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	-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/mc68hc000cei10

LIST OF TABLES

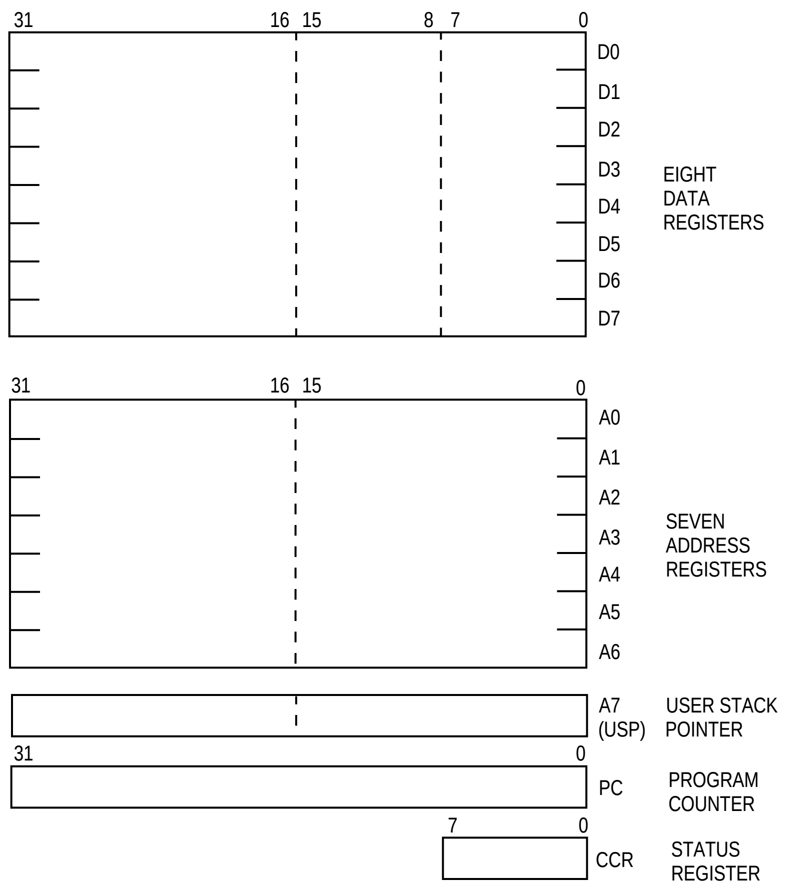


Figure 2-1. User Programmer's Model (MC68000/MC68HC000/MC68008/MC68010)

2.1.2 Supervisor Programmer's Model

The supervisor programmer's model consists of supplementary registers used in the supervisor mode. The M68000 MPUs contain identical supervisor mode register resources, which are shown in Figure 2-2, including the status register (high-order byte) and the supervisor stack pointer (SSP/A7').

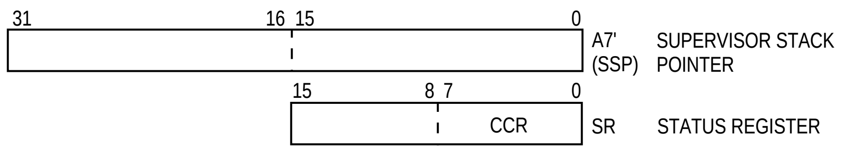


Figure 2-2. Supervisor Programmer's Model Supplement

The supervisor programmer's model supplement of the MC68010 is shown in Figure 2-3. In addition to the supervisor stack pointer and status register, it includes the vector base register (VRB) and the alternate function code registers (AFC).The VBR is used to determine the location of the exception vector table in memory to support multiple vector tables. The SFC and DFC registers allow the supervisor to access user data space or emulate CPU space cycles.

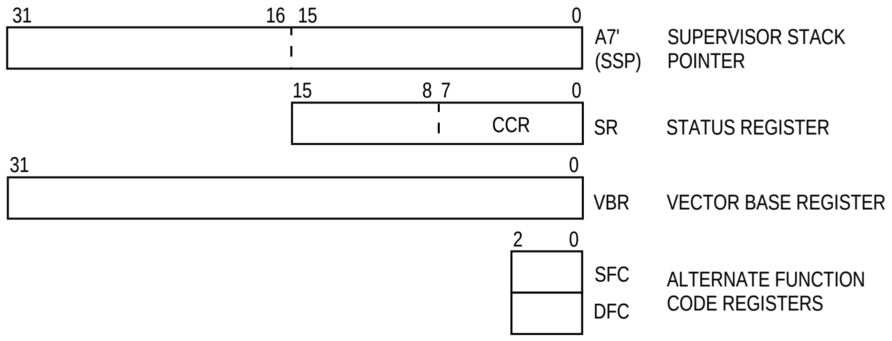


Figure 2-3. Supervisor Programmer's Model Supplement (MC68010)

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

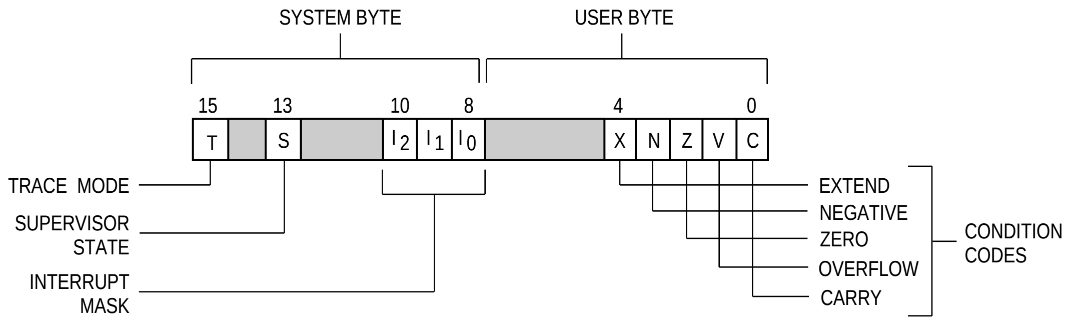


Figure 2-4. Status Register

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)

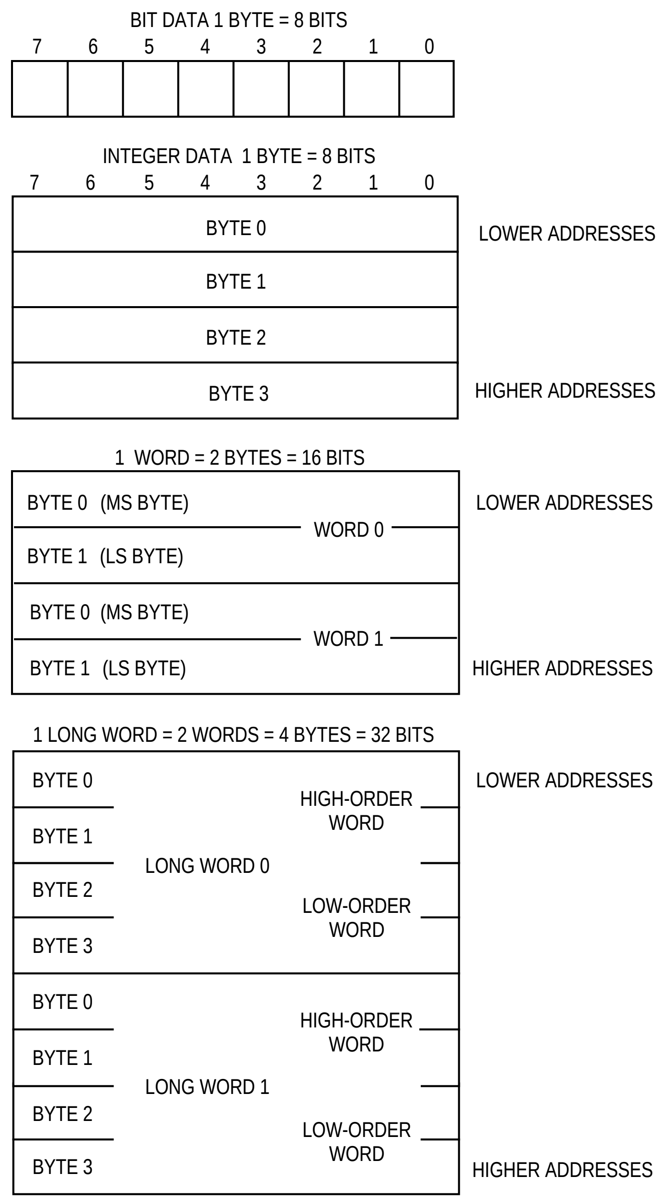


Figure 2-7. Memory Data Organization of the MC68008

2.5 INSTRUCTION SET SUMMARY

Table 2-2 provides an alphabetized listing of the M68000 instruction set listed by opcode, operation, and syntax. In the syntax descriptions, the left operand is the source operand, and the right operand is the destination operand. The following list contains the notations used in Table 2-2.

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

Opcode	Operation	Syntax
DIVS	Destination/Source $\rightarrow$ Destination	DIVS.W <ea>,Dn 32/16 $\rightarrow$ 16r:16q
DIVU	Destination/Source $\rightarrow$ Destination	DIVU.W <ea>,Dn 32/16 $\rightarrow$ 16r:16q
EOR	Source $\oplus$ Destination $\rightarrow$ Destination	EOR Dn,<ea>
EORI	Immediate Data $\oplus$ Destination $\rightarrow$ Destination	EORI # <data>,<ea>
EORI to CCR	Source $\oplus$ CCR $\rightarrow$ CCR	EORI # <data>,CCR
EORI to SR	If supervisor state then Source $\oplus$SR $\rightarrow$ SR else TRAP	EORI # <data>,SR
EXG	Rx $\leftrightarrow$ Ry	EXG Dx,Dy EXG Ax,Ay EXG Dx,Ay EXG Ay,Dx
EXT	Destination Sign-Extended $\rightarrow$ Destination	EXT.W Dn extend byte to word EXT.L Dn extend word to long word

ILLEGAL	SSP – 2 $\rightarrow$ SSP; Vector Offset $\rightarrow$ (SSP);
	SSP – 4 $\rightarrow$ SSP; PC $\rightarrow$ (SSP);
	SSP – 2 $\rightarrow$ SSP; SR $\rightarrow$ (SSP);
	Illegal Instruction Vector Address $\rightarrow$ PC
		ILLEGAL

JMP	Destination Address $\rightarrow$ PC	JMP <ea>
JSR	SP – 4 $\rightarrow$ SP; PC $\rightarrow$ (SP) Destination Address $\rightarrow$ PC	JSR <ea>
LEA	<ea> $\rightarrow$ An	LEA <ea>,An
LINK	SP – 4 $\rightarrow$ SP; An $\rightarrow$ (SP) SP $\rightarrow$ An, SP + d $\rightarrow$ SP	LINK An, # <displacement>
LSL,LSR	Destination Shifted by <count> $\rightarrow$ Destination	LSd[1] Dx,Dy LSd[1] # <data>,Dy LSd[1] <ea>
MOVE	Source $\rightarrow$ Destination	MOVE <ea>,<ea>
MOVEA	Source $\rightarrow$ Destination	MOVEA <ea>,An
MOVE from CCR	CCR $\rightarrow$ Destination	MOVE CCR,<ea>
MOVE to CCR	Source $\rightarrow$ CCR	MOVE <ea>,CCR
MOVE from SR	SR $\rightarrow$ Destination If supervisor state then SR $\rightarrow$ Destination else TRAP (MC68010 only)	MOVE SR,<ea>
MOVE to SR	If supervisor state then Source $\rightarrow$ SR else TRAP	MOVE <ea>,SR

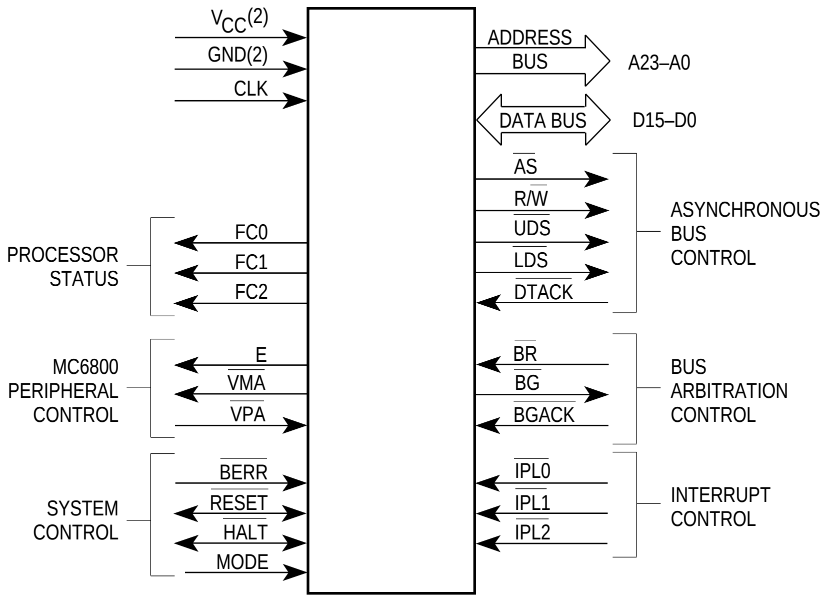


Figure 3-2. Input and Output Signals (MC68HC001)

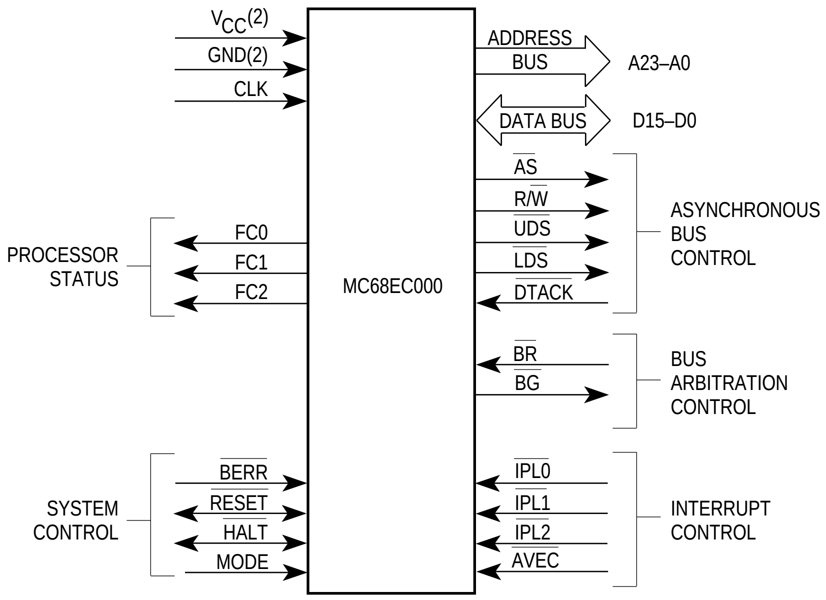


Figure 3-3. Input and Output Signals (MC68EC000)

Table 3-3. Function Code Outputs

Function Code Output			Address Space 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	CPU Space

3.9 CLOCK (CLK)

The clock input is a TTL-compatible signal that is internally buffered for development of the internal clocks needed by the processor. This clock signal is a constant frequency square wave that requires no stretching or shaping. The clock input should not be gated off at any time, and the clock signal must conform to minimum and maximum pulse-width times listed in **Section 10 Electrical Characteristics**.

3.10 POWER SUPPLY (V_{CC} and GND)

Power is supplied to the processor using these connections. The positive output of the power supply is connected to the V_{CC} pins and ground is connected to the GND pins.

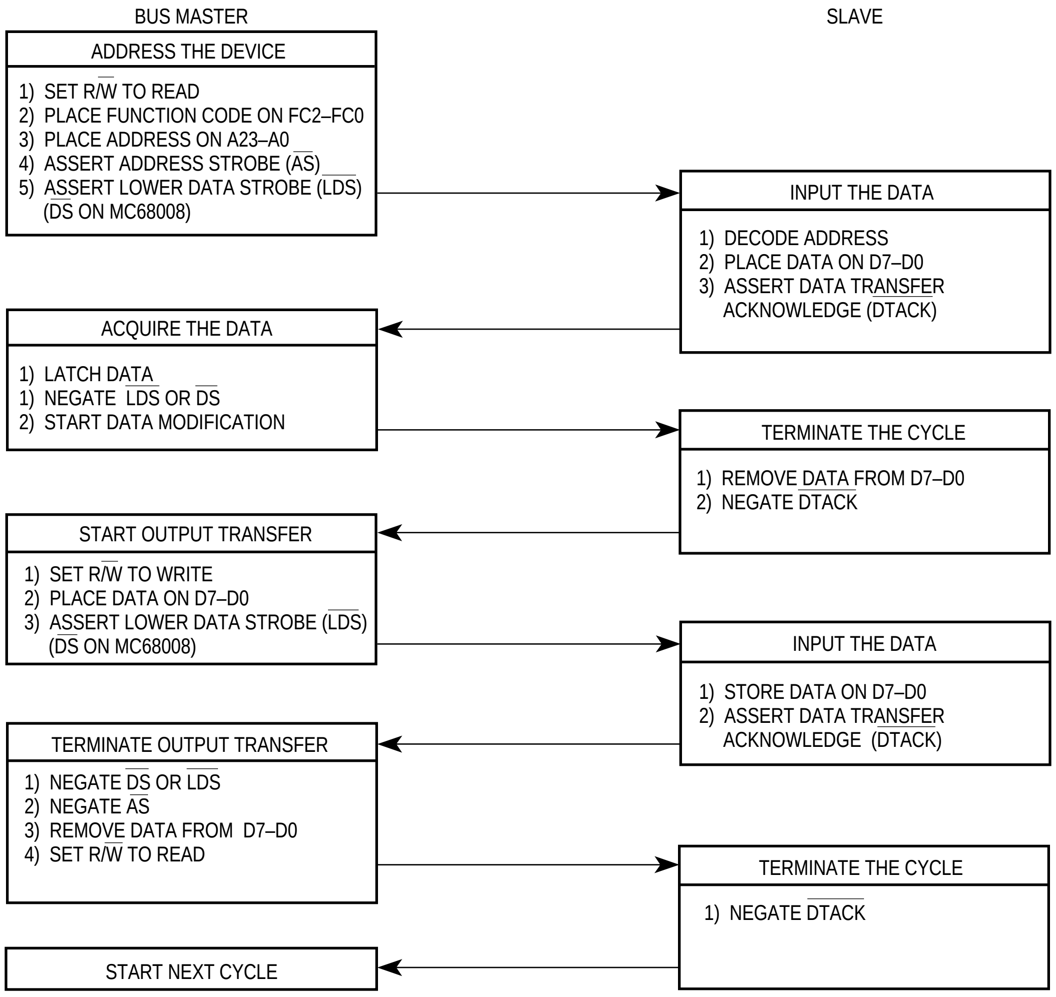


Figure 4-5. Read-Modify-Write Cycle Flowchart

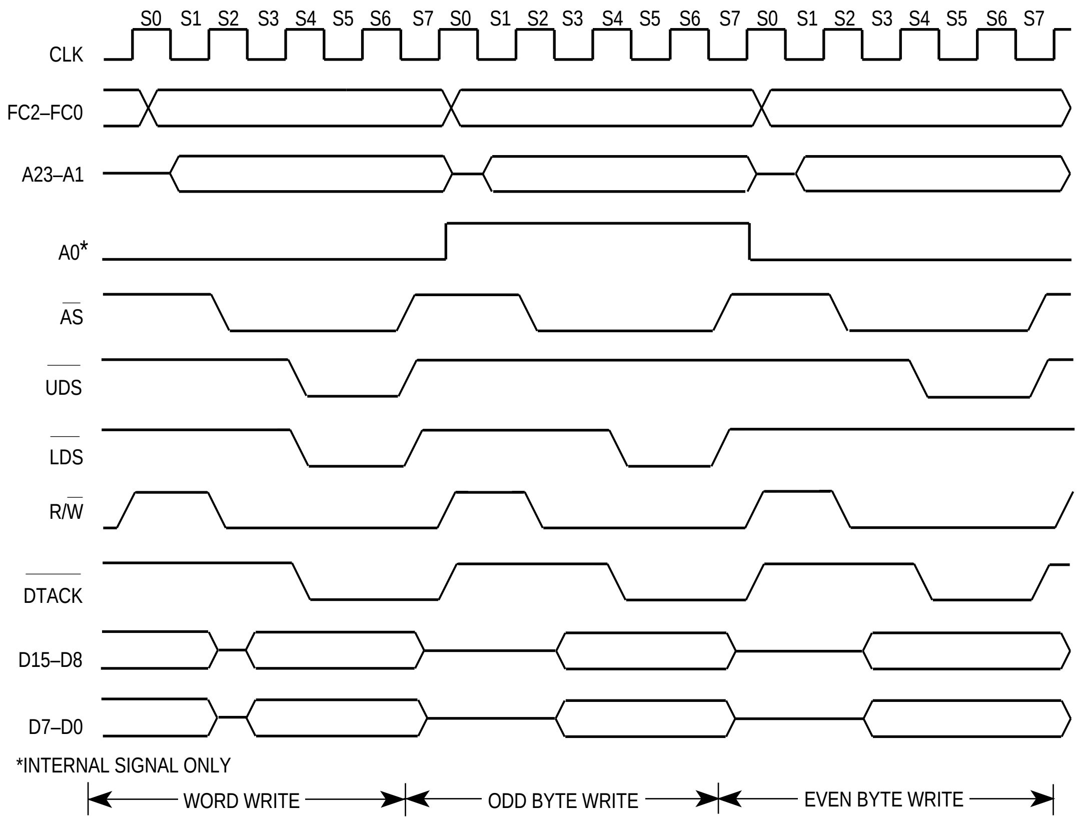


Figure 5-7. Word and Byte Write-Cycle Timing Diagram

The descriptions of the eight states of a write cycle are as follows:

STATE 0	The write cycle starts in S0. The processor places valid function codes on FC2–FC0 and drives R/$\overline{\text{W}}$ high (if a preceding write cycle has left R/$\overline{\text{W}}$ low).
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 $\overline{\text{AS}}$ and drives R/$\overline{\text{W}}$ low.
STATE 3	During S3, the data bus is driven out of the high-impedance state as the data to be written is placed on the bus.
STATE 4	At the rising edge of S4, the processor asserts $\overline{\text{UDS}}$, or $\overline{\text{LDS}}$. The processor waits for a cycle termination signal ($\overline{\text{DTACK}}$ or $\overline{\text{BERR}}$) or $\overline{\text{VPA}}$, an M6800 peripheral signal. When $\overline{\text{VPA}}$ 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 $\overline{\text{DTACK}}$ or $\overline{\text{BERR}}$ is asserted.
STATE 5	During S5, no bus signals are altered.
STATE 6	During S6, no bus signals are altered.

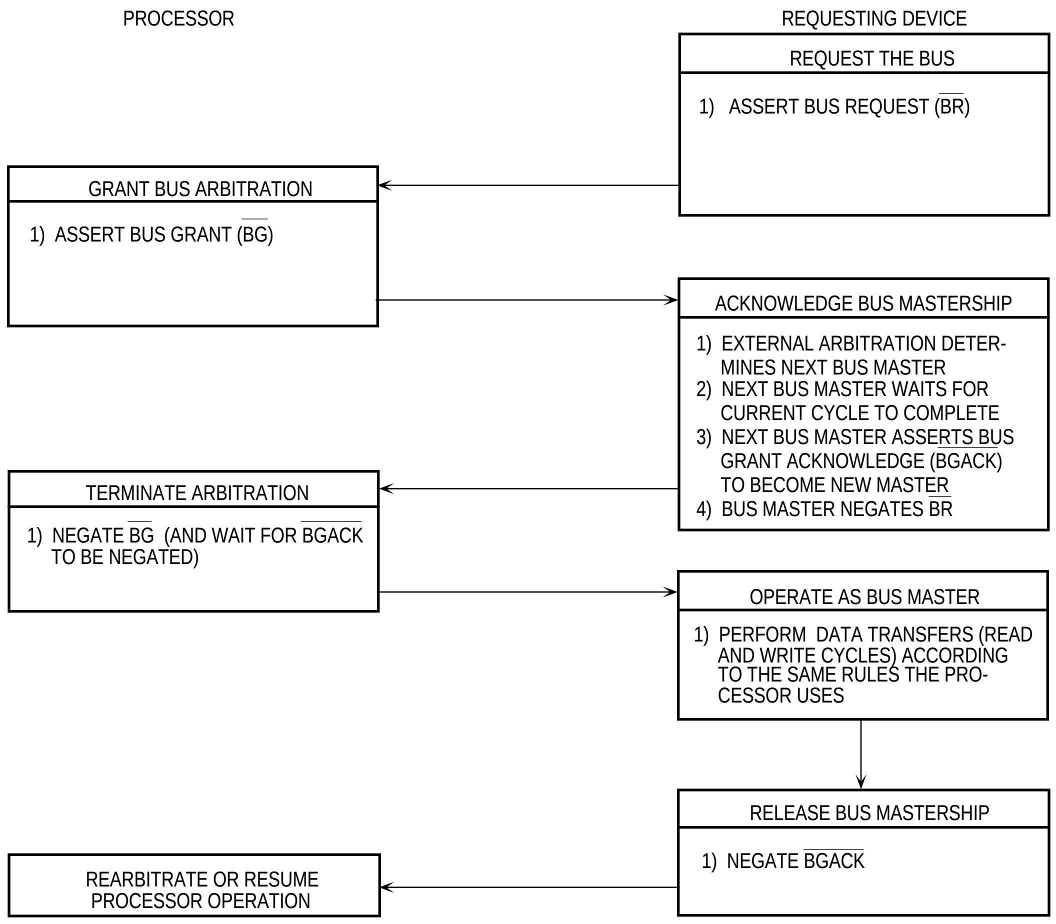

PROCESSOR
REQUESTING DEVICE
REQUEST THE BUS
1) ASSERT BUS REQUEST (BR)
GRANT BUS ARBITRATION
1) ASSERT BUS GRANT (BG)
ACKNOWLEDGE BUS MASTERSHIP
1) EXTERNAL ARBITRATION DETERMINES NEXT BUS MASTER
2) NEXT BUS MASTER WAITS FOR CURRENT CYCLE TO COMPLETE
3) NEXT BUS MASTER ASSERTS BUS GRANT ACKNOWLEDGE (BGACK) TO BECOME NEW MASTER
4) BUS MASTER NEGATES BR
TERMINATE ARBITRATION
1) NEGATE BG (AND WAIT FOR BGACK TO BE NEGATED)
OPERATE AS BUS MASTER
1) PERFORM DATA TRANSFERS (READ AND WRITE CYCLES) ACCORDING TO THE SAME RULES THE PROCESSOR USES
RELEASE BUS MASTERSHIP
1) NEGATE BGACK
REARBITRATE OR RESUME PROCESSOR OPERATION

Figure 5-13. 3-Wire Bus Arbitration Cycle Flowchart (Not Applicable to 48-Pin MC68008 or MC68EC000)

bus request signal. When no acknowledge is received before the bus request signal is negated, the processor continues the use of the bus.

5.2.2 Receiving The Bus Grant

The processor asserts $\overline{BG}$ as soon as possible. Normally, this process immediately follows internal synchronization, except when the processor has made an internal decision to execute the next bus cycle but has not yet asserted $\overline{AS}$ for that cycle. In this case, $\overline{BG}$ is delayed until $\overline{AS}$ is asserted to indicate to external devices that a bus cycle is in progress.

$\overline{BG}$ can be routed through a daisy-chained network or through a specific priority-encoded network. Any method of external arbitration that observes the protocol can be used.

5.2.3 Acknowledgment Of Mastership (3-Wire Bus Arbitration Only)

Upon receiving $\overline{BG}$, the requesting device waits until $\overline{AS}$, $\overline{DTACK}$, and $\overline{BGACK}$ are negated before asserting $\overline{BGACK}$. The negation of $\overline{AS}$ indicates that the previous bus master has completed its cycle. (No device is allowed to assume bus mastership while $\overline{AS}$ is asserted.) The negation of $\overline{BGACK}$ indicates that the previous master has released the bus. The negation of $\overline{DTACK}$ indicates that the previous slave has terminated the connection to the previous master. (In some applications, $\overline{DTACK}$ might not be included in this function; general-purpose devices would be connected using $\overline{AS}$ only.) When $\overline{BGACK}$ is asserted, the asserting device is bus master until it negates $\overline{BGACK}$. $\overline{BGACK}$ should not be negated until after the bus cycle(s) is complete. A device relinquishes control of the bus by negating $\overline{BGACK}$.

The bus request from the granted device should be negated after $\overline{BGACK}$ is asserted. If another bus request is pending, $\overline{BG}$ is reasserted within a few clocks, as described in **5.3 Bus Arbitration Control**. The processor does not perform any external bus cycles before reasserting $\overline{BG}$.

5.3 BUS ARBITRATION CONTROL

All asynchronous bus arbitration signals to the processor are synchronized before being used internally. As shown in Figure 5-17, synchronization requires a maximum of one cycle of the system clock, assuming that the asynchronous input setup time (#47, defined in **Section 10 Electrical Characteristic**) has been met. The input asynchronous signal is sampled on the falling edge of the clock and is valid internally after the next falling edge.

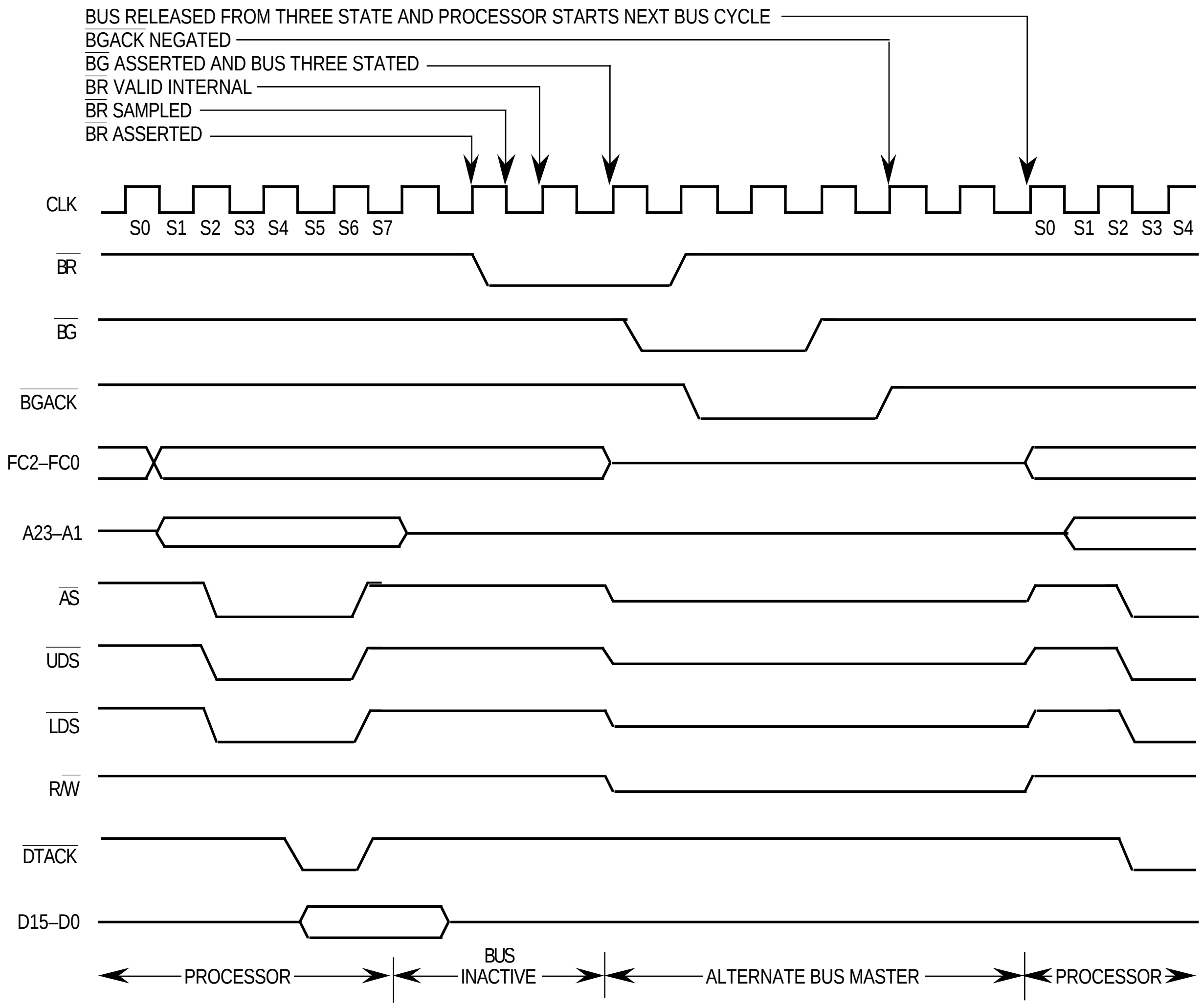


Figure 5-20. 3-Wire Bus Arbitration Timing Diagram—Bus Inactive

4. For an MC68010, return $\overline{\text{DTACK}}$ before data verification. If data is invalid, assert $\overline{\text{BERR}}$ on the next clock cycle (case 4).

Table 5-6. $\overline{\text{BERR}}$ and $\overline{\text{HALT}}$ Negation Results

Conditions of Termination in Table 4-4	Control Signal	Negated on Rising Edge of State			Results—Next Cycle
		N		N+2	
Bus Error	$\overline{\text{BERR}}$	•	or	•	Takes bus error trap.
	$\overline{\text{HALT}}$	•	or	•	
Rerun	$\overline{\text{BERR}}$	•	or	•	Illegal sequence; usually traps to vector number 0.
	$\overline{\text{HALT}}$	•			
Rerun	$\overline{\text{BERR}}$	•			Reruns the bus cycle.
	$\overline{\text{HALT}}$			•	
Normal	$\overline{\text{BERR}}$	•			May lengthen next cycle.
	$\overline{\text{HALT}}$	•	or	•	
Normal	$\overline{\text{BERR}}$			•	If next cycle is started, it will be terminated as a bus error.
	$\overline{\text{HALT}}$	•	or	none	

• = Signal is negated in this bus state.

5.7 ASYNCHRONOUS OPERATION

To achieve clock frequency independence at a system level, the bus can be operated in an asynchronous manner. Asynchronous bus operation uses the bus handshake signals to control the transfer of data. The handshake signals are $\overline{\text{AS}}$, $\overline{\text{UDS}}$, $\overline{\text{LDS}}$, $\overline{\text{DS}}$ (MC68008 only), $\overline{\text{DTACK}}$, $\overline{\text{BERR}}$, $\overline{\text{HALT}}$, $\overline{\text{AVEC}}$ (MC68EC000 only), and $\overline{\text{VPA}}$ (only for M6800 peripheral cycles). $\overline{\text{AS}}$ indicates the start of the bus cycle, and $\overline{\text{UDS}}$, $\overline{\text{LDS}}$, and $\overline{\text{DS}}$ signal valid data for a write cycle. After placing the requested data on the data bus (read cycle) or latching the data (write cycle), the slave device (memory or peripheral) asserts $\overline{\text{DTACK}}$ to terminate the bus cycle. If no device responds or if the access is invalid, external control logic asserts $\overline{\text{BERR}}$, or $\overline{\text{BERR}}$ and $\overline{\text{HALT}}$, to abort or retry the cycle. Figure 5-31 shows the use of the bus handshake signals in a fully asynchronous read cycle. Figure 5-32 shows a fully asynchronous write cycle.

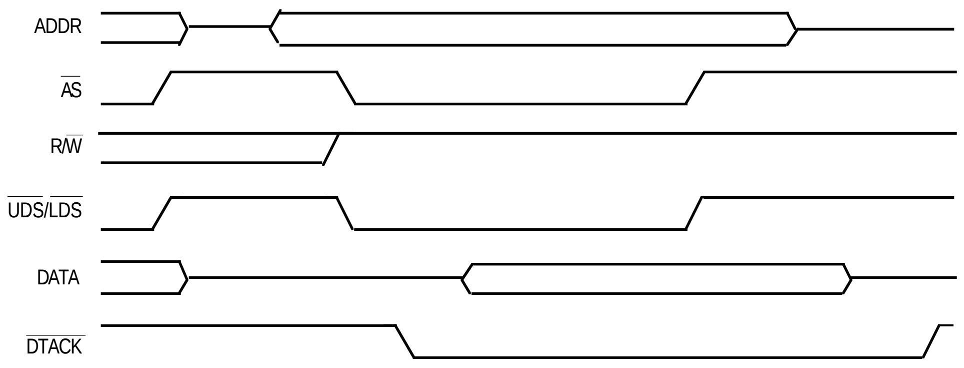


Figure 5-31. Fully Asynchronous Read Cycle

SECTION 7
8-BIT INSTRUCTION EXECUTION TIMES

This section contains listings of the instruction execution times in terms of external clock (CLK) periods for the MC68008 and MC68HC001/MC68EC000 in 8-bit mode. 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 periods 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 instruction.

NOTE

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

7.1 OPERAND EFFECTIVE ADDRESS CALCULATION TIMES

Table 7-1 lists the numbers of clock periods required to compute the effective addresses for instructions. The totals include fetching any extension words, computing the address, and fetching the memory operand. The total number of clock periods, the number of read cycles, and the number of write cycles (zero for all effective address calculations) are shown in the previously described format.

Table 8-3. 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)	**12**(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)	**12**(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, 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, 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.

8.3 STANDARD INSTRUCTION EXECUTION TIMES

The numbers of clock periods shown in Table 8-4 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 8-4, 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 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 (+).

10.11 AC ELECTRICAL SPECIFICATIONS—MC68000 TO M6800 PERIPHERAL (V_{CC} = 5.0 Vdc ±5%; GND=0 Vdc; $T_A = T_L$ TO T_H; refer to figures 10-6)

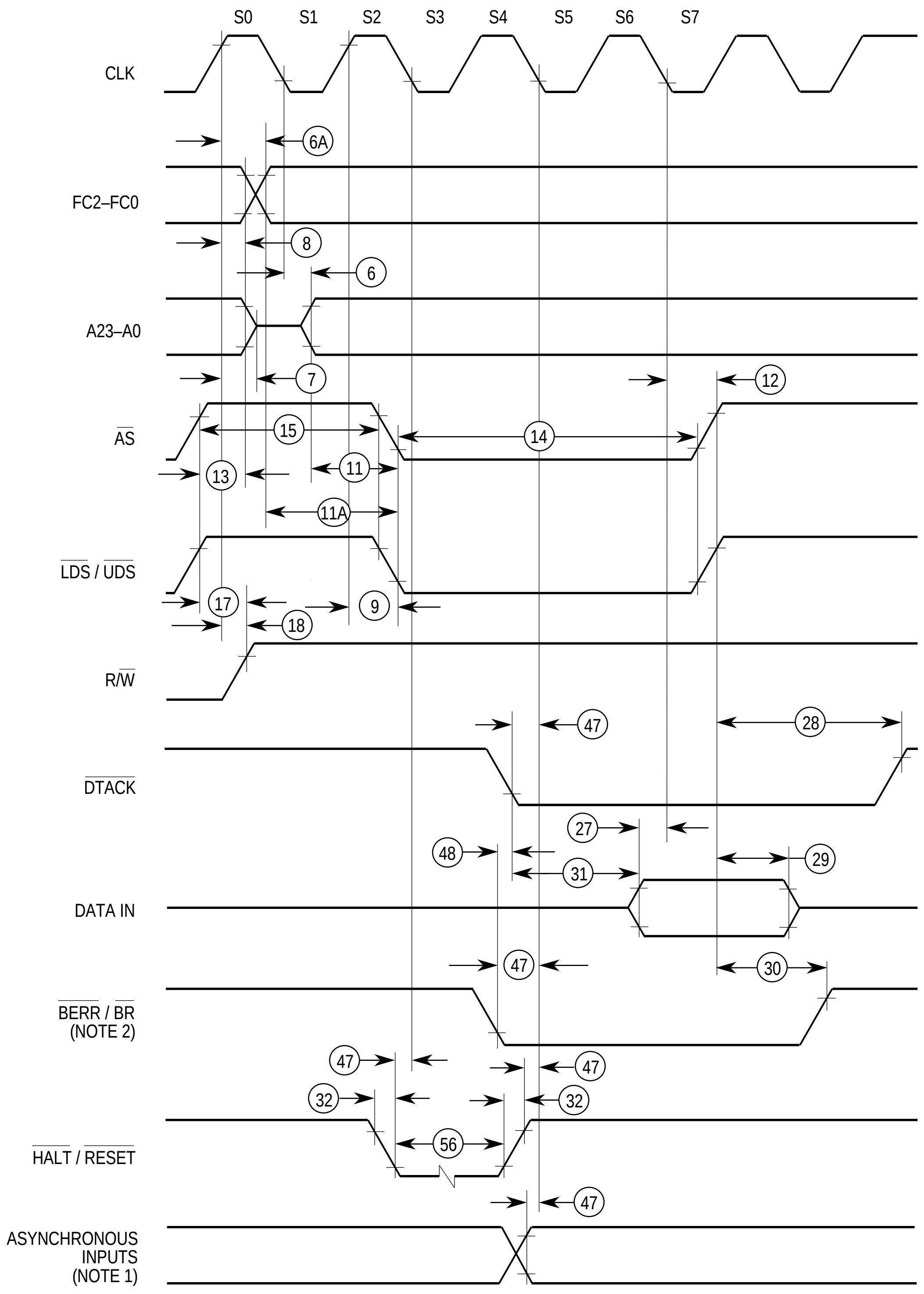
(Applies To All Processors Except The MC68EC000)

Num	Characteristic	8 MHz*		10 MHz*		12.5 MHz*		16.67 MHz `12F'		16 MHz		20 MHz**		Unit
		Min	Max	Min	Max	Min	Max	Min	Max	Min	Max	Min	Max	
12[1]	Clock Low to $\overline{AS}$, $\overline{DS}$ Negated	—	62	—	50	—	40	—	40	3	30	3	25	ns
18[1]	Clock High to R/$\overline{W}$ High (Read)	0	55	0	45	0	40	0	40	0	30	0	25	ns
20[1]	Clock High to R/$\overline{W}$ Low (Write)	0	55	0	45	0	40	0	40	0	30	0	25	ns
23	Clock Low to Data-Out Valid (Write)	—	62	—	50	—	50	—	50	—	30	—	25	ns
27	Data-In Valid to Clock Low (Setup Time on Read)	10	—	10	—	10	—	7	—	5	—	5	—	ns
29	$\overline{AS}$, $\overline{DS}$ Negated to Data-In Invalid (Hold Time on Read)	0	—	0	—	0	—	0	—	0	—	0	—	ns
40	Clock Low to $\overline{VMA}$ Asserted	—	70	—	70	—	70	—	50	—	50	—	40	ns
41	Clock Low to E Transition	—	55	—	45	—	35	—	35	—	35	—	30	ns
42	E Output Rise and Fall Time	—	15	—	15	—	15	—	15	—	15	—	12	ns
43	$\overline{VMA}$ Asserted to E High	200	—	150	—	90	—	80	—	80	—	60	—	ns
44	$\overline{AS}$, $\overline{DS}$ Negated to $\overline{VPA}$ Negated	0	120	0	90	0	70	0	50	0	50	0	42	ns
45	E Low to Control, Address Bus Invalid (Address Hold Time)	30	—	10	—	10	—	10	—	10	—	10	—	ns
47	Asynchronous Input Setup Time	10	—	10	—	10	—	10	—	10	—	5	—	ns
49[2]	$\overline{AS}$, $\overline{DS}$, Negated to E Low	-70	70	-55	55	-45	45	-35	35	-35	35	–30	30	ns
50	E Width High	450	—	350	—	280	—	220	—	220	—	190	—	ns
51	E Width Low	700	—	550	—	440	—	340	—	340	—	290	—	ns
54	E Low to Data-Out Invalid	30	—	20	—	15	—	10	—	10	—	5	—	ns

*These specifications represent 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.

** This frequency applies only to MC68HC000 and MC68HC001.

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



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 10-12. MC68EC000 Read Cycle Timing Diagram

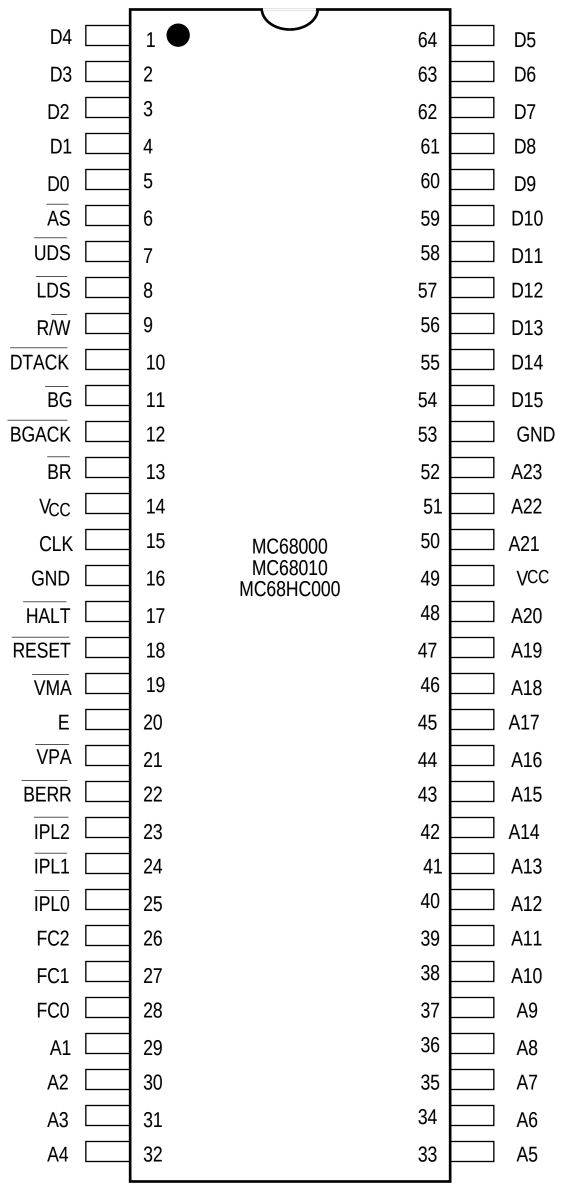


Figure 11-1. 64-Pin Dual In Line

After recognizing $\overline{\text{VPA}}$, the processor assures that enable (E) is low by waiting, if necessary, and subsequently asserts $\overline{\text{VMA}}$. $\overline{\text{VMA}}$ 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 $\overline{\text{VPA}}$ to the E clock.

When external circuitry asserts $\overline{\text{VPA}}$ as soon as possible following the assertion of $\overline{\text{AS}}$, the assertion of $\overline{\text{VPA}}$ is recognized on the falling edge of S4. In this case, no extra wait states are inserted (waiting for the assertion of $\overline{\text{VPA}}$). 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 $\overline{\text{VPA}}$ 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 $\overline{\text{VPA}}$ 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/$\overline{\text{W}}$ is driven high. Logic in the peripheral must remove $\overline{\text{VPA}}$ within one clock after the negation of address strobe.

Data transfer acknowledge ($\overline{\text{DTACK}}$) must not be asserted while VPA is asserted. The state machine in the processor looks for $\overline{\text{DTACK}}$ to identify an asynchronous bus cycle and for $\overline{\text{VPA}}$ 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, $\overline{\text{VMA}}$ is active low for the M68000 Family of processors. The active-low $\overline{\text{VMA}}$ 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, $\overline{\text{VPA}}$ is asserted, and the processor (or external circuitry) asserts $\overline{\text{VMA}}$ 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.

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