1.0 THE i960® MC PROCESSOR

The 80960MC, a member of Intel's i960® 32-bit processor family, is ideally suited for embedded applications. It includes a 512-byte instruction cache and a built-in interrupt controller. The 80960MC has a large register set, multiple parallel execution units and a high-bandwidth burst bus. Using advanced RISC technology, this processor is capable of execution rates in excess of 9.4 million instructions per second*. The 80960MC is well-suited for a wide range of applications including non-impact printers, I/O control and specialty instrumentation. The embedded market includes applications as diverse as industrial automation, avionics, image processing, graphics and networking. These types of applications require high integration, low power consumption, quick interrupt response times and high performance. Since time to market is critical, embedded processors must be easy to use in both hardware and software designs.

All members of the i960 processor family share a common core architecture which utilizes RISC technology so that, except for special functions, the family members are object-code compatible. Each new processor in the family adds its own special set of functions to the core to satisfy the needs of a specific application or range of applications in the embedded market.

The 80960MC includes an integrated Floating Point Unit (FPU), a Memory Management Unit (MMU), multitasking support, and multiprocessor support. Two commercial members of the i960® family provide similar features: the 80960KB processor with integrated FPU and the 80960KA without floating-point.

* Relative to Digital Equipment Corporation's VAX-11/780* at 1 MIPS

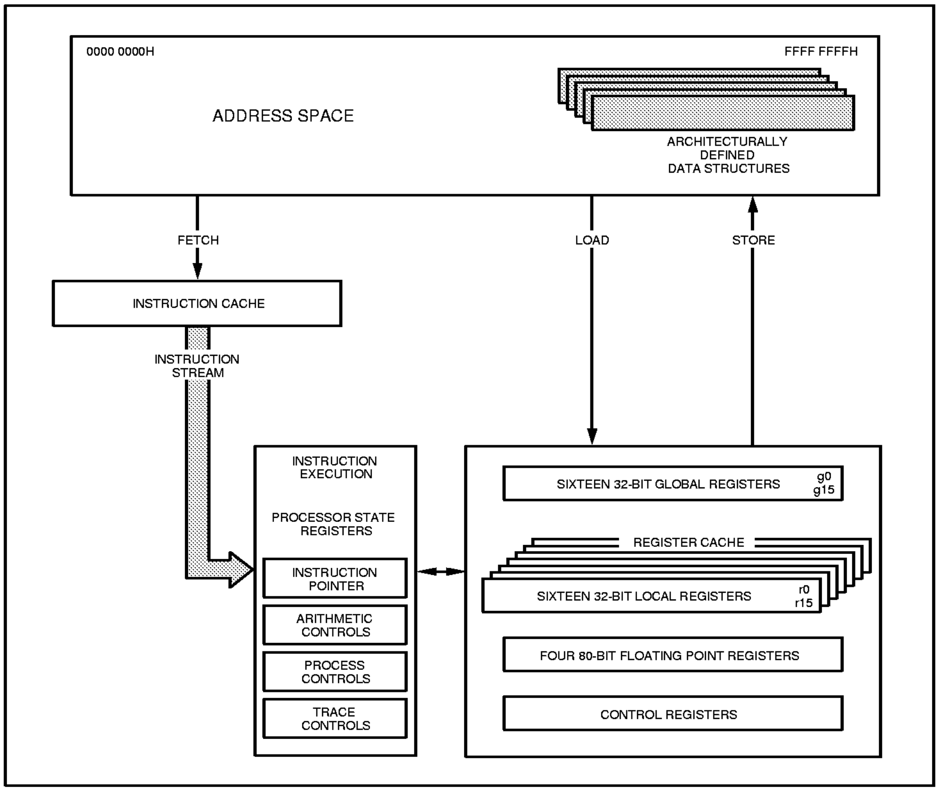


Figure 1. 80960MC Programming Environment

1.1 Key Performance Features

The 80960 architecture is based on the most recent advances in microprocessor technology and is grounded in Intel's long experience in the design and manufacture of embedded microprocessors. Many features contribute to the 80960MC's exceptional performance:

1. **Large Register Set.** Having a large number of registers reduces the number of times that a processor needs to access memory. Modern compilers can take advantage of this feature to optimize execution speed. For maximum flexibility, the 80960MC provides thirty-two 32-bit registers. (See Figure 2.)
2. **Fast Instruction Execution.** Simple functions make up the bulk of instructions in most programs so that execution speed can be improved by ensuring that these core instructions are executed as quickly as possible. The most frequently executed instructions such as register-register moves, add/subtract, logical operations and shifts execute in one to two cycles. (Table 1 contains a list of instructions.)
3. **Load/Store Architecture.** One way to improve execution speed is to reduce the number of times that the processor must access memory to perform an operation. As with other processors based on RISC technology, the 80960MC has a Load/Store architecture. As such, only the LOAD and STORE instructions reference memory; all other instructions operate on registers. This type of architecture simplifies instruction decoding and is used in combination with other techniques to increase parallelism.
4. **Simple Instruction Formats.** All instructions in the 80960MC are 32 bits long and must be aligned on word boundaries. This alignment makes it possible to eliminate the instruction alignment stage in the pipeline. To simplify the instruction decoder, there are only five instruction formats; each instruction uses only one format. (See Figure 3.)
5. **Overlapped Instruction Execution.** Load operations allow execution of subsequent instructions to continue before the data has been returned from memory, so that these instructions can overlap the load. The 80960MC manages this process transparently to software through the use of a register scoreboard. Conditional instructions also make use of a scoreboard so that subsequent unrelated instructions may be executed while the conditional instruction is pending.
6. **Integer Execution Optimization.** When the result of an arithmetic execution is used as an operand in a subsequent calculation, the value is sent immediately to its destination register. Yet at the same time, the value is put on a bypass path to the ALU, thereby saving the time that otherwise would be required to retrieve the value for the next operation.
7. **Bandwidth Optimizations.** The 80960MC gets optimal use of its memory bus bandwidth because the bus is tuned for use with the on-chip instruction cache: instruction cache line size matches the maximum burst size for instruction fetches. The 80960MC automatically fetches four words in a burst and stores them directly in the cache. Due to the size of the cache and the fact that it is continually filled in anticipation of needed instructions in the program flow, the 80960MC is relatively insensitive to memory wait states. The benefit is that the 80960MC delivers outstanding performance even with a low cost memory system.
8. **Cache Bypass.** When a cache miss occurs, the processor fetches the needed instruction then sends it on to the instruction decoder at the same time it updates the cache. Thus, no extra time is spent to load and read the cache.

Table 1. 80960MC Instruction Set

Data Movement	Process Management	Floating Point	Logical
Load Store Move Load Address Load Physical Address	Schedule Process Saves Process Resume Process Load Process Time Modify Process Controls Wait Conditional Wait Signal Receive Conditional Receive Send Send Service Atomic Add Atomic Modify	Add Subtract Multiply Divide Remainder Scale Round Square Root Sine Cosine Tangent Arctangent Log Log Binary Log Natural Exponent Classify Copy Real Extended Compare	And Not And And Not Or Exclusive Or Not Or Or Not Nor Exclusive Nor Not Nand Rotate
Comparison	**Branch**	**Bit and Bit Field**	**String**
Compare Conditional Compare Compare and Increment Compare and Decrement	Unconditional Branch Conditional Branch Compare and Branch	Set Bit Clear Bit Not Bit Check Bit Alter Bit Scan For Bit Scan Over Bit Extract Modify	Move String Move Quick String Fill String Compare String Scan Byte for Equal
Conversion	**Decimal**	**Call/Return**	**Arithmetic**
Convert Real to Integer Convert Integer to Real	Move Add with Carry Subtract with Carry	Call Call Extended Call System Return Branch and Link	Add Subtract Multiply Divide Remainder Modulo Shift
Fault	**Debug**	**Miscellaneous**	
Conditional Fault Synchronize Faults	Modify Trace Controls Mark Force Mark	Flush Local Registers Inspect Access Modify Arithmetic Controls Test Condition Code	

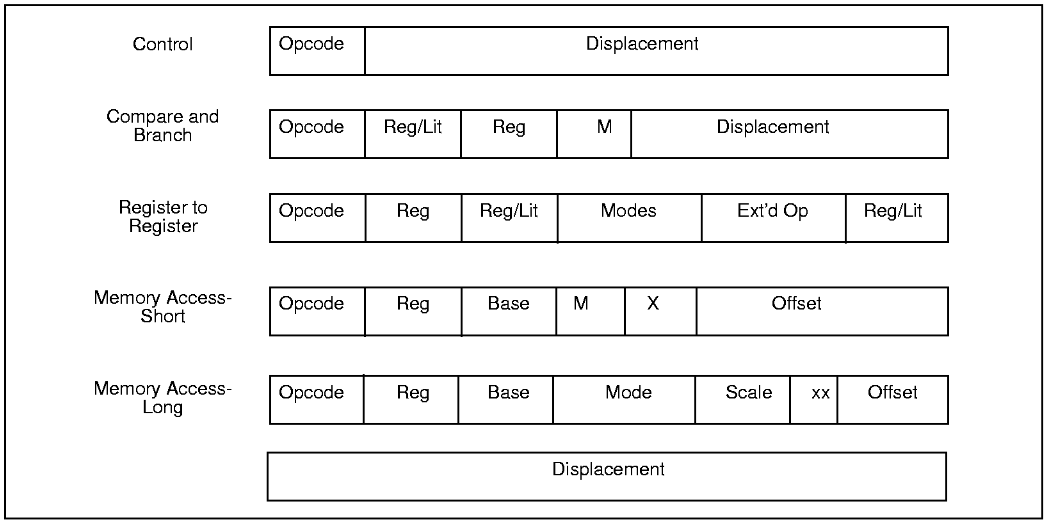


Figure 2. Instruction Formats

1.1.1 Memory Space And Addressing Modes

The 80960MC allows each task (process) to address a logical memory space of up to 4 Gbytes. Each task's address space is divided into four 1 Gbyte regions and each region can be mapped to physical addresses by zero, one, or two levels of page tables. The region with the highest addresses (Region 3) is common to all tasks.

In keeping with RISC design principles, the number of addressing modes is minimal yet includes all those necessary to ensure efficient execution of high-level languages such as Ada, C, and Fortran. Table 2 lists the memory accessing modes.

Table 2. Memory Addressing Modes

• 12-Bit Offset
• 32-Bit Offset
• Register-Indirect
• Register + 12-Bit Offset
• Register + 32-Bit Offset
• Register + (Index-Register x Scale-Factor)
• Register x Scale Factor + 32-Bit Displacement
• Register + (Index-Register x Scale-Factor) + 32-Bit Displacement
• Scale-Factor is 1, 2, 4, 8 or 16

1.1.2 Data Types

The 80960MC recognizes the following data types:

Numeric:

- 8-, 16-, 32- and 64-bit ordinals
- 8-, 16-, 32- and 64-bit integers
- 32-, 64- and 80-bit real numbers

Non-Numeric:

- Bit
- Bit Field
- Triple Word (96 bits)
- Quad-Word (128 bits)

1.1.3 Large Register Set

The 80960MC programming environment includes a large number of registers. 36 registers are available at any time; this greatly reduces the number of memory accesses required to perform algorithms, which leads to greater instruction processing speed.

Two types of general-purpose registers are available: local and global. The 20 global registers consist of sixteen 32-bit registers (G0 though G15) and four 80-bit registers (FP0 through FP3). These registers perform the same function as the general-purpose registers provided in other popular microprocessors. The term *global* refers to the fact that these registers retain their contents across procedure calls.

The local registers are procedure-specific. For each procedure call, the 80960MC allocates 16 local registers (R0 through R15). Each local register is 32 bits wide. Any register can also be used for floating-point operations; the 80-bit floating-point registers are provided for extended precision.

1.1.4 Multiple Register Sets

To further increase the efficiency of the register set, multiple sets of local registers are stored on-chip (See Figure 4). This cache holds up to four local register frames, which means that up to three procedure calls can be made without having to access the procedure stack resident in memory.

Although programs may have procedure calls nested many calls deep, a program typically oscillates back and forth between only two to three levels. As a result, with four stack frames in the cache, the probability of having a free frame available on the cache when a call is made is very high. Runs of representative C-language programs show that 80% of the calls are handled without needing to access memory.

When four or more procedures are active and a new procedure is called, the 80960MC moves the oldest local register set in the stack-frame cache to a procedure stack in memory to make room for a new set of registers. Global register G15 is the frame pointer (FP) to the procedure stack.

Global registers are not exchanged on a procedure call, but retain their contents, making them available to all procedures for fast parameter passing.

1.1.5 Instruction Cache

To further reduce memory accesses, the 80960MC includes a 512-byte on-chip instruction cache. The instruction cache is based on the concept of *locality of reference*; most programs are typically not executed in a steady stream but consist of many branches, loops and procedure calls that lead to jumping back and forth in the same small section of code. Thus, by maintaining a block of instructions in cache, the number of memory references required to read instructions into the processor is greatly reduced.

To load the instruction cache, instructions are fetched in 16-byte blocks; up to four instructions can be fetched at one time. An efficient prefetch algorithm increases the probability that an instruction is already in the cache when it is needed.

Code for small loops often fits entirely within the cache, leading to an increase in processing speed since further memory references might not be necessary until the program exits the loop. Similarly, when calling short procedures, the code for the calling procedure is likely to remain in the cache so it is there on the procedure's return.

1.1.6 Register Scoreboarding

The instruction decoder is optimized in several ways. One optimization method is the ability to overlap instructions by using *register scoreboarding.*

Register scoreboarding occurs when a LOAD moves a variable from memory into a register. When the instruction initiates, a scoreboard bit on the target register is set. Once the register is loaded, the bit is reset. In between, any reference to the register contents is accompanied by a test of the scoreboard bit to ensure that the load has completed before processing continues. Since the processor does not need to wait for the LOAD to complete, it can execute additional instructions placed between the LOAD and the instruction that uses the register contents, as shown in the following example:

```
ld data_2, r4
ld data_2, r5
Unrelated instruction
Unrelated instruction
add R4, R5, R6
```

In essence, the two unrelated instructions between LOAD and ADD are executed "for free" (i.e., take no apparent time to execute) because they are executed while the register is being loaded. Up to three load instructions can be pending at one time with three corresponding scoreboard bits set. By exploiting this feature, system programmers and compiler writers have a useful tool for optimizing execution speed.

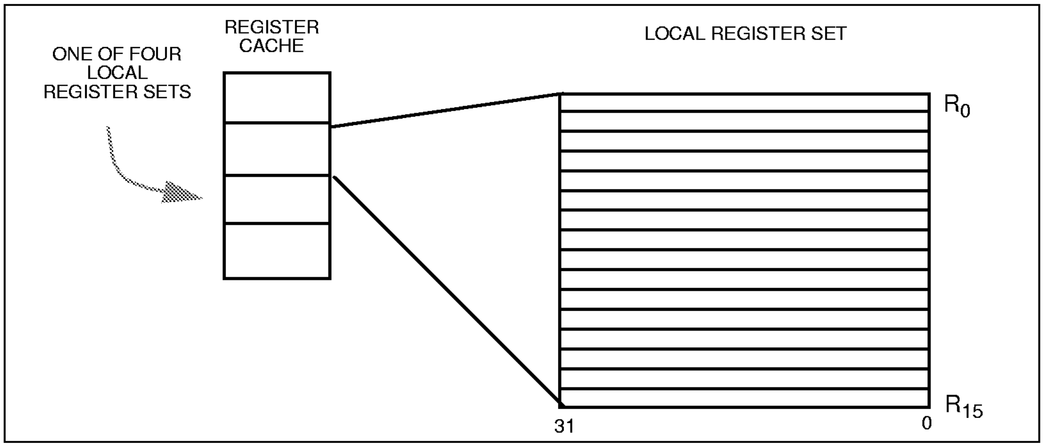


Figure 3. Multiple Register Sets Are Stored On-Chip

1.1.7 Memory Management and Protection

The 80960MC is ideal for multitasking applications that require software protection and a large address space. To ensure the highest level of performance possible, the memory management unit (MMU) and translation look-aside buffer (TLB) are contained on-chip.

The 80960MC supports a conventional form of demand-paged virtual memory in which the address space is divided into 4-Kbyte pages. Studies indicate that a 4-Kbyte page is the optimum size for a broad range of applications.

Each page table entry includes a 2-bit page rights field that specifies whether the page is a no-access, read-only, or read-write page. This field is interpreted differently depending on whether the current task (process) is executing in user or supervisor mode, as shown below:

Rights	User	Supervisor
00	No Access	Read-Only
01	No Access	Read-Write
10	Read-Only	Read-Write
11	Read-Write	Read-Write

1.1.8 Floating-Point Arithmetic

In the 80960MC, floating-point arithmetic is an integral part of the architecture. Having the floating-point unit integrated on-chip provides two advantages. First, it improves the performance of the chip for floating-point applications, since no additional bus overhead is associated with floating-point calculations, thereby leaving more time for other bus operations such as I/O. Second, the cost of using floating-point operations is reduced because a separate coprocessor chip is not required.

The 80960MC floating-point (real-number) data types include single-precision (32-bit), double-precision (64-bit) and extended precision (80-bit) floating-point numbers. Any registers may be used to execute floating-point operations.

The processor provides hardware support for both mandatory and recommended portions of IEEE Standard 754 for floating-point arithmetic, including all arithmetic, exponential, logarithmic and other transcendental functions. Table 3 shows execution times for some representative instructions.

Table 3. Sample Floating-Point Execution Times (µs) at 25 MHz

Function	32-Bit	64-Bit
Add	0.4	0.5
Subtract	0.4	0.5
Multiply	0.7	1.3
Divide	1.3	2.9
Square Root	3.7	3.9
Arctangent	10.1	13.1
Exponent	11.3	12.5
Sine	15.2	16.6
Cosine	15.2	16.6

1.1.9 Multitasking Support

Multitasking programs commonly involve the monitoring and control of an external operation, such as the activities of a process controller or the movements of a machine tool. These programs generally consist of a number of processes that run independently of one another, but share a common database or pass data among themselves.

The 80960MC offers several hardware functions designed to support multitasking systems. One unique feature, called self-dispatching, allows a processor to switch itself automatically among scheduled tasks. When self-dispatching is used, all the operating system is required to do is place the task in the scheduling queue.

When the processor becomes available, it dispatches the task from the beginning of the queue and then executes it until it becomes blocked, interrupted, or until its time-slice expires. It then returns the task to the end of the queue (i.e., automatically reschedules it) and dispatches the next ready task. During these operations, no communication between the processor and the operating system is necessary until the running task is complete or an interrupt is issued.

1.1.10 Synchronization and Communication

The 80960MC also offers instructions to set up and test semaphores to ensure that concurrent tasks remain synchronized and no data inconsistency results. Special data structures, known as communication ports, provide the means for exchanging parameters and data structures. Transmission of information by means of communication ports is asynchronous and automatically buffered by the processor.

Communication between tasks by means of ports can be carried out independently of the operating system. Once the ports have been set up by the programmer, the processor handles the message passing automatically.

1.1.11 High Bandwidth Local Bus

The 80960MC CPU resides on a high-bandwidth address/data bus known as the local bus (L-Bus). The L-Bus provides a direct communication path between the processor and the memory and I/O subsystem interfaces. The processor uses the L-Bus to fetch instructions, manipulate memory and respond to interrupts. L-Bus features include:

- 32-bit multiplexed address/data path
- Four-word burst capability which allows transfers from 1 to 16 bytes at a time
- High bandwidth reads and writes with 66.7 MBytes/s burst (at 25 MHz)
- Special signal to indicate whether a memory transaction can be cached

Table 4 defines L-bus signal names and functions; Table 5 defines other component-support signals such as interrupt lines.

1.1.12 Multiple Processor Support

One means of increasing the processing power of a system is to run two or more processors in parallel. Since microprocessors are not generally designed to run in tandem with other processors, designing such a system is usually difficult and costly.

The 80960MC solves this problem by offering a number of functions to coordinate the actions of multiple processors. First, messages can be passed between processors to initiate actions such as flushing a cache, stopping or starting another processor, or preempting a task. The messages are passed on the bus and allow multiple processors to run together smoothly, with rare need to lock the bus or memory.

Second, a set of synchronization instructions help maintain memory coherency. These instructions permit several processors to modify memory at the same time without inserting inaccuracies or ambiguities into shared data structures.

The self-dispatching mechanism — in addition to being used in single-processor systems — provides the means to increase the performance of a system merely by adding processors. Each processor can either work on the same pool of tasks (sharing the same queue with other processors) or can be restricted to its own queue.

When processors perform system operation, they synchronize themselves by using atomic operations and sending special messages between each other. In theory, changing the number of processors in a system does not require a software change. Software executes correctly regardless of the number of processors in the system; systems with more processors simply execute faster.

1.1.13 Interrupt Handling

The 80960MC can be interrupted in two ways: by the activation of one of four interrupt pins or by sending a message on the processor's data bus.

The 80960MC is unusual in that it automatically handles interrupts on a priority basis and can keep track of pending interrupts through its on-chip interrupt controller. Two of the interrupt pins can be configured to provide 8259A-style handshaking for expansion beyond four interrupt lines.

An interrupt message is made up of a vector number and an interrupt priority. When the interrupt priority is greater than that of the currently running task, the processor accepts the interrupt and uses the vector as an index into the interrupt table. When the priority of the interrupt message is below that of the current task, the processor saves the information in a section of the interrupt table reserved for pending interrupts.

1.1.14 Debug Features

The 80960MC has built-in debug capabilities, including two types of breakpoints and six trace modes. Debug features are controlled by two internal 32-bit registers: the Process-Controls Word and the Trace-Controls Word. By setting bits in these control words, a software debug monitor can closely control how the processor responds during program execution.

The 80960MC has both hardware and software breakpoints. It provides two hardware breakpoint registers on-chip which, by using a special command, can be set to any value. When the instruction pointer matches either breakpoint register value, the breakpoint handling routine is automatically called.

The 80960MC also provides software breakpoints through the use of two instructions: MARK and FMARK. These can be placed at any point in a program and cause the processor to halt execution at that point and call the breakpoint handling routine. The breakpoint mechanism is easy to use and provides a powerful debugging tool.

Tracing is available for instructions (single step execution), calls and returns and branching. Each trace type may be enabled separately by a special debug instruction. In each case, the 80960MC executes the instruction first and then calls a trace handling routine (usually part of a software debug monitor). Further program execution is halted until the routine completes, at which time execution resumes at the next instruction. The 80960MC'S tracing mechanisms, implemented completely in hardware, greatly simplify the task of software test and debug.

1.1.15 Fault Detection

The 80960MC has an automatic mechanism to handle faults. There are ten fault types include floating point, trace and arithmetic faults. When the processor detects a fault, it automatically calls the appropriate fault handling routine and saves the current instruction pointer and necessary state information to make efficient recovery possible. The processor posts diagnostic information on the type of fault to a Fault Record. Like interrupt handling routines, fault handling routines are usually written to meet the needs of specific applications and are often included as part of the operating system or kernel.

For each of the ten fault types, numerous subtypes provide specific information about a fault. For example, a floating point fault may have the subtype set to an Overflow or Zero-Divide fault. The fault handler can use this specific information to respond correctly to the fault.

1.1.16 Inter-Agent Communications (IAC)

To coordinate their actions, processors in a multiple processor system need a means for communicating with each other. The 80960MC does this through a mechanism known as "IACs" — Inter-Agent Communication messages.

IAC messages cause a variety of actions including starting and stopping processors, flushing instruction caches and TLBs, and sending interrupts to other processors in the system. The upper 16 Mbytes of the processor's physical memory space is reserved for sending and receiving IAC messages.

1.1.17 Built-in Testability

Upon reset, the 80960MC automatically conducts an exhaustive internal test of its major blocks of logic. Then, before executing its first instruction, it does a zero check sum on the first eight words in memory to ensure that the memory image was programmed correctly. When a problem is discovered at any point during the self-test, the 80960MC asserts its $\overline{\text{FAILURE}}$ pin and does not begin program execution. Self test takes approximately 47,000 cycles to complete.

System manufacturers can use the 80960MC's self-test feature during incoming parts inspection. No special diagnostic programs need to be written. The test is both thorough and fast. The self-test capability helps ensure that defective parts are discovered before systems are shipped and, once in the field, the self-test makes it easier to distinguish between problems caused by processor failure and problems resulting from other causes.

1.1.18 Compatibility with 80960K-Series

Application programs written for the 80960K-Series microprocessors can be run on the 80960MC without modification. The 80960K-Series instruction set forms the core of the 80960MC's instructions, so binary compatibility is assured.

1.1.19 CHMOS

The 80960MC is fabricated using Intel's CHMOS IV (Complementary High Speed Metal Oxide Semiconductor) process. The 80960MC is currently available at 25 MHz.

Table 4. 80960MC Pin Description: L-Bus Signals (Sheet 1 of 3)

NAME	TYPE	DESCRIPTION
CLK2	I	**SYSTEM CLOCK** provides the fundamental timing for 80960MC systems. It is divided by two inside the 80960MC to generate the internal processor clock. Refer to Figure 16, Processor Clock Pulse (CLK2) (pg. 21)
LAD31:0	I/O T.S.	**LOCAL ADDRESS / DATA BUS** carries 32-bit physical addresses and data to and from memory. During an address (T_a) cycle, bits 2-31 contain a physical word address (bits 0-1 indicate SIZE; see below). During a data (T_d) cycle, bits 0-31 contain read or write data. These pins float to a high impedance state when not active. Bits 0-1 comprise SIZE during a T_a cycle. SIZE specifies burst transfer size in words. **LAD1 LAD0** 0 0 1 Word 0 1 2 Words 1 0 3 Words 1 1 4 Words
$\overline{\text{ALE}}$	O T.S.	**ADDRESS LATCH ENABLE** indicates the transfer of a physical address. $\overline{\text{ALE}}$ is asserted during a T_a cycle and deasserted before the beginning of the T_d state. It is active LOW and floats to a high impedance state during a hold cycle (T_h).

I/O = Input/Output, O = Output, I = Input, O.D. = Open Drain, T.S. = Three-state

Table 4. 80960MC Pin Description: L-Bus Signals (Sheet 2 of 3)

NAME	TYPE	DESCRIPTION
$\overline{\text{ADS}}$	O O.D.	**ADDRESS/DATA STATUS** indicates an address state. $\overline{\text{ADS}}$ is asserted every T_a state and deasserted during the following T_d state. For a burst transaction, $\overline{\text{ADS}}$ is asserted again every T_d state where $\overline{\text{READY}}$ was asserted in the previous cycle.
W/$\overline{\text{R}}$	O O.D.	**WRITE/READ** specifies, during a T_a cycle, whether the operation is a write or read. It is latched on-chip and remains valid during T_d cycles.
DT/$\overline{\text{R}}$	O O.D.	**DATA TRANSMIT / RECEIVE** indicates the direction of data transfer to and from the L-Bus. It is low during T_a and T_d cycles for a read or interrupt acknowledgment; it is high during T_a and T_d cycles for a write. DT/$\overline{\text{R}}$ never changes state when $\overline{\text{DEN}}$ is asserted.
$\overline{\text{DEN}}$	O O.D.	**DATA ENABLE** (active low) enables data transceivers. The processor asserts DEN# during all T_d and T_w states. The DEN# line is an open drain-output of the 80960MC.
$\overline{\text{READY}}$	I	**READY** indicates that data on LAD lines can be sampled or removed. When $\overline{\text{READY}}$ is not asserted during a T_d cycle, the T_d cycle is extended to the next cycle by inserting a wait state (T_w) and $\overline{\text{ADS}}$ is not asserted in the next cycle.
$\overline{\text{LOCK}}$	I/O O.D.	**BUS LOCK** prevents bus masters from gaining control of the L-Bus during Read/Modify/Write (RMW) cycles. The processor or any bus agent may assert $\overline{\text{LOCK}}$. At the start of a RMW operation, the processor examines the $\overline{\text{LOCK}}$ pin. When the pin is already asserted, the processor waits until it is not asserted. When the pin is not asserted, the processor asserts $\overline{\text{LOCK}}$ during the T_a cycle of the read transaction. The processor deasserts $\overline{\text{LOCK}}$ in the T_a cycle of the write transaction. During the time $\overline{\text{LOCK}}$ is asserted, a bus agent can perform a normal read or write but not a RMW operation. The processor also asserts $\overline{\text{LOCK}}$ during interrupt-acknowledge transactions. Do not leave $\overline{\text{LOCK}}$ unconnected. It must be pulled high for the processor to function properly.
$\overline{\text{BE3:0}}$	O O.D.	**BYTE ENABLE LINES** specify the data bytes (up to four) on the bus which are used in the current bus cycle. $\overline{\text{BE3}}$ corresponds to LAD31:24; $\overline{\text{BE0}}$ corresponds to LAD7:0. The byte enables are provided in advance of data: Byte enables asserted during T_a specify the bytes of the first data word. Byte enables asserted during T_d specify the bytes of the next data word, if any (the word to be transmitted following the next assertion of $\overline{\text{READY}}$). Byte enables that occur during T_d cycles that precede the last assertion of $\overline{\text{READY}}$ are undefined. Byte enables are latched on-chip and remain constant from one T_d cycle to the next when $\overline{\text{READY}}$ is not asserted. For reads, byte enables specify the byte(s) that the processor actually uses. L-Bus agents are required to assert only adjacent byte enables (e.g., asserting just $\overline{\text{BE0}}$ and $\overline{\text{BE2}}$ is not permitted) and are required to assert at least one byte enable. Address bits A_0 and A_1 can be decoded externally from the byte enables.

I/O = Input/Output, O = Output, I = Input, O.D. = Open Drain, T.S. = Three-state

Table 4. 80960MC Pin Description: L-Bus Signals (Sheet 3 of 3)

NAME	TYPE	DESCRIPTION
HOLD/ HLDAR	I	**HOLD**: A request from an external bus master to acquire the bus. When the processor receives HOLD and grants bus control to another master, it floats its three-state bus lines and open-drain control lines, asserts HLDA and enters the T_h state. When HOLD deasserts, the processor deasserts HLDA and enters the T_i or T_a state. **HOLD ACKNOWLEDGE RECEIVED**: Indicates that the processor has acquired the bus. When the processor is initialized as the secondary bus master this input is interpreted as HLDAR. Refer to Figure 18, HOLD Timing (pg. 22).
HLDA/ HOLDR	O T.S.	**HOLD ACKNOWLEDGE**: Relinquishes control of the bus to another bus master. When the processor is initialized as the primary bus master this output is interpreted as HLDA. When HOLD is deasserted, the processor deasserts HLDA and goes to either the T_i or T_a state. **HOLD REQUEST**: Indicates a request to acquire the bus. When the processor is initialized as the secondary bus master this output is interpreted as HOLDR. Refer to Figure 18, HOLD Timing (pg. 22).
CACHE/ TAG	O T.S.	**CACHE** indicates when an access is cacheable during a T_a cycle. It is not asserted during any synchronous access, such as a synchronous load or move instruction used for sending an IAC message. The CACHE signal floats to a high impedance state when the processor is idle. **TAG** is an input/output signal that, during T_d and T_w cycles, identifies the contents of a 32-bit word as either data (TAG = 0) or an access descriptor (TAG = 1).

I/O = Input/Output, O = Output, I = Input, O.D. = Open Drain, T.S. = Three-state

Table 5. 80960MC Pin Description: Support Signals (Sheet 1 of 2)

NAME	TYPE	DESCRIPTION
$\overline{\text{BADAC}}$	I	**BAD ACCESS**, when asserted in the cycle following the one in which the last $\overline{\text{READY}}$ of a transaction is asserted, indicates that an unrecoverable error has occurred on the current bus transaction or that a synchronous load/store instruction has not been acknowledged. During system reset the $\overline{\text{BADAC}}$ signal is interpreted differently. When the signal is high, it indicates that this processor will perform system initialization. When low, another processor in the system will perform system initialization instead.
RESET	I	**RESET** clears the processor's internal logic and causes it to reinitialize. During RESET assertion, the input pins are ignored (except for $\overline{\text{BADAC}}$ and $\overline{\text{IAC}}/\overline{\text{INT}_0}$), the three-state output pins are placed in a high impedance state and other output pins are placed in their non-asserted states. RESET must be asserted for at least 41 CLK2 cycles for a predictable RESET. The HIGH to LOW transition of RESET should occur after the rising edge of both CLK2 and the external bus clock and before the next rising edge of CLK2. Refer to Figure 17, RESET Signal Timing (pg. 21).

I/O = Input/Output, O = Output, I = Input, O.D. = Open Drain, T.S. = Three-state

Table 5. 80960MC Pin Description: Support Signals (Sheet 2 of 2)

NAME	TYPE	DESCRIPTION
FAILURE	O O.D.	**INITIALIZATION FAILURE** indicates that the processor did not initialize correctly. After RESET deasserts and before the first bus transaction begins, $\overline{\text{FAILURE}}$ asserts while the processor performs a self-test. When the self-test completes successfully, then $\overline{\text{FAILURE}}$ deasserts. The processor then performs a zero checksum on the first eight words of memory. When it fails, $\overline{\text{FAILURE}}$ asserts for a second time and remains asserted. When it passes, system initialization continues and $\overline{\text{FAILURE}}$ remains deasserted.
$\overline{\text{IAC/INT}}_0$ LOCAL PROCESSOR NUMBER	I	**INTERAGENT COMMUNICATION REQUEST/INTERRUPT 0** indicates an IAC message or an interrupt is pending. The bus interrupt control register determines how the signal is interpreted. To signal an interrupt or IAC request in a synchronous system, this pin — as well as the other interrupt pins — must be enabled by being deasserted for at least one bus cycle and then asserted for at least one additional bus cycle. In an asynchronous system the pin must remain deasserted for at least two bus cycles and then asserted for at least two more bus cycles. **LOCAL PROCESSOR NUMBER** - this signal is interpreted differently during system reset. When the signal is a high voltage level it indicates that this processor is a primary bus master (local processor number = 0). When at a low voltage level it indicates that this processor is a secondary bus master (local processor number = 1).
INT_1	I	**INTERRUPT 1**, like $\overline{\text{INT}}_0$, provides direct interrupt signaling.
INT_2/INTR	I	**INTERRUPT2/INTERRUPT REQUEST**: The interrupt control register determines how this pin is interpreted. When INT_2, it has the same interpretation as the $\overline{\text{INT}}_0$ and INT_1 pins. When INTR, it is used to receive an interrupt request from an external interrupt controller.
$\overline{\text{INT}}_3/\overline{\text{INTA}}$	I/O O.D.	**INTERRUPT3/INTERRUPT ACKNOWLEDGE**: The bus interrupt control register determines how this pin is interpreted. When $\overline{\text{INT}}_3$, it has the same interpretation as the $\overline{\text{INT}}_0$, INT1 and INT2 pins. When $\overline{\text{INTA}}$, it is used as an output to control interrupt-acknowledge transactions. The $\overline{\text{INTA}}$ output is latched on-chip and remains valid during T_d cycles; as an output, it is open-drain.
N.C.	N/A	**NOT CONNECTED** indicates pins should not be connected. Never connect any pin marked N.C. as these pins may be reserved for factory use.

I/O = Input/Output, O = Output, I = Input, O.D. = Open Drain, T.S. = Three-state

2.0 ELECTRICAL SPECIFICATIONS

2.1 Power and Grounding

The 80960MC is implemented in CHMOS IV technology and therefore has modest power requirements. Its high clock frequency and numerous output buffers (address/data, control, error and arbitration signals) can cause power surges as multiple output buffers simultaneously drive new signal levels. For clean on-chip power distribution, V_{CC} and V_{SS} pins separately feed the device's functional units. Power and ground connections must be made to all 80960MC power and ground pins. On the circuit board, all V_{cc} pins must be strapped closely together, preferably on a power plane; all V_{ss} pins should be strapped together, preferably on a ground plane.

2.2 Power Decoupling Recommendations

Place a liberal amount of decoupling capacitance near the 80960MC. When driving the L-bus the processor can cause transient power surges, particularly when connected to a large capacitive load.

Low inductance capacitors and interconnects are recommended for best high frequency electrical performance. Inductance is reduced by shortening board traces between the processor and decoupling capacitors as much as possible.

2.3 Connection Recommendations

For reliable operation, always connect unused inputs to an appropriate signal level. In particular, when one or more interrupt lines are not used, they should be pulled up. No inputs should ever be left floating.

All open-drain outputs require a pull-up device. While in most cases a simple pull-up resistor is adequate, a network of pull-up and pull-down resistors biased to a valid V_{IH} (>3.0 V) and terminated in the characteristic impedance of the circuit board is recommended to limit noise and AC power consumption. Figure 5 and Figure 6 show recommended values for the resistor network for low and high current drive, assuming a characteristic impedance of 100 Ω. Terminating output signals in this fashion limits signal swing and reduces AC power consumption.

NOTE:
Do not connect external logic to pins marked N.C.

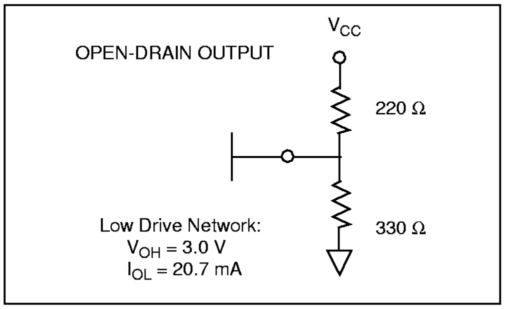


Figure 4. Connection Recommendations for Low Current Drive Network

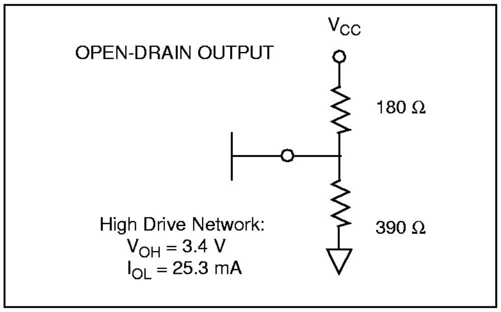


Figure 5. Connection Recommendations for High Current Drive Network

2.4 Characteristic Curves

Figure 7 shows typical supply current requirements over the operating temperature range of the processor at supply voltage (V_{CC}) of 5 V. Figure 8 and Figure 9 show the typical power supply current (I_{CC}) that the 80960MC requires at various operating frequencies when measured at three input voltage (V_{CC}) levels and two temperatures.

For a given output current (I_{OL}) the curve in Figure 10 shows the worst case output low voltage (V_{OL}). Figure 11 shows the typical capacitive derating curve for the 80960MC measured from 1.5V on the system clock (CLK) to 1.5V on the falling edge and 1.5V on the rising edge of the L-Bus address/data (LAD) signals.

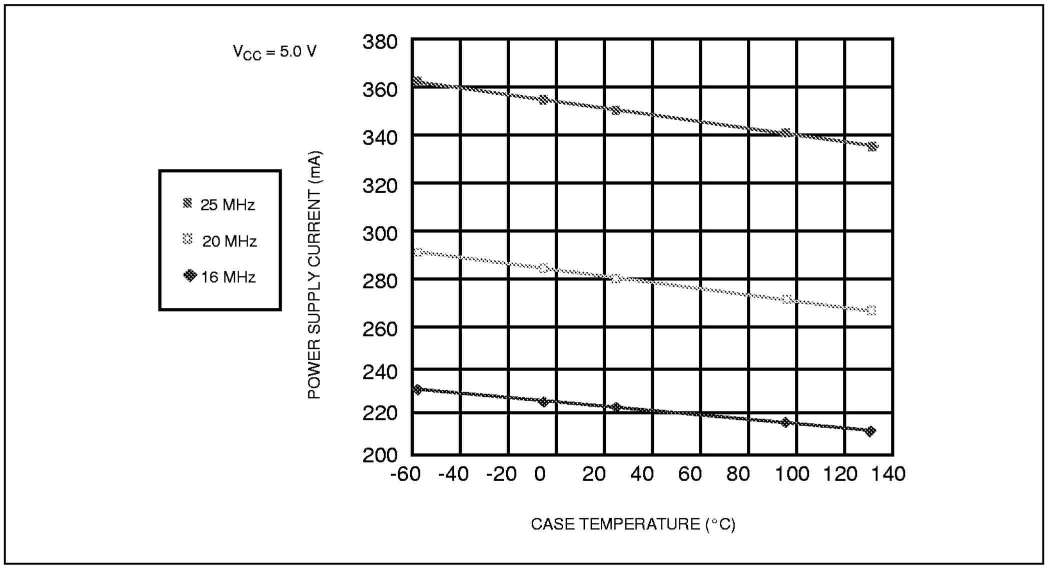


Figure 6. Typical Supply Current vs. Case Temperature

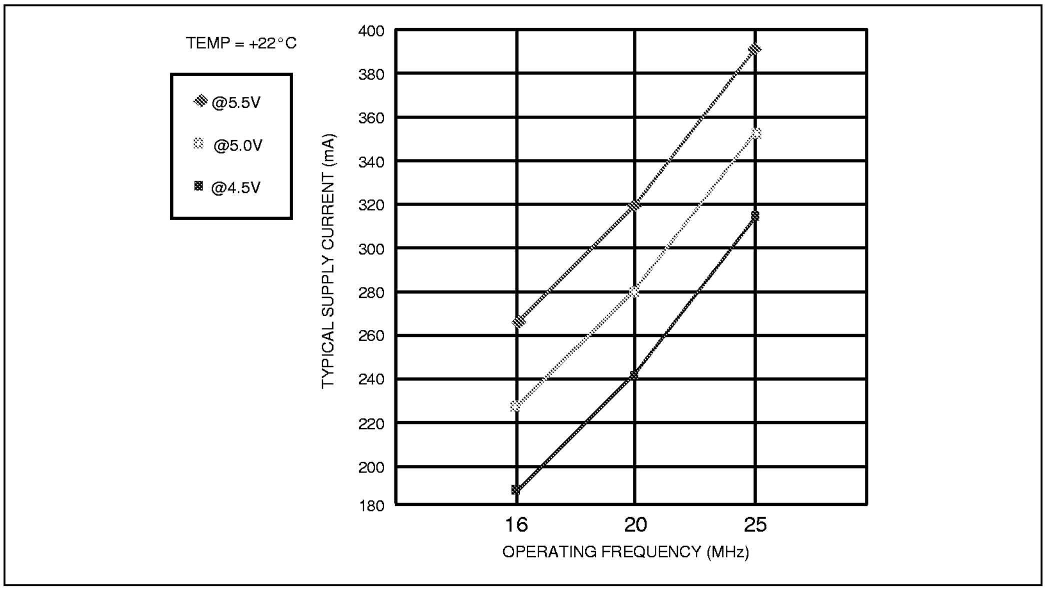


Figure 7. Typical Current vs. Frequency (Room Temp)

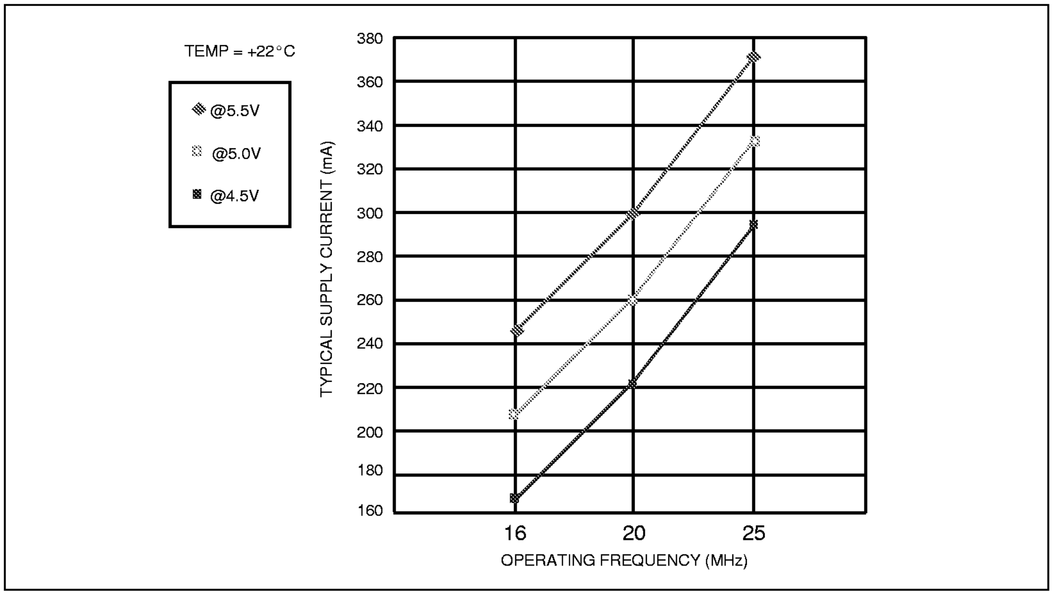


Figure 8. Typical Current vs. Frequency (Hot Temp)

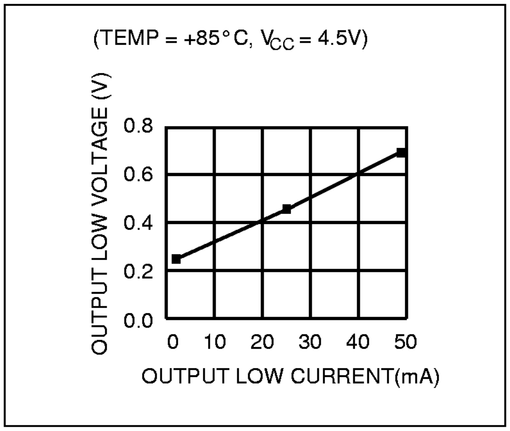


Figure 9. Worst-Case Voltage vs. Output Current on Open-Drain Pins

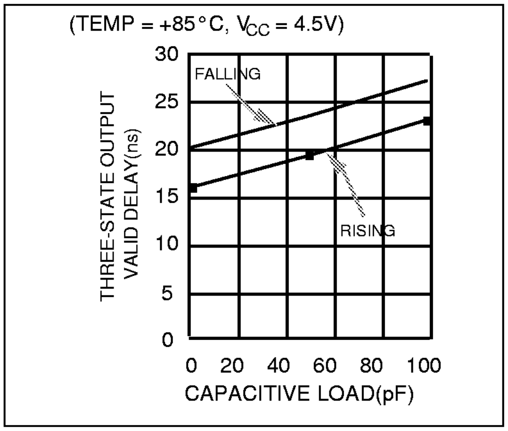


Figure 10. Capacitive Derating Curve

2.5 Test Load Circuit

Figure 12 illustrates the load circuit used to test the 80960MC's three-state pins; Figure 13 shows the load circuit used to test the open drain outputs. The open drain test uses an active load circuit in the form of a matched diode bridge. Since the open-drain outputs sink current, only the I_{OL} legs of the bridge are necessary and the I_{OH} legs are not used. When the 80960MC driver under test is turned off, the output pin is pulled up to V_{REF} (i.e., V_{OH}). Diode D_1 is turned off and the I_{OL} current source flows through diode D_2.

When the 80960MC open-drain driver under test is on, diode D_1 is also on and the voltage on the pin being tested drops to V_{OL}. Diode D_2 turns off and I_{OL} flows through diode D_1.

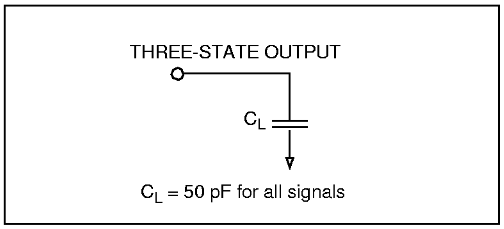


Figure 11. Test Load Circuit for Three-State Output Pins

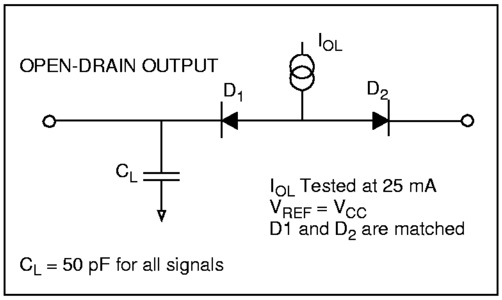


Figure 12. Test Load Circuit for Open-Drain Output Pins

2.6 Absolute Maximum Ratings

Operating Temperature (PGA) ...... 0° C to +85° C Case
Storage Temperature..................... –65° C to +150° C
Voltage on Any Pin .... –0.5 V to VCC +0.5 V
Power Dissipation .... 2.5 W (25 MHz)

NOTICE: This is a production data sheet. The specifications are subject to change without notice.

**WARNING: Stressing the device beyond the "Absolute Maximum Ratings" may cause permanent damage. These are stress ratings only. Operation beyond the "Operating Conditions" is not recommended and extended exposure beyond the "Operating Conditions" may affect device reliability.*

2.7 DC Characteristics

PGA: 80960MC (25 MHz) T_{CASE} = 0° C to +85° C, V_{CC} = 5V ± 5%

Table 6. DC Characteristics

Symbol	Parameter	Min	Max	Units	Notes
V_{IL}	Input Low Voltage	–0.3	+0.8	V	
V_{IH}	Input High Voltage	2.0	V_{CC} + 0.3	V	
V_{CL}	CLK2 Input Low Voltage	–0.3	+0.8	V	
V_{CH}	CLK2 Input High Voltage	0.55 V_{CC}	V_{CC} + 0.3	V	
V_{OL}	Output Low Voltage		0.45	V	(1,2)
V_{OH}	Output High Voltage	2.4		V	(3,4)
I_{CC}	Power Supply Current: 16 MHz 20 MHz 25 MHz		 315 360 420	 mA mA mA	 (5) (5) (5)
I_{LI}	Input Leakage Current		±15	µA	$0 \leq V_{IN} \leq V_{CC}$
I_{LO}	Output Leakage Current		±15	µA	$0.45 \leq V_O \leq V_{CC}$
C_{IN}	Input Capacitance		10	pF	f_C = 1 MHz (6)
C_O	Output Capacitance		12	pF	f_C = 1 MHz (6)
C_{CLK}	Clock Capacitance		10	pF	f_C = 1 MHz (6)

NOTES:

1. For three-state outputs, this parameter is measured at:
   Address/Data 4.0 mA
   Controls 5.0 mA
2. For open-drain outputs 25 mA
3. This parameter is measured at:
   Address/Data –1.0 mA
   Controls –0.9 mA
   $\overline{ALE}$ –5.0 mA
4. Not measured on open-drain outputs.
5. Measured at worst case frequency, V_{CC} and temperature, with device operating and outputs loaded to the test conditions in Figure 12 and Figure 13. Figure 7, Figure 8 and Figure 9 indicate typical values.
6. Input, output and clock capacitance are not tested.

2.8 AC Specifications

This section describes the AC specifications for the 80960MC pins. All input and output timings are specified relative to the 1.5 V level of the rising edge of CLK2. For output timings the specifications refer to the time it takes the signal to reach 1.5 V.

For input timings the specifications refer to the time at which the signal reaches (for input setup) or leaves (for hold time) the TTL levels of LOW (0.8 V) or HIGH (2.0 V). All AC testing should be done with input voltages of 0.4 V and 2.4 V, except for the clock (CLK2), which should be tested with input voltages of 0.45 V and 0.55 V_{CC}.

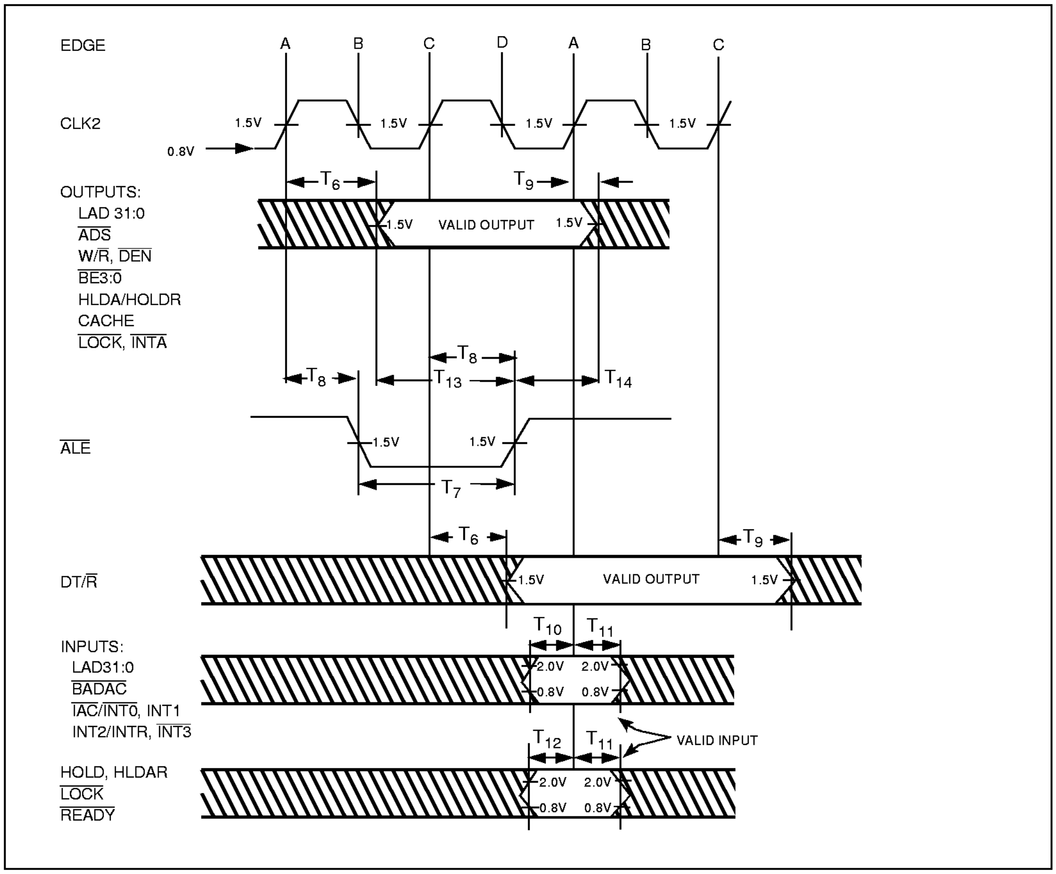


Figure 13. Drive Levels and Timing Relationships for 80960MC Signals

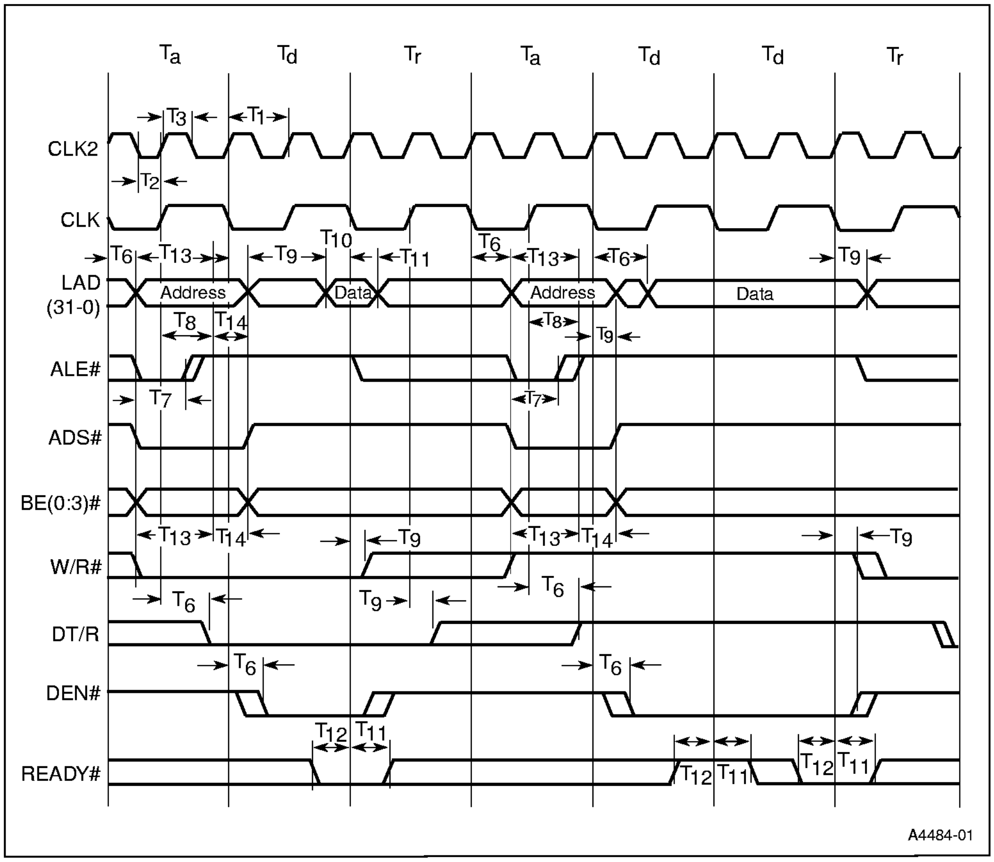


Figure 14. Timing Relationship of L-Bus Signals

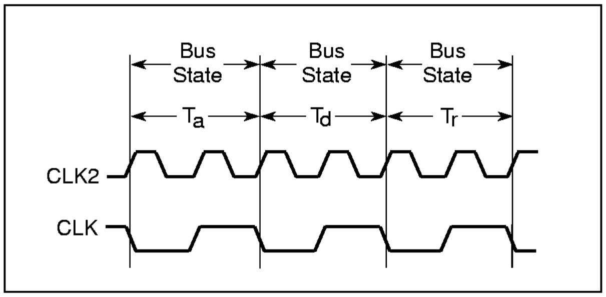


Figure 15. System and Processor Clock Relationship

Table 7. 80960MC AC Characteristics (25 MHz)

Symbol	Parameter	Min	Max	Units	Notes
Input Clock					
T_1	Processor Clock Period (CLK2)	20	125	ns	V_{IN} = 1.5V
T_2	Processor Clock Low Time (CLK2)	5		ns	V_{IL} = 10% Point = 1.2V
T_3	Processor Clock High Time (CLK2)	5		ns	V_{IH} = 90% Point = 0.1V + 0.5 V_{CC}
T_4	Processor Clock Fall Time (CLK2)		10	ns	V_{IN} = 90% Point to 10% Point (1)
T_5	Processor Clock Rise Time (CLK2)		10	ns	V_{IN} = 10% Point to 90% Point (1)
Synchronous Outputs					
T_6	Output Valid Delay	2	18	ns	
T_{6H}	HLDA Output Valid Delay	4	23	ns	
T_7	$\overline{\text{ALE}}$ Width	12		ns	
T_8	$\overline{\text{ALE}}$ Output Valid Delay	2	18	ns	
T_9	Output Float Delay	2	18	ns	(2)
T_{9H}	HLDA Output Float Delay	4	20	ns	(2)
Synchronous Inputs					
T_{10}	Input Setup 1	3		ns	(3)
T_{11}	Input Hold	5		ns	(3)
T_{11H}	HOLD Input Hold	4		ns	
T_{12}	Input Setup 2	7		ns	
T_{13}	Setup to $\overline{\text{ALE}}$ Inactive	8		ns	
T_{14}	Hold after $\overline{\text{ALE}}$ Inactive	8		ns	
T_{15}	Reset Hold	3		ns	
T_{16}	Reset Setup	5		ns	
T_{17}	Reset Width	820		ns	41 CLK2 Periods Minimum

NOTES:

1. Clock rise and fall times are not tested.
2. A float condition occurs when the maximum output current becomes less than I_{LO}. Float delay is not tested; however, it should not be longer than the valid delay.
3. LAD31:0, $\overline{\text{BADAC}}$, HOLD, $\overline{\text{LOCK}}$ and $\overline{\text{READY}}$ are synchronous inputs. $\overline{\text{IAC}}/\overline{\text{INT}}_0$, INT_1, $\text{INT}_2/\text{INT}_R$ and $\overline{\text{INT}}_3$ may be synchronous or asynchronous.

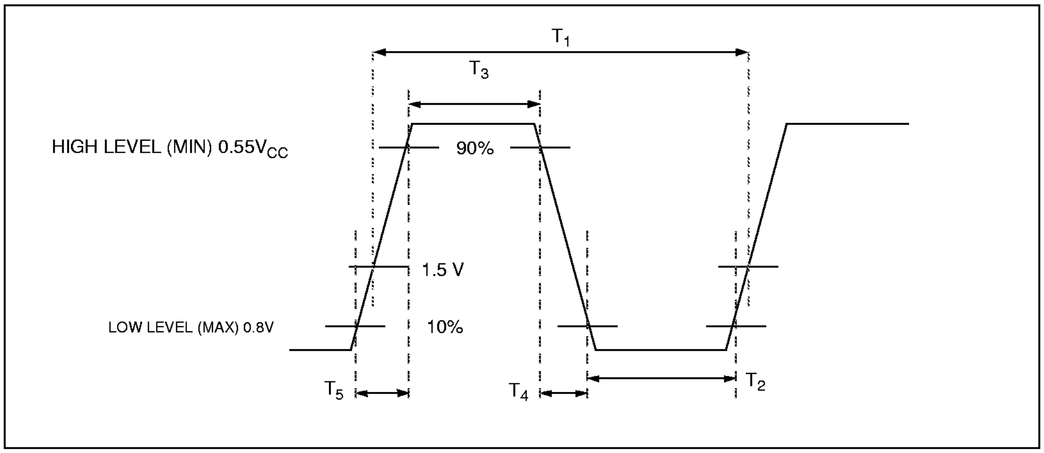


Figure 16. Processor Clock Pulse (CLK2)

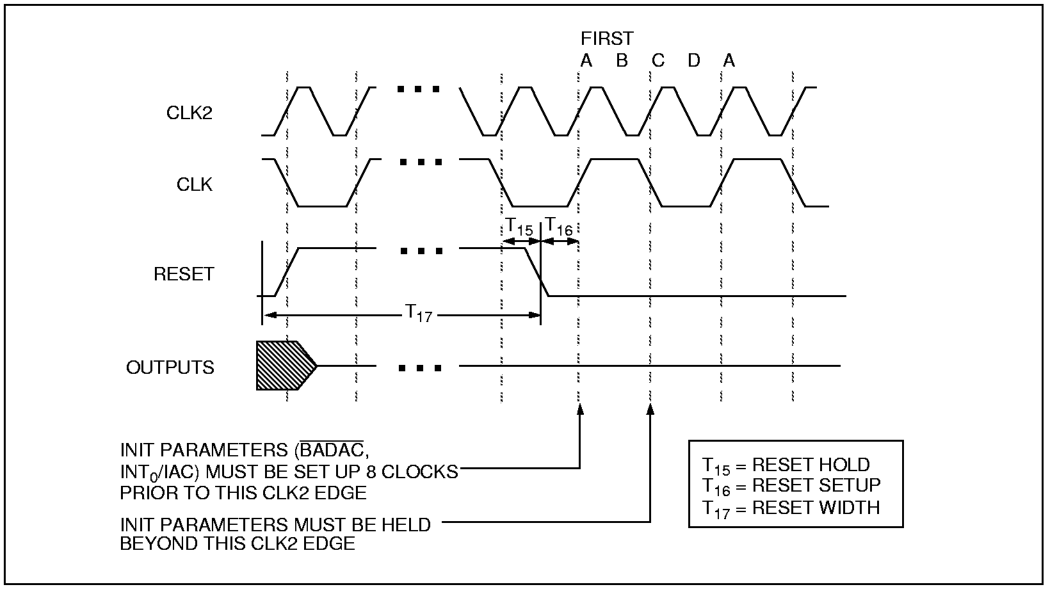


Figure 17. RESET Signal Timing

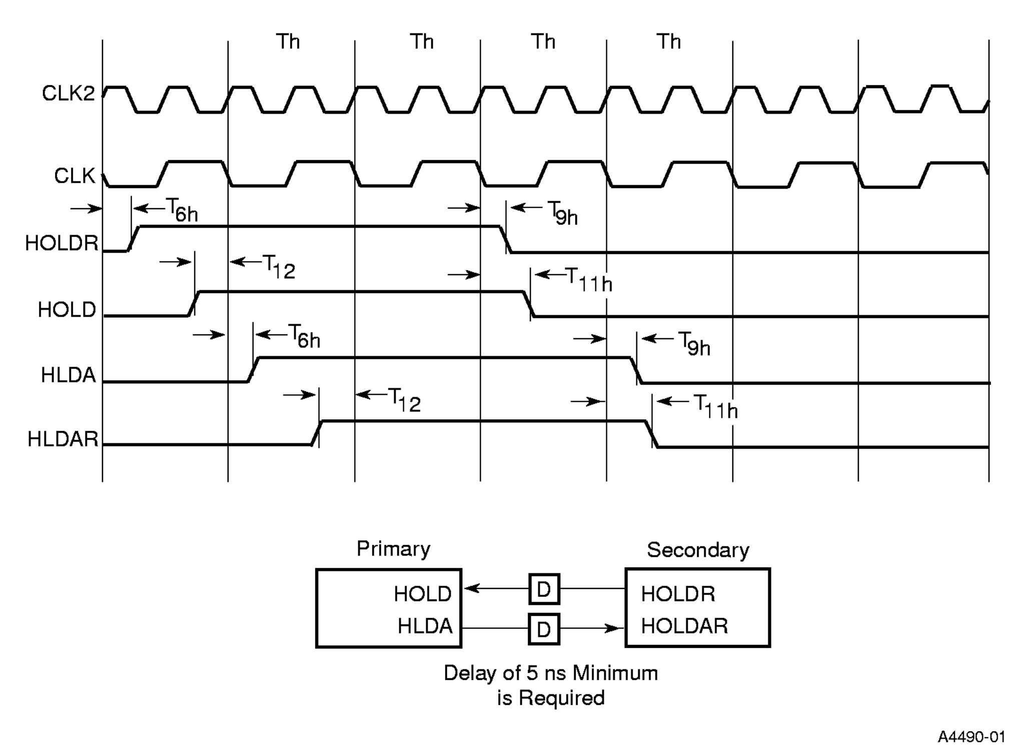


Figure 18. HOLD Timing

2.9 Design Considerations

Input hold times can be disregarded by the designer whenever the input is removed because a subsequent output from the processor is deasserted (e.g., $\overline{\text{DEN}}$ becomes deasserted).

In other words, whenever the processor generates an output that indicates a transition into a subsequent state, the processor must have sampled any inputs for the previous state.

Similarly, whenever the processor generates an output that indicates a transition into a subsequent state, any outputs that are specified to be three stated in this new state are guaranteed to be three stated.

3.0 MECHANICAL DATA

3.1 Packaging

The 80960MC is available in one package type: a 132-lead ceramic pin-grid array (PGA). Pins are arranged 0.100 inch (2.54 mm) center-to-center, in a 14 by 14 matrix, three rows around (see Figure 20). Dimensions for the PGA package type is given in the Intel *Packaging* handbook (Order #240800).

3.1.1 Pin Assignment

Figure 21 shows the view from the PGA bottom (pins facing up). Table 8 and Table 9 list the function of each PGA pin.

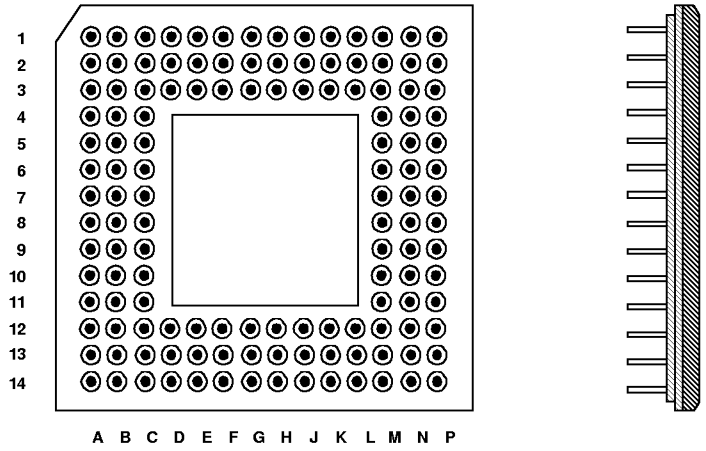


Figure 19. 132-Lead Pin-Grid Array (PGA) Package

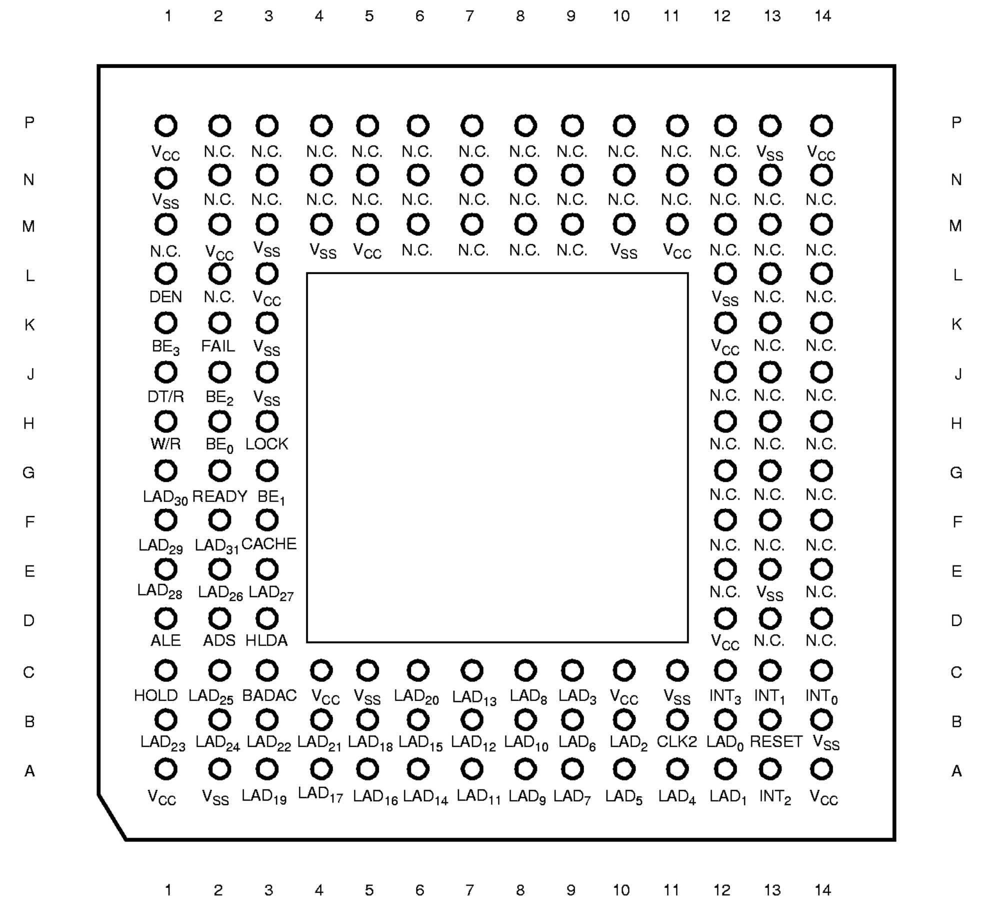
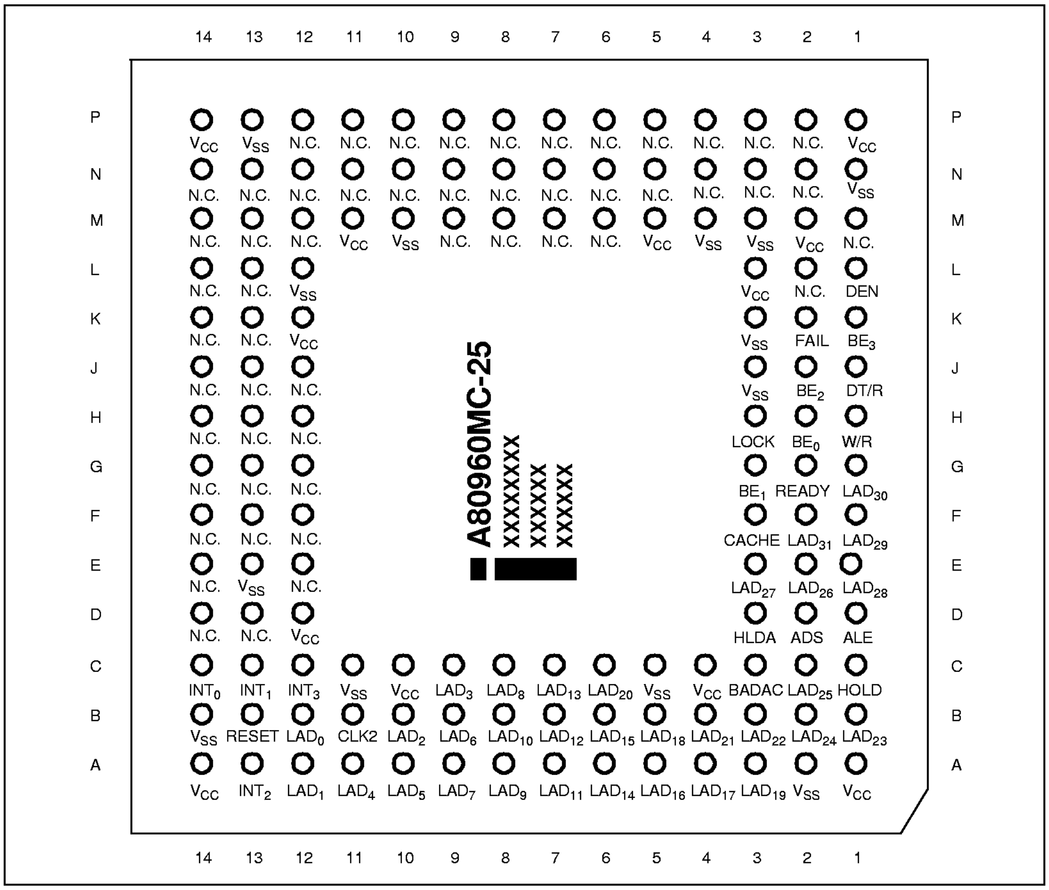


Figure 20. 80960MC PGA Pinout—View from Bottom (Pins Facing Up)



	14	13	12	11	10	9	8	7	6	5	4	3	2	1	
P	V_{CC}	V_{SS}	N.C.	N.C.	N.C.	N.C.	N.C.	N.C.	N.C.	N.C.	N.C.	N.C.	N.C.	V_{CC}	P
N	N.C.	N.C.	N.C.	N.C.	N.C.	N.C.	N.C.	N.C.	N.C.	N.C.	N.C.	N.C.	N.C.	V_{SS}	N
M	N.C.	N.C.	N.C.	V_{CC}	V_{SS}	N.C.	N.C.	N.C.	N.C.	V_{CC}	V_{SS}	V_{SS}	V_{CC}	N.C.	M
L	N.C.	N.C.	V_{SS}									V_{CC}	N.C.	DEN	L
K	N.C.	N.C.	V_{CC}									V_{SS}	FAIL	BE$_3$	K
J	N.C.	N.C.	N.C.									V_{SS}	BE$_2$	DT/R	J
H	N.C.	N.C.	N.C.									LOCK	BE$_0$	W/R	H
G	N.C.	N.C.	N.C.									BE$_1$	READY	LAD$_{30}$	G
F	N.C.	N.C.	N.C.									CACHE	LAD$_{31}$	LAD$_{29}$	F
E	N.C.	V_{SS}	N.C.									LAD$_{27}$	LAD$_{26}$	LAD$_{28}$	E
D	N.C.	N.C.	V_{CC}									HLDA	ADS	ALE	D
C	INT$_0$	INT$_1$	INT$_3$	V_{SS}	V_{CC}	LAD$_3$	LAD$_8$	LAD$_{13}$	LAD$_{20}$	V_{SS}	V_{CC}	BADAC	LAD$_{25}$	HOLD	C
B	V_{SS}	RESET	LAD$_0$	CLK2	LAD$_2$	LAD$_6$	LAD$_{10}$	LAD$_{12}$	LAD$_{15}$	LAD$_{18}$	LAD$_{21}$	LAD$_{22}$	LAD$_{24}$	LAD$_{23}$	B
A	V_{CC}	INT$_2$	LAD$_1$	LAD$_4$	LAD$_5$	LAD$_7$	LAD$_9$	LAD$_{11}$	LAD$_{14}$	LAD$_{16}$	LAD$_{17}$	LAD$_{19}$	V_{SS}	V_{CC}	A
	14	13	12	11	10	9	8	7	6	5	4	3	2	1	

Figure 21. 80960MC PGA Pinout—View from Top (Pins Facing Down)

3.2 Pinout

Table 8. 80960MC PGA Pinout — In Pin Order

Pin	Signal	Pin	Signal	Pin	Signal	Pin	Signal
A1	V_{CC}	C6	LAD_{20}	H1	$W/\overline{R}$	M10	V_{SS}
A2	V_{SS}	C7	LAD_{13}	H2	$\overline{BE}_0$	M11	V_{CC}
A3	LAD_{19}	C8	LAD_8	H3	$\overline{LOCK}$	M12	N.C.
A4	LAD_{17}	C9	LAD_3	H12	N.C.	M13	N.C.
A5	LAD_{16}	C10	V_{CC}	H13	N.C.	M14	N.C.
A6	LAD_{14}	C11	V_{SS}	H14	N.C.	N1	V_{SS}
A7	LAD_{11}	C12	$\overline{INT}_3/\overline{INTA}$	J1	$DT/\overline{R}$	N2	N.C.
A8	LAD_9	C13	INT_1	J2	$\overline{BE}_2$	N3	N.C.
A9	LAD_7	C14	$\overline{IAC}/\overline{INT}_0$	J3	V_{SS}	N4	N.C.
A10	LAD_5	D1	$\overline{ALE}$	J12	N.C.	N5	N.C.
A11	LAD_4	D2	$\overline{ADS}$	J13	N.C.	N6	N.C.
A12	LAD_1	D3	HLDA/HOLDR	J14	N.C.	N7	N.C.
A13	$INT_2/INTR$	D12	V_{CC}	K1	$\overline{BE}_3$	N8	N.C.
A14	V_{CC}	D13	N.C.	K2	$\overline{FAILURE}$	N9	N.C.
B1	LAD_{23}	D14	N.C.	K3	V_{SS}	N10	N.C.
B2	LAD_{24}	E1	LAD_{28}	K12	V_{CC}	N11	N.C.
B3	LAD_{22}	E2	LAD_{26}	K13	N.C.	N12	N.C.
B4	LAD_{21}	E3	LAD_{27}	K14	N.C.	N13	N.C.
B5	LAD_{18}	E12	N.C.	L1	$\overline{DEN}$	N14	N.C.
B6	LAD_{15}	E13	V_{SS}	L2	N.C.	P1	V_{CC}
B7	LAD_{12}	E14	N.C.	L3	V_{CC}	P2	N.C.
B8	LAD_{10}	F1	LAD_{29}	L12	V_{SS}	P3	N.C.
B9	LAD_6	F2	LAD_{31}	L13	N.C.	P4	N.C.
B10	LAD_2	F3	CACHE/TAG	L14	N.C.	P5	N.C.
B11	CLK2	F12	N.C.	M1	N.C.	P6	N.C.
B12	LAD_0	F13	N.C.	M2	V_{CC}	P7	N.C.
B13	RESET	F14	N.C.	M3	V_{SS}	P8	N.C.
B14	V_{SS}	G1	LAD_{30}	M4	V_{SS}	P9	N.C.
C1	HOLD/HLDAR	G2	$\overline{READY}$	M5	V_{CC}	P10	N.C.
C2	LAD_{25}	G3	$\overline{BE}_1$	M6	N.C.	P11	N.C.
C3	$\overline{BADAC}$	G12	N.C.	M7	N.C.	P12	N.C.
C4	V_{CC}	G13	N.C.	M8	N.C.	P13	V_{SS}
C5	V_{SS}	G14	N.C.	M9	N.C.	P14	V_{CC}

NOTES: Do not connect any external logic to any pins marked N.C.

Table 9. 80960MC PGA Pinout — In Signal Order

Signal	Pin	Signal	Pin	Signal	Pin	Signal	Pin
$\overline{ADS}$	D2	LAD$_{15}$	B6	N.C.	J14	N.C.	P9
$\overline{ALE}$	D1	LAD$_{16}$	A5	N.C.	K13	N.C.	P10
$\overline{BADAC}$	C3	LAD$_{17}$	A4	N.C.	K14	N.C.	P11
$\overline{BE}_0$	H2	LAD$_{18}$	B5	N.C.	L13	N.C.	P12
$\overline{BE}_1$	G3	LAD$_{19}$	A3	N.C.	L14	N.C.	L2
$\overline{BE}_2$	J2	LAD$_{20}$	C6	N.C.	M1	$\overline{READY}$	G2
$\overline{BE}_3$	K1	LAD$_{21}$	B4	N.C.	M6	RESET	B13
CACHE	F3	LAD$_{22}$	B3	N.C.	M7	V_{CC}	A1
CLK2	B11	LAD$_{23}$	B1	N.C.	M8	V_{CC}	A14
$\overline{DEN}$	L1	LAD$_{24}$	B2	N.C.	M9	V_{CC}	C4
DT/$\overline{R}$	J1	LAD$_{25}$	C2	N.C.	M12	V_{CC}	C10
$\overline{FAILURE}$	K2	LAD$_{26}$	E2	N.C.	M13	V_{CC}	D12
HLDA/HOLDR	D3	LAD$_{27}$	E3	N.C.	M14	V_{CC}	K12
HOLD/HLDAR	C1	LAD$_{28}$	E1	N.C.	N2	V_{CC}	L3
$\overline{IAC}/\overline{INT}_0$	C14	LAD$_{29}$	F1	N.C.	N3	V_{CC}	M2
INT$_1$	C13	LAD$_{30}$	G1	N.C.	N4	V_{CC}	M5
INT$_2$/INTR	A13	LAD$_{31}$	F2	N.C.	N5	V_{CC}	M11
$\overline{INT}_3/\overline{INTA}$	C12	$\overline{LOCK}$	H3	N.C.	N6	V_{CC}	P1
LAD$_0$	B12	N.C.	D13	N.C.	N7	V_{CC}	P14
LAD$_1$	A12	N.C.	D14	N.C.	N8	V_{SS}	A2
LAD$_2$	B10	N.C.	E12	N.C.	N9	V_{SS}	B14
LAD$_3$	C9	N.C.	E14	N.C.	N10	V_{SS}	C5
LAD$_4$	A11	N.C.	F12	N.C.	N11	V_{SS}	C11
LAD$_5$	A10	N.C.	F13	N.C.	N12	V_{SS}	E11
LAD$_6$	B9	N.C.	F14	N.C.	N13	V_{SS}	J3
LAD$_7$	A9	N.C.	G12	N.C.	N14	V_{SS}	K3
LAD$_8$	C8	N.C.	G13	N.C.	P2	V_{SS}	L12
LAD$_9$	A8	N.C.	G14	N.C.	P3	V_{SS}	M3
LAD$_{10}$	B8	N.C.	H12	N.C.	P4	V_{SS}	M4
LAD$_{11}$	A7	N.C.	H13	N.C.	P5	V_{SS}	M10
LAD$_{12}$	B7	N.C.	H14	N.C.	P6	V_{SS}	N1
LAD$_{13}$	C7	N.C.	J12	N.C.	P7	V_{SS}	P13
LAD$_{14}$	A6	N.C.	J13	N.C.	P8	W/$\overline{R}$	H1

NOTE: Do not connect external logic to any pins marked N.C.

3.3 Package Thermal Specification

The 80960MC is specified for operation when case temperature is within the range 0°C to 85°C (PGA). Measure case temperature at the top center of the package. Ambient temperature can be calculated from:

- $T_J = T_C + P^*\theta_{jc}$
- $T_A = T_J + P^*\theta_{ja}$
- $T_C = T_A + P^*[\theta_{ja}-\theta_{jc}]$

Values for θ_{ja} and θ_{jc} for various airflows are given in Table 10 for the PGA package. The PGA's θ_{ja} can be reduced by adding a heatsink.

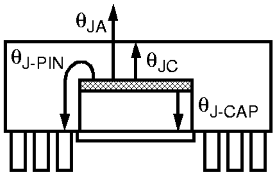
Maximum allowable ambient temperature (T_A) permitted without exceeding T_C is shown by the graphs in Figure 23, Figure 24 and Figure 25. The curves assume the maximum permitted supply current (I_{CC}) at each speed, V_{CC} of +5.0 V and a T_{CASE} of +85° C (PGA).

Table 10. 80960MC PGA Package Thermal Characteristics

Thermal Resistance — °C/Watt							
Parameter	Airflow — ft./min (m/sec)						
	0 (0)	50 (0.25)	100 (0.50)	200 (1.01)	400 (2.03)	600 (3.04)	800 (4.06)
θ Junction-to-Case	2	2	2	2	2	2	2
θ Case-to-Ambient (No Heatsink)	19	18	17	15	12	10	9
θ Case-to-Ambient (Omnidirectional Heatsink)	16	15	14	12	9	7	6
θ Case-to-Ambient (Unidirectional Heatsink)	15	14	13	11	8	6	5

NOTES:

1. This table applies to 80960MC PGA plugged into socket or soldered directly to board.
2. $\theta_{JA} = \theta_{JC} + \theta_{CA}$
3. $\theta_{J\text{-}CAP}$ = 4°C/W (approx.)
   $\theta_{J\text{-}PIN}$ = 4°C/W (inner pins) (approx.)
   $\theta_{J\text{-}PIN}$ = 8°C/W (outer pins) (approx.)



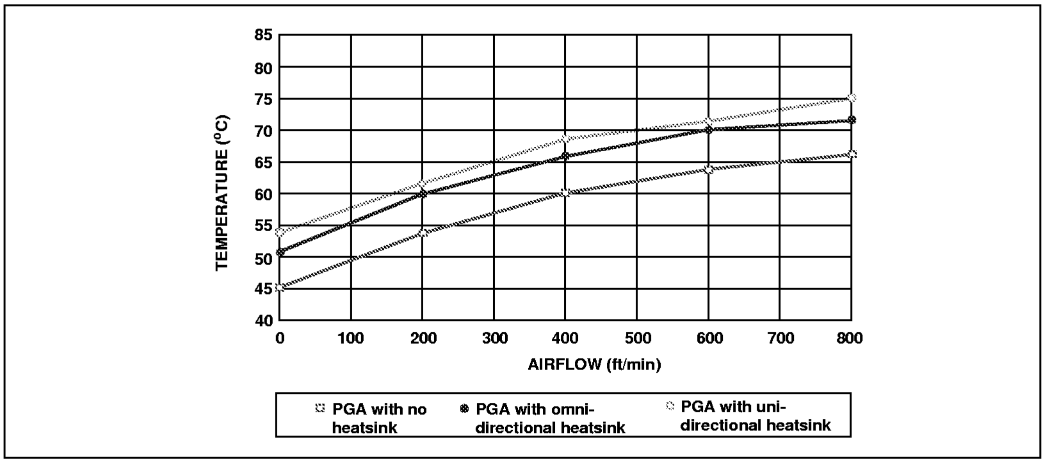


Figure 22. 25 MHz Maximum Allowable Ambient Temperature

4.0 WAVEFORMS

The following figures present waveforms for various transactions on the 80960MC'S local bus:

- Figure 23, Non-Burst Read and Write Transactions Without Wait States (pg. 30)
- Figure 24, Burst Read and Write Transaction Without Wait States (pg. 31)
- Figure 25, Burst Write Transaction with 2, 1, 1, 1 Wait States (pg. 32)
- Figure 26, Accesses Generated by Quad Word Read Bus Request, Misaligned Two Bytes from Quad Word Boundary (1, 0, 0, 0 Wait States) (pg. 33)
- Figure 27, Interrupt Acknowledge Transaction (pg. 34)
- Figure 28, Bus Exchange Transaction (PBM = Primary Bus Master, SBM = Secondary Bus Master) (pg. 35)

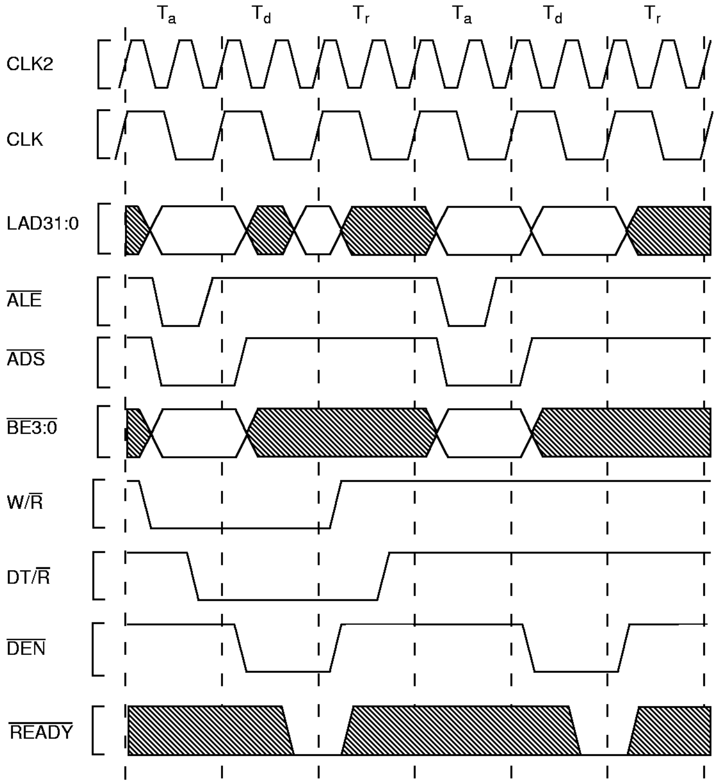


Figure 23. Non-Burst Read and Write Transactions Without Wait States

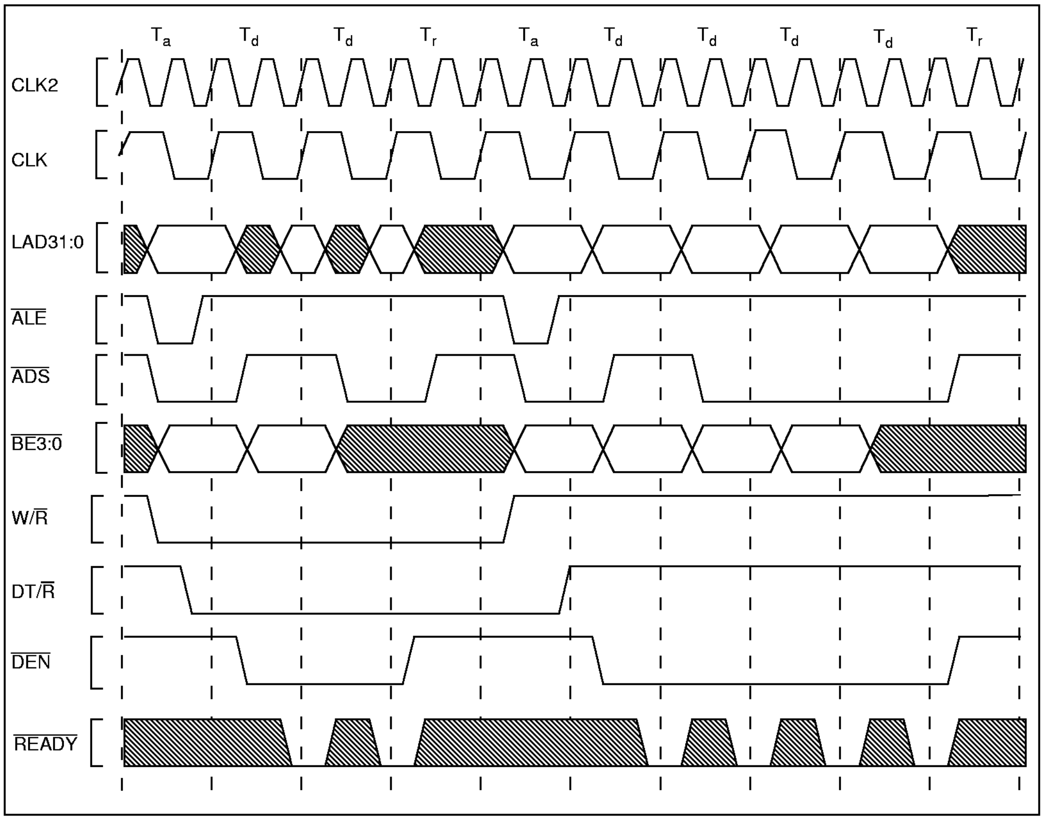


Figure 24. Burst Read and Write Transaction Without Wait States

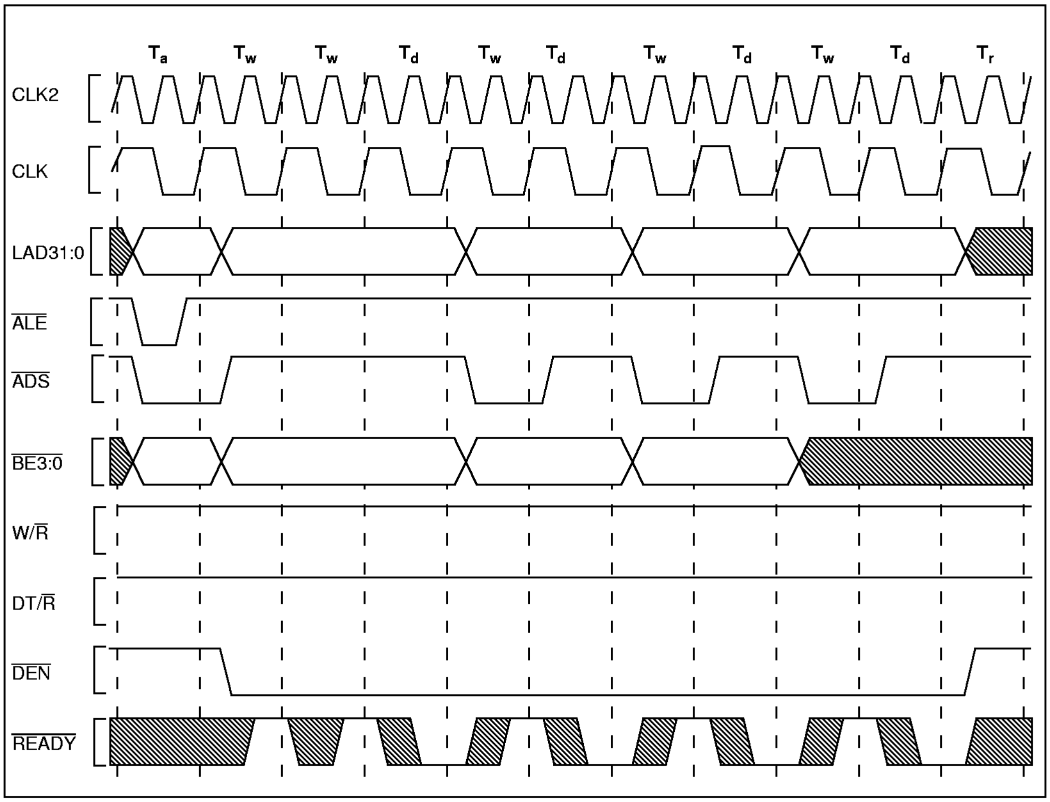


Figure 25. Burst Write Transaction with 2, 1, 1, 1 Wait States

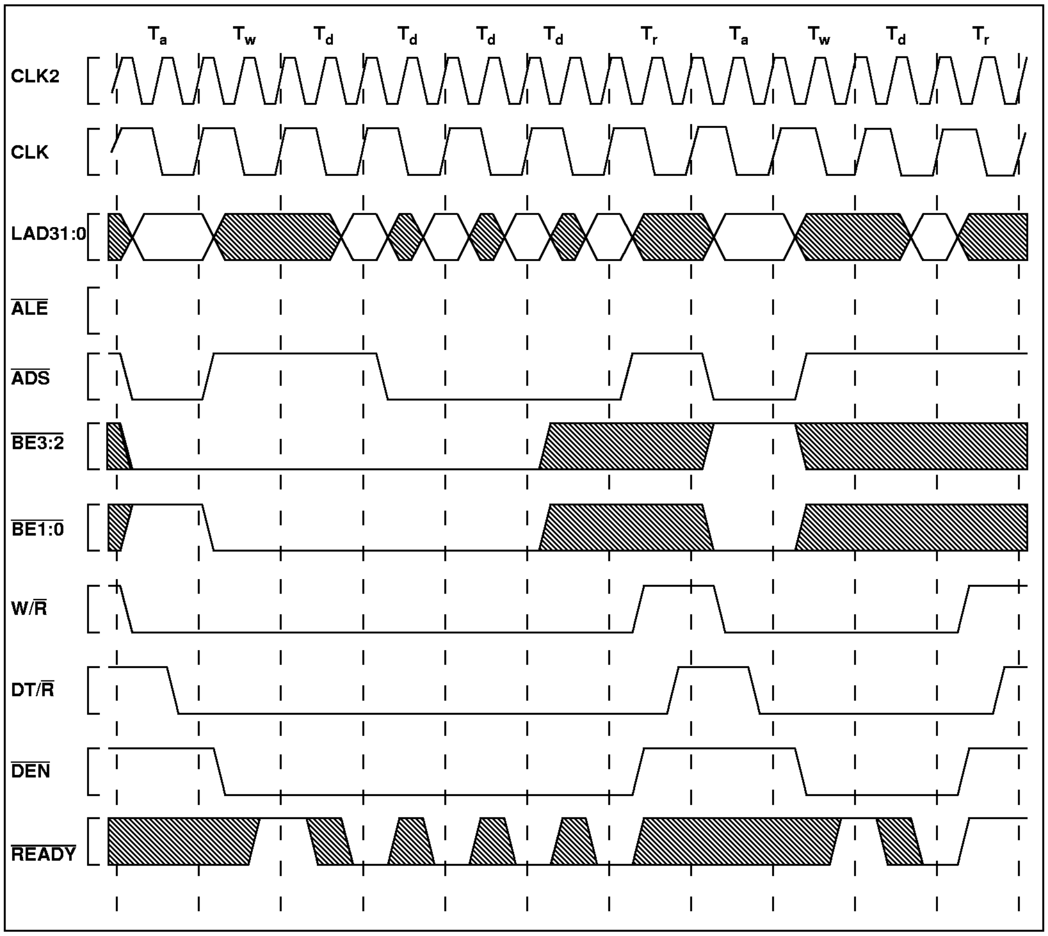
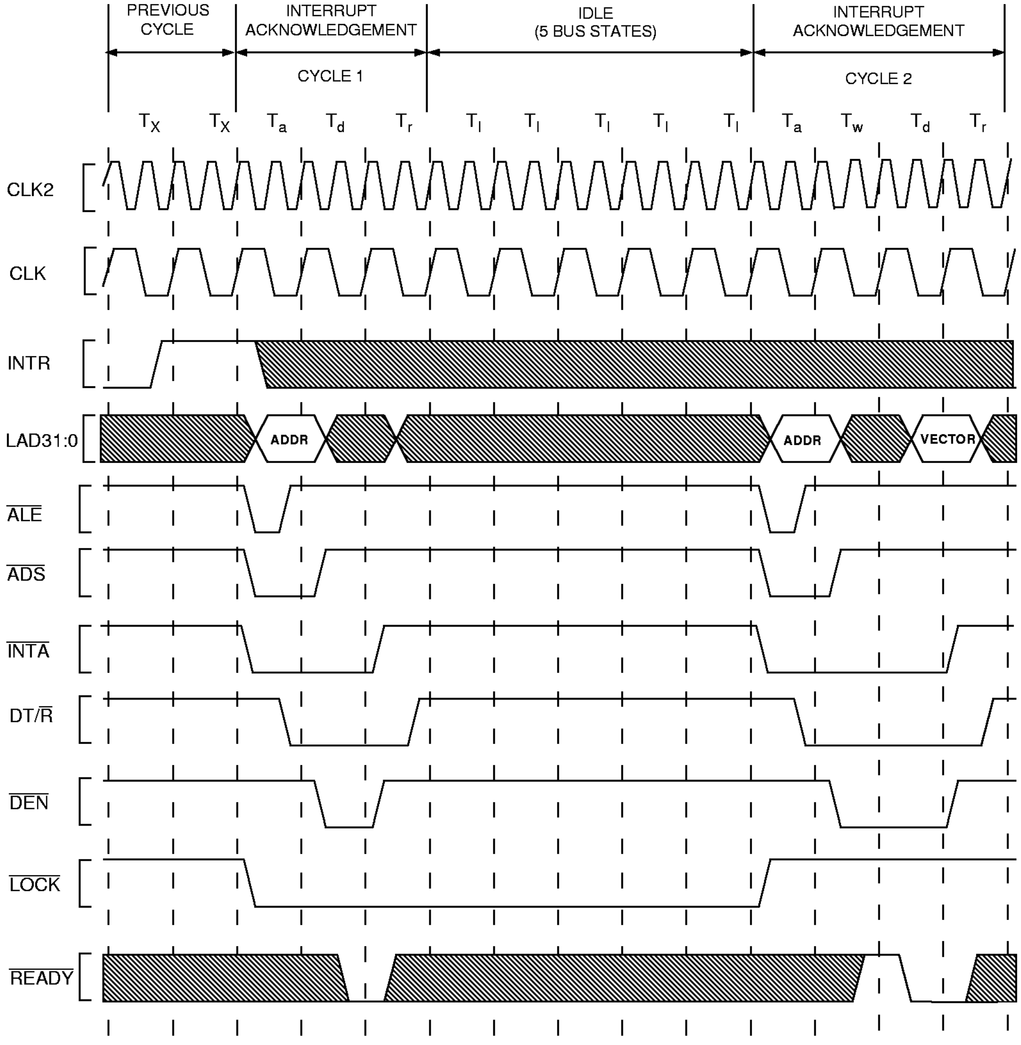


Figure 26. Accesses Generated by Quad Word Read Bus Request, Misaligned Two Bytes from Quad Word Boundary (1, 0, 0, 0 Wait States)



NOTE:
INTR can go low no sooner than the input hold time following the beginning of interrupt acknowledgment cycle 1. For a second interrupt to be acknowledged, INTR must be low for at least three cycles before it can be reasserted.

Figure 27. Interrupt Acknowledge Transaction

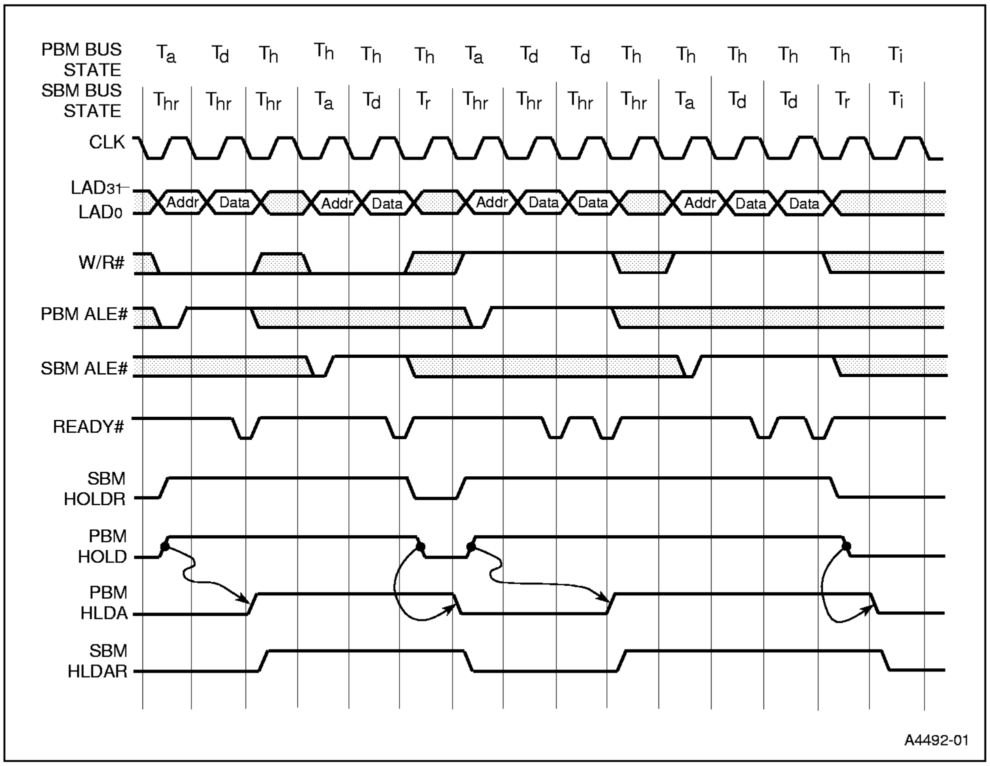

PBM BUS STATE
Ta Td Th Th Th Th Ta Td Td Th Th Th Th Th Ti
SBM BUS STATE
Thr Thr Thr Ta Td Tr Thr Thr Thr Thr Ta Td Td Tr Ti
CLK
LAD31–LAD0
Addr Data Addr Data Addr Data Data Addr Data Data
W/R#
PBM ALE#
SBM ALE#
READY#
SBM HOLDR
PBM HOLD
PBM HLDA
SBM HLDAR
A4492-01

Figure 28. Bus Exchange Transaction (PBM = Primary Bus Master, SBM = Secondary Bus Master)

5.0 REVISION HISTORY

No revision history was maintained in earlier revisions of this data sheet. All errata that has been identified to date is incorporated into this revision. The sections significantly changed since the previous revision are:

Section	Last Rev.	Description
		This is the initial commercial data sheet for the A80960MC.