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	12MHz
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-BCPGA
Supplier Device Package	68-PGA (26.92x26.92)
Purchase URL	https://www.e-xfl.com/product-detail/nxp-semiconductors/mc68hc000crc12

LIST OF ILLUSTRATIONS (Concluded)

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

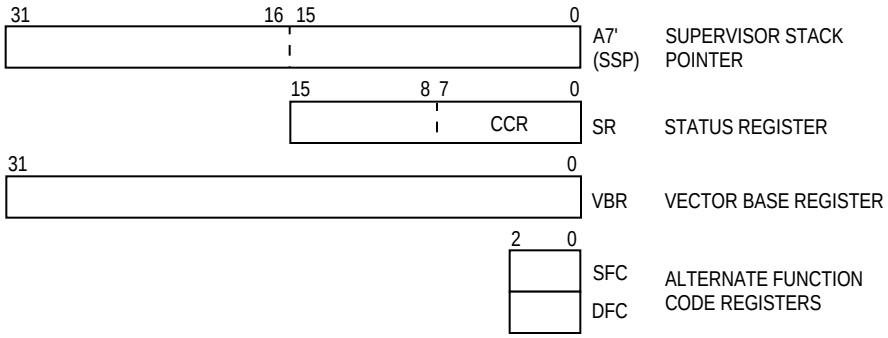


Figure 2-3. Supervisor Programmer's Model Supplement (MC68010)

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

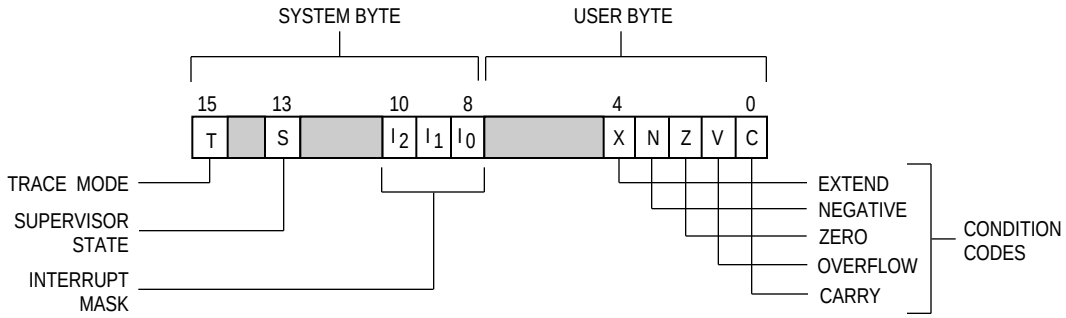


Figure 2-4. Status Register

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)

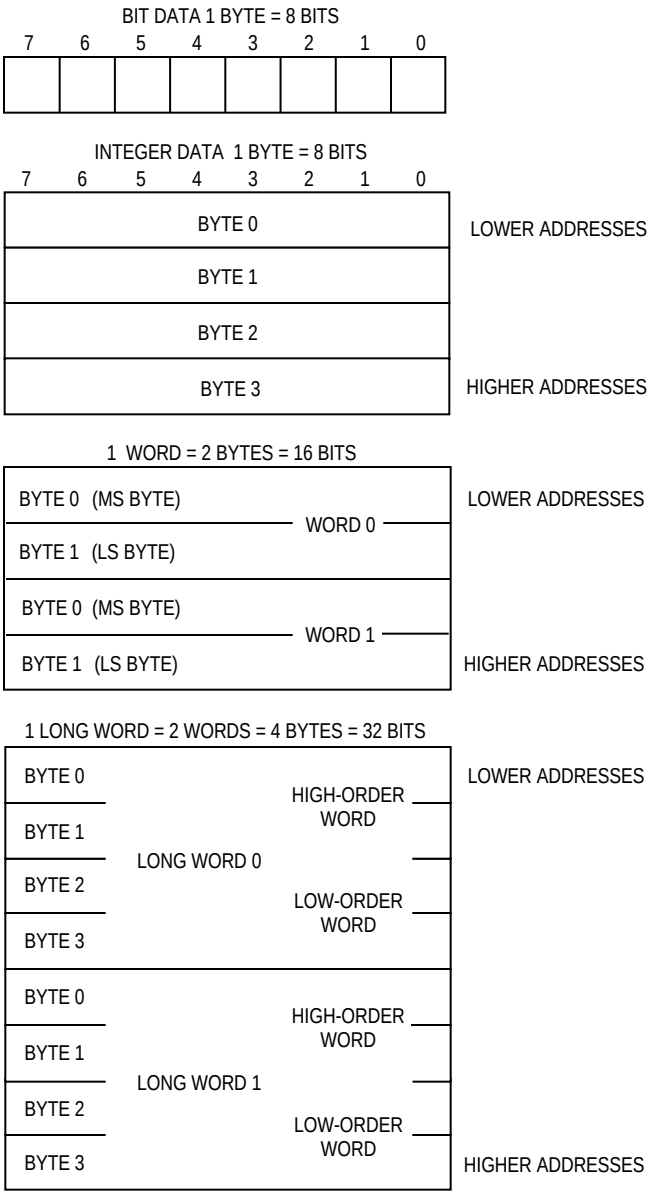


Figure 2-7. Memory Data Organization of the MC68008

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.

shifted by, rotated by — The source operand is shifted or rotated by the number of positions specified by the second operand

Notation for single-operand operations:

~<operand> — The operand is logically complemented

<operand>sign-extended — The operand is sign-extended, all bits of the upper portion are made equal to the high-order bit of the lower portion

<operand>tested — The operand is compared to zero and the condition codes are set appropriately

Notation for other operations:

TRAP — Equivalent to Format/Offset Word → (SSP); SSP–2 → SSP; PC → (SSP); SSP–4 → SSP; SR → (SSP); SSP–2 → SSP; (vector) → PC

STOP — Enter the stopped state, waiting for interrupts

If <condition> then <operations> else <operations> — The condition is tested. If true, the operations after "then" are performed. If the condition is false and the optional "else" clause is present, the operations after "else" are performed. If the condition is false and else is omitted, the instruction performs no operation. Refer to the Bcc instruction description as an example.

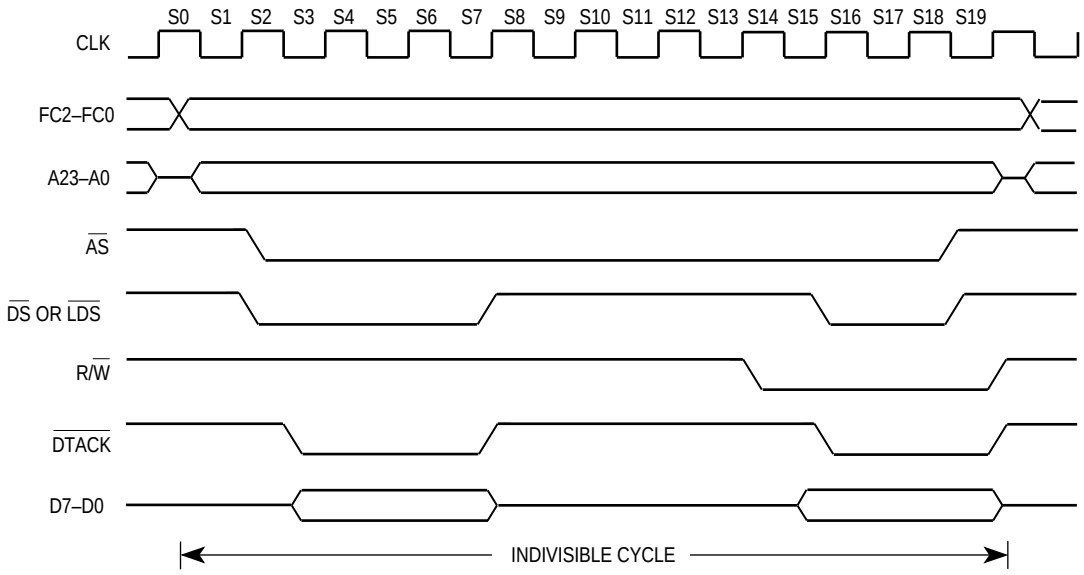


Figure 4-6. Read-Modify-Write Cycle Timing Diagram

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 $\overline{LDS}$, or $\overline{DS}$.

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 $\overline{LDS}$, and $\overline{DS}$. 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.

STATE 12 The write portion of the cycle starts in S12. The valid function codes on FC2–FC0, the address bus lines, $\overline{AS}$, and R/$\overline{W}$ remain unaltered.

STATE 13 During S13, no bus signals are altered.

STATE 14 On the rising edge of S14, the processor drives R/$\overline{W}$ low.

STATE 15 During S15, the data bus is driven out of the high-impedance state as the data to be written are placed on the bus.

STATE 16 At the rising edge of S16, the processor asserts $\overline{LDS}$ or $\overline{DS}$. The processor waits for $\overline{DTACK}$ or $\overline{BERR}$ or $\overline{VPA}$, an M6800 peripheral signal. When $\overline{VPA}$ is asserted during S16, 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 close of S16, the processor inserts wait states (full clock cycles) until either $\overline{DTACK}$ or $\overline{BERR}$ is asserted.

STATE 17 During S17, no bus signals are altered.

STATE 18 During S18, no bus signals are altered.

STATE 19 On the falling edge of the clock entering S19, the processor negates $\overline{AS}$, $\overline{LDS}$, and $\overline{DS}$. As the clock rises at the end of S19, the processor places the address and data buses in the high-impedance state, and drives R/$\overline{W}$ high. The device negates $\overline{DTACK}$ or $\overline{BERR}$ at this time.

4.2 OTHER BUS OPERATIONS

Refer to **Section 5 16-Bit Bus Operations** for information on the following items:

- CPU Space Cycle
- Bus Arbitration
  - — Bus Request
  - — Bus Grant
  - — Bus Acknowledgment
- Bus Control
- Bus Errors and Halt Operations
- Reset Operations
- Asynchronous Operations
- Synchronous Operations

A bus cycle consists of eight states. The various signals are asserted during specific states of a read cycle, as follows:

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

STATE 1 — Entering state 1 (S1), the processor drives a valid address on the address bus.

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

STATE 3 — During state 3 (S3), no bus signals are altered.

STATE 4 — During state 4 (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 state 5 (S5), no bus signals are altered.

STATE 6 — During state 6 (S6), data from the device is driven onto the data bus.

STATE 7 — On the falling edge of the clock entering state 7 (S7), the processor latches data from the addressed device and negates $\overline{AS}$, $\overline{UDS}$, and $\overline{LDS}$. At the rising edge of S7, the processor places the address bus in the high-impedance state. The device negates $\overline{DTACK}$ or $\overline{BERR}$ at this time.

NOTE

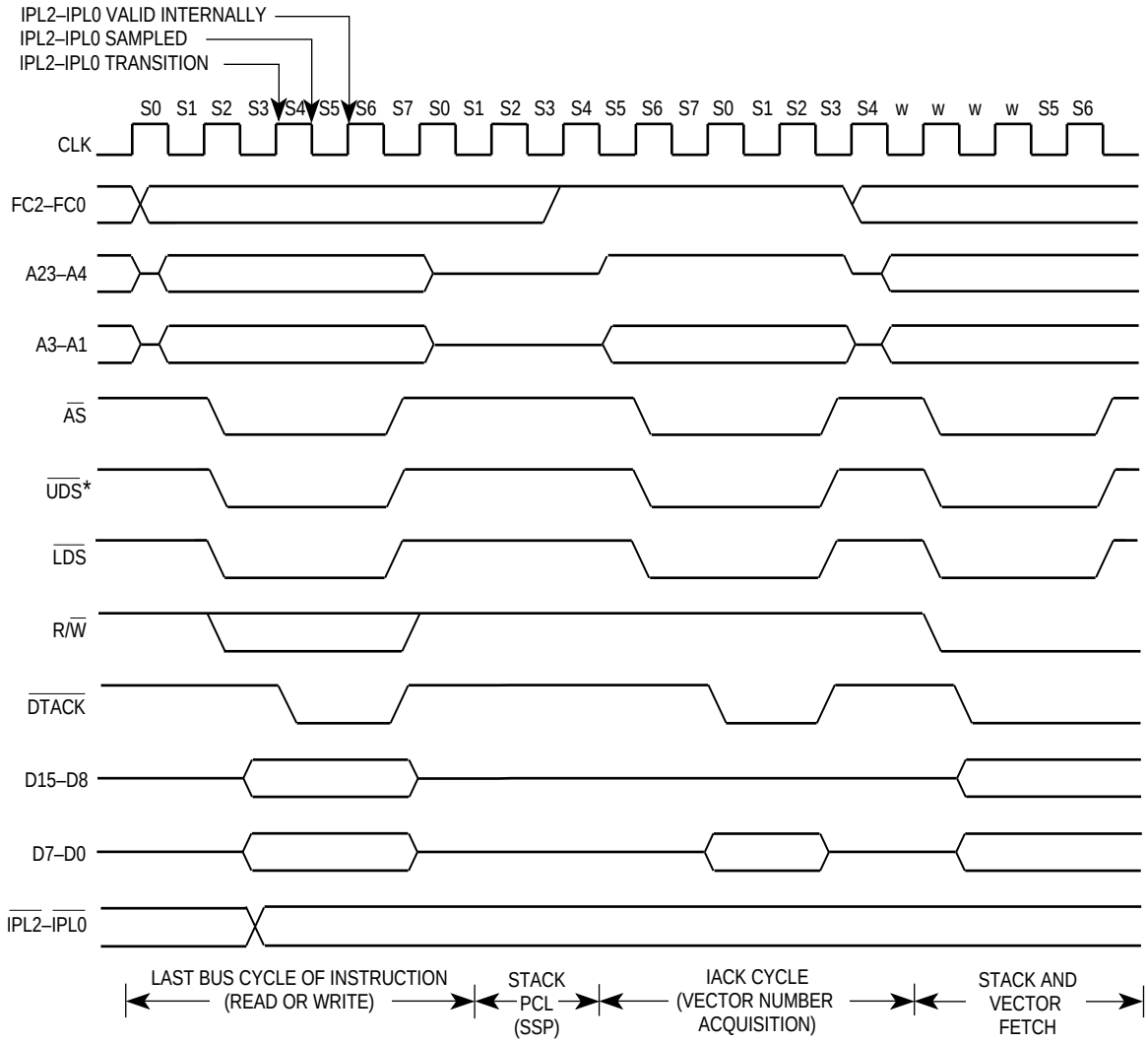
During an active bus cycle, $\overline{VPA}$ and $\overline{BERR}$ are sampled on every falling edge of the clock beginning with S4, and data is latched on the falling edge of S6 during a read cycle. The bus cycle terminates in S7, except when $\overline{BERR}$ is asserted in the absence of $\overline{DTACK}$. In that case, the bus cycle terminates one clock cycle later in S9.

5.1.2 Write Cycle

During a write cycle, the processor sends bytes of data to the memory or peripheral device. If the instruction specifies a word operation, the processor issues both $\overline{UDS}$ and $\overline{LDS}$ and writes both bytes. When the instruction specifies a byte operation, the processor uses the internal A0 bit to determine which byte to write and issues the appropriate data strobe. When the A0 bit equals zero, $\overline{UDS}$ is asserted; when the A0 bit equals one, $\overline{LDS}$ is asserted.

The interrupt acknowledge cycle places the level of the interrupt being acknowledged on address bits A3–A1 and drives all other address lines high. The interrupt acknowledge cycle reads a vector number when the interrupting device places a vector number on the data bus and asserts $\overline{\text{DTACK}}$ to acknowledge the cycle.

The timing diagram for an interrupt acknowledge cycle is shown in Figure 5-11. Alternately, the interrupt acknowledge cycle can be autovectored. The interrupt acknowledge cycle is the same, except the interrupting device asserts $\overline{\text{VPA}}$ instead of $\overline{\text{DTACK}}$. For an autovectored interrupt, the vector number used is $18 plus the interrupt level. This is generated internally by the microprocessor when $\overline{\text{VPA}}$ (or $\overline{\text{AVEC}}$) is asserted on an interrupt acknowledge cycle. $\overline{\text{DTACK}}$ and $\overline{\text{VPA}}$ ($\overline{\text{AVEC}}$) should never be simultaneously asserted.



*Although a vector number is one byte, both data strobes are asserted due to the microcode used for exception processing. The processor does not recognize anything on data lines D8 through D15 at this time.

Figure 5-11. Interrupt Acknowledge Cycle Timing Diagram

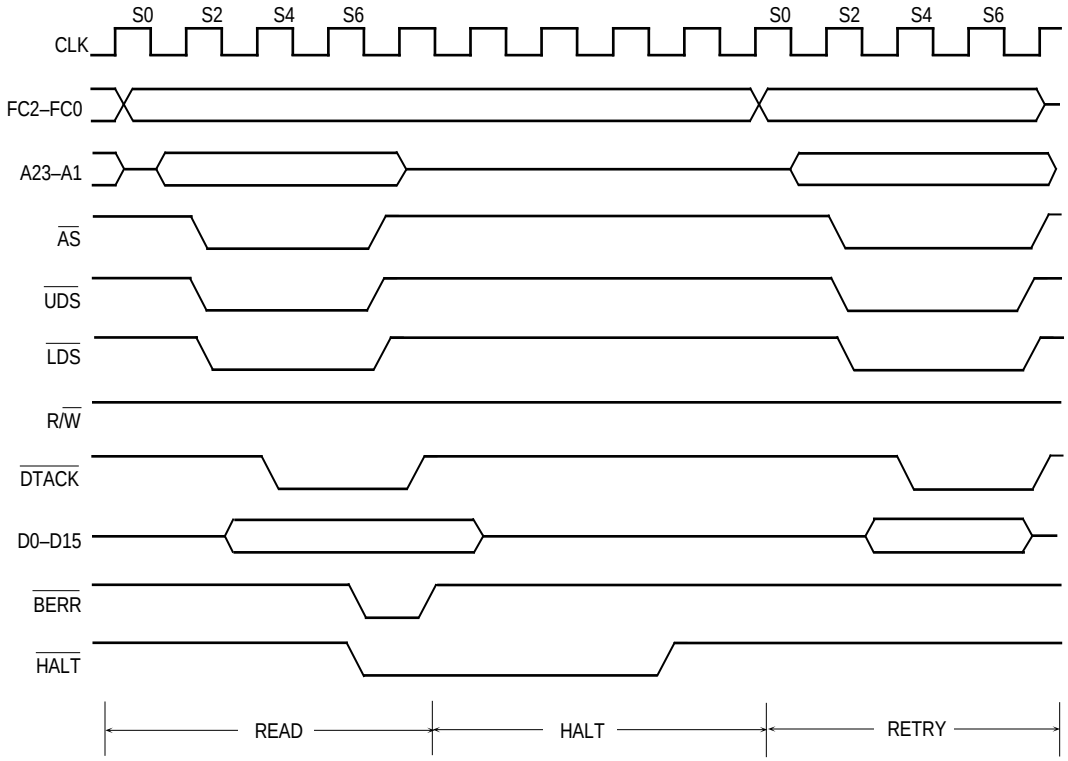


Figure 5-29. Halt Operation Timing Diagram

While the processor is halted, the address bus and the data bus signals are placed in the high-impedance state. Bus arbitration is performed as usual. Should a bus error occur while $\overline{\text{HALT}}$ is asserted, the processor performs the retry operation previously described.

The single-step mode is derived from correctly timed transitions of $\overline{\text{HALT}}$. $\overline{\text{HALT}}$ is negated to allow the processor to begin a bus cycle, then asserted to enter the halt mode when the cycle completes. The single-step mode proceeds through a program one bus cycle at a time for debugging purposes. The halt operation and the hardware trace capability allow tracing of either bus cycles or instructions one at a time. These capabilities and a software debugging package provide total debugging flexibility.

5.4.4 Double Bus Fault

When a bus error exception occurs, the processor begins exception processing by stacking information on the supervisor stack. If another bus error occurs during exception processing (i.e., before execution of another instruction begins) the processor halts and asserts $\overline{\text{HALT}}$. This is called a double bus fault. Only an external reset operation can restart a processor halted due to a double bus fault.

A retry operation does not initiate exception processing; a bus error during a retry operation does not cause a double bus fault. The processor can continue to retry a bus cycle indefinitely if external hardware requests.

On the rising edge of the clock, at the end of S7 (which may be the start of S0 for the next bus cycle), the processor places the address bus in the high-impedance state. During a write cycle, the processor also places the data bus in the high-impedance state and drives R/$\overline{W}$ high. External logic circuitry should respond to the negation of the $\overline{AS}$ and $\overline{UDS}$, $\overline{LDS}$, and/or $\overline{DS}$ by negating $\overline{DTACK}$ and/or $\overline{BERR}$. Parameter #28 is the hold time for $\overline{DTACK}$, and parameter #30 is the hold time for $\overline{BERR}$.

Figure 5-35 shows a synchronous read cycle and the important timing parameters that apply. The timing for a synchronous read cycle, including relevant timing parameters, is shown in Figure 5-36.

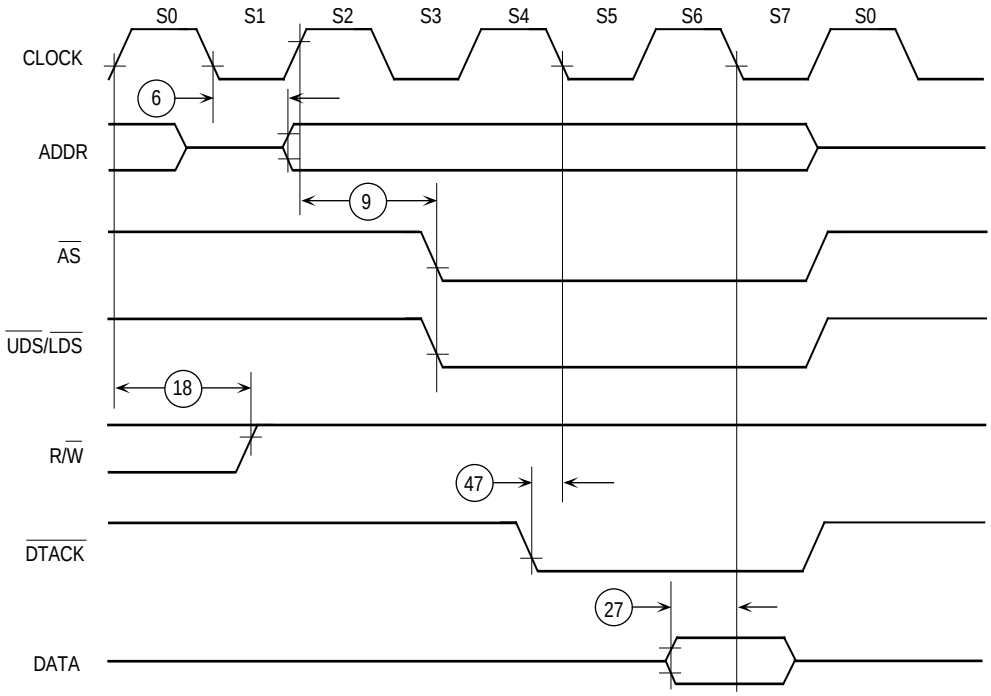


Figure 5-35. Synchronous Read Cycle

After the execution of the instruction is complete and before the start of the next instruction, exception processing for a trace begins. A copy is made of the status register. The transition to supervisor mode is made, and the T bit of the status register is turned off, disabling further tracing. The vector number is generated to reference the trace exception vector, and the current program counter and the copy of the status register are saved on the supervisor stack. On the MC68010, the format/offset word is also saved on the supervisor stack. The saved value of the program counter is the address of the next instruction. Instruction execution commences at the address contained in the trace exception vector.

6.3.9 Bus Error

A bus error exception occurs when the external logic requests that a bus error be processed by an exception. The current bus cycle is aborted. The current processor activity, whether instruction or exception processing, is terminated, and the processor immediately begins exception processing. The bus error facility is identical on the all processors; however, the stack frame produced on the MC68010 contains more information. The larger stack frame supports instruction continuation, which supports virtual memory on the MC68010 processor.

6.3.9.1 BUS ERROR. Exception processing for a bus error follows the usual sequence of steps. The status register is copied, the supervisor mode is entered, and tracing is turned off. The vector number is generated to refer to the bus error vector. Since the processor is fetching the instruction or an operand when the error occurs, the context of the processor is more detailed. To save more of this context, additional information is saved on the supervisor stack. The program counter and the copy of the status register are saved. The value saved for the program counter is advanced 2–10 bytes beyond the address of the first word of the instruction that made the reference causing the bus error. If the bus error occurred during the fetch of the next instruction, the saved program counter has a value in the vicinity of the current instruction, even if the current instruction is a branch, a jump, or a return instruction. In addition to the usual information, the processor saves its internal copy of the first word of the instruction being processed and the address being accessed by the aborted bus cycle. Specific information about the access is also saved: type of access (read or write), processor activity (processing an instruction), and function code outputs when the bus error occurred. The processor is processing an instruction if it is in the normal state or processing a group 2 exception; the processor is not processing an instruction if it is processing a group 0 or a group 1 exception. Figure 6-7 illustrates how this information is organized on the supervisor stack. If a bus error occurs during the last step of exception processing, while either reading the exception vector or fetching the instruction, the value of the program counter is the address of the exception vector. Although this information is not generally sufficient to effect full recovery from the bus error, it does allow software diagnosis. Finally, the processor commences instruction processing at the address in the vector. It is the responsibility of the error handler routine to clean up the stack and determine where to continue execution.

If a bus error occurs during the exception processing for a bus error, an address error, or a reset, the processor halts and all processing ceases. This halt simplifies the detection of a catastrophic system failure, since the processor removes itself from the system to

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.

7.7 BIT MANIPULATION INSTRUCTION EXECUTION TIMES

Table 7-9 lists the timing data for the bit manipulation 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 7-9. Bit Manipulation Instruction Execution Times

Instruction	Size	Dynamic		Static	
		Register	Memory	Register	Memory
BCHG	Byte Long	— **12**(2/0)*	**12**(2/1)+ —	— **20**(4/0)*	**20**(4/1)+ —
BCLR	Byte Long	— **14**(2/0)*	**12**(2/1)+ —	— **22**(4/0)*	**20**(4/1)+ —
BSET	Byte Long	— **12**(2/0)*	**12**(2/1)+ —	— **20**(4/0)*	**20**(4/1)+ —
BTST	Byte Long	— **10**(2/0)	**8**(2/0)+	— **18**(4/0)	**16**(4/0)+ —

+Add effective address calculation time.
* Indicates maximum value; data addressing mode only.

7.8 CONDITIONAL INSTRUCTION EXECUTION TIMES

Table 7-10 lists the timing data for the conditional 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 7-10. Conditional Instruction Execution Times

Instruction	Displacement	Trap or Branch Taken	Trap or Branch Not Taken
Bcc	Byte Word	**18**(4/0) **18**(4/0)	**12**(2/0) **20**(4/0)
BRA	Byte Word	**18**(4/0) **18**(4/0)	— —
BSR	Byte Word	**34**(4/4) **34**(4/4)	— —
DBcc	CC True CC False	— **18**(4/0)	**20**(4/0) **26**(6/0)
CHK	—	**68**(8/6)+*	**14**(2/0)
TRAP	—	**62**(8/6)	**—**
TRAPV	—	**66**(10/6)	**8**(2/0)

+Add effective address calculation time for word operand.
* Indicates maximum base value.

Table 8-6. Single Operand Instruction Execution Times

Instruction	Size	Register	Memory
CLR	Byte, Word	**4**(1/0)	**8**(1/1)+
	Long	**6**(1/0)	**12**(1/2)+
NBCD	Byte	**6**(1/0)	**8**(1/1)+
NEG	Byte, Word	**4**(1/0)	**8**(1/1)+
	Long	**6**(1/0)	**12**(1/2)+
NEGX	Byte, Word	**4**(1/0)	**8**(1/1)+
	Long	**6**(1/0)	**12**(1/2)+
NOT	Byte, Word	**4**(1/0)	**8**(1/1)+
	Long	**6**(1/0)	**12**(1/2)+
Scc	Byte, False	**4**(1/0)	**8**(1/1)+
	Byte, True	**6**(1/0)	**8**(1/1)+
TAS	Byte	**4**(1/0)	**14**(2/1)+
TST	Byte, Word	**4**(1/0)	**4**(1/0)+
	Long	**4**(1/0)	**4**(1/0)+

+Add effective address calculation time.

8.6 SHIFT/ROTATE INSTRUCTION EXECUTION TIMES

Table 8-7 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 8-7. Shift/Rotate Instruction Execution Times

Instruction	Size	Register	Memory
ASR, ASL	Byte, Word	**6+2n**(1/0)	**8**(1/1)+
	Long	**8+2n**(1/0)	—
LSR, LSL	Byte, Word	**6+2n**(1/0)	**8**(1/1)+
	Long	**8+2n**(1/0)	—
ROR, ROL	Byte, Word	**6+2n**(1/0)	**8**(1/1)+
	Long	**8+2n**(1/0)	—
ROXR, ROXL	Byte, Word	**6+2n**(1/0)	**8**(1/1)+
	Long	**8+2n**(1/0)	—

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

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 10-1 summarizes maximum power dissipation and average junction temperature for the curve drawn in Figure 10-1, using the minimum and maximum values of ambient temperature for different packages and substituting $^{\theta}J_C$ for $^{\theta}J_A$ (assuming good thermal management). Table 10-2 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 10-1 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*.

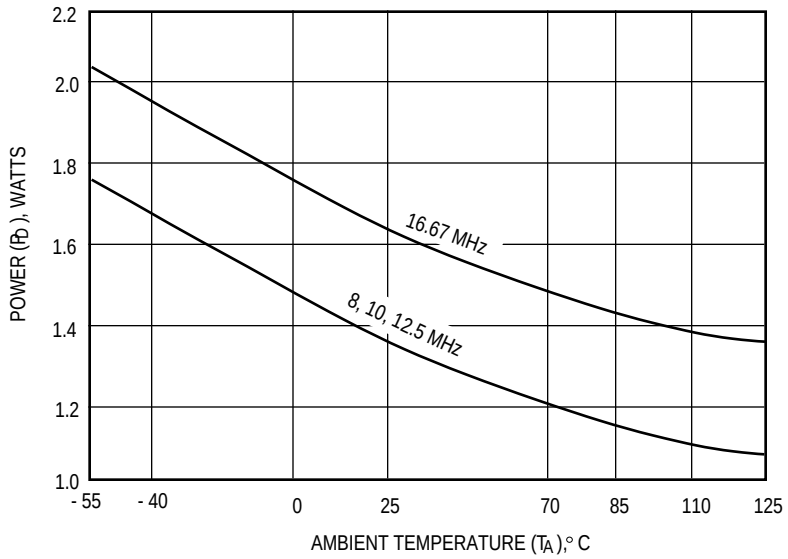


Figure 10-1. MC68000 Power Dissipation (P_D) vs Ambient Temperature (T_A) (Not Applicable to MC68HC000/68HC001/68EC000)

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 and MC68EC000 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; other may require additional circuitry.

The MC68HC000 and MC68EC000, implemented in CMOS, is applicable to designs to which the following considerations are relevant:

1. The MC68HC000 and MC68EC000 completely satisfies the input/output drive requirements of CMOS logic devices.
2. The HCMOS MC68HC000 and MC68EC000 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.

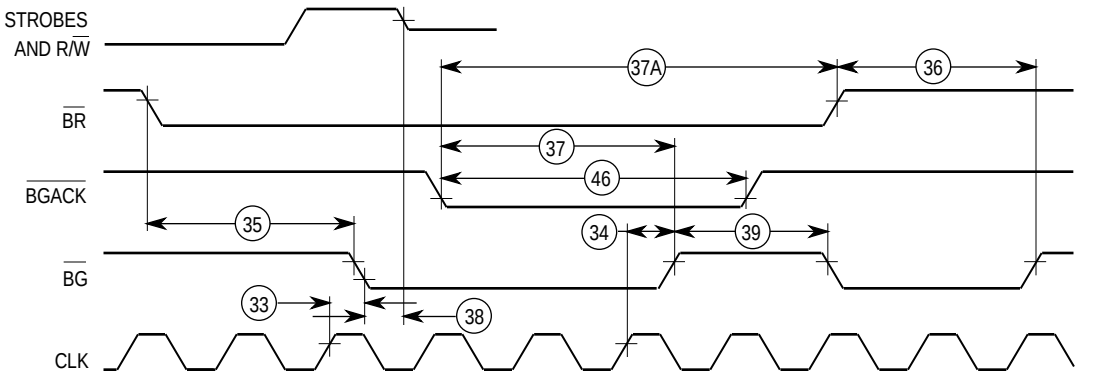
10.5 AC ELECTRICAL SPECIFICATION DEFINITIONS

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 10-2. 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 setup and hold times, and are measured as shown. Finally, the measurement for signal-to-signal specifications are 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.

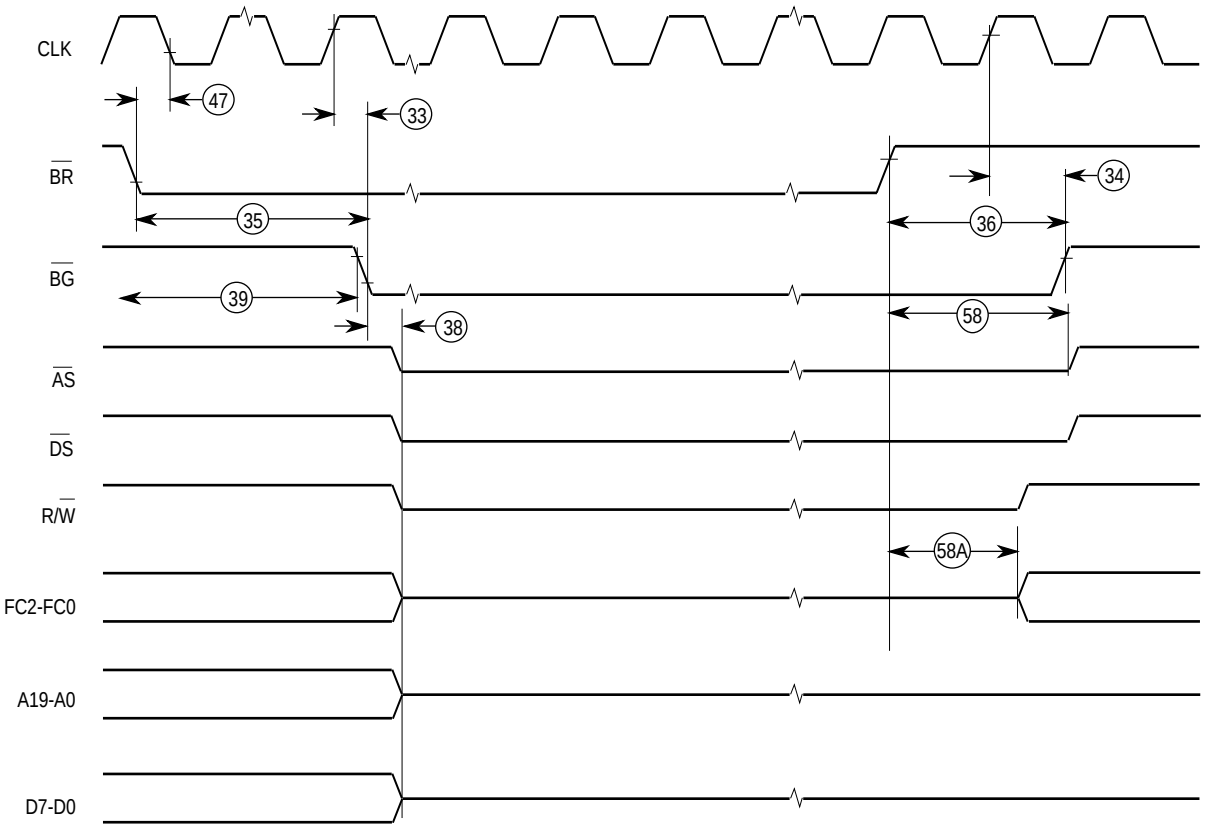


NOTE: Setup time to the clock (#47) for the asynchronous inputs BERR, BGACK, BR, DTACK, IPL2-IPL0, and VPA guarantees their recognition at the next falling edge of the clock.

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

10.14 MC68EC000 AC ELECTRICAL SPECIFICATIONS — READ AND WRITE CYCLES (VCC=5.0 VDC ± 5;PC; GND = 0 VDC; $T_A = T_L$ TO T_H; (See Figures 10-12 and 10-13)

Num	Characteristic	8 MHz		10 MHz		12.5 MHz		16.67 MHz		20 MHz		Unit
		Min	Max	Min	Max	Min	Max	Min	Max	Min	Max	
6	Clock Low to Address Valid	—	35	—	35	—	35	—	30	—	25	ns
6A	Clock High to FC Valid	—	35	—	35	—	35	—	30	0	25	ns
7	Clock High to Address, Data Bus High Impedance (Maximum)	—	55	—	55	—	55	—	50	—	42	ns
8	Clock High to Address, FC Invalid (Minimum)	0	—	0	—	0	—	0	—	0	—	ns
9[1]	Clock High to $\overline{AS}$, $\overline{DS}$ Asserted	3	35	3	35	3	35	3	30	3	25	ns
11[2]	Address Valid to $\overline{AS}$, $\overline{DS}$ Asserted (Read)/$\overline{AS}$ Asserted (Write)	30	—	20	—	15	—	15	—	10	—	ns
11A[2]	FC Valid to $\overline{AS}$, $\overline{DS}$ Asserted (Read)/ $\overline{AS}$ Asserted (Write)	45	—	45	—	45	—	45	—	40	—	ns
12[1]	Clock Low to $\overline{AS}$, $\overline{DS}$ Negated	3	35	3	35	3	35	3	30	3	25	ns
13[2]	$\overline{AS}$, $\overline{DS}$ Negated to Address, FC Invalid	15	—	15	—	15	—	15	—	10	—	ns
14[2]	$\overline{AS}$ (and $\overline{DS}$ Read) Width Asserted	270	—	195	—	160	—	120	—	100	—	ns
14A[2]	$\overline{DS}$ Width Asserted (Write)	140	—	95	—	80	—	60	—	50	—	ns
15[2]	$\overline{AS}$, $\overline{DS}$ Width Negated	150	—	105	—	65	—	60	—	50	—	ns
16	Clock High to Control Bus High Impedance	—	55	—	55	—	55	—	50	—	42	ns
17[2]	$\overline{AS}$, $\overline{DS}$ Negated to R/$\overline{W}$ Invalid	15	—	15	—	15	—	15	—	10	—	ns
18[1]	Clock High to R/$\overline{W}$ High (Read)	0	35	0	35	0	35	0	30	0	25	ns
20[1]	Clock High to R/$\overline{W}$ Low (Write)	0	35	0	35	0	35	0	30	0	25	ns
20A[2,6]	$\overline{AS}$ Asserted to R/$\overline{W}$ Low (Write)	—	10	—	10	—	10	—	10	—	10	ns
21[2]	Address Valid to R/$\overline{W}$ Low (Write)	0	—	0	—	0	—	0	—	0	—	ns
21A[2]	FC Valid to R/$\overline{W}$ Low (Write)	60	—	50	—	30	—	30	—	25	—	ns
22[2]	R/$\overline{W}$ Low to $\overline{DS}$ Asserted (Write)	80	—	50	—	30	—	30	—	25	—	ns
23	Clock Low to Data-Out Valid (Write)	—	35	—	35	—	35	—	30	—	25	ns
25[2]	$\overline{AS}$, $\overline{DS}$ Negated to Data-Out Invalid (Write)	40	—	30	—	20	—	15	—	10	—	ns
26[2]	Data-Out Valid to $\overline{DS}$ Asserted (Write)	40	—	30	—	20	—	15	—	10	—	ns
27[5]	Data-In Valid to Clock Low (Setup Time on Read)	5	—	5	—	5	—	5	—	5	—	ns
28[2]	$\overline{AS}$, $\overline{DS}$ Negated to $\overline{DTACK}$ Negated (Asynchronous Hold)	0	110	0	110	0	110	0	110	0	95	ns
28A	Clock High to $\overline{DTACK}$ Negated	0	110	0	110	0	110	0	110	0	95	ns



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