Welcome to E-XFL.COM

Understanding Embedded - Microprocessors

Embedded microprocessors are specialized computing chips designed to perform specific tasks within an embedded system. Unlike general-purpose microprocessors found in personal computers, embedded microprocessors are tailored for dedicated functions within larger systems, offering optimized performance, efficiency, and reliability. These microprocessors are integral to the operation of countless electronic devices, providing the computational power necessary for controlling processes, handling data, and managing communications.

Applications of Embedded - Microprocessors

Embedded microprocessors are utilized across a broad spectrum of applications, making them indispensable in

Details	
Product Status	Obsolete
Core Processor	EC000
Number of Cores/Bus Width	1 Core, 32-Bit
Speed	10MHz
Co-Processors/DSP	-
RAM Controllers	-
Graphics Acceleration	No
Display & Interface Controllers	-
Ethernet	-
SATA	-
USB	-
Voltage - I/O	5.0V
Operating Temperature	0°C ~ 70°C (TA)
Security Features	-
Package / Case	68-BCPGA
Supplier Device Package	68-PGA (26.92x26.92)
Purchase URL	https://www.e-xfl.com/product-detail/nxp-semiconductors/mc68hc000rc10

1.4 MC68HC000

The primary benefit of the MC68HC000 is 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 bit microprocessor architecture. The MC68HC000 has a 16-bit data bus implementation of the MC68000 and is upward code-compatible with the MC68010 virtual extensions and the MC68020 32-bit implementation of the architecture.

1.5 MC68HC001

The MC68HC001 provides a functional extension to the MC68HC000 HCMOS 16-/32-bit microprocessor with the addition of statically selectable 8- or 16-bit data bus operation. The MC68HC001 is object-code compatible with the MC68HC000, and code written for the MC68HC001 can be migrated without modification to any member of the M68000 Family.

1.6 MC68EC000

The MC68EC000 is an economical high-performance embedded controller designed to suit the needs of the cost-sensitive embedded controller market. The HCMOS MC68EC000 has an internal 32-bit architecture that is supported by a statically selectable 8- or 16-bit data bus. This architecture provides a fast and efficient processing device that can satisfy the requirements of sophisticated applications based on high-level languages.

The MC68EC000 is object-code compatible with the MC68000, and code written for the MC68EC000 can be migrated without modification to any member of the M68000 Family.

The MC68EC000 brings the performance level of the M68000 Family to cost levels previously associated with 8-bit microprocessors. The MC68EC000 benefits from the rich M68000 instruction set and its related high code density with low memory bandwidth requirements.

tables. The SFC and DFC registers allow the supervisor to access user data space or emulate CPU space cycles.

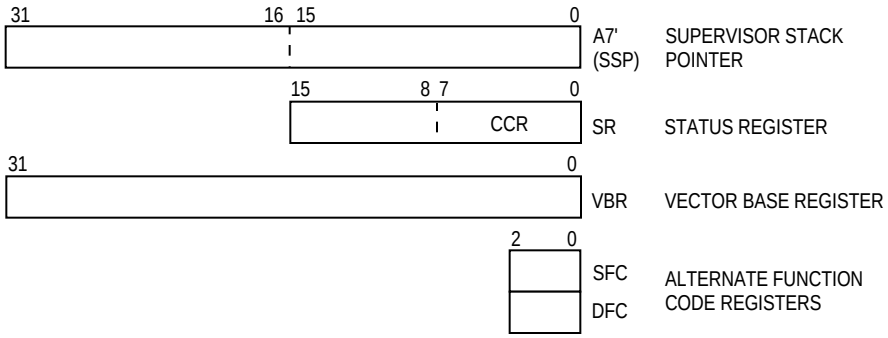


Figure 2-3. Supervisor Programmer's Model Supplement (MC68010)

2.1.3 Status Register

The status register (SR),contains the interrupt mask (eight levels available) and the following condition codes: overflow (V), zero (Z), negative (N), carry (C), and extend (X). Additional status bits indicate that the processor is in the trace (T) mode and/or in the supervisor (S) state (see Figure 2-4). Bits 5, 6, 7, 11, 12, and 14 are undefined and reserved for future expansion

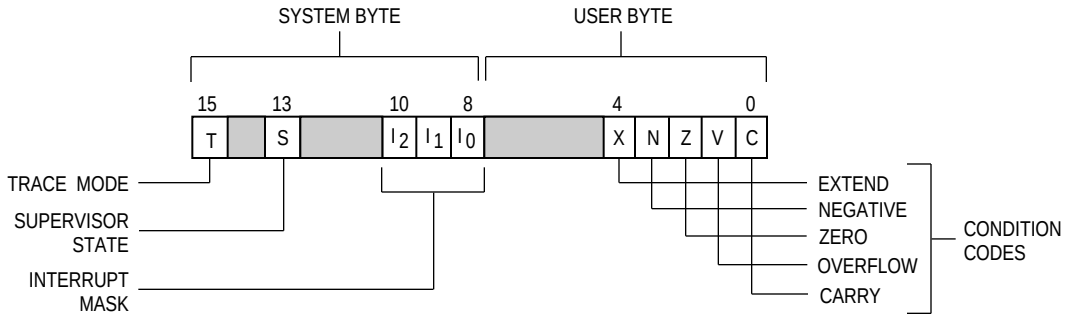


Figure 2-4. Status Register

2.2 DATA TYPES AND ADDRESSING MODES

The five basic data types supported are as follows:

1. Bits
2. Binary-Coded-Decimal (BCD) Digits (4 Bits)
3. Bytes (8 Bits)
4. Words (16 Bits)
5. Long Words (32 Bits)

Table 2-2. Instruction Set Summary (Sheet 1 of 4)

Opcode	Operation	Syntax
ABCD	$Source_{10} + Destination_{10} + X \rightarrow Destination$	ABCD Dy,Dx ABCD –(Ay), –(Ax)
ADD	Source + Destination → Destination	ADD <ea>,Dn ADD Dn,<ea>
ADDA	Source + Destination → Destination	ADDA <ea>,An
ADDI	Immediate Data + Destination → Destination	ADDI # <data>,<ea>
ADDQ	Immediate Data + Destination → Destination	ADDQ # <data>,<ea>
ADDX	Source + Destination + X → Destination	ADDX Dy, Dx ADDX –(Ay), –(Ax)
AND	Source Λ Destination → Destination	AND <ea>,Dn AND Dn,<ea>
ANDI	Immediate Data Λ Destination → Destination	ANDI # <data>, <ea>
ANDI to CCR	Source Λ CCR → CCR	ANDI # <data>, CCR
ANDI to SR	If supervisor state then Source Λ SR → SR else TRAP	ANDI # <data>, SR
ASL, ASR	Destination Shifted by <count> → Destination	ASd Dx,Dy ASd # <data>,Dy ASd <ea>
Bcc	If (condition true) then PC + d → PC	Bcc <label>
BCHG	~ (<number> of Destination) → Z; ~ (<number> of Destination) → <bit number> of Destination	BCHG Dn,<ea> BCHG # <data>,<ea>
BCLR	~ (<bit number> of Destination) → Z; 0 → <bit number> of Destination	BCLR Dn,<ea> BCLR # <data>,<ea>
BKPT	Run breakpoint acknowledge cycle; TRAP as illegal instruction	BKPT # <data>
BRA	PC + d → PC	BRA <label>
BSET	~ (<bit number> of Destination) → Z; 1 → <bit number> of Destination	BSET Dn,<ea> BSET # <data>,<ea>
BSR	SP – 4 → SP; PC → (SP); PC + d → PC	BSR <label>
BTST	– (<bit number> of Destination) → Z;	BTST Dn,<ea> BTST # <data>,<ea>
CHK	If Dn < 0 or Dn > Source then TRAP	CHK <ea>,Dn
CLR	0 → Destination	CLR <ea>
CMP	Destination—Source → cc	CMP <ea>,Dn
CMPA	Destination—Source	CMPA <ea>,An
CMPI	Destination —Immediate Data	CMPI # <data>,<ea>
CMPM	Destination—Source → cc	CMPM (Ay)+, (Ax)+
DBcc	If condition false then (Dn – 1 → Dn; If Dn ≠ –1 then PC + d → PC)	DBcc Dn,<label>

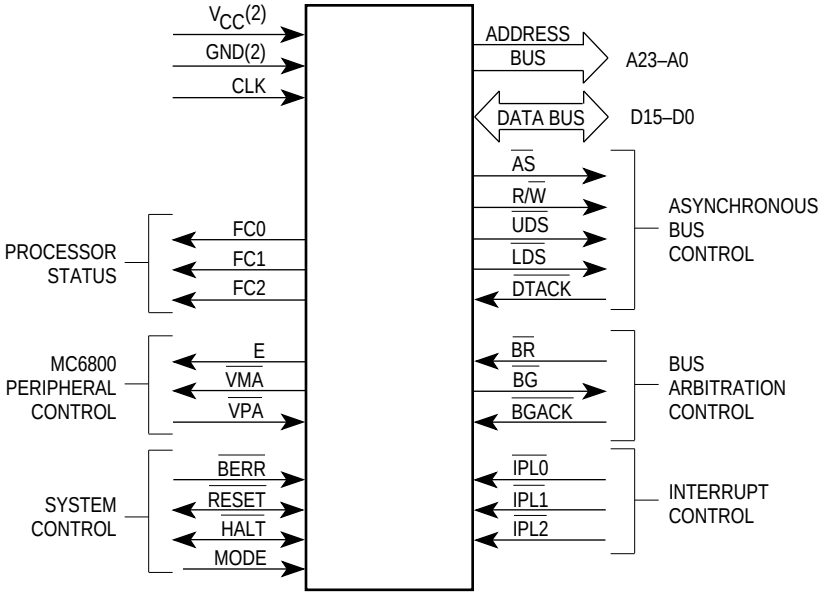


Figure 3-2. Input and Output Signals (MC68HC001)

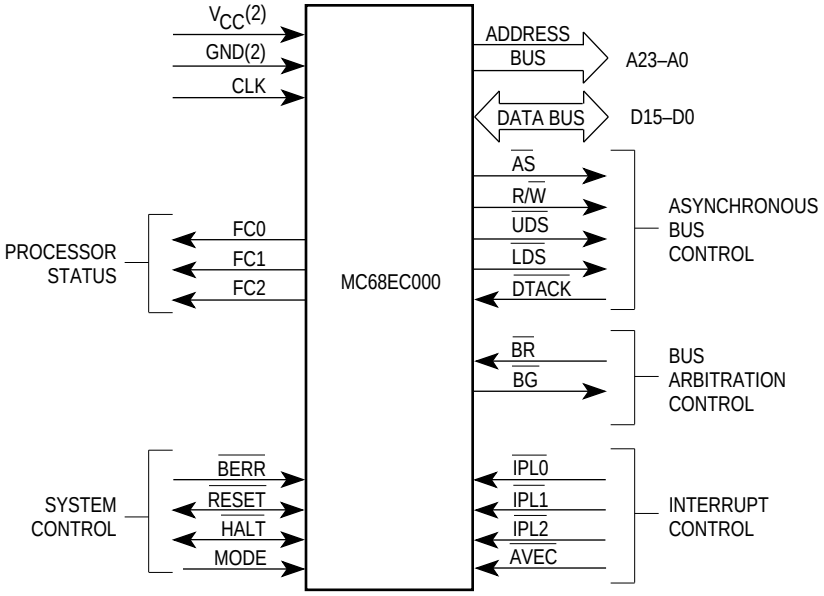


Figure 3-3. Input and Output Signals (MC68EC000)

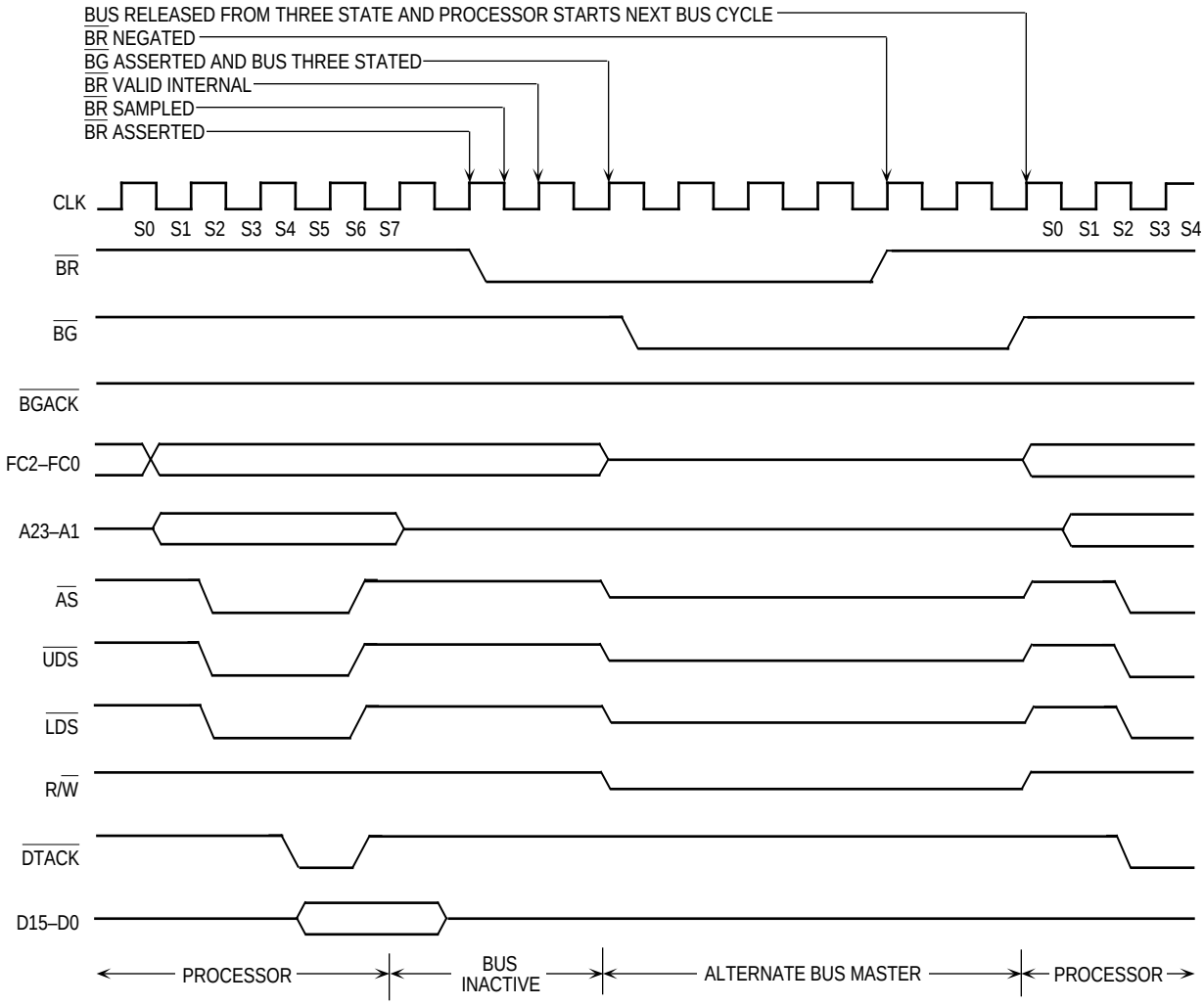


Figure 5-23. 2-Wire Bus Arbitration Timing Diagram—Bus Inactive

SECTION 6
EXCEPTION PROCESSING

This section describes operations of the processor outside the normal processing associated with the execution of instructions. The functions of the bits in the supervisor portion of the status register are described: the supervisor/user bit, the trace enable bit, and the interrupt priority mask. Finally, the sequence of memory references and actions taken by the processor for exception conditions are described in detail.

The processor is always in one of three processing states: normal, exception, or halted. The normal processing state is 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, resulting from execution of a STOP instruction. In this state, no further memory references are made.

An additional, special case of the normal state is the loop mode of the MC68010, optionally entered when a test condition, decrement, and branch (DBcc) instruction is executed. In the loop mode, only operand fetches occur. See **Appendix A MC68010 Loop Mode Operation**.

The exception processing state is associated with interrupts, trap instructions, tracing, and other exceptional 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 provides an efficient context switch so that the processor can handle unusual conditions.

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 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.

6.1 PRIVILEGE MODES

The processor operates in one of two levels of privilege: the supervisor mode or the user mode. The privilege mode determines which operations are legal. The mode is optionally used by an external memory management device to control and translate accesses. The mode is also used to choose between the supervisor stack pointer (SSP) and the user stack pointer (USP) in instruction references.

Table 6-2. Exception Vector Assignment

Vectors Numbers		Address		Space[6]	Assignment
Hex	Decimal	Dec	Hex		
0	0	0	000	SP	Reset: Initial SSP[2]
1	1	4	004	SP	Reset: Initial PC[2]
2	2	8	008	SD	Bus Error
3	3	12	00C	SD	Address Error
4	4	16	010	SD	Illegal Instruction
5	5	20	014	SD	Zero Divide
6	6	24	018	SD	CHK Instruction
7	7	28	01C	SD	TRAPV Instruction
8	8	32	020	SD	Privilege Violation
9	9	36	024	SD	Trace
A	10	40	028	SD	Line 1010 Emulator
B	11	44	02C	SD	Line 1111 Emulator
C	12[1]	48	030	SD	(Unassigned, Reserved)
D	13[1]	52	034	SD	(Unassigned, Reserved)
E	14	56	038	SD	Format Error[5]
F	15	60	03C	SD	Uninitialized Interrupt Vector
10–17	16–23[1]	64	040	SD	(Unassigned, Reserved)
		92	05C		—
18	24	96	060	SD	Spurious Interrupt[3]
19	25	100	064	SD	Level 1 Interrupt Autovector
1A	26	104	068	SD	Level 2 Interrupt Autovector
1B	27	108	06C	SD	Level 3 Interrupt Autovector
1C	28	112	070	SD	Level 4 Interrupt Autovector
1D	29	116	074	SD	Level 5 Interrupt Autovector
1E	30	120	078	SD	Level 6 Interrupt Autovector
1F	31	124	07C	SD	Level 7 Interrupt Autovector
20–2F	32–47	128	080	SD	TRAP Instruction Vectors[4]
		188	0BC		—
30–3F	48–63[1]	192	0C0	SD	(Unassigned, Reserved)
		255	0FF		—
40–FF	64–255	256	100	SD	User Interrupt Vectors
		1020	3FC		—

NOTES:
1. Vector numbers 12, 13, 16–23, and 48–63 are reserved for future enhancements by Motorola. No user peripheral devices should be assigned these numbers.
2. Reset vector (0) requires four words, unlike the other vectors which only require two words, and is located in the supervisor program space.
3. The spurious interrupt vector is taken when there is a bus error indication during interrupt processing.
4. TRAP #n uses vector number 32+ n.
5. MC68010 only. This vector is unassigned, reserved on the MC68000 and MC68008.
6. SP denotes supervisor program space, and SD denotes supervisor data space.

Table 7-3. Move Word Instruction Execution Times

Source	Destination								
	Dn	An	(An)	(An)+	–(An)	(d_{16}, An)	(d_8, An, Xn)*	(xxx).W	(xxx).L
Dn	**8**(2/0)	**8**(2/0)	**16**(2/2)	**16**(2/2)	**16**(2/2)	**24**(4/2)	**26**(4/2)	**24**(4/2)	**32**(6/2)
An	**8**(2/0)	**8**(2/0)	**16**(2/2)	**16**(2/2)	**16**(2/2)	**24**(4/2)	**26**(4/2)	**24**(4/2)	**32**(6/2)
(An)	**16**(4/0)	**16**(4/0)	**24**(4/2)	**24**(4/2)	**24**(4/2)	**32**(6/2)	**34**(6/2)	**32**(6/2)	**40**(8/2)
(An)+	**16**(4/0)	**16**(4/0)	**24**(4/2)	**24**(4/2)	**24**(4/2)	**32**(6/2)	**34**(6/2)	**32**(6/2)	**40**(8/2)
–(An)	**18**(4/0)	**18**(4/0)	**26**(4/2)	**26**(4/2)	**26**(4/2)	**34**(6/2)	**32**(6/2)	**34**(6/2)	**42**(8/2)
(d_{16}, An)	**24**(6/0)	**24**(6/0)	**32**(6/2)	**32**(6/2)	**32**(6/2)	**40**(8/2)	**42**(8/2)	**40**(8/2)	**48**(10/2)
(d_8, An, Xn)*	**26**(6/0)	**26**(6/0)	**34**(6/2)	**34**(6/2)	**34**(6/2)	**42**(8/2)	**44**(8/2)	**42**(8/2)	**50**(10/2)
(xxx).W	**24**(6/0)	**24**(6/0)	**32**(6/2)	**32**(6/2)	**32**(6/2)	**40**(8/2)	**42**(8/2)	**40**(8/2)	**48**(10/2)
(xxx).L	**32**(8/0)	**32**(8/0)	**40**(8/2)	**40**(8/2)	**40**(8/2)	**48**(10/2)	**50**(10/2)	**48**(10/2)	**56**(12/2)
(d_{16}, PC)	**24**(6/0)	**24**(6/0)	**32**(6/2)	**32**(6/2)	**32**(6/2)	**40**(8/2)	**42**(8/2)	**40**(8/2)	**48**(10/2)
(d_8, PC, Xn)*	**26**(6/0)	**26**(6/0)	**34**(6/2)	**34**(6/2)	**34**(6/2)	**42**(8/2)	**44**(8/2)	**42**(8/2)	**50**(10/2)
#<data>	**16**(4/0)	**16**(4/0)	**24**(4/2)	**24**(4/2)	**24**(4/2)	**32**(6/2)	**34**(6/2)	**32**(6/2)	**40**(8/2)

*The size of the index register (Xn) does not affect execution time.

Table 7-4. Move Long Instruction Execution Times

Source	Destination								
	Dn	An	(An)	(An)+	–(An)	(d_{16}, An)	(d_8, An, Xn)*	(xxx).W	(xxx).L
Dn	**8**(2/0)	**8**(2/0)	**24**(2/4)	**24**(2/4)	**24**(2/4)	**32**(4/4)	**34**(4/4)	**32**(4/4)	**40**(6/4)
An	**8**(2/0)	**8**(2/0)	**24**(2/4)	**24**(2/4)	**24**(2/4)	**32**(4/4)	**34**(4/4)	**32**(4/4)	**40**(6/4)
(An)	**24**(6/0)	**24**(6/0)	**40**(6/4)	**40**(6/4)	**40**(6/4)	**48**(8/4)	**50**(8/4)	**48**(8/4)	**56**(10/4)
(An)+	**24**(6/0)	**24**(6/0)	**40**(6/4)	**40**(6/4)	**40**(6/4)	**48**(8/4)	**50**(8/4)	**48**(8/4)	**56**(10/4)
–(An)	**26**(6/0)	**26**(6/0)	**42**(6/4)	**42**(6/4)	**42**(6/4)	**50**(8/4)	**52**(8/4)	**50**(8/4)	**58**(10/4)
(d_{16}, An)	**32**(8/0)	**32**(8/0)	**48**(8/4)	**48**(8/4)	**48**(8/4)	**56**(10/4)	**58**(10/4)	**56**(10/4)	**64**(12/4)
(d_8, An, Xn)*	**34**(8/0)	**34**(8/0)	**50**(8/4)	**50**(8/4)	**50**(8/4)	**58**(10/4)	**60**(10/4)	**58**(10/4)	**66**(12/4)
(xxx).W	**32**(8/0)	**32**(8/0)	**48**(8/4)	**48**(8/4)	**48**(8/4)	**56**(10/4)	**58**(10/4)	**56**(10/4)	**64**(12/4)
(xxx).L	**40**(10/0)	**40**(10/0)	**56**(10/4)	**56**(10/4)	**56**(10/4)	**64**(12/4)	**66**(12/4)	**64**(12/4)	**72**(14/4)
(d_{16}, PC)	**32**(8/0)	**32**(8/0)	**48**(8/4)	**48**(8/4)	**48**(8/4)	**56**(10/4)	**58**(10/4)	**56**(10/4)	**64**(12/4)
(d_8, PC, Xn)*	**34**(8/0)	**34**(8/0)	**50**(8/4)	**50**(8/4)	**50**(8/4)	**58**(10/4)	**60**(10/4)	**58**(10/4)	**66**(12/4)
#<data>	**24**(6/0)	**24**(6/0)	**40**(6/4)	**40**(6/4)	**40**(6/4)	**48**(8/4)	**50**(8/4)	**48**(8/4)	**56**(10/4)

*The size of the index register (Xn) does not affect execution time.

7.3 STANDARD INSTRUCTION EXECUTION TIMES

The numbers of clock periods shown in Table 7-5 indicate the times required to perform the operations, store the results, and read the next instruction. The total number of clock periods, the number of read cycles, and the number of write cycles are shown in the previously described format. The number of clock periods, the number of read cycles, and the number of write cycles, respectively, must be added to those of the effective address calculation where indicated by a plus sign (+).

In Table 7-5, the following notation applies:

An — Address register operand
Dn — Data register operand
ea — An operand specified by an effective address
M — Memory effective address operand

Table 7-5. Standard Instruction Execution Times

Instruction	Size	op<ea>, An	op<ea>, Dn	op Dn, <M>
ADD/ADDA	Byte Word Long	— **12**(2/0)+ **10**(2/0)+**	**8**(2/0)+ **8**(2/0)+ **10**(2/0)+**	**12**(2/1)+ **16**(2/2)+ **24**(2/4)+
AND	Byte Word Long	— — —	**8**(2/0)+ **8**(2/0)+ **10**(2/0)+**	**12**(2/1)+ **16**(2/2)+ **24**(2/4)+
CMP/CMPA	Byte Word Long	— **10**(2/0)+ **10**(2/0)+	**8**(2/0)+ **8**(2/0)+ **10**(2/0)+	— — —
DIVS DIVU	— —	— —	**162**(2/0)+* **144**(2/0)+*	— —
EOR	Byte, Word, Long	— — —	**8**(2/0)+*** **8**(2/0)+*** **12**(2/0)+***	**12**(2/1)+ **16**(2/2)+ **24**(2/4)+
MULS MULU	— —	— —	**74**(2/0)+* **74**(2/0)+*	— —
OR	Byte, Word Long	— — —	**8**(2/0)+ **8**(2/0)+ **10**(2/0)+**	**12**(2/1)+ **16**(2/2)+ **24**(2/4)+
SUB	Byte, Word Long	 **12**(2/0)+ **10**(2/0)+**	**8**(2/0)+ **8**(2/0)+ **10**(2/0)+**	**12**(2/1)+ **16**(2/2)+ **24**(2/4)+

\+ Add effective address calculation time.
* Indicates maximum base value added to word effective address time
** The base time of 10 clock periods is increased to 12 if the effective address mode is register direct or immediate (effective address time should also be added).
*** Only available effective address mode is data register direct.
DIVS, DIVU — The divide algorithm used by the MC68008 provides less than 10% difference between the best- and worst-case timings.
MULS, MULU — The multiply algorithm requires 42+2n clocks where n is defined as:
MULS: n = tag the <ea> with a zero as the MSB; n is the resultant number of 10 or 01 patterns in the 17-bit source; i.e., worst case happens when the source is $5555.
MULU: n = the number of ones in the <ea>

7.4 IMMEDIATE INSTRUCTION EXECUTION TIMES

The numbers of clock periods shown in Table 7-6 include the times to fetch immediate operands, perform the operations, store the results, and read the next operation. The total number of clock periods, the number of read cycles, and the number of write cycles are

shown in the previously described format. The number of clock periods, the number of read cycles, and the number of write cycles, respectively, must be added to those of the effective address calculation where indicated by a plus sign (+).

In Table 7-6, the following notation applies:

— Immediate operand
Dn — Data register operand
An — Address register operand
M — Memory operand

Table 7-6. Immediate Instruction Execution Times

Instruction	Size	op #, Dn	op #, An	op #, M
ADDI	Byte	**16**(4/0)	—	**20**(4/1)+
	Word	**16**(4/0)	—	**24**(4/2)+
	Long	**28**(6/0)	—	**40**(6/4)+
ADDQ	Byte	**8**(2/0)	—	**12**(2/1)+
	Word	**8**(2/0)	**12**(2/0)	**16**(2/2)+
	Long	**12**(2/0)	**12**(2/0)	**24**(2/4)+
ANDI	Byte	**16**(4/0)	—	**20**(4/1)+
	Word	**16**(4/0)	—	**24**(4/2)+
	Long	**28**(6/0)	—	**40**(6/4)+
CMPI	Byte	**16**(4/0)	—	**16**(4/0)
	Word	**16**(4/0)	—	**16**(4/0)
	Long	**26**(6/0)	—	**24**(6/0)
EORI	Byte	**16**(4/0)	—	**20**(4/1)+
	Word	**16**(4/0)	—	**24**(4/2)+
	Long	**28**(6/0)	—	**40**(6/4)+
MOVEQ	Long	**8**(2/0)	—	—
ORI	Byte	**16**(4/0)	—	**20**(4/1)+
	Word	**16**(4/0)	—	**24**(4/2)+
	Long	**28**(6/0)	—	**40**(6/4)+
SUBI	Byte	**16**(4/0)	—	**12**(2/1)+
	Word	**16**(4/0)	—	**16**(2/2)+
	Long	**28**(6/0)	—	**24**(2/4)+
SUBQ	Byte	**8**(2/0)	—	**20**(4/1)+
	Word	**8**(2/0)	**12**(2/0)	**24**(4/2)+
	Long	**12**(2/0)	**12**(2/0)	**40**(6/4)+

+Add effective address calculation time.

7.5 SINGLE OPERAND INSTRUCTION EXECUTION TIMES

Table 7-7 lists the timing data for the single operand instructions. The total number of clock periods, the number of read cycles, and the number of write cycles are shown in the previously described format. The number of clock periods, the number of read cycles, and the number of write cycles, respectively, must be added to those of the effective address calculation where indicated by a plus sign (+).

Table 8-11. Multiprecision Instruction Execution Times

Instruction	Size	op Dn, Dn	op M, M
ADDX	Byte, Word	**4**(1/0)	**18**(3/1)
	Long	**8**(1/0)	**30**(5/2)
CMPM	Byte, Word	—	**12**(3/0)
	Long	—	**20**(5/0)
SUBX	Byte, Word	**4**(1/0)	**18**(3/1)
	Long	**8**(1/0)	**30**(5/2)
ABCD	Byte	**6**(1/0)	**18**(3/1)
SBCD	Byte	**6**(1/0)	**18**(3/1)

8.11 MISCELLANEOUS INSTRUCTION EXECUTION TIMES

Tables 8-12 and 8-13 list the timing data for miscellaneous instructions. The total number of clock periods, the number of read cycles, and the number of write cycles are shown in the previously described format. The number of clock periods, the number of read cycles, and the number of write cycles, respectively, must be added to those of the effective address calculation where indicated by a plus sign (+).

SECTION 9
MC68010 INSTRUCTION EXECUTION TIMES

This section contains listings of the instruction execution times in terms of external clock (CLK) periods for the MC68010. In this data, it is assumed that both memory read and write cycles consist of four clock periods. A longer memory cycle causes the generation of wait states that must be added to the total instruction times.

The number of bus read and write cycles for each instruction is also included with the timing data. This data is shown as

n(r/w)

where:

n is the total number of clock periods
r is the number of read cycles
w is the number of write cycles

For example, a timing number shown as 18(3/1) means that 18 clock cycles are required to execute the instruction. Of the 18 clock periods, 12 are used for the three read cycles (four periods per cycle). Four additional clock periods are used for the single write cycle, for a total of 16 clock periods. The bus is idle for two clock periods during which the processor completes the internal operations required for the instructions.

NOTE

The total number of clock periods (n) includes instruction fetch and all applicable operand fetches and stores.

10.3 POWER CONSIDERATIONS

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

$$T_J = T_A + (P_D \bullet {}^{\theta}J_A) \quad (1)$$

where:

T_A = Ambient Temperature, °C
${}^{\theta}J_A$ = 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:

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

where K is a constant pertaining to the particular part. K can be determined from equation (3) by measuring P_D (at 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 10-1 gives the graphic solution to the above equations for the specified power dissipation of 1.5 W over the ambient temperature range of -55 °C to 125 °C using a maximum ${}^{\theta}J_A$ of 45 °C/W. Ambient temperature is that of the still air surrounding the device. Lower values of ${}^{\theta}J_A$ cause the curve to shift downward slightly; for instance, for ${}^{\theta}J_A$ of 40 °/W, the curve is just below 1.4 W at 25 °C.

The total thermal resistance of a package (${}^{\theta}J_A$) can be separated into two components, ${}^{\theta}J_C$ and ${}^{\theta}C_A$, representing the barrier to heat flow from the semiconductor junction to the package (case) surface (${}^{\theta}J_C$) and from the case to the outside ambient air (${}^{\theta}C_A$). These terms are related by the equation:

$${}^{\theta}J_A = {}^{\theta}J_C + {}^{\theta}C_A \quad (4)$$

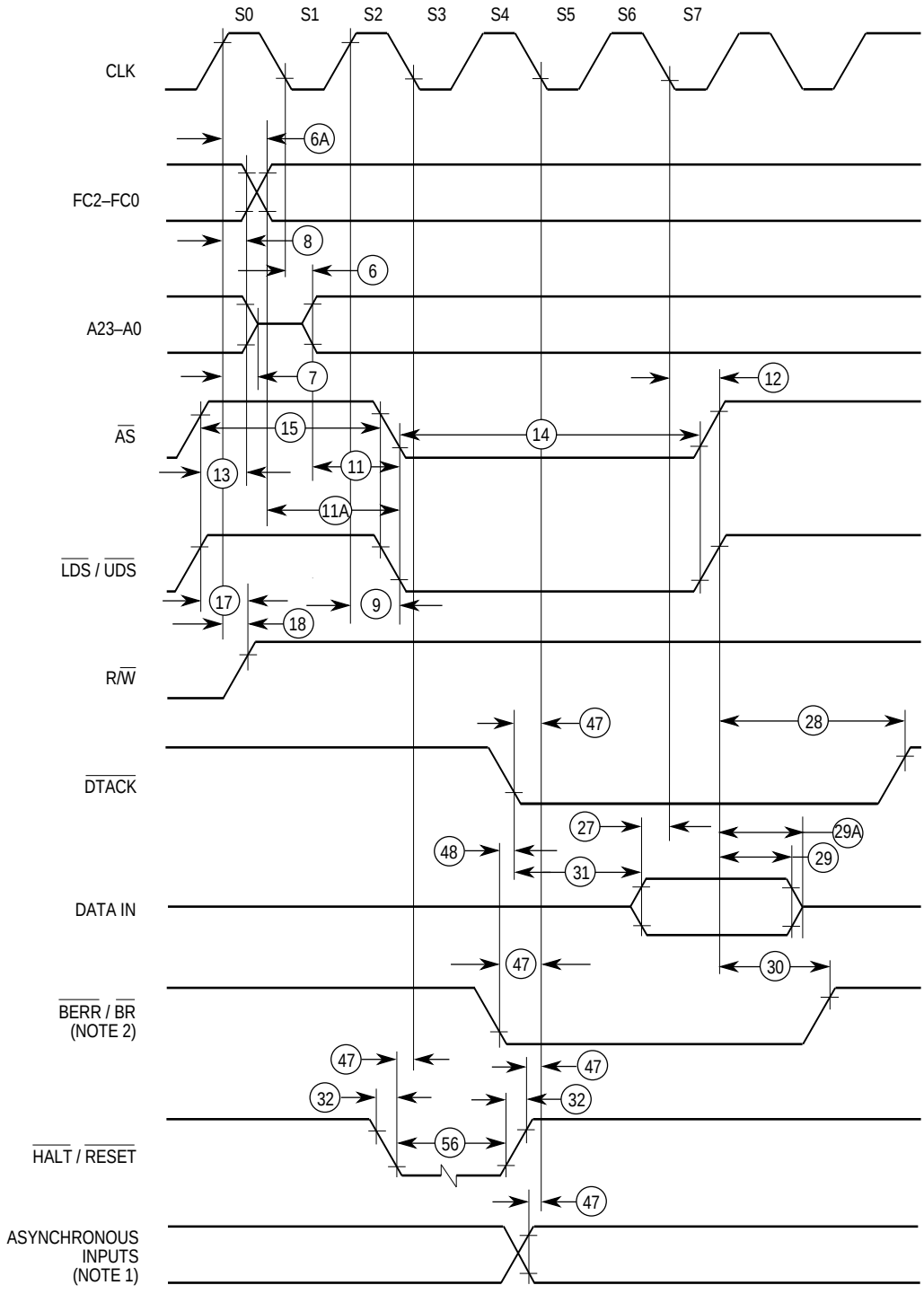
${}^{\theta}J_C$ is device related and cannot be influenced by the user. However, ${}^{\theta}C_A$ is user dependent and can be minimized by such thermal management techniques as heat sinks, ambient air cooling, and thermal convection. Thus, good thermal management on the part of the user can significantly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

10.6 MC68000/68008/68010 DC ELECTRICAL CHARACTERISTICS

(V_{CC}=5.0 VDC±5%; GND=0 VDC; T_A=T_L TO T_H)

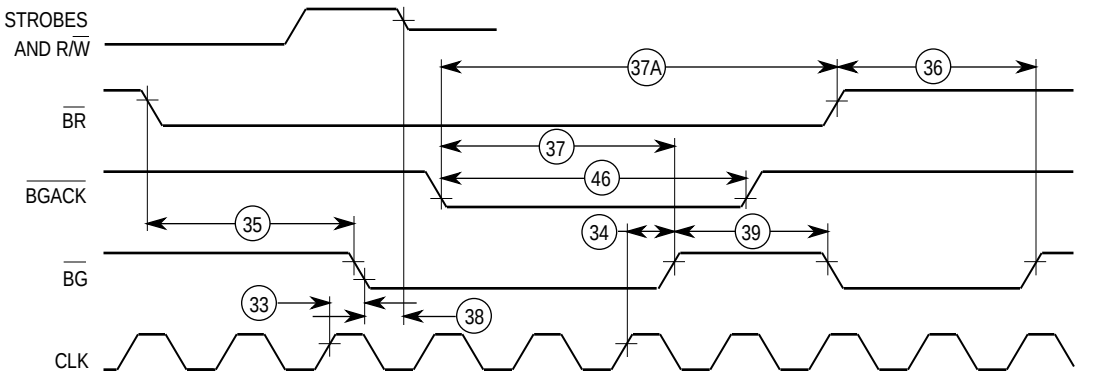
Characteristic	Symbol	Min	Max	Unit
Input High Voltage	V_{IH}	2.0	V_{CC}	V
Input Low Voltage	V_{IL}	GND-0.3	0.8	V
Input Leakage Current @ 5.25 V $\overline{BERR}$, $\overline{BGACK}$, $\overline{BR}$, $\overline{DTACK}$, CLK, $\overline{IPL0}$—$\overline{IPL2}$, $\overline{VPA}$ $\overline{HALT}$, $\overline{RESET}$	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* $\overline{AS}$, 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) $\overline{HALT}$ (I_{OL} = 3.2 mA) A1—A23, $\overline{BG}$, FC0-FC2 (I_{OL} = 5.0 mA) $\overline{RESET}$ (I_{OL} = 5.3 mA) 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.



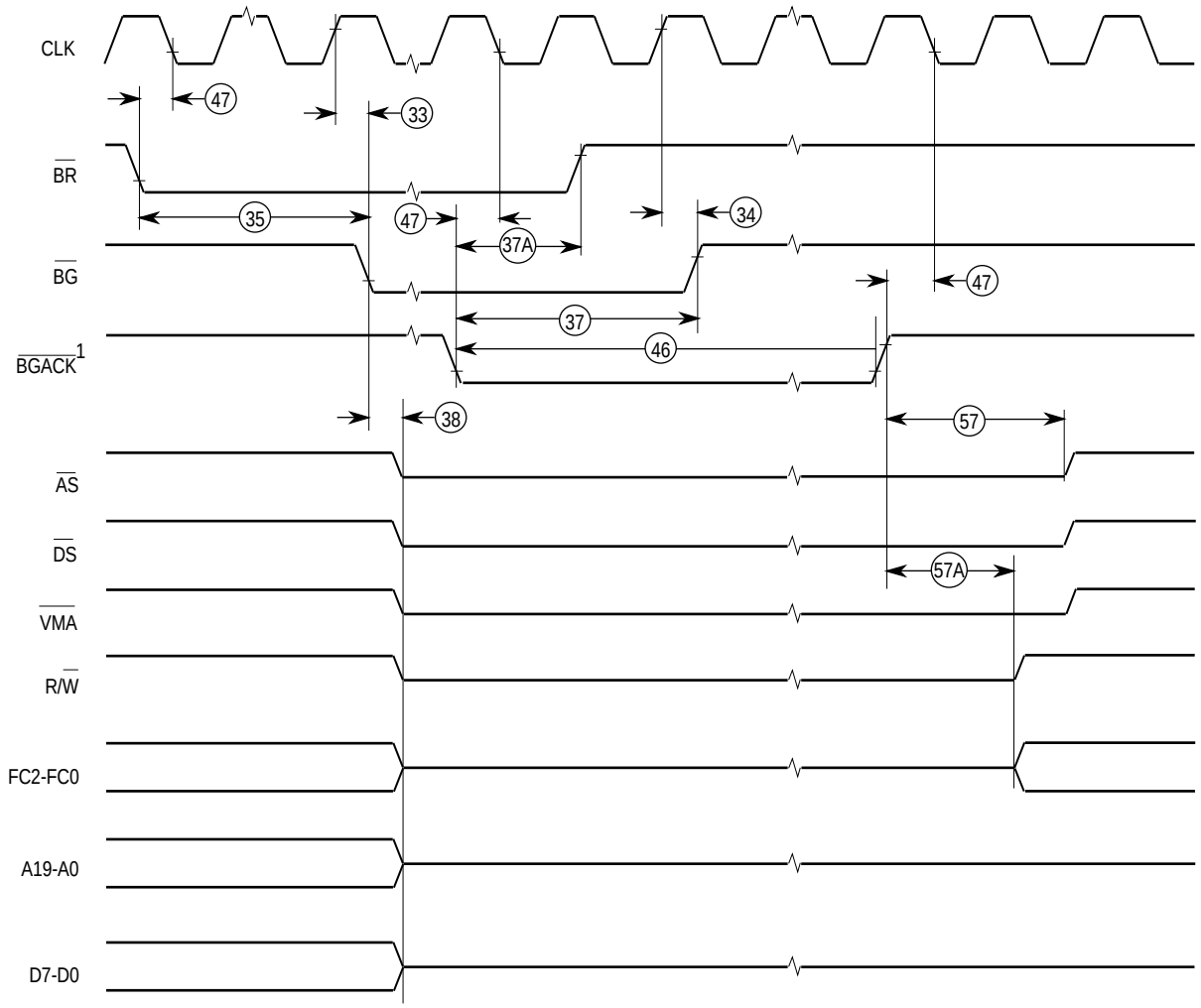
NOTES:
1. Setup time for the asynchronous inputs IPL2–IPL0 and AVEC (#47) guarantees their recognition at the next falling edge of the clock.
2. BR need fall at this time only to insure being recognized at the end of the bus cycle.
3. Timing measurements are referenced to and from a low voltage of 0.8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

Figure 10-4. Read Cycle Timing Diagram
(Applies To All Processors Except The MC68EC000)



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

Figure 10-7. Bus Arbitration Timing
(Applies To All Processors Except The MC68EC000)



NOTES: Waveform measurements for all inputs and outputs are specified at: logic high 2.0 V, logic low = 0.8 V.
1. MC68008 52-Pin Version only.

Figure 10-9. Bus Arbitration Timing — Idle Bus Case
(Applies To All Processors Except The MC68EC000)

10.13 MC68EC000 DC ELECTRICAL SPECIFICATIONS (VCC=5.0 VDC ± 5;PC; 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	$\overline{BERR}$, $\overline{BR}$, $\overline{DTACK}$, CLK, $\overline{IPL2}$–$\overline{IPL0}$, $\overline{AVEC}$ MODE, $\overline{HALT}$, $\overline{RESET}$	I_{in}	— —	2.5 20	µA
Three-State (Off State) Input Current @2.4 V/0.4 V	AS, A23–A0, D15–D0, FC2–FC0, $\overline{LDS}$, R/$\overline{W}$, $\overline{UDS}$	I_{TSI}	—	20	µA
Output High Voltage (I_{OH}=–400 µA)	AS, A23–A0, $\overline{BG}$, D15–D0, FC2–FC0, $\overline{LDS}$, R/$\overline{W}$, $\overline{UDS}$	V_{OH}	VCC –0.75	—	V
Output Low Voltage		V_{OL}			V
(IOL = 1.6 mA)	$\overline{HALT}$		—	0.5	
(IOL = 3.2 mA)	A23–A0, $\overline{BG}$, FC2–FC0		—	0.5	
(IOL = 5.0 mA)	$\overline{RESET}$		—	0.5	
(IOL = 5.3 mA)	$\overline{AS}$, D15–D0, $\overline{LDS}$, R/$\overline{W}$, $\overline{UDS}$		—	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	
	f= 20 MHz		—	70	
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	
	f=20 MHz		—	0.38	
Capacitance (Vin=0 V, TA=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.

Table A-1. MC68010 Loop Mode Instructions

Opcodes	Applicable Addressing Modes
MOVE [BWL]	(Ay) to (Ax) (Ay) to (Ax)+ (Ay) to –(Ax) (Ay)+ to (Ax) (Ay)+ to –(Ax) –(Ay) to (Ax) –(Ay) to (Ax)+ –(Ay) to –(Ax) Ry to (Ax) Ry to (Ax)+
ADD [BWL] AND [BWL] CMP [BWL] OR [BWL] SUB [BWL]	(Ay) to Dx (Ay)+ to Dx –(Ay) to Dx
ADDA [WL] CMPA [WL] SUBA [WL]	(Ay) to Ax –(Ay) to Ax (Ay)+ to Ax
ADD [BWL] AND [BWL] EOR [BWL] OR [BWL] SUB [BWL]	Dx to (Ay) Dx to (Ay)+ Dx to –(Ay)
ABCD [B] ADDX [BWL] SBCD [B] SUBX [BWL]	–(Ay) to –(Ax)
CMP [BWL]	(Ay)+ to (Ax)+
CLR [BWL] NEG [BWL] NEGX [BWL} NOT [BWL] TST [BWL] NBCD [B]	(Ay) (Ay)+ –(Ay)
ASL [W] ASR [W] LSL [W] LSR [W] ROL [W] ROR [W] ROXL [W] ROXR	(Ay) by #1 (Ay)+ by #1 –(Ay) by #1

NOTE: [B, W, or L] indicate an operand size of byte, word, or long word.

After recognizing $\overline{VPA}$, the processor assures that enable (E) is low by waiting, if necessary, and subsequently asserts $\overline{VMA}$. $\overline{VMA}$ is then used as part of the chip-select equation of the peripheral to ensure correct timing for selection and deselection of the M6800 device. Once selected, the peripheral runs its cycle during the high portion of the E signal. Figure B-4 shows the best-case timing of an M6800 cycle, and Figure B-5 shows the worst-case timing. The cycle length is entirely dependent on the relationship of the assertion of $\overline{VPA}$ to the E clock.

When external circuitry asserts $\overline{VPA}$ as soon as possible following the assertion of $\overline{AS}$, the assertion of $\overline{VPA}$ is recognized on the falling edge of S4. In this case, no extra wait states are inserted (waiting for the assertion of $\overline{VPA}$). The only wait states inserted are those required to synchronize with the E clock. The synchronization delay is an integral number of system clock cycles within the following extremes:

1. Best Case—the assertion of $\overline{VPA}$ is recognized on the falling edge three clock cycles before E rises (or three clock cycles after E falls).
2. Worst Case—the assertion of $\overline{VPA}$ is recognized on the falling edge two clock cycles before E rises (or four clock cycles after E falls).

The processor latches the peripheral data in state 6 (S6) during a read cycle. For all cycles, the processor negates the address and data strobes one-half clock cycle later in state 7 (S7), and E goes low at this time. Another half clock later, the address bus is placed in the high-impedance state, and R/$\overline{W}$ is driven high. Logic in the peripheral must remove $\overline{VPA}$ within one clock after the negation of address strobe.

Data transfer acknowledge ($\overline{DTACK}$) must not be asserted while VPA is asserted. The state machine in the processor looks for $\overline{DTACK}$ to identify an asynchronous bus cycle and for $\overline{VPA}$ to identify a synchronous peripheral bus cycle. If both signals are asserted, the operation of the state machine is unpredictable.

To allow the processor to place its buses in the high-impedance state during DMA requests without inadvertently selecting the peripherals, $\overline{VMA}$ is active low for the M68000 Family of processors. The active-low $\overline{VMA}$ is in contrast to the active-high VMA signal of the M6800.

B.2 INTERRUPT INTERFACE OPERATION

During an interrupt acknowledge cycle while the processor is fetching the vector, $\overline{VPA}$ is asserted, and the processor (or external circuitry) asserts $\overline{VMA}$ and completes a normal M6800 read cycle as shown in Figure B-6. For the interrupt vector, the processor uses an internally generated vector number called an autovector. The autovector corresponds to the interrupt level being serviced. The seven autovectors are decimal vector numbers 25–31.

B

The autovector operation, which can be used with all peripherals, is similar to the normal interrupt acknowledge cycle. The autovector capability provides vectors for each of the six maskable interrupt levels and for the nonmaskable interrupt level. Whether the device supplies the vector number or the processor generates an autovector number, the