5	55	5	55	5	45	ns

*Parts with an S suffix have the following characteristics:

69	RXCLK, TXCLK Frequency	.01	10	.01	12.5	—	—	MHz
73	RXCLK, TXCLK Width Low	40	50,000	30	50,000	—	—	ns
74	RXCLK, TXCLK Width High	40	50,000	30	50,000	—	—	ns
75	RXCLK, TXCLK Period	95	100,000	80	100,000	—	—	ns

— CONTINUED —

AC ELECTRICAL CHARACTERISTICS (Concluded)

NOTES:

1. If $\overline{DTACK}$ satisfies the asynchronous setup time (1), then (48) is required for the data-in setup time and (58) for the synchronous exception setup time. Erroneous behavior may occur if (58) is not satisfied.
2. If $\overline{DTACK}$ does not satisfy (1), then (49) is required for data-in and (57) for the exception. Erroneous behaviour may occur if (57) is not satisfied.
3. Active exception when $\overline{DTACK}$ is absent must satisfy the asynchronous setup time (59).
4. R/$\overline{W}$ rises on the end of a write cycle (i.e., on the phase following S7). If the TBC relinquishes the bus, then R/$\overline{W}$ is three-stated one phase later. When the TBC takes the bus, R/$\overline{W}$ is three-stated until S1 and rises on that phase.
5. Data (3) and $\overline{DTACK}$ (7) will be timed from the earliest clock on which $\overline{CS}$ and either data strobe are recognized during an MPU cycle. Data (3) and $\overline{DTACK}$ will be timed from the earliest clock on which $\overline{IACK}$ and either data strobe during an $\overline{IACK}$ cycle.
6. If $\overline{CS}$ or $\overline{IACK}$ is negated before $\overline{UDS}$/$\overline{LDS}$, the data bus will be three-stated (4), possibly before $\overline{UDS}$/$\overline{LDS}$ negation.
7. If an 8-bit bus is used only $\overline{LDS}$ need be considered. If a 16-bit bus is used, both $\overline{UDS}$ and $\overline{LDS}$ must negate to apply to this specification.
8. The clock signal used during test has 5 ns of rise time and 5 ns of fall time. For system implementations that have less clock rise and fall time, the clock pulse minimum should be commensurately wider such that:
   1. System (TCL + (TCR + TCF) ÷ 2) ≥ (minimum TCYC) ÷ 2
   2. System (TCH + (TCR + TCF) ÷ 2) ≥ (minimum TCYC) ÷ 2
9. For performance reasons, it is required that a frequency ratio of systems clock to serial clock be greater than 1:1.

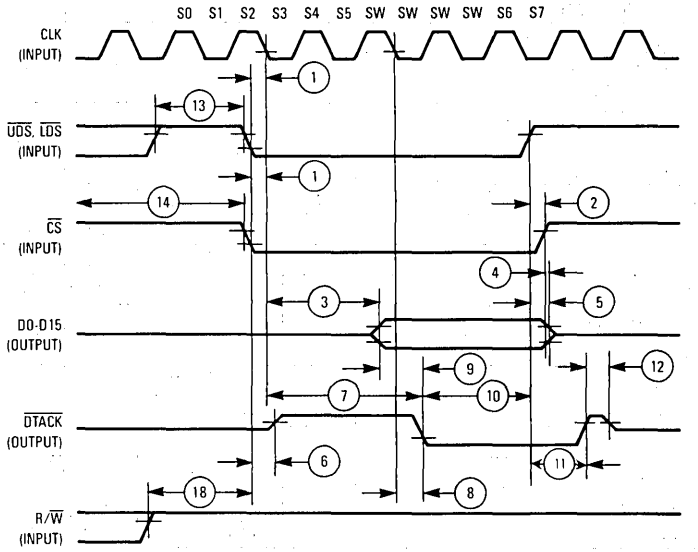


Figure 8. Host Processor Read Cycle

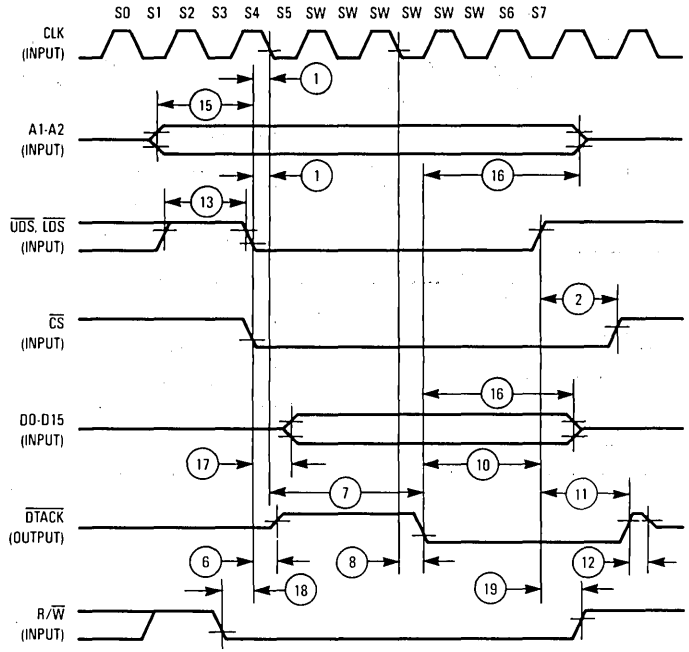


Figure 9. Host Processor Write Cycle

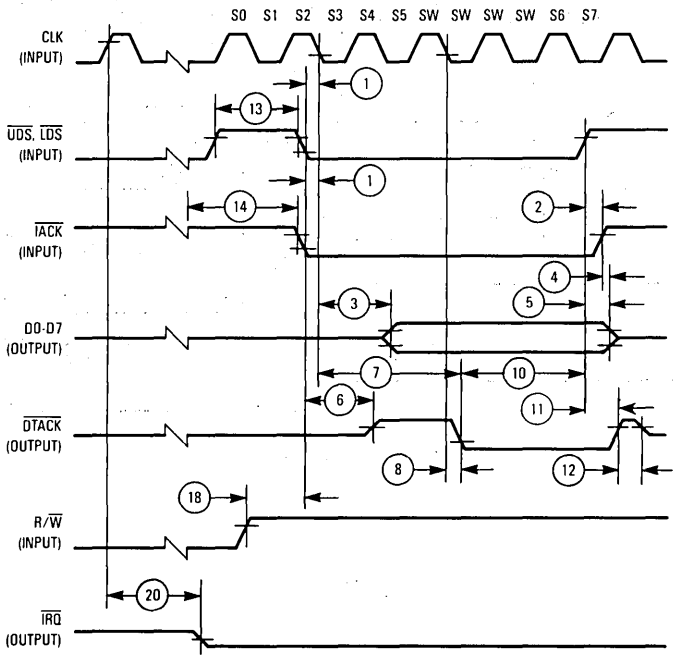
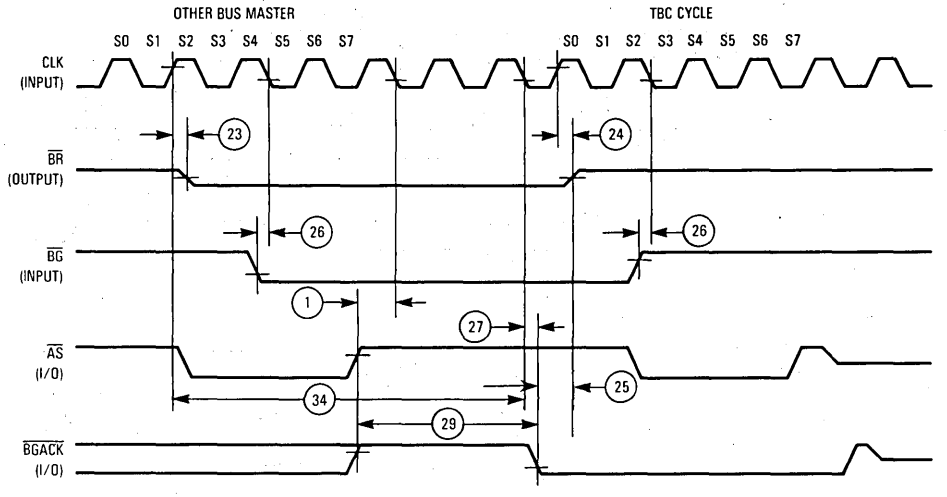


Figure 10. Interrupt Acknowledge Cycle

7



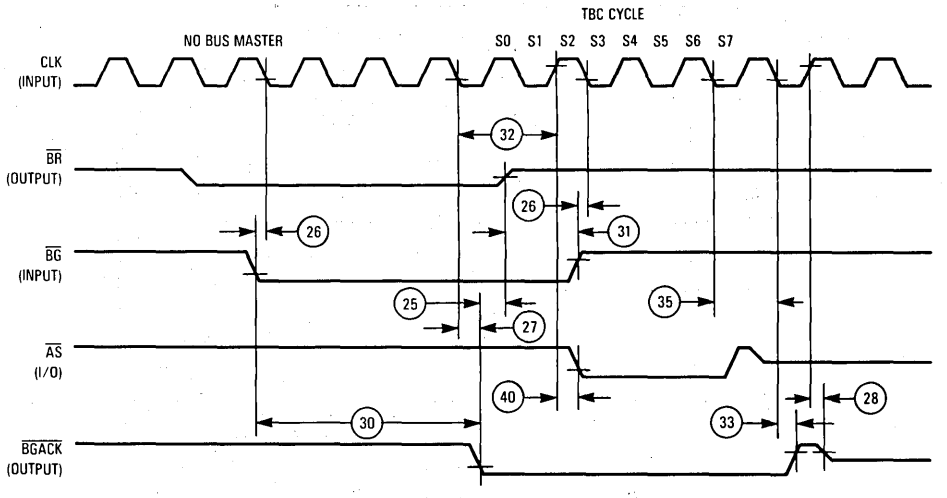
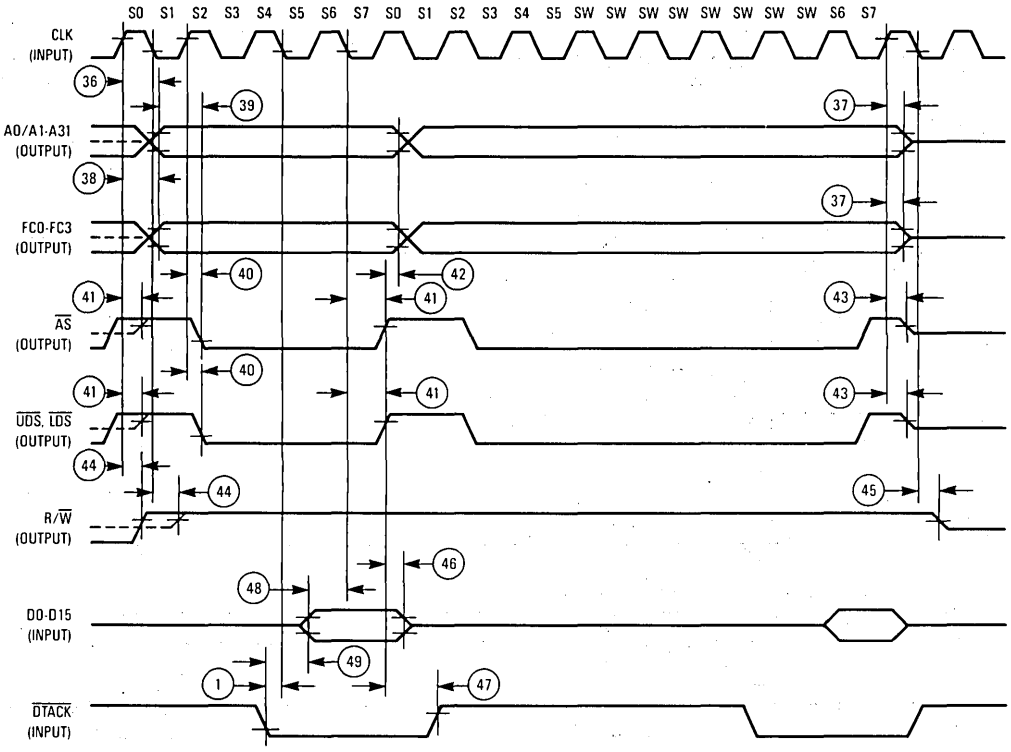


Figure 11. Bus Arbitration



NOTE: The solid lines assume that the communication controller was bus master on the last cycle. The dotted lines assume that there was a different bus master.

Figure 12. Read Cycle and Slow Read Cycle

7

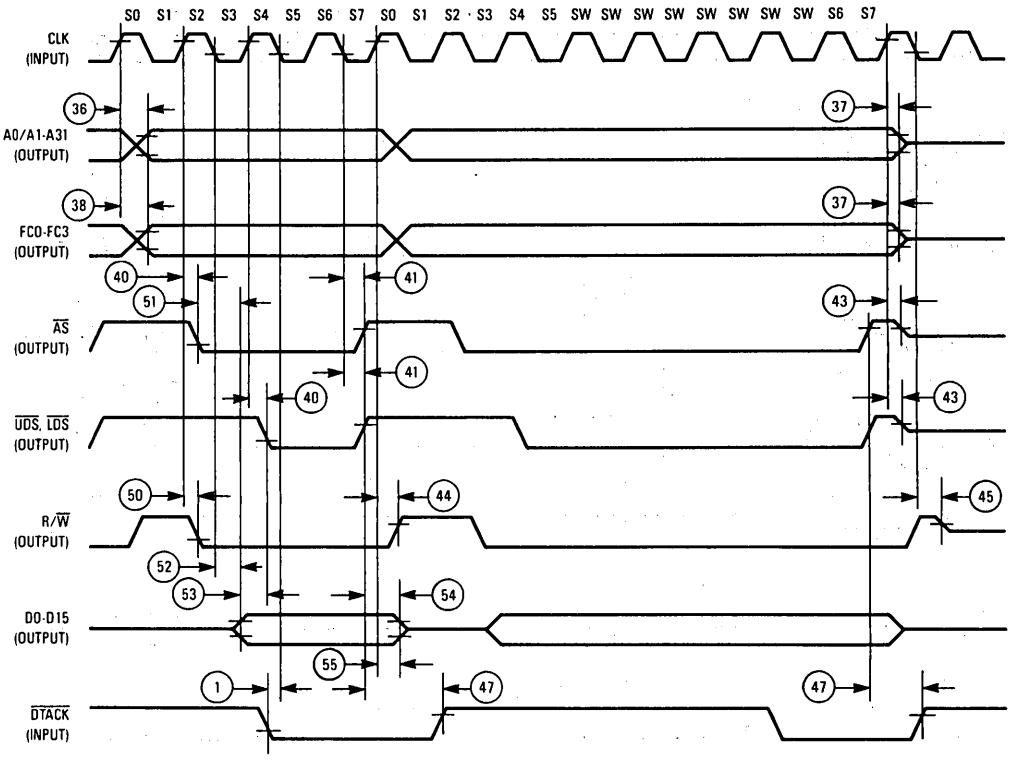
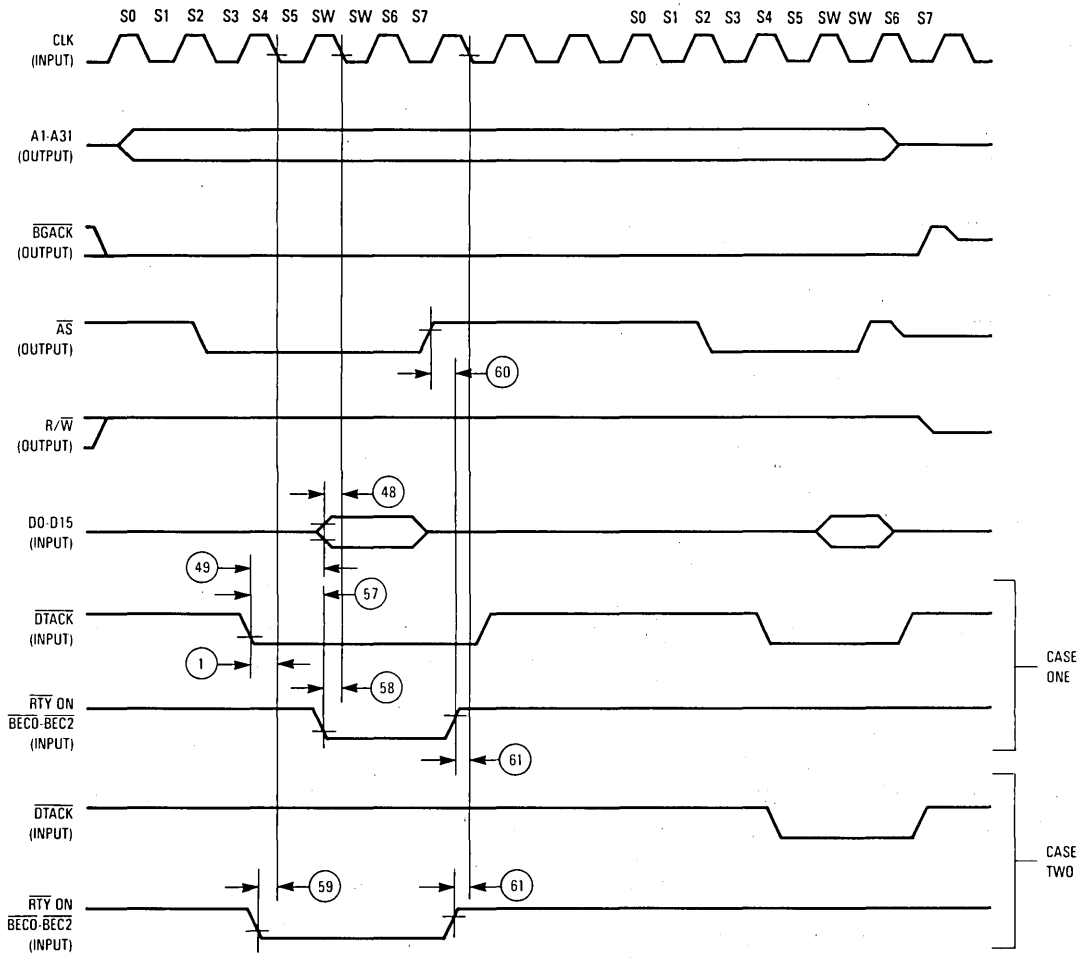


Figure 13. Write Cycle and Slow Write Cycle

7



CASE ONE: If DTACK satisfies 1 then 48 and 58 are required. If DTACK is active but does not satisfy 1 then 49 and 57 are required.

CASE TWO: If DTACK is not active then 59 is required for the exception active set-up time. Parameter 61 is always required for the exception inactive set-up time.

Figure 14. TBC Read Cycle with RETRY

7

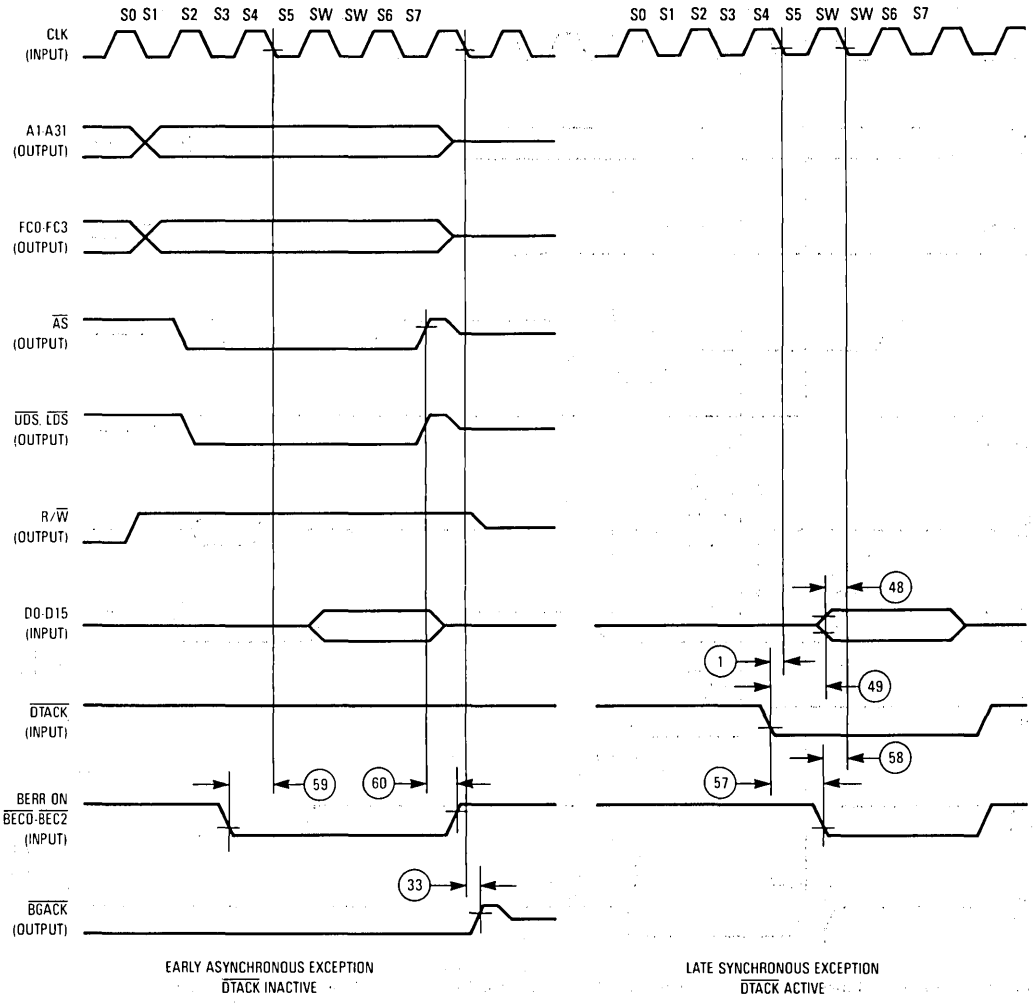
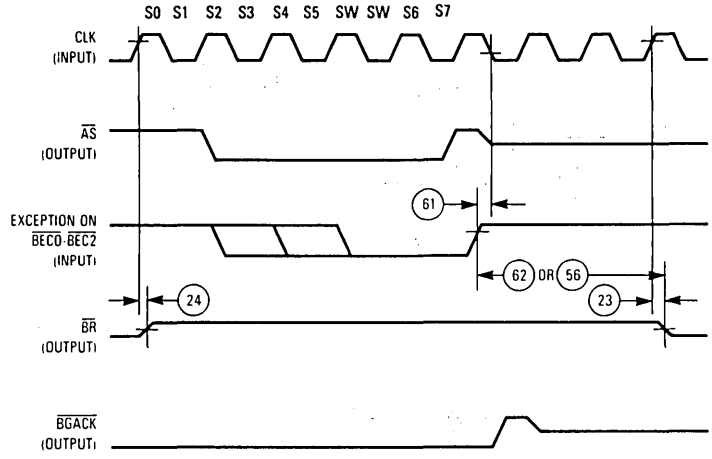


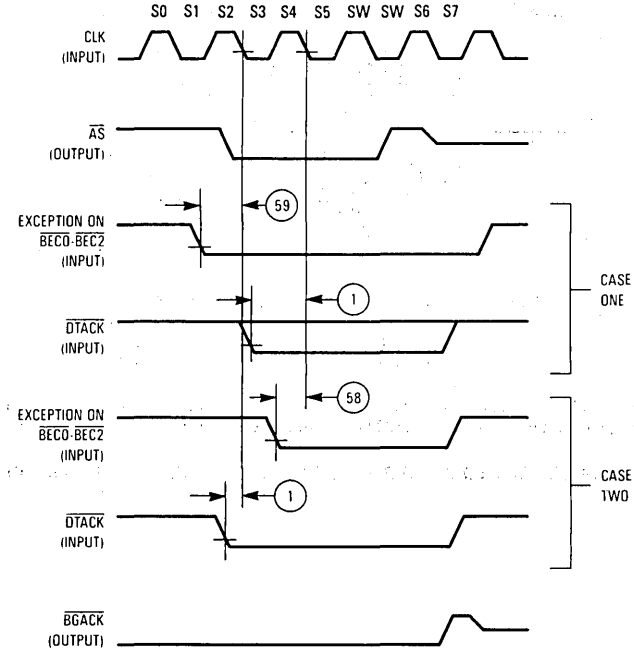
Figure 15. Read Cycle with Bus Error

7



NOTE: The above occurs when the TBC requires the bus cycle after a previous exception.

Figure 16. $\overline{BR}$ After Previous Exception



NOTE: Two alternatives of $\overline{DTACK}$ and exception. Case one has $\overline{DTACK}$ occur after exception and case two has exception occur after $\overline{DTACK}$.

Figure 17. Short Exception Cycle

7

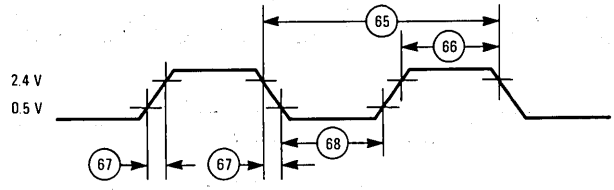
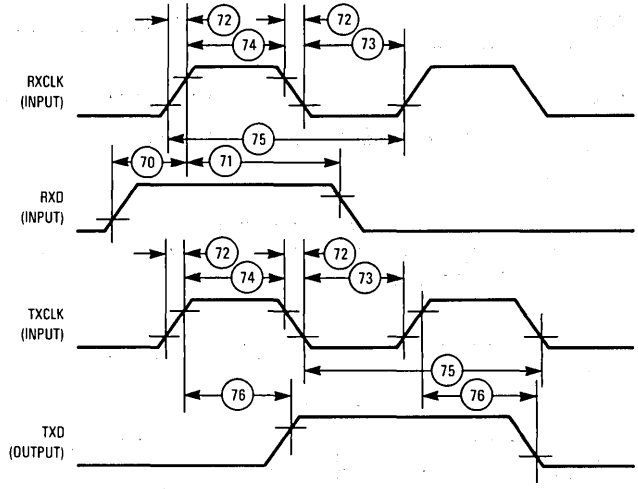


Figure 18. Clock, CLK



RXD SIGNALS: $\overline{\text{SMIND}}$, RXSYM2, RXSYM1, RXSYM0

TXD SIGNALS: $\overline{\text{SMREQ}}$, TXSYM2, TXSYM1, TXSYM0

NOTE: If a serial speed of 10 Mbps is required, then the MC68824 system clock must run at 12.5 MHz or 16.67 MHz. If a serial speed greater than 10 Mbps is required, then the MC68824 system clock must run at 16.67 MHz.

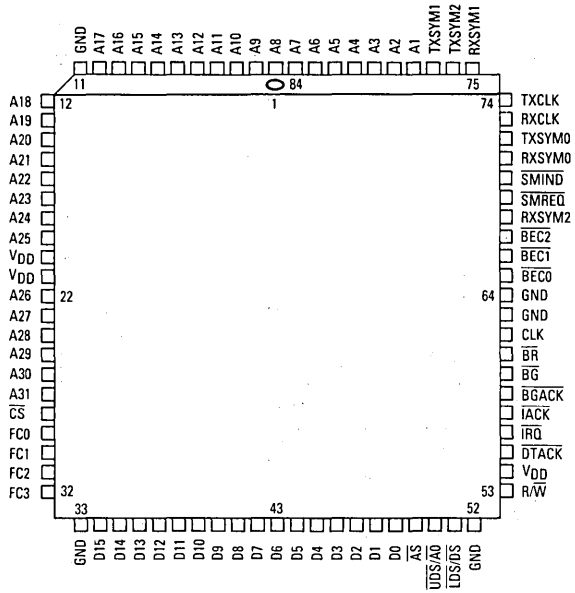
Figure 19. TBC Serial Data (RXD and TXD) and Serial Clocks (RCLK and TCLK)

7

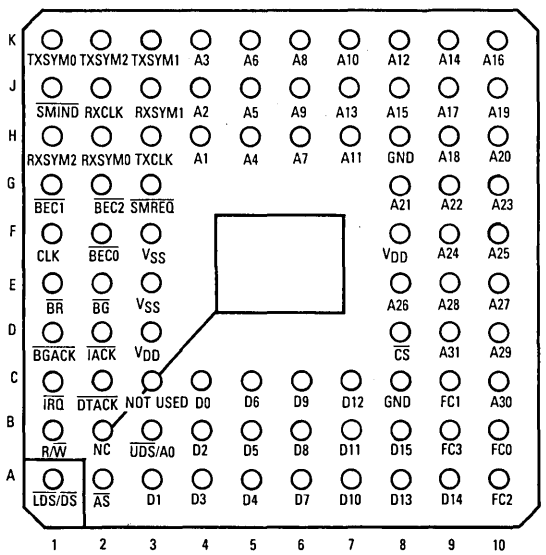
MECHANICAL DATA

PIN ASSIGNMENTS

Plastic-Leaded Chip Carrier



Pin-Grid Array



7

ORDERING INFORMATION

Package Type	Serial Range	System Frequency	Temperature Range	Ordering Designation
Pin Grid Array RC Suffix	1 MHz to 5 MHz	10 MHz	0°C to 70°C	MC68824RC10
	1 MHz to 10 MHz	12.5 MHz	0°C to 70°C	MC68824RC12
	5 MHz to 12.5 MHz	16.67 MHz	0°C to 70°C	MC68824RC16
	10 kHz to 5 MHz	10 MHz	0°C to 70°C	MC68824RC10S
	10 kHz to 10 MHz	12.5 MHz	0°C to 70°C	MC68824RC12S
Pin Grid Array IRC Suffix	1 MHz to 5 MHz	10.0 MHz	0°C to 85°C	MC68824IRC10
	1 MHz to 10 MHz	12.5 MHz	0°C to 85°C	MC68824IRC12
	5 MHz to 12.5 MHz	16.67 MHz	0°C to 85°C	MC68824IRC16
Plastic Lead Chip Carrier FN Suffix	1 MHz to 5 MHz	10.0 MHz	0°C to 70°C	MC68824FN10
	1 MHz to 10 MHz	12.5 MHz	0°C to 70°C	MC68824FN12
	10 kHz to 5 MHz	10 MHz	0°C to 70°C	MC68824FN10S
	10 kHz to 10 MHz	12.5 MHz	0°C to 70°C	MC68824FN12S
Plastic Lead Chip Carrier IFN Suffix	1 MHz to 5 MHz	10.0 MHz	0°C to 70°C	MC68824IFN10
	1 MHz to 10 MHz	12.5 MHz	0°C to 70°C	MC68824IFN12
	10 kHz to 5 MHz	10 MHz	0°C to 70°C	MC68824IFN10S
	10 kHz to 10 MHz	12.5 MHz	0°C to 70°C	MC68824IFN12S
Pin Grid Array R Suffix	1 MHz to 5 MHz	10 MHz	0°C to 70°C	MC68824R10
	1 MHz to 10 MHz	12.5 MHz	0°C to 70°C	MC68824R12
	5 MHz to 12.5 MHz	16.67 MHz	0°C to 70°C	MC68824R16
	10 kHz to 5 MHz	10 MHz	0°C to 70°C	MC68824R10S
	10 kHz to 10 MHz	12.5 MHz	0°C to 70°C	MC68824R12S
Pin Grid Array IR Suffix	1 MHz to 5 MHz	10.0 MHz	0°C to 85°C	MC68824IR10
	1 MHz to 10 MHz	12.5 MHz	0°C to 85°C	MC68824IR12
	5 MHz to 12.5 MHz	16.67 MHz	0°C to 85°C	MC68824IR16

Product Preview

Token Bus Frame Analyzer Software (MC68KTBFA)

The token bus frame analyzer (TBFA) is a software tool to aid in the development and debug of token bus networks and operates in real time. The TBFA can both keep track of statistics, monitoring network performance, and show specific user chosen data frames via an easy to use triggering mechanism. The TBFA can store and display the data units and trigger based on any ISO header or any segment of the MAC data unit. Connecting a TBFA to a LAN is transparent to the LAN's operation because the TBFA is not part of the token-passing ring.

Using the TBFA is straightforward due to its easy-to-use menu-driven interface. In order to execute an option from a menu, the user simply types the required option number and the TBFA prompts the user for any required parameters. At any time, all required information is displayed on the screen, so the user need not manage any notes on paper.

The TBFA is a software package that resides on four EPROMs. These EPROMs fit on a MVME372 MAP Interface Module board. The MVME372 includes among other things a MC68020 MPU and a MC68824 Token Bus Controller (TBC). The TBFA software runs on the MC68020 and controls the MC68824. The MVME372 running the TBFA software is completely stand-alone and requires no backplane bus. Besides the MVME372 and the TBFA EPROM set, the only other components required by the user to run the TBFA are a modem to match the LAN specifications, a VT100 terminal or equivalent, and a power supply.

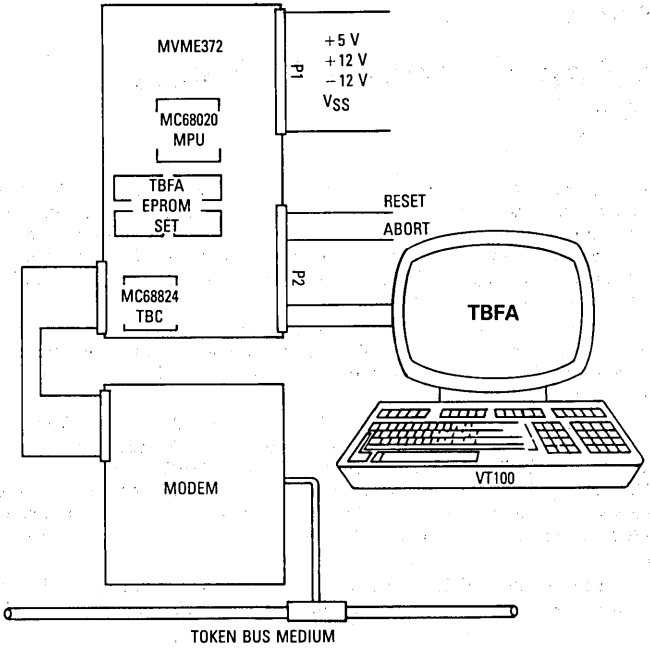
Modems which may be used with the MVME372 include the MVME371FS and MVME371FA, both of which are broadband modems. Any other modem conforming to the IEEE 802.4G physical interface may also be used.

Some of the features of the TBFA are:

- Operation on Any 802.4 LAN at 1, 5, or 10 Mbps
- Menu-Driven, User-Friendly Interface
- Stand-Alone Operation
- Use of a Special MC68824 Mode to Listen to All Activities on the Network
- Important Network Statistics Collection, Including Error Frames
- Collection and Inspection of Data Frames
- Availability of User-Defined Titles to Mark Higher-Layer Headers in MAC Data Units
- Several Useful Triggering Mechanisms, Including MAC Frame Control

A unique and useful function of the TBFA is its triggering mechanism. The TBFA defines four types of triggers which may be combined in any logical manner: starting triggers, qualifiers, stopping triggers, and serial triggers. Starting triggers define when the TBFA will start storing frames. The default is to begin storing frames immediately upon activation. Qualifiers define which frames the TBFA will store. The default is to store all frames. Stopping triggers define when the TBFA will stop storing frames and end the session. The default is when the TBFA runs out of buffers. Serial triggers define relationships between frames (for example, the user could define the TBFA to start storing frames when the TBFA sees two token frames in a row).

TBFA SYSTEM CONFIGURATION



7

GENERAL-PURPOSE PERIPHERAL DEVICES 8

MC68153

Product Preview

MC68153 Bus Interrupter Module

The bipolar LSI MC68153 Bus interrupter interfaces a microcomputer system bus to multiple slave devices requiring interrupt capabilities. It handles up to 4 independent sources of interrupt requests and is fully programmable.

- VERSAbus™/VMEbus™ Compatible
- MC68000 Compatible
- Handles 4 Independent Interrupt Sources
- 8 Programmable Read/Write Registers
- Programmable Interrupt Request Levels
- Programmable Interrupt Vectors
- Supports Interrupt Acknowledge Daisy Chain
- Control Registers Contain Flag Bits
- Single +5.0 Volt Supply
- Total Power Dissipation = 1.0 W Typical
- Temperature Range of 0°C to 70°C
- Chip Access Time = 200 ns Typical with 16 MHz Clock
- 40-Pin Dual-In-Line Packages

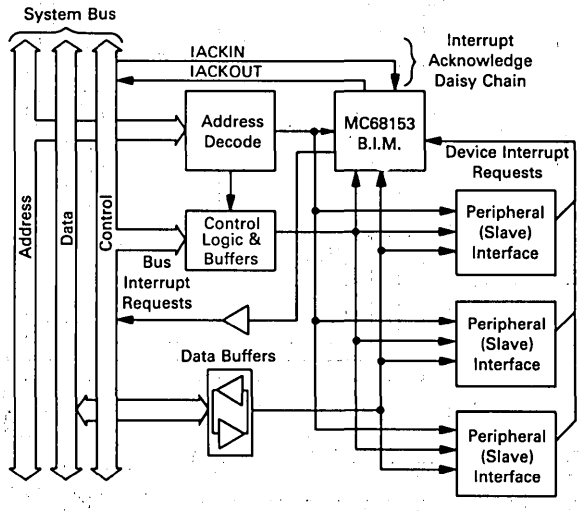


FIGURE 1 — BLOCK DIAGRAM

VERSAbus and VMEbus are trademarks of Motorola Inc.

8

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

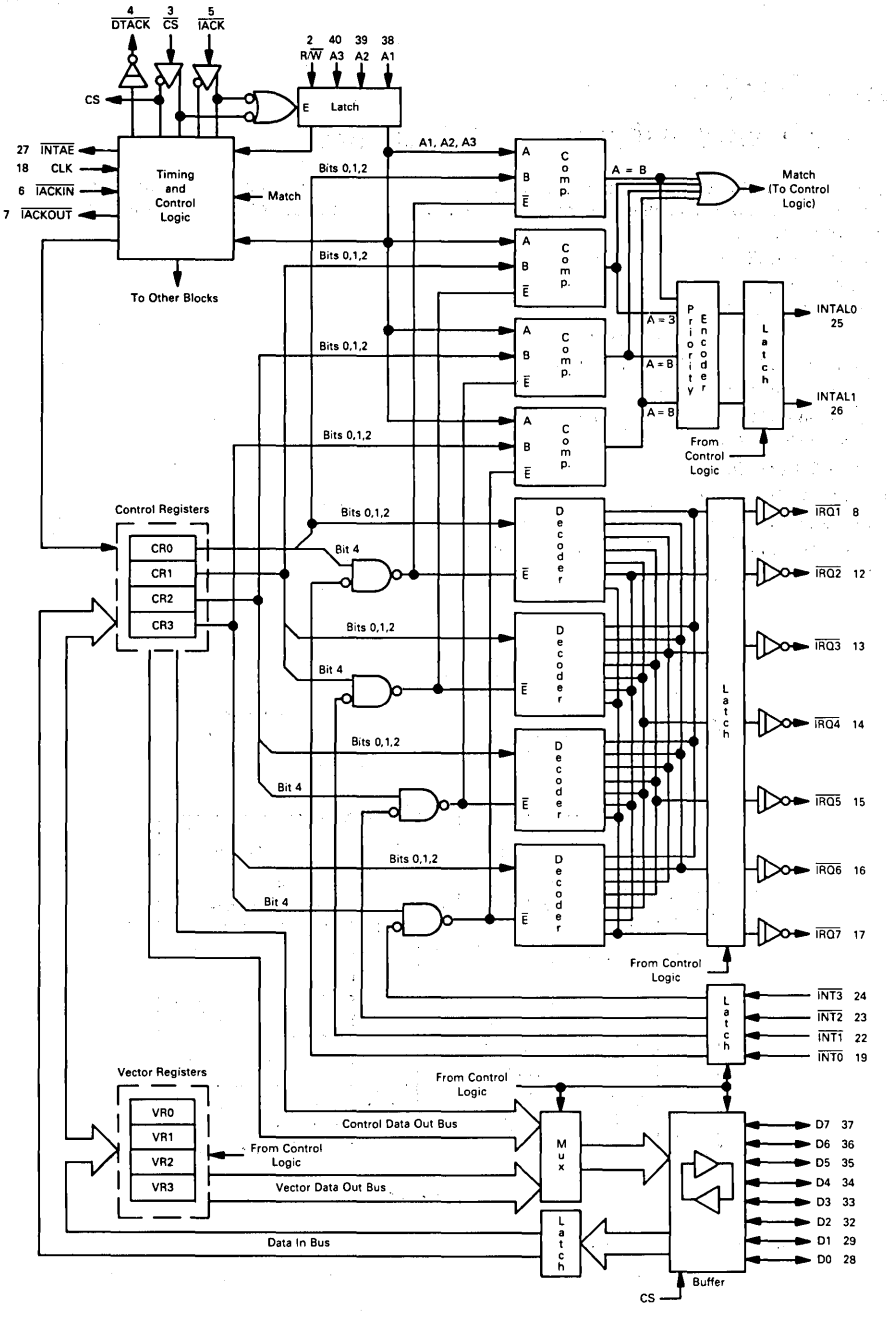


FIGURE 2 — MC68153 FUNCTIONAL BLOCK DIAGRAM

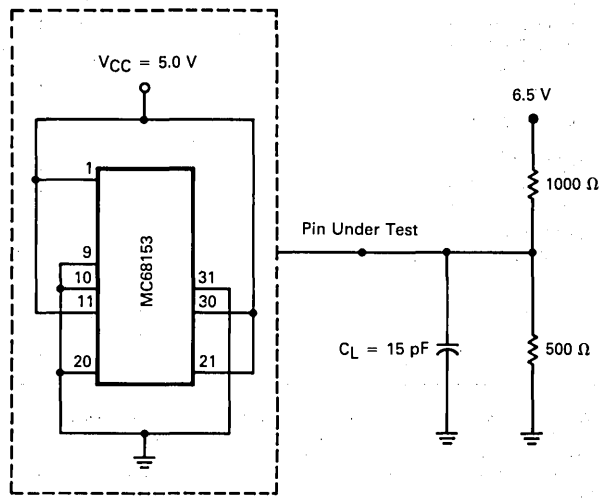
ABSOLUTE MAXIMUM RATINGS (Beyond which useful life may be impaired.)

Parameter	Symbol	Value	Unit
Supply Voltage	V_{CC}	−0.5 to +7.0	V
Input Voltage	V_{in}	−0.5 to +7.0	V
Input Current	I_{in}	−30 to +5.0	mA
Output Voltage	V_{out}	−0.5 to +5.5	V
Output Current	I_{OL}	Twice Rated I_{OL}	mA
Storage Temperature	T_{stg}	−65 to +140	°C
Junction Operating Temperature	T_J	−55 to +140	°C

BURN-IN LIMITS: A maximum T_J of +175°C may be used for periods not to exceed 250 hours.

DC ELECTRICAL SPECIFICATIONS (V_{CC} = 5.0 V ±5%, T_A = 0°C to 70°C)

Parameter	Symbol	Min	Max	Unit	Test Conditions
High Level Input Voltage	V_{IH}	2.0	—	V	
Low Level Input Voltage	V_{IL}	—	0.8	V	
Input Clamp Voltage	V_{IK}	—	−1.5	V	V_{CC} = MIN, I_{IN} = −18 mA
High Level Output Voltage(1)	V_{OH}	2.7	—	V	V_{CC} = MIN, I_{OH} = −400 μA
Low Level Output Voltage	V_{OL}	—	0.4	V	V_{CC} = MIN, I_{OL} = 8.0 mA
Output Short Circuit Current(2)	I_{OS}	−15	−130	mA	V_{CC} = MAX, V_{OUT} = 0 V
High Level Input Current	I_{IH}	—	20	μA	V_{CC} = MAX, V_{IN} = 2.7 V
Low Level Input Current	I_{IL}	—	−0.4	mA	V_{CC} = MAX, V_{IN} = 0.4 V
Supply Current	I_{CC}	225	385	mA	V_{CC} = MAX
Output Off Current (High)	I_{OZH}	—	20	μA	V_{CC} = MAX, V_{OUT} = 2.4 V
Output Off Current (Low)	I_{OZL}	—	−20	μA	V_{CC} = MAX, V_{OUT} = 0.4 V



NOTES:
1. Not applicable to open-collector outputs.
2. Not more than one output should be shorted at a time for longer than one second.
3. $\overline{CS}$ Low to CLK High (Setup Time) of 15 ns Min must be observed.
4. $\overline{IACK}$ Low to CLK High and $\overline{IACKIN}$ Low to CLK High (Setup Times) of 15 ns Min must be observed.
5. See Table 1 for additional performance guidelines.

AC TEST CIRCUIT — AC Testing of All Outputs

8

TABLE 1
AC PERFORMANCE SPECIFICATIONS
(V_{CC} = 5.0 V ±5%, T_A = 0°C to 70°C)

Number	Characteristic	Min	Max	Units	Notes
1	R/$\overline{W}$, A1–A3 Valid to $\overline{CS}$ Low (Setup Time)	10	—	ns	
2	$\overline{CS}$ Low to R/$\overline{W}$, A1–A3 Invalid (Hold Time)	5.0	—	ns	
3	$\overline{CS}$ Low to CLK High (Setup Time)	15	—	ns	1
4	CLK High to Data Out Valid (Delay)	—	55	ns	2
5	CLK High to $\overline{DTACK}$ Low (Delay)	—	40	ns	2
6	$\overline{DTACK}$ Low to $\overline{CS}$ High	0		ns	
7	$\overline{CS}$ High to $\overline{DTACK}$ High (Delay)	—	35	ns	10
8	$\overline{CS}$ High to Data Out Invalid (Hold Time)	0	—	ns	
9	$\overline{CS}$ High to Data Out High-Impedance (Hold Time)	—	50	ns	
10	$\overline{CS}$ High to $\overline{CS}$ or $\overline{IACK}$ Low	20	—	ns	
11	Data In Valid to $\overline{CS}$ Low (Setup Time)	10	—	ns	
12	$\overline{CS}$ Low to Data In Invalid (Hold Time)	5.0	—	ns	
13	$\overline{DTACK}$ High to Data Out High-Impedance	—	25	ns	10
14	$\overline{IACK}$ Low to CLK High (Setup Time)	15	—	ns	1
15	A1–A3 Valid to $\overline{IACK}$ Low (Setup Time)	10	—	ns	
16	$\overline{IACK}$ Low to A1–A3 Invalid (Hold Time)	5.0	—	ns	
17	$\overline{IACKIN}$ Low to CLK High (Setup Time)	15	—	ns	1, 8
18	CLK High to Data Out Valid (Delay)	—	55	ns	3
19	CLK High to $\overline{DTACK}$ Low (Delay)	—	40	ns	3
20	CLK High to $\overline{INTAE}$ Low (Delay)	—	40	ns	3
22	$\overline{DTACK}$ Low to $\overline{IACKIN}$ High	0	—	ns	8
23	$\overline{DTACK}$ Low to $\overline{IACK}$ High	0	—	ns	
24	$\overline{IACK}$ High to Data Out Invalid (Hold Time)	0	—	ns	
25	$\overline{IACK}$ High to Data Out High Impedance (Delay)	—	60	ns	
26	$\overline{IACK}$ High to $\overline{DTACK}$ High (Delay)	—	45	ns	10
27	$\overline{IACK}$ High to $\overline{INTAE}$ High (Delay)	—	35	ns	
28	INTAL0, INTAL1 Valid to $\overline{INTAE}$ Low (Setup Time)	1.0	2.0	CLK Per	
29	$\overline{INTAE}$ High to INTAL0, INTAL1 Invalid (Hold Time)	1.0	2.0	CLK Per	
30	$\overline{IACK}$ High to $\overline{IRQx}$ High (Delay)	—	50	ns	7, 10
31	$\overline{IACK}$ High to $\overline{IACK}$ or $\overline{CS}$ Low	20	—	ns	
32	CLK High to $\overline{IACKOUT}$ Low (Delay)	—	40	ns	5
33	$\overline{IACKIN}$ Low to $\overline{IACKOUT}$ Low (Delay)	—	30	ns	4, 8
34	$\overline{IACKOUT}$ Low to $\overline{IACKIN}$, $\overline{IACK}$ High	0	—	ns	8
35	$\overline{IACK}$ High to $\overline{IACKOUT}$ High (Delay)	—	35	ns	
36	$\overline{IACK}$ and $\overline{CS}$ both Low to CLK High (Setup Time)	15	—	ns	9
37	CLK High to $\overline{IACK}$ or $\overline{CS}$ High (Hold Time)	0	—	ns	
38	$\overline{IACK}$ or $\overline{CS}$ High to $\overline{IACK}$ and $\overline{CS}$ High (Skew)	—	1.0	CLK Per	6
39	Clock Rise Time	—	10	ns	
40	Clock Fall Time	—	10	ns	
41	Clock High Time	20	—	ns	
42	Clock Low Time	20	—	ns	
43	Clock Period	40	—	ns	

NOTES:

1. This specification only applies if the VBIM had completed all operations initiated by the previous bus cycle when $\overline{CS}$ or $\overline{IACK}$ was asserted. Following a normal bus cycle, all operations are completed within 2 clock cycles after $\overline{CS}$ or $\overline{IACK}$ have been negated. If $\overline{IACK}$ or $\overline{CS}$ is asserted prior to completion of these operations, the new cycle, and hence, $\overline{DTACK}$ is postponed.

   If the $\overline{IACK}$, $\overline{IACKIN}$ or $\overline{CS}$ setup time is violated, $\overline{DTACK}$ may be asserted as shown, or may be asserted one clock cycle later (i.e. $\overline{IACK}$ will not be recognized until the next rising edge of the clock).
2. Assumes that 3 has been met.
3. Assumes that 14 and 17 have both been met.
4. Assumes that 14 has been met. ($\overline{IACKOUT}$ cannot go low prior to $\overline{IACKIN}$ going low).
5. Assumes that 14 has been met and $\overline{IACKIN}$ has been low for at least the amount of time specified by 33.
6. 38 is the minimum skew between the last moment when both $\overline{IACK}$ and $\overline{CS}$ are asserted to when both are negated, to insure that an access cycle is not unintentionally started.
7. Assumes no other $\overline{INTx}$ input is causing $\overline{IRQx}$ to be driven low.
8. In non-daisy chain systems, $\overline{IACKIN}$ may be tied low.
9. Failure to meet this spec. causes RESET to be ignored for 1 clock period. It is then necessary to keep these signals low for 3 clock periods instead of 2.
10. Delay time is specified from Input signal to Open-Collector Output pulled High thru 1.0 kΩ resistor to +6.5 V.

8

AC ELECTRICAL SPECIFICATIONS (V_{CC} = 5.0 V ±5%, T_A = 0°C to 70°C)

Parameter	Test Number[5]	Max (ns)
CLK High to Data Out Valid (Delay)[3]	1	55
CLK High to $\overline{\text{DTACK}}$ Low (Delay)[3]	2	40
$\overline{\text{CS}}$ High to $\overline{\text{DTACK}}$ High (Delay)	3	35
CLK High to Data Out Valid (Delay)[4]	4	55
CLK High to $\overline{\text{INTAE}}$ Low (Delay)[4]	5	40
$\overline{\text{IACK}}$ High to Data Out High Impedance (Delay)	6	60
$\overline{\text{IACK}}$ High to $\overline{\text{DTACK}}$ High (Delay)	7	45
$\overline{\text{CS}}$ High to Data Out High (Delay)	8	45
$\overline{\text{CS}}$ High to IRQ High (Delay)	9	60
$\overline{\text{IACK}}$ High to $\overline{\text{INTAE}}$ High (Delay)	10	35

GENERAL DESCRIPTION

The MC68153 Bus Interrupter Module (BIM) is designed to serve as an interrupt requester for peripheral devices in a microcomputer system. Up to 4 independent devices can be interfaced to the system bus by the MC68153. Intended for asynchronous master/slave bus operation, the BIM is compatible with VERSAbus, VMEbus, MC68000 device bus, and other system buses. Figure 1 shows a block diagram of a typical configuration. In this example, three peripheral devices (bus slaves) are connected to the system data bus. Each of these devices could be parallel I/O, serial I/O, or some other function. An interrupt request from any device is routed to the MC68153, and the BIM handles all interface to the system bus. It generates a bus interrupt request as a result of the device interrupt request. When the system interrupt handler or processor responds with an interrupt acknowledge cycle, the MC68153 can answer supplying an interrupt vector and handling all timing.

The functional block diagram of the MC68153 is shown in Figure 2. The device contains circuitry to accept four separate interrupt sources ($\overline{\text{INT0}}$ – $\overline{\text{INT3}}$). Interface to the system bus includes generation of bus interrupt requests ($\overline{\text{IRQ1}}$ – $\overline{\text{IRQ7}}$), response to a bus interrupt acknowledge cycle (either supplying a vector or passing on a daisy chain signal), and releasing the bus interrupt request signal at the proper time. The BIM has flexibility provided by eight programmable read/write registers. Four 8-bit vector registers (VR0 – VR3) contain status/address information and supply a byte vector in response to an interrupt acknowledge cycle for the corresponding interrupt source. Four other 8-bit control registers (CR0 – CR3) contain information that oversees operation of the interrupt circuitry. The control information is programmable and includes interrupt request level and interrupt enable and disable. Also contained in the control registers are flag-bits. These flags are useful for task coordination, resource management, and interprocessor communication.

SIGNAL DESCRIPTION

Throughout the data sheet, signals are presented using the terms asserted and negated independent of whether the signal is asserted in the high voltage or low voltage state. Active low signals are denoted by a superscript bar.

BIDIRECTIONAL DATA BUS — D0 – D7

Pins D0 – D7 form an 8-bit bidirectional data bus to/from the system bus. These are active high, 3-state pins. D7 is the most significant bit.

ADDRESS INPUTS — A1 – A3

These active high inputs serve two functions. One function is to select one of the eight possible registers during a read or write cycle. Secondly, during an interrupt acknowledge A1 – A3 show the level of interrupt being acknowledged, and the BIM uses these to determine if a match exists with an internal level.

CHIP SELECT — $\overline{\text{CS}}$

$\overline{\text{CS}}$ is an active low input used to select the BIM's registers for the current bus cycle. Address strobe, data strobe, and appropriate address bits must be included in the chip select equation.

READ/WRITE — R/$\overline{\text{W}}$

The R/$\overline{\text{W}}$ input is a signal from the system bus used to determine if the current bus cycle is a read (high) or write (low).

DATA TRANSFER ACKNOWLEDGE — $\overline{\text{DTACK}}$

$\overline{\text{DTACK}}$ is an open-collector, active low output that signals the completion of a read, write, or interrupt acknowledge cycle. During read or interrupt acknowledge cycles, $\overline{\text{DTACK}}$ is asserted by the MC68153 after data has been provided on the data bus; during write cycles it is asserted after data has been accepted from the data bus. A pullup resistor is required to maintain $\overline{\text{DTACK}}$ high between bus cycles.

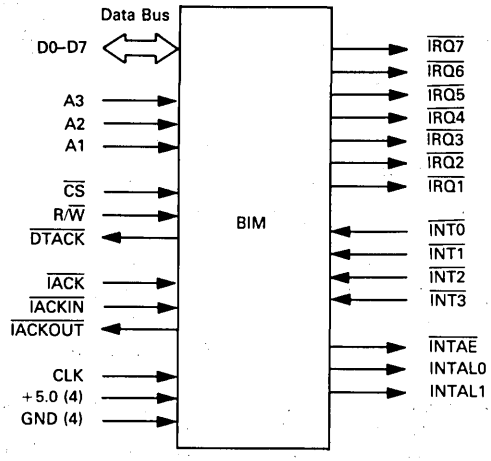


FIGURE 3 — LOGICAL PIN ASSIGNMENT

INTERRUPT ACKNOWLEDGE SIGNALS — IACK, IACKIN, IACKOUT

These three pins support the interrupt acknowledge cycle. A low level on the IACK input indicates an interrupt acknowledge cycle has been initiated. This signal is conditioned externally with Address Strobe and the lower data strobe of an MC68000 type bus. After IACK is asserted the BIM compares the interrupt level presented on address lines A1, A2, and A3 with the current levels generated internally and determines if a match exists. Then, if input IACKIN is asserted (driven low), the BIM will either complete the interrupt acknowledge cycle if a match exists or assert output IACKOUT if *no* match exists.

IACKIN and IACKOUT form part of a prioritized interrupt acknowledge daisy chain. The daisy chain prioritizes interrupters and guarantees that two or more devices requesting an interrupt on the same level will not respond to the same cycle. The requesting device (or interrupter) must wait until IACKIN is asserted and not pass the signal on (assert IACKOUT) if it is to complete the interrupt acknowledge cycle.

BUS INTERRUPT REQUEST SIGNALS — IRQ1 – IRQ7

These open-collector outputs are low when asserted, indicating a bus interrupt is requested at the corresponding level. An open-collector buffer is normally required for sufficient drive when interfacing to a system bus. A pullup resistor is required to maintain IRQ1 – IRQ7 high between interrupt requests.

DEVICE INTERRUPT REQUEST SIGNALS — INT0 – INT3

INT0 – INT3 are active low inputs used to indicate to the BIM that a device wants a bus interrupt.

INTERRUPT ACKNOWLEDGE ENABLE — INTAE

During an interrupt acknowledge cycle, this output pin is asserted low to indicate that outputs INTAL0 and INTAL1 are valid. These two outputs contain an encoded number (x) corresponding to the interrupt (INTx) being acknowledged. This feature can be used to signal interrupting devices, which supply their own vector, when to respond to the interrupt acknowledge cycle with the vector and a DTACK signal.

INTERRUPT ACKNOWLEDGE LEVEL — INTAL0, INTAL1

These active high outputs contain an encoded number corresponding to the interrupt level being acknowledged. They are valid only when INTAE is asserted low.

CLOCK — CLK

The CLK input is used to supply the clock for internal operations of the MC68153.

RESET — CS, IACK

Although a reset input is not supplied, an on-board reset is performed if CS and IACK are asserted simultaneously.

ADDRESS BIT A3	A2	A1	FLAG	FLAG AUTO-CLEAR	EXTERNAL/INTERNAL	INTERRUPT ENABLE	INTERRUPT AUTO-CLEAR	INTERRUPT LEVEL			
0	0	0	F	FAC	X/$\overline{IN}$	IRE	IRAC	L2	L1	L0	CONTROL REGISTER 0
0	0	1	F	FAC	X/$\overline{IN}$	IRE	IRAC	L2	L1	L0	CONTROL REGISTER 1
0	1	0	F	FAC	X/$\overline{IN}$	IRE	IRAC	L2	L1	L0	CONTROL REGISTER 2
1	1	1	F	FAC	X/$\overline{IN}$	IRE	IRAC	L2	L1	L0	CONTROL REGISTER 3
1	0	0	V7	V6	V5	V4	V3	V2	V1	V0	VECTOR REGISTER 0
1	0	1	V7	V6	V5	V4	V3	V2	V1	V0	VECTOR REGISTER 1
1	1	0	V7	V6	V5	V4	V3	V2	V1	V0	VECTOR REGISTER 2
1	1	1	V7	V6	V5	V4	V3	V2	V1	V0	VECTOR REGISTER 3
			7	6	5	4	3	2	1	0	REGISTER NAME
			REGISTER BIT								

FIGURE 4 — MC68153 REGISTER MODEL

REGISTER DESCRIPTION

The MC68153 contains 8 programmable read/write registers. There are four control registers (CR0 – CR3) that govern operation of the device. The other four (VR0 – VR3) are vector registers that contain the vector data used during an interrupt acknowledge cycle. Figure 4 illustrates the device register model.

CONTROL REGISTERS

There is a control register for each interrupt source, i.e., CR0 controls $\overline{INT0}$, CR1 controls $\overline{INT1}$, etc. The control registers are divided into several fields:

1. Interrupt level (L2, L1, L0) — The least significant 3-bit field of the register determines the level at which an interrupt will be generated:

L2	L1	L0	IRQ LEVEL
0	0	0	DISABLED
0	0	1	IRQ1
0	1	0	IRQ2
0	1	1	IRQ3
1	0	0	IRQ4
1	0	1	IRQ5
1	1	0	IRQ6
1	1	1	IRQ7

   A value of zero in the field disables the interrupt.

2. Interrupt Enable (IRE) — This field (Bit 4) must be set (high level) to enable the bus interrupt request associated with the control register. Thus, if the $\overline{INTX}$ line is asserted and IRE is cleared, *no* interrupt request ($\overline{IRQX}$) will be asserted.
3. Interrupt Auto-Clear (IRAC) — If the IRAC is set (Bit 3), IRE (Bit 4) is cleared during an interrupt acknowledge cycle responding to this request. This action of clearing IRE disables the interrupt request. To reenable the interrupt associated with this register, IRE must be set again by writing to the control register.
4. External/Internal (X/$\overline{IN}$) — Bit 5 of the control register determines the response of the MC68153 during an interrupt acknowledge cycle. If the X/$\overline{IN}$ bit is clear (low level) the BIM will respond with vector data and a $\overline{DTACK}$ signal, i.e., an internal response. If X/$\overline{IN}$ is set, the vector is not supplied and no $\overline{DTACK}$ is given by the BIM, i.e., an external device should respond.
5. Flag (F) — Bit 7 is a flag that can be used in conjunction with the test and set instruction of the MC68000. It can be changed without affecting chip operation. It is useful for processor-to-processor communication and resource allocation.
6. Flag Auto-Clear (FAC) — If FAC (Bit 6) is set, the Flag bit is automatically cleared during an interrupt acknowledge cycle.

VECTOR REGISTERS

Each interrupt input has its own associated vector register. Each register is 8 bits wide and supplies a data byte during its interrupt acknowledge cycle if the associated External/Internal (X/$\overline{IN}$) control register bit is clear (zero). This data can be status, identification, or address information depending on system usage. The information is programmed by the system user.

DEVICE RESET

When the MC68153 is reset, the registers are set to a known condition. The control registers are set to all zeros (low). The vector registers are set to $0F. This value is the MC68000 vector for an uninitialized interrupt vector.

8

FUNCTIONAL DESCRIPTION

SYSTEM OVERVIEW

The MC68153 can be used with many system buses, however, it is primarily intended for VMEbus, VERSAbus and MC68000 applications. Figure 5 shows a system configuration similar to VMEbus. In the figure only one system Data Transfer Bus (DTB) master is used. The Priority Interrupt structure provides a means for peripheral slave devices to ask for an interrupt of other processor (DTB master) activity and receive service from the processor. The MC68153 BIM acts as an interface device requesting and responding to interrupt acknowledge cycles for up to 4 independent slaves.

In Figure 5, functional modules are identified as Interrupters and an Interrupt Handler. An Interrupter (such as the MC68153) receives slave requests for an interrupt and handles all interface to the system bus required to ask for and respond to interrupt requests. The Interrupt Handler receives the bus interrupt requests, determines when an interrupt acknowledge will occur and at which level, and finally either performs the interrupt acknowledge (IACK) cycle or tells the DTB master to execute the IACK cycle.

The signal lines in the Priority Interrupt structure include (* — indicates active low):

1. IRQ1*–IRQ7* — seven prioritized interrupt request lines.
2. IACK* — signal line that indicates an interrupt acknowledge cycle is occurring.
3. IACKIN*/IACKOUT* — two signals that form part of a daisy chain that prioritizes interrupters.

In addition Data Transfer Bus control signals are involved in the IACK bus cycle:

1. AS* — the Address Strobe asserted low indicates a valid address is on the bus.
2. DSO* — the lower Data Strobe asserted low indicates a data transfer will occur on bus bits D00–D07.
3. WRITE* — the Read/Write is negated indicating the data is to be read from the Interrupter.
4. A01–A03 — Address lines A01–A03 contain the encoded priority level of the IACK cycle.
5. D00–D07 — Data bus lines D00–D07 are used to pass the interrupt vector from the responding Interrupter to the Interrupt Handler.
6. DTACK* — Data Transfer Acknowledge asserted low signals that the Interrupter has put the vector on the data bus.

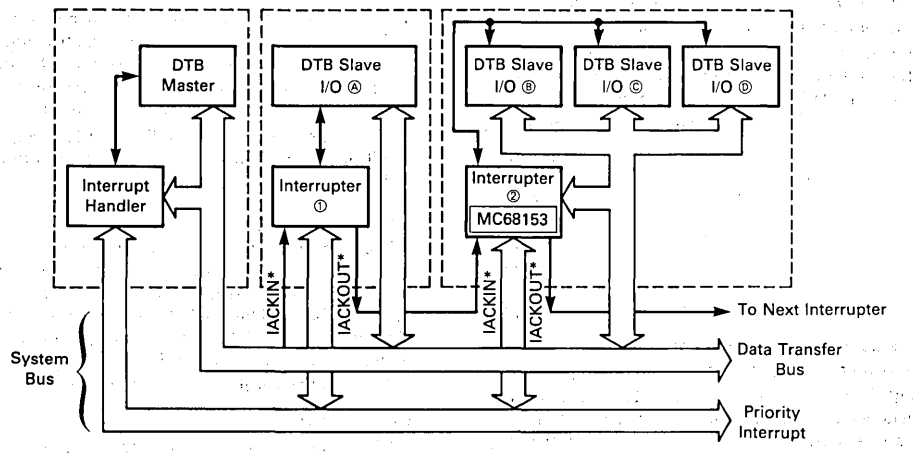


FIGURE 5 — SIMPLE VMEbus CONFIGURATION

8

Figure 6 shows a flow diagram of a typical interrupt request and acknowledge operation. Briefly, the sequence of events is first, an Interrupter makes a request, next the Handler responds with an IACK cycle, then the Interrupter passes a vector to the Handler completing the IACK cycle, and finally the Handler uses the vector to determine additional action. Typically, an interrupt service routine is stored in software and the vector determines where its starting address is stored.

Note the daisy chain operation. If the IACK level (on A01–A03) does not match the Interrupter's request level or if no request is pending, the Interrupter passes the IACKIN* signal on and asserts IACKOUT*. This sequential action automatically prioritizes Requesters on the same level (first one in line with a request pending gets serviced) and prevents two or more Interrupters from responding simultaneously.

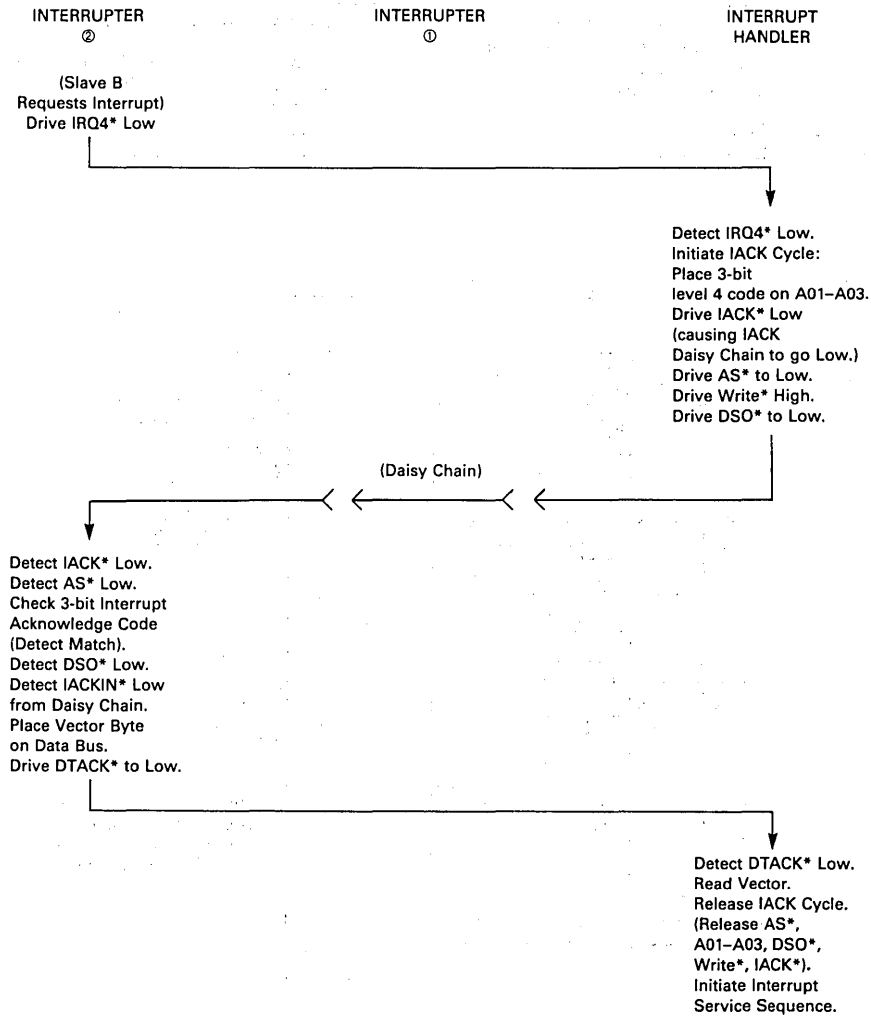


FIGURE 6 — INTERRUPT REQUEST AND ACKNOWLEDGE OPERATION FLOW DIAGRAM

8

This discussion is a very cursory look at the bus operation. For more details including situations with multiple bus masters, the user is directed to the VMEbus Specification MVMEBS or VERSAbus Specification M68KVBS. Also, the MC68153 can be used with other buses having similar interrupt structures.

BIM BUS INTERFACE

Figure 7 shows a simplified block diagram of the MC68153 interface to VERSAbus or VMEbus. Address Decode and Control Logic are dependent on the application and must be designed to guarantee BIM ac specifications. It is possible in most cases that the decode logic can be shared with the slave devices. Buffers are provided where shown to comply with bus loading and drive specifications. It is also possible that buffers can be shared with the slave bus interface.

READ/WRITE OPERATION

All eight BIM registers can be accessed from the system bus in both read and write modes. The BIM has an asynchronous bus interface, primarily designed for MC68000-like buses. The following BIM signals generate read and write cycles: Chip Select ($\overline{CS}$), Read/Write (R/$\overline{W}$), Address Inputs (A1–A3), Data Bus (D0–D7), and Data Transfer Acknowledge ($\overline{DTACK}$). During read and write cycles the internal registers are selected by A1, A2, and A3 in compliance with the Figure 4 Truth Table.

Figure 8 shows the device timing for a read cycle. R/$\overline{W}$ and A1–A3 are latched on the falling edge of $\overline{CS}$ and must meet specified setup and hold times. Chip access time for valid data and $\overline{DTACK}$ are dependent on the clock frequency as shown in the figure.

Figure 9 shows the device timing for a write cycle. R/$\overline{W}$, A1–A3, and D0–D7 are latched on the falling edge of $\overline{CS}$ and must meet specified setup and hold times. Chip access time for $\overline{DTACK}$ is dependent on the clock frequency as shown in the figure.

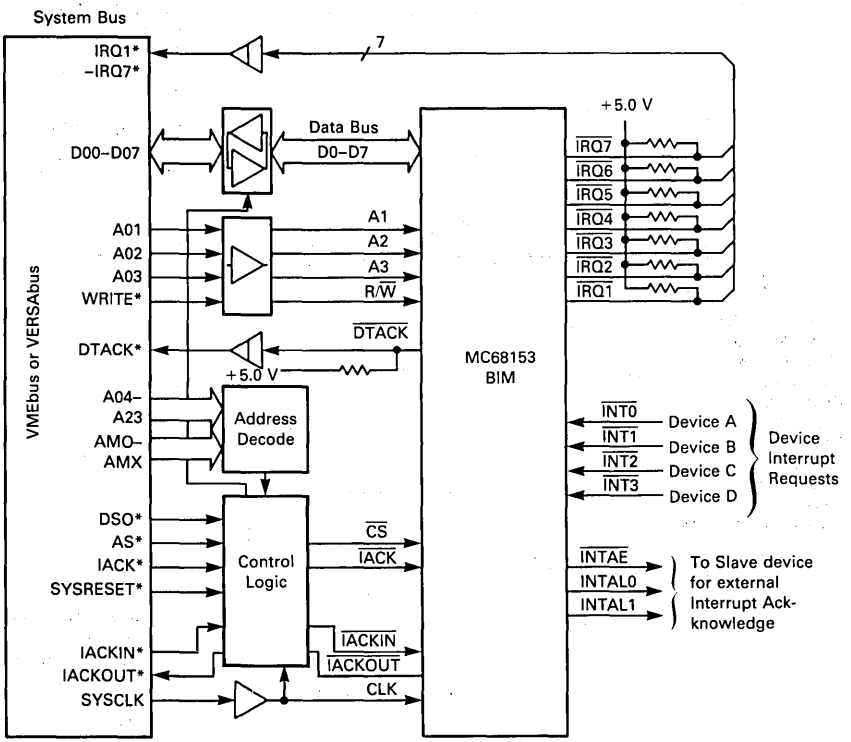


FIGURE 7 — VMEbus/VERSAbus INTERFACE BLOCK DIAGRAM

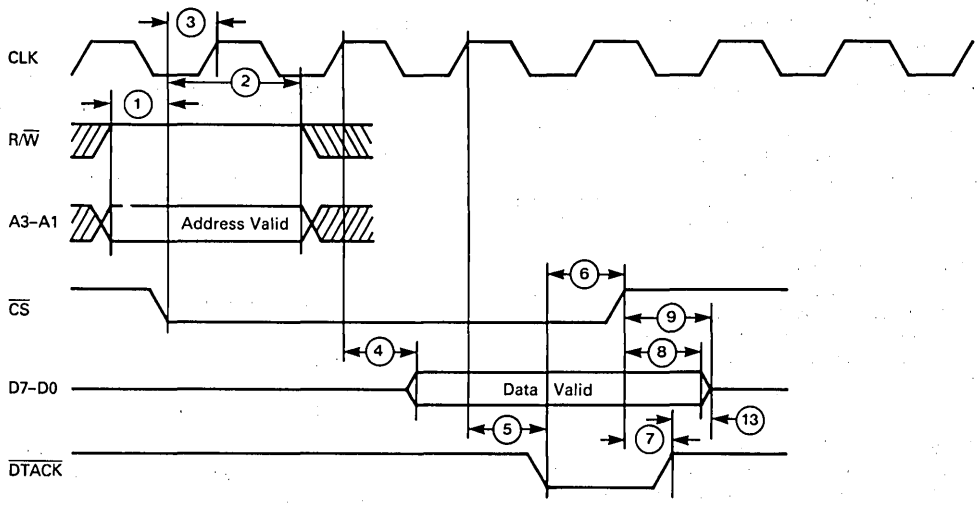


FIGURE 8 — READ CYCLE

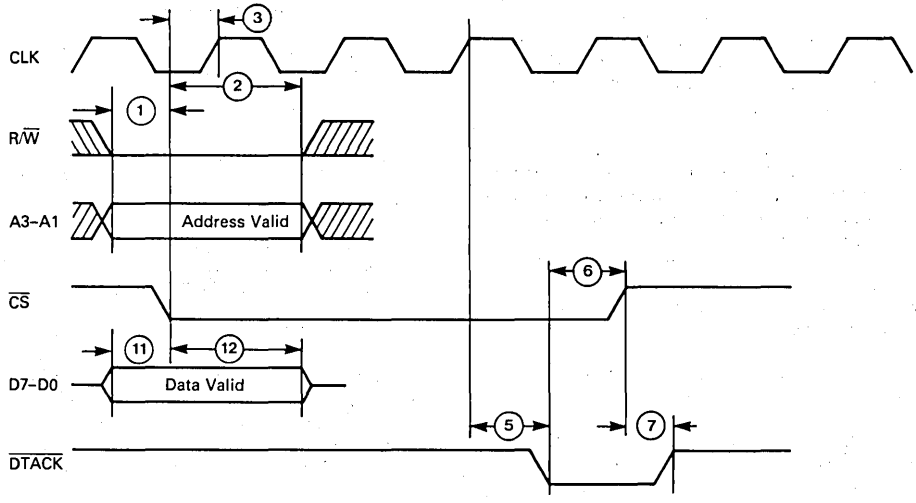


FIGURE 9 — WRITE CYCLE

8

INTERRUPT REQUESTS

The MC68153 accepts device interrupt requests on inputs $\overline{INT0}$, $\overline{INT1}$, $\overline{INT2}$, and $\overline{INT3}$. Each input is regulated by Bit 4 (IRE) of the associated control register (CR0 controls $\overline{INT0}$, CR1 controls $\overline{INT1}$, etc). If IRE (Interrupt Enable) is set and a device input is asserted, an Interrupt Request open-collector output ($\overline{IRQ1}$–$\overline{IRQ7}$) is asserted. The asserted $\overline{IRQX}$ output is selected by the value programmed in Bits 0, 1, and 2 of the control register (L0, L1, and L2). This 3-bit field determines the interrupt request level as set by software.

Two or more interrupt sources can be programmed to the same request level. The corresponding $\overline{IRQX}$ output will remain asserted until multiple interrupt acknowledge cycles respond to all requests.

If the interrupt request level is set to zero, the interrupt is disabled because there is no corresponding IRQ output.

INTERRUPT ACKNOWLEDGE

The response of an Interrupt Handler to a bus interrupt request is an interrupt acknowledge cycle. The IACK cycle is initiated in the MC68153 by receiving $\overline{IACK}$ low. R/$\overline{W}$, A1, A2, A3 are latched, and the interrupt level on line A1–A3 is compared with any interrupt requests pending in the chip. Further activity can be one of four cases:

1. No further action required — This occurs if $\overline{IACKIN}$ is **not** asserted. Asserting $\overline{IACK}$ only starts the BIM activity. If the daisy chain signal never reaches the MC68153 ($\overline{IACKIN}$ is not asserted), another Interrupter has responded to the $\overline{IACK}$ cycle. The cycle will end, the chip $\overline{IACK}$ is negated, and no additional action is required.

2. Pass on the interrupt acknowledge daisy chain — For this case, $\overline{IACKIN}$ input is asserted by the preceding daisy chain Interrupter, and $\overline{IACKOUT}$ output is in turn asserted. The daisy chain signal is passed on when no interrupts are pending on a matching level or when any possible interrupts are disabled. The Interrupt Enable (IRE) bit of a control register can disable any interrupt requests, and in turn, any possible matches.

3. Respond internally — For this case, $\overline{IACKIN}$ is asserted and a match is found. The MC68153 completes the IACK cycle by supplying an interrupt vector from the proper vector register followed by a $\overline{DTACK}$ signal asserted. $\overline{IACKOUT}$ is not asserted because the interrupt acknowledge cycle is completed by this device.

   For the MC68153 to respond in this mode of operation, the EXTERNAL/$\overline{INTERNAL}$ control register bit (X/$\overline{IN}$) must be zero. For each source of interrupt request, the associated control register determines the BIM response to an IACK cycle, and the X/$\overline{IN}$ bit sets this response either internally (X/$\overline{IN}$ = 0) or externally (X/$\overline{IN}$ = 1).

4. Respond externally — For the final case, $\overline{IACKIN}$ is also asserted, a match is found and the associated control register has X/$\overline{IN}$ bit set to one. The MC68153 does not assert $\overline{IACKOUT}$ and does assert $\overline{INTAE}$ low. $\overline{INTAE}$ signals that the requesting device must complete the IACK cycle (supplying a vector and $\overline{DTACK}$) and that the 2-bit code contained on outputs INTAL0 and INTAL1 shows which interrupt source is being acknowledged.

These cases are discussed in more detail in the following paragraphs.

Internal Interrupt Acknowledge

For an internal interrupt acknowledge to occur, the following conditions must be met:

1. One or more device interrupt inputs ($\overline{INT0}$–$\overline{INT3}$) has been asserted and corresponding control bit IRE value is one.

2. $\overline{IACK}$ asserted.

3. A match exists between [A3, A2, A1] and the [L2, L1, L0] field of an enabled, requesting control register. If two or more devices are requesting at the same interrupt level, preference is given to the highest number requester, that is, $\overline{INT3}$ has highest priority and $\overline{INT0}$ has lowest.

4. Control register bit X/$\overline{IN}$ of matching interrupt source must be zero.

5. $\overline{IACKIN}$ asserted.

The internal interrupt acknowledge cycle timing is shown in Figure 10. The 8-bit interrupt acknowledge vector is presented to the data bus and $\overline{DTACK}$ is asserted. Note also that INTAL0 and INTAL1 are valid and $\overline{INTAE}$ is asserted during this cycle although they would normally not be used. The cycle is terminated (data and $\overline{DTACK}$ released) after $\overline{IACK}$ is negated.

During the IACK cycle, the INTERRUPT AUTO-CLEAR control bit (IRAC) comes into play. If the IRAC = one for the responding interrupt source, the INTERRUPT ENABLE (IRE) bit is automatically cleared during the IACK cycle, thus disabling the associated interrupt input and any $\overline{IRQX}$ output asserted due to this interrupt input. Before another interrupt can be requested from this source, IRE must be set to one by writing to the control register.

Note that $\overline{IACKOUT}$ is not asserted because this device is responding to the IACK and does not pass the daisy chain signal on. Also, new device interrupt requests occurring on $\overline{INT0}$–$\overline{INT3}$ after $\overline{IACK}$ is asserted are locked out to prevent any race conditions on the daisy chain.

8

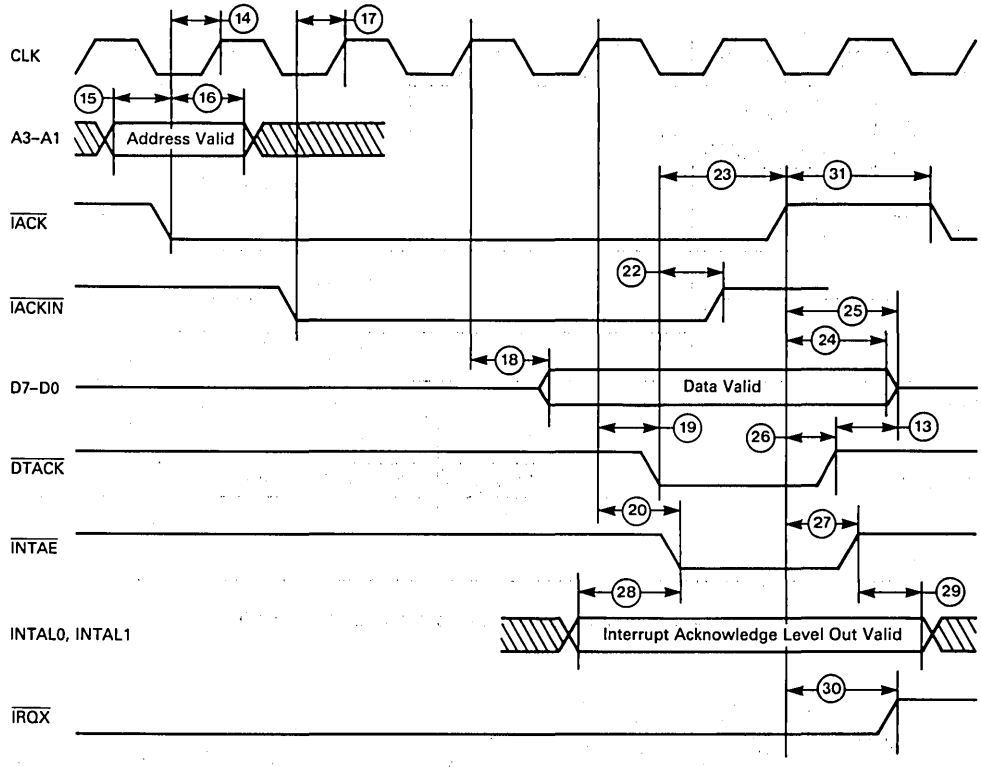


FIGURE 10 — INTERRUPT ACKNOWLEDGE CYCLE — INTERNAL VECTOR

External Interrupt Acknowledge

For an external interrupt acknowledge, the same conditions as listed above are met with one exception. Control register bit X/$\overline{IN}$ of matching interrupt source must be set to one. The timing is shown in Figure 11. For this cycle, the interrupt vector and $\overline{DTACK}$ must be supplied by an external device. $\overline{INTAE}$ is asserted indicating that INTAL0 and INTAL1 are valid. The external device can use these signals to enable the vector and $\overline{DTACK}$. The cycle is terminated after $\overline{IACK}$ is negated.

The IRAC control bit acts in the external interrupt acknowledge the same as described for the internal response (see above). Also, $\overline{IACKOUT}$ is not asserted and new device interrupts are disabled for reasons discussed above.

Pass On IACK Daisy Chain

If the MC68153 has no interrupt request pending at the same level as the interrupt acknowledge, the IACK daisy chain signal is passed on to the next device if $\overline{IACKIN}$ is asserted. The following conditions are thus met:

1. $\overline{IACK}$ asserted.
2. No match exists between [A3, A2, A1] and the [L2, L1, L0] field of an enabled, requesting control register.
3. $\overline{IACKIN}$ is asserted.

$\overline{IACKOUT}$ is asserted if these conditions are valid. This output drives $\overline{IACKIN}$ of the next Interrupter on the daisy chain, passing the signal along. Figure 12 shows the timing for this case. $\overline{IACKOUT}$ is negated after $\overline{IACK}$ is negated.

8

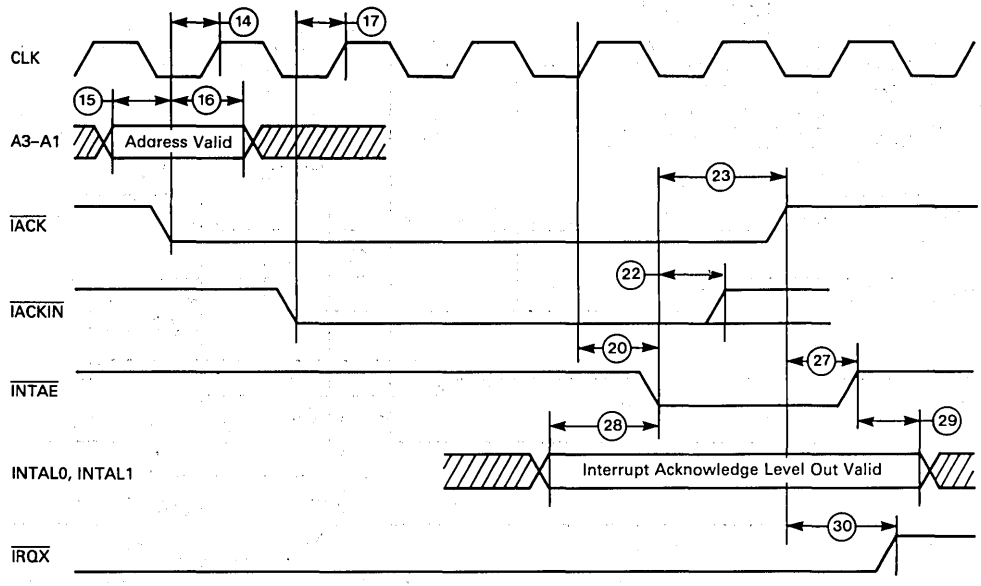


FIGURE 11 — INTERRUPT ACKNOWLEDGE CYCLE — EXTERNAL VECTOR

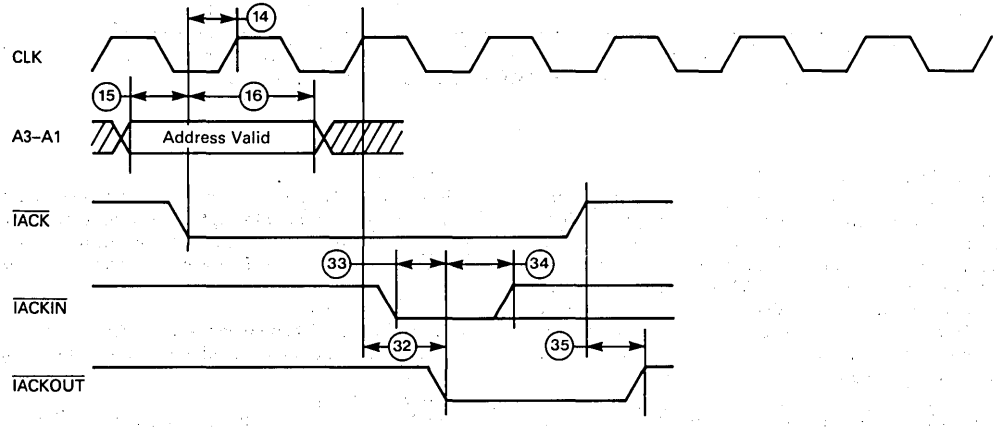


FIGURE 12 — INTERRUPT ACKNOWLEDGE CYCLE — IACKOUT

8

CONTROL REGISTER FLAGS

Each control register contains a Flag bit (F) and a Flag Auto-Clear bit (FAC). Both bits can be read or altered via a register write without affecting the interrupt operation of the device. The Flag is useful as a status indicator for resource management and as a semaphor in multitasking or multiprocessor systems. Flag (F) is located in bit position 7 and can be used with the MC68000 Test and Set (TAS) instruction.

The Flag Auto-Clear (FAC) is used to manipulate the Flag bit. If the Flag is set to one and the FAC is also one, an interrupt acknowledge cycle to the associated interrupt source clears the Flag bit. This feature is useful in determining the interrupt status and passing messages.

RESET

There is no reset input, however, a chip reset is activated by asserting both $\overline{CS}$ and $\overline{IACK}$ simultaneously (Figure 13). These inputs should be held low for a minimum of two clock cycles for a full reset function. The control registers are reset to all zeroes and the Vector Registers are set to a value of $0F. This vector value is the uninitialized vector for the MC68000. See the MC68000 Users Manual for more details on this vector.

CLOCK

The chip clock is required for internal operation to occur. Typical frequency is 16 MHz in VMEbus and VERSAbus applications derived from the system clock. Any frequency can be used, however, up to 25 MHz (Figure 14).

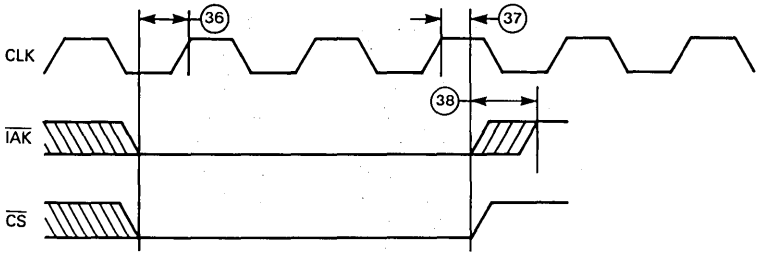


FIGURE 13 — RESET

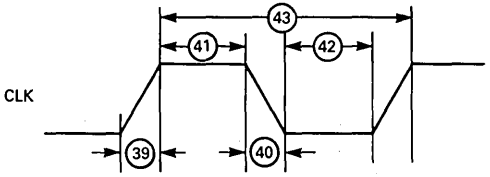


FIGURE 14 — CLOCK WAVEFORM

8

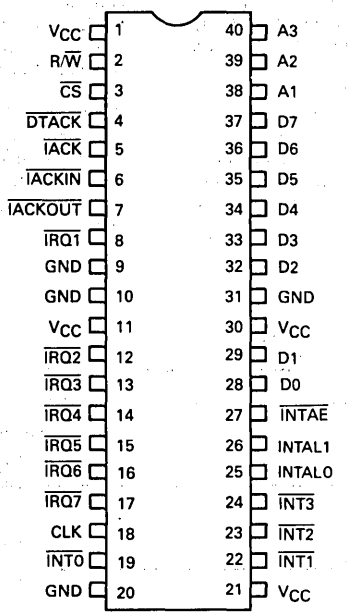
TYPICAL THERMAL CHARACTERISTICS

Package	θJA (Junction to Ambient) Still Air	Junction Temperature Still Air @ 70°C Ambient
L Suffix	40°C/W	147°C
P Suffix[1]	35°C/W	137°C

NOTES:

1. For reliable system operation the maximum allowable junction temperature (T_J) for plastic encapsulated packages has been limited to +140°C. Exceeding this limit will accellerate "wear-out" mechanisms associated with industry standard assembly methods using thermosonic ball bonds to attach gold (Aμ) bond wire to aluminum (Al) bond pads on the die surface.
2. At T_J = 140°C, time to 0.1% failure due to Aμ/Al interconnect = 8,920 Hours.

PIN ASSIGNMENTS



8

MC68230

Technical Summary

Parallel Interface/Timer (PI/T)

The MC68230 parallel interface/timer (PI/T) provides versatile double buffered parallel interfaces and a system oriented timer for MC68000 systems. The parallel interfaces operate in unidirectional or bidirectional modes, either 8 or 16 bits wide. In the unidirectional modes, an associated data direction register determines whether each port pin is an input or output. In the bidirectional modes the data direction registers are ignored and the direction is determined dynamically by the state of four handshake pins. These programmable handshake pins provide an interface flexible enough for connection to a wide variety of low, medium, or high speed peripherals or other computer systems. The PI/T ports allow use of vectored or autovectored interrupts, and also provide a DMA request pin for connection to the MC68450 direct memory access controller (DMAC) or a similar circuit. The PI/T timer contains a 24-bit wide counter and a 5-bit prescaler. The timer may be clocked by the system clock (PI/T CLK pin) or by an external clock (TIN pin), and a 5-bit prescaler can be used. It can generate periodic interrupts, a square wave, or a single interrupt after a programmed time period. It can also be used for elapsed time measurement or as a device watchdog.

Features of the PI/T include:

- M68000 Bus Compatible
- Port Modes Include:
  Bit I/O
  Unidirectional 8 Bit and 16 Bit
  Bidirectional 8 Bit and 16 Bit
- Programmable Handshaking Options
- 24-Bit Programmable Timer Modes
- Five Separate Interrupt Vectors, Four of Which May be Dedicated to External Interrupt Service Requests
- Separate Port and Timer Interrupt Service Requests
- Registers are Read/Write and Directly Addressable
- Registers are Addressed for MOVEP (Move Peripheral) and DMAC Compatibility

8

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

INTRODUCTION

The PI/T consists of two logically independent sections: the ports and the timer. The port section consists of port A (PAO-PA7), port B (PBO-PB7), four handshake pins (H1, H2, H3, and H4), two general input/output (I/O) pins, and six dual-function pins. The dual-function pins can individually operate as a third port (port C) or an alternate function related to either port A, port B, or the timer. The four programmable handshake pins, depending on the mode, can control data transfer to and from the ports, can be used as general-purpose I/O pins, or can be used as interrupt-generating edge-sensitive inputs with corresponding interrupt vector numbers. Refer to Figure 1.

The timer consists of a 24-bit counter, optionally clocked by a 5-bit prescaler. Three pins provide complete timer I/O: PC2/TIN, PC3/TOUT, and PC7/$\overline{\text{TIACK}}$. Only the ones needed for the given configuration perform the timer function, while the others remain port C I/O.

The system bus interface provides for asynchronous transfer of data from the PI/T to a bus master over the data bus (DO-D7). Data transfer acknowledge ($\overline{\text{DTACK}}$), register selects (RS1-RS5), timer interrupt acknowledge ($\overline{\text{TIACK}}$), read/write line (R/$\overline{\text{W}}$), chip select ($\overline{\text{CS}}$), or port interrupt acknowledge ($\overline{\text{PIACK}}$) control data transfers between the PI/T and an M68000 processor.

PORT MODE DESCRIPTION

The primary focus of most applications will be on port A, port B, the handshake pins, the port interrupt pins, and the DMA request pin. They are controlled in the following way: the port general control register contains a 2-bit field that specifies one of four operation modes. These govern the overall operation of the ports and determine

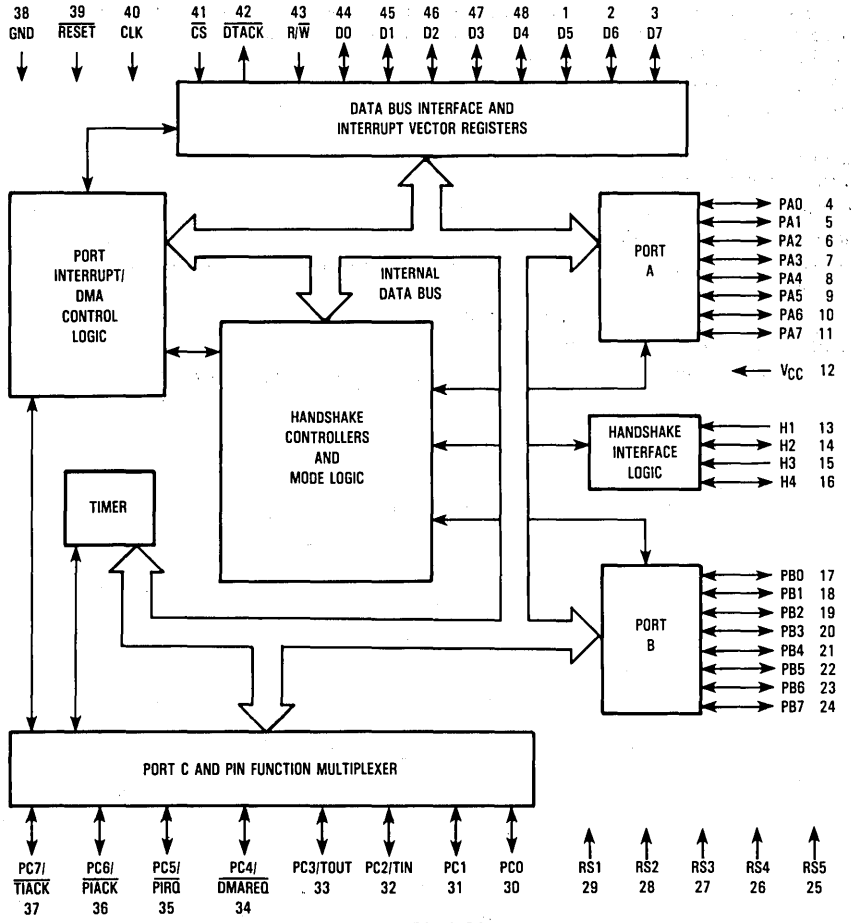


Figure 1. Block Diagram

their interrelationships. Some modes require additional information from each port's control register to further define its operation. In each port control register, there is a 2-bit submode field that serves this purpose. Each port mode/submode combination specifies a set of programmable characteristics that fully define the behavior of that port and two of the handshake pins. This structure is summarized in Table 1 and Figure 2.

Table 1. Port Mode Control Summary

Mode 0 (Unidirectional 8-Bit Mode)
- Port A
  - Submode 00 — Pin-Definable Double-Buffered Input or Single-Buffered Output
    - H1 — Latches input data
    - H2 — Status/interrupt generating input, general-purpose output, or operation with H1 in the interlocked or pulsed handshake protocols
  - Submode 01 — Pin-Definable Double-Buffered Output or Non-Latched Input
    - H1 — Indicates data received by peripheral
    - H2 — Status/interrupt generating input, general-purpose output, or operation with H1 in the interlocked or pulsed handshake protocols
  - Submode 1X — Pin-Definable Single-Buffered Output or Non-Latched Input
    - H1 — Status/interrupt generating input
    - H2 — Status/interrupt generating input or general-purpose output
- Port B
  - H3 and H4 — Identical to port A, H1 and H2

Mode 1 (Unidirectional 16-Bit Mode)
- Port A — Most Significant Data Byte or Non-Latched Input or Single Buffered Output
  - Submode XX — (Not Used)
    - H1 — Status/interrupt generating input
    - H2 — Status/interrupt generating input or general-purpose output
- Port B — Least-Significant Data Byte
  - Submode X0 — Pin-Definable Double-Buffered Input or Single- Buffered Output
    - H3 — Latches input data
    - H4 — Status/interrupt generating input, general-purpose output, or opereation with H3 in the interlocked or pulsed handshake protocols
  - Submode X1 — Pin-Definable Double-Buffered Output or Non-Latched Input
    - H3 — Indicates data received by peripheral
    - H4 — Status/interrupt generating input, general-purpose output, or operation with H3 in the interlocked or pulsed handshake protocols

Mode 2 (Bidirectional 8-Bit Mode)
- Port A — Bit I/O
  - Submode XX — (Not Used)
- Port B — Double-Buffered Bidirectional Data
  - Submode XX — (Not Used)
    - H1 — Indicates output data received by the peripheral and controls output drivers
    - H2 — Operating with H1 in the interlocked or pulsed output handshake protocols
    - H3 — Latches input data
    - H4 — Operation with H3 in the interlocked or pulsed input protocols

Mode 3 (Bidirectional 16-Bit Mode)
- Port A — Double-Buffered Bidirectional Data (Most-Significant Data Byte)
  - Submode XX — (Not Used)
- Port B — Double-Buffered Bidirectional Data (Least-Significant Data Byte)
  - Submode XX — (Not Used)
    - H1 — Indicates output data received by peripheral and controls output drivers
    - H2 — Operation with H1 in the interlocked or pulsed output handshake protocols
    - H3 — Latches input data
    - H4 — Operation with H3 in the interlocked or pulsed input handshake protocols

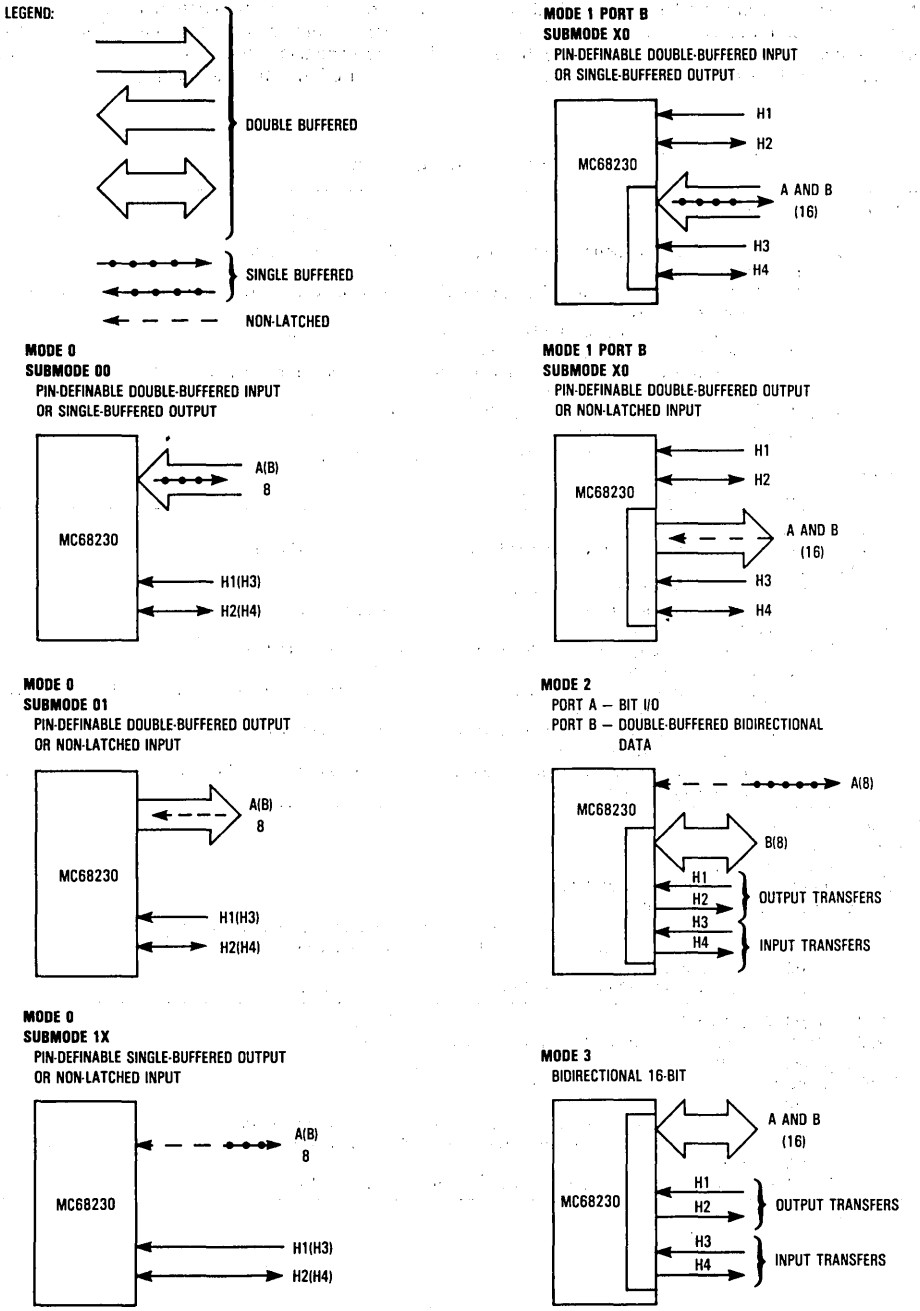
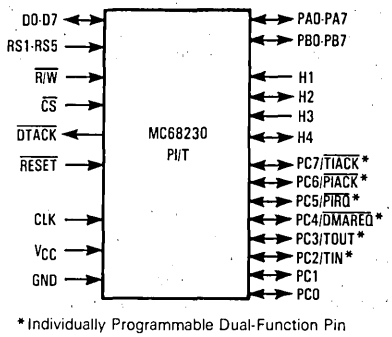


Figure 2. Port Mode Layout

SIGNAL DESCRIPTION

The input and output signals are illustrated functionally in Figure 3 and described in the following paragraphs.



*Individually Programmable Dual-Function Pin

Figure 3. Logical Pin Assignment

BIDIRECTIONAL DATA BUS (D0-D7)

The data bus pins, D0-D7, form an 8-bit bidirectional data bus to/from an M68000 bus master. These pins are active high.

REGISTER SELECTS (RS1-RS5)

The register select pins, RS1-RS5, are active high high-impedance inputs that determine which of the 23 internal registers is being selected. They are provided by the M68000 bus master or other bus master.

READ/WRITE (R/$\overline{W}$)

R/$\overline{W}$ is a high-impedance read/write input signal from the M68000 bus master, indicating whether the current bus cycle is a read (high) or write (low) cycle.

CHIP SELECT ($\overline{CS}$)

$\overline{CS}$ is a high-impedance input that selects the PI/T registers for the current bus cycle. The data strobe (upper or lower) of the bus master, along with the appropriate address bits, must be included in the chip-select equation. A low level corresponds to an asserted chip select.

DATA TRANSFER ACKNOWLEDGE ($\overline{DTACK}$)

$\overline{DTACK}$ is an active low output that signals the completion of the bus cycle. During read or interrupt acknowledge cycles, $\overline{DTACK}$ is asserted after data has been provided on the data bus; during write cycles it is asserted after data has been accepted at the data bus. Data transfer acknowledge is compatible with the MC68000 and with other M68000 bus mastsers such as the MC68450 direct memory access controller (DMAC). A pullup resistor is required to maintain $\overline{DTACK}$ high between bus cycles.

RESET ($\overline{RESET}$)

$\overline{RESET}$ is a high-impedance input used to initialize all PI/T functions. All control and data direction registers are cleared and most internal operations are disabled by the assertion of $\overline{RESET}$ (low).

CLOCK (CLK)

The clock pin is a high-impedance TTL-compatible signal with the same specifications as the MC68000. The PI/T contains dynamic logic throughout, and hence this clock must not be gated off at any time. It is not necessary that this clock maintain any particular phase relationship with the M68000 system clock. It may be connected to an independent frequency source (faster or slower) as long as all bus specifications are met.

PORT A AND PORT B (PA0-PA7 and PB0-PB7)

Ports A and B are 8-bit ports that may be concatenated to form a 16-bit port in certain modes. The ports may be controlled in conjunction with the handshake pins H1-H4. For stabilization during system power up, ports A and B have internal pullup resistors to V_{CC}. All port pins are active high.

HANDSHAKE PINS (H1-H4)

Handshake pins H1-H4 are multi-purpose pins that (depending on the operational mode) may provide an interlocked handshake, a pulsed handshake, interrupt-generating edge-sensitive inputs (independent of data transfers), or simple I/O pins. For stabilization during system power up, H2 and H4 have internal pullup resistors to V_{CC}. The sense of H1-H4 (active high or low) may be programmed in bits 3-0 of the port general control register. Independent of the mode, the instantaneous level of the handshake pins can be read from the port status register.

PORT C (PC0-PC7/ALTERNATE FUNCTION)

This port can be used as eight general purpose I/O pins (PC0-PC7) or any combination of six special function pins and two general purpose I/O pins (PC0-PC1). Each dual-function pin can be a standard I/O or a special function independent of the other port C pins. When used as a port C pin, these pins are active high. They may be individually programmed as inputs or outputs by the port C data direction register. The dual function pins are defined in the following paragraphs.

The alternate functions TIN, TOUT, and $\overline{TIACK}$ are timer I/O pins. TIN may be used as a rising-edge triggered external clock input or an external run/halt control pin (the timer is in the run state if run/halt is high and in the halt state if run/halt is low). TOUT may provide an active low timer interrupt request output or a general-purpose square-wave output, initially high. $\overline{TIACK}$ is an active low high-impedance input used for timer interrupt acknowledge.

The port functions of the PI/T (ports A and B) have an independent pair of active low interrupt request ($\overline{PIRQ}$) and interrupt acknowledge ($\overline{PIACK}$) pins.

8

The $\overline{\text{DMAREQ}}$ (direct memory access request) pin provides an active low direct memory access controller request pulse for three clock cycles, completely compatible with the MC68450 DMAC. Note that if these pins are used for an alternate function, the corresponding bit in the Port C Data Direction Register must be programmed as an input (0).

SIGNAL SUMMARY

Table 2 is a summary of all the signals discussed in the previous paragraphs.

BUS INTERFACE OPERATION

The PI/T has an asynchronous bus interface, primarily designed for use with an MC68000 microprocessor. With care, however, it can be connected to synchronous microprocessor buses.

In an asynchronous system the PI/T clock may operate at a significantly different frequency, either higher or lower, than the bus master and other system components, as long as all bus specifications are met. The MC68230 CLK pin has the same specifications as the MC68000 CLK pin, and must not be gated off at any time.

The following signals generate normal read and write cycles to the PI/T: $\overline{\text{CS}}$ (chip select), R/$\overline{\text{W}}$ (read/write), RS1-RS5 (five register select bits), DO-D7 (the 8-bit bidirectional data bus), and $\overline{\text{DTACK}}$ (data transfer acknowledge). To generate interrupt acknowledge cycles, PC6/$\overline{\text{PIACK}}$ or PC7/$\overline{\text{TIACK}}$ is used instead of $\overline{\text{CS}}$, and the register select pins are ignored. No combination of the following pin functions may be asserted simultaneously: $\overline{\text{CS}}$, $\overline{\text{PIACK}}$, or $\overline{\text{TIACK}}$.

TIMER OPERATION

The MC68230 timer can provide several facilities needed by MC68000 operating systems. It can generate periodic interrupts, a square wave, or a single interrupt after a programmed time period. Also, it can be used for elapsed time measurement or as a device watchdog.

The PI/timer contains a 24-bit synchronous down counter that is loaded from three 8-bit counter preload registers. The 24-bit counter may be clocked by the output of a 5-bit (divide-by-32) prescaler or by an external timer input (TIN). If the prescaler is used, it may be clocked by the system clock (CLK pin) or by the TIN external input. The counter signals the occurrence of an event primarily

Table 2. Signal Summary

Signal Name	Input/Output	Active State	Edge/Level Sensitive	Output States
CLK	Input		Falling and Rising Edge	
$\overline{\text{CS}}$	Input	Low	Level	
D0-D7	Input/Output	High = 1, low = 0	Level	High, Low, High Impedance
$\overline{\text{DMAREQ}}$	Output	Low		High, Low
$\overline{\text{DTACK}}$	Output	Low		High, Low, High Impedance*
H1(H3)***	Input	Low or High	Asserted Edge	
H2)H4**	Input or Output	Low or High	Asserted Edge	High, Low, High Impedance
PA0-PA7**, PB0-PB7** PC0-PC7	Input/Output Input or Output	High = 1, Low = 0	Level	High, Low, High Impedance
$\overline{\text{PIACK}}$	Input	Low	Level	
$\overline{\text{PIRQ}}$	Output	Low		Low, High Impedance*
RS1-RS5	Input	High = 1, Low = 0	Level	
R/$\overline{\text{W}}$	Input	High Read, Low Write	Level	
$\overline{\text{RESET}}$	Input	Low	Level	
$\overline{\text{TIACK}}$	Input	Low	Level	
TIN (External Clock)	Input		Rising Edge	
TIN (Run/Halt)	Input	High	Level	
TOUT (Square Wave)	Output	Low		High, Low
TOUT ($\overline{\text{TIRQ}}$)	Output	Low		Low, High Impedance*

*Pullup resistors required.
**Note these pins have internal pullup resistors.
***H1 is level sensitive for output buffer control in modes 2 and 3.

through zero detection. (A zero is when the counter of the 24-bit timer is equal to zero). This sets the zero detect status (ZDS) bit in the timer status register. It may be checked by the processor or may be used to generate a timer interrupt. The ZDS bit can be reset by writing a one to the timer status register in that bit position independent of timer operation.

The general operation of the timer is flexible and easily programmable. The timer is fully configured and controlled by programming the 8-bit timer control register. It controls: 1) the choice between the port C operation of three timer pins, 2) whether the counter is loaded from the counter preload register or rolls over when zero detect is reached, 3) the clock input, 4) whether the prescaler is used, and 5) whether the timer is enabled.

REGISTER MODEL

A register model that includes the corresponding register selects is shown in Table 3.

Table 3. Register Model (Sheet 1 of 2)

Register Select Bits 5 4 3 2 1	7	6	5	4	3	2	1	0	Register Value After RESET (Hex Value)	
0 0 0 0 0	Port Mode Control		H34 Enable	H12 Enable	H4 Sense	H3 Sense	H2 Sense	H1 Sense	0 0	Port General Control Register
0 0 0 0 1	*	SVCRQ Select		IPF Select		Port Interrupt Priority Control			0 0	Port Service Request Register
0 0 0 1 0	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	0 0	Port A Data Direction Register
0 0 0 1 1	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	0 0	Port B Data Direction Register
0 0 1 0 0	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	0 0	Port C Data Direction Register
0 0 1 0 1	Interrupt Vector Number						*	*	0 F	Port Interrupt Vector Register
0 0 1 1 0	Port A Submode		H2 Control			H2 Int Enable	H1 SVCRQ Enable	H1 Stat Control	0 0	Port A Control Register
0 0 1 1 1	Port B Submode		H4 Control			H4 Int Enable	H3 SVCRQ Enable	H3 Stat Control	0 0	Port B Control Register
0 1 0 0 0	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	* *	Port A Data Register
0 1 0 0 1	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	* *	Port B Data Register
0 1 0 1 0	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	***	Port A Alternate Register
0 1 0 1 1	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	***	Port B Alternate Register
0 1 1 0 0	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	****	Port C Data Register
0 1 1 0 1	H4 Level	H3 Level	H2 Level	H1 Level	H4S	H3S	H2S	H1S	****	Port Status Register
0 1 1 1 0	*	*	*	*	*	*	*	*	0 0	(Null)
0 1 1 1 1	*	*	*	*	*	*	*	*	0 0	(Null)

*Unused, read as zero
**Value before RESET
***Current value on pins
****Undetermined value

8

Table 3. Register Model (Sheet 2 of 2)

Register Select Bits 5 4 3 2 1	7	6	5	4	3	2	1	0	Register Value After RESET (Hex Value)	
1 0 0 0 0	TOUT/TIACK Control			Z D Control	Clock Control			Timer Enable	0 0	Timer Control Register
1 0 0 0 1	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	0 F	Timer Interrupt Vector Register
1 0 0 1 0	*	*	*	*	*	*	*	*	0 0	(Null)
1 0 0 1 1	Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16	* *	Counter Preload Register (High)
1 0 1 0 0	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	* *	Counter Preload Register (Mid)
1 0 1 0 1	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	* *	Counter Preload Register (Low)
1 0 1 1 0	*	*	*	*	*	*	*	*	0 0	(Null)
1 0 1 1 1	Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16	* *	Count Register (High)
1 1 0 0 0	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	* *	Count Register (Mid)
1 1 0 0 1	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	* *	Count Register (Low)
1 1 0 1 0	*	*	*	*	*	*	*	ZDS	0 0	Timer Status
1 1 0 1 1	*	*	*	*	*	*	*	*	0 0	(Null)
1 1 1 0 0	*	*	*	*	*	*	*	*	0 0	(Null)
1 1 1 0 1	*	*	*	*	*	*	*	*	0 0	(Null)
1 1 1 1 0	*	*	*	*	*	*	*	*	0 0	(Null)
1 1 1 1 1	*	*	*	*	*	*	*	*	0 0	(Null)

*Unused, read as zero
**Value before RESET

8

ELECTRICAL SPECIFICATIONS

MAXIMUM RATINGS

Characteristic	Symbol	Value	Unit
Supply Voltage	V_{CC}	−0.3 to +7.0	V
Input Voltage	V_{in}	−0.3 to +7.0	V
Operating Temperature Range	T_A	0 to 70	°C
Storage Temperature	T_{stg}	−55 to +150	°C

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

THERMAL CHARACTERISTICS

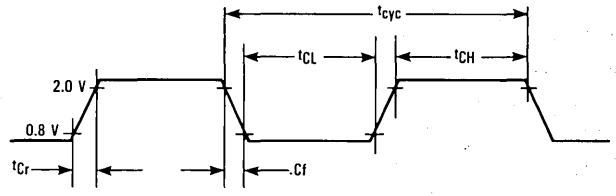
Characteristic	Symbol	Value (Max)	Symbol	Value (Max)	Rating
Thermal Resistance Ceramic (L/LC) Plastic (P)	θ_{JA}	 40 40	θ_{JC}	 15* 20*	°C/W

DC ELECTRICAL CHARACTERISTICS (V_{CC} = 5.0 Vdc ± 5%; T_A = 0 to 70°C; unless otherwise noted)

Characteristic		Symbol	Min	Max	Unit
Input High Voltage	All Inputs	V_{IH}	GND +2.0	V_{CC}	V
Input Low Voltage	All Inputs	V_{IL}	GND −0.3	GND +0.8	V
Input Leakage Current (V_{in} = 0 to 5.25 V)	H1, H3, R/$\overline{W}$, $\overline{RESET}$, CLK, RS1-RS5, $\overline{CS}$	Iin	—	10.0	μA
Hi-Z (Off State) Input Current (V_{in} = 0.4 to 2.4 V)	$\overline{DTACK}$, PC0-PC7, D0-D7 H2, H4, PA0-PA7, PB0-PB7	I_{TSI}	— −0.1	20 −1.0	μA mA
Output High Voltage (I_{Load} = −400 μA, V_{CC} = Min) (I_{Load} = −150 μA, V_{CC} = Min) (I_{Load} = −100 μA, V_{CC} = Min)	 $\overline{DTACK}$, D0-D7 H2, H4, PB0-PB7, PA0-PA7 PC0-PC7	V_{OH}	GND +2.4	—	V
Output low Voltage (I_{Load} = 8.8 mA, V_{CC} = Min) (I_{Load} = 5.3 mA, V_{CC} = Min) (I_{Load} = 2.4 mA, V_{CC} = Min)	 PC3/TOUT, PC5/$\overline{PIRQ}$ D0-D7, $\overline{DTACK}$ PA0-PA7, PB0-PB7, H2, H4, PC0-PC2, PC4, PC6, PC7	V_{OL}	—	0.5	V
Maximum Internal Power Dissipation (Measured at T_A = 0°C)		P_{INT}	—	750	mW
Input Capacitance (V_{in} = 0, T_A = 25°C, f = 1 MHz)		C_{in}	—	15	pF

AC ELECTRICAL SPECIFICATIONS — CLOCK TIMING (see Figure 4)

Characteristic	Symbol	8 MHz		10 MHz		Unit
		Min	Max	Min	Max	
Frequency of Operation	f	2.0	8.0	2.0	10.0	MHz
Cycle Time	t_{cyc}	125	500	100	500	ns
Clock Pulse Width	t_{CL} t_{CH}	55 55	250 250	45 45	250 250	ns
Clock Rise and Fall Time	t_{Cr} t_{Cf}	— —	10 10	— —	10 10	ns



NOTE:
Timing measurements are referenced to and from a low voltage of 0.8 volt and a high voltage of 2.0 volts, unless otherwise noted. The voltage swing through this range should start outside, and pass through, the range such that the rise or fall will be linear between 0.8 volt and 2.0 volts.

Figure 4. Clock Input Timing Diagram

AC ELECTRICAL SPECIFICATIONS (V_{CC} = 5.0 Vdc ± 5%; GND = 0 Vdc; T_A = 0°C to 70°C; unless otherwise noted)
(see Figures 5, 6, 7, 8, and 9)

Num.	Characteristic	8 MHz		10 MHz		Unit
		Min	Max	Min	Max	
1	R/$\overline{W}$, RS1-RS5 Valid of CS Low (Setup Time)	0	—	0	—	ns
2	$\overline{CS}$ Low to R/$\overline{W}$ and RS1-RS5 Invalid (Hold Time	100	—	65	—	ns
3[1]	$\overline{CS}$ Low to CLK Low (Setup Time)	30	—	20	—	ns
4[2]	$\overline{CS}$ Low to Data Out Valid	—	75	—	60	ns
5	RS1-RS5 Valid to Data Out Valid	—	140	—	100	ns
6	CLK Low to $\overline{DTACK}$ Low (Read/Write Cycle)	0	70	0	60	ns
7[3]	$\overline{DTACK}$ Low to $\overline{CS}$ High (Hold Time)	0	—	0	—	ns
8	$\overline{CS}$ or $\overline{PIACK}$ or $\overline{TIACK}$ High to Data Out Invalid (Hold Time	0	—	0	—	ns
9	$\overline{CS}$ or $\overline{PIACK}$ or $\overline{TIACK}$ High to D0-D7 High Impedance	—	50	—	45	ns
10	$\overline{CS}$ or $\overline{PIACK}$ or $\overline{TIACK}$ High to $\overline{DTACK}$ High	—	50	—	45	ns
11	$\overline{CS}$ or PIACK or $\overline{TIACK}$ High to $\overline{DTACK}$ High Impedance	—	100	—	55	ns
12	Data In Valid to $\overline{CS}$ Low (Setup Time)	0	—	0	—	ns
13	$\overline{CS}$ Low to Data In Invalid (Hold Time)	100	—	65	—	ns
14	Port Input Data Valid to H1(H3) Asserted (Setup Time)	100	—	60	—	ns
15	H1(H3) Asserted to Port Input Data Invalid (Hold Time)	20	—	20	—	ns
16	Handshake Input H1(H4) Pulse Width Asserted	40	—	40	—	ns
17	Handshake Input H1(H4) Pulse Width Negated	40	—	40	—	ns
18	H1(H3) Asserted to H2(H4) Negated (Delay Time)	—	150	—	120	ns
19	CLK Low to H2(H4) Asserted (Delay Time)	—	100	—	100	ns
20[4]	H2(H4) Asserted to H1(H3) Asserted	0	—	0	—	ns
21[5]	CLK Low to H2(H4) Pulse Negated (Delay Time)	—	125	—	125	ns
22[9,10]	Synchronized H1(H3) to CLK Low on which $\overline{DMAREQ}$ is Asserted	2.5	3.5	2.5	3.5	Clk. Per.
23	CLK Low on which $\overline{DMAREQ}$ is Asserted to CLK Low on which $\overline{DMAREQ}$ is Negated	2.5	3	2.5	3	Clk. Per.
24	CLK Low to Port Output Data valid (Delay Time) (Modes 0 and 1)	—	150	—	120	ns
25[9,10]	Synchronized H1(H3) to Port Output Data Invalid (Modes 0 and 1)	1.5	2.5	1.5	2.5	Clk. Per.
26	H1 Negated to Port Output Data Valid (Modes 2 and 3)	—	70	—	50	ns
27	H1 Asserted to Port Output Data High Impedance (Modes 2 and 3)	0	70	0	70	ns
28	Read Data Valid to $\overline{DTACK}$ Low (Setup Time)	0	—	0	—	ns
29	CLK Low to Data Output Valid, Interrupt Acknowledge Cycle	—	120	—	100	ns
30[7]	H1(H3) Asserted to CLK High (Setup Time)	50	—	40	—	ns
31	$\overline{PIACK}$ to $\overline{TIACK}$ Low to CLK Low (Setup Time)	50	—	40	—	ns
32[10]	Synchronized $\overline{CS}$ to CLK Low on which $\overline{DMAREQ}$ is Asserted	3	3	3	3	Clk. Per.
33[9,10]	Synchronized H1(H3) to CLK Low on which H2(H4) is Asserted	3.5	4.5	3.5	4.5	Clk. Per.
34	CLK Low to $\overline{DTACK}$ Low Interrupt Acknowledge Cycle (Delay Time)	—	100	—	100	ns

8

AC ELECTRICAL SPECIFICATIONS (Continued)

Num.	Characteristic	8 MHz		10 MHz		Unit
		Min	Max	Min	Max	
35	CLK Low to $\overline{DMAREQ}$ Low (Delay Time	0	120	0	100	ns
36	CLK Low to $\overline{DMAREQ}$ High (Delay Time)	0	120	0	100	ns
37[10]	Synchronized H1(H3) to CLK Low on which $\overline{PIRQ}$ is Asserted	3.5	3.5	3.5	3.5	Clk. Per.
38[10]	Synchronized $\overline{CS}$ to CLK Low on which $\overline{PIRQ}$ is High Impedance	3	3	3	3	Clk. Per.
39	CLK Low to $\overline{PIRQ}$ Low or High Impedance	0	250	0	225	ns
40[8]	TIN Frequency (External Clock) — Prescaler Used	0	1	0	1	f_{clk} (Hz)[6]
41	TIN Frequency (External Clock) — Prescaler Not Used	0	1/8	0	1/8	f_{clk} (Hz)[6]
42	TIN Pulse Width High or Low (External Clock)	55	—	45	—	ns
43	TIN Pulse Width Low (Run/Halt Control)	1	—	1	—	Clk. Per.
44	CLK Low to TOUT High, Low, or High Impedance	0	250	0	225	ns
45	$\overline{CS}$, $\overline{PIACK}$, or $\overline{TIACK}$ High to $\overline{CS}$, $\overline{PIACK}$, or $\overline{TIACK}$ Low	50	—	30	—	ns

NOTES:

1. This specification only applies if the PI/T had completed all operations initiated by the previous bus cycle when $\overline{CS}$ was asserted. Following a normal read or write busy cycle, all operations are complete within three clocks after the falling edge of the CLK pin on which $\overline{DTACK}$ was asserted. If $\overline{CS}$ is asserted prior to completion of these operations, the new bus cycle, and hence, $\overline{DTACK}$ is postponed. If all operations of the previous bus cycle were complete when $\overline{CS}$ was asserted, this specification is made only to insure that $\overline{DTACK}$ is asserted with respect to the falling edge of the CLK pin as shown in the timing diagram not to guarantee operation of the part. If the $\overline{CS}$ setup time is violated, DTACK may be asserted as shown, or may be asserted one clock later.
2. Assuming the RS1-RS5 to data valid time has also expired.
3. This specification imposes a lower bound on $\overline{CS}$ low time, guaranteeing that $\overline{CS}$ will be low for at least 1 CLK period.
4. This specification assures recognition of the asserted edge of H1(H3).
5. This specification applies only when a pulsed handshake option is chosen and the pulse is not shortened due to any early asserted edge of H1 (H3).
6. CLK refers to the actual frequency of the CLK pin, not the maximum allowable CLK frequency.
7. If the setup time on the rising edge of the clock is not met, H1(H3) may not be recognized until the next rising of the clock.
8. This limit applies to the frequency of the signal at TIN compared to the frequency of the CLK signal during each clock cycle. If any period of the waveform at TIN is smaller than the period of the CLK signal at that instant, then it is likely that the timer circuit will completely ignore one cycle of the TIN signal. If these two signals are derived from different sources they will have different instantaneous frequency variations. In this case the frequency applied to the TIN pin must be distinctly less than the frequency at the CLK pin to avoid lost cycles of the TIN signal. With signals derived from different crystal oscillators applied to the TIN and CLK pins with fast rise and fall times, the TIN frequency can approach 80 to 90% of the frequency of the CLK signal without a loss of a cycle of the TIN signal. If these two signals are derived from the same frequency source then the frequency of the signal applied to TIN can be 100% of the frequency at the CLK pin. They may be generated by different buffers from the same signal or one may be an inverted version of the other. The TIN signal may be generated by an "AND" function of the clock and a control signal.
9. The maximum value is caused by a peripheral access (H1(H3) asserted) and bus access ($\overline{CS}$ asserted) occurring at the same time.
10. Synchronized means that the input signal has been seen by the PI/T on the appropriate edge of the clock (rising edge for H1(H3) and falling edge for $\overline{CS}$).

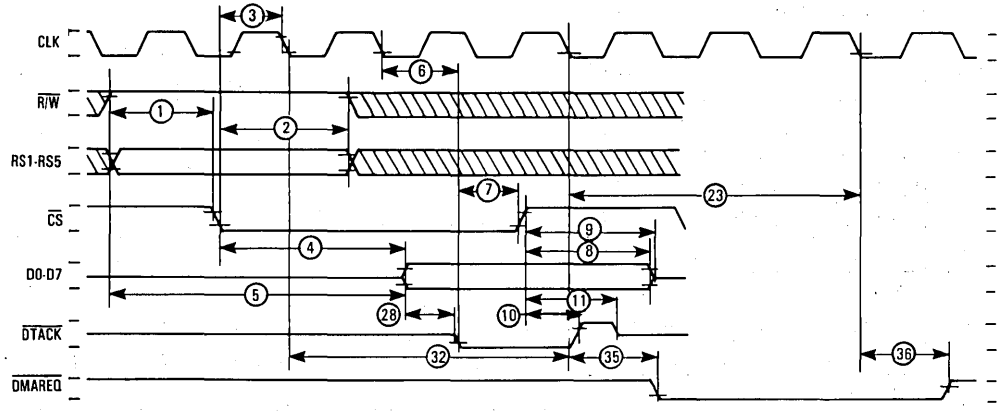


Figure 5. Read Cycle Timing Diagram

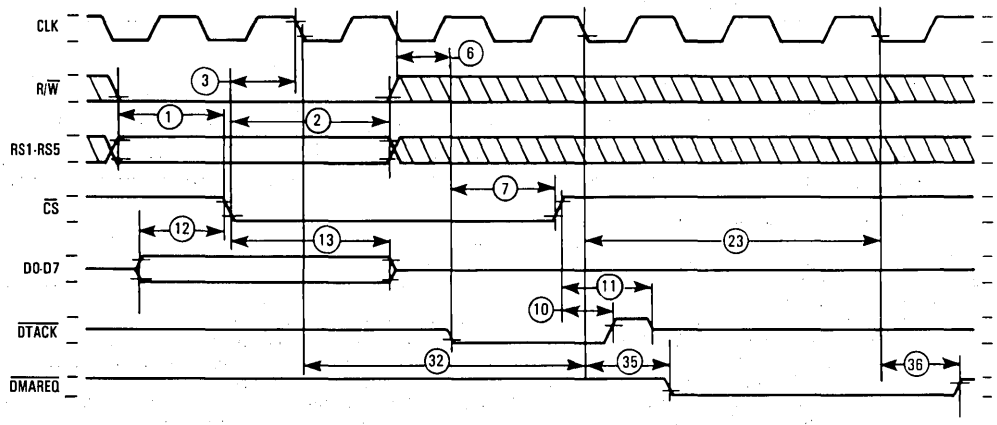
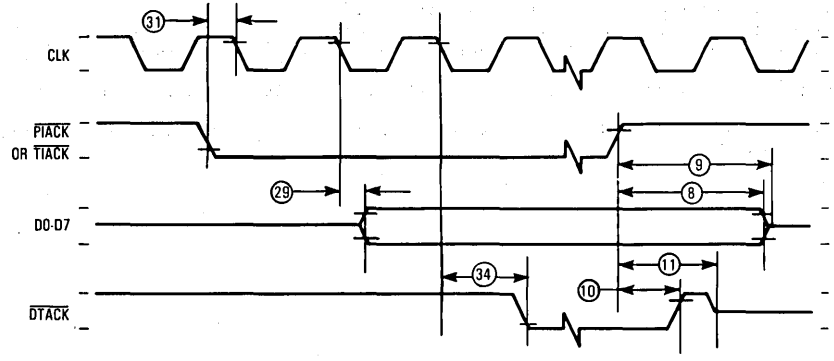
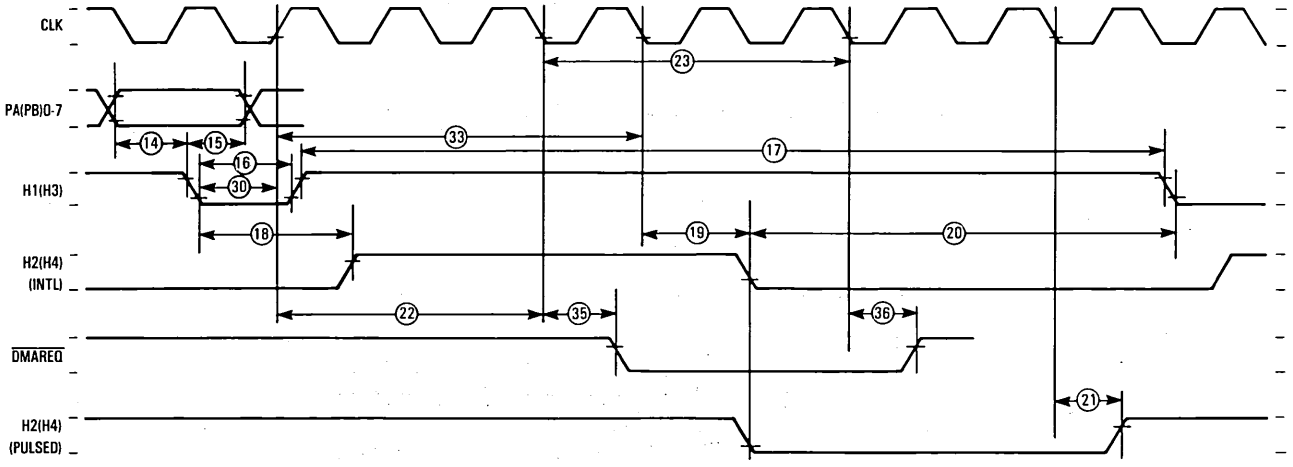


Figure 6. Write Cycle Timing Diagram



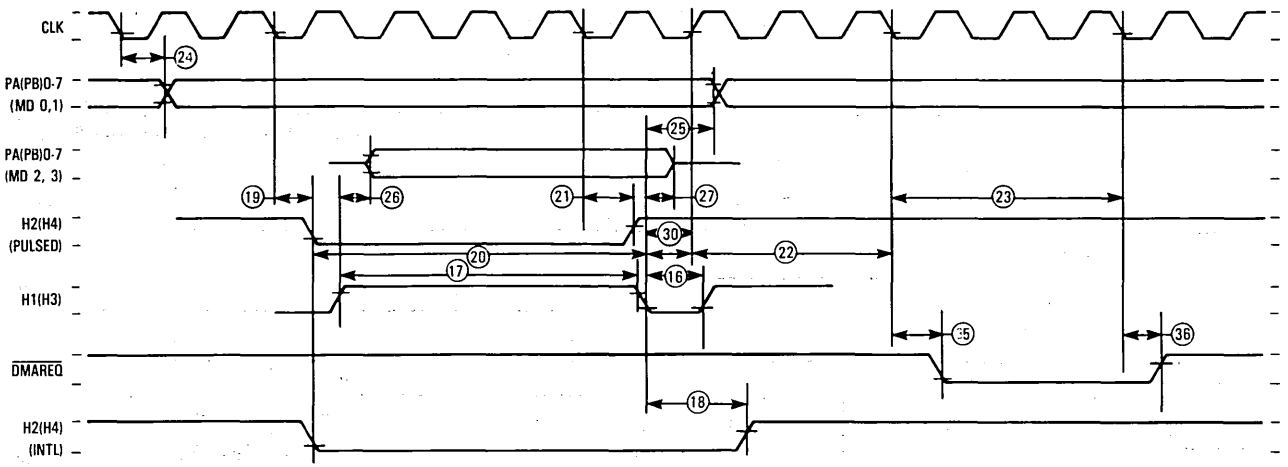
NOTE: Timing measurements are referenced to and from a low voltage of 0.8 volt and a high voltage of 2.0 volts, unless otherwise noted.

Figure 7. IACK Timing Diagram



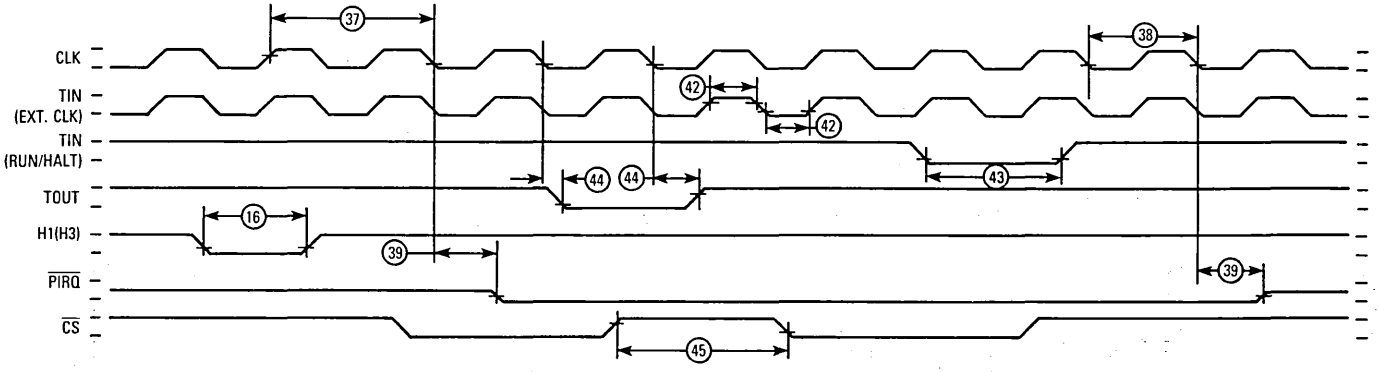
NOTES
1. Timing diagram shows H1, H2, H3 and H4 asserted low.
2. Timing measurements are referenced to and from a low voltage of 0.8 volt and a high voltage of 2.0 volts, unless otherwise noted.

Figure 8. Peripheral Input Timing Diagram



NOTES
1. Timing diagram shows H1, H2, H3 and H4 asserted low.
2. Timing measurements are referenced to and from a low voltage of 0.8 volt and a high voltage of 2.0 volts, unless otherwise noted.

Figure 9. Peripheral Output Timing Diagram

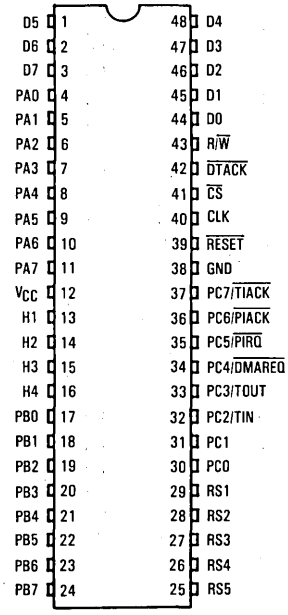


NOTES:
1. Timing diagram shows H1, H2, H3, and H4 asserted low.
2. Timing measurements are referenced to and from a low voltage of 0.8 volt and a high voltage of 2.0 volts, unless otherwise noted.

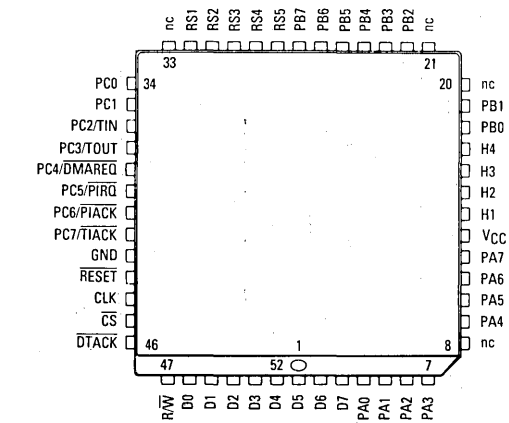
Figure 10. TIN, TOUT, $\overline{\text{PIRQ}}$ Timing Diagram

PIN ASSIGNMENTS

DUAL-IN-PACKAGE



QUAD PACK



8

MC68452

Bus Arbitration Module

The MC68452 is a bipolar asynchronous bus controller which allows multiple local MPU buses to be multiplexed onto a common global bus enabling the local buses to share memory, I/O devices, and communicate with each other easily and efficiently.

- Performs Arbitration For Eight Users Of A Global Bus
- Expandable
- Implements Fixed Physical Priority
- Supports Cycle By Cycle Or Block Mode Arbitration
- 52 ns Max Arbitration Time
- Performs Arbitration For Eight Users Of A 68000 Bus
- 28 Pin Package
- +5.0 Volt Only Operation

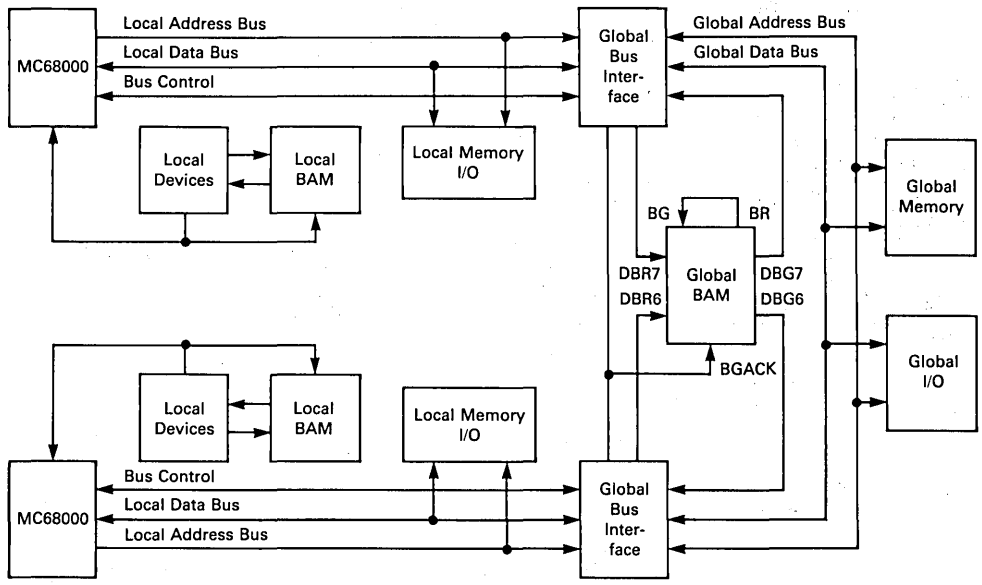


FIGURE 1 — MC68452 IN A MODERATELY COUPLED MULTI-PROCESSOR SYSTEM

8

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

ABSOLUTE MAXIMUM RATINGS

Parameter	Symbol	Value	Unit
Supply Voltage	V_{CC}	−0.5 to +7.0	V
Input Voltage	V_{in}	−0.5 to +7.0	V
Operating Temperature	T_A	0 to +70	°C
Storage Temperature	T_{STG}	−65 to +150	°C

DC ELECTRICAL SPECIFICATIONS (V_{CC} = 5.0 V ±5%, T_A = 0°C to 70°C)

Parameter	Min	Max	Unit	Test Conditions
V_{IH} High Level Input Voltage	2.0		V	
V_{IL} Low Level Input Voltage		0.8	V	
V_{IK} Input Clamp Voltage		−1.5	V	V_{CC} = MIN, I_{in} = −18 mA
V_{OH} High Level Output Voltage	2.4		V	V_{CC} = MIN, I_{OH} = 2.6 mA
V_{OL} Low Level Output Voltage		0.5	V	V_{CC} = MIN, I_{OL} = 24 mA
I_{OS} Output Short Circuit Current(2)		−130	mA	V_{CC} = MAX, V_{OUT} = 0 V
I_{IH} High Level Input Current		20	μA	V_{CC} = MAX, V_{in} = 2.7 V
I_{IL} Low Level Input Current		−0.4	mA	V_{CC} = MAX, V_{in} = 0.4 V
I_{CC} Supply Current	−15	−130	mA	V_{CC} = MAX

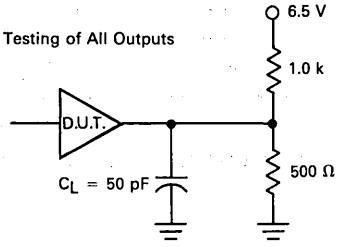
AC ELECTRICAL SPECIFICATIONS (V_{CC} = 5.0 V ± 5%, T_A = 0°C to 70°C, t_{rise} = t_{fall} = 6 ns max)

Parameter	Number*	Min	Max	Units
$\overline{DBRn}$ Low to $\overline{BR}$ Low	1	—	24	ns
$\overline{DBRn}$ High to $\overline{BR}$ High	2	—	31	ns
$\overline{BG}$ Low to $\overline{DBGn}$ Low	3	—	28	ns
$\overline{BGACK}$ Low to $\overline{DBRn}$ High	4	2.0	—	ns
$\overline{BGACK}$ Low to $\overline{DBGn}$ High	5	—	52	ns
$\overline{BGACK}$ High to $\overline{DBGn}$ Low	6	—	52	ns
$\overline{DBRn}$ Low to $\overline{BCLR}$ Low(1)	7	—	28	ns
$\overline{BGACK}$ High to $\overline{BCLR}$ High	8	—	24	ns
$\overline{DBRn}$ High to $\overline{BGACK}$ High	9	0	—	ns

*See Figure 2.

NOTES:
1. Assuming pending request is higher priority than current user.
2. No more than one output should be shorted at a time for no more than one second.

AC TEST CIRCUIT - Functional and Ac Testing of All Outputs



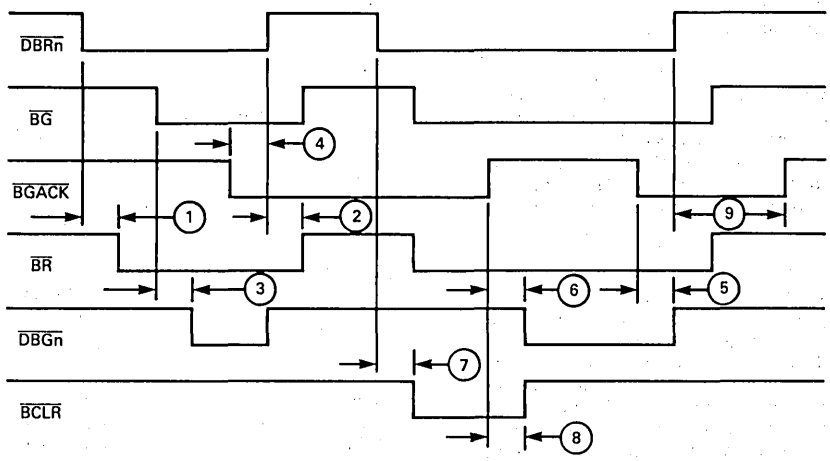


FIGURE 2 — TIMING DIAGRAM

SIGNAL DESCRIPTION

DEVICE BUS REQUEST (DBR7 – DBR0) — These eight inputs are active low and are used to indicate that a user demands a bus cycle(s). The DBR inputs are prioritized with DBR7 as the highest and DBR0 the lowest. This priority scheme is only used when two or more devices have pending requests.

DEVICE BUS GRANT (DBG7 – DBG0) — These active low outputs indicate that a user has obtained the bus, should bring BGACK active, and begin the transfer. The DBGn is removed when the user brings BGACK active.

BUS REQUEST (BR) — This output is the logical AND of all the DBRn inputs. This active low signal indicates that one or more of the DBR inputs are active.

BUS GRANT (BG) — This active low input is used to enable the DBGn outputs.

BUS GRANT ACKNOWLEDGE (BGACK) — This active low input indicates that a user has taken control of the bus. Each user must be able to generate this signal. When BGACK becomes active the DBGn will be removed.

BUS CLEAR (BCLR) — This active low output indicates that a higher priority device has a request pending. This signal is enabled by the BGACK being active. How this signal is used is totally up to the system designer. The BAM cannot force any device off the bus.

LATCH ENABLE INPUT (LEI) — This active low input is used to cascade BAMs to allow more than eight bus users. This signal should be the logical AND of all BR outputs of the BAM circuits. This signal is used to close the input request latch in all BAM circuits whenever there is a request pending in any part. This allows the encoding/decoding to proceed in all parts without spiking the outputs of any parts.

8

THEORY OF OPERATION

The BAM provides a central arbitration function by utilizing a separate request-grant pair for each user as opposed to multiplexing all requests onto a single line and daisy-chaining the grant. Each BAM circuit has eight DBR-DBG request-grant pairs. When a device desires to use the bus it brings its DBR low (active). Since the BAM circuit operates asynchronously there are no restrictions placed on the active transition of the DBRn signals. There are however, two minor restrictions placed on the inactive transition. The restrictions are: 1) all requests must remain active until they receive their grant signal and bring the BGACK active, and 2) the request is removed before the BGACK is released.

Each bus request line has a corresponding bus grant line (DBGn). After a requesting device brings its request active it must monitor the DBGn signal. When this signal becomes active the user has obtained the bus, should bring BGACK active, and begin transferring. The device can maintain control of the shared bus as long as the BGACK signal remains active. This three level handshake (request-grant-acknowledge) allows the BAM to support single or block type transfers with equal ease.

When the device transfer is complete it should remove the BGACK signal to allow the arbiter to determine the next user. In order for the BAM to operate properly the three level handshake must be used even if only single transfers are supported. When the BAM detects the BGACK going inactive it initiates another arbitration cycle, therefore the BGACK signal must be used properly. Although the DBGn outputs are disabled when BGACK is active the BAM has the current user number latched internally. During the transfer the current user number is compared with the highest active pending request. If the pending device is higher priority the BUS CLEAR (BCLR) will become active. Any circuitry for forcing devices off the bus and re-arbitrating must be provided external to the BAM.

Since the BAM is an asynchronous device, the bus grant outputs are *not* guaranteed to be spike free, although the internal delay paths have been equalized to minimize the occurrence of output spikes. The spikes are caused by metastabling of the internal request input latch. The arbitration cycle begins when one of the request inputs makes an active going transition. The request input latch is closed to prevent the encoder/decoder path from changing during the cycle. Requests that change just as the latches are closing may cause the latch to metastable or in essence attempt to latch in an analog level in the feedback path of the latch. If this metastable occurs in a request that is higher priority than the request that initiated the cycle the two bus grant outputs will spike alternately for approximately 50 ns. The metastable state should resolve itself into a valid digital state within this time and the outputs will stabilize to a valid state. These spikes can be removed by disabling the bus grant outputs during the arbitration cyle as shown in the circuit of Figure 3.

The BAM is an asynchronous state machine and is sensitive to noise at certain state transition times. The first critical time is the active transition of the DBRn inputs. These inputs must have a 6.0 ns maximum fall time to insure proper state transitions inside the part. This 6.0 ns specification can be easily met by having an LSTTL buffer drive a single DBRn input directly. Violation of this parameter *may* force the BAM into an invalid state. In this state the part will ignore all inputs except BGACK. The state can be cleared by bringing BGACK active for at least 20 ns and returning it to the inactive state.

The second critical time is the inactive transition of BGACK. This signal must have a 6.0 ns maximum rise time with no transitions for a minimum of 15 ns after reaching threshold. Violation of these two parameters *may* force the BAM into the same invalid state as discussed in the previous paragraph. Recovery procedures are the same as above.

MODES OF OPERATION

Local Central Arbiter

Figure 4 shows the BAM circuit in a local bus configuration. In this mode, the BAM serves as a central bus controller as opposed to the distributed control of a daisy-chain arbitration scheme. As shown, the BAM provides the interface between the local bus masters and the MC68000 MPU. The BR-BG pair of each local

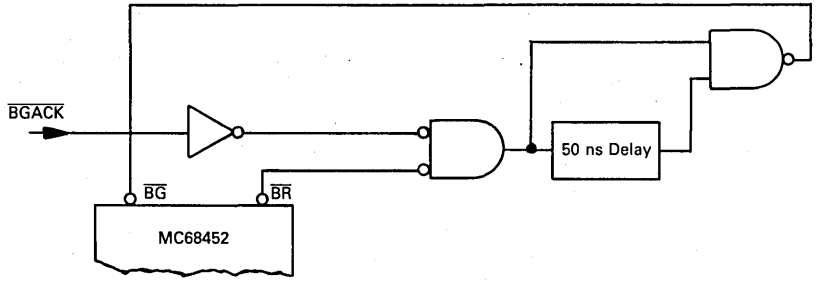


FIGURE 3 — CIRCUIT TO DISABLE GRANT OUTPUTS DURING ARBITRATION CYCLE

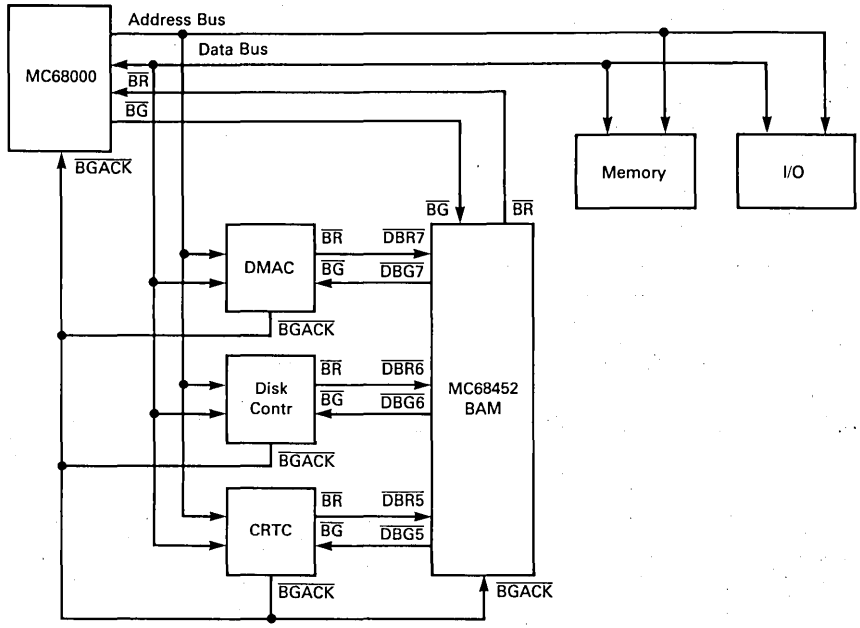


FIGURE 4 — LOCAL BUS ARBITER CONFIGURATION

bus master connect directly to the DBRn-DBGn pair of BAM. The BR-BG pair of the processor connect directly to the BR-BG pair of the BAM.

Whenever a device desires bus access it brings its BR (DBRn) active. The BAM detects the active request and makes a bus request (BR) to the processor. Within 1.5 clocks the MPU will return the BG to the BAM. This signifies that the requesting device can obtain control of the bus at the end of the current bus cycle. When the BAM receives the BG from the processor it will issue the DBGn to the highest priority requesting device. When the next bus master observes the end of the current bus cycle, it should bring BGACK active and begin to transfer. As long as the BGACK is held active the user can continue bus cycles indefinitely. When the current user completes the transfer it should release the BGACK to allow others to use the bus. If there is another request pending, the MC68000 MPU will remain in the idle state and another device will be allowed to become bus master. When no devices have a pending request the BAM will remove the BR from the MPU and allow the processor to resume execution.

Global Bus Arbiter

Figure 1 shows a moderately coupled multiprocessor system utilizing the BAM circuit. The global BAM serves as the central bus controller of the shared global bus. This allows multiple local buses to share mass storage and the addition of the more local processors to increase system throughput.

Figure 5 shows the global interface of each local bus. Please note that this circuit is used as an example. It will only support local bus masters with arbitration at the end of every bus cycle. However, the BAM does not preclude global bus masters and will certainly support block transfers with the proper interface circuitry. The local bus generates the DBRn by detecting some global address on the address bus. As shown, the AS is used in the DBRn equation to eliminate any switching noise on the address decode signal. The local DTACK signal also enters the DBRn equation to remove the request before BGACK is released. In this configuration there is no MPU to be removed from control of the global bus. Therefore, the BR output is connected directly to the BG input of the BAM. This allows a short delay before the DBGn outputs are enabled to allow encoding/decoding to proceed without switching noise appearing on the outputs. The rest of the circuitry shown in Figure 4 is used to generate the output enable for address/data three-state drivers and the BGACK for this user. Since the DBGn signal is removed when BGACK becomes active the OEn is the logical OR of the DBGn and the BGACKn for each user. The BGACK is generated when the global bus generates the global DTACK (GDTACK).

8

This will bring the clear active on the flip-flop which brings BGACK low and removes the DBGn. After the local bus recognizes the DTACK it will finish the cycle by removing ASn. The rising edge of the AS clocks the flip-flop and removes the BGACK which initiates another arbitration cycle. By removing BGACK at the end of every cycle this circuit implements single cycle transfers. This is probably the most efficient way to operate if the MPU is performing the transfers. However, this circuit can support local bus masters other than the MPU. If a DMA is performing the transfer, the clock to the BGACK flop might be the END signal from the DMA controller. Certainly both modes could be supported easily by multiplexing these two clocks together using single cycle for program I/O cycles and block mode for DMA cycles.

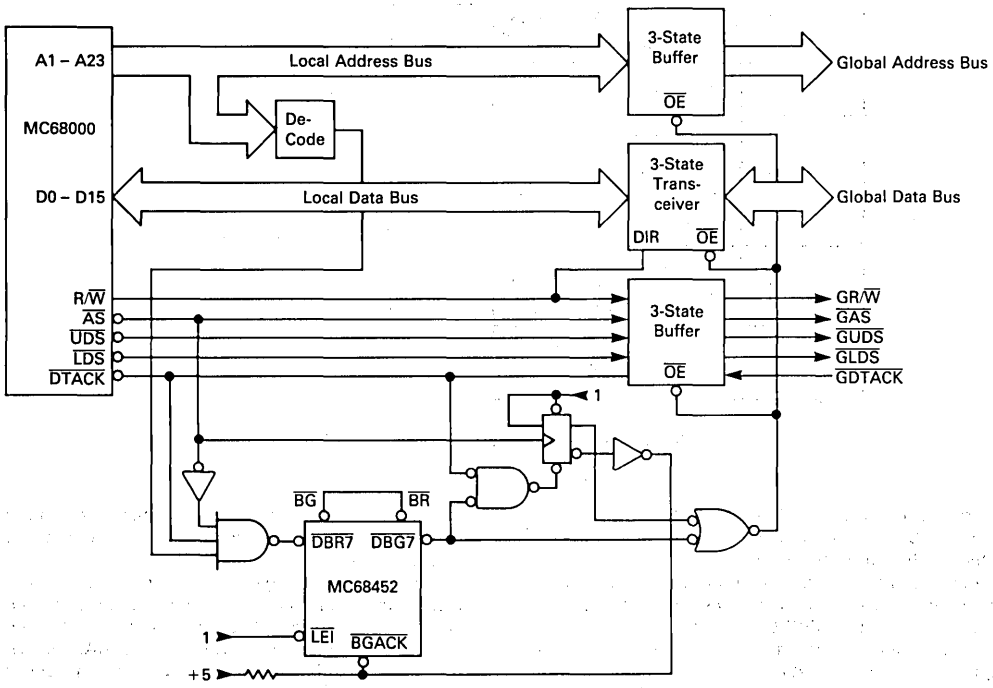


FIGURE 5 — GLOBAL BUS ARBITER CONFIGURATION

8

Expanding the Arbiter

If a system requires more than eight bus masters the BAMs can be cascaded to any number of users required. Figure 6 shows a circuit that can support up to 64 bus masters. To support 64 users, nine BAMs are required. Eight parts supply the DBRn-DBGn request-grant pairs, and one part monitors status of the eight parts in parallel to supply the BCLR signal and handle the eight asynchronous BR outputs. As shown, the BR outputs of the parallel parts are connected to the DBRn inputs of the expansion BAM. This preserves the physical priority and handles the asynchronous expansion. By expanding in this way, the BAMs can operate correctly, however, the arbitration cycle will now require 80 ns max.

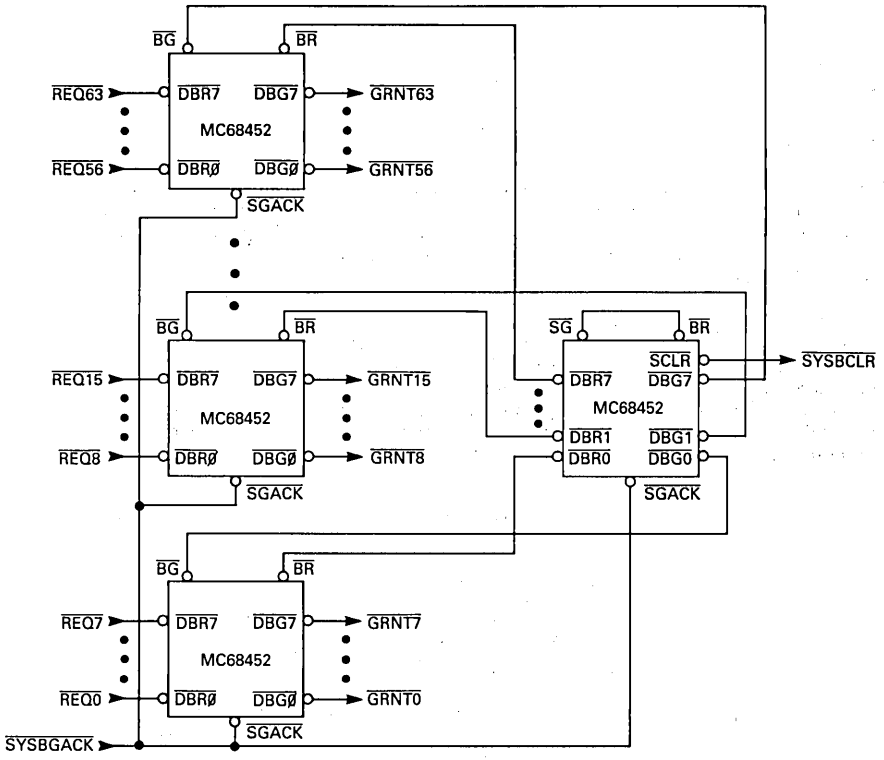
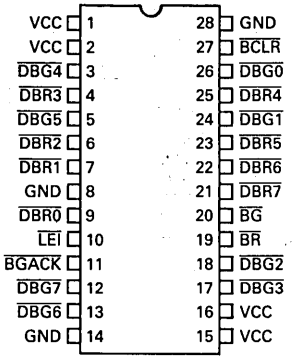


FIGURE 6 — EXPANDED BAM CONFIGURATION

BAM SIGNAL DIAGRAM



8

MC68901

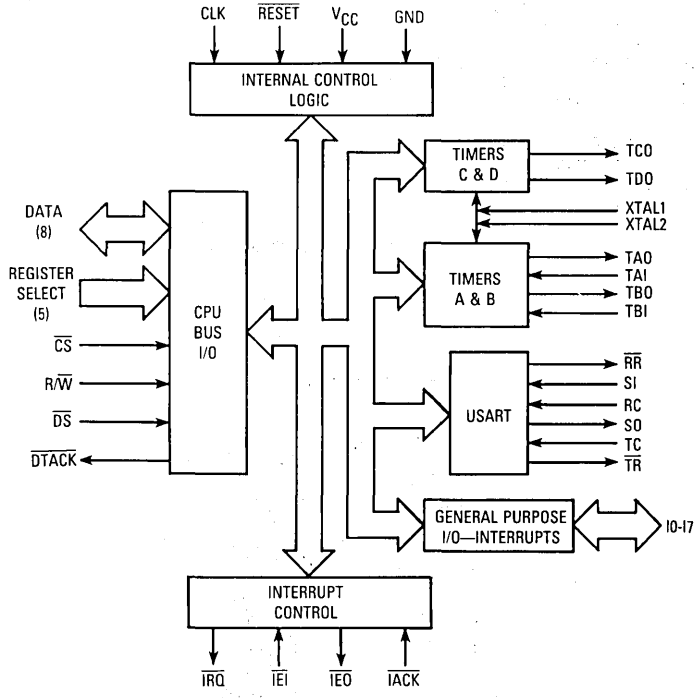
Technical Summary
Multi-Function Peripheral

The MC68901 multi-function peripheral (MFP) is a member of the M68000 Family of peripherals. The MFP directly interfaces with the MC68000 microprocessor via the asynchronous bus structure. Both vectored and polled interrupt schemes are supported with the MFP providing unique vector number generation for each of its 16 interrupt sources. Additionally, handshake lines are provided to facilitate DMAC interfacing.

The MC68901 performs many of the functions common to most microprocessor-based systems. The resources available to the user include:

- Eight Individually Programmable I/O Pins with Interrupt Capability
- 16-Source Interrupt Controller with Individual Source Enable and Masking
- Four Timers, Two of which are Multi-Mode Timers
- Single-Channel Full-Duplex Universal Synchronous/Asynchronous Receiver-Transmitter (USART) that Supports Asynchronous and with the Addition of a Polynomial Generator Checker that Supports Byte Synchronous Formats

BLOCK DIAGRAM



8

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

By incorporating multiple functions within the MFP, the system designer retains flexibility while minimizing device count.

From a programmer's point of view, the versatility of the MFP may be attributed to its register set. The registers are well organized and allow the MFP to be easily tailored to a variety of applications. All of the 24 registers are also directly addressable which simplifies programming. The register map is shown in Table 1.

SIGNAL DESCRIPTION

The following paragraphs contain a brief description of the input and output signals. These signals can be functionally organized into groups as shown in Figure 1.

NOTE

The terms **assertion** and **negation** will be used extensively. This is done to avoid confusion when dealing with a mixture of "active low" and "active high" signals. The term **assert** or **assertion** is used to indicate that a signal is active or true, independent of whether that level is represented by a high or low voltage. The term **negate** or **negation** is used to indicate that a signal is inactive or false.

V_{CC} AND GND

These inputs supply power to the MFP. The V_{CC} is powered at +5 volts, and GND is the ground connection.

CLOCK (CLK)

The clock input is a single-phase TTL-compatible signal used for internal timing. This input should not be gated off at any time and must conform to minimum and maximum pulse width times. The clock is not necessarily the system clock in frequency or phase.

Table 1. MFP Register Map

Address						Abbreviation	Register Name
Hex	Binary						
	RS5	RS4	RS3	RS2	RS1		
01	0	0	0	0	0	GPDR	General Purpose I/O Data Register
03	0	0	0	0	1	AER	Active Edge Register
05	0	0	0	1	0	DDR	Data Direction Register
07	0	0	0	1	1	IERA	Interrupt Enable Register A
09	0	0	1	0	0	IERB	Interrupt Enable Register B
0B	0	0	1	0	1	IPRA	Interrupt Pending Register A
0D	0	0	1	1	0	IPRB	Interrupt Pending Register B
0F	0	0	1	1	1	ISRA	Interrupt In-Service Register A
11	0	1	0	0	0	ISRB	Interrupt In-Service Register B
13	0	1	0	0	1	IMRA	Interrupt Mask Register A
15	0	1	0	1	0	IMRB	Interrupt Mask Register B
17	0	1	0	1	1	VR	Vector Register
19	0	1	1	0	0	TACR	Timer A Control Register
1B	0	1	1	0	1	TBCR	Timer B Control Register
1D	0	1	1	1	0	TCDCR	Timers C and D Control Register
1F	0	1	1	1	1	TADR	Timer A Data Register
21	1	0	0	0	0	TBDR	Timer B Data Register
23	1	0	0	0	1	TCDR	Timer C Data Register
25	1	0	0	1	0	TDDR	Timer D Data Register
27	1	0	0	1	1	SCR	Synchronous Character Register
29	1	0	1	0	0	UCR	USART Control Register
2B	1	0	1	0	1	RSR	Receiver Status Register
2D	1	0	1	1	0	TSR	Transmitter Status Register
2F	1	0	1	1	1	UDR	USART Data Register

NOTE: Hex addresses assume that RS1 connects with A1, RS2 connects with A2, etc. and that $\overline{DS}$ is connected to $\overline{LDS}$ on the MC68000 or $\overline{DS}$ is connected to $\overline{DS}$ on the MC68008.

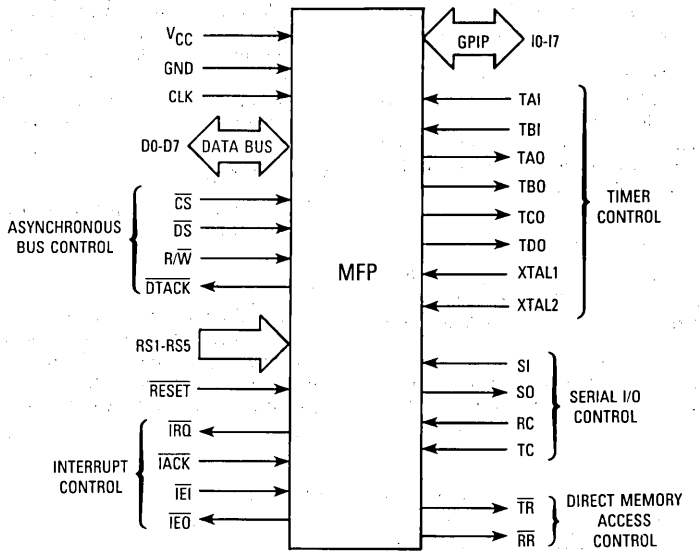


Figure 1. Input and Output Signals

DATA BUS (D0 through D7)

This three-state bidirectional bus is used to transmit data to or receive data from the MFP's internal registers during a processor read or write cycle, respectively. During an interrupt acknowledge cycle, the data bus is used to pass a vector number to the processor. The MFP must be located on data bus lines D0-D7 when used with an MC68000, MC68008, or MC68010 and on data lines D24-D31 when used with an MC68020, if vectored interrupts are to be used.

ASYNCHRONOUS BUS CONTROL

Asynchronous data transfers are controlled by chip select, data strobe, read/write, and data transfer acknowledge. The register select lines, RS5-RS1, select an internal MFP register for a read or write operation. The reset line initializes the MFP registers and the internal control signals.

Chip Select ($\overline{CS}$)

This active low input activates the MFP for internal register access. $\overline{CS}$ and $\overline{IACK}$ must not be asserted at the same time.

Data Strobe ($\overline{DS}$)

This active low input is part of the internal chip select and interrupt acknowledge functions.

Read/Write (R/$\overline{W}$)

This input defines the current bus cycle as a read (high) or a write (low) cycle.

Data Transfer Acknowledge ($\overline{DTACK}$)

This active low, three-state output signals the completion of the operation phase of a bus cycle to the processor. If the bus cycle is a processor read, the MFP asserts $\overline{DTACK}$ to indicate that the information on the data bus is valid. If the bus cycle is a processor write to the MFP, $\overline{DTACK}$ acknowledges the acceptance of the data by the MFP. $\overline{DTACK}$ will be asserted only by an MFP that has $\overline{CS}$ or $\overline{IACK}$ (and $\overline{IEI}$) asserted.

REGISTER SELECT BUS (RS1 through RS5)

The register select bus selects an internal MFP register during a read or write operation.

RESET ($\overline{RESET}$)

This active low input will initialize the MFP during powerup or in response to a total system reset.

INTERRUPT CONTROL

The interrupt request and interrupt acknowledge signals are handshake lines for a vectored interrupt scheme. Interrupt enable in and interrupt enable out implement a daisy-chained interrupt structure.

Interrupt Request ($\overline{IRQ}$)

This active low, open-drain output signals to the processor that an interrupt is pending from the MFP. There are 16 interrupt channels that can generate an interrupt request. Clearing the interrupt pending registers (IPRA and IPRB) or clearing the interrupt mask registers (IMRA and IMRB) will cause the $\overline{IRQ}$ to be negated. $\overline{IRQ}$ will also be negated as the result of an interrupt acknowledge cycle, unless additional interrupts are pending in the MFP.

Interrupt Acknowledge ($\overline{IACK}$)

If both $\overline{IRQ}$ and $\overline{IEI}$ are asserted, the MFP will begin an interrupt acknowledge cycle when $\overline{IACK}$ and $\overline{DS}$ are asserted. The MFP will supply a unique vector number to

the processor which corresponds to the particular channel requesting interrupt service. In a daisy-chained interrupt structure, all devices in the chain must have a common IACK. CS and IACK must not be asserted at the same time.

Interrupt Enable In (IEI)

This active low input, together with the IEO signal, provides a daisy-chained interrupt structure for a vectored interrupt scheme. IEI indicates that no higher priority device is requesting interrupt service. So, the highest priority MFP in the chain should have its IEI pin tied low. During an interrupt acknowledge cycle, an MFP with a pending interrupt is not allowed to pass a vector number to the processor until its IEI pin is asserted. When the daisy-chain option is not implemented, all MFPs should have their IEI pin tied low.

Interrupt Enable Out (IEO)

This active low output, together with the IEI signal, provides a daisy-chained interrupt structure for a vectored interrupt scheme. The IEO of a particular MFP signals lower priority devices that neither it nor any other higher priority device is requesting interrupt service. When a daisy-chain is implemented, IEO is tied to the next lower priority MFP's IEI input. The lowest priority MFP's IEO is not connected. When the daisy-chain option is not implemented, IEO is not connected.

GENERAL PURPOSE I/O INTERRUPT LINES (I0 through I7)

These lines constitute an 8-bit pin-programmable I/O port with interrupt capability. The data direction register (DDR) individually defines each line as either a high-impedance input or a TTL-compatible output. As an input, each line can generate an interrupt on the user selected transition of the input signal.

TIMER CONTROL

These lines provide internal timing and auxiliary timer control inputs required for certain operating modes. Additionally, the timer outputs are included in this group.

Timer Inputs (TAI and TBI)

These inputs are control signals for timers A and B in the pulse width measurement mode and the event count mode. These signals generate interrupts at the same priority level as the general purpose I/O interrupt lines I4 and I3, respectively, when in the pulse width measurement mode. While I4 and I3 do not have interrupt capability when the timers are operated in this mode, they may still be used for I/O.

Timer Outputs (TAO, TBO, TCO, and TDO)

Each timer has an associated output which toggles when its main counter counts through 01 (hexadecimal) regardless of which operational mode is selected. When in the delay mode, the timer output will be a square wave with a period equal to two timer cycles. This output may be used to supply the universal synchronous/asynchronous receiver-transmitter (USART) baud rate clocks.

Timer Clock (XTAL1 and XTAL2)

This input provides the timing signal for the four timers. A crystal can be connected between the timer clock inputs, XTAL1 and XTAL2, or XTAL1 can be driven with a TTL-level clock while XTAL2 is not connected. The following crystal parameters are suggested:

a) Parallel resonance, fundamental mode AT-cut, HC6 or HC33 holder
b) Frequency tolerance measured with 18 picofarads load (0.1% accuracy) — drive level 10 microwatts
c) Shunt capacitance equals 7 picofarads
d) Series resistance:
   $2.0 < f < 2.7$ MHz; $Rs \leq 300$ ohms
   $2.8 < f < 4.0$ MHz; $Rs \leq 150$ ohms

SERIAL I/O

The full duplex serial channel is implemented by a serial input line. The independent receive and transmit sections may be clocked by separate timing signals on the receive clock input and the transmitter clock input.

Serial Input (SI)

This input line is the USART receiver data input. This input is not used in the USART loopback mode.

Serial Output (SO)

This output line is the USART transmitter data output. This output is in a high-impedance state after a device reset.

Receiver Clock (RC)

This input controls the serial bit rate of the receiver. The signal may be supplied by the timer output lines or by an external TTL-level clock which meets the minimum and maximum cycle times. This clock is not used in the USART loopback mode.

Transmitter Clock (TC)

This input controls the serial bit rate of the transmitter. This signal may be supplied by the timer output lines or by an external TTL-level clock which meets the minimum and maximum cycle times.

DIRECT MEMORY ACCESS CONTROL

The USART section of the MFP supports direct memory access transfers through its receiver ready and transmitter ready status lines.

Receiver Ready (RR)

This active low output reflects the receiver buffer full status (bit 7 in the Receiver Status Register) for DMA operations.

Transmitter Ready (TR)

This active low output reflects the transmitter buffer empty (bit 7 in the Transmitter Status Register) for DMA operations.

SIGNAL SUMMARY

Table 2 is a summary of all the signals discussed in the previous paragraphs.

Table 2. Signal Summary

Signal Name	Mnemonic	I/O	Active State
Power Input	V_{CC}	Input	High
Ground	GND	Input	Low
Clock	CLK	Input	N/A
Chip Select	$\overline{CS}$	Input	Low
Data Strobe	$\overline{DS}$	Input	Low
Read/Write	R/$\overline{W}$	Input	Read – High, Write – Low
Data Transfer Acknowledge	$\overline{DTACK}$	Output	Low
Register Select Bus	RS1–RS5	Input	N/A
Data Bus	D0-D7	I/O	N/A
Reset	$\overline{RESET}$	Input	Low
Interrupt Request	$\overline{IRQ}$	Output	Low
Interrupt Acknowledge	$\overline{IACK}$	Input	Low
Interrupt Enable In	$\overline{IEI}$	Input	Low
Interrupt Enable Out	$\overline{IEO}$	Output	Low
General Purpose I/O	I0–I7	I/O	N/A
Timer Clock	XTAL1, XTAL2	Input	N/A
Timer Inputs	TAI, TBI	Input	N/A
Timer Outputs	TAO, TBO, TCO, TDO	Output	N/A
Serial Input	SI	Input	N/A
Serial Output	SO	Output	N/A
Receiver Clock	RC	Input	N/A
Transmitter Clock	TC	Input	N/A
Receiver Ready	$\overline{RR}$	Output	Low
Transmitter Ready	$\overline{TR}$	Output	Low

BUS OPERATION

The following paragraphs explain the control signals and bus operation during data transfer operations and reset.

8

DATA TRANSFER OPERATIONS

Transfer of data between devices involves the following pins:

Register Select Bus — RS1 through RS5
Data Bus — D0 through D7
Control Signals

The address and data buses are separate parallel buses used to transfer data using an asynchronous bus structure. In all cases, the bus master assumes responsibility for deskewing all signals it issues at both the start and end of a cycle. Additionally, the bus master is responsible for deskewing the acknowledge and data signals from the peripheral devices.

Read Cycle

To read an MFP register, $\overline{CS}$ and $\overline{DS}$ must be asserted, and R/$\overline{W}$ must be high. The MFP will place the contents of the register which is selected by the register select bus (RS1 through RS5) on the data bus (D0 through D7) and then assert $\overline{DTACK}$. The register addresses are shown in Table 1.

After the processor has latched the data, it negates $\overline{DS}$. The negation of either $\overline{CS}$ or $\overline{DS}$ will terminate the read operation. The MFP will drive $\overline{DTACK}$ high and place it and the data bus in the high-impedance state. The timing for a read cycle is shown in Figure 2.

Write Cycle

To write a register, $\overline{CS}$ and $\overline{DS}$ must be asserted, and R/$\overline{W}$ must be low. The MFP will decode the address bus to determine which register is selected. Then the register will be loaded with the contents of the data bus, and $\overline{DTACK}$ will be asserted. When the processor recognizes $\overline{DTACK}$, it will negate $\overline{DS}$. The write cycle is terminated when either $\overline{CS}$ or $\overline{DS}$ is negated. The MFP will drive $\overline{DTACK}$ high and place it in the high-impedance state. The timing for a write cycle is shown in Figure 3.

INTERRUPT ACKNOWLEDGE OPERATION

The MFP has 16 interrupt sources: eight internal and eight external. When an interrupt request is pending, the MFP will assert $\overline{IRQ}$. In a vectored interrupt scheme, the processor will acknowledge the interrupt request by performing an interrupt acknowledge cycle. $\overline{IACK}$ and $\overline{DS}$

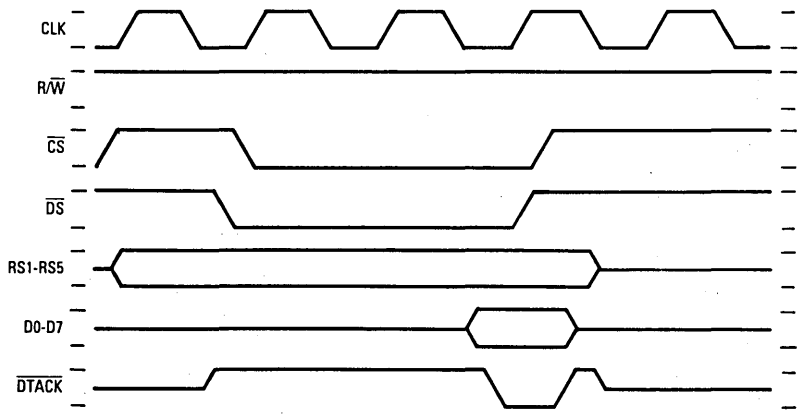


Figure 2. Read Cycle Timing

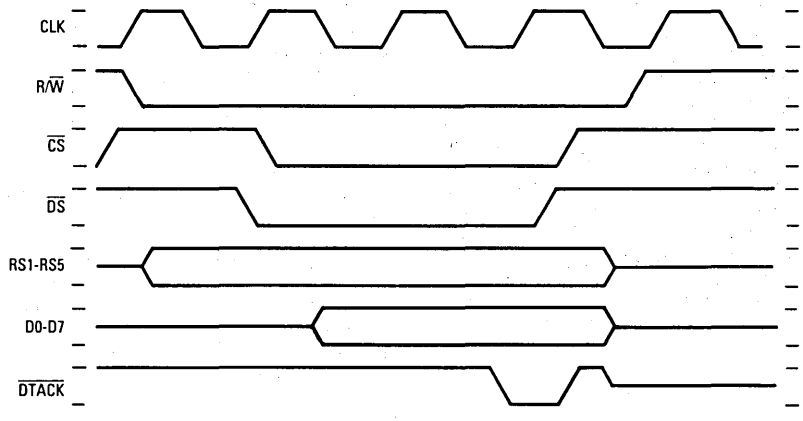


Figure 3. Write Cycle Timing

8

will be asserted. The MFP responds to the IACK signal by placing a vector number on the data bus. This vector number corresponds to the particular interrupt channel requesting service.

When the MFP asserts DTACK to indicate that valid data is on the bus, the processor will latch the data and terminate the bus cycle by negating DS. When either DS or IACK is negated, the MFP will terminate the interrupt acknowledge operation by driving DTACK high and placing it in the high-impedance state. IRQ will be negated as a result of the IACK cycle unless additional interrupts are pending.

The MFP can be part of a daisy-chain interrupt structure which allows multiple MFPs to be placed at the same interrupt level by sharing a common IACK signal. A daisy-chain priority scheme is implemented with signals IEI and IEO. IEI indicates that no higher priority device is requesting interrupt service. IEO signals lower priority devices that neither this device nor any higher priority MFP is requesting service. To daisy-chain MFPs, the highest priority MFP has its IEI tied low and successive MFPs have their IEI connected to the next higher priority MFP's IEO. Note that when the daisy-chain interrupt structure is not implemented, the IEIs of all MFPs must be tied low and the IEOs left unconnected.

When the processor initiates an interrupt acknowledge cycle by driving IACK and DS, the MFP, whose IEI is low, may respond with a vector number if an interrupt is pending. If this device does not have a pending interrupt, IEO is asserted which allows the next lower priority device to respond to the interrupt acknowledge. When an MFP propagates IEO, it will not drive the data bus nor DTACK during the interrupt acknowledge cycle. The timing for an IACK cycle is shown in Figure 4.

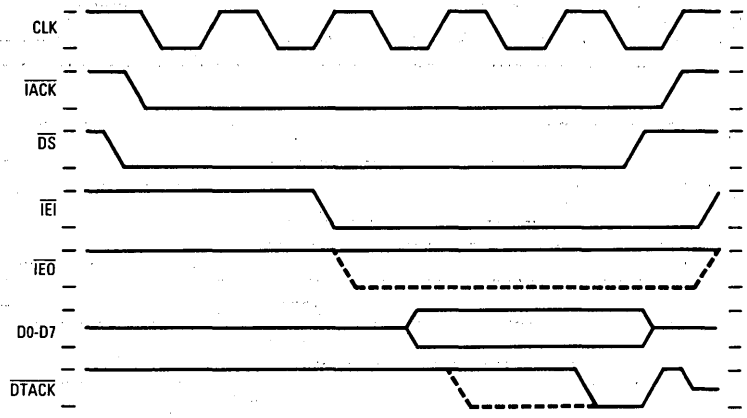


Figure 4. IACK Cycle Timing

RESET OPERATION

The reset operation will initiate the MFP to a known state. The reset operation requires that the RESET input be asserted for a minimum of two microseconds. During a device reset condition, all internal MFP registers are cleared except for the timer data registers (TADR, TBDR, TCDR, and TDDR), the USART data register (UDR), and the transmitter status register (TSR). All timers are stopped, the USART receiver and transmitter are disabled, and the serial output (SO) line is placed in high impedance. The interrupt channels are also disabled and any pending interrupts are cleared. In addition, the general purpose interrupt I/O lines are placed in the high-impedance input mode, and the timer outputs are driven low. External MFP signals are negated. Since the vector register (VR) is initialized to a $00, an uninitialized MFP may not respond to an interrupt acknowledge cycle with the uninitialized interrupt vector, $0F.

8

INTERRUPT STRUCTURE

In an M68000 system, the MFP will be assigned to one of the seven possible interrupt levels. All interrupt service requests from the MFP's 16 interrupt channels will be presented at this level. As an interrupt controller, the MFP will internally prioritize its 16 interrupt sources. Additional interrupt sources may be placed at the same interrupt level by daisy-chaining multiple MFPs. The MFPs will be prioritized by their position in the chain.

INTERRUPT PROCESSING

Each MFP provides individual interrupt capability for its various functions. When an interrupt is received on one of the external interrupt channels or from one of the eight internal sources, the MFP will request interrupt service. The 16 interrupt channels are assigned a fixed priority so that multiple pending interrupts are serviced according to their relative importance. Since the MFP can internally generate 16 vector numbers, the unique vector number which corresponds to the highest priority channel that has a pending interrupt is presented to the processor during an interrupt acknowledge cycle. This unique vector number allows the processor to immediately begin execution of the interrupt handler for the interrupting source, decreasing interrupt latency.

Interrupt Channel Prioritization

The 16 interrupt channels are prioritized from highest to lowest, with General Purpose Interrupt 7 (I7) being the highest and I0 the lowest. The priority of the interrupt is determined by the least-significant four bits in the interrupt vector number which are internally generated by the MC68901. Pending interrupts are presented to the processor in order of priority unless they have been masked. By selectively masking interrupts, the channels are in effect re-prioritized.

Interrupt Vector Number

During an interrupt acknowledge cycle, a unique 8-bit interrupt vector number is presented to the system which corresponds to the specific interrupt source that is requesting service.

7	6	5	4	3	2	1	0
V7	V6	V5	V4	IV3	IV2	IV1	IV0

V7–V4 — Copied from the vector regiser
IV3–IV0 — Determine highest priority channel requesting interrupt.

Vector Register (VR, $17)

This 8-bit register determines the four most-significant bits in the interrupt vector format and which end-of-interrupt mode is used in a vectored interrupt scheme. The vector register should be written to before writing to the interrupt mask or enable registers to ensure that the MC68901 responds to an interrupt acknowledge cycle with

IV3	IV2	IV1	IV0	Description
1	1	1	1	General Purpose Interrupt 7 (I7)
1	1	1	0	General Purpose Interrupt 6 (I6)
1	1	0	1	Timer A
1	1	0	0	Receiver Buffer Full
1	0	1	1	Receive Error
1	0	1	0	Transmit Buffer Empty
1	0	0	1	Transmit Error
1	0	0	0	Timer B
0	1	1	1	General Purpose Interrupt 5 (I5)
0	1	1	0	General Purpose Interrupt 4 (I4)
0	1	0	1	Timer C
0	1	0	0	Timer D
0	0	1	1	General Purpose Interrupt 3 (I3)
0	0	1	0	General Purpose Interrupt 2 (I2)
0	0	0	1	General Purpose Interrupt 1 (I1)
0	0	0	0	General Purpose Interrupt 0 (I0)

a vector number not in the range of allowable user vectors.

7	6	5	4	3	2	1	0
V7	V6	V5	V4	S	*	*	*

Reset:

0	0	0	0	0	U	U	U

V7–V4 — Written by user to set the most-significant four bits of interrupt vector number.

S — In-Service Register Enable

1 = Software end-of-interrupt mode and in-service register bits enabled.

0 = Automatic end-of-interrupt mode and in-service register bits forced low.

2–0 — Not used

DAISY-CHAINING MFPs

As an interrupt controller, the MC68901 MFP will support eight external interrupt sources in addition to its eight internal interrupt sources. When a system requires more than eight external interrupt sources to be placed at the same interrupt level, sources may be added to the prioritized structure by daisy-chaining MFPs. Interrupt sources are prioritized internally within each MFP, and the MFPs are prioritized by their position in the chain. Unique vector numbers are provided for each interrupt source.

The $\overline{IEI}$ and $\overline{IEO}$ signals implement the daisy-chained structure. The $\overline{IEI}$ of the highest priority MFP is tied low and the $\overline{IEO}$ output of this device is tied to the next highest priority MFP's $\overline{IEI}$. The $\overline{IEI}$ and $\overline{IEO}$ signals are daisy-chained in this manner for all the MFPs in the chain with the lowest priority MFP's $\overline{IEO}$ left unconnected. Figure 5 shows a diagram of the interrupt daisy-chain.

Daisy-chaining requires that all parts in the chain have a common $\overline{IACK}$. When the common $\overline{IACK}$ is asserted during an interrupt acknowledge cycle, all parts will prioritize interrupts in parallel. When the $\overline{IEI}$ signal to an MFP is asserted, the part may respond to the $\overline{IACK}$ cycle if it requires interrupt service. Otherwise, the part will assert $\overline{IEO}$ to the next lower priority device. Thus, priority is passed down the chain via $\overline{IEI}$ and $\overline{IEO}$ until a part which has a pending interrupt is reached. The part with the pending interrupt passes a vector number to the processor and does not propagate $\overline{IEO}$.

INTERRUPT CONTROL REGISTERS

MFP interrupt processing is managed by the enable registers A and B, interrupt pending registers A and B, and interrupt mask registers A and B. These registers allow the programmer to enable or disable individual interrupt channels, mask individual interrupt channels, and access pending interrupt status information. In-service registers A and B allow interrupts to be nested. The interrupt control registers are shown in the following paragraphs.

Interrupt Enable Registers (IREA, $07, IERB, $09)

The interrupt channels are individually enabled or disabled by writing a one or a zero, respectively, to the appropriate bit of interrupt enable register A or B (IERA or IERB). The processor may read these registers at any time.

When a channel is enabled, interrupts received on the channel will be recognized by the MFP, and $\overline{IRQ}$ will be asserted to the processor indicating that interrupt service

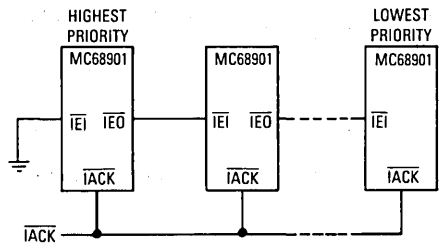


Figure 5. Daisy-Chained Interrupt Structure

is required. On the other hand, a disabled channel is completely inactive; interrupts received on the channel are ignored by the MFP.

Writing a zero to a bit of interrupt enable register A or B will cause the corresponding bit of the interrupt pending register to be cleared. This will terminate all interrupt service requests for the channel and also negate $\overline{IRQ}$ unless interrupts are pending from other sources. Disabling a channel, however, does not affect the corresponding bit in interrupt in-service registers A or B. So, if the MFP is in the software end-of-interrupt mode and an interrupt is in service when a channel is disabled, the in-service bit of that channel will remain set until cleared by software.

7	6	5	4	3	2	1	0
GPIP7	GPIP6	TIMER A	RCV BUFFER FULL	RCV ERROR	XMIT BUFFER EMPTY	XMIT ERROR	TIMER B

7	6	5	4	3	2	1	0
GPIP5	GPIP6	TIMER C	TIMER D	GPIP3	GPIP2	GPIP1	GPIP0

RESET:

0	0	0	0	0	0	0	0

Interrupt Pending Registers (IPRA, $0B, IPRB, $0D)

When an interrupt is received on an enabled channel, the corresponding interrupt pending bit is set in interrupt pending register A or B (IPRA or IPRB). In a vectored interrupt scheme, this bit will be cleared when the processor acknowledges the interrupting channel and the MFP responds with a vector number. In a polled interrupt system, the interrupt pending registers must be read to determine the interrupting channel, and then the interrupt pending bit is cleared by the interrupt handling routine without performing an interrupt acknowledge sequence.

7	6	5	4	3	2	1	0
GPIP7	GPIP6	TIMER A	RCV BUFFER FULL	RCV ERROR	XMIT BUFFER EMPTY	XMIT ERROR	TIMER B

7	6	5	4	3	2	1	0
GPIP5	GPIP6	TIMER C	TIMER D	GPIP3	GPIP2	GPIP1	GPIP0

RESET:

0	0	0	0	0	0	0	0

Interrupt Mask Registers (IMRA, $13, IMRB, $15)

Interrupts are masked for a channel by clearing the appropriate bit in interrupt mask register A or B (IMRA or IMRB). Even though an enabled channel is masked, the channel will recognize subsequent interrupts and set its interrupt pending bit. However, the channel is prevented from requesting interrupt service ($\overline{IRQ}$ to the processor) as long as the mask bit for that channel is cleared. If a channel is requesting interrupt service at the time that its corresponding bit in IMRA or IMRB is cleared, the request will cease, and $\overline{IRQ}$ will be negated unless another channel is requesting interrupt service. Later, when the mask bit is set, any pending interrupt on the channel will be processed according to the channel's assigned priority. IMRA and IMRB may be read at any time. Figure 6 shows a conceptual circuit of an MC68901 interrupt channel.

7	6	5	4	3	2	1	0
GPIP7	GPIP6	TIMER A	RCV BUFFER FULL	RCV ERROR	XMIT BUFFER EMPTY	XMIT ERROR	TIMER B

7	6	5	4	3	2	1	0
GPIP5	GPIP6	TIMER C	TIMER D	GPIP3	GPIP2	GPIP1	GPIP0

RESET:

0	0	0	0	0	0	0	0

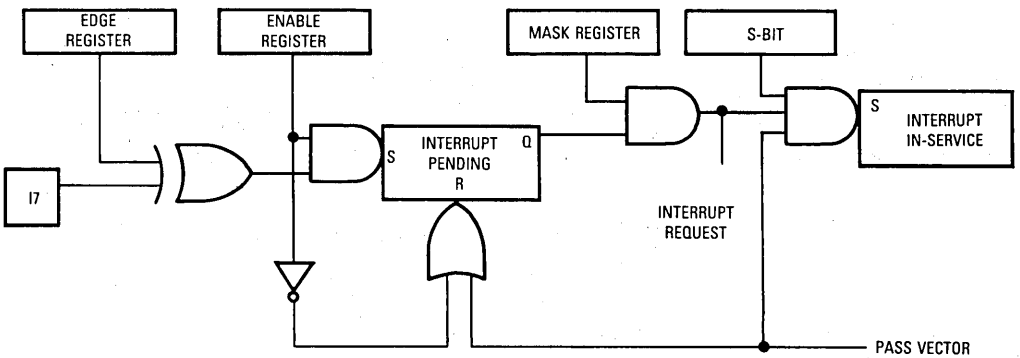


Figure 6. Conceptual Circuits of an Interrupt Channel

Interrupt In-Service Registers (ISRA, $0F, ISRB, $11)

These registers indicate whether interrupt processing is in progress for a certain channel. A bit is set whenever an interrupt vector number is passed for a interrupt channel during an IACK cycle and the S bit of the vector register is a one. The bit is cleared whenever interrupt service is complete for an associated interrupt channel, the S bit of the vector register is cleared, or the processor writes a zero to the bit.

7	6	5	4	3	2	1	0
GPIP7	GPIP6	TIMER A	RCV BUFFER FULL	RCV ERROR	XMIT BUFFER EMPTY	XMIT ERROR	TIMER B

7	6	5	4	3	2	1	0
GPIP5	GPIP6	TIMER C	TIMER D	GPIP3	GPIP2	GPIP1	GPIP0
RESET:							
0	0	0	0	0	0	0	0

NESTING MFP INTERRUPTS

In an M68000 vectored interrupt system, the MFP is assigned to one of seven possible interrupt levels. When an interrupt is received from the MFP, an interrupt acknowledge for that level is initiated. Once an interrupt is recognized at a particular level, interrupts at the same level or below are masked by the processor. As long as the processor's interrupt mask is unchanged, the M68000 interrupt structure will prohibit nesting the interrupts at the same interrupt level. However, additional interrupt requests from the MFP can be recognized before a previous channel's interrupt service routine is finished by lowering the processor's interrupt mask to the next lower interrupt level within the interrupt handler.

When nesting MFP interrupts, it may be desirable to permit interrupts on any MFP channel regardless of its priority, to preempt or delay interrupt processing of an earlier channel's interrupt service request. Or, it may be desirable to only allow subsequent higher priority channel interrupt requests to supercede previously recognized lower priority interrupt requests. The MFP interrupt structure provides the flexibility by offering two end-of-interrupt options for vectored interrupt schemes. Note that the end-of-interrupt modes are not active in a polled interrupt scheme.

GENERAL PURPOSE INPUT/OUTPUT PORT

The general purpose input/output (I/O) port (GPIP) provides eight I/O lines (I0 through I7) that may be operated as either inputs or outputs under software control. In addition, these lines may optionally generate an interrupt on either a positive transition or a negative transition of the input signal. The flexibility of the GPIP allows it to be configured as an 8-bit I/O port or for bit I/O. Since interrupts are enabled on a bit-by-bit basis, a subset of the GPIP could be programmed as handshake lines or the port could be connected to as many as eight external interrupt sources, which would be prioritized by the MFP interrupt controller for the interrupt service.

M6800 INTERRUPT CONTROLLER

The MFP interrupt controller is particularly useful in a system which has many M6800-type devices. Typically, in a vectored M68000 system, M6800 peripherals use the autovector which corresponds to their assigned interrupt level since they can not provide a vector number in response to an interrupt acknowledge cycle. The autovector interrupt handler must then poll all M6800 devices at that interrupt level to determine which device is requesting service. However, by tying the $\overline{IRQ}$ output from an M6800 peripheral to the general purpose I/O port (GPIP) of an MFP, a unique vector number will be provided to the processor during an interrupt acknowledge cycle. This interrupt structure will significantly reduce interrupt latency for M6800 devices and other peripherals which do not support vectored interrupts.

GPIP CONTROL REGISTERS

The GPIP is programmed via three control registers. These registers control the data direction, provide user access to the port, and specify the active edge for each bit of the GPIP which will produce an interrupt. These registers are described in detail in the following paragraphs.

General Purpose I/O Data Register (GPDR, $01)

The general purpose I/O data register is used to input data from or output data to the port. When data is written to the GPDR, those pins which are defined as inputs will remain in the high-impedance state. Pins which are defined as outputs will assume the state (high or low) of their corresponding bit in the data register. When the GPDR is read, data will be passed directly from the bits of the data register for pins which are defined as outputs. Data from pins defined as inputs will come from the input buffers.

7	6	5	4	3	2	1	0
GPIP7	GPIP6	GPIP5	GPIP4	GPIP3	GPIP2	GPIP1	GPIP0
RESET:							
0	0	0	0	0	0	0	0

Active Edge Register (AER, $03)

The active edge register (AER) allows each of the GPIP lines to produce an interrupt on either a one-to-zero or a zero-to-one transition. Writing a zero to the appropriate edge bit of the active edge register will cause the associated input to generate an interrupt on the one-to-zero transition. Writing a one to the edge bit will produce an interrupt on the zero-to-one transition of the corresponding line. When the processor sets a bit, interrupts will be generated on the rising edge of the associated input signal. When the processor clears a bit, interrupts will be generated on the falling edge of the associated input signal.

7	6	5	4	3	2	1	0
GPIP7	GPIP6	GPIP5	GPIP4	GPIP3	GPIP2	GPIP1	GPIP0
RESET:							
0	0	0	0	0	0	0	0

8

NOTE

The inputs to the exclusive-OR of the transition detector are the edge bit and the input buffer. As a result, writing the AER may cause an interrupt-producing transition, depending upon the state of the input. So, the AER should be configured before enabling interrupts via the interrupt enable registers (IERA and IERB). Also, changing the edge bit while interrupts are enabled may cause an interrupt on the corresponding channel.

Data Direction Register (DDR, $05)

The data direction register (DDR) allows the programmer to define I0 through I7 as inputs or outputs by writing the corresponding bit. When a bit of the data direction register is written as a zero, the corresponding interrupt I/O pin will be a high-impedance input. Writing a one to any bit of the data direction register will cause the corresponding pin to be configured as a push-pull output.

7	6	5	4	3	2	1	0
GPIP7	GPIP6	GPIP5	GPIP4	GPIP3	GPIP2	GPIP1	GPIP0
RESET:							
0	0	0	0	0	0	0	0

TIMERS

The MFP contains four 8-bit timers which provide many functions typically required in microprocessor systems. The timers can supply the baud rate clocks for the on-chip serial I/O channel, generate periodic interrupts, measure elapsed time, and count signal transitions. In addition, two timers have waveform generation capability.

All timers are prescaler/counter timers with a common independent clock input (XTAL1 and XTAL2) and are not required to be operated from the system clock. Each timer's output signal toggles when the timer's main counter times out. Additionally, timers A and B have auxiliary control signals which are used in two of the operation modes. An interrupt channel is assigned to each timer, and when the auxiliary control signals are used in the pulse width measurement mode, a separate interrupt channel will respond to transitions on these inputs.

8

OPERATION MODES

Timers A and B are full function timers which, in addition to the delay mode, operate in the pulse width measurements mode and the event count mode. Timers C and D are delay timers only. A brief discussion of each of the timer modes follows.

Delay Mode Operation

All timers may operate in the delay mode. In this mode, the prescaler is always active. The prescaler specifies the number of timer clock cycles which must elapse before a count pulse is applied to the main counter. A count pulse causes the main counter to decrement by one. When the timer has decremented down to 01 (hexadecimal), the next count pulse will cause the main counter to be reloaded from the timer data register and a time out pulse will be produced. This time out pulse is coupled to the timer's interrupt channel and, if the channel is enabled, an interrupt will occur. The time out pulse also causes the timer output pin to toggle. The output will remain in this new state until the next time out pulse occurs.

Pulse Width Measurement Operation

Besides the delay mode, timers A and B may be programmed to operate in the pulse width measurement mode. In this mode, an auxiliary control input is required; timers A and B auxiliary input lines are TAI and TBI. Also, in the pulse width measurement mode, interrupt channels normally associated with I4 and I3 will instead respond to transitions on TAI and TBI, respectively. General purpose lines I3 and I4 may still be used for I/O, but may not be used as interrupt generating inputs. A conceptual circuit of the selection of the interrupt source is shown in Figure 7.

The pulse width measurement mode functions similarly to the delay mode, with the auxiliary control signal acting as an enable to the timer. When the control signal is active, the prescaler and main counter are allowed to operate. When the control signal is negated, the timer is stopped. So, the width of the active pulse on TAI or TBI is measured by the number of timer counts which occur while the timer is allowed to operate.

The active state of the auxiliary input line is defined by the associated interrupt channel's edge bit in the active edge register (AER). GPIP4 of the AER is the edge bit associated with TAI, and GPIP3 is associated with TBI. When the edge bit is a one, the auxiliary input will be active high, enabling the timer while the input signal is at a high level. If the edge bit is zero, the auxiliary input will be active low and the timer will operate while the input signal is at a low level.

The state of the active edge bit also specifies whether a zero-to-one transition or a one-to-zero transition of the auxiliary input pin will produce an interrupt when the interrupt channel is enabled. In normal operation, programming the active edge bit to a one will produce an interrupt on the zero-to-one transition of the associated input signal. Alternately, programming the edge bit to a zero will produce an interrupt on the one-to-zero transition of the input signal. However, in the pulse width measurement mode, the interrupt generated by a transition on TAI or TBI will occur on the opposite transition as that normally defined by the edge bit.

Event Count Mode Operation

In addition to the delay mode and the pulse width measurement mode, timers A and B may be programmed to operate in the event count mode. Like the pulse width measurement mode, the event count mode requires an auxiliary input signal, TAI or TBI. General purpose lines I3 and I4 can be used for I/O or as interrupt producing inputs.

In the event count mode, the prescaler is disabled allowing each active transition on TAI and TBI to produce a count pulse. The count pulse causes the main counter

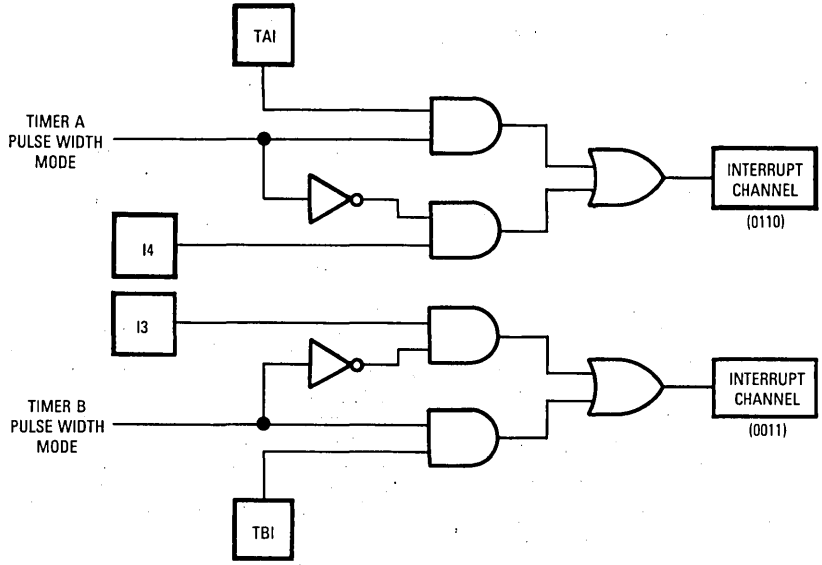


Figure 7. Conceptual Circuit of Interrupt Source Selection

to decrement by one. When the timer counts through 01 (hexadecimal), a time out pulse is generated which will cause the output signal to toggle and may optionally produce an interrupt via the associated timer interrupt channel. The timer's main counter is also reloaded from the timer data register. To count transitions reliably, the input signal may only transition once every four timer clock periods. For this reason, the input signal must have a maximum frequency of one-fourth that of the timer clock.

TIMER REGISTERS

The four timers are programmed via three control registers and four data registers. The following paragraphs describe the different registers.

Timer Data Registers (TxDR) ($1F, $21, $23, $25)

The four timer data registers (TDRs) are designed as Timer A data register (TADR), Timer B (TBDR), Timer C (TCDR), and Timer D (TDDR). Each timer's main counter is an 8-bit binary down counter. The timer data registers contain the value of their respective main counter. This value was captured on the last low-to-high transition of the data strobe pin.

The main counter is initialized by writing to the TDR. If the timer is stopped, data is loaded simultaneously into both the TDR and main counter. If the TDR is written to while the timer is enabled, the value is not loaded into the timer until the timer counts through 01 (hexadecimal). If a write is performed while the timer is counting through 01, then an indeterminate value will be loaded into the timer's main counter.

7	6	5	4	3	2	1	0
D7	D6	D5	D4	D3	D2	D1	D0

Timer Control Registers (TxCR) ($19, $1B, $1D)

Timer control register A (TACR) and timer control register B (TBCR) are associated with timers A and B, respectivey. Timers C and D are programmed using one control register—the timer C and D control register (TCDCR). The bits in the control register select the operation mode, prescaler value, and disable the timers. Both control registers have bits which allow the programmer to reset output lines TA0 and TB0.

TACR

7	6	5	4	3	2	1	0
*	*	*	RESET TA0	AC3	AC2	AC1	AC0

TBCR

7	6	5	4	3	2	1	0
*	*	*	RESET TB0	BC3	BC2	BC1	BC0
RESET: 0	0	0	0	0	0	0	0

*Unused bits read as zero.

Reset TA0/TB0 — Reset Timer A and B Output Lines. TA0 and TB0 may be forced low at any time by writing a one to the reset location in TACR and TBCR. Output is held low during the write operation, and at the end of the bus cycle the output is allowed to

8

toggle in response to a time-out pulse. When resetting TA0 and TB0, the other bits in the TCR must be written with their previous value to avoid altering the operating mode.

AC3-AC0, BC3-BC0 — Select Timer A and B Operation Mode.

When the timer is stopped, counting is inhibited. The contents of the timer's main counter is not affected, although any residual count in the prescaler.

AC3 BC3	AC2 BC2	AC1 BC1	AC0 BC0	Operation Mode
0	0	0	0	Timer Stopped
0	0	0	1	Delay Mode, /4 Prescaler
0	0	1	0	Delay Mode, /10 Prescaler
0	0	1	1	Delay Mode, /16 Prescaler
0	1	0	0	Delay Mode, /50 Prescaler
0	1	0	1	Delay Mode, /64 Prescaler
0	1	1	0	Delay Mode, /100 Prescaler
0	1	1	1	Delay Mode, /200 Prescaler
1	0	0	0	Event Count Mode
1	0	0	1	Pulse Width Mode, /4 Prescaler
1	0	1	0	Pulse Width Mode, /10 Prescaler
1	0	1	1	Pulse Width Mode, /16 Prescaler
1	1	0	0	Pulse Width Mode, /50 Prescaler
1	1	0	1	Pulse Width Mode, /64 Prescaler
1	1	1	0	Pulse Width Mode, /100 Prescaler
1	1	1	1	Pulse Width Mode, /200 Prescaler

TCDCR

7	6	5	4	3	2	1	0
*	CC2	CC1	CC0	*	DC2	DC1	DC0

CC2-CC0, DC3-DC0 — Select Timer C and D Operation Mode.

When the timer is stopped, counting is inhibited. The contents of the timer's main counter is not affected, although any residual count in the prescaler is lost.

CC2 DC2	CC1 DC1	CC0 DC0	Operation Mode
0	0	0	Timer Stopped
0	0	1	Delay Mode, /4 Prescaler
0	1	0	Delay Mode, /10 Prescaler
0	1	1	Delay Mode, /16 Prescaler
1	0	0	Delay Mode, /50 Prescaler
1	0	1	Delay Mode, /64 Prescaler
1	1	0	Delay Mode, /100 Prescaler
1	1	1	Delay Mode, /200 Prescaler

UNIVERSAL SYNCHRONOUS/ASYNCHRONOUS RECEIVER-TRANSMITTER

The universal synchronous asynchronous receiver-transmitter (USART) is a single, full-duplex serial channel with a double-buffered receiver and transmitter. There are separate receive and transmit clocks and, also, separate receive and transmit status and data bytes. The receive and transmit sections are also assigned separate interrupt channels. Each section has two interrupt channels: one for normal conditions and the other for error conditions. All interrupt channels are edge-triggered. Generally, it is the output of a flag bit or bits which is coupled to the interrupt channel. Thus, if an interrupt-producing event occurs while the associated interrupt channel is disabled, no interrupt would be produced, even if the channel was subsequently enabled because a transition did not occur while the channel was enabled. That particular event would have to occur again, generating another edge, before an interrupt would be generated. The interrupt channels may be disabled and instead, a DMA device can be used to transfer the data via the control signals receiver ready ($\overline{RR}$) and transmitter ready ($\overline{TR}$).

CHARACTER PROTOCOLS

The MFP USART supports asynchronous and, with the help of a polynomial generator checker, byte synchronous character formats. These formats are selected independently of the divide-by-one and divide-by-18 clock modes. It is possible to clock data synchronously into the MC68901 but still use start and stop bits. After a start bit is detected, data will be shifted in and a stop bit will be checked to determine proper framing. In this mode, all normal asynchronous format features apply.

When the divide-by-one clock mode is selected, synchronization must be accomplished externally. The receiver will sample serial data on the rising edge of the receiver clock. In the divide-by-18 clock mode, the data is sampled at mid-bit time to increase transient noise rejection. Also, when the divide-by-18 clock mode is selected, the USART resynchronization logic is enabled. This logic increases the channels clock skew tolerance.

Asynchronous Format

Variable character length and start/stop bit configurations are available under software control for asynchronous operation. The user can choose a character length from five to eight bits and a stop bit length of one, one and one-half, or two bits. The user can also select odd, even, or no parity.

In the asynchronous format, start bit detection is always enabled. New data is not shifted into the receive shift register until a zero bit is received. When the divide-by-18 clock mode is selected, the false start bit logic is also active. Any transition must be stable for three positive receive clock edges to be considered valid. For a start bit to be good, a valid zero-to-one transition must not occur for eight positive receiver clock transitions after the initial one-to-zero transition. After a valid start bit has been detected, the data is checked continuously for valid

transitions. When a valid transition is detected, an internal counter is forced to state zero, and no further transition checking is initiated until state four. At state eight, the "previous state" of the transition checking logic is clocked into the receiver. As a result of this resynchronization logic, it is possible to run with asynchronous clocks without start and stop bits if there are sufficient valid transitions in the data streams.

Synchronous Format

When the synchronous character format is selected, the 8-bit synchronous character loaded into the synchronous character register (SCR, $27) is compared to received serial data until a match is found. Once synchronization is established, incoming data is clocked into the receiver. The synchronous word will be continuously transmitted during an underrun condition. All synchronous characters can be optionally stripped from the receive buffer (i.e. taken out of the data stream and thrown away) by clearing the appropriate bit in the receive status register (RSR).

7	6	5	4	3	2	1	0
D7	D6	D5	D4	D3	D2	D1	D0
RESET:							
0	0	0	0	0	0	0	0

The synchronous character should be written after the character length is selected, since unused bits in the synchronous character register are zeroed out. When parity is enabled, synchronous word length is the character length plus one. The MFP will compute and append the parity bit for the synchronous character when a character length of eight is selected. However, if the character length is less than eight, the user must determine the synchronous word parity and write it into the synchronous character register along with the synchronous character. The parity bit must be the most-significant bit. The MFP will then transmit the extra bit in the synchronous word as a parity bit.

USART Control Register (UCR, $29)

This register selects the clock mode and the character format for the receive and transmit sections.

7	6	5	4	3	2	1	0
CLK	CL1	CL0	ST1	ST0	PE	E/O	*
RESET:							
0	0	0	0	0	0	0	U

*Unused bits read as zero.

CLK — Clock Mode
- 1 = Data clocked into and out of receiver and transmitter at one sixteenth the frequency of their respective clocks.
- 0 = Data clocked into and out of receiver and transmitter at the frequency of their respective clocks.

CL1,CL0 — Character Length

These bits specify the length of the character exclusive of start bits, and parity.

CL1	CL0	Character Length
0	0	8 Bits
0	1	7 Bits
1	0	6 Bits
1	1	5 Bits

ST0-ST1 — Start/Stop Bit and Format Control

These bits select the number of start and stop bits and specify the character format.

ST1	ST0	Start Bits	Stop Bits	Format
0	0	0	0	Synchronous
0	1	1	1	Asynchronous
1	0	1	1-1/2	Asynchronous*
1	1	1	2	Asynchronous

*Used with Divide-by-16 mode only.

PE — Parity Enable

Parity is not automatically appended to the synchronous character for character lengths of less than eight bits. Therefore, parity should be written into the SCR along with the synchronous character.
- 1 = Parity checked by receiver and parity calculated and inserted during data transmission.
- 0 = No parity check and no parity bit computed for transmission.

E/O — Even/Odd Parity
- 1 = Even parity is selected.
- 0 = Odd parity is selected.

Bit 0 — Not used

RECEIVER

As data is received on the serial input line (SI), it is clocked into an internal 8-bit shift register until the specified number of data bits have been assembled. The character will then be transferred to the receive buffer, assuming that the last word in the receive buffer has been read. This transfer will set the buffer full bit in the Receiver Status Register (RSR) and produce a buffer full interrupt to the processor, assuming this interrupt has been enabled.

Reading the receive buffer satisfies the buffer full condition and allows a new data word to be transferred to the receive buffer when it is assembled. The receive buffer is accessed by reading the USART data register (UDR). The UDR is simply an 8-bit data register used when transferring data between the MFP and the CPU.

Each time a word is transferred to the receive buffer, its status information is latched into the receiver status register (RSR). The RSR is not updated again until the data word in the receive buffer has been read. When a buffer full condition exists, the RSR should always be read before the receive buffer (UDR) to maintain the correct correspondence between data and flags. Otherwise, it is possible that after reading the UDR and prior to reading the RSR, a new word could be received and transferred to the receive buffer. Its associated flags would be

8

latched into the RSR, writing over the flags for the previous data word. Thus, when the RSR was read to access the status information for the first data word, the flags for the new word would be retrieved.

Receiver Interrupt Channels

The USART receiver section is assigned two interrupt channels. One indicates the buffer full condition while the other channel indicates an error condition. Error conditions include overrun, parity error, frame error, synchronous found, and break. These interrupting conditions correspond to the OE, PE, FE, and F/S or B bits of the receiver status register. These flags will function whether the receiver interrupt channels are enabled or disabled.

While only one interrupt is generated per character received, two dedicated interrupt channels allow separate vector numbers to be assigned for normal and abnormal receiver conditions. When a received word has an error associated with it and the error interrupt channel is enabled, an interrupt will be generated on the error channel only. However, if the error channel is disabled, an interrupt for an error condition will be generated on the buffer full interrupt channel along with interrupts produced by the buffer full condition. The receiver status register must always be read to determine which error condition produced the interrupt.

Receiver Status Register (RSR, $2B)

The RSR contains the receiver buffer full flag, the synchronous strip enable, the various status information associated with the data word in the receive buffer. The RSR is latched each time a data word is transferred to the receive buffer. RSR flags cannot change again until the new data word has been read. However, the M/CIP bit is allowed to change.

7	6	5	4	3	2	1	0
BF	OE	PE	FE	F/S OR B	M/CIP	SS	RE
RESET: 0	0	0	0	0	0	0	0

BF — Buffer Full
- Receiver word is transferred to the receive buffer.
- Receiver buffer is read by accessing the USART data register.

OE — Overrun Error
- Overrun error occurs when a received word is to be transferred to the receive buffer, but the buffer is full. Neither the receiver buffer nor the RSR is overwritten.
- 1 = Receiver buffer full.
- 0 = Read by RSR or MFP reset.

PE — Parity Error
- 1 = Parity error detected on character transfer to receiver buffer.
- 0 = No parity error detected on character transfer to receiver buffer.

FE — Frame Error
- A frame error exists when a non-zero data character is not followed by a stop bit in the asynchronous character format.
- 1 = Frame error detected on character transfer to receiver buffer.
- 0 = No frame error detected on character transfer to receiver buffer.

F/S or B — Found/Search or Break Detect
- The F/S bit is used in the synchronous character format. When set to zero, the USART receiver is placed in the search mode. F/S is cleared when the incoming character does not match the synchronous character.
- 1 = Match found. Character length counter enabled.
- 0 = Incoming data compared to SCR. Character length counter disabled.
- The B bit is used in the asynchronous character format. This flag indicates a break condition which continues until a non-zero data bit is received.
- 1 = Character transferred to the receive buffer is a break condition.
- 0 = Non-zero data bit received and break condition was acknowledged by reading the RSR at least once.

M or CIP — Match/Character in Progress
- The M bit is used in the synchronous character format and indicates a synchronous character has been received.
- 1 = Character transferred to the receive buffer matches the synchronous character.
- 0 = Character transferred to the receive buffer does not match the synchronous character.
- The CIP bit is used in the asynchronous character format and indicates that a character is being assembled.
- 1 = Start bit is detected.
- 0 = Final stop bit has been received.

SS — Synchronous Strip Enable
- 1 = Characters that match the synchronous character will not be loaded into the receiver buffer and no buffer full condition will be produced.
- 0 = Characters that match the synchronous character will be transferred to the receive buffer and a buffer full condition will be produced.

RE — Receiver Enable
- This bit should not be set until the receiver clock is active. When the transmitter is disabled in auto-turnaround mode this bit is set.
- 1 = Receiver operation is enabled.
- 0 = Receiver is disabled.

Special Receive Conditions

Certain receive conditions relating to the overrun error flag and the break detect flag require further explanation. Consider the following examples:

1) A break is received while the receive buffer is full. This does not produce an overrun condition. Only the B flag will be set after the receiver buffer is read.
2) A new word is received, and the receive buffer is full. A break is received before the receive buffer is read.

Both the B and OE flags will be set when the buffer full condition is satisfied.

TRANSMITTER

The transmit buffer is loaded by writing to the USART data register (UDR). The data character will be transferred to an internal 8-bit shift register when the last character in the shift register has been transmitted. This transfer will produce a buffer empty condition. If the transmitter completes the transmission of the character in the shift register before a new character is written to the transmit buffer, an underrun error will occur. In the asynchronous character format, the transmitter will send a mark until the transmit buffer is written. In the synchronous character format, the transmitter will continuously send the synchronous character until the transmit buffer is written.

The transmit buffer can be loaded prior to enabling the transmitter. After the transmitter is enabled, there is a delay before the first bit is output. The serial output line (SO) should be programmed to be high, low, or high impedance (by setting the appropriate bits in the Transmitter Status Register (TSR)) before the transmitter is enabled forcing the output line to the desired state until the first bit of the first character is shifted out. The state of the H and L bits in the TSR determine the state of the first transmitted bit after the transmitter is enabled. If the high impedance mode is selected prior to the transmitter being enabled, the first bit transmitted is indeterminate. Note that the SO line will always be driven high for one bit time prior to the character in the transmit shift register being transmitted when the transmitter is first enabled.

When the transmitter is disabled, any character currently being transmitted will continue to completion. However, any character in the transmit buffer will not be transmitted and will remain in the buffer. Thus, no buffer empty condition will occur. If the buffer is empty when the transmitter is disabled, the buffer empty condition will remain, but no underrun condition will be generated when the character in transmission is completed. If no character is being transmitted when the transmitter is disabled, the transmitter will stop at the next rising edge of the internal shift clock.

In the asynchronous character format, the transmitter can be programmed to send a break. The break will be transmitted once the word currently in the shift register has been sent. If the shift register is empty, the break command will be effective immediately. A transmit error interrupt will be generated at every normal character boundary to aid in timing the break transmission. The contents of the TSR are not affected, however. The break will continue until the break bit is cleared. The underrun error (UE) must be cleared from the TSR. Also, the interrupt pending register must be cleared of pending transmitter errors at the beginning of the break transmission, or no interrupts will be generated at the character boundary time. The break (B) bit cannot be set until the transmitter has been enabled and has had sufficient time (one transmitter clock cycle) to perform internal reset and initialization functions.

Any character in the transmit buffer at the start of a break will be transmitted when the break is terminated, assuming the transmitter is still enabled. If the transmit buffer is empty at the start of a break, it may be written at any time during the break. If the buffer is still empty at the end of the break, an underrun condition will exist.

Disabling the transmitter during a break condition causes the transmitter to cease transmission of the break character at the end of the current character. No end-of-break stop bit will be transmitted. Even if the transmit buffer is empty, no buffer empty condition will occur nor will an underrun condition occur. Also, any word in the transmit buffer will remain.

Transmitter Interrupt Channels

The USART transmit section is assigned two interrupt channels. The normal channel indicates a buffer empty condition, and the error channel indicates an underrun or end condition. These interrupting conditions correspond to the BE, UE, and END flags in the TSR. The flag bits will function whether their associated interrupt channel is enabled or disabled.

Transmitter Status Register (TSR, $2D)

The TSR contains various transmitter error flags and transmitter control bits for selecting auto-turnaround and loopback mode.

7	6	5	4	3	2	1	0
BE	UE	AT	END	B	H	L	TE
RESET:							
0	0	0	0	0	0	0	0

BE — Buffer Empty
1 = Character in the transmit buffer transferred to TSR.
0 = Transmit buffer reloaded by writing to the USR.

U — Underrun Error
One full transmitter clock cycle is required after UE bit is set before it can be cleared. This bit does not require clearing before writing to the UDR.
1 = Character in the TSR was transmitted before a new word was loaded into the transmit buffer.
0 = Transmitter disabled or read performed on TSR.

AT — Auto-Turnaround
When set, the receiver will be enabled automatically after the transmitter has been disabled and the last character being transmitted is complete.

END — End of Transmission
If the transmitter is disabled while a character is being transmitted, this bit is set after transmission is complete. If no character was being transmitted, then this bit is set immediately. Reenabling the transmitter clears this bit.

B — Break
This bit only functions in the asynchronous format. When B is set, BE cannot be set. A break consists of all zeros with no stop bit. This bit cannot be set until transmitter is enabled and internal reset and initialization is complete.
1 = Break transmitted and transmission stops.
0 = Break ceases and normal transmission resumes.

H, L — High and Low
These bits configure the transmitter output (SO) when the transmitter is disabled. Changing these bits after

the transmitter is enabled will alter the output state until END is cleared.

H	L	Output State
0	0	High Impedance
0	1	Low
1	0	High
1	1	Loopback Mode

TE — Transmitter Enable

The serial output will be driven according to H and L bits until transmission begins. A one bit is transmitted before character transmission in the TSR begins.

1 = Transmitter enabled.

0 = Transmitter disabled. UE bit cleared and END bit set.

DMA OPERATION

USART error conditions are valid only for each character boundary. When the USART performs block data transfers by using the DMA handshake lines receiver ready ($\overline{RR}$) and transmitter ready ($\overline{TR}$), errors must be saved and checked at the end of a block. This is accomplished by enabling the error channel for the receiver or transmitter and by masking interrupts for this channel. Once the transfer is complete, interrupt pending register A is read. Any pending receiver or transmitter error indicates an error in the data transfer.

$\overline{RR}$ is asserted when the buffer full bit is set in the RSR unless a parity error or frame error is detected by the receiver. $\overline{TR}$ is asserted when the buffer empty bit is set in the TSR unless a break is currently being transmitted.

ELECTRICAL CHARACTERISTICS

MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Supply Voltage	V_{CC}	−0.3 to 7.0	V
Input Voltage	V_{in}	−0.3 to 7.0	V
Operating Temperature Range	T_A	0 to 70	°C
Storage Temperature Range	T_{stg}	−65 to 150	°C
Power Dissipation	P_D	1.5	W

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

THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Symbole	Value	Rating
Thermal Resistance	θ_{JA}		θ_{JC}		°C/W
Ceramic		40		15*	
Plastic		40		20*	

*Estimated

POWER CONSIDERATIONS

The average chip-junction temperature, T_J, in °C can be obtained from:

$$T_J = T_A + (P_D \cdot \theta_{JA}) \quad (1)$$

where:

T_A = Ambient Temperature, °C

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

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

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

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

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

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

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

Solving equations (1) and (2) for K gives:

$$K = P_D \cdot (T_A + 273°C) + \theta_{JA} \cdot P_D^2 \quad (3)$$

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.

The total thermal resistance of a package (θ_{JA}) can be separated into two components, θ_{JC} and θ_{CA}, representing the barrier to heat flow from the semiconductor junction to the package (case) surface (θ_{JC}) and from the case to the outside ambient (θ_{CA}). These terms are related by the equation:

$$\theta_{JA} = \theta_{JC} + \theta_{CA} \quad (4)$$

θ_{JC} is device related and cannot be influenced by the user. However, θ_{CA} is user dependent and can be minimized by such thermal management techniques as heat sinks, ambient air cooling and thermal convection. Thus, good thermal management on the part of the user can significantly reduce θ_{CA} so that θ_{JA} approximately equals θ_{JC}. Substitution of θ_{JC} for θ_{JA} in equation (1) will result in a lower semiconductor junction temperature.

Values for thermal resistance presented in this document, unless estimated, were derived using the procedure described in Motorola Reliability Report 7843, "Thermal Resistance Measurement Method for MC68XX Microcomponent Devices," and are provided for design purposes only. Thermal measurements are complex and dependent on procedure and setup. User derived values for thermal resistance may differ.

8

DC ELECTRICAL CHARACTERISTICS ($T_A = 0°C$ to $70°C$, $V_{CC} = -5\ V \pm 5\%$, unless otherwise noted)

Characteristic	Symbol	Min	Max	Unit
Input High Voltage	V_{IH}	2.0	$V_{CC}+0.3$	V
Input Low Voltage	V_{IL}	−0.3	0.8	V
Output High Voltage, Except $\overline{DTACK}$ ($I_{OH} = -120\ \mu A$)	V_{OH}	2.4	—	V
Output Low Voltage, Except $\overline{DTACK}$ ($I_{OL} = 2.0$ mA)	V_{OL}	—	0.5	V
Power Supply Current (Outputs Open)	I_{LL}	—	180	mA
Input Leakage Current ($V_{in} = 0$ to V_{CC})	I_{LI}	—	10	μA
Hi-Z Output Leakage Current in Float ($V_{out} = 2.4$ to V_{CC})	I_{LOH}	—	10	μA
Hi-Z Output Leakage Current in Float ($V_{out} = 0.5$ V)	I_{LOL}	—	−10	μA
$\overline{DTACK}$ Output Source Current ($V_{out} = 2.4$ V)	I_{OH}	—	−400	μA
$\overline{DTACK}$ Output Sink Current ($V_{out} = 0.5$ V)	I_{OL}	—	5.3	mA

CAPACITANCE ($T_A = 25°C$, f = 1 MHz, unmeasured pins returned to ground)

Characteristic		Symbol	Min	Max	Unit
Input Capacitance		C_{in}	—	10	pF
Hi-Z Output Capacitance		C_{out}	—	10	pF
Load Capacitance	$\overline{IRQ}$, $\overline{DTACK}$	C	—	100	pF
	All Other Outputs		—	130	

CLOCK TIMING (see Figures 8 and 9)

Characteristic	Symbol	Min	Max	Unit
Frequency of Operation	f	1.0	4.0	MHz
Cycle Time	t_{cyc}	250	1000	ns
Clock Pulse Width	t_{CL}, t_{CH}	110	250	ns
Rise and Fall Times	t_{Cr}, t_{Cf}	—	15	ns

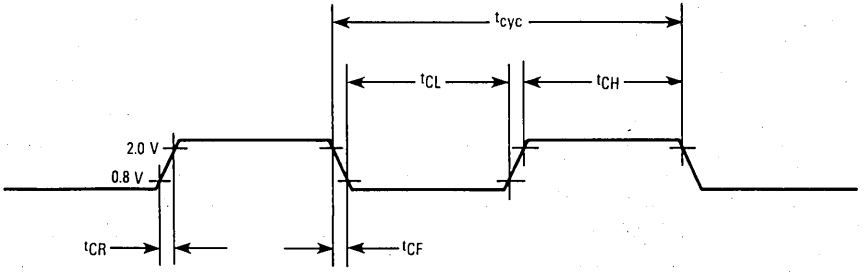
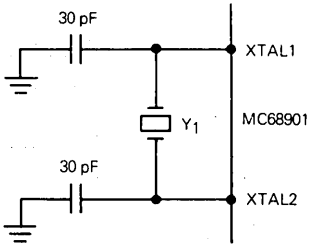


Figure 8. Clock Input Timing Diagram



Crystal Parameters
Parallel resonance fundamen
$R_S \leq 150\ \Omega$ (f = 2.8 – 4.0 MHz
$R_S \leq 300\ \Omega$ (f = 2.0 – 2.7 MHz
$C_L = 18$ pF, $C_M = 0.02$ pF, C_I
f (typical) = 2.4576 MHz

Figure 8. MFP External Oscillator Components

8

AC ELECTRICAL CHARACTERISTICS (V_{CC} = 5.0 Vdc ± 5%, GND = 0°C to 70°C unless otherwise noted; see Figures 10 through 19)

Num	Characteristic	Min	Max	Unit
1[4]	$\overline{CS}$, $\overline{DS}$ Width High	50	—	ns
2	R/$\overline{W}$, RS1-RS5 Valid to Falling $\overline{CS}$ Setup Time	0	—	ns
3	Data Valid Prior to Falling CLK Setup Time (Write Cycle Only)	0	—	ns
4[1]	$\overline{CS}$, $\overline{IACK}$ Valid to Falling CLK Setup Time	50	—	ns
5	CLK Low to $\overline{DTACK}$ Low	—	220	ns
6	$\overline{CS}$ or $\overline{DS}$ or $\overline{IACK}$ High to $\overline{DTACK}$ High	—	60	ns
7	$\overline{CS}$ or $\overline{DS}$ or $\overline{IACK}$ High to $\overline{DTACK}$ High Impedance	—	100	ns
8	$\overline{DTACK}$ Low to Data Invalid Hold Time (Write)	0	—	ns
9	$\overline{CS}$ or $\overline{DS}$ or $\overline{IACK}$ High to Data High Impedance (Read)	—	50	ns
10	$\overline{CS}$ or $\overline{DS}$ High to RS1-RS5, R/$\overline{W}$ Invalid Hold Time	0	—	ns
11[5]	Data Valid from $\overline{CS}$ Low (Read)	—	310	ns
12	Data Valid to $\overline{DTACK}$ Low Setup Time (Read)	50	—	ns
13	$\overline{DTACK}$ Low to $\overline{DS}$ or $\overline{CS}$ or $\overline{IACK}$ High Hold Time	0	—	ns
14	$\overline{IEI}$ Low to Falling CLK Setup Time	50	—	ns
15	$\overline{IEO}$ Valid from CLK Low Delay Time	—	180	ns
16	Data Valid from CLK Low Delay Time	—	300	ns
17	$\overline{IEO}$ Invalid from $\overline{IACK}$ High Delay Time	—	150	ns
18	$\overline{DTACK}$ Low from CLK High Delay Time	—	180	ns
19	$\overline{IEO}$ Valid from $\overline{IEI}$ Low Delay Time	—	100	ns
20	Data Valid from $\overline{IEI}$ Low Delay Time	—	220	ns
21	CLK Cycle Time	250	1000	ns
22	CLK Width Low	110	—	ns
23	CLK Width High	110	—	ns
24[2,4]	$\overline{CS}$, $\overline{IACK}$ Inactive to Rising CLK Setup Time	100	—	ns
25	I/O Minimum Active Pulse Width	100	—	ns
26[3]	$\overline{IACK}$ Width High	2	—	t_{cyc}
27	I/O Data Valid from Rising $\overline{CS}$ or $\overline{DS}$ (Write)	—	450	ns
28	Receiver Ready ($\overline{RR}$) Delay from Rising RC	—	600	ns
29	Transmitter Ready ($\overline{TR}$) Delay from Rising TC	—	600	ns
30	TxO (A or B) Low from Rising Edge of $\overline{CS}$ or $\overline{DS}$ (Reset Time)	—	450	ns
31[3]	Timer Output (TxO) Valid from Falling t_{clk} that Causes Timeout	—	2 t_{clk} + 300	ns
32	Timer Clock (t_{clk}) Low Time	110	—	ns
33	Timer Clock (t_{clk}) High Time	110	—	ns
34	Timer Clock (t_{clk}) Cycle Time	250	1000	ns
35	$\overline{RESET}$ Low Time	2	—	μs
36	Delay to Falling $\overline{IRQ}$ from Ix Active Transition	—	380	ns
37	Transmitter Interrupt Delay from Falling Edge of TC	550	—	ns
38	Receiver Interrupt Delay from Rising Edge of RC (Buffer Full)	800	—	ns
39	Receiver Interrupt Delay from Falling Edge of RC (Error)	800	—	ns

— Continued —

8

AC ELECTRICAL CHARACTERISTICS (Continued)

Num	Characteristic	Min	Max	Unit
40	SI Setup Time from Rising Edge of RC (Divide by 1 Only)	80	—	ns
41	SI Hold Time from Rising Edge of RC (Divide by 1 Only)	350	—	ns
42	SO Data Valid from Falling Edge of TC (Divide by 1 Only)	—	440	ns
43	TC Low Time	500	—	ns
44	TC High Time	500	—	ns
45	TC Cycle Time	1.05	∞	µs
46	RC Low Time	500	—	ns
47	RC High Time	500	—	ns
48	RC Cycle Time	1.05	∞	µs
49[3]	$\overline{CS}$, $\overline{IACK}$, $\overline{DS}$ Width Low	—	80	t_{cyc}
50	SO Data Valid from Falling Edge of TC (Divide by 16 Only)	—	490	ns

NOTES:
1. If the setup time is not met, $\overline{CS}$ will not be recognized until the next falling clock.
2. If this setup time is met (for consecutive cycles), the minimum hold-off time of one clock cycle will be obtained. If not met, the hold-off time will be two clock cycles.
3. t_{cyc} refers to the clock signal applied to the MFP CLK input pin. t_{clk} refers to the timer clock signal, regardless of whether that signal comes from the XTAL1/XTAL2 crystal clock inputs or the TAI or TBI timer inputs
4. $\overline{CS}$ is latched internally, therefore if specifications 1 and 24 are met, then $\overline{CS}$ may be negated before the falling clock and still initiate a bus cycle.
5. Although $\overline{CS}$ and $\overline{DTACK}$ are synchronized with the clock, the data out during a read cycle is asynchronous to the clock, relying only on $\overline{CS}$ for timing.

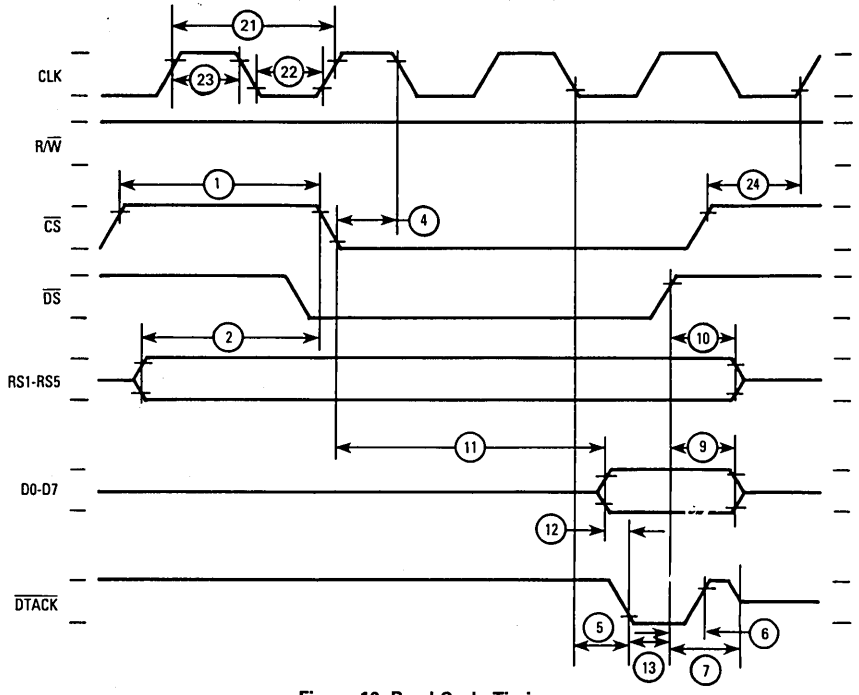


Figure 10. Read Cycle Timing

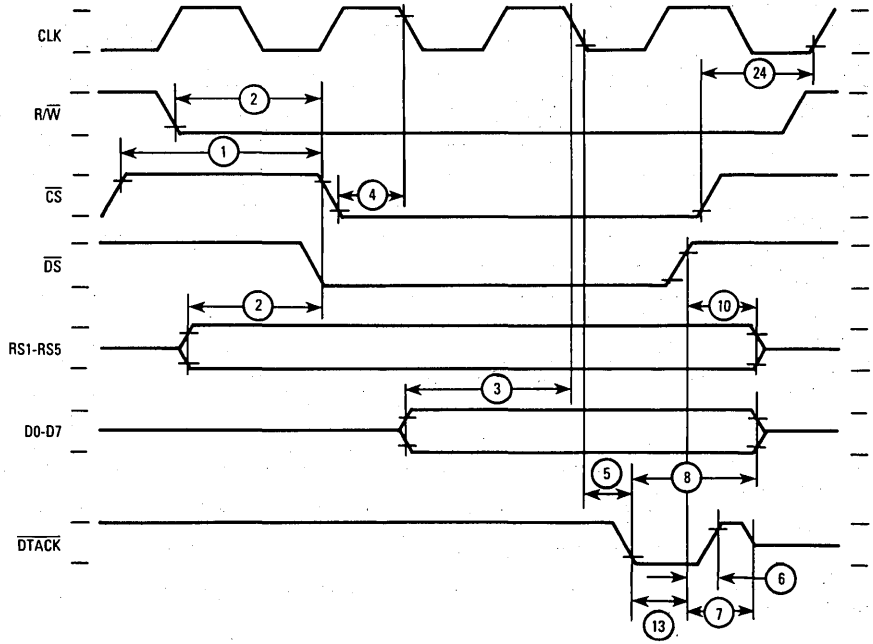
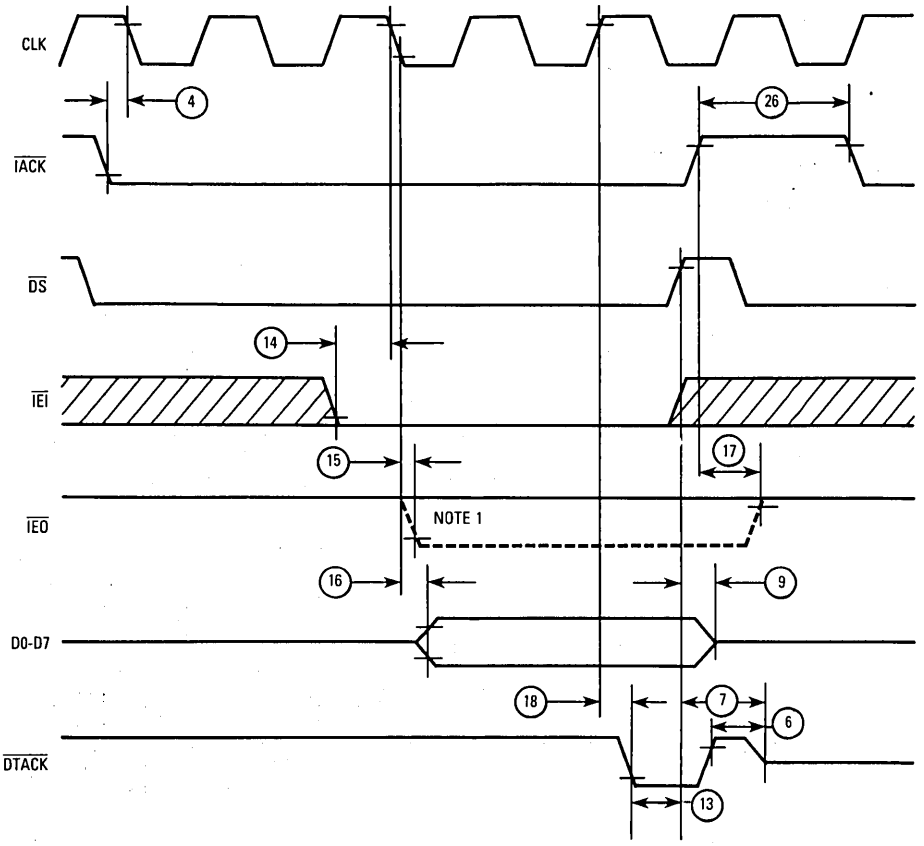


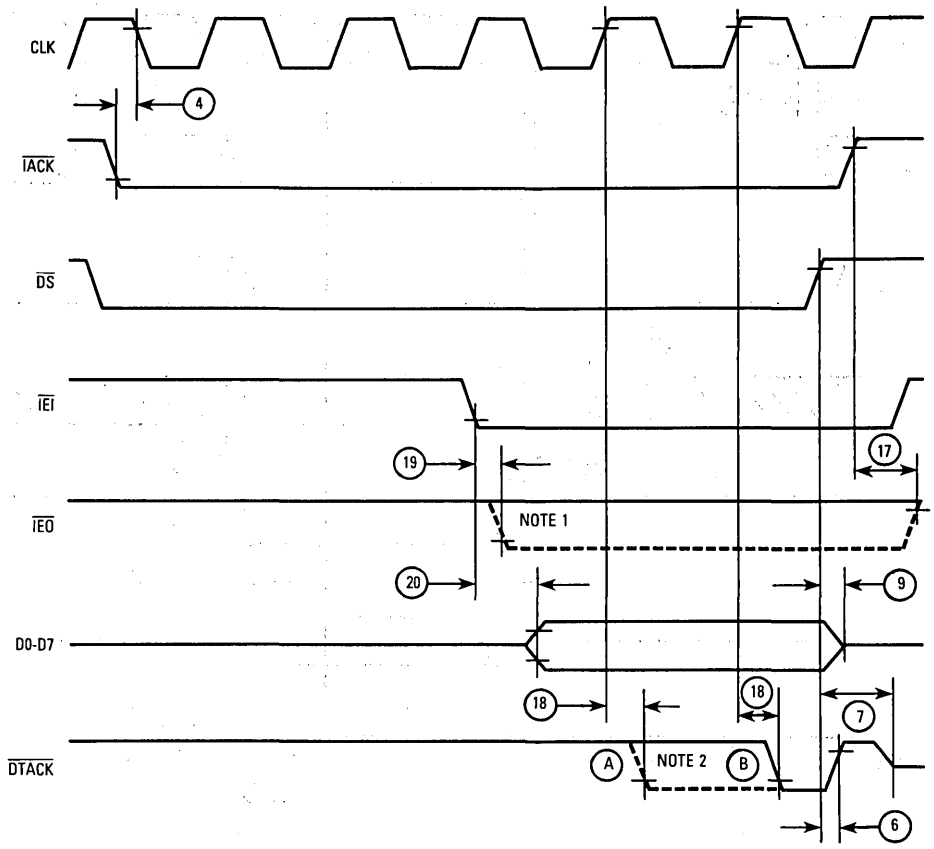
Figure 11. Write Cycle Timing



NOTE 1: $\overline{IEO}$ only goes low if no acknowledgable interrupt is pending. If $\overline{IEO}$ goes low, $\overline{DTACK}$ and the data bus remain in the high-impendance state.

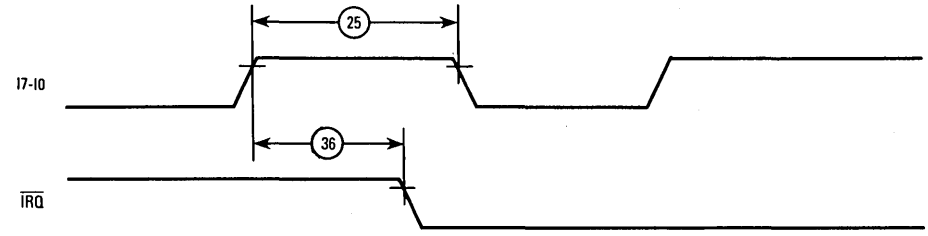
Figure 12. Write Cycle Timing

8



NOTES:
1. $\overline{IEO}$ only goes low if no acknowledge interrupt is pending. If $\overline{IEO}$ goes low, $\overline{DTACK}$ and the data bus remain in the high-impedance state.
2. $\overline{DTACK}$ will go low at (A) if specification number (14) is met. Otherwise, $\overline{DTACK}$ will go low at (B).

Figure 13. Interrupt Acknowledge Cycle ($\overline{IEI}$ High)



NOTE: Active Edge is assumed to be the rising edge.

Figure 14. Interrupt Timing

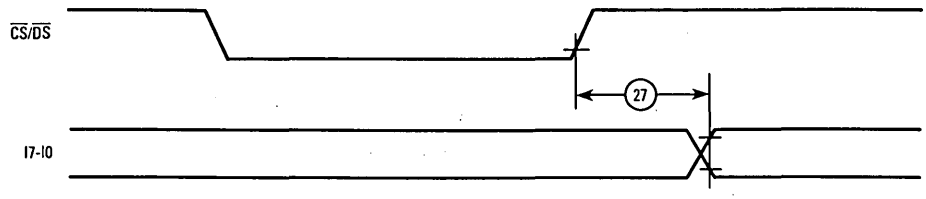


Figure 15. Port Timing

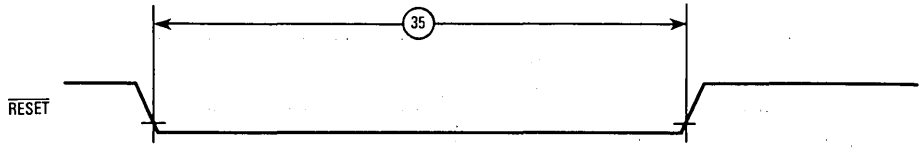


Figure 16. Reset Timing

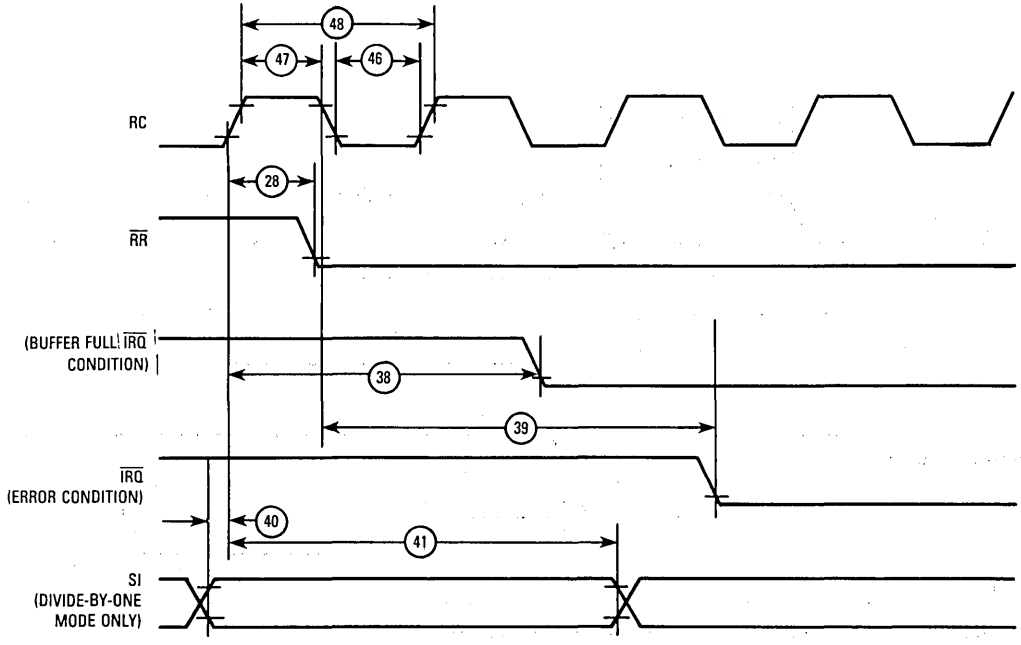


Figure 17. Receiver Timing

8

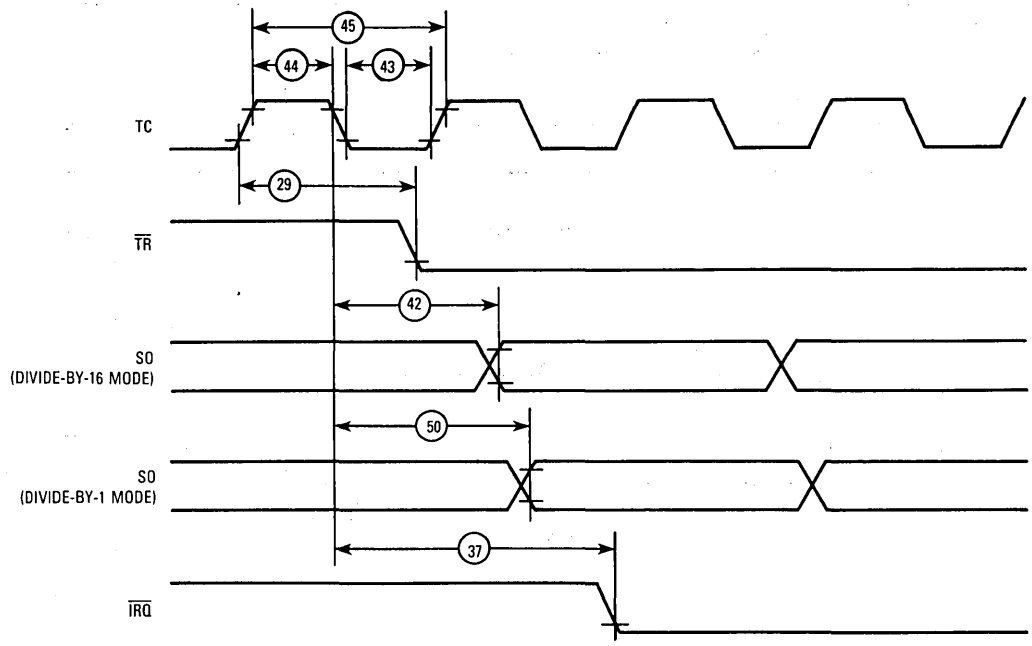
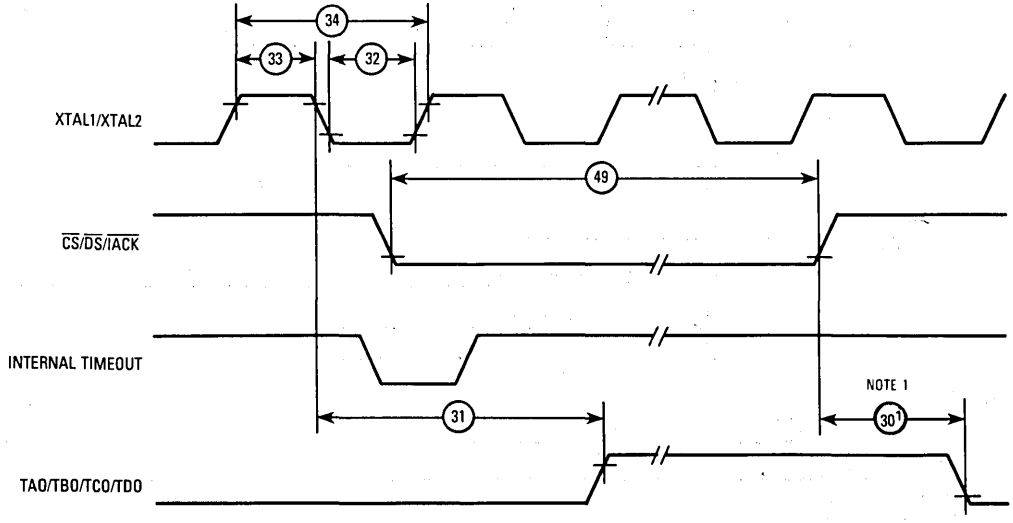


Figure 18. Transmitter Timing



NOTE 1: Specification number 30 applies to timer outputs TAO and TBO only.

Figure 19. Timer Timing

TIMER AC CHARACTERISTICS

Definitions:
Error = Indicated time value — actual time value
$t_{psc} = t_{CLK} \times$ Prescale Value

Internal Timer Mode:

Single Interval Error (Free Running) (See Note 2)	± 100 ns
Cumulative Internal Error	0
Error Between Two Timer Reads	± (t_{psc} − 4 t_{CLK})
Start Timer to Stop Timer Error	2 t_{CLK} + 100 ns to − (t_{psc} + 6 t_{CLK} + 100 ns)
Start Timer to Read Timer Error	0 to − (t_{psc} + 6 t_{CLK} + 400 ns)
Start Timer to Interrupt Request Error (See Note 3)	−2 t_{CLK} to −(4 t_{CLK} + 800 ns)

Pulse Width Measurement Mode:

Measurement Accuracy (See Note 1)	2 t_{CLK} to − (t_{psc} + 4 t_{CLK})
Minimum Pulse Width	4 t_{CLK}

Event Counter Mode:

Minimum Active Time of TAI and TBI	4 t_{CLK}
Minimum Inactive Time of AI and TBI	4 t_{CLK}

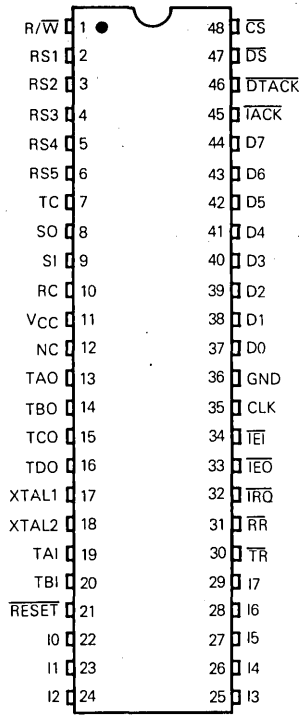
NOTES:
1. Error may be cumulative if repetitively performed.
2. Error with respect to t_{out} or $\overline{IRQ}$ if note 3 is true.
3. Assuming it is possible for the timer to make an interrupt request immediately.

8

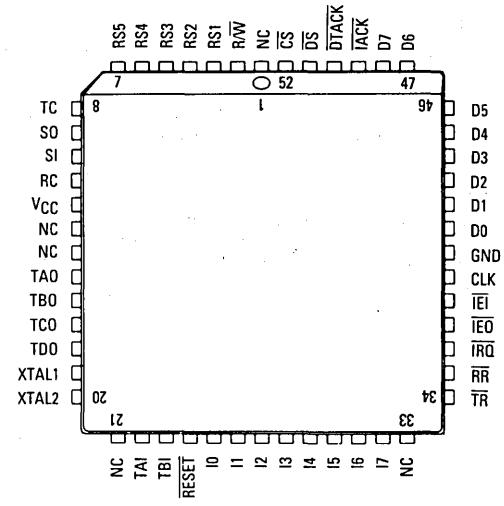
MECHANICAL DATA AND ORDERING INFORMATION

PIN ASSIGNMENTS

48-PIN
Dual-in-Line Package



52-Lead Quad MC68901
(Top View)



8

ORDERING INFORMATION

Package Type	Maximum Clock Frequency	Temperature Range	Order Number
Ceramic L Suffix	4.0 MHz	0°C to 70°C	MC68901L
Plastic P Suffix	4.0 MHz	0°C to 70°C	MC68901P
Quad Pack FN Suffix	4.0 MHz	0°C to 70°C	MC68901FN

MECHANICAL DATA 9

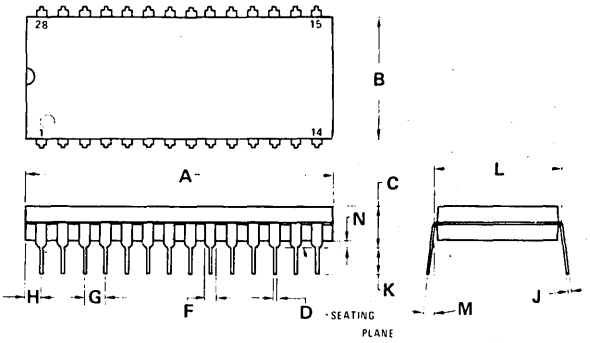
SECTION 9
MECHANICAL DATA

Package availability, for each device, is shown in the individual technical summaries. Dimensions for the packages are given in this section.

9.1 DUAL-IN-LINE PACKAGES

28-Pin

PLASTIC
CASE 710-02

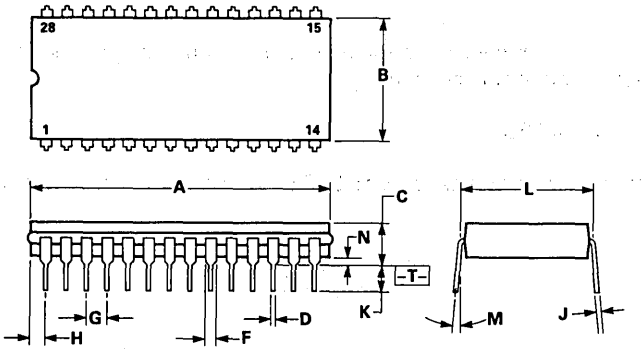


DIM	MILLIMETERS		INCHES	
	MIN	MAX	MIN	MAX
A	36.45	37.21	1.435	1.465
B	13.72	14.22	0.540	0.560
C	3.94	5.08	0.155	0.200
D	0.36	0.56	0.014	0.022
F	1.02	1.52	0.040	0.060
G	2.54 BSC		0.100 BSC	
H	1.65	2.16	0.065	0.085
J	0.20	0.38	0.008	0.015
K	2.92	3.43	0.115	0.135
L	15.24 BSC		0.600 BSC	
M	0^0	15^0	0^0	15^0
N	0.51	1.02	0.020	0.040

NOTES:

1. POSITIONAL TOLERANCE OF LEADS (D), SHALL BE WITHIN 0.25mm(0.010) AT MAXIMUM MATERIAL CONDITION, IN RELATION TO SEATING PLANE AND EACH OTHER.
2. DIMENSION L TO CENTER OF LEADS WHEN FORMED PARALLEL.
3. DIMENSION B DOES NOT INCLUDE MOLD FLASH.

CERAMIC
CASE 733-04

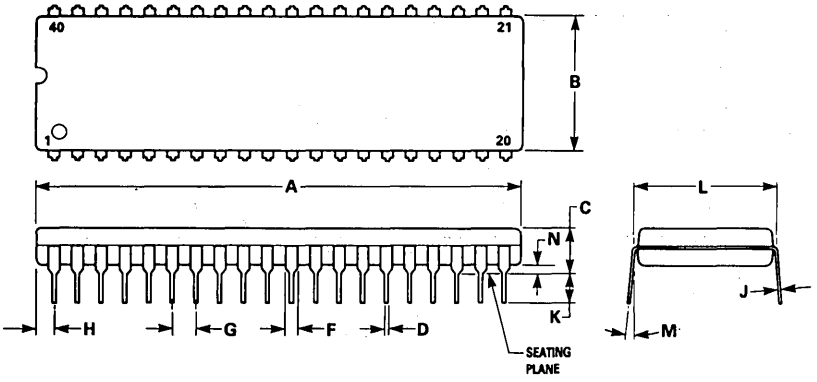


NOTES:
1. DIM [-A-] IS DATUM.
2. POSITIONAL TOL FOR LEADS:
   [⌖ | ϕ 0.25 (0.010) Ⓜ | T | A Ⓜ]
3. [-T-] IS SEATING PLANE.
4. DIM A AND B INCLUDES MENISCUS.
5. DIM -L- TO CENTER OF LEADS WHEN FORMED PARALLEL.
6. DIMENSIONING & TOLERANCING PER Y14.5, 1982.
7. CONTROLLING DIM: INCH.

	MILLIMETERS		INCHES	
DIM	MIN	MAX	MIN	MAX
A	36.45	37.84	1.435	1.490
B	12.70	15.36	0.500	0.605
C	4.06	5.84	0.160	0.230
D	0.38	0.55	0.015	0.022
F	1.27	1.65	0.050	0.065
G	2.54 BSC		0.100 BSC	
J	0.20	0.30	0.008	0.012
K	3.18	4.06	0.125	0.160
L	15.24 BSC		0.600 BSC	
M	0°	15°	0°	15°
N	0.51	1.27	0.020	0.050

9

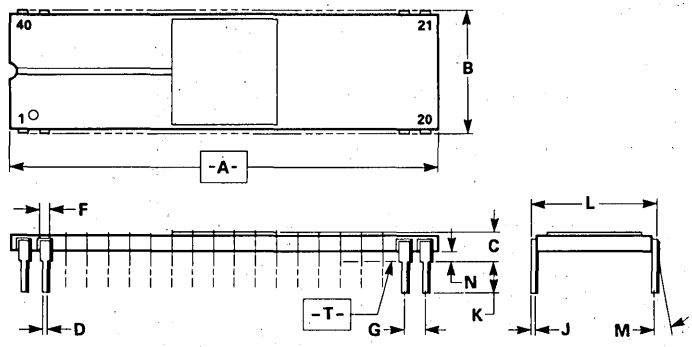
PLASTIC
CASE 711-03



NOTES:
1. POSITIONAL TOLERANCE OF LEADS (D), SHALL BE WITHIN 0.25 mm (0.010) AT MAXIMUM MATERIAL CONDITION, IN RELATION TO SEATING PLANE AND EACH OTHER.
2. DIMENSION L TO CENTER OF LEADS WHEN FORMED PARALLEL.
3. DIMENSION B DOES NOT INCLUDE MOLD FLASH.

	MILLIMETERS		INCHES	
DIM	MIN	MAX	MIN	MAX
A	51.69	52.45	2.035	2.065
B	13.72	14.22	0.540	0.560
C	3.94	5.08	0.155	0.200
D	0.36	0.56	0.014	0.022
F	1.02	1.52	0.040	0.060
G	2.54 BSC		0.100 BSC	
H	1.65	2.16	0.065	0.085
J	0.20	0.38	0.008	0.015
K	2.92	3.43	0.115	0.135
L	15.24 BSC		0.600 BSC	
M	0°	15°	0°	15°
N	0.51	1.02	0.020	0.040

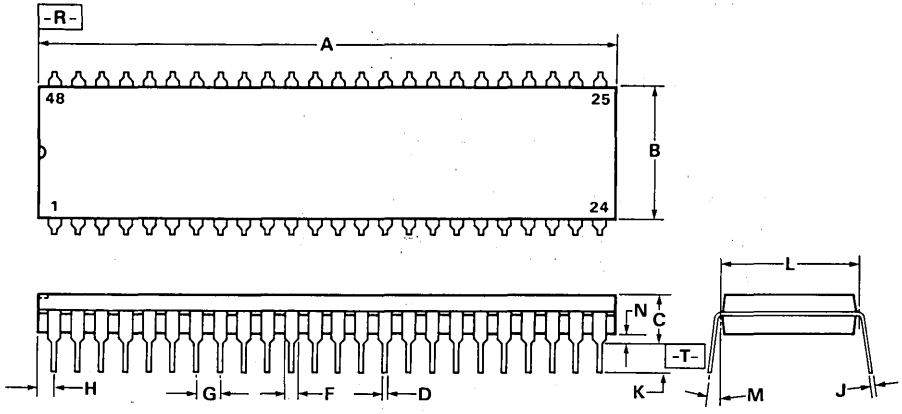
CERAMIC
CASE 715-05



NOTES:
1. DIMENSION -A- IS DATUM.
2. POSITIONAL TOLERANCE FOR LEADS:
   ⌖ 0.25 (0.010) Ⓜ | T | A Ⓜ
3. -T- IS SEATING PLANE.
4. DIMENSION "L" TO CENTER OF LEADS WHEN FORMED PARALLEL.
5. DIMENSIONING AND TOLERANCING PER ANSI Y14.5, 1973.

	MILLIMETERS		INCHES	
DIM	MIN	MAX	MIN	MAX
A	50.29	51.31	1.980	2.020
B	14.63	15.49	0.576	0.610
C	2.79	4.32	0.110	0.170
D	0.38	0.53	0.015	0.021
F	0.76	1.52	0.030	0.060
G	2.54 BSC		0.100 BSC	
J	0.20	0.33	0.008	0.013
K	2.54	4.57	0.100	0.180
L	14.99	15.65	0.590	0.616
M	—	10°	—	10°
N	1.02	1.52	0.040	0.060

PLASTIC
CASE 767-02

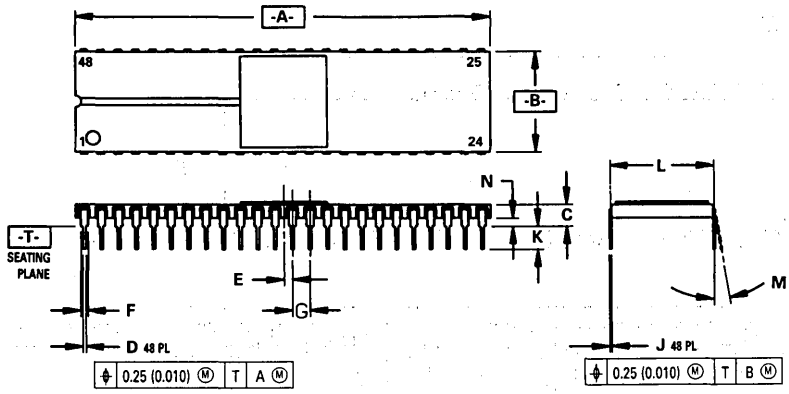


NOTES:

1. -R- IS END OF PACKAGE DATUM PLANE. -T- IS BOTH A DATUM AND SEATING PLANE.
2. POSITIONAL TOLERANCE FOR LEADS 1 AND 48:
   ⌖ | 0.51 (0.020) | T | B Ⓜ | R
   POSITIONAL TOLERANCE FOR LEAD PATTERN:
   ⌖ | 0.25 (0.010) | T | B Ⓜ
3. DIMENSION B DOES NOT INCLUDE MOLD FLASH.
4. DIMENSION L IS TO CENTER OF LEADS WHEN FORMED PARALLEL.
5. DIMENSIONING AND TOLERANCING PER ANSI Y14.5, 1982.
6. CONTROLLING DIMENSION: INCH.

	MILLIMETERS		INCHES	
DIM	MIN	MAX	MIN	MAX
A	61.34	62.10	2.415	2.445
B	13.72	14.22	0.540	0.560
C	3.94	5.08	0.155	0.200
D	0.36	0.55	0.014	0.022
F	1.02	1.52	0.040	0.060
G	2.54 BSC		0.100 BSC	
H	1.79 BSC		0.070 BSC	
J	0.20	0.38	0.008	0.015
K	2.92	3.42	0.115	0.135
L	15.24 BSC		0.600 BSC	
M	0°	15°	0°	15°
N	0.51	1.01	0.020	0.040

CERAMIC
CASE 740-03

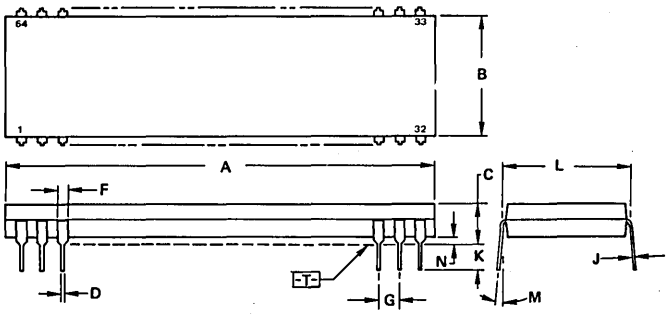


NOTES:
1. DIMENSIONING AND TOLERANCING PER ANSI Y14.5M, 1982.
2. CONTROLLING DIMENSION: INCH.
3. DIM L TO CENTER OF LEAD WHEN FORMED PARALLEL.

	MILLIMETERS		INCHES	
DIM	MIN	MAX	MIN	MAX
A	60.36	61.56	2.376	2.424
B	14.64	15.34	0.576	0.604
C	3.05	4.31	0.120	0.170
D	0.381	0.533	0.015	0.021
E	1.27 BSC		0.050 BSC	
F	0.762	1.397	0.030	0.055
G	2.54 BSC		0.100 BSC	
J	0.204	0.330	0.008	0.013
K	2.54	4.19	0.100	0.165
L	15.24 BSC		0.600 BSC	
M	0°	10°	0°	10°
N	1.016	1.524	0.040	0.060

9

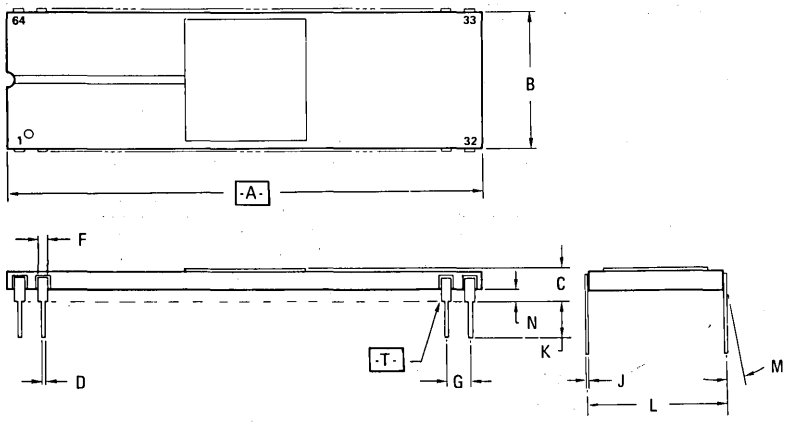
PLASTIC
CASE 754-01



NOTES:
1. DIMENSIONS A AND B ARE DATUMS.
2. -T- IS SEATING PLANE.
3. POSITIONAL TOLERANCE FOR LEADS (DIMENSION D):
   ⌖ Ø 0.25 (0.010) Ⓜ T A Ⓜ B Ⓜ
4. DIMENSION L TO CENTER OF LEADS WHEN FORMED PARALLEL.
5. DIMENSION B DOES NOT INCLUDE MOLD FLASH.
6. DIMENSIONING AND TOLERANCING PER ANSI Y14.5, 1973.

	MILLIMETERS		INCHES	
DIM	MIN	MAX	MIN	MAX
A	81.16	81.91	3.195	3.225
B	20.17	20.57	0.790	0.810
C	4.83	5.84	0.190	0.230
D	0.33	0.53	0.013	0.021
F	1.27	1.77	0.050	0.070
G	2.54 BSC		0.100 BSC	
J	0.20	0.38	0.008	0.015
K	3.05	3.55	0.120	0.140
L	22.86 BSC		0.900 BSC	
M	0°	15°	0°	15°
N	0.51	1.01	0.020	0.040

CERAMIC
CASE 746-01

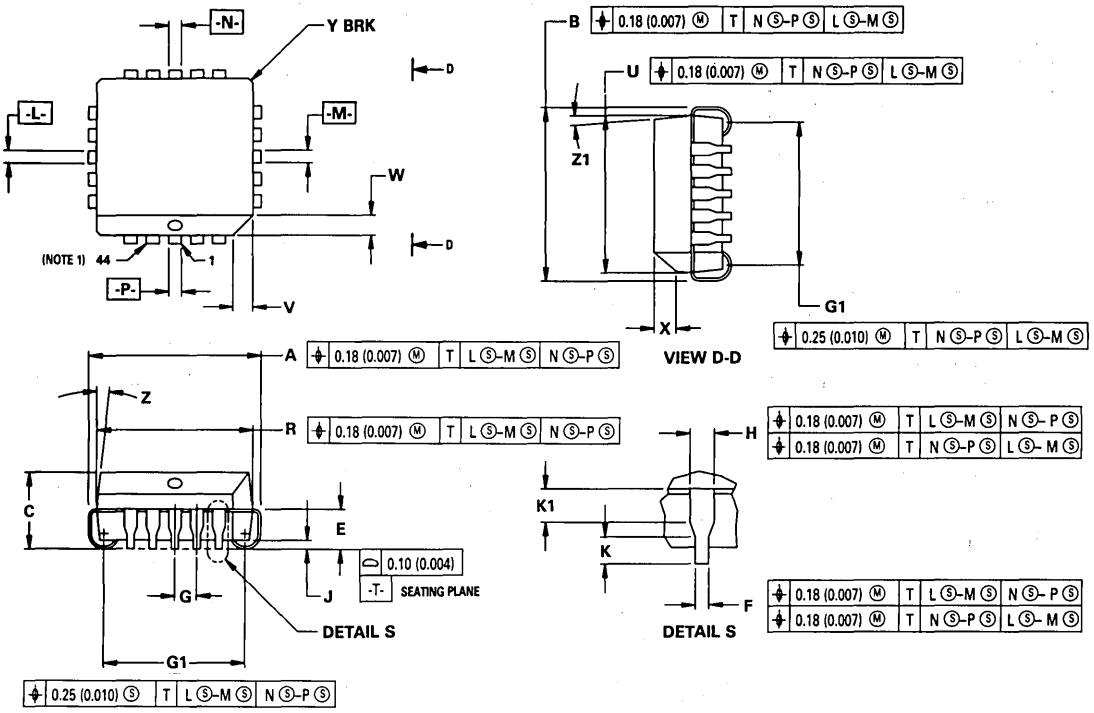


NOTES:
1. DIMENSION -A- IS DATUM.
2. POSITIONAL TOLERANCE FOR LEADS:
   ⌖ 0.25 (0.010) Ⓜ T | A Ⓜ
3. -T- IS SEATING PLANE.
4. DIMENSION "L" TO CENTER OF LEADS WHEN FORMED PARALLEL.
5. DIMENSIONING AND TOLERANCING PER ANSI Y14.5, 1973.

	MILLIMETERS		INCHES	
DIM	MIN	MAX	MIN	MAX
A	80.52	82.04	3.170	3.230
B	22.25	22.96	0.876	0.904
C	3.05	4.32	0.120	0.170
D	0.38	0.53	0.015	0.021
F	0.76	1.40	0.030	0.055
G	2.54 BSC		0.100 BSC	
J	0.20	0.33	0.008	0.013
K	2.54	4.19	0.100	0.165
L	22.61	23.11	0.890	0.910
M	–	10°	–	10°
N	1.02	1.52	0.040	0.060

44-Leads

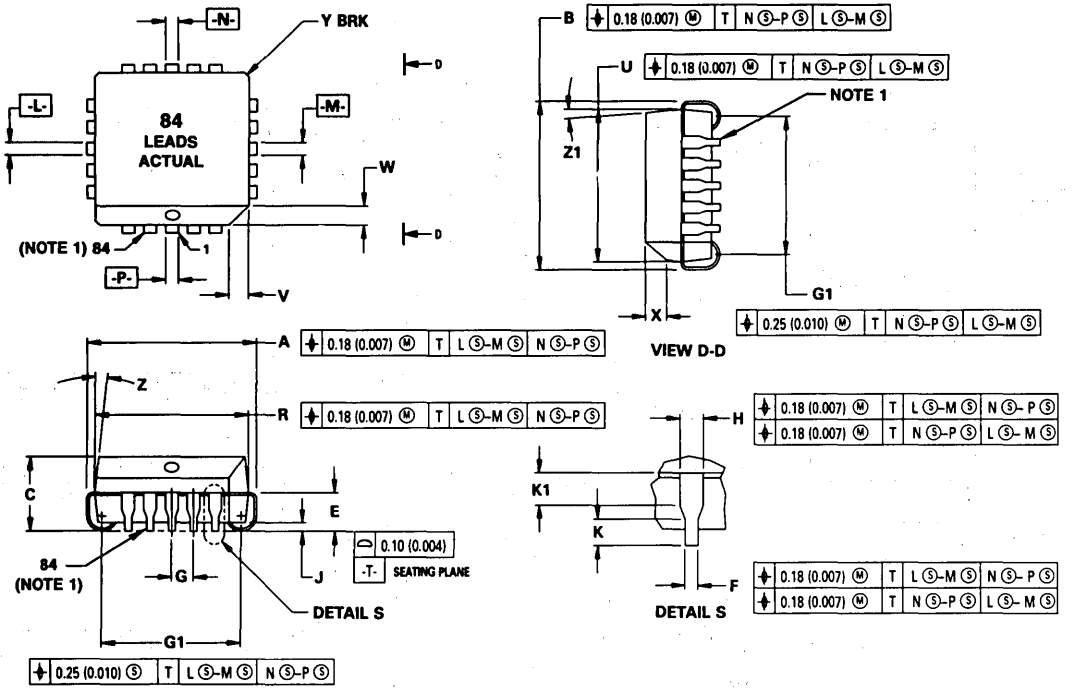
CASE 777-02



DIM	MILLIMETERS		INCHES	
	MIN	MAX	MIN	MAX
A	17.40	17.65	0.685	0.695
B	17.40	17.65	0.685	0.695
C	4.20	4.57	0.165	0.180
E	2.29	2.79	0.090	0.110
F	0.33	0.48	0.013	0.019
G	1.27 BSC		0.050 BSC	
H	0.66	0.81	0.026	0.032
J	0.51	—	0.020	—
K	0.64	—	0.025	—
R	16.51	16.66	0.650	0.656
U	16.51	16.66	0.650	0.656
V	1.07	1.21	0.042	0.048
W	1.07	1.21	0.042	0.048
X	1.07	1.42	0.042	0.056
Y	—	0.50	—	0.020
Z	2°	10°	2°	10°
G1	15.50	16.00	0.610	0.630
K1	1.02	—	0.040	—
Z1	2°	10°	2°	10°

NOTES:
1. DUE TO SPACE LIMITATION, CASE 777-02 SHALL BE REPRESENTED BY A GENERAL (SMALLER) CASE OUTLINE DRAWING RATHER THAN SHOWING ALL 44 LEADS.
2. DATUMS -L-, -M-, -N-, AND -P- DETERMINED WHERE TOP OF LEAD SHOULDER EXIT PLASTIC BODY AT MOLD PARTING LINE.
3. DIM G1, TRUE POSITION TO BE MEASURED AT DATUM -T-, SEATING PLANE.
4. DIM R AND U DO NOT INCLUDE MOLD PROTRUSION. ALLOWABLE MOLD PROTRUSION IS 0.25 (0.010) PER SIDE.
5. DIMENSIONING AND TOLERANCING PER ANSI Y14.5M, 1982.
6. CONTROLLING DIMENSION: INCH.

CASE 780-01



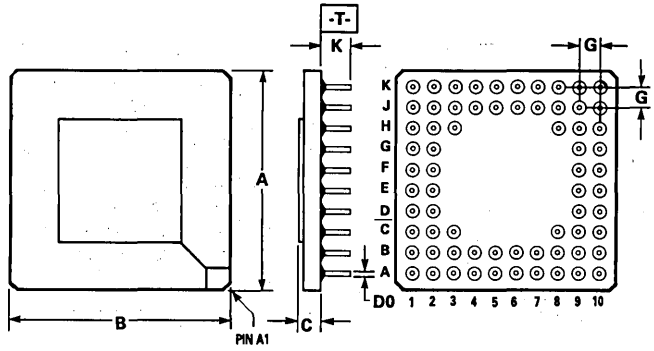
DIM	MILLIMETERS		INCHES	
	MIN	MAX	MIN	MAX
A	30.10	30.35	1.185	1.195
B	30.10	30.35	1.185	1.195
C	4.20	4.57	0.165	0.180
E	2.29	2.79	0.090	0.110
F	0.33	0.48	0.013	0.019
G	1.27 BSC		0.050 BSC	
H	0.66	0.81	0.026	0.032
J	0.51	—	0.020	—
K	0.64	—	0.025	—
R	29.21	29.36	1.150	1.156
U	29.21	29.36	1.150	1.156
V	1.07	1.21	0.042	0.048
W	1.07	1.21	0.042	0.048
X	1.07	1.42	0.042	0.056
Y	—	0.50	—	0.020
Z	2°	10°	2°	10°
G1	28.20	28.70	1.110	1.130
K1	1.02	—	0.040	—
Z1	2°	10°	2°	10°

NOTES:

1. **DUE TO SPACE LIMITATION, CASE 780-01 SHALL BE REPRESENTED BY A GENERAL (SMALLER) CASE OUTLINE DRAWING RATHER THAN SHOWING ALL 84 LEADS.**
2. DATUMS -L-, -M-, -N-, AND -P- DETERMINED WHERE TOP OF LEAD SHOULDER EXIT PLASTIC BODY AT MOLD PARTING LINE.
3. DIM G1, TRUE POSITION TO BE MEASURED AT DATUM -T-, SEATING PLANE.
4. DIM R AND U DO NOT INCLUDE MOLD PROTRUSION. ALLOWABLE MOLD PROTRUSION IS 0.25 (0.010) PER SIDE.
5. DIMENSIONING AND TOLERANCING PER ANSI Y14.5M, 1982.
6. CONTROLLING DIMENSION: INCH.

68-Terminal

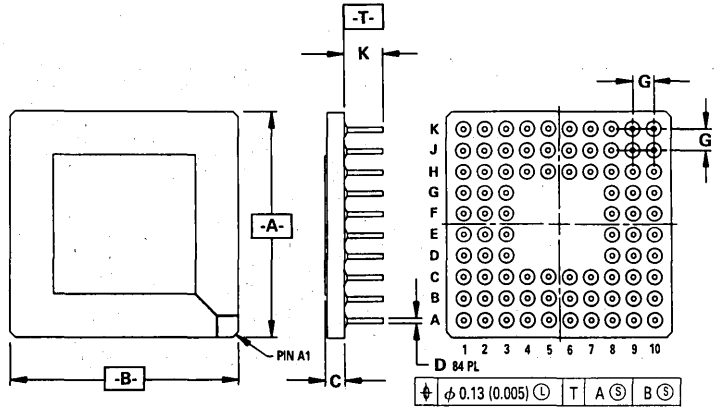
CASE 765A-05



NOTES:
1. DIMENSIONS A AND B ARE DATUMS AND -T- IS DATUM SURFACE.
2. POSITIONAL TOLERANCE FOR LEADS (68 PLACES):
   ⌖ | ϕ 0.13 (0.005) Ⓜ | T | A Ⓢ | B Ⓢ
3. DIMENSIONING AND TOLERANCING PER ANSI Y14.5M, 1982.
4. CONTROLLING DIMENSION: INCH.

	MILLIMETERS		INCHES	
DIM	MIN	MAX	MIN	MAX
A	26.67	27.17	1.050	1.070
B	26.67	27.17	1.050	1.070
C	2.09	2.59	0.082	0.102
D	0.43	0.50	0.017	0.020
G	2.54 BSC		0.100 BSC	
K	4.32	4.82	0.170	0.190

CASE 793-03

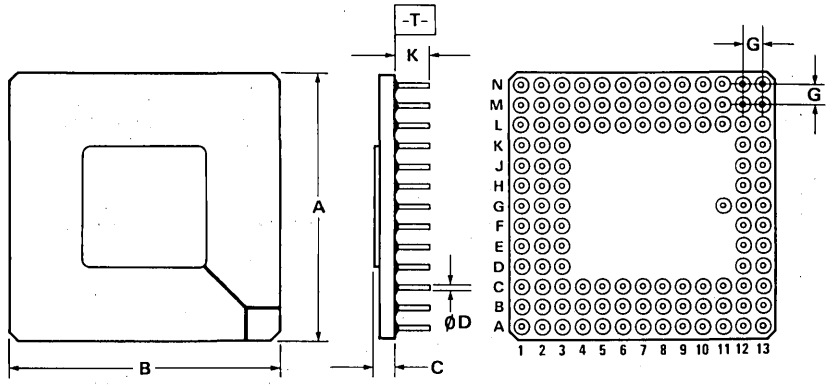


NOTES:
1. DIMENSIONING AND TOLERANCING PER ANSI Y14.5M, 1982.
2. CONTROLLING DIMENSION: INCH.

	MILLIMETERS		INCHES	
DIM	MIN	MAX	MIN	MAX
A	—	27.43	—	1.080
B	—	27.43	—	1.080
C	2.04	2.66	0.080	0.105
D	0.44	0.60	0.017	0.024
G	2.54 BSC		0.100 BSC	
K	4.32	4.82	0.170	0.190

9

CASE 791-01



NOTES:

1. A AND B ARE DATUMS AND T IS A DATUM SURFACE.
2. POSITIONAL TOLERANCE FOR LEADS (114 PLACES).

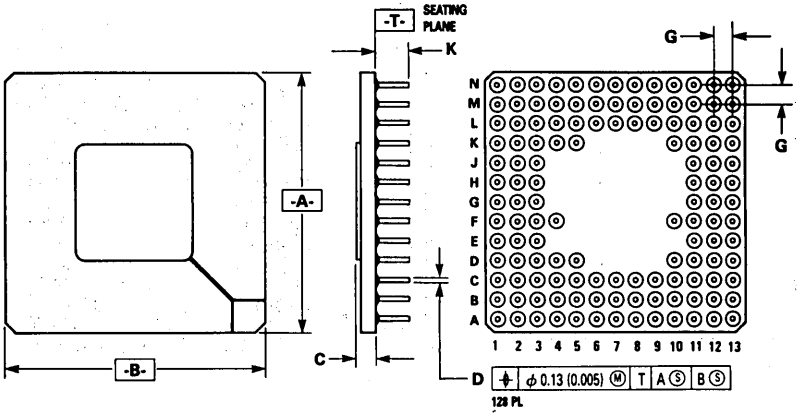
⌖	0.13 (0.005) Ⓜ	T	A Ⓢ	B Ⓢ

3. DIMENSIONING AND TOLERANCING PER Y14.5M, 1982.
4. CONTROLLING DIMENSION: INCH

	MILLIMETERS		INCHES	
DIM	MIN	MAX	MIN	MAX
A	34.04	35.05	1.340	1.380
B	34.04	35.05	1.340	1.380
C	2.54	3.81	0.100	0.150
D	0.43	0.55	0.017	0.022
G	2.54 BSC		0.100 BSC	
K	4.32	4.95	0.170	0.195

9

CASE 789C-01

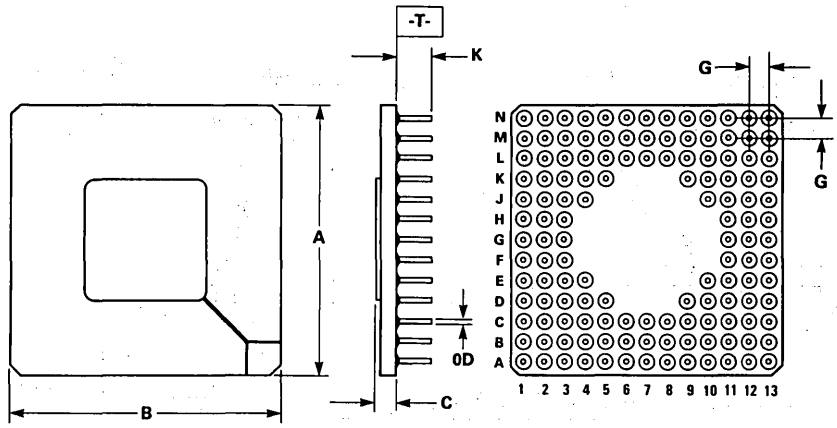


NOTES:
1. A AND B ARE DATUMS AND T IS A DATUM SURFACE.
2. DIMENSIONING AND TOLERANCING PER Y14.5M, 1982.
3. CONTROLLING DIMENSION: INCH.

	MILLIMETERS		INCHES	
DIM	MIN	MAX	MIN	MAX
A	34.04	35.05	1.340	1.380
B	34.04	35.05	1.340	1.380
C	2.54	3.81	0.100	0.150
D	0.44	0.55	0.017	0.022
G	2.54 BSC		0.100 BSC	
K	4.32	4.95	0.170	0.195

9

CASE 789B-01

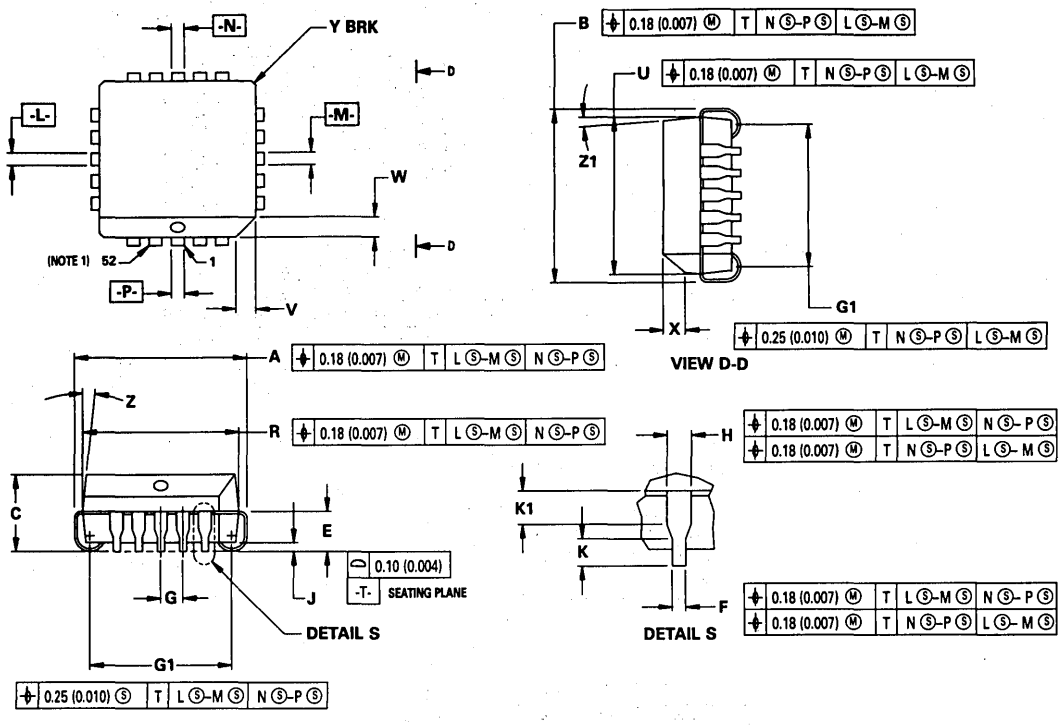


NOTES:
1. A AND B ARE DATUMS AND T IS A DATUM SURFACE.
2. POSITIONAL TOLERANCE FOR LEADS (132 PL).
   | ⌖ | ϕ 0.13 (0.005) Ⓜ | T | A Ⓢ | B Ⓢ |
3. DIMENSIONING AND TOLERANCING PER Y14.5M, 1982.
4. CONTROLLING DIMENSION: INCH.

	MILLIMETERS		INCHES	
DIM	MIN	MAX	MIN	MAX
A	34.04	35.05	1.340	1.380
B	34.04	35.05	1.340	1.380
C	2.54	3.81	0.100	0.150
D	0.43	0.55	0.017	0.022
G	2.54 BSC		0.100 BSC	
K	4.32	4.95	0.170	0.195

52-Lead

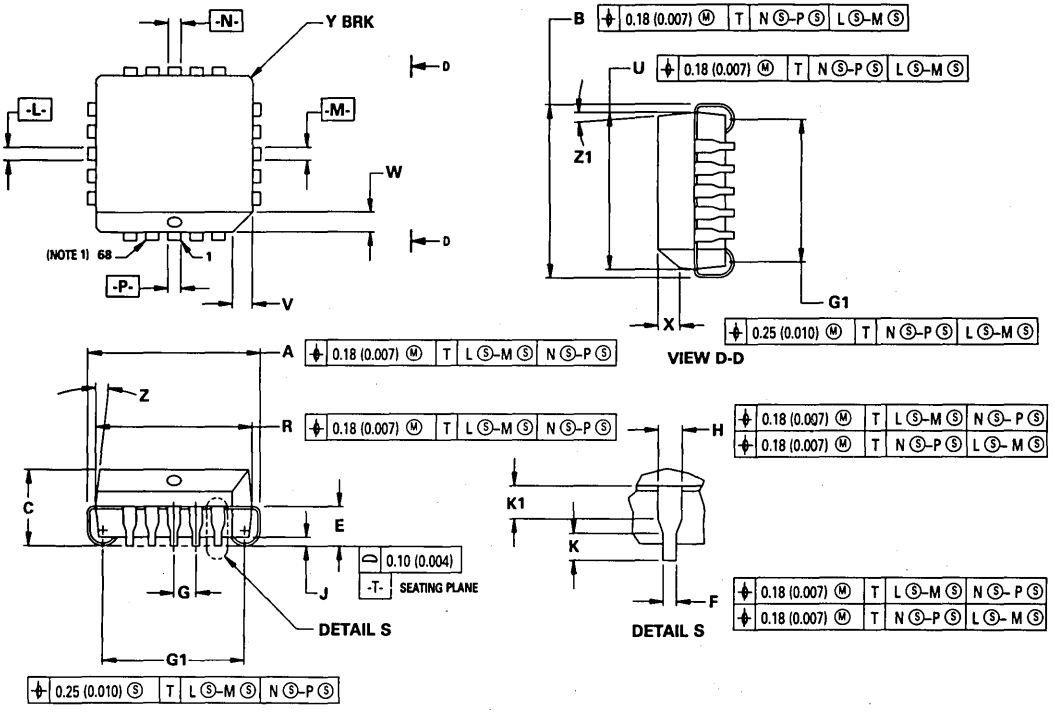
CASE 778-02



DIM	MILLIMETERS		INCHES	
	MIN	MAX	MIN	MAX
A	19.94	20.19	0.785	0.795
B	19.94	20.19	0.785	0.795
C	4.20	4.57	0.165	0.180
E	2.29	2.79	0.090	0.110
F	0.33	0.48	0.013	0.019
G	1.27 BSC		0.050 BSC	
H	0.66	0.81	0.026	0.032
J	0.51	—	0.020	—
K	0.64	—	0.025	—
R	19.05	19.20	0.750	0.756
U	19.05	19.20	0.750	0.756
V	1.07	1.21	0.042	0.048
W	1.07	1.21	0.042	0.048
X	1.07	1.42	0.042	0.056
Y	—	0.50	—	0.020
Z	2°	10°	2°	10°
G1	18.04	18.54	0.710	0.730
K1	1.02	—	0.040	—
Z1	2°	10°	2°	10°

NOTES:
1. DUE TO SPACE LIMITATION, CASE 778-02 SHALL BE REPRESENTED BY A GENERAL (SMALLER) CASE OUTLINE DRAWING RATHER THAN SHOWING ALL 52 LEADS.
2. DATUMS -L-, -M-, -N-, AND -P- DETERMINED WHERE TOP OF LEAD SHOULDER EXIT PLASTIC BODY AT MOLD PARTING LINE.
3. DIM G1, TRUE POSITION TO BE MEASURED AT DATUM -T-, SEATING PLANE.
4. DIM R AND U DO NOT INCLUDE MOLD PROTRUSION. ALLOWABLE MOLD PROTRUSION IS 0.25 (0.010) PER SIDE.
5. DIMENSIONING AND TOLERANCING PER ANSI Y14.5M, 1982.
6. CONTROLLING DIMENSION: INCH.

CASE 779-02



DIM	MILLIMETERS		INCHES	
	MIN	MAX	MIN	MAX
A	25.02	25.27	0.985	0.995
B	25.02	25.27	0.985	0.995
C	4.20	4.57	0.165	0.180
E	2.29	2.79	0.090	0.110
F	0.33	0.48	0.013	0.019
G	1.27 BSC		0.050 BSC	
H	0.66	0.81	0.026	0.032
J	0.51	—	0.020	—
K	0.64	—	0.025	—
R	24.13	24.28	0.950	0.956
U	24.13	24.28	0.950	0.956
V	1.07	1.21	0.042	0.048
W	1.07	1.21	0.042	0.048
X	1.07	1.42	0.042	0.056
Y	—	0.50	—	0.020
Z	2°	10°	2°	10°
G1	23.12	23.62	0.910	0.930
K1	1.02	—	0.040	—
Z1	2°	10°	2°	10°

NOTES:
1. DUE TO SPACE LIMITATION, CASE 779-02 SHALL BE REPRESENTED BY A GENERAL (SMALLER) CASE OUTLINE DRAWING RATHER THAN SHOWING ALL 68 LEADS.
2. DATUMS -L-, -M-, -N-, AND -P- DETERMINED WHERE TOP OF LEAD SHOULDER EXIT PLASTIC BODY AT MOLD PARTING LINE.
3. DIM G1, TRUE POSITION TO BE MEASURED AT DATUM -T-, SEATING PLANE.
4. DIM R AND U DO NOT INCLUDE MOLD PROTRUSION. ALLOWABLE MOLD PROTRUSION IS 0.25 (0.010) PER SIDE.
5. DIMENSIONING AND TOLERANCING PER ANSI Y14.5M, 1982.
6. CONTROLLING DIMENSION: INCH.

TECHNICAL SUPPORT 10

SECTION 10
TECHNICAL SUPPORT

The technical support described in this section includes literature and training offered by Motorola. The section also lists sources of literature and training and the Motorola Sales Offices.

10.1 LITERATURE

The M68000 Family, along with the entire high-end MPU product line, is supported by publicly available technical documentation at three levels: *product previews*, *technical summaries*, and *user's manuals*.

During the early stages of the development of a product, a *product preview* is prepared and published. A *product preview* provides a very high-level generalization of the product and its major features. The purpose of the *product preview* is to document specific information about Motorola's future plans. The audience for this document ranges from the technical and trade press to the engineering community.

The *product preview* is supplied in response to requests for information about a product under development. The information in the *product preview* is purposely limited in order to avoid disclosure of patentable or other proprietary information and to withhold information from competitors. The availability of a *product preview* for a new product usually indicates that further information required to support specific design-ins is available to those willing to sign nondisclosure agreements.

As the development of a new product approaches completion, the *technical summary* replaces the *product preview*. Typically, the *technical summary* provides a block diagram of the component and a brief description of each of the major elements of the block diagram. A summary of the major features of the device is included with sufficient information to allow a system architect, planner, or engineering manager to determine whether the product meets design objectives. The *technical summary* may show a signal list, register models, instruction set summaries, etc. The product description in the *technical summary* may vary in the amount of detail included, as required for a specific product, but typically does not discuss implementation details. For example, signal lists and register descriptions may be shown in general tables but are not addressed in any detail. Detail is limited to provide a good understanding of the important aspects of the product without the exhaustive detail appropriate for the *user's manual*. The audience of the *technical summary* includes design engineers, systems architects, all levels of engineering management, the press, etc.

The *technical summary* is intended to provide enough information about the product to allow decision-makers to determine whether the product meets the gross requirements for their systems. It is also intended to inform the technical and trade press, enabling them to answer commonly asked questions and to write accurate editorial comments. The *technical summary* is not intended to provide enough information for detailed design.

Additionally, the *technical summary* is the vehicle for publishing the initial electrical specifications and updating electrical specifications throughout the life cycle of the product. Prior to publication

of a *user's manual*, the *technical summary* is the means by which Motorola publishes preliminary specifications to facilitate design activities with the product. These preliminary specifications are typically used with the proprietary design specifications (available under nondisclosure) for designs that include the product. As new electrical specifications become available, the specifications in the *technical summary* are updated. Because the *technical summary* can be quickly revised, it is used for fast publication of new information.

The *user's manual* provides the comprehensive, detailed information required to design a system that uses an M68000 component. It includes a complete description of the device and its operation, information about the use of the part, and relevant programming information. The *user's manual* is the authoritative source of product information. The audience for the *user's manual* includes systems architects and designers, software engineers and programmers, and the first tier of engineering management. Component engineering and other support functions also require the information in the *user's manual*. The *user's manual* is expected to provide a detailed description of all user-visible aspects of the component and to answer applications-related questions about each of these aspects.

User's manuals vary in length, but often contain several hundred pages. These manuals are updated infrequently because of their size. Motorola recommends that design engineers always have up-to-date copies of the *user's manual* and the *technical summary* at hand during the design activities.

Advance information data sheets provide the user's manual information for M68000 Family peripheral products. *User's manuals* are not available for these devices. Table 10-1 summarizes the literature provided for the high-end products.

To fill the need for information about third-party software that is available for Motorola microprocessors, Motorola publishes a catalog of software products. The catalog lists the products and the suppliers from whom they can be purchased. Listings in the catalog contain descriptions supplied by the vendors. To obtain a copy, order:

BR506/D The SOURCE

Literature is listed in this manual for your information. Call or write the Motorola Semiconductor Product Sector Literature Distribution Center for latest listings and price information at:

Motorola Semiconductor Products Sector
Literature Distribution Center
P.O. Box 20924
Phoenix, AZ 85036-0924
Phone: (602) 994-6561

Literature is available to the educational community on a special basis, described in the University Support Program brochure. For a copy, or for details and prices, call or write the University Support Organization:

Motorola Technical Operations
University Support Program/HW68
P.O. Box 21007
Phoenix, AZ 85036
Phone: (602) 244-6777 or (602) 244-7130

Table 10-1. MPU Product Literature

Product	Technical Summary	Data Sheet	User's Manual
MC68000	BR245/D	—	M68000UM/AD Rev. 5
MC68HC000	BR275/D	—	M68000UM/AD Rev. 5
MC68008	BR259/D	—	M68000UM/AD Rev. 5
MC68010	BR269/D	—	M68000UM/AD Rev. 5
MC68020	BR243/D	—	MC68020UM/AD
MC68030	BR508/D	—	MC68030UM/AD
MC68851	BR299/D	—	MC68851UM/AD
MC68881	BR265/D	—	MC68881UM/AD
MC68882	BR509/D	—	MC68881UM/AD
MC68153	BR530/D	MC68153/D	—
MC68184	BR286/D	MC68184/D	—
MC68194	BR287/D	MC68194/D	—
MC68230	BR263/D	MC68230/D	—
MC68440	BR260/D	MC68440/D	—
MC68450	BR251/D	MC68450/D	—
MC68452	—	MC68452/D	—
MC68605	BR272/D	MC68605/D	—
MC68606	BR520/D	MC68606/D	—
MC68652	—	MC68652/D	—
MC68661	—	MC68661/D	—
MC68681	BR220/D	MC68681/D	—
MC68824	BR280/D	MC68824/D	—
MC68901	BR587/D	MC68901/D	—

10.2 TECHNICAL TRAINING

Training is available on audio cassettes as well as in-class sessions at Motorola's training centers. Motorola technical training courses are scheduled throughout the world, with training centers in the United States, Canada, South America, Europe, Australia, and the Asia-Pacific area. Motorola also conducts courses at customer's facilities. Training is provided under the direction of Technical Operations whose address is:

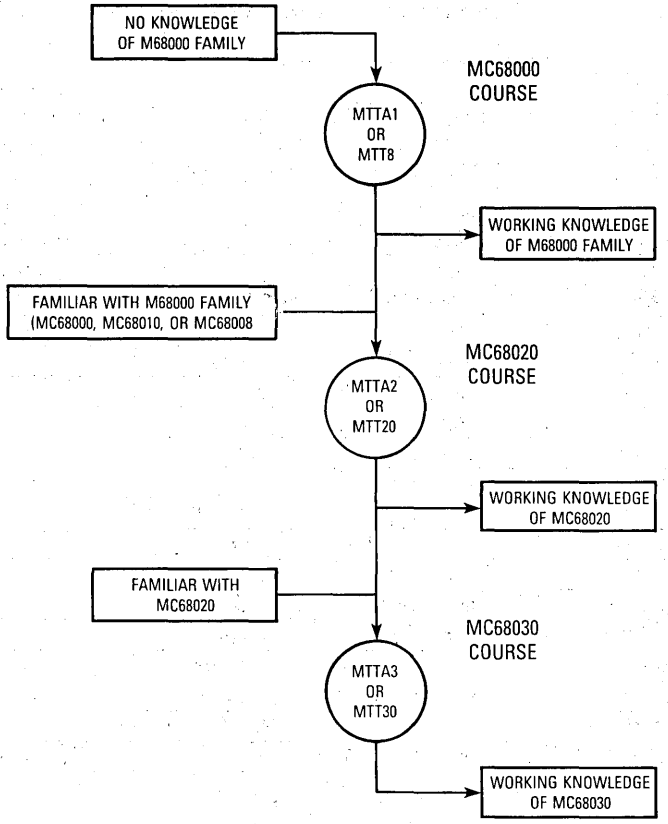
Motorola Semiconductor Products Sector
Technical Operations, HW68
P.O. Box 21007
Phoenix, AZ 85036

Complete information about Motorola's technical training, including descriptions, schedules, and prices of all courses, is listed in the **Technical Training Catalog** (BR348/D).

10

10.2.1 Audio Cassette Courses

Three M68000 Family audio cassette courses follow the evolutionary, upward-compatible growth of the Family. Figure 10-1 shows the flow from the first MPU on the Family, the MC68000, through



NOTE: While the material in an audio or instructor-led course is similar, the audio casette courses are intended for those who do not require the full content and labs available in an instructor-led course.

Figure 10-1. Audio Cassette Courses

the 32-bit MC68020, to the MC68030, the newest member of the Family. The material in the audio cassettes is similar to the content of courses taught at the training centers, but those who require an instructor-led course with lab sessions should choose training center courses rather than audio cassette training.

The courses are:

MTTA1 An Introduction to the MC68000 16-Bit Microprocessor

10

Prerequisites: The student should be familiar with memory concepts, binary numbers, hexadecimal number notation, binary arithmetic, and standard logic operations. Experience with an 8-bit microprocessor, a 16-bit minicomputer, or a mainframe is helpful.

Content: This course covers the major functions of the MC68000: pins and bus operation, programming model, addressing modes, instruction set, and exception processing (including interrupts). Software and hardware examples are included. Upon successful completion, the student will have a working technical knowledge of the MC68000.

MTTA2 An Introduction to the MC68020 32-Bit Microprocessor

Prerequisites: MTTA1 or equivalent knowledge of the MC68000 is required.

Content: This course covers the major features of the MC68020: internal architecture, programming model, pins and bus operation, addressing modes, instruction set, and exception processing. Upon successful completion, the student will have a working technical knowledge of the MC68020.

MTTA3 An Introduction to the MC68030 32-Bit Microprocessor

Prerequisites: MTTA2 or equivalent knowledge of the MC68020 is required.

Content: This course covers the major features of the MC68030: data cache, burst mode, synchronous bus, and the internal memory management unit. Upon successful completion, the student will have a working technical knowledge of the MC68030.

10.2.2 Classroom Courses

The following courses, particularly oriented to the M68000 Family of microprocessors, are offered at Motorola's Technical Training Centers:

MTT7 Understanding Microprocessor Basics — 1 day

Prerequisites: None.

Content: This course is for individuals who do not already possess a working knowledge of microprocessors. It is a nontechnical course to acquaint managers, secretaries, buyers, salespeople, and others with microprocessors. Upon successful completion, the student will know the terminology and understand basic concepts.

MTT8 MC68000 16-/32-Bit Microprocessor — 4 days

Prerequisites: The student should be familiar with memory concepts, binary numbers, hexadecimal number notation, binary arithmetic, and standard logic operations. Experience with an 8-bit microprocessor, a 16-bit minicomputer, or a mainframe is helpful.

Content: This course introduces the student to the MC68000 microprocessor. It describes the general features of the MC68000, such as pin functions, registers, addressing modes, and the instruction set. It also discusses the unique features such as primitive instructions for high-level software, exception handling, and position independent machine code generation. The course includes two lab sessions to provide hardware and software experience. Upon successful completion, the student will be prepared to use and design with the MC68000, the MC68008, or the MC68010.

MTT20 MC68020 32-Bit Microprocessor — 4 days

Prerequisites: The student must have detailed knowledge of the MC68000, MC68008, or MC68010 (course MTT8 or equivalent).

Content: This course introduces the student to the MC68020 32-bit microprocessor. It discusses the general features: pin functions, registers, addressing modes, and the instruction set. It also

describes the unique features, such as primitive instructions for high-level software, exception handling, and modular code generation. The course includes one lab session, which provides experience with the hardware and software. Upon successful completion, the student will be prepared to use and design with the MC68020.

MTT30 MC68030 Enhanced 32-Bit Microprocessor

Prerequisites: The student must have detailed knowledge of the MC68020 (course MTT20 or equivalent).

Content: This course introduces the student to the MC68030 enhanced 32-bit microprocessor. It discusses the major features: data cache, burst mode, synchronous bus, and the internal memory management unit. Upon successful completion, the student will be prepared to use and design with the MC68030.

To enroll in a class at a domestic training center, call (800) 521-6274. For more information about courses, including schedules and prices, call (602) 244-7126. Outside the U.S., call the number of the training center nearest you; numbers of these training centers are listed with their addresses in the first part of this section.

Training is offered at the following training centers in the United States:

Austin
Motorola Training and Education Center
1701 Director's Blvd., Suite 480
Austin, TX 78744
Phone: (512) 444-7725

Boston
Motorola Technical Training Center
300 Unicorn Park Dr.
Woburn, MA 01801
Phone: (617) 932-9700

Chicago
Motorola Technical Training Center
1295 E. Algonquin
Schaumburg, IL 60196
Phone: (312) 576-8600

Dallas
Motorola Technical Training Center
1200 E. Campbell Rd., Suite 108
Richardson, TX 75081
Phone: (214) 699-3900

Phoenix
Motorola Technical Training Center
4902 E. McDowell Rd., Suite 115
Phoenix, AZ 85008
Phone: (602) 244-7126

San Jose
Motorola Technical Training Center
1150 Kifer Road
Sunnyvale, CA 94086
Phone: (408) 749-0510

Toronto
Motorola Technical Training Center
4000 Victoria Park Ave.
North York, Ontario M2H 3P4
Phone: (416) 497-8181

Washington, D.C.
Motorola Technical Training Center
8200 Professional Place, Suite 114
Hyattsville, MD 20785
Phone: (301) 577-2600

Worldwide, Motorola operates training centers in the following countries:

FRANCE
Motorola Semiconducteurs Commercial S.A.
Main Sales Office
2 Rue Auguste Comte BP39
92173 Vanves Cedex, France
Phone: (1) 47 36 03 41

GERMANY
Motorola GMBH
Schulungszentrum
Arabellastr. 17
D-8000 Muenchen 81, Germany
Phone: (89) 9272-142

HONG KONG
Motorola Technical Training
Profit Industrial Building,
7th Floor, Phase 2,
1-15 Kwai Fung Crescent
Kwai Chung, N.T.
Hong Kong
Phone: 0-223111

ISRAEL
Motorola Israel Semiconductors & Systems (SPS) Ltd.
145 Bialik Street
Ramat, Israel
Phone: 972-3-7538288

ITALY
Motorola S.P.A.
Divisione Semiconduttori
Centro Milanofioro-Strade 2-C2
20090 Assago Milano, Italy
Phone: 928 22 01

SPAIN
Motorola Espania S.A.
Albert Alocer, 46 DPDO
28016 Madrid, Spain
Phone: 457 82 04

SWEDEN
Motorola AB, Dalvaegen 2,
S-171 36 Solna, Sweden
Phone: (8) 8300200

ENGLAND
Motorola Technical Training
Fairfax House
69 Buckingham Street
Aylesbury
Buckinghamshire, United Kingdom
Phone: (0296) 393312

10.3 MOTOROLA SALES OFFICES

All Motorola semiconductor products are supported throughout the U. S. and worldwide by local and regional sales offices. In addition to Motorola's professional sales staff, highly experienced field application engineers provide local techical support through these sales offices.

The cities in which Motorola Semiconductor Products Sector sales offices are located and the telephone numbers of those offices are:

DISTRICT OFFICES

ALABAMA, Huntsville .....(205) 830-1050
ARIZONA, Phoenix .....(602) 244-7100
CALIFORNIA, Agoura Hills .....(818) 706-1929
CALIFORNIA, Los Angeles .....(213) 417-8848
CALIFORNIA, Orange .....(714) 634-2844
CALIFORNIA, Sacramento .....(916) 922-7152
CALIFORNIA, San Diego .....(619) 560-4644
CALIFORNIA, San Jose .....(408) 985-0510
COLORADO, Colorado Springs .....(303) 599-7404
COLORADO, Denver .....(303) 337-3434
CONNECTICUT, Wallingford .....(203) 284-0810
FLORIDA, Maitland .....(305) 628-2636
FLORIDA, Pompano Beach/Ft. Lauderdale .....(305) 486-9775
FLORIDA, St. Petersburg .....(813) 576-6030
GEORGIA, Atlanta .....(404) 449-0493
ILLINOIS, Chicago/Schaumburg .....(312) 576-7800
INDIANA, Fort Wayne .....(219) 484-0436
INDIANA, Indianapolis .....(317) 849-7060
INDIANA, Kokomo .....(317) 457-6634
IOWA, Cedar Rapids .....(319) 373-1328
KANSAS, Kansas City/Mission .....(913) 384-3050
MASSACHUSETTS, Marlborough .....(617) 481-8100
MASSACHUSETTS, Woburn .....(617) 932-9700
MICHIGAN, Detroit/Westland .....(313) 261-6200
MINNESOTA, Minneapolis .....(612) 941-6800
MISSOURI, St. Louis .....(314) 872-7681
NEW JERSEY, Hackensack .....(201) 488-1200
NEW YORK, Fairport .....(716) 425-4000
NEW YORK, Hauppage .....(516) 361-7000
NEW YORK, Poughkeepsie/Fishkill .....(914) 473-8102
NORTH CAROLINA, Raleigh .....(919) 876-6025
OHIO, Cleveland .....(216) 349-3100

DISTRICT OFFICES (Continued)

OHIO, Columbus/Worthington .....(614) 846-9460
OHIO, Dayton.....(513) 294-2231
OKLAHOMA, Tulsa.....(918) 664-5227
OREGON, Portland.....(503) 641-3681
PENNSYLVANIA, Philadelphia/Horsham.....(215) 443-9400
TENNESSEE, Knoxville.....(615) 690-5592
TEXAS, Austin .....(512) 452-7673
TEXAS, Houston.....(713) 783-6400
TEXAS, Irving .....(214) 550-0770
TEXAS, Richardson .....(214) 699-3900
VIRGINIA, Charlottesville .....(804) 977-3691
WASHINGTON, Bellevue .....(206) 454-4160
Seattle Access .....(206) 622-9960
WASHINGTON, DC/MARYLAND, Hyattsville.....(301) 577-2600
WISCONSIN, Milwaukee/Wauwatosa.....(414) 792-0122

CANADA

BRITISH COLUMBIA, Burnaby.....(604) 434-9134
MANITOBA, Winnipeg .....(204) 783-3388
ONTARIO, North York.....(416) 497-8181
ONTARIO, Ottawa .....(613) 226-3491
QUEBEC, Montreal .....(514) 731-5483

INTERNATIONAL SALES OFFICES

AUSTRALIA, Melbourne .....(03) 561-3555
AUSTRALIA, Sydney.....(02) 438-1955
AUSTRIA, Vienna.....(0222) 31 65 45
BRAZIL, Sao Paulo .....(011) 572 3553
DENMARK, Soborg .....(02) 92 00 99
FINLAND, Helsinki.....(0) 69 48 455
FRANCE, Grenoble .....(076) 90 22 81
FRANCE, Paris.....(014) 736-01-99
FRANCE, Toulouse .....(061) 41 90 00
GERMANY, Langenhagen/Hannover.....(0511) 78-99-11
GERMANY, Munich .....(089) 92720

INTERNATIONAL SALES OFFICES (Continued)

GERMANY, Nuremberg ..... (0911) 643044
GERMANY, Sindelfingen ..... (07031) 83074
GERMANY, Wiesbaden ..... (06121) 76-1921
HONG KONG, Kwai Chung ..... (0) 223111
ISRAEL, Tel Aviv 03-7538222
ITALY, Milan ..... (2) 82201
ITALY, Rome ..... (06) 831 4746
JAPAN, Osaka ..... (06) 305 1801
JAPAN, Tokyo 03-440-3311
KOREA, Pusan ..... (51) 463-5035
KOREA, Seoul ..... (2) 554-5118-21
MALAYSIA, Penang 04-374514
MEXICO, D.F. ..... (525) 540-5187/(525) 540-5429
NETHERLANDS, Maarssen ..... (030) 439 653
NORWAY, Oslo ..... (02) 19 80 70
PHILIPPINES, Manila ..... (2) 827-59-11
PUERTO RICO, San Juan 809-721-3070
SCOTLAND, East Kilbride ..... (03552) 39 101
SINGAPORE 2945438
SPAIN, Madrid ..... (01) 458 1061
SWEDEN, Solna ..... (08) 83 02 00
SWITZERLAND, Geneva ..... (022) 991 111
SWITZERLAND, Zurich ..... (01) 730 40 74
TAIWAN, Taipei ..... (02) 752 8944
UNITED KINGDOM, Aylesbury ..... (296) 395252

DEVELOPMENT SYSTEMS 11

SECTION 11
DEVELOPMENT SYSTEMS

Motorola's total development system solutions include in-circuit emulators, host computer systems, and control stations. The host computer system integrates the tool set into a complete development system for hardware and software development.

The Motorola tool set includes:

1. **In-Circuit-Emulators**, for all the M68000 Family of microprocessors: 16-bit MC68000, MC68HC000, MC68008, and MC68010; and 32-bit MC68020 and MC68030.
2. **Hardware Control Station**, a controller that services each emulator and establishes the linkage to the host computer and user terminal.
3. **System Performance Analyzer or Bus State Monitor**, which monitors all the microprocessor activity and traces each selected bus cycle of the target microprocessor.
4. **Source Level Debug**, used with a host computer system to provide unprecedented debugging capability at the source code level.
5. **Cross Support Software**, consisting of the assemblers, disassemblers, linkers, "C" compilers, and debuggers used for software development on different host computers.
6. **Development System Host Computer**, a multi-user, multi-tasking computer system running UNIX™ System V, that hosts the hardware control station and cross support software.

Motorola's approach to development systems is to be "host independent." The development systems equipment uses the EIA RS-232-C asynchronous serial interface to connect the control station and its emulator to the host computer system, as shown in Figure 11-1. The control station

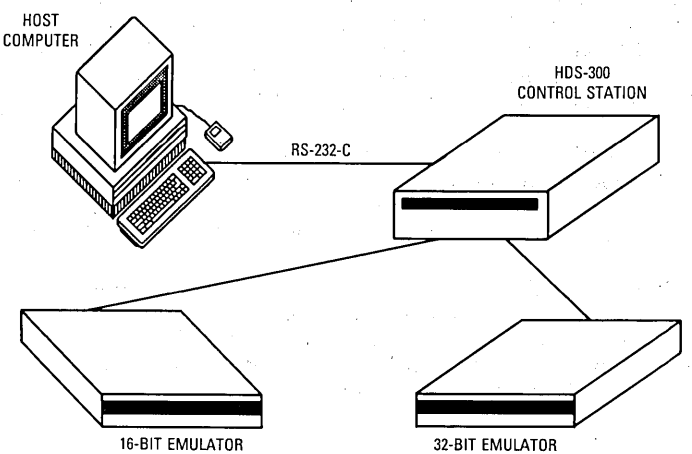


Figure 11.1 The Total Development System Solution

UNIX is a registered trademark of AT&T Bell Laboratories

11

and emulator look like an ASCII asynchronous terminal to the host computer. For standalone operation, an ASCII asynchronous terminal can be connected to the control station in place of the host computer. With the host-independent approach, the user is free to use, as the host, any computer system that can execute basic software development tools, like an assembler, and download code to the emulator/control station via the asynchronous link.

The centerpiece of Motorola development systems is the HDS-300™ hardware control station. The HDS-300 connects the emulator to the host computer to provide a universal debug capability for all Motorola 16- and 32-bit microprocessors. The system performance analyzer plugs into the HDS-300 to monitor 32-bit microprocessor activity by tracing the bus. The source level debug software executes on the host computer and works with the HDS-300 and emulator to give program visibility in the target system at the source code level.

11.1 HOST SYSTEMS

Microprocessor development is typically done with the aid of a host computer that is used to develop software for the target microprocessor system. Software developed on the host computer is downloaded through the HDS-300 via an asynchronous terminal interface on the host computer connected to the HDS-300 host port. The HDS-300 looks like an ASCII terminal to the host computer. Future enhancements will allow high-speed download via a Centronics interface.

Motorola's cross development software has been ported to several host computer systems. The microcomputer designer can use either Motorola's M68DVLP, a VAX™ system, a Macintosh™ computer, or a SUN-3™ workstation as the host.

11.1.1 M68DVLP Host Computer System

The Motorola M68DVLP host computer system is a multi-user, multi-tasking development system that supports cross software development for Motorola's M68000 Family of microprocessors and serves as host for the HDS-300 and emulator. The M68DVLP runs the SYSTEM V/68™ Operating System, the first AT&T-validated version of UNIX System V, and is based on the 32-bit MC68020 VME system built and serviced by Motorola. This system includes a 70-megabyte Winchester disk, a 5.25 inch 655K byte floppy, a QIC-02 streaming tape, eight serial ports and 2 megabytes of memory. The system includes cross development software consisting of 16- and 32-bit assemblers, linkers, C compilers, and source level debug (SLD) packages.

Motorola offers the M68DVLP host development system for software development and hosted debug with the HDS-300 control station and emulator. Using the software control facilities of the SYSTEM V/68 and Motorola's assemblers and compilers for cross software development, M68DVLP support for a team of designers contributes to a well-coordinated development project.

The HDS-300 control station, with an appropriate emulator, can be used standalone or in the hosted configuration to support microprocessor application studies without requiring a target system. Using the emulator's memory, test code can be downloaded from a host computer and benchmarks can be run. In this way, benchmarks that do not require interfacing to a target I/O device can be run and measured in the emulator.

HDS-300 is a trademark of Motorola Inc.
VAX is a trademark of Digital Equipment Corporation
Macintosh is a trademark of Apple Computer Inc.
SUN-3 is a trademark of Sun Microsystems, Inc.
SYSTEM V/68 is a trademark of Motorola Inc.

Motorola's cross development software has been ported for other popular computer systems as described previously.

11.1.2 VAX Host System

Digital Equipment Corporation's VAX system is a multi-user, multi-tasking minicomputer that can serve as the host computer system. VMS™, release 4.6 or later, is required.

11.1.3 Macintosh Host System

The Apple Macintosh is a single-user desktop system that runs the Apple Operating System. The cross software runs on the Macintosh Plus, SE, and Macintosh II models with a minimum of 1 megabytes of memory and a hard disk (required to contain the volume of cross software). Apple Operating System, release 5.0 or later, is required.

11.1.4 SUN-3 Host System

Another alternative host computer system is the SUN-3 workstation. The SUN-3 is a single-user multi-tasking networking workstation.

11.2 HDS-300 CONTROL STATION

The HDS-300 is the universal control station supporting all Motorola emulators for the M68000 Family: 16-bit MC68000, MC68HC000, MC68008, and MC68010; and 32-bit MC68020 and MC68030. An HDS-300 consists of the control station and station software, which are connected to an emulator via a four foot cable set. This configuration can operate standalone by simply attaching an ASCII terminal to the asynchronous serial port of the HDS-300.

The powerful HDS-300 control station greatly simplifies the development task. There is no need to memorize operating procedures, since a powerful HELP command in the HDS-300 can guide inexperienced users through the most complex function sequences. An in-depth tutorial can be called up instantly to provide an "on-line manual" for any function requested. Menus guide the user through the options of an operation. The HDS-300 supports user-defined macros for various operations. For example, a macro can save blocks of code or record complex repetitive tasks for recall later with a single command. The HDS-300 provides easy-to-use "window" displays that are user controlled, and show the status of a variety of activities simultaneously. The HDS-300 is a universal control station for all the M68000 Family emulators. Using BNC connectors mounted on the HDS-300, up to eight control stations, each with a different emulator, can be connected for a single coordinated multi-processor debug session.

Because the HDS-300 control station and emulator replicates all functions of the target system microprocessor in its application environment, it can operate in a non-invasive mode supporting the traditional mechanisms for hardware and software debugging. This mode includes the fundamental capabilities of starting and stopping code execution in order to exercise the entire target system, of examining and altering processor registers and system memory, and of stepping through code by executing one instruction at a time.

VMS is a trademark of Digital Equipment Corporation

The design features of the HDS-300 are:

1. Supports all Motorola M68000 Family emulators
2. Macro utility, for user-defined macro sequences of commands
3. On-line HELP facility, to provide a quick reference, always available on the screen
4. Two asynchronous RS-232-C serial ports, one for host connection and one for local terminal connection.
5. Built-in Centronics-compatible parallel printer port for hard copy of screen displays.
6. 5.25 inch floppy disk drive supports user-written macros, terminal configuration definition, storage of downloaded code, and powerful multi-level HELP files.
7. External synchronization input/output, for operation with multiple control stations in multi-processor development projects.
8. Power-up self-test, with more extensive diagnostics available on floppy disk.

11.2.1 Real-Time Bus Analysis

In-circuit emulation is supported by the bus state monitor (BSM), for the 16-bit microprocessors, and by the system performance analyzer (SPA) for the MC68020. The BSM and SPA work much like a standalone logic analyzer, but are integrated with the emulators and capture bus cycle data via the emulator's in-circuit probe. Both the BSM and SPA can capture all bus cycles in a trace buffer or selectively capture bus cycles as specified by the user. The user-specified qualifiers decide which bus cycles are captured. Qualifiers may be combined for more complex triggers, which may be sequenced for very sophisticated debugging of system software.

11.2.1.1 BUS STATE MONITOR. The BSM is integrated into the HDS-300 and is used with the emulators for the MC68000, MC68HC000, MC68008, and MC68010 microprocessors. The capabilities of the BSM provide:

1. Capture of microprocessor bus cycles
2. Selection of specific bus cycles by user-selected qualifiers
3. Display of the bus cycles captured in the trace buffer
4. Reverse assembly of the microprocessor instructions
5. Performance analysis of the most/least heavily executed program code

The BSM offers two bus cycle recording modes. In one mode, all bus cycle data is recorded continuously from initiation of emulated execution until execution is stopped manually or by a breakpoint. In the other mode, the BSM is triggered by a pre-defined event and continues for a specified number of occurrences of that event or for a specified number of bus cycles after being triggered.

A trigger event is a collective logical state of the emulated microprocessor's data, address, control, and other lines used during emulation for starting or stopping the acquisition of data representing bus cycle activity. Hence, an event is defined by specifying the logical level of particular data and address lines to be used for triggering. An additional cable is provided so eight additional lines may be connected to points of interest in the target system. The logical states of these points as well as an external signal can be included in the trace and in the trigger event. The BSM provides an external trigger event signal for use by external equipment. The BSM can also trigger the emulator to halt emulation, which provides an additional hardware breakpoint.

The trace buffer can record 1024 selected bus cycles. Display of the buffer may be modified to show disassembled instructions, hexadecimal dump, or a combination. Filter patterns may be used to search through the buffer data.

Histograms are used to display relative activity for address ranges with up to 16 user-selected sub-divisions. The BSM captures data in the trace buffer and optionally can halt the emulator each time the BSM records the selected bus cycle. Alternatively, the BSM is allowed to continue to process the recorded bus cycles non-invasively as the test program continues execution.

11.2.1.2 SYSTEM PERFORMANCE ANALYZER. The SPA is an option of the HDS-300 used with the MC68020 emulator. The capabilities of the SPA are similar to the BSM with enhanced definition of the bus event used as a trigger and enhanced recording of bus cycle data.

For event definition, 112 address, data, and control signals, and eight user-option input signals are available. An additional 32 bits in the trace buffer are used to time tag the collected data. Address, data, and control signals of the MC68020 or MC68030 are used to define simple events. These inputs, in combination with an external trigger, a timer value, and up to three hardware breakpoints, are used to define complex events. Simple and complex events can be combined for even more precise event definition to more accurately analyze the target activity. The SPA also allows dynamic control over the emulator's hardware breakpoints and definition of breakpoint events.

The SPA trace buffer is 144 bits wide and 4096 events deep. The 4096 events can be divided into as many as 16 separate blocks. For each block, a unique event can be defined to trigger data capture, and any detected occurrence of the event from the first to the 256th can be selected as the recording trigger. Finally, events can be enabled dynamically under program control, including the control of the first three hardware breakpoints.

The SPA can be used with the MC68030, but the synchronous bus and burst mode can sequence faster than the SPA. In such cases, the SPA can miss some bus cycles of the MC68030.

11.2.2 User Interface

The HDS-300 design employs "fill in the blanks" screen layouts and provides several helpful utilities to simplify the use of its many capabilities. The various capabilities of related HDS-300 commands are grouped according to major function into "operating environments" each of which displays a unique screen used to enter commands. The screens are organized as templates that plainly indicate what information is required and where it is to be entered. A scroll region is used by HDS-300 to display register values, bus analysis histograms, or other results of command execution. The scroll region is also used by the HELP utility.

Several capabilities extremely useful for hardware development are provided by the macro utility. The macro utility offers a full screen text editing capability as a means of creating macros from sequences of commands, a macro library buffer in memory, and support of floppy files for creating and invoking macros. Macro capabilities are often used for automatic initialization and setup at powerup and for the recording and re-execution of a sequence of commands. The macro editor is used for creating named macros and saving them in the macro library.

11.2.3 HDS-300 as a Test Tool

While the power of a debug tool as sophisticated as the HDS-300 is obviously well applied in the microprocessor design activity, it likewise applies effectively in the production test and service/ repair activities. Here, the floppy drive is very important to hold production final tests, more narrow system/board diagnostics, and all software downloaded to floppy for use in the target

test phases. Also, the Macro utility is invaluable to sequence the diagnostic tests and interact with the diagnostician. Macros can:

1. Start and stop test execution
2. Accept commands and directives from the diagnostician
3. Execute commands and directives and display requested information
4. Call and execute other macros when a breakpoint is encountered
5. Copy all data sent to the screen on a printer for a paper trail
6. Restart test code
7. Display understandable, self-explanatory error messages as well as provide extensive help information for the diagnostician

11.3 IN-CIRCUIT EMULATION

In-circuit emulation allows the microprocessor chip to be lifted from the target system and the emulator to be plugged directly into the same socket, which provides the proper electrical connections to duplicate the normal microprocessor functions in the target system. With real-time in-circuit emulation, hardware and software debug is supported by the standard emulator techniques including breakpoints, single-cycle instruction execution, tracing bus cycles, and substitution by emulator memory and certain emulator control signals for those nonfunctioning elements of the target system.

Emulation memory is used as a substitute for the target memory. The HDS-300 remaps emulation memory to exactly reflect the target memory map. In support of ROM applications, emulation memory provides a means to detect unwanted writes to target memory as well as writes to blocks of emulation memory that are write protected.

The HDS-300 provides 32K bytes of emulation memory for the 16-bit emulators. This is expandable to 64K, 128K, or 256K bytes. The MC68020 and MC68030 emulators carry emulation memory within the emulator enclosure; memory configurations are 64K minimum, with 256K- and 1M-byte configurations available.

11.3.1 16-Bit Emulators: MC68000, MC68HC000, MC68008, MC68010

The general features of the 16-bit emulators include:

1. Real-time emulation to 12.5 MHz
2. Probes available for dual-in-line, PGA, and PLCC package types
3. Zero wait states to 10 MHz; one wait state at 12.5 MHz
   a. When operating in emulation memory
   b. Target memory may meet zero wait states at 12.5 MHz
4. Sixteen software breakpoints can be assigned to any four blocks of 4K bytes each anywhere in memory
5. 32K bytes of emulation memory is standard
   a. Expandable to 68K, 128K, or 256K bytes
   b. Mapped as substitute memory in 4K-byte blocks anywhere in 16M-byte address space
6. Built-in bus state monitor with analysis
   a. 63-bit wide trace buffer is 1024 entries deep
   b. Works like a "sliding window"
   c. Tracing controlled by user-defined qualifiers, which can also control the emulator and be used with an external trigger
   d. Analysis highlights code bottlenecks via histograms and instruction disassembly is supported

11.3.2 32-Bit Emulator: MC68020

The general features of the MC68020 emulator include:

1. Real-time emulation to 25 MHz
2. Zero wait states to 20 MHz; one wait state maximum at 25 MHz when using emulation memory; target memory may meet zero wait states at 25 MHz
3. 64K standard emulation memory, with 256K and 1M bytes available
   a. First 256K is high-speed memory with zero wait states to 20 MHz (synchronous and asynchronous) and one wait state above 20 MHz (asynchronous only)
   b. Additional emulation memory has 1 wait state
   c. Selectable, with 8-/16-/32-bit port sizes, write protection, up to seven wait states
4. Three hardware breakpoints and 64 software breakpoints
5. Optional system performance analysis
   a. 144-bit wide trace buffer is 4000 entries deep
   b. Works like a "sliding window"
   c. Tracing controlled by user-defined qualifiers, which can also control the emulator and be used with an external trigger
   d. Instruction disassembly supported

The following table lists the part numbers for MC68020 emulator configurations by speed and memory size.

Maximum Speed	64K Bytes	256K Bytes	1M Bytes
16 MHz	M68020HM3C-1	M68020HM3C-2	M68020HM3C-3
25 MHz	M68020HM3C-4	M68020HM3C-5	M68020HM3C-6

11.3.3 32-Bit Emulator: MC68030

The general features of the MC68030 emulator include:

1. Real-time emulation to 25 MHz
2. Emulation memory with zero wait states to 20 MHz (synchronous and asynchronous); target memory access may see one wait state maximum (ynchronous and asynchronous)
3. 64K standard emulation memory, with 256K, and 1M bytes available
   a. First 256K is high-speed memory with zero wait state to 20 MHz (synchronous and asynchronous) and one wait state above 20 MHz (asynchronous only)
   b. Additional emulation memory has one wait state
   c. Selectable, with 8-/16-/32-bit port sizes, write-rrotection, up to seven wait states
4. Three hardware breakpoints and 64 software breakpoints.
5. Power-up self test, with sophisticated diagnostic set included on floppy disk

The following table lists the part numbers for MC68030 emulator configurations by speed and memory size.

Maximum Speed	64K Bytes	256K Bytes	1M Bytes
25 MHz	M68030HM3C-4	M68030HM3C-5	M68030HM3C-6

Details of MC68030 emulator operation are listed under the following five headings.

Target Memory Access:

1. User application can make use of full 4G-byte address space
2. Maximum of one wait cycle required when accessing target memory (both asynchronous and synchronous)
3. Target (and emulation) memory can be displayed, modified, dumped, loaded, filled, searched, and tested
4. Emulator supports all address spaces
5. Emulator supports dynamic bus sizing (byte, word, and long)
6. Emulator supports three data widths (byte, word, and long) independent of bus width

Emulation Memory Access:

1. First 256K bytes can be used for synchronous emulation memory and supports burst mode with zero wait states up to 20 MHz
2. Emulation memory can be mapped anywhere in the 4G-byte address space
3. Emulation memory can be mapped in 4K-byte blocks
4. All emulation memory must be in the same 16M-byte sector
5. Up to seven wait states can be inserted for each block of asynchronous emulation memory
6. User-selectable bus sizes for each block of emulation memory
7. Four primary address spaces supported: user/supervisor program/data
8. Each block can be write protected

Signal Support:

1. To overcome delays from the target to the emulator, the emulator can provide the $\overline{\text{STERM}}$, $\overline{\text{CIIN}}$, and $\overline{\text{CBACK}}$, signals
2. These signals are independent of the mapping of synchronous emulation memory
3. Up to two wait states can be inserted for synchronous signals supplied by the emulator

Execution Control:

1. Single-step mode is available
2. Three hardware breakpoints are available
3. Hardware breakpoints do not alter target memory
4. Hardware breakpoints may be enabled/disabled with caches enabled or disabled
5. Hardware breakpoint event counter counts up to 64K events
6. Hardware breakpoints can be used as triggers by the SPA
7. Up to 64 software breakpoints can be used
8. Breakpoints can be specific to user or supervisor space

Processor Signals:

Signals that can be enabled or disabled:

$\overline{\text{DSACKx}}$	$\overline{\text{MMUDIS}}$
$\overline{\text{BERR}}$	$\overline{\text{CDIS}}$
$\overline{\text{BR}}$	$\overline{\text{BGACK}}$
$\overline{\text{CIOUT}}$	$\overline{\text{RESET}}$
$\overline{\text{CBREQ}}$	$\overline{\text{HALT}}$
$\overline{\text{STERM}}$	$\overline{\text{CBACK}}$
$\overline{\text{AVEC}}$	$\overline{\text{CIIN}}$
Interrupts	

Signals that are displayed by the emulator:

Target Power	Bus Error
$\overline{BR}$	$\overline{BGACK}$
Target Clock Stopped	Interrupt Line Status
Bus Timeout	$\overline{HALT}$
Double-Bus Fault Condition	$\overline{RESET}$

Signals that can be substituted by the emulator:

$\overline{BERR}$	$\overline{CDIS}$
$\overline{DSACKx}$	$\overline{CIOUT}$
$\overline{CBREQ}$	$\overline{MMUDIS}$
CLK	

Signals that can be substituted by the emulator based on selected address ranges:

$\overline{STERM}$	$\overline{CBACK}$
$\overline{CIIN}$	

11.4 DEVELOPMENT SOFTWARE

Usually a host computer system is used for software program development, editing, assembling, compiling, linking, and downloading to the HDS-300. The HDS-300 supports Motorola S record and UNIX "coff" download formats.

The 5.25 inch floppy disk drive can be used to save the downloaded code and to store macros written by the user for the target under development. Downloaded code may be patched using the HDS-300 reverse assembler and line-by-line assembler, then saved on the floppy disk as testing continues. Code can be quickly reloaded into the emulator from the floppy disk when software and hardware bugs cause target errors.

11.4.1 Source-Level Debug

The SLD works with Motorola's well-known C high-level language compiler to permit debug at the C statement level rather than at the assembly or hexadecimal levels. Debugging at this level provides breakpoints at the statement level and displays of program variables by name and typed value. The host-resident SLD works with the HDS-300 control station to provide a powerful, yet easy-to-use debugging tool that takes advantage of all the advanced capabilities of the HDS-300.

This versatile debugging tool allows the user to view and manipulate the target system via source language when programming in assembler or in C and downloading code in the UNIX "coff" format. Single lines may be stepped through, breakpoints set, and variables displayed and set, using the program source code rather than hexadecimal memory representations. Positions within the source code (such as those required for setting breakpoints or moving the browse cursor) can be referred to by line number or address or by a line number within a named function or source module.

Cursor movement, windowing, and other capabilities are provided. Windowing provides continuous source context display as code is executed. High-level mode windowing displays source while mixed mode windowing displays both source and the corresponding assembler level code.

Within the source window, two cursors may be displayed: an execution cursor that indicates the next source line to be executed and a browse cursor that may be positioned on any line in any text file for which the user has host operating system read-access permission.

SLD provides two additional windows. The variable window displays the current values of one or more user-specified variables at selected times as a check on the state of the program being debugged. The command log window displays the results obtained from executing the most recently entered commands.

11.4.2 Cross-Support Software

The system includes cross-development software consisting of 16- and 32-bit assemblers, linkers, C compilers, and SLD packages.

Cross-support software allows development of software and hardware in parallel. Software can be developed on a host, such as the M68DVLP system or an existing microcomputer, without having to wait for the microcomputer under development to be operational. Only the final testing and verification of the software needs to be done on the completed hardware.

All the cross-development software has been ported to several host computer systems. The microcomputer designer can use either Motorola's M68DVLP, a VAX system, a Macintosh computer, or a SUN-3 workstation as the host.

11.5 PART NUMBERS

SOFTWARE DEVELOPMENT HOST

Part Number	Description
M68DVLP	Multi-User UNIX Host System Including ALL 8-/16-/32-Bit Languages, SLD, and Tools
M68DVLP2	220V Version of M68DVLP

HARDWARE/SOFTWARE DEVELOPMENT STATION

Part Number	Description
M68HDS300	HDS-300 Control Station
M68HDS302	220V Version of HDS-300 Control Station

16-BIT EMULATOR MODULES

Part Number	Description
M68000HM3A	MC68000 Emulator and BSM with DIP Cable for HDS-300
M68000HM3B	MC68000 Emulator and BSM with PLCC Cable for HDS-300
M68000HM3C	MC68000 Emulator and BSM with PGA Cable for HDS-300
M68008HM3A	MC68008 Emulator and BSM with DIP Cable for HDS-300
M68008HM3B	MC68008 Emulator and BSM with PLCC Cable for HDS-300
M68010HM3A	MC68010 Emulator and BSM with DIP Capbe for HDS-300
M68010HM3B	MC68010 Emulator and BSM with PLCC Cable for HDS-300
M68010HM3C	MC68010 Emulator and BSM with PGA Cable for HDS-300

32-BIT EMULATOR MODULES

Part Number	Description
M68020HM3C-1	MC68020 Emulator with 64K RAM and PGA Cable for HDS-300 (16 MHz)
M68020HM3C-2	MC68020 Emulator with 256K RAM and PGA Cable for HDS-300 (16 MHz)
M68020HM3C-3	MC68020 Emulator with 1M RAM and PGA Cable for HDS-300 (16 MHz)
M68020HM3C-4	MC68020 Emulator with 64K RAM and PGA Cable for HDS-300 (25 MHz)
M68020HM3C-5	MC68020 Emulator with 256K RAM and PGA Cable for HDS-300 (25 MHz)
M68020HM3C-6	MC68020 Emulator with 1M RAM and PGA Cable for HDS-300 (25 MHz)
M68030HM3C-4	MC68030 Emulator with 64K RAM and PGA Cable for HDS-300 (25 MHz)
M68030HM3C-5	MC68030 Emulator with 256K RAM and PGA Cable for HDS-300 (25 MHz)
M68030HM3C-6	MC68030 Emulator with 1M RAM and PGA Cable for HDS-300 (25 MHz)

SYSTEM PERFORMANCE ANALYZER

Part Number	Description
M68HDS300SPA	System Performance Analyzer for 32-Bit Emulation

MEMORY EXPANSION

Part Number	Description
M68HDS3EMM1	64K Emulation Memory Expansion for HDS-300; MC68000/MC68008/MC68010
M68HDS3EMM2	128K Emulation Memory Expansion for HDS-300; MC68000/MC68008/MC68010
M68HDS3EMM3	256K Emulation Memory Expansion for HDS-300; MC68000/MC68008/MC68010
M68HDS3FDKT	HDS-300 Second Floppy Drive Kit

CABLES/HARDWARE

Part Number	Description
M68000/10DIPT	MC68000/MC68010 DIP Cable Probe for HDS-300
M68000/10PCCT	MC68000/MC68010 PLCC Cable Probe for HDS-300
M68000/10PGAT	MC68000/MC68010 PGA Cable Probe for HDS-300
M68008DIPT	MC68008 DIP Cable Probe for HDS-300
M68008PCCT	MC68008 PLCC Cable Probe for HDS-300

CROSS SOFTWARE

Part Number	Description
M68KTUTOR-D4	Tutor™ Source Listing, Rev. 1.3
M68KTUTORS	Tutor Source Code for MEX68KECB or VERSAdos™ 8-Inch Diskette

APPLE MACINTOSH CROSS SOFTWARE

Part Number	Description
M68HGBASM2	M68000 Family Macro Assembler/Linker for Apple Macintosh, 3.5 Inch Diskette
M68JGBCC2A	MC68020/MC68030 Cross C Compiler/Linker for Apple Macintosh, 3.5 Inch Diskette, 1-2 Users
M68HGBSLD00	MC68000/MC68008 Source Level Debug for HDS-300 and Apple Macintosh, 3.5 Inch Diskette
M68JGBSLD10	MC68010 Source Level Debug for HDS-300 and Apple Macintosh, 3.5 Inch Diskette
M68JGBSLD20	MC68020 Source Level Debug for HDS-300 and Apple Macintosh, 3.5 Inch Diskette
M68JGBSLD30	MC68030 Source Level Debug for HDS-300 and Apple Macintosh, 3.5 Inch Diskette

Tutor is a trademark of Motorola Inc.
VERSAdos is a trademark of Motorola Inc.

M68DVLP CROSS SOFTWARE

Part Number	Description
M68N2XBASM	MC68020 Macro Assembler for M68DVLP,2, Object Code on 5.25 Inch Diskette
M68N2QSASM	MC68020 Macro Assembler for M68DVLP,2, Source Code on Mag Tape
M68N2XSASM	MC68020 Macro Assembler for M68DVLP,2, Source Code on 5.25 Inch Diskette
M68NXBASM2	M68000 Family Macro Assembler/Linker for M68DVLP,2, Object Code, 5.25 Inch Diskette
M68NNXBPASMLK	PAL Port Assembler/Linker for M68DVLP,2, on 5.25 Inch Diskette
M68NNXSPASMLK	PAL Port Assembler/Linker Source Code for M68DVLP,2, on 5.25 Inch Diskette
M68N2XBCC	MC68000/MC68008/MC68010 C Compiler/Assembler/Linker for SYSTEM V/68, 1-8 Users
M68NNXBCC20B	MC68020 C Compiler/Assembler/Linker on 5.25 Inch Diskette, 1-8 Users
M68NNXBCC20C	MC68020 C Compiler/Assembler/Linker on 5.25 Inch Diskette, 1-6 Users
M68NNXSCC20-2	MC68020 C Compiler/Assembler/Linker, Source on 5.25 Inch Diskette
M68BNQBCC20C	MC68020 C Compiler/Assembly/Linker on Mag Tape, 1-16 Users
M68BNQBCC20D	MC68020 C Compiler/Assembler/Linker on Mag Tape, 1-32 Users
M68BNQBCC20E	MC68020 C Compiler/Assembler/Linker on Mag Tape, 1-64 Users
M68BNQBCC20F	MC68020 C Compiler/Assembler/Linker on Mag Tape, >64 Users
M68BNQSCC20-2	MC68020 C Compiler/Assembler/Linker Source Code on Mag Tape
M68BNQSOPT	Hi-Level C Optimizer, Source on Mag Tape
M68NXBCC2A	MC68020/MC68030 C Compiler/Assembler/Linker on 5.25 Inch Diskette, 1-2 Users
M68NXBCC2B	MC68020/MC68030 C Compiler/Assembler/Linker on 5.25 Inch Diskette, >2 Users
M68NNXBSLD00	M68000/MC68008/MC68010 Source Level Debug for HDS-300 and M68DVLP,2
M68NNXBSLD20	M68020 Source Level Debug for HDS-300 and M68DVLP,2
M68NXBSLD30	MC68030 Source Level Debug for HDS-300 and M68DVLP,2, 5.25 Inch Diskette
M68NNXBTLKT	VERSAdos Tool Kit, SYSTEM V/68, on 5.25 Inch Diskette
M68NNXSTLKT	VERSAdos Tool Kit, SYSTEM V/68, Source on 5.25 Inch Diskette

VAX VMS CROSS SOFTWARE

Part Number	Description
M68DOBASM2	M68000 Macro Assembler/Linker for VAX VMS, 600 Foot 9-Track Tape
M68DOBCC2A	MC68020/MC68030 C Compiler/Linker for VAX VMS, 1-2 Users, 600 Foot 9-Track Tape
M68DOBCC2B	MC68020/MC68030 C Compiler/Linker for VAX VMS, >2 Users, 600 Foot 9-Track Tape
M68DOBSLD00	MC68000/MC68008 Source Level Debug for HDS-300 and VAX VMS, 600 Foot 9-Track Tape
M68DOBSLD10	MC68010 Source Level Debug for HDS-300 and VAX VMS, 600 Foot 9-Track Tape
M68DOBSLD20	MC68020 Source Level Debug for HDS-300 and VAX VMS, 600 Foot 9-Track Tape
M68DOBSLD30	MC68030 Source Level Debug for HDS-300 and VAX VMS, 600 Foot 9-Track Tape
M68DHBASM2	M68000 Macro Assembler/Linker for VAX VMS, TK50 Tape
M68DHBCC2A	MC68020/MC68030 Cross C Compiler/Linker for VAX VMS, 1-2 Users, TK50 Tape
M68DHBCC2B	MC68020/MC68030 Cross C Compiler/Linker for VAX VMS, >2 Users, TK50 Tape
M68DHBSLD00	MC68000/MC68008 Source Level Debug for HDS-300 and VAX VMS, TK50 Tape
M68DHBSLD10	MC68010 Source Level Debug for HDS-300 and VAX VMS, TK50 Tape
M68DHBSLD20	MC68020 Source Level Debug for HDS-300 and VAX VMS, TK50 Tape
M68DHBSLD30	MC68030 Source Level Debug for HDS-300 and VAX VMS, TK50 Tape
M68DOSASM2	M68000 Family Macro Assembler/Linker for VAX VMS, 600 Foot 9-Track Tape, Source Code
M68DOSCC2	MC68020/MC68030 Cross C Compiler/Linker for VAX VMS, 600 Foot 9-Tack Tape, Source Code
M68DOSSLD00	MC68000/MC68008 Source Level Debug for VAX VMS, 600 Foot 9-Track Tape, Source Code
M68DOSSLD10	MC68010 Source Level Debug for VAX VMS, 600 Foot 9-Track Tape, Source Code
M68DOSSLD20	MC68020 Source Level Debug for VAX VMS, 600 Foot 9-Track Tape, Source Code
M68DOSSLD30	MC68030 Source Level Debug for VAX VMS, 600 Foot 9-Track Tape, Source Code
M68DHSASM2	M68000 Family Macro Assembler/Linker for VAX VMS, TK50 Tape, Source Code
M68DHSCC2	MC68020/MC68030 Cross C Compiler/Linker for VAX VMS, TK50 Tape, Source Code
M68DHSSLD00	MC68000/MC68008 Source Level Debug for VAX VMS, TK50 Tape, Source Code
M68DHSSLD10	MC68010 Source Level Debug for VAX VMS, TK50 Tape, Source Code
M68DHSSLD20	MC68020 Source Level Debug for VAX VMS, TK50 Tape, Source Code
M68DHSSLD30	MC68030 Source Level Debug for VAX VMS, TK50 Tape, Source Code

MPU SOFTWARE SUPPORT

Part Number	Description
MC68KTBFA	Token Bus Frame Analyzer

EVALUATION MODULES

Part Number	Description
MEX68KECB	MC68000 Educational Computer Board (includes Tutor S/W in ROM)