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	8MHz
Co-Processors/DSP	-
RAM Controllers	-
Graphics Acceleration	No
Display & Interface Controllers	-
Ethernet	-
SATA	-
USB	-
Voltage - I/O	5.0V
Operating Temperature	-40°C ~ 85°C (TA)
Security Features	-
Package / Case	68-LCC (J-Lead)
Supplier Device Package	68-PLCC (24.21x24.21)
Purchase URL	https://www.e-xfl.com/product-detail/nxp-semiconductors/mc68hc001cei8

LIST OF ILLUSTRATIONS

LIST OF ILLUSTRATIONS (Concluded)

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

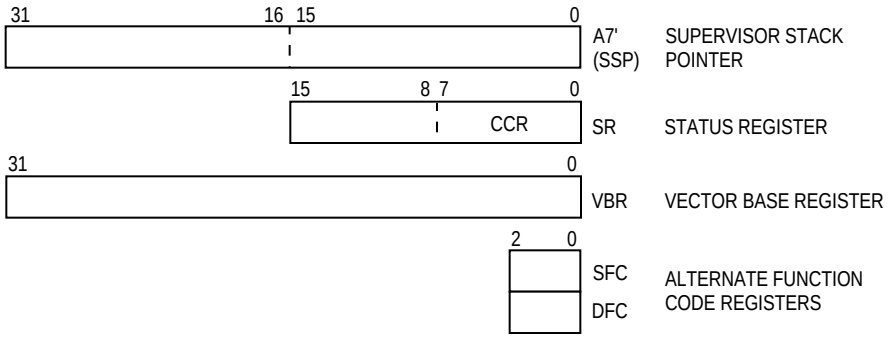


Figure 2-3. Supervisor Programmer's Model Supplement (MC68010)

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

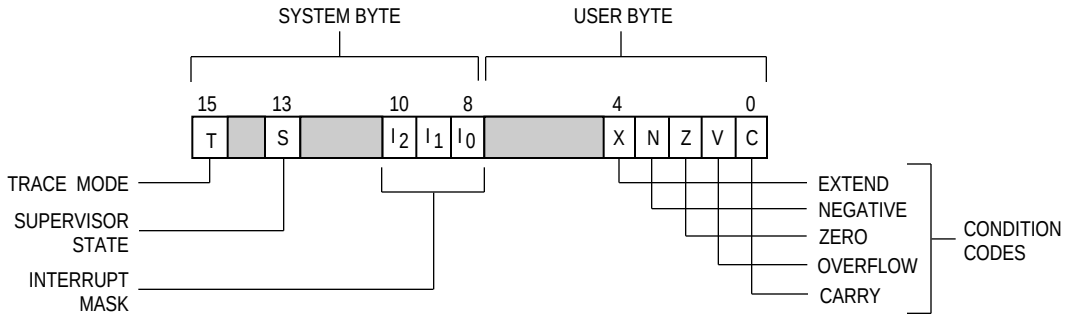


Figure 2-4. Status Register

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)

SECTION 4
8-BIT BUS OPERATION

The following paragraphs describe control signal and bus operation for 8-bit operation during data transfer operations, bus arbitration, bus error and halt conditions, and reset operation. The 8-bit bus operations devices are the MC68008, MC68HC001 in 8-bit mode, and MC68EC000 in 8-bit mode. The MC68HC001 and MC68EC000 select 8-bit mode by grounding mode during reset.

4.1 DATA TRANSFER OPERATIONS

Transfer of data between devices involves the following signals:

1. Address bus A0 through highest numbered address line
2. Data bus D0 through D7
3. 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 must deskew all signals it issues at both the start and end of a bus cycle. In addition, the bus master must deskew the acknowledge and data signals from the slave device. For the MC68HC001 and MC68EC000, $\overline{\text{UDS}}$ is held negated and D15–D8 are undefined in 8-bit mode.

The following paragraphs describe the read, write, read-modify-write, and CPU space cycles. The indivisible read-modify-write cycle implements interlocked multiprocessor communications. A CPU space cycle is a special processor cycle.

4.1.1 Read Cycle

During a read cycle, the processor receives one byte of data from the memory or from a peripheral device. When the data is received, the processor internally positions the byte appropriately.

The 8-bit operation must perform two or four read cycles to access a word or long word, asserting the data strobe to read a single byte during each cycle. The address bus in 8-bit operation includes A0, which selects the appropriate byte for each read cycle. Figure 4-1 and 4-2 illustrate the byte read-cycle operation.

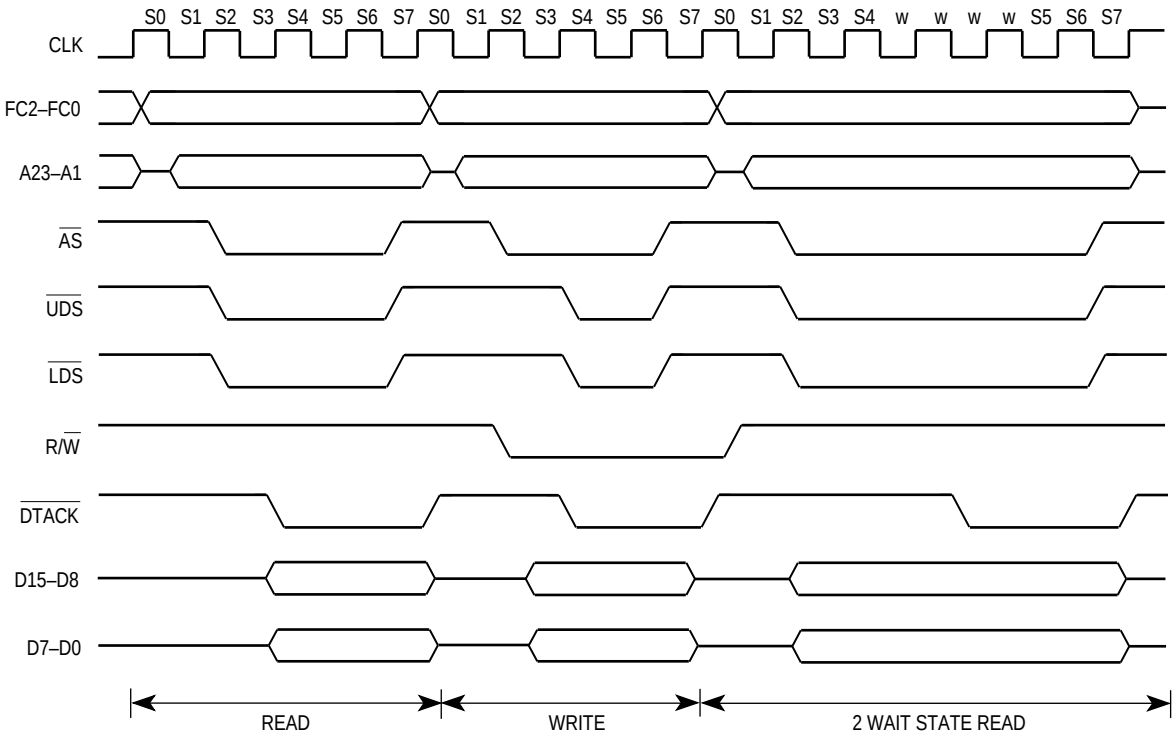


Figure 5-3. Read and Write-Cycle Timing Diagram

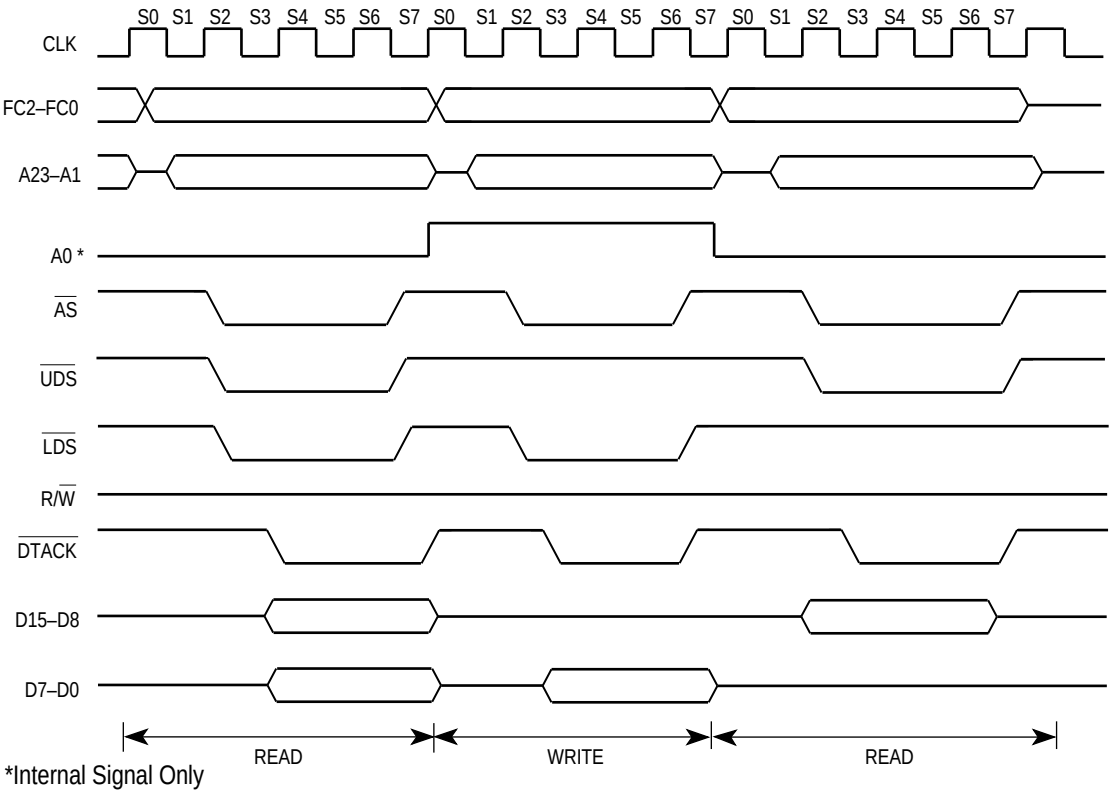


Figure 5-4. Word and Byte Read-Cycle Timing Diagram

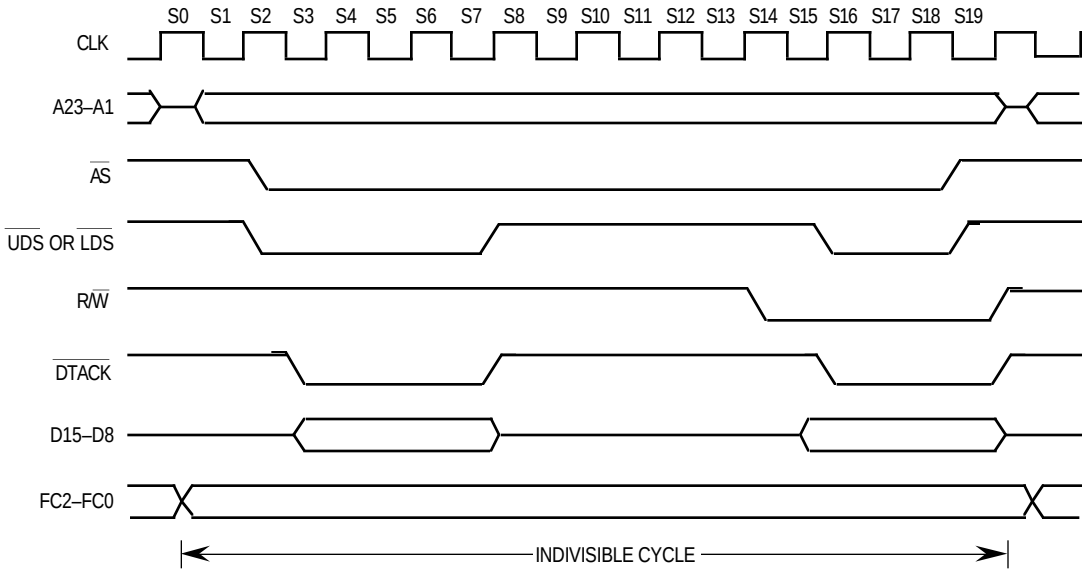


Figure 5-9. Read-Modify-Write Cycle Timing Diagram

The descriptions of the read-modify-write cycle states are as follows:

STATE 0 The read cycle starts in S0. The processor places valid function codes on FC2–FC0 and drives R/$\overline{W}$ high to identify a read cycle.

STATE 1 Entering S1, the processor drives a valid address on the address bus.

STATE 2 On the rising edge of S2, the processor asserts $\overline{AS}$ and $\overline{UDS}$, or $\overline{LDS}$.

STATE 3 During S3, no bus signals are altered.

STATE 4 During S4, the processor waits for a cycle termination signal ($\overline{DTACK}$ or $\overline{BERR}$) or $\overline{VPA}$, an M6800 peripheral signal. When $\overline{VPA}$ is asserted during S4, the cycle becomes a peripheral cycle (refer to **Appendix B M6800 Peripheral Interface**). If neither termination signal is asserted before the falling edge at the end of S4, the processor inserts wait states (full clock cycles) until either $\overline{DTACK}$ or $\overline{BERR}$ is asserted.

STATE 5 During S5, no bus signals are altered.

STATE 6 During S6, data from the device are driven onto the data bus.

STATE 7 On the falling edge of the clock entering S7, the processor accepts data from the device and negates $\overline{UDS}$, and $\overline{LDS}$. The device negates $\overline{DTACK}$ or $\overline{BERR}$ at this time.

STATES 8–11

The bus signals are unaltered during S8–S11, during which the arithmetic logic unit makes appropriate modifications to the data.

supervisor mode. Therefore, when instruction execution resumes at the address specified to process the exception, the processor is in the supervisor privilege mode.

NOTE

The transition from supervisor to user mode can be accomplished by any of four instructions: return from exception (RTE) (MC68010 only), move to status register (MOVE to SR), AND immediate to status register (ANDI to SR), and exclusive OR immediate to status register (EORI to SR). The RTE instruction in the MC68010 fetches the new status register and program counter from the supervisor stack and loads each into its respective register. Next, it begins the instruction fetch at the new program counter address in the privilege mode determined by the S bit of the new contents of the status register.

The MOVE to SR, ANDI to SR, and EORI to SR instructions fetch all operands in the supervisor mode, perform the appropriate update to the status register, and then fetch the next instruction at the next sequential program counter address in the privilege mode determined by the new S bit.

6.1.4 Reference Classification

When the processor makes a reference, it classifies the reference according to the encoding of the three function code output lines. This classification allows external translation of addresses, control of access, and differentiation of special processor states, such as CPU space (used by interrupt acknowledge cycles). Table 6-1 lists the classification of references.

Table 6-1. Reference Classification

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

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

6.4 RETURN FROM EXCEPTION (MC68010)

In addition to returning from any exception handler routine on the MC68010, the RTE instruction resumes the execution of a suspended instruction by returning to the normal processing state after restoring all of the temporary register and control information stored during a bus error. For the RTE instruction to execute properly, the stack must contain valid and accessible data. The RTE instruction checks for data validity in two ways. First, the format/offset word is checked for a valid stack format code. Second, if the format code indicates the long stack format, the validity of the long stack data is checked as it is loaded into the processor. In addition, the data is checked for accessibility when the processor starts reading the long data. Because of these checks, the RTE instruction executes as follows:

1. Determine the stack format. This step is the same for any stack format and consists of reading the status register, program counter, and format/offset word. If the format code indicates a short stack format, execution continues at the new program counter address. If the format code is not an MC68010-defined stack format code, exception processing starts for a format error.
2. Determine data validity. For a long-stack format, the MC68010 begins to read the remaining stack data, checking for validity of the data. The only word checked for validity is the first of the 16 internal information words (SP + 26) shown in Figure 5-8. This word contains a processor version number (in bits 10–13) and proprietary internal information that must match the version number of the MC68010 attempting to read the data. This validity check is used to ensure that the data is properly interpreted by the RTE instruction. If the version number is incorrect for this processor, the RTE instruction is aborted and exception processing begins for a format error exception. Since the stack pointer is not updated until the RTE instruction has successfully read all the stack data, a format error occurring at this point does not stack new data over the previous bus error stack information.
3. Determine data accessibility. If the long-stack data is valid, the MC68010 performs a read from the last word (SP + 56) of the long stack to determine data accessibility. If this read is terminated normally, the processor assumes that the remaining words on the stack frame are also accessible. If a bus error is signaled before or during this read, a bus error exception is taken. After this read, the processor must be able to load the remaining data without receiving a bus error; therefore, if a bus error occurs on any of the remaining stack reads, the error becomes a double bus fault, and the MC68010 enters the halted state.

Table 7-7. Single Operand Instruction Execution Times

Instruction	Size	Register	Memory
CLR	Byte Word Long	**8**(2/0) **8**(2/0) **10**(2/0)	**12**(2/1)+ **16**(2/2)+ **24**(2/4)+
NBCD	Byte	**10**(2/0)	**12**(2/1)+
NEG	Byte Word Long	**8**(2/0) **8**(2/0) **10**(2/0)	**12**(2/1)+ **16**(2/2)+ **24**(2/4)+
NEGX	Byte Word Long	**8**(2/0) **8**(2/0) **10**(2/0)	**12**(2/1)+ **16**(2/2)+ **24**(2/4)+
NOT	Byte Word Long	**8**(2/0) **8**(2/0) **10**(2/0)	**12**(2/1)+ **16**(2/2)+ **24**(2/4)+
Scc	Byte, False Byte, True	**8**(2/0) **10**(2/0)	**12**(2/1)+ **12**(2/1)+
TAS	Byte	**8**(2/0)	**14**(2/1)+
TST	Byte Word Long	**8**(2/0) **8**(2/0) **8**(2/0)	**8**(2/0)+ **8**(2/0)+ **8**(2/0)+

+Add effective address calculation time.

7.6 SHIFT/ROTATE INSTRUCTION EXECUTION TIMES

Table 7-8 lists the timing data for the shift and rotate 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 7-8. Shift/Rotate Instruction Execution Times

Instruction	Size	Register	Memory
ASR, ASL	Byte Word Long	**10+2n**(2/0) **10+2n**(2/0) **12+n2**(2/0)	— **16**(2/2)+ —
LSR, LSL	Byte Word Long	**10+2n**(2/0) **10+2n**(2/0) **12+n2**(2/0)	— **16**(2/2)+ —
ROR, ROL	Byte Word Long	**10+2n**(2/0) **10+2n**(2/0) **12+n2**(2/0)	— **16**(2/2)+ —
ROXR, ROXL	Byte Word Long	**10+2n**(2/0) **10+2n**(2/0) **12+n2**(2/0)	— **16**(2/2)+ —

+Add effective address calculation time for word operands.
n is the shift count.

7.9 JMP, JSR, LEA, PEA, AND MOVEM INSTRUCTION EXECUTION TIMES

Table 7-11 lists the timing data for the jump (JMP), jump to subroutine (JSR), load effective address (LEA), push effective address (PEA), and move multiple registers (MOVEM) 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.

Table 7-11. JMP, JSR, LEA, PEA, and MOVEM Instruction Execution Times

Instruction	Size	(An)	(An)+	–(An)	(d_{16},An)	(d_8,An,Xn)+	(xxx).W	(xxx).L	(d_{16} PC)	(d_8, PC, Xn)*
JMP	—	**16** (4/0)	—	—	**18** (4/0)	**22** (4/0)	**18** (4/0)	**24** (6/0)	**18** (4/0)	**22** (4/0)
JSR	—	**32** (4/4)	—	—	**34** (4/4)	**38** (4/4)	**34** (4/4)	**40** (6/4)	**34** (4/4)	**32** (4/4)
LEA	—	**8**(2/0)	—	—	**16** (4/0)	**20** (4/0)	**16** (4/0)	**24** (6/0)	**16** (4/0)	**20** (4/0)
PEA	—	**24** (2/4)	—	—	**32** (4/4)	**36** (4/4)	**32** (4/4)	**40** (6/4)	**32** (4/4)	**36** (4/4)
MOVEM M → R	Word	**24+8n** (6+2n/0)	**24+8n** (6+2n/0)	—	**32+8n** (8+2n/0)	**34+8n** (8+2n/0)	**32+8n** (10+n/0)	**40+8n** (10+2n/0)	**32+8n** (8+2n/0)	**34+8n** (8+2n/0)
	Long	**24+16n** (6+4n/0)	**24+16n** (6+4n/0)	—	**32+16n** (8+4n/0)	**34+16n** (8+4n/0)	**32+16n** (8+4n/0)	**40+16n** (8+4n/0)	**32+16n** (8+4n/0)	**34+16n** (8+4n/0)
MOVEM R → M	Word	**16+8n** (4/2n)	— —	**16+8n** (4/2n)	**24+8n** (6/2n)	**26+8n** (6/2n)	**24+8n** (6/2n)	**32+8n** (8/2n)	— —	— —
	Long	**16+16n** (4/4n)	— —	**16+16n** (4/4n)	**24+16n** (6/4n)	**26+16n**	**24+16n** (8/4n)	**32+16n** (6/4n)	— —	— —

n is the number of registers to move.
*The size of the index register (Xn) does not affect the instruction's execution time.

7.10 MULTIPRECISION INSTRUCTION EXECUTION TIMES

Table 7-12 lists the timing data for multiprecision instructions. The numbers of clock periods include the times to fetch both operands, perform the operations, store the results, and read the next 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 following notation applies in Table 7-12:

Dn — Data register operand

M — Memory operand

Table 8-4. Standard Instruction Execution Times

Instruction	Size	op<ea>, An†	op<ea>, Dn	op Dn, <M>
ADD/ADDA	Byte, Word	**8**(1/0)+	**4**(1/0)+	**8**(1/1)+
	Long	**6**(1/0)+**	**6**(1/0)+**	**12**(1/2)+
AND	Byte, Word	—	**4**(1/0)+	**8**(1/1)+
	Long	—	**6**(1/0)+**	**12**(1/2)+
CMP/CMPA	Byte, Word	**6**(1/0)+	**4**(1/0)+	—
	Long	**6**(1/0)+	**6**(1/0)+	—
DIVS	—	—	**158**(1/0)+*	—
DIVU	—	—	**140**(1/0)+*	—
EOR	Byte, Word	—	**4**(1/0)***	**8**(1/1)+
	Long	—	**8**(1/0)***	**12**(1/2)+
MULS	—	—	**70**(1/0)+*	—
MULU	—	—	**70**(1/0)+*	—
OR	Byte, Word	—	**4**(1/0)+	**8**(1/1)+
	Long	—	**6**(1/0)+**	**12**(1/2)+
SUB	Byte, Word	**8**(1/0)+	**4**(1/0)+	**8**(1/1)+
	Long	**6**(1/0)+**	**6**(1/0)+**	**12**(1/2)+

\+ Add effective address calculation time.
† Word or long only
* Indicates maximum basic value added to word effective address time
** The base time of six clock periods is increased to eight 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 MC68000 provides less than 10% difference between the best- and worst-case timings.
MULS, MULU — The multiply algorithm requires 38+2n clocks where n is defined as:
MULU: n = the number of ones in the <ea>
MULS: n=concatenate the <ea> with a zero as the LSB; 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.

8.4 IMMEDIATE INSTRUCTION EXECUTION TIMES

The numbers of clock periods shown in Table 8-5 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 8-5, the following notation applies:

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

the handler routine. 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 8-14. Exception Processing Execution Times

Exception	Periods
Address Error	**50**(4/7)
Bus Error	**50**(4/7)
CHK Instruction	**40**(4/3)+
Divide by Zero	**38**(4/3)+
Illegal Instruction	**34**(4/3)
Interrupt	**44**(5/3)*
Privilege Violation	**34**(4/3)
$\overline{\text{RESET}}$**	**40**(6/0)
Trace	**34**(4/3)
TRAP Instruction	**34**(4/3)
TRAPV Instruction	**34**(5/3)

\+ Add effective address calculation time.
* The interrupt acknowledge cycle is assumed to take four clock periods.
** Indicates the time from when $\overline{\text{RESET}}$ and $\overline{\text{HALT}}$ are first sampled as negated to when instruction execution starts.

Table 9-4. Move Long Instruction Execution Times

Source	Destination								
	Dn	An	(An)	(An)+	–(An)	(d_{16}, An)	(d_8, An, Xn)*	(xxx).W	(xxx).L
Dn	**4**(1/0)	**4**(1/0)	**12**(1/2)	**12**(1/2)	**14**(1/2)	**16**(2/2)	**18**(2/2)	**16**(2/2)	**20**(3/2)
An	**4**(1/0)	**4**(1/0)	**12**(1/2)	**12**(1/2)	**14**(1/2)	**16**(2/2)	**18**(2/2)	**16**(2/2)	**20**(3/2)
(An)	**12**(3/0)	**12**(3/0)	**20**(3/2)	**20**(3/2)	**20**(3/2)	**24**(4/2)	**26**(4/2)	**24**(4/2)	**28**(5/2)
(An)+	**12**(3/0)	**12**(3/0)	**20**(3/2)	**20**(3/2)	**20**(3/2)	**24**(4/2)	**26**(4/2)	**24**(4/2)	**28**(5/2)
–(An)	**14**(3/0)	**14**(3/0)	**22**(3/2)	**22**(3/2)	**22**(3/2)	**26**(4/2)	**28**(4/2)	**26**(4/2)	**30**(5/2)
(d_{16}, An)	**16**(4/0)	**16**(4/0)	**24**(4/2)	**24**(4/2)	**24**(4/2)	**28**(5/2)	**30**(5/2)	**28**(5/2)	**32**(6/2)
(d_8, An, Xn)*	**18**(4/0)	**18**(4/0)	**26**(4/2)	**26**(4/2)	**26**(4/2)	**30**(5/2)	**32**(5/2)	**30**(5/2)	**34**(6/2)
(xxx).W	**16**(4/0)	**16**(4/0)	**24**(4/2)	**24**(4/2)	**24**(4/2)	**28**(5/2)	**30**(5/2)	**28**(5/2)	**32**(6/2)
(xxx).L	**20**(5/0)	**20**(5/0)	**28**(5/2)	**28**(5/2)	**28**(5/2)	**32**(6/2)	**34**(6/2)	**32**(6/2)	**36**(7/2)
(d_{16}, PC)	**16**(4/0)	**16**(4/0)	**24**(4/2)	**24**(4/2)	**24**(4/2)	**28**(5/2)	**30**(5/2)	**28**(5/2)	**32**(5/2)
(d_8, PC, Xn)*	**18**(4/0)	**18**(4/0)	**26**(4/2)	**26**(4/2)	**26**(4/2)	**30**(5/2)	**32**(5/2)	**30**(5/2)	**34**(6/2)
#<data>	**12**(3/0)	**12**(3/0)	**20**(3/2)	**20**(3/2)	**20**(3/2)	**24**(4/2)	**26**(4/2)	**24**(4/2)	**28**(5/2)

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

Table 9-5. Move Long Instruction Loop Mode Execution Times

Source	Loop Continued			Loop Terminated					
	Valid Count, cc False			Valid count, cc True			Expired Count		
	Destination								
	(An)	(An)+	–(An)	(An)	(An)+	–(An)	(An)	(An)+	–(An)
Dn	**14**(0/2)	**14**(0/2)	—	**20**(2/2)	**20**(2/2)	—	**18**(2/2)	**18**(2/2)	—
An	**14**(0/2)	**14**(0/2)	—	**20**(2/2)	**20**(2/2)	—	**18**(2/2)	**18**(2/2)	—
(An)	**22**(2/2)	**22**(2/2)	**24**(2/2)	**28**(4/2)	**28**(4/2)	**30**(4/2)	**24**(4/2)	**24**(4/2)	**26**(4/2)
(An)+	**22**(2/2)	**22**(2/2)	**24**(2/2)	**28**(4/2)	**28**(4/2)	**30**(4/2)	**24**(4/2)	**24**(4/2)	**26**(4/2)
–(An)	**24**(2/2)	**24**(2/2)	**26**(2/2)	**30**(4/2)	**30**(4/2)	**32**(4/2)	**26**(4/2)	**26**(4/2)	**28**(4/2)

9.3 STANDARD INSTRUCTION EXECUTION TIMES

The numbers of clock periods shown in tables 9-6 and 9-7 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 Tables 9-6 and 9-7, the following notation applies:

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

and read the next 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 following notation applies in Table 9-17:

Dn — Data register operand
M — Memory operand

Table 9-17. Multiprecision Instruction Execution Times

		Nonlooped		Loop Mode		
				Continued	Terminated	
				Valid Count, cc False	Valid Count, cc True	Expired Count
Instruction	**Size**	**op Dn, Dn**	**op M, M***			
ADDX	Byte, Word	**4**(1/0)	**18**(3/1)	**22**(2/1)	**28**(4/1)	**26**(4/1)
	Long	**6**(1/0)	**30**(5/2)	**32**(4/2)	**38**(6/2)	**36**(6/2)
CMPM	Byte, Word	—	**12**(3/0)	**14**(2/0)	**20**(4/0)	**18**(4/0)
	Long	—	**20**(5/0)	**24**(4/0)	**30**(6/0)	**26**(6/0)
SUBX	Byte, Word	**4**(1/)	**18**(3/1)	**22**(2/1)	**28**(4/1)	**26**(4/1)
	Long	**6**(1/0)	**30**(5/2)	**32**(4/2)	**38**(6/2)	**36**(6/2)
ABCD	Byte	**6**(1/0)	**18**(3/1)	**24**(2/1)	**30**(4/1)	**28**(4/1)
SBCD	Byte	**6**(1/0)	**18**(3/1)	**24**(2/1)	**30**(4/1)	**28**(4/1)

*Source and destination ea are (An)+ for CMPM and –(An) for all others.

9.11 MISCELLANEOUS INSTRUCTION EXECUTION TIMES

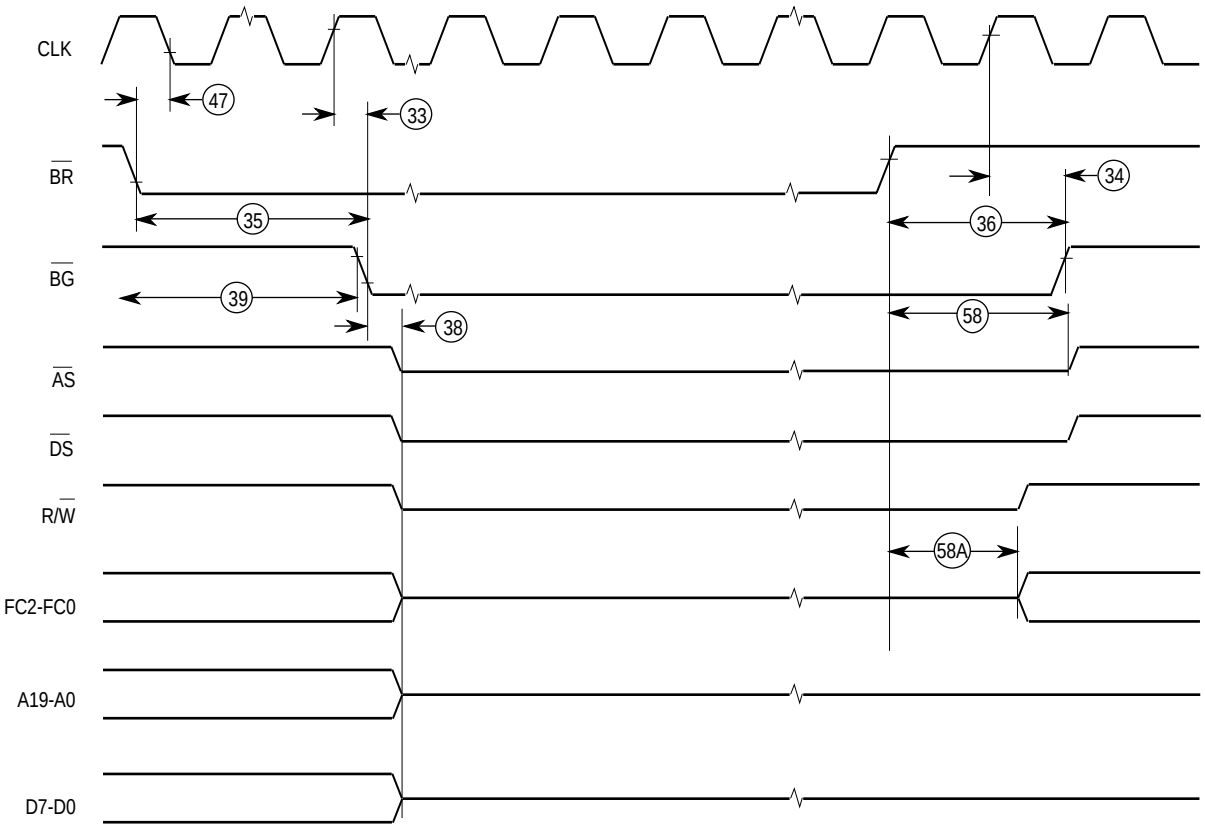
Table 9-18 lists the timing data for miscellaneous instructions. The total number of clock periods, the number of read cycles, and the number of write cycles are shown in the previously described format. The number of clock periods, the number of read cycles, and the number of write cycles, respectively, must be added to those of the effective address calculation where indicated by a plus sign (+).

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: Waveform measurements for all inputs and outputs are specified at: logic high 2.0 V, logic low = 0.8 V.

Figure 10-14. MC68EC000 Bus Arbitration Timing Diagram

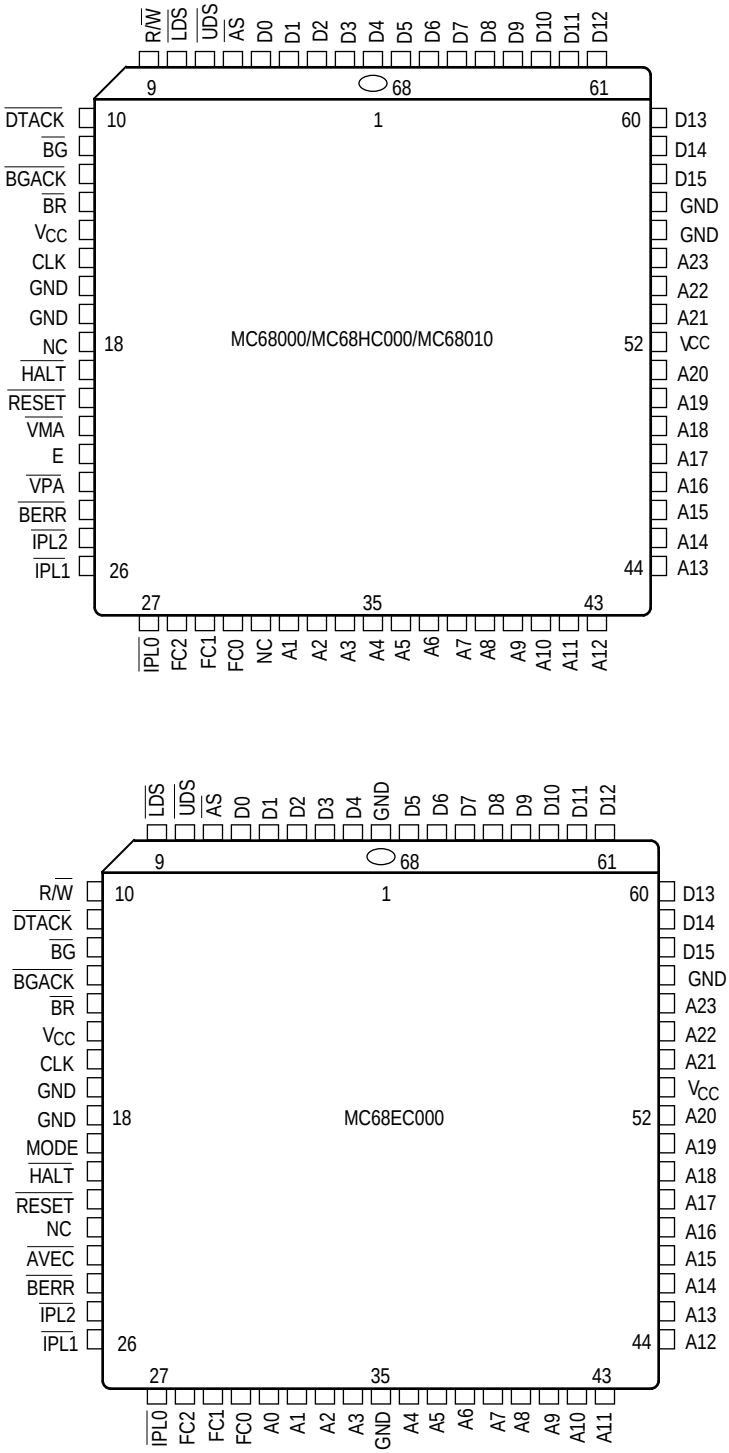
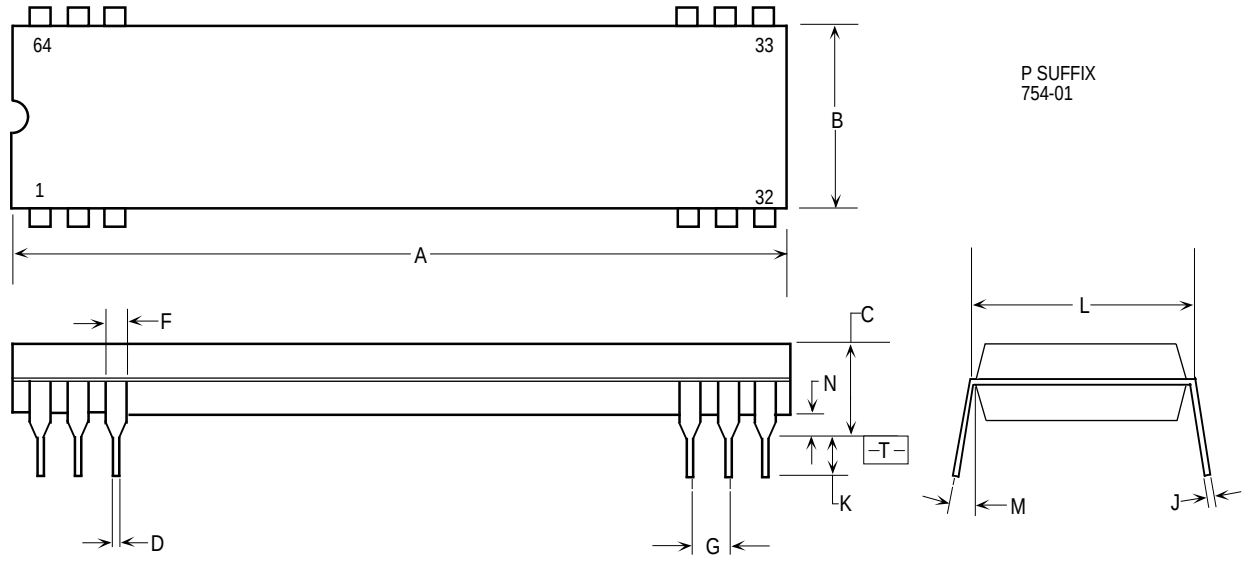


Figure 11-3. 68-Lead Quad Pack (1 of 2)



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 B DOES NOT INCLUDEMOLD FLASH.
5. DIMENSION L IS TO CENTER OF LEADS WHEN FORMED PARALLEL.
6. DIMENSIONING AND TOLERANCING PER ANSI Y14.5, 1982.

	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		0.9 00 BSC	
M	0°	15°	0°	15°
N	0.51	1.02	0.020	0.040

Figure 11-10. Case 754-01—R and P Suffix

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.

processors. Enable has a 60/40 duty cycle; that is, it is low for six system clocks and high for four system clocks. This duty cycle allows $\overline{VPA}$ accesses on successive E pulses.

In the MC68000, MC68HC000, MC68HC001, and the MC68010, $\overline{VMA}$ is provided to indicate synchronization with E. The MC68008 does not provide a $\overline{VMA}$ signal; external circuitry similar to that shown in Figure B-2 using transistor-to-transistor (TTL) logic must be included in the system to provide $\overline{VMA}$. The $\overline{VMA}$ signal indicates to the M6800 devices that the address on the address bus is a valid device address and that the processor is synchronized to the enable clock. The VPA decode input is an active-high signal that is asserted when address strobe $\overline{AS}$ has been asserted and the address on the address bus is that of a peripheral device. The flip-flop on the left sets at the falling edge of E; the flip-flop on the right sets at the next fall of system clock, asserting $\overline{VMA}$. $\overline{VMA}$ remains asserted until the fall of system clock immediately following the negation of VPA decode. Figure B-3 shows the timing for the $\overline{VMA}$ signal provided by this circuitry.

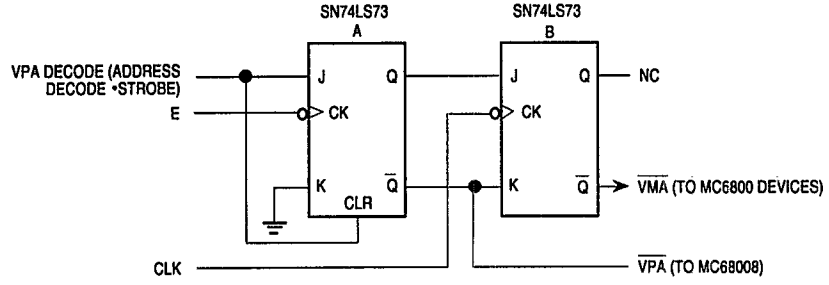


Figure B-2. Example External $\overline{VMA}$ Circuit

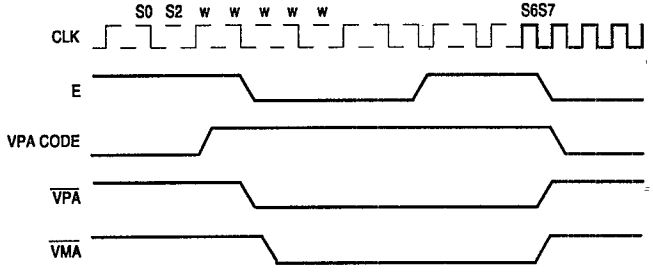


Figure B-3 External $\overline{VMA}$ Timing

B

M6800 cycle timing is shown in Figures B-4 and B-5. At state 0 (S0) in the cycle, the address bus is in the high-impedance state. A function code is asserted on the function code output lines. In state 1 (S1), the address is placed on the address bus. During state 2 (S2), the address strobe ($\overline{AS}$) is asserted to indicate that the address on the bus is valid. If the bus cycle is a read cycle, the upper and/or lower data strobe ($\overline{UDS}$, $\overline{LDS}$) (MC68000/MC68HC000/MC68HC001/MC68010) or data strobe ($\overline{DS}$) (MC68008) is also