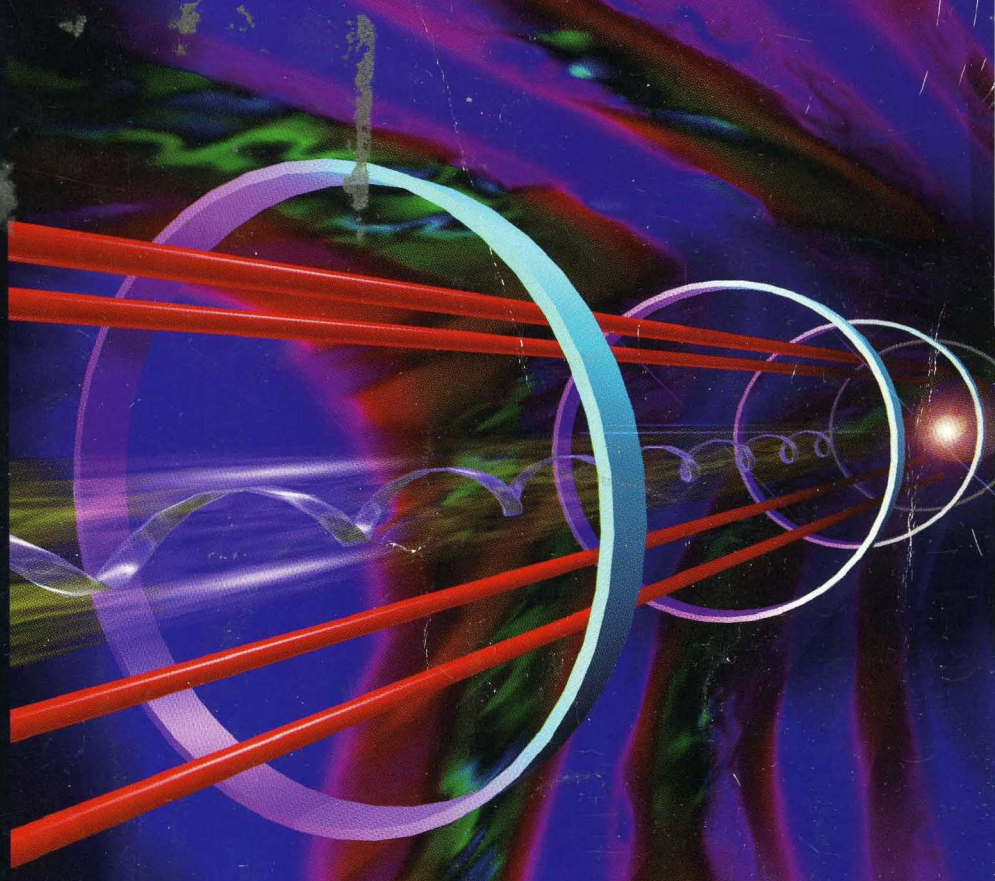
Pentium™ Processors and Related Products

Microprocessors

PCIsets

Peripheral Components



intel®

intel®

Pentium™ Processors and Related Products

Microprocessors, PCIsets, Peripheral Components

1995

Additional copies of this document or other Intel literature may be obtained from:

Intel Corporation
Literature Sales
P.O. Box 7641
Mt. Prospect, IL 60056-7641

or call 1-800-879-4683

DATASHEET DESIGNATIONS

Intel uses various datasheet markings to designate each phase of the document as it relates to the product. The markings appear in the lower inside corner of each datasheet page. Following are the definitions of each marking:

Datasheet Marking	Description
Product Preview	Contains information on products in the design phase of development. Do not finalize a design with this information. Revised information will be published when the product becomes available.
Advanced Information	Contains information on products being sampled or in the initial production phase of development.*
Preliminary	Contains preliminary information on new products in production.*
No Marking	Contains information on products in full production.*

* Specifications within these datasheets are subject to change without notice. Verify with your local Intel sales office that you have the latest datasheet before finalizing a design.

intel®

Table of Contents

Alphanumeric Index

intel®

1

Pentium™ Processor Overview

THE Pentium™ FAMILY—A TECHNICAL OVERVIEW

Intel's new Pentium processor combines the performance traditionally associated with minicomputers and workstations with the flexibility and compatibility that characterize the personal computer platform. Designed to meet the needs of today's and tomorrow's sophisticated software applications, the Pentium processor extends the range of Intel's microprocessor architecture to new heights, blurring previous distinctions between hardware platforms and creating an entirely new realm of possibilities for notebook computers, desktop PCs, and servers.

This paper begins by presenting an overview of the Pentium processor. Afterwards, it details the key technological features that enable this Intel solution to meet the market's evolving requirements for high performance, continued software compatibility, and advanced functionality.

THE WORLD'S BEST PERFORMANCE FOR ALL PC SOFTWARE

The Pentium processor family includes the highest performing members of Intel's family of microprocessors—the Pentium processor at iCOMP™ index 510\60 MHz, Pentium processor at iCOMP index 567\66 MHz, Pentium processor at iCOMP index 610\75 MHz, Pentium processor at iCOMP index 735\90 MHz, and Pentium processor at iCOMP index 815\100 MHz. While incorporating new features and improvements made possible by advances in semiconductor technology, the Pentium processor is fully software compatible with previous members of the Intel microprocessor family—thereby preserving the value of users' software investments which are worth billions of dollars. The Pentium processor meets the demands of computing in a number of areas: advanced operating systems, such as Windows 4.0*, UNIX*, Windows-NT*, OS/2*, Solaris* and NEXTstep*; compute-intensive graphics applications such as 3-D modeling, computer-aided design/engineering (CAD/CAE), large-scale financial analysis, high-throughput client/server, handwriting, and voice recognition; network applications; virtual reality; electronic mail that combines many of the above areas; and new applications yet to be developed.

The Pentium processor family was designed using an advanced process technology and has features that are less than a micron (one-millionth of a meter) in size. The Pentium processor (510\60, 567\66) was developed utilizing 5V, 0.8 micron technology with 3.1 million transistors, while the Pentium processor (610\75, 735\90, 815\100) was designed using 3.3V, 0.6 micron technology with 3.3 million transistors.

THE PENTIUM PROCESSOR: TECHNICAL INNOVATIONS

A number of innovative product features contribute to the Pentium processor's unique combination of high performance, compatibility, data integrity and upgradability. These include:

- Superscalar architecture
- Separate code and data caches
- Branch prediction
- High-performance floating-point unit
- Enhanced 64-bit data bus
- Data integrity features
- SL technology power management features
- Multiprocessing support
- Performance monitoring
- Memory page size feature
- Intel OverDrive™ processor upgradability

Superscalar Architecture

The Pentium processor's superscalar architecture enables the processor to achieve new levels of performance by executing more than one instruction per clock cycle. The term "superscalar" refers to a microprocessor architecture that contains more than one execution unit. These execution units—or pipelines—are where the chip processes the data and instructions that are fed to it by the rest of the system.

December 1994
Order Number: 242423-001

The Pentium processor's superscalar implementation represents a natural progression from previous generations of processors in the 32-bit Intel architecture. The Intel486™ processor, for example, is able to execute many of its instructions in one clock cycle, while previous generations of Intel microprocessors require multiple clock cycles to execute a single instruction.

This ability to execute multiple instructions per clock cycle is due to the fact that the Pentium processor's two pipelines can execute two instructions simultaneously. As with the Intel486 processor's single pipeline, the Pentium processor's dual pipelines execute integer instructions in five stages: *prefetch, decode 1, decode 2, execute* and *writeback.* This permits several instructions to be in various stages of execution, thus increasing processing performance.

The Pentium processor also uses hardwired instructions to replace many of the microcoded instructions used in previous microprocessor generations. Hardwired instructions are simple and commonly used, and can be executed by the processor's hardware without requiring microcode. This improves performance without affecting compatibility. In the case of more complex instructions, the Pentium processor's enhanced microcode further boosts performance by employing both dual integer pipelines to execute instructions.

Separate Code and Data Caches

Another significant advancement is the Pentium processor's innovative on-chip cache implementation. On-chip caches increase performance by acting as temporary storage places for commonly-used instructions and data, replacing the need to go off-chip to the system's main memory to fetch information. The Intel486 microprocessor, for example, contains a single 8-Kbyte on-chip cache to handle both code and data caching functions. Intel Pentium Processor designers improved on this implementation by creating separate on-chip code and data caches. This increases performance because bus conflicts are reduced (with a single cache, conflicts can occur between instruction pre-fetches and data accesses) and the caches are available more often when they are needed.

The Pentium processor's code and data caches each contain 8 Kbytes of information, and both are organized as two-way set associative caches—meaning that they save time by searching only pre-specified 32-byte segments rather than the entire cache. This performance-enhancing feature is in turn supplemented by the Pentium processor's 64-bit data bus, which ensures that the dual caches and superscalar execution pipelines are continually supplied with data.

The Pentium processor's data cache uses two other important techniques: "writeback" caching and an algorithm called the MESI (Modified, Exclusive, Shared, Invalid) protocol. The writeback method transfers data to the cache without going out to main memory (data is written to main memory only when it is removed from the cache). In contrast, previous-generation "write-through" cache implementations transfer data to the external memory each time the processor writes data to the cache. The writeback technique increases performance by reducing bus utilization and preventing needless bottlenecks in the system.

To ensure that data in the cache and in main memory are consistent, the data cache implements the MESI protocol. By obeying the rules of the protocol during reads/writes, the Pentium processor can maintain cache consistency and circumvent problems that might be caused by multiple processors using the same data.

Branch Prediction

Branch prediction is an advanced computing technique that boosts performance by keeping the execution pipelines full. This is accomplished by predetermining the most likely set of instructions to be executed. The Pentium processor is the first PC-compatible microprocessor to use branch prediction, which until now has traditionally been associated with the mainframe computers.

For a better understanding of this concept, consider a typical application program. After each pass through a software loop, the program performs a conditional test to determine whether to return to the beginning of the loop or to exit and continue on to the next execution step. These two choices, or paths, are called branches. Branch prediction forecasts which branch the software will require, based on the assumption that the previous branch that was taken will be used again. The Pentium processor makes branch predictions using a Branch Target Buffer (BTB). This software-transparent innovation eliminates the need for recompiling code, thus increasing overall speed and application software performance.

To efficiently predict branches, the Pentium processor uses two prefetch buffers. One buffer prefetches code in a linear fashion (for the next execution step) while the other prefetches instructions based on addresses in the

Branch Target Buffer (to jump to the beginning of the loop). As a result, the needed code is always prefetched before it is required for execution.

The Pentium processor's prediction algorithm can not only forecast simple branch choices, but also support more complex branch prediction—for example, within nested loops. This is accomplished by storing multiple branch addresses in the Branch Prediction Buffer. The BTB's design allows 256 addresses to be recorded, and thus the prediction algorithm can forecast up to 256 branches.

High-Performance Floating-Point Unit

The emerging wave of 32-bit compute-intensive software applications require a high degree of floating-point processing power to handle mathematical calculations. As the floating-point requirements of personal computer software have steadily increased, advances in microprocessor technology have been introduced to satisfy these needs. The Intel486 DX processor, for example, was the first Intel microprocessor to integrate math coprocessing functions on-chip; previous-generation Intel processors used off-chip math coprocessors when floating-point calculations were required.

The Pentium processor family takes math computational ability to the next performance level by using an enhanced on-chip floating-point unit that incorporates sophisticated eight-stage pipelining and hardwired functions. A three-stage floating-point instruction pipeline is appended to the integer pipelines. Most floating-point instructions begin execution in one of the integer pipelines, then move on to the floating-point pipeline. In addition, common floating-point functions—such as add, multiply and divide—are hardwired for faster execution.

As a result of these innovations, the Pentium processor (815\100) executes floating-point instructions five to ten times faster than the 33-MHz Intel486 DX processor, optimizing it for the high-speed numeric calculations inherent in advanced visual applications such as CAD and 3D graphics.

Enhanced 64-Bit Data Bus

The data bus is the highway that carries information between the processor and the memory subsystem. Because of its external 64-bit data bus, the Pentium processor can transfer data to and from memory at rates up to 528 Mbytes/second, a more than five-fold increase over the peak transfer rate of the 66-MHz Intel486 DX2 microprocessor (105 Mbytes/second). This wider data bus facilitates high-speed processing by maintaining the flow of instructions and data to the processor's superscalar execution unit. As a result, the Pentium processor's (815\100) overall performance is over two and one-half times that of the Intel486 DX2-66 microprocessor.

In addition to having a wider data bus, the Pentium processor implements bus cycle pipelining to increase bus bandwidth. Bus cycle pipelining allows a second cycle to start before the first one is completed. This gives the memory subsystem more time to decode the address, which allows slower and less-expensive memory components to be used—resulting in a lower overall system cost. Burst reads and writes, parity on address and data, and a simple cycle identification all contribute to providing better bandwidth and improved system reliability.

The Pentium processor also has two write buffers, one corresponding to each pipeline, to enhance the performance of consecutive writes to memory. Write buffers improve performance by allowing the processor to proceed with the next pair of instructions, even though one of the current instructions needs to write to memory while the bus is busy.

Data Integrity Features

Protecting important data and ensuring its integrity has become increasingly important as mission-critical applications continue to proliferate. To ensure the Pentium processor's reliability, Intel ran millions of simulations and tests. In addition, designers integrated two advanced features traditionally associated with mainframe-class designs—internal error detection and functional redundancy testing—to help preserve data integrity in today's evolving PC-based networks.

Internal error detection places parity bits on the internal code and data caches, translation look aside buffers, microcode, and branch target buffer. This feature helps detect errors in a manner that remains transparent to both the user and the system.

For situations where data integrity is especially crucial, the Pentium processor supports Functional Redundancy Checking (FRC). FRC requires the use of two Pentium chips, one acting as the master and the other as the "checker". The two chips run in tandem, and the checker compares its output with that of the master Pentium processor to assure that errors have not occurred. If a discrepancy is discovered, the system is notified.

SL Enhanced Power Management Features

The Pentium processors (610\75, 735\90, 815\100) incorporate new SL technology features for superior power-management capabilities. These features operate at two levels: the microprocessor and the system. Power management at the processor level involves putting the processor into low power state during non-processor intensive tasks (such as word processing), or into a very low-power state when the computer is not in use ("sleep" mode). At the system level, Intel's SL technology uses system management mode (SMM) to control the way power is used by the computer (including peripherals). This mode provides intelligent system management that allow the microprocessor to slow down, suspend, or completely shut down various system components so as to maximize energy savings. All members of the Pentium processor family include SMM.

Multiprocessor Support

The Pentium processor is ideal for the increasing wave of multiprocessing systems. Multiprocessing applications that combine two or more Pentium processors are well served by the chip's advanced architecture, separate on-chip code and data caches, chip sets for controlling external caches, and sophisticated data integrity features.

As previously discussed, the Pentium processor family uses the MESI protocol to maintain cache consistency among several processors. The Pentium processor also ensures that instructions are seen by the system in the order that they were programmed. This strong ordering helps software designed to run on a single-processor system to work correctly in a multiprocessing environment.

The Pentium processors (610\75, 735\90, 815\100) also include two new multiprocessor (MP) features: a multiprocessor interrupt controller on-chip and the dual processor mode. The processor's on-chip MP interrupt controller can support up to 60 processors. The dual processor mode enables two processors to share a single second-level cache, allowing the development of low-cost shared-cache multiprocessor systems for workstations and low-end servers.

Performance Monitoring

Performance monitoring is a feature of the Pentium processor that enables system designers and application developers to optimize their hardware and software products by identifying potential code bottlenecks. Designers can observe and count clocks for internal processor events that affect the performance of data reads and writes, cache hits and misses, interrupts, and bus utilization. This allows them to measure the effect that their code has on both the Pentium processor architecture and their product, and to fine-tune their application or system for optimal performance. The benefit to end users is better value and higher performance, due to the greater synergy between the Pentium processor, its host system, and application software.

Memory Page Size Feature

The Pentium processor offers the option of supporting either the traditional memory page size of 4 Kbytes, or a larger 4-Mbyte page. This feature—which is transparent to the application software—was provided to reduce the frequency of page swapping in complex graphics applications, frame buffers, and operating system kernels, where the increased page size allows users to map large, previously unwieldy objects. The larger page enables an increased page hit rate, which results in higher performance.

Upgradability

As with all new implementations of the Intel 32-bit microprocessor architecture, the Pentium processor has been designed for easy upgradability using Intel's upgrade technology. This innovation protects user investments by adding performance that helps to maintain the productivity levels of Intel processor-based systems over their entire lifespans.

Upgrade technology makes it possible for users to take advantage of more advanced processor technology in their existing systems with an easy-to-install, single-chip performance upgrade. For example, Intel's first upgrade options, the OverDrive processors developed for Intel486 SX and Intel486 DX processors, apply the same speed-doubling technology used in the development of the Intel486 DX2 microprocessor.

Intel's upgrade processors will also be available for systems based on the Pentium processor family, ensuring an easy future upgrade path based on even more advanced processor technology. In addition, Pentium processor technology will be the basis of the upgrade processors now being developed for Intel486 processor-based systems.

INCREASED PERFORMANCE: BY THE NUMBERS

The Pentium processor stands alone on the performance ladder when compared to all other PC-compatible microprocessors, raising the standard for the Intel 32-bit architecture. While there are many ways to measure performance, three different examples are offered here to demonstrate the Pentium processor's speed and processing power: Intel's iCOMP index, the SPECint92 UNIX benchmark, and the SPECfp92 UNIX benchmark.

One indication of the Pentium processor's high performance is provided by Intel's iCOMP (Intel Comparative Microprocessor Performance) index. The iCOMP index, which measures the performance of the members of the Intel 32-bit architecture, was created so that computer users can more easily identify relative performance differences among Intel's microprocessors. The index is based on four distinct aspects of processor performance: integer, floating-point, graphics and video performance. Each of the four elements is considered for both 16- and 32-bit software, and is weighted relative to the estimated percentage of time it occupies the processor's attention (based on a mix of today's commonly used application software). As shown in Figure 1, the Pentium processor at iCOMP index 815\100 MHz has more than 2.7 times the relative performance of the 66-MHz Intel486 DX2 microprocessor, which has an iCOMP rating of 297.

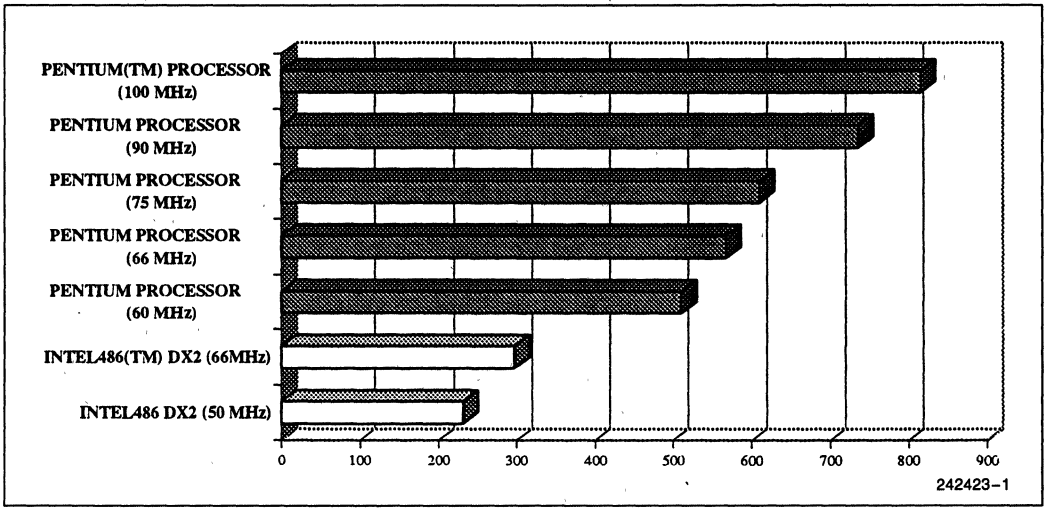


Figure 1. iCOMP™ Index Ratings for Intel Processors

SPECint92 is a processor-intensive UNIX benchmark (Figure 2) that evaluates desktop performance using a representative mix of application instructions. With a SPECint92 rating of 100.0, the Pentium processor at iCOMP index 815\100 MHz outperforms many workstation-class, RISC-based processors, including members of the IBM, MIPS and Sun SPARC processor families.

The SPECfp92 UNIX benchmark (Figure 3) is a useful measure of floating-point performance. The SPECfp92 rating for the Pentium processor at iCOMP index 815\100 MHz is 80.6. This is comparable to that of today's RISC architectures, and is more than 4.3 times that of the 66-MHz Intel486 DX2 processor.

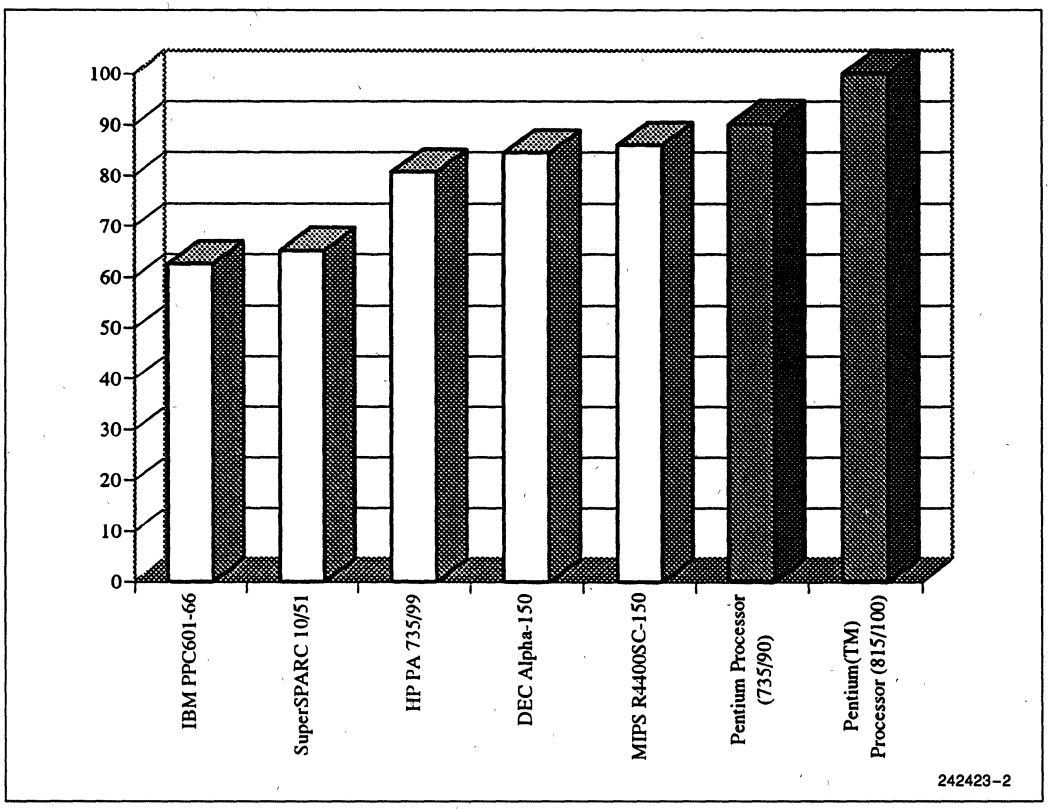


Figure 2. UNIX SPECint92 Benchmark Performance

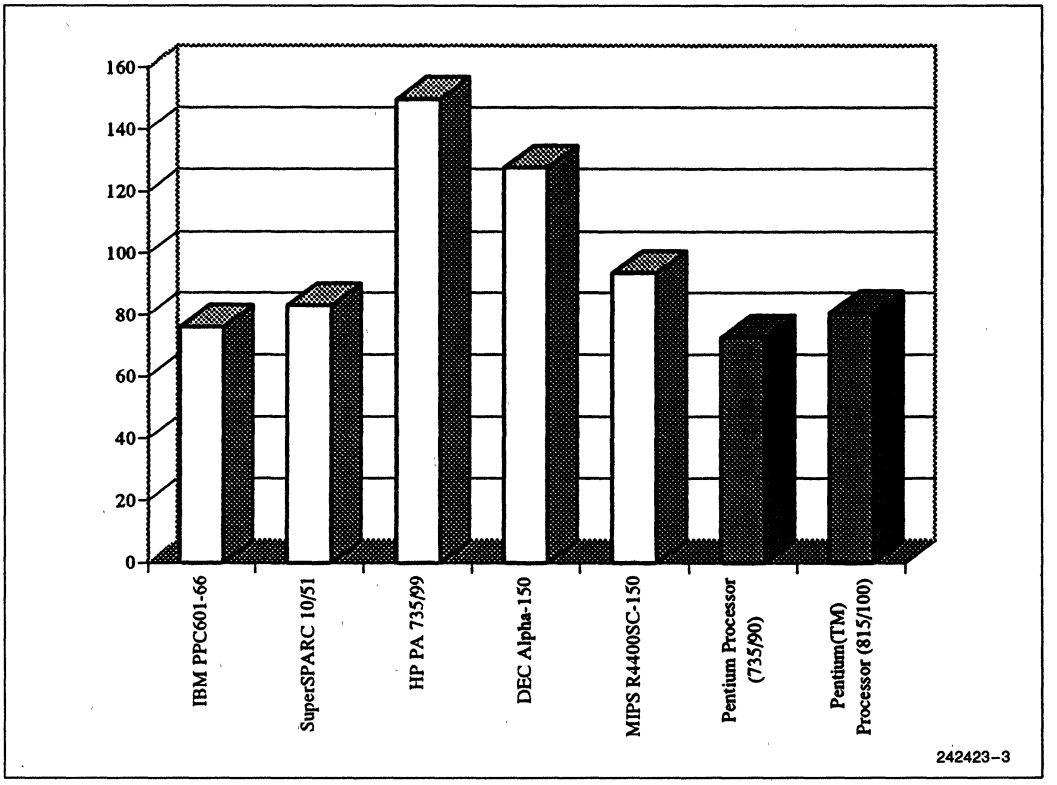


Figure 3. UNIX SPECfp92 Benchmark Performance

HIGH PERFORMANCE WHILE MAINTAINING COMPATIBILITY

The Pentium processor family provides extremely high-performance because it incorporates the latest state-of-the-art design principles. With its superscalar architecture, separate code and data caches, branch prediction, and an enhanced floating-point unit, the Pentium processor can meet the performance needs of today's—and tomorrow's—applications software. Meanwhile, it maintains complete compatibility with the $50 billion installed base of software currently running on members of the Intel family.

The Pentium processor's combination of performance and compatibility uniquely positions it to meet the needs of the emerging wave of notebook, desktop, and server applications. Not only will users experience dramatic performance improvements while running their current software, but they can also anticipate that new applications will take even further advantage of the Pentium processor's high-performance features.

intel®

2

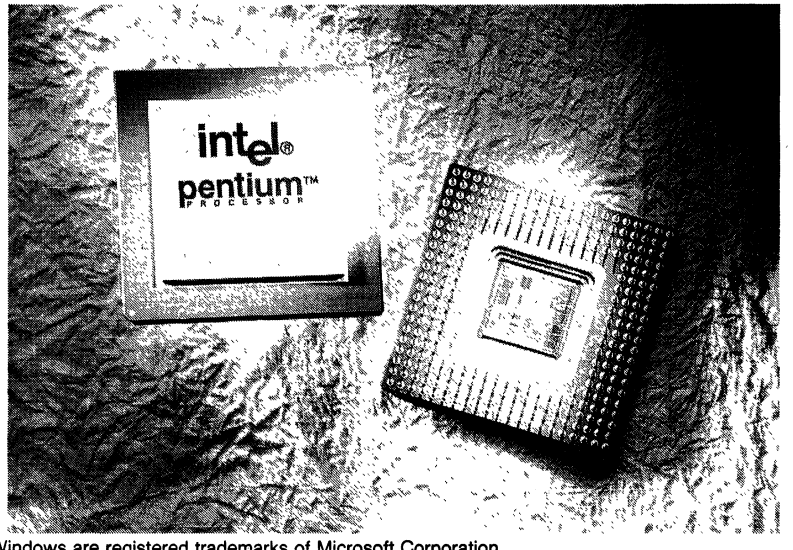
Pentium™ Processor and Peripherals

PENTIUM™ PROCESSOR AT iCOMP INDEX 510\60 MHz
PENTIUM™ PROCESSOR AT iCOMP INDEX 567\66 MHz

- **Binary Compatible with Large Software Base**
  - **DOS, OS/2, UNIX, and WINDOWS**
- **32-Bit Microprocessor**
  - **32-Bit Addressing**
  - **64-Bit Data Bus**
- **Superscalar Architecture**
  - **Two Pipelined Integer Units**
  - **Capable of under One Clock per Instruction**
  - **Pipelined Floating Point Unit**
- **Separate Code and Data Caches**
  - **8K Code, 8K Write Back Data**
  - **2-Way 32-Byte Line Size**
  - **Software Transparent**
  - **MESI Cache Consistency Protocol**
- **Advanced Design Features**
  - **Branch Prediction**
  - **Virtual Mode Extensions**
- **273-Pin Grid Array Package**
- **BiCMOS Silicon Technology**
- **Increased Page Size**
  - **4M for Increased TLB Hit Rate**
- **Multi-Processor Support**
  - **Multiprocessor Instructions**
  - **Support for Second Level Cache**
- **Internal Error Detection**
  - **Functional Redundancy Checking**
  - **Built in Self Test**
  - **Parity Testing and Checking**
- **IEEE 1149.1 Boundary Scan Compatibility**
- **Performance Monitoring**
  - **Counts Occurrence of Internal Events**
  - **Traces Execution through Pipelines**

2

The Pentium processor (510\60, 567\66) provides the next generation of power for high-end workstations and servers. The Pentium processor (510\60, 567\66) is compatible with the entire installed base of applications for DOS, Windows, OS/2, and UNIX. The Pentium processor's superscalar architecture can execute two instructions per clock cycle. Branch Prediction and separate caches also increase performance. The pipelined floating point unit of the Pentium processor (510\60, 567\66) delivers workstation level performance. Separate code and data caches reduce cache conflicts while remaining software transparent. The Pentium processor (510\60, 567\66) has 3.1 million transistors and is built on Intel's 0.8 Micron BiCMOS silicon technology.



241595-1

November 1994
Order Number: 241595-002

Pentium™ Processor (510\60, 567\66)

CONTENTS

CONTENTS

1.0 MICROPROCESSOR ARCHITECTURE OVERVIEW

The Pentium™ processor (510\60, 567\66) is the next generation member of the Intel386™ and Intel486™ microprocessor family. It is 100% binary compatible with the 8086/88, 80286, Intel386 DX CPU, Intel386 SX CPU, Intel486 DX CPU, Intel486 SX and the Intel486 DX2 CPUs.

The Pentium processor (510\60, 567\66) contains all of the features of the Intel486 CPU, and provides significant enhancements and additions including the following:

- Superscalar Architecture
- Dynamic Branch Prediction
- Pipelined Floating-Point Unit
- Improved Instruction Execution Time
- Separate 8K Code and Data Caches
- Writeback MESI Protocol in the Data Cache
- 64-Bit Data Bus
- Bus Cycle Pipelining
- Address Parity
- Internal Parity Checking
- Functional Redundancy Checking
- Execution Tracing
- Performance Monitoring
- IEEE 1149.1 Boundary Scan
- System Management Mode
- Virtual Mode Extensions

The application instruction set of the Pentium processor (510\60, 567\66) includes the complete Intel486 CPU instruction set with extensions to accommodate some of the additional functionality of the Pentium processor (510\60, 567\66). All application software written for the Intel386 and Intel486 microprocessors will run on the Pentium processor (510\60, 567\66) without modification. The on-chip memory management unit (MMU) is completely compatible with the Intel386 and Intel486 CPUs.

The Pentium processor (510\60, 567\66) implements several enhancements to increase performance. The two instruction pipelines and floating-point unit on the Pentium processor (510\60, 567/66) are capable of independent operation. Each pipeline issues frequently used instructions in a single clock. Together, the dual pipes can issue two integer instructions in one clock, or one floating point instruction (under certain circumstances, 2 floating point instructions) in one clock.

Branch prediction is implemented in the Pentium processor (510\60, 567\66). To support this, the Pentium processor (510\60, 567\66) implements two prefetch buffers, one to prefetch code in a linear fashion, and one that prefetches code according to the BTB so the needed code is almost always prefetched before it is needed for execution.

The floating-point unit has been completely redesigned over the Intel486 CPU. Faster algorithms provide up to 10X speed-up for common operations including add, multiply, and load.

The Pentium processor (510\60, 567\66) includes separate code and data caches integrated on chip to meet its performance goals. Each cache is 8 Kbytes in size, with a 32-byte line size and is 2-way set associative. Each cache has a dedicated Translation Lookaside Buffer (TLB) to translate linear addresses to physical addresses. The data cache is configurable to be writeback or writethrough on a line by line basis and follows the MESI protocol. The data cache tags are triple ported to support two data transfers and an inquire cycle in the same clock. The code cache is an inherently write protected cache. The code cache tags are also triple ported to support snooping and split line accesses. Individual pages can be configured as cacheable or non-cacheable by software or hardware. The caches can be enabled or disabled by software or hardware.

The Pentium processor (510\60, 567\66) has increased the data bus to 64-bits to improve the data transfer rate. Burst read and burst writeback cycles are supported by the Pentium processor (510\60, 567\66). In addition, bus cycle pipelining has been added to allow two bus cycles to be in progress simultaneously. The Pentium processor (510\60, 567\66) Memory Management Unit contains optional extensions to the architecture which allow 4 Mbyte page sizes.

The Pentium processor (510\60, 567\66) has added significant data integrity and error detection capability. Data parity checking is still supported on a byte by byte basis. Address parity checking, and internal parity checking features have been added along with a new exception, the machine check exception. In addition, the Pentium processor (510\60, 567\66) has implemented functional redundancy checking to provide maximum error detection of the processor and the interface to the processor. When functional redundancy checking is used, a second processor, the "checker" is used to execute in lock step with the "master" processor. The checker samples the master's outputs and compares those values with the values it computes internally, and asserts an error signal if a mismatch occurs.

As more and more functions are integrated on chip, the complexity of board level testing is increased. To address this, the Pentium processor (510\60, 567\66) has increased test and debug capability. Like many of the Intel486 CPUs, the Pentium processor (510\60, 567\66) implements IEEE Boundary Scan (Standard 1149.1). In addition, the Pentium processor (510\60, 567\66) has specified 4 breakpoint pins that correspond to each of the debug registers and externally indicate a breakpoint match. Execution tracing provides external indications when an instruction has completed execution in either of the two internal pipelines, or when a branch has been taken.

System management mode has been implemented along with some extensions to the SMM architecture. Enhancements to the Virtual 8086 mode have been made to increase performance by reducing the number of times it is necessary to trap to a virtual 8086 monitor.

Figure 1-1 shows a block diagram of the Pentium processor (510\60, 567\66).

The block diagram shows the two instruction pipelines, the "u" pipe and the "v" pipe. The u-pipe can execute all integer and floating point instructions. The v-pipe can execute simple integer instructions and the FXCH floating point instructions.

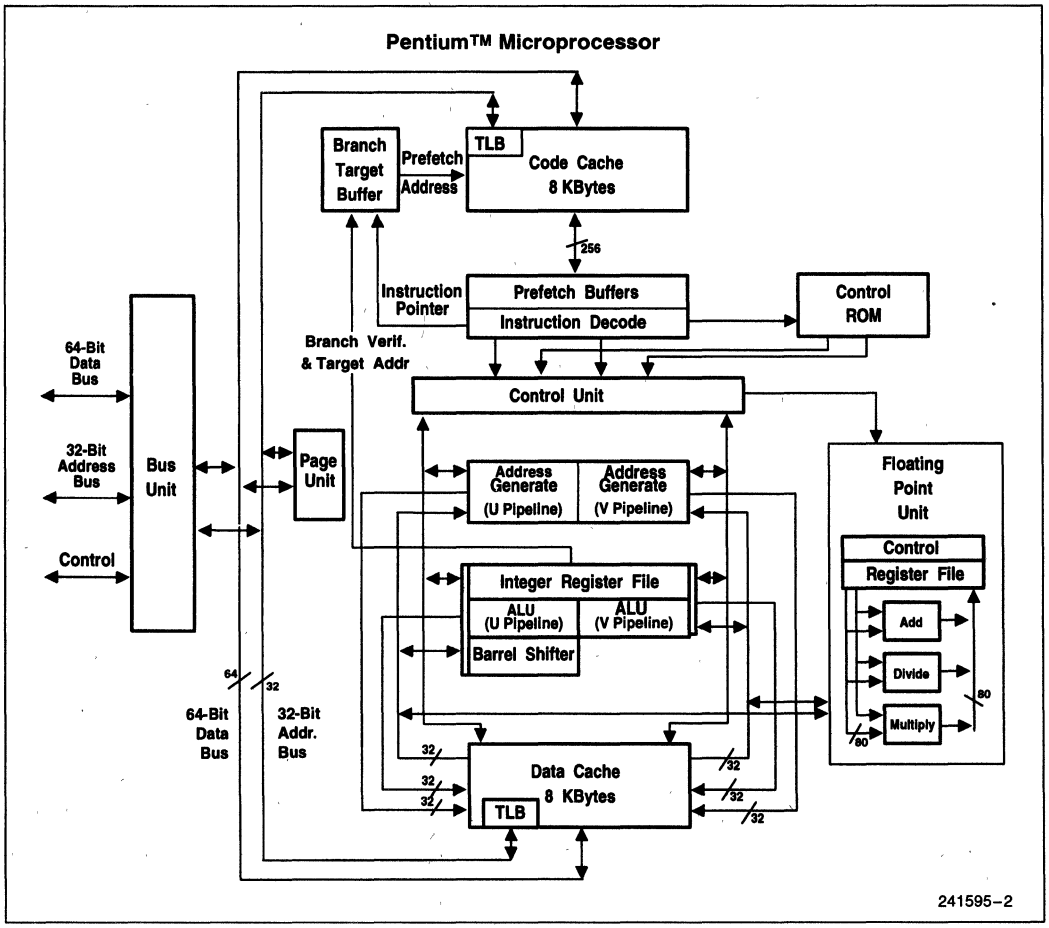


Figure 1-1. Pentium™ Processor (510\60, 567\66) Block Diagram

The separate caches are shown, the code cache and data cache. The data cache has two ports, one for each of the two pipes (the tags are triple ported to allow simultaneous inquire cycles). The data cache has a dedicated Translation Lookaside Buffer (TLB) to translate linear addresses to the physical addresses used by the data cache.

The code cache, branch target buffer and prefetch buffers are responsible for getting raw instructions into the execution units of the Pentium processor (510\60, 567\66). Instructions are fetched from the code cache or from the external bus. Branch addresses are remembered by the branch target buffer. The code cache TLB translates linear addresses to physical addresses used by the code cache.

The decode unit decodes the prefetched instructions so the Pentium processor (510\60, 567\66) can execute the instruction. The control ROM contains the microcode which controls the sequence of operations that must be performed to implement the Pentium processor (510\60, 567\66) architecture. The control ROM unit has direct control over both pipelines.

The Pentium processor (510\60, 567\66) contains a pipelined floating point unit that provides a significant floating point performance advantage over previous generations of the Pentium processor (510\60, 567\66).

The architectural features introduced in this chapter are more fully described in the *Pentium™ Processor (510\60, 567\66) User's Manual.*

2.0 PINOUT

2.1 Pinout and Pin Descriptions

2.1.1 Pentium™ PROCESSOR (510\60, 567\66) PINOUT

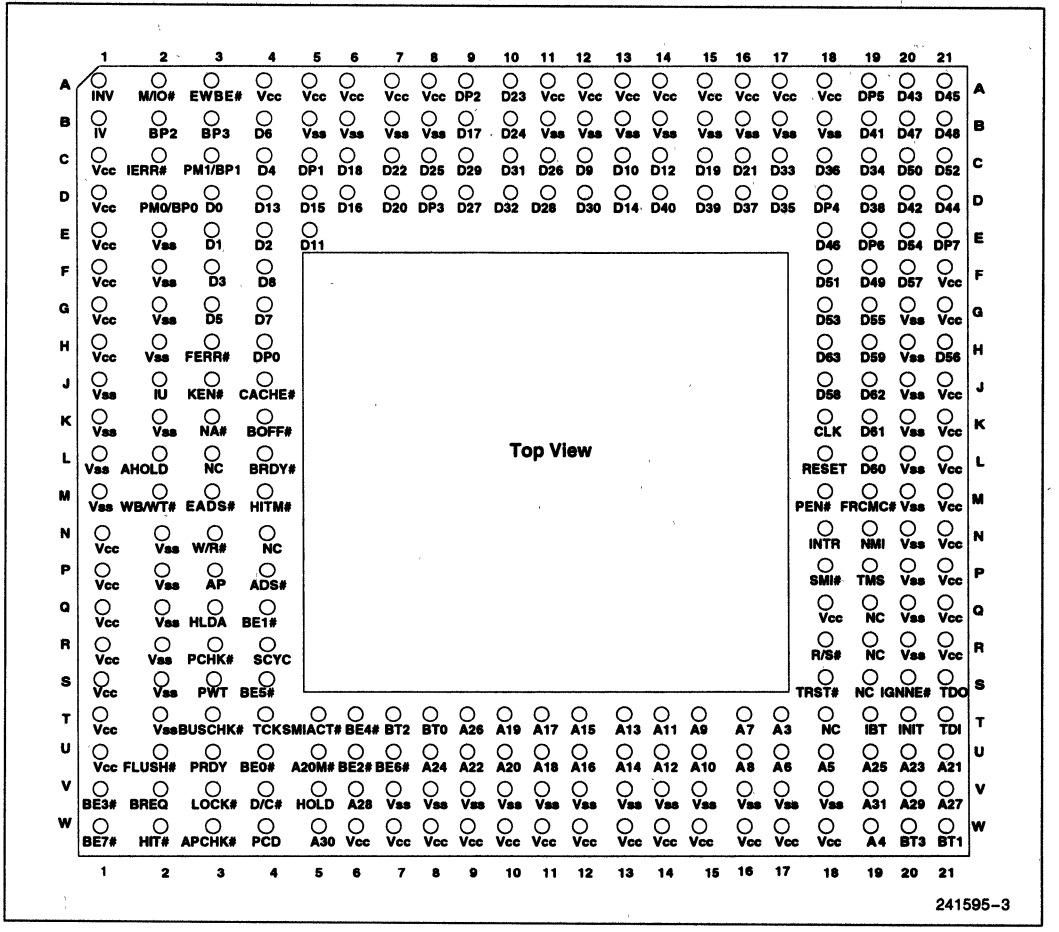


Figure 2-1. Pentium™ Processor (510\60, 567\66) Pinout (Top View)

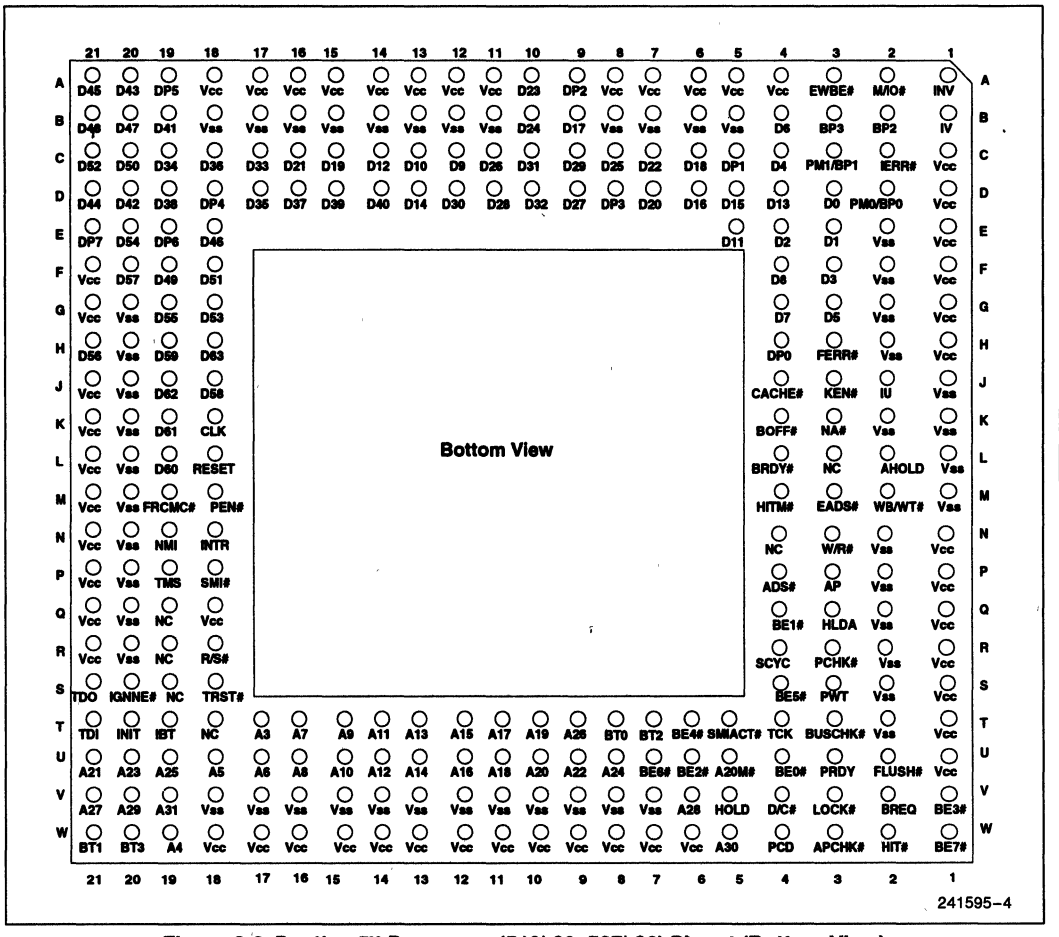


Figure 2-2. Pentium™ Processor (510\60, 567\66) Pinout (Bottom View)

Table 2-1. Pentium™ Processor (510\60, 567\66) Pin Cross Reference Table by Pin Name

Signal	Location	Signal	Location	Signal	Location	Signal	Location
A3	T17	BE2#	U06	D18	C06	D54	E20
A4	W19	BE3#	V01	D19	C15	D55	G19
A5	U18	BE4#	T06	D20	D07	D56	H21
A6	U17	BE5#	S04	D21	C16	D57	F20
A7	T16	BE6#	U07	D22	C07	D58	J18
A8	U16	BE7#	W01	D23	A10	D59	H19
A9	T15	BOFF#	K04	D24	B10	D60	L19
A10	U15	BP2	B02	D25	C08	D61	K19
A11	T14	BP3	B03	D26	C11	D62	J19
A12	U14	BRDY#	L04	D27	D09	D63	H18
A13	T13	BREQ	V02	D28	D11	D/C#	V04
A14	U13	BT0	T08	D29	C09	DP0	H04
A15	T12	BT1	W21	D30	D12	DP1	C05
A16	U12	BT2	T07	D31	C10	DP2	A9
A17	T11	BT3	W20	D32	D10	DP3	D08
A18	U11	BUSCHK#	T03	D33	C17	DP4	D18
A19	T10	CACHE#	J04	D34	C19	DP5	A19
A20	U10	CLK	K18	D35	D17	DP6	E19
A21	U21	D0	D03	D36	C18	DP7	E21
A22	U09	D1	E03	D37	D16	EADS#	M03
A23	U20	D2	E04	D38	D19	EWBE#	A03
A24	U08	D3	F03	D39	D15	FERR#	H03
A25	U19	D4	C04	D40	D14	FLUSH#	U02
A26	T09	D5	G03	D41	B19	FRCMC#	M19
A27	V21	D6	B04	D42	D20	HIT#	W02
A28	V06	D7	G04	D43	A20	HITM#	M04
A29	V20	D8	F04	D44	D21	HLDA	Q03
A30	W05	D9	C12	D45	A21	HOLD	V05
A31	V19	D10	C13	D46	E18	IBT	T19
A20M#	U05	D11	E05	D47	B20	IERR#	C02
ADS#	P04	D12	C14	D48	B21	IGNNE#	S20
AHOLD	L02	D13	D04	D49	F19	INIT	T20
AP	P03	D14	D13	D50	C20	INTR	N18
APCHK#	W03	D15	D05	D51	F18	INV	A01
BE0#	U04	D16	D06	D52	C21	IU	J02
BE1#	Q04	D17	B09	D53	G18	IV	B01

Table 2-1. Pentium™ Processor (510\60, 567\66) Pin Cross Reference Table by Pin Name (Continued)

Signal	Location
KEN#	J03
LOCK#	V03
M/IO#	A02
NA#	K03
NMI	N19
PCD	W04
PCHK#	R03
PEN#	M18
PM0/BP0	D02
PM1/BP1	C03
PRDY	U03
PWT	S03

Signal	Location
RESET	L18
R/S#	R18
SCYC	R04
SMI#	P18
SMIACT#	T05
TCK	T04
TDI	T21
TDO	S21
TMS	P19
TRST#	S18
WB/WT#	M02
W/R#	N03

Signal	Location
NC	L03, N04, Q19, R19, S19, T18
V_{CC}	A04, A05, A06, A07, A08, A11, A12, A13, A14, A15, A16, A17, A18, C01, D01, E01, F01, F21, G01, G21, H01, J21 K21, L21, M21, N01, N21, P01, P21, Q01, Q18, Q21, R01, R21, S01, T01, U01, W06, W07, W08, W09, W10, W11, W12, W13, W14, W15, W16, W17, W18

Signal	Location
V_{SS}	B05, B06, B07, B08, B11, B12, B13, B14, B15, B16, B17, B18, E02, F02, G02, G20, H02, H20, J01, J20, K01, K02, K20, L01, L20, M01, M20, N02, N20, P02, P20, Q02, Q20, R02, R20, S02, T02, V07, V08, V09, V10, V11, V12, V13, V14, V15, V16, V17, V18

2.2 Design Notes

For reliable operation, always connect unused inputs to an appropriate signal level. Unused active LOW inputs should be connected to V_{CC}. Unused active HIGH inputs should be connected to GND.

No Connect (NC) pins must remain unconnected. Connection of NC pins may result in component failure or incompatibility with processor steppings.

NOTE:
The No Connect pin located at L03 (BRDYC#) along with BUSCHK# are sampled by the Pentium processor (510\60, 567\66) at RESET to configure the I/O buffers of the processor for use with the 82496 Cache Controller/82491 Cache SRAM secondary cache as a chip set (refer to the *82496 Cache Controller/82491 Cache SRAM Data Book for Use with the Pentium™ Processor (510\60, 567\66)* for further information).

2.3 Quick Pin Reference

This section gives a brief functional description of each of the pins. For a detailed description, see the Hardware Interface chapter in the *Pentium™ Processor (510\60, 567\66) User's Manual*, Vol. 1. **Note that all input pins must meet their AC/DC specifications to guarantee proper functional behavior.** In this section, the pins are arranged in alphabetical order. The functional grouping of each pin is listed at the end of this chapter.

The # symbol at the end of a signal name indicates that the active, or asserted state occurs when the signal is at a low voltage. When a # symbol is not present after the signal name, the signal is active, or asserted at the high voltage level.

Table 2-2. Quick Pin Reference

Symbol	Type*	Name and Function
A20M#	I	When the *address bit 20 mask* pin is asserted, the Pentium™ Processor (510\60, 567\66) emulates the address wraparound at one Mbyte which occurs on the 8086. When A20M# is asserted, the Pentium processor (510\60, 567\66) masks physical address bit 20 (A20) before performing a lookup to the internal caches or driving a memory cycle on the bus. The effect of A20M# is undefined in protected mode. A20M# must be asserted only when the processor is in real mode.
A31–A3	I/O	As outputs, the *address* lines of the processor along with the byte enables define the physical area of memory or I/O accessed. The external system drives the inquire address to the processor on A31–A5.
ADS#	O	The *address status* indicates that a new valid bus cycle is currently being driven by the Pentium processor (510\60, 567\66).
AHOLD	I/O	In response to the assertion of *address hold*, the Pentium processor (510\60, 567\66) will stop driving the address lines (A31–A3), and AP in the next clock. The rest of the bus will remain active so data can be returned or driven for previously issued bus cycles.
AP	I/O	*Address parity* is driven by the Pentium processor (510\60, 567\66) with even parity information on all Pentium processor (510\60, 567\66) generated cycles in the same clock that the address is driven. Even parity must be driven back to the Pentium processor (510\60, 567\66) during inquire cycles on this pin in the same clock as EADS# to ensure that the correct parity check status is indicated by the Pentium processor (510\60, 567\66).
APCHK#	O	The *address parity check* status pin is asserted two clocks after EADS# is sampled active if the Pentium processor (510\60, 567\66) has detected a parity error on the address bus during inquire cycles. APCHK# will remain active for one clock each time a parity error is detected.
BE7#–BE0#	O	The *byte enable* pins are used to determine which bytes must be written to external memory, or which bytes were requested by the CPU for the current cycle. The byte enables are driven in the same clock as the address lines (A31–3).
BOFF#	I	The *backoff* input is used to abort all outstanding bus cycles that have not yet completed. In response to BOFF#, the Pentium processor (510\60, 567\66) will float all pins normally floated during bus hold in the next clock. The processor remains in bus hold until BOFF# is negated at which time the Pentium processor (510\60, 567\66) restarts the aborted bus cycle(s) in their entirety.
BP[3:2] PM/BP[1:0]	O	The *breakpoint* pins (BP3–0) correspond to the debug registers, DR3–DR0. These pins externally indicate a breakpoint match when the debug registers are programmed to test for breakpoint matches. BP1 and BP0 are multiplexed with the Performance Monitoring pins (PM1 and PM0). The PB1 and PB0 bits in the Debug Mode Control Register determine if the pins are configured as breakpoint or performance monitoring pins. The pins come out of reset configured for performance monitoring.
BRDY#	I	The *burst ready* input indicates that the external system has presented valid data on the data pins in response to a read or that the external system has accepted the Pentium processor (510\60, 567\66) data in response to a write request. This signal is sampled in the T2, T12 and T2P bus states.

Table 2-2. Quick Pin Reference (Continued)

Symbol	Type*	Name and Function
BREQ	O	The *bus request* output indicates to the external system that the Pentium processor (510\60, 567\66) has internally generated a bus request. This signal is always driven whether or not the Pentium processor (510\60, 567\66) is driving its bus.
BT3–BT0	O	The *branch trace* outputs provide bits 2-0 of the branch target linear address (BT2-BT0) and the default operand size (BT3) during a branch trace message special cycle.
BUSCHK#	I	The *bus check* input allows the system to signal an unsuccessful completion of a bus cycle. If this pin is sampled active, the Pentium processor (510\60, 567\66) will latch the address and control signals in the machine check registers. If in addition, the MCE bit in CR4 is set, the Pentium processor (510\60, 567\66) will vector to the machine check exception.
CACHE#	O	For Pentium processor (510\60, 567\66)-initiated cycles the *cache* pin indicates internal cacheability of the cycle (if a read), and indicates a burst writeback cycle (if a write). If this pin is driven inactive during a read cycle, Pentium processor (510\60, 567\66) will not cache the returned data, regardless of the state of the KEN# pin. This pin is also used to determine the cycle length (number of transfers in the cycle).
CLK	I	The *clock* input provides the fundamental timing for the Pentium processor (510\60, 567\66). Its frequency is the internal operating frequency of the Pentium processor (510\60, 567\66) and requires TTL levels. All external timing parameters except TDI, TDO, TMS and TRST# are specified with respect to the rising edge of CLK.
D/C#	O	The *Data/Code* output is one of the primary bus cycle definition pins. It is driven valid in the same clock as the ADS# signal is asserted. D/C# distinguishes between data and code or special cycles.
D63–D0	I/O	These are the 64 *data lines* for the processor. Lines D7-D0 define the least significant byte of the data bus; lines D63-D56 define the most significant byte of the data bus. When the CPU is driving the data lines, they are driven during the T2, T12, or T2P clocks for that cycle. During reads, the CPU samples the data bus when BRDY# is returned.
DP7–DP0	I/O	These are the *data parity* pins for the processor. There is one for each byte of the data bus. They are driven by the Pentium processor (510\60, 567\66) with even parity information on writes in the same clock as write data. Even parity information must be driven back to the Pentium processor (510\60, 567\66) on these pins in the same clock as the data to ensure that the correct parity check status is indicated by the Pentium processor (510\60, 567\66). DP7 applies to D63–D56, DP0 applies to D7–D0.
EADS#	I	This signal indicates that a *valid external address* has been driven onto the Pentium processor (510\60, 567\66) address pins to be used for an inquire cycle.
EWBE#	I	The *external write buffer empty* input, when inactive (high), indicates that a write cycle is pending in the external system. When the Pentium processor (510\60, 567\66) generates a write, and EWBE# is sampled inactive, the Pentium processor (510\60, 567\66) will hold off all subsequent writes to all E or M-state lines in the data cache until all write cycles have completed, as indicated by EWBE# being active.

Table 2-2. Quick Pin Reference (Continued)

Symbol	Type*	Name and Function
FERR#	O	The *floating point error* pin is driven active when an unmasked floating point error occurs. FERR# is similar to the ERROR# pin on the Intel387™ math coprocessor. FERR# is included for compatibility with systems using DOS type floating point error reporting.
FLUSH#	I	When asserted, the *cache flush* input forces the Pentium processor (510\60, 567\66) to writeback all modified lines in the data cache and invalidate its internal caches. A Flush Acknowledge special cycle will be generated by the Pentium processor (510\60, 567\66) indicating completion of the writeback and invalidation. If FLUSH# is sampled low when RESET transitions from high to low, tristate test mode is entered.
FRCMC#	I	The *Functional Redundancy Checking Master/Checker* mode input is used to determine whether the Pentium processor (510\60, 567\66) is configured in master mode or checker mode. When configured as a master, the Pentium processor (510\60, 567\66) drives its output pins as required by the bus protocol. When configured as a checker, the Pentium processor (510\60, 567\66) tristates all outputs (except IERR# and TDO) and samples the output pins. The configuration as a master/checker is set after RESET and may not be changed other than by a subsequent RESET.
HIT#	O	The *hit* indication is driven to reflect the outcome of an inquire cycle. If an inquire cycle hits a valid line in either the Pentium processor (510\60, 567\66) data or instruction cache, this pin is asserted two clocks after EADS# is sampled asserted. If the inquire cycle misses Pentium processor (510\60, 567\66) cache, this pin is negated two clocks after EADS#. This pin changes its value only as a result of an inquire cycle and retains its value between the cycles.
HITM#	O	The *hit to a modified line* output is driven to reflect the outcome of an inquire cycle. It is asserted after inquire cycles which resulted in a hit to a modified line in the data cache. It is used to inhibit another bus master from accessing the data until the line is completely written back.
HLDA	O	The *bus hold acknowledge* pin goes active in response to a hold request driven to the processor on the HOLD pin. It indicates that the Pentium processor (510\60, 567\66) has floated most of the output pins and relinquished the bus to another local bus master. When leaving bus hold, HLDA will be driven inactive and the Pentium processor (510\60, 567\66) will resume driving the bus. If the Pentium processor (510\60, 567\66) has bus cycle pending, it will be driven in the same clock that HLDA is deasserted.
HOLD	I	In response to the *bus hold request*, the Pentium processor (510\60, 567\66) will float most of its output and input/output pins and assert HLDA after completing all outstanding bus cycles. The Pentium processor (510\60, 567\66) will maintain its bus in this state until HOLD is deasserted. HOLD is not recognized during LOCK cycles. The Pentium processor (510\60, 567\66) will recognize HOLD during reset.
IBT	O	The *instruction branch taken* pin is driven active (high) for one clock to indicate that a branch was taken. This output is always driven by the Pentium processor (510\60, 567\66).

Table 2-2. Quick Pin Reference (Continued)

Symbol	Type*	Name and Function
IERR#	O	The *internal error* pin is used to indicate two types of errors, internal parity errors and functional redundancy errors. If a parity error occurs on a read from an internal array, the Pentium processor (510\60, 567\66) will assert the IERR# pin for one clock and then shutdown. If the Pentium processor (510\60, 567\66) is configured as a checker and a mismatch occurs between the value sampled on the pins and the corresponding value computed internally, the Pentium processor (510\60, 567\66) will assert IERR# two clocks after the mismatched value is returned.
IGNNE#	I	This is the *ignore numeric error* input. This pin has no effect when the NE bit in CR0 is set to 1. When the CR0.NE bit is 0, and the IGNNE# pin is asserted, the Pentium processor (510\60, 567\66) will ignore any pending unmasked numeric exception and continue executing floating point instructions for the entire duration that this pin is asserted. When the CR0.NE bit is 0, IGNNE# is not asserted, a pending unmasked numeric exception exists (SW.ES = 1), and the floating point instruction is one of FINIT, FCLEX, FSTENV, FSAVE, FSTSW, FSTCW, FENI, FDISI, or FSETPM, the Pentium processor (510\60, 567\66) will execute the instruction in spite of the pending exception. When the CR0.NE bit is 0, IGNNE# is not asserted, a pending unmasked numeric exception exists (SW.ES = 1), and the floating point instruction is one other than FINIT, FCLEX, FSTENV, FSAVE, FSTSW, FSTCW, FENI, FDISI, or FSETPM, the Pentium processor (510\60, 567\66) will stop execution and wait for an external interrupt.
INIT	I	The Pentium processor (510\60, 567\66) *initialization* input pin forces the Pentium processor (510\60, 567\66) to begin execution in a known state. The processor state after INIT is the same as the state after RESET except that the internal caches, write buffers, and floating point registers retain the values they had prior to INIT. INIT may NOT be used in lieu of RESET after power-up. If INIT is sampled high when RESET transitions from high to low the Pentium processor (510\60, 567\66) will perform built-in self test prior to the start of program execution.
INTR	I	An active *maskable interrupt* input indicates that an external interrupt has been generated. If the IF bit in the EFLAGS register is set, the Pentium processor (510\60, 567\66) will generate two locked interrupt acknowledge bus cycles and vector to an interrupt handler after the current instruction execution is completed. INTR must remain active until the first interrupt acknowledge cycle is generated to assure that the interrupt is recognized.
INV	I	The *invalidation* input determines the final cache line state (S or I) in case of an inquire cycle hit. It is sampled together with the address for the inquire cycle in the clock EADS# is sampled active.
IU	O	The *u-pipe instruction complete* output is driven active (high) for 1 clock to indicate that an instruction in the u-pipeline has completed execution. This pin is always driven by the Pentium processor (510\60, 567\66).
IV	O	The *v-pipe instruction complete* output is driven active (high) for one clock to indicate that an instruction in the v-pipeline has completed execution. This pin is always driven by the Pentium processor (510\60, 567\66).
KEN#	I	The *cache enable* pin is used to determine whether the current cycle is cacheable or not and is consequently used to determine cycle length. When the Pentium processor (510\60, 567\66) generates a cycle that can be cached (CACHE# asserted) and KEN# is active, the cycle will be transformed into a burst line fill cycle.

Table 2-2. Quick Pin Reference (Continued)

Symbol	Type*	Name and Function
LOCK#	O	The *bus lock* pin indicates that the current bus cycle is locked. The Pentium processor (510\60, 567\66) will not allow a bus hold when LOCK# is asserted (but AHOLD and BOFF# are allowed). LOCK# goes active in the first clock of the first locked bus cycle and goes inactive after the BRDY# is returned for the last locked bus cycle. LOCK# is guaranteed to be deasserted for at least one clock between back to back locked cycles.
M/IO#	O	The *Memory/Input-Output* is one of the primary bus cycle definition pins. It is driven valid in the same clock as the ADS# signal is asserted. M/IO# distinguishes between memory and I/O cycles.
NA#	I	An active *next address* input indicates that the external memory system is ready to accept a new bus cycle although all data transfers for the current cycle have not yet completed. The Pentium processor (510\60, 567\66) will drive out a pending cycle two clocks after NA# is asserted. The Pentium processor (510\60, 567\66) supports up to 2 outstanding bus cycles.
NMI	I	The *non-maskable interrupt* request signal indicates that an external non-maskable interrupt has been generated.
PCD	O	The *page cache disable* pin reflects the state of the PCD bit in CR3, the Page Directory Entry, or the Page Table Entry. The purpose of PCD is to provide an external cacheability indication on a page by page basis.
PCHK#	O	The *parity check* output indicates the result of a parity check on a data read. It is driven with parity status two clocks after BRDY# is returned. PCHK# remains low one clock for each clock in which a parity error was detected. Parity is checked only for the bytes on which valid data is returned.
PEN#	I	The *parity enable* input (along with CR4.MCE) determines whether a machine check exception will be taken as a result of a data parity error on a read cycle. If this pin is sampled active in the clock a data parity error is detected, the Pentium processor (510\60, 567\66) will latch the address and control signals of the cycle with the parity error in the machine check registers. If in addition the machine check enable bit in CR4 is set to "1", the Pentium processor (510\60, 567\66) will vector to the machine check exception before the beginning of the next instruction.
PM/BP[1:0]BP[3:2]	O	These pins function as part of the Performance Monitoring feature. The breakpoint pins BP[1:0] are multiplexed with the Performance Monitoring pins PM[1:0]. The PB1 and PB0 bits in the Debug Mode Control Register determine if the pins are configured as breakpoint or performance monitoring pins. The pins come out of reset configured for performance monitoring.
PRDY	O	The PRDY output pin indicates that the processor has stopped normal execution in response to the R/S# pin going active, or Probe Mode being entered.
PWT	O	The *page write through* pin reflects the state of the PWT bit in CR3, the Page Directory Entry, or the Page Table Entry. The PWT pin is used to provide an external writeback indication on a page by page basis.
R/S#	I	The R/S# input is an asynchronous, edge sensitive interrupt used to stop the normal execution of the processor and place it into an idle state. A high to low transition on the R/S# pin will interrupt the processor and cause it to stop execution at the next instruction boundary.

Table 2-2. Quick Pin Reference (Continued)

Symbol	Type*	Name and Function
RESET	I	*Reset* forces the Pentium processor (510\60, 567\66) to begin execution at a known state. All the Pentium processor (510\60, 567\66) internal caches will be invalidated upon the RESET. Modified lines in the data cache are not written back. FLUSH#, FRCMC# and INIT are sampled when RESET transitions from high to low to determine if tristate test mode or checker mode will be entered, or if BIST will be run.
SCYC	O	The *split cycle* output is asserted during misaligned LOCKed transfers to indicate that more than two cycles will be locked together. This signal is defined for locked cycles only. It is undefined for cycles which are not locked.
SMI#	I	The system Management Interrupt causes a system management interrupt request to be latched internally. When the latched SMI# is recognized on an instruction boundary, the processor enters System Management Mode.
SMIACT#	O	An active *system management interrupt active* output indicates that the processor is operating in System Management Mode (SMM).
TCK	I	The *testability clock* input provides the clocking function for the Pentium processor (510\60, 567\66) boundary scan in accordance with the IEEE Boundary Scan interface (Standard 1149.1). It is used to clock state information and data into and out of the Pentium processor (510\60, 567\66) during boundary scan.
TDI	I	The *test data input* is a serial input for the test logic. TAP instructions and data are shifted into the Pentium processor (510\60, 567\66) on the TDI pin on the rising edge of TCK when the TAP controller is in an appropriate state.
TDO	O	The *test data output* is a serial output of the test logic. TAP instructions and data are shifted out of the Pentium processor (510\60, 567\66) on the TDO pin on the falling edge of TCK when the TAP controller is in an appropriate state.
TMS	I	The value of the *test mode select* input signal sampled at the rising edge of TCK controls the sequence of TAP controller state changes.
TRST#	I	When asserted, the *test reset* input allows the TAP controller to be asynchronously initialized.
W/R#	O	*Write/Read* is one of the primary bus cycle definition pins. It is driven valid in the same clock as the ADS# signal is asserted. W/R# distinguishes between write and read cycles.
WB/WT#	I	The writeback/writethrough input allows a data cache line to be defined as write back or write through on a line by line basis. As a result, it determines whether a cache line is initially in the S or E state in the data cache.

NOTE:
*The pins are classified as Input or Output based on their function in Master Mode. See the Functional Redundancy Checking section in the "Error Detection" chapter of the *Pentium™ Processor (510\60, 567\66) User's Manual*, Vol. 1, for further information.

2.4 Pin Reference Tables

Table 2-3. Output Pins

Name	Active Level	When Floated
ADS#	LOW	Bus Hold, BOFF#
APCHK#	LOW	
BE7#-BE0#	LOW	Bus Hold, BOFF#
BREQ	HIGH	
BT3-BT0	n/a	
CACHE#	LOW	Bus Hold, BOFF#
FERR#	LOW	
HIT#	LOW	
HITM#	LOW	
HLDA	HIGH	
IBT	HIGH	
IERR#	LOW	
IU	HIGH	
IV	HIGH	
LOCK#	LOW	Bus Hold, BOFF#
M/IO#, D/C#, W/R#	n/a	Bus Hold, BOFF#
PCHK#	LOW	
BP3-2, PM1/BP1, PM0/BP0	HIGH	
PRDY	HIGH	
PWT, PCD	HIGH	Bus Hold, BOFF#
SCYC	HIGH	Bus Hold, BOFF#
SMIACT#	LOW	
TDO	n/a	All states except Shift-DR and Shift-IR

NOTE:
All output and input/output pins are floated during tri-state test mode and checker mode (except IERR#).

Table 2-4. Input Pins

Name	Active Level	Synchronous/ Asynchronous	Internal resistor	Qualified
A20M#	LOW	Asynchronous		
AHOLD	HIGH	Synchronous		
BOFF#	LOW	Synchronous		
BRDY#	LOW	Synchronous		Bus State T2, T12, T2P
BUSCHK#	LOW	Synchronous	Pullup	BRDY#
CLK	n/a			
EADS#	LOW	Synchronous		
EWBE#	LOW	Synchronous		BRDY#
FLUSH#	LOW	Asynchronous		
FRCMC#	LOW	Asynchronous		
HOLD	HIGH	Synchronous		
IGNNE#	LOW	Asynchronous		
INIT	HIGH	Asynchronous		
INTR	HIGH	Asynchronous		
INV	HIGH	Synchronous		EADS#
KEN#	LOW	Synchronous		First BRDY#/ NA#
NA#	LOW	Synchronous		Bus State T2, TD, T2P
NMI	HIGH	Asynchronous		
PEN#	LOW	Synchronous		BRDY#
R/S#	n/a	Asynchronous	Pullup	
RESET	HIGH	Asynchronous		
SMI#	LOW	Asynchronous	Pullup	
TCK	n/a		Pullup	
TDI	n/a	Synchronous/TCK	Pullup	TCK
TMS	n/a	Synchronous/TCK	Pullup	TCK
TRST#	LOW	Asynchronous	Pullup	
WB/WT#	n/a	Synchronous		First BRDY#/ NA#

Table 2-5. Input/Output Pins

Name	Active Level	When Floated	Qualified (when an input)
A31–A3	n/a	Address hold, Bus Hold, BOFF#	EADS#
AP	n/a	Address hold, Bus Hold, BOFF#	EADS#
D63–D0	n/a	Bus Hold, BOFF#	BRDY#
DP7–DP0	n/a	Bus Hold, BOFF#	BRDY#

NOTE:
All output and input/output pins are floated during tristate test mode (except TDO) and checker mode (except IERR# and TDO).

2.5 Pin Grouping According to Function

Table 2-6 organizes the pins with respect to their function.

Table 2-6. Pin Functional Grouping

Function	Pins
Clock	CLK
Initialization	RESET, INIT
Address Bus	A31–A3, BE7#–BE0#
Address Mask	A20M#
Data Bus	D63–D0
Address Parity	AP, APCHK#
Data Parity	DP7–DP0, PCHK#, PEN#
Internal Parity Error	IERR#
System Error	BUSCHK#
Bus Cycle Definition	M/IO#, D/C#, W/R#, CACHE#, SCYC, LOCK#
Bus Control	ADS#, BRDY#, NA#
Page Cacheability	PCD, PWT
Cache Control	KEN#, WB/WT#
Cache Snooping/ Consistency	AHOLD, EADS#, HIT#, HITM#, INV
Cache Flush	FLUSH#
Write Ordering	EWBE#
Bus Arbitration	BOFF#, BREQ, HOLD, HLDA
Interrupts	INTR, NMI
Floating Point Error Reporting	FERR#, IGNNE#
System Management Mode	SMI#, SMIACT#
Functional Redundancy Checking	FRCMC# (IERR#)
TAP Port	TCK, TMS, TDI, TDO, TRST#
Breakpoint/ Performance Monitoring	PM0/BP0, PM1/BP1, BP3-2
Execution Tracing	BT3–BT0, IU, IV, IBT
Probe Mode	R/S#, PRDY

2.6 Output Pin Grouping According to when Driven

This section groups the output pins according to when they are driven.

Group 1

The following output pins are driven active at the beginning of a bus cycle with ADS#. A31–A3 and AP are guaranteed to remain valid until AHOLD is asserted or until the earlier of the clock after NA# or the last BRDY#. The remaining pins are guaranteed to remain valid until the earlier of the clock after NA# or the last BRDY#:

A31–A3, AP, BE7#–0#, CACHE#, M/IO#, W/R#, D/C#, SCYC, PWT, PCD.

Group 2

As outputs, the following pins are driven in T2, T12, and T2P. As inputs, these pins are sampled with BRDY#:

D63–0, DP7–0.

Group 3

These are the status output pins. They are always driven:

BREQ, HIT#, HITM#, IU, IV, IBT, BT3-BT0, PM0/BP0, PM1/BP1, BP3, BP2, PRDY, SMIACT#.

Group 4

These are the glitch free status output pins.

APCHK#, FERR#, HLDA, IERR#, LOCK#, PCHK#.

3.0 ELECTRICAL SPECIFICATIONS

3.1 Power and Ground

For clean on-chip power distribution, the Pentium processor (510\60, 567\66) has 50 V_{CC} (power) and 49 V_{SS} (ground) inputs. Power and ground connections must be made to all external V_{CC} and V_{SS} pins of the Pentium processor (510\60, 567\66). On the circuit board, all V_{CC} pins must be connected to a V_{CC} plane. All V_{SS} pins must be connected to a V_{SS} plane.

3.2 Decoupling Recommendations

Liberal decoupling capacitance should be placed near the Pentium processor (510\60, 567\66). The Pentium processor (510\60, 567\66) driving its large address and data buses at high frequencies can cause transient power surges, particularly when driving large capacitive loads.

Low inductance capacitors (i.e. surface mount capacitors) and interconnects are recommended for best high frequency electrical performance. Inductance can be reduced by connecting capacitors directly to the V_{CC} and V_{SS} planes, with minimal trace length between the component pads and vias to the plane. Capacitors specifically for PGA packages are also commercially available.

These capacitors should be evenly distributed among each component. Capacitor values should be chosen to ensure they eliminate both low and high frequency noise components.

3.3 Connection Specifications

All NC pins must remain unconnected.

For reliable operation, always connect unused inputs to an appropriate signal level. Unused active low inputs should be connected to V_{CC}. Unused active high inputs should be connected to ground.

3.4 Maximum Ratings

Table 3-1 is a stress rating only. Functional operation at the maximums is not guaranteed. Functional operating conditions are given in the A.C. and D.C. specification tables.

Extended exposure to the maximum ratings may affect device reliability. Furthermore, although the Pentium processor (510\60, 567\66) contains protective circuitry to resist damage from static electric discharge, always take precautions to avoid high static voltages or electric fields.

Table 3-1. Absolute Maximum Ratings

Case temperature under bias	−65°C to +110°C
Storage temperature	−65°C to +150°C
Voltage on any pin with respect to ground	−0.5 V_{CC} to V_{CC} + 0.5 (V)
Supply voltage with respect to V_{SS}	−0.5V to +6.5V

2

3.5 D.C. Specifications

Table 3-2 lists the D.C. specifications associated with the Pentium processor (510\60, 567\66).

Table 3-2. Pentium™ Processor (510\60, 567\66) D.C. Specifications
V_{CC} = See Notes 10, 11; T_{case} = See Notes 12, 13

Symbol	Parameter	Min	Max	Unit	Notes
V_{IL}	Input Low Voltage	−0.3	+0.8	V	TTL Level
V_{IH}	Input High Voltage	2.0	V_{CC} + 0.3	V	TTL Level
V_{OL}	Output Low Voltage		0.45	V	TTL Level, (1)
V_{OH}	Output High Voltage	2.4		V	TTL Level, (2)
I_{CC}	Power Supply Current		3200 2910	mA mA	66 MHz, (7), (8) 60 MHz, (7), (9)
I_{LI}	Input Leakage Current		±15	uA	$0 \le V_{IN} \le V_{CC}$, (4)
I_{LO}	Output Leakage Current		±15	uA	$0 \le V_{OUT} \le V_{CC}$ Tristate, (4)
I_{IL}	Input Leakage Current		−400	uA	V_{IN} = 0.45V, (5)
I_{IH}	Input Leakage Current		200	uA	V_{IN} = 2.4V, (6)
C_{IN}	Input Capacitance		15	pF	
C_O	Output Capacitance		20	pF	
$C_{I/O}$	I/O Capacitance		25	pF	
C_{CLK}	CLK Input Capacitance		8	pF	
C_{TIN}	Test Input Capacitance		15	pF	
C_{TOUT}	Test Output Capacitance		20	pF	
C_{TCK}	Test Clock Capacitance		8	pF	

NOTES:
1. Parameter measured at 4 mA load.
2. Parameter measured at 1 mA load.
4. This parameter is for input without pullup or pulldown.
5. This parameter is for input with pullup.
6. This parameter is for input with pulldown.
7. Worst case average I_{CC} for a mix of test patterns.
8. (16W max.) Typical Pentium processor (510\60, 567\66) supply current is 2600 mA (13W) at 66 MHz.
9. (14.6W max.) Typical Pentium processor (510\60, 567\66) supply current is 2370 mA (11.9W) at 60 MHz.
10. V_{CC} = 5V ± 5% at 60 MHz.
11. V_{CC} = 4.9V to 5.40V at 66 MHz.
12. T_{case} = 0°C to +80°C at 60 MHz.
13. T_{case} = 0°C to +70°C at 66 MHz.

3.6 A.C. Specifications

The 66 MHz and 60 MHz A.C. specifications given in Tables 3-3 and 3-4 consist of output delays, input setup requirements and input hold requirements. All A.C. specifications (with the exception of those for the TAP signals) are relative to the rising edge of the CLK input.

All timings are referenced to 1.5 volts for both "0" and "1" logic levels unless otherwise specified. Within the sampling window, a synchronous input must be stable for correct Pentium processor (510\60, 567\66) operation.

Care should be taken to read all notes associated with a particular timing parameter. In addition, the following list of notes apply to the timing specification tables in general and are not associated with any one timing. They are 2, 5, 6, and 14.

Table 3-3. 66 MHz Pentium™ Processor (510\60, 567\66) A.C. Specifications

V_{CC} = 4.9V to 5.40V; T_{case} = 0°C to +70°C; C_L = 0 pF

Symbol	Parameter	Min	Max	Unit	Figure	Notes
	Frequency	33.33	66.66	MHz		1x CLK
t_1	CLK Period	15		ns	3.1	
t_{1a}	CLK Period Stability		±250	ps		(18), (19), (20), (21)
t_2	CLK High Time	4		ns	3.1	@2V, (1)
t_3	CLK Low Time	4		ns	3.1	@0.8V, (1)
t_4	CLK Fall Time	0.15	1.5	ns	3.1	(2.0V–0.8V), (1)
t_5	CLK Rise Time	0.15	1.5	ns	3.1	(0.8V–2.0V), (1)
t_6	ADS#, A3–A31, BT0–3, PWT, PCD, BE0–7#, M/IO#, D/C#, W/R#, CACHE#, SCYC, LOCK# Valid Delay	1.5	8.0	ns	3.2	
t_{6a}	AP Valid Delay	1.5	9.5	ns	3.2	
t_7	ADS#, AP, A3–A31, BT0–3, PWT, PCD, BE0–7#, M/IO#, D/C#, W/R#, CACHE#, SCYC, LOCK# Float Delay		10	ns	3.3	(1)
t_8	PCHK#, APCHK#, IERR#, FERR# Valid Delay	1.5	8.3	ns	3.2	(4)
t_9	BREQ, HLDA, SMIACT# Valid Delay	1.5	8.0	ns	3.2	(4)
t_{10}	HIT#, HITM# Valid Delay	1.5	8.0	ns	3.2	
t_{11}	PM0–1, BP0–3, IU, IV, IBT Valid Delay	1.5	10	ns	3.2	
t_{11a}	PRDY Valid Delay	1.5	8.0	ns	3.2	
t_{12}	D0–D63, DP0–7 Write Data Valid Delay	1.5	9	ns	3.2	
t_{13}	D0–63, DP0–7 Write Data Float Delay		10	ns	3.3	(1)
t_{14}	A5–A31 Setup Time	6.5		ns	3.4	
t_{15}	A5–A31 Hold Time	1.5		ns	3.4	
t_{16}	EADS#, INV, AP Setup Time	5		ns	3.4	
t_{17}	EADS#, INV, AP Hold Time	1.5		ns	3.4	
t_{18}	KEN#, WB/WT# Setup Time	5		ns	3.4	
t_{18a}	NA# Setup Time	4.5		ns	3.4	
t_{19}	KEN#, WB/WT#, NA# Hold Time	1.5		ns	3.4	
t_{20}	BRDY# Setup Time	5		ns	3.4	
t_{21}	BRDY# Hold Time	1.5		ns	3.4	
t_{22}	AHOLD, BOFF# Setup Time	5.5		ns	3.4	
t_{23}	AHOLD, BOFF# Hold Time	1.5		ns	3.4	

Table 3-3. 66 MHz Pentium™ Processor (510\60, 567\66) A.C. Specifications
V_{CC} = 4.9V to 5.40V; T_{case} = 0°C to +70°C; C_L = 0 pF (Continued)

Symbol	Parameter	Min	Max	Unit	Figure	Notes
t_{24}	BUSCHK#, EWBE#, HOLD, PEN# Setup Time	5		ns	3.4	
t_{25}	BUSCHK#, EWBE#, HOLD, PEN# Hold Time	1.5		ns	3.4	
t_{26}	A20M#, INTR, Setup Time	5		ns	3.4	(12), (16)
t_{27}	A20M#, INTR, Hold Time	1.5		ns	3.4	(13)
t_{28}	INIT, FLUSH#, NMI, SMI#, IGNNE# Setup Time	5		ns	3.4	(16), (17)
t_{29}	INIT, FLUSH#, NMI, SMI#, IGNNE# Hold Time	1.5		ns	3.4	
t_{30}	INIT, FLUSH#, NMI, SMI#, IGNNE# Pulse Width, Async	2		CLKs		(15), (17)
t_{31}	R/S# Setup Time	5		ns	3.4	(12), (16), (17)
t_{32}	R/S# Hold Time	1.5		ns	3.4	(13)
t_{33}	R/S# Pulse Width, Async.	2		CLKs		(15), (17)
t_{34}	D0–D63 Read Data Setup Time	3.8		ns	3.4	
t_{34a}	DP0–7 Read Data Setup Time	3.8		ns	3.4	
t_{35}	D0–D63, DP0–7 Read Data Hold Time	2		ns	3.4	
t_{36}	RESET Setup Time	5		ns	3.5	(11), (12), (16)
t_{37}	RESET Hold Time	1.5		ns	3.5	(11), (13)
t_{38}	RESET Pulse Width, V_{CC} & CLK Stable	15		CLKs	3.5	(11)
t_{39}	RESET Active After V_{CC} & CLK Stable	1		ms	3.5	power up, (11)
t_{40}	Pentium processor (510\60, 567\66) Reset Configuration Signals (INIT, FLUSH#, FRCMC#) Setup Time	5		ns	3.5	(12), (16), (17)
t_{41}	Pentium processor (510\60, 567\66) Reset Configuration Signals (INIT, FLUSH#, FRCMC#) Hold Time	1.5		ns	3.5	(13)
t_{42}	Pentium processor (510\60, 567\66) Reset Configuration Signals (INIT, FLUSH#, FRCMC#) Setup Time, Async.	2		CLKs	3.5	(16)
t_{43}	Pentium processor (510\60, 567\66) Reset Configuration Signals (INIT, FLUSH#, FRCMC#) Hold Time, Async.	2		CLKs	3.5	
t_{44}	TCK Frequency		16	MHz		
t_{45}	TCK Period	62.5		ns	3.1	

Table 3-3. 66 MHz Pentium™ Processor (510\60, 567\66) A.C. Specifications
V_{CC} = 4.9V to 5.40V; T_{case} = 0°C to +70°C; C_L = 0 pF (Continued)

Symbol	Parameter	Min	Max	Unit	Figure	Notes
t_{46}	TCK High Time	25		ns	3.1	@2V, (1)
t_{47}	TCK Low Time	25		ns	3.1	@0.8V, (1)
t_{48}	TCK Fall Time		5	ns	3.1	(2.0V–0.8V), (1), (8), (9)
t_{49}	TCK Rise Time		5	ns	3.1	(0.8V–2.0V), (1), (8), (9)
t_{50}	TRST# Pulse Width	40		ns	3.7	Asynchronous, (1)
t_{51}	TDI, TMS Setup Time	5		ns	3.6	(7)
t_{52}	TDI, TMS Hold Time	13		ns	3.6	(7)
t_{53}	TDO Valid Delay	3	20	ns	3.6	(8)
t_{54}	TDO Float Delay		25	ns	3.6	(1), (8)
t_{55}	All Non-Test Outputs Valid Delay	3	20	ns	3.6	(3), (8), (10)
t_{56}	All Non-Test Outputs Float Delay		25	ns	3.6	(1), (3), (8), (10)
t_{57}	All Non-Test Inputs Setup Time	5		ns	3.6	(3), (7), (10)
t_{58}	All Non-Test Inputs Hold Time	13		ns	3.6	(3), (7), (10)

NOTES:
1. Not 100% tested. Guaranteed by design/characterization.
2. TTL input test waveforms are assumed to be 0 to 3 Volt transitions with 1Volt/ns rise and fall times.
3. Non-Test Outputs and Inputs are the normal output or input signals (besides TCK, TRST#, TDI, TDO, and TMS). These timings correspond to the response of these signals due to boundary scan operations.
4. APCHK#, FERR#, HLDA, IERR#, LOCK#, and PCHK# are glitch free outputs. Glitch free signals monotonically transition without false transitions (i.e. glitches).
5. 0.8 V/ns ≤ CLK input rise/fall time ≤ 8 V/ns.
6. 0.3 V/ns ≤ Input rise/fall time ≤ 5 V/ns.
7. Referenced to TCK rising edge.
8. Referenced to TCK falling edge.
9. 1 ns can be added to the maximum TCK rise and fall times for every 10 MHz of frequency below 16 MHz.
10. During probe mode operation, use the normal specified timings. Do not use the boundary scan timings ($t_{55\text{-}58}$).
11. FRCMC# should be tied to V_{CC} (high) to ensure proper operation of the Pentium processor (510\60, 567\66) as a master Pentium processor (510\60, 567\66).
12. Setup time is required to guarantee recognition on a specific clock.
13. Hold time is required to guarantee recognition on a specific clock.
14. All TTL timings are referenced from 1.5 V.
15. To guarantee proper asynchronous recognition, the signal must have been deasserted (inactive) for a minimum of 2 clocks before being returned active and must meet the minimum pulse width.
16. This input may be driven asynchronously.
17. When driven asynchronously, NMI, FLUSH#, R/S#, INIT, and SMI# must be deasserted (inactive) for a minimum of 2 clocks before being returned active.
18. Functionality is guaranteed by design/characterization.
19. Measured on rising edge of adjacent CLKs at 1.5V.
20. To ensure a 1:1 relationship between the amplitude of the input jitter and the internal and external clocks, the jitter frequency spectrum should not have any power spectrum peaking between 500 KHz and 1/3 of the CLK operating frequency.
21. The amount of jitter present must be accounted for as a component of CLK skew between devices.

Table 3-4. 60 MHz Pentium™ Processor (510\60, 567\66) A.C. Specifications

V_{CC} = 5V ±5%; T_{case} = 0°C to +80°C; C_L = 0 pF

Symbol	Parameter	Min	Max	Unit	Figure	Notes
	Frequency	33.33	60	MHz		1x CLK
t_1	CLK Period	16.67		ns	3.1	
t_{1a}	CLK Period Stability		±250	ps		(18), (19), (20), (21)
t_2	CLK High Time	4		ns	3.1	@2V, (1)
t_3	CLK Low Time	4		ns	3.1	@0.8V,(1)
t_4	CLK Fall Time	0.15	1.5	ns	3.1	(2.0V–0.8V), (1)
t_5	CLK Rise Time	0.15	1.5	ns	3.1	(0.8V–2.0V), (1)
t_6	ADS#, A3–A31, BT0–3, PWT, PCD, BE0–7#, M/IO#, D/C#, W/R#, CACHE#, SCYC, LOCK# Valid Delay	1.5	9.0	ns	3.2	
t_{6a}	AP Valid Delay	1.5	10.5	ns	3.2	
t_7	ADS#, AP, A3–A31, BT0–3, PWT, PCD, BE0–7#, M/IO#, D/C#, W/R#, CACHE#, SCYC, LOCK# Float Delay		11	ns	3.3	(1)
t_8	PCHK#, APCHK#, IERR#, FERR# Valid Delay	1.5	9.3	ns	3.2	(4)
t_9	BREQ, HLDA, SMIACT# Valid Delay	1.5	9.0	ns	3.2	(4)
t_{10}	HIT#, HITM# Valid Delay	1.5	9.0	ns	3.2	
t_{11}	PM0–1, BP0–3, IU, IV, IBT Valid Delay	1.5	11	ns	3.2	
t_{11a}	PRDY Valid Delay	1.5	9.0	ns	3.2	
t_{12}	D0–D63, DP0–7 Write Data Valid Delay	1.5	10	ns	3.2	
t_{13}	D0–D63, DP0–7 Write Data Float Delay		11	ns	3.3	(1)
t_{14}	A5–A31 Setup Time	7		ns	3.4	
t_{15}	A5–A31 Hold Time	1.5		ns	3.4	
t_{16}	EADS#, INV, AP Setup Time	5.5		ns	3.4	
t_{17}	EADS#, INV, AP Hold Time	1.5		ns	3.4	
t_{18}	KEN#, WB/WT# Setup Time	5.5		ns	3.4	
t_{18a}	NA# Setup Time	5.0		ns	3.4	
t_{19}	KEN#, WB/WT#, NA# Hold Time	1.5		ns	3.4	
t_{20}	BRDY# Setup Time	5.5		ns	3.4	
t_{21}	BRDY# Hold Time	1.5		ns	3.4	
t_{22}	AHOLD, BOFF# Setup Time	6		ns	3.4	
t_{23}	AHOLD, BOFF# Hold Time	1.5		ns	3.4	
t_{24}	BUSCHK#, EWBE#, HOLD, PEN# Setup Time	5.5		ns	3.4	
t_{25}	BUSCHK#, EWBE#, HOLD, PEN# Hold Time	1.5		ns	3.4	

Table 3-4. 60 MHz Pentium™ Processor (510\60, 567\66) A.C. Specifications
V_{CC} = 5V ±5%; T_{CASE} = 0°C to +80°C; C_L = 0 pF (Continued)

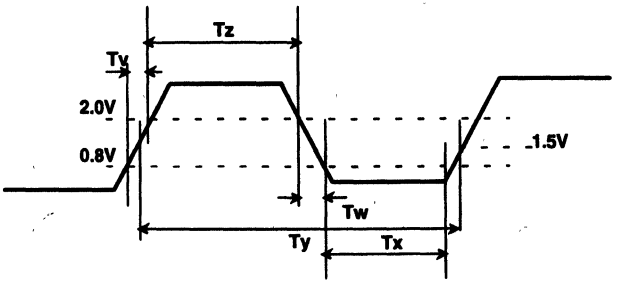
Symbol	Parameter	Min	Max	Unit	Figure	Notes
t_{26}	A20M#, INTR, Setup Time	5.5		ns	3.4	(12), (16)
t_{27}	A20M#, INTR, Hold Time	1.5		ns	3.4	(13)
t_{28}	INIT, FLUSH#, NMI, SMI#, IGNNE# Setup Time	5.5		ns	3.4	(16), (17)
t_{29}	INIT, FLUSH#, NMI, SMI#, IGNNE# Hold Time	1.5		ns	3.4	
t_{30}	INIT, FLUSH#, NMI, SMI#, IGNNE# Pulse Width, Async	2		CLKs		(15), (17)
t_{31}	R/S# Setup Time	5.5		ns	3.4	(12), (16), (17)
t_{32}	R/S# Hold Time	1.5		ns	3.4	(13)
t_{33}	R/S# Pulse Width, Async.	2		CLKs		(15), (17)
t_{34}	D0–D63 Read Data Setup Time	4.3		ns	3.4	
t_{34a}	DP0–7 Read Data Setup Time	4.3		ns	3.4	
t_{35}	D0–D63, DP0–7 Read Data Hold Time	2		ns	3.4	
t_{36}	RESET Setup Time	5.5		ns	3.5	(11), (12), (16)
t_{37}	RESET Hold Time	1.5		ns	3.5	(11), (13)
t_{38}	RESET Pulse Width, V_{CC} & CLK Stable	15		CLKs	3.5	(11)
t_{39}	RESET Active after V_{CC} & CLK Stable	1		ms	3.5	Power Up, (11)
t_{40}	Pentium Processor (510\60, 567\66) Reset Configuration Signals (INIT, FLUSH#, FRCMC#) Setup Time	5.5		ns	3.5	(12), (16), (17)
t_{41}	Pentium Processor (510\60, 567\66) Reset Configuration Signals (INIT, FLUSH#, FRCMC#) Hold Time	1.5		ns	3.5	(13)
t_{42}	Pentium Processor (510\60, 567\66) Reset Configuration Signals (INIT, FLUSH#, FRCMC#) Setup Time, Async.	2		CLKs	3.5	(16)
t_{43}	Pentium Processor (510\60, 567\66) Reset Configuration Signals (INIT, FLUSH#, FRCMC#) Hold Time, Async.	2		CLKs	3.5	
t_{44}	TCK Frequency		16	MHz		
t_{45}	TCK Period	62.5		ns	3.1	
t_{46}	TCK High Time	25		ns	3.1	@2V, (1)
t_{47}	TCK Low Time	25		ns	3.1	@0.8V, (1)
t_{48}	TCK Fall Time		5	ns	3.1	(2.0V–0.8V), (1), (8), (9)
t_{49}	TCK Rise Time		5	ns	3.1	(0.8V–2.0V), (1), (8), (9)

Table 3-4. 60 MHz Pentium™ Processor (510\60, 567\66) A.C. Specifications
V_{CC} = 5V ±5%; T_{CASE} = 0°C to +80°C; C_L = 0 pF (Continued)

Symbol	Parameter	Min	Max	Unit	Figure	Notes
t_{50}	TRST# Pulse Width	40		ns	3.7	Async, (1)
t_{51}	TDI, TMS Setup Time	5		ns	3.6	(7)
t_{52}	TDI, TMS Hold Time	13		ns	3.6	(7)
t_{53}	TDO Valid Delay	3	20	ns	3.6	(8)
t_{54}	TDO Float Delay		25	ns	3.6	(1), (8)
t_{55}	All Non-Test Outputs Valid Delay	3	20	ns	3.6	(3), (8), (10)
t_{56}	All Non-Test Outputs Float Delay		25	ns	3.6	(1), (3), (8), (10)
t_{57}	All Non-Test Inputs Setup Time	5		ns	3.6	(3), (7), (10)
t_{58}	All Non-Test Inputs Hold Time	13		ns	3.6	(3), (7), (10)

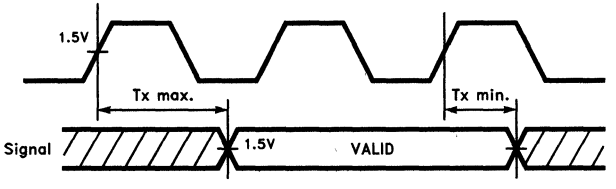
NOTES:
1. Not 100% tested. Guaranteed by design/characterization.
2. TTL input test waveforms are assumed to be 0 to 3 Volt transitions with 1Volt/ns rise and fall times.
3. Non-Test Outputs and Inputs are the normal output or input signals (besides TCK, TRST#, TDI, TDO, and TMS). These timings correspond to the response of these signals due to boundary scan operations.
4. APCHK#, FERR#, HLDA, IERR#, LOCK#, and PCHK# are glitch free outputs. Glitch free signals monotonically transition without false transitions (i.e. glitches).
5. 0.8 V/ns ≤ CLK input rise/fall time ≤ 8 V/ns.
6. 0.3 V/ns ≤ Input rise/fall time ≤ 5 V/ns.
7. Referenced to TCK rising edge.
8. Referenced to TCK falling edge.
9. 1 ns can be added to the maximum TCK rise and fall times for every 10 MHz of frequency below 16 MHz.
10. During probe mode operation, use the normal specified timings. Do not use the boundary scan timings ($t_{55\text{-}58}$).
11. FRCMC# should be tied to V_{CC} (high) to ensure proper operation of the Pentium processor (510\60, 567\66) as a master Pentium processor (510\60, 567\66).
12. Setup time is required to guarantee recognition on a specific clock.
13. Hold time is required to guarantee recognition on a specific clock.
14. All TTL timings are referenced from 1.5 V.
15. To guarantee proper asynchronous recognition, the signal must have been deasserted (inactive) for a minimum of 2 clocks before being returned active and must meet the minimum pulse width.
16. This input may be driven asynchronously.
17. When driven asynchronously, NMI, FLUSH#, R/S#, INIT, and SMI# must be deasserted (inactive) for a minimum of 2 clocks before being returned active.
18. Functionality is guaranteed by design/characterization.
19. Measured on rising edge of adjacent CLKs at 1.5V.
20. To ensure a 1:1 relationship between the amplitude of the input jitter and the internal and external clocks, the jitter frequency spectrum should not have any power spectrum peaking between 500 KHz and 1/3 of the CLK operating frequency.
21. The amount of jitter present must be accounted for as a component of CLK skew between devices.

Each valid delay is specified for a 0 pF load. The system designer should use I/O buffer modeling to account for signal flight time delays.



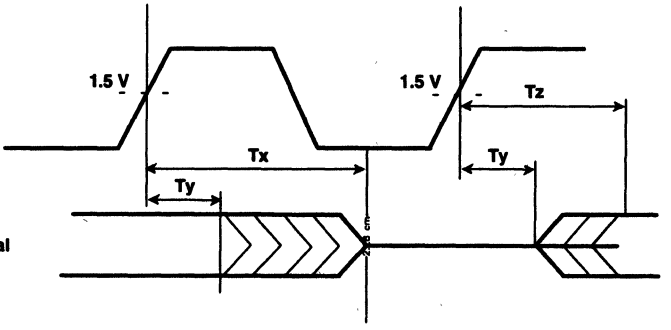
Tv = t5, t49
Tw = t4, t48
Tx = t3, t47
Ty = t1, t45
Tz = t2, t46

Figure 3-1. Clock Waveform



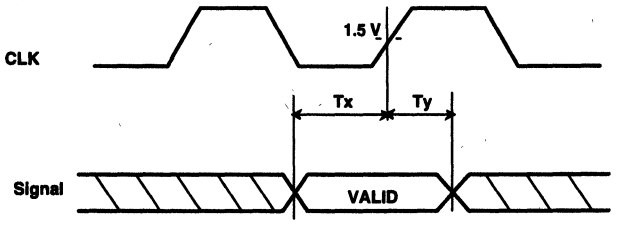
Tx = t6, t6a, t8, t9, t10, t11, t11a, t12

Figure 3-2. Valid Delay Timings



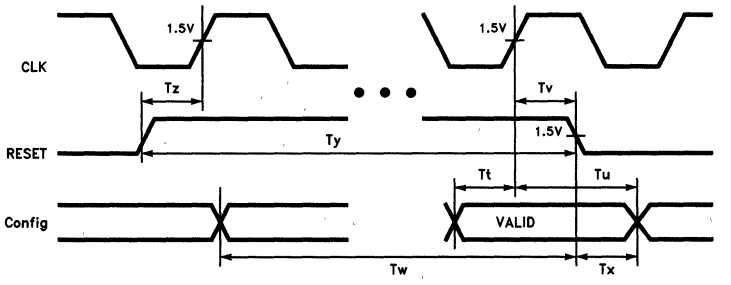
Tx = t7, t13
Ty = t6min, t12min

Figure 3-3. Float Delay Timings



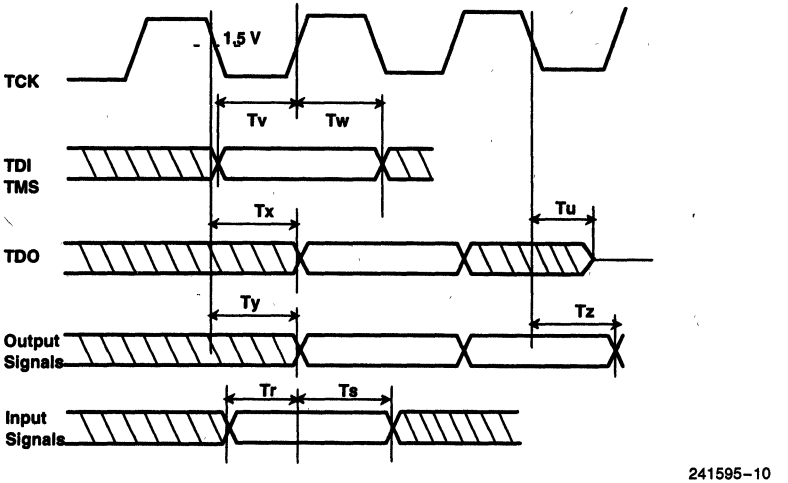
Tx = t14, t16, t18, t18a, t20, t22, t24, t26, t28, t31, t34, t34a
Ty = t15, t17, t19, t21, t23, t25, t27, t29, t32, t35

Figure 3-4. Setup and Hold Timings



Tt = t40
Tu = t41
Tv = t37
Tw = t42
Tx = t43
Ty = t38, t39
Tz = t36

Figure 3-5. Reset and Configuration Timings



Tr = t57
Ts = t58
Tu = t54
Tv = t51
Tw = t52
Tx = t53
Ty = t55
Tz = t56

Figure 3-6. Test Timings

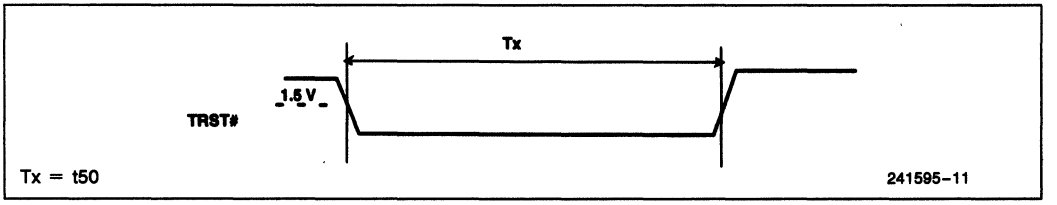


Figure 3-7. Test Reset Timings

4.0 MECHANICAL SPECIFICATIONS

The Pentium processor (510\60, 567\66) is packaged in a 273 pin ceramic pin grid array (PGA). The pins are arranged in a 21 by 21 matrix and the package dimensions are 2.16" × 2.16" (Table 4-1).

Figure 4-1 shows the package dimensions for the Pentium processor (510\60, 567\66). The mechanical specifications are provided in Table 4-2.

Table 4-1. Pentium™ Processor (510\60, 567\66) Package Information Summary

	Package Type	Total Pins	Pin Array	Package Size	Estimated Wattage
Pentium Processor (510\60, 567\66)	PGA	273	21 × 21	2.16" × 2.16" 5.49 cm × 5.49 cm	16

NOTE:
See D.C. Specifications for more detailed power specifications.

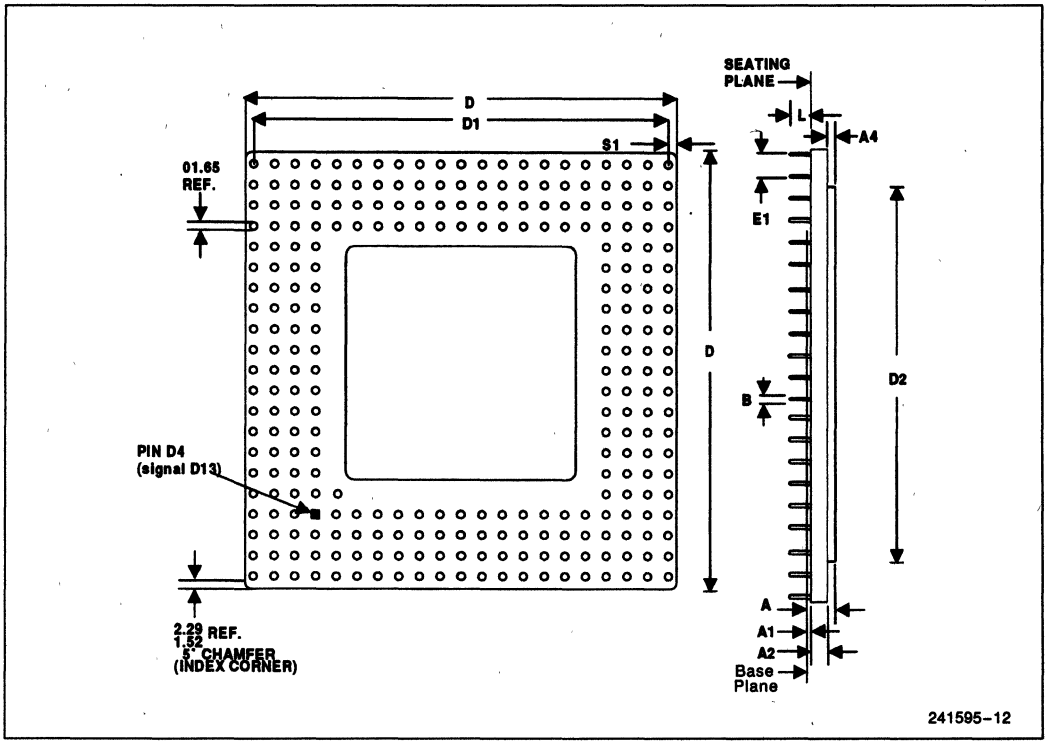


Figure 4-1. Pentium™ Processor (510\60, 567\66) Package Dimensions

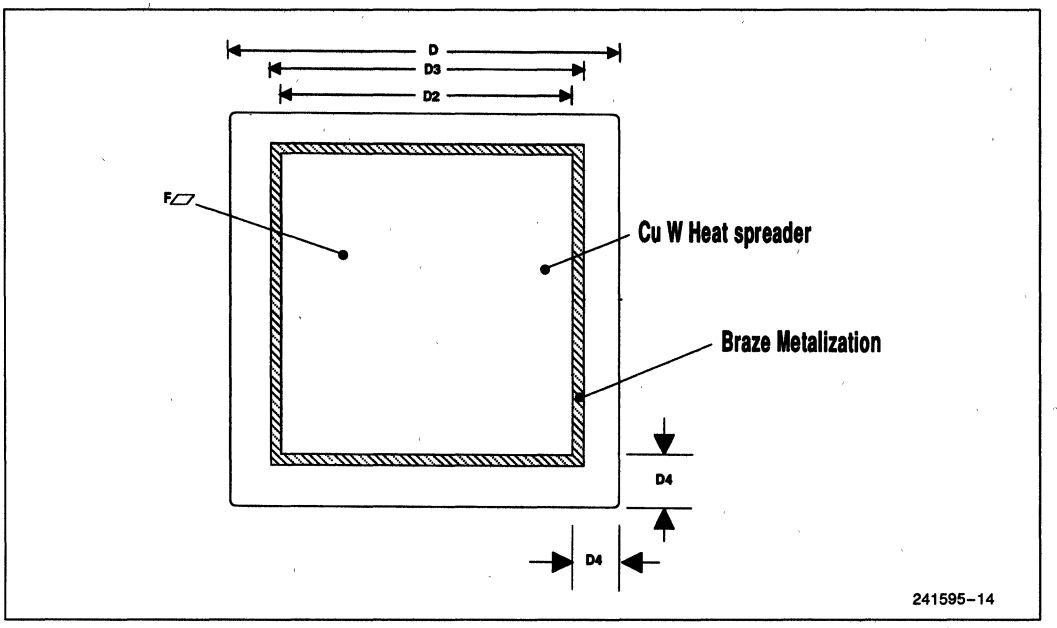


Figure 4-2. Pentium™ Processor (510\60, 567\66) Package Dimensions

Table 4-2. Pentium™ Processor (510\60, 567\66) Mechanical Specifications

Family: Ceramic Pin Grid Array Package						
Symbol	Millimeters			Inches		
	Min	Max	Notes	Min	Max	Notes
A	3.91	4.70	Solid Lid	0.154	0.185	Solid Lid
A1	0.38	0.43	Solid Lid	0.015	0.017	Solid Lid
A2	2.62	4.30		0.103	0.117	
A4	0.97	1.22		0.038	0.048	
B	0.43	0.51		0.017	0.020	
D	54.66	55.07		2.152	2.168	
D1	50.67	50.93		1.995	2.005	
D2	37.85	38.35	Spreader Size	1.490	1.510	Spreader Size
D3	40.335	40.945	Braze	1.588	1.612	Braze
D4	8.382			0.330		
E1	2.29	2.79		0.090	0.110	
F	0.127		Flatness of spreader measured diagonally		0.005	Flatness of spreader measured diagonally
L	2.54	3.30		0.120	0.130	
N	273		Total Pins	273		Total Pins
S1	1.651	2.16		0.065	0.085	

5.0 THERMAL SPECIFICATIONS

The Pentium processor (510\60, 567\66) is specified for proper operation when T_C (case temperature) is within the specified range. To verify that the proper T_C is maintained, it should be measured at the center of the top surface (opposite of the pins) of the device in question. To minimize the measurement errors, it is recommended to use the following approach:

- Use 36 gauge or finer diameter k, t, or j type thermocouples. The laboratory testing was done using a thermocouple made by Omega (part number: 5TC-TTK-36-36).
- Attach the thermocouple bead or junction to the center of the package top surface using high thermal conductivity cements. The laboratory testing was done by using Omega Bond (part number: OB-100).
- The thermocouple should be attached at a 90 degrees angle as shown in Figure 5-1. When a heat sink is attached, a hole (no larger than 0.15") should be drilled through the heat sink to allow probing the center of the package as shown in Figure 5-1.
- If the case temperature is measured with a heat sink attached to the package, drill a hole through the heat sink to route the thermocouple wire out.

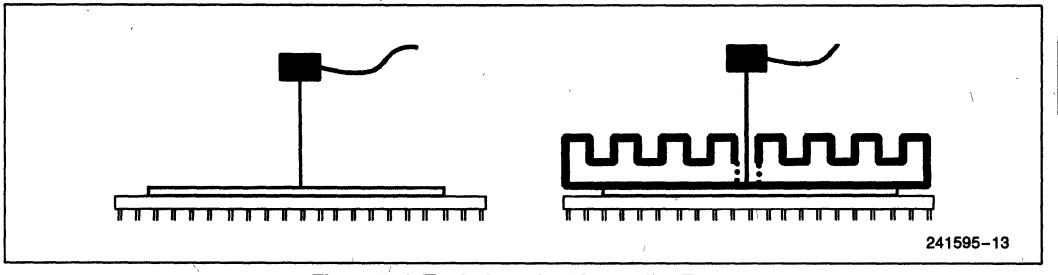


Figure 5-1. Technique for Measuring T_{case}

An ambient temperature T_A is not specified directly. The only restriction is that T_C is met. To determine the allowable T_A values, the following equations may be used:

$$T_J = TC + (P * \theta_{JC})$$

$$T_A = T_J - (P * \theta_{JA})$$

$$\theta_{CA} = \theta_{JA} - \theta_{JC}$$

$$T_A = T_C - (P * \theta_{CA})$$

where, T_J, T_A, and T_C = Junction, Ambient and Case Temperature, respectively.

θ_{JC}, θ_{JA}, and θ_{CA} = Junction-to-Case, Junction-to-Ambient, and Case-to-Ambient Thermal Resistance, respectively.

P = Maximum Power Consumption

Table 5-1 lists the θ_{JC} and θ_{CA} values for the Pentium processor (510\60, 567\66).

Table 5-1. Junction-to-Case and Case-to-Ambient Thermal Resistances for the Pentium™ Processor (510\60, 567\66) (With and Without a Heat Sink)

	θ_{JC}	θ_{CA} vs Airflow (ft/min)					
		0	200	400	600	800	1000
With 0.25" Heat Sink	0.6	8.3	5.4	3.5	2.6	2.1	1.8
With 0.35" Heat Sink	0.6	7.4	4.5	3.0	2.2	1.8	1.6
With 0.65" Heat Sink	0.6	5.9	3.0	1.9	1.5	1.2	1.1
Without Heat Sink	1.2	10.5	7.9	5.5	3.8	2.8	2.4

NOTES:
1. Heat Sink: 2.1 sq. in. base, omni-directional pin Al heat sink with 0.050 in. pin width, 0.143 in pin-to-pin center spacing and 0.150 in. base thickness. Heat sinks are attached to the package with a 2 to 4 mil thick layer of typical thermal grease. The thermal conductivity of this grease is about 1.2 w/m °C.
2. θ_{CA} values shown in Table 5-1. are typical values.The actual θ_{CA} values depend on the air flow in the system (which is typically unsteady, non uniform and turbulent) and thermal interactions between Pentium™ processor (510\60, 567\66) and surrounding components though PCB and the ambient.

PENTIUM™ PROCESSOR AT iCOMP™ INDEX 610\75 MHz

- **Compatible with Large Software Base**
  - **MS-DOS‡, Windows‡, OS/2‡, UNIX‡**
- **32-Bit CPU with 64-Bit Data Bus**
- **Superscalar Architecture**
  - **Two Pipelined Integer Units Are Capable of 2 Instructions/Clock**
  - **Pipelined Floating Point Unit**
- **Separate Code and Data Caches**
  - **8K Code, 8K Writeback Data**
  - **MESI Cache Protocol**
- **Advanced Design Features**
  - **Branch Prediction**
  - **Virtual Mode Extensions**
- **3.3V BiCMOS Silicon Technology**
- **4M Pages for Increased TLB Hit Rate**
- **IEEE 1149.1 Boundary Scan**
- **Internal Error Detection Features**
- **SL Enhanced Power Management Features**
  - **System Management Mode**
  - **Clock Control**
- **Fractional Bus Operation**
  - **75-MHz Core/50-MHz Bus**

The Pentium™ processor is fully compatible with the entire installed base of applications for DOS‡, Windows‡, OS/2‡, and UNIX‡, and all other software that runs on any earlier Intel 8086 family product. The Pentium processor's superscalar architecture can execute two instructions per clock cycle. Branch prediction and separate caches also increase performance. The pipelined floating-point unit delivers workstation level performance. Separate code and data caches reduce cache conflicts while remaining software transparent. The Pentium processor (610\75) has 3.3 million transistors, is built on Intel's advanced 3.3V BiCMOS silicon technology, and has full SL Enhanced power management features, including System Management Mode (SMM) and clock control. The additional SL Enhanced features, 3.3V operation, and the TCP package, which are not available in the Pentium processor (510\60, 567\66), make the Pentium proces sor (610\75) TCP ideal for enabling mobile Pentium processor designs.

242323-21

‡ Other brands and trademarks are the property of their respective owners.

December 1994
Order Number: 242323-001

PENTIUM™ PROCESSOR AT iCOMP™ INDEX 610\75 MHz

CONTENTS

CONTENTS

1.0 INTRODUCTION

Intel is now manufacturing its latest version of the Pentium™ processor family that is designed specifically for mobile systems, with a core frequency of 75 MHz and a bus frequency of 50 MHz. The Pentium processor (610\75) is provided in the TCP (Tape Carrier Package) and SPGA packages, and has all of the advanced features of the Pentium processor (735\90, 815\100).

The new Pentium processor (610\75) TCP package has several features which allow high-performance notebooks to be designed with the Pentium processor, including the following:

- TCP package dimensions are ideal for small form-factor designs.
- The TCP package has superior thermal resistance characteristics.
- 3.3V V_{CC} reduces power consumption by half (in both the TCP and SPGA packages).
- The SL Enhanced feature set, which was initially implemented in the Intel486™ CPU.

The architecture and internal features of the Pentium processor (610\75) TCP and SPGA packages are identical to those of the Pentium processor (735\90, 815\100), although several features have been eliminated for the Pentium processor (610\75) TCP, as described in section 1.1.

This document should be used in conjunction with the Pentium processor documents listed below.

List of related documents:

- *Pentium™ Family User's Manual, Vol. 1: Data Book* (Order Number: 241428)
- *Pentium™ Family User's Manual, Vol. 3: Architecture and Programming Manual* (Order Number: 241430)

1.1 Pentium™ Processor (610\75) SPGA Specifications and Dif ferences from the TCP Package

This section provides references to the Pentium processor (610\75) SPGA specifications and describes the major differences between the Pentium processor (610\75) SPGA and TCP packages.

All Pentium processor (610\75) SPGA specifications, with the exception of power consumption, are identical to the Pentium processor (735\90, 815\100) specifications provided in the *Pentium™ Family User's Manual, Volume 1: Data Book*. See Tables 8 and 11 in sectio n 4.2 for the Pentium processor (610\75) SPGA and TCP power specifications.

The following features have been eliminated for the Pentium processor (610\75) TCP: the Upgrade feature, the Dual Processing (DP) feature, and the Master/Checker functional redundancy feature. Table 1 lists the corresponding pins which exist on the Pentium processor (610\75) SPGA but have been removed on the Pentium processor (610\75) TCP.

Table 1. SPGA Signals Removed in TCP

Signal	Function
ADSC#	Additional Address Status. This signal is mainly used for large or standalone L2 cache memory subsystem support required for high-performance desktop or server models.
BRDYC#	Additional Burst Ready. This signal is mainly used for large or standalone L2 cache memory subsystem support required for high-performance desktop or server models.
CPUTYP	CPU Type. This signal is used for dual processing systems.
D/P#	Dual/Primary processor identification. This signal is only used for an Upgrade processor.
FRCMC#	Functional Redundancy Checking. This signal is only used for error detection via processor redundancy, and requires two Pentium processors (master/checker).
PBGNT#	Private Bus Grant. This signal is only used for dual processing systems.
PBREQ# PHIT#	Private Bus Request. This signal is used only for dual processing systems. Private Hit. This signal is only used for dual processing systems.
PHITM#	Private Modified Hit. This signal is only used for dual processing systems.

The I/O buffer models provided in section 4.4 of this document apply to both the Pentium processor (610\75) TCP and SPGA packages, although the capacitance (C_p) and inductance (L_p) parameter values differ between the two packages. Also, the thermal parameters, T_{CASE} max and θ_{CA}, differ between the TCP and SPGA packages. For Pentium processor (610\75) SPGA values, refer to Chapters 24 and 26 of the *Pentium™ Family User's Manual, Volume 1: Data Book.*

2.0 MICROPROCESSOR ARCHITECTURE OVERVIEW

The Pentium™ processor at iCOMP™ rating 610\75 MHz extends the Intel Pentium family of microprocessors. It is compatible with the 8086/88, 80286, Intel386™ DX CPU, Intel386 SX CPU, Intel486™ DX CPU, Intel486 SX CPU, Intel486 DX2 CPUs, the Pentium processor at iCOMP Index 510\60 MHz and iCOMP Index 567\66 MHz, and the Pentium processor at iCOMP Index 735\90 MHz and iCOMP Index 815\100 MHz.

The Pentium processor family consists of the new Pentium processor at iCOMP rating 610\75 MHz, described in this document, the original Pentium processor (510\60, 567\66), and the Pentium processor (735\90, 815\100). The name "Pentium processor (610\75)" will be used in this document to refer to the Pentium processor at iCOMP rating 610\75 MHz. "Pentium Processor" will be used in this document to refer to the entire Pentium processor family in general.

The Pentium processor family architecture contains all of the features of the Intel486 CPU family, and provides significant enhancements and additions including the following:

- Superscalar Architecture
- Dynamic Branch Prediction
- Pipelined Floating-Point Unit
- Improved Instruction Execution Time
- Separate 8K Code and 8K Data Caches
- Writeback MESI Protocol in the Data Cache
- 64-Bit Data Bus
- Bus Cycle Pipelining
- Address Parity
- Internal Parity Checking
- Execution Tracing
- Performance Monitoring
- IEEE 1149.1 Boundary Scan
- System Management Mode
- Virtual Mode Extensions

2.1 Pentium™ Processor Family Architecture

The application instruction set of the Pentium processor family includes the complete Intel486 CPU family instruction set with extensions to accommodate some of the additional functionality of the Pentium processors. All application software written for the Intel386 and Intel486 family microprocessors will run on the Pentium processors without modification. The on-chip memory management unit (MMU) is completely compatible with the Intel386 family and Intel486 family of CPUs.

The Pentium processors implement several enhancements to increase performance. The two instruction pipelines and floating-point unit on Pentium processors are capable of independent operation. Each pipeline issues frequently used instructions in a single clock. Together, the dual pipes can issue two integer instructions in one clock, or one floating point instruction (under certain circumstances, two floating-point instructions) in one clock.

Branch prediction is implemented in the Pentium processors. To s upport this, Pentium processors implement two prefetch buffers, one to prefetch code in a linear fashion, and one that prefetches code according to the BTB so the needed code is almost always prefetched before it is needed for execution.

The floating-point unit has been completely redesigned over the Intel486 CPU. Faster algorithms provide up to 10X speed-up for common operations including add, multiply, and load.

Pentium processors include separate code and data caches integrated on-chip to meet performance goals. Each cache is 8 Kbytes in size, with a 32-byte line size and is 2-way set associative. Each cache has a dedicated Translation Lookaside Buffer (TLB) to translate linear addresses to physical addresses. The data cache is configurable to be writeback or writethrough on a line-by-line basis and follows the MESI protocol. The data cache tags are triple ported to support two data transfers and an inquire cycle in the same clock. The code cache is an inherently write-protected cache. The code cache tags are also triple ported to support snooping and split line accesses. Individual pages can be configured as cacheable or non-cacheable by software or hardware. The caches can be enabled or disabled by software or hardware.

The Pentium processors have increased the data bus to 64 bits to improve the data transfer rate. Burst read and burst writeback cycles are supported by the Pentium processors. In addition, bus cycle pipelining has been added to allow two bus cycles to be in progress simultaneously. The Pentium processors' Memory Management Unit contains optional extensions to the architecture which allow 2-Mbyte and 4-Mbyte page sizes.

The Pentium processors have added significant data integrity and error detection capability. Data parity checking is still supported on a byte-by-byte basis. Address parity checking, and internal parity checking features have been added along with a new exception, the machine check exception.

As more and more functions are integrated on chip, the complexity of board level testing is increased. To address this, the Pentium processors have increased test and debug capability. The Pentium processors implement IEEE Boundary Scan (Standard 1149.1). In addition, the Pentium processors have specified 4 breakpoint pins that correspond to each of the debug registers and externally indicate a breakpoint match. Execution tracing provides external indications when an instruction has completed execution in either of the two internal pipelines, or when a branch has been taken.

System Management Mode (SMM) has been implemented along with some extensions to the SMM architecture. Enhancements to the virtual 8086 mode have been made to increase performance by reducing the number of times it is necessary to trap to a virtual 8086 monitor.

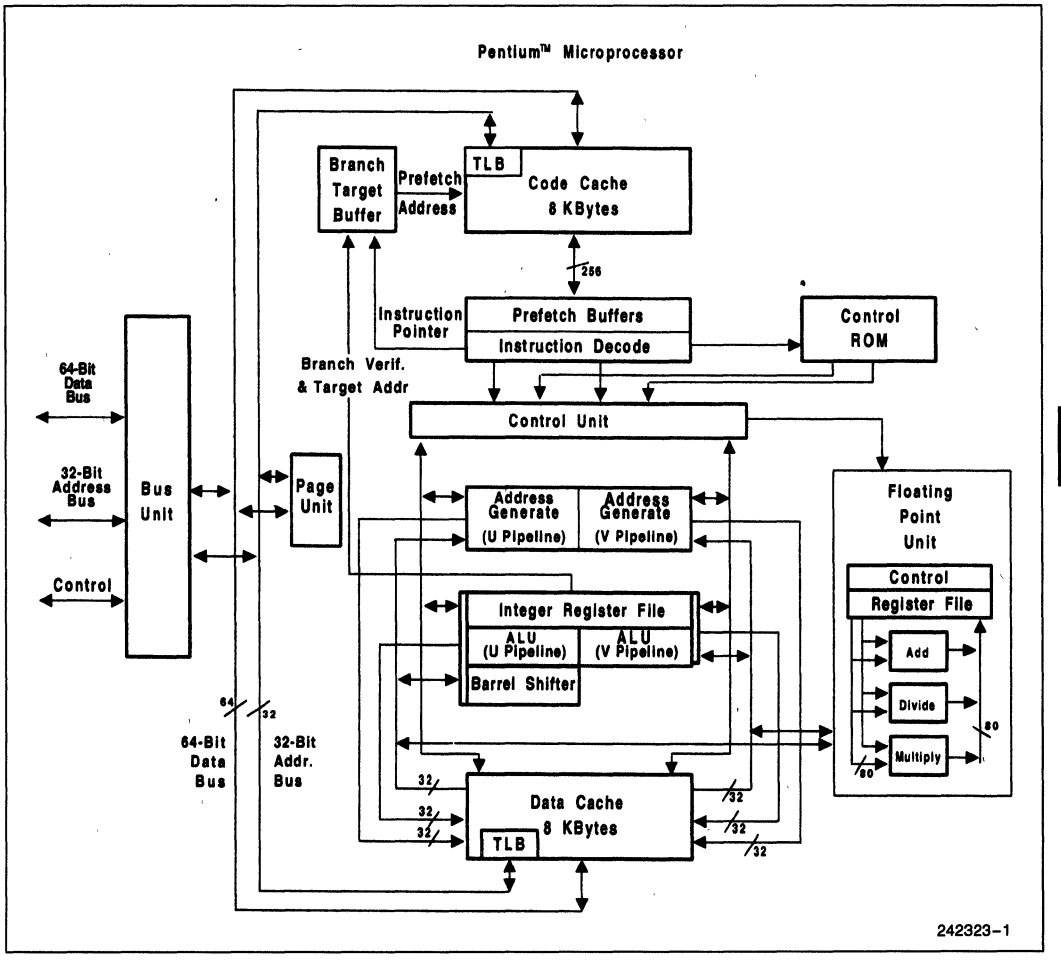


Figure 1. Pentium™ Processor Block Diagram

The block diagram shows the two instruction pipelines, the "u" pipe and the "v" pipe. The u-pipe can execute all integer and floating point instructions. The v-pipe can execute simple integer instructions and the FXCH floating-point instructions.

The separate caches are shown, the code cache and data cache. The data cache has two ports, one for each of the two pipes (the tags are triple ported to allow simultaneous inquire cycles). The data cache has a dedicated Translation Lookaside Buffer (TLB) to translate linear addresses to the physical addresses used by the data cache.

The code cache, branch target buffer and prefetch buffers are responsible for getting raw instructions into the execution units of the Pentium processor. Instructions are fetched from the code cache or from the external bus. Branch addresses are remembered by the branch target buffer. The code cache TLB translates linear addresses to physical addresses used by the code cache.

The decode unit decodes the prefetched instructions so the Pentium processor can execute the instruction. The control ROM contains the microcode which controls the sequence of operations that must be performed to implement the Pentium processor architecture. The control ROM unit has direct control over both pipelines.

The Pentium processors contain a pipelined floating-point unit that provides a significant floating-point performance advantage over previous generations of processors.

The architectural features introduced in this section are more fully described in the *Pentium™ Family User's Manual*.

3.0 TCP PINOUT

3.1 TCP Pinout and Pin Descriptions

3.1.1 Pentium™ Processor (610\75) TCP PINOUT

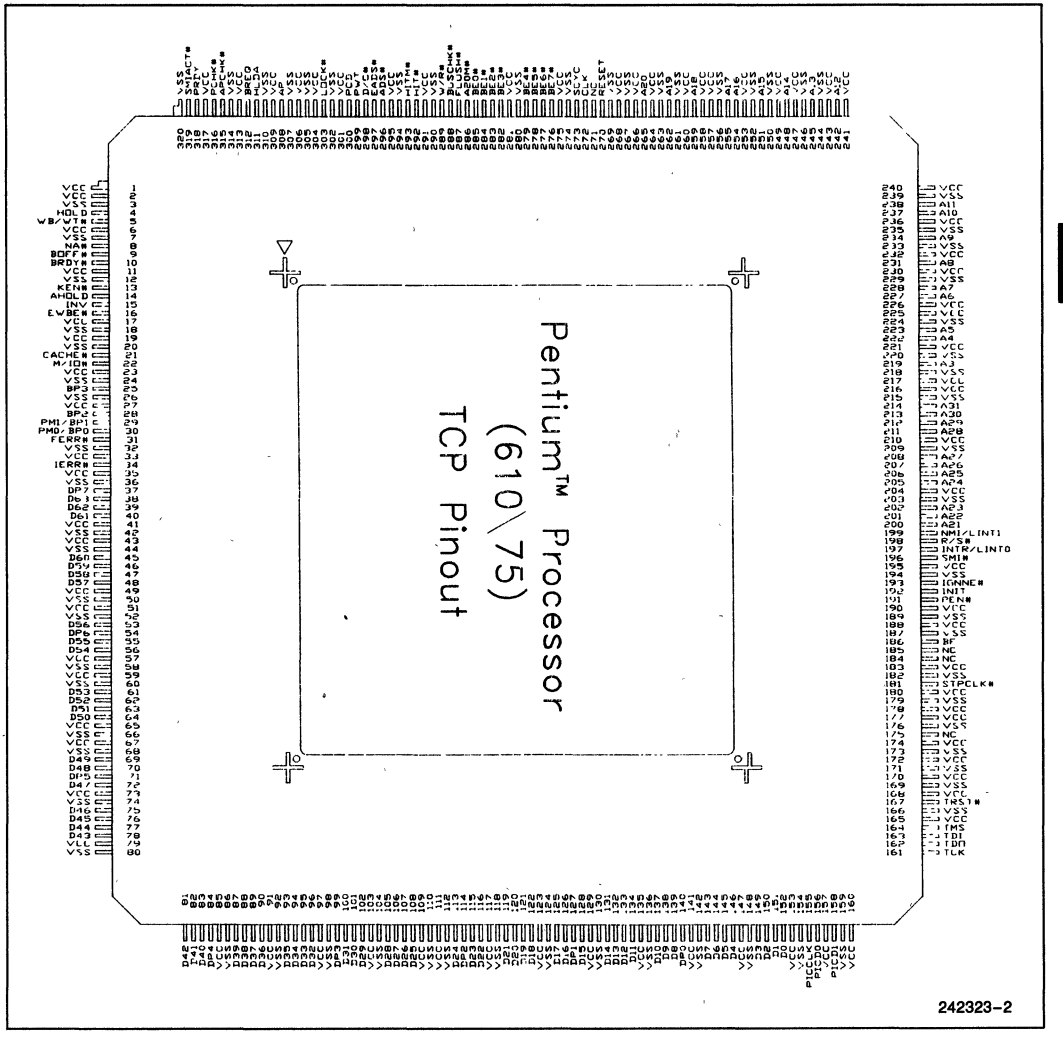


Figure 2. Pentium™ Processor (610\75) TCP Pinout

3.1.2 PIN CROSS REFERENCE TABLE FOR Pentium™ Processor (610\75) TCP

Table 2. TCP Pin Cross Reference by Pin Name

Address									
A3	219	A9	234	A15	251	A21	200	A27	208
A4	222	A10	237	A16	254	A22	201	A28	211
A5	223	A11	238	A17	255	A23	202	A29	212
A6	227	A12	242	A18	259	A24	205	A30	213
A7	228	A13	245	A19	262	A25	206	A31	214
A8	231	A14	248	A20	265	A26	207		
Data									
D0	152	D13	132	D26	107	D39	87	D52	62
D1	151	D14	131	D27	106	D40	83	D53	61
D2	150	D15	128	D28	105	D41	82	D54	56
D3	149	D16	126	D29	102	D42	81	D55	55
D4	146	D17	125	D30	101	D43	78	D56	53
D5	145	D18	122	D31	100	D44	77	D57	48
D6	144	D19	121	D32	96	D45	76	D58	47
D7	143	D20	120	D33	95	D46	75	D59	46
D8	139	D21	119	D34	94	D47	72	D60	45
D9	138	D22	116	D35	93	D48	70	D61	40
D10	137	D23	115	D36	90	D49	69	D62	39
D11	134	D24	113	D37	89	D50	64	D63	38
D12	133	D25	108	D38	88	D51	63		

Table 2. TCP Pin Cross Reference by Pin Name (Continued)

Control							
A20M#	286	BREQ	312	HITM#	293	PM1/BP1	29
ADS#	296	BUSCHK#	288	HLDA	311	PRDY	318
AHOLD	14	CACHE#	21	HOLD	4	PWT	299
AP	308	D/C#	298	IERR#	34	R/S#	198
APCHK#	315	DP0	140	IGNNE#	193	RESET	270
BE0#	285	DP1	127	INIT	192	SCYC	273
BE1#	284	DP2	114	INTR/LINT0	197	SMI#	196
BE2#	283	DP3	99	INV	15	SMIACT#	319
BE3#	282	DP4	84	KEN#	13	TCK	161
BE4#	279	DP5	71	LOCK#	303	TDI	163
BE5#	278	DP6	54	M/IO#	22	TDO	162
BE6#	277	DP7	37	NA#	8	TMS	164
BE7#	276	EADS#	297	NMI/LINT1	199	TRST#	167
BOFF#	9	EWBE#	16	PCD	300	W/R#	289
BP2	28	FERR#	31	PCHK#	316	WB/WT#	5
BP3	25	FLUSH#	287	PEN#	191		
BRDY#	10	HIT#	292	PM0/BP0	30		
APIC				**Clock Control**			
PICCLK	155	PICD1	158	BF	186	STPCLK#	181
PICD0	156	[APICEN]		CLK	272		
[DPEN#]							

Table 2. TCP Pin Cross Reference by Pin Name (Continued)

V_{CC}								
1*	35	73	123	168*	190*	230	257*	295
2	41*	79	129	170*	195*	232*	258	301
6*	43	85	135	172*	204	236	260*	304*
11*	49*	91	141	174*	210	240*	264	306
17*	51	97	147	177*	216	241	266*	309*
19	57*	103	153*	178	217*	243*	268*	313
23	59	109	157*	18 0*	221	247	275	317*
27*	65*	111*	160	183*	225*	249*	281	
33*	67	117	165*	188*	226	253	291	

V_{SS}						
3	50	104	166	209	250	302
7	52	110	169	215	252	305
12	58	112	171	218	256	307
18	60	118	173	220	261	310
20	66	124	176	224	263	314
24	68	130	179	229	267	320
26	74	136	182	233	269	
32	80	142	187	235	274	
36	86	148	189	239	280	
42	92	154	194	244	290	
44	98	159	203	246	294	

NC			
175	184	185	271

NOTE:
*These V_{CC} pins are 3.3V supplies for the Pentium processor (610\75) TCP but will be lower voltage pins on future offerings of this microprocessor family. All other V_{CC} pins will remain at 3.3V.

3.2 Design Notes

For reliable operation, always connect unused inputs to an appropriate signal level. Unused active low inputs should be connected to V_{CC}. Unused active HIGH inputs should be connected to GND (V_{SS}).

No Connect (NC) pins must remain unconnected. Connection of NC pins may result in component failure or incompatibility with processor steppings.

3.3 Quick Pin Reference

This section gives a brief functional description of each of the pins. For a detailed description, see the "Hardware Interface" chapter in the *Pentium™ Family User's Manual, Volume 1*.

Note that all input pins must meet their AC/DC specifications to guarantee proper functional behavior.

The # symbol at the end of a signal name indicates that the active, or asserted state occurs when the signal is at a low voltage. When a # symbol is not present after the signal name, the signal is active, or asserted at the high voltage level.

Table 3. Quick Pin Reference

Symbol	Type	Name and Function
A20M#	I	When the **address bit 20 mask** pin is asserted, the Pentium processor (610\75) emulates the address wraparound at 1 Mbyte which occurs on the 8086. When A20M# is asserted, the Pentium processor (610\75) masks physical address bit 20 (A20) before performing a lookup to the internal caches or driving a memory cycle on the bus. The effect of A20M# is undefined in protected mode. A20M# must be asserted only when the processor is in real mode.
A31–A3	I/O	As outputs, the **address** lines of the processor along with the byte enables define the physical area of memory or I/O accessed. The external system drives the inquire address to the processor on A31-A5.
ADS#	O	The **address status** indicates that a new valid bus cycle is currently being driven by the Pentium processor (610\75).
AHOLD	I	In response to the assertion of **address hold,** the Pentium processor (610\75) will stop driving the address lines (A31-A3), and AP in the next clock. The rest of the bus will remain active so data can be returned or driven for previously issued bus cycles.
AP	I/O	**Address parity** is driven by the Pentium processor (610\75) with even parity information on all Pentium processor (610\75) generated cycles in the same clock that the address is driven. Even parity must be driven back to the Pentium processor (610\75) during inquire cycles on this pin in the same clock as EADS# to ensure that correct parity check status is indicated by the Pentium processor (610\75).
APCHK#	O	The **address parity check** status pin is asserted two clocks after EADS# is sampled active if the Pentium processor (610\75) has detected a parity error on the address bus during inquire cycles. APCHK# will remain active for one clock each time a parity error is detected.
[APICEN] PICD1	I	The **Advanced Programmable Interrupt Controller Enable** pin enables or disables the on-chip APIC interrupt controller. If sampled high at the falling edge of RESET, the APIC is enabled. APICEN shares a pin with the **Programmable Interrupt Controller Data 1** signal.
BE7#–BE5# BE4#-BE0#	O I/O	The **byte enable** pins are used to determine which bytes must be written to external memory, or which bytes were requested by the CPU for the current cycle. The byte enables are driven in the same clock as the address lines (A31–3). The lower four byte enable pins (BE3#–BE0#) are used on the Pentium processor (610\75) also as APIC ID inputs and are sampled at RESET for that function.

Table 3. Quick Pin Reference (Continued)

Symbol	Type	Name and Function
[BF]	I	**Bus Frequency** determines the bus-to-core frequency ratio. BF is sampled at RESET, and cannot be changed until another non-warm (1 ms) assertion of RESET. Additionally, BF must not change values while RESET is active. For proper operation of the Pentium processor (610\75) this pin should be strapped high or low. When BF is strapped to V_{CC}, the processor will operate at a 2 to 3 bus to core frequency ratio. When BF is strapped to V_{SS}, the processor will operate at a 1 to 2 bus to core frequency ratio. If BF is left floating, the Pentium processor (610\75) defaults to a 2 to 3 bus ratio. Note the Pentium processor (610\75) will not operate at a 1 to 2 bus to core frequency ratio.
BOFF#	I	The **backoff** input is used to abort all outstanding bus cycles that have not yet completed. In response to BOFF#, the Pentium processor (610\75) will float all pins normally floated during bus hold in the next clock. The processor remains in bus hold until BOFF# is negated, at which time the Pentium processor (610\75) restarts the aborted bus cycle(s) in their entirety.
BP[3:2] PM/BP[1:0]	O	The **breakpoint** pins (BP30–0) correspond to the debug registers, DR3–DR0. These pins externally indicate a breakpoint match when the debug registers are programmed to test for breakpoint matches. BP1 and BP0 are multiplexed with the **performance monitoring** pins (PM1 and PM0). The PB1 and PB0 bits in the Debug Mode Control Register determine if the pins are configured as breakpoint or performance monitoring pins. The pins come out of RESET configured for performance monitoring.
BRDY#	I	The **burst ready** input indicates that the external system has presented valid data on the data pins in response to a read or that the external system has accepted the Pentium processor (610\75) data in response to a write request. This signal is sampled in the T2, T12 and T2P bus states.
BREQ	O	The **bus request** output indicates to the external system that the Pentium processor (610\75) has internally generated a bus request. This signal is always driven whether or not the Pentium processor (610\75) is driving its bus.
BUSCHK#	I	The **bus check** input allows the system to signal an unsuccessful completion of a bus cycle. If this pin is sampled active, the Pentium processor (610\75) will latch the address and control signals in the machine check registers. If, in addition, the MCE bit in CR4 is set, the Pentium processor (610\75) will vector to the machine check exception.
CACHE#	O	For Pentium processor (610\75)-initiated cycles the **cache** pin indicates internal cacheability of the cycle (if a read), and indicates a burst writeback cycle (if a write). If this pin is driven inactive during a read cycle, the Pentium processor (610\75) will not cache the returned data, regardless of the state of the KEN# pin. This pin is also used to determine the cycle length (number of transfers in the cycle).

Table 3. Quick Pin Reference (Continued)

Symbol	Type	Name and Function
CLK	I	The **clock** input provides the fundamental timing for the Pentium processor (610\75).Its frequency is the operating frequency of the Pentium processor (610\75) external bus and requires TTL levels. All external timing parameters except TDI, TDO, TMS, TRST#, and PICD0–1 are specified with respect to the rising edge of CLK.
D/C#	O	The **data/code** output is one of the primary bus cycle definition pins. It is driven valid in the same clock as the ADS# signal is asserted. D/C# distinguishes between data and code or special cycles.
D63–D0	I/O	These are the 64 **data lines** for the processor. Lines D7–D0 define the least significant byte of the data bus; lines D63–D56 define the most significant byte of the data bus. When the CPU is driving the data lines, they are driven during the T2, T12, or T2P clocks for that cycle. During reads, the CPU samples the data bus when BRDY# is returned.
DP7–DP0	I/O	These are the **data parity** pins for the processor. There is one for each byte of the data bus. They are driven by the Pentium processor (610\75) with even parity information on writes in the same clock as write data. Even parity information must be driven back to the Pentium processor (610\75) on these pins in the same clock as the data to ensure that the correct parity check status is indicated by the Pentium processor (610\75). DP7 applies to D63–D56; DP0 applies to D7–D0.
[DPEN#] PICD0	I/O	**Dual processing enable** is an output of the Dual processor and an input of the Primary processor. The Dual processor drives DPEN# low to the Primary processor at RESET to indicate that the Primary processor should enable dual processor mode. Since the dual processing feature is not supported on the Pentium processor (610\75) TCP package, DPEN# should never be asserted (low) at RESET. DPEN# shares a pin with PICD0.
EADS#	I	This signal indicates that a valid **external address** has been driven onto the Pentium processor (610\75) address pins to be used for an inquire cycle.
EWBE#	I	The **external write buffer empty** input, when inactive (high), indicates that a write cycle is pending in the external system. When the Pentium processor (610\75) generates a write, and EWBE# is sampled inactive, the Pentium processor (610\75) will hold off all subsequent writes to all E- or M-state lines in the data cache until all write cycles have completed, as indicated by EWBE# being active.
FERR#	O	The **floating point error** pin is driven active when an unmasked floating point error occurs. FERR# is similar to the ERROR# pin on the Intel387™ math coprocessor. FERR# is included for compatibility with systems using DOS-type floating point error reporting.

Table 3. Quick Pin Reference (Continued)

Symbol	Type	Name and Function
FLUSH#	I	When asserted, the **cache flush** input forces the Pentium processor (610\75) to write back all modified lines in the data cache and invalidate its internal caches. A Flush Acknowledge special cycle will be generated by the Pentium processor (610\75) indicating completion of the writeback and invalidation. If FLUSH# is sampled low when RESET transitions from high to low, tristate test mode is entered.
HIT#	O	The **hit** indication is driven to reflect the outcome of an inquire cycle. If an inquire cycle hits a valid line in either the Pentium processor (610\75) data or instruction cache, this pin is asserted two clocks after EADS# is sampled asserted. If the inquire cycle misses the Pentium processor (610\75) cache, this pin is negated two clocks after EADS#. This pin changes its value only as a result of an inquire cycle and retains its value between the cycles.
HITM#	O	The **hit to a modified line** output is driven to reflect the outcome of an inquire cycle. It is asserted after inquire cycles which resulted in a hit to a modified line in the data cache. It is used to inhibit another bus master from accessing the data until the line is completely written back.
HLDA	O	The **bus hold acknowledge** pin goes active in response to a hold request driven to the processor on the HOLD pin. It indicates that the Pentium processor (610\75) has floated most of the output pins and relinquished the bus to another local bus master. When leaving bus hold, HLDA will be driven inactive and the Pentium processor (610\75) will resume driving the bus. If the Pentium processor (610\75) has a bus cycle pending, it will be driven in the same clock that HLDA is de-asserted.
HOLD	I	In response to the **bus hold request,** the Pentium processor (610\75) will float most of its output and input/output pins and assert HLDA after completing all outstanding bus cycles. The Pentium processor (610\75) will maintain its bus in this state until HOLD is de-asserted. HOLD is not recognized during LOCK cycles. The Pentium processor (610\75) will recognize HOLD during reset.
IERR#	O	The **internal error** pin is used to indicate internal parity errors. If a parity error occurs on a read from an internal array, the Pentium processor (610\75) will assert the IERR# pin for one clock and then shutdown.
IGNNE#	I	This is the **ignore numeric error** input. This pin has no effect when the NE bit in CR0 is set to 1. When the CR0.NE bit is 0, and the IGNNE# pin is asserted, the Pentium processor (610\75) will ignore any pending unmasked numeric exception and continue executing floating-point instructions for the entire duration that this pin is asserted. When the CR0.NE bit is 0, IGNNE# is not asserted, a pending unmasked nume ric exception exists (SW.ES = 1), and the floating-point instruction is one of FINIT, FCLEX, FSTENV, FSAVE, FSTSW, FSTCW, FENI, FDISI, or FSETPM, the Pentium processor (610\75) will execute the instruction in spite of the pending exception. When the CR0.NE bit is 0, IGNNE# is not asserted, a pending unmasked numeric exception exists (SW.ES = 1), and the floating-point instruction is one other than FINIT, FCLEX, FSTENV, FSAVE, FSTSW, FSTCW, FENI, FDISI, or FSETPM, the Pentium processor (610\75) will stop execution and wait for an external interrupt.

Table 3. Quick Pin Reference (Continued)

Symbol	Type	Name and Function
INIT	I	The Pentium processor (610\75) **initialization** input pin forces the Pentium processor (610\75) to begin execution in a known state. The processor state after INIT is the same as the state after RESET except that the internal caches, write buffers, and floating point registers retain the values they had prior to INIT. INIT may NOT be used in lieu of RESET after power up. If INIT is sampled high when RESET transitions from high to low, the Pentium processor (610\75) will perform built-in self test prior to the start of program execution.
INTR/LINT0	I	An active **maskable interrupt** input indicates that an external interrupt has been generated. If the IF bit in the EFLAGS register is set, the Pentium processor (610\75) will generate two locked interrupt acknowledge bus cycles and vector to an interrupt handler after the current instruction execution is completed. INTR must remain active until the first interrupt acknowledge cycle is generated to assure that the interrupt is recognized. If the local APIC is enabled, this pin becomes local interrupt 0.
INV	I	The **invalidation** input determines the final cache line state (S or I) in case of an inquire cycle hit. It is sampled together with the address for the inquire cycle in the clock EADS# is sampled active.
KEN#	I	The **cache enable** pin is used to determine whether the current cycle is cacheable or not and is consequently used to determine cycle length. When the Pentium processor (610\75) generates a cycle that can be cached (CACHE# asserted) and KEN# is active, the cycle will be transformed into a burst line fill cycle.
LINT0/INTR	I	If the APIC is enabled, this pin is **local interrupt 0.** If the APIC is disabled, this pin is **interrupt.**
LINT1/NMI	I	If the APIC is enabled, this pin is **local interrupt 1.** If the APIC is disabled, this pin is **non-maskable interrupt.**
LOCK#	O	The **bus lock** pin indicates that the current bus cycle is locked. The Pentium processor (610\75) will not allow a bus hold when LOCK# is asserted (but AHOLD and BOFF# are allowed). LOCK# goes active in the first clock of the first locked bus cycle and goes inactive after the BRDY# is returned for the last locked bus cycle. LOCK# is guaranteed to be de-asserted for at least one clock between back-to-back locked cycles.
M/IO#	O	The **memory/input-output** is one of the primary bus cycle definition pins. It is driven valid in the same clock as the ADS# signal is asserted. M/IO# distinguishes between memory and I/O cycles.

Table 3. Quick Pin Reference (Continued)

Symbol	Type	Name and Function
NA#	I	An active **next address** input indicates that the external memory system is ready to accept a new bus cycle although all data transfers for the current cycle have not yet completed. The Pentium processor (610\75) will issue ADS# for a pending cycle two clocks after NA# is asserted. The Pentium processor (610\75) supports up to 2 outstanding bus cycles.
NMI/LINT1	I	The **non-maskable interrupt** request signal indicates that an external non-maskable interrupt has been generated. If the local APIC is enabled, this pin becomes local interrupt 1.
PCD	O	The **page cache disable** pin reflects the state of the PCD bit in CR3, the Page Directory Entry, or the Page Table Entry. The purpose of PCD is to provide an external cacheability indication on a page-by-page basis.
PCHK#	O	The **parity check** output indicates the result of a parity check on a data read. It is driven with parity status two clocks after BRDY# is returned. PCHK# remains low one clock for each clock in which a parity error was detected. Parity is checked only for the bytes on which valid data is returned.
PEN#	I	The **parity enable** input (along with CR4.MCE) determines whether a machine check exception will be taken as a result of a data parity error on a read cycle. If this pin is sampled active in the clock a data parity error is detected, the Pentium processor (610\75) will latch the address and control signals of the cycle with the parity error in the machine check registers. If, in addition, the machine check enable bit in CR4 is set to "1", the Pentium processor (610\75) will vector to the machine check exception before the beginning of the next instruction.
PICCLK	I	The APIC interrupt controller serial data bus clock is driven into the **programmable interrupt controller clock** input of the Pentium processor (610\75).
PICD0-1 [DPEN#] [APICEN]	I/O	**Programmable interrupt controller data lines 0–1** of the Pentium processor (610\75) comprise the data portion of the APIC 3-wire bus. They are open-drain outputs that require external pull-up resistors. These signals share pins with DPEN# and APICEN.
PM/BP[1:0]	O	These pins function as part of the performance monitoring feature. The breakpoint 1-0 pins are multiplexed with the **performance monitoring 1–0 pins.** The PB1 and PB0 bits in the Debug Mode Control Register determine if the pins are configured as breakpoint or performance monitoring pins. The pins come out of RESET configured for performance monitoring.
PRDY	O	The **probe ready** output pin indicates that the processor has stopped normal execution in response to the R/S# pin going active, or Probe Mode being entered.
PWT	O	The **page writethrough** pin reflects the state of the PWT bit in CR3, the page directory entry, or the page table entry. The PWT pin is used to provide an external writeback indication on a page-by-page basis.

Table 3. Quick Pin Reference (Continued)

Symbol	Type*	Name and Function
R/S#	I	The **run/stop** input is an asynchronous, edge-sensitive interrupt used to stop the normal execution of the processor and place it into an idle state. A high to low transition on the R/S# pin will interrupt the processor and cause it to stop execution at the next instruction boundary.
RESET	I	**RESET** forces the Pentium processor (610\75) to begin execution at a known state. All the Pentium processor (610\75) internal caches will be invalidated upon the RESET. Modified lines in the data cache are not written back. FLUSH# and INIT are sampled when RESET transitions from high to low to determine if tristate test mode will be entered, or if BIST will be run.
SCYC	O	The **split cycle** output is asserted during misaligned LOCKed transfers to indicate that more than two cycles will be locked together. This signal is defined for locked cycles only. It is undefined for cycles which are not locked.
SMI#	I	The **system management interrupt** causes a system management interrupt request to be latched internally. When the latched SMI# is recognized on an instruction boundary, the processor enters System Management Mode.
SMIACT#	O	An active **system management interrupt active** output indicates that the processor is operating in System Management Mode.
STPCLK#	I	Assertion of the **stop clock** input signifies a request to stop the internal clock of the Pentium processor (610\75) thereby causing the core to consume less power. When the CPU recognizes STPCLK#, the processor will stop execution on the next instruction boundary, unless superseded by a higher priority interrupt, and generate a Stop Grant Acknowledge cycle. When STPCLK# is asserted, the Pentium processor (610\75) will still respond to external snoop requests.
TCK	I	The **testability clock** input provides the clocking function for the Pentium processor (610\75) boundary scan in accordance with the IEEE Boundary Scan interface (Standard 1149.1). It is used to clock state information and data into and out of the Pentium processor (610\75) during boundary scan.
TDI	I	The **test data input** is a serial input for the test logic. TAP instructions and data are shifted into the Pentium processor (610\75) on the TDI pin on the rising edge of TCK when the TAP controller is in an appropriate state.
TDO	O	The **test data output** is a serial output of the test logic. TAP instructions and data are shifted out of the Pentium processor (610\75) on the TDO pin on TCK's falling edge when the TAP controller is in an appropriate state.
TMS	I	The value of the **test mode select** input signal sampled at the rising edge of TCK controls the sequence of TAP controller state changes.
TRST#	I	When asserted, the **test reset** input allows the TAP controller to be asynchronously initialized.

Table 3. Quick Pin Reference (Continued)

Symbol	Type	Name and Function
V_{CC} V_{SS}	I I	The Pentium processor (610\75) has 79 3.3V **power** inputs. The Pentium processor (610\75) has 72 **ground** inputs.
W/R#	O	**Write/read** is one of the primary bus cycle definition pins. It is driven valid in the same clock as the ADS# signal is asserted. W/R# distinguishes between write and read cycles.
WB/WT#	I	The **writeback/writethrough** input allows a data cache line to be defined as writeback or writethrough on a line-by-line basis. As a result, it determines whether a cache line is initially in the S or E state in the data cache.

3.4 Pin Reference Tables

Table 4. Output Pins

Name	Active Level	When Floated
ADS#	Low	Bus Hold, BOFF#
APCHK#	Low	
BE7#–BE5#	Low	Bus Hold, BOFF#
BREQ	High	
CACHE#	Low	Bus Hold, BOFF#
FERR#	Low	
HIT#	Low	
HITM#	Low	
HLDA	High	
IERR#	Low	
LOCK#	Low	Bus Hold, BOFF#
M/IO#, D/C#, W/R#	n/a	Bus Hold, BOFF#
PCHK#	Low	
BP3-2, PM1/BP1, PM0/BP0	High	
PRDY	High	
PWT, PCD	High	Bus Hold, BOFF#
SCYC	High	Bus Hold, BOFF#
SMIACT#	Low	
TDO	n/a	All states except Shift-DR and Shift-IR

NOTE:
All output and input/output pins are floated during tristate test mode (except TDO).

Table 5. Input Pins

Name	Active Level	Synchronous/ Asynchronous	Internal resistor	Qualified
A20M#	Low	Asynchronous		
AHOLD	High	Synchronous		
BF	High	Synchronous/RESET	Pullup	
BOFF#	Low	Synchronous		
BRDY#	Low	Synchronous		Bus State T2, T12, T2P
BUSCHK#	Low	Synchronous	Pullup	BRDY#
CLK	n/a			
EADS#	Low	Synchronous		
EWBE#	Low	Synchronous		BRDY#
FLUSH#	Low	Asynchronous		
HOLD	High	Synchronous		
IGNNE#	Low	Asynchronous		
INIT	High	Asynchronous		
INTR	High	Asynchronous		
INV	High	Synchronous		EADS#
KEN#	Low	Synchronous		First BRDY#/NA#
NA#	Low	Synchronous		Bus State T2,TD,T2P
NMI	High	Asynchronous		
PEN#	Low	Synchronous		BRDY#
PICCLK	High	Asynchronous	Pullup	
R/S#	n/a	Asynchronous	Pullup	
RESET	High	Asynchronous		
SMI#	Low	Asynchronous	Pullup	
STPCLK#	Low	Asynchronous	Pullup	
TCK	n/a		Pullup	
TDI	n/a	Synchronous/TCK	Pullup	TCK
TMS	n/a	Synchronous/TCK	Pullup	TCK
TRST#	Low	Asynchronous	Pullup	
WB/WT#	n/a	Synchronous		First BRDY#/NA#

Table 6. Input/Output Pins

Name	Active Level	When Floated	Qualified (when an Input)	Internal Resistor
A31-A3	n/a	Address Hold, Bus Hold, BOFF#	EADS#	
AP	n/a	Address Hold, Bus Hold, BOFF#	EADS#	
BE4#-BE0#	Low	Bus Hold, BOFF#	RESET	Pulldown*
D63-D0	n/a	Bus Hold, BOFF#	BRDY#	
DP7-DP0	n/a	Bus Hold, BOFF#	BRDY#	
PICD0[DPEN#]				Pullup
PICD1[APICEN]				Pulldown

NOTES:
All output and input/output pins are floated during tristate test mode (except TDO).
*BE3#–BE0# have pulldowns during RESET only.

3.5 Pin Grouping According to Function

Table 7 organizes the pins with respect to their function.

Table 7. Pin Functional Grouping

Function	Pins
Clock	CLK
Initialization	RESET, INIT
Address Bus	A31-A3, BE7#-BE0#
Address Mask	A20M#
Data Bus	D63-D0
Address Parity	AP, APCHK#
APIC Support	PICCLK, PICD0-1
Data Parity	DP7-DP0, PCHK#, PEN#
Internal Parity Error	IERR#
System Error	BUSCHK#
Bus Cycle Definition	M/IO#, D/C#, W/R#, CACHE#, SCYC, LOCK#
Bus Control	ADS#, BRDY#, NA#
Page Cacheability	PCD, PWT
Cache Control	KEN#, WB/WT#
Cache Snooping/Consistency	AHOLD, EADS#, HIT#, HITM#, INV
Cache Flush	FLUSH#
Write Ordering	EWBE#
Bus Arbitration	BOFF#, BREQ, HOLD, HLDA
Interrupts	INTR, NMI
Floating Point Error Reporting	FERR#, IGNNE#
System Management Mode	SMI#, SMIACT#
TAP Port	TCK, TMS, TDI, TDO, TRST#
Breakpoint/Performance Monitoring	PM0/BP0, PM1/BP1, BP3-2
Clock Control	STPCLK#
Probe Mode	R/S#, PRDY

4.0 Pentium™ Processor (610\75) TCP ELECTRICAL SPECIFICATIONS

4.1 Absolute Maximum Ratings

The following values are stress ratings only. Functional operation at the maximum ratings is not implied or guaranteed. Functional operating conditions are given in the AC and DC specification tables.

Extended exposure to the maximum ratings may affect device reliability. Furthermore, although the Pentium processor (610\75) contains protective circuitry to resist damage from static electric discharge, always take precautions to avoid high static voltages or electric fields.

Case temperature under bias ..... −65°C to 110°C

Storage temperature ..... −65°C to +150°C

3V Supply voltage with respect to V_{SS} ..... −0.5V to +4.6V

3V Only Buffer DC Input Voltage..... −0.5V to V_{CC} +0.5; not to exceed 4.6V(2)

5V Safe Buffer DC Input Voltage..... −0.5V to 6.5V(1,3)

NOTES:

1. Applies to CLK and PICCLK.
2. Applies to all Pentium processor (610\75) inputs except CLK and PICCLK.
3. See Table 9.

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.

4.2 DC Specifications

Tables 8, 9, and 10 list the DC specifications which apply to the Pentium processor (610\7 5). The Pentium processor (610\75) is a 3.3V part internally. The CLK and PICCLK inputs may be a 3.3V or 5V inputs. Since the 3.3V (5V safe) input levels defined in Table 9 are the same as the 5V TTL levels, the CLK and PICCLK inputs are compatible with existing 5V clock drivers. The power dissipation specification in Table 11 is provided for design of thermal solutions during operation in a sustained maximum level. This is the worst-case power the device would dissipate in a system for a sustain ed period of time. This number is used for design of a thermal solution for the device.

Table 8. 3.3V DC Specifications

T_{CASE} = 0 to 95°C; V_{CC} = 3.3V ±5%

Symbol	Parameter	Min	Max	Unit	Notes
V_{IL3}	Input Low Voltage	−0.3	0.8	V	TTL Level (3)
V_{IH3}	Input High Voltage	2.0	V_{CC}+0.3	V	TTL Level(3)
V_{OL3}	Output Low Voltage		0.4	V	TTL Level(1) (3)
V_{OH3}	Output High Voltage	2.4		V	TTL Level(2) (3)
I_{CC3}	Power Supply Current		2650	mA	@75 MHz(4)

NOTES:

1. Parameter measured at 4 mA.
2. Parameter measured at 3 mA.
3. 33V TTL levels apply to all signals except CLK and PICCLK.
4. This value should be used for power supply design. It was determined usi ng a worst-case instruction mix and V_{CC} + 5%. Power supply transient response and decoupling capacitors must be sufficient to handle the instantaneous current changes occurring during transitions from stop clock to full active modes. For more information, refer to section 4.3.2.

Table 9. 3.3V (5V Safe) DC Specifications

Symbol	Parameter	Min	Max	Unit	Notes
V_{IL5}	Input Low Voltage	−0.3	0.8	V	TTL Level (1)
V_{IH5}	Input High Voltage	2.0	5.55	V	TTL Level (1)

NOTES:
1.Applies to CLK and PICCLK only.

Table 10. Input and Output Characteristics

Symbol	Parameter	Min	Max	Unit	Notes
C_{IN}	Input Capacitance		15	pF	(4)
C_O	Output Capacitance		20	pF	(4)
$C_{I/O}$	I/O Capacitance		25	pF	(4)
C_{CLK}	CLK Input Capacitance		15	pF	(4)
C_{TIN}	Test Input Capacitance		15	pF	(4)
C_{TOUT}	Test Output Capacitance		20	pF	(4)
C_{TCK}	Test Clock Capacitance		15	pF	(4)
I_{LI}	Input Leakage Current		±15	μA	$0 < V_{IN} < V_{CC3}$(1)
I_{LO}	Output Leakage Current		±15	μA	$0 < V_{IN} < V_{CC3}$(1)
I_{IH}	Input Leakage Current		200	μA	$V_{IN} = 2.4V$(3)
I_{IL}	Input Leakage Current		−400	μA	$V_{IN} = 0.4V$(2)

NOTES:
1. This parameter is for input without pull up or pull down.
2. This parameter is for input with pull up.
3. This parameter is for input with pull down.
4. Guaranteed by design.

Table 11. Power Dissipation Requirements for Thermal Solution Design

Parameter	Typical(1)	Max(2)	Unit	Notes
Active Power Dissipation	3	8.0	Watts	@ 75 MHz
Stop Grant and Auto Halt Powerdown Power Dissipation		1.2	Watts	@ 75 MHz(3)
Stop Clock Power Dissipation	0.02	≤.05	Watts	(4) (5)

NOTES:
1. This is the typical power dissipation in a system. This value was t he average value measured in a system using a typical device at V_{CC} = 3.3V running typical applications. This value is highly dependent upon the specific system configuration.
2. Systems must be designed to thermally dissipate the maximum active power dissipation. It is determined using a worst-case instruction mix with V_{CC} = 3.3V. The use of nominal V_{CC} in this measurement takes into account the thermal time constant of the package.
3. Stop Grant/Auto Halt Powerdown Power Dissipation is determined by asserting the STPCLK# pin or executing the HALT instruction.
4. Stop Clock Power Dissipation is determined by asserting the STPCLK# pin and then removing the external CLK input.
5. Complete characterization of the specification was still in process at the time of print. Please contact Intel for the latest information. The final specification may be less than 50 mW.

4.3 AC Specifications

The AC specifications of the Pentium processor (610\75) consist of setup times, hold times, and valid delays at 0 pF.

WARNING

Do not exceed the Pentium processor (610\75) internal maximum frequency of 75 MHz by either selecting the 1/2 bus fraction or providing a clock greater than 50 MHz.

4.3.1 POWER AND GROUND

For clean on-chip power distribution, the Pentium processor (610\75) has 79 V_{CC} (power) and 72 V_{SS} (ground) inputs. Power and ground connections must be made to all external V_{CC} and V_{SS} pins of the Pentium processor (610\75). On the circuit board all V_{CC} pins must be connected to a 3.3V V_{CC} plane. All V_{SS} pins must be connected to a V_{SS} plane.

4.3.2 DECOUPLING RECOMMENDATIONS

Liberal decoupling capacitance should be placed near the Pentium processor (610\75). The Pentium processor (610\75) driving its large address and data buses at high frequencies can cause transient power surges, particularly when driving large capacitive loads.

Low inductance capacitors and interconnects are recommended for best high frequency electrical performance. Inductance can be reduced by shortening circuit board traces between the Pentium processor (610\75) and decoupling capacitors as much as possible.

These capacitors should be evenly distributed around each component on the 3.3V plane. Capacitor values should be chosen to ensure they eliminate both low and high frequency noise components.

For the Pentium processor (610\ 75), the power consumption can transition from a low level of power to a much higher level (or high to low power) very rapidly. A typical example would be entering or exiting the Stop Grant state. Another example would be executing a HALT instruction, causing the Pentium processor (610\75) to enter the Auto HALT Powerdown state, or transitioning from HALT to the Normal state. All of these examples may cause abrupt changes in the power being consumed by the Pentium processor (610\75). Note that the Auto HALT Powerdown feature is always enabled even when other power management features are not implemented.

Bulk storage capacitors with a low ESR (Effective Series Resistance) in the 10 μf to 100 μf range are required to maintain a regulated supply voltage during the interval between the time the current load changes and the point that the regulated power supply output can react to the change in load. In order to reduce the ESR, it may be necessary to place several bulk storage capacitors in parallel.

These capacitors should be placed near the Pentium processor (610\75) (on the 3.3V plane) to ensure that the supply voltage stays within specified limits during changes in the supply current during operation.

4.3.3 CONNECTION SPECIFICATIONS

All NC pins must remain unconnected.

For reliable operation, always connect unused inputs to an appropriate signal level. Unused active low inputs should be connected to V_{CC}. Unused active high inputs should be connected to ground.

4.3.4 AC TIMINGS FOR A 50-MHZ BUS

The AC specifications given in Table 12 consist of output delays, input setup requirements and input hold requirements for a 50-MHz external bus. All AC specifications (with the exception of those for the TAP signals and APIC signals) are relative to the rising edge of the CLK input.

All timings are referenced to 1.5V for both "0" and "1" logic levels unless otherwise specified. Within the sampling window, a synchronous input must be stable for correct Pentiu m processor (610\75) operation.

Table 12. Pentium™ Processor (610\75) TCP AC Specifications for 50-MHz Bus Operation

V_{CC} = 3.3V± 5%, T_{CASE} = 0°C to 95°C, C_L = 0 pF

Symbol	Parameter	Min	Max	Unit	Figure	Notes
	Frequency	25.0	50.0	MHz		Max Core Freq. = 75 MHz @ 2/3
t_{1a}	CLK Period	20.0	40.0	nS	3	
t_{1b}	CLK Period Stability		250	pS		(1), (25)
t_2	CLK High Time	4.0		nS	3	@2V, (1)
t_3	CLK Low Time	4.0		nS	3	@0.8V, (1)
t_4	CLK Fall Time	0.15	1.5	nS	3	(2.0V–0.8V), (1), (5)
t_5	CLK Rise Time	0.15	1.5	nS	3	(0.8V–2.0V), (1), (5)
t_{6a}	PWT, PCD, M/IO#, CACHE# Valid Delay	1.0	7.0	nS	4	
t_{6b}	AP Valid Delay	1.0	8.5	nS	4	
t_{6c}	A3–A31, BE0–7# Valid Delay	0.7	7.0	nS	4	
t_{6d}	D/C#, SCYC, LOCK# Valid Delay	0.9	7.0	nS		
t_{6e}	ADS# Valid Delay	0.8	7.0	nS		
t_{6f}	W/R# Valid Delay	0.5	7.0	nS		

Table 12. Pentium™ Processor (610\75) TCP AC Specifications for 50-MHz Bus Operation (Continued)

V_{CC} = 3.3V ± 5%, T_{CASE} = 0°C to 95°C, C_L = 0 pF

Symbol	Parameter	Min	Max	Unit	Figure	Notes
t_7	ADS#, AP, A3-A31, PWT, PCD, BE0-7#, M/IO#, D/C#, W/R#, CACHE#, SCYC, LOCK# Float Delay		10.0	nS	5	(1)
t_8	APCHK#, IERR#, FERR#, PCHK# Valid Delay	1.0	8.3	nS	4	(4)
t_{9a}	BREQ, HLDA, SMIACT# Valid Delay	1.0	8.0	nS	4	(4)
t_{10a}	HIT# Valid Delay	1.0	8.0	nS	4	
t_{10b}	HITM# Valid Delay	0.5	6.0	nS	4	
t_{11a}	PM0-1, BP0-3 Valid Delay	1.0	10.0	nS	4	
t_{11b}	PRDY Valid Delay	1.0	8.0	nS	4	
t_{12}	D0-D63, DP0-7 Write Data Valid Delay	1.3	8.5	nS	4	
t_{13}	D0-D63, DP0-3 Write Data Float Delay		10.0	nS	5	(1)
t_{14}	A5-A31 Setup Time	6.5		nS	6	(26)
t_{15}	A5-A31 Hold Time	1.0		nS	6	
t_{16a}	INV, AP Setup Time	5.0		nS	6	
t_{16b}	EADS# Setup Time	6.0		nS	6	
t_{17}	EADS#, INV, AP Hold Time	1.0		nS	6	
t_{18a}	KEN# Setup Time	5.0		nS	6	
t_{18b}	NA#, WB/WT# Setup Time	4.5		nS	6	
t_{19}	KEN#, WB/WT#, NA# Hold Time	1.0		nS	6	
t_{20}	BRDY# Setup Time	5.0		nS	6	
t_{21}	BRDY# Hold Time	1.0		nS	6	
t_{22}	BOFF# Setup Time	5.5		nS	6	
t_{22a}	AHOLD Setup Time	6.0		nS	6	
t_{23}	AHOLD, BOFF# Hold Time	1.1		nS	6	

Table 12. Pentium™ Processor (610\75) TCP AC Specifications for 50-MHz Bus Operation (Continued)

V_{CC} = 3.3V ± 5%, T_{CASE} = 0°C to 95°C, C_L = 0 pF

Symbol	Parameter	Min	Max	Unit	Figure	Notes
t_{24}	BUSCHK#, EWBE#, HOLD, PEN# Setup Time	5.0		nS	6	
t_{25}	BUSCHK#, EWBE #, PEN# Hold Time	1.0		nS	6	
t_{25a}	HOLD Hold Time	1.5		nS	6	
t_{26}	A20M#, INTR, STPCLK# Setup Time	5.0		nS	6	(12), (16)
t_{27}	A20M#, INTR, STPCLK# Hold Time	1.0		nS	6	(13)
t_{28}	INIT, FLUSH#, NMI, SMI#, IGNNE# Setup Time	5.0		nS	6	(12), (16), (17)
t_{29}	INIT, FLUSH#, NMI, SMI#, IGNNE# Hold Time	1.1		nS	6	(13)
t_{30}	INIT, FLUSH#, NMI, SMI#, IGNNE# Pulse Width, Async	2.0		CLKs	6	(15), (17)
t_{31}	R/S# Setup Time	5.0		nS	6	(12), (16), (17)
t_{32}	R/S# Hold Time	1.0		nS	6	(13)
t_{33}	R/S# Pulse Width, Async.	2.0		CLKs	6	(15), (17)
t_{34}	D0–D63, DP0–7 Read Data Setup Time	3.8		nS	6	
t_{35}	D0–D63, DP0–7 Read Data Hold Time	2.0		nS	6	
t_{36}	RESET Setup Time	5.0		nS	7	(11), (12), (16)
t_{37}	RESET Hold Time	1.0		nS	7	(11), (13)
t_{38}	RESET Pulse Width, Vcc & CLK Stable	15		CLKs	7	(11), (17)
t_{39}	RESET Active After Vcc & CLK Stable	1.0		mS	7	Power up
t_{40}	Reset Configuration Signals (INIT, FLUSH#) Setup Time	5.0		nS	7	(12), (16), (17)
t_{41}	Reset Configuration Signals (INIT, FLUSH#) Hold Time	1.0		nS	7	(13)

Table 12. Pentium™ Processor (610\75) TCP AC Specifications for 50-MHz Bus Operation (Continued)

V_{CC} = 3.3V ± 5%, T_{CASE} = 0°C to 95°C, C_L = 0 pF

Symbol	Parameter	Min	Max	Unit	Figure	Notes
t_{42a}	Reset Configuration Signals (INIT, FLUSH#) Setup Time, Async.	2.0		CLKs	7	To RESET falling edge (16)
t_{42b}	Reset Configuration Signals (INIT, FLUSH#, BRDY#, BUSCHK#) Hold Time, Async.	2.0		CLKs	7	To RESET falling edge (27)
t_{42c}	Reset Configuration Signal (BRDY#, BUSCHK#) Setup Time, Async.	3.0		CLKs	7	To RESET falling edge (27)
t_{42d}	Reset Configuration Signal BRDY# Hold Time, RESET driven synchronously	1.0		ns		To RESET falling edge (1), (27)
t_{43a}	BF Setup Time	1.0		ms	7	(22) to RESET falling edge
t_{43b}	BF Hold Time	2.0		CLKs	7	(22) to RESET falling edge
t_{43c}	APICEN Setup Time	2.0		CLKs	7	To RESET falling edge
t_{43d}	APICEN Hold Time	2.0		CLKs	7	To RESET falling edge
t_{44}	TCK Frequency		16.0	MHz		
t_{45}	TCK Period	62.5		ns	3	
t_{46}	TCK High Time	25.0		ns	3	@2V, (1)
t_{47}	TCK Low Time	25.0		ns	3	@0.8V, (1)
t_{48}	TCK Fall Time		5.0	ns	3	(2.0V–0.8V), (1), (8), (9)
t_{49}	TCK Rise Time		5.0	ns	3	(0.8V–2.0V), (1), (8), (9)
t_{50}	TRST# Pulse Width	40.0		ns	9	(1), Asynchronous
t_{51}	TDI, TMS Setup Time	5.0		ns	8	(7)
t_{52}	TDI, TMS Hold Time	13.0		ns	8	(7)
t_{53}	TDO Valid Delay	3.0	20.0	ns	8	(8)
t_{54}	TDO Float Delay		25.0	ns	8	(1), (8)
t_{55}	All Non-Test Outputs Valid Delay	3.0	20.0	ns	8	(3), (8), (10)

Table 12. Pentium™ Processor (610\75) TCP AC Specifications for 50-MHz Bus Operation (Continued)

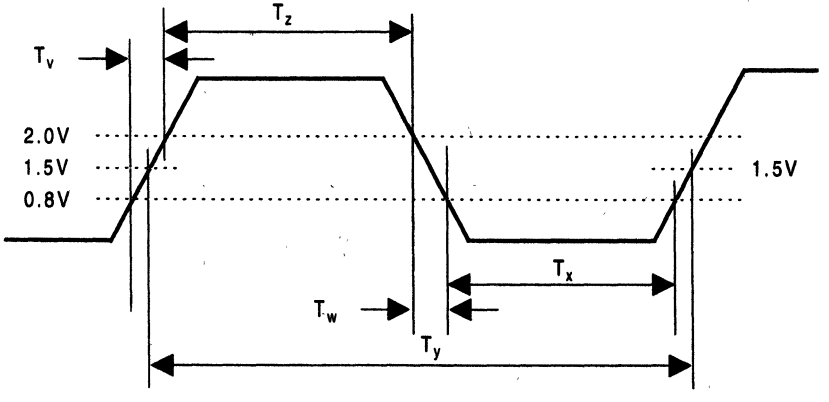
V_{CC} = 3.3V ± 5%, T_{CASE} = 0°C to 95°C, C_L = 0 pF

Symbol	Parameter	Min	Max	Unit	Figure	Notes
t_{56}	All Non-Test Outputs Float Delay		25.0	ns	8	(1), (3), (8), (10)
t_{57}	All Non-Test Inputs Setup Time	5.0		ns	8	(3), (7), (10)
t_{58}	All Non-Test Inputs Hold Time	13.0		ns	8	(3), (7), (10)
APIC AC Specifications						
t_{60a}	PICCLK Frequency	2.0	16.66	MHz		
t_{60b}	PICCLK Period	60.0	500.0	ns	3	
t_{60c}	PICCLK High Time	9.0		ns	3	
t_{60d}	PICCLK Low Time	9.0		ns	3	
t_{60e}	PICCLK Rise Time	1.0	5.0	ns	3	
t_{60f}	PICCLK Fall Time	1.0	5.0	ns	3	
t_{60g}	PICD0-1 Setup Time	3.0		ns	6	to PICCLK
t_{60h}	PICD0-1 Hold Time	2.5		ns	6	to PICCLK
t_{60i}	PICD0-1 Valid Delay (LtoH)	4.0	38.0	ns	4	from PICCLK, (28)
t_{60j}	PICD0-1 Valid Delay (HtoL)	4.0	22.0	ns	4	from PICCLK, (28)

NOTES:
Notes 2, 6, and 14 are general and apply to all standard TTL signals used with the Pentium Processor family.
Notes 11, 18, 19, 20, 23, and 24 do not apply to the TCP package and have been removed in this document.
1. Not 100% tested. Guaranteed by design.
2. TTL input test waveforms are assumed to be 0 to 3V transitions with 1V/ns rise and fall times.
3. Non-test outputs and inputs are the normal output or input signals (besides TCK, TRST#, TDI, TDO, and TMS). These timings correspond to the response of these signals due to boundary scan operations.
4. APCHK#, FERR#, HLDA, IERR#, LOCK#, and PCHK# are glitch-free outputs. Glitch-free signals monotonically transition without false transitions (i.e., glitches).
5. 0.8V/ns ≤ CLK input rise/fall time ≤ 8V/ns.
6. 0.3V/ns ≤ input rise/fall time ≤ 5V/ns.
7. Referenced to TCK rising edge.
8. Referenced to TCK falling edge.
9. 1 ns can be added to the maximum TCK rise and fall times for every 10 MHz of frequency below 33 MHz.
10. During probe mode operation, do not use the boundary scan timings (t_{55-58}).
12. Setup time is required to guarantee recognition on a specific clock.
13. Hold time is required to guarantee recognition on a specific clock.
14. All TTL timings are referenced from 1.5V.
15. To guarantee proper asynchronous recognition, the signal must have been de-asserted (inactive) for a minimum of 2 clocks before being returned active and must meet the minimum pulse width.
16. This input may be driven asynchronously.
17. When driven asynchronously, RESET, NMI, FLUSH#, R/S#, INIT, and SMI# must be de-asserted (inactive) for a minimum of 2 clocks before being returned active.

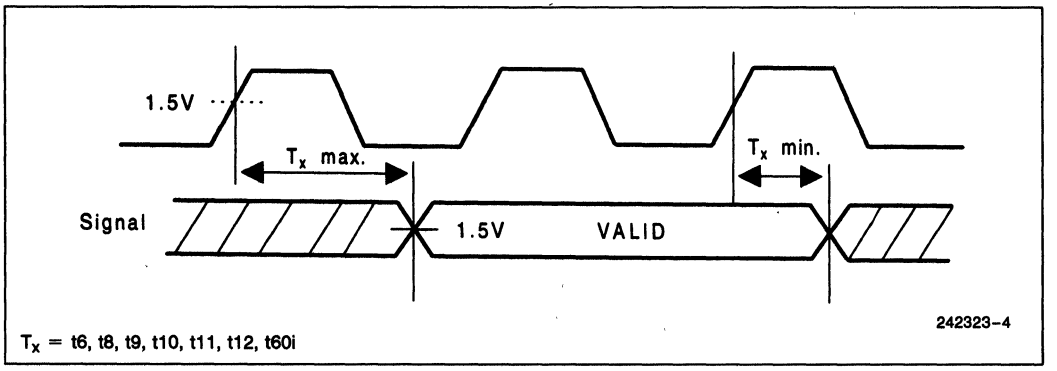
21. The D/C#, M/IO#, W/R#, CACHE#, and A5–A31 signals are sampled only on the CLK that ADS# is active.
22. BF should be strapped to V_{CC} or V_{SS}.
25. These signals are measured on the rising edge of adjacent CLKs at 1.5V. To ensure a 1:1 relationship between the amplitude of the input jitter and the internal and external clocks, the jitter frequency spectrum should not have any power spectrum peaking between 500 KHz and 1/3 of the CLK operating frequency. The amount of jitter present must be accounted for as a component of CLK skew between devices.
26. Timing t_{14} is required for external snooping (e.g., address setup to the CLK in which EADS# is sampled active).
27. BUSCHK# is used as a reset configuration signal to select buffer size.
28. This assumes an external pullup resistor to V_{CC} and a lumped capacitive load. The pullup resistor must be between 150 ohms and 1K ohms, the capacitance must be between 20 pF and 240 pF, and the RC product must be between 3 ns and 36 ns.

** Each valid delay is specified for a 0 pF load. The system designer should use I/O buffer modeling to account for signal flight time delays.



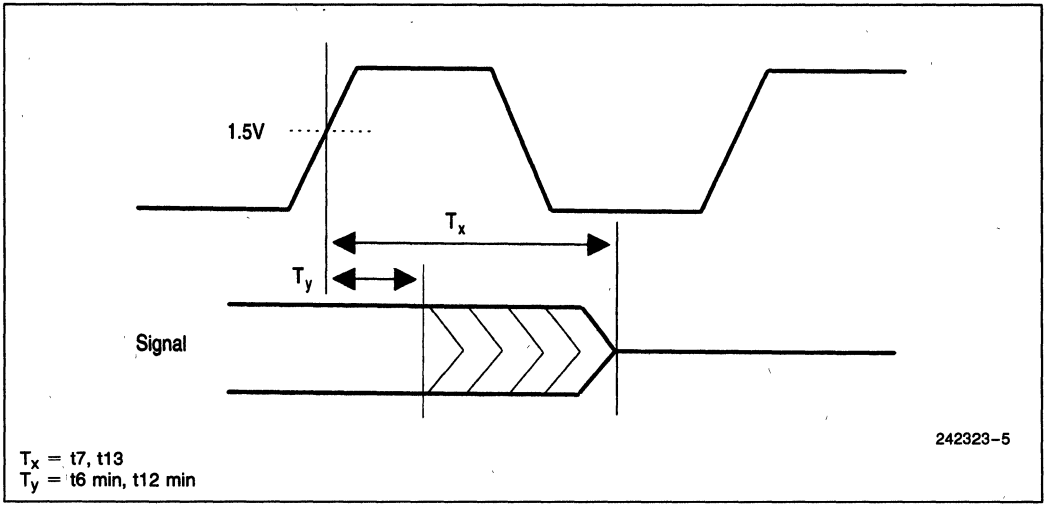
T_v = t5, t49, t60e
T_w = t4, t8,t60f
T_x = t3, t47, t60d
T_y = t1, t45, t60b
T_z = t2, t46, t60c

Figure 3. Clock Waveform



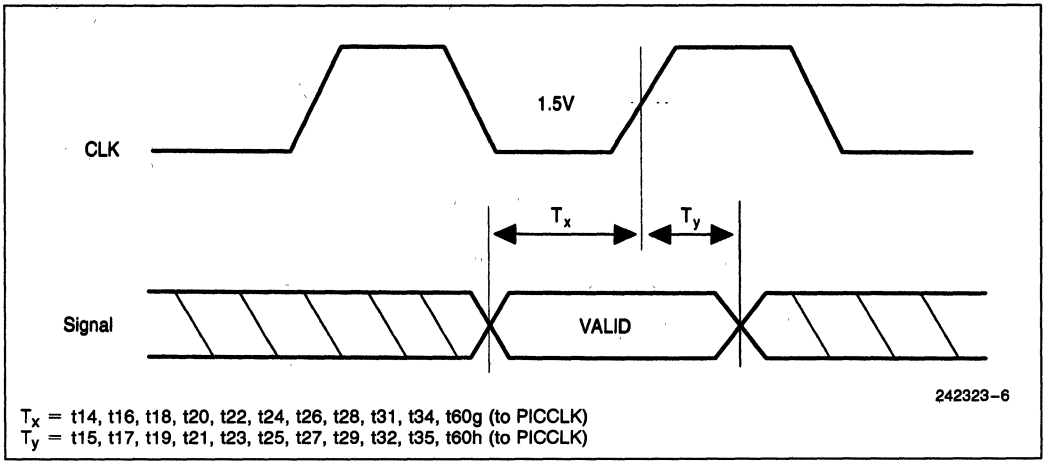
T_x = t6, t8, t9, t10, t11, t12, t60i

Figure 4. Valid Delay Timings



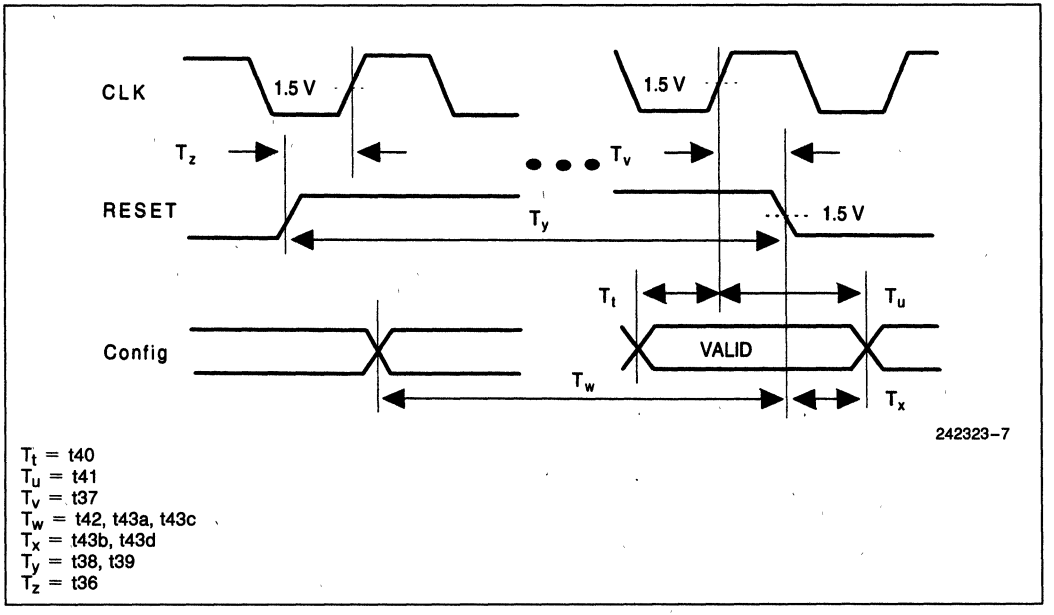
T_x = t7, t13
T_y = t6 min, t12 min

Figure 5. Float Delay Timings



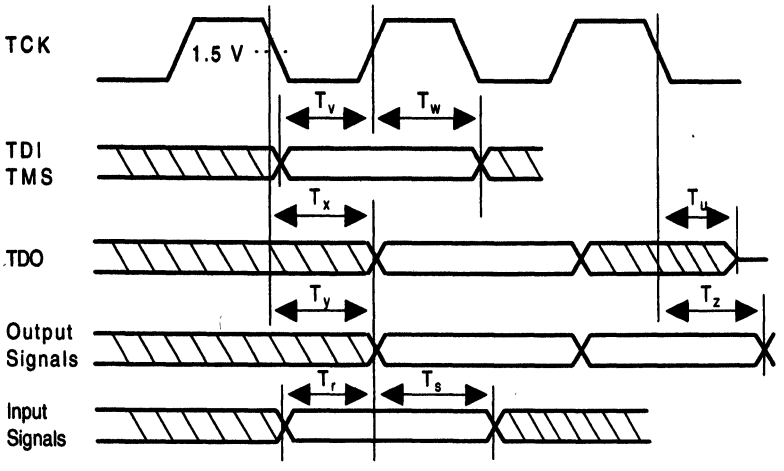
T_x = t14, t16, t18, t20, t22, t24, t26, t28, t31, t34, t60g (to PICCLK)
T_y = t15, t17, t19, t21, t23, t25, t27, t29, t32, t35, t60h (to PICCLK)

Figure 6. Setup and Hold Timings



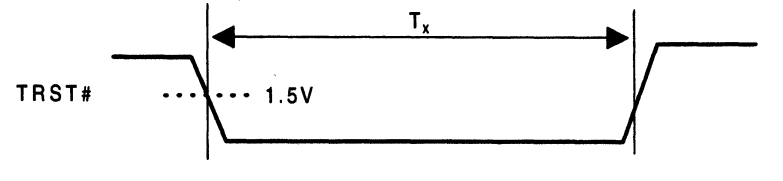
T_t = t40
T_u = t41
T_v = t37
T_w = t42, t43a, t43c
T_x = t43b, t43d
T_y = t38, t39
T_z = t36

Figure 7. Reset and Configuration Timings



T_r = t57
T_s = t58
T_u = t54
T_v = t51
T_w = t52
T_x = t53
T_y = t55
T_z = t56

Figure 8. Test Timings



T_x = t50

Figure 9. Test Reset Timings

4.4 I/O Buffer Models

This section describes the I/O buffer models of the Pentium processor (610\75).

The first order I/O buffer model is a simplified representation of the complex input and output buffers used in the Pentium processor (610\75). Figures 10 and 11 show the structure of the input buffer model and Figure 12 shows the output buffer model. Tables 13 and 14 show the parameters used to specify these models.

Although simplified, these buffer models will accurately model flight time and signal quality. For these parameters, there is very little added accuracy in a complete transistor model.

The following two models represent the input buffer models. The first model, Figure 10, represents all of the input buffers of the Pentium processor (610\75) except for a special group of input buffers. The second model, Figure 11, represents these special buffers. These buffers are the inputs: AHOLD, EADS#, KEN#, WB/WT#, INV, NA#, EWBE#, BOFF#, CLK, and PICCLK.

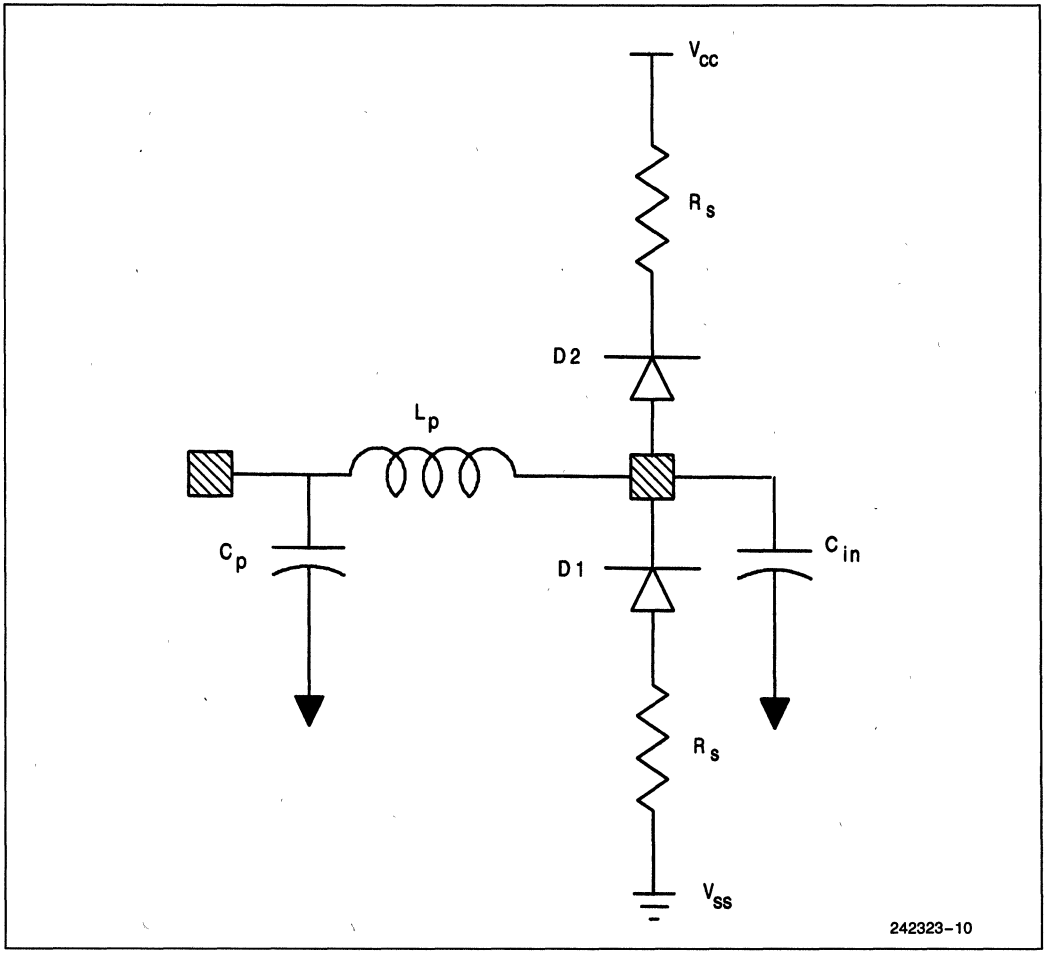


Figure 10. Input Buffer Model, Except Special Group

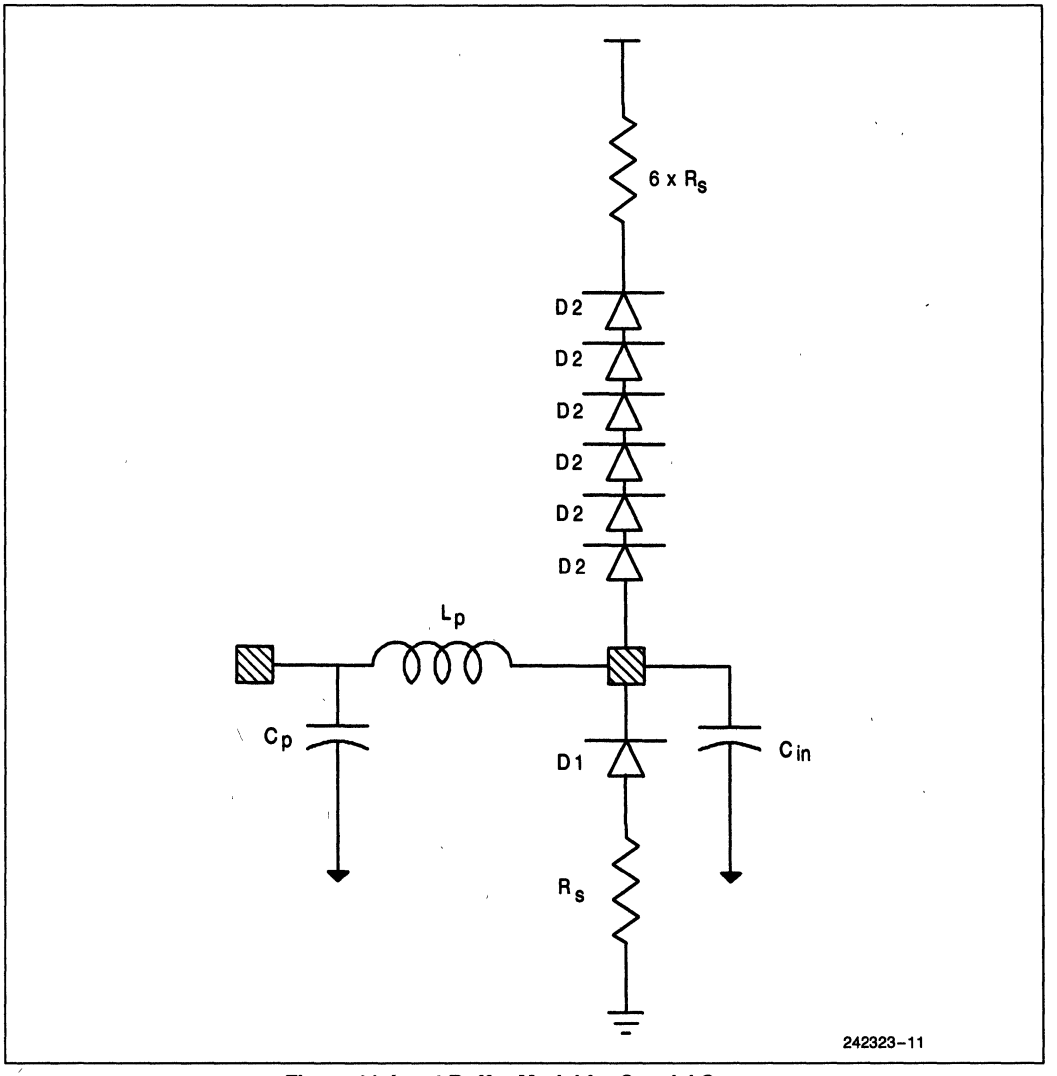


Figure 11. Input Buffer Model for Special Group

Table 13. Parameters Used in the Specification of the First Order Input Buffer Model

Parameter	Description
Cin	Minimum and Maximum value of the capacitance of the input buffer model.
Lp	Minimum and Maximum value of the package inductance.
Cp	Minimum and Maximum value of the package capacitance.
Rs	Diode Series Resistance
D1, D2	Ideal Diodes

Figure 12 shows the structure of the output buffer model. This model is used for all of the output buffers of the Pentium processor (610\75).

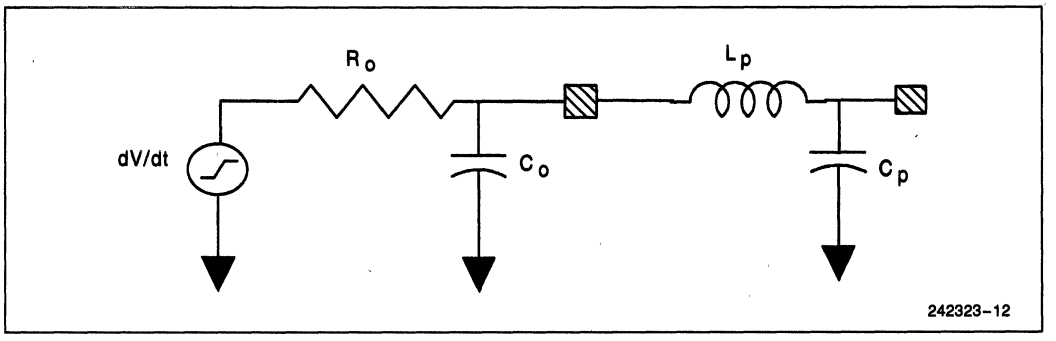


Figure 12. First Order Output Buffer Model

Table 14. Parameters Used in the Specification of the First Order Output Buffer Model

Parameter	Description
dV/dt	Minimum and maximum value of the rate of change of the open circuit voltage source used in the output buffer model.
Ro	Minimum and maximum value of the output impedance of the output buffer model.
Co	Minimum and Maximum value of the capacitance of the output buffer model.
Lp	Minimum and Maximum value of the package inductance.
Cp	Minimum and Maximum value of the package capacitance.

In addition to the input and output buffer parameters, input protection diode models are provided for added accuracy. These diodes have been optimized to provide ESD protection and provide some level of clamping. Although the diodes are not required for simulation, it may be more difficult to meet specifications without them.

Note, however, some signal quality specifications require that the diodes be removed from the input model. The series resistors (Rs) are a part of the diode model. Remove these when removing the diodes from the input model.

4.4.1 BUFFER MODEL PARAMETERS

This section gives the parameters for each Pentium processor (610\75) input, output, and bidirectional signal, as well as the settings for the configurable buffers.

Some pins on the Pentium processor (610\75) have selectable buffer sizes. These pins use the configurable output buffer EB2. Table 15 shows the drive level for BRDY# required at the falling edge of RESET to select the buffer strength. The buffer sizes selected should be the appropriate size required; otherwise AC timings might not be met, or too much overshoot and ringback may occur. There are no other selection choices; all of the configurable buffers get set to the same size at the same time.

Table 15. Buffer Selection Chart

Environment	BRDY#	Buffer Selection
Typical Stand Alone Component	1	EB2
Loaded Component	0	EB2A

NOTES:
For correct buffer selection, the BUSCHK# signal must be held inactive (high) at the falling edge of RESET.
For the Pentium processor (610\75) SPGA version, BRDYC# is used to configure selectable buffer sizes.

Please refer to Table 16 for the groupings of the buffers.

Table 16. Signal to Buffer Type

Signals	Type	Driver Buffer Type	Receiver Buffer Type
CLK	I		ER0
A20M#, AHOLD, BF, BOFF#, BRDY#, BUSCHK#, EADS#, EWBE#, FLUSH#, HOLD, IGNNE#, INIT, INTR, INV, KEN#, NA#, NMI, PEN#, PICCLK, R/S#, RESET, SMI#, STPCLK#, TCK, TDI, TMS, TRST#, WB/WT#	I		ER1
APCHK#, BE[7:5]#, BP[3:2], BREQ, FERR#, IERR#, PCD, PCHK#, PM0/BP0, PM1/BP1, PRDY, PWT, SMIACT#, TDO, U/O#	O	ED1	
A[31:21], AP, BE[4:0]#, CACHE#, D/C#, D[63:0], DP[8:0], HLDA, LOCK#, M/IO#, SCYC	I/O	EB1	EB1
A[20:3], ADS#, HITM#, W/R#	I/O	EB2A	EB2
HIT#	I/O	EB3	EB3
PID0, PICD1	I/O	EB4	EB4

The input, output and bidirectional buffer values are listed in Table 17. This table contains listings for all three types, do not get them confused during simulation. When a bidirectional pin is operating as an input, just use the Cin, Cp and Lp values; if it is operating as a driver, use all of the data parameters.

Table 17. Input, Output and Bidirectional Buffer Model Parameters

Buffer Type	Transition	dV/dt (V/nsec)		Ro (Ohms)		Cp (pF)		Lp (nH)		Co/Cin (pF)	
		min	max	min	max	min	max	min	max	min	max
ER0	Rising					0.3	0.4	3.9	5.0	0.8	1.2
(input)	Falling					0.3	0.4	3.9	5.0	0.8	1.2
ER1	Rising					0.2	0.5	3.1	6.0	0.8	1.2
(input)	Falling					0.2	0.5	3.1	6.0	0.8	1.2
ED1	Rising	3/3.0	3.7/0.9	21.6	53.1	0.3	0.6	3.7	6.6	2.0	2.6
(output)	Falling	3/2.8	3.7/0.8	17.5	50.7	0.3	0.6	3.7	6.6	2.0	2.6
EB1	Rising	3/3.0	3.7/	21.6.9	53.1	0.2	0.5	2.9	6.1	2.0	2.6
(bidir)	Falling	3/2.8	3.7/0.8	17.5	50.7	0.2	0.5	2.9	6.1	2.0	2.6
EB2	Rising	3/3.0	3.7/0.9	21.6	53.1	0.2	0.5	3.1	6.4	9.1	9.7
(bidir)	Falling	3/2.8	3.7/0.8	17.5	50.7	0.2	0.5	3.1	6.4	9.1	9.7
EB2A	Rising	3/2.4	3.7/0.9	10.1	22.4	0.2	0.5	3.1	6.4	9.1	9.7
(bidir)	Falling	3/2.4	3.7/0.9	9.0	21.2	0.2	0.5	3.1	6.4	9.1	9.7
EB3	Rising	3/3.0	3.7/0.9	21.6	53.1	0.2	0.4	3.2	4.1	3.3	3.9
(bidir)	Falling	3/2.8	3.7/0.8	17.5	50.7	0.2	0.4	3.2	4.1	3.3	3.9
EB4	Rising	3/3.0	3.7/0.9	21.6	53.1	0.3	0.4	4.0	4.1	5.0	7.0
(bidir)	Falling	3/2.8	3.7/0.8	17.5	50.7	0.3	0.4	4.0	4.1	5.0	7.0

Table 18. Input Buffer Model Parameters: D (Diodes)

Symbol	Parameter	D1	D2
IS	Saturation Current	1.4e-14A	2.78e-16A
N	Emission Coefficient	1.19	1.00
RS	Series Resistance	6.5 ohms	6.5 ohms
TT	Transit Time	3 ns	6 ns
VJ	PN Potential	0.983V	0.967V
CJ0	Zero Bias PN Capacitance	0.281 pF	0.365 pF
M	PN Grading Coefficient	0.385	0.376

4.4.2 SIGNAL QUALITY SPECIFICATIONS

Signals driven by the system into the Pentium processor (610\75) must meet signal quality specifications to guarantee that the components read data properly and to ensure that incoming signals do not affect the reliability of the component. There are two signal quality parameters: Ringback and Settling Time.

4.4.2.1 Ringback

Excessive ringback can contribute to long-term reliability degradation of the Pentium processor (610\75), and can cause false signal detection. Ringback is simulated at the input pin of a component using the input buffer model. Ringback can be simulated with or without the diodes that are in the input buffer model.

Ringback is the absolute value of the maximum voltage at the receiving pin below V_{CC} (or above V_{SS}) relative to V_{CC} (or V_{SS}) level after the signal has reached its maximum voltage level. The input diodes are assumed present.

> Maximum Ringback on Inputs = 0.8V (with diodes)

If simulated without the input diodes, follow the Maximum Overshoot/Undershoot specification. By meeting the overshoot/undershoot specification, the signal is guaranteed not to ringback excessively.

If simulated with the diodes present in the input model, follow the maximum ringback specification.

Overshoot (Undershoot) is the absolute value of the maximum voltage above V_{CC} (below V_{SS}). The guideline assumes the absence of diodes on the input.

- Maximum Overshoot/Undershoot on 5V 82497 Cache Controller, and 82492 Cache SRAM Inputs (CLK and PICCLK only) = 1.6V above V_{CC5} (without diodes)
- Maximum Overshoot/Undershoot on 3.3V Pentium processor (610\75) Inputs (not CLK and PICCLK) = 1.4V above V_{CC3} (without diodes)

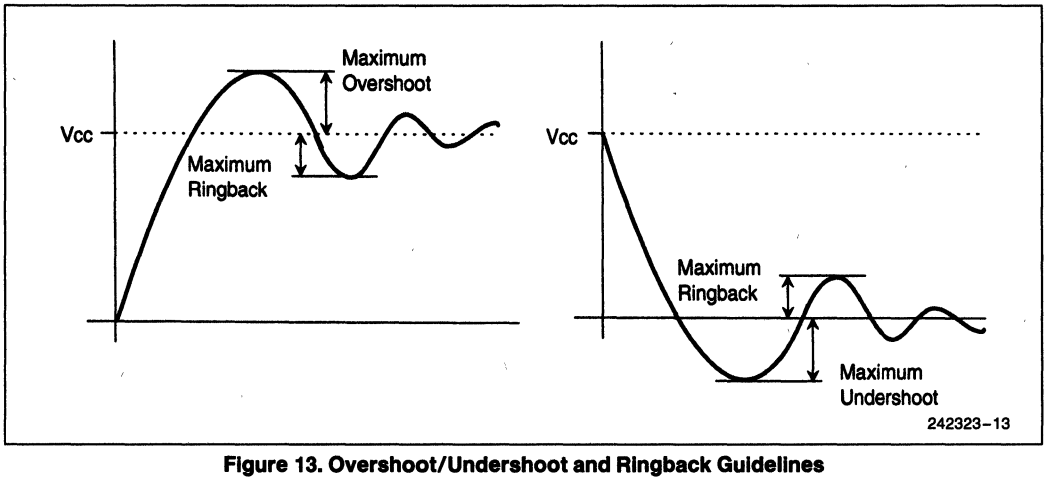


Figure 13. Overshoot/Undershoot and Ringback Guidelines

4.4.2.2 Settling Time

The settling time is defined as the time a signal requires at the receiver to settle within 10% of V_{CC} or V_{SS}. Settling time is the maximum time allowed for a signal to reach within 10% of its final value.

Most available simulation tools are unable to simulate settling time so that it accurately reflects silicon measurements. On a physical board, second-order effects and other effects serve to dampen the signal at the receiver. Because of all these concerns, settling time is a recommendation or a tool for layout tuning and not a specification.

Settling time is simulated at the slow corner, to make sure that there is no impact on the flight times of the signals if the waveform has not settled. Settling time may be simulated with the diodes included or excluded from the input buffer model. If diodes

are included, settling time recommendation will be easier to meet.

Although simulated settling time has not shown good correlation with physical, measured settling time, settling time simulations can still be used as a tool to tune layouts.

Use the following procedure to verify board simulation and tuning with concerns for settling time.

1. Simulate settling time at the slow corner for a particular signal.
2. If settling time violations occur, simulate signal trace with D.C. diodes in place at the receiver pin. The D.C. diode behaves almost identically to the actual (non-linear) diode on the part as long as excessive overshoot does not occur.
3. If settling time violations still occur, simulate flight times for 5 consecutive cycles for that particular signal.
4. If flight time values are consistent over the 5 simulations, settling time should not be a concern. If however, flight times are not consistent over the 5 simulations, tuning of the layout is required.
5. Note that, for signals that are allocated 2 cycles for flight time, the recommended settling time is doubled.

A typical design method would include a settling time that ensures a signal is within 10% of V_{CC} or V_{SS} for at least 2.5 ns prior to the end of the CLK period.

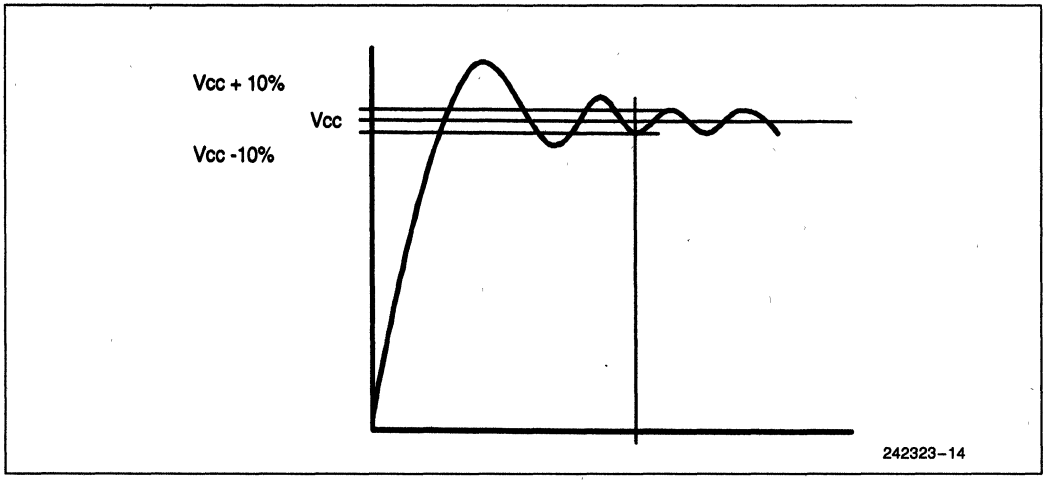


Figure 14. Settling Time

5.0 Pentium™ Processor (610\75) TCP MECHANICAL SPECIFICATIONS

Today's portable computers face the challenge of meeting desktop performance in an environment that is constrained by thermal, mechanical, and electrical design considerations. These considerations have driven the development and implementation of Intel's Tape Carrier Package (TCP). The Intel TCP package has been designed to offer a high pin count, low profile, reduced footprint package with uncompromised thermal and electrical performance. Intel continues to provide packaging solutions that meet our rigorous criteria for quality and performance, and this new entry into the Intel package portfolio is no exception.

Key features of the TCP package include: surface mount technology design, lead pitch of 0.25 mm, polyimide body size of 24 mm and polyimide up for pick&place handling. TCP components are shipped with the leads flat in slide carriers, and are designed to be excised and lead formed at the customer manufacturing site. Recommendations for the manufacture of this package are included in the Pentium™ Processor (610\75) *Tape Carrier Package User' s Guide*.

Figure 15 shows a cross-sectional view of the TCP package as mounted on the Printed Circuit Board. Figures 16 and 17 show the TCP as shipped in its slide carrier, and key dimensions of the carrier and package. Figure 18 shows a blow up detail of the package in cross-section. Figure 19 shows an enlarged view of the outer lead bond area of the package.

Tables 19 and 20 provide Pentium processor (610\75) TCP package dimensions.

2

5.1 TCP Package Mechanical Diagrams

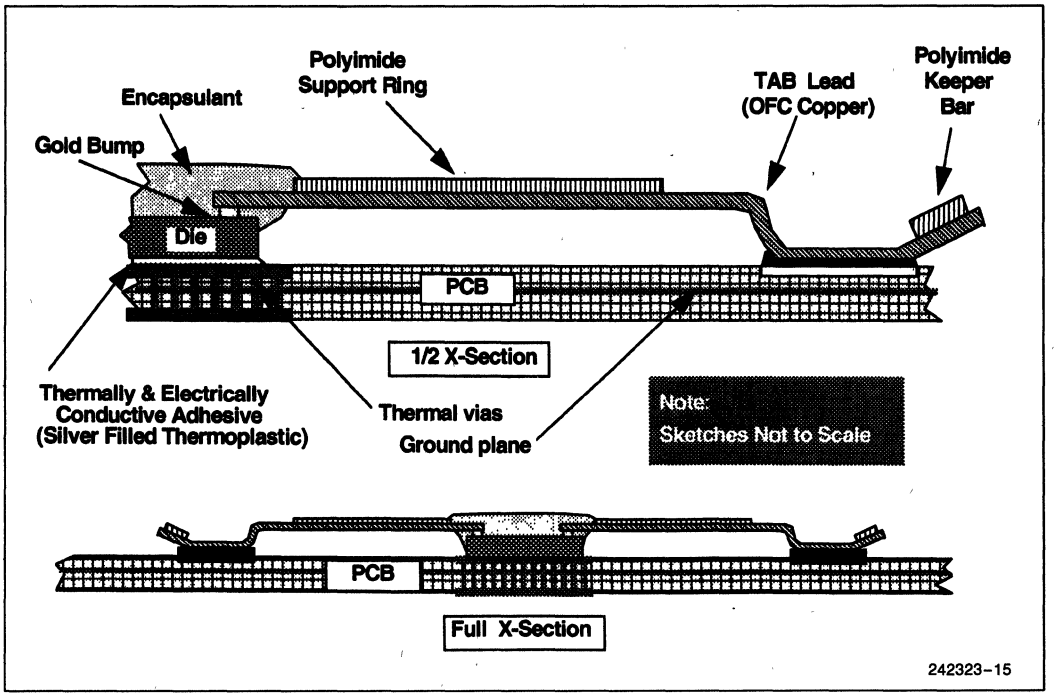


Figure 15. Cross-Sectional View of the Mounted TCP Package

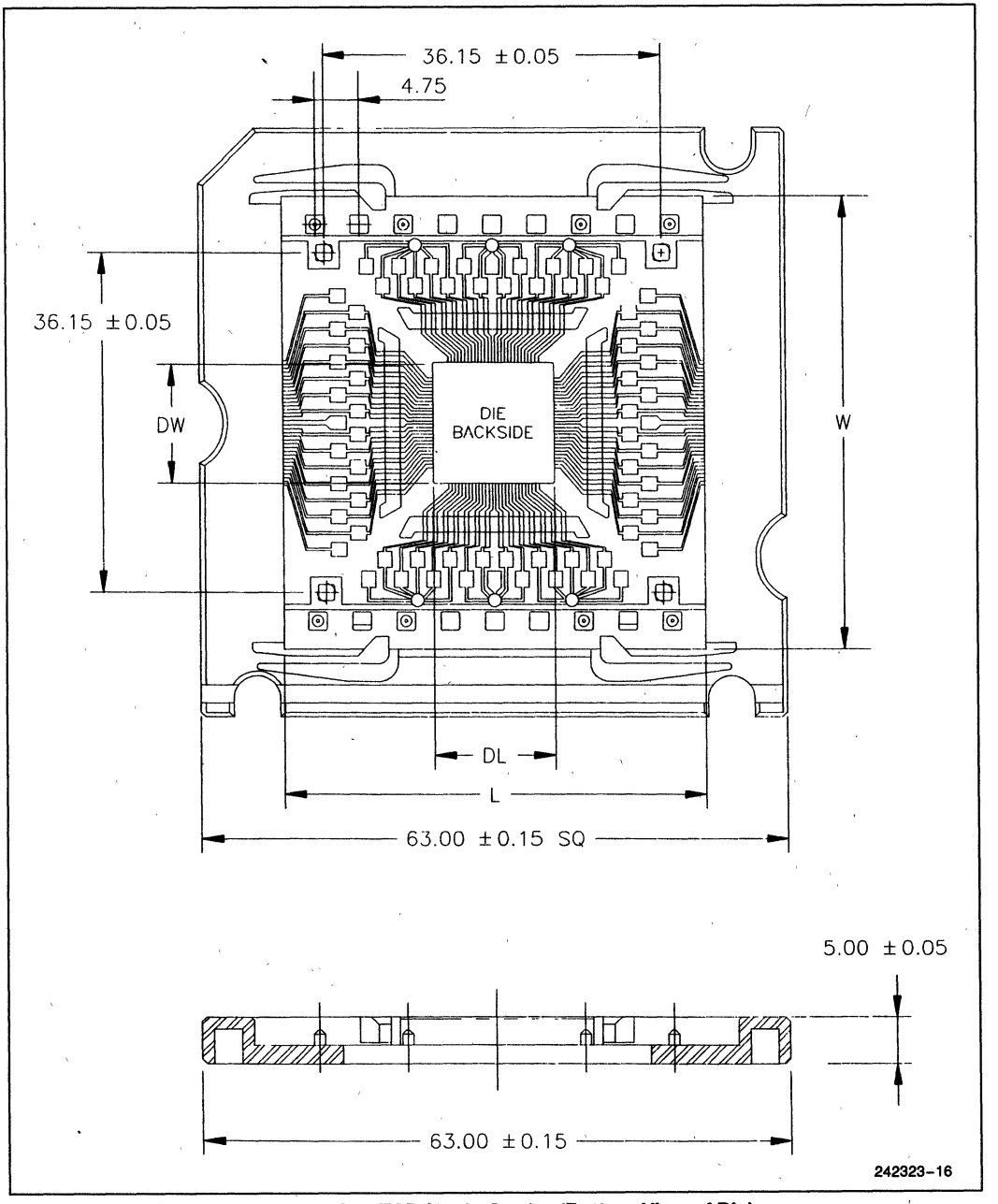


Figure 16. One TCP Site in Carrier (Bottom View of Die)

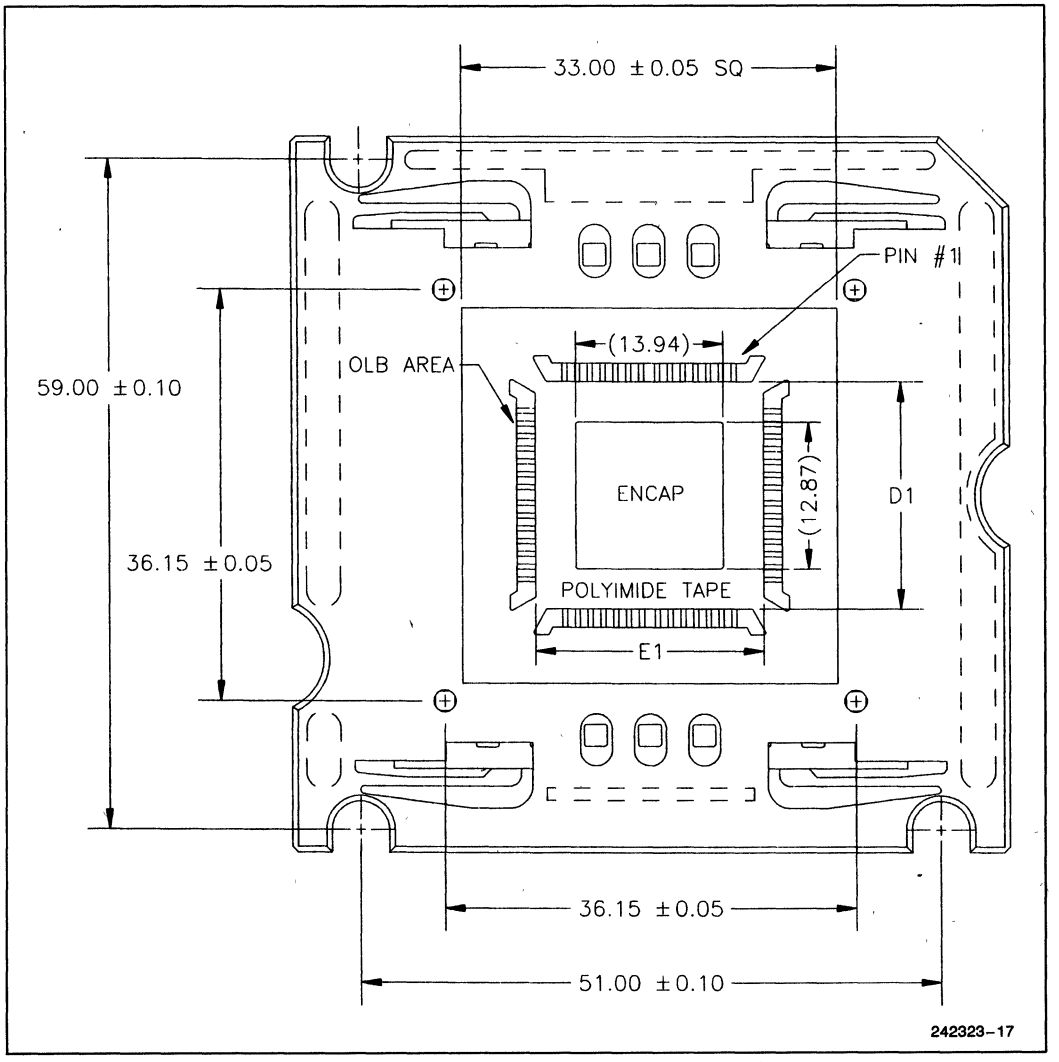


Figure 17. One TCP Site in Carrier (Top View of Die)

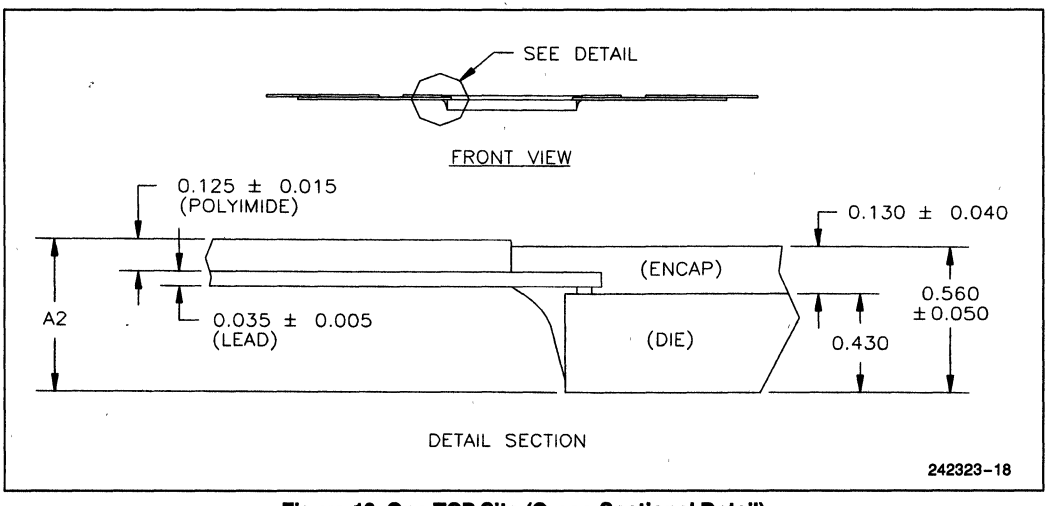


Figure 18. One TCP Site (Cross-Sectional Detail)

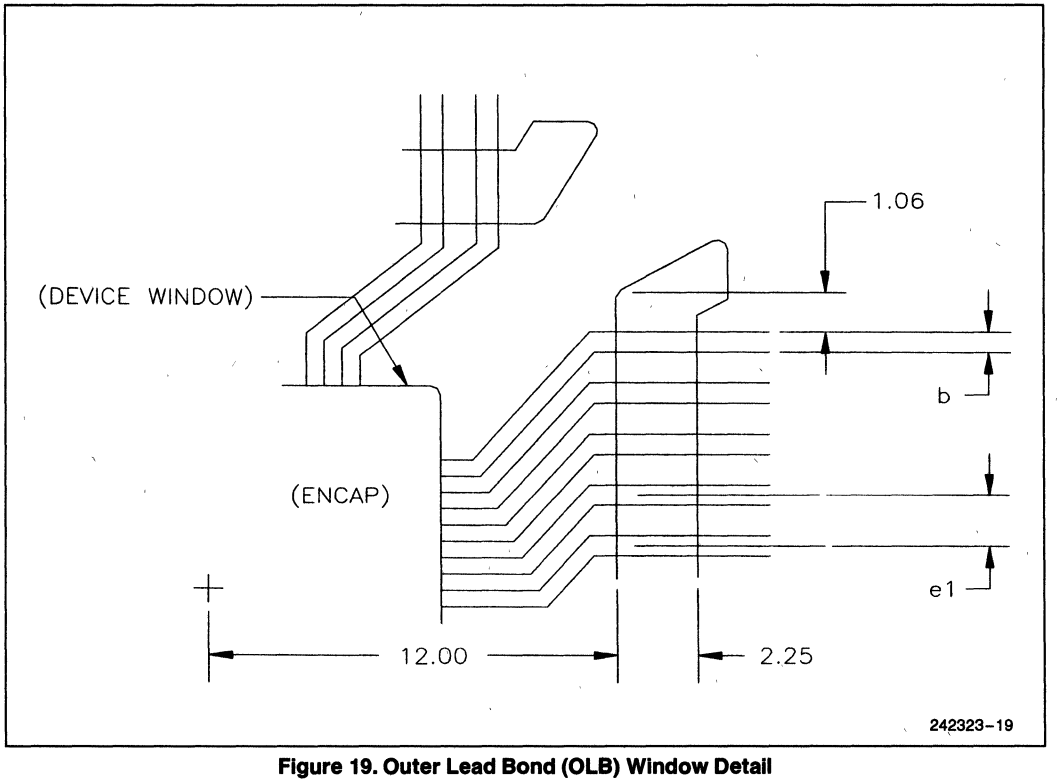


Figure 19. Outer Lead Bond (OLB) Window Detail

Table 19. TCP Key Dimensions

Symbol	Description	Dimension
N	Leadcount	320 leads
W	Tape Width	48.18 ± 0.12
L	Site Length	(43.94) ref.
e1	Outer Lead Pitch	0.25 nom.
b	Outer Lead Width	0.10 ± 0.01
D1,E1	Package Body Size	24.0 ± 0.1
A2	Package Height	0.615 ± 0.030
DL	Die Length	13.302 ± 0.015
DW	Die Width	12.235 ± 0.015

NOTES:
Dimensions are in millimeters unless otherwise noted.
Dimensions in parentheses are for reference only.

Table 20. Mounted TCP Package Dimensions

Description	Dimension
Package Height	0.75 max.
Terminal Dimension	29.5 nom.
Package Weight	0.5 g max.

NOTE:
Dimensions are in millimeters unless otherwise noted.
Package terminal dimension (lead tip-to-lead tip) assumes the use of a keeper bar.

6.0 Pentium™ Processor (610\75) TCP THERMAL SPECIFICATIONS

The Pentium processor (610\75) is specified for proper operation when the case temperature, T_{CASE}, (T_C) is within the specified range of 0°C to 95°C.

6.1 Measuring Thermal Values

To verify that the proper T_C (case temperature) is maintained for the Pentium processor (610\75), it should be measured at the center of the package top surface (encapsulant). To minimize any measurement errors, the following techniques are recommended:

- Use 36 gauge or finer diameter K, T, or J type thermocouples. Intel's laboratory testing was done using a thermocouple made by Omega (part number: 5TC-TTK-36-36).
- Attach the thermocouple bead or junction to the center of the package top surface using highly thermally conductive cements. Intel's laboratory testing was done by using Omega Bond (part number: OB-100).
- The thermocouple should be attached at a 90° angle as shown in Figure 20.

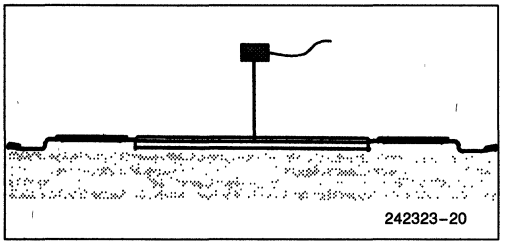


Figure 20. Technique for Measuring Case Temperature (T_C)

6.2 Thermal Equations

For the Pentium processor (610\75), an ambient temperature (T_A) is not specified directly. The only requirement is that the case temperature (T_C) is met. The ambient temperature can be calculated from the following equations:

$$T_J = T_C + P \times \theta_{JC}$$
$$T_A = T_J - P \times \theta_{JA}$$
$$T_A = T_C - (P \times \theta_{CA}$$
$$T_C = T_A + P \times [\theta_{JA} - \theta_{JC}]$$
$$\theta_{CA} = \theta_{JA} - \theta_{JC}$$

where,

T_A and T_C are ambient and case temperatures (°C)
θ_{CA} = Case-to-Ambient thermal resistance (°C/W)
θ_{JA} = Junction-to-Ambient thermal resistance (°C/W)
θ_{JC} = Junction-to-Case thermal resistance (°C/W)
P = maximum power consumption (Watts)

P (maximum power consumption) is specified in section 4.2.

6.3 TCP Thermal Characteristics

The primary heat transfer path from the die of the Tape Carrier Package (TCP) is through the back side of the die and into the PC board. There are two thermal paths traveling from the PC board to the ambient air. One is the spread of heat within the board and the dissipation of heat by the board to the ambient air. The other is the transfer of heat through the board and to the opposite side where thermal enhancements (e.g., heat sinks, pipes) are attached. To prevent the possibility of damaging the TCP component, the thermal enhancements should be attached to the opposite side of the TCP site not directly mounted to the package surface.

6.4 PC Board Enhancements

Copper planes, thermal pads, and vias are design options that can be used to improve heat transfer from the PC board to the ambient air. Tables 21 and 22 present thermal resistance data for copper plane thickness and via effects. It should be noted that although thicker copper planes will reduce the θ_{ca} of a system without any thermal enhancements, they have less effect on the θ_{ca} of a system with thermal enhancements. However, placing vias under the die will reduce the θ_{ca} of a system with and without thermal enhancements.

Table 21. Thermal Resistance vs. Copper Plane Thickness with and without Enhancements

Copper Plane Thickness*	θ_{CA} (°C/W) No Enhancements	θ_{CA} (°C/W) With Heat Pipe
1 oz. Cu	18	8
3 oz. Cu	14	8

NOTES:
*225 vias underneath the die
(1 oz = 1.3 ml)

Table 22. Thermal Resistance vs. Thermal Vias underneath the Die

No. of Vias Under the Die*	θ_{CA} (°C/W) No Enhancements
0	15
144	13

NOTE:
*3 oz. copper planes in test boards

6.4.1 STANDARD TEST BOARD CONFIGURATION

All Tape Carrier Package (TCP) thermal measurements provided in the following tables were taken with the component soldered to a 2" x 2" test board outline. This six-layer board contains 225 vias (underneath the die) in the die attach pad which are connected to two 3 oz. copper planes located at layers two and five. For the Pentium processor (610\75) TCP, the vias in the die attach pad should be connected without thermal reliefs to the ground plane(s). The die is attached to the die attach pad using a thermally and electrically conductive adhesive. This test board was designed to optimize the heat spreading into the board and the heat transfer through to the opposite side of the board.

NOTE:
Thermal resistance values should be used as guidelines only, and are highly system dependent. Final system verification should always refer to the case temperature specification.

Table 23. Pentium™ Processor (610\75) TCP Package Thermal Resistance without Enhancements

	θ_{JC} (°C/W)	θ_{CA} (°C/W)
Thermal Resistance without Enhancements	0.8	13.9

Table 24. Pentium™ Processor (610\75) TCP Package Thermal Resistance with Enhancements (without Airflow)

Thermal Enhancements	θ_{CA} (°C/W)	Notes
Heat sink	11.7	1.2" ×1.2" ×0.35
Al Plate	8.7	4" ×4" ×0.030"
Al Plate with Heat Pipe	7.8	0.3×1" ×4"

Table 25. Pentium™ Processor (610\75) TCP Package Thermal Resistance with Enhancements (with Airflow)

Thermal Enhancements	θ_{CA} (°C/W)	Notes
Heat sink with Fan @ 1.7 CFM	5.0	1.2" ×1.2" ×0.35" HS 1" ×1" ×0.4" Fan
Heat sink with Airflow @ 400 LFM	5.1	1.2" ×1.2" ×0.35" HS
Heat sink with Airflow @ 600 LFM	4.3	1.2" ×1.2" ×0.35" HS

HS = heat sink
LFM = Linear Feet/Minute
CFM = Cubic Feet/Minute

PENTIUM™ PROCESSOR at iCOMP™ INDEX 610\75 MHz
PENTIUM™ PROCESSOR at iCOMP™ INDEX 735\90 MHz
PENTIUM™ PROCESSOR at iCOMP™ INDEX 815\100 MHz

- Compatible with Large Software Base
  - MS-DOS‡, Windows‡, OS/2‡, UNIX‡
- 32-Bit CPU with 64-Bit Data Bus
- Superscalar Architecture
  - Two Pipelined Integer Units Are Capable of 2 Instructions/Clock
  - Pipelined Floating Point Unit
- Separate Code and Data Caches
  - 8K Code, 8K Write Back Data
  - MESI Cache Protocol
- Advanced Design Features
  - Branch Prediction
  - Virtual Mode Extensions
- 3.3V BiCMOS Silicon Technology
- 4M Pages for Increased TLB Hit Rate
- IEEE 1149.1 Boundary Scan
- Dual Processing Configuration
- Multi-Processor Support
  - Multiprocessor Instructions
  - Support for Second Level Cache
- On-Chip Local APIC Controller
  - MP Interrupt Management
  - 8259 Compatible
- Internal Error Detection Features
- Upgradable with a Future Pentium™ OverDrive™ Processor
- Power Management Features
  - System Management Mode
  - Clock Control
- Fractional Bus Operation
  - 100-MHz Core/66-MHz Bus
  - 100-MHz Core/50-MHz Bus
  - 90-MHz Core/60-MHz Bus
  - 75-MHz Core/50-MHz Bus

The Pentium processor (610\75, 735\90, 815\100) extends the Pentium processor family, providing performance needed for mainstream desktop applications as well as for workstations and servers. The Pentium processor is compatible with the entire installed base of applications for DOS, Windows, OS/2, and UNIX. The Pentium processor (610\75, 735\90, 815\100) superscalar architecture can execute two instructions per clock cycle. Branch prediction and separate caches also increase performance. The pipelined floating point unit delivers workstation level performance. Separate code and data caches reduce cache conflicts while remaining software transparent. The Pentium processor (610\75, 735\90, 815\100) has 3.3 million transistors and is built on Intel's advanced 3.3V BiCMOS silicon technology. The Pentium processor (610\75, 735\90, 815\100) has on-chip dual processing support, a local multiprocessor interrupt controller, and SL power management features.

241997-17

‡Other brands and trademarks are the property of their respective owners.

December 1994
Order Number: 241997-003

PENTIUM™ PROCESSOR at iCOMP™ INDEX 610\75 MHz
PENTIUM™ PROCESSOR at iCOMP™ INDEX 735\90 MHz
PENTIUM™ PROCESSOR at iCOMP™ INDEX 815\100 MHz

CONTENTS

CONTENTS

1.0 MICROPROCESSOR ARCHITECTURE OVERVIEW

The Pentium™ processor at iCOMP™ rating 610\75 MHz, iCOMP rating 735\90 MHz, and iCOMP rating 815\100 MHz extends the Intel Pentium family of microprocessors. It is 100% binary compatible with the 8086/88, 80286, Intel386™ DX CPU, Intel386 SX CPU, Intel486™ DX CPU, Intel486 SX CPU, Intel486 DX2 CPUs, and Pentium processor at iCOMP Index 510\60 MHz and iCOMP Index 567\66 MHz.

The Pentium processor family consists of the new Pentium processor at iCOMP rating 610\75 MHz, iCOMP rating 735\90 MHz, and iCOMP rating 815\100 MHz (product order code 80502), described in this document, and the original Pentium processor (510\60, 567\66) (order code 80501). The name "Pentium processor (610\75, 735\90, 815\100)" will be used in this document to refer to the Pentium processor at iCOMP rating 610\75 MHz, iCOMP rating 735\90 MHz and iCOMP rating 815\100 MHz. Also, the name "Pentium processor (510\60, 567\66)" will be used to refer to the original 60- & 66-MHz version product.

The Pentium processor family architecture contains all of the features of the Intel486 CPU family, and provides significant enhancements and additions including the following:

- Superscalar Architecture
- Dynamic Branch Prediction
- Pipelined Floating-Point Unit
- Improved Instruction Execution Time
- Separate 8K Code and 8K Data Caches
- Writeback MESI Protocol in the Data Cache
- 64-Bit Data Bus
- Bus Cycle Pipelining
- Address Parity
- Internal Parity Checking
- Functional Redundancy Checking
- Execution Tracing
- Performance Monitoring
- IEEE 1149.1 Boundary Scan
- System Management Mode
- Virtual Mode Extensions

In addition to the features listed above, the Pentium processor (610\75, 735\90, 815\100) offers the following enhancements over the Pentium processor (510\60, 567\66):

- iCOMP performance rating of 815 at 100 MHz in single processor configuration
- iCOMP performance rating of 735 at 90 MHz in single processor configuration
- iCOMP performance rating of 610 at 75 MHz in single processor configuration
- Dual processing support
- SL power management features
- Upgradable with a Future Pentium OverDrive processor
- Fractional bus operation
- On-chip local APIC device

1.1 Pentium Processor Family Architecture

The application instruction set of the Pentium processor family includes the complete Intel486 CPU family instruction set with extensions to accommodate some of the additional functionality of the Pentium processors. All application software written for the Intel386 and Intel486 family microprocessors will run on the Pentium processors without modification. The on-chip memory management unit (MMU) is completely compatible with the Intel386 family and Intel486 family of CPUs.

The Pentium processors implement several enhancements to increase performance. The two instruction pipelines and floating-point unit on Pentium processors are capable of independent operation. Each pipeline issues frequently used instructions in a single clock. Together, the dual pipes can issue two integer instructions in one clock, or one floating point instruction (under certain circumstances, two floating-point instructions) in one clock.

Branch prediction is implemented in the Pentium processors. To support this, Pentium processors implement two prefetch buffers, one to prefetch code in a linear fashion, and one that prefetches code according to the BTB so the needed code is almost always prefetched before it is needed for execution.

The floating-point unit has been completely redesigned over the Intel486 CPU. Faster algorithms provide up to 10X speed-up for common operations including add, multiply, and load.

Pentium processors include separate code and data caches integrated on-chip to meet performance goals. Each cache is 8 Kbytes in size, with a 32-byte line size and is 2-way set associative. Each cache has a dedicated Translation Lookaside Buffer (TLB) to translate linear addresses to physical addresses. The data cache is configurable to be write back or write through on a line-by-line basis and follows the MESI protocol. The data cache tags are triple ported to support two data transfers and an inquire cycle in the same clock. The code cache is an inherently write-protected cache. The code cache tags are also triple ported to support snooping and split line accesses. Individual pages can be configured as cacheable or non-cacheable by software or hardware. The caches can be enabled or disabled by software or hardware.

The Pentium processors have increased the data bus to 64 bits to improve the data transfer rate. Burst read and burst write back cycles are supported by the Pentium processors. In addition, bus cycle pipelining has been added to allow two bus cycles to be in progress simultaneously. The Pentium processors' Memory Management Unit contains optional extensions to the architecture which allow 2-Mbyte and 4-Mbyte page sizes.

The Pentium processors have added significant data integrity and error detection capability. Data parity checking is still supported on a byte-by-byte basis. Address parity checking, and internal parity checking features have been added along with a new exception, the machine check exception. In addition, the Pentium processors have implemented functional redundancy checking to provide maximum error detection of the processor and the interface to the processor. When functional redundancy checking is used, a second processor, the "checker" is used to execute in lock step with the "master" processor. The checker samples the master's outputs and compares those values with the values it computes internally, and asserts an error signal if a mismatch occurs.

As more and more functions are integrated on chip, the complexity of board level testing is increased. To address this, the Pentium processors have increased test and debug capability. The Pentium processors implement IEEE Boundary Scan (Standard 1149.1). In addition, the Pentium processors have specified 4 breakpoint pins that correspond to each of the debug registers and externally indicate a breakpoint match. Execution tracing provides external indications when an instruction has completed execution in either of the two internal pipelines, or when a branch has been taken.

System Management Mode (SMM) has been implemented along with some extensions to the SMM architecture. Enhancements to the virtual 8086 mode have been made to increase performance by reducing the number of times it is necessary to trap to a virtual 8086 monitor.

Figure 1 shows a block diagram of the Pentium processor (610\75, 735\90, 815\100).

For Pentium Processor (610\75) designs which use the TCP package, Intel document 242323-001 must be referenced for correct TCP pinout, mechanical, thermal, and AC specifications.

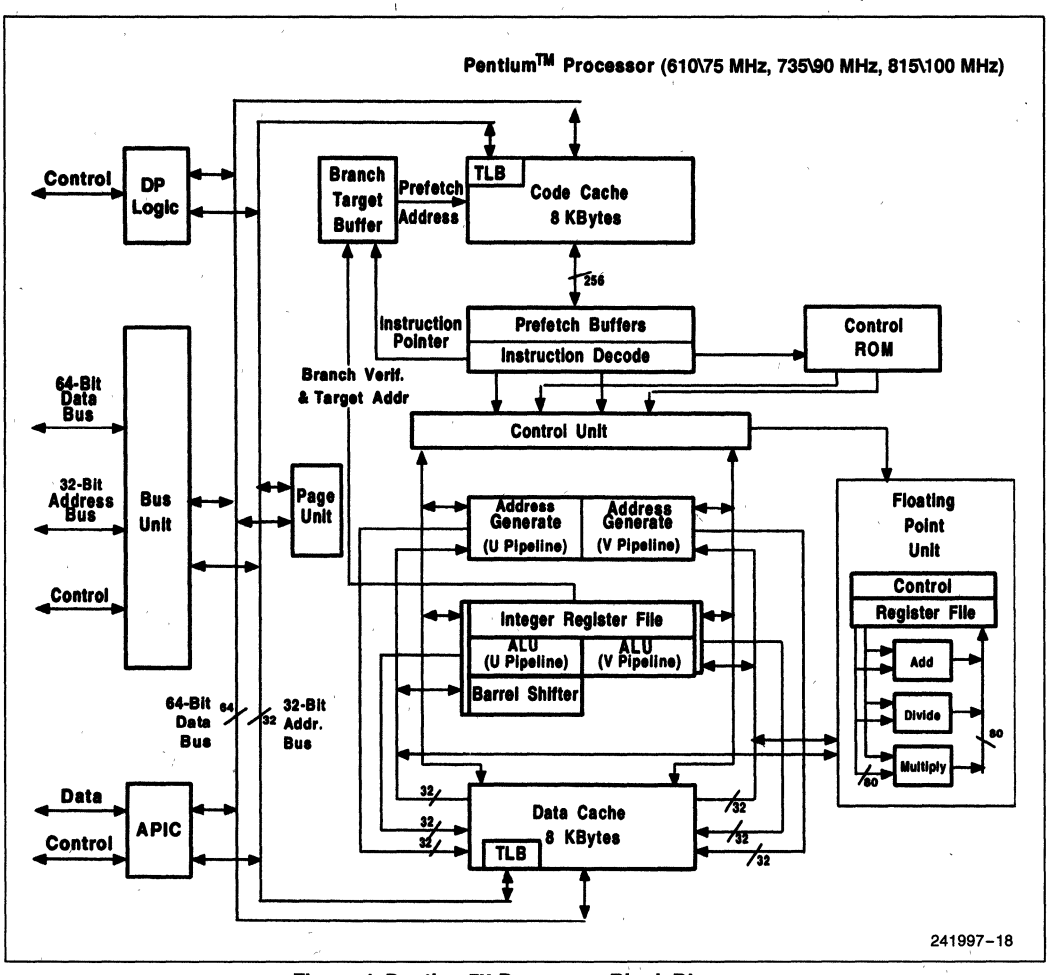


Figure 1. Pentium™ Processor Block Diagram

The block diagram shows the two instruction pipelines, the "u" pipe and the "v" pipe. The u-pipe can execute all integer and floating point instructions. The v-pipe can execute simple integer instructions and the FXCH floating-point instructions.

The separate caches are shown, the code cache and data cache. The data cache has two ports, one for each of the two pipes (the tags are triple ported to allow simultaneous inquire cycles). The data cache has a dedicated Translation Lookaside Buffer (TLB) to translate linear addresses to the physical addresses used by the data cache.

The code cache, branch target buffer and prefetch buffers are responsible for getting raw instructions into the execution units of the Pentium processor. Instructions are fetched from the code cache or from the external bus. Branch addresses are remembered by the branch target buffer. The code cache TLB translates linear addresses to physical addresses used by the code cache.

The decode unit decodes the prefetched instructions so the Pentium processors can execute the instruction. The control ROM contains the microcode which controls the sequence of operations that

must be performed to implement the Pentium processor architecture. The control ROM unit has direct control over both pipelines.

The Pentium processors contain a pipelined floating-point unit that provides a significant floating-point performance advantage over previous generations of processors.

The architectural features introduced in this chapter are more fully described in the *Pentium™ Processor User's Manual.*

1.2 Pentium™ Processor (610\75, 735\90, 815\100)

In addition to the architecture described above for the Pentium processor family, the Pentium processor (610\75, 735\90, 815\100) has additional features which are described in this section.

The Pentium processor (610\75, 735\90, 815\100) offers higher performance and higher operating frequencies than the Pentium processor (510\60, 567\66). The 100-MHz version of the Pentium processor (610\75, 735\90, 815\100) offers core operation at 100 MHz, external bus interface at 66 MHz, and achieves an iCOMP index of 815, while the 90-MHz version offers core operation at 90 MHz, external bus interface at 60 MHz, and achieves an iCOMP index of 735, and the Pentium processor (610\75, 735\90, 815\100) core operates at 75 MHz and the external bus operates at 50 MHz.

Symmetric dual processing in a system is supported with two Pentium processors (610\75, 735\90, 815\100). The two processors appear to the system as a single Pentium processor (610\75, 735\90, 815\100). Operating systems with dual processing support properly schedule computing tasks between the two processors. This scheduling of tasks is transparent to software applications and the end-user. Logic built into the processors support a "glueless" interface for easy system design. Through a private bus, the two Pentium processors (610\75, 735\90, 815\100) arbitrate for the external bus and maintain cache coherency. **Dual processing is supported in a system only if both processors are operating at identical core and bus frequencies. Within these restrictions, two processors of different steppings may operate together in a system.**

In this document, in order to distinguish between two Pentium processors (610\75, 735\90, 815\100) in dual processing mode, one CPU will be designated as the Primary processor with the other being the Dual processor. Note that this is a different concept than that of "master" and "checker" processors described above in the discussion on functional redundancy.

Due to the advanced 3.3V BiCMOS process that it is produced on, the Pentium processor (610\75, 735\90, 815\100) dissipates less power than the Pentium processor (510\60, 567\66). In addition to the SMM features described above, the Pentium processor (610\75, 735\90, 815\100) supports clock control. When the clock to the Pentium processor (610\75, 735\90, 815\100) is stopped, power dissipation is virtually eliminated. The combination of these improvements makes the Pentium processor (610\75, 735\90, 815\100) a good choice for energy-efficient desktop designs.

Supporting an upgrade socket (Socket 5) in the system will provide end-user upgradability by the addition of a Future Pentium OverDrive processor. Typical applications will realize a 40%–70% performance increase by addition of a Future Pentium OverDrive processor.

The Pentium processor (610\75, 735\90, 815\100) supports fractional bus operation. This allows the internal processor core to operate at high frequencies, while communicating with the external bus at lower frequencies. The external bus frequency operates at a selectable one-half or two-thirds fraction of the internal core frequency.

The Pentium processor (610\75, 735\90, 815\100) contains an on-chip Advanced Programmable Interrupt Controller (APIC). This APIC implementation supports multiprocessor interrupt management (with symmetric interrupt distribution across all processors), multiple I/O subsystem support, 8259A compatibility, and inter-processor interrupt support.

2.0 PINOUT

2.1 Pinout and Pin Descriptions

2.1.1 PENTIUM™ PROCESSOR (610\75, 735\90, 815\100) PINOUT

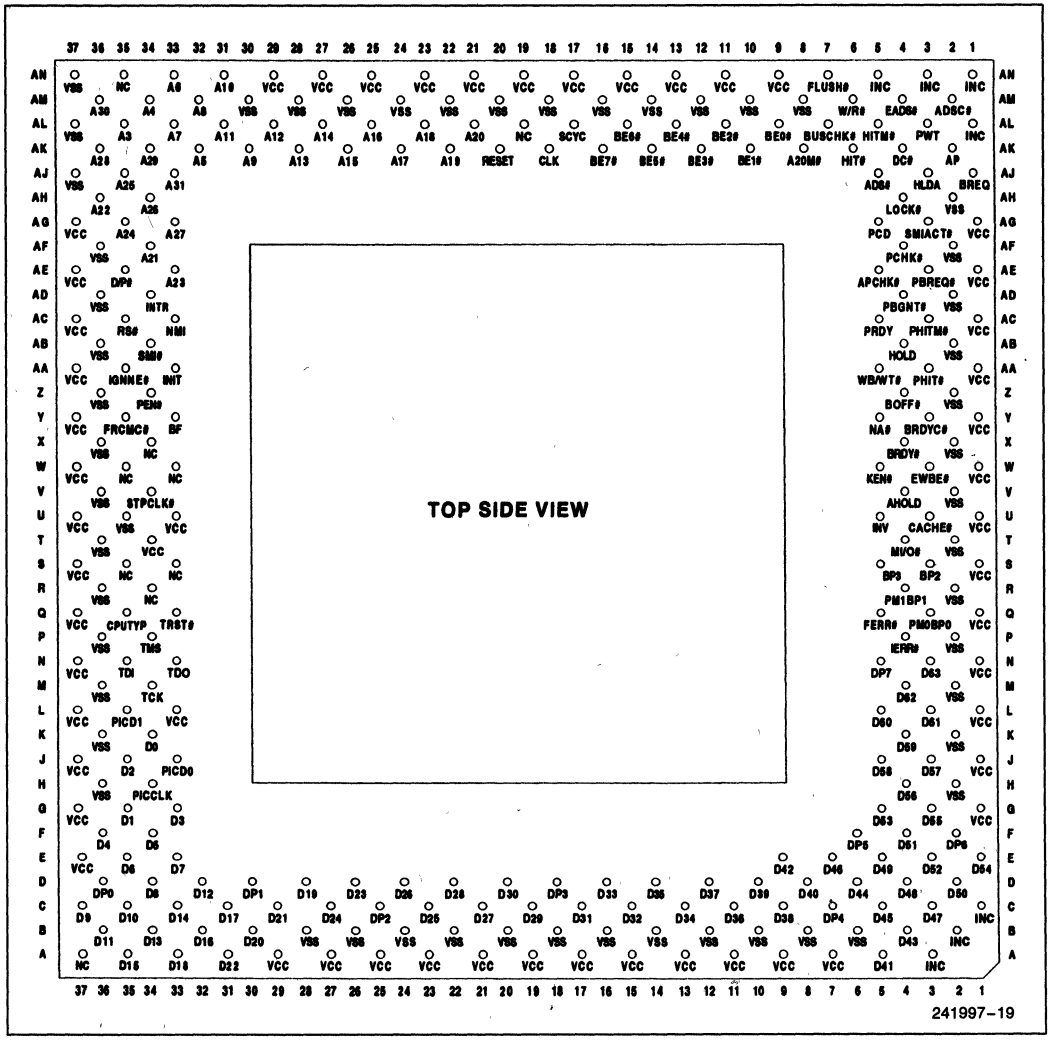


Figure 2. Pentium™ Processor (610\75, 735\90, 815\100) Pinout (Top Side View)

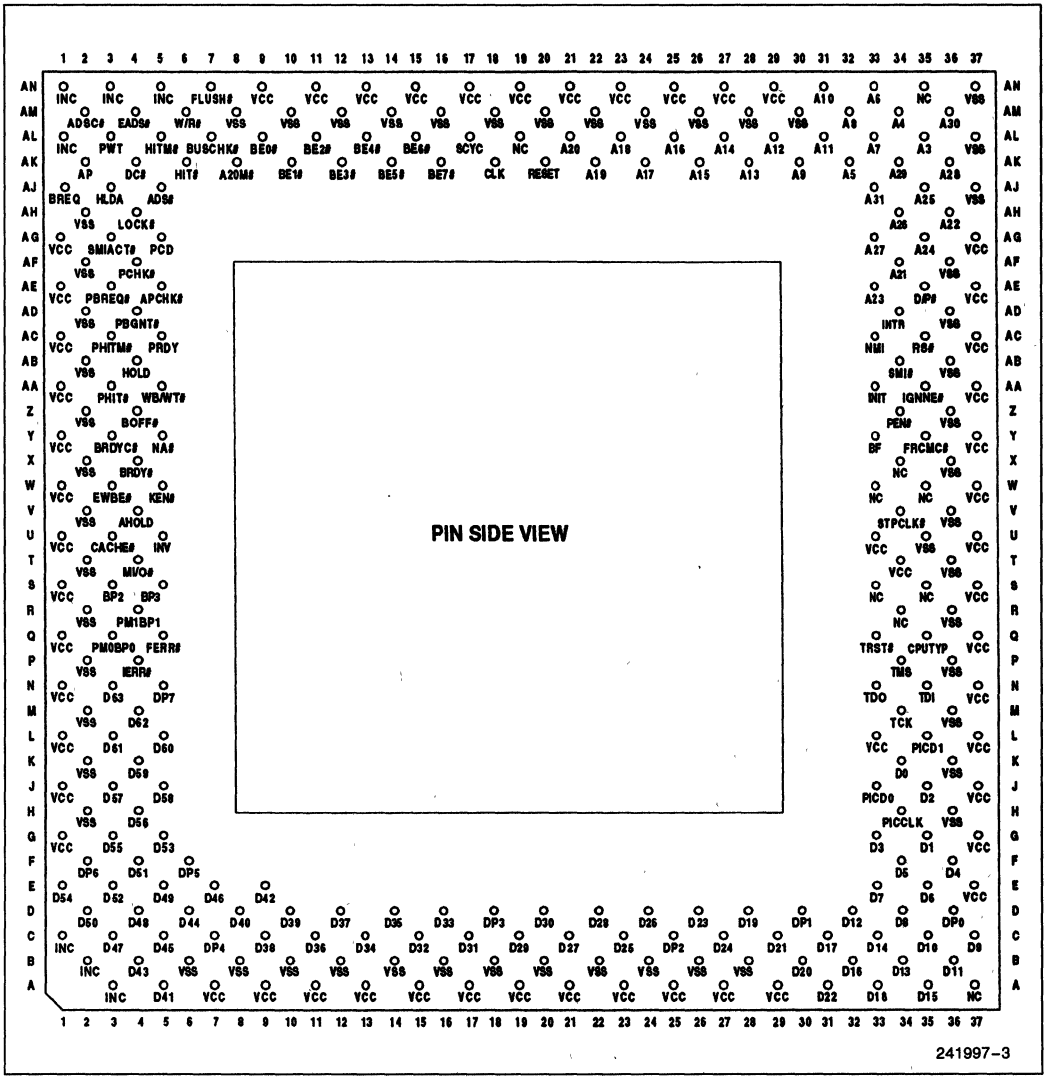


Figure 3. Pentium™ Processor (610\75, 735\90, 815\100) (Pin Side View)

2.1.2 PIN CROSS REFERENCE TABLE FOR PENTIUM™ PROCESSOR (610\75, 735\90, 815\100)

Table 1. Pin Cross Reference by Pin Name

Address									
A3	AL35	A9	AK30	A15	AK26	A21	AF34	A27	AG33
A4	AM34	A10	AN31	A16	AL25	A22	AH36	A28	AK36
A5	AK32	A11	AL31	A17	AK24	A23	AE33	A29	AK34
A6	AN33	A12	AL29	A18	AL23	A24	AG35	A30	AM36
A7	AL33	A13	AK28	A19	AK22	A25	AJ35	A31	AJ33
A8	AM32	A14	AL27	A20	AL21	A26	AH34		
Data									
D0	K34	D13	B34	D26	D24	D39	D10	D52	E03
D1	G35	D14	C33	D27	C21	D40	D08	D53	G05
D2	J35	D15	A35	D28	D22	D41	A05	D54	E01
D3	G33	D16	B32	D29	C19	D42	E09	D55	G03
D4	F36	D17	C31	D30	D20	D43	B04	D56	H04
D5	F34	D18	A33	D31	C17	D44	D06	D57	J03
D6	E35	D19	D28	D32	C15	D45	C05	D58	J05
D7	E33	D20	B30	D33	D16	D46	E07	D59	K04
D8	D34	D21	C29	D34	C13	D47	C03	D60	L05
D9	C37	D22	A31	D35	D14	D48	D04	D61	L03
D10	C35	D23	D26	D36	C11	D49	E05	D62	M04
D11	B36	D24	C27	D37	D12	D50	D02	D63	N03
D12	D32	D25	C23	D38	C09	D51	F04		

Table 1. Pin Cross Reference by Pin Name (Contd.)

Control							
A20M#	AK08	BRDYC#	Y03	FLUSH#	AN07	PEN#	Z34
ADS#	AJ05	BREQ	AJ01	FRCMC#	Y35	PM0/BP0	Q03
ADSC#	AM02	BUSCHK#	AL07	HIT#	AK06	PM1/BP1	R04
AHOLD	V04	CACHE#	U03	HITM#	AL05	PRDY	AC05
AP	AK02	CPUTYP	Q35	HLDA	AJ03	PWT	AL03
APCHK#	AE05	D/C#	AK04	HOLD	AB04	R/S#	AC35
BE0#	AL09	D/P#	AE35	IERR#	P04	RESET	AK20
BE1#	AK10	DP0	D36	IGNNE#	AA35	SCYC	AL17
BE2#	AL11	DP1	D30	INIT	AA33	SMI#	AB34
BE3#	AK12	DP2	C25	INTR/LINT0	AD34	SMIACT#	AG03
BE4#	AL13	DP3	D18	INV	U05	TCK	M34
BE5#	AK14	DP4	C07	KEN#	W05	TDI	N35
BE6#	AL15	DP5	F06	LOCK#	AH04	TDO	N33
BE7#	AK16	DP6	F02	M/IO#	T04	TMS	P34
BOFF#	Z04	DP7	N05	NA#	Y05	TRST#	Q33
BP2	S03	EADS#	AM04	NMI/LINT1	AC33	W/R#	AM06
BP3	S05	EWBE#	W03	PCD	AG05	WB/WT#	AA05
BRDY#	X04	FERR#	Q05	PCHK#	AF04		

APIC		Clock Control		Dual Processor Private Interface	
PICCLK	H34	CLK	AK18	PBGNT#	AD04
PICD0	J33	[BF]	Y33	PBREQ#	AE03
[DPEN#]		STPCLK#	V34	PHIT#	AA03
PICD1	L35			PHITM#	AC03
[APICEN]					

2

Table 1. Pin Cross Reference by Pin Name (Contd.)

V_{CC}								
A07	A19	E37	L33	S01	W01	AC01	AN09	AN21
A09	A21	G01	L37	S37	W37	AC37	AN11	AN23
A11	A23	G37	N01	T34	Y01	AE01	AN13	AN25
A13	A25	J01	N37	U01	Y37	AE37	AN15	AN27
A15	A27	J37	Q01	U33	AA01	AG01	AN17	AN29
A17	A29	L01	Q37	U37	AA37	AG37	AN19	

V_{SS}						
B06	B22	M02	U35	AB36	AM08	AM24
B08	B24	M36	V02	AD02	AM10	AM26
B10	B26	P02	V36	AD36	AM12	AM28
B12	B28	P36	X02	AF02	AM14	AM30
B14	H02	R02	X36	AF36	AM16	AN37
B16	H36	R36	Z02	AH02	AM18	
B18	K02	T02	Z36	AJ37	AM20	
B20	K36	T36	AB02	AL37	AM22	

NC/INC						
A03	C01	S35	W35	AL01	AN01	AN05
A37	R34	W33	X34	AL19	AN03	AN35
B02	S33					

2.2 Design Notes

For reliable operation, always connect unused inputs to an appropriate signal level. Unused active low inputs should be connected to V_{CC}. Unused active HIGH inputs should be connected to GND.

No Connect (NC) pins must remain unconnected. Connection of NC pins may result in component failure or incompatibility with processor steppings.

2.3 Quick Pin Reference

This section gives a brief functional description of each of the pins. For a detailed description, see the "Hardware Interface" chapter in the *Pentium™ Processor User's Manual,* Volume. 1. **Note that all input pins must meet their AC/DC specifications to guarantee proper functional behavior.**

The # symbol at the end of a signal name indicates that the active, or asserted state occurs when the signal is at a low voltage. When a # symbol is not present after the signal name, the signal is active, or asserted at the high voltage level.

The following pins exist on the Pentium processor (510\60, 567\66) but have been removed from the Pentium processor (610\75, 735\90, 815\100):

- IBT, IU, IV, BT0-3

The following pins become I/O pins when two Pentium processors (610\75, 735\90, 815\100) are operating in a dual processing environment:

- ADS#, CACHE#, HIT#, HITM#, HLDA#, LOCK#, M/IO#, D/C#, W/R#, SCYC

Table 2. Quick Pin Reference

Symbol	Type*	Name and Function
A20M#	I	When the **address bit 20 mask** pin is asserted, the Pentium processor (610\75, 735\90, 815\100) emulates the address wraparound at 1 Mbyte which occurs on the 8086. When A20M# is asserted, the Pentium processor (610\75, 735\90, 815\100) masks physical address bit 20 (A20) before performing a lookup to the internal caches or driving a memory cycle on the bus. The effect of A20M# is undefined in protected mode. A20M# must be asserted only when the processor is in real mode. A20M# is internally masked by the Pentium processor (610\75, 735\90, 815\100) when configured as a Dual processor.
A31-A3	I/O	As outputs, the **address** lines of the processor along with the byte enables define the physical area of memory or I/O accessed. The external system drives the inquire address to the processor on A31-A5.
ADS#	O	The **address status** indicates that a new valid bus cycle is currently being driven by the Pentium processor (610\75, 735\90, 815\100).
ADSC#	O	ADSC# is functionally identical to ADS#.
AHOLD	I	In response to the assertion of **address hold,** the Pentium processor (610\75, 735\90, 815\100) will stop driving the address lines (A31-A3), and AP in the next clock. The rest of the bus will remain active so data can be returned or driven for previously issued bus cycles.
AP	I/O	**Address parity** is driven by the Pentium processor (610\75, 735\90, 815\100) with even parity information on all Pentium processor (610\75, 735\90, 815\100) generated cycles in the same clock that the address is driven. Even parity must be driven back to the Pentium processor (610\75, 735\90, 815\100) during inquire cycles on this pin in the same clock as EADS# to ensure that correct parity check status is indicated by the Pentium processor (610\75, 735\90, 815\100).
APCHK#	O	The **address parity check** status pin is asserted two clocks after EADS# is sampled active if the Pentium processor (610\75, 735\90, 815\100) has detected a parity error on the address bus during inquire cycles. APCHK# will remain active for one clock each time a parity error is detected (including during dual processing private snooping).
[APICEN] PICD1	I	**Advanced Programmable Interrupt Controller Enable** is a new pin that enables or disables the on-chip APIC interrupt controller. If sampled high at the falling edge of RESET, the APIC is enabled. APICEN shares a pin with the **Programmable Interrupt Controller Data** 1 signal.
BE7#-BE5# BE4#-BE0#	O I/O	The **byte enable** pins are used to determine which bytes must be written to external memory, or which bytes were requested by the CPU for the current cycle. The byte enables are driven in the same clock as the address lines (A31-3). Unlike the Pentium processor (510\60, 567\66), the lower 4-byte enables (BE3#-BE0#) are used on the Pentium processor (610\75, 735\90, 815\100) as APIC ID inputs and are sampled at RESET. After RESET, these behave exactly like the Pentium processor (510\60, 567\66) byte enables. In dual processing mode, BE4# is used as an input during Flush cycles.

Table 2. Quick Pin Reference (Contd.)

Symbol	Type*	Name and Function
[BF]	I	**Bus Frequency** determines the bus-to-core frequency ratio. BF is sampled at RESET, and cannot be changed until another non-warm (1 ms) assertion of RESET. Additionally, BF must not change values while RESET is active. For proper operation of the Pentium processor (610\75, 735\90, 815\100) this pin should be strapped high or low. When BF is strapped to V_{CC}, the processor will operate at a 2/3 bus/core frequency ratio. When BF is strapped to V_{SS}, the processor will operate at a 1/2 bus/core frequency ratio. If BF is left floating, the Pentium processor (610\75, 735\90, 815\100) defaults to a 2/3 bus ratio. Note that core operation at either 75 MHz or 90 MHz does not allow 1/2 bus/core frequency.
BOFF#	I	The **backoff** input is used to abort all outstanding bus cycles that have not yet completed. In response to BOFF#, the Pentium processor (610\75, 735\90, 815\100) will float all pins normally floated during bus hold in the next clock. The processor remains in bus hold until BOFF# is negated, at which time the Pentium processor (610\75, 735\90, 815\100) restarts the aborted bus cycle(s) in their entirety.
BP[3:2] PM/BP[1:0]	O	The **breakpoint** pins (BP3-0) correspond to the debug registers, DR3-DR0. These pins externally indicate a breakpoint match when the debug registers are programmed to test for breakpoint matches. BP1 and BP0 are multiplexed with the **performance monitoring** pins (PM1 and PM0). The PB1 and PB0 bits in the Debug Mode Control Register determine if the pins are configured as breakpoint or performance monitoring pins. The pins come out of RESET configured for performance monitoring.
BRDY#	I	The **burst ready** input indicates that the external system has presented valid data on the data pins in response to a read or that the external system has accepted the Pentium processor (610\75, 735\90, 815\100) data in response to a write request. This signal is sampled in the T2, T12 and T2P bus states.
BRDYC#	I	This signal has the same functionality as BRDY#.
BREQ	O	The **bus request** output indicates to the external system that the Pentium processor (610\75, 735\90, 815\100) has internally generated a bus request. This signal is always driven whether or not the Pentium processor (610\75, 735\90, 815\100) is driving its bus.
BUSCHK#	I	The **bus check** input allows the system to signal an unsuccessful completion of a bus cycle. If this pin is sampled active, the Pentium processor (610\75, 735\90, 815\100) will latch the address and control signals in the machine check registers. If, in addition, the MCE bit in CR4 is set, the Pentium processor (610\75, 735\90, 815\100) will vector to the machine check exception.
CACHE#	O	For Pentium processor (610\75, 735\90, 815\100)-initiated cycles the **cache** pin indicates internal cacheability of the cycle (if a read), and indicates a burst write back cycle (if a write). If this pin is driven inactive during a read cycle, the Pentium processor (610\75, 735\90, 815\100) will not cache the returned data, regardless of the state of the KEN# pin. This pin is also used to determine the cycle length (number of transfers in the cycle).

Table 2. Quick Pin Reference (Contd.)

Symbol	Type*	Name and Function
CLK	I	The **clock** input provides the fundamental timing for the Pentium processor (610\75, 735\90, 815\100). Its frequency is the operating frequency of the Pentium processor (610\75, 735\90, 815\100) external bus, and requires TTL levels. All external timing parameters except TDI, TDO, TMS, TRST#, and PICD0-1 are specified with respect to the rising edge of CLK. **NOTE:** **It is recommended that CLK begin toggling within 150 ms after V_{CC} reaches its proper operating level. This recommendation is only to ensure long-term reliability of the device.**
CPUTYP	I	**CPU type** distinguishes the Primary processor from the Dual processor. In a single processor environment, or when the Pentium processor (610\75, 735\90, 815\100) is acting as the Primary processor in a dual processing system, CPUTYP should be strapped to V_{SS}. The Dual processor should have CPUTYP strapped to V_{CC}. For the Future Pentium OverDrive processor, CPUTYP will be used to determine whether the bootup handshake protocol will be used (in a dual socket system) or not (in a single socket system).
D/C#	O	The **data/code** output is one of the primary bus cycle definition pins. It is driven valid in the same clock as the ADS# signal is asserted. D/C# distinguishes between data and code or special cycles.
D/P#	O	The **dual/primary** processor indication. The Primary processor drives this pin low when it is driving the bus, otherwise it drives this pin high. D/P# is always driven. D/P# can be sampled for the current cycle with ADS# (like a status pin). This pin is defined only on the Primary processor.
D63-D0	I/O	These are the 64 **data lines** for the processor. Lines D7-D0 define the least significant byte of the data bus; lines D63-D56 define the most significant byte of the data bus. When the CPU is driving the data lines, they are driven during the T2, T12, or T2P clocks for that cycle. During reads, the CPU samples the data bus when BRDY# is returned.
DP7-DP0	I/O	These are the **data parity** pins for the processor. There is one for each byte of the data bus. They are driven by the Pentium processor (610\75, 735\90, 815\100) with even parity information on writes in the same clock as write data. Even parity information must be driven back to the Pentium processor (610\75, 735\90, 815\100) on these pins in the same clock as the data to ensure that the correct parity check status is indicated by the Pentium processor (610\75, 735\90, 815\100). DP7 applies to D63-56, DP0 applies to D7-0.
[DPEN#] PICD0	I/O	**Dual processing enable** is an output of the Dual processor and an input of the Primary processor. The Dual processor drives DPEN# low to the Primary processor at RESET to indicate that the Primary processor should enable dual processor mode. DPEN# may be sampled by the system at the falling edge of RESET to determine if Socket 5 is occupied. DPEN# shares a pin with PICD0.
EADS#	I	This signal indicates that a valid **external address** has been driven onto the Pentium processor (610\75, 735\90, 815\100) address pins to be used for an inquire cycle.
EWBE#	I	The **external write buffer empty** input, when inactive (high), indicates that a write cycle is pending in the external system. When the Pentium processor (610\75, 735\90, 815\100) generates a write, and EWBE# is sampled inactive, the Pentium processor (610\75, 735\90, 815\100) will hold off all subsequent writes to all E- or M-state lines in the data cache until all write cycles have completed, as indicated by EWBE# being active.

Table 2. Quick Pin Reference (Contd.)

Symbol	Type*	Name and Function
FERR#	O	The **floating point error** pin is driven active when an unmasked floating point error occurs. FERR# is similar to the ERROR# pin on the Intel387™ math coprocessor. FERR# is included for compatibility with systems using DOS type floating point error reporting. FERR# is never driven active by the Dual processor.
FLUSH#	I	When asserted, the **cache flush** input forces the Pentium processor (610\75, 735\90, 815\100) to write back all modified lines in the data cache and invalidate its internal caches. A Flush Acknowledge special cycle will be generated by the Pentium processor (610\75, 735\90, 815\100) indicating completion of the write back and invalidation. If FLUSH# is sampled low when RESET transitions from high to low, tristate test mode is entered. If two Pentium processors (610\75, 735\90, 815\100) are operating in dual processing mode in a system and FLUSH# is asserted, the Dual processor will perform a flush first (without a flush acknowledge cycle), then the Primary processor will perform a flush followed by a flush acknowledge cycle.
FRCMC#	I	The **functional redundancy checking master/checker** mode input is used to determine whether the Pentium processor (610\75, 735\90, 815\100) is configured in master mode or checker mode. When configured as a master, the Pentium processor (610\75, 735\90, 815\100) drives its output pins as required by the bus protocol. When configured as a checker, the Pentium processor (610\75, 735\90, 815\100) tristates all outputs (except IERR# and TDO) and samples the output pins. The configuration as a master/checker is set after RESET and may not be changed other than by a subsequent RESET.
HIT#	O	The **hit** indication is driven to reflect the outcome of an inquire cycle. If an inquire cycle hits a valid line in either the Pentium processor (610\75, 735\90, 815\100) data or instruction cache, this pin is asserted two clocks after EADS# is sampled asserted. If the inquire cycle misses the Pentium processor (610\75, 735\90, 815\100) cache, this pin is negated two clocks after EADS#. This pin changes its value only as a result of an inquire cycle and retains its value between the cycles.
HITM#	O	The **hit to a modified line** output is driven to reflect the outcome of an inquire cycle. It is asserted after inquire cycles which resulted in a hit to a modified line in the data cache. It is used to inhibit another bus master from accessing the data until the line is completely written back.
HLDA	O	The **bus hold acknowledge** pin goes active in response to a hold request driven to the processor on the HOLD pin. It indicates that the Pentium processor (610\75, 735\90, 815\100) has floated most of the output pins and relinquished the bus to another local bus master. When leaving bus hold, HLDA will be driven inactive and the Pentium processor (610\75, 735\90, 815\100) will resume driving the bus. If the Pentium processor (610\75, 735\90, 815\100) has a bus cycle pending, it will be driven in the same clock that HLDA is de-asserted.

Table 2. Quick Pin Reference (Contd.)

Symbol	Type*	Name and Function
HOLD	I	In response to the **bus hold request,** the Pentium processor (610\75, 735\90, 815\100) will float most of its output and input/output pins and assert HLDA after completing all outstanding bus cycles. The Pentium processor (610\75, 735\90, 815\100) will maintain its bus in this state until HOLD is de-asserted. HOLD is not recognized during LOCK cycles. The Pentium processor (610\75, 735\90, 815\100) will recognize HOLD during reset.
IERR#	O	The **internal error** pin is used to indicate two types of errors, internal parity errors and functional redundancy errors. If a parity error occurs on a read from an internal array, the Pentium processor (610\75, 735\90, 815\100) will assert the IERR# pin for one clock and then shutdown. If the Pentium processor (610\75, 735\90, 815\100) is configured as a checker and a mismatch occurs between the value sampled on the pins and the corresponding value computed internally, the Pentium processor (610\75, 735\90, 815\100) will assert IERR# two clocks after the mismatched value is returned.
IGNNE#	I	This is the **ignore numeric error** input. This pin has no effect when the NE bit in CR0 is set to 1. When the CR0.NE bit is 0, and the IGNNE# pin is asserted, the Pentium processor (610\75, 735\90, 815\100) will ignore any pending unmasked numeric exception and continue executing floating-point instructions for the entire duration that this pin is asserted. When the CR0.NE bit is 0, IGNNE# is not asserted, a pending unmasked numeric exception exists (SW.ES = 1), and the floating point instruction is one of FINIT, FCLEX, FSTENV, FSAVE, FSTSW, FSTCW, FENI, FDISI, or FSETPM, the Pentium processor (610\75, 735\90, 815\100) will execute the instruction in spite of the pending exception. When the CR0.NE bit is 0, IGNNE# is not asserted, a pending unmasked numeric exception exists (SW.ES = 1), and the floating-point instruction is one other than FINIT, FCLEX, FSTENV, FSAVE, FSTSW, FSTCW, FENI, FDISI, or FSETPM, the Pentium processor (610\75, 735\90, 815\100) will stop execution and wait for an external interrupt. IGNNE# is internally masked when the Pentium processor (610\75, 735\90, 815\100) is configured as a Dual processor.
INIT	I	The Pentium processor (610\75, 735\90, 815\100) **initialization** input pin forces the Pentium processor (610\75, 735\90, 815\100) to begin execution in a known state. The processor state after INIT is the same as the state after RESET except that the internal caches, write buffers, and floating point registers retain the values they had prior to INIT. INIT may NOT be used in lieu of RESET after power-up. If INIT is sampled high when RESET transitions from high to low, the Pentium processor (610\75, 735\90, 815\100) will perform built-in self test prior to the start of program execution.
INTR/LINT0	I	An active **maskable interrupt** input indicates that an external interrupt has been generated. If the IF bit in the EFLAGS register is set, the Pentium processor (610\75, 735\90, 815\100) will generate two locked interrupt acknowledge bus cycles and vector to an interrupt handler after the current instruction execution is completed. INTR must remain active until the first interrupt acknowledge cycle is generated to assure that the interrupt is recognized. If the local APIC is enabled, this pin becomes **local interrupt 0.**

Table 2. Quick Pin Reference (Contd.)

Symbol	Type*	Name and Function
INV	I	The **invalidation** input determines the final cache line state (S or I) in case of an inquire cycle hit. It is sampled together with the address for the inquire cycle in the clock EADS# is sampled active.
KEN#	I	The **cache enable** pin is used to determine whether the current cycle is cacheable or not and is consequently used to determine cycle length. Whe n the Pentium processor (610\75, 735\90, 815\100) generates a cycle that can be cached (CACHE# asserted) and KEN# is active, the cycle will be transformed into a burst line fill cycle.
LINT0/INTR	I	If the APIC is enabled, this pin is **local interrupt 0.** If the APIC is disabled, this pin is **interrupt.**
LINT1/NMI	I	If the APIC is enabled, this pin is **local interrupt 1.** If the APIC is disabled, this pin is **non-maskable interrupt.**
LOCK#	O	The **bus lock** pin indicates that the current bus cycle is locked. The Pentium processor (610\75, 735\90, 815\100) will not allow a bus hold when LOCK# is asserted (but AHOLD and BOFF# are allowed). LOCK# goes active in the first clock of the first locked bus cycle and goes inactive after the BRDY# is returned for the last locked bus cycle. LOCK# is guaranteed to be de-asserted for at least one clock between back-to-back locked cycles.
M/IO#	O	The **memory/input-output** is one o f the primary bus cycle definition pins. It is driven valid in the same clock as the ADS# signal is asserted. M/IO# distinguishes between memory and I/O cycles.
NA#	I	An active **next address** input indicates that the external memory system is ready to accept a new bus cycle although all data transfers for the current cycle have not yet completed. The Pentium processor (610\75, 735\90, 815\100) will issue ADS# for a pending cycle two clocks after NA# is asserted. The Pen tium processor (610\75, 735\90, 815\100) supports up to 2 outstanding bus cycles.
NMI/LINT1	I	The **non-maskable interrupt** request signal indicates that an external non-maskable interrupt has been generated. If the local APIC is enabled, this pin becomes **local interrupt 1.**
PBGNT#	I/O	**Private bus grant** is the grant line that is used when two Pentium processors (610\75, 735\90, 815\100) are configured in dual processing mode, in order to perform private bus arbitration. PBGNT# should be left unconnected if only one Pentium processor (610\75, 735\90, 815\100) exists in a system.
PBREQ#	I/O	**Private bus request** is the request line that is used when two Pentium processors (610\75, 735\90, 815\100) are configured in dual processing mode, in order to perform private bus arbitration. PBREQ# should be left unconnected if only one Pentium processor (610\75, 735\90, 815\100) exists in a system.
PCD	O	The **page cache disable** pin reflects the state of the PCD bit in CR3, the Page Directory Entry, or the Page Table Entry. The purpose of PCD is to provide an external cacheability indication on a page by page basis.

Table 2. Quick Pin Reference (Contd.)

Symbol	Type*	Name and Function
PCHK#	O	The **parity check** output indicates the result of a parity check on a data read. It is driven with parity status two clocks after BRDY# is returned. PCHK# remains low one clock for each clock in which a parity error was detected. Parity is checked only for the bytes on which valid data is returned. When two Pentium processors (610\75, 735\90, 815\100) are operating in dual processing mode, PCHK# may be driven two or three clocks after BRDY# is returned.
PEN#	I	The **parity enable** input (along with CR4.MCE) determines whether a machine check exception will be taken as a result of a data parity error on a read cycle. If this pin is sampled active in the clock a data parity error is detected, the Pentium processor (610\75, 735\90, 815\100) will latch the address and control signals of the cycle with the parity error in the machine check registers. If, in addition, the machine check enable bit in CR4 is set to "1", the Pentium processor (610\75, 735\90, 815\100) will vector to the machine check exception before the beginning of the next instruction.
PHIT#	I/O	**Private hit** is a hit indication used when two Pentium processors (610\75, 735\90, 815\100) are configured in dual processing mode, in order to maintain local cache coherency. PHIT# should be left unconnected if only one Pentium processor (610\75, 735\90, 815\100) exists in a system.
PHITM#	I/O	**Private modified hit** is a hit indication used when two Pentium processors (610\75, 735\90, 815\100) are configured in dual processing mode, in order to maintain local cache coherency. PHITM# should be left unconnected if only one Pentium processor (610\75, 735\90, 815\100) exists in a system.
PICCLK	I	The APIC interrupt controller serial data bus clock is driven into the **programmable interrupt controller clock** input of the Pentium processor (610\75, 735\90, 815\100).
PICD0-1 [DPEN#] [APICEN]	I/O	**Programmable interrupt controller data lines 0-1** of the Pentium processor (610\75, 735\90, 815\100) comprise the data portion of the APIC 3-wire bus. They are open-drain outputs that require external pull-up resistors. These signals share pins with DPEN# and APICEN.
PM/BP[1:0]	O	These pins function as part of the performance monitoring feature. The breakpoint 1-0 pins are multiplexed with the **performance monitoring 1-0** pins. The PB1 and PB0 bits in the Debug Mode Control Register determine if the pins are configured as breakpoint or performance monitoring pins. The pins come out of RESET configured for performance monitoring.
PRDY	O	The **probe ready** output pin indicates that the processor has stopped normal execution in response to the R/S# pin going active, or Probe Mode being entered.
PWT	O	The **page write through** pin reflects the state of the PWT bit in CR3, the page directory entry, or the page table entry. The PWT pin is used to provide an external write back indication on a page-by-page basis.

Table 2. Quick Pin Reference (Contd.)

Symbol	Type*	Name and Function
R/S#	I	The **run/stop** input is an asynchronous, edge-sensitive interrupt used to stop the normal execution of the processor and place it into an idle state. A high to low transition on the R/S# pin will interrupt the processor and cause it to stop execution at the next instruction boundary.
RESET	I	**RESET** forces the Pentium processor (610\75, 735\90, 815\100) to begin execution at a known state. All the Pentium processor (610\75, 735\90, 815\100) internal caches will be invalidated upon the RESET. Modified lines in the data cache are not written back. FLUSH#, FRCMC# and INIT are sampled when RESET transitions from high to low to determine if tristate test mode or checker mode will be entered, or if BIST will be run.
SCYC	O	The **split cycle** output is asserted during misaligned LOCKed transfers to indicate that more than two cycles will be locked together. This signal is defined for locked cycles only. It is undefined for cycles which are not locked.
SMI#	I	The **system management interrupt** causes a system management interrupt request to be latched internally. When the latched SMI# is recognized on an instruction boundary, the processor enters System Management Mode.
SMIACT#	O	An active **system management interrupt active** output indicates that the processor is operating in System Management Mode.
STPCLK#	I	Assertion of the **stop clock** input signifies a request to stop the internal clock of the Pentium processor (610\75, 735\90, 815\100) thereby causing the core to consume less power. When the CPU recognizes STPCLK#, the processor will stop execution on the next instruction boundary, unless superseded by a higher priority interrupt, and generate a stop grant acknowledge cycle. When STPCLK# is asserted, the Pentium processor (610\75, 735\90, 815\100) will still respond to interprocessor and external snoop requests.
TCK	I	The **testability clock** input provides the clocking function for the Pentium processor (610\75, 735\90, 815\100) boundary scan in accordance with the IEEE Boundary Scan interface (Standard 1149.1). It is used to clock state information and data into and out of the Pentium processor (610\75, 735\90, 815\100) during boundary scan.

Table 2. Quick Pin Reference (Contd.)

Symbol	Type*	Name and Function
TDI	I	The **test data input** is a serial input for the test logic. TAP instructions and data are shifted into the Pentium processor (610\75, 735\90, 815\100) on the TDI pin on the rising edge of TCK when the TAP controller is in an appropriate state.
TDO	O	The **test data output** is a serial output of the test logic. TAP instructions and data are shifted out of the Pentium processor (610\75, 735\90, 815\100) on the TDO pin on TCK's falling edge when the TAP controller is in an appropriate state.
TMS	I	The value of the **test mode select** input signal sampled at the rising edge of TCK controls the sequence of TAP controller state changes.
TRST#	I	When asserted, the **test reset** input allows the TAP controller to be asynchronously initialized.
V_{CC}	I	The Pentium processor (610\75, 735\90, 815\100) has 53 3.3V **power** inputs.
V_{SS}	I	The Pentium processor (610\75, 735\90, 815\100) has 53 **ground** inputs.
W/R#	O	**Write/read** is one of the primary bus cycle definition pins. It is driven valid in the same clock as the ADS# signal is asserted. W/R# distinguishes between write and read cycles.
WB/WT#	I	The **write back/write through** input allows a data cache line to be defined as write back or write through on a line-by-line basis. As a result, it determines whether a cache line is initially in the S or E state in the data cache.

* The pins are classified as Input or Output based on their function in Master Mode. See the Functional Redundancy Checking section in the "Error Detection" chapter of the *Pentium™ Processor User's Manual,* Vol. 1, for further information.

2.4 Pin Reference Tables

Table 3. Output Pins

Name	Active Level	When Floated
ADS#*	Low	Bus Hold, BOFF#
ADSC#	Low	Bus Hold, BOFF#
APCHK#	Low	
BE7#-BE5#	Low	Bus Hold, BOFF#
BREQ	High	
CACHE#*	Low	Bus Hold, BOFF#
D/P#**	n/a	
FERR#**	Low	
HIT#*	Low	
HITM#*	Low	
HLD A*	High	
IERR#	Low	
LOCK#*	Low	Bus Hold, BOFF#
M/IO#*, D/C#*, W/R#*	n/a	Bus Hold, BOFF#
PCHK#	Low	
BP3-2, PM1/BP1, PM0/BP0	High	
PRDY	High	
PWT, PCD	High	Bus Hold, BOFF#
SCYC*	High	Bus Hold, BOFF#
SMIACT#	Low	
TDO	n/a	All states except Shift-DR and Shift-IR

NOTES:
All output and input/output pins are floated during tristate test mode and checker mode (except IERR#).
* These are I/O signals when two Pentium processors (610\75, 735\90, 815\100) are operating in dual processing mode.
** These signals are undefined when the CPU is configured as a Dual Processor.

Table 4. Input Pins

Name	Active Level	Synchronous/ Asynchronous	Internal resistor	Qualified
A20M#*	Low	Asynchronous		
AHOLD	High	Synchronous		
BF	High	Synchronous/RESET	Pullup	
BOFF#	Low	Synchronous		
BRDY#	Low	Synchronous		Bus State T2, T12, T2P
BRDYC#	Low	Synchronous	Pullup	Bus State T2, T12, T2P
BUSCHK#	Low	Synchronous	Pullup	BRDY#
CLK	n/a			
CPUTYP	High	Synchronous/RESET		
EADS#	Low	Synchronous		
EWBE#	Low	Synchronous		BRDY#
FLUSH#	Low	Asynchronous		
FRCMC#	Low	Asynchronous		
HOLD	High	Synchronous		
IGNNE#*	Low	Asynchronous		
INIT	High	Asynchronous		
INTR	High	Asynchronous		
INV	High	Synchronous		EADS#
KEN#	Low	Synchronous		First BRDY#/NA#
NA#	Low	Synchronous		Bus State T2,TD,T2P
NMI	High	Asynchronous		
PEN#	Low	Synchronous		BRDY#
PICCLK	High	Asynchronous	Pullup	
R/S#	n/a	Asynchronous	Pullup	
RESET	High	Asynchronous		
SMI#	Low	Asynchronous	Pullup	
STPCLK#	Low	Asynchronous	Pullup	
TCK	n/a		Pullup	
TDI	n/a	Synchronous/TCK	Pullup	TCK
TMS	n/a	Synchronous/TCK	Pullup	TCK
TRST#	Low	Asynchronous	Pullup	
WB/WT#	n/a	Synchronous		First BRDY#/NA#

* Undefined when the CPU is configured as a Dual processor.

Table 5. Input/Output Pins

Name	Active Level	When Floated	Qualified (when an input)	Internal Resistor
A31-A3	n/a	Address hold, Bus Hold, BOFF#	EADS#	
AP	n/a	Address hold, Bus Hold, BOFF#	EADS#	
BE4#-BE0#	Low	Address hold, Bus Hold, BOFF#	RESET	Pulldown*
D63-D0	n/a	Bus Hold, BOFF#	BRDY#	
DP7-DP0	n/a	Bus Hold, BOFF#	BRDY#	
PICD0[DPEN#]				Pullup
PICD1[APICEN]				Pulldown

NOTES:
All output and input/output pins are floated during tristate test mode (except TDO) and checker mode (except IERR# and TDO).
* BE3#-BE0# have Pulldowns during RESET only.

Table 6. Inter-Processor I/O Pins

Name	Active Level	Internal Resistor
PHIT#	Low	Pullup
PHITM#	Low	Pullup
PBGNT#	Low	Pullup
PBREQ#	Low	Pullup

NOTE:
For proper inter-processor operation, the system cannot load these signals.

2.5 Pin Grouping According to Function

Table 7 organizes the pins with respect to their function.

Table 7. Pin Functional Grouping

Function	Pins
Clock	CLK
Initialization	RESET, INIT
Address Bus	A31-A3, BE7#-BE0#
Address Mask	A20M#
Data Bus	D63-D0
Address Parity	AP, APCHK#
APIC Support	PICCLK, PICD0-1
Data Parity	DP7-DP0, PCHK#, PEN#
Internal Parity Error	IERR#
System Error	BUSCHK#
Bus Cycle Definition	M/IO#, D/C#, W/R#, CACHE#, SCYC, LOCK#
Bus Control	ADS#, ADSC#, BRDY#, BRDYC#, NA#
Page Cacheability	PCD, PWT
Cache Control	KEN#, WB/WT#
Cache Snooping/Consistency	AHOLD, EADS#, HIT#, HITM#, INV
Cache Flush	FLUSH#
Write Ordering	EWBE#
Bus Arbitration	BOFF#, BREQ, HOLD, HLDA
Dual Processing Private Bus Control	PBGNT#, PBREQ#, PHIT#, PHITM#
Interrupts	INTR, NMI
Floating Point Error Reporting	FERR#, IGNNE#
System Management Mode	SMI#, SMIACT#
Functional Redundancy Checking	FRCMC# (IERR#)
TAP Port	TCK, TMS, TDI, TDO, TRST#
Breakpoint/Performance Monitoring	PM0/BP0, PM1/BP1, BP3-2
Power Management	STPCLK#
Miscellaneous Dual Processing	CPUTYP, D/P#
Probe Mode	R/S#, PRDY

3.0 ELECTRICAL SPECIFICATIONS

This section describes the electrical differences between the Pentium processor (510\60, 567\66) and the Pentium processor (610\75, 735\90, 815\100), and the DC and AC specifications.

3.1 Electrical Differences Between Pentium™ Processor (610\75, 735\90, 815\100) and Pentium™ Processor (510\60, 567\66)

Pentium™ Processor (510\60, 567\66) Electrical Characteristic	Difference in Pentium™ Processor (610\75, 735\90, 815\100)
5V Power Supply	3.3V Power Supply*
5V TTL Inputs/Outputs	3.3V Inputs/Outputs
Pentium processor (510\60, 567\66) Buffer Models	Pentium processor (610\75, 735\90, 815\100) Buffer Models

* The upgrade socket specifies two 5V inputs (section 6.0.).

The sections that follow will briefly point out some ways to design with these electrical differences.

3.1.1 3.3V POWER SUPPLY

The Pentium processor (610\75, 735\90, 815\100) has all V_{CC} 3.3V inputs. By connecting all Pentium processor (510\60, 567\66) V_{CC} inputs to a common and dedicated power plane, that plane can be converted to 3.3V for the Pentium processor (610\75, 735\90, 815\100).

The CLK and PICCLK inputs can tolerate a 5V input signal. This allows the Pentium processor (610\75, 735\90, 815\100) to use 5V or 3.3V clock drivers.

3.1.2 3.3V INPUTS AND OUTPUTS

The inputs and outputs of the Pentium processor (610\75, 735\90, 815\100) are 3.3V JEDEC standard levels. Both inputs and outputs are also TTL-compatible, although the inputs cannot tolerate voltage swings above the 3.3V V_{IN} max.

For Pentium processor (610\75, 735\90, 815\100) outputs, if the Pentium processor (510\60, 567\66) system support components use TTL-compatible inputs, they will interface to the Pentium processor (610\75, 735\90, 815\100) without extra logic. This is because the Pentium processor (610\75, 735\90, 815\100) drives according to the 5V TTL specification (but not beyond 3.3V).

For Pentium processor (610\75, 735\90, 815\100) inputs, the voltage must not exceed the 3.3V V_{IH3} maximum specification. System support components can consist of 3.3V devices or open-collector devices. 3.3V support components may interface to the Pentium processor (510\60, 567\66) since they typically meet 5V TTL specifications. In an open-collector configuration, the external resistor may be biased with the CPU V_{CC}; as the CPU's V_{CC} changes from 5V to 3.3V, so does this signal's maximum drive.

The CLK and PICCLK inputs of the Pentium processor (610\75, 735\90, 815\100) are 5V tolerant, so they are electrically identical to the Pentium processor (510\60, 567\66) clock input. This allows a Pentium processor (510\60, 567\66) clock driver to drive the Pentium processor (610\75, 735\90, 815\100).

All pins, other than the CLK and PICCLK inputs, are 3.3V-only. If an 8259A interrupt controller is used, for example, the system must provide level converters between the 8259A and the Pentium processor (610\75, 735\90, 815\100).

3.1.3 3.3V PENTIUM™ PROCESSOR (610\75, 735\90, 815\100) BUFFER MODELS

The structure of the buffer models of the Pentium processor (610\75, 735\90, 815\100) are the same as those of the Pentium processor (510\60, 567\66), but the values of the components change since the Pentium processor (610\75, 735\90, 815\100) buffers are 3.3V buffers on a different process.

Despite this difference, the simulation results of Pentium processor (610\75, 735\90, 815\100) buffers and Pentium processor (510\60, 567\66) buffers look nearly identical. Since the 0pF AC specifications of the Pentium processor (610\75, 735\90, 815\100) are derived from the Pentium processor (510\60, 567\66) specifications, the system should see little difference between the AC behavior of the Pentium processor (610\75, 735\90, 815\100) and the Pentium processor (510\60, 567\66).

To meet specifications, simulate the AC timings with Pentium processor (610\75, 735\90, 815\100) buffer models. Pay special attention to the new signal quality restrictions imposed by 3.3V buffers.

3.2 Absolute Maximum Ratings

The values listed below are stress ratings only. Functional operation at the maximums is not implied or guaranteed. Functional operating conditions are given in the AC and DC specification tables.

Extended exposure to the maximum ratings may affect device reliability. Furthermore, although the Pentium processor (610\75, 735\90, 815\100) contains protective circuitry to resist damage from static electric discharge, always take precautions to avoid high static voltages or electric fields.

Case temperature under bias ..... −65°C to 110°C

Storage temperature ..... −65°C to 150°C

3V Supply voltage
with respect to V_{SS} ..... −0.5V to +4.6V

3V Only Buffer DC Input Voltage
−0.5V to V_{CC} +0.5; not to exceed V_{CC3} max(2)

5V Safe Buffer
DC Input Voltage ..... −0.5V to 6.5V(1,3)

NOTES:

1. Applies to CLK and PICCLK.
2. Applies to all Pentium processor (610\75, 735\90, 815\100) inputs except CLK and PICCLK.
3. See overshoot/undershoot transient spec.

** 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.*

3.3 DC Specifications

Tables 8, 9, and 10 list the DC specifications which apply to the Pentium processor (610\75, 735\90, 815\100). The Pentium processor (610\75, 735\90, 815\100) is a 3.3V part internally. The CLK and PICCLK inputs may be a 3.3V or 5V inputs. Since the 3.3V (5V-safe) input levels defined in Table 9 are the same as the 5V TTL levels, the CLK and PICCLK inputs are compatible with existing 5V clock drivers. The power dissipation specification in Table 11 is provided for design of thermal solutions during operation in a sustained maximum level. This is the worst case power the device would dissipate in a system. This number is used for design of a thermal solution for the device.

2

Table 8. 3.3V DC Specifications

T_{CASE} = 0 to 70°C; V_{CC} = 3.3V +5%

Symbol	Parameter	Min	Max	Unit	Notes
V_{IL3}	Input Low Voltage	−0.3	0.8	V	TTL Level(3)
V_{IH3}	Input High Voltage	2.0	V_{CC}+0.3	V	TTL Level(3)
V_{OL3}	Output Low Voltage		0.4	V	TTL Level(1, 3)
V_{OH3}	Output High Voltage	2.4		V	TTL Level(2, 3)
I_{CC3}	Power Supply Current		3250 2950 2650	mA mA mA	@100 MHz(4) @90 MHz(4) @75 MHz(4)

NOTES:

1. Parameter measured at 4 mA.
2. Parameter measured at 3 mA.
3. 3.3V TTL levels apply to all signals except CLK and PICCLK.
4. This value should be used for power supply design. It was determined using a worst case instruction mix and V_{CC} + 5%. Power supply transient response and decoupling capacitors must be sufficient to handle the instantaneous current changes occurring during transitions from stop clock to full active modes. For more information, refer to section 3.4.3.

Table 9. 3.3V (5V-Safe) DC Specifications

Symbol	Parameter	Min	Max	Unit	Notes
V_{IL5}	Input Low Voltage	−0.3	0.8	V	TTL Level(1)
V_{IH5}	Input High Voltage	2.0	5.55	V	TTL Level(1)

NOTES:
1. Applies to CLK and PICCLK only.

Table 10. Input and Output Characteristics

Symbol	Parameter	Min	Max	Unit	Notes
C_{IN}	Input Capacitance		15	pF	4
C_O	Output Capacitance		20	pF	4
$C_{I/O}$	I/O Capacitance		25	pF	4
C_{CLK}	CLK Input Capacitance		15	pF	4
C_{TIN}	Test Input Capacitance		15	pF	4
C_{TOUT}	Test Output Capacitance		20	pF	4
C_{TCK}	Test Clock Capacitance		15	pF	4
I_{LI}	Input Leakage Current		±15	μA	$0 < V_{IN} < V_{CC3}$(1)
I_{LO}	Output Leakage Current		±15	μA	$0 < V_{IN} < V_{CC3}$(1)
I_{IH}	Input Leakage Current		200	μA	$V_{IN} = 2.4V$(3)
I_{IL}	Input Leakage Current		−400	μA	$V_{IN} = 0.4V$(2)

NOTES:
1. This parameter is for input without pullup or pulldown.
2. This parameter is for input with pullup.
3. This parameter is for input with pulldown.
4. Guaranteed by design.

Table 11. Power Dissipation Requirements for Thermal Solution Design

Parameter	Typical(1)	Max(2)	Unit	Notes
Active Power Dissipation	3.9	10.1	Watts	@100 MHz
	3.5	9.0	Watts	@90 MHz
	3.0	8.0	Watts	@75 MHz
Stop Grant and Auto Halt Powerdown Power Dissipation		1.55	Watts	@100 MHz(3)
		1.40	Watts	@90 MHz(3)
		1.20	Watts	@75 MHz
Stop Clock Power Dissipation	0.02	<0.05	Watts	4

NOTES:
1. This is the typical power dissipation in a system. This value was the average value measured in a system using a typical device at V_{CC} = 3.3V running typical applications. This value is highly dependent upon the specific system configuration.
2. Systems must be designed to thermally dissipate the maximum active power dissipation. It is determined using worst case instruction mix with V_{CC} = 3.3V and also takes into account the thermal time constants of the package.
3. Stop Grant/Auto Halt Powerdown Power Dissipation is determined by asserting the STPCLK# pin or executing the HALT instruction.
4. Stop Clock Power Dissipation is determined by asserting the STPCLK# pin and then removing the external CLK input.

3.4 AC Specifications

The AC specifications of the Pentium processor (610\75, 735\90, 815\100) consist of setup times, hold times, and valid delays at 0 pF.

3.4.1 PRIVATE BUS

When two Pentium processors (610\75, 735\90, 815\100) are operating in dual processor mode, a "private bus" exists to arbitrate for the CPU bus and maintain local cache coherency. The private bus consists of two pinout changes:

1. Five pins are added: PBREQ#, PBGNT#, PHIT#, PHITM#, D/P#.
2. Ten output pins become I/O pins: ADS#, D/C#, W/R#, M/IO#, CACHE#, LOCK#, HIT#, HITM#, HLDA, SCYC.

The new pins are given AC specifications of valid delays at 0 pF, setup times, and hold times. Simulate with these parameters and their respective I/O buffer models to guarantee that proper timings are met.

The AC specification gives input setup and hold times for the ten signals that become I/O pins. These setup and hold times must only be met when a dual processor is present in the system.

3.4.2 POWER AND GROUND

For clean on-chip power distribution, the Pentium processor (610\75, 735\90, 815\100) has 53 V_{CC} (power) and 53 V_{SS} (ground) inputs. Power and ground connections must be made to all external V_{CC} and V_{SS} pins of the Pentium processor (610\75, 735\90, 815\100). On the circuit board all V_{CC} pins must be connected to a 3.3V V_{CC} plane. All V_{SS} pins must be connected to a V_{SS} plane.

3.4.3 DECOUPLING RECOMMENDATIONS

Liberal decoupling capacitance should be placed near the Pentium processor (610\75, 735\90, 815\100). The Pentium processor (610\75, 735\90, 815\100) driving its large address and data buses at high frequencies can cause transient power surges, particularly when driving large capacitive loads.

Low inductance capacitors and interconnects are recommended for best high frequency electrical performance. Inductance can be reduced by shortening circuit board traces between the Pentium processor (610\75, 735\90, 815\100) and decoupling capacitors as much as possible.

These capacitors should be evenly distributed around each component on the 3.3V plane. Capacitor values should be chosen to ensure they eliminate both low and high frequency noise components.

For the Pentium processor (610\75, 735\90, 815\100), the power consumption can transition from a low level of power to a much higher level (or high to low power) very rapidly. A typical example would be entering or exiting the Stop Grant state. Another example would be executing a HALT instruction, causing the Pentium processor (610\75, 735\90, 815\100) to enter the Auto HALT Powerdown state, or transitioning from HALT to the Normal state. All of these examples may cause abrupt changes in the power being consumed by the Pentium processor (610\75, 735\90, 815\100). Note that the Auto HALT Powerdown feature is always enabled even when other power management features are not implemented.

Bulk storage capacitors with a low ESR (Effective Series Resistance) in the 10 to 100 μf range are required to maintain a regulated supply voltage during the interval between the time the current load changes and the point that the regulated power supply output can react to the change in load. In order to reduce the ESR, it may be necessary to place several bulk storage capacitors in parallel.

These capacitors should be placed near the Pentium processor (610\75, 735\90, 815\100) (on the 3.3V plane) to ensure that the supply voltage stays within specified limits during changes in the supply current during operation.

3.4.4 CONNECTION SPECIFICATIONS

All NC and INC pins must remain unconnected.

For reliable operation, always connect unused inputs to an appropriate signal level. Unused active low inputs should be connected to V_{CC}. Unused active high inputs should be connected to ground.

3.4.5 AC TIMING TABLES

3.4.5.1 AC Timing Table for a 50-MHz Bus

The AC specifications given in Tables 12 and 13 consist of output delays, input setup requirements and input hold requirements for a 50-MHz external bus. All AC specifications (with the exception of those for the TAP signals and APIC signals) are relative to the rising edge of the CLK input.

All timings are referenced to 1.5V for both "0" and "1" logic levels unless otherwise specified. Within the sampling window, a synchronous input must be stable for correct Pentium processor (610\75, 735\90, 815\100) operation.

Table 12. Pentium™ Processor 610\75, 815\100 AC Specifications for 50-MHz Bus Operation
$3.135 < V_{CC} < 3.465V$, T_{CASE} = 0 to 70°C, C_L = 0 pF

Symbol	Parameter	Min	Max	Unit	Figure	Notes
	Frequency	25.0	50.0	MHz		Max Core Freq = 100 MHz @1/2
t_{1a}	CLK Period	20.0	40.0	nS	4	
t_{1b}	CLK Period Stability		±250	pS		1, 25
t_2	CLK High Time	4.0		nS	4	@2V,(1)
t_3	CLK Low Time	4.0		nS	4	@0.8V,(1)
t_4	CLK Fall Time	0.15	1.5	nS	4	(2.0V–0.8V),(1,5)
t_5	CLK Rise Time	0.15	1.5	nS	4	(0.8V–2.0V), (1,5)
t_{6a}	ADS#, ADSC#, PWT, PCD, BE0-7#, M/IO#, D/C#, CACHE#, SCYC, W/R# Valid Delay	1.0	7.0	nS	5	

Table 12. Pentium™ Processor 610\75, 815\100 AC Specifications for 50-MHz Bus Operation (Contd.)
$3.135 < V_{CC} < 3.465V$, T_{CASE} = 0 to 70°C, C_L = 0 pF

Symbol	Parameter	Min	Max	Unit	Figure	Notes
t_{6b}	AP Valid Delay	1.0	8.5	nS	5	
t_{6c}	A3-A31, LOCK# Valid Delay	1.1	7.0	nS	5	
t_7	ADS#, ADSC#, AP, A3-A31, PWT, PCD, BE0-7#, M/IO#, D/C#, W/R#, CACHE#, SCYC, LOCK# Float Delay		10.0	nS	6	1
t_8	APCHK#, IERR#, FERR#, PCHK# Valid Delay	1.0	8.3	nS	5	4
t_{9a}	BREQ, HLDA, SMIACT# Valid Delay	1.0	8.0	nS	5	4
t_{10a}	HIT# Valid Delay	1.0	8.0	nS	5	
t_{10b}	HITM# Valid Delay	1.1	6.0	nS	5	
t_{11a}	PM0-1, BP0-3 Valid Delay	1.0	10.0	nS	5	
t_{11b}	PRDY Valid Delay	1.0	8.0	nS	5	
t_{12}	D0-D63, DP0-7 Write Data Valid Delay	1.3	8.5	nS	5	
t_{13}	D0-D63, DP0-3 Write Data Float Delay		10.0	nS	6	1
t_{14}	A5-A31 Setup Time	6.5		nS	7	26
t_{15}	A5-A31 Hold Time	1.0		nS	7	
t_{16a}	INV, AP Setup Time	5.0		nS	7	
t_{16b}	EADS# Setup Time	6.0		nS	7	
t_{17}	EADS#, INV, AP Hold Time	1.0		nS	7	
t_{18a}	KEN# Setup Time	5.0		nS	7	
t_{18b}	NA#, WB/WT# Setup Time	4.5		nS	7	
t_{19}	KEN#, WB/WT#, NA# Hold Time	1.0		nS	7	
t_{20}	BRDY#, BRDYC# Setup Time	5.0		nS	7	
t_{21}	BRDY#, BRDYC# Hold Time	1.0		nS	7	
t_{22}	BOFF# Setup Time	5.5		nS	7	
t_{22a}	AHOLD Setup Time	6.0		nS	7	
t_{23}	AHOLD, BOFF# Hold Time	1.0		nS	7	

Table 12. Pentium™ Processor 610\75, 815\100 AC Specifications for 50-MHz Bus Operation (Contd.)
3.135 < V_{CC} < 3.465V, T_{CASE} = 0 to 70°C, C_L = 0 pF

Symbol	Parameter	Min	Max	Unit	Figure	Notes
t_{24}	BUSCHK#, EWBE#, HOLD, PEN# Setup Time	5.0		nS	7	
t_{25}	BUSCHK#, EWBE#, PEN# Hold Time	1.0		nS	7	
t_{25a}	HOLD Hold Time	1.5		nS	7	
t_{26}	A20M#, INTR, STPCLK# Setup Time	5.0		nS	7	12, 16
t_{27}	A20M#, INTR, STPCLK# Hold Time	1.0		nS	7	13
t_{28}	INIT, FLUSH#, NMI, SMI#, IGNNE# Setup Time	5.0		nS	7	12, 16, 17
t_{29}	INIT, FLUSH#, NMI, SMI#, IGNNE# Hold Time	1.0		nS	7	13
t_{30}	INIT, FLUSH#, NMI, SMI#, IGNNE# Pulse Width, Async	2.0		CLKs	7	15, 17
t_{31}	R/S# Setup Time	5.0		nS	7	12, 16, 17
t_{32}	R/S# Hold Time	1.0		nS	7	13
t_{33}	R/S# Pulse Width, Async.	2.0		CLKs	7	15, 17
t_{34}	D0-D63, DP0-7 Read Data Setup Time	3.8		nS	7	
t_{35}	D0-D63, DP0-7 Read Data Hold Time	2.0		nS	7	
t_{36}	RESET Setup Time	5.0		nS	8	11, 12, 16
t_{37}	RESET Hold Time	1.0		nS	8	11, 13
t_{38}	RESET Pulse Width, V_{CC} & CLK Stable	15		CLKs	8	11, 17
t_{39}	RESET Active After V_{CC} & CLK Stable	1.0		mS	8	Power up
t_{40}	Reset Configuration Signals (INIT, FLUSH#, FRCMC#) Setup Time	5.0		nS	8	12, 16, 17
t_{41}	Reset Configuration Signals (INIT, FLUSH#, FRCMC#) Hold Time	1.0		nS	8	13

Table 12. Pentium™ Processor 610\75, 815\100 AC Specifications for 50-MHz Bus Operation (Contd.)
$3.135 < V_{CC} < 3.465V$, T_{CASE} = 0 to 70°C, C_L = 0 pF

Symbol	Parameter	Min	Max	Unit	Figure	Notes
t_{42a}	Reset Configuration Signals (INIT, FLUSH#, FRCMC#) Setup Time, Async.	2.0		CLKs	8	To RESET falling edge(16)
t_{42b}	Reset Configuration Signals (INIT, FLUSH#, FRCMC#, BRDYC#, BUSCHK#) Hold Time, Async.	2.0		CLKs	8	To RESET falling edge(27)
t_{42c}	Reset Configuration Signals (BRDYC#, BUSCHK#) Setup Time, Async.	3.0		CLKs	8	To RESET falling edge(27)
t_{42d}	Reset Configuration Signal BRDYC# Hold Time, RESET driven synchronously	1.0		nS		To RESET falling edge(1,27)
t_{43a}	BF, CPUTYP Setup Time	1.0		mS	8	To RESET falling edge(22)
t_{43b}	BF, CPUTYP Hold Time	2.0		CLKs	8	To RESET falling edge(22)
t_{43c}	APICEN Setup Time	2.0		CLKs	8	To RESET falling edge
t_{43d}	APICEN Hold Time	2.0		CLKs	8	To RESET falling edge
t_{44}	TCK Frequency		16.0	MHz		
t_{45}	TCK Period	62.5		nS	4	
t_{46}	TCK High Time	25.0		nS	4	@2V(1)
t_{47}	TCK Low Time	25.0		nS	4	@0.8V(1)
t_{48}	TCK Fall Time		5.0	nS	4	(2.0V–0.8V)(1,8,9)
t_{49}	TCK Rise Time		5.0	nS	4	(0.8V–2.0V)(1,8,9)
t_{50}	TRST# Pulse Width	40.0		nS	10	Asynchronous(1)
t_{51}	TDI, TMS Setup Time	5.0		nS	9	7
t_{52}	TDI, TMS Hold Time	13.0		nS	9	7
t_{53}	TDO Valid Delay	3.0	20.0	nS	9	8
t_{54}	TDO Float Delay		25.0	nS	9	1, 8
t_{55}	All Non-Test Outputs Valid Delay	3.0	20.0	nS	9	3, 8, 10

Table 12. Pentium™ Processor 610\75, 815\100 AC Specifications for 50-MHz Bus Operation (Contd.)
$3.135 < V_{CC} < 3.465V$, T_{CASE} = 0 to 70°C, C_L = 0 pF

Symbol	Parameter	Min	Max	Unit	Figure	Notes
t_{56}	All Non-Test Outputs Float Delay		25.0	nS	9	1, 3, 8, 10
t_{57}	All Non-Test Inputs Setup Time	5.0		nS	9	3, 7, 10
t_{58}	All Non-Test Inputs Hold Time	13.0		nS	9	3, 7, 10
APIC AC Specifications						
t_{60a}	PICCLK Frequency	2.0	16.66	MHz		
t_{60b}	PICCLK Period	60.0	500.0	nS	4	
t_{60c}	PICCLK High Time	9.0		nS	4	
t_{60d}	PICCLK Low Time	9.0		nS	4	
t_{60e}	PICCLK Rise Time	1.0	5.0	nS	4	
t_{60f}	PICCLK Fall Time	1.0	5.0	nS	4	
t_{60g}	PICD0-1 Setup Time	3.0		nS	7	To PICCLK
t_{60h}	PICD0-1 Hold Time	2.5		nS	7	To PICCLK
t_{60i}	PICD0-1 Valid Delay (LtoH)	4.0	38.0	nS	5	From PICCLK(28,29)
t_{60j}	PICD0-1 Valid Delay (HtoL)	4.0	22.0	nS	5	From PICCLK(28,29)

Table 13. Pentium™ Processor 610\75, 815\100 Dual Processor Mode AC Specifications for 50 MHz Bus Operation

$3.135 < V_{CC} < 3.465V$, T_{CASE} = 0 to 70°C, C_L = 0 pF

Symbol	Parameter	Min	Max	Unit	Figure	Notes
t_{80a}	PBREQ#, PBGNT# Valid Delay	1.0	5.0	nS	5	18
t_{80b}	PHIT#, PHITM# Flight Time	0	2.0	nS	11	30
t_{81a}	PBREQ#, PBGNT# Setup Time	8.0		nS	7	18
t_{82}	PBREQ#, PBGNT# Hold Time	1.0		nS	7	18, 24
t_{83a}	A5-A31 Setup Time	6.5		nS	7	18, 21, 26
t_{83b}	D/C#, W/R#, CACHE#, LOCK#, SCYC Setup Time	6.0		nS	7	18, 21
t_{83c}	ADS#, M/IO# Setup Time	8.0		nS	7	18, 21
t_{83d}	HIT#, HITM# Setup Time	8.0		nS	7	18, 21
t_{83e}	HLDA Setup Time	6.0		nS	7	18, 21
t_{84}	ADS#, D/C#, W/R#, M/IO#, CACHE#, LOCK#, A5-A31, HLDA, HIT#, HITM#, SCYC Hold Time	1.0		nS	7	18, 21
t_{85}	DPEN# Valid Time		10.0	CLKs		18, 19, 23
t_{86}	DPEN# Hold Time	2.0		CLKs		18, 20, 23
t_{87}	APIC ID (BE0#-BE3#) Setup Time	2.0		CLKs	8	To RESET falling edge(23)
t_{88}	APIC ID (BE0#-BE3#) Hold Time	2.0		CLKs	8	From RESET falling edge(23)
t_{89}	D/P# Valid Delay	1.0	8.0	nS	5	Primary Processor Only

3.4.5.2 AC Timing Tables for a 60-MHz Bus

The AC specifications given in Tables 14 and 15 consist of output delays, input setup requirements and input hold requirements for a 60-MHz external bus. All AC specifications (with the exception of those for the TAP signals and APIC signals) are relative to the rising edge of the CLK input.

All timings are referenced to 1.5V for both "0" and "1" logic levels unless otherwise specified. Within the sampling window, a synchronous input must be stable for correct Pentium processor (610\75, 735\90, 815\100) operation.

Table 14. Pentium™ Processor 735\90 AC Specifications for 60-MHz Bus Operation
$3.135 < V_{CC} < 3.465V$, T_{CASE} = 0 to 70°C, C_L = 0 pF

Symbol	Parameter	Min	Max	Unit	Figure	Notes
	Frequency	30.0	60.0	MHz	4	
t_{1a}	CLK Period	16.67	33.33	nS	4	
t_{1b}	CLK Period Stability		±250	pS	4	Adjacent Clocks,(1,25)
t_2	CLK High Time	4.0		nS	4	@2V(1)
t_3	CLK Low Time	4.0		nS	4	@0.8V(1)
t_4	CLK Fall Time	0.15	1.5	nS	4	(2.0V–0.8V)(1,5)
t_5	CLK Rise Time	0.15	1.5	nS	4	(0.8V–2.0V)(1,5)
t_{6a}	ADS#, ADSC#, PWT, PCD, BE0-7#, M/IO#, D/C#, CACHE#, SCYC, W/R# Valid Delay	1.0	7.0	nS	5	
t_{6b}	AP Valid Delay	1.0	8.5	nS	5	
t_{6c}	A3-A31, LOCK# Valid Delay	1.1	7.0	nS	5	
t_7	ADS#, ADSC#, AP, A3-A31, PWT, PCD, BE0-7#, M/IO#, D/C#, W/R#, CACHE#, SCYC, LOCK# Float Delay		10.0	nS	5	1
t_{8a}	APCHK#, IERR#, FERR# Valid Delay	1.0	8.3	nS	5	4
t_{8b}	PCHK# Valid Delay	1.0	7.0	nS	5	4
t_{9a}	BREQ, HLDA Valid Delay	1.0	8.0	nS	5	4
t_{9b}	SMIACT# Valid Delay	1.0	7.6	nS	5	
t_{10a}	HIT# Valid Delay	1.0	8.0	nS	5	
t_{10b}	HITM# Valid Delay	1.1	6.0	nS	5	
t_{11a}	PM0-1, BP0-3 Valid Delay	1.0	10.0	nS	5	
t_{11b}	PRDY Valid Delay	1.0	8.0	nS	5	
t_{12}	D0-D63, DP0-7 Write Data Valid Delay	1.3	7.5	nS	5	

Table 14. Pentium™ Processor 735\90 AC Specifications for 60-MHz Bus Operation (Contd.)
$3.135 < V_{CC} < 3.465V$, T_{CASE} = 0 to 70°C, C_L = 0 pF

Symbol	Parameter	Min	Max	Unit	Figure	Notes
t_{13}	D0-D63, DP0-3 Write Data Float Delay		10.0	nS	6	1
t_{14}	A5-A31 Setup Time	6.0		nS	7	26
t_{15}	A5-A31 Hold Time	1.0		nS	7	
t_{16a}	INV, AP Setup Time	5.0		nS	7	
t_{16b}	EADS# Setup Time	5.5		nS	7	
t_{17}	EADS#, INV, AP Hold Time	1.0		nS	7	
t_{18a}	KEN# Setup Time	5.0		nS	7	
t_{18b}	NA#, WB/WT# Setup Time	4.5		nS	7	
t_{19}	KEN#, WB/WT#, NA# Hold Time	1.0		nS	7	
t_{20}	BRDY#, BRDYC# Setup Time	5.0		nS	7	
t_{21}	BRDY#, BRDYC# Hold Time	1.0		nS	7	
t_{22}	AHOLD, BOFF# Setup Time	5.5		nS	7	
t_{23}	AHOLD, BOFF# Hold Time	1.0		nS	7	
t_{24}	BUSCHK#, EWBE#, HOLD, PEN# Setup Time	5.0		nS	7	
t_{25}	BUSCHK#, EWBE#, PEN# Hold Time	1.0		nS	7	
t_{25a}	HOLD Hold Time	1.5		nS	7	
t_{26}	A20M#, INTR, STPCLK# Setup Time	5.0		nS	7	12, 16
t_{27}	A20M#, INTR, STPCLK# Hold Time	1.0		nS	7	13
t_{28}	INIT, FLUSH#, NMI, SMI#, IGNNE# Setup Time	5.0		nS	7	12, 16, 17
t_{29}	INIT, FLUSH#, NMI, SMI#, IGNNE# Hold Time	1.0		nS	7	13
t_{30}	INIT, FLUSH#, NMI, SMI#, IGNNE# Pulse Width, Async	2.0		CLKs		15, 17
t_{31}	R/S# Setup Time	5.0		nS	7	12, 16, 17
t_{32}	R/S# Hold Time	1.0		nS	7	13
t_{33}	R/S# Pulse Width, Async.	2.0		CLKs	7	15, 17

Table 14. Pentium™ Processor 735\90 AC Specifications for 60-MHz Bus Operation (Contd.)
$3.135 < V_{CC} < 3.465V$, T_{CASE} = 0 to 70°C, C_L = 0 pF

Symbol	Parameter	Min	Max	Unit	Figure	Notes
t_{34}	D0-D63, DP0-7 Read Data Setup Time	3.0		nS	7	
t_{35}	D0-D63, DP0-7 Read Data Hold Time	2.0		nS	8	
t_{36}	RESET Setup Time	5.0		nS	8	11, 12, 16
t_{37}	RESET Hold Time	1.0		nS	8	11, 13
t_{38}	RESET Pulse Width, V_{CC} & CLK Stable	15		CLKs	8	11, 17
t_{39}	RESET Active After V_{CC} & CLK Stable	1.0		mS	8	Power up
t_{40}	Reset Configuration Signals (INIT, FLUSH#, FRCMC#) Setup Time	5.0		nS	8	12, 16, 17
t_{41}	Reset Configuration Signals (INIT, FLUSH#, FRCMC#) Hold Time	1.0		nS	8	13
t_{42a}	Reset Configuration Signals (INIT, FLUSH#, FRCMC#) Setup Time, Async.	2.0		CLKs	8	To RESET falling edge(16)
t_{42b}	Reset Configuration Signals (INIT, FLUSH#, FRCMC#, BRDYC#, BUSCHK#) Hold Time, Async.	2.0		CLKs	8	To RESET falling edge (27)
t_{42c}	Reset Configuration Signals (BRDYC#, BUSCHK#) Setup Time, Async.	3.0		CLKs	8	To RESET falling edge(27)
t_{42d}	Reset Configuration Signal BRDYC# Hold Time, RESET driven synchronously	1.0		nS		To RESET falling edge(1,27)
t_{43a}	BF, CPUTYP Setup Time	1.0		mS	8	To RESET falling edge(22)
t_{43b}	BF, CPUTYP Hold Time	2.0		CLKs	8	To RESET falling edge(22)
t_{43c}	APICEN Setup Time	2.0		CLKs	8	To RESET falling edge
t_{43d}	APICEN Hold Time	2.0		CLKs	8	To RESET falling edge
t_{44}	TCK Frequency		16.0	MHz	8	
t_{45}	TCK Period	62.5		nS	4	
t_{46}	TCK High Time	25.0		nS	4	@2V(1)
t_{47}	TCK Low Time	25.0		nS	4	@0.8V(1)

Table 14. Pentium™ Processor 735\90 AC Specifications for 60-MHz Bus Operation (Contd.)
$3.135 < V_{CC} < 3.465V$, T_{CASE} = 0 to 70°C, C_L = 0 pF

Symbol	Parameter	Min	Max	Unit	Figure	Notes
t_{48}	TCK Fall Time		5.0	nS	4	(2.0V–0.8V)(1,8,9)
t_{49}	TCK Rise Time		5.0	nS	4	(0.8V–2.0V)(1,8,9)
t_{50}	TRST# Pulse Width	40.0		nS	10	Asynchronous(1)
t_{51}	TDI, TMS Setup Time	5.0		nS	9	7
t_{52}	TDI, TMS Hold Time	13.0		nS	9	7
t_{53}	TDO Valid Delay	3.0	20.0	nS	9	8
t_{54}	TDO Float Delay		25.0	nS	9	1, 8
t_{55}	All Non-Test Outputs Valid Delay	3.0	20.0	nS	9	3, 8, 10
t_{56}	All Non-Test Outputs Float Delay		25.0	nS	9	1, 3, 8, 10
t_{57}	All Non-Test Inputs Setup Time	5.0		nS	9	3, 7, 10
t_{58}	All Non-Test Inputs Hold Time	13.0		nS	9	3, 7, 10
	APIC AC Specifications					
t_{60a}	PICCLK Frequency	2.0	16.66	MHz	4	
t_{60b}	PICCLK Period	60.0	500.0	nS	4	
t_{60c}	PICCLK High Time	9.0		nS	4	
t_{60d}	PICCLK Low Time	9.0		nS	4	
t_{60e}	PICCLK Rise Time	1.0	5.0	nS	4	
t_{60f}	PICCLK Fall Time	1.0	5.0	nS	4	
t_{60g}	PICD0-1 Setup Time	3.0		nS	7	To PICCLK
t_{60h}	PICD0-1 Hold Time	2.5		nS	7	To PICCLK
t_{60i}	PICD0-1 Valid Delay (LtoH)	4.0	38.0	nS	5	From PICCLK(28,29)
t_{60j}	PICD0-1 Valid Delay (HtoL)	4.0	22.0	nS	5	From PICCLK(28,29)

Table 15. Pentium™ Processor 735\90 Dual Processor Mode AC Specifications for 60-MHz Bus Operation

3.135 < V_{CC} < 3.465V, T_{CASE} = 0 to 70°C, C_L = 0 pF

Symbol	Parameter	Min	Max	Unit	Figure	Notes
t_{80a}	PBREQ#, PBGNT# Valid Delay	1.0	5.0	nS	5	18
t_{80b}	PHIT#, PHITM# Flight Time	0	2.0	nS	11	30
t_{81a}	PBREQ#, PBGNT# Setup Time	8.0		nS	7	18
t_{82}	PBREQ#, PBGNT# Hold Time	1.0		nS	7	18, 24
t_{83a}	A5-A31 Setup Time	3.0		nS	7	18, 21, 26
t_{83b}	D/C#, W/R#, CACHE#, LOCK#, SCYC Setup Time	4.0		nS	7	18, 21
t_{83c}	ADS#, M/IO# Setup Time	6.0		nS	7	18, 21
t_{83d}	HIT#, HITM# Setup Time	6.0		nS	7	18, 21
t_{83e}	HLDA Setup Time	6.0		nS	7	18, 21
t_{84}	ADS#, D/C#, W/R#, M/IO#, CACHE#, LOCK#, A5-A31, HLDA, HIT#, HITM#, SCYC Hold Time	1.0		nS	7	18, 21
t_{85}	DPEN# Valid Time		10.0	CLKs		18, 19, 23
t_{86}	DPEN# Hold Time	2.0		CLKs		18, 20, 23
t_{87}	APIC ID (BE0#–BE3#) Setup Time	2.0		CLKs	8	To RESET falling edge(23)
t_{88}	APIC ID (BE0#-BE3#) Hold Time	2.0		CLKs	8	From RESET falling edge(23)
t_{89}	D/P# Valid Delay	1.0	8.0	nS	5	Primary Processor Only

3.4.5.3 AC Timing Tables for a 66-MHz Bus

The AC specifications given in Tables 16 and 17 consist of output delays, input setup requirements and input hold requirements for a 66-MHz external bus. All AC specifications (with the exception of those for the TAP signals and AP IC signals) are relative to the rising edge of the CLK input.

All timings are referenced to 1.5V for both "0" and "1" logic levels unless otherwise specified. Within the sampling window, a synchronous input must be stable for correct Pentium processor (610\75, 735\90, 815\100) operation.

Table 16. Pentium™ Processor 815\100 AC Specifications for 66-MHz Bus Operation
$3.135 < V_{CC} < 3.465V$, T_{CASE} = 0 to 70°C, C_L = 0 pF

Symbol	Parameter	Min	Max	Unit	Figure	Notes
	Frequency	33.33	66.6	MHz		
t_{1a}	CLK Period	15.0	30.0	nS	4	
t_{1b}	CLK Period Stability		±250	pS		Adjacent Clocks(1,25)
t_2	CLK High Time	4.0		nS	4	@2V(1)
t_3	CLK Low Time	4.0		nS	4	@0.8V(1)
t_4	CLK Fall Time	0.15	1.5	nS	5	(2.0V–0.8V)(1)
t_5	CLK Rise Time	0.15	1.5	nS	4	(0.8V–2.0V)(1)
t_{6a}	ADSC#, PWT, PCD, BE0-7#, D/C#, W/R#, CACHE#, SCYC Valid Delay	1.0	7.0	nS	5	
t_{6b}	AP Valid Delay	1.0	8.5	nS	5	
t_{6c}	A3-A31, LOCK# Valid Delay	1.1	7.0	nS	5	
t_{6d}	ADS#, M/IO# Valid Delay	1.0	6.0	nS	5	
t_7	ADS#, ADSC#, AP, A3-A31, PWT, PCD, BE0-7#, M/IO#, D/C#, W/R#, CACHE#, SCYC, LOCK# Float Delay		10.0	nS	6	1
t_{8a}	APCHK#, IERR#, FERR# Valid Delay	1.0	8.3	nS	5	4
t_{8b}	PCHK# Valid Delay	1.0	7.0	nS	5	4
t_{9a}	BREQ, HLDA Valid Delay	1.0	8.0	nS	5	4
t_{9b}	SMIACT# Valid Delay	1.0	7.6	nS	5	4
t_{10a}	HIT# Valid Delay	1.0	8.0	nS	5	
t_{10b}	HITM# Valid Delay	1.1	6.0	nS	5	
t_{11a}	PM0-1, BP0-3 Valid Delay	1.0	10.0	nS	5	
t_{11b}	PRDY Valid Delay	1.0	8.0	nS	5	
t_{12}	D0-D63, DP0-7 Write Data Valid Delay	1.3	7.5	nS	5	
t_{13}	D0-D63, DP0-3 Write Data Float Delay		10.0	nS	6	1
t_{14}	A5-A31 Setup Time	6.0		nS	7	26
t_{15}	A5-A31 Hold Time	1.0		nS	7	

Table 16. Pentium™ Processor 815\100 AC Specifications for 66-MHz Bus Operation (Contd.)
$3.135 < V_{CC} < 3.465V$, T_{CASE} = 0 to 70°C, C_L = 0 pF

Symbol	Parameter	Min	Max	Unit	Figure	Notes
t_{16a}	INV, AP Setup Time	5.0		nS	7	
t_{16b}	EADS# Setup Time	5.5		nS	7	
t_{17}	EADS#, INV, AP Hold Time	1.0		nS	7	
t_{18a}	KEN# Setup Time	5.0		nS	7	
t_{18b}	NA#, WB/WT# Setup Time	4.5		nS	7	
t_{19}	KEN#, WB/WT#, NA# Hold Time	1.0		nS	7	
t_{20}	BRDY#, BRDYC# Setup Time	5.0		nS	7	
t_{21}	BRDY#, BRDYC# Hold Time	1.0		nS	7	
t_{22}	AHOLD, BOFF# Setup Time	5.5		nS	7	
t_{23}	AHOLD, BOFF# Hold Time	1.0		nS	7	
t_{24}	BUSCHK#, EWBE#, HOLD, PEN# Setup Time	5.0		nS	7	
t_{25a}	BUSCHK#, EWBE#, PEN# Hold Time	1.0		nS	7	
t_{25b}	HOLD Hold Time	1.5		nS	7	
t_{26}	A20M#, INTR, STPCLK# Setup Time	5.0		nS	7	12, 16
t_{27}	A20M#, INTR, STPCLK# Hold Time	1.0		nS	7	13
t_{28}	INIT, FLUSH#, NMI, SMI#, IGNNE# Setup Time	5.0		nS	7	12, 16, 17
t_{29}	INIT, FLUSH#, NMI, SMI#, IGNNE# Hold Time	1.0		nS	7	13
t_{30}	INIT, FLUSH#, NMI, SMI#, IGNNE# Pulse Width, Async	2.0		CLKs		15, 17
t_{31}	R/S# Setup Time	5.0		nS	7	12, 16, 17
t_{32}	R/S# Hold Time	1.0		nS	7	13
t_{33}	R/S# Pulse Width, Async.	2.0		CLKs		15, 17
t_{34}	D0-D63, DP0-7 Read Data Setup Time	3.0		nS	7	
t_{35}	D0-D63, DP0-7 Read Data Hold Time	2.0		nS	7	
t_{36}	RESET Setup Time	5.0		nS	8	11, 12, 16

Table 16. Pentium™ Processor 815\100 AC Specifications for 66-MHz Bus Operation (Contd.)
$3.135 < V_{CC} < 3.465V$, T_{CASE} = 0 to 70°C, C_L = 0 pF

Symbol	Parameter	Min	Max	Unit	Figure	Notes
t_{37}	RESET Hold Time	1.0		nS	8	11, 13
t_{38}	RESET Pulse Width, V_{CC} & CL K Stable	15.0		CLKs	8	11, 17
t_{39}	RESET Active After V_{CC} & CLK Stable	1.0		mS	8	Power up
t_{40}	Reset Configuration Signals (INIT, FLUSH#, FRCMC#) Setup Time	5.0		nS	8	12, 16, 17
t_{41}	Reset Configuration Signals (INIT, FLUSH#, FRCMC#) Hold Time	1.0		nS	8	13
t_{42a}	Reset Configuration Signals (INIT, FLUSH#, FRCMC#) Setup Time, Async.	2.0		CLKs	8	To RESET falling edge(16)
t_{42b}	Reset Configuration Signals (INIT, FLUSH#, FRCMC#, BRDYC#, BUSCHK#) Hold Time, Async.	2.0		CLKs	8	To RESET falling edge(27)
t_{42c}	Reset Configuration Signals (BRDYC#, BUSCHK#) Setup Time, Async.	3.0		CLKs	8	To RESET falling edge(27)
t_{42d}	Reset Configuration Signal BRDYC# Hold Time, RESET driven synchronously	1.0		nS		To RESET falling edge(1,27)
t_{43a}	BF, CPUTYP Setup Time	1.0		mS	8	To RESET falling edge(22)
t_{43b}	BF, CPUTYP Hold Time	2.0		CLKs	8	To RESET falling edge(22)
t_{43c}	APICEN Setup Time	2.0		CLKs	8	To RESET falling edge
t_{43d}	APICEN Hold Time	2.0		CLKs	8	To RESET falling edge
t_{44}	TCK Frequency		16.0	MHz		
t_{45}	TCK Period	62.5		nS	4	
t_{46}	TCK High Time	25.0		nS	4	@2V(1)
t_{47}	TCK Low Time	25.0		nS	4	@0.8V(1)
t_{48}	TCK Fall Time		5.0	nS	4	(2.0V–0.8V)(1,8,9)
t_{49}	TCK Rise Time		5.0	nS	4	(0.8V–2.0V)(1,8,9)
t_{50}	TRST# Pulse Width	40.0		nS	10	Asynchronous(1)
t_{51}	TDI, TMS Setup Time	5.0		nS	9	7

Table 16. Pentium™ Processor 815\100 AC Specifications for 66-MHz Bus Operation (Contd.)
$3.135 < V_{CC} < 3.465V$, T_{CASE} = 0 to 70°C, C_L = 0 pF

Symbol	Parameter	Min	Max	Unit	Figure	Notes
t_{52}	TDI, TMS Hold Time	13.0		nS	9	7
t_{53}	TDO Valid Delay	3.0	20.0	nS	9	8
t_{54}	TDO Float Delay		25.0	nS	9	1, 8
t_{55}	All Non-Test Outputs Valid Delay	3.0	20.0	nS	9	3, 8, 10
t_{56}	All Non-Test Outputs Float Delay		25.0	nS	9	1, 3, 8, 10
t_{57}	All Non-Test Inputs Setup Time	5.0		nS	9	3, 7, 10
t_{58}	All Non-Test Inputs Hold Time	13.0		nS	9	3, 7, 10
APIC AC Specifications						
t_{60a}	PICCLK Frequency	2.0	16.66	MHz		
t_{60b}	PICCLK Period	60.0	500.0	nS	4	
t_{60c}	PICCLK High Time	9.0		nS	4	
t_{60d}	PICCLK Low Time	9.0		nS	4	
t_{60e}	PICCLK Rise Time	1.0	5.0	nS	4	
t_{60f}	PICCLK Fall Time	1.0	5.0	nS	4	
t_{60g}	PICD0-1 Setup Time	3.0		nS	7	To PICCLK
t_{60h}	PICD0-1 Hold Time	2.5		nS	7	To PICCLK
t_{60i}	PICD0-1 Valid Delay (LtoH)	4.0	38.0	nS	5	From PICCLK(28,29)
t_{60j}	PICD0-1 Valid Delay (HtoL)	4.0	22.0	nS	5	From PICCLK(28,29)

Table 17. Pentium™ Processor 815\100 Dual Processor Mode AC Specifications for 66-MHz Bus Operation

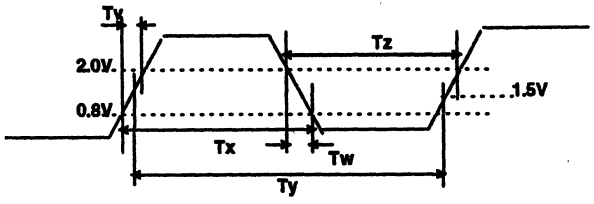
$3.135 < V_{CC} < 3.465V$, T_{CASE} = 0 to 70°C, C_L = 0 pF

Symbol	Parameter	Min	Max	Unit	Figure	Notes
t_{80a}	PBREQ#, PBGNT# Valid Delay	1.0	5.0	nS	5	18
t_{80b}	PHIT#, PHITM# Flight Time	0	2.0	nS	11	30
t_{81a}	PBREQ#, PBGNT# Setup Time	8.0		nS	7	18
t_{82}	PBREQ#, PBGNT# Hold Time	1.0		nS	7	18, 24
t_{83a}	A5-A31 Setup Time	3.0		nS	7	18, 21, 26
t_{83b}	D/C#, W/R#, CACHE#, LOCK#, SCYC Setup Time	4.0		nS	7	18, 21
t_{83c}	ADS#, M/IO# Setup Time	5.8		nS	7	18, 21
t_{83d}	HIT#, HITM# Setup Time	6.0		nS	7	18, 21
t_{83e}	HLDA Setup Time	6.0		nS	7	18, 21
t_{84}	ADS#, D/C#, W/R#, M/IO#, CACHE#, LOCK#, A5-A31, HLDA, HIT#, HITM#, SCYC Hold Time	1.0		nS	7	18, 21
t_{85}	DPEN# Valid Time		10.0	CLKs		18, 19, 23
t_{86}	DPEN# Hold Time	2.0		CLKs		18, 20, 23
t_{87}	APIC ID (BE0#-BE3#) Setup Time	2.0		CLKs	8	To RESET falling edge(23)
t_{88}	APIC ID (BE0#-BE3#) Hold Time	2.0		CLKs	8	From RESET falling edge(23)
t_{89}	D/P# Valid Delay	1.0	8.0	nS	5	Primary Processor Only

NOTES:
Notes 2, 6, and 14 are general and apply to all standard TTL signals used with the Pentium Processor family.
1. Not 100% tested. Guaranteed by design/characterization.
2. TTL input test waveforms are assumed to be 0 to 3V transitions with 1V/nS rise and fall times.
3. Non-test outputs and inputs are the normal output or input signals (besides TCK, TRST#, TDI, TDO, and TMS). These timings correspond to the response of these signals due to boundary scan operations.
4. APCHK#, FERR#, HLDA, IERR#, LOCK#, and PCHK# are glitch-free outputs. Glitch-free signals monotonically transition without false transitions (i.e., glitches).
5. 0.8V/ns ≤ CLK input rise/fall time ≤ 8V/ns.
6. 0.3V/ns ≤ input rise/fall time ≤ 5V/ns.
7. Referenced to TCK rising edge.
8. Referenced to TCK falling edge.
9. 1 ns can be added to the maximum TCK rise and fall times for every 10 MHz of frequency below 33 MHz.
10. During probe mode operation, do not use the boundary scan timings ($t_{55\text{-}58}$).
11. FRCMC# should be tied to V_{CC} (high) to ensure proper operation of the Pentium processor (610\75, 735\90, 815\100) as a primary processor.
12. Setup time is required to guarantee recognition on a specific clock. Pentium processor (610\75, 735\90, 815\100) must meet this specification for dual processor operation for the FLUSH# and RESET signals.

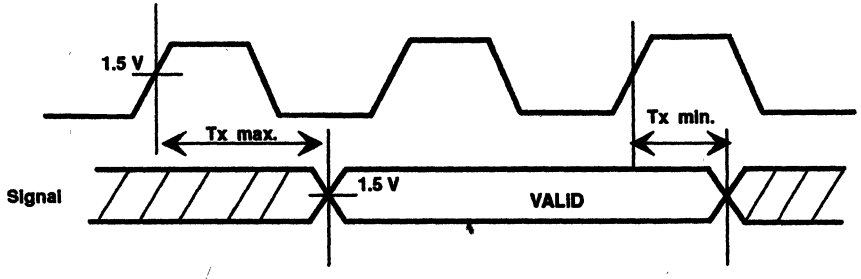
13. Hold time is required to guarantee recognition on a specific clock. Pentium processor (610\75, 735\90, 815\100) must meet this specification for dual processor operation for the FLUSH# and RESET signals.
14. All TTL timings are referenced from 1.5V.
15. To guarantee proper asynchronous recognition, the signal must have been de-asserted (inactive) for a minimum of 2 clocks before being returned active and must meet the minimum pulse width.
16. This input may be driven asynchronously. However, when operating two processors in dual processing mode, FLUSH# and RESET must be asserted synchronously to both processors.
17. When driven asynchronously, RESET, NMI, FLUSH#, R/S#, INIT, and SMI# must be de-asserted (inactive) for a minimum of 2 clocks before being returned active.
18. Timings are valid only when dual processor is present.
19. Maximum time DPEN# is valid from rising edge of RESET.
20. Minimum time DPEN# is valid after falling edge of RESET.
21. The D/C#, M/IO#, W/R#, CACHE#, and A5-A31 signals are sampled only on the CLK that ADS# is active.
22. BF and CPUTYP should be strapped to V_{CC} or V_{SS}.
23. RESET is synchronous in dual processing mode and functional redundancy checking mode. All signals which have a setup or hold time with respect to a falling or rising edge of RESET in UP mode, should be measured with respect to the first processor clock edge in which RESET is sampled either active or inactive in dual processing and functional redundancy checking modes.
24. The PHIT# and PHITM# signals operate at the core frequency (75, 90 or 100 MHz).
25. These signals are measured on the rising edge of adjacent CLKs at 1.5V. To ensure a 1:1 relationship between the amplitude of the input jitter and the internal and external clocks, the jitter frequency spectrum should not have any power spectrum peaking between 500 KHz and 1/3 of the CLK operating frequency. The amount of jitter present must be accounted for as a component of CLK skew between devices.
26. In dual processing mode, timing t_{14} is replaced by t_{83a}. Timing t_{14} is required for external snooping (e.g., address setup to the CLK in which EADS# is sampled active) in both uniprocessor and dual processor modes.
27. BRDYC# and BUSCHK# are used as reset configuration signals to select buffer size.
28. This assumes an external pullup resistor to V_{CC} and a lumped capacitive load such that the maximum RC product does not exceed R = 150Ω, C = 240 pF.
29. This assumes an external pullup resistor to V_{CC} and a lumped capacitive load such that the minimum RC product does not fall below R = 150Ω, C = 20 pF.
30. This is a flight time specification, that includes both flight time and clock skew. The flight time is the time from where the unloaded driver crosses 1.5V (50% of min V_{CC}), to where the receiver crosses the 1.5V level (50% of min V_{CC}). See Figure 11.

** Each valid delay is specified for a 0 pF load. The system designer should use I/O buffer modeling to account for signal flight time delays.



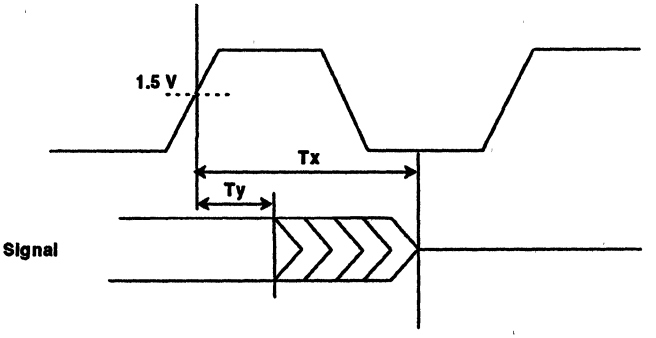
Tv = t5, t49, t60e; Tw = t4, t48, t60f; Tx = t3, t47, t60d
Ty = t1, t45, t60b; Tz = t2, t46, t60c

Figure 4. Clock Waveform



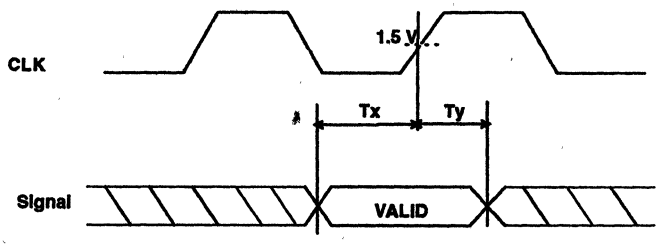
Tx = t6, t8, t9, t10, t11, t12, t60i, t80, t89

Figure 5. Valid Delay Timings



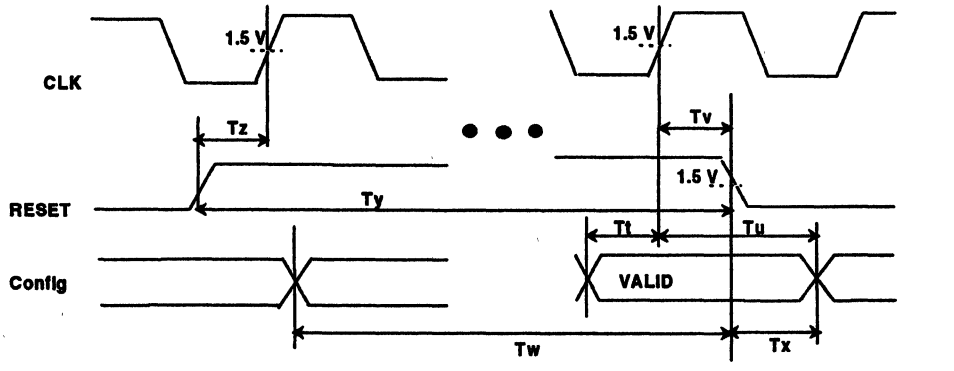
Tx = t7, t13; Ty = t6min, t12min

Figure 6. Float Delay Timings



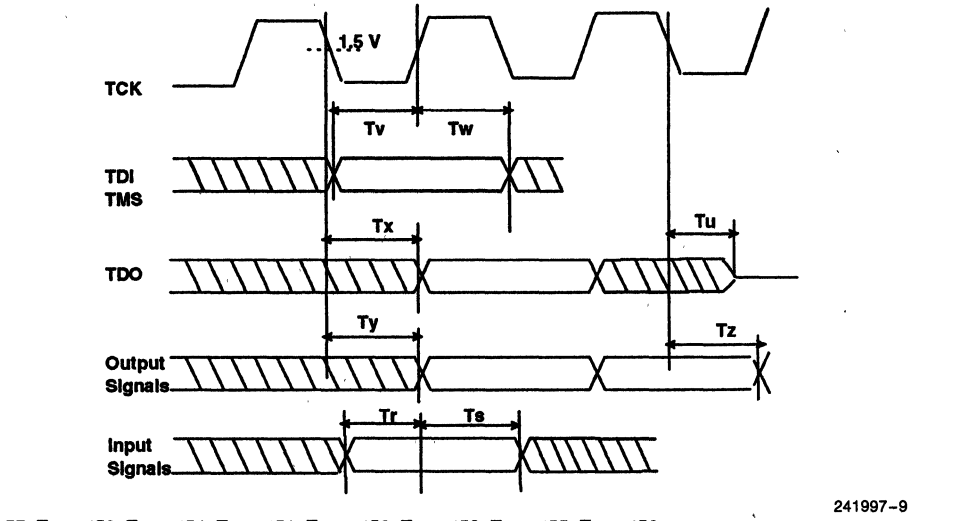
Tx = t14, t16, t18, t20, t22, t24, t26, t28, t31, t34, t60g (to PICCLK),t81, t83
Ty = t15, t17, t19, t21, t23, t25, t27, t29, t32, t35, t60h (to PICCLK), t82, t84

Figure 7. Setup and Hold Timings



Tt = t40, Tu = t41, Tv = t37, T w =t42, t43a, t43c, t87, Tx = t43b, t43d, t88, Ty = t38, t39, Tz = t36

Figure 8. Reset and Configuration Timings



Tr = t57, Ts = t58, Tu = t54, Tv = t51, Tw = t52, Tx = t53, Ty = t55, Tz = t56

Figure 9. Test Timings

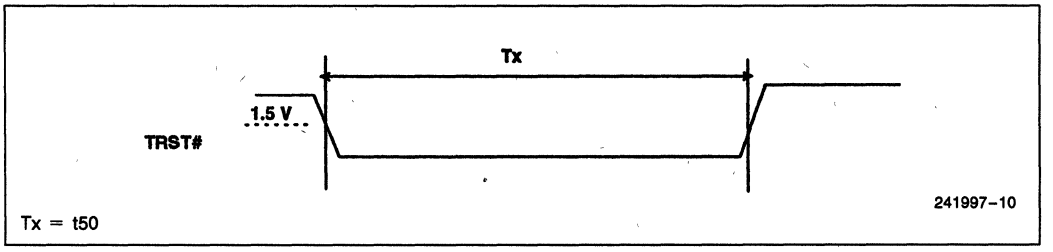


Figure 10. Test Reset Timings

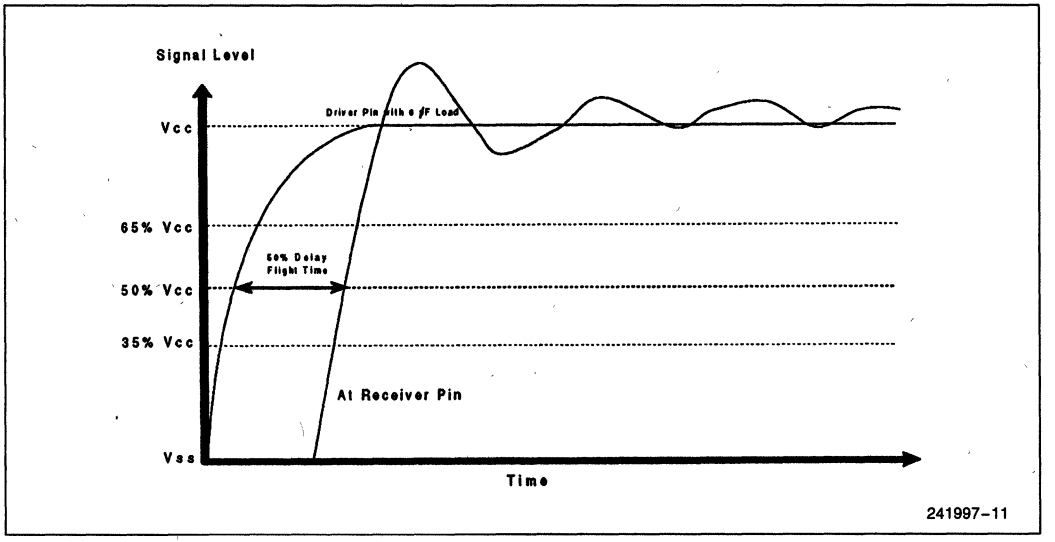


Figure 11. 50% V_{CC} Measurement of Flight Time

4.0 MECHANICAL SPECIFICATIONS

The Pentium processor (610\75, 735\90, 815\100) is packaged in a 296-pin staggered pin grid array (SPGA). The pins are arranged in a 37 x 37 matrix and the package dimensions are 1.95" x 1.95" (Table 18). A 1.25" x 1.25" copper tungsten heat spreader is brazed to the top of the ceramic to improve the package's thermal efficiency.

Table 18. Package Information Summary

	Package Type	Total Pins	Pin Array	Package Size
Pentium processor (610\75, 735\90, 815\100)	SPGA	296	37 x 37	1.95" x 1.95" 4.95 cm x 4.95 cm

The mechanical specifications for the Pentium processor (610\75, 735\90, 815\100) are provided in Table 19. Figure 12 shows the package dimensions.

Table 19. Package Dimensions

Family: Ceramic Staggered Pin Grid Array Package						
Symbol	Millimeters			Inches		
	Min	Max	Notes	Min	Max	Notes
A	3.91	4.70	Solid Lid	0.154	0.185	Solid Lid
A1	0.33	0.43	Solid Lid	0.013	0.017	Solid Lid
A2	2.62	2.97		0.103	0.117	
B	0.43	0.51		0.017	0.020	
D	49.28	49.91		1.940	1.965	
D1	45.47	45.97		1.790	1.810	
D2	31.50	32.00	Square	1.240	1.260	Square
D3	33.99	34.59		1.338	1.362	
D4	8.00	9.91		0.315	0.390	
E1	2.41	2.67		0.095	0.105	
E2	1.14	1.40		0.045	0.055	
F		.127	Diagonal		0.005	Diagonal
L	3.05	3.30		0.120	0.130	
N	296			296		
S1	1.52	2.54		0.060	0.100	

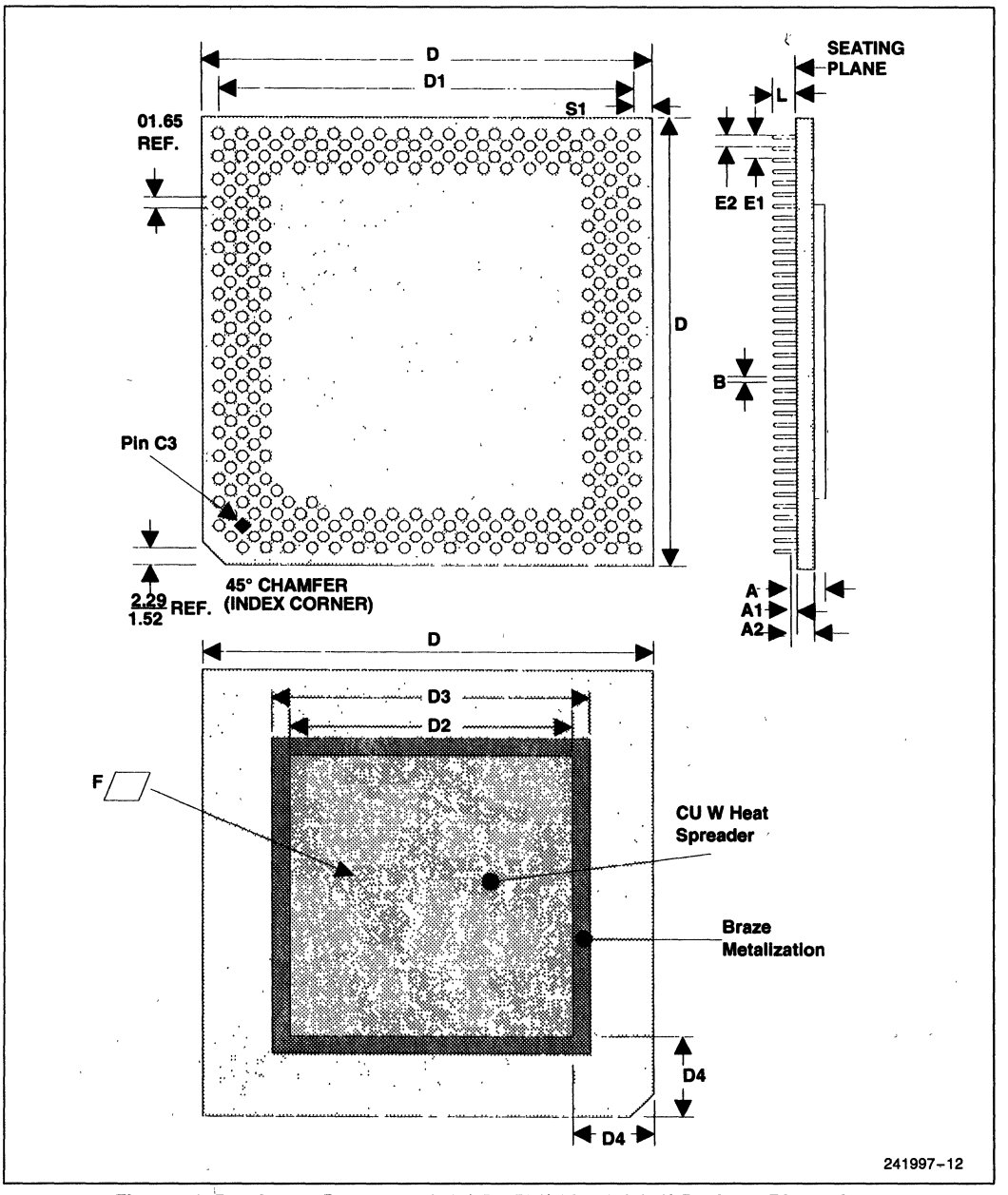


Figure 12. Pentium™ Processor (610\75, 735\90, 815\100) Package Dimensions

5.0 THERMAL SPECIFICATIONS

Due to the advanced 3.3V BiCMOS process that it is produced on, the Pentium processor (610\75, 735\90, 815\100) dissipates less power than the Pentium processor (510\60, 567\66).

The Pentium processor (610\75, 735\90, 815\100) is specified for proper operation when case temperature, T_{CASE}, (T_C) is within the specified range of 0°C to 70°C.

5.1 Measuring Thermal Values

The Pentium processor (610\75, 735\90, 815\100) package will include a heat spreader. To verify that the proper T_C (case temperature) is maintained, it should be measured at the center of the package top surface on the heat spreader (opposite of the pins). The measurement is made in the same way with or without a heat sink attached. When a heat sink is attached a hole (smaller than 0.150″ diameter) should be drilled through the heat sink to allow probing the center of the package. See Figure 13 for an illustration of how to measure T_C.

To minimize the measurement errors, it is recommended to use the following approach:

- Use 36-gauge or finer diameter K, T, or J type thermocouples. The laboratory testing was done using a thermocouple made by Omega (part number: 5TC-TTK-36-36).
- Attach the thermocouple bead or junction to the center of the package top surface using high thermal conductivity cements. The laboratory testing was done by using Omega Bond (part number: OB-100).
- The thermocouple should be attached at a 90-degree angle as shown in Figure 13.
- The hole size should be smaller than 0.150″ in diameter.

5.1.1 THERMAL EQUATIONS AND DATA

For the Pentium processor (610\75, 735\90, 815\100), an ambient temperature, T_A (air temperature around the processor), is not specified directly. The only restriction is that T_C is met. To calculate T_A values, the following equations may be used:

$$T_A = T_C - (P * \theta_{CA})$$

$$\theta_{CA} = \theta_{JA} - \theta_{JC}$$

where:

T_A and T_C = ambient and case temperature. (°C)

θ_{CA} = case-to-ambient thermal resistance. (°C/Watt)

θ_{JA} = junction-to-ambient thermal resistance. (°C/Watt)

θ_{JC} = junction-to-case thermal resistance. (°C/Watt)

P = maximum power consumption (Watt)

Table 20 lists the θ_{CA} values for the Pentium processor (610\75, 735\90, 815\100) with passive heat sinks. Figure 14 shows Table 20 in graphic format.

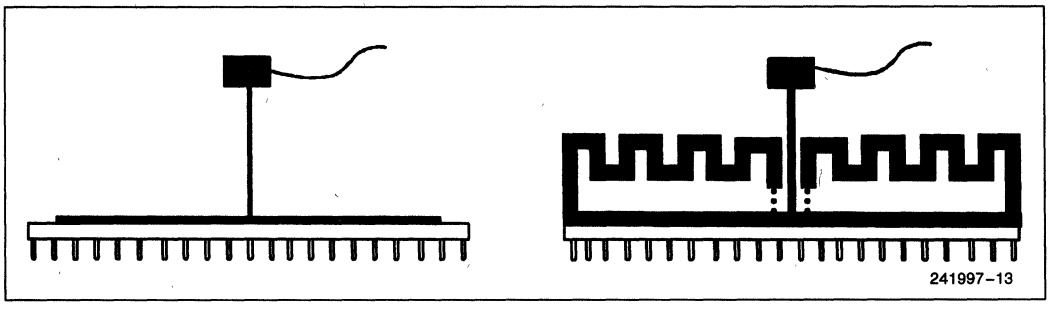


Figure 13. Technique for Measuring TC

Table 20. Thermal Resistances

Heat Sink in Inches	θ_{JC} (°C/Watt)	θ_{CA}(°C/Watt) vs. Laminar Airflow (linear ft/min)					
		0	100	200	400	600	800
1.95x1.95x0.25	0.9	8.7	7.6	6.2	4.0	3.2	2.6
1.95x1.95x0.35	0.9	8.4	7.1	5.6	3.6	2.9	2.4
1.95x1.95x0.45	0.9	8.0	6.6	4.9	3.2	2.5	2.1
1.95x1.95x0.55	0.9	7.7	6.1	4.3	2.8	2.2	1.9
1.95x1.95x0.65	0.9	7.3	5.6	3.9	2.6	2.0	1.7
1.95x1.95x0.80	0.9	6.6	4.9	3.5	2.2	1.8	1.6
1.95x1.95 x1.00	0.9	5.9	4.2	3.2	2.2	1.7	1.4
1.95x1.95x1.20	0.9	5.5	3.9	2.9	2.0	1.6	1.4
1.95x1.95x1.40	0.9	5.0	3.5	2.6	1.8	1.5	1.3
Without Heat Sink	1.4	11.4	10.5	8.7	5.7	4.5	3.8

NOTES:
Estimation for Pentium processor (610\75, 735\90, 815\100) with 1.25" x 1.25" x 0.040" CuW heat spreader.
Heat sinks are omni directional pin aluminum alloy.
Features were based on standard extrusion practices for a given height
Pin size ranged from 50 to 129 mils
Pin spacing ranged from 93 to 175 mils
Based thickness ranged from 79 to 200 mils
Heat sink attach was 0.005" of thermal grease.
Attach thickness of 0.002" will improve performance approximately 0.3°C/Watt

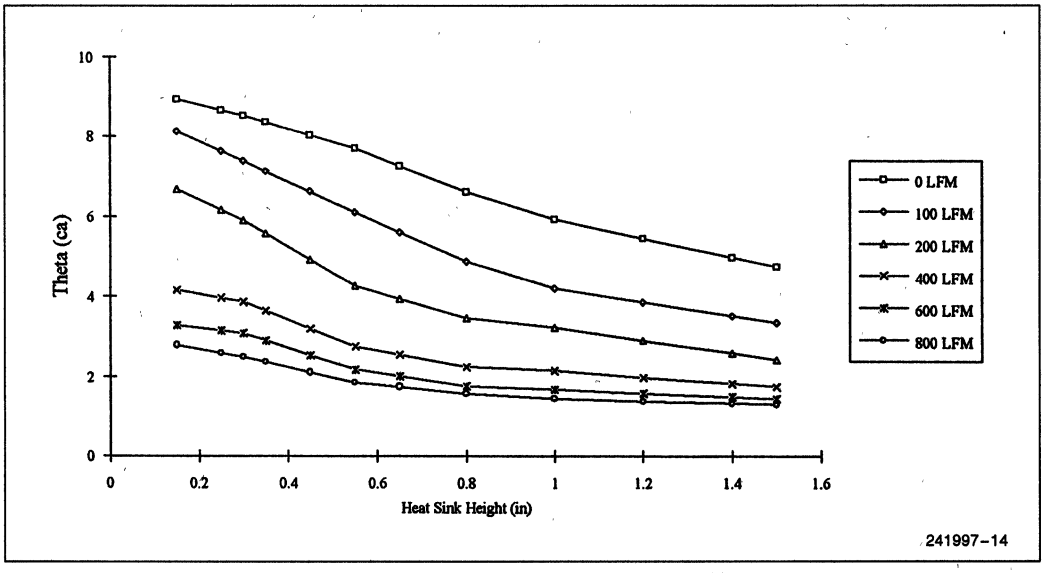


Figure 14. Thermal Resistance vs. Heatsink Height

6.0 FUTURE PENTIUM™ OVERDRIVE™ PROCESSOR SOCKET SPECIFICATION

6.1 Introduction

The Future Pentium OverDrive processor is an end-user single-chip CPU upgrade product for Pentium processor (610\75, 735\90, 815\100)-based systems. The Future Pentium OverDrive processor will speed up most software applications by 40% to 70%. It is binary compatible with the Pentium processor (610\75, 735\90, 815\100).

An upgrade socket (Socket 5) has been defined along with the Pentium processor (610\75, 735\90, 815\100) as part of the processor architecture. Upgradability can be supported by implementing either a single socket or a dual socket strategy for Pentium processor (610\75, 735\90, 815\100)-based systems. A single socket system will include a 320-pin SPGA Socket 5. When this system configuration is upgraded, the Pentium processor (610\75, 735\90, 815\100) is simply replaced by the Future Pentium OverDrive processor. A dual socket system will include a 296-pin SPGA socket for the Pentium processor (610\75, 735\90, 815\100) and a 320-pin SPGA Socket 5 for the second processor. In dual socket systems, Socket 5 can be filled with either the Dual processor or the Future Pentium OverDrive processor.

6.1.1 UPGRADE OBJECTIVES

Systems using the Pentium processor (610\75, 735\90, 815\100), and equipped with only one processor socket, must use socket 5 to also accept the Future Pentium OverDrive processor. Systems equipped with two processor sockets must use Socket 5 as the second socket to contain either the Pentium processor (610\75, 735\90, 815\100) Dual processor or the Future Pentium OverDrive processor.

Inclusion of Socket 5 in Pentium processor (610\75, 735\90, 815\100) systems provides the end-user with an easy and cost-effective way to increase system performance. The paradigm of simply installing an additional component into an easy to use Zero Insertion Force (ZIF) Socket to achieve enhanced system performance is familiar to the millions of end-users and dealers who have purchased Intel math coprocessor upgrades to boost system floating point performance.

The majority of upgrade installations which take advantage of Socket 5 will be performed by end users and resellers. Therefore, it is important that the design be "end-user easy," and that the amount of training and technical expertise required to install the upgrade processors be minimized. Upgrade installation instructions should be clearly described in the system user's manual. In addition, by making installation simple and foolproof, PC manufacturers can reduce the risk of system damage, warranty claims and service calls. Feedback from Intel's math coprocessor upgrade customers highlights three main characteristics of end user easy designs:

- accessible socket location
- clear indication of upgrade component orientation
- minimization of insertion force

The Future Pentium OverDrive processor will support the 82430NX PCIset. Unlike the Pentium processor (610\75, 735\90, 815\100), the Future Pentium OverDrive processor will not support the 82497 Cache Controller and 82492 Cache SRAM chip set.

6.1.2 INTEL VERIFICATION PROGRAM

The Intel Verification Program ensures that a Pentium processor (610\75, 735\90, 815\100)-based personal computer meets a minimum set of design criteria for reliable and straight-forward CPU upgradability with a Future Pentium OverDrive processor. Testing performed at the Intel Verification Labs confirms that Future Pentium OverDrive processor specifications for mechanical, thermal, electrical, functional, and end-user installation attributes have been met. While system designs may exceed these minimum design criteria, the intent is to provide end-users with confidence that computer systems based on verified designs can be upgraded with Future Pentium OverDrive processors.

The OEM submits production-ready designs to one of Intel's worldwide Verification Labs for testing. The OEM benefits from advance testing of the design prior to availablitiy of the Future Pentium OverDrive processor. By identifying and resolving upgradability problems before a system is introduced, the OEM increases system quality and reduces future support costs associated with end-user calls and complications when the CPU upgrade is ultimately installed.

Contact your local Intel representative for more information on the Intel Verification Program for Pentium processor (610\75, 735\90, 815\100)-based systems.

6.2 Future Pentium OverDrive Processor (Socket 5) Pinout

This section contains pinouts of the Future Pentium OverDrive Socket (Socket 5) when used as a single-socket turbo upgrade.

6.2.1 PIN DIAGRAMS

6.2.1.1 Socket 5 Pinout

For systems with a single socket for the Pentium processor (610\75, 735\90, 815\100) and Future Pentium OverDrive processor, the following pinout *must* be followed for the single socket location. Note that to be Intel Verified for a Future Pentium OverDrive processor, this must be a ZIF socket. The socket footprint contains V_{CC}, V_{CC5}, and V_{SS} pins that are internal no connects on the Pentium processor (610\75, 735\90, 815\100). These pins *must* be connected to the appropriate PCB power and ground layers to ensure Future Pentium OverDrive processor compatibility.

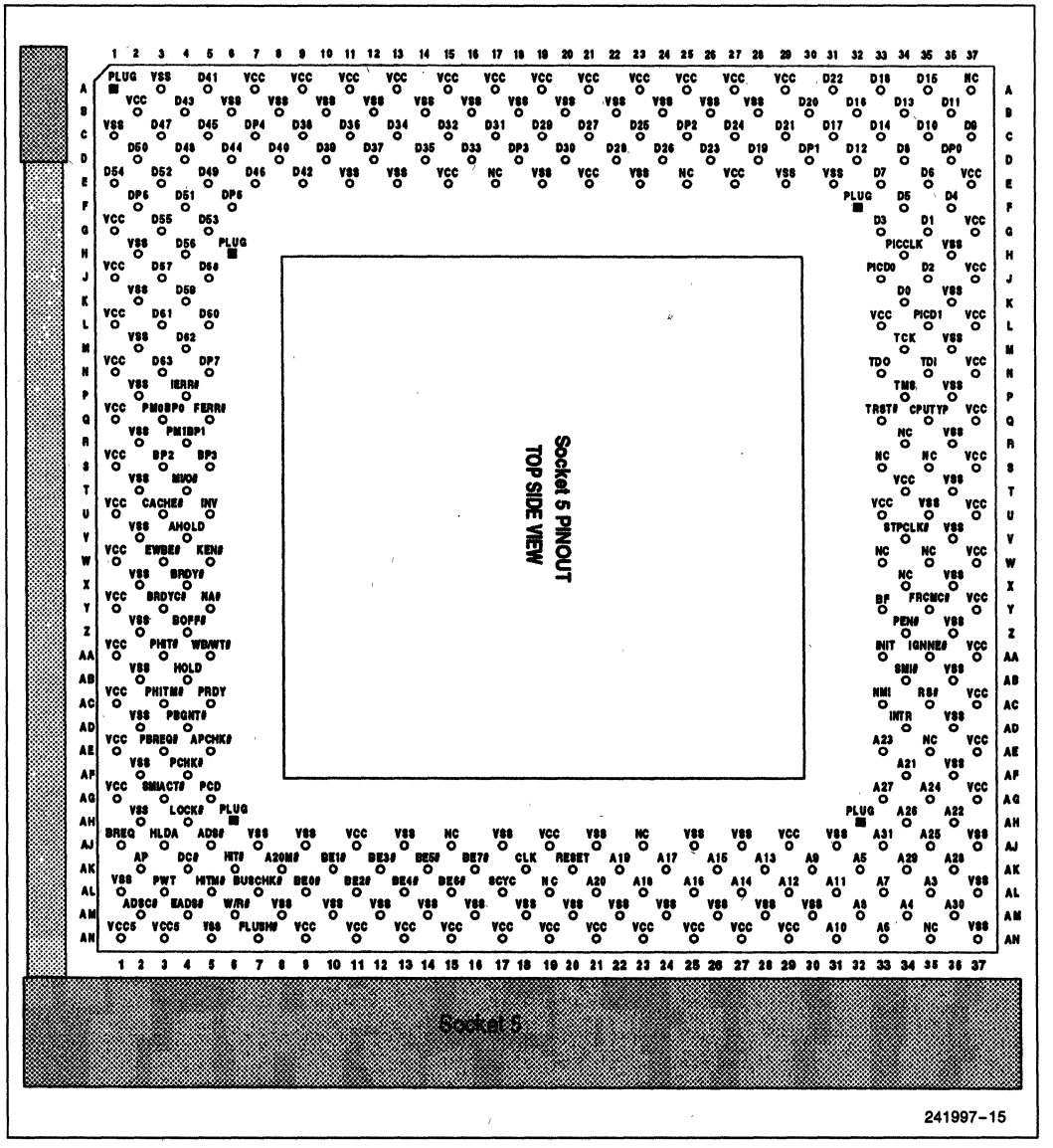


Figure 15. Socket 5 Pinout Top Side View

NOTE:

The "Socket 5 PINOUT TOP SIDE VIEW" text orientation on the top side view drawing in this section represents the orientation of the ink mark on the actual packages. (Note that the text shown in this section is not the actual text which will be marked on the packages).

Table 21. Pentium™ Processor (610\75, 735\90, 815\100) vs. Socket 5 Pins

Pentium™ Processor (610\75, 735\90, 815\100) Signal	Socket 5 Signal	Pin Number
INC	V_{SS}	A03
D/P#	NC	AE35
NO PIN	V_{SS}	AJ07
NO PIN	V_{SS}	AJ09
NO PIN	V_{CC}	AJ11
NO PIN	V_{SS}	AJ13
NO PIN	NC	AJ15
NO PIN	V_{SS}	AJ17
NO PIN	V_{CC}	AJ19
NO PIN	V_{SS}	AJ21
NO PIN	NC	AJ23
NO PIN	V_{SS}	AJ25
NO PIN	V_{SS}	AJ27
NO PIN	V_{CC}	AJ29
NO PIN	V_{SS}	AJ31

Pentium™ Processor (610\75, 735\90, 815\100) Signal	Socket 5 Signal	Pin Number
INC	V_{CC5}	AN01
INC	V_{CC5}	AN03
INC	V_{CC}	B02
NO PIN	V_{SS}	E11
NO PIN	V_{SS}	E13
NO PIN	V_{CC}	E15
NO PIN	NC	E17
NO PIN	V_{SS}	E19
NO PIN	V_{CC}	E21
NO PIN	V_{SS}	E23
NO PIN	NC	E25
NO PIN	V_{CC}	E27
NO PIN	V_{SS}	E29
NO PIN	V_{SS}	E31

NOTE:
All INCs are internal no connects. These signals are guaranteed to remain internally not connected in the Pentium processor (610\75, 735\90, 815\100).

6.3 Electrical Specifications

The Future Pentium OverDrive processor will have the same AC specifications, power and ground specifications and decoupling recommendations as the Pentium processor (610\75, 735\90, 815\100).

6.3.1 V_{CC5} PIN DEFINITION

The Future Pentium OverDrive processor pinout contains two 5V V_{CC} pins (V_{CC5}) used to provide power to the fan/heatsink. These pins should be connected to +5V ±5% regardless of the system design. Failure to connect V_{CC5} to 5V may cause the component to shut down.

6.4 Absolute Maximum Ratings of Upgrade

The on-chip Voltage Regulation and fan/heatsink devices included with the Future Pentium OverDrive processor require different stress ratings than the Pentium processor (610\75, 735\90, 815\100). The voltage regulator is surface mounted on the Future Pentium OverDrive processor and is, therefore, an integral part of the assembly. The Future Pentium OverDrive processor storage temperature ratings are tightened as a result. The fan is a detachable unit, and the storage temperature is stated separately in the table below. Functional operation of the Future Pentium OverDrive processor remains 0°C to 70°C.

Future Pentium OverDrive processor and Voltage Regulator Assembly

Storage Temperature ..... −30°C to 100°C

Case Temperature Under Bias ..... −30°C to 100°C

Fan

Storage Temperature ..... −30°C to 75°C

Case Temperature Under Bias ..... −30°C to 75°C

** 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.*

6.4.1 DC SPECIFICATIONS

The Future Pentium OverDrive processor will have compatible DC specifications to the Pentium processor (610\75, 735\90, 815\100), except that I_{CC3} (Power Supply Current), I_{SS5} (Fan/Heatsink Current), and V_{CC} are the following:

Table 22. Future Pentium™ OverDrive™ Processor I_{CC} Specification

$V_{CC5} = 5V \pm 5\%$, T_{CASE} = 0 to 70°C

Symbol	Parameter	Min	Max	Unit
I_{CC3}(1)	Power Supply Current		4330	mA
I_{CC5}	Fan/Heatsink Current		200	mA

NOTE:
1. V_{CC} = 3.135V to 3.6V

6.5 Mechanical Specifications

The Future Pentium OverDrive processor will be packaged in a 320-pin ceramic staggered pin grid array (SPGA). The pins will be arranged in a 37 x 37 matrix and the package dimensions will be 1.95″ x 1.95″ (4.95 cm x 4.95 cm).

Table 23. Processor Package Information Summary

	Package Type	Total Pins	Pin Array	Package Size
Future Pentium™ OverDrive™ Processor	SPGA	320	37 x 37	1.95″ x 1.95″ 4.95 cm x 4.95 cm

Table 24. Future Pentium™ OverDrive™ Processor Package Dimensions

Family: Ceramic Staggered Pin Grid Array Package						
Symbol	Millimeters			Inches		
	Min	Max	Notes	Min	Max	Notes
A*		33.88	Solid Lid		1.334	Solid Lid
A1	0.33	0.43	Solid Lid	0.013	0.017	Solid Lid
A2	2.62	2.97		0.103	0.117	
A4		20.32			0.800	
A5	10.16		Air Space	0.400		Air Space
B	0.43	0.51		0.017	0.020	
D	49.28	49.91		1.940	1.965	
D1	45.47	45.97		1.790	1.810	
E1	2.41	2.67		0.095	0.105	
E2	1.14	1.40		0.045	0.055	
L	3.05	3.30		0.120	0.130	
N	320		SPGA pins	320		SPGA pins
S1	1.52	2.54		0.060	0.100	

NOTES:
* Assumes the minimum air space above the fan/heatsink
A 0.2" clearance around three of four sides of the package is also required to allow free airflow through the fan/heatsink.

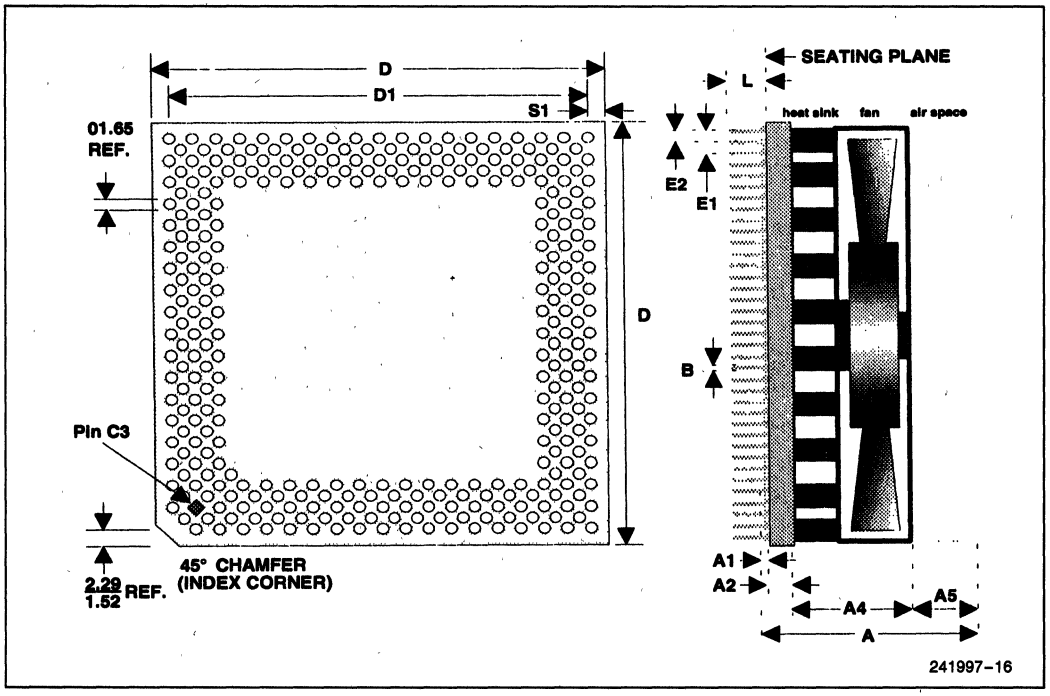


Figure 16. Future Pentium™ OverDrive™ Processor Package Dimensions

6.6 Thermal Specifications

The Future Pentium OverDrive processor will be cooled with a fan/heatsink cooling solution. The Future Pentium OverDrive processor with a fan/heatsink is specified for proper operation when T_A (air temperature entering the fan/heatsink) is a maximum of 45°C. When the $T_A(max) \leq 45°C$ specification is met, the fan/heatsink will keep T_C (case temperature) within the specified range of 0°C to 70°C provided airflow through the fan/heatsink is unimpeded.

6.7 Upgradability with Socket 5

6.7.1 INTRODUCTION

- Built-in upgradability for Pentium processor (610\75, 735\90, 815\100) based systems
  - — Supports the Future Pentium OverDrive processor 320-pin socket (Socket 5)

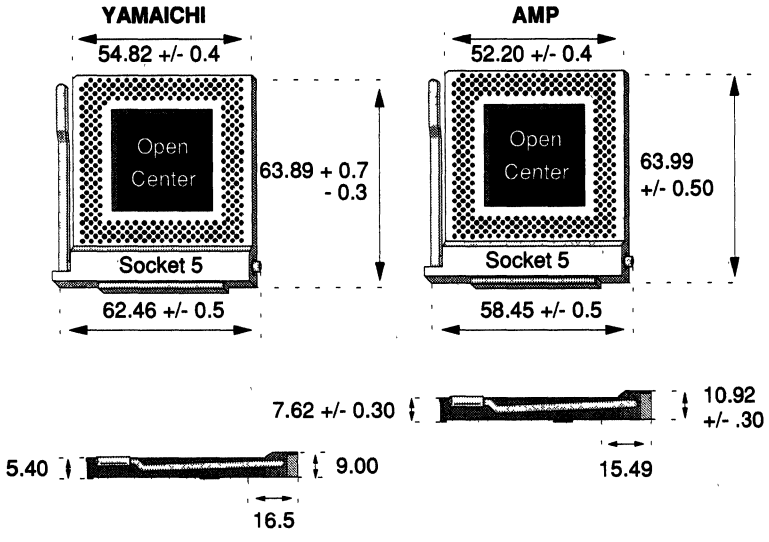
6.7.2 SOCKET 5 VENDORS

OEMs should contact Intel for the most current list of Intel-qualified socket vendors. For a complete list of Qualified Sockets and Vendor Order Numbers, call the Intel Faxback number for your geographical area and have document #7209 automatically faxed to you. Figure 17 shows preliminary dimensions for AMP and Yamaichi sockets. OEMs should directly contact the socket vendors for the most current socket information. Figure 18 shows the upgrade processor's orientation in Socket 5.

To order Socket 5 from AMP and Yamaichi, the phone numbers and part numbers are as follows:

AMP: 1-800-522-6752
part#: 916513-1

Yamaichi: 1-800-769-0797
part#: NP210-320K13625



NOTE:
All dimensions are in mm.

Figure 17. Socket 5 Footprint Dimensions

WARNING:
See socket manufacturer for the most current information.

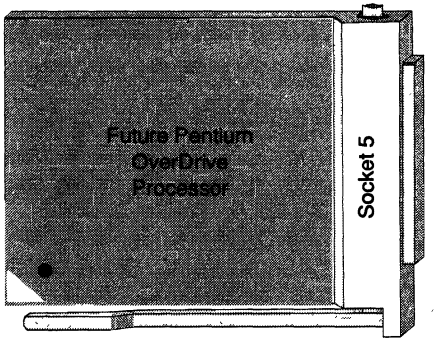


Figure 18. Chip Orientation in Socket 5

6.8 Testability

6.8.1 BOUNDARY SCAN

The Future Pentium OverDrive processor supports the IEEE Standard 1149.1 boundary scan using the Test Access Port (TAP) and TAP Controller. The boundary scan register for the Future Pentium OverDrive processor contains a cell for each pin. The turbo upgrade component will have a different bit order than the Pentium processor (610\75, 735\90, 815\100). If the TAP port on your system will be used by an end user following installation of the Future Pentium OverDrive processor, please contact Intel for the bit order of the upgrade processor boundary scan register.

FUTURE PENTIUM™ OVERDRIVE™ PROCESSOR FOR PENTIUM™ PROCESSOR (510\60, 567\66)-BASED SYSTEMS SOCKET SPECIFICATIONS

1.0 INTRODUCTION

The Future Pentium™ OverDrive™ processor is an end user single-chip CPU upgrade product for Pentium processor (510\60, 567\66)-based systems. The Future Pentium OverDrive processor will speed up most software applications by 40% to 70%. It is binary compatible with the Pentium processor (510\60, 567\66).

An upgrade socket (Socket 4) has been defined along with the Pentium processor (510\60, 567\66) as part of the processor architecture. The Future Pentium OverDrive processor will be socket compatible with the Pentium processor (510\60, 567\66). The Future Pentium OverDrive processor is packaged in a 273-pin ceramic pin grid array package with an attached fan/heatsink present on the turbo upgrade processor component.

Execution tracing is not supported in the Future Pentium OverDrive processor, and performance monitoring is implemented differently than in the Pentium processor (510\60, 567\66).

1.1 Upgrade Objectives

Systems using the Pentium processor (510\60, 567\66) must use Socket 4 to also accept the Future Pentium OverDrive processor. Inclusion of upgrade Socket 4 in Pentium processor (510\60, 567\66) systems provides the end user with an easy and cost effective way to increase system performance. The process of simply installing an upgrade component into an easy to use Zero Insertion Force (ZIF) socket to achieve enhanced system performance is familiar to the millions of end users and dealers who have purchased Intel Math CoProcessor upgrades to boost system floating-point performance.

Inclusion of Socket 4 in Pentium processor (510\60, 567\66) systems provides the end-user with an easy and cost-effective way to increase system performance. The paradigm of simply installing an additional component into an easy to use Zero Insertion Force (ZIF) Socket to achieve enhanced system performance is familiar to the millions of end-users and dealers who have purchased Intel math coprocessor upgrades to boost system floating point performance.

The majority of upgrade installations which take advantage of Socket 4 will be performed by end users and resellers. Therefore, it is important that the design be "end user easy," and that the amount of training and technical expertise required to install the upgrade processors be minimized. Upgrade installation instructions should be clearly described in the system user's manual. In addition, by making installation simple and foolproof, PC manufacturers can reduce the risk of system damage, warranty claims and service calls.

Feedback from Intel's Math CoProcessor upgrade customers highlights three main characteristics of end user easy designs:

- accessible socket location
- clear indication of upgrade component orientation
- minimization of insertion force

The Future Pentium OverDrive processor will support the Intel 82430 PCIset. Unlike the Pentium processor (510\60, 567\66), the Future Pentium OverDrive processor will not support the 82496 Cache Controller and 82491 Cache SRAM chip set.

1.2 Intel Verification Program

The Intel Verification Program ensures that a Pentium processor (510\60, 567\66)-based personal computer meets a minimum set of design criteria for reliable and straightforward CPU upgradability with a Future Pentium OverDrive processor. Testing performed at the Intel Verification Labs confirms that future Pentium OverDrive processor specifications for mechanical, thermal, electrical, functional, and end-user installation attributes have been met. While system designs may exceed these minimum design criteria, the intent is to provide end-users with confidence that computer systems based on verified designs can be upgraded with Futute Pentium OverDrive processors.

The OEM submits production-ready designs to one of Intel's worldwide Verification Labs for testing. The OEM benefits from advance testing of the design prior to availability of the Future Pentium OverDrive processor. By identifying and resolving upgradability problems before a system is introduced, the OEM increases system quality and reduces future support costs associated with end-user calls and complications when the CPU upgrade is ultimately installed.

Contact your local Intel representative for more information on the Intel Verification Program for Pentium processor (510\60, 567\66)-based systems.

2.0 Future Pentium™ OverDrive™ Processor Socket

The following drawings in Figure 1 show the preliminary worst case socket footprints from two qualified Socket 4 vendors, AMP and Yamaichi. OEMs should work directly with socket vendors for the most current socket information.

To order Socket 4 from AMP and Yamaichi, the phone numbers and part numbers are:

AMP:	1-800-522-6752	Part#: 916510-1
Yamaichi:	1-800-769-0797	Part#: NP11J-273K13221

Figure 2 shows the Future Pentium OverDrive processor chip's orientation in the Socket 4.

For a complete list of Qualified Sockets and Vendor Order Numbers, call the Intel Faxback number for your geographical area and have document #7209 automatically faxed to you.

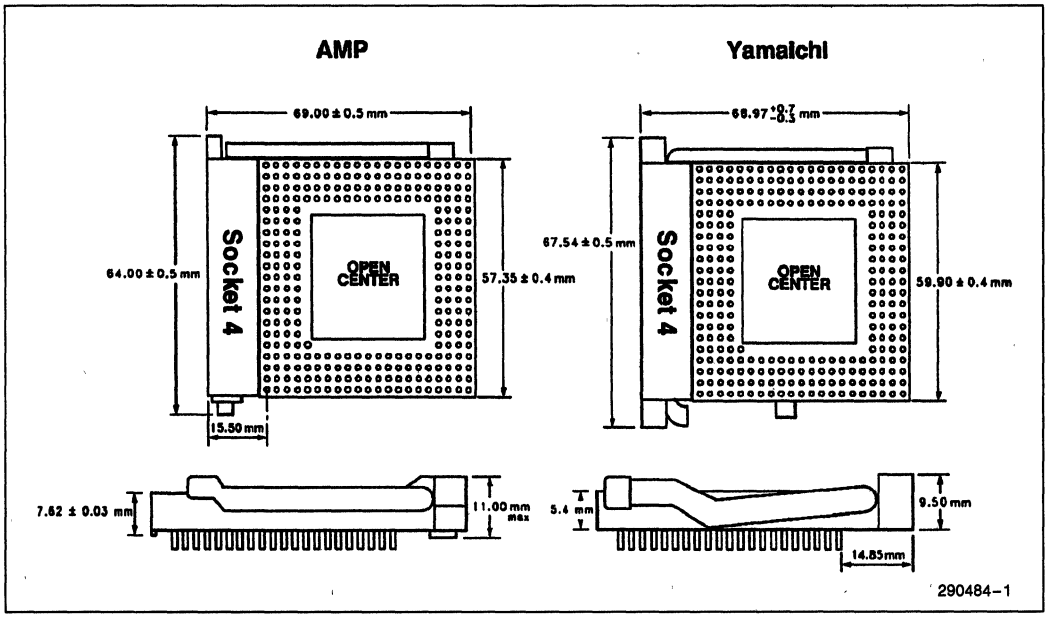


Figure 1. Socket 4 Footprint Dimensions (See socket manufacturer for the most current information.)

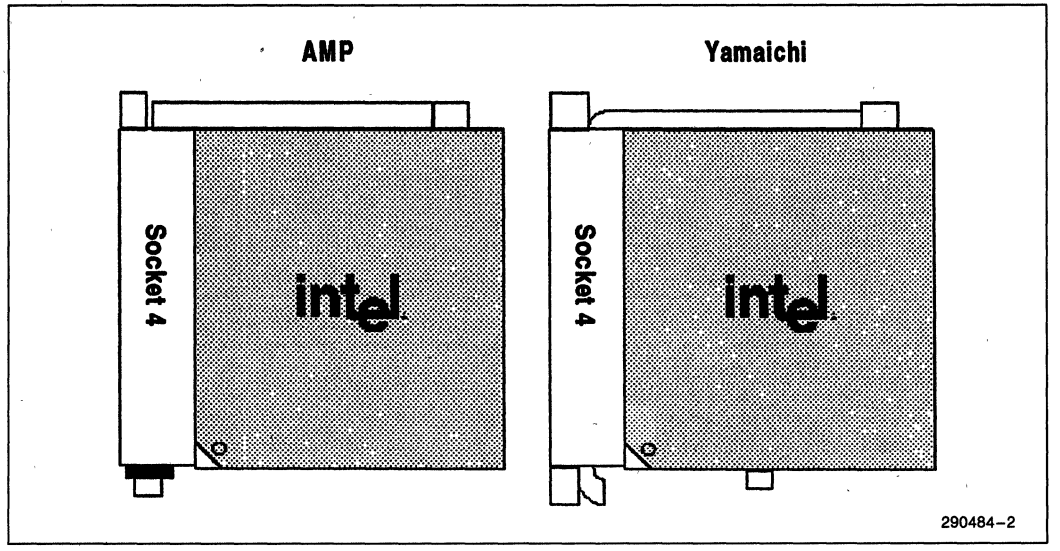


Figure 2. Chip Orientation in Socket 4

3.0 SOCKET 4 PINOUT

The Future Pentium OverDrive processor pinout is identical to that of the Pentium processor (510\60, 567\66). Note that all input pins must meet their AC/DC specifications to guarantee proper functional behavior. Figure 3 shows the Socket 4 pinout.

Locations E17, S17 and S5 should be plugged on Socket 4 in order to ensure that the Pentium processor (510\60, 567\66) or OverDrive processor chip is installed in the socket with the correct orientation.

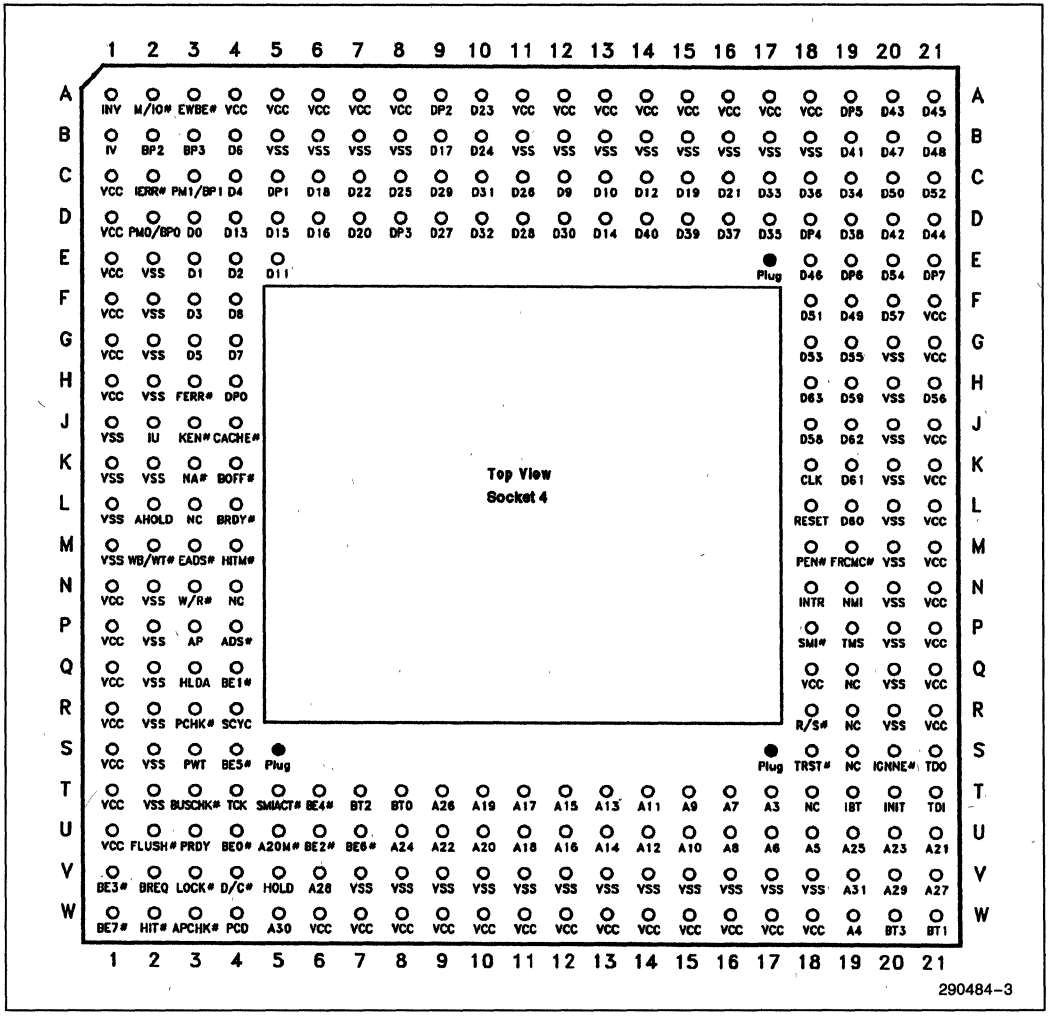


Figure 3. Socket 4 Pinout (Top View)

4.0 ELECTRICAL SPECIFICATIONS

The Future Pentium OverDrive processor will have the same power and ground specifications, decoupling recommendations, connection specifications and maximum ratings as the Pentium processor (510\60, 567\66).

4.1 Absolute Maximum Ratings For Upgrade

The on-chip voltage regulation and fan/heatsink devices included on the Future Pentium OverDrive processor require different stress ratings than the Pentium processor (510\60, 567\66). The voltage regulator is surface mounted on the Future Pentium OverDrive processor and is, therefore, an integral part of the assembly. The Future Pentium OverDrive processor storage temperature ratings area tightened as a result. The fan is a detachable unit, and the storage temperature is stated separately in the table below. Functional operation of the Future Pentium OverDrive processor remains 0°C to 70°C.

Table 1. Absolute Maximum Ratings

Future Pentium™ OverDrive™ Processor and Voltage Regulator Assembly:					
	Parameter	**Min**	**Max**	**Unit**	**Notes**
	Storage Temperature	−30	100	°C	
	Case Temperature Under Bias	−30	100	°C	
Fan:					
	Parameter	**Min**	**Max**	**Unit**	**Notes**
	Storage Temperature	−30	75	°C	
	Case Temperature Under Bias	−30	75	°C	

WARNING:
Stressing the devices 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.

4.2 DC Specifications

The Future Pentium OverDrive processor will have the same DC specifications as the Pentium processor (510\60, 567\66), including I_{CC} (Power Supply Current) as shown in Table 2.

Table 2. OverDrive™ Processor I_{CC} Specifications(2)

Symbol	Parameter	Min	Max	Unit	Notes
I_{CC}	Power Supply Current		3200 2910	mA mA	66 MHz(1) 60 MHz(1)

NOTES:
1. Worst case average I_{CC} for a mix of test patterns. (The mix of test patterns will be determined after silicon is examined.)
2. Refer to Pentium processor at iCOMP index 510\60 MHz, and Pentium processor at iCOMP index 567\66 MHz datasheet for remaining Pentium processor (510\60, 567\66) DC and V_{CC} specifications.

Refer to Pentium processor at iCOMP index 510\60 MHz, and Pentium processor at iCOMP index 567\66 MHz datasheets for a listing of the remaining DC specifications.

4.3 AC Specifications

The Future Pentium OverDrive processor will have the same AC specifications as the Pentium processor (510\60, 567\66). Refer to Pentium processor at iCOMP index 510\60 MHz, and Pentium processor at iCOMP index 567\66 MHz datasheets for a listing of the AC specifications. The functional parameters for the Future Pentium OverDrive processor's AC specifications are the same as those for Pentium processor (510\60, 567\66) except T_{SINK} as shown below:

$$T_{SINK} = 0°C \text{ to } + 70°C$$

5.0 MECHANICAL SPECIFICATIONS

The Future Pentium OverDrive processor for Pentium processor (510\60, 567\66)-based systems is packaged in a 273-pin ceramic pin grid array (PGA) with attached fan/heatsink. The pins are arranged in a 21 x 21 matrix and the package dimensions will be 2.16" x 2.16" (5.49 cm x 5.49 cm). See Table 3.

Table 3. OverDrive™ Processor Package Information Summary

Package Type	Total Pins	Pin Array	Package Size
PGA	273	21 x 21	2.16" x 2.16" (5.49 cm x 5.49 cm)

NOTE:
See DC Specifications for more detailed power specifications.

As can be seen in the mechanical dimensions in Table 4 and Figure 4, the actual height required by the heatsink and fan is less than the total space allotted. Since the Future Pentium OverDrive processor for Pentium processor (510\60, 567\66)-based systems employs a fan/heatsink, a certain amount of space is required above the fan/heatsink unit to ensure that the airflow is not blocked. Figure 5 shows unacceptable blocking of the airflow for the Future Pentium OverDrive processor fan/heatsink. Figure 6 details the minimum space needed around the PGA package to ensure proper heatsink airflow.

As shown in Figure 6, it is acceptable to allow any device (i.e., add-in cards, surface mount device, chassis, etc.) to enter within the free space distance of 0.2" from the PGA package if it is not taller than the level of the heatsink base. In other words, if a component is taller than height "B," it cannot be closer to the PGA package than distance "A." This applies to three of the four sides of the PGA package, although the back and handle sides of a ZIF socket will generally automatically meet this specification since they have widths larger than distance "A."

Table 4. OverDrive™ Processor Mechanical Specifications

Family: Ceramic Pin Grid Array Package						
Symbol	Millimeters			Inches		
	Min	Max	Notes	Min	Max	Notes
A		33.98	Solid Lid		1.338	Solid Lid
A1	2.84	3.50	Solid Lid	0.112	0.138	Solid Lid
A2	0.33	0.43	Solid Lid	0.013	0.017	Solid Lid
A3	2.51	3.07		0.099	0.121	
A4		20.32			0.800	
A5	10.16			0.400		
B	0.43	0.51		0.017	0.020	
D	54.61	55.11		2.150	2.170	
D1	50.67	50.93		1.995	2.005	
E1	2.29	2.79		0.090	0.110	
L	3.05	3.30		0.120	0.130	
N	273			273		
S1	1.65	2.16		0.065	0.085	

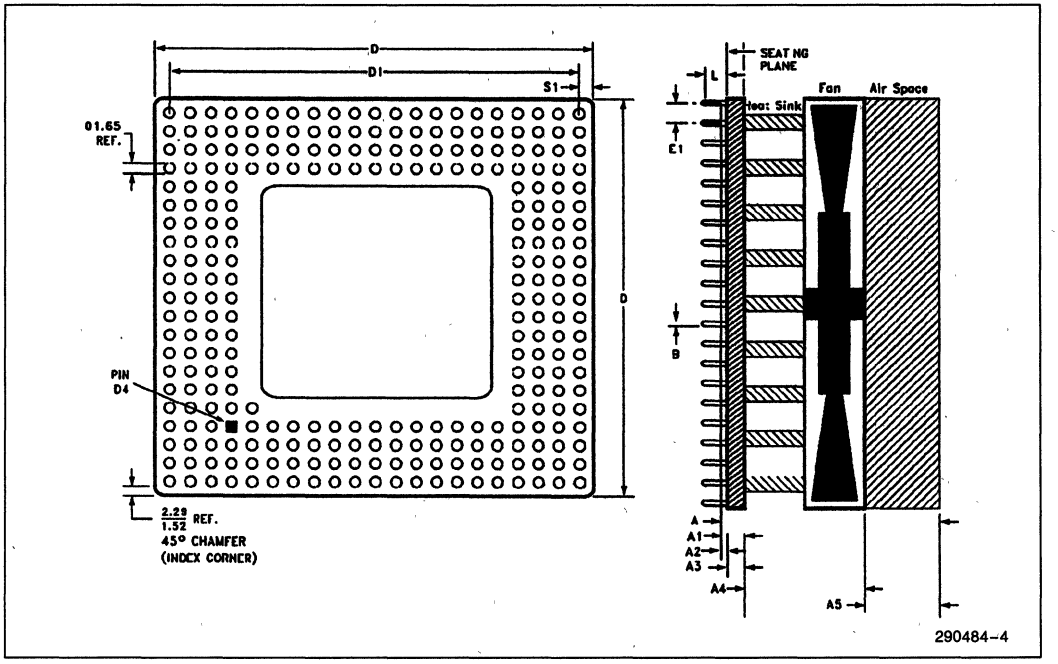


Figure 4. Processor Package Dimensions

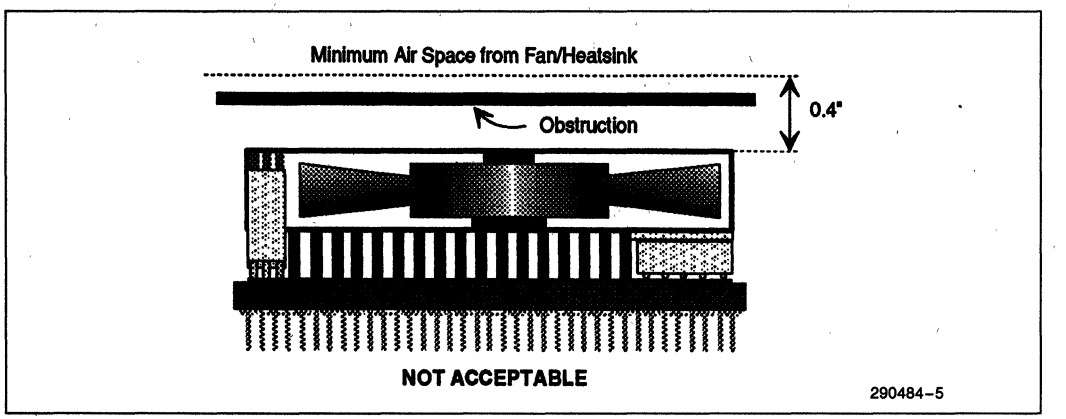


Figure 5. Fan/Heatsink Top Space Requirements

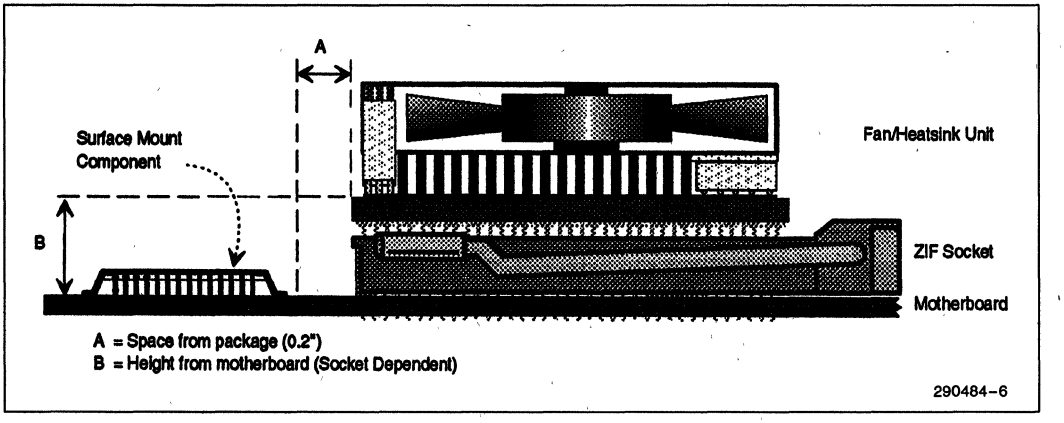


Figure 6. Required Free Space from Sides of PGA Package

6.0 THERMAL SPECIFICATIONS

The fan/heatsink cooling solution will properly cool the Future Pentium OverDrive processor as long as the maximum air temperature entering the fan/heatsink cooling solution (T_A(In)) does not exceed 45°C. It is left up to the OEM to ensure that T_A(In) meets this specification by providing sufficient airflow around the Future Pentium OverDrive processor heatsink unit.

Intel's fan/heatsink will dissipate approximately 1W and is powered by the chip such that no external wires or connections are required. The extra power needed for the fan/heatsink is taken into account in the I_{CC} numbers of the processor. Additionally, Intel is evaluating the feasibility of having the Future Pentium OverDrive processor monitor its temperature. No BIOS or hardware changes will be needed for this thermal protection mechanism. The shutdown temperature will be greater than the maximum temperature specification of the processor. The fan/heatsink unit will be designed to be removable so that if fan failure should occur, the unit may be easily replaced. Figure 7 gives a functional representation of the processor and fan/heatsink unit. The actual fan/heatsink unit may be different from the one shown in the figure.

Since the Future Pentium OverDrive processor for Pentium processor (510\60, 567\66)-based systems employs a fan/heatsink, it is not as important that the processor heatsink receive direct airflow, rather that the system has sufficient capability to remove the warm air that the Future Pentium OverDrive processor will generate. This implies that enough airflow exists at the Socket 4 to keep localized heating from occurring. This can be accomplished by a standard power supply fan with a clear path to the processor. Figure 8 shows how system design can cause localized heating to occur by limiting the airflow in the area of the processor. The airflow supplied in the system should also be enough to ensure that the OEM processor shipped with the system will meet the OEM processor thermal specifications before the system is upgraded with the Future Pentium OverDrive processor.

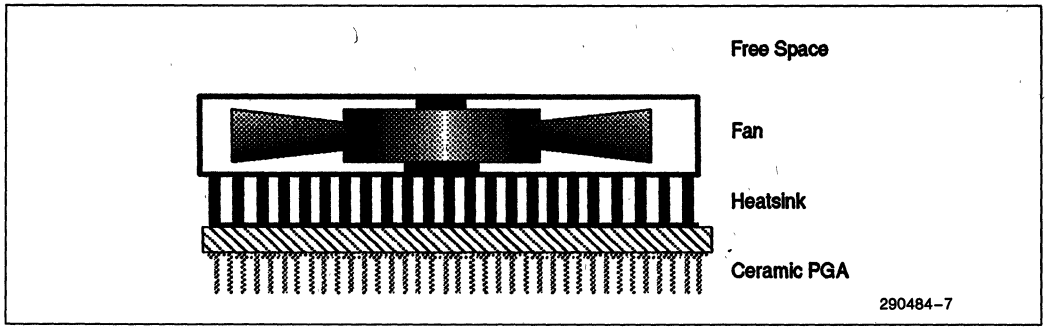


Figure 7. Fan/Heatsink Example

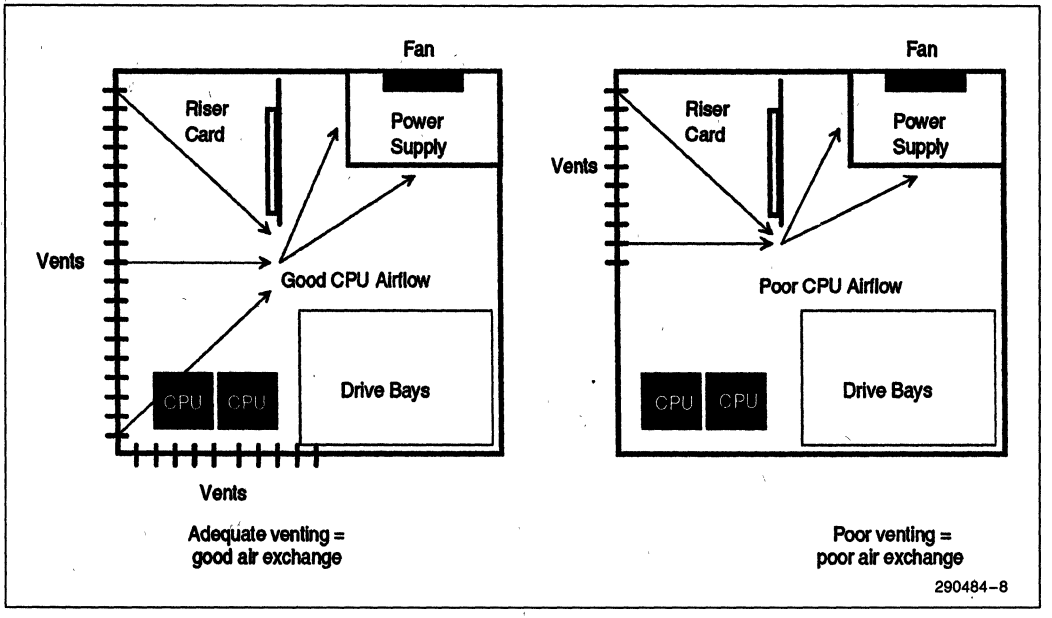


Figure 8. Airflow Design Examples

7.0 TESTABILITY

7.1 Boundary Scan

The Future Pentium OverDrive processor supports the IEEE Standard 1149.1 boundary scan using the Test Access Port (TAP) and TAP Controller. The boundary scan register for the Future Pentium OverDrive processor contains a cell for each pin. The turbo upgrade component will have a different bit order than the Pentium processor (510\60, 567\66). If the TAP port on your system will be used by an end user following installation of the Future Pentium OverDrive processor, please contact Intel for the bit order of the upgrade processor boundary scan register.

Pentium™ PROCESSOR REFERENCE SHEET

For more information regarding the Pentium™ Processor, you may obtain the *Pentium Processor User's Manual* by calling our toll-free literature distribution center at 1-800-548-4725. The *Pentium Processor User's Manual* is a three-volume set with details on Pentium Processor hardware and software design parameters, with specifications also for the 82496 Cache Controller/82491 Cache SRAM, and 82497 Cache Controller/82492 Cache SRAM chipsets.

Also listed in this chapter are brief technical profiles of other Pentium Processor peripheral products, including the 82489DX Advanced Interrupt Controller and the 82430 PCIset. For more information on these products, please consult the 1995 Peripherals Handbook.

241815-1

December 1994
Order Number: 241815-002

82430LX/82430NX PCIset

- Supports the Pentium™ Processor at 60 and 66 MHz (82430LX)
- Supports the Pentium Processor at ICOMP™ Index 735\90 MHz, Pentium Processor at ICOMP Index 815\100 MHz, and Pentium Processor at ICOMP Index 610\75 MHz
- Supports Uni-Processor (UP) or Dual-Processor (DP) Configurations
- Interfaces the Host and Standard Buses to the PCI Local Bus
  - Up to 132 MBytes/Sec Transfer Rate
  - Full Concurrency Between CPU Host Bus and PCI Bus Transactions
- Integrated Cache Controller Provided for Optional Second Level Cache
  - 256 KByte or 512 KByte Cache
  - Write-Back or Write-Through Policy (82430LX)
  - Write-Back Policy (82430NX)
  - Standard or Burst SRAM
- Integrated Tag RAM for Cost Savings on Second Level Cache
- Supports the Pipelined Address Mode of the Pentium Processor for Higher Performance
- Provides a 64-Bit Interface to DRAM Memory
  - From 2 MBytes to 512 MBytes of Main Memory
  - 70 ns and 60 ns DRAMs Supported
- Optional ISA or EISA Standard Bus Interface
  - Single Component ISA Controller
  - Two Component EISA Bus Interface
  - Minimal External Logic Required
- Supports Burst Read and Writes of Memory from the CPU and PCI Buses
- Five Integrated Write Posting and Read Prefetch Buffers Increase CPU and PCI Performance
- Host CPU Writes to PCI Converted to Zero Wait-State PCI Bursts with Optional TRDY# Connection
- Integrated Low Skew Host Bus Clock Driver for Cost and Board Space Savings
- PCIset Operates Synchronous to the CPU and PCI Clocks
- Byte Parity Support for the Host and Main Memory Buses
  - Optional Parity on the Second Level Cache

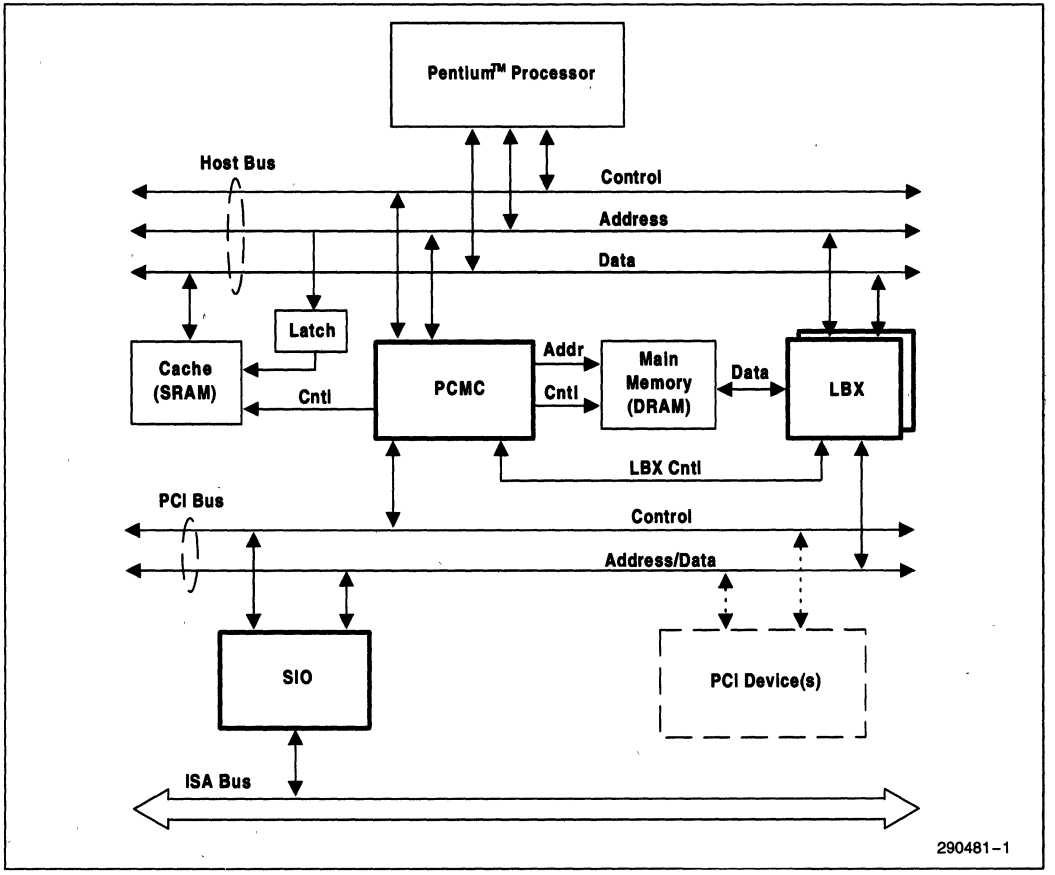
The 82430LX/82430NX PCIsets provide the Host/PCI bridge, cache/main memory controller, and an I/O subsystem core (either PCI/EISA or PCI/ISA bridge) for the next generation of high-performance personal computers based on the Pentium processor. System designers can take advantage of the power of the PCI Local bus for the local I/O while maintaining access to the large base of EISA and ISA expansion cards, and corresponding software applications. Extensive buffering and buffer management within the bridges ensures maximum efficiency in all three bus environments (Host CPU, PCI, and EISA/ISA Buses).

The 82430LX PCIset consists of the 82434LX PCI/Cache Memory Controller (PCMC) and the 82433LX Local Bus Accelerator (LBX) components, plus, either a PCI/ISA bridge or a PCI/EISA bridge. The PCMC and LBX provide the core cache and main memory architecture and serve as the Host/PCI bridge. For an ISA-based system, the 82430LX PCIset includes the 82378ZB System I/O (SIO) component as the PCI/ISA bridge. For an EISA-based system, the 82430LX PCIset includes the 82375EB/SB PCI/EISA Bridge (PCEB) and the 82374EB/SB EISA System Component (ESC). The PCEB and ESC work in tandem to form the complete PCI/EISA bridge. Both the ISA and EISA-based systems are shown on the following pages.

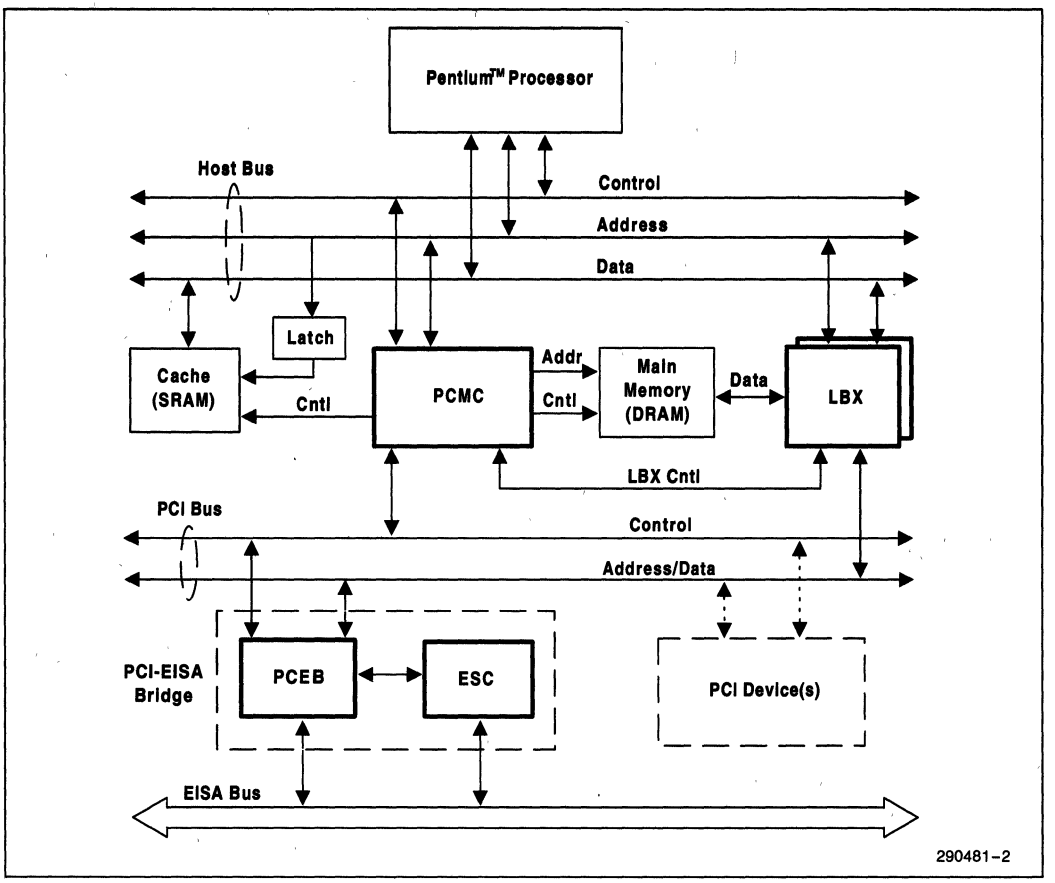
The 82430NX PCIset consists of the 82434NX PCI/Cache Memory Controller (PCMC) and the 82433NX Local Bus Accelerator (LBX) components, plus, either a PCI/ISA bridge or a PCI/EISA bridge. For an ISA-based system, the 82430NX PCIset includes the 82378ZB System I/O (SIO) component as the PCI/ISA bridge. For the DP ISA based system, the 82430NX PCIset includes the 82379AB. For UP or DP EISA-based systems, the 82430NX PCIset includes the 82375EB/SB PCI/EISA Bridge (PCEB) and the 82374EB/SB EISA System Component (ESC).

This document describes both the 82430LX and 82430NX. Unshaded areas describe the 82434LX. Shaded areas, like this one, describe 82430NX operations that differ from the 82434LX.

November 1994
Order Number: 290481-005



82430LX or 82430NX PCIset ISA Block Diagram



82430LX or Uni-Processor 82430NX PCIset EISA Block Diagram

82496 CACHE CONTROLLER AND 82491 CACHE SRAM FOR USE WITH THE Pentium™ PROCESSOR

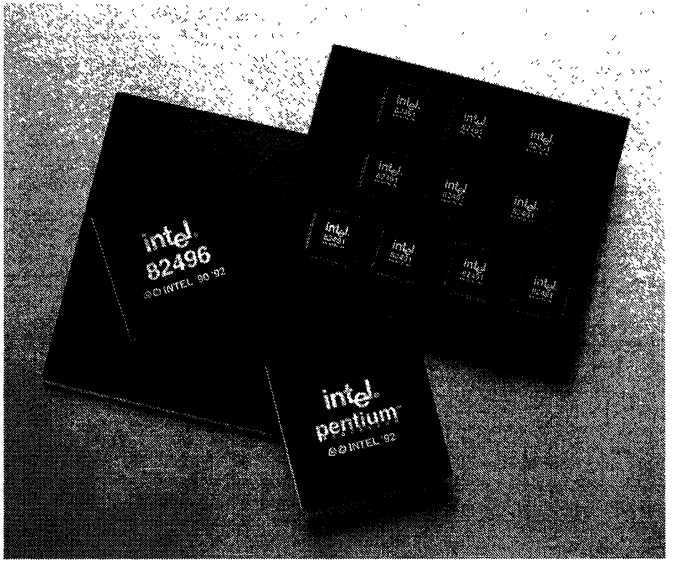
- **High Performance Second Level Cache**
  - **Zero Wait States at 66 MHz**
  - **Two-way Set Associative**
  - **Write-Back with MESI Protocol**
  - **Concurrent CPU Bus and Memory Bus Operation**
  - **Boundary Scan**
- **Pentium Processor**
  - **Chip Set Version of Pentium Processor**
  - **Superscalar Architecture**
  - **Enhanced Floating Point**
  - **On-chip 8K Code and 8K Data Caches**
  - **See Pentium™ Processor User's Manual Volume 2 for more information**
- **Highly Flexible**
  - **256K to 512K with parity**
  - **32, 64, or 128-Bit Wide Memory Bus**
  - **Synchronous, Asynchronous, and Strobed Memory Bus Operation**
  - **Selectable Bus Widths, Line Sizes, Transfers, and Burst Orders**
- **Full Multiprocessing Support**
  - **Concurrent CPU, Memory Bus, and Snoop Operations**
  - **Complete MESI Protocol**
  - **Internal/External Parity Generation/ Checking**
  - **Supports Read-for Ownership, Write-Allocation, and Cache-to-Cache Transfers**

2

The 82496 Cache Controller and multiple 82491 Cache SRAMs combine with the Pentium processor to form a CPU Cache chip set designed for high performance servers and function-rich desktops. The high speed interconnect between the CPU and cache components has been optimized to provide zero-wait state operation. This CPU Cache chip set is fully compatible with existing software, and has new data integrity features for mission critical applications.

The 82496 cache controller implements the MESI write-back protocol for full multiprocessing support. Dual ported buffers and registers allow the 82496 to concurrently handle CPU bus, memory bus, and internal cache operation for maximum performance.

The 82491 is a customized high-performance SRAM that supports 32, 64, and 128-bit wide memory bus widths, 16, 32, and 64 byte line sizes, and optional sectoring. The data path between the CPU bus and memory bus is separated by the 82491, allowing the CPU bus to handshake synchronously, asynchronously, or with a strobed protocol, and allowing concurrent CPU bus and memory bus operations.



241814-1

October 1993
Order Number: 241814-001

82497 CACHE CONTROLLER AND 82492 CACHE SRAM

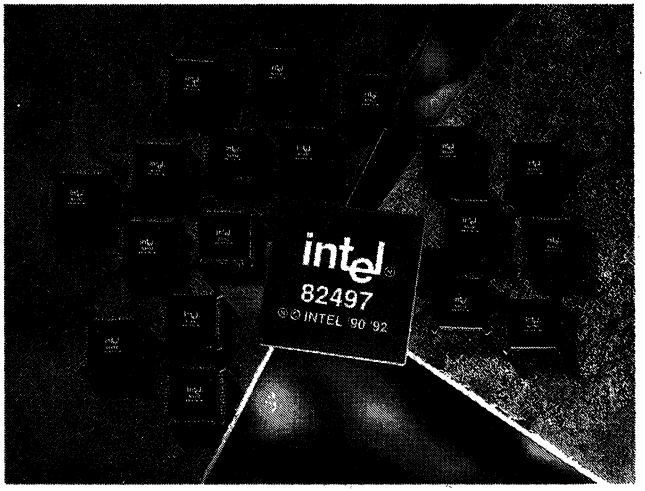
For Use with the Pentium™ Processor (735\90, 815\100)

- **High Performance Second Level Cache**
  - **Zero Wait States at 66 MHz**
  - **Two-Way Set Associative**
  - **Writeback with MESI Protocol**
  - **Concurrent CPU Bus and Memory Bus Operation**
  - **Boundary Scan**
- **Pentium™ Processor (735\90, 815\100)**
  - **Chip Set Version of Pentium™ Processor (735\90, 815\100)**
  - **Superscalar Architecture**
  - **Enhanced Floating Point**
  - **On-Chip 8K Code and 8K Data Caches**
  - **See *Pentium™ Processor Family Data Book* for More Information**
- **Highly Flexible**
  - **256K to 512K with Parity**
  - **32-, 64-, or 128-Bit Wide Memory Bus**
  - **Synchronous, Asynchronous and Strobed Memory Bus Operation**
  - **Selectable Bus Widths, Line Sizes, Transfers and Burst Orders**
- **Full Multiprocessing Support**
  - **Concurrent CPU, Memory Bus and Snoop Operations**
  - **Complete MESI Protocol**
  - **Internal/External Parity Generation/ Checking**
  - **Supports Read For Ownership, Write-Allocation and Cache-to-Cache Transfers**

The 82497 Cache Controller and multiple 82492 Cache SRAMs combine with the Pentium processor (735\90, 810\100) to form a CPU Cache chip set designed for high performance servers and function-rich desktops. The high-speed interconnect between the CPU and cache components has been optimized to provide zero-wait state operation. This CPU Cache chip set is fully compatible with existing software, and has new data integrity features for mission critical applications.

The 82497 cache controller implements the MESI write-back protocol for full multiprocessing support. Dual ported buffers and registers allow the 82497 to concurrently handle CPU bus, memory bus, and internal cache operation for maximum performance.

The 82492 is a customized high-performance SRAM that supports 32-, 64-, 128-bit wide memory bus widths, 16-, 32-, and 64-byte line sizes, and optional sectoring. The data path between the CPU bus and memory bus is separated by the 82492, allowing the CPU bus to handshake synchronously, asynchronously, or with a strobed protocol, and allowing concurrent CPU bus and memory bus operations.



242425-1

The complete document for this product is available on Intel's "Data-on-Demand" CD-ROM product. Contact your local Intel field sales office, Intel technical distributor, or call 1-800-548-4725.

November 1994
Order Number: 242425-001

82498 CACHE CONTROLLER AND 82493 CACHE SRAM

For use with the Pentium™ Processor (735/90, 815/100)

- **High Performance Second Level Cache**
  - **Zero Wait States at 66 MHz**
  - **Two-Way Set Associative**
  - **Writeback with MESI Protocol**
  - **Concurrent CPU Bus and Memory Bus Operation**
  - **Boundary Scan**
- **Pentium™ Processor (735/90, 815/100)**
  - **Chip Set Version of Pentium™ Processor (735/90, 815/100)**
  - **Superscalar Architecture**
  - **Enhanced Floating Point**
  - **On-Chip 8K Code and 8K Data Caches**
  - **See *Pentium™ Processor Family Data Book* for More Information**
- **Highly Flexible**
  - **1 Mbyte to 2 Mbyte**
  - **64-, or 128-Bit Wide Memory Bus**
  - **Synchronous, Asynchronous and Strobed Memory Bus Operation**
  - **Selectable Bus Widths, Line Sizes, Transfers and Burst Orders**
- **Full Multiprocessing Support**
  - **Concurrent CPU, Memory Bus and Snoop Operations**
  - **Complete MESI Protocol**
  - **Internal/External Parity Generation/Checking**
  - **Supports Read For Ownership, Write-Allocation and Cache-to-Cache Transfers**

2

The 82498 Cache Controller and multiple 82493 Cache SRAMs combine with the Pentium processor (735/90, 815/100) and future Pentium Processors to form a CPU Cache chip set designed for high performance servers and function-rich desktops. The high-speed interconnect between the CPU and cache components has been optimized to provide zero-wait state operation. This CPU Cache chip set is fully compatible with existing software, and has new data integrity features for mission critical applications.

The 82498 Cache Controller implements the MESI write-back protocol for full multiprocessing support. Dual ported buffers and registers allow the 82498 to concurrently handle CPU bus, memory bus, and internal cache operation for maximum performance.

The 82493 is a customized high-performance SRAM that supports 64-, and 128-bit wide memory bus widths, 32-, and 64-byte line sizes, and optional sectoring. The data path between the CPU bus and memory bus is separated by the 82493, allowing the CPU bus to handshake synchronously, asynchronously, or with a strobed protocol, and allowing concurrent CPU bus and memory bus operations.

The complete document for this product is available on Intel's "Data-on-Demand" CD-ROM product. Contact your local Intel field sales office, Intel technical distributor, or call 1-800-548-4725.

November 1994
Order Number: 242426-001

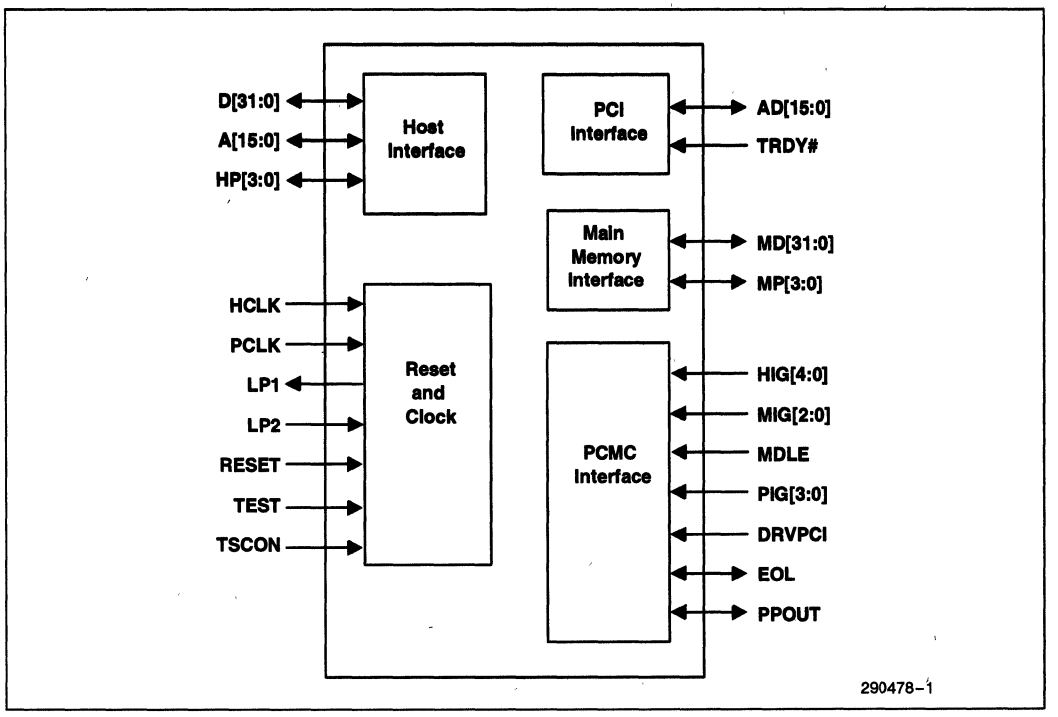
82433LX/82433NX
LOCAL BUS ACCELERATOR (LBX)

- **Supports the Full 64-bit Pentium™ Processor Data Bus at Frequencies up to 66 MHz (82433LX and 82433NX)**
- **Drives 3.3V Signal Levels on the CPU Data and Address Buses (82433NX)**
- **Provides a 64-Bit Interface to DRAM and a 32-Bit Interface to PCI**
- **Five Integrated Write Posting and Read Prefetch Buffers Increase CPU and PCI Performance**
  - **— CPU-to-Memory Posted Write Buffer 4 Qwords Deep**
  - **— PCI-to-Memory Posted Write Buffer Two Buffers, 4 Dwords Each**
  - **— PCI-to-Memory Read Prefetch Buffer 4 Qwords Deep**
  - **— CPU-to-PCI Posted Write Buffer 4 Dwords Deep**
  - **— CPU-to-PCI Read Prefetch Buffer 4 Dwords Deep**
- **CPU-to-Memory and CPU-to-PCI Write Posting Buffers Accelerate Write Performance**
- **Dual-Port Architecture Allows Concurrent Operations on the Host and PCI Buses**
- **Operates Synchronously to the CPU and PCI Clocks**
- **Supports Burst Read and Writes of Memory from the Host and PCI Buses**
- **Sequential CPU Writes to PCI Converted to Zero Wait-State PCI Bursts with Optional TRDY# Connection**
- **Byte Parity Support for the Host and Memory Buses**
  - **— Optional Parity Generation for Host to Memory Transfers**
  - **— Optional Parity Checking for the Secondary Cache**
  - **— Parity Checking for Host and PCI Memory Reads**
  - **— Parity Generation for PCI to Memory Writes**
- **160-Pin QFP Package**

Two 82433LX or 82433NX Local Bus Accelerator (LBX) components provide a 64-bit data path between the host CPU/Cache and main memory, a 32-bit data path between the host CPU bus and PCI Local Bus, and a 32-bit data path between the PCI Local Bus and main memory. The dual-port architecture allows concurrent operations on the host and PCI Buses. The LBXs incorporate three write posting buffers and two read prefetch buffers to increase CPU and PCI performance. The LBX supports byte parity for the host and main memory buses. The 82433NX is intended to be used with the 82434NX PCI/Cache/Memory Controller (PCMC). The 82433LX is intended to be used with the 82434LX PCMC. During bus operations between the host, main memory and PCI, the PCMC commands the LBXs to perform functions such as latching address and data, merging data, and enabling output buffers. Together, these three components form a "Host Bridge" that provides a full function dual-port data path interface, linking the host CPU and PCI bus to main memory.

This document describes both the 82433LX and 82433NX. Shaded areas, like this one, describe the 82433NX operations that differ from the 82433LX.

December 1994
Order Number: 290478-004



LBX Simplified Block Diagram

82433LX/82433NX
LOCAL BUS ACCELERATOR (LBX)

CONTENTS

CONTENTS

1.0 ARCHITECTURAL OVERVIEW

The 82430 PCIset consists of the 82434LX PCMC and 82433LX LBX components plus either a PCI/ISA bridge or a PCI/EISA bridge. The 82430NX PCIset consists of the 82434NX PCMC and 82433NX LBX components plus either a PCI/ISA bridge or a PCI/EISA bridge. The PCMC and LBX provide the core cache and main memory architecture and serves as the Host/PCI bridge. An overview of the PCMC follows the system overview section.

The Local Bus Accelerator (LBX) provides a high performance data and address path for the 82430LX/82430NX PCIset. The LBX incorporates five integrated buffers to increase the performance of the Pentium processor and PCI master devices. Two LBXs in the system support the following areas:

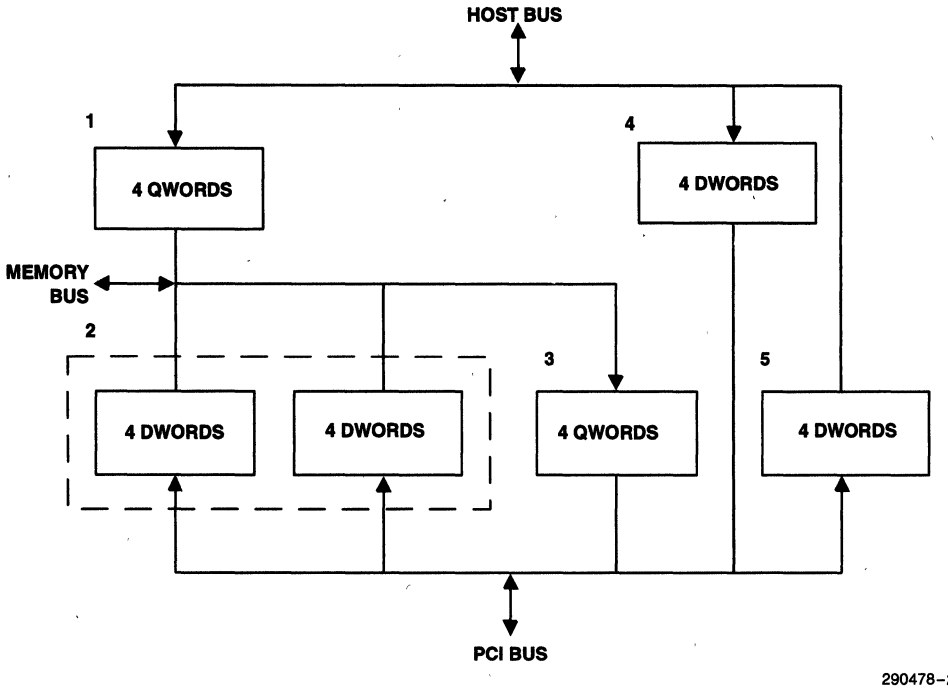
1. 64-bit data and 32-bit address bus of the Pentium processor.
2. 32-bit multiplexed address/data bus of PCI.
3. 64-bit data bus of the main memory.

In addition, the LBXs provide parity support for the three areas noted above (discussed further in Section 1.4).

1.1 Buffers in the LBX

The LBX components have five integrated buffers designed to increase the performance of the Host and PCI Interfaces of the 82430LX/82430NX PCIset.

With the exception of the PCI-to-Memory write buffer and the CPU-to-PCI write buffer, the buffers in the LBX store data only, addresses are stored in the PCMC component.



NOTES:
1. **CPU-to-Memory Posted Write Buffer:** This buffer is 4 Qwords deep, enabling the Pentium processor to write back a whole cache line in 4-1-1-1 timing, a total of 7 CPU clocks.
2. **PCI-to-Memory Posted Write Buffer:** A PCI master can post two consecutive sets of 4 Dwords (total of one cache line) or two single non-consecutive transactions.
3. **PCI-to-Memory Read Prefetch Buffer:** A PCI master to memory read transaction will cause this prefetch buffer to read up to 4 Qwords of data from memory, allowing up to 8 Dwords to be read onto PCI in a single burst transaction.
4. **CPU-to-PCI Posted Write Buffer:** The Pentium processor can post up to 4 Dwords into this buffer. The TRDY# connect option allows zero-wait state burst writes to PCI, making this buffer especially useful for graphic write operations.
5. **CPU-to-PCI Read Prefetch Buffer:** This prefetch buffer is 4 Dwords deep, enabling faster sequential Pentium processor reads when targeting PCI.

Figure 1. Simplified Block Diagram of the LBX Data Buffers

1.2 Control Interface Groups

The LBX is controlled by the PCMC via the control interface group signals. There are three interface groups: Host, Memory, and PCI. These control groups are signal lines that carry binary codes which the LBX internally decodes in order to implement specific functions such as latching data and steering data from PCI to memory. The control interfaces are described below.

1. **Host Interface Group:** These control signals are named HIG[4:0] and define a total of 29 (30 for the 82433NX) discrete commands. The PCMC sends HIG commands to direct the LBX to perform functions related to buffering and storing host data and/or address.
2. **Memory Interface Group:** These control signals are named MIG[2:0] and define a total of 7 discrete commands. The PCMC sends MIG commands to direct the LBX to perform functions related to buffering, storing, and retiring data to memory.
3. **PCI Interface Group:** These control signals are named PIG[3:0] and define a total of 15 discrete commands. The PCMC sends PIG commands to direct the LBX to perform functions related to buffering and storing PCI data and/or address.

1.3 System Bus Interconnect

The architecture of the 82430/82430NX PCIset splits the 64-bit memory and host data buses into logical halves in order to manufacture LBX devices with manageable pin counts. The two LBXs interface to the 32-bit PCI AD[31:0] bus with 16 bits each. Each LBX connects to 16 bits of the AD[31:0] bus and 32-bits of both the MD[0:63] bus and the D[0:63] bus. The lower order LBX (LBXL) connects to the low word of the AD[31:0] bus, while the high order LBX (LBXH) connects to the high word of the AD[31:0] bus.

Since the PCI connection for each LBX falls on 16-bit boundaries, each LBX does not simply connect to either the low Dword or high Dword of the Qword memory and host buses. Instead, the low order LBX buffers the first and third words of each 64-bit bus while the high order LBX buffers the second and fourth words of the memory and host buses.

As shown in Figure 2, LBXL connects to the first and third words of the 64-bit main memory and host data buses. The same device also drives the first 16 bits of the host address bus, A[15:0]. The LBXH device connects to the second and fourth words of the 64-bit main memory and host data buses. Correspondingly, LBXH drives the remaining 16 bits of the host address bus, A[31:16].

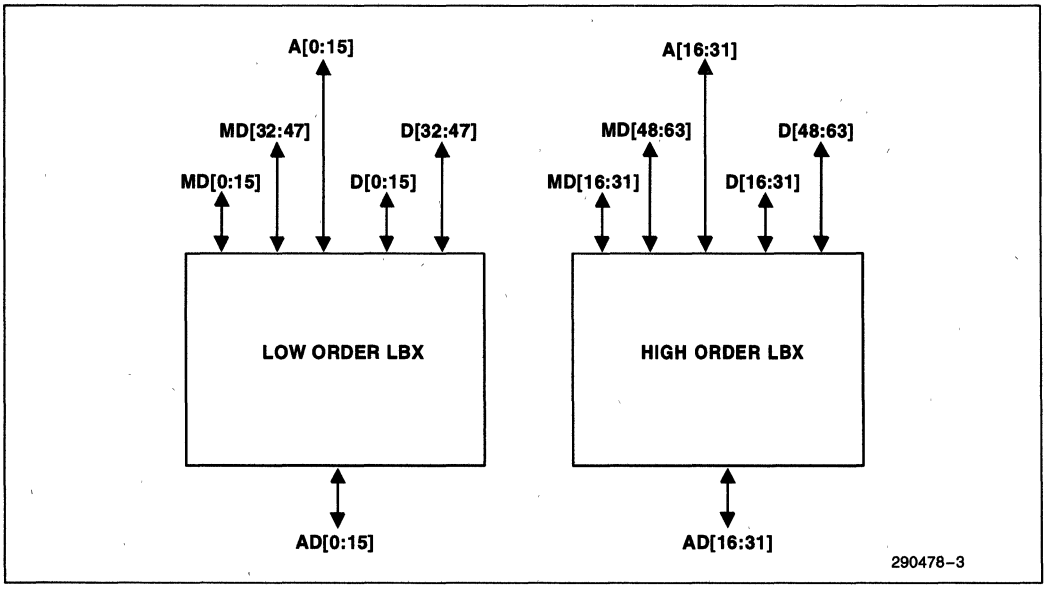


Figure 2. Simplified Interconnect Diagram of LBXs to System Buses

1.4 PCI TRDY# Interface

The PCI control signals do not interface to the LBXs, instead these signals connect to the 82434LX PCMC component. The main function of the LBXs PCI interface is to drive address and data onto PCI when the CPU targets PCI and to latch address and data when a PCI master targets main memory.

The TRDY# option provides the capability for zero-wait state performance on PCI when the Pentium processor performs sequential writes to PCI. This option requires that PCI TRDY# be connected to each LBX, for a total of two additional connections in the system. These two TRDY# connections are in addition to the single TRDY# connection that the PCMC requires.

1.5 Parity Support

The LBXs support byte parity on the host bus (CPU and second level cache) and main memory buses (local DRAM). The LBXs support parity during the address and data phases of PCI transactions to/from the host bridge.

2.0 SIGNAL DESCRIPTIONS

This section provides a detailed description of each signal. The signals (Figure 3) are arranged in functional groups according to their associated interface.

The '#' symbol at the end of a signal name indicates that the active, or asserted state occurs when the signal is at a low voltage level. When '#' is not present after the signal name, the signal is asserted when at the high voltage level.

The terms assertion and negation are used extensively. This is done to avoid confusion when working with a mixture of 'active-low' and 'active-high' signals. The term **assert**, or **assertion** indicates that a signal is active, independent of whether that level is represented by a high or low voltage. The term **negate**, or **negation** indicates that a signal is inactive.

The following notations are used to describe the signal type.

in Input is a standard input-only signal.

out Totem Pole output is a standard active driver.

t/s Tri-State is a bi-directional, tri-state input/output pin.

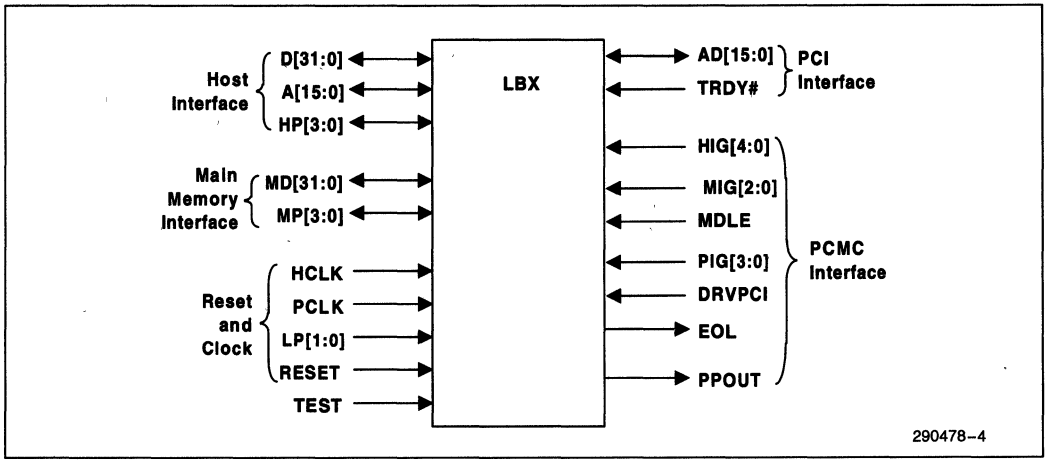


Figure 3. LBX Signals

2.1 Host Interface Signals

Signal	Type	Description
A[15:0]	t/s	**ADDRESS BUS:** The bi-directional A[15:0] lines are connected to the address lines of the host bus. The high order LBX (determined at reset time using the EOL signal) is connected to A[31:16], and the low order LBX is connected to A[15:0]. The host address bus is common with the Pentium processor, second level cache, PCMC and the two LBXs. During CPU cycles A[31:3] are driven by the CPU and A[2:0] are driven by the PCMC, all are inputs to the LBXs. During inquire cycles the LBX drives the PCI master address onto the host address lines A[31:0]. This snoop address is driven to the CPU and the PCMC by the LBXs to snoop L1 and the integrated second level tags, respectively. During PCI configuration cycles bound for the PCMC, the LBXs will send or receive the configuration data to/from the PCMC by copying the host data bus to/from the host address bus. The LBX drives both halves of the Qword host data bus with data from the 32-bit address during PCMC configuration read cycles. The LBX drives the 32-bit address with either the low Dword or the high Dword during PCMC configuration write cycles. In the 82433NX, these pins contain weak internal pull-down resistors. The high order 82433NX LBX samples A11 at the falling edge of reset to configure the LBX for PLL test mode. When A11 is sampled low, the LBX is in normal operating mode. When A11 is sampled high, the LBX drives the internal HCLK from the PLL on the EOL pin. Note that A11 on the high order LBX is connected to the A27 line on the CPU address bus. This same address line is used to put the PCMC into PLL test mode.
D[31:0]	t/s	**HOST DATA:** The bi-directional D[31:0] lines are connected to the data lines of the host data bus. The high order LBX (determined at reset time using the EOL signal) is connected to the host data bus D[63:48] and D[31:16] lines, and the low order LBX is connected to the host data bus D[47:32] and D[15:0] lines. In the 82433LX, these pins contain weak internal pull-up resistors. In the 82433NX, these pins contain weak internal pull-down resistors.
HP[3:0]	t/s	**HOST DATA PARITY:** HP[3:0] are the bi-directional byte parity signals for the host data bus. The low order parity bit HP[0] corresponds to D[7:0] while the high order parity bit HP[3] corresponds to D[31:24]. The HP[3:0] signals function as parity inputs during write cycles and as parity outputs during read cycles. Even parity is supported and the HP[3:0] signals follow the same timings as D[31:0]. In the 82433LX, these pins contain weak internal pull-up resistors. In the 82433NX, these pins contain weak internal pull-down resistors.

2.2 Main Memory (Dram) Interface Signals

Signal	Type	Description
MD[31:0]	t/s	**MEMORY DATA BUS:** MD[31:0] are the bi-directional data lines for the memory data bus. The high order LBX (determined at reset time using the EOL signal) is connected to the memory data bus MD[63:48] and MD[31:16] lines, and the low order LBX is connected to the memory data bus MD[47:32] and MD[15:0] lines. The MD[31:0] signals drive data destined for either the host data bus or the PCI bus. The MD[31:0] signals input data that originated from either the host data bus or the PCI bus. These pins contain weak internal pull-up resistors.
MP[3:0]	t/s	**MEMORY PARITY:** MP[3:0] are the bi-directional byte enable parity signals for the memory data bus. The low order parity bit MP[0] corresponds to MD[7:0] while the high order parity bit MP[3] corresponds to MD[31:24]. The MP[3:0] signals are parity outputs during write cycles to memory and parity inputs during read cycles from memory. Even parity is supported and the MP[3:0] signals follow the same timings as MD[31:0]. These pins contain weak internal pull-up resistors.

2.3 PCI Interface Signals

Signal	Type	Description
AD[15:0]	t/s	**ADDRESS AND DATA:** AD[15:0] are bi-directional data lines for the PCI bus. The AD[15:0] signals sample or drive the address and data on the PCI bus. The high order LBX (determined at reset time using the EOL signal) is connected to the PCI bus AD[31:16] lines, and the low order LBX is connected to the PCI AD[15:0] lines.
TRDY#	in	**TARGET READY:** TRDY# indicates the selected (targeted) device's ability to complete the current data phase of the bus operation. For normal operation, TRDY# is tied asserted low. When the TRDY# option is enabled in the PCMC (for zero wait-state PCI burst writes), TRDY# should be connected to the PCI bus.

2.4 PCMC Interface Signals

Signal	Type	Description
HIG[4:0]	in	**HOST INTERFACE GROUP:** These signals are driven from the PCMC and control the host interface of the LBX. The 82433LX decodes the binary pattern of these lines to perform 29 unique functions (30 for the 83433NX). These signals are synchronous to the rising edge of HCLK.
MIG[2:0]	in	**MEMORY INTERFACE GROUP:** These signals are driven from the PCMC and control the memory interface of the LBX. The LBX decodes the binary pattern of these lines to perform 7 unique functions. These signals are synchronous to the rising edge of HCLK.
PIG[3:0]	in	**PCI INTERFACE GROUP:** These signals are driven from the PCMC and control the PCI interface of the LBX. The LBX decodes the binary pattern of these lines to perform 15 unique functions. These signals are synchronous to the rising edge of HCLK.
MDLE	in	**MEMORY DATA LATCH ENABLE:** During CPU reads from DRAM, the LBX uses a clocked register to transfer data from the MD[31:0] and MP[3:0] lines to the D[31:0] and HP[3:0] lines. MDLE is the clock enable for this register. Data is clocked into this register when MDLE is asserted. The register retains its current value when MDLE is negated. During CPU reads from main memory, the LBX tri-states the D[31:0] and HP[3:0] lines on the rising edge of MDLE when HIG[4:0] = NOPC.
DRVPCI	in	**DRIVE PCI BUS:** This signals enables the LBX to drive either address or data information onto the PCI AD[15:0] lines.

2.4 PCMC Interface Signals (Continued)

Signal	Type	Description
EOL	t/s	**End Of Line:** This signal is asserted when a PCI master read or write transaction is about to overrun a cache line boundary. The low order LBX will have this pin connected to the PCMC (internally pulled up in the PCMC). The high order LBX connects this pin to a pull-down resistor. With one LBX EOL line being pulled down and the other LBX EOL pulled up, the LBX samples the value of this pin on the negation of the RESET signal to determine if it's the high or low order LBX.
PPOUT	t/s	**LBX PARITY:** This signal reflects the parity of the 16 AD lines driven from or latched into the LBX, depending on the command driven on PIG[3:0]. The PCMC uses PPOUT from both LBXs (called PPOUT[1:0]) to calculate the PCI parity signal (PAR) for CPU to PCI transactions during the address phase of the PCI cycle. The LBX uses PPOUT to check the PAR signal for PCI master transactions to memory during the address phase of the PCI cycle. When transmitting data to PCI the PCMC uses PPOUT to calculate the proper value for PAR. When receiving data from PCI the PCMC uses PPOUT to check the value received on PAR. If the L2 cache does not implement parity, the LBX will calculate parity so the PCMC can drive the correct value on PAR during L2 reads initiated by a PCI master. The LBX samples the PPOUT signal at the negation of reset and compares that state with the state of EOL to determine whether the L2 cache implements parity. The PCMC internally pulls down PPOUT[0] and internally pulls up PPOUT[1]. The L2 supports parity if PPOUT[0] is connected to the high order LBX and PPOUT[1] is connected to the low order LBX. The L2 is defined to not support parity if these connections are reversed, and for this case, the LBX will calculate parity. For normal operations either connection allows proper parity to be driven to the PCMC.

2.5 Reset and Clock Signals

Signal	Type	Description
HCLK	in	**HOST CLOCK:** HCLK is input to the LBX to synchronize command and data from the host and memory interfaces. This input is derived from a buffered copy of the PCMC HCLKx output.
PCLK	in	**PCI CLOCK:** All timing on the LBX PCI interface is referenced to the PCLK input. All output signals on the PCI interface are driven from PCLK rising edges and all input signals on the PCI interface are sampled on PCLK rising edges. This input is derived from a buffered copy of the PCMC PCLK output.
RESET	in	**RESET:** Assertion of this signal resets the LBX. After RESET has been negated the LBX configures itself by sampling the EOL and PPOUT pins. RESET is driven by the PCMC CPURST pin. The RESET signal is synchronous to HCLK and must be driven directly by the PCMC.
LP1	out	**LOOP 1:** Phase Lock Loop Filter pin. The filter components required for the LBX are connected to these pins.
LP2	in	**LOOP 2:** Phase Lock Loop Filter pin. The filter components required for the LBX are connected to these pins.
TEST	in	**TEST:** The TEST pin must be tied low for normal system operation.
TSCON	in	**TRI-STATE CONTROL:** This signal enables the output buffers on the LBX. This pin must be held high for normal operation. If TSCON is negated, all LBX outputs will tri-state.

3.0 FUNCTIONAL DESCRIPTION

3.1 LBX Post and Prefetch Buffers

This section describes the five write posting and read prefetching buffers implemented in the LBX. The discussion in this section refers to the operation of both LBXs in the system.

3.1.1 CPU-TO-MEMORY POSTED WRITE BUFFER

The write buffer is a queue 4 Qwords deep, it loads Qwords from the CPU and stores Qwords to memory. It is 4 Qwords deep to accommodate write-backs from the first or second level cache. It is organized as a simple FIFO. Commands driven on the HIG[4:0] lines store Qwords into the buffer, while commands on the MIG[2:0] lines retire Qwords from the buffer. While retiring Qwords to memory, the DRAM controller unit of the PCMC will assert the appropriate MA, CAS[7:0]#, and WE# signals. The PCMC keeps track of full/empty states, status of the data and address.

Byte parity for data to be written to memory is either propagated from the host bus or generated by the LBX. The LBX generates parity for data from the second level cache when the second level cache does not implement parity.

3.1.2 PCI-TO-MEMORY POSTED WRITE BUFFER

The buffer is organized as 2 buffers (4 Dwords each). There is an address storage register for each buffer. When an address is stored one of the two buffers is allocated and subsequent Dwords of data are stored beginning at the first location in that buffer. Buffers are retired to memory strictly in order, Qword at a time.

Commands driven on the PIG[3:0] lines post addresses and data into the buffer. Commands driven on HIG[4:0] result in addresses being driven on the host address bus. Commands driven on MIG[2:0] result in data being retired to DRAM.

For cases where the address targeted by the first Dword is odd, i.e. A[2] = 1, and the data is stored in an even location in the buffer, the LBX correctly aligns the Dword when retiring the data to DRAM. In other words the buffer is capable of retiring a Qword to memory where the data in the buffer is shifted by 1 Dword (Dword is position 0 shifted to 1, 1 shifted to 2 etc.). The DRAM controller of the PCMC asserts the correct CAS[7:0]# signals depending on the PCI C/BE[3:0]# signals stored in the PCMC for that Dword.

The End Of Line (EOL) signal is used to prevent PCI master writes from bursting past the cache line boundary. The device that provides "warning" to the PCMC is the low order LBX. This device contains the PCI master write low order address bits necessary to determine how many Dwords are left to the end of the line. Consequently, the LBX protocol uses the EOL signal from the low order LBX to provide this "end-of-line" warning to the PCMC, so that it may retry a PCI master write when it bursts past the cache line boundary. This protocol is described fully in Section 3.3.6.

The LBX calculates Dword parity on PCI write data, sending the proper value to the PCMC on PPOUT. The LBX generates byte parity on the MP signals for writing into DRAM.

3.1.3 PCI-TO-MEMORY READ PREFETCH BUFFER

This buffer is organized as a line buffer (4 Qwords) for burst transfers to PCI. The data is transferred into the buffer a Qword at a time and read out a Dword at a time. The LBX then effectively decouples the memory read rate from the PCI rate to increase concurrence.

Each new transaction begins by storing the first Dword in the first location in the buffer. The starting Dword for reading data out of the buffer onto PCI must be specified within a Qword boundary; that is the first requested Dword on PCI could be an even or odd Dword. If the snoop for a PCI master read results in a write-back from first or second level caches, this write back is sent directly to PCI and main memory. The following two paragraphs describe this process for cache line write-backs.

Since the write-back data from L1 is in linear order, writing into the buffer is straightforward. Only those Qwords to be transferred into PCI are latched into the PCI-to-memory read buffer. For example, if the address targeted by PCI is in the 3rd or 4th Qword in the line, the first 2 Qwords of write back data are discarded and not written into the read buffer. The primary cache write-back must always be written

completely to the CPU-to-Memory posted Write Buffer.

If the PCI master read data is read from the secondary cache, it is not written back to memory. Write-backs from the second level cache, when using burst SRAMs, are in Pentium processor burst order (the order depending on which Qword of the line is targeted by the PCI read). The buffer is directly addressed when latching second level cache write-back data to accommodate this burst order. For example, if the requested Qword is Qword 1, then the burst order is 1-0-3-2. Qword 1 is latched in buffer location 0, Qword 0 is discarded, Qword 3 is latched into buffer location 2 and Qword 2 is latched into buffer location 1.

Commands driven on MIG[2:0] and HIG[4:0] enter data into the buffer from the DRAM interface and the host interface (i.e. the caches), respectively. Commands driven on the PIG[3:0] lines drive data from the buffer onto the PCI AD[31:0] lines.

Parity driven on the PPOUT signal is calculated from the byte parity received on the host bus or the memory bus, whichever is the source. If the second level cache is the source of the data and does not implement parity, the parity driven on PPOUT is generated by the LBX from the second level cache data. If main memory is the source of the read data, PCI parity is calculated from the DRAM byte parity. Main memory must implement byte parity to guarantee correct PCI parity generation.

3.1.4 CPU-TO-PCI POSTED WRITE BUFFER

The CPU-to-PCI Posted Write Buffer is 4 Dwords deep. The buffer is constructed as a simple FIFO, with some performance enhancements. An address is stored in the LBX with each Dword of data. The structure of the buffer accommodates the packetization of writes to be burst on PCI. This is accomplished by effectively discarding addresses of data Dwords driven within a burst. Thus, while an address is stored for each Dword, an address is not necessarily driven on PCI for each Dword. The PCMC determines when a burst write may be performed based on consecutive addresses. The buffer also enables consecutive bytes to be merged within a single Dword, accommodating byte, word, and misaligned Dword string store and string move operations. Qword writes on the host bus are stored within the buffer as two individual Dword writes, with separate addresses.

The storing of an address with each Dword of data allows burst writes to be retried easily. In order to retry transactions, the FIFO is effectively "backed up" by one Dword. This is accomplished by making the FIFO physically one entry larger than it is logically. Thus, the buffer is physically 5 entries deep (an entry consists of an address and a Dword of data), while logically it is considered full when 4 entries have been posted. This design allows the FIFO to be backed up one entry when it is logically full.

Commands driven on HIG[4:0] post addresses and data into the buffer, and commands driven on PIG[3:0] retire addresses and data from the buffer and drive them onto the PCI AD[31:0] lines. As discussed previously, when bursting, not all addresses are driven onto PCI.

Data parity driven on the PPOUT signal is calculated from the byte parity received on the host bus. Address parity driven on PPOUT is calculated from the address received on the host bus.

3.1.5 CPU-TO-PCI READ PREFETCH BUFFER

This prefetch buffer is organized as a single buffer 4 Dwords deep. The buffer is organized as a simple FIFO. reads from the buffer are sequential; the buffer does not support random access of its contents. To support reads of less than a Dword the FIFO read pointer can function with or without a pre-increment. The pointer can also be reset to the first entry before a Dword is driven. When a Dword is read, it is driven onto both halves of the host data bus.

Commands driven on the HIG[4:0] lines enable read addresses to be sent onto PCI, the addresses are driven using PIG[3:0] commands. Read data is latched into the LBX by commands driven on the PIG[3:0] lines and the data is driven onto the host data bus using commands driven on the HIG[4:0] lines.

The LBX calculates Dword parity on PCI read data, sending the proper value to the PCMC on PPOUT. The LBX does not generate byte parity on the host data bus when the CPU reads PCI.

3.2 LBX Interface Command Descriptions

This section describes the functionality of the HIG, MIG and PIG commands driven by the PCMC to the LBXs.

3.2.1 HOST INTERFACE GROUP: HIG[4:0]

The Host Interface commands are shown in Table 1. These commands are issued by the host interface of the PCMC to the LBXs in order to perform the following functions:

- Reads from CPU-to-PCI read prefetch buffer when the CPU reads from PCI.
- Stores write-back data to PCI-to-memory read prefetch buffer when PCI read address results in a hit to a modified line in first or second level caches.
- Posts data to CPU-to-memory write buffer in the case of a CPU to memory write.
- Posts data to CPU-to-PCI write buffer in the case of a CPU to PCI write.
- Drives host address to Data lines and data to address lines for programming the PCMC configuration registers.

Table 1. HIG Commands

Command	Code	Description
NOPC	00000b	No Operation on CPU Bus
CMR	11100b	CPU Memory Read
CPRF	00100b	CPU Read First Dword from CPU-to-PCI Read Prefetch Buffer
CPRA	00101b	CPU Read Next Dword from CPU-to-PCI Read Prefetch Buffer, Toggle A
CPRB	00110b	CPU Read Next Dword from CPU-to-PCI Read Prefetch Buffer, Toggle B
CPRQ	00111b	CPU Read Qword from CPU-to-PCI Read Prefetch Buffer
SWB0	01000b	Store Write-Back Data Qword 0 to PCI-to-Memory Read Buffer
SWB1	01001b	Store Write-Back Data Qword 1 to PCI-to-Memory Read Buffer
SWB2	01010b	Store Write-Back Data Qword 2 to PCI-to-Memory Read Buffer
SWB3	01011b	Store Write-Back Data Qword 3 to PCI-to-Memory Read Buffer
PCMWQ	01100b	Post to CPU-to-Memory Write Buffer Qword
PCMWFQ	01101b	Post to CPU-to-Memory Write and PCI-to-Memory Read Buffer First Qword
PCMWNQ	01110b	Post to CPU-to-Memory Write and PCI-to-Memory Read Buffer Next Qword
PCPWL	10000b	Post to CPU-to-PCI Write Low Dword
MCP3L	10011b	Merge to CPU-to-PCI Write Low Dword 3 Bytes
MCP2L	10010b	Merge to CPU-to-PCI Write Low Dword 2 Bytes
MCP1L	10001b	Merge to CPU-to-PCI Write Low Dword 1 Byte
PCPWH	10100b	Post to CPU-to-PCI Write High Dword
MCP3H	10111b	Merge to CPU-to-PCI Write High Dword 3 Bytes
MCP2H	10110b	Merge to CPU-to-PCI Write High Dword 2 Bytes
MCP1H	10101b	Merge to CPU-to-PCI Write High Dword 1 Byte
LCPRAD	00001b	Latch CPU-to-PCI Read Address
DPRA	11000b	Drive Address from PCI A/D Latch to CPU Address Bus
DPWA	11001b	Drive Address from PCI-to-Memory Write Buffer to CPU Address Bus
ADCPY	11101b	Address to Data Copy in the LBX
DACPYH	11011b	Data to Address Copy in the LBX High Dword
DACPYL	11010b	Data to Address Copy in the LBX Low Dword
PSCD	01111b	Post Special Cycle Data
DRVFF	11110b	Drive FF..FF (All 1's) onto the Host Data Bus
PCPWHC	00011b	Post to CPU-to-PCI Write High Dword Configuration

NOTE:
All other patterns are reserved.

NOPC No Operation is performed on the host bus by the LBX hence it tri-states its host bus drivers.

CMR This command effectively drives DRAM data onto the host data bus. The LBX acts as a transparent latch in this mode, depending on MDLE for latch control. With the MDLE signal high the CMR command will cause the LBXs to buffer memory data onto the host bus. When MDLE is low. The LBX will drive onto the host bus whatever memory data that was latched when MDLE was negated.

CPRF This command reads the first Dword of the CPU-to-PCI read prefetch buffer. The read pointer of the FIFO is set to point to the first Dword. The Dword is driven onto the high and low halves of the host data bus.

CPRA This command increments the read pointer of the CPU-to-PCI read prefetch buffer FIFO and drives that Dword onto the host bus when it is driven after a CPRF or CPRB command. If driven after another CPRA command, the LBX drives the current Dword while the read pointer of the FIFO is not incremented. The Dword is driven onto the upper and lower halves of the host data bus.

CPRB This command increments the read pointer of the CPU-to-PCI read prefetch buffer FIFO and drives that Dword onto the host bus when it is driven after a CPRA command. If driven after another CPRB command, the LBX drives the current Dword while the read pointer of the FIFO is not incremented. The Dword is driven onto the upper and lower halves of the host data bus.

CPRQ This command drives the first Dword stored in the CPU-to-PCI read prefetch buffer onto the lower half of the host data bus, and drives the second Dword onto the upper half of the host data bus, regardless of the state of the read pointer. The read pointer is not affected by this command.

SWB0 This command stores a Qword from the host data lines into location 0 of the PCI-to-Memory Read Buffer. Parity is either generated for the data or propagated from the host bus based on the state of the PPOUT signals sampled at the negation of RESET when the LBXs were initialized.

SWB1 This command, (similar to SWB0), stores a Qword from the host data lines into location 1 of the PCI-to-Memory Read Buffer. Parity is either generated from the data or propagated from the host bus based on the state of the PPOUT signal sampled at the falling edge of RESET.

SWB2 This command, (similar to SWB0), stores a Qword written back from the first or second level cache into location 2 of the PCI-to-memory read buffer. Parity is either generated from the data or propagated from the host bus based on the state of the PPOUT signal sampled at the falling edge of RESET.

SWB3 This command stores a Qword from the host data lines into location 3 of the PCI-to-Memory Read Buffer. Parity is either generated for the data or propagated from the host bus based on the state of the PPOUT signal sampled at the falling edge of RESET.

PCMWQ This command posts one Qword of data from the host data lines to CPU-to-Memory Write Buffer in case of a CPU memory write or a write-back from the second level cache.

PCMWFQ If the PCI Memory read address leads to a hit on a modified line in the first level cache, then a write-back is scheduled and this data has to be written into the CPU-to-Memory Write Buffer and PCI-to-Memory Read Buffer at the same time. The write-back of the first Qword is done by this command to both the buffers.

PCMWNQ This command follows the previous command to store or post subsequent write-back Qwords.

PCPWL This command posts the low Dword of a CPU-to-PCI write. The CPU-to-PCI Write Buffer stores a Dword of PCI address for every Dword of data. Hence, this command also stores the address of the Low Dword in the address location for the data. Address bit 2 (A2) is not stored directly. This command assumes a value of 0 for A2 and this is what is stored.

MCP3L This command merges the 3 most significant bytes of the low Dword of the host data bus into the last Dword posted to the CPU-to-PCI write buffer. The address is not modified.

MCP2L This command merges the 2 most significant bytes of the low Dword of the host data bus into the last Dword posted to the CPU-to-PCI write buffer. The address is not modified.

MCP1L This command merges the most significant byte of the low Dword of the host data bus into the last Dword posted to the CPU-to-PCI write buffer. The address is not modified.

PCPWH This command posts the upper Dword of a CPU-to-PCI write, with its address, into the address location. Hence, to do a Qword write PCPWL has to be followed by a PCPWH. Address bit 2 (A2) is not stored directly. This command forces a value of 1 for A2 and this is what is stored.

MCP3H This command merges the 3 most significant bytes of the high Dword of the host data bus into the last Dword posted to the CPU-to-PCI Write Buffer. The address is not modified.

MCP2H This command merges the 2 most significant bytes of the high Dword of the host data bus into the last Dword posted to the CPU-to-PCI Write Buffer. The address is not modified.

MCP1H This command merges the most significant byte of the high Dword of the host data bus into the last Dword posted to the CPU-to-PCI Write Buffer. The address is not modified.

LCPRAD This command latches the host address to drive on PCI for a CPU-to-PCI read. It is necessary to latch the address in order to drive inquire addresses on the host address bus before the CPU address is driven onto PCI.

DPRA The PCI memory read address is latched in the PCI A/D latch by a PIG command LCPRAD, this address is driven onto the host address bus by DPRA. Used in PCI to memory read transaction.

DPWA The DPWA command drives the address of the current PCI Master Write Buffer onto the host address bus. This command is potentially driven for multiple cycles. When it is no longer driven, the read pointer will increment to point to the next buffer, and a subsequent DPWA command will read the address from that buffer.

ADCPY This command drives the host data bus with the host address. The address is copied on the high and low halves of the Qword data bus; i.e. A[31:0] is copied onto D[31:0] and D[63:32]. This command is used when the CPU writes to the PCMC configuration registers.

DACPYH This command drives the host address bus with the high Dword of host data. This command is used when the CPU writes to the PCMC configuration registers.

DACPYL This command drives the host address bus with the low Dword of host data. This command is used when the CPU writes to the PCMC configuration registers.

PSCD This command is used to post the value of the Special Cycle code into the CPU-to-PCI Posted Write Buffer. The value is driven onto the A[31:0] lines by the PCMC, after acquiring the address bus by asserting AHOLD. The value on the A[31:0] lines is posted into the DATA location in the CPU-to-PCI Posted Write Buffer.

DRVFF This command causes the LBX to drive all "1s" (i.e. FFFFFFFFh) onto the host data bus. It is used for CPU reads from PCI that terminate with master abort.

PCPWHC This command posts the high half of the CPU data bus. The LBXs post the high half of the data bus even if A2 from the PCMC is low. This command is used during configuration writes when using PCI configuration access mechanism #1.

3.2.2 MEMORY INTERFACE GROUP: MIG[2:0]

The Memory Interface commands are shown in Table 2. These commands are issued by the DRAM controller of the PCMC to perform the following functions:

- Retires data from CPU-to-Memory Write Buffer to DRAM.
- Stores data into PCI-to-Memory Read Buffer when the PCI read address is targeted to DRAM.
- Retires PCI-to-Memory Write Buffer to DRAM.

Table 2. MIG Commands

Command	Code	Description
NOPM	000b	No Operation on Memory Bus
PMRFQ	001b	Place into PCI-to-Memory Read Buffer First Qword
PMRNQ	010b	Place into PCI-to-Memory Read Buffer Next Qword
RCMWQ	100b	Retire CPU-to-Memory Write Buffer Qword
RPMWQ	101b	Retire PCI-to-Memory Write Buffer Qword
RPMWQS	110b	Retire PCI-to-Memory Write Buffer Qword Shifted
MEMDRV	111b	Drive Latched Data Onto Memory Bus for 1 Clock Cycle

NOTE:
All other patterns are reserved.

NOPMN Operation on the memory bus. The LBX tri-states its drivers driving the memory bus.

PMRFQ The PCI-to-Memory read address targets memory if there is a miss on first and second caches. This command stores the first Qword of data starting at the first location in the buffer. This buffer is 8 Dwords or 1 cache line deep.

PMRNQ This command stores subsequent Qwords from memory starting at the next available location in the PCI-to-Memory Read Buffer. It is always used after PMRFQ.

RCMWQ This command retires one Qword from the CPU-to-Memory Write Buffer to DRAM. The address is stored in the address queue for this buffer in the PCMC.

RPMWQ This command retires one Qword of data from one line of the PCI-to-Memory write buffer to DRAM. When all the valid data in one buffer is retired, the next RPMWQ (or RPMWQS) will read data from the next buffer.

RPMWQS This command retires one Qword of data from one line of PCI-to-Memory write buffer to DRAM. For this command the data in the buffer is shifted by one Dword (Dword in position 0 is shifted to 1, 1 to 2 etc.). This is because the address targeted by the first Dword of the write could be an odd Dword (i.e., address bit[2] is a 1). To retire a misaligned line this command has to be used for all the data in the buffer. When all the valid data in one buffer is retired, the next RPMWQ (or RPMWQS) will read data from the next buffer.

MEMDRV For a memory write operation the data on the memory bus is required for more than one clock cycle hence all DRAM retires are latched and driven to the memory bus in subsequent cycles by this command.

3.2.3 PCI INTERFACE GROUP: PIG[3:0]

The PCI Interface commands are shown in Table 3. These commands are issued by the PCI master/slave interface of the PCMC to perform the following functions:

- Slave posts address and data to PCI-to-Memory Write Buffer.
- Slave sends PCI-to-Memory read data on the AD bus.
- Slave latches PCI master memory address so that it can be gated to the host address bus.
- Master latches CPU-to-PCI read data from the AD bus.
- Master retires CPU-to-PCI write buffer.
- Master sends CPU-to-PCI address to the AD bus.

The PCI AD[31:0] lines are driven by asserting the signal DRVPCI. This signal is used for both master and slave transactions.

Parity is calculated on either the value being driven onto PCI or the value being received on PCI, depending on the command. In Table 3, the PAR column has been included to indicate the value that the PPOUT signals are based on. An "I" indicates that the PPOUT signals reflect the parity of the AD lines as inputs to the LBX. An "O" indicates that the PPOUT signals reflect the value being driven on the PCI AD lines. See Section 3.3.4 for the timing relationship between the PIG[3:0] command, the AD[31:0] lines, and the PPOUT signals.

Table 3. PIG Commands

Command	Code	PAR	Description
PPMWA	1000b	I	Post to PCI-to-Memory Write Buffer Address
PPMWD	1001b	I	Post to PCI-to-Memory Write Buffer Data
SPMRH	1101b	O	Send PCI Master Read Data High Dword
SPMRL	1100b	O	Send PCI Master Read Data Low Dword
SPMRN	1110b	O	Send PCI Master Read Data Next Dword
LCPRF	0000b	I	Latch CPU Read from PCI into Read Prefetch Buffer First Dword
LCPRA	0001b	I	Latch CPU Read from PCI into Prefetch Buffer Next Dword, A Toggle
LCPRB	0010b	I	Latch CPU Read from PCI into Prefetch Buffer Next Dword, B Toggle
DCPWA	0100b	O	Drive CPU-to-PCI Write Buffer Address
DCPWD	0101b	O	Drive CPU-to-PCI Write Buffer Data
DCPWL	0110b	O	Drive CPU-to-PCI Write Buffer Last Data
DCCPD	1011b	O	Discard Current CPU-to-PCI Write Buffer Data
BCPWR	1010b	O	Backup CPU-to-PCI Write Buffer for Retry
SCPA	0111b	O	Send CPU-to-PCI Address
LPMA	0011b	I	Latch PCI Master Address

NOTE:
All other patterns are reserved.

PPMWA This command selects a new buffer and places the PCI master address latch value into the address register for that buffer. The next PPMWD command posts write data in the first location of this newly selected buffer. This command also causes the EOL logic to decrement the count of Dwords remaining in the line.

PPMWD This command stores the value in the AD latch into the next data location in the currently selected buffer. This command also causes the EOL logic to decrement the count of Dwords remaining in the line.

SPMRH This command sends the high order Dword from the first Qword of the PCI-to-Memory Read Buffer onto PCI. This command also causes the EOL logic to decrement the count of Dwords remaining in the line.

SPMRL This command sends the low order Dword from the first Qword of the PCI-to-Memory Read Buffer onto PCI. This command also selects the Dword alignment for the transaction and causes the EOL logic to decrement the count of Dwords remaining in the line.

SPMRN This command sends the next Dword from the PCI-to-Memory Read Buffer onto PCI. This command also causes the EOL logic to decrement the count of Dwords remaining in the line. This command is used for the second and all subsequent Dwords of the current transaction.

LCPRF This command acquires the value of the AD[31:0] lines into the first location in the CPU-to-PCI Read Prefetch Buffer until a different command is driven.

LCPRA When driven after a LCPRF or LCPRB command, this command latches the value of the AD[31:0] lines into the next location into the CPU-to-PCI Read Prefetch Buffer. When driven after another LCPRA command, this command latches the value on AD[31:0] into the same location in the CPU-to-PCI Read Prefetch Buffer, overwriting the previous value.

LCPRB When driven after a LCPRA command, this command latches the value of the AD[31:0] lines into the next location into the CPU-to-PCI Read Prefetch Buffer. When driven after another LCPRB command, this command latches the value on AD[31:0] into the same location in the CPU-to-PCI Read Prefetch Buffer, overwriting the previous value.

DCPWA This command drives the next address in the CPU-to-PCI Write Buffer onto PCI. The read pointer of the FIFO is not incremented.

DCPWD This command drives the next data Dword in the CPU-to-PCI Write Buffer onto PCI. The read pointer of the FIFO is incremented on the next PCLK if TRDY# is asserted.

DCPWL This command drives the previous data Dword in the CPU-to-PCI Write Buffer onto PCI. This is the data which was driven by the last DCPWD command. The read pointer of the FIFO is not incremented.

DCCPD This command discards the current Dword in the CPU-to-PCI Write Buffer. This is used to clear write data when the write transaction terminates with master abort, where TRDY# is never asserted.

BCPWR For this command the CPU-to-PCI Write Buffer is "backed up" one entry such that the address/data pair last driven with the DCPWA and DCPWD commands will be driven again on the AD[31:0] lines when the commands are driven again. This command is used when the target has retried the write cycle.

SCPA This command drives the value on the host address bus onto PCI.

LPMA This command stores the previous AD[31:0] value into the PCI master address latch. If the EOL logic determines that the requested Dword is the last Dword of a line, then the EOL signal will be asserted; otherwise the EOL signal will be negated.

3.3 LBX Timing Diagrams

This section describes the timing relationship between the LBX control signals and the interface buses.

3.3.1 HIG[4:0] COMMAND TIMING

The commands driven on HIG[4:0] can cause the host address bus and/or the host data bus to be driven and latched. The following timing diagram illustrates the timing relationship between the driven command and the buses. The "host bus" in Figure 4 could be address and/or data.

Note that the Drive command takes two cycles to drive the host data bus, but only one to drive the address. When the NOPC command is sampled, the LBX takes only one cycle to release the host bus.

The Drive commands in Figure 4 are any of the following:

CMR	**CPRF**	**CPRA**	**CPRB**
CPRQ	**DPRA**	**DPWA**	**ADCPY**
DACPYH	**DACPYL**	**DRVFF**	

The Latch command in Figure 4 is any of the following:

SWB0	**SWB1**	**SWB2**	**SWB3**
PCMWQ	**PCMWFQ**	**PCMWNQ**	**PCPWL**
MCP3L	**MCP2L**	**MCP1L**	**PCPWH**
MCP3H	**MCP2H**	**LCPRAD**	**PSCD**

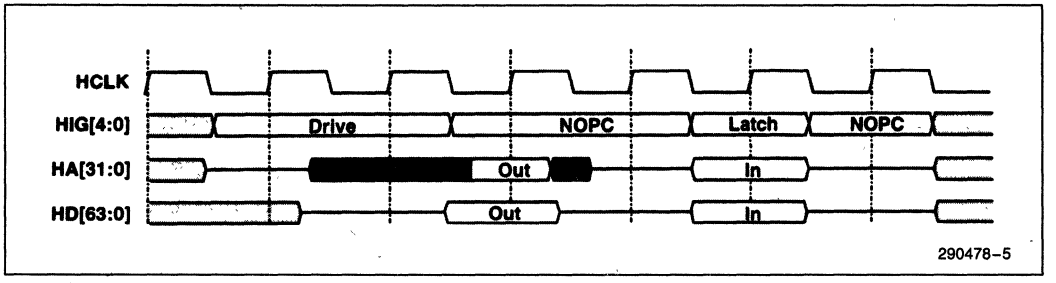


Figure 4. HIG[4:0] Command Timing

3.3.2 HIG[4:0] MEMORY READ TIMING

Figure 5 illustrates the timing relationship between the HIG[4:0], MIG[2:0], CAS[7:0]#, and MDLE signals for DRAM memory reads. The delays shown in the diagram do not represent the actual AC timings, but are intended only to show how the delay affects the sequencing of the signals.

When the CPU is reading from DRAM, the HIG[4:0] lines are driven with the CMR command that causes the LBX to drive memory data onto the HD bus. Until the MD bus is valid, the HD bus is driven with invalid data. When CAS[7:0]# assert, the MD bus becomes valid after the DRAM CAS[7:0]# access time. The MD and MP lines are directed through a synchronous register inside the LBX to the HD and HP lines. MDLE acts as a clock enable for this register. When MDLE is asserted, the LBX samples the MD and MP lines. When MDLE is negated, the MD and HD register retains its current value.

The LBX releases the HD bus based on sampling the NOPC command on the HIG[4:0] lines and MDLE being asserted. By delaying the release of the HD bus until MDLE is asserted, the LBX provides hold time for the data with respect to the write enable strobes (CWE[7:0]#) of the second level cache.

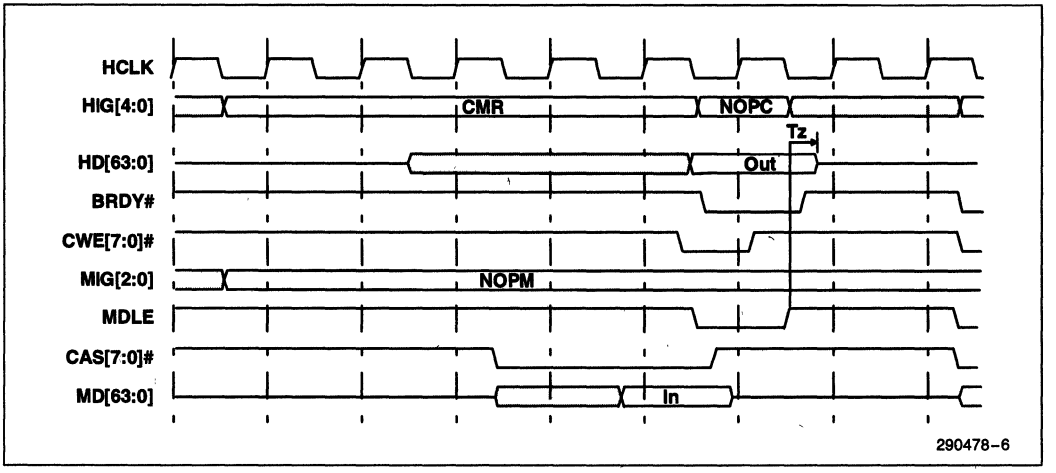


Figure 5. CPU Read from Memory

3.3.3 MIG[2:0] COMMAND

Figure 6 illustrates the timing of the MIG[2:0] commands with respect to the MD bus, CAS[7:0]#, and WE#. Figure 6 shows the MD bus transitioning from a read to a write cycle.

The Latch command in Figure 6 is any of the following:
PMRFQ PMRNQ

The Retire command in Figure 6 is any of the following:
RCMWQ RPMWQ RPMWQS

The data on the MD bus is sampled at the end of the first cycle into the LBX based on sampling the Latch command. The CAS[7:0]# signals can be negated in the next cycle. The WE# signal is asserted in the next cycle. The required delay between the assertion of WE# and the assertion of CAS[7:0]# means that the MD bus has 2 cycles to turn around; hence the NOPM command driven in the second clock. The LBX starts to drive the MD bus based on sampling the Retire command at the end of the third clock. After the Retire command is driven for 1 cycle, the data is held at the output by the MEMDRV command. The LBX releases the MD bus based on sampling the NOPM command at the end of the sixth clock.

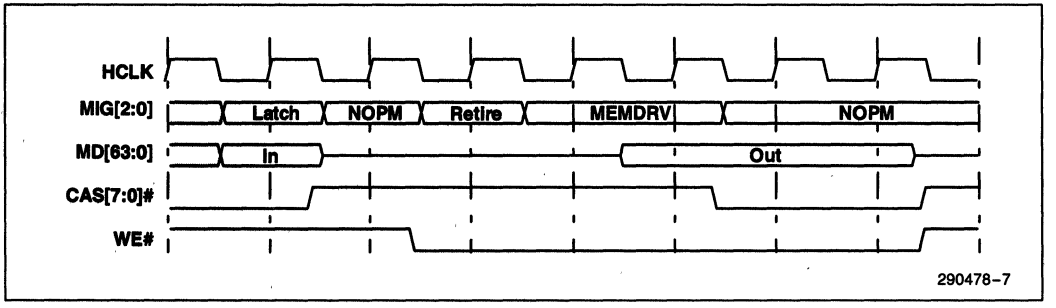


Figure 6. MIG[2:0] Command Timing

3.3.4 PIG[3:0] COMMAND, DRVPCI, AND PPOUT TIMING

Figure 7 illustrates the timing of the PIG[3:0] commands, the DRVPCI signal, and the PPOUT[1:0] signal relative to the PCI AD[31:0] lines.

The Drive commands in Figure 7 are any of the following:
SPMRH SPMRL SPMRN
DCPWA DCPWD DCPWL
SCPA

The Latch commands in Figure 7 are any of the following:
PPMWA PPMWD LPMA

The following commands do not fit in either category, although they function like Latch type commands with respect to the PPOUT[1:0] signals. They are described in Section 3.3.5.

LCPRF LCPRA LCPRB

The DRVPCI signal is driven synchronous to the PCI bus, enabling the LBXs to initiate driving the PCI AD[31:0] lines one clock after DRVPCI is asserted. As shown in Figure 7, if DRVPCI is asserted in cycle N, the PCI AD[31:0] lines are driven in cycle N+1. The negation of the DRVPCI signal causes the LBXs to asynchronously release the PCI bus, enabling the LBXs to cease driving the PCI AD[31:0] lines in the same clock that DRVPCI is negated. As shown in Figure 7, if DRVPCI is negated in cycle N, the PCI AD[31:0] lines are released in cycle N.

PCI address and data parity is available at the LBX interface on the PPOUT lines from the LBX. The parity for data flow from PCI to LBX is valid 1 clock cycle after data on the AD bus. The parity for data flow from LBX to PCI is valid in the same cycle as the data. When the AD[31:0] lines transition from input to output, there is no conflict on the parity lines due to the dead cycle for bus turnaround. This is illustrated in the sixth and seventh clock of Figure 7.

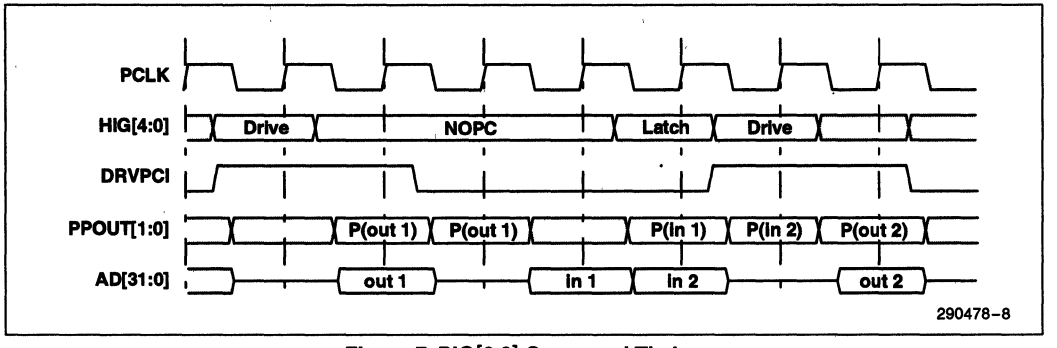


Figure 7. PIG[3:0] Command Timing

3.3.5 PIG[3:0]: READ PREFETCH BUFFER COMMAND TIMING

The structure of the CPU-to-PCI read prefetch buffer requires special considerations due to the partition of the PCMC and LBX. The PCMC interfaces only to the PCI control signals, while the LBXs interface only to the data. Therefore, it is not possible to latch a Dword of data into the prefetch buffer after it is qualified by TRDY#. Instead, the data is repetitively latched into the same location until TRDY# is sampled asserted. Only after TRDY# is sampled asserted is data valid in the buffer. A toggling mechanism is implemented to advance the write pointer to the next Dword after the current Dword has been qualified by TRDY#.

Other considerations of the partition are taken into account on the host side as well. When reading from the buffer, the command to drive the data onto the host bus is sent before it is known that the entry is valid. This method avoids the wait-state that would be introduced by waiting for an entry's TRDY# to be asserted before sending the command to drive the entry onto the host bus. The FIFO structure of the buffer also necessitates a toggling scheme to advance to the next buffer entry after the current entry has been successfully driven. Also, this method gives the LBX the ability to drive the same Dword twice, enabling reads of less than a Dword to be serviced by the buffer; reads of individual bytes of a Dword would read the same Dword 4 times.

The HIG[4:0] and PIG[3:0] lines are defined to enable the features described previously. The LCPRF PIG[3:0] command latches the first PCI read Dword into the first location in the CPU-to-PCI read prefetch buffer. This command is driven until TRDY# is sampled asserted. The valid Dword would then be in the first location of the buffer. The cycle after TRDY# is sampled asserted, the PCMC drives the LCPRA command on the PIG[3:0] lines. This action latches the value on the PCI AD[31:0] lines into the *next* Dword location in the buffer. Again, the LCPRA command is driven until TRDY# is sampled asserted. Each cycle the LCPRA command is driven, data is latched into the same location in the buffer. When TRDY# is sampled asserted, the PCMC drives the LCPRB command on the PIG[3:0] lines. This latches the value on the AD[31:0] lines into the next location in the buffer, the one *after* the location that the previous LCPRA command latched data into. After TRDY# has been sampled asserted again, the command switches back to LCPRA. In this way, the same location in the buffer can be filled repeatedly until valid, and when it is known that the location is valid, the next location can be filled.

The commands for the HIG[4:0], CPRF, CPRA, and CPRB, work exactly the same way. If the same command is driven, the same data is driven. Driving an appropriately different command results in the next data being driven. Figure 8 illustrates the usage of these commands.

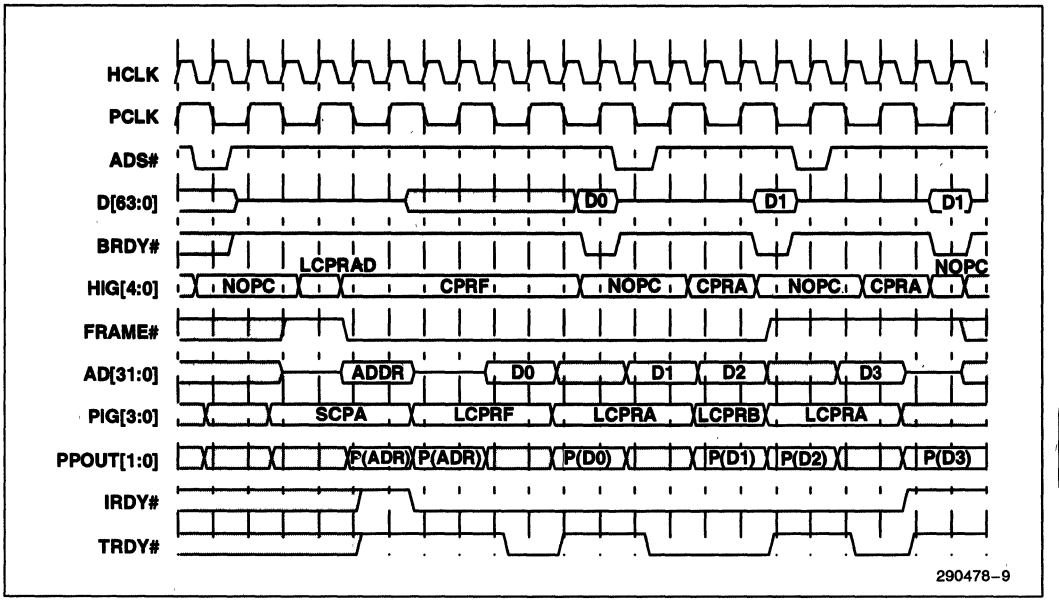


Figure 8. PIG[3:0] CPU-to-PCI Read Prefetch Buffer Commands

Figure 8 shows an example of how the PIG commands function on the PCI side. The LCPRF command is driven on the PIG[3:0] lines until TRDY# is sampled asserted at the end of the fifth PCI clock. The LCPRA command is then driven until TRDY# is again sampled asserted at the end of the seventh PCI clock. TRDY# is sampled asserted again so LCPRB is driven only once. Finally, LCPRA is driven again until the last TRDY# is asserted at the end of the tenth PCI clock. In this way, 4 Dwords are latched in the read CPU-to-PCI prefetch buffer.

Figure 8 also shows an example of how the HIG commands function on the host side of the LBX. Two clocks after sampling the CPRF command, the LBX drives the host data bus. The data takes two cycles to become stable. The first data driven in this case is invalid, since the data has not arrived on PCI. The data driven on the host bus changes in the seventh host clock, since the LCPRF command has been driven on the PIG[3:0] lines the previous cycle, latching a new value into the first location of the read prefetch buffer. At this point the data is not the correct value, since TRDY# has not yet been asserted on PCI. The LCPRF command is driven again in the fifth PCI clock while TRDY# is sampled asserted at the end of this clock. The requested data for the read is then latched into the first location of the read prefetch buffer and driven onto the host data bus, becoming valid at the end of CPU clock 12. The BRDY# signal can therefore be driven asserted in this clock. The following read transaction (issued in CPU clock 15) requests the next Dword, and so the CPRA command is driven on the HIG[4:0] lines, advancing to read the next location in the read prefetch buffer. As the correct data is already there, the command is driven only once for this transaction. The next read transaction requests data in the same Dword as the previous. Therefore, the CPRA command is driven again, the buffer is not advanced, and the same Dword is driven onto the host bus.

3.3.6 PIG[3:0]: END-OF-LINE WARNING SIGNALS: EOL

When posting PCI master writes, the PCMC must be informed when the line boundary is about to be overrun, as it has no way of determining this itself (recall that the PCMC does not receive any address bits from PCI). The low order LBX determines this, as it contains the low order bits of the PCI master write address and also tracks how many Dwords of write data have been posted. Therefore, the low order LBX component sends the "end-of-line" warning to the PCMC. This is accomplished with the EOL signal driven from the low order LBX to the PCMC. Figure 9 illustrates the timing of this signal.

1. The FRAME# signal is sampled asserted in the first cycle. The LPMA command is driven on the PIG[3:0] signals to hold the address while it is being decoded (e.g. in the MEMCS# decode circuit of the 82378 SIO). The first data (D0) remains on the bus until TRDY# is asserted in response to MEMCS# being sampled asserted in the third clock.
2. The PPMWA command is driven in response to sampling MEMCS# asserted. TRDY# is asserted in this cycle indicating that D0 has been latched at the end of the fourth clock. The action of the PPMWA command is to transfer the PCI address captured in the PCI AD latch at the end of the first clock to the posting buffer, and open the PCI AD latch in order to capture the data. This data will be posted to the write buffer in the following cycle by the PPMWD command.
3. The EOL signal is first negated when the LPMA command is driven on the PIG[3:0] signals. However, if the first data Dword accepted is also the last that should be accepted, the EOL signal will be asserted in the third clock. This is the "end-of-line" indication. In this case, the EOL signal is asserted as soon as the LPMA command has been latched. The action by the PCMC in response is to negate TRDY# and assert STOP# in the fifth clock. Note that the EOL signal is asserted even before the MEMCS# signal is sampled asserted in this case. The EOL signal will remain asserted until the next time the LPMA command is driven.
4. If the second Dword is the last that should be accepted, the EOL signal will be asserted in the fifth clock to negate TRDY# and assert STOP# on the following clock. The EOL signal is asserted in response to the PPMWA command being sampled, and relies on the knowledge that TRDY# for the first Dword of data will be sampled asserted by the master in the same cycle (at the end of the fourth clock). Therefore, to prevent a third assertion of TRDY# in the sixth clock, the EOL signal must be asserted in the fifth clock.

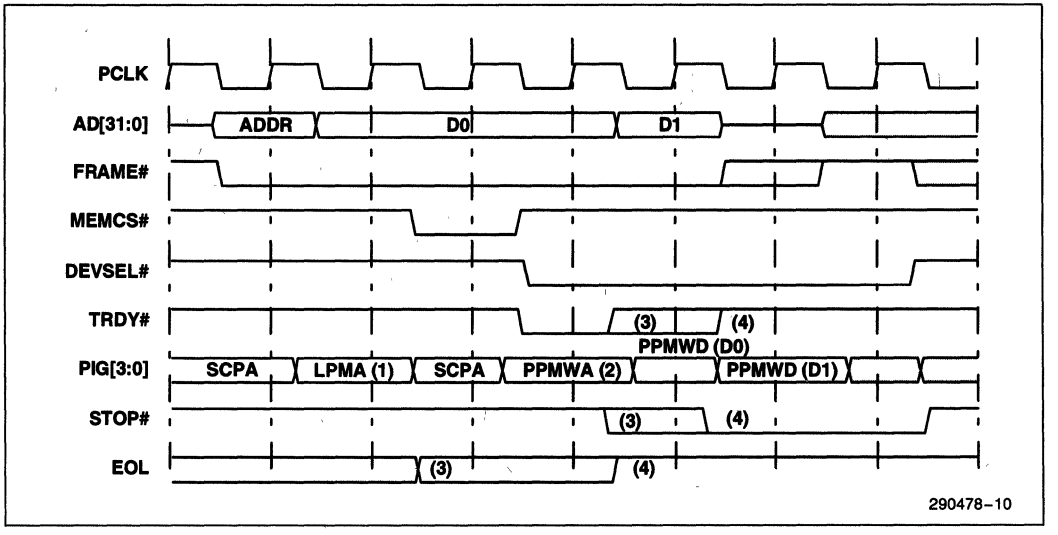


Figure 9. EOL Signal Timing for PCI Master Writes

A similar sequence is defined for PCI master reads. While it is possible to know when to stop driving read data due to the fact that the read address is latched into the PCMC before any read data is driven on PCI, the use of the EOL signal for PCI master reads simplifies the logic internal to the PCMC. Figure 10 illustrates the timing of EOL with respect to the PIG[3:0] commands to drive out PCI read data.

Note that unlike the PCI master write sequence, the STOP# signal is asserted with the last data transfer, not after.

1. The LPMA command sampled at the end of the second clock causes the EOL signal to assert if there is only one Dword left in the line, otherwise it will be negated. The first TRDY# will also be the last, and the STOP# signal will be asserted with TRDY#.
2. The SPMRH command causes the count of the number of Dwords left in the line to be decremented. If this count reaches one, the EOL signal is asserted. The next TRDY# will be the last, and STOP# is asserted with TRDY#.

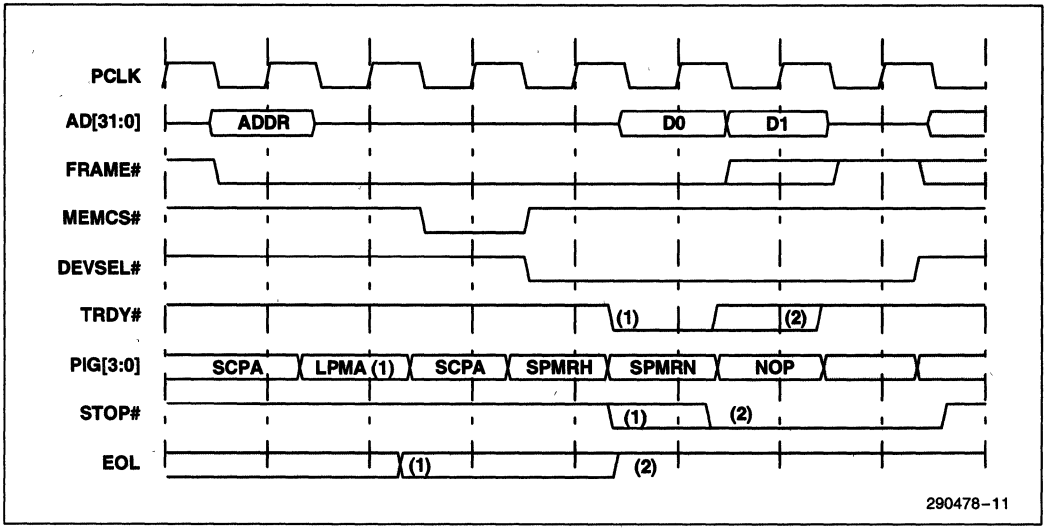


Figure 10. EOL Signal Timing for PCI Master Reads

3.4 PLL Loop Filter Components

As shown in Figure 11, loop filter components are required on the LBX components. A 4.7 KΩ 5% resistor is typically connected between pins LP1 and LP2. Pin LP2 has a path to the PLLAGND pin through a 100Ω 5% series resistor and a 0.01 μF 10% series capacitor. The ground side of capacitor C1 and the PLLVSS pin should connect to the ground plane at a common point. All PLL loop filter traces should be kept to minimal length and should be wider than signal traces. Inductor L1 is connected to the 5V power supply on both the 82433LX and 82433NX.

Some circuit boards may require filtering the power circuit to the LBX PLL. The circuit shown in Figure 11 will typically enable the LBX PLL to have higher noise immunity than without. Pin PLLVDD is connected to the 5V V_{CC} through a 10Ω 5% resistor. The PLLVDD and PLLVSS pins are bypassed with a 0.01 μF 10% series capacitor.

The high order 82433NX LBX samples A11 at the falling edge of reset to configure the LBX for PLL test mode. When A11 is sampled low, the LBX is in normal operating mode. When A11 is sampled high, the LBX drives the internal HCLK from the PLL on the EOL pin. Note that A11 on the high order LBX is connected to the A27 line on the CPU address bus. This same address line is used to put the PCMC into PLL test mode.

	Mercury 60 MHz	Mercury 66 MHz	Neptune
R1	4.7 KΩ	2.2 KΩ	4.7 KΩ
R2	100Ω	100Ω	100Ω
C2	0.01 μF	0.01 μF	0.01 μF
R3	10Ω	10Ω	10Ω
C1	0.47 μF	0.47 μF	0.47 μF
C1[1]	0.01 μF	0.01 μF	0.01 μF

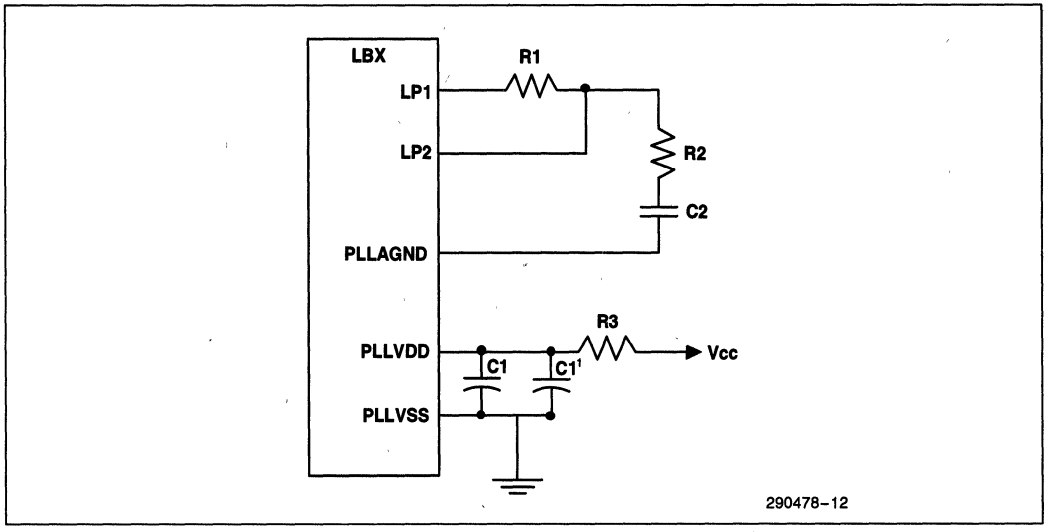


Figure 11. Loop Filter Circuit

3.5 PCI Clock Considerations

There is a 1.25 ns clock skew specification between the PCMC and the LBX that must be adhered to for proper operation of the PCMC/LBX timing. As shown in Figure 12, the PCMC drives PCLKOUT to an external clock driver which supplies copies of PCLK to PCI devices, the LBXs, and back to the PCMC. The skew specification is defined as the difference in timing between the signal that appears at the PCMC PCLKIN input pin and the signal that appears at the LBX PCLK input pin. For both the low order LBX and the high order LBX, the PCLK rising and falling edges must not be more than 1.25 ns apart from the rising and falling edge of the PCMC PCLKIN signal.

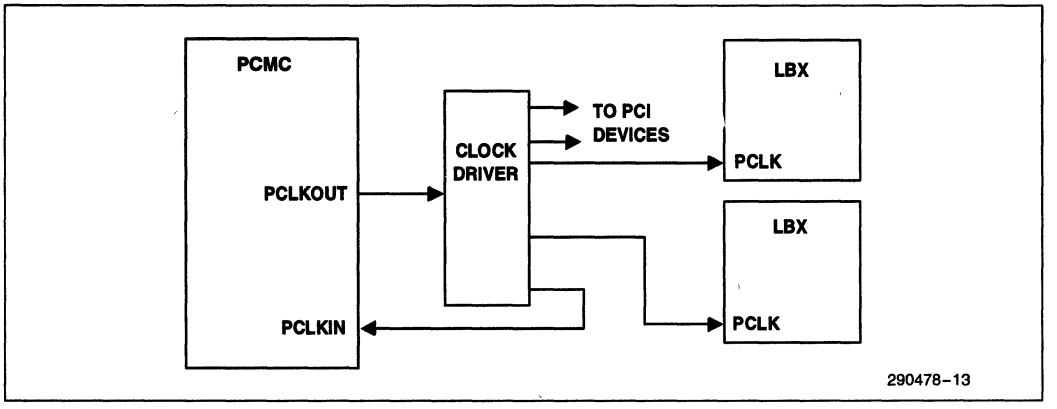


Figure 12. Clock Considerations

4.0 ELECTRICAL CHARACTERISTICS

4.1 Absolute Maximum Ratings

Table 4 lists stress ratings only. Functional operation at these maximums is not guaranteed. Functional operation conditions are given in Sections 4.2 and 4.3.

Extended exposure to the Absolute Maximum Ratings may affect device reliability.

Case Temperature under Bias .....0°C to +85°C

Storage Temperature .....−40°C to +125°C

Voltage on Any Pin with Respect to Ground.....−0.3 to V_{CC} + 0.3V

Supply Voltage with Respect to V_{SS} .....−0.3 to +7.0V

Maximum Power Dissipation: 1.4W (82433LX)

Maximum Total Power Dissipation .1.4W (82433NX)

Maximum Power Dissipation, V_{CC3} .....430 mW

The maximum total power dissipation in the 82433NX on the V_{CC} and V_{CC3} pins is 1.4W. The V_{CC3} pins may draw as much as 430 mW, however, total power will not exceed 1.4W.

NOTICE: This data sheet contains information on products in the sampling and initial production phases of development. The specifications are subject to change without notice. Verify with your local Intel Sales office that you have the latest data sheet before finalizing a design.

**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.*

4.2 Thermal Characteristics

The LBX is designed for operation at case temperatures between 0°C and 85°C. The thermal resistances of the package are given in the following tables.

Table 4. Thermal Resistance

Parameter	Air Flow Rate (Linear Feet per Minute)		
	0	400	600
θ_{JA} (°C/Watt)	51.9	37.1	34.8
θ_{JC} (°C/Watt)		10	

4.3 DC Characteristics

Host Interface Signals
A[15:0](t/s), D[31:0](t/s), HIG[4:0](in), HP[3:0](t/s)

Main Memory (DRAM) Interface Signals
MD[31:0](t/s), MP[3:0](t/s), MIG[2:0](in), MDLE(in)

PCI Interface Signals
AD[15:0](t/s), TRDY#(in), PIG[3:0](in), DRVPCI(in), EOL(t/s), PPOUT(t/s)

Reset and Clock Signals
HCLK(in), PCLK(in), RESET(in), LP1(out), LP2(in), TEST(in)

4.3.1 82433LX LBX DC CHARACTERISTICS

Functional Operating Range: V_{CC} = 4.75 V to 5.25V; T_{CASE} = 0°C to +85°C

Symbol	Parameter	Min	Typical	Max	Unit	Notes
V_{IL1}	Input Low Voltage	−0.3		0.8	V	1
V_{IH1}	Input High Voltage	2.0		V_{CC} + 0.3	V	1
V_{IL2}	Input Low Voltage	−0.3		0.3 × V_{CC}	V	2
V_{IH2}	Input High Voltage	0.7 × V_{CC}		V_{CC} + 0.3	V	2
V_{OL1}	Output Low Voltage			0.4	V	3
V_{OH1}	Output High Voltage	2.4			V	3
V_{OL2}	Output Low Voltage			0.5	V	4
V_{OH2}	Output High Voltage	V_{CC} − 0.5			V	4
I_{OL1}	Output Low Current			1	mA	5
I_{OH1}	Output High Current	−1			mA	5
I_{OL2}	Output Low Current			3	mA	6
I_{OH2}	Output High Current	−2			mA	6

Functional Operating Range: V_{CC} = 4.75V to 5.25V; T_{CASE} = 0°C to +85°C (Continued)

Symbol	Parameter	Min	Typical	Max	Unit	Notes
I_{OL3}	Output Low Current			3	mA	7
I_{OH3}	Output High Current	−1			mA	7
I_{IH}	Input Leakage Current			+10	μA	
I_{IL}	Input Leakage Current			−10	μA	
C_{IN}	Input Capacitance		4.6		pF	
C_{OUT}	Output Capacitance		4.3		pF	
$C_{I/O}$	I/O Capacitance		4.6		pF	

NOTES:
1. V_{IL1} and V_{IH1} apply to the following signals: AD[15:0], A[15:0], D[31:0], HP[3:0], MD[31:0], MP[3:0], TRDY#, RESET, HCLK, PCLK
2. V_{IL2} and V_{IH2} apply to the following signals: HIG[4:0], PIG[3:0], MIG[2:0], MDLE, DRVPCI
3. V_{OL1} and V_{OH1} apply to the following signals: AD[15:0], A[15:0], D[31:0], HP[3:0], MD[31:0], MP[3:0]
4. V_{OL2} and V_{OH2} apply to the following signals: PPOUT, EOL
5. I_{OL1} and I_{OH1} apply to the following signals: PPOUT, EOL
6. I_{OL2} and I_{OH2} apply to the following signals: AD[15:0]
7. I_{OL3} and I_{OH3} apply to the following signals: A[15:0], D[31:0], HP[3:0], MD[31:0], MP[3:0]

4.3.2 82433NX LBX DC CHARACTERISTICS

Functional Operating Range: V_{CC} = 4.75V to 5.25V; V_{CC3} = 3.135 to 3.465V, T_{CASE} = 0°C to +85°C

Symbol	Parameter	Min	Typical	Max	Unit	Notes
V_{IL1}	Input Low Voltage	−0.3		0.8	V	1
V_{IH1}	Input High Voltage	2.0		V_{CC} + 0.3	V	1
V_{IL2}	Input Low Voltage	−0.3		0.3 x V_{CC}	V	2
V_{IH2}	Input High Voltage	0.7 x V_{CC}		V_{CC} + 0.3	V	2
V_{IL3}	Input Low Voltage	−0.3		0.8	V	3
V_{IH3}	Input High Voltage	2.0		V_{CC3} + 0.3	V	3
V_{OL1}	Output Low Voltage			0.4	V	4
V_{OH1}	Output High Voltage	2.4			V	4
V_{OL2}	Output Low Voltage			0.5	V	5
V_{OH2}	Output High Voltage	V_{CC} − 0.5			V	5
I_{OL1}	Output Low Current			1	mA	6
I_{OH1}	Output High Current	−1			mA	6
I_{OL2}	Output Low Current			3	mA	7
I_{OH2}	Output High Current	−2			mA	7

Functional Operating Range: V_{CC} = 4.75V to 5.25V; V_{CC3} = 3.135V to 3.465V, T_{CASE} = 0°C to +85°C (Continued)

Symbol	Parameter	Min	Typical	Max	Unit	Notes
I_{OL3}	Output Low Current			3	mA	8
I_{OH3}	Output High Current	−1			mA	8
I_{IH}	Input Leakage Current			+10	µA	
I_{IL}	Input Leakage Current			−10	µA	
C_{IN}	Input Capacitance		4.6		pF	
C_{OUT}	Output Capacitance		4.3		pF	
$C_{I/O}$	I/O Capacitance		4.6		pF	

NOTES:
1. V_{IL1} and V_{IH1} apply to the following signals: AD[15:0], MD[31:0], MP[3:0], TRDY#, RESET, HCLK, PCLK
2. V_{IL2} and V_{IH2} apply to the following signals: HIG[4:0], PIG[3:0], MIG[2:0], MDLE, DRVPCI
3. V_{IL3} and V_{IH3} apply to the following signals: A[15:0], D[31:0], HP[3:0]
4. V_{OL1} and V_{OH1} apply to the following signals: AD[15:0], A[15:0], D[31:0], HP[3:0], MD[31:0], MP[3:0]
5. V_{OL2} and V_{OH2} apply to the following signals: PPOUT, EOL
6. I_{OL1} and I_{OH1} apply to the following signals: PPOUT, EOL
7. I_{OL2} and I_{OH2} apply to the following signals: AD[15:0]
8. I_{OL3} and I_{OH3} apply to the following signals: A[15:0], D[31:0], HP[3:0], MD[31:0], MP[3:0]
9. The output buffers for A[15:0], D[31:0] and HP[3:0] are powered with V_{CC3} and therefore drive 3.3V signal levels.

2

4.4 82433LX AC Characteristics

The AC specifications given in this section consist of propagation delays, valid delays, input setup requirements, input hold requirements, output float delays, output enable delays, clock high and low times and clock period specifications. Figure 13 through Figure 21 define these specifications. Sections 4.3.1 through 4.3.3 list the AC Specifications.

In Figure 13 through Figure 21 VT = 1.5V for the following signals: MD[31:0], MP[3:0], D[31:0], HP[3:0], A[15:0], AD[15:0], TRDY#, HCLK, PCLK, RESET, TEST.

VT = 2.5V for the following signals: HIG[4:0], PIG[3:0], MIG[2:0], MDLE, DRVPCI, PPOUT, EOL.

4.4.1 HOST AND PCI CLOCK TIMING, 66 MHZ (82433LX)

Functional Operating Range: V_{CC} = 4.9V to 5.25V; T_{CASE} = 0°C to +70°C

Symbol	Parameter	Min	Max	Figure	Notes
t1a	HCLK Period	15	20	18	
t1b	HCLK High Time	5		18	
t1c	HCLK Low Time	5		18	
t1d	HCLK Rise Time		1.5	19	
t1e	HCLK Fall Time		1.5	19	
t1f	HCLK Period Stability		±100		ps[1]
t2a	PCLK Period	30		18	
t2b	PCLK High Time	12		18	
t2c	PCLK Low Time	12		18	
t2d	PCLK Rise Time		3	19	
t2e	PCLK Fall Time		3	19	
t3	HCLK to PCLK Skew	−7.2	5.8	21	

NOTE:
1. Measured on rising edge of adjacent clocks at 1.5 Volts.

4.4.2 COMMAND TIMING, 66 MHZ (82433LX)

Functional Operating Range: V_{CC} = 4.9V to 5.25V; T_{CASE} = 0°C to +70°C

Symbol	Parameter	Min	Max	Figure	Notes
t10a	HIG[4:0] Setup Time to HCLK Rising	5.4		15	
t10b	HIG[4:0] Hold Time from HCLK Rising	0		15	
t11a	MIG[2:0] Setup Time to HCLK Rising	5.4		15	
t11b	MIG[2:0] Hold Time from HCLK Rising	0		15	
t12a	PIG[3:0] Setup Time to PCLK Rising	15.6		15	
t12b	PIG[3:0] Hold Time from PCLK Rising	−1.0		15	
t13a	MDLE Setup Time to HCLK Rising	5.7		15	
t13b	MDLE Hold Time to HCLK Rising	−0.3		15	
t14a	DRVPCI Setup Time to PCLK Rising	6.5		15	
t14b	DRVPCI Hold Time from PCLK Rising	−0.5		15	
t15a	RESET Setup Time to HCLK Rising	3.1		15	
t15b	RESET Hold Time from HCLK Rising	0.3		15	

4.4.3 ADDRESS, DATA, TRDY#, EOL, TEST, TSCON AND PARITY TIMING, 66 MHz (82433LX)

Functional Operating Range: V_{CC} = 4.9V to 5.25V; T_{CASE} = 0°C to +70°C

Symbol	Parameter	Min	Max	Figure	Notes
t20a	AD[15:0] Output Enable Delay from PCLK Rising	2		17	
t20b	AD[15:0] Valid Delay from PCLK Rising	2	11	14	1
t20c	AD[15:0] Setup Time to PCLK Rising	7		15	
t20d	AD[15:0] Hold Time from PCLK Rising	0		15	
t20e	AD[15:0] Float Delay from DRVPCI Falling	2	10	16	
t21a	TRDY# Setup Time to PCLK Rising	7		15	
t21b	TRDY# Hold Time from PCLK Rising	0		15	
t22a	D[31:0], HP[3:0] Output Enable Delay from HCLK Rising	0	7.7	17	2
t22b	D[31:0], HP[3:0] Float Delay from HCLK Rising	3.1	15.5	16	
t22c	D[31:0], HP[3:0] Float Delay from MDLE Rising	2	11.0	16	3
t22d	D[31:0], HP[3:0] Valid Delay from HCLK Rising	0	7.7	14	2
t22e	D[31:0], HP[3:0] Setup Time to HCLK Rising	3.0		15	
t22f	D[31:0], HP[3:0] Hold Time from HCLK Rising	0.3		15	
t23a	HA[15:0] Output Enable Delay from HCLK Rising	0	15.2	17	
t23b	HA[15:0] Float Delay from HCLK Rising	0	15.2	16	
t23c	HA[15:0] Valid Delay from HCLK Rising	0	16	14	7
t23cc	HA[15:0] Valid Delay from HCLK Rising	0	14.5		8
t23d	HA[15:0] Setup Time to HCLK Rising	15		15	4
t23e	HA[15:0] Setup Time to HCLK Rising	4.1		15	5
t23f	HA[15:0] Hold Time from HCLK Rising	0.3		15	
t24a	MD[31:0], MP[3:0] Valid Delay from HCLK Rising	0	12.0	14	6
t24b	MD[31:0], MP[3:0] Setup Time to HCLK Rising	4.0		15	
t24c	MD[31:0], MP[3:0] Hold Time from HCLK Rising	0.4		15	
t25	EOL, PPOUT Valid Delay from PCLK Rising	2.3	17.2	14	2
t26a	All Outputs Float Delay from TSCON Falling	0	30	16	
t26b	All Outputs Enable Delay from TSCON Rising	0	30	17	

NOTES:
1. Min: 0 pF, Max: 50 pF
2. 0 pF
3. When NOPC command sampled on previous rising HCLK on HIG[4:0]
4. CPU to PCI Transfers
5. When ADCPY command is sampled on HIG[4:0]
6. 50 pF
7. When DACPYL or DACPYH commands are sampled on HIG[4:0]
8. Inquire cycle

4.4.4 HOST AND PCI CLOCK TIMING, 60 MHz (82433LX)

Functional Operating Range: V_{CC} = 4.75V to 5.25V; T_{CASE} = 0°C to +85°C

Symbol	Parameter	Min	Max	Figure	Notes
t1a	HCLK Period	16.6	20	18	
t1b	HCLK High Time	5.5		18	
t1c	HCLK Low Time	5.5		18	
t1d	HCLK Rise Time		1.5	19	
t1e	HCLK Fall Time		1.5	19	
t1f	HCLK Period Stability		±100		ps[1]
t2a	PCLK Period	33.33		18	
t2b	PCLK High Time	13		18	
t2c	PCLK Low Time	13		18	
t2d	PCLK Rise Time		3	19	
t2e	PCLK Fall Time		3	19	
t3	PCLK to PCMC PCLKIN: Input to Input Skew	−7.2	5.8	21	

NOTES:
1. Measured on rising edge of adjacent clocks at 1.5 Volts

4.4.5 COMMAND TIMING, 60 MHZ (82433LX)

Functional Operating Range: V_{CC} = 4.75V to 5.25V; T_{CASE} = 0°C to +85°C

Symbol	Parameter	Min	Max	Figure	Notes
t10a	HIG[4:0] Setup Time to HCLK Rising	6.0		15	
t10b	HIG[4:0] Hold Time from HCLK Rising	0		15	
t11a	MIG[2:0] Setup Time to HCLK Rising	6.0		15	
t11b	MIG[2:0] Hold Time from HCLK Rising	0		15	
t12a	PIG[3:0] Setup Time to PCLK Rising	16.0		15	
t12b	PIG[3:0] Hold Time from PCLK Rising	0		15	
t13a	MDLE Setup Time to HCLK Rising	5.9		15	
t13b	MDLE Hold Time to HCLK Rising	−0.3		15	
t14a	DRVPCI Setup Time to PCLK Rising	7.0		15	
t14b	DRVPCI Hold Time from PCLK Rising	−0.5		15	
t15a	RESET Setup Time to HCLK Rising	3.4		15	
t15b	RESET Hold Time from HCLK Rising	0.4		15	

4.4.6 ADDRESS, DATA, TRDY#, EOL, TEST, TSCON AND PARITY TIMING, 60 MHz (82433LX)

Functional Operating Range: V_{CC} = 4.75V to 5.25V; T_{CASE} = 0°C to +85°C

Symbol	Parameter	Min	Max	Figure	Notes
t20a	AD[15:0] Output Enable Delay from PCLK Rising	2		17	
t20b	AD[15:0] Valid Delay from PCLK Rising	2	11	14	1
t20c	AD[15:0] Setup Time to PCLK Rising	7		15	
t20d	AD[15:0] Hold Time from PCLK Rising	0		15	
t20e	AD[15:0] Float Delay from DRVPCI Falling	2	10	16	
t21a	TRDY# Setup Time to PCLK Rising	7		15	
t21b	TRDY# Hold Time from PCLK Rising	0		15	
t22a	D[31:0], HP[3:0] Output Enable Delay from HCLK Rising	0	7.9	17	2
t22b	D[31:0], HP[3:0] Float Delay from HCLK Rising	3.1	15.5	16	
t22c	D[31:0], HP[3:0] Float Delay from MDLE Rising	2	11.0	16	3
t22d	D[31:0], HP[3:0] Valid Delay from HCLK Rising	0	7.8	14	2
t22e	D[31:0],HP[3:0] Setup Time to HCLK Rising	3.4		15	
t22f	D[31:0], HP[3:0] Hold Time from HCLK Rising	0.3		15	
t23a	HA[15:0] Output Enable Delay from HCLK Rising	0	15.2	17	
t23b	HA[15:0] Float Delay from HCLK Rising	0	15.2	16	
t23c	HA[15:0] Valid Delay from HCLK Rising	0	18.5	14	7
t23cc	HA[15:0] Valid Delay from HCLK Rising	0	15.5		8
t23d	HA[15:0] Setup Time to HCLK Rising	15.0		15	4
t23e	HA[15:0] Setup Time to HCLK Rising	4.1		15	5
t23f	HA[15:0] Hold Time from HCLK Rising	0.3		15	
t24a	MD[31:0], MP[3:0] Valid Delay from HCLK Rising	0	12.0	14	6
t24b	MD[31:0], MP[3:0] Setup Time to HCLK Rising	4.4		15	
t24c	MD[31:0], MP[3:0] Hold Time from HCLK Rising	1.0		15	
t25	EOL, PPOUT Valid Delay from PCLK Rising	2.3	17.2	14	2
t26a	All Outputs Float Delay from TSCON Falling	0	30	16	
t26b	All Outputs Enable Delay from TSCON Rising	0	30	17	

NOTES:
1. Min: 0 pF, Max: 50 pF
2. 0 pF
3. When NOPC command sampled on previous rising HCLK on HIG[4:0]
4. CPU to PCI Transfers
5. When ADCPY command is sampled on HIG[4:0]
6. 50 pF
7. When DACPYL or DACPYH commands are sampled on HIG[4:0]
8. Inquire cycle

4.4.7 TEST TIMING (82433LX)

Functional Operating Range: V_{CC} = 4.75V to 5.25V; T_{CASE} = 0°C to +85°C

Symbol	Parameter	Min	Max	Figure	Notes
t30	All Test Signals Setup Time to HCLK/PCLK Rising	10.0			In PLL Bypass Mode
t31	All Test Signals Hold Time to HCLK/PCLK Rising	12.0			In PLL Bypass Mode
t32	Test Setup Time to HCLK/PCLK Rising	15.0		15	
t33	Test Hold Time to HCLK/PCLK Rising	5.0		15	
t34	PPOUT Valid Delay from PCLK Rising	0.0	500	15	In PLL Bypass Mode

4.5 82433NX AC Characteristics

The AC specifications given in this section consist of propagation delays, valid delays, input setup requirements, input hold requirements, output float delays, output enable delays, clock high and low times and clock period specifications. Figure 13 through Figure 21 define these specifications. Section 4.5 lists the AC Specifications.

In Figure 13 through Figure 21 VT = 1.5V for the following signals: MD[31:0], MP[3:0], D[31:0], HP[3:0], A[15:0], AD[15:0], TRDY#, HCLK, PCLK, RESET, TEST.

VT = 2.5V for the following signals: HIG[4:0], PIG[3:0], MIG[2:0], MDLE, DRVPCI, PPOUT, EOL.

4.5.1 HOST AND PCI CLOCK TIMING, (82433NX)

Functional Operating Range: V_{CC} = 4.75V to 5.25V; V_{CC3} = 3.135V to 3.465V, T_{CASE} = 0°C to +85°C

Symbol	Parameter	Min	Max	Figure	Notes
t1a	HCLK Period	15	20	18	
t1b	HCLK High Time	5		18	
t1c	HCLK Low Time	5		18	
t1d	HCLK Rise Time		1.5	19	
t1e	HCLK Fall Time		1.5	19	
t1f	HCLK Period Stability		±100		ps[1]
t2a	PCLK Period	30		18	
t2b	PCLK High Time	12		18	
t2c	PCLK Low Time	12		18	
t2d	PCLK Rise Time		3	19	
t2e	PCLK Fall Time		3	19	
t3	HCLK to PCLK Skew	−7.2	5.8	21	

NOTE:
1. Measured on rising edge of adjacent clocks at 1.5 Volts.

4.5.2 COMMAND TIMING, (82433NX)

Functional Operating Range: V_{CC} = 4.75V to 5.25V; V_{CC3} = 3.135V to 3.465V, T_{CASE} = 0°C to +85°C

Symbol	Parameter	Min	Max	Figure	Notes
t10a	HIG[4:0] Setup Time to HCLK Rising	5.5		15	
t10b	HIG[4:0] Hold Time from HCLK Rising	0		15	
t11a	MIG[2:0] Setup Time to HCLK Rising	5.5		15	
t11b	MIG[2:0] Hold Time from HCLK Rising	0		15	
t12a	PIG[3:0] Setup Time to PCLK Rising	14.5		15	
t12b	PIG[3:0] Hold Time from PCLK Rising	0.0		15	
t13a	MDLE Setup Time to HCLK Rising	5.5		15	
t13b	MDLE Hold Time to HCLK Rising	−0.3		15	
t14a	DRVPCI Setup Time to PCLK Rising	7.0		15	
t14b	DRVPCI Hold Time from PCLK Rising	−0.5		15	
t15a	RESET Setup Time to HCLK Rising	3.4		15	
t15b	RESET Hold Time from HCLK Rising	0.4		15	

4.5.3 ADDRESS, DATA, TRDY#, EOL, TEST, TSCON AND PARITY TIMING, (82433NX)

Functional Operating Range: V_{CC} = 4.75V to 5.25V; V_{CC3} = 3.135V to 3.465V, T_{CASE} = 0°C to +85°C

Symbol	Parameter	Min	Max	Figure	Notes
t20a	AD[15:0] Output Enable Delay from PCLK Rising	2		17	
t20b	AD[15:0] Valid Delay from PCLK Rising	2	11	14	1
t20c	AD[15:0] Setup Time to PCLK Rising	7		15	
t20d	AD[15:0] Hold Time from PCLK Rising	0		15	
t20e	AD[15:0] Float Delay from DRVPCI Falling	2	10	16	
t21a	TRDY# Setup Time to PCLK Rising	7		15	
t21b	TRDY# Hold Time from PCLK Rising	0		15	
t22a	D[31:0], HP[3:0] Output Enable Delay from HCLK Rising	0	7.5	17	2
t22b	D[31:0], HP[3:0] Float Delay from HCLK Rising	3.1	15.5	16	
t22c	D[31:0], HP[3:0] Float Delay from MDLE Rising	2	9.5	16	3
t22d	D[31:0], HP[3:0] Valid Delay from HCLK Rising	0	7.5	14	2
t22e	D[31:0],HP[3:0] Setup Time to HCLK Rising	3.1		15	
t22f	D[31:0], HP[3:0] Hold Time from HCLK Rising	0.3		15	

Functional Operating Range: V_{CC} = 4.75V to 5V; V_{CC3} = 3.135V to 3.465V, T_{CASE} = 0°C to +85°C (Continued)

Symbol	Parameter	Min	Max	Figure	Notes
t23a	HA[15:0] Output Enable Delay from HCLK Rising	0	13.5	17	
t23b	HA[15:0] Float Delay from HCLK Rising	0	13.5	16	
t23c	HA[15:0] Valid Delay from HCLK Rising	0	17.5	14	7
t23cc	HA[15:0] Valid Delay from HCLK Rising	0	13.5		8
t23d	HA[15:0] Setup Time to HCLK Rising	15		15	4
t23e	HA[15:0] Setup Time to HCLK Rising	4.2		15	5
t23f	HA[15:0] Hold Time from HCLK Rising	0.3		15	
t24a	MD[31:0], MP[3:0] Valid Delay from HCLK Rising	0	12.0	14	6
t24b	MD[31:0], MP[3:0] Setup Time to HCLK Rising	4.4		15	
t24c	MD[31:0], MP[3:0] Hold Time from HCLK Rising	1.0		15	
t25	EOL, PPOUT Valid Delay from PCLK Rising	2.3	17.2	14	2
t26a	All Outputs Float Delay from TSCON Falling	0	30	16	
t26b	All Outputs Enable Delay from TSCON Rising	0	30	17	

NOTE:
1. Min: 0 pF, Max: 50 pF
2. 0 pF
3. When NOPC command sampled on previous rising HCLK on HIG[4:0]
4. CPU to PCI Transfers
5. When ADCPY command is sampled on HIG[4:0]
6. 50 pF
7. When DACPYL or DACPYH commands are sampled on HIG[4:0]
8. Inquire cycle

4.5.4 TEST TIMING (82433NX)

Functional Operating Range: V_{CC} = 4.75V to 5.25V; V_{CC3} = 3.135V to 3.465V, T_{CASE} = 0°C to +85°C

Symbol	Parameter	Min	Max	Figure	Notes
t30	All Test Signals Setup Time to HCLK/PCLK Rising	10.0			In PLL Bypass Mode
t31	All Test Signals Hold Time to HCLK/PCLK Rising	12.0			In PLL Bypass Mode
t32	Test Setup Time to HCLK/PCLK Rising	15.0		15	
t33	Test Hold Time to HCLK/PCLK Rising	5.0		15	
t34	PPOUT Valid Delay from PCLK Rising	0.0	500	15	In PLL Bypass Mode

4.5.5 TIMING DIAGRAMS

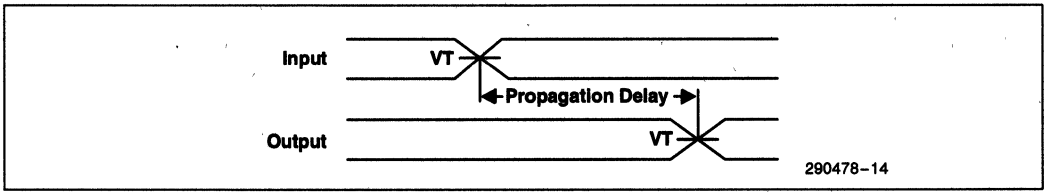


Figure 13. Propagation Delay

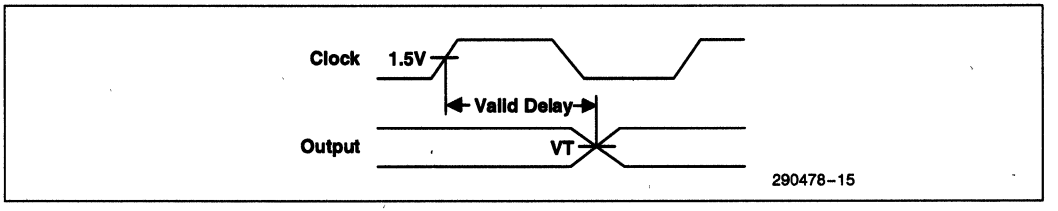


Figure 14. Valid Delay from Rising Clock Edge

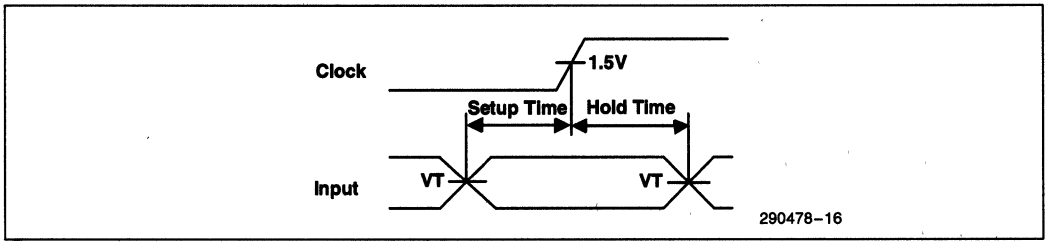


Figure 15. Setup and Hold Times

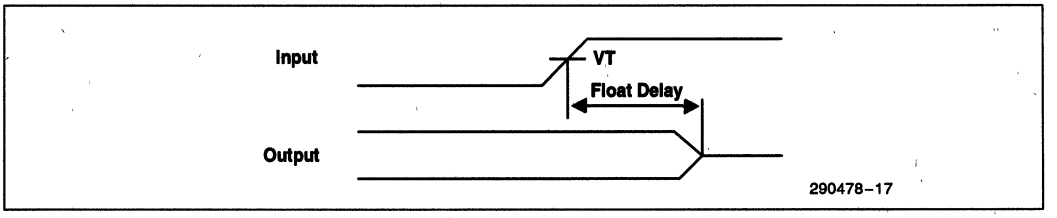


Figure 16. Float Delay

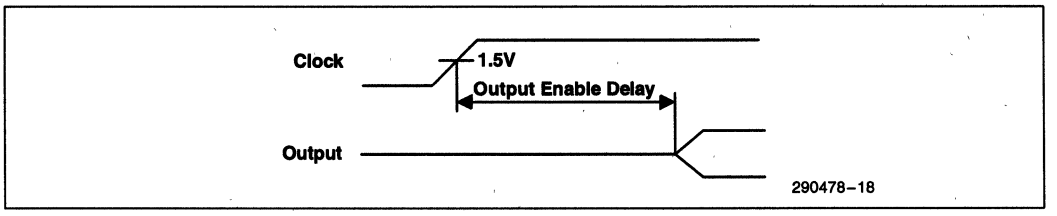


Figure 17. Output Enable Delay

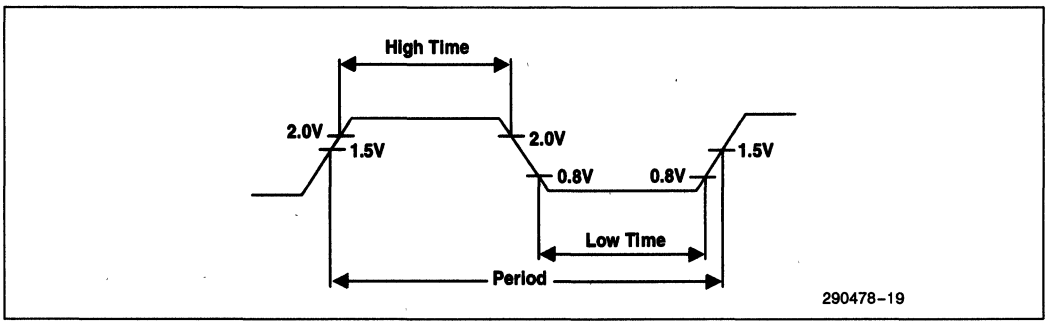


Figure 18. Clock High and Low Times and Period

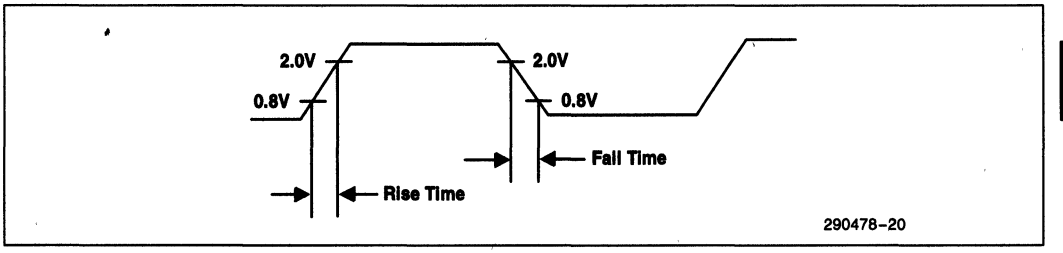


Figure 19. Clock Rise and Fall Times

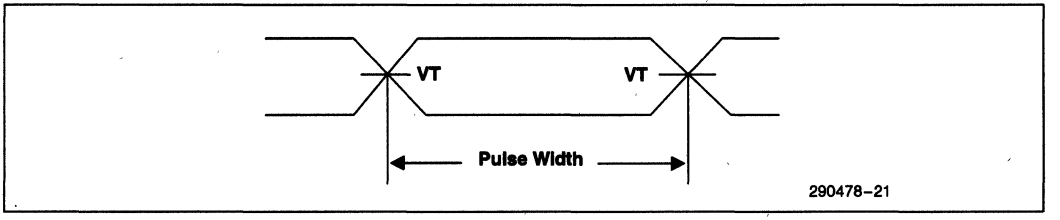


Figure 20. Pulse Width

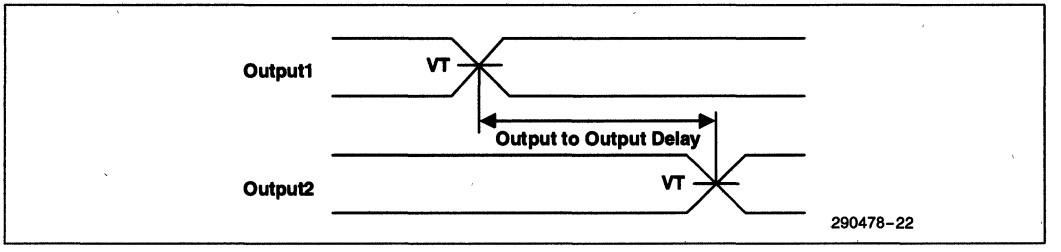


Figure 21. Output to Output Delay

5.0 PINOUT AND PACKAGE INFORMATION

5.1 Pin Assignment

Pins 1, 22, 41, 61, and 150 are VDD3 pins on the 82433NX. These pins must be connected to the 3.3V power supply. All other VDD pins on the 82433NX must be connected to the 5V power supply.

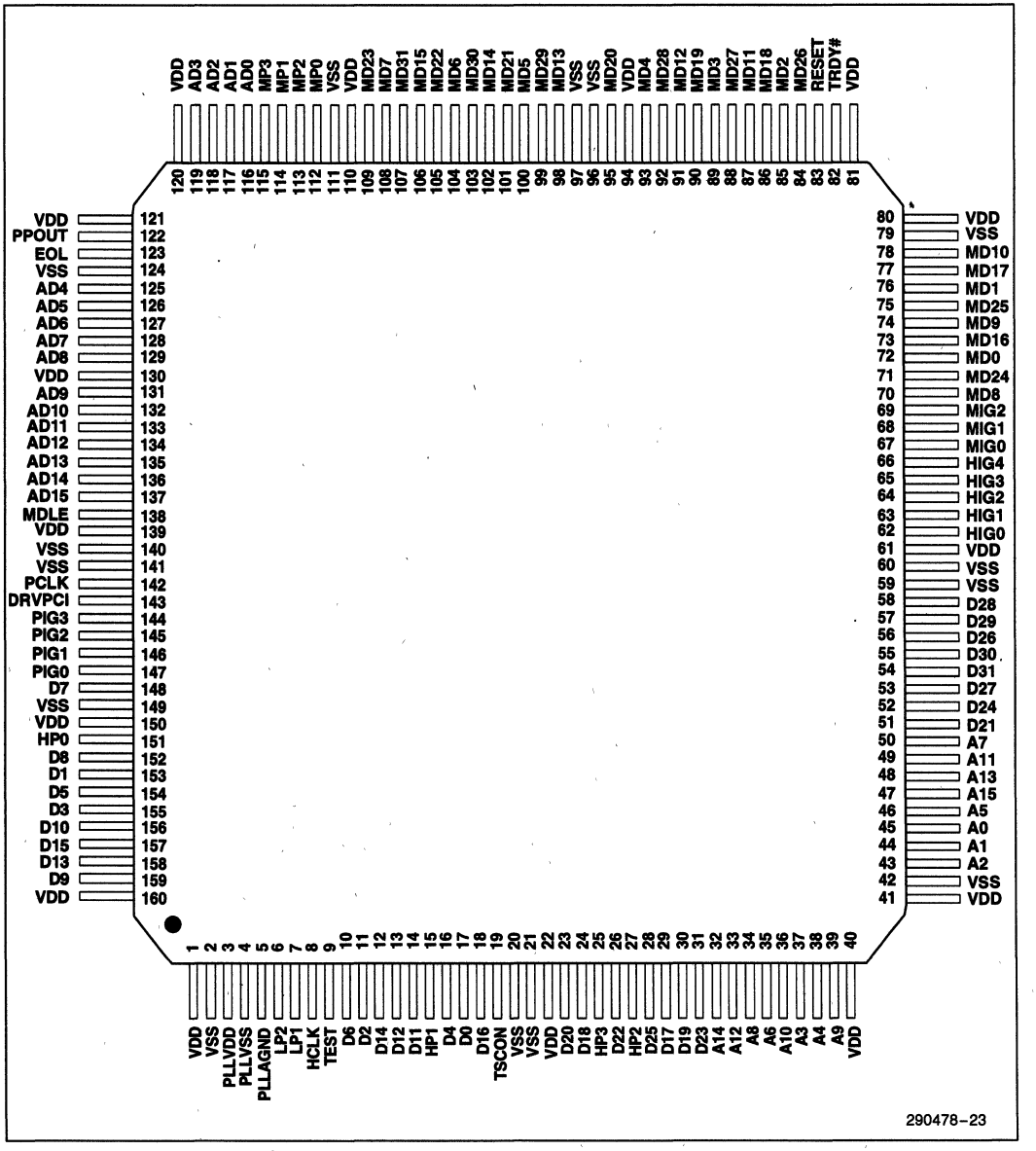


Figure 22. 82433LX and 82433NX Pin Assignment

Table 5. 82433LX and 82433NX Numerical Pin Assignment

Pin Name	Pin #	Type
V_{DD} (82433LX) V_{DD3} (82433NX)	1	V
V_{SS}	2	V
$PLLV_{DD}$	3	V
$PLLV_{SS}$	4	V
PLLAGND	5	V
LP2	6	in
LP1	7	out
HCLK	8	in
TEST	9	in
D6	10	t/s
D2	11	t/s
D14	12	t/s
D12	13	t/s
D11	14	t/s
HP1	15	t/s
D4	16	t/s
D0	17	t/s
D16	18	t/s
TSCON	19	in
V_{SS}	20	V
V_{SS}	21	V
V_{DD} (82433LX V_{DD3} (82433NX)	22	V
D20	23	t/s
D18	24	t/s
HP3	25	t/s
D22	26	t/s
HP2	27	t/s
D25	28	t/s
D17	29	t/s
D19	30	t/s
D23	31	t/s
A14	32	t/s
A12	33	t/s
A8	34	t/s
A6	35	t/s
A10	36	t/s
A3	37	t/s
A4	38	t/s
A9	39	t/s
V_{DD}	40	V
V_{DD} (82433LX) V_{DD3} (82433NX)	41	V
V_{SS}	42	V
A2	43	t/s
A1	44	t/s
A0	45	t/s
A5	46	t/s
A15	47	t/s
A13	48	t/s
A11	49	t/s
A7	50	t/s
D21	51	t/s
D24	52	t/s
D27	53	t/s
D31	54	t/s
D30	55	t/s
D26	56	t/s
D29	57	t/s
D28	58	t/s
V_{SS}	59	V
V_{SS}	60	V
V_{DD} (82433LX) V_{DD3} (82433NX)	61	V
HIG0	62	in
HIG1	63	in
HIG2	64	in
HIG3	65	in
HIG4	66	in
MIG0	67	in
MIG1	68	in
MIG2	69	in
MD8	70	t/s
MD24	71	t/s
MD0	72	t/s
MD16	73	t/s
MD9	74	t/s
MD25	75	t/s

Table 5. 82433LX and 82433NX Numerical Pin Assignment (Continued)

Pin Name	Pin #	Type
MD1	76	t/s
MD17	77	t/s
MD10	78	t/s
V_{SS}	79	V
V_{DD}	80	V
V_{DD}	81	V
TRDY#	82	in
RESET	83	in
MD26	84	t/s
MD2	85	t/s
MD18	86	t/s
MD11	87	t/s
MD27	88	t/s
MD3	89	t/s
MD19	90	t/s
MD12	91	t/s
MD28	92	t/s
MD4	93	t/s
V_{DD}	94	V
MD20	95	t/s
V_{SS}	96	V
V_{SS}	97	V
MD13	98	t/s
MD29	99	t/s
MD5	100	t/s
MD21	101	t/s
MD14	102	t/s
MD30	103	t/s
MD6	104	t/s

Pin Name	Pin #	Type
MD22	105	t/s
MD15	106	t/s
MD31	107	t/s
MD7	108	t/s
MD23	109	t/s
V_{DD}	110	V
V_{SS}	111	V
MP0	112	t/s
MP2	113	t/s
MP1	114	t/s
MP3	115	t/s
AD0	116	t/s
AD1	117	t/s
AD2	118	t/s
AD3	119	t/s
V_{DD}	120	V
V_{DD}	121	V
PPOUT	122	t/s
EOL	123	t/s
V_{SS}	124	V
AD4	125	t/s
AD5	126	t/s
AD6	127	t/s
AD7	128	t/s
AD8	129	t/s
V_{DD}	130	V
AD9	131	t/s
AD10	132	t/s

Pin Name	Pin #	Type
AD11	133	t/s
AD12	134	t/s
AD13	135	t/s
AD14	136	t/s
AD15	137	t/s
MDLE	138	in
V_{DD}	139	V
V_{SS}	140	V
V_{SS}	141	V
PCLK	142	in
DRVPCI	143	in
PIG3	144	in
PIG2	145	in
PIG1	146	in
PIG0	147	in
D7	148	t/s
V_{SS}	149	V
V_{DD} (82433LX) $V_{DD}3$ (82433NX)	150	V
HP0	151	t/s
D8	152	t/s
D1	153	t/s
D5	154	t/s
D3	155	t/s
D10	156	t/s
D15	157	t/s
D13	158	t/s
D9	159	t/s
V_{DD}	160	V

Table 6. 82433LX and 82433NX Alphabetical Pin Assignment List

Pin Name	Pin #	Type
A0	45	t/s
A1	44	t/s
A2	43	t/s
A3	37	t/s
A4	38	t/s
A5	46	t/s
A6	35	t/s
A7	50	t/s
A8	34	t/s
A9	39	t/s
A10	36	t/s
A11	49	t/s
A12	33	t/s
A13	48	t/s
A14	32	t/s
A15	47	t/s
AD0	116	t/s
AD1	117	t/s
AD2	118	t/s
AD3	119	t/s
AD4	125	t/s
AD5	126	t/s
AD6	127	t/s
AD7	128	t/s
AD8	129	t/s
AD9	131	t/s
AD10	132	t/s
AD11	133	t/s
AD12	134	t/s

Pin Name	Pin #	Type
AD13	135	t/s
AD14	136	t/s
AD15	137	t/s
D0	17	t/s
D1	153	t/s
D2	11	t/s
D3	155	t/s
D4	16	t/s
D5	154	t/s
D6	10	t/s
D7	148	t/s
D8	152	t/s
D9	159	t/s
D10	156	t/s
D11	14	t/s
D12	13	t/s
D13	158	t/s
D14	12	t/s
D15	157	t/s
D16	18	t/s
D17	29	t/s
D18	24	t/s
D19	30	t/s
D20	23	t/s
D21	51	t/s
D22	26	t/s
D23	31	t/s
D24	52	t/s
D25	28	t/s

Pin Name	Pin #	Type
D26	56	t/s
D27	53	t/s
D28	58	t/s
D29	57	t/s
D30	55	t/s
D31	54	t/s
DRVPCI	143	in
EOL	123	t/s
HCLK	8	in
HIG0	62	in
HIG1	63	in
HIG2	64	in
HIG3	65	in
HIG4	66	in
HP0	151	t/s
HP1	15	t/s
HP2	27	t/s
HP3	25	t/s
LP1	7	out
LP2	6	in
MD0	72	t/s
MD1	76	t/s
MD2	85	t/s
MD3	89	t/s
MD4	93	t/s
MD5	100	t/s
MD6	104	t/s
MD7	108	t/s
MD8	70	t/s

Table 6. 82433LX and 82433NX Alphabetical Pin Assignment List (Continued)

Pin Name	Pin #	Type
MD9	74	t/s
MD10	78	t/s
MD11	87	t/s
MD12	91	t/s
MD13	98	t/s
MD14	102	t/s
MD15	106	t/s
MD16	73	t/s
MD17	77	t/s
MD18	86	t/s
MD19	90	t/s
MD20	95	t/s
MD21	101	t/s
MD22	105	t/s
MD23	109	t/s
MD24	71	t/s
MD25	75	t/s
MD26	84	t/s
MD27	88	t/s
MD28	92	t/s
MD29	99	t/s
MD30	103	t/s
MD31	107	t/s
MDLE	138	in
MIG0	67	in

Pin Name	Pin #	Type
MIG1	68	in
MIG2	69	in
MP0	112	t/s
MP1	114	t/s
MP2	113	t/s
MP3	115	t/s
PCLK	142	in
PIG0	147	in
PIG1	146	in
PIG2	145	in
PIG3	144	in
PLLAGND	5	V
$PLLV_{DD}$	3	V
$PLLV_{SS}$	4	V
PPOUT	122	t/s
RESET	83	in
TEST	9	in
TRDY	82	in
TSCON	19	in
V_{DD} (82433LX) $V_{DD}3$ (82433NX)	1	V
V_{DD} (82433LX) $V_{DD}3$ (82433NX)	22	V
V_{DD}	40	V
V_{DD} (82433LX) $V_{DD}3$ (82433NX)	41	V
V_{DD} (82433LX) $V_{DD}3$ (82433NX)	61	V

Pin Name	Pin #	Type
V_{DD}	80	V
V_{DD}	81	V
V_{DD}	94	V
V_{DD}	110	V
V_{DD}	120	V
V_{DD}	121	V
V_{DD}	130	V
V_{DD}	139	V
V_{DD} (82433LX) $V_{DD}3$ (82433NX)	150	V
V_{DD}	160	V
V_{SS}	2	V
V_{SS}	20	V
V_{SS}	21	V
V_{SS}	42	V
V_{SS}	59	V
V_{SS}	60	V
V_{SS}	79	V
V_{SS}	96	V
V_{SS}	97	V
V_{SS}	111	V
V_{SS}	124	V
V_{SS}	140	V
V_{SS}	141	V
V_{SS}	149	V

5.2 Package Information

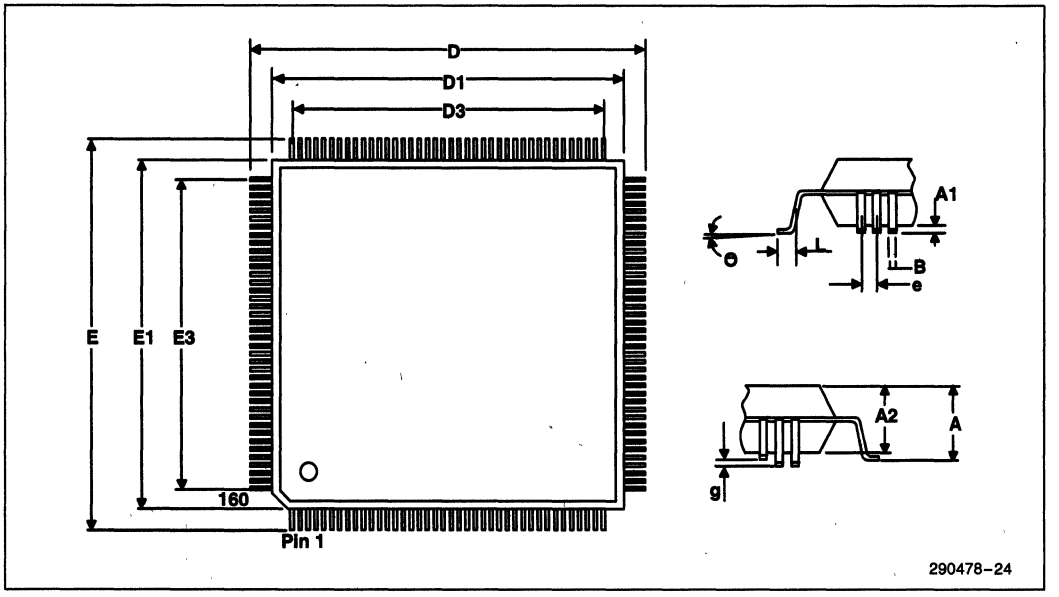


Figure 23. 82433LX and 82433NX 160-Pin QFP Package

Table 7. 160-Pin QFP Package Values

Symbol	Min Value (mm)	Max Value (mm)
A		4.45
A1	0.25	0.65
A2	3.30	3.80
B	0.20	0.40
D	31.00	32.40
D1	27.80	28.20
D3		25.55

Symbol	Min Value (mm)	Max Value (mm)
E	31.60	32.40
E1	27.80	28.20
E3		25.55
e		0.65
L	0.60	1.00
θ	0°	10°
g		0.1

6.0 TESTABILITY

The TSCON pin may be used to help test circuits surrounding the LBX. During normal operations, the TSCON pin must be tied to VCC or connected to VCC through a pull-up resistor. All LBX outputs are tri-stated when the TSCON pin is held low or grounded.

6.1 NAND Tree

A NAND tree is provided in the LBX for Automated Test Equipment (ATE) board level testing. The NAND tree allows the tester to set the connectivity of each of the LBX signal pins.

The following steps must be taken to put the LBX into PLL bypass mode and enable the NAND tree. First, to enable PLL bypass mode, drive RESET inactive, TEST active, and the DCPWA command (0100) on the PIG[3:0] lines. Then drive PCLK from low to high. DRVPCI must be held low on all rising edges of PCLK during testing in order to ensure that the LBX does not drive the AD[15:0] lines. The host and memory buses are tri-stated by driving NOPM (000) and NOPC (00000) on the MIG[2:0] and HIG[4:0] lines and driving two rising edges on HCLK. A rising edge on PCLK with RESET high will cause the LBXs to exit PLL bypass mode. TEST must remain high throughout the use of the NAND tree. The combination of TEST and DRVPCI high with a rising edge of PCLK must be avoided. TSCON must be driven high throughout testing since driving it low would tri-state the output of the NAND tree. A 10 ns hold time is required on all inputs sampled by PCLK or HCLK when in PLL bypass mode.

6.1.1 TEST VECTOR TABLE

The following test vectors can be applied to the 82433LX and 82433NX to put it into PLL bypass mode and to enable NAND tree testing.

6.1.2 NAND TREE TABLE

Table 9 shows the sequence of the NAND tree in the 82433LX and 82433NX. Non-inverting inputs are driven directly into the input of a NAND gate in the tree. Inverting inputs are driven into an inverter before going into the NAND tree. The output of the NAND tree is driven on the PPOUT pin.

Table 8. Test Vectors to put LBX Into PLL Bypass and Enable NAND Tree Testing

LBX Pin/Vector#	1	2	3	4	5	6	7	8	9	10	11
PCLK	0	1	0	0	1	1	1	1	1	1	1
PIG[3:0]	0h	0h	0h	4h	4h	4h	4h	4h	4h	4h	4h
RESET	1	1	1	0	0	0	1	1	1	1	1
HCLK	0	0	0	0	0	0	0	1	0	1	0
MIG[2:0]	0h	0h	0h	0h	0h	0h	0h	0h	0h	0h	0h
HIG[4:0]	0h	0h	0h	0h	0h	0h	0h	0h	0h	0h	0h
TEST	1	1	1	1	1	1	1	1	1	1	1
DRVPCI	0	0	0	0	0	0	0	0	0	0	0

Table 9. NAND Tree Sequence

Order	Pin #	Signal	Non-Inverting
1	10	D6	Y
2	11	D2	Y
3	12	D14	Y
4	13	D12	Y
5	14	D11	Y
6	15	HP1	Y
7	16	D4	Y
8	17	D0	Y
9	18	D16	Y
10	23	D20	Y
11	24	D18	Y
12	25	HP3	Y
13	26	D22	Y
14	27	HP2	Y
15	28	D25	Y
16	29	D17	Y
17	30	D19	Y
18	31	D23	Y
19	32	A14	Y
20	33	A12	Y
21	34	A8	Y
22	35	A6	Y
23	36	A10	Y
24	37	A3	Y
25	38	A4	Y
26	39	A9	Y
27	43	A2	Y
28	44	A1	Y
29	45	A0	Y
30	46	A5	Y
31	47	A15	Y
32	48	A13	Y
33	49	A11	Y
34	50	A7	Y
35	51	D21	Y
36	52	D24	Y
37	53	D27	Y
38	54	D31	Y
39	55	D30	Y
40	56	D26	Y
41	57	D29	Y
42	58	D28	Y
43	62	HIG0	Y
44	63	HIG1	Y
45	64	HIG2	Y
46	65	HIG3	Y
47	66	HIG4	Y
48	67	MIG0	N
49	68	MIG1	N
50	69	MIG2	N
51	70	MD8	N
52	71	MD24	N
53	72	MD0	N
54	73	MD16	N
55	74	MD9	N
56	75	MD25	N
57	76	MD1	N
58	77	MD17	N
59	78	MD10	N
60	82	TRDY#	Y
61	83	RESET	N
62	84	MD26	N
63	85	MD2	N
64	86	MD18	N
65	87	MD11	N
66	88	MD27	N
67	89	MD3	N
68	90	MD19	N
69	91	MD12	N
70	92	MD28	N
71	93	MD4	N
72	95	MD20	N
73	98	MD13	N
74	99	MD29	N
75	100	MD5	N
76	101	MD21	N
77	102	MD14	N
78	103	MD30	N

Table 9. NAND Tree Sequence (Continued)

Order	Pin #	Signal	Non-Inverting
79	104	MD6	N
80	105	MD22	N
81	106	MD15	N
82	107	MD31	N
83	108	MD7	N
84	109	MD23	N
85	112	MP0	N
86	113	MP2	N
87	114	MP1	N
88	115	MP3	N
89	116	AD0	Y
90	117	AD1	Y
91	118	AD2	Y
82	119	AD3	Y
93	123	EOL	Y
94	125	AD4	Y
95	126	AD5	Y
96	127	AD6	Y
97	128	AD7	Y
98	129	AD8	Y
99	131	AD9	Y
100	132	AD10	Y
101	133	AD11	Y
102	134	AD12	Y
103	135	AD13	Y
104	136	AD14	Y
105	137	AD15	Y
106	138	MDLE	Y
107	143	DRVPCI	N
108	144	PIG3	N
109	145	PIG2	N
110	146	PIG1	N
111	147	PIG0	N
112	148	D7	Y
113	151	HP0	Y
114	152	D8	Y
115	153	D1	Y
116	154	D5	Y
117	155	D3	Y
118	156	D10	Y
119	157	D15	Y
120	158	D13	Y
121	159	D9	Y

6.2 PLL Test Mode

The high order 82433NX LBX samples A11 at the falling edge of reset to configure the LBX for PLL test mode. When A11 is sampled low, the LBX is in normal operating mode. When A11 is sampled high, the LBX drives the internal HCLK from the PLL on the EOL pin.

82434LX/82434NX PCI, CACHE AND MEMORY CONTROLLER (PCMC)

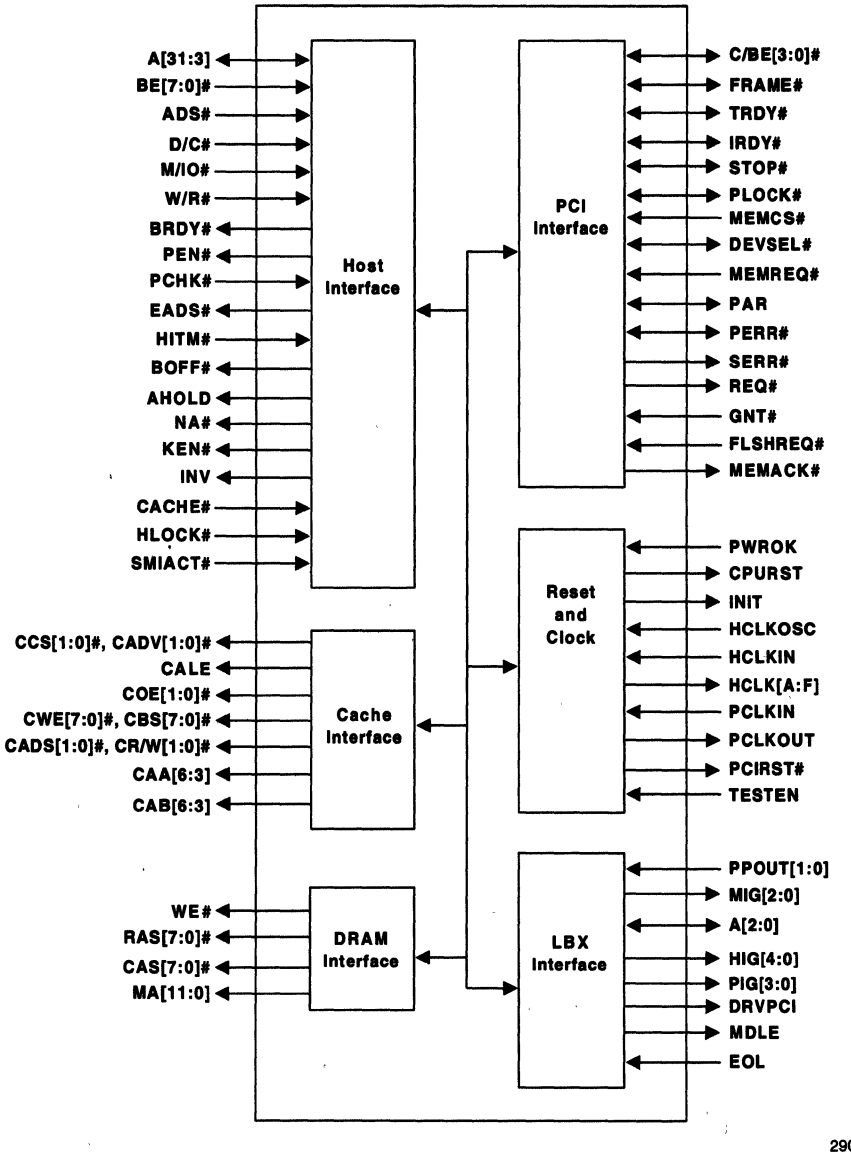
- Supports the Pentium™ Processor at ICOMP™ Index 510\60 MHz and ICOMP Index 567\66 MHz
- Supports the Pentium Processor at ICOMP Index 735\90 MHz, ICOMP Index 815\100 MHz, and ICOMP Index 610\75 MHz
- Supports Pipelined Addressing Capability of the Pentium Processor
- The 82430NX Drives 3.3V Signal Levels on the CPU and Cache Interfaces
- High Performance CPU/PCI/Memory Interfaces via Posted Write and Read Prefetch Buffers
- Fully Synchronous PCI Interface with Full Bus Master Capability
- Supports the Pentium Processor Internal Cache in Either Write-Through or Write-Back Mode
- Programmable Attribute Map of DOS and BIOS Regions for System Flexibility
- Integrated Low Skew Clock Driver for Distributing Host Clock
- Integrated Second Level Cache Controller
  - Integrated Cache Tag RAM
  - Write-Through and Write-Back Cache Modes for the 82434LX
  - Write-Back for the 82434NX
  - 82434NX Supports Low-Power Cache Standby
  - Direct Mapped Organization
  - Supports Standard and Burst SRAMs
  - 256-KByte and 512-KByte Sizes
  - Cache Hit Cycle of 3-1-1-1 on Reads and Writes Using Burst SRAMs
  - Cache Hit Cycle of 3-2-2-2 on Reads and 4-2-2-2 on Writes Using Standard SRAMs
- Integrated DRAM Controller
  - Supports 2 MBytes to 192 MBytes of Cacheable Main Memory for the 82434LX
  - Supports 2 MBytes to 512 MBytes of Cacheable Main Memory for the 82434NX
  - Supports DRAM Access Times of 70 ns and 60 ns
  - CPU Writes Posted to DRAM 4-1-1-1
  - Refresh Cycles Decoupled from ISA Refresh to Reduce the DRAM Access Latency
  - Six RAS# Lines (82434LX)
  - Eight RAS# Lines (82434NX)
  - Refresh by RAS#-Only, or CAS-Before-RAS#, in Single or Burst of Four
- Host/PCI Bridge
  - Translates CPU Cycles into PCI Bus Cycles
  - Translates Back-to-Back Sequential CPU Memory Writes into PCI Burst Cycles
  - Burst Mode Writes to PCI in Zero PCI Wait-States (i.e. Data Transfer Every Cycle)
  - Full Concurrency Between CPU-to-Main Memory and PCI-to-PCI Transactions
  - Full Concurrency Between CPU-to-Second Level Cache and PCI-to-Main Memory Transactions
  - Same Cache and Memory System Logic Design for ISA and EISA Systems
  - Cache Snoop Filter Ensures Data Consistency for PCI-to-Main Memory Transactions
- 208-Pin QFP Package

2

*Other brands and names are the property of their respective owners.

This document describes both the 82434LX and 82434NX. Unshaded areas describe the 82434LX. Shaded areas, like this one, describe 82434NX operations that differ from the 82434LX.

The 82434LX/82434NX PCI, Cache, Memory Controllers (PCMC) integrate the cache and main memory DRAM control functions and provide bus control for transfers between the CPU, cache, main memory, and the PCI Local Bus. The cache controller supports write-back (or write-through for 82434LX) cache policy and cache sizes of 256-KBytes and 512-KBytes. The cache memory can be implemented with either standard or burst SRAMs. The PCMC cache controller integrates a high-performance Tag RAM to reduce system cost.



NOTE:
RAS[7:6]# and MA11 are only on the 82434NX. CCS[1:0] functionality is only on the 82434NX.

Simplified Block Diagram of the PCMC

82434LX/82434NX PCI, CACHE AND MEMORY CONTROLLER (PCMC)

CONTENTS

CONTENTS

CONTENTS

PAGE

CONTENTS

CONTENTS

CONTENTS

1.0 ARCHITECTURAL OVERVIEW

This section provides an 82430LX/82430NX PCIset system overview that includes a description of the bus hierarchy and bridges between the buses. The 82430LX PCIset consists of the 82434LX PCMC and 82433LX LBX components plus either a PCI/ISA bridge or a PCI/EISA bridge. The 82430NX PCIset consists of the 82434NX PCMC and 82433NX LBX components plus either a PCI/ISA bridge or a PCI/EISA bridge. The PCMC and LBX provide the core cache and main memory architecture and serve as the Host/PCI bridge. An overview of the PCMC follows the system overview section.

1.1 System Overview

The 82430LX/82430NX PCIset provides the Host/PCI bridge, cache and main memory controller, and an I/O subsystem core (either PCI/EISA or PCI/ISA bridge) for the next generation of high-performance personal computers based on the Pentium processor. System designers can take advantage of the power of the PCI (Peripheral Component Interconnect) local bus while maintaining access to the large base of EISA and ISA expansion cards. Extensive buffering and buffer management within the bridges ensures maximum efficiency in all three buses (Host CPU, PCI, and EISA/ISA Buses).

For an ISA-based system, the PCIset includes the System I/O (82378IB SIO) component (Figure 1) as the PCI/ISA bridge. For an EISA-based system (Figure 2), the PCIset includes the PCI-EISA bridge (82375EB PCEB) and the EISA System Component (82374EB ESC). The PCEB and ESC work in tandem to form the complete PCI/EISA bridge.

1.1.1. BUS HIERARCHY—CONCURRENT OPERATIONS

Systems based on the 82430LX/82430NX PCIset contain three levels of buses structured in the following hierarchy:

- Host Bus as the execution bus
- PCI Bus as a primary I/O bus
- ISA or EISA Bus as a secondary I/O bus.

This bus hierarchy allows concurrency for simultaneous operations on all three buses. Data buffering permits concurrency for operations that crossover into another bus. For example, the Pentium processor could post data destined to the PCI in the LBX. This permits the Host transaction to complete in minimum time, freeing up the Host Bus for further transactions. The Pentium processor does not have to wait for the transfer to complete to its final destination. Meanwhile, any ongoing PCI Bus transactions are permitted to complete. The posted data is then transferred to the PCI Bus when the PCI Bus is available. The LBX implements extensive buffering for Host-to-PCI, Host-to-main memory, and PCI-to-main memory transactions. In addition, the PCEB/ESC chip set and the SIO implement extensive buffering for transfers between the PCI Bus and the EISA and ISA Buses, respectively.

Host Bus

Designed to meet the needs of high-performance computing, the Host Bus features:

- 64-bit data path
- 32-bit address bus with address pipelining
- Synchronous frequencies of 60 MHz and 66 MHz
- Synchronous frequency of 50 MHz (82430NX)
- Burst read and write transfers
- Support for first level and second level caches
- Capable of full concurrency with the PCI and memory subsystems
- Byte data parity
- Full support for Pentium processor machine check and DOS compatible parity reporting
- Support for Pentium processor System Management Mode (SMM).

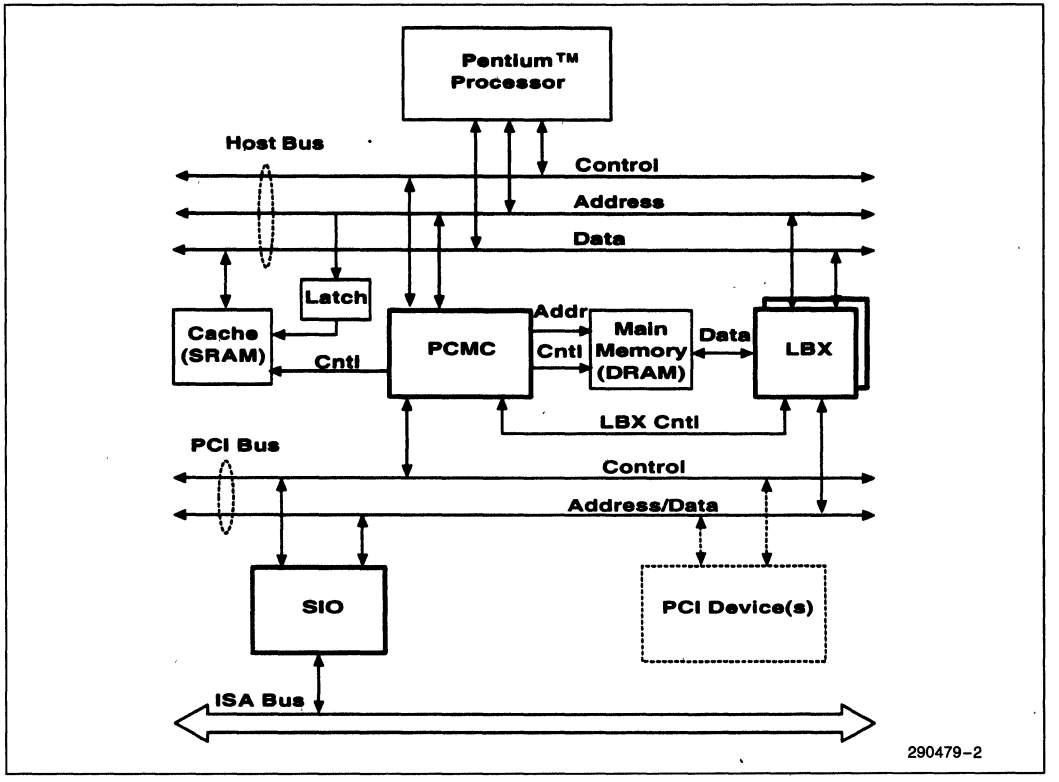


Figure 1. Block Diagram of a 82430LX/82430NX PCIset ISA System

2

PCI Bus

The PCI Bus is designed to address the growing industry needs for a standardized *local bus* that is not directly dependent on the speed and the size of the processor bus. New generations of personal computer system software such as Windows™ and Win-NT™ with sophisticated graphical interfaces, multi-tasking, and multi-threading bring new requirements that traditional PC I/O architectures cannot satisfy. In addition to the higher bandwidth, reliability and robustness of the I/O subsystem are becoming increasingly important. PCI addresses these needs and provides a future upgrade path. PCI features include:

- Processor independent
- Multiplexed, burst mode operation
- Synchronous at frequencies up to 33 MHz
- 120 MByte/sec usable throughput (132 MByte/sec peak) for a 32-bit data path

- Low latency random access (60 ns write access latency to slave registers from a master parked on the bus)
- Capable of full concurrency with the processor/memory subsystem
- Full multi-master capability allowing any PCI master peer-to-peer access to any PCI slave
- Hidden (overlapped) central arbitration
- Low pin count for cost effective component packaging (multiplexed address/data)
- Address and data parity
- Three physical address spaces: memory, I/O, and configuration
- Comprehensive support for autoconfiguration through a defined set of standard configuration functions.

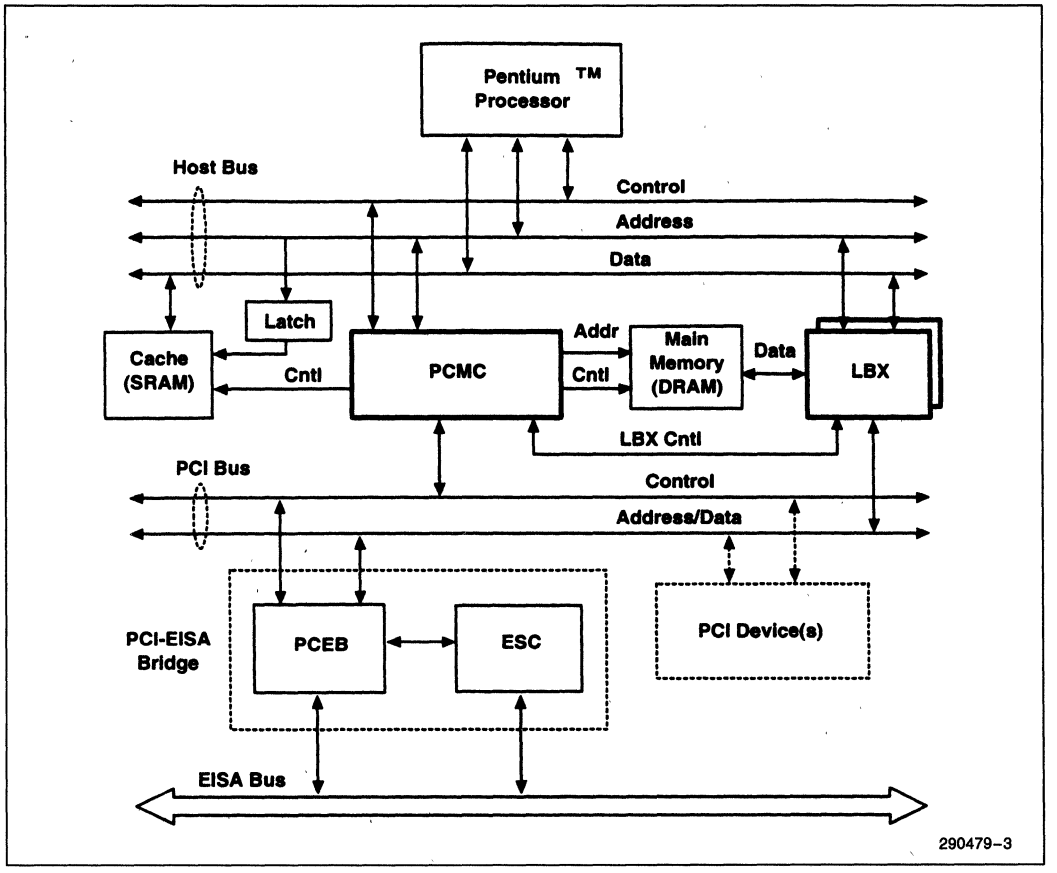


Figure 2. Block Diagram of the 82430LX/82430NX PCIset EISA System

ISA Bus

Figure 1 represents a system using the ISA Bus as the second level I/O bus. It allows personal computer platforms built around the PCI as a primary I/O bus to leverage the large ISA product base. The ISA Bus has 24-bit addressing and a 16-bit data path.

EISA Bus

Figure 2 represents a system using the EISA Bus as the second level I/O bus. It allows personal computer platforms built around the PCI as a primary I/O bus to leverage the large EISA/ISA product base. Combinations of PCI and EISA buses, both of which can be used to provide expansion functions, will satisfy even the most demanding applications.

Along with compatibility for 16-bit and 8-bit ISA hardware and software, the EISA bus provides the following key features:

- 32-bit addressing and 32-bit data path
- 33 MByte/sec bus bandwidth
- Multiple bus master support through efficient arbitration
- Support for autoconfiguration.

1.1.2 BUS BRIDGES

Host/PCI Bridge Chip Set (PCMC and LBX)

The PCMC and LBX enhance the system performance by allowing for concurrency between the Host CPU Bus and PCI Bus, giving each greater bus throughput and decreased bus latency. The LBX contains posted write buffers for Host-to-PCI, Host-to-main memory, and PCI-to-main memory transfers. The LBX also contains read prefetch buffers for Host reads of PCI, and PCI reads of main memory. There are two LBXs per system. The LBXs are controlled by commands from the PCMC. The PCMC/LBX Host/PCI bridge chip set is covered in more detail in Section 1.2, PCMC Overview.

PCI-EISA Bridge Chip Set (PCEB and ESC)

The PCEB provides the master/slave functions on both the PCI Bus and the EISA Bus. Functioning as a bridge between the PCI and EISA buses, the PCEB provides the address and data paths, bus controls, and bus protocol translation for PCI-to-EISA and EISA-to-PCI transfers. Extensive data buffering in both directions increase system performance by maximizing PCI and EISA Bus efficiency and allowing concurrency on the two buses. The PCEB's buffer management mechanism ensures data coherency. The PCEB integrates central bus control functions including a programmable bus arbiter for the PCI Bus and EISA data swap buffers for the EISA Bus. Integrated system functions include PCI parity generation, system error reporting, and programmable PCI and EISA memory and I/O address space mapping and decoding. The PCEB also contains a BIOS Timer that can be used to implement timing loops. The PCEB is intended to be used with the ESC to provide an EISA I/O subsystem interface.

The ESC integrates the common I/O functions found in today's EISA-based PCs. The ESC incorporates the logic for EISA Bus controller, enhanced seven channel DMA controller with scatter-gather support, EISA arbitration, 14 level interrupt controller, Advanced Programmable Interrupt Controller (APIC), five programmable timer/counters, non-maskable-interrupt (NMI) control, and power management. The ESC also integrates support logic to decode peripheral devices (e.g., the flash BIOS, real time clock, keyboard/mouse controller, floppy controller, two serial ports, one parallel port, and IDE hard disk drive).

PCI/ISA Bridge (SIO):

The SIO component provides the bridge between the PCI Bus and the ISA Bus. The SIO also integrates many of the common I/O functions found in today's ISA-based PCs. The SIO incorporates the logic for a PCI interface (master and slave), ISA interface (master and slave), enhanced seven channel DMA controller that supports fast DMA transfers and scatter-gather, data buffers to isolate the PCI Bus from the ISA Bus and to enhance performance, PCI and ISA arbitration, 14 level interrupt controller, a 16-bit BIOS timer, three programmable timer/counters, and non-maskable-interrupt (NMI) control logic. The SIO also provides decode for peripheral devices (e.g., the flash BIOS, real time clock, keyboard/mouse controller, floppy controller, two serial ports, one parallel port, and IDE hard disk drive).

1.2 PCMC Overview

The PCMC (along with the LBX) provides three basic functions: a cache controller, a main memory DRAM controller, and a Host/PCI bridge. This section provides an overview of these functions. Note that, in this document, operational descriptions assume that the PCMC and LBX components are used together.

1.2.1 CACHE OPERATIONS

The PCMC provides the control for a second level cache memory array implemented with either standard asynchronous SRAMs or synchronous burst SRAMs. The data memory array is external to the PCMC and located on the Host address/data bus. Since the Pentium processor contains an internal cache, there can be two separate caches in a Host subsystem. The cache inside the Pentium processor is referred to as the first level cache (also called primary cache). A detailed description of the first level cache is beyond the scope of this document. The PCMC cache control circuitry and associated external memory array is referred to as the second level cache (also called secondary cache). The second level cache is unified, meaning that both CPU data and instructions are stored in the cache. The 82434LX PCMC supports both write-through and write-back caching policies and the 82434NX supports write-back.

The optional second level cache memory array can be either 256-KBytes or 512-KBytes in size. The cache is direct-mapped and is organized as either 8K or 16K cache lines of 32 bytes per line.

In addition to the cache data RAM, the second level cache contains a 4K set of cache tags that are internal to the PCMC. Each tag contains an address that is associated with the corresponding data sector (2 lines for a 256 KByte cache and 4 lines for a 512 KByte cache) and two status bits for each line in the sector.

During a main memory read or write operation, the PCMC first searches the cache. If the addressed code or data is in the cache, the cycle is serviced by the cache. If the addressed code or data is not in the cache, the cycle is forwarded to main memory.

For the write-through (82434LX only) and write-back (both 82434LX and 82434NX) policies, the cache operation is determined by the CPU read or write cycle as follows:

Write Cycle

If the caching policy is write-through and the write cycle hits in the cache, both the cache and main memory are updated. Upon a cache miss, only main memory is updated. The cache is not updated (no write-allocate).

If the caching policy is write-back and the write cycle hits in the cache, only the cache is updated; main memory is not affected. Upon a cache miss, only main memory is updated. The cache is not updated (no write-allocate).

Read Cycle

Upon a cache hit, the cache operation is the same for both write-through and write-back. In this case, data is transferred from the cache to the CPU. Main memory is not accessed.

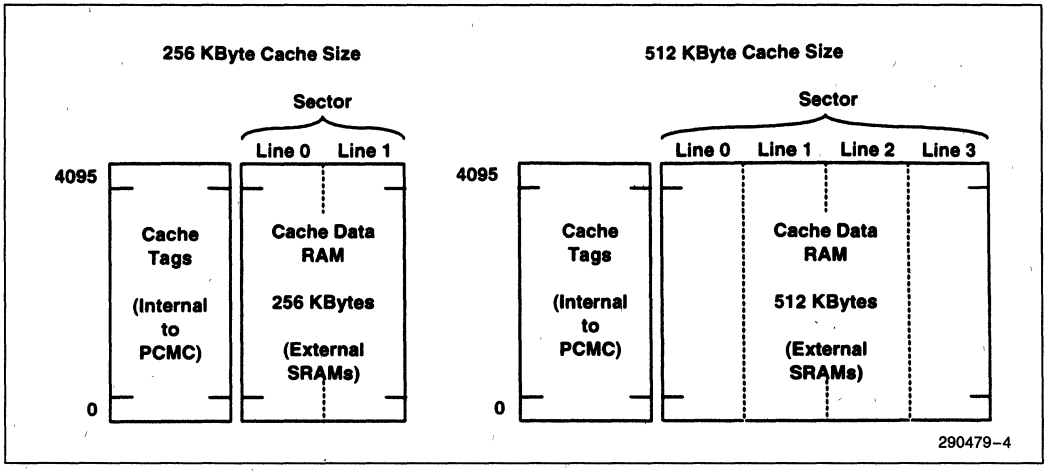


Figure 3. Second Level Cache Organization

If the read cycle causes a cache miss, the line containing the requested data is transferred from main memory to the cache and to the CPU. In the case of a write-back cache, if the cache line fill is to a sector containing one or more modified lines, the modified lines are written back to main memory and the new line is brought into the cache. For a modified line write-back operation, the PCMC transfers the modified cache lines to main memory via a write buffer in the LBX. Before writing the last modified line from the write buffer to main memory, the PCMC updates the first and second level caches with the new line, allowing the CPU access to the requested data with minimum latency.

1.2.1.1 Cache Consistency

The Snoop mechanism in the PCMC ensures data consistency between cache (both first level and second level) and main memory. The PCMC monitors PCI master accesses to main memory and when needed, initiates an inquire (snoop) cycle to the first and second level caches. The snoop mechanism guarantees that consistent data is always delivered to both the host CPU and PCI masters.

1.2.2 ADDRESS/DATA PATHS

Address paths between the CPU/cache and PCI and data paths between the CPU/cache, PCI, and main memory are supplied by two LBX components. The LBX is a companion component to the PCMC. Together, they form a Host/PCI bridge. The PCMC (via the PCMC/LBX interface signals), controls the address and data flow through the LBXs. Refer to the LBX data sheet for more details on the address and data paths.

Data is transferred to and from the PCMC internal registers via the PCMC address lines. When the Host CPU performs a write operation, the data is sent to the LBXs. When the PCMC decodes the cycle as an access to one of its internal registers, it asserts AHOLD to the CPU and instructs the LBXs to copy the data onto the Host address lines. When the PCMC decodes a Host read as an access to a PCMC internal register, it asserts AHOLD to the CPU. The PCMC then places the register data on its address lines and instructs the LBX to copy the data on the Host address bus to the Host data bus. When the register data is on the Host data bus, the PCMC negates AHOLD and completes the cycle.

1.2.2.1 Read/Write Buffers

The LBX provides an interface for the CPU address and data buses, PCI Address/Data bus, and the main memory DRAM data bus. There are three posted write buffers and one read-prefetch buffers implemented in the LBXs to increase performance and to maximize concurrency. The buffers are:

- CPU-to-Main Memory Posted Write Buffer (4 Qwords)
- CPU-to-PCI Posted Write Buffer (4 Dwords)
- PCI-to-Main Memory Posted Write Buffer (2 x 4 Dwords)
- PCI-to-Main Memory Read Prefetch Buffer (line buffer, 4 Qwords).

Refer to the LBX data sheet for details on the operation of these buffers.

2

1.2.3 HOST/PCI BRIDGE OPERATIONS

The PCMC permits the Host CPU to access devices on the PCI Bus. These accesses can be to PCI I/O space, PCI memory space, or PCI configuration space.

As a PCI device, the PCMC can be either a master initiating a PCI Bus operation or a target responding to a PCI Bus operation. The PCMC is a PCI Bus master for Host-to-PCI cycles and a target for PCI-to-main memory transfers. Note that the PCMC does not permit peripherals to be located on the Host Bus. CPU I/O cycles, other than to PCMC internal registers, are forwarded to the PCI Bus and PCI Bus accesses to the Host Bus are not supported.

When the CPU initiates a bus cycle to a PCI device, the PCMC becomes a PCI Bus master and translates the CPU cycle into the appropriate PCI Bus cycle. The Host/PCI Posted write buffer in the LBXs permits the CPU to complete CPU-to-PCI Dword memory writes in three CPU clocks (1 wait-state), even if the PCI Bus is currently busy. The posted data is written to the PCI device when the PCI Bus is available.

When a PCI Bus master initiates a main memory access, the PCMC (and LBXs) become the target of the PCI Bus cycle and responds to the read/write access. During PCI-to-main memory accesses, the PCMC automatically performs cache snoop operations on the Host Bus, when needed, to maintain data consistency.

As a PCI device, the PCMC contains all of the required PCI configuration registers. The Host CPU reads and writes these registers as described in Section 3.0, Register Description.

1.2.4 DRAM MEMORY OPERATIONS

The PCMC contains a DRAM controller that supports CPU and PCI master accesses to main memory. The PCMC DRAM interface supplies the control signals and address lines and the LBXs supply the data path. DRAM parity is generated for main memory writes and checked for memory reads.

For the 82434LX, the memory array is 64-bits wide and ranges in size from 2 MBytes–192 MBytes. The array can be implemented with either single-sided or double-sided SIMMs. DRAM SIMM sizes of 256K x 36, 1M x 36, and 4M x 36 are supported.

For the 82434NX, the memory array is 64-bits wide and ranges in size from 2 MBytes–512 MBytes. The array can be implemented with either single-sided or double-sided SIMMs. DRAM SIMM sizes of 256K x 36, 1M x 36, 4M x 36, and 16M x 36 are supported.

To provide optimum support for the various cache configurations, and the resultant mix of bus cycles, the system designer can select between 0-active RAS# and 1-active RAS# modes. These modes affect the behavior of the RAS# signal following either CPU-to-main memory cycles or PCI-to-main memory cycles.

The PCMC also provides programmable memory and cacheability attributes on 14 memory segments of various sizes in the ISA compatibility range (512 KByte–1 MByte address range). Access rights to these memory segments from the PCI Bus are controlled by the expansion bus bridge.

The PCMC permits a gap to be created in main memory within the 1 MByte–16 MBytes address range, accommodating ISA devices which are mapped into this range (e.g., ISA LAN card or an ISA frame buffer).

1.2.5 3.3V SIGNALS

The 82434NX PCMC drives 3.3V signal levels on the CPU and second level cache interfaces. Thus, no extra logic (i.e. 5V/3.3V translation) is required when interfacing to 3.3V processors and SRAMs. Six of the power pins on the 82434NX are VDD3 pins. These pins are connected to a 3.3V power supply. The VDD3 pins power the output buffers on the CPU and second level cache interfaces. The VDD3 pins also power the output buffers for the HCLK[A-F] outputs.

2.0 SIGNAL DESCRIPTIONS

This section provides a detailed description of each signal. The signals are arranged in functional groups according to their associated interface. The states of all of the signals during hard reset are provided in Section 8.0, System Clocking and Reset.

The "#" symbol at the end of a signal name indicates that the active, or asserted state occurs when the signal is at a low voltage level. When "#" is not present after the signal name, the signal is asserted when at the high voltage level.

The terms assertion and negation are used extensively. This is done to avoid confusion when working with a mixture of "active-low" and "active-high" signals. The term **assert**, or **assertion** indicates that a signal is active, independent of whether that level is represented by a high or low voltage. The term **negate**, or **negation** indicates that a signal is inactive.

The following notations are used to describe the signal type.

in Input is a standard input-only signal

out Totem pole output is a standard active driver

o/d Open drain

t/s Tri-State is a bi-directional, tri-state input/output pin

s/t/s Sustained tri-state is an active low tri-state signal owned and driven by one and only one agent at a time. The agent that drives a s/t/s pin low must drive it high for at least one clock before letting it float. A new agent can not start driving a s/t/s signal any sooner than one clock after the previous owner tri-states it. An external pull-up is required to sustain the inactive state until another agent drives it and must be provided by the central resource.

2.1 Host Interface

Signal	Type	Description
A[31:0]	t/s	**ADDRESS BUS:** A[31:0] are the address lines of the Host Bus. A[31:3] are connected to the CPU A[31:3] lines and to the LBXs. A[2:0] are only connected to the LBXs. Along with the byte enable signals, the A[31:3] lines define the physical area of memory or I/O being accessed. During CPU cycles, the A[31:3] lines are inputs to the PCMC. They are used for address decoding and second level cache tag lookup sequences. Also during CPU cycles, A[2:0] are outputs and are generated from BE[7:0]#. A[27:24] provide hardware strapping options for test features. For more details on theses options, refer to Section 11.0 Testability. During inquire cycles, A[31:5] are inputs from the LBXs to the CPU and the PCMC to snoop the first and the second level cache tags, respectively. In response to a Flush or Flush Acknowledge Special Cycle, the PCMC asserts AHOLD and drives the addresses of the second level cache lines to be written back to main memory on A[18:7]. During CPU to PCI configuration cycles, the PCMC drives A[31:0] with the PCI configuration space address that is internally derived from the CPU physical I/O address. All PCMC internal configuration registers are accessed via A[31:0]. During CPU reads from PCMC internal configuration registers, the PCMC asserts AHOLD and drives the contents of the addressed register on A[31:0]. The PCMC then signals the LBXs to copy this value from the address lines onto the host data lines. During writes to PCMC internal configuration registers, the PCMC asserts AHOLD and signals the LBXs to copy the write data onto the A[31:0] lines. Finally, when in deturbo mode, the PCMC periodically asserts AHOLD and then drives A[31:0] to valid logic levels to keep these lines from floating for an extended period of time. A[31:28] provide hardware strapping options at powerup. For more details on strapping options, refer to Section 8.0, System Clocking and Reset. A[27:24] provide hardware strapping options for test features. For more details on these options, refer to Section 11.0 Testability.

Signal	Type	Description
BE[7:0]#	in	**BYTE ENABLES:** The byte enables indicate which byte lanes on the CPU data bus carry valid data during the current bus cycle. In the case of cacheable reads, all 8 bytes of data are driven to the Pentium processor, regardless of the state of the byte enables. The byte enable signals indicate the type of special cycle when M/IO# = D/C# = 0 and W/R# = 1. During special cycles, only one byte enable is asserted by the CPU. The following table depicts the special cycle types and their byte enable encodings: **Special Cycle Type** — **Asserted Byte Enable** Shutdown — BE0# Flush — BE1# Halt/Stop Grant — BE2# Write Back — BE3# Flush Acknowledge — BE4# Branch Trace Message — BE5# When the PCMC decodes a Shutdown Special Cycle, it asserts AHOLD, drives 000...000 (the PCI Shutdown Special Cycle Encoding) on the A[31:0] lines and signals the LBXs to latch the host address bus. The PCMC then drives a Special Cycle on PCI, signaling the LBXs to drive the latched address (00...00) on the AD[31:0] lines during the data phase. The PCMC then asserts INIT for 16 HCLKs. In response to Flush and Flush Acknowledge Special Cycles, the PCMC internally inspects the Valid and Modified bits for each of the Second Level Cache Sectors. If a line is both valid and modified, the PCMC drives the cache address of the line on the A[18:7] and CAA/CAB[6:3] lines and writes the line back to main memory. The valid and modified bits are both reset to 0. All valid and unmodified lines are simply marked invalid. In response to a write back special cycle, the PCMC simply returns BRDY# to the CPU. The second level cache will be written back to main memory in response to the following flush special cycle. If BE2# is asserted during a special cycle, the 82434NX uses A4 to determine if the cycle is a Halt or Stop Grant Special Cycle. If A4 = 0, the cycle is a Halt Special Cycle and if A4 = 1, the cycle is a Stop Grant Special cycle. In response to a halt special cycle, the PCMC asserts AHOLD, drives 000...001 (the PCI halt special cycle encoding) on the A[31:0] lines, and signals the LBXs to latch the host address bus. The PCMC then drives a special cycle on PCI, signaling the LBXs to drive the latched address (00...01) on the AD[31:0] lines during the data phase. When the 82434NX PCMC detects a CPU Stop Grant Special Cycle (M/IO# = 0, D/C# = 0, W/R# = 1, A4 = 1, BE[7:0]# = FBh), it generates a PCI Stop Grant Special cycle, with 0002h in the message field (AD[15:0]) and 0012h in the message dependent data field (AD[31:16]) during the first data phase (IRDY# asserted).
ADS#	in	**ADDRESS STROBE:** The Pentium processor asserts ADS# to indicate that a new bus cycle is beginning. ADS# is driven active in the same clock as the address, byte enable, and cycle definition signals. The PCMC ignores a floating low ADS# that may occur when BOFF# is asserted as the CPU is asserting ADS#.

Signal	Type	Description
BRDY#	out	**BURST READY:** BRDY# indicates that the system has responded in one of three ways: 1. valid data has been placed on the Pentium processor data pins in response to a read, 2. CPU write data has been accepted by the system, or 3. the system has responded to a special cycle.
NA#	out	**NEXT ADDRESS:** The PCMC asserts NA# for one clock when the memory system is ready to accept a new address from the CPU, even if all data transfers for the current cycle have not completed. The CPU may drive out a pending cycle two clocks after NA# is asserted and has the ability to support up to two outstanding bus cycles.
AHOLD	out	**ADDRESS HOLD:** The PCMC asserts AHOLD to force the Pentium processor to stop driving the address bus so that either the PCMC or LBXs can drive the bus. During PCI master cycles, AHOLD is asserted to allow the LBXs to drive a snoop address onto the address bus. If the PCI master locks main memory, AHOLD remains asserted until the PCI master locked sequence is complete and the PCI master negates PLOCK#. AHOLD is asserted during all accesses to PCMC internal configuration registers to allow configuration register accesses to occur over the A[31:0] lines. When in deturbo mode, the PCMC periodically asserts AHOLD to prevent the processor from initiating bus cycles in order to emulate a slower system. The duration of AHOLD assertion in deturbo mode is controlled by the Deturbo Frequency Control Register (offset 51h). When PWROK is negated, the PCMC asserts AHOLD to allow the strapping options on A[31:28] to be read. For more details on strapping options, see the System Clocking and Reset section.
EADS#	out	**EXTERNAL ADDRESS STROBE:** The PCMC asserts EADS# to indicate to the Pentium processor that a valid snoop address has been driven onto the CPU address lines to perform an inquire cycle. During PCI master cycles, the PCMC signals the LBXs to drive a snoop address onto the host address lines and then asserts EADS# to cause the CPU to sample the snoop address.
INV	out	**INVALIDATE:** The INV signal specifies the final state (invalid or shared) that a first level cache line transitions to in the event of a cache line hit during a snoop cycle. When snooping the caches during a PCI master write, the PCMC asserts INV with EADS#. When INV is asserted with EADS#, an inquire hit results in the line being invalidated. When snooping the caches during a PCI master read, the PCMC does not assert INV with EADS#. In this case, an inquire cycle hit results in a line transitioning to the shared state.
BOFF#	out	**BACKOFF:** The PCMC asserts BOFF# to force the Pentium processor to abort all outstanding bus cycles that have not been completed and float its bus in the next clock. The PCMC uses this signal to force the CPU to re-order a write-back due to a snoop cycle around a currently outstanding bus cycle. The PCMC also asserts BOFF# to obtain the CPU data bus for write-back cycles from the secondary cache due to a snoop hit. The CPU remains in bus hold until BOFF# is negated.
HITM#	in	**HIT MODIFIED:** The Pentium processor asserts HITM# to inform the PCMC that the current inquire cycle hit a modified line. HITM# is asserted by the Pentium processor two clocks after the assertion of EADS# if the inquire cycle hits a modified line in the primary cache.

Signal	Type	Description
M/IO# D/C# W/R#	in	**BUS CYCLE DEFINITION (MEMORY/INPUT-OUTPUT, DATA/CONTROL, WRITE/READ):** M/IO, D/C# and W/R# define Host Bus cycles as shown in the table below. **M/IO# D/C# W/R# Bus Cycle Type** Low Low Low Interrupt Acknowledge Low Low High Special Cycle Low High Low I/O Read Low High High I/O Write High Low Low Code Read High Low High Reserved High High Low Memory Read High High High Memory Write Interrupt acknowledge cycles are forwarded to the PCI Bus as PCI interrupt acknowledge cycles (i.e. C/BE[3:0]# = 0000 during the address phase). All I/O cycles and any memory cycles that are not directed to memory controlled by the PCMC DRAM controller are forwarded to PCI. The Pentium processor generates six different types of special cycles. The special cycle type is encoded on the BE[7:0]# lines.
HLOCK#	in	**HOST BUS LOCK:** The Pentium processor asserts HLOCK# to indicate the current bus cycle is locked. HLOCK# is asserted in the first clock of the first locked bus cycle and is negated after the BRDY# is returned for the last locked bus cycle. The Pentium processor guarantees HLOCK# to be negated for at least one clock between back-to-back locked operations. When a CPU locked cycle is directed to main memory, the PCMC guarantees that once the locked operation begins in main memory, the CPU has exclusive access to main memory (i.e., PCI master accesses to main memory will not be initiated until the CPU locked operation completes). When a CPU locked cycle is directed to PCI, the PCMC arbitrates for PLOCK# (PCI LOCK#) before initiating the cycle on PCI, except when the cycle is to the memory range defined by the Frame Buffer Range Register and the No Lock Requests bit in that register is set to 1.
CACHE#	in	**CACHEABILITY:** The Pentium processor asserts CACHE# to indicate the internal cacheability of a read cycle or that a write cycle is a burst write-back cycle. If the CPU drives CACHE# inactive during a read cycle, the returned data is not cached, regardless of the state of KEN#. The CPU asserts CACHE# for cacheable data reads, cacheable code fetches, and cache line write-backs. CACHE# is driven along with the cycle definition pins.
KEN#	out	**CACHE ENABLE:** The PCMC asserts KEN# to indicate to the CPU that the current cycle is cacheable. KEN# is asserted for all accesses to memory ranges 0–512-KBytes and 1024-KBytes to the top of main memory controlled by the PCMC when the Primary Cache Enable bit is set to 1, except in the following case: KEN# is not asserted for accesses to the top 64-KByte of main memory controlled by the PCMC when the SMRAM Enable bit in the DRAM Control Register (Offset 57h) is set to 1 and the area is not write protected. If the area is write protected and cacheable, KEN# is asserted for code read cycles, but is not asserted during data read cycle. KEN# is asserted for any CPU access within the range of 512-KBytes–1024-KBytes if the corresponding Cache Enable bit in the PAM[6:0] Registers (offsets 59h–5Fh) is set to 1. When the Pentium processor indicates that the current read cycle can be cached by asserting CACHE# and the PCMC responds with KEN#, the cycle is converted into a burst cache line fill. The CPU samples KEN# with the first of either BRDY# or NA#.

Signal	Type	Description
SMIACT#	in	**SYSTEM MANAGEMENT INTERRUPT ACTIVE:** The Pentium processor asserts SMIACT# to indicate that the processor is operating in System Management Mode (SMM). When the SMRAM Enable bit in the DRAM Control Register (offset 57h) is set to 1, the PCMC allows CPU accesses SMRAM as permitted by the SMRAM Space Register at configuration space offset 72h.
PEN#	out	**PARITY ENABLE:** The PEN# signal, along with the MCE bit in CR4 of the Pentium processor, determines whether a machine check exception will be taken by the CPU as a result of a parity error on a read cycle. The PCMC asserts PEN# during DRAM read cycles if the MCHK on DRAM/L2 Cache Data Parity Error Enable bit in the Error Command Register (offset 70h) is set to 1. The PCMC asserts PEN# during CPU second level cache read cycles if the MCHK on DRAM/L2 Cache Data Parity Error Enable and the L2 Cache Parity Enable bits in the Error Command Register (offset 70h) are both set to 1.
PCHK#	in	**DATA PARITY CHECK:** PCHK# is sampled by the PCMC to detect parity errors on CPU read cycles from main memory if the Parity Error Mask Enable bit in the DRAM Control Register (offset 57h) is reset to 0. PCHK# is sampled by the PCMC to detect parity errors on CPU read cycles from the second level cache if the L2 Cache Parity Enable bit in the Error Command Register (offset 70h) is set to 1. If incorrect parity was detected on a data read, the PCHK# signal is asserted by the Pentium processor two clocks after BRDY# is returned. PCHK# is asserted for one clock for each clock in which a parity error was detected.

2.2 DRAM Interface

Signal	Type	Description
RAS[5:0]#	out	**ROW ADDRESS STROBES:** The RAS[5:0]# signals are used to latch the row address on the MA[10:0] lines into the DRAMs. Each RAS[5:0]# signal corresponds to one DRAM row. The 82434LX PCMC supports up to 6 rows in the DRAM array. Each row is eight bytes wide. These signals drive the RAS# lines of the DRAM array directly, without external buffers.
RAS[7:6]#	out	**ROW ADDRESS STROBES:** The 82434NX supports up to eight rows of DRAM. RAS[7:6]# are used with RAS[5:0] to latch the row address on the MA[11:0] lines into the DRAMs. Each row is eight bytes wide. These signals drive the RAS# lines of the DRAM array directly, without external buffers.
CAS[7:0]#	out	**COLUMN ADDRESS STROBES:** The CAS[7:0]# signals are used to latch the column address on the MA[10:0] lines into the DRAMs. Each CAS[7:0]# signal corresponds to one byte of the eight byte-wide array. These signals drive the CAS# lines of the DRAM array directly, without external buffers. In a minimum configuration, each CAS[7:0]# line only has one SIMM load, while the maximum configuration has 6 SIMM loads.
WE#	out	**DRAM WRITE ENABLE:** WE# is asserted during both CPU and PCI master writes to main memory. During burst writes to main memory, WE# is asserted before the first assertion of CAS[7:0]# and is negated with the last CAS[7:0]#. The WE# signal is externally buffered to drive the WE# inputs on the DRAMs.
MA[10:0]	out	**DRAM MULTIPLEXED ADDRESS:** MA[10:0] provide the row and column address to the DRAM array. The 82434LX uses MA[10:0] for the complete DRAM address bus. The MA[10:0] lines are externally buffered to drive the multiplexed address lines of the DRAM array.
MA11	out	**DRAM MULTIPLEXED ADDRESS:** MA11 provides the extra addressability for the 16M x 36 SIMMs that are supported by the 82434NX. MA[11:0] provide the row and column address to the DRAM array. Like MA[10:0], MA11 is externally buffered to drive the multiplexed address lines of the DRAM array.

2.3 Cache Interface

Signal	Type	Description
CALE	out	**CACHE ADDRESS LATCH ENABLE:** CALE controls the external latch between the host address lines and the cache address lines. CALE is asserted to open the external latch, allowing the host address lines to propagate to the cache address lines. CALE is negated to latch the cache address lines.
CADS[1:0]#, CR/W[1:0]#	out	This signal pin has two functions, depending on the type of SRAMs used for the second level cache. **CACHE ADDRESS STROBE:** CADS[1:0]# are used with burst SRAMs. When asserted, CADS[1:0]# cause the burst SRAMs to latch the cache address on the rising edge of HCLK. CADS[1:0]# are glitch-free synchronous signals. CADS[1:0]# functionality is selected by the SRAM type bit in the Secondary Cache Control Register. Two copies of this signal are provided for timing reasons only. **CACHE READ/WRITE:** CR/W# provide read/write control to the second level cache when using asynchronous dual-byte select SRAMs. This functionality is selected by the SRAM Type and Cache Byte Control Bits in the Secondary Cache Control Register. The two copies of this signal are always driven to the same logic level.
CADV[1:0]#, CCS[1:0]#	out	This signal pin has two functions. The Cache Chip Select function is only enabled when the SRAM connectivity bit (bit 2) in the SCC Register is set to 1. **CACHE ADVANCE:** CADV[1:0]# are used with burst SRAMs to advance the internal two bit address counter inside the SRAMs to the next address of the burst sequence. Two copies of this signal are provided for timing reasons only. The two copies are always driven to the same logic level. **CACHE CHIP SELECT:** CCS[1:0]# are used with asynchronous SRAMs to de-select the SRAMs, placing them in a low power standby mode. When the CPU runs a halt or stop grant special cycle, the 82434NX negates CCS[1:0]#, placing the second level cache in a power saving mode. The PCMC then asserts CCS[1:0]# (activating the SRAMs) when the CPU asserts ADS#. When using burst SRAMs, only CCS1# implements the CCS# function. CADV0# retains the address advance function. CCS1# serve two purposes with burst SRAMs: 1) It is used (along with CADS[1:0]#) to place the SRAMs in a low power standby mode. When the CPU runs a halt or stop grant special cycle, the 82434NX negates CCS1# and asserts CADS[1:0]# for one clock, placing the SRAMs in a power saving mode. The PCMC then asserts CCS1# so that the next ADS# from the CPU places the SRAMs in an active mode. 2) CCS1# is used to block pipelined cycles from the SRAMs when the SRAMs are servicing a cycle. After NA# is asserted, the PCMC negates CCS1# preventing the SRAMs from sampling a new address. CCS1# is asserted again when the SRAMs have completed the current cycle.
CAA[6:3] CAB[6:3]	out	**CACHE ADDRESS [6:3]:** CAA[6:3] and CAB[6:3] are connected to address lines A[3:0] on the second level cache SRAMs. CAA[4:3] and CAB[4:3] are used with standard SRAMs to advance through the burst sequence. CAA[6:5] and CAB[6:5] are used during second level cache write-back cycles to address the modified lines within the addressed sector. Two copies of these signals are provided for timing reasons only. The two copies are always driven to the same logic level.

Signal	Type	Description
COE[1:0]#	out	**CACHE OUTPUT ENABLE:** COE[1:0]# are asserted when data is to be read from the second level cache and are negated at all other times. Two copies of this signal are provided for timing reasons only. The two copies are always driven to the same logic level.
CWE[7:0]#, CBS[7:0]#	out	This signal pin has two functions, depending on the type of SRAMs used for the second level cache. **CACHE WRITE ENABLES:** CWE[7:0]# are asserted to write data to the second level cache SRAMs on a byte-by-byte basis. CWE7# controls the most significant byte while CWE0# controls the least significant byte. These signals are cache write enables when using burst SRAMs (SRAM Type bit in SCC Register is 1) or when using asynchronous SRAMs (SRAM Type bit in SCC Register is 0) and the Cache Byte Control Bit is 1. **CACHE BYTE SELECTS:** The CBS[7:0]# lines provide byte control to the secondary cache when using dual-byte select asynchronous SRAMs. These signals are Cache Byte select lines when the SRAM Type and Cache Byte Control Bits in the SCC Register are both 0.

2.4 PCI Interface

Signal	Type	Description
C/BE[3:0]#	t/s	**PCI BUS COMMAND AND BYTE ENABLES:** C/BE[3:0]# are driven by the current bus master during the address phase of a PCI cycle to define the PCI command, and during the data phase as the PCI byte enables. The PCI commands indicate the current cycle type, and the PCI byte enables indicate which byte lanes carry meaningful data. C/BE[3:0]# are outputs of the PCMC during CPU cycles that are directed to PCI. C/BE[3:0]# are inputs when the PCMC acts as a slave. The command encodings and types are listed below. **C/BE[3:0]# Command** 0000 Interrupt Acknowledge 0001 Special Cycle 0010 I/O Read 0011 I/O Write 0100 Reserved 0101 Reserved 0110 Memory Read 0111 Memory Write 1000 Reserved 1001 Reserved 1010 Configuration Read 1011 Configuration Write 1100 Memory Read Multiple 1101 Reserved 1110 Memory Read Line 1111 Memory Write and Invalidate

Signal	Type	Description
FRAME#	s/t/s	**CYCLE FRAME:** FRAME# is driven by the current bus master to indicate the beginning and duration of an access. FRAME# is asserted to indicate that a bus transaction is beginning. While FRAME# is asserted, data transfers continue. When FRAME# is negated, the transaction is in the final data phase. FRAME# is an output of the PCMC during CPU cycles which are directed to PCI. FRAME# is an input to the PCMC when the PCMC acts as a slave.
IRDY#	s/t/s	**INITIATOR READY:** The assertion of IRDY# indicates the current bus master's ability to complete the current data phase. IRDY# works in conjunction with TRDY# to indicate when data has been transferred. On PCI, data is transferred on each clock that both IRDY# and TRDY# are asserted. During read cycles, IRDY# is used to indicate that the master is prepared to accept data. During write cycles, IRDY# is used to indicate that the master has driven valid data on the AD[31:0] lines. Wait states are inserted until both IRDY# and TRDY# are asserted together. IRDY# is an output of the PCMC when the PCMC is the PCI master. IRDY# is an input to the PCMC when the PCMC acts as a slave.
TRDY#	s/t/s	**TARGET READY:** TRDY# indicates the target device's ability to complete the current data phase of the transaction. It is used in conjunction with IRDY#. A data phase is completed on each clock that TRDY# and IRDY# are both sampled asserted. During read cycles, TRDY# indicates that valid data is present on AD[31:0] lines. During write cycles, TRDY# indicates the target is prepared to accept data. Wait states are inserted on the bus until both IRDY# and TRDY# are asserted together. TRDY# is an output of the PCMC when the PCMC is the PCI slave. TRDY# is an input to the PCMC when the PCMC is a master.
DEVSEL#	s/t/s	**DEVICE SELECT:** When asserted, DEVSEL# indicates that the driving device has decoded its address as the target of the current access. DEVSEL# is an output of the PCMC when PCMC is a PCI slave and is derived from the MEMCS# input. MEMCS# is generated by the expansion bus bridge as a decode to the main memory address space. During CPU-to-PCI cycles, DEVSEL# is an input. It is used to determine if any device has responded to the current bus cycle, and to detect a target abort cycle. Master-Abort termination results if no subtractive decode agent exists in the system, and no one asserts DEVSEL# within a programmed number of clocks.
STOP#	s/t/s	**STOP:** STOP# indicates that the current target is requesting the master to stop the current transaction. This signal is used in conjunction with DEVSEL# to indicate disconnect, target-abort, and retry cycles. When PCMC is acting as a master on PCI, if STOP# is sampled active on a rising edge of PCLKIN, FRAME# is negated within a maximum of 3 clock cycles. STOP# may be asserted by the PCMC in three cases. If a PCI master attempts to access main memory when another PCI master has locked main memory, the PCMC asserts STOP# to signal retry. The PCMC detects this condition when sampling FRAME# and LOCK# both active during an address phase. When a PCI master is reading from main memory, the PCMC asserts STOP# when the burst cycle is about to cross a cache line boundary. When a PCI master is writing to main memory, the PCMC asserts STOP# upon filling either of the two PCI-to-main memory posted write buffers. Once asserted, STOP# remains asserted until FRAME# is negated.

Signal	Type	Description
PLOCK#	s/t/s	**PCI LOCK:** PLOCK# is used to indicate an atomic operation that may require multiple transactions to complete. PCI provides a mechanism referred to as "resource lock" in which only the target of the PCI transaction is locked. The assertion of GNT# on PCI does not guarantee control of the PLOCK# signal. Control of PLOCK# is obtained under its own protocol. When the PCMC is the PCI slave, PLOCK# is sampled as an input on the rising edge of PCLKIN when FRAME# is sampled active. If PLOCK# is sampled asserted, the PCMC enters into a locked state and remains in the locked state until PLOCK# is sampled negated on a following rising edge of PCLKIN, when FRAME# is sampled asserted.
REQ#	out	**REQUEST:** The PCMC asserts REQ# to indicate to the PCI bus arbiter that the PCMC is requesting use of the PCI Bus in response to a CPU cycle directed to PCI.
GNT#	in	**GRANT:** When asserted, GNT# indicates that access to the PCI Bus has been granted to the PCMC by the PCI Bus arbiter.
MEMCS#	in	**MAIN MEMORY CHIP SELECT:** When asserted, MEMCS# indicates to the PCMC that a PCI master cycle is targeting main memory. MEMCS# is generated by the expansion bus bridge. MEMCS# is sampled by the PCMC on the rising edge of PCLKIN on the first and second cycle after FRAME# has been asserted.
FLSHREQ#	in	**FLUSH REQUEST:** When asserted, FLSHREQ# instructs the PCMC to flush the CPU-to-PCI posted write buffer in the LBXs and to disable further posting to this buffer as long as FLSHREQ# remains active. The PCMC acknowledges completion of the CPU-to-PCI write buffer flush operation by asserting MEMACK#. MEMACK# remains asserted until FLSHREQ# is negated. FLSHREQ# is driven by the expansion bus bridge and is used to avoid deadlock conditions on the PCI Bus.
MEMREQ#	in	**MEMORY REQUEST:** When asserted, MEMREQ# instructs the PCMC to flush the CPU-to-PCI and CPU-to-main memory posted write buffers and to disable posting in these buffers as long as MEMREQ# is active. The PCMC acknowledges completion of the flush operations by asserting MEMACK#. MEMACK# remains asserted until MEMREQ# is negated. MEMREQ# is driven by the expansion bus bridge.
MEMACK#	out	**MEMORY ACKNOWLEDGE:** When asserted, MEMACK# indicates the completion of the operations requested by an active FLSHREQ# and/or MEMREQ#.
PAR	t/s	**PARITY:** PAR is an even parity bit across the AD[31:0] and C/BE[3:0]# lines. Parity is generated on all PCI transactions. As a master, the PCMC generates even parity on CPU writes to PCI, based on the PPOUT[1:0] inputs from the LBXs. During CPU read cycles from PCI, the PCMC checks parity by checking the value sampled on the PAR input with the PPOUT[1:0] inputs from the LBXs. As a slave, the PCMC generates even parity on PAR, based on the PPOUT[1:0] inputs during PCI master reads from main memory. During PCI master writes to main memory, the PCMC checks parity by checking the value sampled on PAR with the PPOUT[1:0] inputs.

Signal	Type	Description
PERR#	s/t/s	**PARITY ERROR:** PERR# may be pulsed by any agent that detects a parity error during an address phase, or by the master or the selected target during any data phase in which the AD lines are inputs. The PERR# signal is enabled when the PERR# on Receiving Data Parity Error bit in the Error Command Register (offset 70h) and the Parity Error Enable bit in the PCI Command Register (offset 04h) are both set to 1. When enabled, CPU-to-PCI write data is checked for parity errors by sampling the PERR# signal two PCI clocks after data is driven. Also, when enabled, PERR# is asserted by the PCMC when it detects a data parity error on CPU read data from PCI and PCI master write data to main memory. PERR# is neither sampled nor driven by the PCMC when either the PERR# on Receiving Data Parity Error bit in the Error Command Register or the Parity Error Enable bit in the PCI Command Register is reset to 0.
SERR#	o/d	**SYSTEM ERROR:** SERR# may be pulsed by any agent for reporting errors other than parity. SERR# is asserted by the PCMC whenever a serious system error (not necessarily a PCI error) occurs. The intent is to have the PCI central agent (for example, the expansion bus bridge) assert NMI to the processor. Control over the SERR# signal is provided via the Error Command Register (offset 70h) when the Parity Error Enable bit in the PCI Command Register (offset 04h) is set to 1. When the SERR# DRAM/L2 Cache Data Parity Error bit is set to 1, SERR# is asserted upon detecting a parity error on CPU read cycles from DRAM. If the L2 Cache Parity bit is also set to 1, SERR# will be asserted upon detecting a parity error on CPU read cycles from the second level cache. The Pentium processor indicates these parity errors to the PCMC via the PCHK# signal. When the SERR# on PCI Address Parity Error bit is set to 1, the PCMC asserts SERR# if a parity error is detected during the address phase of a PCI master cycle. When the SERR# on Received PCI Data Parity bit is set to 1, the PCMC asserts SERR# if a parity error is detected on PCI during a CPU read from PCI. During CPU to PCI write cycles, when the SERR# on Transmitted PCI Data Parity Error bit is set to 1, the PCMC asserts SERR# in response to sampling PERR# active. When the SERR# on Received Target Abort bit is set to 1, the PCMC asserts SERR# when the PCMC receives a target abort on a PCMC initiated PCI cycle. If the Parity Error Enable bit in the PCI Command Register is reset to 0, SERR# is disabled and is never asserted by the PCMC.

2.5 LBX Interface

Signal	Type	Description
HIG[4:0]	out	**HOST INTERFACE GROUP:** HIG[4:0] are outputs of the PCMC used to control the LBX HA (Host Address) and HD (Host Data) buses. Commands driven on HIG[4:0] cause the host data and/or address lines to be either driven or latched by the LBXs. See the 82433LX (LBX) Local Bus Accelerator Data Sheet for a listing of the HIG[4:0] commands.
MIG[2:0]	out	**MEMORY INTERFACE GROUP:** MIG[2:0] are outputs of the PCMC and control the LBX MD (Memory Data) bus. Commands driven on the MIG[2:0] lines cause the memory data lines to be either driven or latched by the LBXs. See the 82433LX (LBX) Local Bus Accelerator Data Sheet for a listing of the MIG[2:0] commands.
MDLE	out	**MEMORY DATA LATCH ENABLE:** During CPU reads from main memory, MDLE is used to control the latching of memory read data on the CPU data bus. MDLE is negated as CAS[7:0]# are negated to close the latch between the memory data bus and the host data bus. During CPU reads from main memory, the PCMC closes the memory data to host data latch in the LBXs as BRDY# is asserted and opens the latch after the CPU has sampled the data.
PIG[3:0]	out	**PCI INTERFACE GROUP:** PIG[3:0] are outputs of the PCMC used to control the LBX AD (PCI Address/Data) bus. Commands driven on the PIG[3:0] lines cause the AD lines to be either driven or latched. See the 82433LX (LBX) Local Bus Accelerator Data Sheet for a listing of the PIG[3:0] commands.
DRVPCI	out	**DRIVE PCI:** DRVPCI acts as an output enable for the LBX AD lines. When sampled asserted, the LBXs begin driving the PCI AD lines. When negated, the AD lines on the LBXs are tri-stated. The LBX AD lines are tri-stated asynchronously from the falling edge of DRVPCI.
EOL	in	**END OF LINE:** EOL is asserted by the low order LBX when a PCI master read or write transaction is about to overrun a cache line boundary. EOL has an internal pull-up resistor inside the PCMC. The low order LBX EOL signal connects to this PCMC input. The high order LBX EOL signal is connected to ground through an external pull-down resistor.
PPOUT[1:0]	in	**PCI PARITY OUT:** These signals reflect the parity of the 32 AD lines driven from or latched in the LBXs, depending on the command driven on PIG[3:0]. The PPOUT0 pin has a weak internal pull-down resistor. The PPOUT1 pin has a weak internal pull-up resistor.

2.6 Reset And Clock

Signal	Type	Description
HCLKOSC	in	**HOST CLOCK OSCILLATOR:** The HCLKOSC input is driven externally by a crystal oscillator. The PCMC generates six copies of HCLK from HCLKOSC (HCLKA–HCLKF). During power-up, HCLKOSC must stabilize for 1 ms before PWROK is asserted. If an external clock driver is used to clock the CPU, PCMC, LBXs and second level cache SRAMs instead of the HCLKA–HCLKF outputs, HCLKOSC must be tied either high or low.
HCLKA–HCLKF	out	**HOST CLOCK OUTPUTS:** HCLKA–HCLKF are six low skew copies of the host clock. These outputs eliminate the need for an external low skew clock driver.

Signal	Type	Description
HCLKIN	in	**HOST CLOCK INPUT:** All timing on the host, DRAM and second level cache interfaces is based on HCLKIN. If an external clock driver is used to clock the CPU, PCMC, LBXs and second level cache SRAMs, the externally generated clock must be connected to HCLKIN. During power-up HCLKIN must stabilize for 1 ms before PWROK is asserted.
CPURST	out	**CPU HARD RESET:** The CPURST pin is asserted in response to one of two conditions. **Powerup** 82434LX: During powerup the 82434LX asserts CPURST when PWROK is negated. When PWROK is asserted, the 82434LX first ensures that it has been initialized before negating CPURST. 82434NX: During powerup, the 82434NX PCMC negates CPURST while PWROK is negated. When PWROK is asserted, the 82434NX asserts CPURST for 2 ms. **Software** CPURST is also asserted when the System Hard Reset Enable bit in the Turbo-Reset Control Register (I/O address 0CF9h) is set to 1 and the Reset CPU bit toggles from 0 to 1 (82434LX and 82434NX). CPURST is driven synchronously to the rising edge of HCLKIN.
INIT	out	**INITIALIZATION:** INIT is asserted in response to any one of two conditions. When the System Hard Reset Enable bit in the Turbo-Reset Control Register is reset to 0 and the Reset CPU bit toggles from 0 to 1, the PCMC initiates a soft reset by asserting INIT. The PCMC also initiates a soft reset by asserting INIT in response to a shutdown special cycle. In both cases, INIT is asserted for a minimum of 2 Host clocks.
PWROK	in	**POWER OK:** When asserted, PWROK is an indication to the PCMC that power and HCLKIN have stabilized for at least 1 ms. PWROK can be driven asynchronously. 82434LX: When PWROK is negated, the 82434LX asserts both CPURST and PCIRST#. When PWROK is driven high, the 82434LX ensures that it is initialized before negating CPURST and PCIRST#. 82434NX: When PWROK is negated, the 82434NX negates CPURST and asserts PCIRST#. When PWROK is asserted, the 82434NX asserts CPURST for 2 ms. PCIRST# is negated 1 ms after PWROK is asserted.
PCLKOUT	out	**PCI CLOCK OUTPUT:** PCLKOUT is internally generated by a Phase Locked Loop (PLL) that divides the frequency of HCLKIN by 2. This output must be buffered externally to generate multiple copies of the PCI Clock. One of the copies must be connected to the PCLKIN pin.

Signal	Type	Description
PCLKIN	in	**PCI CLOCK INPUT:** An internal PLL locks PCLKIN in phase with HCLKIN. All timing on the PCMC PCI interface is referenced to the PCLKIN input. All output signals on the PCI interface are driven from PCLKIN rising edges and all input signals on the PCI interface are sampled on PCLKIN rising edges.
PCIRST#	out	**PCI RESET:** PCIRST# is asserted to initiate hard reset on PCI. PCIRST# is asserted in response to one of two conditions. **Power-up** During power-up the PCMC asserts PCIRST# when PWROK is negated. 82434LX: When PWROK is asserted the PCMC will first ensure that it has been initialized before negating PCIRST#. 82434NX: When PWROK is negated, the 82434NX asserts PCIRST#. The 82434NX then negates PCIRST# 1 ms after PWROK is asserted. **Software** PCIRST# is also asserted when the System Hard Reset Enable bit in the Turbo/Reset Control Register is set to 1 and the Reset CPU bit toggles from 0 to 1 (82434LX and 82434NX). PCIRST# is driven asynchronously.
TESTEN	in	**TEST ENABLE:** TESTEN must be tied low for normal system operation.

3.0 REGISTER DESCRIPTION

The 82434LX/82434NX PCMC contains two sets of software accessible registers. These registers are accessed via the Host CPU I/O address space. The PCMC also contains a set of configuration registers that reside in PCI configuration space and are used to specify PCI configuration, DRAM configuration, cache configuration, operating parameters and optional system features (see Section 3.2, PCI Configuration Space Mapped Registers). The PCMC internal registers (both I/O Mapped and Configuration registers) are only accessible by the Host CPU and cannot be accessed by PCI masters. The registers can be accessed as Byte, Word (16-bit), or Dword (32-bit) quantities. All multi-byte numeric fields use "little-endian" ordering (i.e., lower addresses contain the least significant parts of the field).

Some of the PCMC registers described in this section contain reserved bits. These bits are labeled "R". Software must deal correctly with fields that are reserved. On reads, software must use appropriate masks to extract the defined bits and not rely on reserved bits being any particular value. On writes, software must ensure that the values of reserved bit positions are preserved. That is, the values of reserved bit positions must first be read, merged with the new values for other bit positions and then written back.

In addition to reserved bits within a register, the PCMC contains address locations in the PCI configuration space that are marked "Reserved" (Table 1). The PCMC responds to accesses to these address locations by completing the Host cycle. When a reserved register location is read, 0000h is returned. Writes to reserved registers have no affect on the PCMC.

Upon receiving a hard reset via the PWROK signal, the PCMC sets its internal configuration registers to predetermined **default** states. The default state represents the minimum functionality feature set required to successfully bring up the system. Hence, it does not represent the optimal system configuration. It is the responsibility of the system initialization software (usually BIOS) to properly determine the DRAM configurations, cache configuration, operating parameters and optional system features that are applicable, and to program the PCMC registers accordingly.

The following nomenclature is used for access attributes.

RO **Read Only.** If a register is read only, writes to this register have no effect.

R/W **Read/Write.** A register with this attribute can be read and written.

R/WC **Read/Write Clear.** A register bit with this attribute can be read and written. However, a write of a 1 clears (sets to 0) the corresponding bit and a write of a 0 has no effect.

3.1 I/O Mapped Registers

The 82434LX PCMC contains three registers that reside in the CPU I/O address space—the Configuration Space Enable (CSE) Register, the Turbo-Reset Control (TRC) Register and the Forward (FORW) Register. These registers can not reside in PCI configuration space because of the special functions they perform. The CSE Register enables/disables the configuration space and, hence, can not reside in that space. The TRC Register enables/disables deturbo mode which effectively slows the processor to accommodate software programs that rely on the slow speed of PC/XT systems to time certain events. The FORW Register determines which of the possible hierarchical PCI Buses a cycle is directed. The 82434LX uses mechanism #2 for accessing PCI configuration space.

The 82434NX PCMC contains five registers that reside in the CPU I/O address spacethe Configuration Address (CONFADD) Register, the Configuration Space Enable (CSE) Register, the Turbo-Reset Control (TRC) Register, the Forward (FORW) Register, and the PCI Mechanism Control (PMC) Register. The CSE, TRC, and FORW Registers are the same for both the 82434LX and 82434NX PCMCs. The 82434NX can use either Configuration Access Mechanism #1 or #2 for accessing PCI configuration space. When Configuration Access Mechanism #1 is used (See Section 3.2, PCI Configuration Space Mapped Registers), The CONFADD Register enables/disables the configuration space and determines what portion of configuration space is visible through the Configuration Data (CONFDATA) window. The CSE and FORW Registers are used for Configuration Access Mechanism #2. The PCI Mechanism Control (PMC) Register selects whether Configuration Access Mechanism 1 or 2 is used (see the Rev 2.0 PCI Local Bus Specification).

3.1.1 CONFADD—CONFIGURATION ADDRESS REGISTER

I/O Address: 0CF8h Accessed as a Dword
Default Value: 00000000h
Access: Read/Write
Size: 32 bits

CONFADD is a 32-bit register used in Configuration Access Mechanism #1. It is accessed only when referenced as a Dword and PCAMS in the PMC Register is set to 1. Byte or Word references "pass through" the CONFADD Register to the I/O locations "behind" it. For example a byte access to 0CF8h will access the CSE Register, while a word access to CF8h will access both the CSE and TRC Registers. The CONFADD Register contains the Bus Number, Device Number, Function Number, and Register Number where the CONFDATA window is located.

Bit	Description
31	**CONFIGURATION ENABLE (CONE)—R/W:** When CONE = 1, accesses to PCI configuration space are enabled, if the PCAMS bit of the PMC register is also 1. When CONE = 0, accesses to PCI configuration space are disabled, if the PCAMS bit is 1. If the PCAMS bit is 0, this bit has no effect.
30:24	**RESERVED**
23:16	**BUS NUMBER (BUSNUM)—R/W:** When the BUSNUM is programmed to 00h, the target of the Configuration Cycle is either the PCMC or the PCI Local Bus that is directly connected to the PCMC. PCI Access Mechanism #1 can generate either type 0 or type 1 configuration cycles on PCI. A type 0 Configuration Cycle is generated on PCI if the Bus Number is programmed to 00h and the PCMC is not the target. If the Bus Number is non-zero a type 1 configuration cycle is generated on PCI with the Bus Number mapped to AD[23:16] during the address phase.
15:11	**DEVICE NUMBER (DEVNUM)—R/W:** This field selects one agent on the PCI Bus selected by the Bus Number. During a Type 1 Configuration cycle this field is mapped to AD[15:11]. During a Type 0 Configuration Cycle this field is decoded and one of AD[31:17] is driven to a 1. The PCMC is always Device Number 0.
10:8	**FUNCTION NUMBER (FUNCNUM)—R/W:** This field is mapped to AD[10:8] during PCI configuration cycles. This allows the configuration registers of a particular function in a multi-function device to be accessed.
7:2	**REGISTER NUMBER (REGNUM)—R/W:** This field selects one register within a particular Bus, Device, and Function as specified by the other fields in the Configuration Address Register. REGNUM is mapped to AD[7:2] during PCI configuration cycles.
1:0	**RESERVED**

3.1.2 CSE—CONFIGURATION SPACE ENABLE REGISTER

I/O Address: 0CF8h
Default Value: 00h
Attribute: Read/Write
Size: 8 bits

The CSE Register enables/disables configuration space access and provides access to specific functions within a PCI agent. The register is located in the CPU I/O address space. The PCMC, as a Host/PCI Bridge, supports multi-function devices on the PCI Bus. The function number permits individual configuration spaces for up to eight functions within an agent. The register is located in the CPU I/O address space.

Bit	Description
7:4	**KEY FIELD (KEY)—R/W:** This field is used only when the PCI Mechanism Control Register (PMC) indicates Configuration Access Mechanism 2 is to be used. When the key field is programmed to 0h, the PCI configuration space is disabled. When the key field is programmed to a non-zero value, all CPU accesses to CnXXh (where n is a non zero value) are forwarded to PCI as configuration space accesses. Additionally, when the key field is programmed to a non-zero value, all CPU accesses to C0XXh are intercepted by the PCMC and directed to a PCMC internal register.
3:1	**FUNCTION NUMBER (FN)—R/W:** For multi-function devices, this field selects a particular function within a PCI device. During a configuration cycle, bits[3:1] become part of the PCI Bus address and correspond to AD[10:8].
0	**RESERVED**

3.1.3 TRC—TURBO-RESET CONTROL REGISTER

I/O Address: 0CF9h
Default Value: 00h
Attribute: Read/Write
Size: 8 bits

The TRC Register is an 8-bit read/write register that selects turbo/deturbo mode of the CPU, initiates PCI Bus and CPU reset cycles, and initiates the CPU Built In Self Test (BIST). TRC is located in CPU I/O address space.

Bit	Description
7:3	**RESERVED**
2	**RESET CPU (RCPU)—R/W:** RCPU is used to initiate a hard reset or soft reset to the CPU. During a hard reset, the PCMC asserts CPURST and PCIRST#. The PCMC initiates a hard reset when this register is programmed for a hard reset or when the PWROK signal is asserted. During a soft reset, the PCMC asserts INIT. The PCMC initiates a soft reset when this register is programmed for a soft reset and in response to a shutdown special cycle. Note that a hard reset initializes the entire system and invalidates the CPU cache. A soft reset initializes only the CPU. The contents of the CPU cache are unaffected. This bit is used in conjunction with bit 1 of this register. Bit 1 must be set up prior to writing a 1 to this register. Thus, two write operations are required to initiate a reset using this bit. The first write operation programs bit 1 to the appropriate state while setting this bit to 0. The second write operation keeps bit 1 at the programmed state (1 or 0) while setting this bit to a 1. When RCPU transitions from a 0 to a 1, a hard reset is initiated if bit 1 = 1 and a soft reset is initiated if bit 1 = 0.
1	**SYSTEM HARD RESET ENABLE (SHRE)—R/W:** This bit is used in conjunction with bit 2 of this register to initiate either a hard or soft reset. When SHRE = 1, the PCMC initiates a hard reset to the CPU when bit 2 transitions from 0 to 1. When SHRE = 0, the PCMC initiates a soft reset when bit 2 transitions from 0 to 1.
0	**DETURBO MODE (DM)—R/W:** This bit enables and disables deturbo mode. When DM = 1, the PCMC is in the deturbo mode. In this mode, the PCMC periodically asserts the AHOLD signal to slow down the effective speed of the CPU. The AHOLD duty cycle is programmable through the Deturbo Frequency Control (DFC) Register. When DM = 0, the deturbo mode is disabled. Deturbo mode can be used to maintain backward compatibility with older software packages that rely on the operating speed of older processors. For accurate speed emulation, caching should be disabled. If caching is disabled during runtime, the following steps should be performed to make sure that modified lines have been flushed from the cache to main memory before entering deturbo mode. Disable the primary cache via the PCE bit in the HCS Register. This prevents the KEN# signal from being asserted, which prevents any further first and second level cache line fills. At this point, software executes the WBINVD instruction to flush the caches, and then sets DM to 1. When exiting the deturbo mode, the system software must first set DM to 0, then enable first and second level caching by writing to the HCS Register.

3.1.4 FORW—FORWARD REGISTER

I/O Address: 0CFAh
Default Value: 00h
Attribute: Read/Write
Size: 8 Bits

This 8-bit register specifies which PCI Bus configuration space is enabled in a multiple PCI Bus configuration. The default value for the FORW Register enables the configuration space of the PCI Bus connected to the PCMC.

Bit	Description
7:0	**FORWARD BUS NUMBER—R/W:** When this register value is 00h, the configuration space of the PCI Bus connected to the PCMC is enabled and the PCMC initiates a type 0 configuration cycle. If the value of this register is not 00h, the PCMC initiates a type 1 configuration cycle to forward the cycle (via one or more PCI/PCI Bridges) to the PCI Bus specified by the contents of this register. For non-zero values, bits[7:0] are mapped to AD[23:16], respectively.

3.1.5 PMC—PCI MECHANISM CONTROL REGISTER

I/O Address: 0CFBh
Default Value: 00h
Access: Read/Write
Size: 8 bits

The PMC Register selects whether PCI Configuration Access Mechanism 1 or 2 is to be used. The register is located in the CPU I/O address space.

Bit	Description
7:1	**RESERVED**
0	**PCI CONFIGURATION ACCESS MECHANISM SELECT (PCAMS)—R/W:** When PCAMS = 0, the PCMC uses to PCI Configuration Access Mechanism #2. When PCAMS = 1, the PCMC uses to PCI Configuration Access Mechanism #1. The CONFADD and CONFDATA Registers are only accessible when PCAMS = 1.

3.1.6 CONFDATA—CONFIGURATION DATA REGISTER

I/O Address: 0CFCh
Default Value: 00h
Access: Read/Write
Size: 32 bits

CONFDATA is a 32 bit read/write window into configuration space. The portion of configuration space that is referenced by CONFDATA is determined by the contents of CONFADD.

Bit	Description
31:0	**CONFIGURATION DATA WINDOW (CDW)—R/W:** When using Configuration Access Mechanism #1 if bit 31 of CONFADD is 1 any I/O reference that falls in the CONFDATA I/O space will be mapped to configuration space using the contents of CONFADD.

3.2 PCI Configuration Space Mapped Registers

The PCI Bus defines a slot based "configuration space" that allows each device to contain up to 256 8-bit configuration registers. The PCI specification defines two bus cycles to access the PCI configuration space—**Configuration Read** and **Configuration Write**. While memory and I/O spaces are supported by the Pentium processor, configuration space is not supported. For PCI configuration space access, the PCMC translates the Pentium processor I/O cycles into PCI configuration cycles. Table 1 shows the PCMC configuration space.

Table 1. PCMC Configuration Space

Address Offset	Register Symbol	Register Name	Access
00–01h	VID	Vendor Identification	RO
02–03h	DID	Device Identification	RO
04–05h	PCICMD	Command Register	R/W
06–07h	PCISTS	Status Register	RO, R/WC
08h	RID	Revision Identification	RO
09h	RLPI	Register-Level Programming Interface	RO
0Ah	SCCD	Sub-Class Code	RO
0Bh	BCCD	Base Class Code	RO
0Ch	—	Reserved	—
0Dh	MLT	Master Latency Timer	R/W
0Eh	—	Reserved	—
0Fh	BIST	BIST Register	RO
10–4Fh	—	Reserved	—
50h	HCS	Host CPU Selection	R/W
51h	DFC	Deturbo Frequency Control	R/W
52h	SCC	Secondary Cache Control	R/W
53h	HBC	Host Read/Write Buffer Control	R/W
54h	PBC	PCI Read/Write Buffer Control	R/W
55h	—	Reserved	—
56h	—	Reserved	—
57h	DRAMC	DRAM Control	R/W
58h	DRAMT	DRAM Timing	R/W
59–5Fh	PAM[6:0]	Programmable Attribute Map (7 Registers)	R/W
60–65h	DRB[5:0]	DRAM Row Boundary (6 Registers)	R/W
66–67h	DRB[7:6]	DRAM Row Boundary (2 Registers)	R/W
68–6Bh	DRBE	DRAM Row Boundary Extension	R/W
6C–6Fh	—	Reserved	—
70h	ERRCMD	Error Command	R/W

Table 1. PCMC Configuration Space (Continued)

Address Offset	Register Symbol	Register Name	Access
71h	ERRSTS	Error Status	R/WC
72h	SMRS	SMRAM Space Control	R/W
73–77h	—	Reserved	—
78–79h	MSG	Memory Space Gap	R/W
7A–7B	—	Reserved	—
7C–7Fh	FBR	Frame Buffer Range	R/W
80–FFh	—	Reserved	—

NOTE:
Shaded rows indicate register differences between the 82434LX and 82434NX devices. For non-shaded rows, the registers are the same for the two devices.

3.2.1 CONFIGURATION SPACE ACCESS MECHANISM

The 82434LX supports Configuration Space Access Mechanism #2 and the 82434NX supports both configuration space access mechanisms #1 and #2. The mechanism is selected via the PCAMS bit in the PMC Register. The bus cycles used to access PCMC internal configuration registers are described in Section 7.0, PCI Interface.

3.2.1.1 Access Mechanism #1:

For configuration access mechanism #1, the 82434NX PCMC uses the CONFADD and CONFDATA Registers. Note that while the CONFADD and PMC Register address spaces overlap, the CONFADD Register is referenced only by a Dword read or write to CF8h. This allows the PMC Register to be accessed by a byte write to CFBh, even when using configuration access mechanism #1.

To reference a configuration register with access mechanism #1, a Dword I/O write loads the CONFADD Register with a 32-bit value that specifies the PCI Bus, the device on that bus, the function within the device, and a specific configuration register of the device function being accessed (Figure 4). Bit 31 of the CONFADD Register must be 1 to enable a configuration cycle. CONFDATA then becomes a four byte window of configuration space specified by the contents of the CONFADD Register. A read or write to CONFDATA results in the PCMC translating CONFADD into a PCI configuration cycle.

Type 0 Access

If the BUSNUM field is 0, a Type 0 configuration cycle is performed on the PCI. Bus CONFADD[10:2] are mapped directly to AD[10:2]. The DEVNUM field is decoded onto AD[31:17] and AD[15:11] (for accesses to device 1, AD17 is asserted; for accesses to device #2, AD18 is asserted; etc.). The PCMC is Device #0 and does not pass its configuration cycles to the PCI Bus. Thus, AD16 is never asserted. For accesses to device 15, AD31 is asserted, etc. This mapping allows the same Device Number to activate the same AD line in either configuration access mechanism. All other AD lines are 0.

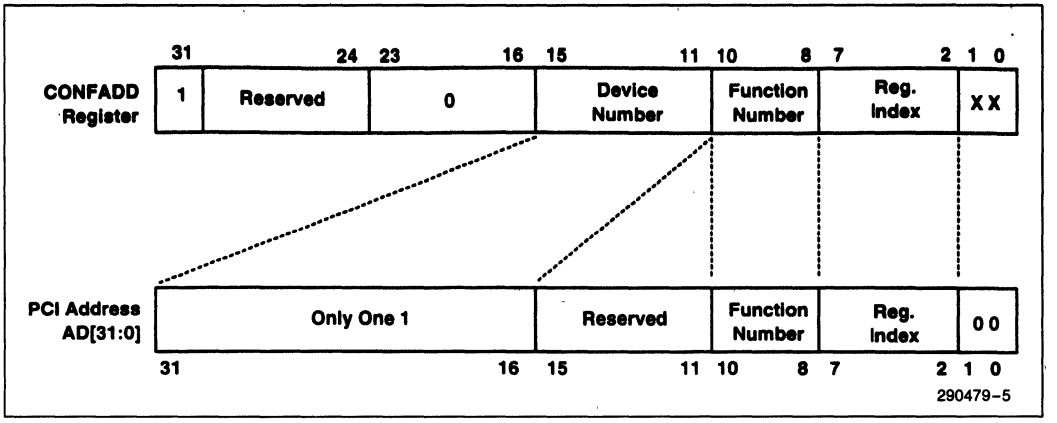


Figure 4. Mechanism #1 Type 0 Configuration Address to PCI Address Mapping

2

Type 1 Access

If the BUSNUM field of the CONFADD Register is non-zero, a Type 1 configuration cycle is performed on the PCI Bus. CONFADD[23:2] are mapped directly to AD[23:2] (Figure 5). AD[1:0] are driven to 01 to indicate a Type 1 Configuration cycle. All other lines are driven to 0.

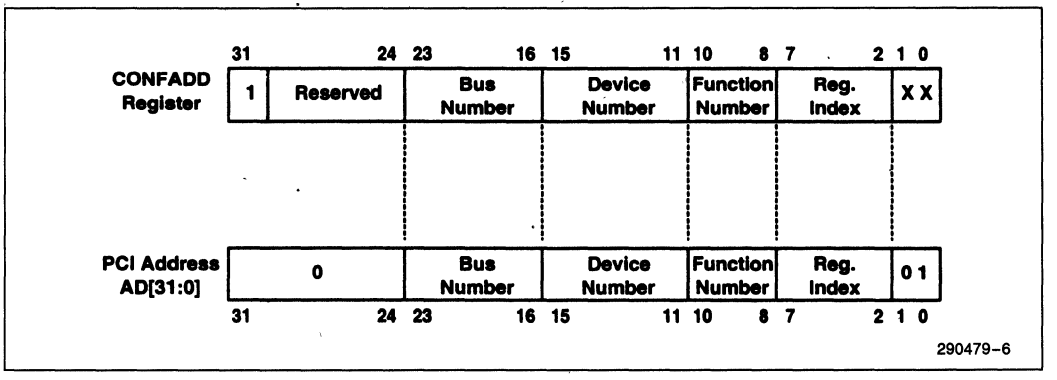


Figure 5. Mechanism #1 Type 1 Configuration Address to PCI Address Mapping

3.2.1.2 Access Mechanism #2

The 82434LX/82434NX PCMC uses the CSE and Forward Registers for configuration access mechanism #2. When PCI configuration space is enabled via the CSE Register, the PCMC maps PCI configuration space into 4-KBytes of CPU I/O space. Each PCI device has its own 256-Byte configuration space. When configuration space is enabled, CPU accesses to I/O locations CXXXh are translated into configuration space accesses. In this mode, the PCMC translates all I/O cycles in the C100h–CFFFh range into configuration cycles on the PCI Bus. I/O accesses within the C000h–C0FFh range are intercepted by the PCMC and are directed to the PCMC internal configuration registers. These cycles are not forwarded to the PCI Bus.

When configuration space access is disabled, CPU accesses to I/O locations CXXXh are forwarded to the PCI Bus I/O space. CPU cycles to I/O locations other than CXXXh are unaffected by whether the configuration mode is enabled or disabled. These cycles are always treated as ordinary I/O cycles by the PCMC.

Type 0 Access

If the Forward Register contains 00h a Type 0 configuration access is generated on the PCI Bus (Figure 6). For type 0 configuration cycles, AD[1:0] = 00. Host CPU address bits A[7:2] are not translated and become AD[7:2] on the PCI Bus. AD[7:2] select one of the 256 8-bit I/O locations in the PCI configuration space. The FUNCTION NUMBER field from the CSE Register (CSE[3:1]) is driven on AD[10:8]. Host CPU address bits A[11:8] are mapped to an IDSEL input for each of the 16 possible PCI devices. The IDSEL input for each PCI device must be hard-wired to one of the AD[31:16] signals on the PCI Bus. AD16 is reserved for the PCMC. When CPU address A[11:8] = Fh, PCI address bits A31 = 1 and A[30:16] = 00h. Other devices on the PCI Bus should not use AD16. Note that when A[11:8] = 0h, an access to the PCMC internal registers occurs and the cycle is not forwarded to the PCI Bus.

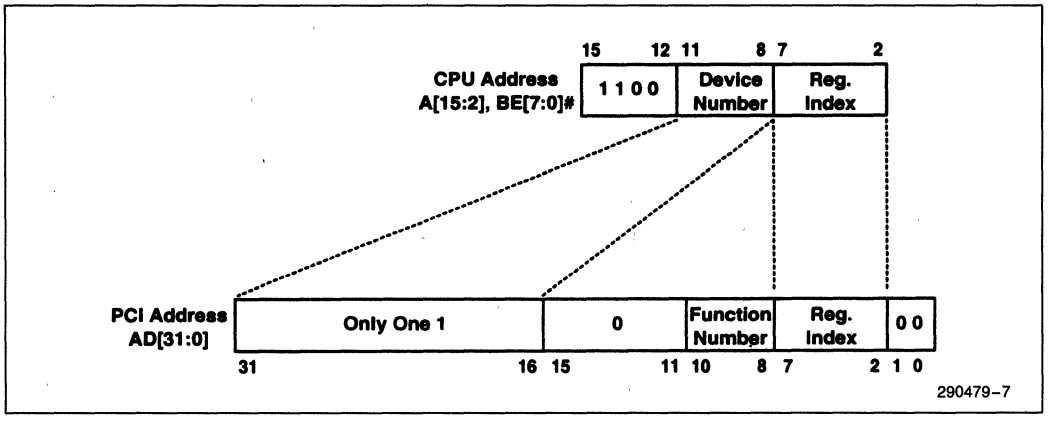


Figure 6. Mechanism #2 Type 0 Host-to-PCI Address Mapping

Type 1 Access

If the Forward Register is non-zero a Type 1 configuration access is generated on PCI. For type 1 configuration cycles, AD[1:0] = 01. AD[10:2] are generated the same as for the type 0 configuration cycle. Host CPU address bits A[11:8] contain the specific device number and are mapped to AD[14:11]. AD[23:16] contain the Bus Number of the PCI Bus that is to be accessed and corresponds to the Forward Address Register bits [7:0].

During a Type 1 configuration access AD[1:0] = 01 (Figure 7). The Register Index and Function Number are mapped to the AD lines the same way in Type 1 configuration access as in a Type 0 configuration access. CPU address bits A[11:8] are mapped directly to PCI lines AD[14:11] as the Device Number. The contents of the Forward Register are mapped to AD[23:16] to form the Bus Number.

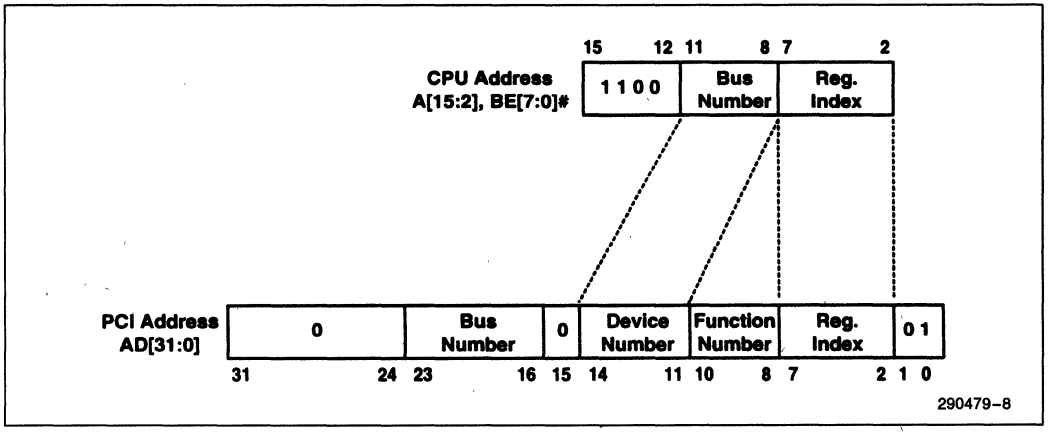


Figure 7. Mechanism #2 Type 1 Host-to-PCI Address Mapping

3.2.2 VID—VENDOR IDENTIFICATION REGISTER

Address Offset: 00–01h
Default Value: 8086h
Attribute: Read Only
Size: 16 bits

The VID Register contains the vendor identification number. This 16-bit register combined with the Device Identification Register uniquely identify any PCI device. Writes to this register have no effect.

Bits	Description
15:0	**VENDOR IDENTIFICATION NUMBER:** This is a 16-bit value assigned to Intel.

3.2.3 DID—DEVICE IDENTIFICATION REGISTER

Address Offset: 02–03h
Default Value: 04A3h
Attribute: Read Only
Size: 16 bits

This 16-bit register combined with the Vendor Identification Register uniquely identifies any PCI device. Writes to this register have no effect.

Bits	Description
15:0	**DEVICE IDENTIFICATION NUMBER:** This is a 16 bit value assigned to the PCMC.

3.2.4 PCICMD—PCI COMMAND REGISTER

Address Offset: 04–05h
Default: 06h
Attribute: Read/Write
Size: 16 bits

This 16-bit register provides basic control over the PCMC's ability to respond to PCI cycles. The PCICMD Register enables and disables the SERR# signal, the parity error signal (PERR#), PCMC response to PCI special cycles, and enables and disables PCI master accesses to main memory.

Bits	Description
15:9	**RESERVED**
8	**SERR# ENABLE (SERRE):** SERRE enables/disables the SERR# signal. When SERRE = 1 and PERRE = 1, SERR# is asserted if the PCMC detects a PCI Bus address/data parity error, or main memory (DRAM) or cache parity error, and the corresponding errors are enabled in the Error-Command Register. When SERRE = 1 and bit 7 in the Error Command Register is set to 1, the PCMC asserts SERR# when it detects a target abort on a PCMC-initiated PCI cycle. When SERRE = 0, SERR# is never asserted.
7	**RESERVED**
6	**PARITY ERROR ENABLE (PERRE):** PERRE controls the PCMC's response to PCI parity errors. This bit is a master enable for bit 3 of the ERRCMD Register. PERRE works in conjunction with the SERRE bit to enable SERR# assertion when the PCMC detects a PCI bus parity error, or a main memory or cache parity error.
5:3	**RESERVED**
2	**BUS MASTER ENABLE (BME):** The PCMC does not support disabling of its bus master capability on the PCI Bus. This bit is always set to 1, permitting the PCMC to function as a PCI Bus master. Writes to this bit position have no affect.
1	**MEMORY ACCESS ENABLE (MAE):** This bit enables/disables PCI master access to main memory (DRAM). When MAE = 1, the PCMC permits PCI masters to access main memory if the MEMCS# signal is asserted. When MAE = 0, the PCMC does not respond to PCI master main memory accesses (MEMCS# asserted).
0	**I/O ACCESS ENABLE (IOAE):** The PCMC does not respond to PCI I/O cycles, hence this command is not supported. PCI master access to I/O space on the Host Bus is always disabled.

3.2.5 PCISTS—PCI STATUS REGISTER

Address Offset: 06–07h
Default Value: 40h
Attribute: Read Only, Read/Write Clear
Size: 16 bits

PCISTS is a 16-bit status register that reports the occurrence of a PCI master abort, PCI target abort, and DRAM or cache parity error. PCISTS also indicates the DEVSEL# timing that has been set by the PCMC hardware. Bits[15:12] are read/write clear and bits[10:9] are read only.

Bits	Attribute	Description
15		**RESERVED**
14	R/WC	**SIGNALED SYSTEM ERROR (SSE):** When the PCMC asserts the SERR# signal, this bit is also set to 1. Software sets SSE to 0 by writing a 1 to this bit.
13	R/WC	**RECEIVED MASTER ABORT STATUS (RMAS):** When the PCMC terminates a Host-to-PCI transaction (PCMC is a PCI master), which is not a special cycle, with a master abort, this bit is set to 1. Software resets this bit to 0 by writing a 1 to it.
12	R/WC	**RECEIVED TARGET ABORT STATUS (RTAS):** When a PCMC-initiated PCI transaction is terminated with a target abort, RTAS is set to 1. The PCMC also asserts SERR# if the SERR# Target Abort bit in the ERRCMD Register is 1. Software resets RTAS to 0 by writing a 1 to it.
11		**RESERVED**
10:9	RO	**DEVSEL# TIMING (DEVT):** This 2-bit field indicates the timing of the DEVSEL# signal when the PCMC responds as a target. The PCI specification defines three allowable timings for assertion of DEVSEL#: 00 = fast, 01 = medium, and 10 = slow (DEVT = 11 is reserved). DEVT indicates the slowest time that a device asserts DEVSEL# for any bus command, except configuration read and write cycles. Note that these two bits determine the slowest time that the PCMC asserts DEVSEL#. However, the PCMC can also assert DEVSEL# in medium time. The PCMC asserts DEVSEL# in response to sampling MEMCS# asserted. The PCMC samples MEMCS# one and two clocks after FRAME# is asserted. If MEMCS# is asserted one PCI clock after FRAME# is asserted, then the PCMC responds with DEVSEL# in slow time.
8	R/WC	**DATA PARITY DETECTED (DPD):** This bit is set to 1 when all of the following conditions are met: 1). The PCMC asserted PERR# or sampled PERR# asserted. 2). The PCMC was the bus master for the operation in which the error occurred. 3). The PERRE bit in the Command Register is set to 1. Software resets DPD to 0 by writing a 1 to it.
7:0		**RESERVED**

3.2.6 RID—REVISION IDENTIFICATION REGISTER

Address Offset: 08h
Default Value: 03h for A-3 Stepping (82434LX)
01h for A-1 Stepping (82434LX)
10h for A-0 Stepping (82434NX)
11h for A-1 Stepping (82434NX)
Attribute: Read Only
Size: 8 bits

This register contains the revision number of the PCMC. These bits are read only and writes to this register have no effect. For the A-2 Stepping of the 82434LX, this value is 03h.

For the A-1 Stepping of the 82434NX, this value is 11h.

Bits	Description
7:0	**REVISION IDENTIFICATION NUMBER:** This is an 8-bit value that indicates the revision identification number for the PCMC.

3.2.7 RLPI—REGISTER-LEVEL PROGRAMMING INTERFACE REGISTER

Address Offset: 09h
Default Value: 00h
Attribute: Read Only
Size: 8 bits

This register defines the PCMC as having no defined register-level programming interface.

Bits	Description
7:0	**REGISTER-LEVEL PROGRAMMING INTERFACE (RLPI):** The value of 00h defines the PCMC as having no defined register-level programming interface.

3.2.8 SUBC—SUB-CLASS CODE REGISTER

Address Offset: 0Ah
Default Value: 00h
Attribute: Read Only
Size: 8 bits

This register defines the PCMC as a host bridge.

Bits	Description
7:0	**SUB-CLASS CODE (SCCD):** The value of this register is 00h defining the PCMC as host bridge.

3.2.9 BASEC—BASE CLASS CODE REGISTER

Address Offset: 0Bh
Default Value: 06h
Attribute: Read Only
Size: 8 bits

This register defines the PCMC as a bridge device.

Bits	Description
7:0	**BASE CLASS CODE (BCCD):** The value in this register is 06h defining the PCMC as bridge device.

3.2.10 MLT—MASTER LATENCY TIMER REGISTER

Address Offset: 0Dh
Default Value: 20h
Attribute: Read/Write
Size: 8 bits

MLT is an 8-bit register that controls the amount of time the PCMC, as a bus master, can burst data on the PCI Bus. MLT is used when the PCMC becomes the PCI Bus master and is cleared and suspended when the PCMC is not asserting FRAME#. When the PCMC asserts FRAME#, the counter is enabled and begins counting. If the PCMC finishes its transaction before the count expires, the MLT count is ignored. If the count expires before the transaction completes, the PCMC initiates a transaction termination as soon as its GNT# is removed. The number of clocks programmed in the MLT represents the guaranteed time slice (measured in PCI clocks) allotted to the PCMC, after which it must surrender the bus as soon as its GNT# is taken away. The number of clocks in the Master Latency Timer is the count value field multiplied by 16.

Bits	Description
7:4	**MASTER LATENCY TIMER COUNT VALUE:** If GNT# is negated after the burst cycle is initiated, the PCMC limits the duration of the burst cycle to the number of PCI Bus clocks specified by this field multiplied by 16.
3:0	**RESERVED**

3.2.11 BIST—BIST REGISTER

Address Offset: 0Fh
Default Value: 0h
Attribute: Read Only
Size: 8 bits

The BIST function is not supported by the PCMC. Writes to this register have no affect.

Bits	Attribute	Description
7	RO	**BIST SUPPORTED:** This read only bit is always set to 0, disabling the BIST function. Writes to this bit position have no affect.
6	RW	**START BIST:** This function is not supported and writes have no affect.
5:4		**RESERVED**
3:0	RO	**COMPLETION CODE:** This read only field always returns 0 when read and writes have no affect.

3.2.12 HCS—HOST CPU SELECTION REGISTER

Address Offset: 50h
Default Value: 82h (82434LX)
A2h (82434NX)
Access: Read/Write, Read Only
Size: 8 bits

The HCS Register is used to specify the Host CPU type and speed. This 8-bit register is also used to enable and disable the first level cache.

Bits	Access	Description
7:5	RO	**HOST CPU TYPE (HCT):** This field defines the Host CPU type. **82434LX** These bits are hardwired to 100 which selects the Pentium processor. All other combinations are reserved. **82434NX** In the 82434NX, these bits are reserved. Reads and writes to these bits have no effect.
4:3		**RESERVED**
2	R/W	**FIRST LEVEL CACHE ENABLE (FLCE):** FLCE enables and disables the first level cache. When FLCE = 1, the PCMC responds to CPU cycles with KEN# asserted for cacheable memory cycles. When FLCE = 0, KEN# is always negated. This prevents new cache line fills to either the first level or second level caches.
1:0	R/W	**HOST OPERATING FREQUENCY (HOF):** The DRAM refresh rate is adjusted according to the frequency selected by this field. For the 82434LX, only bit 0 is used and bit 1 is reserved. **82434LX** Bit 1 is reserved. If bit 0 is 1, the 82434LX supports a 66 MHz CPU. If bit 0 is 0, the 82434LX supports a 60 MHz CPU. **82434NX** These bits select the Host CPU frequency supported as follows: **Bits[1:0] Host CPU Frequency** 00 Reserved 01 50 MHz 10 60 MHz 11 66 MHz

2

3.2.13 DFC—DETURBO FREQUENCY CONTROL REGISTER

Address Offset: 51h
Default Value: 80h
Attribute: Read/Write
Size: 8 bits

Some software packages rely on the operating speed of the processor to time certain system events. To maintain backward compatibility with these software packages, the PCMC provides a mechanism to emulate a slower operating speed. This emulation is achieved with the PCMC's deturbo mode. The deturbo mode is enabled and disabled via the DM bit in the Turbo-Reset Control Register. When the deturbo mode is enabled, the PCMC periodically asserts AHOLD to slow down the effective speed of the CPU. The duty cycle of the AHOLD active period is controlled by the DFC Register.

Bits	Description
7:6	**DETURBO MODE FREQUENCY ADJUSTMENT VALUE:** This 8-bit value effectively defines the duty cycle of the AHOLD signal. DFC[7:6] are programmable and DFC[5:0] are 0. The value programmed into this register is compared against a free running 8-bit counter running at 1/8 the CPU clock. When the counter is greater than the value specified in this register, AHOLD is asserted. AHOLD is negated when the counter value is equal to or smaller than the contents of this register. AHOLD is negated when the counter rolls over to 00h. The deturbo emulation speed is directly proportional to the value in this register. Smaller values in this register yield slower deturbo emulation speed. The value of 00h is reserved.
5:0	**RESERVED**

3.2.14 SCC—SECONDARY CACHE CONTROL REGISTER

Address Offset: 52h
Default Value: SSS01R10 (82434LX)
SSS01010 (82434NX)
(S = Strapping option)
Attribute: Read/Write
Size: 8 bits

This 8-bit register defines the secondary cache operations. The SCC Register enables and disables the second level cache, adjusts cache size, selects the cache write policy, and defines the cache SRAM type. After hard reset, SCC[7:5] contain the opposite of the signal levels sampled on the Host address lines A[31:29].

Bits	Description
7:6	**SECONDARY CACHE SIZE (SCS):** This field defines the size of the second level cache. The values sampled on the A[31:30] lines at the rising edge of the PWROK signal are inverted and stored in this field. **Bits[7:6] — Secondary Cache Size** 00 — Cache not populated 01 — Reserved 10 — 256-KBytes 11 — 512-KBytes

Bits	Description
5	**SRAM TYPE (SRAMT):** This bit selects between standard SRAMs or burst SRAMS to implement the second level cache. When SRAMT = 0, standard SRAMs are selected. When SRAMT = 1, burst SRAMs are selected. This bit reflects the signal level on the A29 pin at the rising edge of the PWROK signal. This value can be overwritten with subsequent writes to the SCC Register.
4	**82434LX: SECONDARY CACHE ALLOCATION (SCA):** SCA controls when the PCMC performs line fills in the second level cache. When SCA is set to 0, only CPU reads of cacheable main memory with CACHE# asserted are cached in the second level cache. When SCA is set to 1, all CPU reads of cacheable main memory are cached in the second level cache.
3	**CACHE BYTE CONTROL (CBC):** When programmed for asynchronous SRAMs, this bit defines whether the cache uses individual write enables per byte or has a single write enable and byte select lines per byte. When CBC is set to 1, write enable control is used. When CBC is set to 0, byte select control is used.
2	**82434LX: RESERVED** **82434NX: SRAM CONNECTIVITY (SRAMC):** This bit enables different connectivities for the second level cache. When SRAMC is set to 0, the second level cache is in 82434LX compatible mode and all connections between the PCMC and second level cache SRAMs are the same as the 82434LX. When asynchronous SRAMs are used, setting this bit to 1 enables the CCS[1:0]# functionality. CCS[1:0]# are used with asynchronous SRAMs to de-select the SRAMs, placing them in a low power standby mode. When the CPU runs a halt or stop grant special cycle, the 82434NX negates CCS[1:0]#, placing the second level cache in a power saving mode. The PCMC then asserts CCS[1:0]# (activating the SRAMs) when the CPU asserts ADS#. When using burst SRAMs, setting this bit to 1 enables the CCS1# functionality and indicates to the PCMC that no external address latch is present.
1	**82434LX: SECONDARY CACHE WRITE POLICY (SCWP):** SCWP selects between write-back and write-through cache policies for the second level cache. When SCWP = 0 and the second level cache is enabled (bit 0 = 1), the second level cache is configured for write-through mode. When SCWP = 1 and the second level cache is enabled (bit 0 = 1), the second level cache is configured for write-back mode. **82434NX: RESERVED:** Secondary cache write-through mode is not supported. The secondary cache is always in write-back mode and this bit has no affect. SCWP can be set to 0, however, the 82434NX will still operate the secondary cache in write-back mode.
0	**SECONDARY CACHE ENABLE (SCE):** SCE enables and disables the secondary cache. When SCE = 1, the secondary cache is enabled. When SCE = 0, the secondary cache is disabled. When the secondary cache is disabled, the PCMC forwards all main memory cycles to the DRAM interface. Note that setting this bit to 0 does not affect existing valid cache lines. If a cache line contains modified data, the data is not written back to memory. Valid lines in the cache remain valid. When the secondary cache is disabled, the CWE[7:0]# lines remain negated. COE[1:0]# may still toggle. When system software disables secondary caching through this register during run-time, the software should first flush the second level cache. This process is accomplished by first disabling first level caching via the PCE bit in the HCS Register. This prevents the KEN# signal from being asserted, which disables any further line fills. At this point, software executes the WBINVD instruction to flush the caches. When the instruction completes, bit 0 of this register can be reset to 0, disabling the secondary cache. The first level cache can then be enabled by writing the PCE bit in the HCS Register.

3.2.15 HBC—HOST READ/WRITE BUFFER CONTROL

Address Offset: 53h
Default Value: 00h
Attribute: Read/Write
Size: 8 bits

The HBC Register enables and disables Host-to-main memory and Host-to-PCI posting of write cycles. When posting is enabled, the write buffers in the LBX devices post the data that is destined for either main memory or PCI. This register also permits a CPU-to-main memory read cycle to be performed before any pending posted write data is written to memory.

Bits	Description
7:4	**RESERVED**
3	**READ-AROUND-WRITE ENABLE (RAWCM):** If enabled, the PCMC, during a CPU read cycle to memory where posted write cycles are pending, internally snoops the write buffers. If the address of the read differs from the posted write addresses, the PCMC initiates the memory read cycle ahead of the pending posted memory write. When RAWCM = 0, the pending posted write is written to memory before the memory read is performed. When RAWCM = 1, the PCMC initiates the memory read ahead of the pending posted memory writes.
2	**RESERVED**
1	**HOST-TO-PCI POSTING ENABLE (HPPE):** This bit enables/disables the posting of Host-to-PCI write data in the LBX posting buffers. When HPPE = 1, up to 4 Dwords of data can be posted to PCI. HPPE = 0 is reserved. Buffering is disabled and each CPU write does not complete until the PCI transaction completes (TRDY# is asserted).
0	**82434LX: HOST-TO-MEMORY POSTING ENABLE (HMPE):** This bit enables/disables the posting of Host-to-main memory write data in the LBX buffers. When HMPE = 1, the CPU can post a single write or a burst write (4 Qwords). The CPU burst write completes at 4-1-1-1 when the second level cache is in write-back mode and at 3-1-1-1 when the second level cache is either disabled or in write-through mode. When HMPE = 0, Host-to-main memory posting is disabled and the CPU write cycles do not complete until the data is written to memory. **82434NX: RESERVED:** For the 82434NX, posting is always enabled and this bit has no affect. The CPU can post a single write or burst write (4 Qwords). HMPE can be set to 0, however, the 82434NX will still allow posting of CPU-to-main memory writes.

3.2.16 PBC—PCI READ/WRITE BUFFER CONTROL REGISTER

Address Offset: 54h
Default Value: 00h
Attribute: Read/Write
Size: 8 bits

The PBC Register enables and disables PCI-to-main memory write posting and permits single CPU-to-PCI writes to be assembled into PCI burst cycles.

Bits	Description
7:3	**RESERVED**
2	**LBXs CONNECTED TO TRDY#:** The TRDY# pin on the LBXs can be connected either to the PCI TRDY# signal or to ground. The cycle time for CPU-to-PCI writes is improved if TRDY# is connected to the LBXs. Since there are two LBXs used in a system, connecting this signal to the LBXs increases the electrical loading of TRDY# by two loads. When the LBXs are externally hard-wired to TRDY#, this bit should be set to 1. Note that this should be done prior to the first Host-to-PCI write or data corruption will occur. Setting this bit to 1 enables the capability of CPU-to-PCI writes at 2-1-1-1 . . . (PCI clocks). When this bit is 0, the LBXs are not connected to TRDY# and CPU-to-PCI writes are completed at 2-2-2-2 . . . timing.
1	**PCI BURST WRITE ENABLE (PBWE):** This bit enables and disables PCI Burst memory write cycles for back-to-back sequential CPU memory write cycles to PCI. When PBWE is set to 1, PCI burst writes are enabled. When PBWE is reset to 0, PCI burst writes are disabled and each single CPU write to PCI invokes a single PCI write cycle (each cycle has an associated FRAME# sequence).
0	**PCI-TO-MEMORY POSTING ENABLE (PMPE):** This bit enables and disables posting of PCI-to-memory write cycles. The posting occurs in a pair of four Dword-deep buffers in the LBXs. When PMPE is set to 1, these buffers are used to post PCI-to-main memory write data. When PMPE is reset to 0, PCI write transactions to main memory are limited to single transfers. The PCMC asserts STOP# with the first TRDY# to disconnect the PCI Master.

3.2.17 DRAMC—DRAM CONTROL REGISTER

Address Offset: 57h
Default Value: 31h
Attribute: Read/Write
Size: 8 bits

This 8-bit register controls main memory DRAM operating modes and features.

Bits	Description
7:6	**82434LX: RESERVED** **82434NX: DRAM BURST TIMING (DBT):** The DRAM interface can be configured for 3 different burst timings. The CAS# pulse width for X-3-3-3 timing is one clock shorter than the CAS# pulse width for X-4-4-4 timing. **Bits[7:6] Burst Timing** 0 0 X-4-4-4 Read/Write timing (default) 0 1 X-4-4-4 Read, X-3-3-3 Write timing 1 0 Reserved 1 1 X-3-3-3 Read/Write timing
5	**PARITY ERROR MASK (PERRM):** When PERRM = 1, parity errors generated during DRAM read cycles initiated by either the CPU request or a PCI Master are masked. This bit affects bits 0 and 1 of the Error Command Register and the ability of the PCMC to respond to PCHK# and assert SERR# when a DRAM parity error occurs. When PERRM is reset to 0, parity errors are not masked.
4	**0-ACTIVE RAS# MODE:** This bit determines if the DRAM page for a particular row remains open (i.e. RAS# remains asserted after a DRAM cycle) enabling the possibility that the next DRAM access may be either a page hit, a page miss, or a row miss. The DRAM interface is then in 1-active RAS# mode. If this bit is reset to 0, RAS# remains asserted after a DRAM cycle. If this bit is set to 1, RAS# is negated after every DRAM cycle, resulting in a row miss for every DRAM cycle. The DRAM interface is then in 0-active RAS# mode.
3	**SMRAM ENABLE (SMRE):** When SMRE = 1, CPU accesses to SMM space are qualified with the SMIACT# pin of the CPU. The location of this space is determined by the SBS field of the SMRAM Register. Read and write cycles to SMM space function normally if SMIACT# is asserted. If SMIACT# is negated when accessing this space, the cycle is forwarded to PCI. When SMRE = 0, accesses to SMM space are treated normally and SMIACT# has no effect. SMRE must be set to 1 to enable the use of the SMRAM Register at configuration space offset 72h.
2	**BURST OF FOUR REFRESH (BFR):** When BFR is set to 1, refreshes are performed in sets of four, at a frequency 1/4 of the normal refresh rate. The PCMC defers refreshes to idle times, if possible. When BFR is reset to 0, single refreshes occur at 15.6 μs refresh rate.
1	**82434LX: REFRESH TYPE (RT):** When RT = 1, the PCMC uses CAS#-before-RAS# timing to refresh the DRAM array. For this refresh type, the PCMC does not supply refresh addresses. When RT = 0, RAS# Only refresh is used and the PCMC drives refresh addresses on the MA[10:0] lines. RAS# only refresh can be used with any type of second level cache configuration (i.e., no second level cache is present, or either a burst SRAM or standard SRAM second level cache is implemented). CAS#-before-RAS# refresh should not be used when a standard SRAM second level cache is implemented. **82434NX: REFRESH TYPE (RT):** In addition to above, when RT = 0, RAS# only refresh is used and the PCMC drives refresh addresses on the MA[11:0] lines. Also, CAS#-before-RAS# refresh can be used with a standrad SRAM second level cache.
0	**REFRESH ENABLE (RE):** When RE is set to 1, the main memory array is refreshed as configured via bits 1 and 2 of this register. When RE is reset to 0, DRAM refresh is disabled. Note that disabling refresh results in the loss of DRAM data.

3.2.18 DRAMT—DRAM TIMING REGISTER

Address Offset: 58h
Default Value: 00h
Attribute: Read/Write
Size: 8 bits

For the 82434LX, this register controls the leadoff latency for CPU DRAM accesses.

For the 82434NX, this register provides additional control over DRAM timings. One additional wait-state can be independently added before the assertion of RAS#, the assertion of the first CAS#, or both. This is to allow more flexibility in the layout of the motherboard and in the selection of DRAM speed grades.

Bits	Description
7:2	**RESERVED**
1	**82434LX: RESERVED** **82434NX: RAS# WAIT-STATE (RWS):** When RWS = 1, one additional wait state will be inserted before RAS# is asserted for row misses or page misses in 1-Active RAS mode and all cycles in 0-Active RAS mode. This provides additional MA[11:0] setup time to RAS# assertion.
0	**CAS# WAIT-STATE (CWS):** When CWS = 1, one additional wait state will be inserted before the first assertion of CAS# within a burst cycle. There is no additional delay between CAS# assertions. This provides additional MA[11:0] setup time to CAS# assertion. The CWS bit is typically reset to 0 for 60 MHz operation and set to 1 for 66 MHz operation.

3.2.19 PAM—PROGRAMMABLE ATTRIBUTE MAP REGISTERS (PAM[6:0])

Address Offset: 59–5Fh
Default Value: PAM0 = 0Fh, PAM[1:6] = 00h
Attribute: Read/Write

The PCMC allows programmable memory and cacheability attributes on 14 memory segments of various sizes in the 512 KByte–1 MByte address range. Seven Programmable Attribute Map (PAM) Registers are used to support these features. Three bits are used to specify cacheability and memory attributes for each memory segment. These attributes are:

RE: **Read Enable.** When RE = 1, the CPU read accesses to the corresponding memory segment are directed to main memory. Conversely, when RE = 0, the CPU read accesses are directed to PCI.

WE: **Write Enable.** When WE = 1, the CPU write accesses to the corresponding memory segment are directed to main memory. Conversely, when WE = 0, the CPU write accesses are directed to PCI.

CE: **Cache Enable.** When CE = 1, the corresponding memory segment is cacheable. CE must not be set to 1 when RE is reset to 0 for any particular memory segment. When CE = 1 and WE = 0, the corresponding memory segment is cached in the first and second level caches only on CPU coded read cycles.

The RE and WE attributes permit a memory segment to be Read Only, Write Only, Read/Write, or disabled. For example, if a memory segment has RE = 1 and WE = 0, the segment is Read Only. The characteristics for memory segments with these read/write attributes are described in Table 2.

Table 2. Attribute Definition

Read/Write Attribute	Definition
Read Only	Read cycles: CPU cycles are serviced by the DRAM in a normal manner. Write cycles: CPU initiated write cycles are ignored by the DRAM interface as well as the cache. Instead, the cycles are passed to PCI for termination. Areas marked as Read Only are cacheable for Code accesses only. These regions may be cached in the second level cache, however as noted above, writes are forwarded to PCI, effectively write protecting the data.
Write Only	Read cycles: All read cycles are ignored by the DRAM interface as well as the second level cache. CPU-initiated read cycles are passed onto PCI for termination. The write only state can be used while copying the contents of a ROM, accessible on PCI, to main memory for shadowing, as in the case of BIOS shadowing. Write cycles: CPU write cycles are serviced by the DRAM and cache in a normal manner.
Read/Write	This is the normal operating mode of main memory. Both read and write cycles from the CPU and PCI are serviced by the DRAM and cache interface.
Disabled	All read and write cycles to this area are ignored by the DRAM and cache interface. These cycles are forwarded to PCI for termination.

Each PAM Register controls two regions, typically 16-KByte in size. Each of these regions have a 4-bit field. The four bits that control each region have the same encoding and are defined in Table 3.

Table 3. Attribute Bit Assignment

Bits[7,3] Reserved	Bits[6,2] Cache Enable	Bits[5,1] Write Enable	Bits[4,0] Read Enable	Description
x	x	0	0	DRAM Disabled, Accesses Directed to PCI
x	0	0	1	Read Only, DRAM Write Protected, Non-Cacheable
x	1	0	1	Read Only, DRAM Write Protected, Cacheable for Code Accesses Only
x	0	1	0	Write Only
x	0	1	1	Read/Write, Non-Cacheable
x	1	1	1	Read/Write, Cacheable

NOTE:
To enable PCI master access to the DRAM address space from C0000h to FFFFFh the MEMCS# configuration registers of the ISA or EISA bridge must be properly configured. These registers must correspond to the PAM Registers in the PCMC.

As an example, consider a BIOS that is implemented on the expansion bus. During the initialization process the BIOS can be shadowed in main memory to increase the system performance. When a BIOS is shadowed in main memory, it should be copied to the same address location. To shadow the BIOS, the attributes for that address range should be set to write only. The BIOS is shadowed by first doing a read of that address. This read is forwarded to the expansion bus. The CPU then does a write of the same address, which is directed to main memory. After the BIOS is shadowed, the attributes for that memory area are set to read only so that all writes are forwarded to the expansion bus.

Table 4. PAM Registers and Associated Memory Segments

PAM Reg	Attribute Bits		Memory Segment		Comments		Offset
PAM0[3:0]	R	CE	WE	RE	080000h–09FFFFh	512K–640K	59h
PAM0[7:4]	R	CE	WE	RE	0F0000h–0FFFFFh	BIOS Area	59h
PAM1[3:0]	R	CE	WE	RE	0C0000h–0C3FFFh	ISA Add-on BIOS	5Ah
PAM1[7:4]	R	CE	WE	RE	0C4000h–0C7FFFh	ISA Add-on BIOS	5Ah
PAM2[3:0]	R	CE	WE	RE	0C8000h–0CBFFFh	ISA Add-on BIOS	5Bh
PAM2[7:4]	R	CE	WE	RE	0CC000h–0CFFFFh	ISA Add-on BIOS	5Bh
PAM3[3:0]	R	CE	WE	RE	0D0000h–0D3FFFh	ISA Add-on BIOS	5Ch
PAM3[7:4]	R	CE	WE	RE	0D4000h–0D7FFFh	ISA Add-on BIOS	5Ch
PAM4[3:0]	R	CE	WE	RE	0D8000h–0DBFFFh	ISA Add-on BIOS	5Dh
PAM4[7:4]	R	CE	WE	RE	0DC000h–0DFFFFh	ISA Add-on BIOS	5Dh
PAM5[3:0]	R	CE	WE	RE	0E0000h–0E3FFFh	BIOS Extension	5Eh
PAM5[7:4]	R	CE	WE	RE	0E4000h–0E7FFFh	BIOS Extension	5Eh
PAM6[3:0]	R	CE	WE	RE	0E8000h–0EBFFFh	BIOS Extension	5Fh
PAM6[7:4]	R	CE	WE	RE	0EC000h–0EFFFFh	BIOS Extension	5Fh

DOS Application Area (00000h-9FFFh)

The 640-KByte DOS application area is split into two regions. The first region is 0–512-KByte and the second region is 512–640 KByte. Read, write, and cacheability attributes are always enabled and are not programmable for the 0–512 KByte region.

Video Buffer Area (A0000h-BFFFFh)

This 128-KByte area is not controlled by attribute bits. CPU-initiated cycles in this region are always forwarded to PCI for termination. This area is not cacheable.

Expansion Area (C0000h-DFFFFh)

This 128-KByte area is divided into eight 16-KByte segments. Each segment can be assigned one of four Read/Write states: read-only, write-only, read/write, or disabled Memory that is disabled is not remapped. Cacheability status can also be specified for each segment.

Extended System BIOS Area (E0000h-EFFFFh)

This 64-KByte area is divided into four 16-KByte segments. Each segment can be assigned independent cacheability, read, and write attributes. Memory segments that are disabled are not remapped elsewhere.

System BIOS Area (F0000h-FFFFFh)

This area is a single 64-KByte segment. This segment can be assigned cacheability, read, and write attributes. When disabled, this segment is not remapped.

Extended Memory Area (100000h-FFFFFFFFh)

The extended memory area can be split into several parts:

- Flash BIOS area from 4 GByte to 4 GByte–512-KByte (aliased on ISA at 16 MBytes–15.5 MBytes)
- DRAM Memory from 1 MByte to a maximum of 192 MBytes
- PCI Memory space from the top of DRAM to 4 GByte – 512-KByte
- Memory Space Gap between the range of 1 MByte up to 15.5 MBytes
- Frame Buffer Range mapped into PCI Memory Space or the Memory Space Gap.

On power-up or reset the CPU vectors to the Flash BIOS area, mapped in the range of 4 GByte to 4 GByte – 512-KByte. This area is physically mapped on the expansion bus. Since these addresses are in the upper 4 GByte range, the request is directed to PCI.

The DRAM memory space can occupy extended memory from a minimum of 2 MBytes up to 192 MBytes. This memory is cacheable.

The address space on PCI between the Flash BIOS (4 GByte to 4 GByte – 512 KByte) and the top of DRAM (including any remapped memory) may be occupied by PCI memory. This memory space is not cacheable.

3.2.20 DRB—DRAM ROW BOUNDARY REGISTERS

Address Offset: 60–65h (82434LX)
60–67h (82434NX)
Default Value: 02h
Attribute: Read/Write
Size: 8 bits

Note the address offset for each DRB Register is DRB0=60h, DRB1=61h, DRB2=62h, DRB3=63h, DRB4=64h, DRB5=65h, DRB6=66h, and DRB7=67h.

3.2.20.1 82434LX Description

The PCMC supports 6 rows of DRAM. Each row is 64 bits wide. The DRAM Row Boundary Registers define upper and lower addresses for each DRAM row. Contents of these 8-bit registers represent the boundary addresses in MBytes.

DRB0 = Total amount of memory in row 0 (in MBytes)
DRB1 = Total amount of memory in row 0 + row 1 (in MBytes)
DRB2 = Total amount of memory in row 0 + row 1 + row 2 (in MBytes)
DRB3 = Total amount of memory in row 0 + row 1 + row 2 + row 3 (in MBytes)
DRB4 = Total amount of memory in row 0 + row 1 + row 2 + row 3 + row 4 (in MBytes)
DRB5 = Total amount of memory in row 0 + row 1 + row 2 + row 3 + row 4 + row 5 (in MBytes)

The DRAM array can be configured with 256K x 36, 1M x 36 and 4M x 36 SIMMs. Each register defines an address range that will cause a particular RAS# line to be asserted (e.g. if the first DRAM row is 2 MBytes in size then accesses within the 0 MByte–2 MBytes range will cause RAS0# to be asserted). The DRAM Row

Boundary (DRB) Registers are programmed with an 8-bit upper address limit value. This upper address limit is compared to A[27:20] of the Host address bus, for each row, to determine if DRAM is being targeted. Since this value is 8 bits and the resolution is 1 MByte, the total bits compared span a 256 MByte space. However, only 192 MBytes of main memory is supported.

Bits	Description
7:0	**ROW BOUNDARY ADDRESS IN MBYTES:** This 8-bit value is compared against address lines A[27:20] to determine the upper address limit of a particular row, i.e. DRB − previous DRB = row size.

Row Boundary Address in MBytes

These 8-bit values represent the upper address limits of the six rows (i.e., this row - previous row = row size). Unpopulated rows have a value equal to the previous row (row size = 0). The value programmed into DRB5 reflects the maximum amount of DRAM in the system. Memory remapped at the top of DRAM, as a result of setting the Memory Space Gap Register, is not reflected in the DRB Registers. The top of memory is always determined by the value written into DRB5 added to the memory space gap size (if enabled).

As an example of a general purpose configuration where 3 physical rows are configured for either single-sided or double-sided SIMMs, the memory array would be configured like the one shown in Figure 8. In this configuration, the PCMC drives two RAS# signals directly to the SIMM rows. If single-sided SIMMs are populated, the even RAS# signal is used and the odd RAS# is not connected. If double-sided SIMMs are used, both RAS# signals are used.

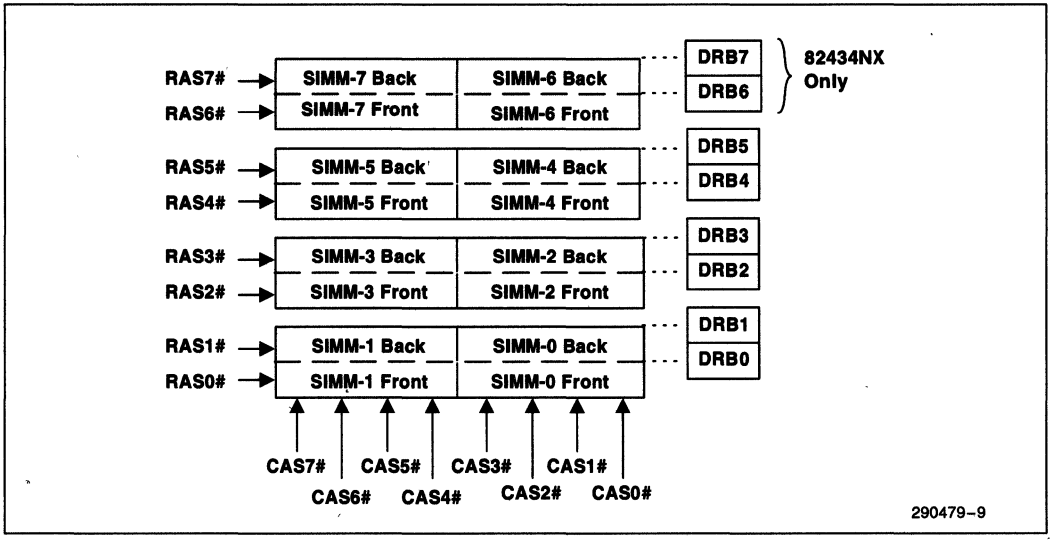


Figure 8. SIMMs and Corresponding DRB Registers

The following 2 examples describe how the DRB Registers are programmed for cases of single-sided and double-sided SIMMs on a motherboard having a total of 6 SIMM sockets.

Example #1

The memory array is populated with six single-sided 256-KByte x 36 SIMMs. Two SIMMs are required for each populated row making each populated row 2 MBytes in size. Filling the array yields 6 MBytes total DRAM. The DRB Registers are programmed as follows:

DRB0 = 02h populated
DRB1 = 02h empty row, not double-sided SIMMs
DRB2 = 04h populated
DRB3 = 04h empty row, not double-sided SIMMs
DRB4 = 06h populated
DRB5 = 06h empty row, not double-sided SIMMs, maximum memory = 6 MBytes.

Example #2

As an another example, if the first four SIMM sockets are populated with 2 MBytes x 36 double-sided SIMMs and the last two SIMM sockets are populated with 4 MBytes x 36 single-sided SIMMs then filling the array yields 64 MBytes total DRAM. The DRB Registers are programmed as follows:

DRB0 = 08h populated with 8 MBytes, ½ of the double-sided SIMMs
DRB1 = 10h the other 8 MBytes of the double-sided SIMMs
DRB2 = 18h populated with 8 MBytes, ½ of the double-sided SIMMs
DRB3 = 20h the other 8 MBytes of the double-sided SIMMs
DRB4 = 40h populated with 32 MBytes
DRB5 = 40h empty row, not double-sided SIMMs, maximum memory = 64 MBytes.

3.2.20.2 82434NX Description

The PCMC supports 8 rows of DRAM. Each row is 64 bits wide. The DRAM Row Boundary Registers define upper and lower addresses for each DRAM row. Contents of these 8-bit registers are concatenated with the associated nibble of the DRBE Register to form 12 bit quantities that represent the row boundary addresses in MBytes.

DRBE[3:0]	DRB0 =	Total amount of memory in row 0 (in MBytes)
DRBE[7:4]	DRB1 =	Total amount of memory in row 0 + row 1 (in MBytes)
DRBE[11:8]	DRB2 =	Total amount of memory in row 0 + row 1 + row 2 (in MBytes)
DRBE[15:12]	DRB3 =	Total amount of memory in row 0 + row 1 + row 2 + row 3 (in MBytes)
DRBE[19:16]	DRB4 =	Total amount of memory in row 0 + row 1 + row 2 + row 3 + row 4 (in MBytes)
DRBE[23:20]	DRB5 =	Total amount of memory in row 0 + row 1 + row 2 + row 3 + row 4 + row 5 (in Bytes)
DRBE[27:24]	DRB6 =	Total amount of memory in row 0 + row 1 + row 2 + row 3 + row 4 + row 5 + row 6 (in MBytes)
DRBE[31:28]	DRB7 =	Total amount of memory in row 0 + row 1 + row 2 + row 3 + row 4 + row 5 + row 6 + row 7 (in MBytes)

The DRAM array can be configured with 256K x 36, 1M x 36, 4M x 36, and 16M x 36 SIMMs. Each register defines an address range that will cause a particular RAS# line to be asserted (e.g. if the first DRAM row is 2 MBytes in size then accesses within the 0 to 2 MBytes range will cause RAS0# to be asserted). The DRAM Row Boundary (DRB) Registers are programmed with an 8-bit upper address limit value. The DRBE Register extends the programming model of this mechanism to 12 bits, however only 10 bits are implemented at this time. This upper address limit is compared to A[29:20] of the Host address bus, for each row, to determine if DRAM is being targeted. Since this value is 10 bits and the resolution is 1 MByte, the total bits compared span a 1 GByte space. However, other resource limits in the PCMC cap the total usable DRAM space at 512 MBytes.

Bits	Description
7:0	**ROW BOUNDARY ADDRESS IN MBYTES:** This 8-bit value is concatenated with a nibble from the DRBE Register and then compared against address lines A[29:20] to determine the upper address limit of a particular row (i.e. DRB − previous DRB = row size).

Row Boundary Address in MBytes

These 10-bit values represent the upper address limits of the 8 rows (i.e., this row - previous row = row size). Unpopulated rows have a value equal to the previous row (row size = 0). The value programmed into DRBE[31:28] || DRB7 reflects the maximum amount of DRAM in the system. Memory remapped at the top of DRAM, as a result of setting the Memory Space Gap Register, is not reflected in the DRB Registers. The top of memory is determined by the value written into DRBE[31:28] || DRB7 added to the memory space gap size (if enabled). If DRBE[31:28] || DRB7 plus the memory space gap is greater than 512 MBytes then 512 MBytes of DRAM are available.

The following 2 examples describe how the DRB Registers are programmed for cases of single-sided and double-sided SIMMs on a motherboard having a total of 8 SIMM sockets.

Example #1

The memory array is populated with eight single-sided 256-KByte x 36 SIMMs. Two SIMMs are required for each populated row making each populated row 2 MBytes in size. Filling the array yields 8 MBytes total DRAM. The DRB Registers are programmed as follows:

DRBE[3:0] = 0h	DRB0 = 02h	populated
DRBE[7:4] = 0h	DRB1 = 02h	empty row, not double-sided SIMMs
DRBE[11:8] = 0h	DRB2 = 04h	populated
DRBE[15:12] = 0h	DRB3 = 04h	empty row, not double-sided SIMMs
DRBE[19:16] = 0h	DRB4 = 06h	populated
DRBE[23:20] = 0h	DRB5 = 06h	empty row, not double-sided SIMMs
DRBE[27:24] = 0h	DRB6 = 08h	populated
DRBE[31:28] = 0h	DRB7 = 08h	empty row, not double-sided SIMMs, max memory = 8 MBytes.

Example #2

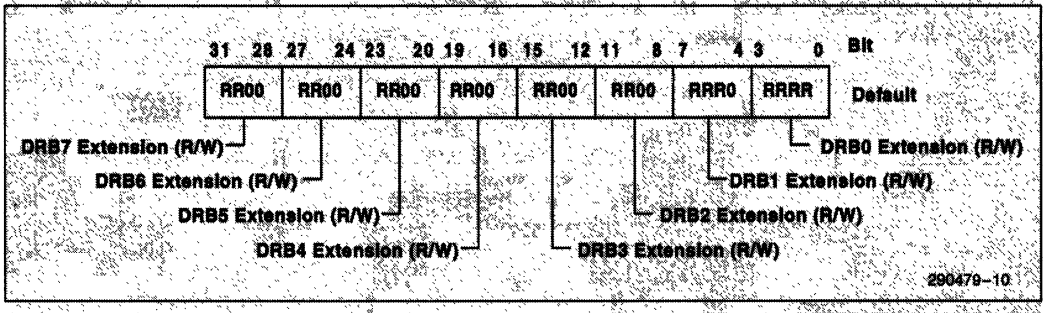
As an another example, if the first four SIMM sockets are populated with 2 MByte x 36 double-sided SIMMs and the last four SIMM sockets are populated with 16 MByte x 36 single-sided SIMMs then filling the array yields 288 MBytes total DRAM. The DRB Registers are programmed as follows:

DRBE[3:0] = 0h	DRB0 = 08h	populated with 8 MBytes, ½ of double-sided SIMMs
DRBE[7:4] = 0h	DRB1 = 10h	the other 8 MBytes of the double-sided SIMMs
DRBE[11:8] = 0h	DRB2 = 18h	populated with 8 MBytes, ½ of double-sided SIMMs
DRBE[15:12] = 0h	DRB3 = 20h	the other 8 MBytes of the double-sided SIMMs
DRBE[19:16] = 0h	DRB4 = A0h	populated with 128 MBytes
DRBE[23:20] = 0h	DRB5 = A0h	empty row, not double-sided SIMMs
DRBE[27:24] = 1h	DRB6 = 20h	populated with 128 MBytes
DRBE[31:28] = 1h	DRB7 = 20h	empty row, not double-sided SIMMs, max memory = 288 MBytes.

3.2.21 DRBE—DRAM ROW BOUNDARY EXTENSION REGISTER

Address Offset: 68-6Bh
Default Value: 0000h
Attribute: Read/Write
Size: 32 bits

The DRBE Register is not implemented in the 82434LX. This register contains an extension for each of the DRAM Row Boundary (DRB) Registers. Each nibble of the DRBE Register is concatenated with a DRB Register (see DRB Register section for details on the use of the DRB and DRBE Registers).



Bits	Description
31:0	**EXTENSIONS FOR DRB0 THROUGH DRB7:** Each nibble corresponds to a DRB. The nibble of the DRBE and its corresponding DRB are concatenated and used to indicate the boundaries between rows of DRAM.

3.2.22 ERRCMD—ERROR COMMAND REGISTER

Address Offset: 70h
Default Value: 00h
Attribute: Read/Write
Size: 8 bits

The Error Command Register controls the PCMC responses to various system errors. Bit 6 of the PCICMD Register is the master enable for bit 3 of this register. Bit 6 of the PCICMD Register must be set to 1 to enable the error reporting function defined by bit 3 of this register. Bits 6 and 8 of the PCICMD Register are the master enables for bits 7, 6, 5, 4, and 1 of this register. Both bits 6 and 8 of the PCICMD Register must be set to 1 to enable the error reporting functions defined by bits 7, 6, 5, 4, and 1 of this register.

Bits	Description
7	**SERR# ON RECEIVED TARGET ABORT:** When this bit is set to 1 (and bit 8 of the PCICMD Register is 1), the PCMC asserts SERR# upon receiving a target abort. When this bit is set to 0, the PCMC is disabled from asserting SERR# upon receiving a target abort.
6	**SERR# ON TRANSMITTED PCI DATA PARITY ERROR:** When this bit is set to 1 (and bits 6 and 8 of the PCICMD Register are both 1), the PCMC asserts SERR# when it detects a data parity error as a result of a CPU-to-PCI write (PERR# detected asserted). When this bit is set to 0, the PCMC is disabled from asserting SERR# when data parity errors are detected via PERR#.
5	**82434LX: RESERVED** **82434NX: SERR# ON RECEIVED PCI DATA PARITY ERROR:** When this bit is set to 1 (and bits 6 and 8 of the PCICMD Register are both 1), the PCMC asserts SERR# when it detects a data parity error as a result of a CPU-to-PCI read (PAR incorrect with received data). In this case, the SERR# signal is asserted when parity errors are detected on PCI return data. When this bit is set to 0, the PCMC is disabled from asserting SERR# when data parity errors are detected during a CPU-to-PCI read.
4	**82434LX: RESERVED** **82434NX: SERR# ON PCI ADDRESS PARITY ERROR:** When this bit is set to 1 (and bits 6 and 8 of the PCICMD Register are both 1), the PCMC asserts SERR# when it detects an address parity error on PCI transactions. When this bit is set to 0, the PCMC is disabled from asserting SERR# when address parity errors are detected on PCI transactions.
3	**82434LX: RESERVED** **82434NX: PERR# ON RECEIVING A DATA PARITY ERROR:** This bit indicates whether the PERR# signal is implemented in the system. When this bit is set to 1 (and bit 6 of the PCICMD Register is 1), the PCMC asserts PERR# when it detects a data parity error (PAR incorrect with received data), either from a CPU-to-PCI read or a PCI master write to memory. When this bit is set to 0 (or bit 6 of the PCICMD Register is set to 0), the PERR# signal is not asserted by the PCMC.
2	**L2 CACHE PARITY ENABLE:** This bit indicates that the second level cache implements parity. When this bit is set to 1, bits 0 and 1 of this register control the checking of parity errors during CPU reads from the second level cache. If this bit is 0, parity is not checked when the CPU reads from the second level cache (PCHK# ignored) and neither bit 1 nor bit 0 apply.
1	**SERR# ON DRAM/L2 CACHE DATA PARITY ERROR ENABLE:** This bit enables/disables the SERR# signal for parity errors on reads from main memory or the second level cache. When this bit is set to 1 and bit 0 of this register is set to 1 (and bits 6 and 8 of the PCICMD Register are set to 1), SERR# is enabled upon a PCHK# assertion from the CPU when reading from main memory or the second level cache. The processor indicates that a parity error was received by asserting PCHK#. The PCMC then latches status information in the Error Status Register and asserts SERR#. When this bit is 0, SERR# is not asserted upon detecting a parity error. Bits[1:0] = 10 is a reserved combination. 0 = Disable assertion of SERR# upon detecting a DRAM/second level cache read parity error. 1 = Enable assertion of SERR# upon detecting a DRAM/second level cache read parity error.
0	**MCHK ON DRAM/L2 CACHE DATA PARITY ERROR ENABLE:** When this bit is set to 1, PEN# is asserted for data returned from main memory or the second level cache. The processor indicates that a parity error was received by asserting the PCHK# signal. In addition, the processor invokes a machine check exception, if enabled via the MCE bit in CR4 in the Pentium processor. The PCMC then latches status information in the Error Status register. When this bit is 0, PEN# is not asserted. Bits[1:0] = 10 is a reserved combination.

3.2.23 ERRSTS—ERROR STATUS REGISTER

Address Offset: 71h
Default Value: 00h
Attribute: Read/Write Clear
Size: 8 bits

The Error Status Register is an 8-bit register that reports the occurrence of PCI, second level cache, and DRAM parity errors. This register also reports the occurrence of a CPU shutdown cycle.

Bits	Description
7	**RESERVED**
6	**PCI TRANSMITTED DATA PARITY ERROR:** The PCMC sets this bit to a 1 when it detects a data parity error (PERR# asserted) as a result of a CPU-to-PCI write. Software resets this bit to 0 by writing a 1 to it.
5	**82434LX: RESERVED** **82434NX: PCI RECEIVED DATA PARITY ERROR:** The PCMC sets this bit to a 1 when it detects a data parity error (PAR incorrect with received data) as a result of a CPU-to-PCI read. Software resets this bit to 0 by writing a 1 to it.
4	**82434LX: RESERVED** **82434NX: PCI ADDRESS PARITY ERROR:** The PCMC sets this bit to a 1 when it detects an address parity error (PAR incorrect with received address and C/BE# lines) on a PCI master transaction. Software resets this bit to 0 by writing a 1 to it.
3	**MAIN MEMORY DATA PARITY ERROR:** The PCMC sets this bit to a 1 when it detects a parity error from the CPU PCHK# signal resulting from a CPU-to-main memory read. Software resets this bit to 0 by writing a 1 to it.
2	**L2 CACHE DATA PARITY ERROR:** The PCMC sets this bit to a 1 when it detects a parity error from the CPU PCHK# signal resulting from a CPU read access that hit in the second level cache. Software resets this bit to 0 by writing a 1 to it.
1	**RESERVED**
0	**SHUTDOWN CYCLE DETECTED:** The PCMC sets this bit to a 1 when it detects a shutdown special cycle on the Host Bus. Under this condition the PCMC drives a shutdown special cycle on PCI and asserts INIT. Software resets this bit to 0 by writing a 1 to it.

3.2.24 SMRS—SMRAM SPACE REGISTER

Address Offset: 72h
Default Value: 00h
Attribute: Read/Write
Size: 8 bits

The PCMC supports a 64-KByte SMRAM space that can be selected to reside at the top of main memory, segment A0000–AFFFFh or segment B0000–BFFFFh. The SMM space defined by this register is not cacheable. This register defines a mechanism that allows the CPU to execute code out of the SMM space at either A0000h or B0000h while accessing the frame buffer on PCI. The SMRAM Enable bit in the DRAM Control Register must be 1 to enable the features defined by this register. Register bits[5:3] apply only when segment A0000-AFFFFh or B0000-BFFFFh are selected.

Bits	Description
7:6	**RESERVED**
5	**OPEN SMRAM SPACE (OSS):** When OSS = 1, the CPU can access SMM space without being in SMM mode. That is, accesses to SMM space are permitted even with SMIACT# negated. This bit is intended to be used during POST to allow the CPU to initialize SMRAM space before the first SMI# interrupt is issued.
4	**CLOSE SMRAM SPACE (CSS):** When CSS = 1 and SMRAM is enabled, CPU code accesses to the SMM memory range are directed to SMM space in main memory and data accesses are forwarded to PCI. This bit allows the CPU to read and write the frame buffer on PCI while executing SMM code. When CSS = 0 and SMRAM is enabled, all accesses to the SMRAM memory range, both code and data, are directed to SMRAM (main memory).
3	**LOCK SMRAM SPACE (LSS):** When LSS = 1, this bit prevents the SMM space from being manually opened, effectively disabling bit 5 of this register. Only a power-on reset can set this bit to 0.
2:0	**SMM BASE SEGMENT (SBS):** This field defines the 64 KByte base segment where SMM space is located. The memory that is defined by this field is non-cacheable.

Bits[2:0]	SMRAM Location	Bits[2:0]	SMRAM Location
000	Top of main memory	100	Reserved
001	Reserved	101	Reserved
010	A0000–AFFFFh	110	Reserved
011	B0000–BFFFFh	111	Reserved

3.2.25 MSG—MEMORY SPACE GAP REGISTER

Address Offset: 78-79h
Default Value: 00h
Attribute: Read/Write
Size: 16 bits

The Memory Space Gap Register defines the starting address and size of a gap in main memory. This register accommodates ISA devices that have their memory mapped into the 1 MByte–15.5 MByte range (e.g., an ISA LAN card or an ISA frame buffer). The Memory Space Gap Register defines a hole in main memory that transfers the cycles in this address space to the PCI Bus instead of main memory. This area is not cacheable.

The memory space gap starting address must be a multiple of the memory space gap size. For example, a 2 MByte gap must start at 2, 4, 6, 8, 10, 12, or 14 MBytes.

NOTE:
Memory that is disabled by the gap created by this register is remapped to the top of memory. This remapped memory is accessible, except in the case where this would cause the top of main memory to exceed 192 MBytes (or 512 MBytes for the 82434NX).

Bits	Description
15	**MEMORY SPACE GAP ENABLE (MSGE):** MSGE enables and disables the memory space gap. When MSGE is set to 1, the CPU accesses to the address range defined by this register are forwarded to PCI bus. The size of the gap created in main memory causes a corresponding amount of DRAM to be remapped at the top of main memory (top specified by DRB Registers). If the Frame Buffer Range is programmed below 16 MBytes and within main memory space, the MSG register must include the Frame Buffer Range. When MSGE is reset to 0, the memory space gap is disabled.
14:12	**MEMORY SPACE GAP SIZE (MSGS):** This 3 bit field defines the size of the memory space gap. If the Frame Buffer Range is programmed below 16 MBytes and within main memory space, this register must include the frame buffer range. The amount of main memory specified by these bits is remapped to the top of main memory. **Bit[14:12] Memory Gap Size** 000 1 MByte 001 2 MBytes 011 4 MBytes 111 8 MBytes **NOTE:** All other combinations are reserved.
11:8	**RESERVED**
7:4	**MEMORY SPACE GAP STARTING ADDRESS (MSGSA):** These 4 bits define the starting address of the memory space gap in the space from 1 MByte–16 MBytes. These bits are compared against A[23:20]. The memory space gap starting address must be a multiple of the memory space gap size. For example, a 2 MBytes gap must start at 2, 4, 6, 8, 10, 12, or 14 MBytes.
3:0	**RESERVED**

3.2.26 FBR—FRAME BUFFER RANGE REGISTER

Address Offset: 7C-7Fh
Default Value: 0000h
Attribute: Read/Write
Size: 32 bits

This 32-bit register enables and disables a frame buffer area and provides attribute settings for the frame buffer area. The attributes defined in this register are intended to increase the performance of the frame buffer. The FBR Register can be used to accommodate PCI devices that have their memory mapped onto PCI from the top of main memory to 4 GByte–512-KByte range (e.g., a linear frame buffer). If the Frame Buffer Range is located within the 1 MByte–16 MBytes main memory region where DRAM is populated, the Memory Space Gap Register must be programmed to include the Frame Buffer Range.

Bits	Description
31:20	**BUFFER OFFSET (BO):** BO defines the starting address of the frame buffer address space in increments of 1 MByte. This 12-bit field is compared directly against A[31:20]. The frame buffer range can either be located at the top of memory, including remapped memory or within the memory space gap (i.e., frame buffer range programmed below 16 MBytes and within main memory space. When bits [31:20] = 0000h and bit 12 = 0, all features defined by this register are disabled.
19:14	**RESERVED**
13	**BYTE MERGING (BM):** Byte merging permits CPU-to-PCI byte writes to the LBX posted write buffer to be combined into a single transfer on the PCI Bus, when appropriate. When BM is set to 1, byte merging on CPU-to-PCI posted write cycles is enabled. When BM is reset to 0, byte merging is disabled.
12	**128K VGA RANGE ATTRIBUTE ENABLE (VRAE):** When VRAE = 1, the attributes defined in this register (bits [13, 10:7]) also apply to the VGA memory range of A0000h–BFFFFh regardless of the value programmed in the Buffer Offset field. When VRAE = 0, the attributes do not apply to the VGA memory range. Note that this bit only affects the mentioned attributes of the VGA memory range and does not enable or disable accesses to the VGA memory range.
11:10	**RESERVED**
9	**NO LOCK REQUESTS (NLR):** When NLR is set to 1, the PCMC never requests exclusive access to a PCI resource via the PCI LOCK# signal in the range defined by this register. When NLR is reset to 0, exclusive access via the PCI LOCK# signal in the range defined by this register is enabled.
8	**RESERVED**
7	**TRANSPARENT BUFFER WRITES (TBW):** When set to a 1, this bit indicates that writes to the Frame Buffer Range need not be flushed for deadlock or coherence reasons on synchronization events (i.e., PCI master reads, and the FLSHBUF#/MEMREQ# protocol). When reset to 0, this bit indicates that upon synchronization events, flushing is required for Frame Buffer writes posted in the CPU-to-PCI Write Buffer in the LBX
6:4	**RESERVED**
3:0	**BUFFER RANGE (BR):** These bits define the size of the frame buffer address space, allowing up to 16 MBytes of frame buffer. If the Frame Buffer Range is within the memory space gap, the buffer range is limited to 8 MBytes and must be included within the memory space gap. The bits listed below in the Reserved Buffer Offset (BO) Bits column are ignored by the PCMC for the corresponding buffer sizes. **Bits[3:0] — Buffer Size — Reserved Buffer Offset (BO) Bits** 0000 — 1 MByte — None 0001 — 2 MBytes — [20] 0011 — 4 MBytes — [21:20] 0111 — 8 MBytes — [22:20] 1111 — 16 MBytes — [23:20] **NOTE:** (all other combinations are reserved)

4.0 PCMC ADDRESS MAP

The Pentium processor has two distinct physical address spaces: Memory and I/O. The memory address space is 4 GBytes and the I/O address space is 64 KBytes. The PCMC maps accesses to these address spaces as described in this section.

4.1 CPU Memory Address Map

Figure 9 shows the address map for the 4 GByte Host CPU memory address space. Depending on the address range and whether a memory gap is enabled via the MSG Register, the PCMC forwards CPU memory accesses to either main memory or PCI memory. Accesses forwarded to main memory invoke operations on the DRAM interface and accesses forwarded to PCI memory invoke operations on PCI. Mapping to the PCI Bus permits PCI or EISA/ISA Bus-based memory.

The main memory size ranges from 2 MBytes–192 MBytes for the 82434LX and 2 MBytes–512 MBytes for the 82434NX. Memory accesses above 192 MBytes (512 MBytes for the 82434NX) are always forwarded to PCI. In addition, a memory gap can be created in the 1 MByte–16 MBytes region that provides a window to PCI-based memory. The location and size of the gap is programmable. Accesses to addresses in the gap are ignored by the DRAM controller and forwarded to PCI. Note that CPU memory accesses that are forwarded to PCI (including the Memory Space Gap) are not cacheable. Only main memory controlled by the PCMC DRAM interface is cacheable.

4.2 System Management RAM—SMRAM

The PCMC supports the use of main memory as System Management RAM (SMRAM) enabling the use of System Management Mode. This function is enabled and disabled via the DRAM Control Register. When this function is disabled, the PCMC memory map is defined by the DRB and PAM Registers. When SMRAM is enabled, the PCMC reserves the top 64-KBytes of main memory for use as SMRAM.

SMRAM can also be placed at A0000–AFFFFh or B0000–BFFFFh via the SMRAM Space Register. Enhanced SMRAM features can also be enabled via this register. PCI masters can not access SMRAM when it is programmed to the A or B segments.

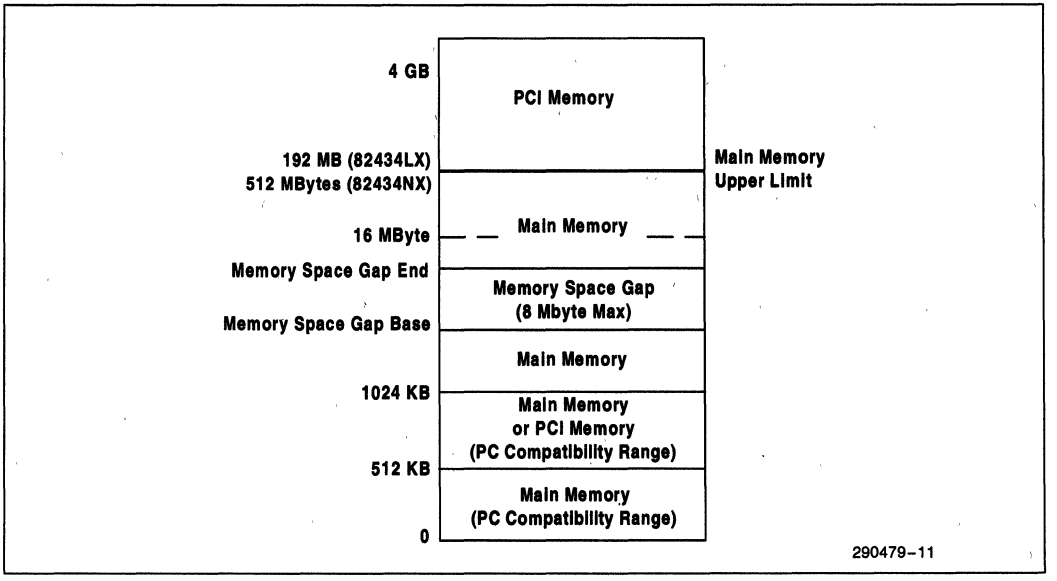


Figure 9. CPU Memory Address Map—Full Range

However, PCI masters can access SMRAM when the top of memory is selected.

When the 82434NX PCMC detects a CPU stop grant special cycle (M/IO# = 0, D/C# = 0, W/R# = 1, A4 = 1, BE[7:0]# = FBh), it generates a PCI Stop Grant Special cycle, with 0002h in the message field (AD[15:0]) and 0012h in the message dependent data field (AD[31:16]) during the first data phase (IRDY# asserted).

4.3 PC Compatibility Range

The PC Compatibility Range is the first MByte of the Memory Map. The 512 KByte–1 MByte range is subdivided into several regions as shown in Figure 10. Each region is provided with programmable attributes in the PAM Registers. The attributes are Read Enable (RE), Write Enable (WE) and Cache Enable (CE). The attributes determine readability, writeability and cacheability of the corresponding memory region. When the associated bit in the PAM Register is set to a 1, the attribute is enabled and when set to a 0 the attribute is disabled. The following rules apply for cacheability in the first level and second level caches:

1. If RE = 1, WE = 1, and CE = 1, the region is cacheable in the first level and second level caches.
2. If RE = 1, WE = 0, and CE = 1, the region is cacheable only on code reads (i.e., D/C# = 0). Data reads do not result in a line fill. Writes to the region are not serviced by the secondary cache, but are forwarded to PCI.

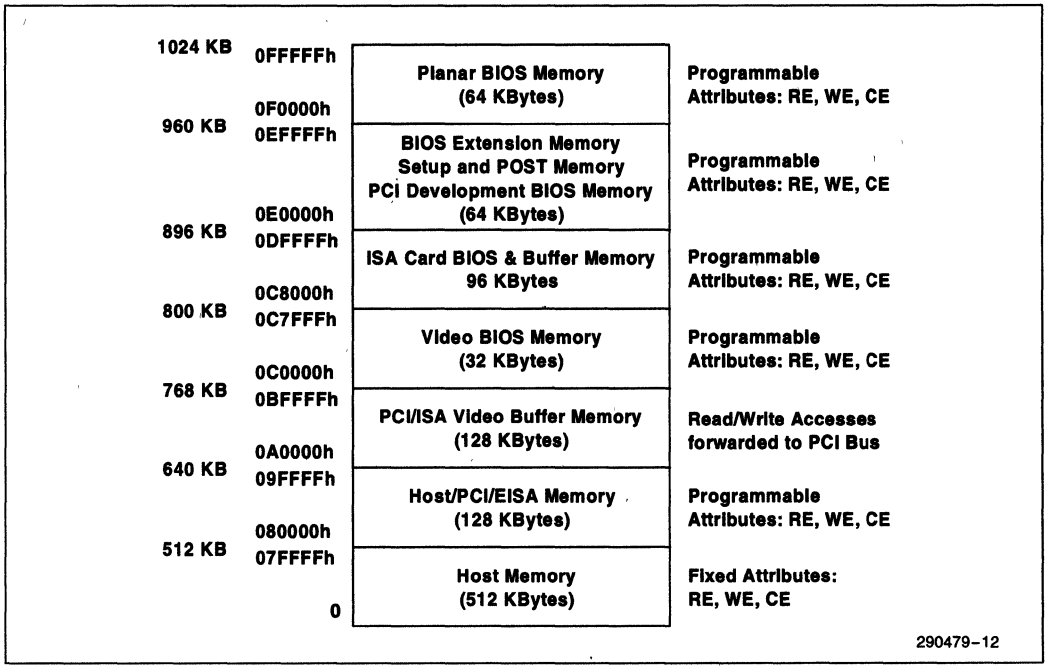
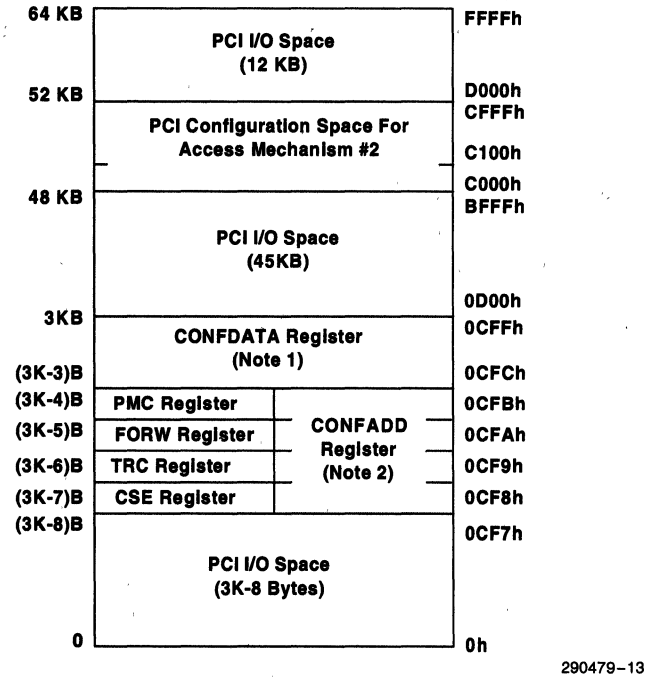


Figure 10. CPU Memory Address Map—PC Compatibility Range

The RE and WE bits for each region are used to shadow BIOS ROM in main memory for improved system performance. To shadow a BIOS area, RE is reset to 0 and WE is set to 1. RE is set to 1 and WE is reset to 0. Any writes to the BIOS area are forwarded to PCI.

4.4 I/O Address Map

I/O devices (other than the PCMC) are not supported on the Host Bus. The PCMC generates PCI Bus cycles for all CPU I/O accesses, except to the PCMC internal registers. Figure 11 shows the mapping for the CPU I/O address space. For the 82434LX, three PCMC registers are located in the CPU I/O address space—the Configuration Space Enable (CSE) Register, the Turbo-Reset Control (TRC) Register, and the Forward (FORW) Register.



NOTES:
1. This 82434NX register is only visible when configuration access mechanism #1 is enabled (via bit 31 of the CONFADD Register). Otherwise, this I/O range is in PCI I/O space.
2. This 82434NX register is accessed during Dword read/writes to 0CF8h. Byte or word cycles access the corresponding 8-bit registers, even if configuration access mechanism #1 is enabled.

Figure 11. CPU I/O Address Map

For the 82434NX, six PCMC registers are located in the CPU I/O address space—the Configuration Space Enable (CSE) Register, the Configuration Address Register (CONFADD), the Turbo-Reset Control (TRC) Register, the Forward (FORW) Register, the PCI Mechanism Control (PMC) Register, and the Configuration Data (CONFDATA) Register.

Except for the I/O locations of the above mentioned registers, all other CPU I/O accesses are mapped to either PCI I/O space or PCI configuration space. If the access is to PCI I/O space, the PCI address is the same as the CPU address. If the access is to PCI configuration space, the CPU address is mapped to a configuration space address as described in Section 3.0, Register Description.

If configuration space is enabled via the CSE Register (access mechanism #2), the PCMC maps accesses in the address range of C100h to CFFFh to PCI configuration space. Accesses to the PCMC configuration register range (C000h to C0FFh) are intercepted by the PCMC and not forwarded to PCI. If the configuration space is disabled in the CSE Register, CPU accesses to the configuration address range (C000h to CFFFh) are forwarded to PCI I/O space.

5.0 SECOND LEVEL CACHE INTERFACE

This section describes the second level cache interface for the 82434LX Cache (Section 5.1) and the 82434NX Cache (Section 5.2). The differences are in the following areas:

1. The 82434LX supports both write-through and write-back cache policies. The 82434NX only supports the write-back policy.
2. The 82434LX timings are for 60 and 66 MHz and the 82434NX timings are for 50, 60, and 66 MHz. Note that the cycle latencies for 60 and 66 MHz are the same for both devices.
3. When burst SRAMs are used to implement the secondary cache, address latches are not needed for the 82434NX type SRAM connectivity. However, a control bit has been added to the 82434NX that permits address latches for 82434LX type SRAM connectivity.
4. A low-power second level cache standby mode has been added to the 82434NX.
5. There are new or changed cache control bits as indicated by the shading in Section 3.0, Register Description. For example, the 82434NX supports zero wait-state cache at 50 MHz via the zero wait-state control bit.

NOTE:

- Second level cache sizes and organization are the same for the 82434LX and 82434NX.
- The general operation of the second level cache write-back policy is the same for the 82434LX and 82434NX. For example, the Valid and Modified bits operate the same for both devices. In addition, snoop operations are the same for both devices, as well as the handling of flush, flush acknowledge, and write-back special cycles.

5.1 82434LX Cache

The 82434LX PCMC integrates a high performance write-back/write-through second level cache controller providing integrated tags and a full first level and second level cache coherency mechanism. The second level cache controller can be configured to support either a 256-KByte cache or a 512 KByte cache using either synchronous burst SRAMs or standard asynchronous SRAMs. The cache is direct mapped and can be configured to support either a write-back or write-through write policy. Parity on the second level cache data SRAMs is optional.

The 82434LX contains 4096 address tags. Each tag represents a *sector* in the second level cache. If the second level cache is 256-KByte, each tag represents two cache lines. If the second level cache is 512-KByte, each tag represents four cache lines. Thus, in the 256-KByte configuration each sector contains two lines. In the 512-KByte configuration, each sector contains four lines. *Valid* and *modified* status bits are kept on a per line basis. Thus, in the case of a 256-KByte cache each tag has two valid bits and two modified bits associated with it. In the case of a 512-KByte cache each tag has four valid and four modified bits associated with it. Upon a CPU read cache miss, the PCMC inspects the valid and modified bits within the addressed sector and writes back to main memory only the lines marked both valid and modified. All of the lines in the sector are then invalidated. The line fill will then occur and the valid bit associated with the allocated line will be set. Only the requested line will be fetched from main memory and written into the cache. If no write-back is required, all of the lines in the sector are marked invalid. The line fill then occurs and the valid bit associated with the allocated line will be set. Lines are not allocated on write misses. When a CPU write hits a line in the second level cache, the modified bit for the line is set.

The second level cache is optional to allow the 82434LX PCMC to be used in a low cost configuration. A 256-KByte cache is implemented with a single bank of eight 32K x 9 SRAMs if parity is supported or 32K x 8 SRAMs if parity is not supported on the cache. A 512-KByte cache is implemented with four 64K x 18 SRAMs if parity is supported or 64K x 16 SRAMs if parity is not supported on the cache.

Two 74AS373 latches complete the cache. Only main memory controlled by the PCMC DRAM interface is cached. Memory on PCI is not cached.

Figure 12 and Figure 13 depict the organization of the internal tags in the PCMC configured for a 256 KByte cache and a 512-KByte cache.

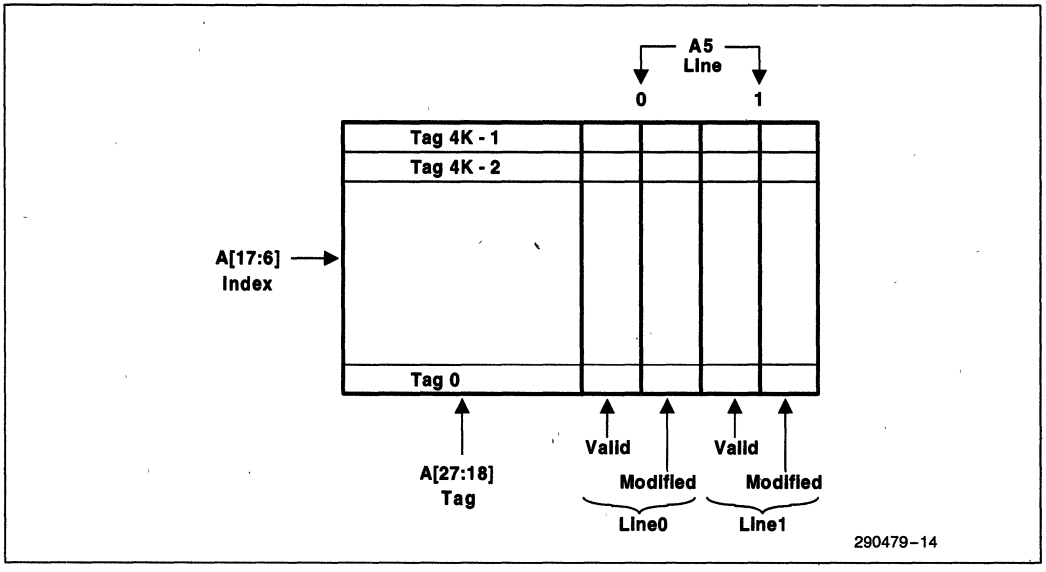


Figure 12. PCMC Internal Tags with 256-KByte Cache

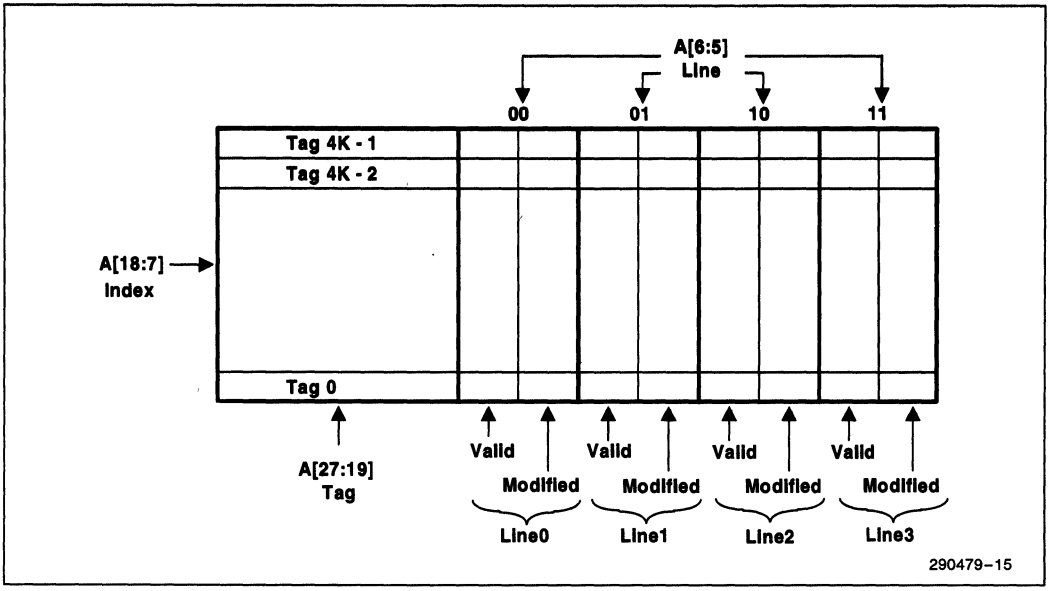


Figure 13. PCMC Internal Tags with 512-KByte Cache

In the 256-KByte cache configuration A[17:6] form the tag RAM index. The ten tag bits read from the tag RAM are compared against A[27:18] from the host address bus. Two valid bits and two modified bits are kept per tag in this configuration. Host address bit 5 is used to select between lines 0 and 1 within a sector. In the 512-KByte cache configuration A[18:7] form the tag RAM index. The nine bits read from the tag RAM are compared against A[27:19] from the host bus. Four valid bits and four modified bits are kept per tag. Host address bits 5 and 6 are used to select between lines 0, 1, 2 and 3 within a sector.

The Secondary Cache Controller Register at offset 52h in configuration space controls the secondary cache size, write and allocation policies, and SRAM type. The cache can also be enabled and disabled via this register.

Figure 14 through Figure 18 show the connections between the PCMC and the external cache data SRAMs and latches.

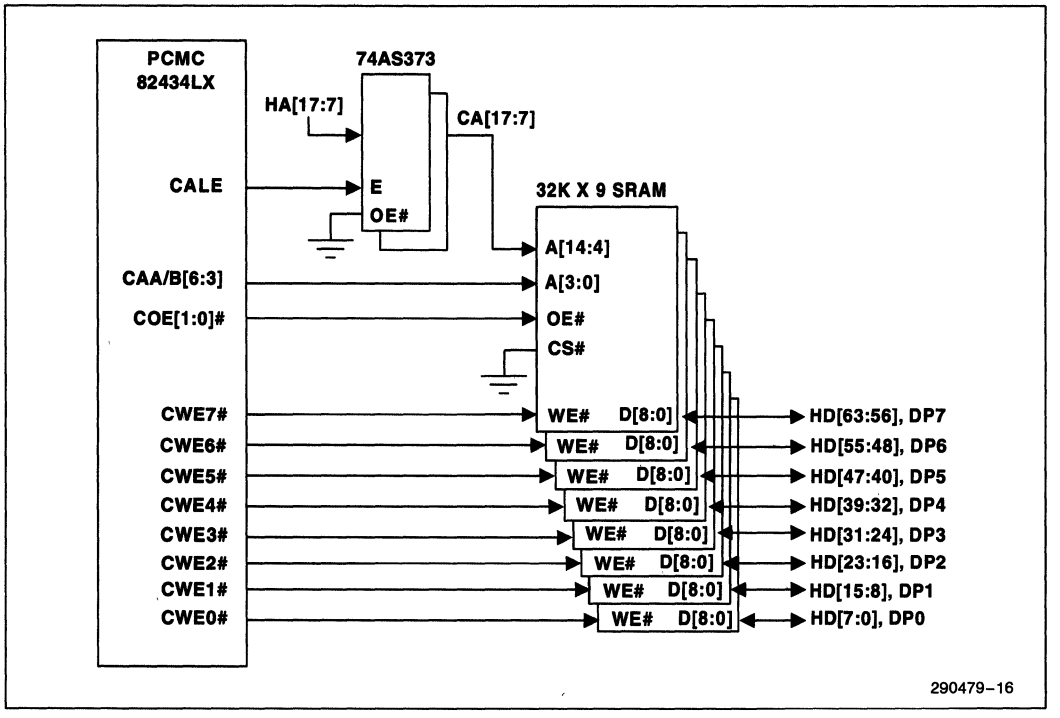


Figure 14. 82434LX Connections to 256-KByte Cache with Standard SRAM

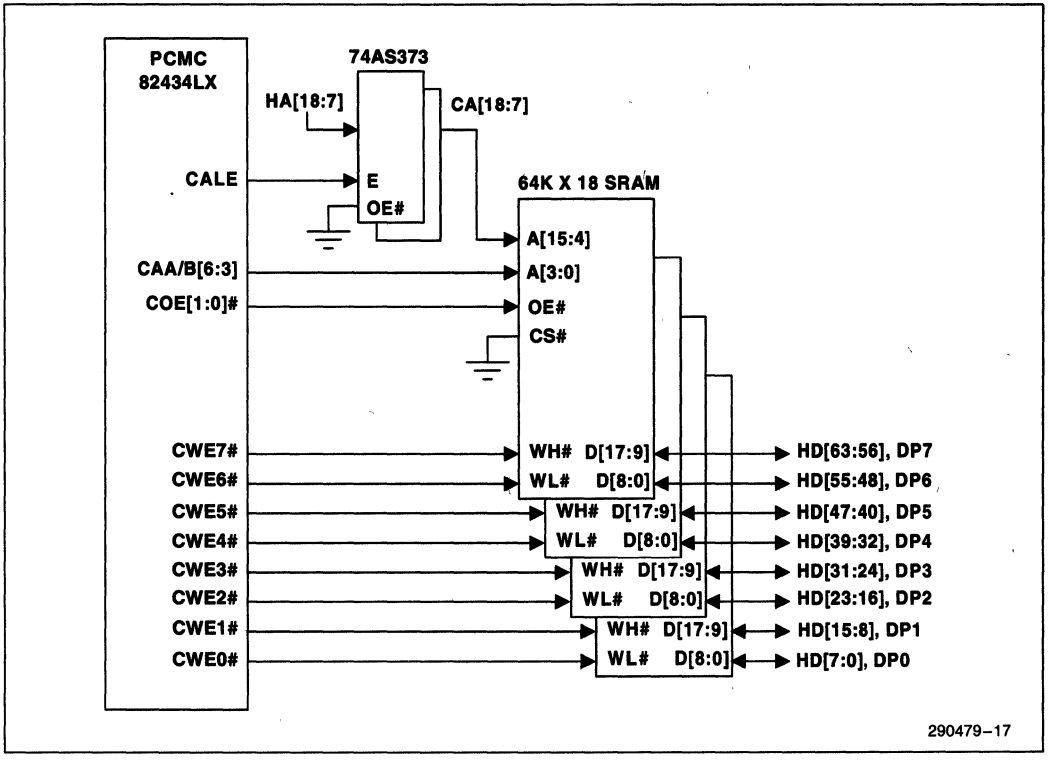


Figure 15. 82434LX Connections to 512-KByte Cache with Standard SRAM

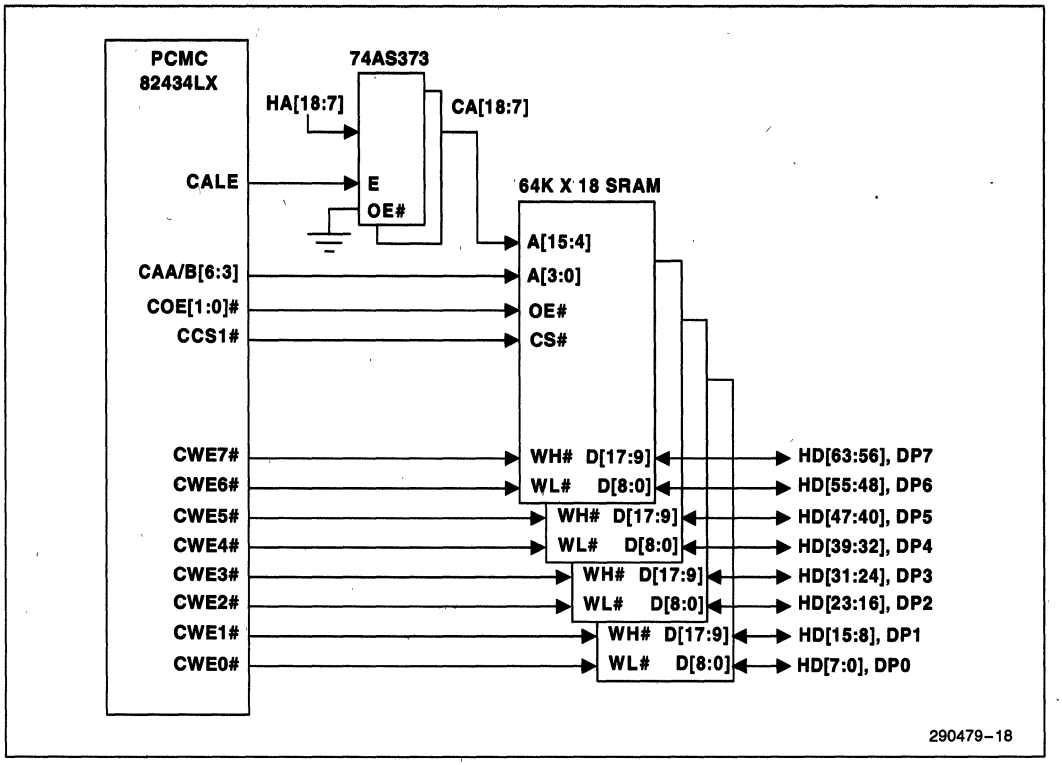


Figure 16. 82434LX Connections to 512-KByte Cache with Dual-Byte Select Standard SRAMs

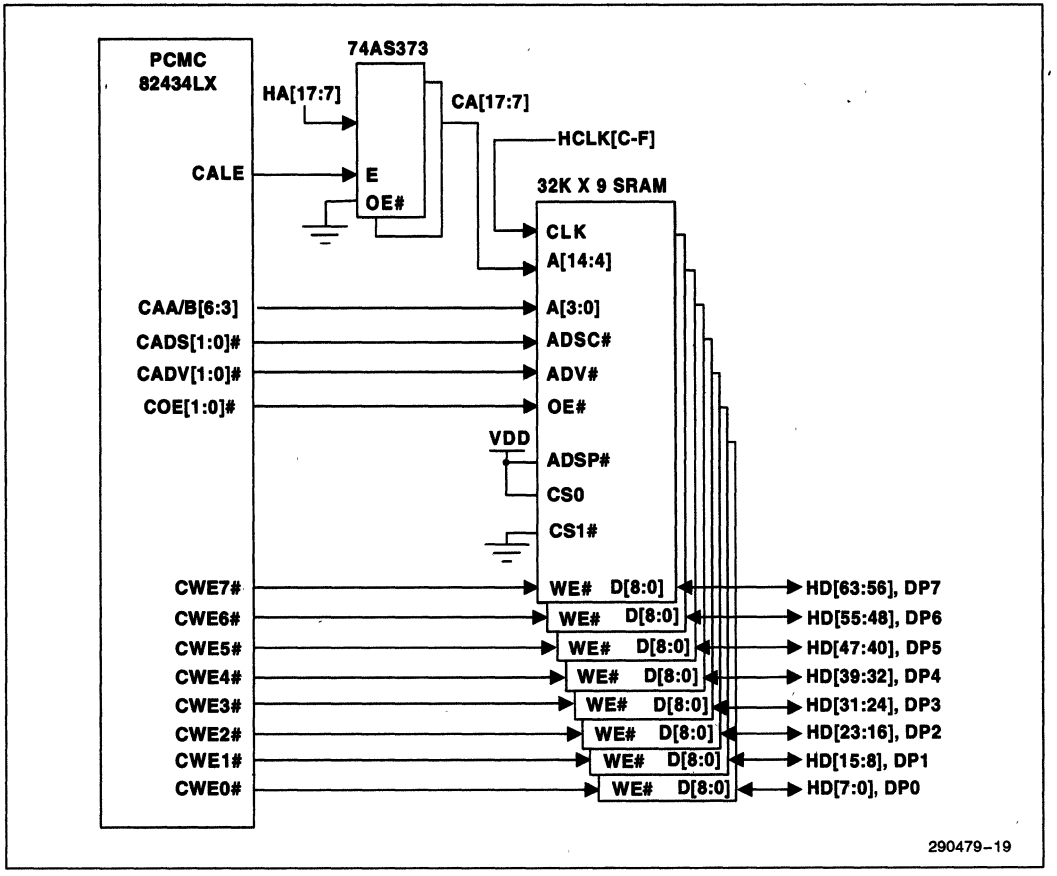


Figure 17. 82434LX Connections to 256-KByte Cache with Burst SRAM

2

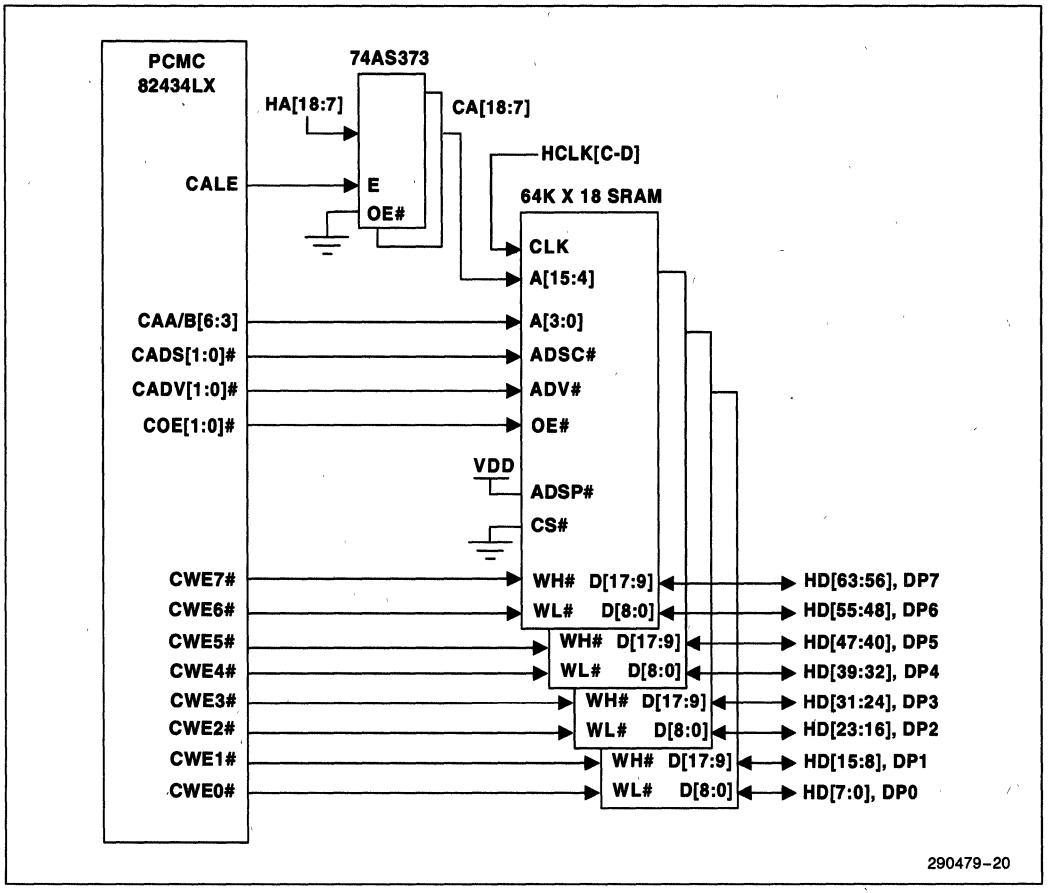


Figure 18. 82434LX Connections for 512-KByte Cache with Burst SRAM

When CALE is asserted, HA[18:7] flow through the address latch. When CALE is negated the address is captured in the latch allowing the processor to pipeline the next bus cycle onto the address bus. Two copies of CA[6:3], COE#, CADS# and CADV# are provided to reduce capacitive loading. Both copies should be used when the second level cache is implemented with eight 32K x 8 or 32K x 9 SRAMs. Either both copies or only one copy can be used with 64K x 18 or 64K x 16 SRAMs as determined by the system board layout and timing analysis. The two copies are always driven to the same logic level. CAA[4:3] and CAB[4:3] are used to count through the Pentium processor burst order when standard SRAMs are used to implement the cache.

With burst SRAMs, the address counting is provided inside the SRAMs. In this case, CAA[4:3] and CAB[4:3] are only used at the beginning of a cycle to load the initial low order address bits into the burst SRAMs. During CPU accesses, host address lines 6 and 5 are propagated to the CAA[6:5] and CAB[6:5] lines and are internally latched. When a CPU read cycle forces a line replacement in the second level cache, all modified lines within the addressed sector are written back to main memory. The PCMC uses CAA[6:5] and CAB[6:5] to select among the lines within the sector. The Cache Output Enables (COE[1:0]#) are asserted to enable the SRAMs to drive data onto the host data bus. The Cache Write Enables (CWE[7:0]#) allow byte control during CPU writes to the second level cache.

An asynchronous SRAM 512-KByte cache can be implemented with two different types of SRAM byte control. Figure 15 depicts the PCMC connections to a 512 KByte cache using 64K x 18 SRAMs or 64K x 16 SRAMs with two write enables per SRAM. Each SRAM has a high and low write enable. Figure 16 depicts the PCMC connections to a 512-KByte cache using 64K x 18 SRAMs or 64K x 16 SRAMs with two byte select lines per SRAM. Each SRAM has a high and low byte select.

The type of cache byte control (write enable or byte select) is programmed in the Cache Byte Control bit in the Secondary Cache Control Register at configuration space offset 52h. When this bit is set to 0, byte select control is used. In this mode, the CBS[7:0]# lines are multiplexed onto pins 90, 91, and 95-100 and CR/W[1:0]# pins are multiplexed onto pins 93 and 94. When this bit is set to 1, byte write enable control is used. In this mode, the CWE[7:0]# lines are multiplexed onto pins 90, 91, and 95-100. CADS[1:0]# and CADV[1:0]# are only used with burst SRAMs. The Cache Address Strobes (CADS[1:0]#) are asserted to cause the burst SRAMs to latch the cache address at the beginning of a second level cache access. CADS[1:0]# can be connected to either ADSP# or ADSC# on the SRAMs. The Cache Advance signals (CADV[1:0]#) are asserted to cause the burst SRAMs to advance to the next address of the burst sequence.

5.1.1 CLOCK LATENCIES (82434LX)

Table 5 and Table 6 list the latencies for various CPU transfers to or from the second level cache for standard SRAMs and burst SRAMs. Standard SRAM access times of 12 ns and 15 ns are recommended for 66 MHz and 60 MHz operation, respectively. Burst SRAM clock access times of 8 ns and 9 ns are recommended for 66 MHz and 60 MHz operation, respectively. Precise SRAM timing requirements should be determined by system board electrical simulation with SRAM I/O buffer models.

Table 5. Second Level Cache Latencies with Standard SRAM (82434LX)

Cycle Type	HCLK Count
Burst Read	3-2-2-2
Burst Write	4-2-2-2
Single Read	3
Single Write	4
Pipelined Back to Back Burst Reads	3-2-2-2/3-2-2-2
Burst Read followed by Pipelined Write	3-2-2-2/4

Table 6. Second Level Cache Latencies with Burst SRAM (82434LX)

Cycle Type	HCLK Count
Burst Read	3-1-1-1
Burst Write	3-1-1-1
Single Read	3
Single Write	3
Pipelined Back to Back Burst Reads	3-1-1-1/1-1-1-1
Read Followed by Pipelined Write	3-1-1-1/2

5.1.2 STANDARD SRAM CACHE CYCLES (82434LX)

The following sections describe the activity of the second level cache interface when standard asynchronous SRAMs are used to implement the cache.

5.1.2.1 Burst Read (82434LX)

Figure 19 depicts a burst read from the second level cache with standard SRAMs. The CPU initiates the read cycle by driving address and status onto the bus and asserting ADS#. Initially, the CA[6:3] are a propagation delay from the host address lines A[6:3]. Upon sampling W/R# active and M/IO# inactive, while ADS# is asserted, the PCMC asserts COE# to begin a read cycle from the SRAMs. CALE is negated, latching the address lines on the SRAM address inputs, allowing the CPU to pipeline a new address onto the bus. CA[4:3] cycle through the Pentium processor burst order, completing the cycle. PEN# is asserted with the first BRDY# and negated with the last BRDY# if parity is implemented on the second level cache data SRAMs and the MCHK DRAM/Second Level Cache Data Parity bit in the Error Command Register (offset 70h) is set.

Figure 20 depicts a burst read from the second level cache with standard 16- or 18-bit wide dual-byte select SRAMs. A single read cycle from the second level cache is very similar to the first transfer of a burst read cycle. CALE is not negated throughout the cycle. COE# is asserted as shown above, but is negated with BRDY#.

When the Secondary Cache Allocation (SCA) bit in the Secondary Cache Control Register is set to 1, the PCMC performs a line fill in the secondary cache, even if the CACHE# signal from the CPU is inactive. In this case, AHOLD is asserted to prevent the CPU from beginning a new cycle while the second level cache line fill is completing.

Back-to-back pipelined burst reads from the second level cache are shown in the Figure 21.

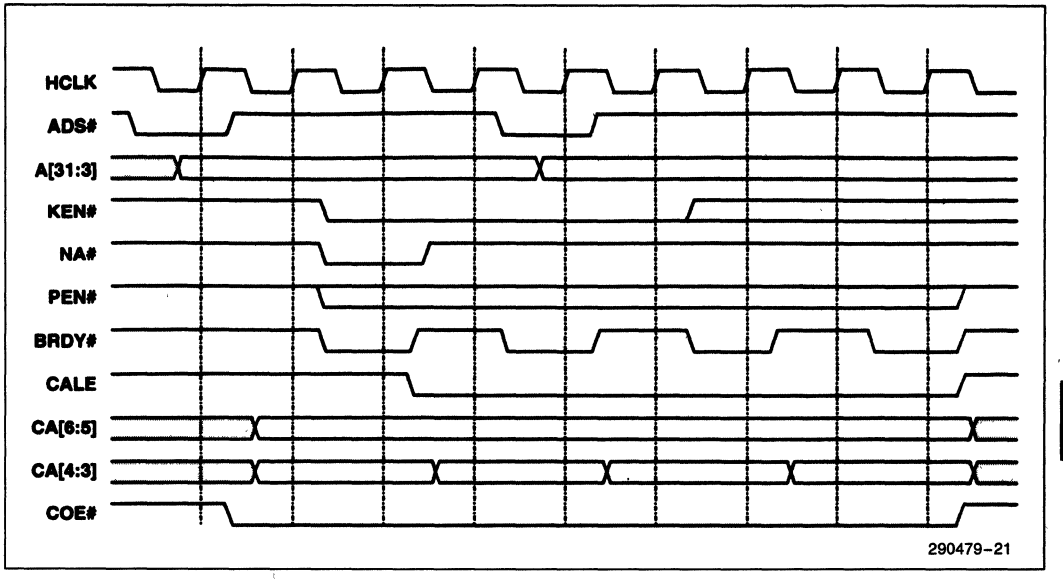


Figure 19. CPU Burst Read from Second Level Cache with Standard SRAM (82434LX)

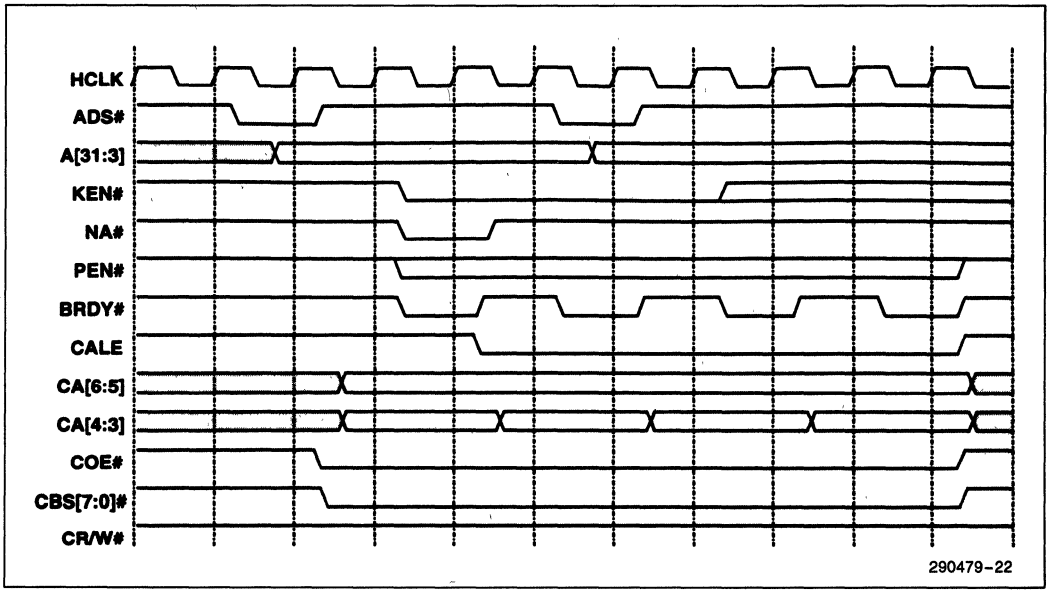


Figure 20. Burst Read from Second Level Cache with Dual-Byte Select SRAMs (82434LX)

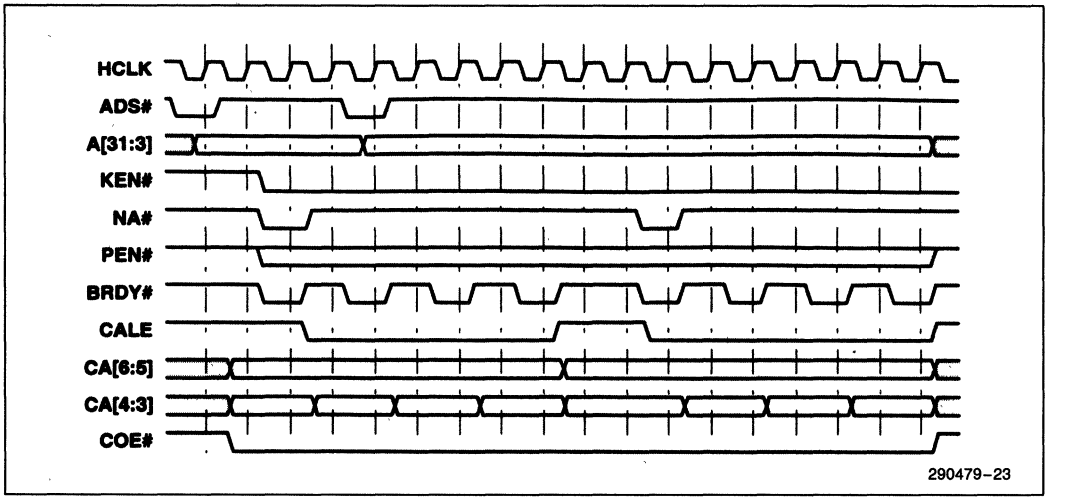


Figure 21. Pipelined Back-to-Back Burst Reads from Second Level Cache with Standard SRAM (82434LX)

Due to assertion of NA#, the CPU drives a new address onto the bus before the first cycle is complete. In this case, the second cycle is a hit in the second level cache. Immediately upon completion of the first read cycle, the PCMC begins the second cycle. When the first cycle completes, the PCMC drives the new address to the SRAMs on CA[6:3] and asserts CALE. The second cycle is very similar to the first, completing at a rate of 3-2-2-2. The cache address lines must be held at the SRAM address inputs until the first cycle completes. Only after the last BRDY# is returned, can CALE be asserted and CA[6:3] be changed. Thus, the pipelined cycle completes at the same rate as a non-pipelined cycle.

5.1.2.2 Burst Write (82434LX)

A burst write cycle is used to write back a cache line from the first level cache to either the second level cache or DRAM. Figure 22 depicts a burst write cycle to the second level cache with standard SRAMs. The CPU initiates the write cycle by driving address and status onto the bus and asserting ADS#. Initially, the CA[6:3] propagate from the host address lines A[6:3]. CALE is negated, latching the address lines on the SRAM address inputs, allowing the CPU to pipeline a new address onto the bus. Burst write cycles from the Pentium processor always begin with the low order Qword and advances to the high order Qword. CWE[7:0]# are generated from an internally delayed version of HCLK, providing address setup time to CWE[7:0]# falling and data setup time to CWE[7:0]# rising edges. HIG[4:0] are driven to PCMWQ (Post CPU to Memory Write Buffer Qword) only when the PCMC is programmed for a write-through write policy. When programmed for write-back mode, the modified bit associated with the line is set within the PCMC. The single write cycle is very similar to the first write of a burst write cycle. A burst read cycle followed by a pipelined write cycle with standard SRAMs is depicted in Figure 24.

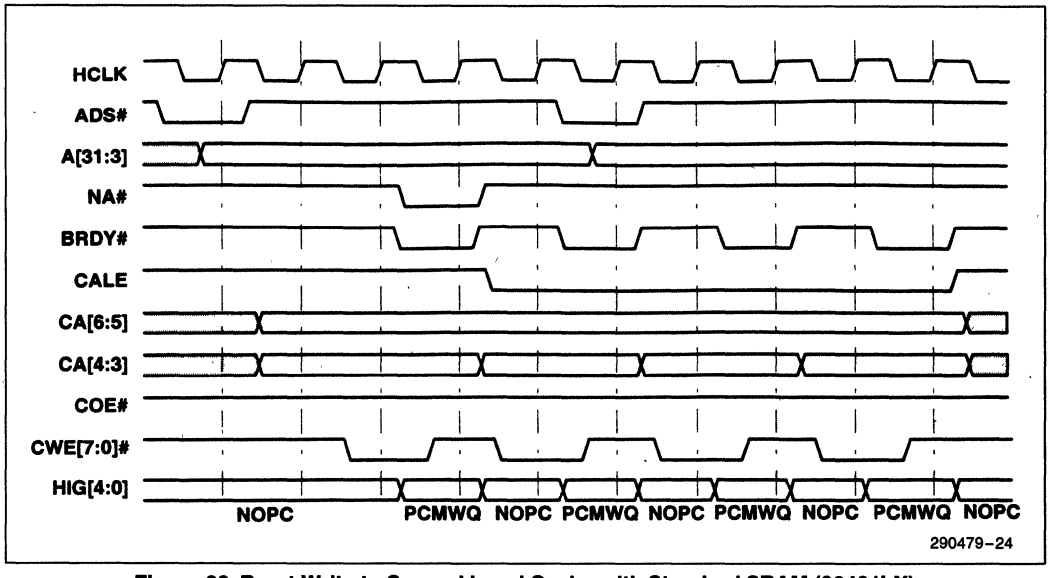


Figure 22. Burst Write to Second Level Cache with Standard SRAM (82434LX)

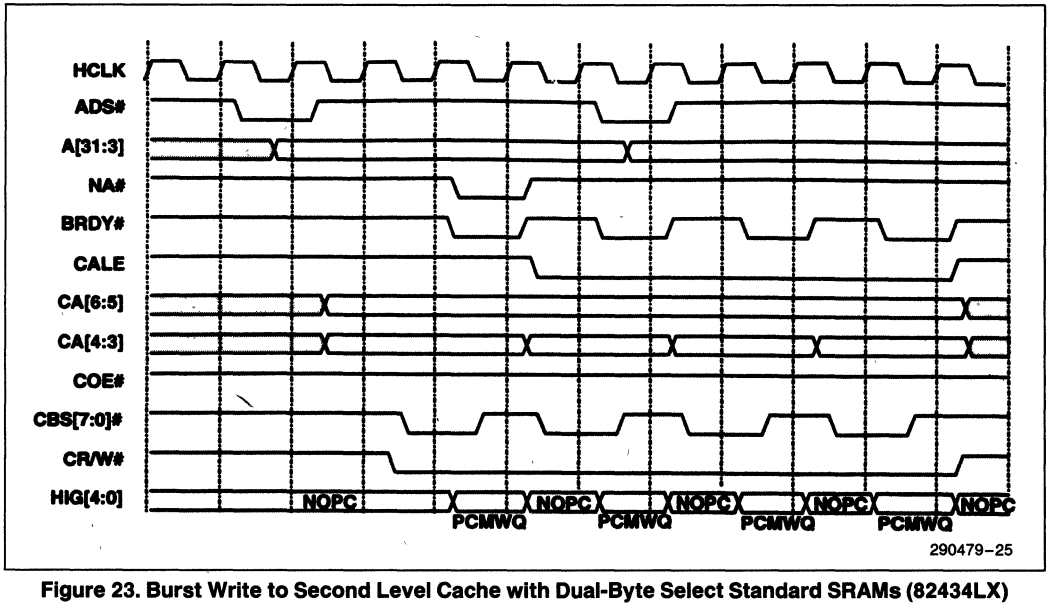


Figure 23. Burst Write to Second Level Cache with Dual-Byte Select Standard SRAMs (82434LX)

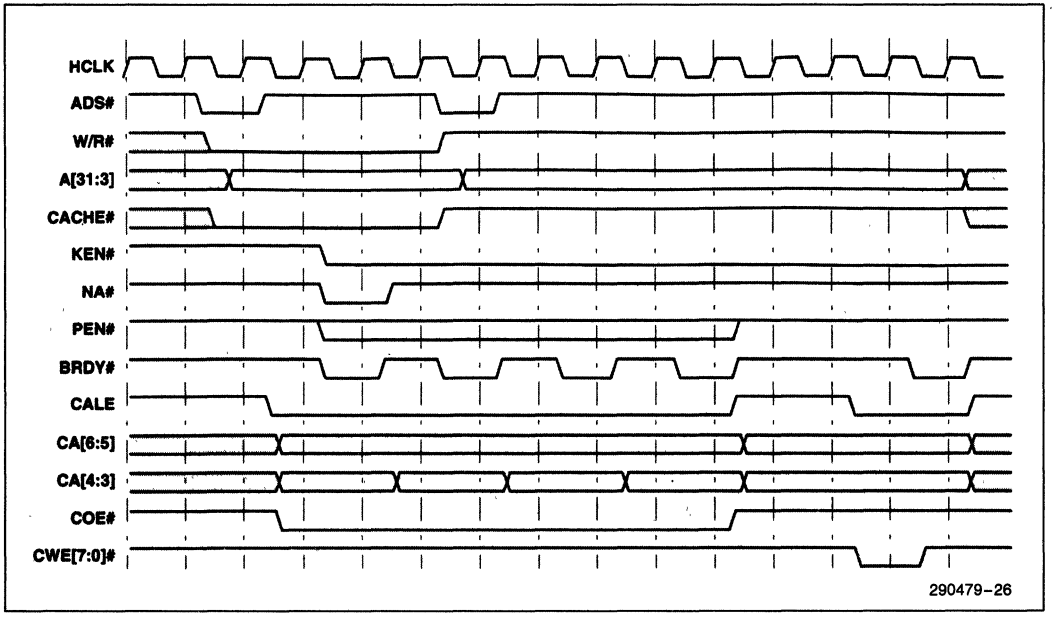


Figure 24. Burst Read Followed by Pipelined Write with Standard SRAM (82434LX)

5.1.2.3 Cache Line Fill (82434LX)

If the CPU issues a memory read cycle to cacheable memory that is not in the second level cache, a first and second level cache line fill occurs. Figure 25 depicts a CPU read cycle that results in a line fill into the first and second level caches.

Figure 27 depicts the host bus activity during a CPU read cycle that forces a write-back from the second level cache to the CPU-to-memory posted write buffer as the DRAM read cycle begins.

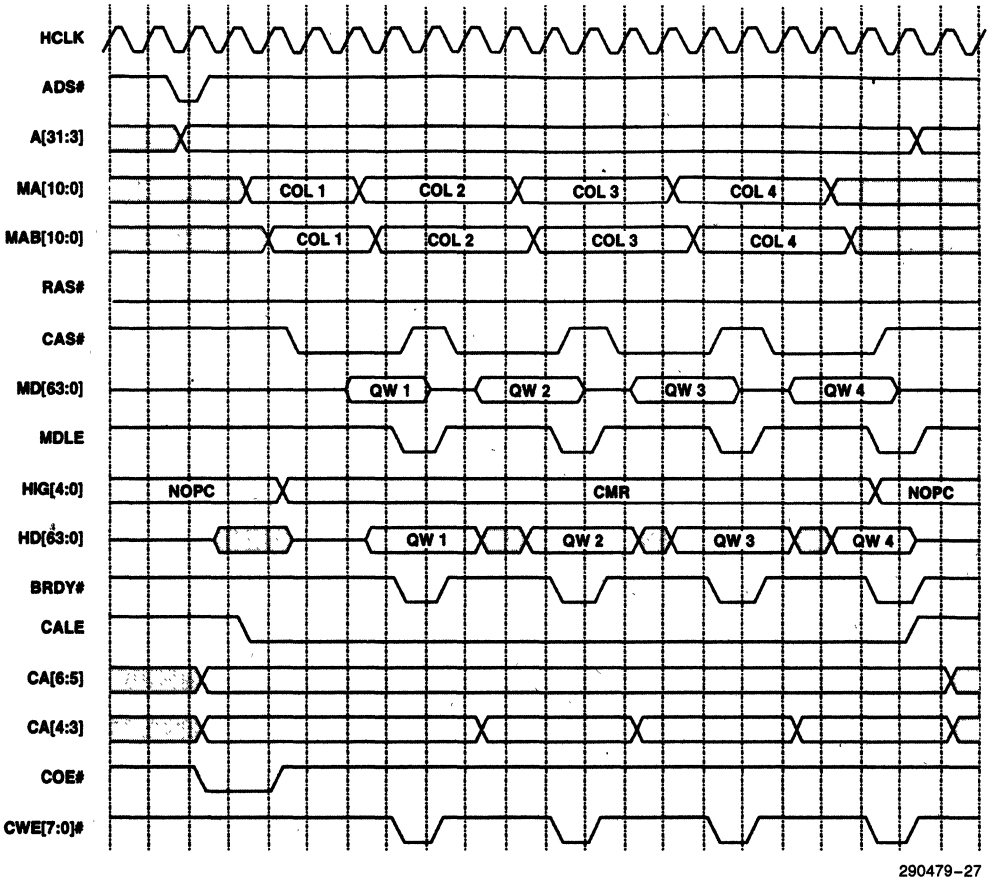


Figure 25. Cache Line Fill with Standard SRAM, DRAM Page Hit (82434LX)

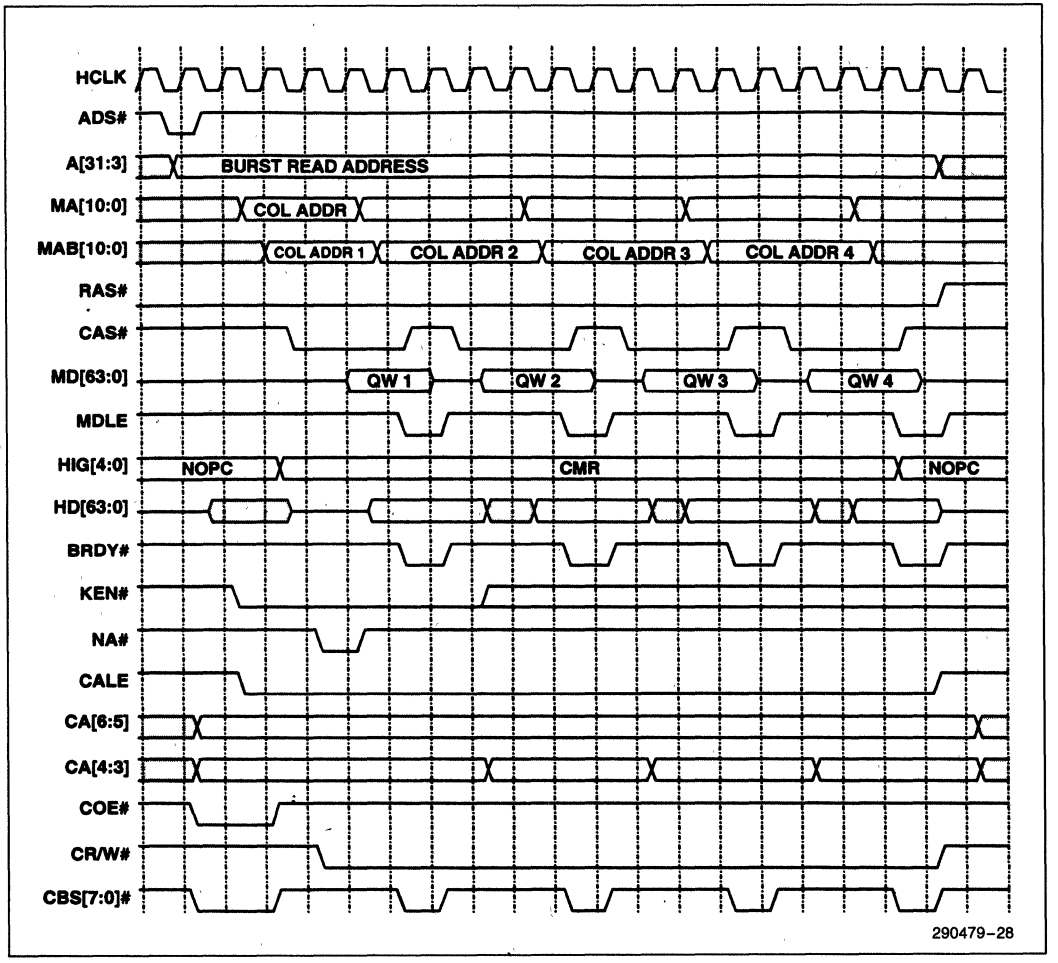


Figure 26. Cache Line Fill with Dual-Byte Select Standard SRAM, DRAM Page Hit (82434LX)

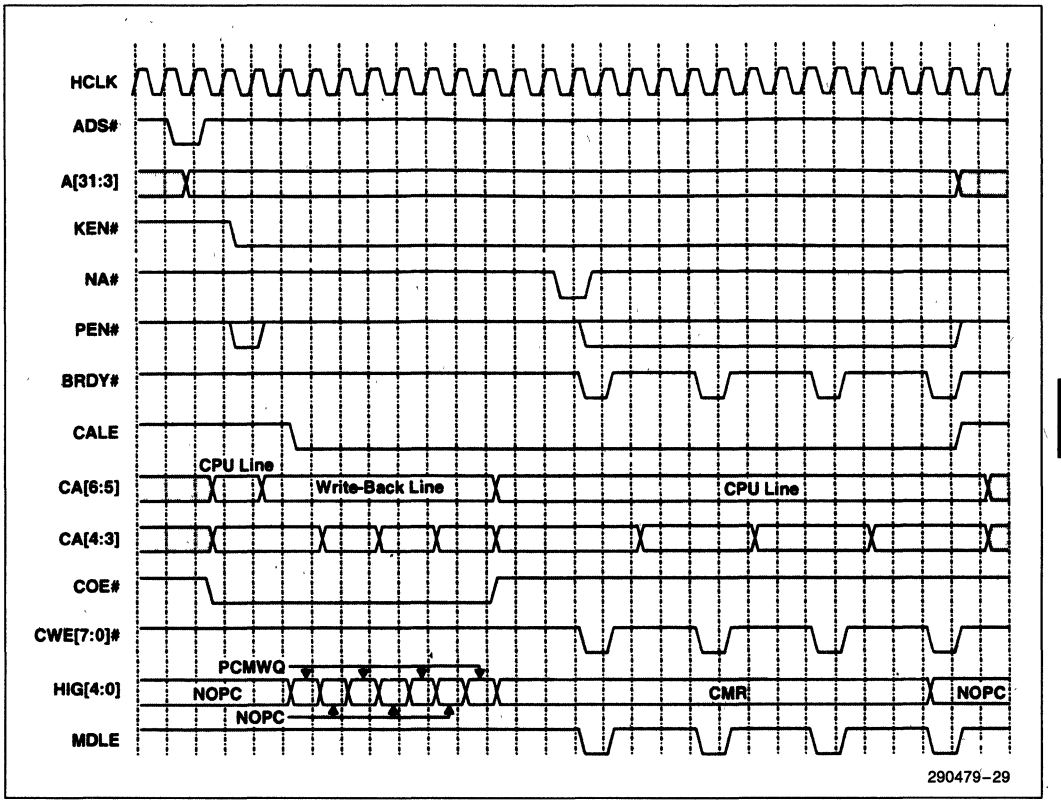


Figure 27. CPU Cache Read Miss, Write-Back, Line Fill with Standard SRAM (82434LX)

The CPU issues a memory read cycle that misses in the second level cache. In this instance, a modified line in the second level cache must be written back to main memory before the new line can be filled into the cache. The PCMC inspects the valid and modified bits for each of the lines within the addressed sector and writes back only the valid lines within the sector that are in the modified state. During the write-back cycle, CA[4:3] begin with the initial value driven by the Pentium processor and proceed in the Pentium processor burst order. CA[6:5] are used to count through the lines within the addressed sector. When two or more lines must be written back to main memory, CA[6:5] count in the direction from line 0 to line 3. CA[6:5] advance to the next line to be written back to main memory, skipping lines that are not modified. Figure 23 depicts the case of just one of the lines in a sector being written back to main memory. In this case, the entire line can be posted in the CPU-to-Main memory posted write buffer by driving the HIG[4:0] lines to the PCMWQ command as each Qword is read from the cache. At the same time, the required DRAM read cycle is beginning. As soon as the de-allocated line is written into the posted write buffer, the HIG[4:0] lines are driven to CMR (CPU Memory Read) to allow data to propagate from the DRAM data lines to the CPU data lines. The CWE[7:0]# lines are not generated from a delayed version of HCLK (as they are in the case of CPU to second level cache burst write), but from ordinary HCLK rising edges. CMR is driven on the HIG[4:0] lines

throughout the DRAM read portion of the cycle. With the fourth assertion of BRDY# the HIG[4:0] lines change to NOPC. The LBXs however, do not tri-state the host data lines until MDLE rises. CWE[7:0]# and MDLE track such that MDLE will not rise before CWE[7:0]#. Thus, the LBXs continue to drive the host data lines until CWE[7:0]# are negated. CA[6:3] remain at the valid values until the clock after the last BRDY#, providing address hold time to CWE[7:0]# rising.

PEN# is asserted as shown if the MCHK DRAM/L2 Cache Data Parity Error bit in the Error Command Register (offset 70h) is set. If the second level cache supports parity, PEN# is always asserted during CPU read cycles in the third clock in case the cycle hits in the cache.

If more than one line must be written back to main memory, the PCMC fills the CPU-to-Main Memory Posted Write Buffer and loads another Qword into the buffer as each Qword write completes into main memory. The writes into DRAM proceed as page hit write cycles from one line to the next, completing at a rate of X-4-4-4-5-4-4-4-5-4-4-4 for a three line write-back. All modified lines except for the last one to be written back are posted and written to memory before the DRAM read cycle begins. The last line to be written back is posted as the DRAM read cycle begins. Thus, the read data is returned to the CPU before the last line is retired to memory.

The line which was written into the second level cache is marked valid and unmodified by the PCMC. All the other lines in the sector are marked invalid. A subsequent CPU read cycle which hits in the same sector (but a different line) in the second level cache would then simply result in a line fill without any write-back.

5.1.3 BURST SRAM CACHE CYCLES (82434LX)

The following sections show the activity of the second level cache interface when burst SRAMs are used for the second level cache.

5.1.3.1 Burst Read (82434LX)

Figure 28 depicts a burst read from the second level cache with burst SRAMs.

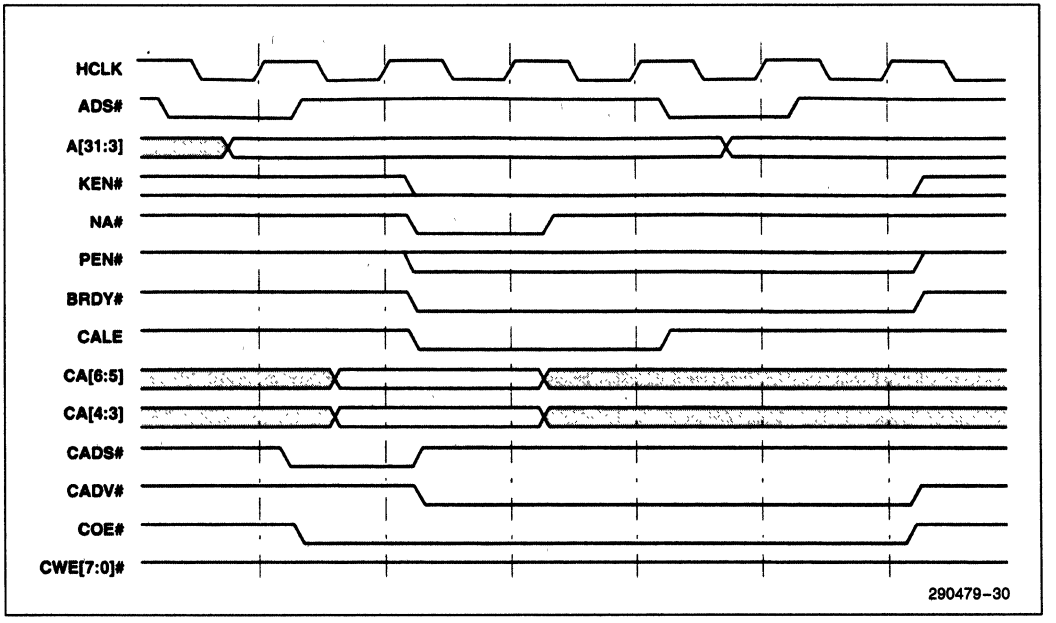


Figure 28. CPU Burst Read from Second Level Cache with Burst SRAM (82434LX)

The cycle begins with the CPU driving address and status onto Host Bus and asserting ADS#. The PCMC asserts CADS# and COE# in the second clock. After the address is latched by the burst SRAMs and the PCMC determines that no write-back cycles are required from the second level cache, CALE is negated. Back-to-back burst reads from the second level cache are shown in Figure 29.

When the Secondary Cache Allocation (SCA) bit in the Secondary Cache Control Register is set to 1, the PCMC performs a line fill in the secondary cache, even if the CACHE# signal from the CPU is negated. In this case, AHOLD is asserted to prevent the CPU from beginning a new cycle while the second level cache line fill is completing.

Back-to-back burst reads which hit in the second level cache complete at a rate of 3-1-1-1/1-1-1-1 with burst SRAMs. As the last BRDY# is being returned to the CPU, the PCMC asserts CADS# causing the SRAMs to latch the new address. This allows the data for the second cycle to be transferred to the CPU on the clock after the first cycle completes.

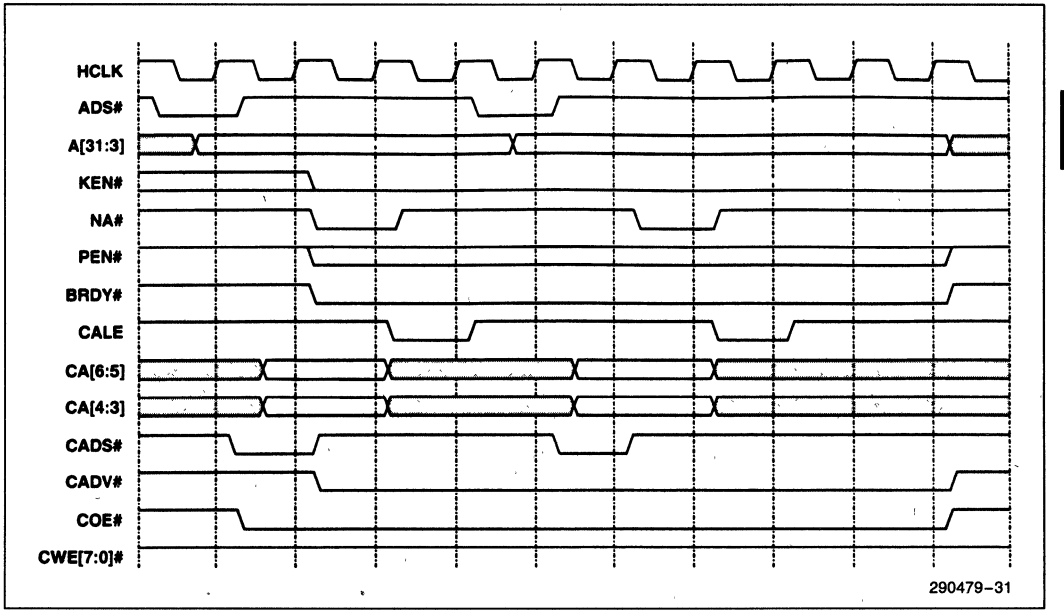


Figure 29. Pipelined Back-to-Back Burst Reads from Second Level Cache (82434LX)

5.1.3.2 Burst Write (82434LX)

A burst write cycle is used to write back a line from the first level cache to either the second level cache or DRAM. A burst write cycle from the first level cache to the second level cache is shown in Figure 30.

The Pentium processor always writes back lines starting with the low order Qword advancing to the high order Qword. CADS# is asserted in the second clock. CWE[7:0]# and BRDY# are asserted in the third clock. CADV# assertion is delayed by one clock relative to the burst read cycle. HIG[4:0] are driven to PCMWQ (Post CPU-to-Memory Write Buffer Qword) only when the PCMC is programmed for a write-through write policy. When programmed for write-back mode, the modified bit associated with the line is set within the PCMC. The single write is very similar to the first write in a burst write. CADS# is asserted in the second clock. BRDY# and CWE[7:0]# are asserted in the third clock. A burst read cycle followed by a pipelined single write cycle is depicted in Figure 31.

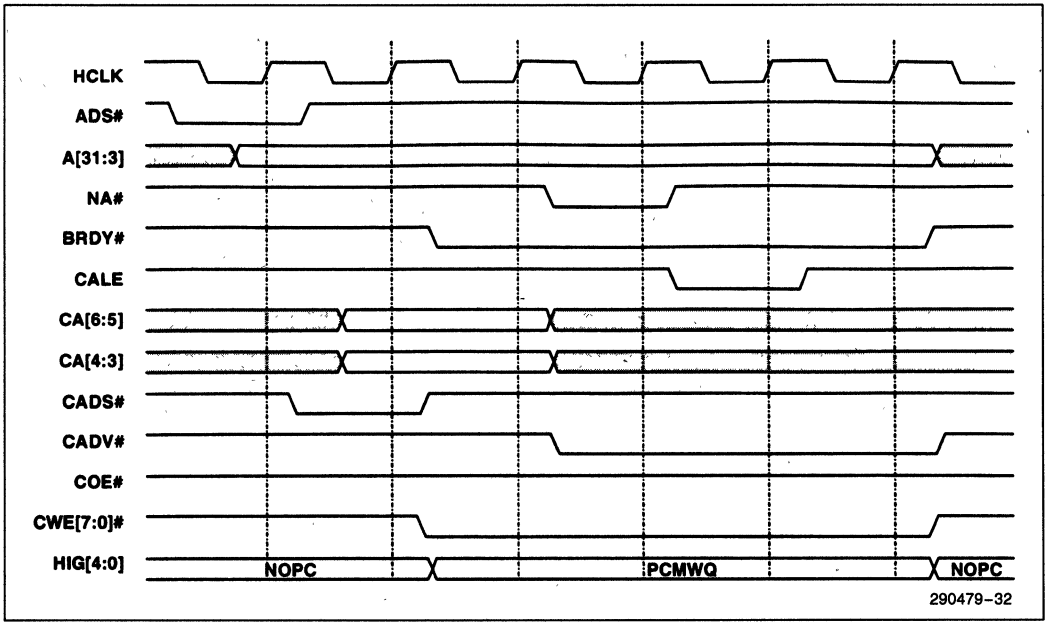


Figure 30. Burst Write to Second Level Cache with Burst SRAM (82434LX)

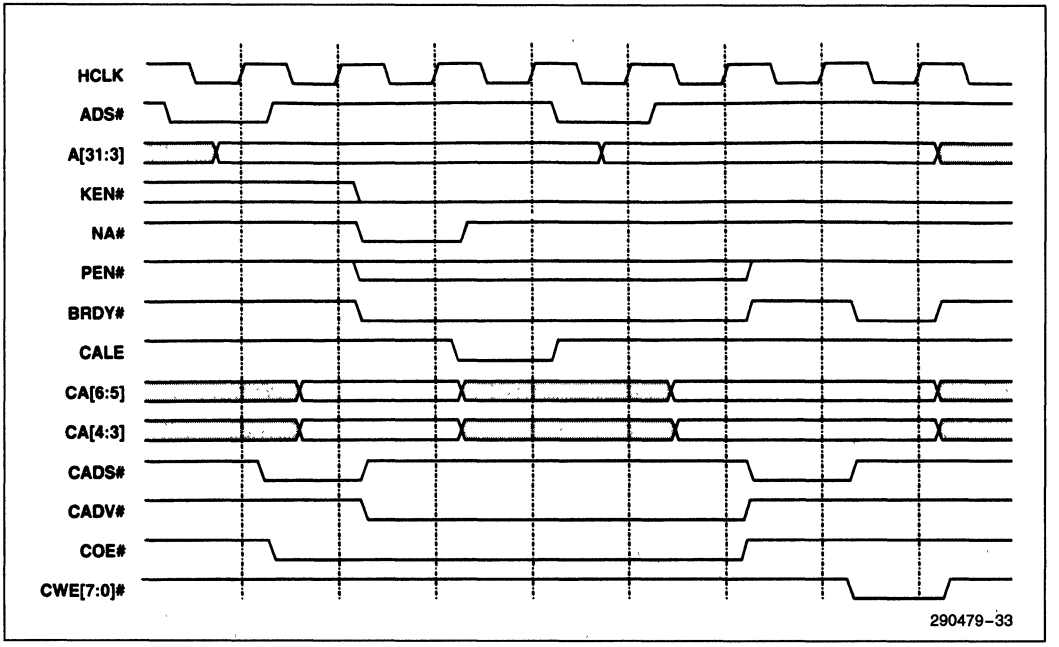


Figure 31. Burst Read Followed by Pipelined Single Write Cycle with Burst SRAM (82434LX)

5.1.3.3 Cache Line Fill (82434LX)

If the CPU issues a memory read cycle to cacheable memory which does not hit in the second level cache, a cache line fill occurs. Figure 32 depicts a first and second level cache line fill with burst SRAMs.

Figure 33 depicts a CPU read cycle which forces a write-back in the second level cache.

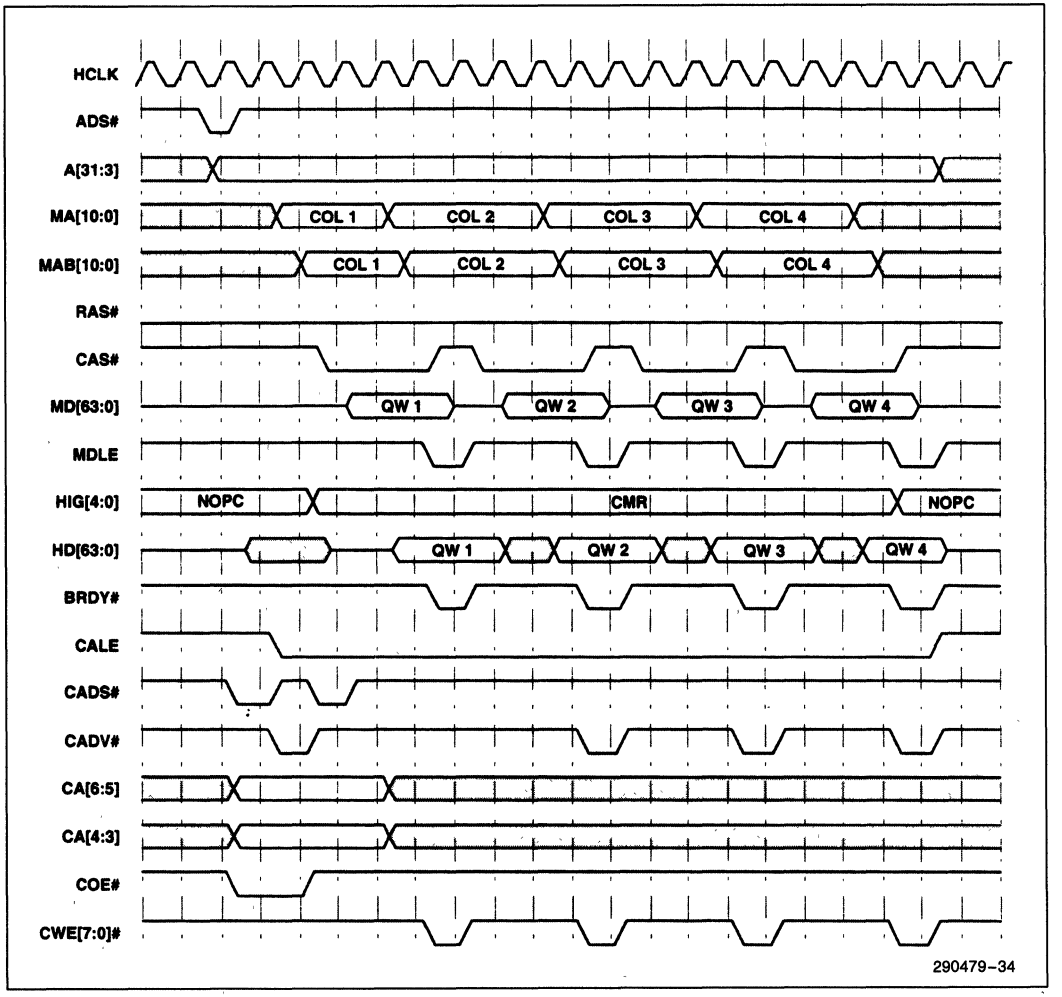


Figure 32. Cache Line Fill with Burst SRAM, DRAM Page Hit, 7-4-4-4 Timing (82434LX)

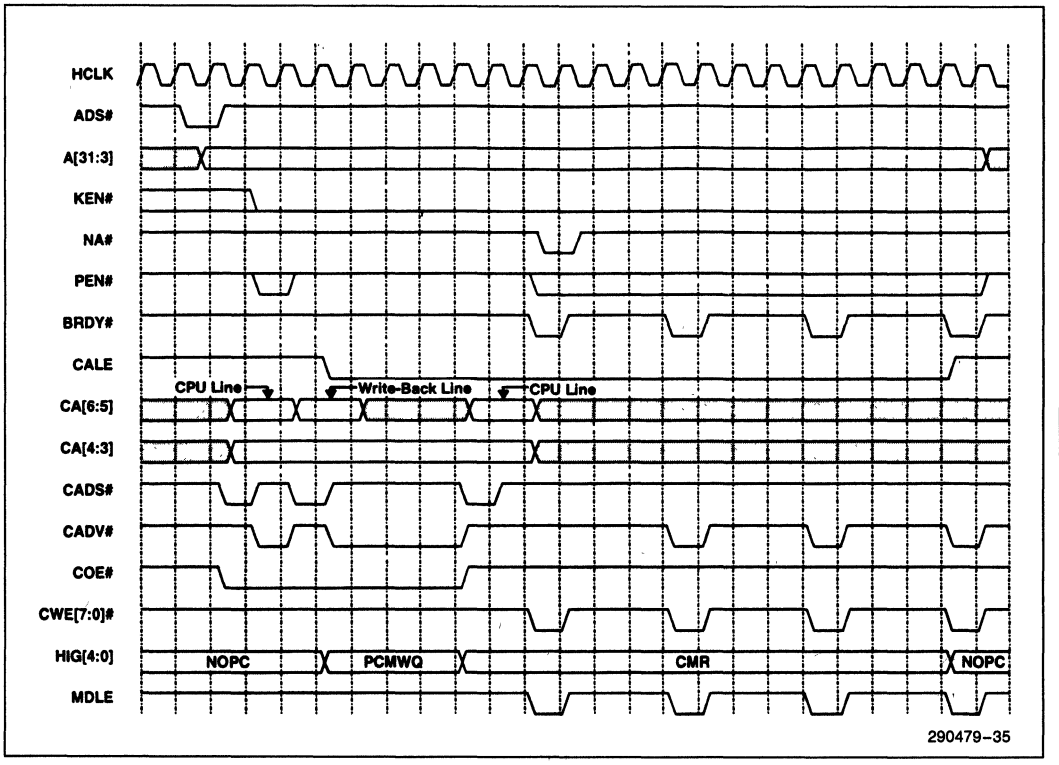


Figure 33. CPU Cache Read Miss, Write-Back, Line Fill with Burst SRAM (82434LX)

The CPU issues a memory read cycle which misses in the second level cache. In this instance, a modified line in the second level cache must be written back to main memory before the new line can be filled into the cache. The PCMC inspects the valid and modified bits for each of the lines within the addressed sector and writes back only the valid lines within the sector that are marked modified. CA[6:5] are used to count through the lines within the addressed sector. When two or more lines must be written back to main memory, CA[6:5] count in the direction from line 0 to line 3 after each line is written back. Figure 29 depicts the case of just one

of the lines in a sector being written back to main memory. In this case, the entire line can be posted in the CPU-to-Memory Posted Write Buffer by driving the HIG[4:0] lines to PCMWQ as each Qword is read from the cache. At the same time, the required DRAM read cycle is beginning. After the de-allocated line is written into the posted write buffer, the HIG[4:0] lines are driven to CMR (CPU Memory Read) to allow data to propagate from the DRAM data lines to the CPU data lines. Figure 29 assumes that the read from DRAM is a page hit and thus the first Qword is already read from the DRAMs when the transfer from cache to the CPU to Memory posting buffer is complete. The rest of the DRAM cycle completes at a -4-4-4 rate. CADV# is asserted with the last three BRDY# assertions. CMR is driven on the HIG[4:0] lines throughout the DRAM read portion of the cycle. Upon the fourth assertion of BRDY# the HIG[4:0] lines change to NOPC.

PEN# is asserted as shown if the MCHK DRAM/L2 Cache Data Parity Error bit in the Error Command Register (offset 70h) is set. If the second level cache supports parity, PEN# is always asserted during CPU read cycles in clock 3 in case the cycle hits in the cache.

If more than one line must be written back to main memory, the PCMC fills the CPU-to-Main Memory Posted Write Buffer and loads another Qword into the buffer as each Qword write completes into main memory. The writes into DRAM proceed as page hit write cycles from one line to the next, completing at a rate of X-4-4-4-5-4-4-4-5-4-4-4 for a three line write-back when programmed for X-4-4-4 DRAM write timing or X-3-3-3-4-3-3-3-4-3-3-3 when programmed for X-3-3-3 DRAM write timing. All modified lines except for the last one to be written back to memory are posted and retired to memory before the DRAM read cycle begins. The last line to be written back is posted as the DRAM read cycle begins. Thus, the read data is returned to the CPU before the last line is retired to memory.

The line which was written into the second level cache is marked valid and unmodified by the PCMC. All the other lines in the block are marked invalid. A subsequent CPU read cycle which hits the same sector (but a different line) in the second level cache results in a line fill without any write-back.

5.1.4 SNOOP CYCLES

Snoop cycles are the same for the 82434LX and 82434NX. The inquire cycle is used to probe the first level and second level caches when a PCI master attempts to access main memory. This is done to maintain coherency between the first and second level caches and main memory. When a PCI master first attempts to access main memory a snoop request is generated inside the PCMC. The PCMC supports up to two outstanding cycles on the CPU address bus at a time. Outstanding cycles include both CPU initiated cycles and snoop cycles. Thus, if the Pentium processor pipelines a second cycle onto the host address bus, the PCMC will not issue a snoop cycle until the first CPU cycle terminates. If the PCMC were to initiate a snoop cycle before the first CPU cycle were complete then for a brief period of time, three cycles would be outstanding. Thus, a snoop request is serviced with a snoop cycle only when either no cycle is outstanding on the CPU bus or one cycle is outstanding.

Snoop cycles are performed by driving the PCI master address onto the CPU address bus and asserting EADS#. The Pentium processor then performs a tag lookup to determine if the addressed memory is in the first level cache. At the same time the PCMC performs an internal tag lookup to determine if the addressed memory is in the second level cache. Table 7 describes how a PCI master read from main memory is serviced by the PCMC.

Table 7. Data Transfers for PCI Master Reads from Main Memory

Snoop Result		Action
First Level Cache	Second Level Cache	
Miss	Miss	Data is transferred from DRAM to PCI.
Miss	Hit Unmodified Line	Data is transferred directly from second level cache to PCI. The line remains valid and unmodified in the second level cache.
Miss	Hit Modified Line	Data is transferred directly from second level cache to PCI. Line remains valid and modified in the second level cache. The line is not written to DRAM.
Hit Unmodified Line	Miss	Data is transferred from DRAM to PCI.
Hit Unmodified Line	Hit Unmodified Line	Data is transferred directly from second level cache to PCI. The line remains valid and unmodified in the second level cache.
Hit Unmodified Line	Hit Modified Line	Data is transferred directly from second level cache to PCI. Line remains valid and modified in the second level cache. The line is not written to DRAM.
Hit Modified Line	Miss	A write-back from first level cache occurs. The data is sent to both PCI and the CPU-to-Memory Posted Write Buffer. The CPU-to-Memory Posted Write Buffer is then written to memory.
Hit Modified Line	Hit Unmodified Line	A write-back from first level cache occurs. The data is posted to PCI and written into the second level cache. When the second level cache is in write-back mode, the line is marked modified and is not written to DRAM. When the second level cache is in write-through mode, the line is posted and then written to DRAM.
Hit Modified Line	Hit Modified Line	A write-back from first level cache occurs. The data is posted to PCI and written into the second level cache. The line is not written to DRAM. This scenario can only occur when the second level cache is in write-back mode.

PCI master write cycles never result in a write directly into the second level cache. A snoop hit to a modified line in either the first level or second level cache results in a write-back of the line to main memory. The line is invalidated and the PCI write to main memory occurs after the write-back completes. The other lines in the sector are not written back to main memory or invalidated. A PCI master write snoop hit to an unmodified line in either the first level or second level cache results in the line being invalidated. Table 8 describes the actions taken by the PCMC when a PCI master writes to main memory.

Table 8. Data Transfers for PCI Master Writes to Main Memory

Snoop Result		Action
First Level Cache	Second Level Cache	
Miss	Miss	The PCI master write data is transferred from PCI to DRAM.
Miss	Hit Unmodified Line	The PCI master write data is transferred from PCI to DRAM. The line is invalidated in the second level cache.
Miss	Hit Modified Line	A write-back from second level cache to DRAM occurs. The PCI master write data is then written to DRAM. The line is invalidated in the second level cache.
Hit Unmodified Line	Miss	The first level cache line is invalidated. The PCI master write data is written to DRAM.
Hit Unmodified Line	Hit Unmodified Line	The line is invalidated in both the first level and second level caches. The PCI master write data is written to DRAM.
Hit Unmodified Line	Hit Modified Line	The first level cache line is invalidated. The second level cache line is written back to main memory and invalidated. The PCI master write data is then written to DRAM.
Hit Modified Line	Miss	The first level cache line is written back to DRAM and invalidated. The PCI master write data is then written to DRAM.
Hit Modified Line	Hit Unmodified Line	The first level cache line is written back to DRAM and invalidated. The second level cache line is invalidated. The PCI master write data is then written to DRAM.
Hit Modified Line	Hit Modified Line	The first level cache line is written back to DRAM and invalidated. The second level cache line is invalidated. The PCI master write data is then written to DRAM.

A snoop hit results in one of three transfers; a write-back from the first level cache posted to the LBXs, a write-back from the second level cache posted to the LBXs or a write-back from the first level cache posted to the LBXs and written to the second level cache. A snoop cycle that does not result in a write-back is depicted in Figure 34.

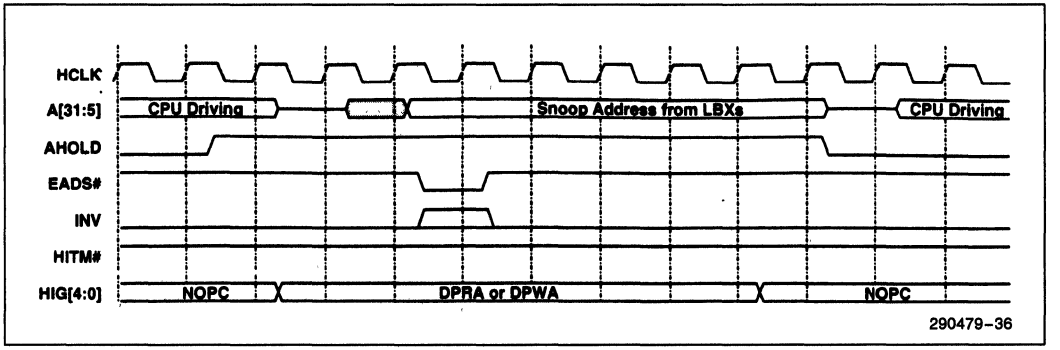


Figure 34. Snoop Hit to Unmodified Line in First Level Cache or Snoop Miss

The PCMC begins to service the snoop request by asserting AHOLD, causing the Pentium processor to tri-state the address bus in the clock after assertion. In the case of a PCI master read cycle, the PCMC drives the DPRA (Drive PCI Read Address) command onto the HIG[4:0] lines causing the LBXs to drive the PCI address onto the host address bus. For a write cycle, the PCMC drives the DPWA (Drive PCI Write Address to CPU Address Bus) command on the HIG[4:0] lines, also causing the LBXs to begin driving the host address bus. The PCMC then asserts EADS#, initiating the snoop cycle to the CPU. The INV signal is asserted by the PCMC only during snoops due to PCI master writes. INV remains negated during snoops due to PCI master reads. If the snoop results in a hit to a modified line in the first level cache, the Pentium processor asserts HITM#. The PCMC samples the HITM# signal two clocks after the CPU samples EADS# asserted to determine if the snoop hit in the first level cache. By this time the PCMC has completed an internal tag lookup to determine if the line is in the second level cache. Since this snoop does not result in a write-back, the NOPC command is driven on the HIG[4:0] lines, causing the LBXs to tri-state the address bus. The sequence ends with AHOLD negation.

If the Pentium processor asserts ADS# in the same clock as the PCMC asserts AHOLD, the PCMC will assert BOFF# in two cases. First, if the snoop cycle hits a modified line in the first level cache, the PCMC will assert BOFF# for 1 HCLK to re-order the write-back around the currently sending cycle. Second, if the snoop requires a write-back from the second level cache, the PCMC will assert BOFF# to enable the write-back from the secondary cache SRAMs.

Figure 35 depicts a snoop hit to a modified line in the first level cache due to a PCI master memory read cycle.

The snoop cycle begins when the PCMC asserts AHOLD causing the CPU to tri-state the address bus. The PCMC drives the DPRA (Drive PCI Read Address) command on to the HIG[4:0] lines causing the LBXs to drive the PCI address onto the host address bus. The PCMC then asserts EADS#, initiating the snoop to the first level cache. INV is not asserted since this is a PCI master read cycle. INV is only asserted with EADS# when the snoop cycle is in response to a PCI master write cycle. As the CPU is sampling EADS# asserted, the PCMC latches the address. Two clocks later, the PCMC completes the

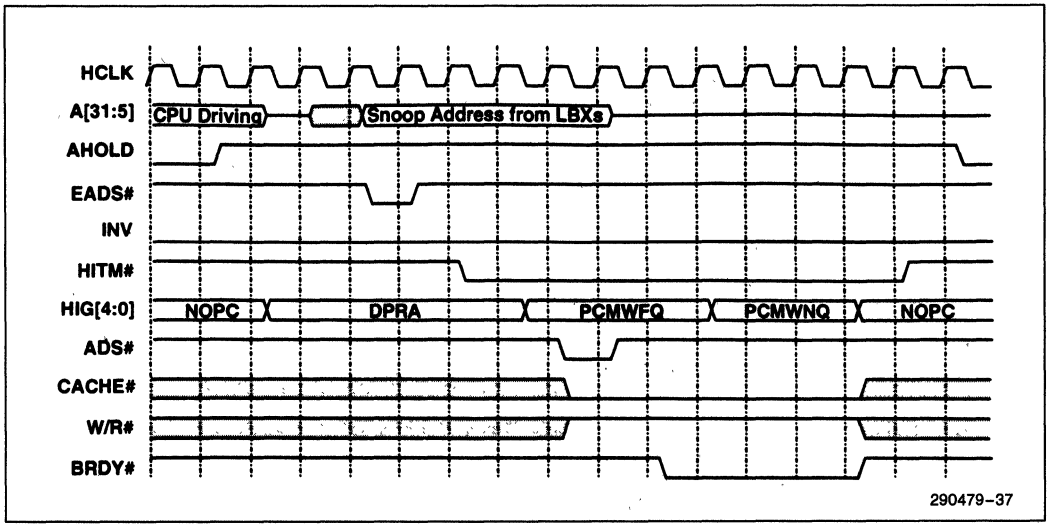


Figure 35. Snoop Hit to Modified Line in First Level Cache, Post Memory and PCI

internal tag lookup to determine if the line is in the second level cache. In this instance, the snoop hits a modified line in the first level cache and misses in the second level cache. Thus, the second level cache is not involved in the write-back cycle. The PCMC allows the LBXs to stop driving the address lines by driving NOPC command on the HIG[4:0] lines. The CPU then drives the write-back cycle onto the bus by asserting ADS# and driving the write-back data on the data lines even though AHOLD is still asserted. The write-back into the LBX buffers occurs at a rate of 3-1-1-1. The PCMC drives PCMWFQ on the HIG[4:0] lines for one clock causing the write data to be posted to both PCI and main memory. For the next three clocks, the HIG[4:0] lines are driven to PCMWNQ, posting the final three Qwords to both PCI and main memory.

A similar transfer from first level cache to the LBXs occurs when a snoop due to a PCI master write hits a modified line in the first level cache. In this case, the write-back is transferred to the CPU-to-Memory Posted Write Buffer. If the line is in the second level cache, it is invalidated. The cycle is similar to the snoop cycle shown above with two exceptions. The PCMC drives the DPWA command on the HIG[4:0] lines instead of the DPRA command. During the four clocks where the PCMC drives BRDY# active to the CPU, it also drives PCMWQ on the HIG[4:0] lines, causing the write to be posted to main memory.

In both of the above cases where a write-back from the first level cache is required, AHOLD is asserted until the write-back is complete. If the CPU has begun a read cycle directed to PCI and the snoop results in a hit to a modified line in the first level cache, BOFF# is asserted for one clock to abort the CPU read cycle and re-order the write-back cycle before the read cycle.

When a PCI master read or write cycle hits a modified line in the second level cache and either misses in the first level cache or hits an unmodified line in the first level cache, a write-back from the second level cache to the LBXs occurs. When a PCI master write snoop hits an unmodified line in the second level cache and either misses in the first level cache or hits an unmodified line in the first level cache, no data transfer from the second level cache occurs. The line is simply invalidated. In the case of a PCI master write cycle, the line is invalidated in both the first level and second level caches. In the case of a PCI master memory read cycle, neither cache is invalidated. A PCI master read from main memory which hits either a modified or unmodified line in the second level cache is shown in Figure 36.

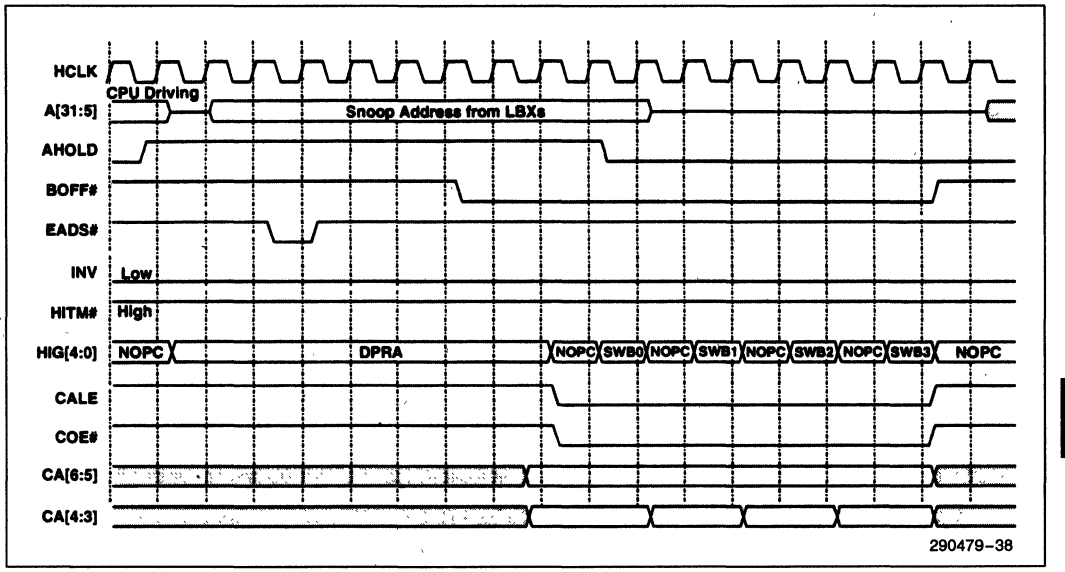


Figure 36. Snoop Hit to Modified Line in Second Level Cache, Store in PCI Read Prefetch Buffer

The snoop cycle begins with the PCMC asserting AHOLD, causing the CPU to tri-state the host address bus. The PCMC drives the DPRA command enabling the LBXs to drive the snoop address onto the host address bus. The PCMC asserts EADS#. INV is not asserted in this case since the snoop cycle is in response to a PCI master read cycle. If the snoop were in response to a PCI master write cycle then INV would be asserted with EADS#. Two clocks after the CPU samples EADS# active, the PCMC completes the internal tag lookup. In this case the snoop hit either an unmodified line or a modified line in the second level cache. Since HITM# is inactive, the snoop did not hit in the first level cache. The PCMC then schedules a read from the second level cache to be written to the LBXs. When the CPU burst cycle completes the PCMC negates the control signals to the second level cache and asserts CALE opening the cache address latch and allowing the snoop address to flow through to the SRAMs. The second level cache executes a read sequence which completes at 3-2-2-2 in the case of standard SRAMs and 3-1-1-1 in the case of burst SRAMs. During all snoop cycles where a write-back from the second level cache is required, BOFF# is asserted throughout the write-back cycle. This prevents the deadlock that would occur if the CPU is in the middle of a non-postable write and the data bus is required for the second level cache write-back.

When using burst SRAMs, the read from the SRAMs follows the Pentium processor burst order. However, the memory to PCI read prefetch buffer in the LBXs is organized as a FIFO and cannot accept data out of order. The SWB0, SWB1, SWB2 and SWB3 commands are used to write data into the buffer in ascending order. In the above example, the PCI master requests a data item which hits Qword 0 in the cache, thus CA[4:3] count through the following sequence: 0, 1, 2, 3 (00, 01, 10, 11). If the PCI mas-

ter requests a data item that hits Qword 1, the SWB0 command is sent via the HIG[4:0] lines to store Qword 1in the first buffer location. The next read from the cache is not in ascending order, thus a NOPC is sent on the HIG[4:0] lines. This Qword is not posted in the buffer. The next read from the cache is to Qword 3. SWB2 is sent on the HIG[4:0] lines. The final read from the cache is Qword 2. SWB1 is sent on the HIG[4:0] lines. Thus, Qword 1 is placed in entry 0 in the buffer, Qword 2 is placed in entry 1 in the buffer and Qword 3 is placed in entry 2 in the buffer. The ordering between the Qwords read from the cache and the HIG[4:0] commands when using burst SRAMs is summarized in Table 9.

Table 9. HIG[4:0] Command Sequence for Second Level Cache to PCI Master Read Prefetch Buffer Transfer

Burst Order from Cache	HIG[4:0] Command Sequence
0, 1, 2, 3	SWB0, SWB1, SWB2, SWB3
1, 0, 3, 2	SWB0, NOPC, SWB2, SWB1
2, 3, 0, 1	SWB0, SWB1, NOPC, NOPC
3, 2, 1, 0	SWB0, NOPC, NOPC, NOPC

When using standard asynchronous SRAMs, the read from the SRAMs occurs in a linear burst order. Thus, CAA[4:3] and CAB[4:3] count in a linear burst order and the Store Write Buffer commands are sent in linear order. The burst ends at the cache line boundary and does not wrap around and continue with the beginning of the cache line.

A PCI master write cycle which hits a modified line in the second level cache and either hits an unmodified line in the first level cache or misses in the first level cache will also cause a transfer from the second level cache to the LBXs. In this case, the read from the SRAMs is posted to main memory and the line is invalidated in the second level cache. The cycle would differ only slightly from the above cycle. INV would be asserted with EADS#. Instead of the DPRA command, the PCMC would use the DPWA command to drive the snoop address onto the host address bus. The write would be posted to the DRAM, thus the PCMC would drive the PCMWQ command on the HIG[4:0] lines to post the write to DRAM.

A snoop cycle can result in a write-back from the first level cache to both the second level and LBXs in the case of a PCI master read cycle which hits a modified line in the first level cache and hits either a modified or unmodified line in the second level cache. The line is written to both the second level cache and the memory to PCI read prefetch buffer. The cycle is shown in Figure 37.

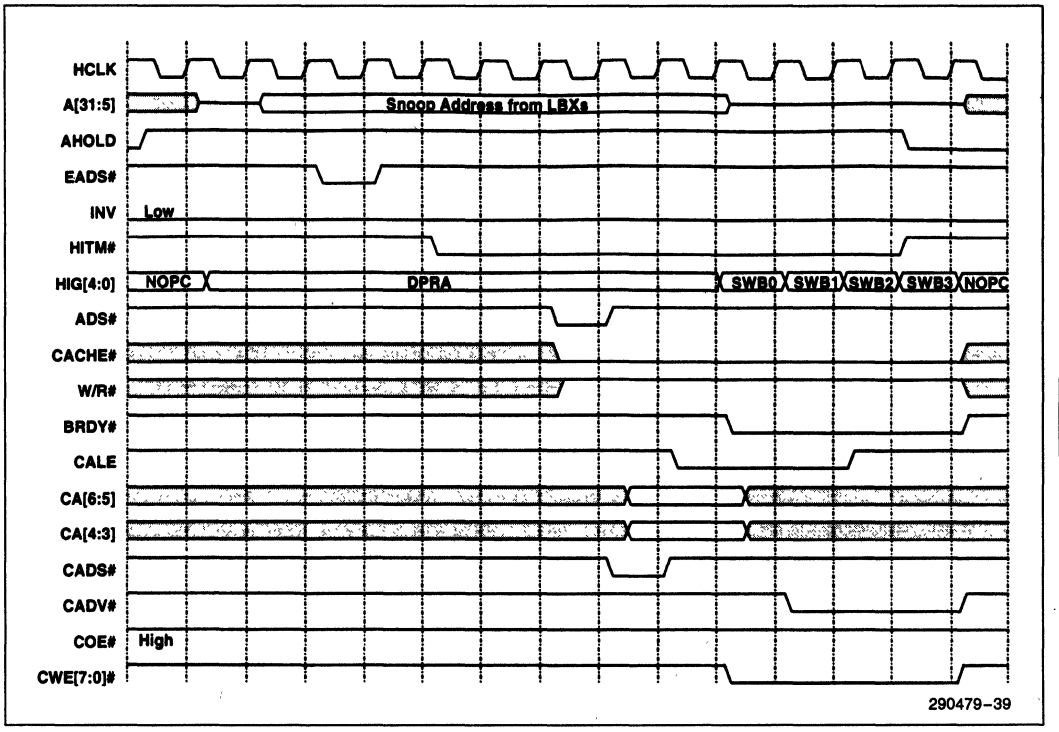


Figure 37. Snoop Hit to Modified Line in First Level Cache, Write-Back from First Level Cache to Second Level Cache and Send to PCI

This cycle is shown for the case of a second level cache with burst SRAMs. In this case, as it completes the second level cache tag lookup, the PCMC samples HITM# active. The write-back is written to the second level cache and simultaneously stored in the memory to PCI prefetch buffer. In the case shown in Figure 33, the PCI master requests a data item which is contained in Qword 0 of the cache line. Note that a write-back from the first level cache always starts with Qword 0 and finishes with Qword 3. Thus the HIG[4:0] lines are sequenced through the following order: SWB0, SWB1, SWB2, SWB3. If the PCI master requests a data item which is contained in Qword 1, the HIG[4:0] lines sequence through the following order: NOPC, SWB0, SWB1, SWB2. If the PCI master requests a data item which is contained in Qword 2, the HIG[4:0] lines sequence through the following order: NOPC, NOPC, SWB0, SWB1. If the PCI master requests a data item which is contained in Qword 3, the HIG[4:0] lines sequence through the following order: NOPC, NOPC, NOPC, SWB0. AHOLD is negated after the write-back cycle is complete.

If the CPU has begun a read cycle directed to PCI and the snoop results in a hit to a modified line in the first level cache, BOFF# is asserted for one clock to abort the CPU read cycle and re-order the write-back cycle before the pending read cycle.

5.1.5 FLUSH, FLUSH ACKNOWLEDGE AND WRITE-BACK SPECIAL CYCLES

There are three special cycles that affect the second level cache, flush, flush acknowledge, and write-back. If the processor executes an INVD instruction, it will invalidate all unmodified first level cache lines and issue a flush special cycle. If the processor executes a WBINVD instruction, it will write back all modified first level cache lines, invalidate the first level cache, and issue a write-back special cycle followed by a flush special cycle. If the Pentium processor FLUSH# pin is asserted, the CPU will write-back all modified first level cache lines, invalidate the first level cache, and issue a flush acknowledge special cycle.

The second level cache behaves the same way in response to the flush special cycle and flush acknowledge special cycle. Each tag is read and the valid and modified bits are examined. If the line is both valid and modified it is written back to main memory and the valid bit for that line is reset. All valid and unmodified lines are simply marked invalid. The PCMC advances to the next tag when all lines within the current sector have been examined. BRDY# is returned to the Pentium processor after all modified lines in the second level cache have been written back to main memory and all of the valid bits for the second level cache are reset. The sequence of write-back cycles will only be interrupted to service a PCI master cycle.

The write-back special cycle is ignored by the PCMC because all modified lines will be written back to main memory by the following flush special cycle. Upon decoding a write-back special cycle, the PCMC simply returns BRDY# to the Pentium processor.

5.2 82434NX Cache

The 82434NX PCMC integrates a high performance write-back second level cache controller, tag RAM and a full first and second level cache coherency mechanism. The cache is either 256 KBytes or 512 KBytes using either synchronous burst SRAMs or standard asynchronous SRAMs. Parity on the data SRAMs is optional. The cache uses a write-back write policy. Write-through mode is not supported.

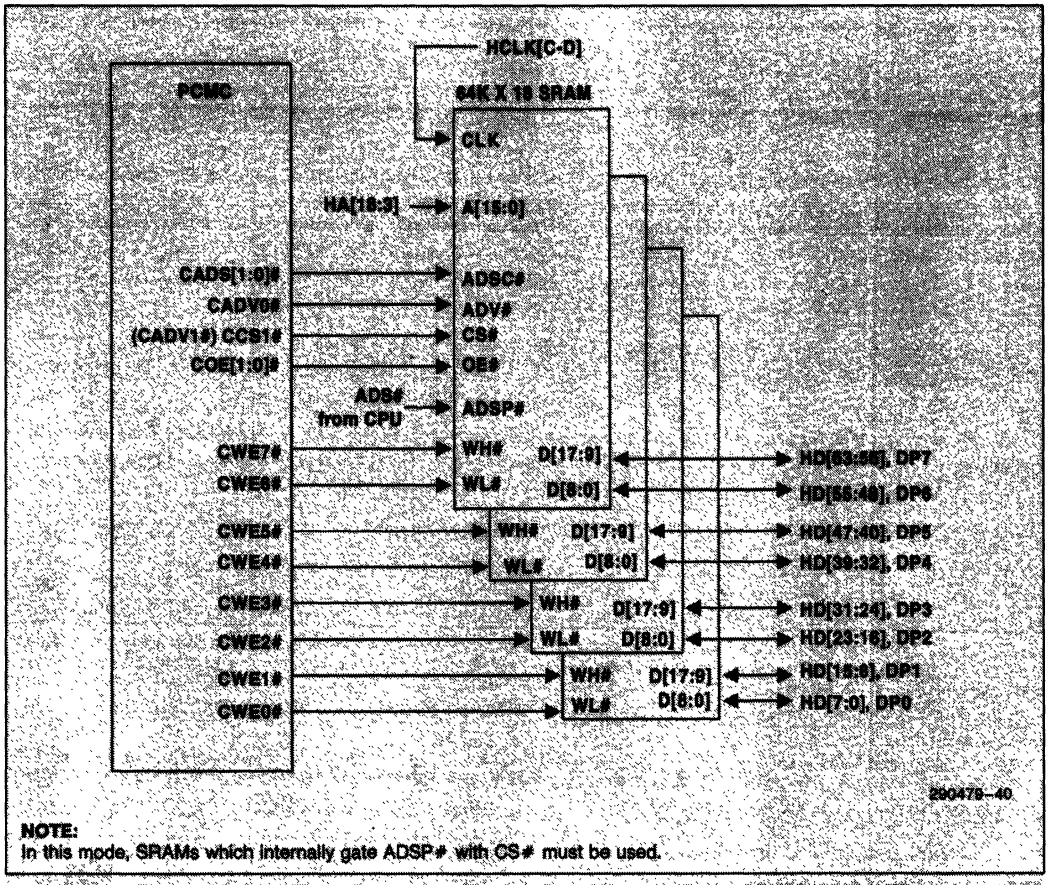
The 82434NX PCMC supports a direct mapped secondary cache. The PCMC contains 4096 tags. Each tag represents a sector in the cache. If the cache is 512 KB, each sector contains four cache lines. If the cache is 256 KB, each sector contains two cache lines. *Valid* and *Modified* bits are kept on a per line basis. The 82434NX Tag RAM is 1 bit wider than the 82434LX Tag RAM.

The PCMC can be configured to cache main memory on read cycles even when CACHE# is not asserted. When bit 4 in the Secondary Cache Control Register (offset 52h) is set to 1, all accesses to main memory, except those to SMM memory or any range marked non-cacheable via the PAM registers, are cached in the secondary cache. Accesses with CACHE# asserted result in a line fill in both the first and second level cache while accesses with CACHE# negated result in a line fill only in the second level cache. When bit 4 in the SCC Register is set to 0, only access with CACHE# asserted can generate a first and second level cache line fill.

When a Halt or Stop Grant Special Cycle is detected from the CPU, the 82434NX PCMC places the second level cache into the low power stand-by mode by deselecting the SRAMs and then generates the corresponding special cycle on PCI. (i.e., if the CPU cycle was a halt special cycle then the PCMC generates a halt special cycle on PCI and if the CPU cycle is a stop grant special cycle the PCMC generates a stop grant special cycle on PCI).

When a burst SRAM secondary cache is implemented, bit 2 of the Secondary Cache Control Register (offset 52h) is used to select between 82434LX SRAM connectivity and the new 82434NX SRAM connectivity. When set to 0, the secondary cache interface is in 82430-compatible mode. (i.e., the four low order address lines on the SRAMs are connected to CAA/B[6:3] on the PCMC. When set to 1, second level cache stand-by is enabled and no latch is used between the host CPU address lines and the SRAM address lines. All of the SRAM address lines are then connected directly to the CPU address lines. Write-back addresses are driven by the PCMC over the host address lines. When a standard SRAM secondary cache is implemented, bit 2 of the Secondary Cache Control Register (offset 52h) is used to enable second level cache stand-by. The default value of this bit is 0.

Figure 38 and Figure 41 show the connections between the PCMC and the external cache data SRAMs and latch for the case of an asynchronous SRAM cache.



NOTE:
In this mode, SRAMs which internally gate ADSP# with CS# must be used.

Figure 38. 512 KByte Secondary Cache, Synchronous Burst SRAM (82434NX)

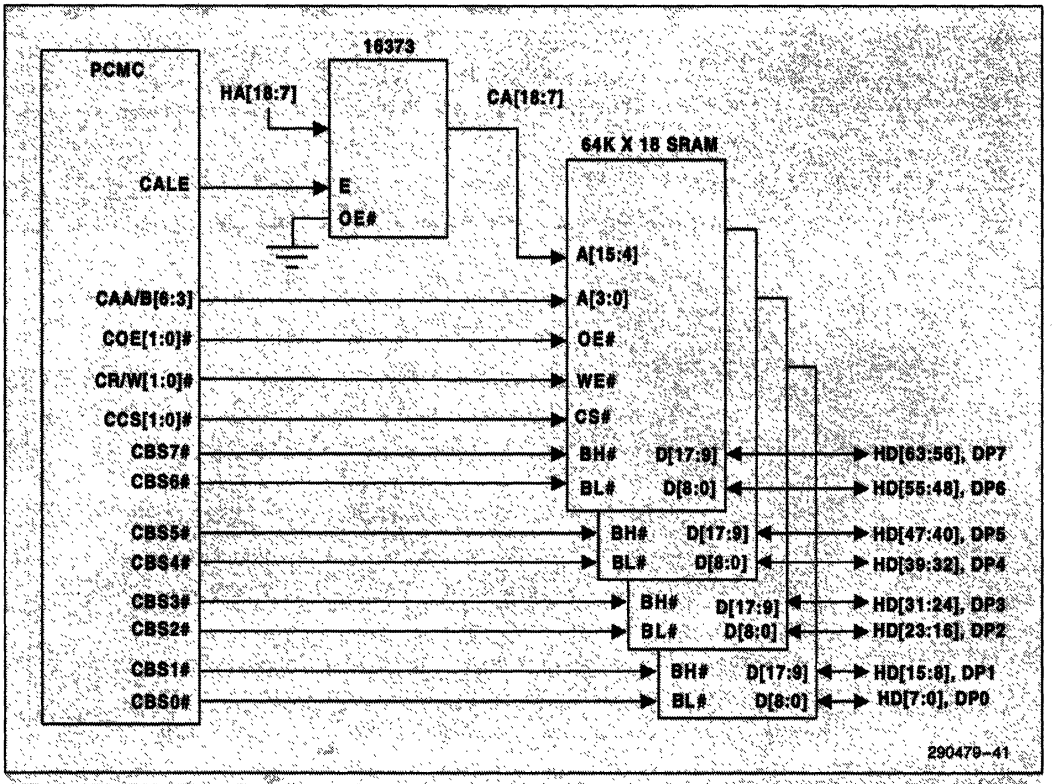


Figure 39. 512 KByte Secondary Cache, Standard Dual-Byte-Select (Asynch) SRAM, 50, 60 & 66 MHz

Figure 38 depicts the PCMC connections to a 512 KByte burst SRAM secondary cache when the PCMC is configured for 50, 60, or 66 MHz operation. Host address lines HA[18:3] are connected directly to the SRAM address lines, A[15:0]. ADS# from the CPU is connected to ADSP# on the SRAMs. CADV0# implements the address advance (ADV#) functionality. A new signal, CCS#, is multiplexed onto the CADV1# pin. When bit 2 in the SCC register is set to 1, SRAMs containing logic which gates ADSP# with CS# must be used. When negated, CCS# prevents the SRAMs from latching a new address due to a pipelined ADS# from the CPU during cache line fills. Note that, unlike the burst SRAM configuration with the 82430 PCIset, no external latch is used between the CPU address bus and the SRAM address lines. The SRAM Connectivity bit (bit 2) in the Secondary Cache Control register (offset 52h) must be set to 1 when using this cache configuration.

If the tag lookup results in a miss in the cache and the sector to be replaced contains one or more modified lines, the PCMC drives the write-back address from the A[18:3] lines on the host bus. Although not used in the write-back, A[31:19] (or A[31:18] in the case of a 256 KB cache) are driven to valid logic levels by the PCMC.

Figure 39 depicts the 82434NX PCMC connections to a 512 KByte standard asynchronous SRAM secondary cache. Figure 40 depicts the 82434NX connections to a 256 KByte asynchronous SRAM secondary cache. Host address lines HA[18:7] are driven through an external latch to form the upper SRAM address lines, CA[18:7]. CA[6:3] are driven from the PCMC. Figure 41 depicts the 82434NX PCMC connections to a 512 KByte standard SRAM secondary cache with dual-write-enable SRAMs.

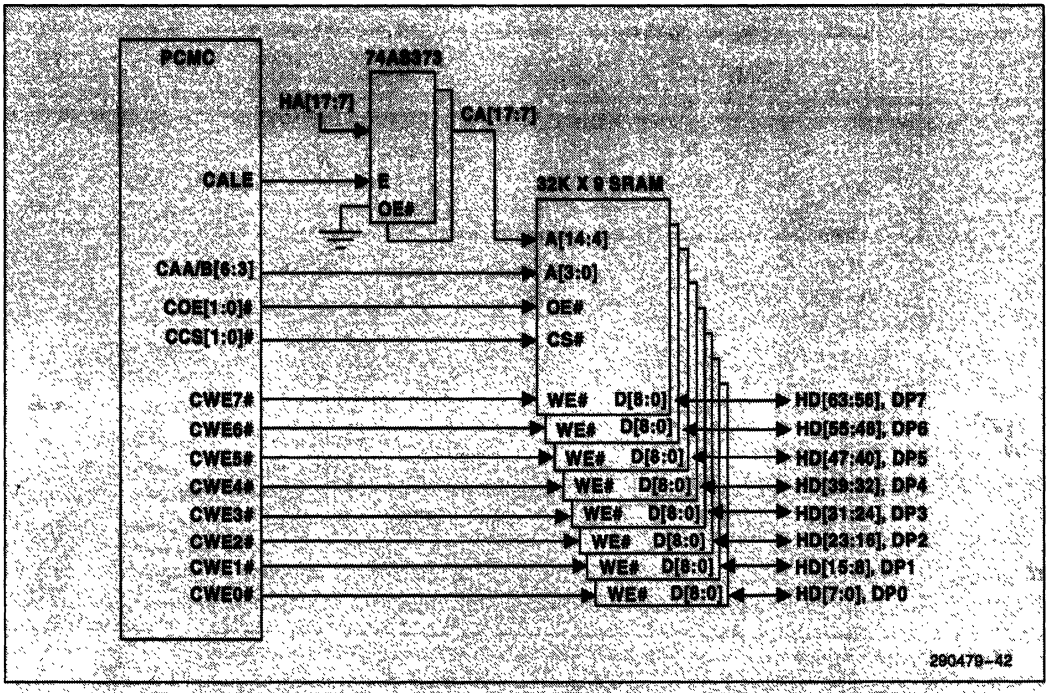


Figure 40. 82434NX Connections to 256 KByte Cache with Standard SRAM

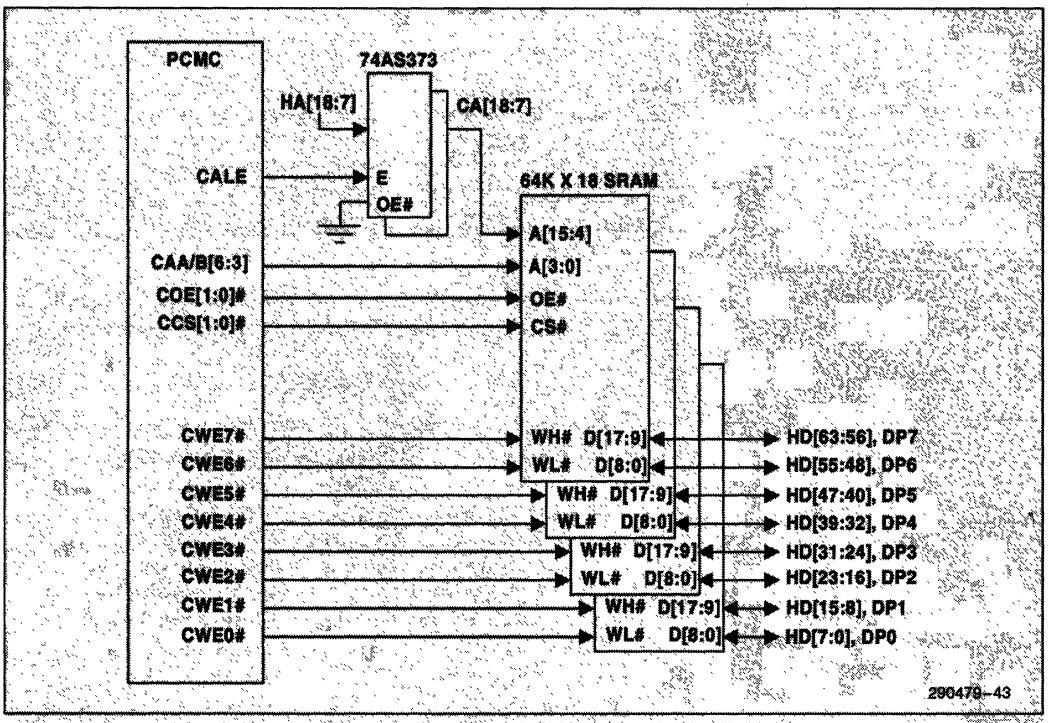


Figure 41. 82434NX Connections to 512 KByte Cache with Standard SRAM

5.2.1 CYCLE LATENCY SUMMARY (82434NX)

Table 10 and Table 11 summarize the clock latencies for CPU memory cycles which hit in the secondary cache.

Table 10. Secondary Cache Latencies with Synchronous Burst SRAM

Cycle Type	50, 60 and 66 MHz
Burst Read	3-1-1-1
Burst Write	3-1-1-1
Single Read	3
Single Write	3
Pipelined Back-to-Back Burst Reads	3-1-1-1-1-1-1-1
Burst Read Followed by Pipelined Write	3-1-1-1-2

Table 11. Secondary Cache Latencies with Standard Asynchronous SRAM (82434NX)

Cycle Type	50, 60 and 66 MHz
Burst Read	3-2-2-2
Burst Write	4-2-2-2
Single Read	3
Single Write	4
Pipelined Back-to-Back Burst Reads	3-2-2-2-3-2-2-2
Burst Read Followed by Pipelined Write	3-2-2-2-4

The 60 MHz and 66 MHz asynchronous SRAM latencies require 15 ns and 12 ns SRAMs, respectively. The 82434NX PCMC supports asynchronous SRAMs at 50 MHz. The 50 MHz (1 wait-state) timings require 20 ns SRAMs. The burst SRAMs speeds for 66 MHz, 60 MHz and 50 MHz operation are 8 ns, 9 ns, and 13 ns clock-to-output valid into a 0 pF test load. The SRAM access times listed in this paragraph are recommendations. Actual access time requirements are a function of system board layout and routing and should be validated with electrical simulation.

5.2.2 STANDARD SRAM CACHE CYCLES (82434NX)

At 50, 60 and 66 MHz, the timing of the second level cache interface with standard asynchronous SRAMs is identical to the timing in the 82430LX PCIset. Compared to the 82434LX second level cache, one additional connection can be made from the PCMC to the SRAMs. The CCS[1:0]# pins, in the case of asynchronous SRAMs, are multiplexed onto the CADV[1:0]# pins. These are then connected to the SRAM CS# pins. The two copies are functionally identical. The two copies are provided for timing reasons. These pins allow the PCMC to deselect the SRAMs, putting them into standby mode. When a halt special cycle or a stop grant special cycle is detected from the CPU, the PCMC negates CCS[1:0]#, placing the SRAMs into the low power standby mode. The PCMC then generates a halt or stop grant special cycle on PCI.

5.2.3 SECOND LEVEL CACHE STANDBY

When the PCMC detects a halt or stop grant special cycle from the CPU, it first places the second level cache into the low power stand-by mode by deselecting the SRAMs and then generates a halt or stop grant special cycle on PCI.

With a standard SRAM secondary cache, a halt or stop grant special cycle from the CPU causes the PCMC to negate CCS[1:0]#, deselecting the SRAMs and placing them in a low power standby mode. When the cache is in stand-by mode, the first bus cycle from the CPU brings the cache out of stand-by and into active mode, enabling the SRAMs to service the cycle in the case of a hit to the cache. The PCMC asserts CCS[1:0]# as a propagation delay from the falling edge of ADS#. CCS[1:0]# are then left asserted until the next halt or stop grant special cycle is occurs. When exiting the powerdown state, the PCMC ignores the Secondary Cache Leadoff wait-states bit and executes a 3-2-2-2 read or 4-2-2-2 write in order to allow the SRAMs time to power up. In the case of a read cycle, COE[1:0]# are asserted in clock two as in the case of ordinary read cycles.

When the SRAMs are powered down, the PCMC asserts CCS[1:0]# when performing a snoop cycle, regardless of whether the cycle hits in the second level cache. The PCMC then negates CCS# after the snoop cycle is complete.

With a burst SRAM secondary cache, a halt or stop grant special cycle from the CPU causes the PCMC to negate CCS# and assert CADS[1:0]#, deselecting the SRAMs, placing them in a low power standby mode. CCS# is then asserted and is left asserted by the PCMC. Thus, when the first cycle is driven from the CPU, the SRAMs sample ADSP# and CS# active, placing them in active mode and initiating the first access.

If the SRAMs are required to service a snoop, they are brought out of power-down when the PCMC asserts CADS[1:0]#. The PCMC always asserts CADS[1:0]# with CCS# negated after a snoop cycle is complete, regardless of whether the SRAMs were powered down prior to the snoop cycle.

5.2.4 SNOOP CYCLES

For snoop operations, refer to Section 5.1, 82434LX Cache.

5.2.5 FLUSH, FLUSH ACKNOWLEDGE, AND WRITE-BACK SPECIAL CYCLES

For flush, flush acknowledge, and write-back special cycles, refer to Section 5.1, 82434LX Cache.

6.0 DRAM INTERFACE

This section describes the DRAM interface for the 82434LX DRAM Interface (Section 6.1) and the 82434NX DRAM Interface (Section 6.2). The differences are in the following areas:

1. Increased maximum DRAM memory size to 512 MBytes. An extra address line (MA11) has been added to the 82434NX.
2. Two additional RAS# lines for a total of eight (RAS[0:7]#.
3. Addition of 50 MHz host-bus optimized DRAM timing sets. Thus, the 82434LX supports 60 and 66 MHz frequencies and the 82434NX supports 50, 60, and 66 MHz.

6.1 82434LX DRAM Interface

The 82434LX PCMC integrates a high performance DRAM controller supporting from 2–192 MBytes of main memory. The PCMC generates the RAS#, CAS#, WE# and multiplexed addresses for the DRAM array, while the data path to DRAM is provided by two 82433LX LBXs. The DRAM controller interface is fully configurable through a set of control registers. Complete descriptions of these registers are given in Section 3.0, Register Description. A brief overview of the registers which configure the DRAM interface is provided in this section.

The 82434LX controls a 64-bit memory array (72-bit including parity) ranging in size from 2 MBytes up to 192 MBytes using industry standard 36-bit wide memory modules with fast page-mode DRAMs. Both single- and double-sided SIMMs are supported. The eleven multiplexed address lines, MA[10:0] allow the PCMC to support 256K x 36, 1M x 36, and 4M x 36 SIMMs. The PCMC has six RAS# lines enabling the support of up to six rows of DRAM. Eight CAS# lines allow byte control over the array during read and write operations. The PCMC supports 70 and 60 ns DRAMs. The PCMC DRAM interface is synchronous to the CPU clock and supports page mode accesses to efficiently transfer data in bursts of four Qwords.

The DRAM interface of the PCMC is configured by the DRAM Control Mode Register (offset 57h) and the six DRAM Row Boundary (DRB) Registers (offsets 60h–65h). The DRAM Control Mode Register contains bits to configure the DRAM interface for RAS# modes and refresh options. In addition, DRAM Parity Error Reporting and System Management RAM space can be enabled and disabled. When System Management RAM is enabled, if SMIACT# from the Pentium processor is not asserted, all CPU read and write accesses to SMM memory are directed to PCI. The SMRAM Space Register at configuration space offset 72h provides additional control over the SMRAM space. The six DRB Registers define the size of each row in the memory array, enabling the PCMC to assert the proper RAS# line for accesses to the array.

CPU-to-Memory write posting and read-around-write operations are enabled and disabled via the Host Read/Write Buffer Control Register (offset 53h). PCI-to-Memory write posting is enabled and disabled via the PCI Read/Write Buffer Control Register (offset 54h). PCI master reads from main memory always result in the PCMC and LBXs reading the requested data and prefetching the next seven Dwords.

Seven Programmable Attribute Map (PAM) Registers (offsets 59h–5Fh) are used to specify the cacheability and read/write status of the memory space between 512 KBytes and 1 MByte. Each PAM Register defines a specific address area enabling the system to selectively mark specific memory ranges as cacheable, read-only, write-only, read/write or disabled. When a memory range is disabled, all CPU accesses to that range are directed to PCI.

Two other registers also affect the DRAM interface, the Memory Space Gap Register (offsets 78h–79h) and the Frame Buffer Range Register (offsets 7Ch–7Fh). The Memory Space Gap Register is used to place a logical hole in the memory space between 1 MByte to 16 MBytes to accommodate memory mapped ISA boards. The Frame Buffer Range Register, is used to map a linear frame buffer into the Memory Space Gap or above main memory. When enabled, accesses to these ranges are never directed to the DRAM interface, but are always directed to PCI.

6.1.1 DRAM CONFIGURATIONS

Figure 42 illustrates a 12-SIMM configuration which supports single-sided SIMMs. A row in the DRAM array is made up of two SIMMs which share a common RAS# line. SIMM0 and SIMM1 are connected to RAS0# and therefore, comprise row 0. SIMM10 and SIMM11 form row 5. Within any given row, the two SIMMs must be the same size. Among the six rows, SIMM densities can be mixed in any order. That is, there are no restrictions on the ordering of SIMM densities among the six rows.

The low order LBX (LBXL) is connected to byte lanes 5, 4, 1, and 0 of the host and memory data buses, and the lower two bytes of the PCI AD bus. The high order LBX (LBXH) is connected to byte lanes 7, 6, 3, and 2 of the host and memory data buses, and the upper two bytes of the PCI AD bus. Thus, SIMMs connected to LBXL are connected to CAS[5:4,1:0]# and SIMMs connected to LBXH are connected to CAS[7:6, 3:2]#.

The MA[10:0] and WE# lines are externally buffered to drive the large capacitance of the memory array. Three buffered copies of the MA[10:0] and WE# signals are required to drive the six row array.

Figure 43 illustrates a 6-SIMM configuration that supports either single- or double-sided SIMMs. In this configuration, single- and double-sided SIMMs can be mixed. For example, if single-sided SIMMs are installed into the sockets marked SIMM0 and SIMM1, then RAS0# is connected to the SIMMs and RAS1# is not connected. Row 0 is then populated and row 1 is empty. Two double-sided SIMMs could then be installed in the sockets marked SIMM2 and SIMM3, populating rows 2 and 3.

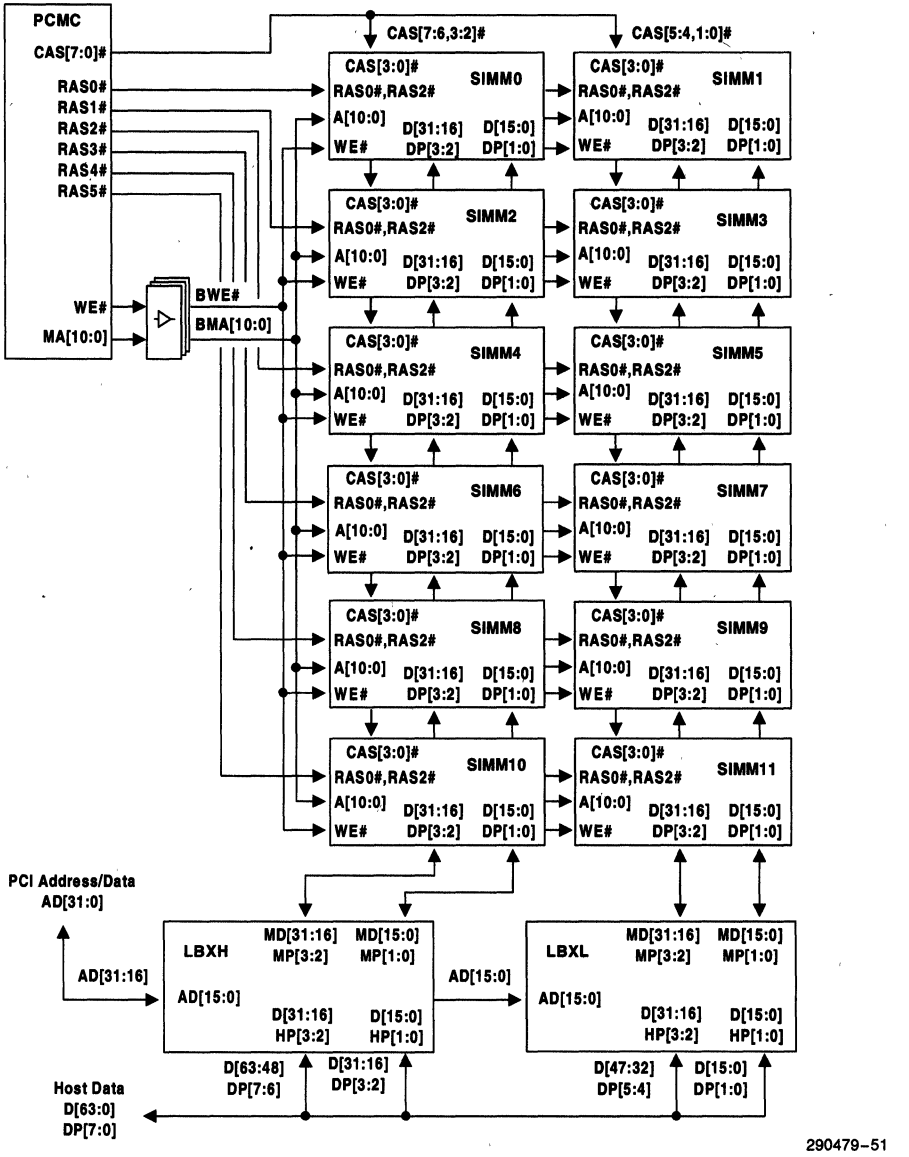
6.1.2 DRAM ADDRESS TRANSLATION

The 82434LX multiplexed row/column address to the DRAM memory array is provided by the MA[10:0] signals. The MA[10:0] bits are derived from the host address bus as defined by Table 12.

MA[10:0] are translated from the host address A[24:3] for all memory accesses, except those targeted to memory that has been remapped as a result of the creation of a memory space gap in the lower extended memory area. In the case of a cycle targeting remapped memory, the least significant bits come directly from the host address, while the more significant bits depend on the memory space gap start address, gap size, and the size of main memory.

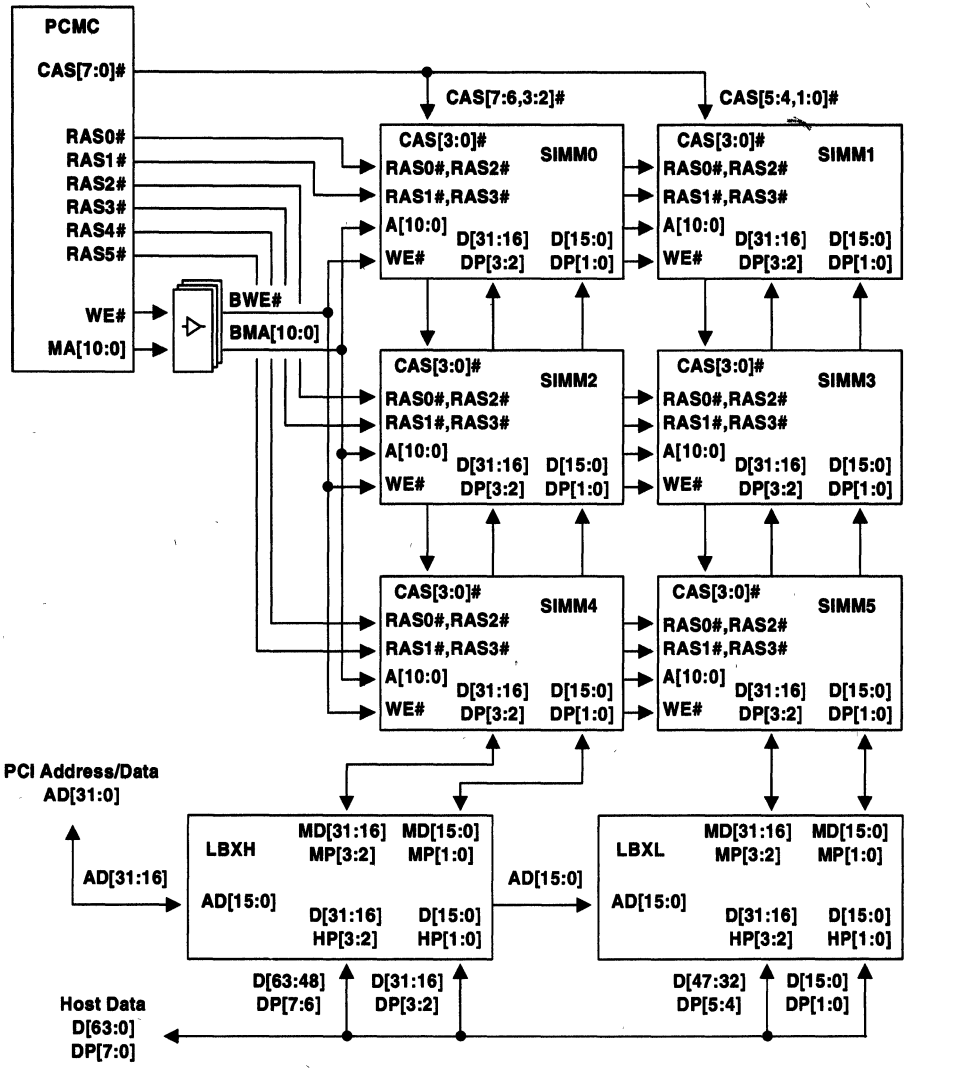
Table 12. DRAM Address Translation

Memory Address, MA[10:0]	10	9	8	7	6	5	4	3	2	1	0
Row Address	A24	A22	A20	A19	A18	A17	A16	A15	A14	A13	A12
Column Address	A23	A21	A11	A10	A9	A8	A7	A6	A5	A4	A3



NOTE:
The figure shows the connections for the 82434LX. For the 82434NX, there are two additional RAS lines (RAS[7:6]#) and one additional address line (MA11).

Figure 42. 82434LX DRAM Configuration Supporting Single-Sided SIMMs



NOTE:
The figure shows the connections for the 82434LX. For the 82434NX, there are two additional RAS lines (RAS[7:6]#) and one additional address line (MA11).

Figure 43. 82434LX DRAM Configuration Supporting Single- or Double-Sided SIMMs

6.1.3 CYCLE TIMING SUMMARY

The 82434LX PCMC DRAM performance is summarized in Table 13 for all CPU read and write cycles.

Table 13. CPU to DRAM Performance Summary

Cycle Type	Burst, x-4-4-4 Timing	Single, x-4-4-4 Timing
Read Page Hit	7-4-4-4	7
Read Row Miss	11-4-4-4	11
Read Page Miss	14-4-4-4	14
Posted Write, WT L2	3-1-1-1	3
Posted Write, WB L2	4-1-1-1	4
Write Page Hit	12-4-4-4	12
Write Row Miss	13-4-4-4	13
Write Page Miss	16-4-4-4	16
0-Active RAS# Mode Read	10-4-4-4	10
0-Active RAS# Mode Write	12-4-4-4	12

CPU writes to the CPU-to-Memory Posted Write Buffer are completed at 3-1-1-1 when the second level cache is configured for write-through mode and 4-1-1-1 when the cache is configured for write-back mode. Table 14 shows the refresh performance in CPU clocks.

Table 14. Refresh Cycle Performance

Refresh Type	Hidden Refresh	RAS# only Refresh	CAS# before RAS#
Single	12	13	14
Burst of Four	48	52	56

6.1.4 CPU TO DRAM BUS CYCLES

This section describes the CPU-to-DRAM cycles for the 82434LX.

6.1.4.1 Read Page Hit

Figure 44 depicts a CPU burst read page hit from DRAM. The 82434LX PCMC decodes the CPU address as a page hit and drives the column address onto the MA[10:0] lines. CAS[7:0]# are then asserted to cause the DRAMs to latch the column address and begin the read cycle. CMR (CPU Memory Read) is driven on the HIG[4:0] lines to enable the memory data to host data path through the LBXs. The PCMC advances the MA[1:0] lines through the Pentium processor burst order, negating and asserting CAS[7:0]# to read each Qword. The host data is latched on the falling edge of MDLE, when CAS[7:0]# are negated. The latch is opened again when MDLE is sampled asserted by the LBXs. The LBXs tri-state the host data bus when HIG[4:0] change to NOPC and MDLE rises. A single read page hit from DRAM is similar to the first read of this sequence. The HIG[4:0] lines are driven to NOPC when BRDY# is asserted.

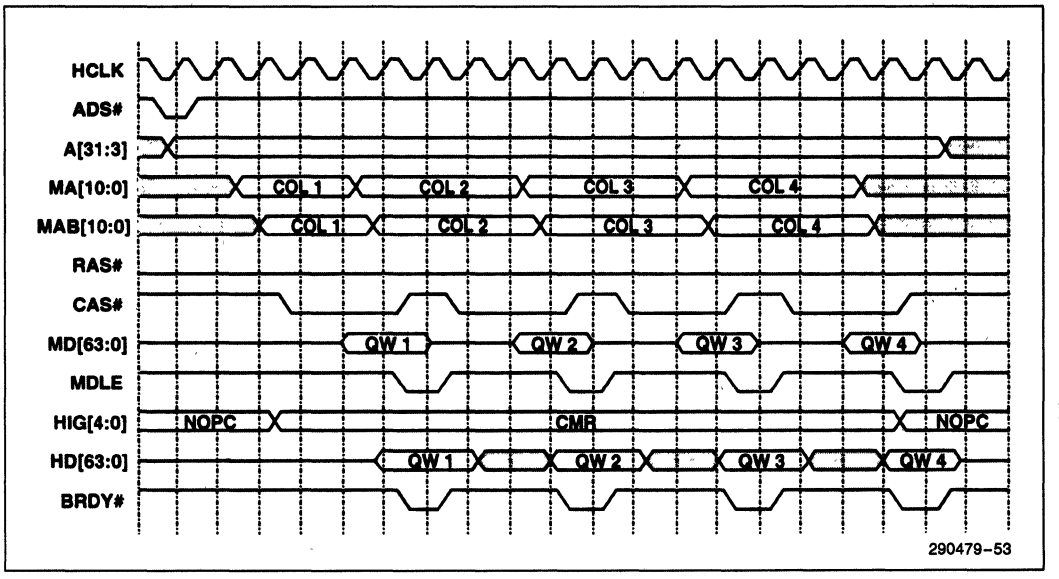


Figure 44. Burst DRAM Read Cycle-Page Hit

6.1.4.2 Read Page Miss

Figure 45 depicts a CPU burst read page miss from DRAM. The 82434LX decodes the CPU address as a page miss and switches from initially driving the column address to driving the row address on the MA[10:0] lines. RAS# is then negated to precharge the DRAMs and then asserted to cause the DRAMs to latch the new row address. The PCMC then switches the MA[10:0] lines to drive the column address and asserts CAS[7:0]#. CMR (CPU Memory Read) is driven on the HIG[4:0] lines to enable the memory data to host data path through the LBXs. The PCMC advances the MA[1:0] lines through the Pentium processor burst order, negating and asserting CAS[7:0]# to read each Qword. The host data is latched on the falling edge of MDLE, when CAS[7:0]# are negated. The latch is opened again when MDLE is sampled asserted by the LBXs. The LBXs tri-state the host data bus when HIG[4:0] change to NOPC and MDLE rises. A single read page miss from DRAM is similar to the first read of this sequence. The HIG[4:0] lines are driven to NOPC when BRDY# is asserted.

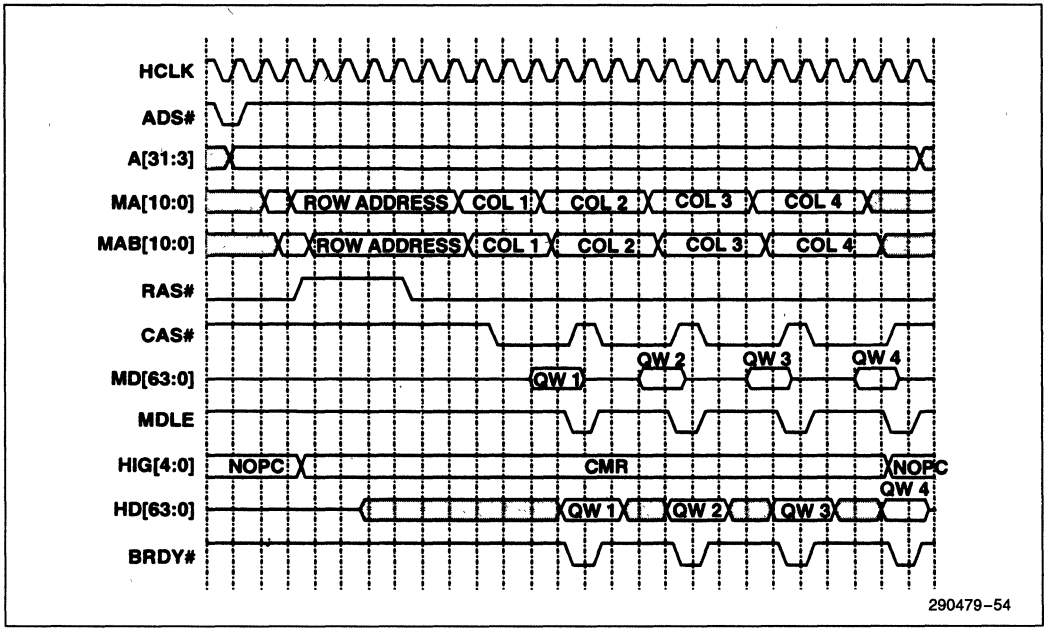


Figure 45. DRAM Read Cycle-Page Miss

6.1.4.3 Read Row Miss

Figure 46 depicts a CPU burst read row miss from DRAM. The 82434LX decodes the CPU address as a row miss and switches from initially driving the column address to driving the row address on the MA[10:0] lines. The RAS# signal that was asserted is negated and the RAS# for the currently accessed row is asserted. The PCMC then switches the MA[10:0] lines to drive the column address and asserts CAS[7:0]#. CMR (CPU Memory Read) is driven on the HIG[4:0] lines to enable the memory data to host data path through the LBXs. The PCMC advances the MA[1:0] lines through the Pentium processor burst order, negating and asserting CAS[7:0]# to read each Qword. The host data is latched on the falling edge of MDLE, when CAS[7:0]# are negated. The latch is opened again when MDLE is sampled asserted by the LBXs. The LBXs tri-state the host data bus when HIG[4:0] change to NOPC and MDLE rises. A single read row miss from DRAM is similar to the first read of this sequence. The HIG[4:0] lines are driven to NOPC when BRDY# is asserted.

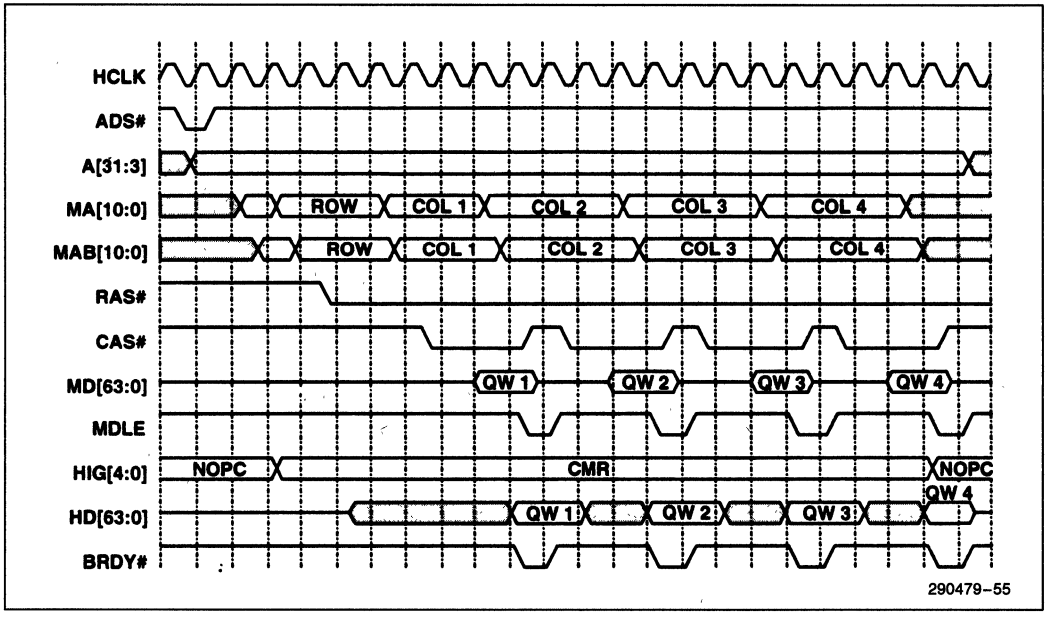


Figure 46. Burst DRAM Read Cycle-Row Miss

6.1.4.4 Write Page Hit

Figure 47 depicts a CPU burst write page hit from DRAM. The 82434LX decodes the CPU write cycle as a DRAM page hit. The HIG[4:0] lines are driven to PCMWQ to post the write to the LBXs. In the figure, the write cycle is posted to the CPU-to-Memory Posted Write Buffer at 4-1-1-1. The write is posted at 4-1-1-1 when the second level cache is configured for a write-back policy. The write is posted to DRAM at 3-1-1-1 when the second level cache is configured for a write-through policy. When the cycle is decoded as a page hit, the PCMC asserts WE# and drives the RCMWQ command on MIG[2:0] to enable the LBXs to drive the first Qword of the write onto the memory data lines. MEMDRV is then driven to cause the LBXs to continue to drive the first Qword for three more clocks. CAS[7:0]# are then negated and asserted to perform the writes to the DRAMs as the MA[1:0] lines advance through the Pentium processor burst order. A single write is similar to the first write of the burst sequence. MIG[2:0] are driven to NOPM in the clock after CAS[7:0]# are asserted.

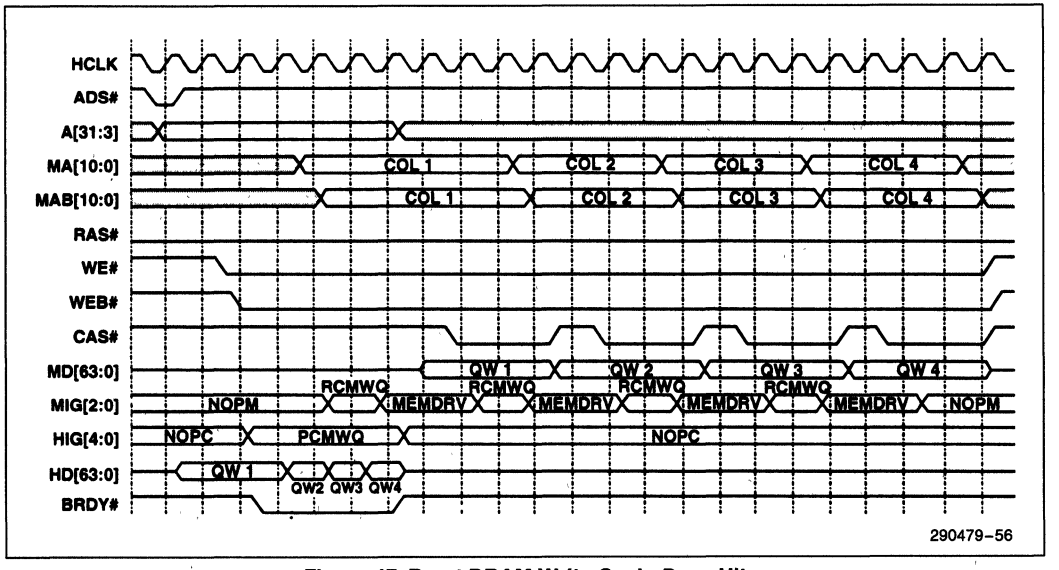


Figure 47. Burst DRAM Write Cycle-Page Hit

6.1.4.5 Write Page Miss

Figure 48 depicts a CPU burst write page miss to DRAM. The 82434LX decodes the CPU write cycle as a DRAM page miss. The HIG[4:0] lines are driven to PCMWQ to post the write to the LBXs. In the figure, the write cycle is posted to the CPU-to-Memory Posted Write Buffer at 4-1-1-1. The write is posted at 4-1-1-1 when the second level cache is configured for a write-back policy. The write is posted to DRAM at 3-1-1-1 when the second level cache is configured for a write-through policy. When the cycle is decoded as a page miss, the PCMC switches the MA[10:0] lines from the column address to the row address and asserts WE#. The PCMC drives the RCMWQ command on MIG[2:0] to enable the LBXs to drive the first Qword of the write onto the memory data lines. MEMDRV is then driven to cause the LBXs to continue to drive the first Qword. The RAS# signal for the currently decoded row is negated to precharge the DRAMs. RAS# is then asserted to cause the DRAMs to latch the row address. The PCMC then switches the MA[10:0] lines to the column address and asserts CAS[7:0]# to initiate the first write. CAS[7:0]# are then negated and asserted to perform the writes to the DRAMs as the MA[1:0] lines advance through the Pentium processor burst order. A single write is similar to the first write of the burst sequence. MIG[2:0] are driven to NOPM in the clock after CAS[7:0]# are asserted.

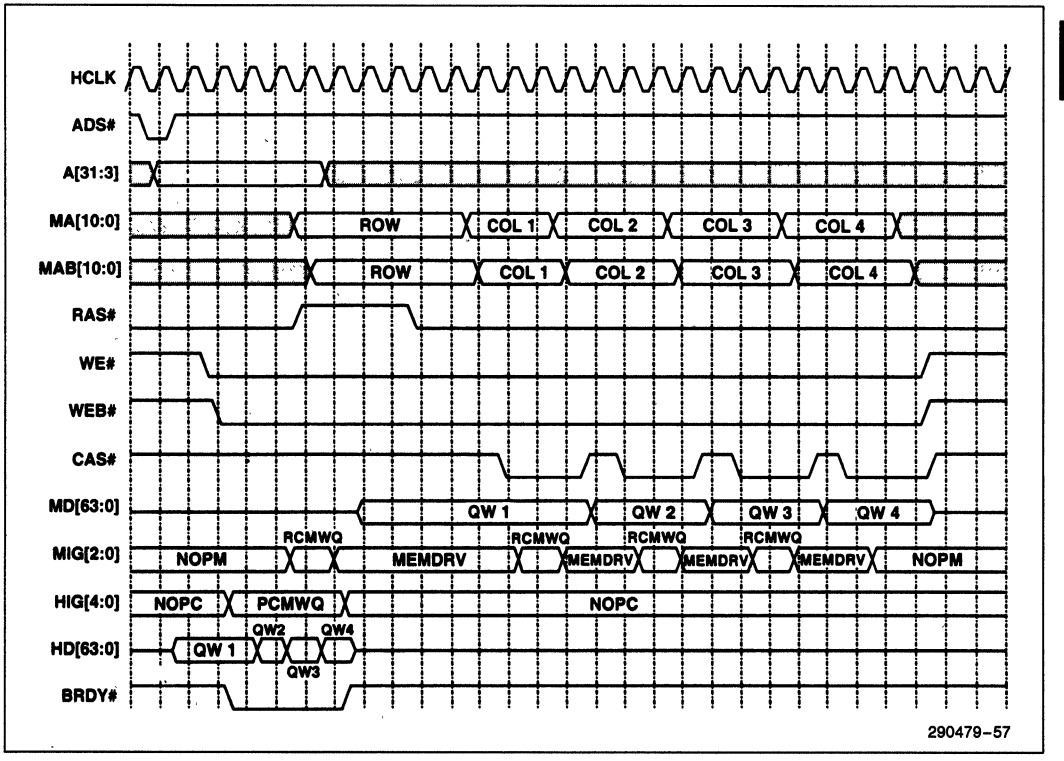


Figure 48. Burst DRAM Write Cycle-Page Miss

6.1.4.6 Write Row Miss

Figure 49 depicts a CPU burst write row miss to DRAM. The 82434LX decodes the CPU write cycle as a DRAM row miss. The HIG[4:0] lines are driven to PCMWQ to post the write to the LBXs. In the figure, the write cycle is posted to the CPU-to-Memory Posted Write Buffer at 4-1-1-1. The write is posted at 4-1-1-1 when the second level cache is configured for a write-back policy. The write is posted to DRAM at 3-1-1-1 when the second level cache is configured for a write-through policy. When the cycle is decoded as a row miss, the PCMC negates the already active RAS# signal, switches the MA[10:0] lines from the column address to the row address and asserts the RAS# signal for the currently decoded row. The PCMC asserts WE# and drives the RCMWQ command on MIG[2:0] to enable the LBXs to drive the first Qword of the write onto the memory data lines. MEMDRV is then driven to cause the LBXs to continue to drive the first Qword. The PCMC then switches the MA[10:0] lines to the column address and asserts CAS[7:0]# to initiate the first write. CAS[7:0]# are then negated and asserted to perform the writes to the DRAMs as the MA[1:0] lines advance through the Pentium processor burst order. A single write is similar to the first write of the burst sequence. MIG[2:0] are driven to NOPM in the clock after CAS[7:0]# are asserted.

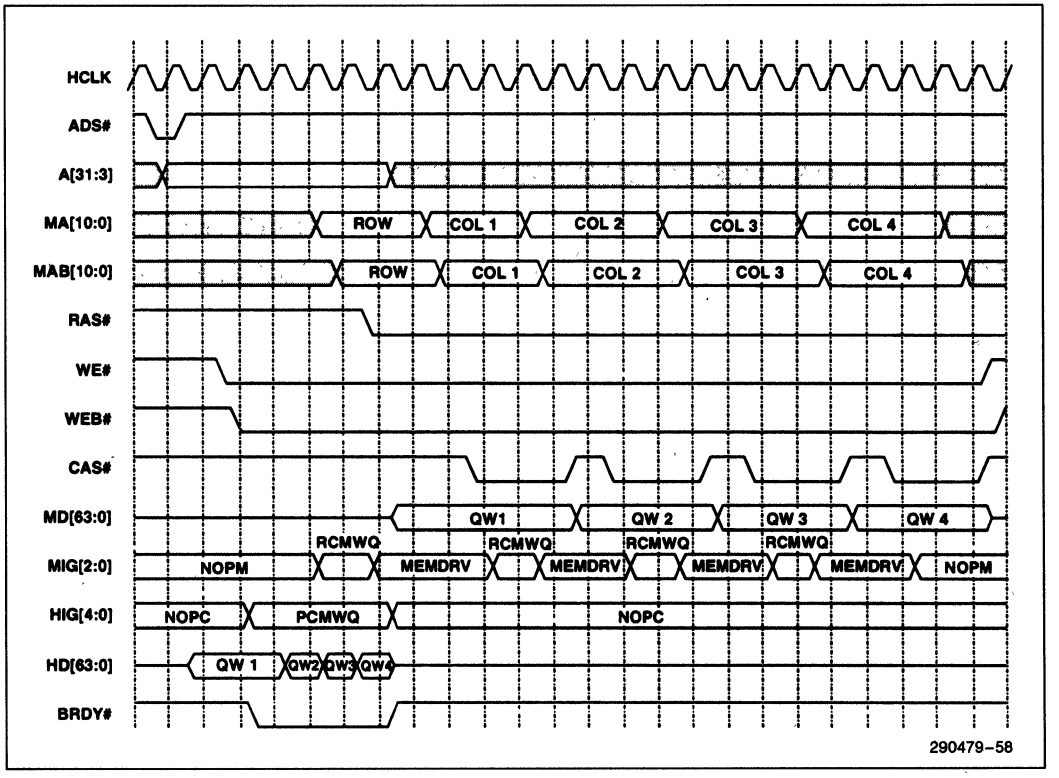


Figure 49. Burst DRAM Write Cycle-Row Miss

6.1.4.7 Read Cycle, 0-Active RAS# Mode

When in 0-active RAS# mode, every CPU cycle to DRAM results in a RAS# and CAS# sequence. RAS# is always negated after a cycle completes. Figure 50 depicts a CPU burst read cycle from DRAM where the 82434LX is configured for 0-active RAS# mode. When in 0-active RAS# mode, the PCMC defaults to driving the row address on the MA[10:0] lines. The PCMC asserts the RAS# signal for the currently decoded row causing the DRAMs to latch the row address. The PCMC then switches the MA[10:0] lines to drive the column address and asserts CAS[7:0]#. CMR (CPU Memory Read) is driven on the HIG[4:0] lines to enable the memory data to host data path through the LBXs. The PCMC advances the MA[1:0] lines through the Pentium processor burst order, negating and asserting CAS[7:0]# to read each Qword. The host data is latched on the falling edge of MDLE, when CAS[7:0]# are negated. The latch is opened again when MDLE is sampled asserted by the LBXs. The LBXs tri-state the host data bus when HIG[4:0] change to NOPC and MDLE rises. A single read row miss from DRAM is similar to the first read of this sequence. The HIG[4:0] lines are driven to NOPC when BRDY# is asserted. RAS# is negated with CAS[7:0]#.

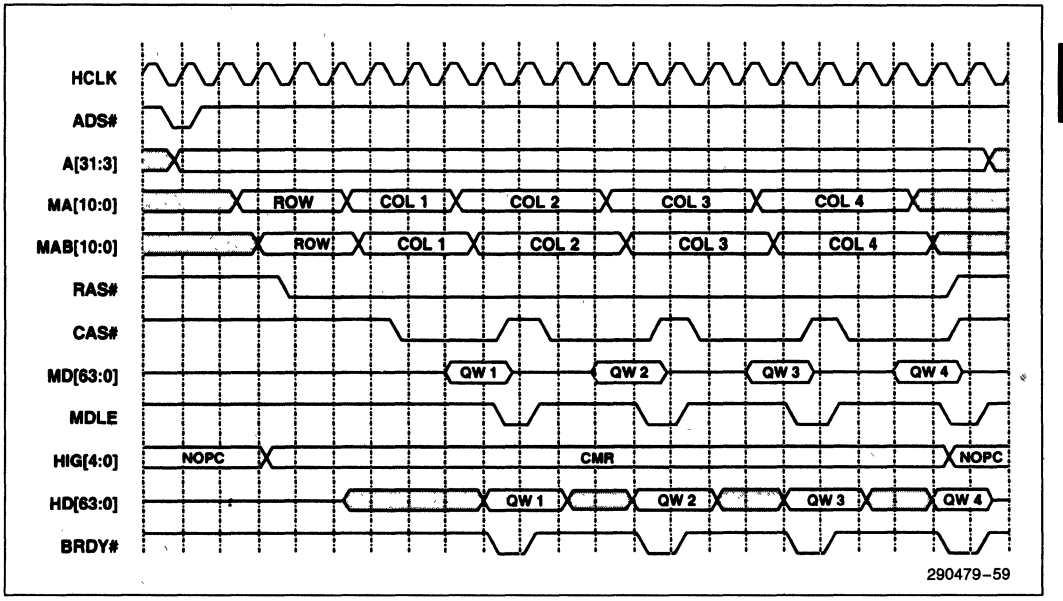


Figure 50. Burst DRAM Read Cycle, 0-Active RAS# Mode

6.1.4.8 Write Cycle, 0-Active RAS# Mode

When in 0-active RAS# mode, every CPU cycle to DRAM results in a RAS# and CAS# sequence. RAS# is always negated after a cycle completes. Figure 51 depicts a CPU Burst Write Cycle to DRAM where the 82434LX is configured for 0-active RAS# mode. The HIG[4:0] lines are driven to PCMWQ to post the write to the LBXs. In the figure, the write cycle is posted to the CPU-to-Memory Posted Write Buffer at 4-1-1-1. The write is posted at 4-1-1-1 when the second level cache is configured for a write-back policy. The write is posted to DRAM at 3-1-1-1 when the second level cache is configured for a write-through policy. When in 0-active RAS# mode, the PCMC defaults to driving the row address on the MA[10:0] lines. The PCMC asserts the RAS# signal for the currently decoded row causing the DRAMs to latch the row address. The PCMC asserts WE# and drives the RCMWQ command on MIG[2:0] to enable the LBXs to drive the first Qword of the write onto the memory data lines. MEMDRV is then driven to cause the LBXs to continue to drive the first Qword. The PCMC then switches the MA[10:0] lines to the column address and asserts CAS[7:0]# to initiate the first write. CAS[7:0]# are then negated and asserted to perform the writes to the DRAMs as the MA[1:0] lines advance through the Pentium processor burst order. A single write is similar to the first write of the burst sequence. MIG[2:0] are driven to NOPM in the clock after CAS[7:0] are asserted.

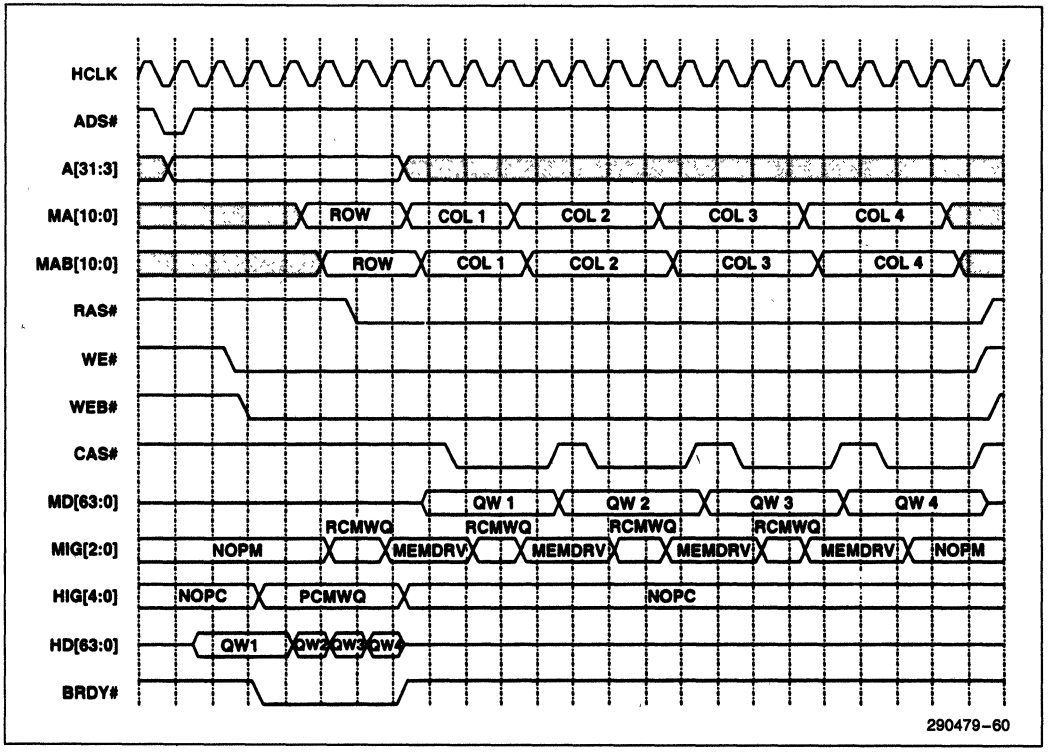


Figure 51. Burst DRAM Write Cycle, 0-Active RAS# Mode

6.1.5 REFRESH

The refresh of the DRAM array can be performed by either using RAS#-only or CAS#-before-RAS# refresh cycles. When programmed for CAS#-before-RAS# refresh, hidden refresh cycles are initiated when possible. RAS# only refresh can be used with any type of second level cache configuration (i.e., no second level cache is present, or either a burst SRAM or standard SRAM second level cache is implemented). CAS#-before-RAS# refresh can be enabled when either no second level cache is present or a burst SRAM second level cache is implemented. CAS#-before-RAS# refresh should not be used when a standard SRAM second level cache is implemented. The timing of internally generated refresh cycles is derived from HCLK and is independent of any expansion bus refresh cycles.

The DRAM controller contains an internal refresh timer which periodically requests the refresh control logic to perform either a single refresh or a burst of four refreshes. The single refresh interval is 15.6 μs. The interval for burst of four refreshes is four times the single refresh interval, or 62.4 μs. The PCMC is configured for either single or burst of four refresh and either RAS#-only or CAS#-before-RAS# refresh via the DRAM Control Register (offset 57h).

To minimize performance impact, refresh cycles are partially deferred until the DRAM interface is idle. The deferment of refresh cycles is limited by the DRAM maximum RAS# low time of 100 μs. Refresh cycles are initiated such that the RAS# maximum low time is never violated.

Hidden refresh cycles are run whenever all eight CAS# lines are active when the refresh cycle is internally requested. Normal CAS#-before-RAS# refresh cycles are run whenever the DRAM interface is idle when the refresh is requested, or when any subset of the CAS# lines is inactive as the refresh is internally requested.

To minimize the power surge associated with refreshing a large DRAM array the DRAM interface staggers the assertion of the RAS# signals during both CAS#-before-RAS# and RAS#-only refresh cycles. The order of RAS# edges is dependent on which RAS# was most recently asserted prior to the refresh sequence. The RAS# that was active will be the last to be activated during the refresh sequence. All RAS[5:0]# lines are negated at the end of refresh cycles, thus, the first DRAM cycle after a refresh sequence is a row miss.

6.1.5.1 RAS#-Only Refresh-Single

Figure 52 depicts a RAS#-only refresh cycle when the 82434LX is programmed for single refresh cycles. The diagram shows a CPU read cycle completing as the refresh timing inside the PCMC generates a refresh request. The refresh address is driven on the MA[10:0] lines. Since the CPU cycle was to row 0, RAS0# is negated. RAS1# is the first to be asserted. RAS2# through RAS5# are then asserted sequentially while RAS0# is driven high, precharging the DRAMs in row 0. RAS0# is then asserted after RAS5#. Each RAS# line is asserted for six host clocks.

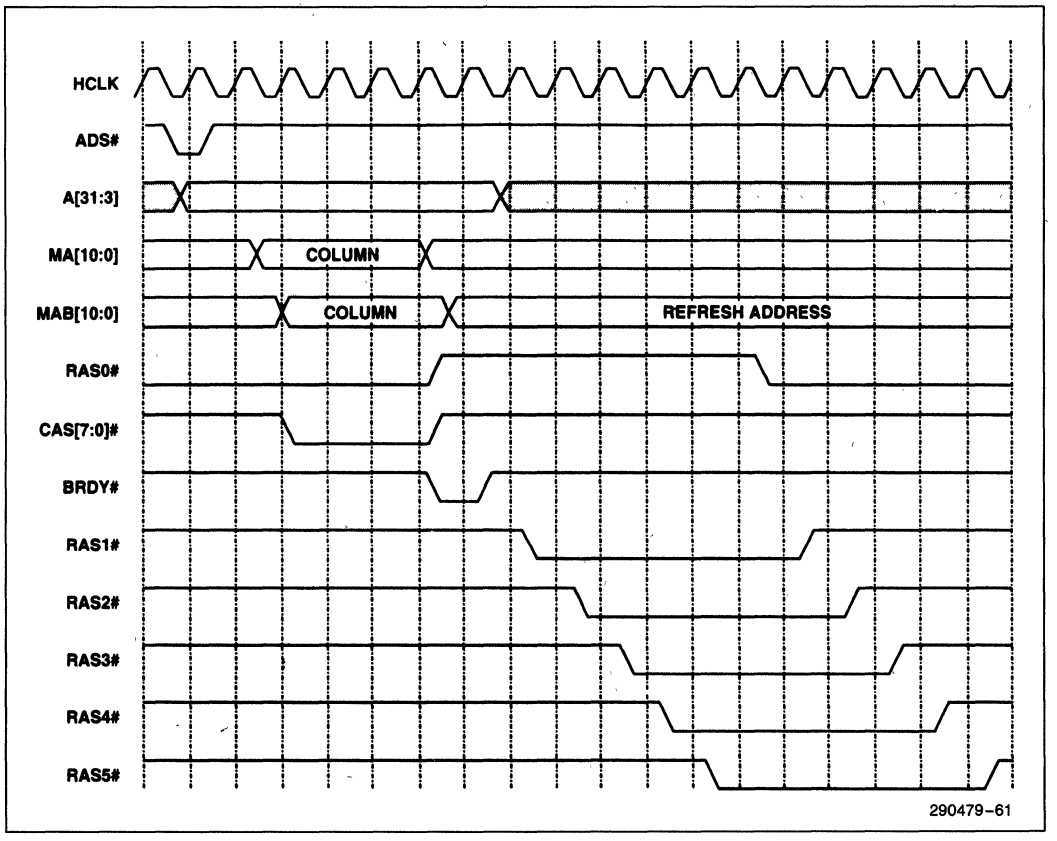


Figure 52. RAS# Only Refresh-Single

6.1.5.2 CAS#-before-RAS# Refresh-Single

Figure 53 depicts a CAS#-before-RAS# refresh cycle when the 82434LX is programmed for single refresh cycles. The diagram shows a CPU read cycle completing as the refresh timing inside the PCMC generates a refresh request. The CPU read cycle is less than a Qword, therefore a hidden refresh is not initiated. After the CPU read cycle completes, all of the RAS# and CAS# lines are negated. The PCMC then asserts CAS[7:0]# and then sequentially asserts the RAS# lines, starting with RAS1# since RAS0# was the last RAS# line asserted. Each RAS# line is asserted for six clocks.

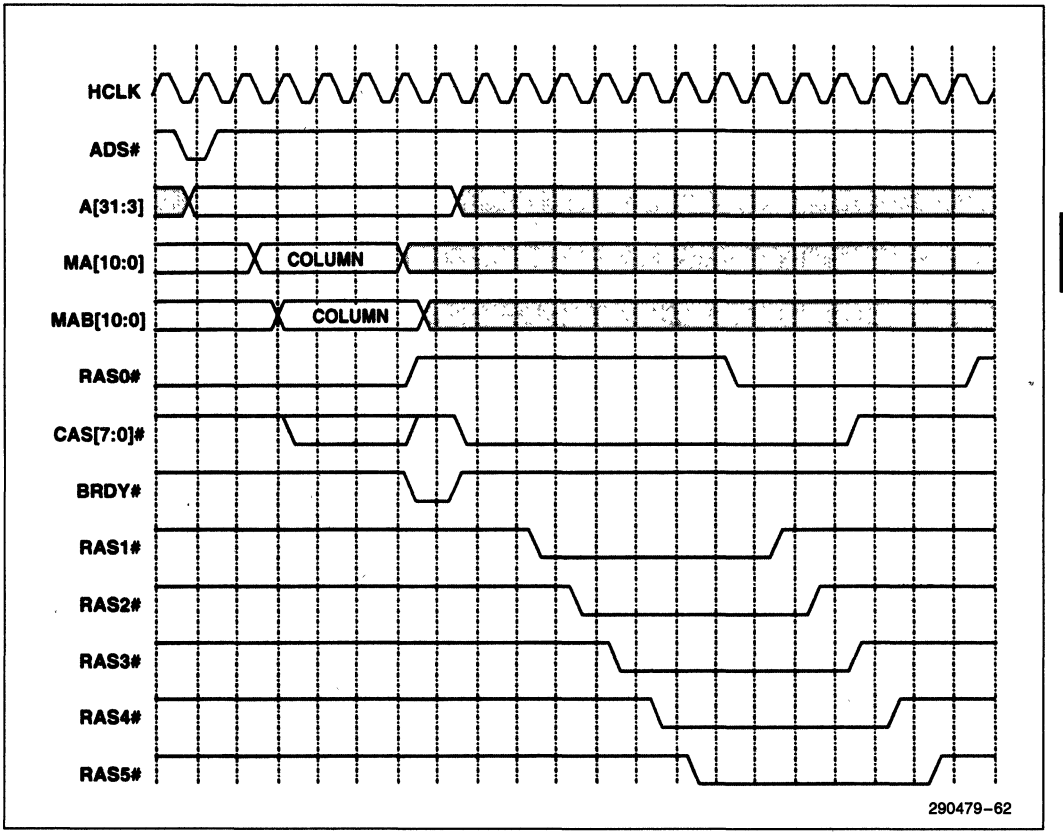


Figure 53. CAS#-before-RAS# Refresh-Single

6.1.5.3 Hidden Refresh-Single

Figure 54 depicts a hidden refresh cycle which takes place after a DRAM read page hit cycle. The diagram shows a CPU read cycle completing as the refresh timing inside the 82434LX generates a refresh request. The CPU read cycle is an entire Qword, therefore a hidden refresh is initiated. After the CPU read cycle completes, RAS# is negated, but all eight CAS# lines remain asserted. The PCMC then sequentially asserts the RAS# lines, starting with RAS1# since RAS0# was the last active RAS# line. Each RAS# line is asserted for six clocks.

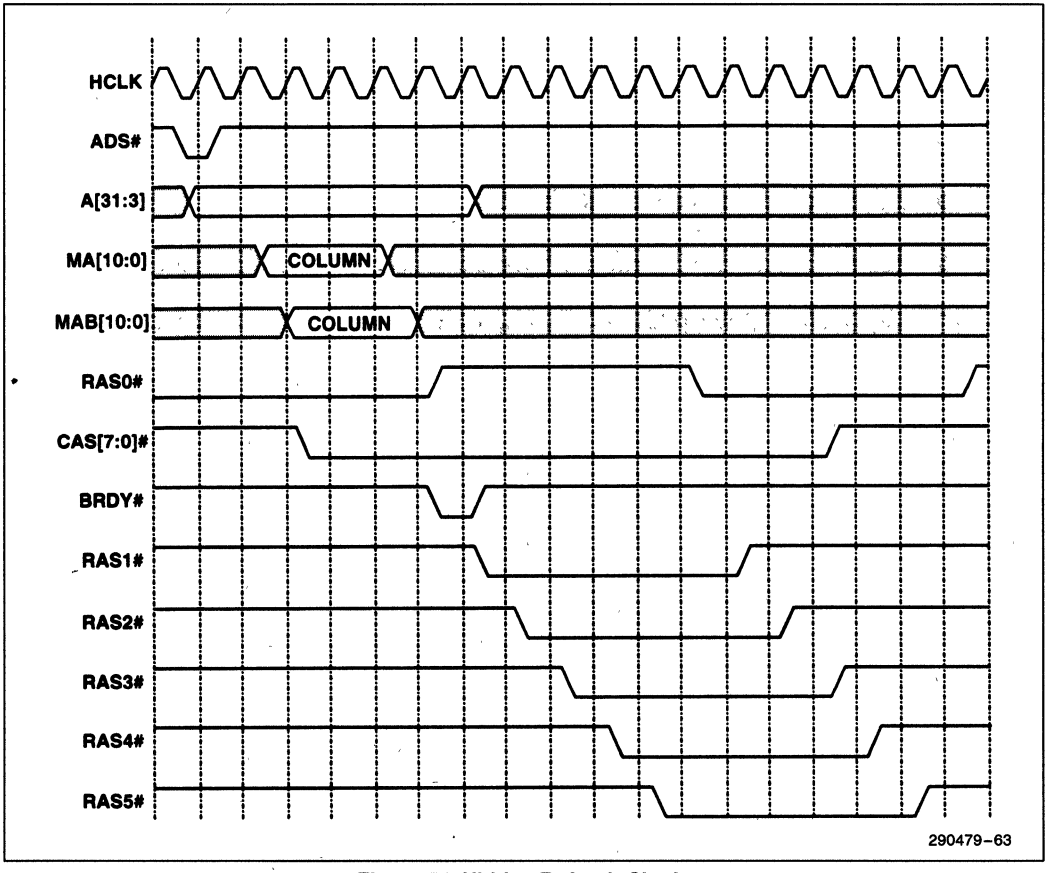


Figure 54. Hidden Refresh-Single

6.2 82434NX DRAM Interface

This section describes the 82434NX DRAM interface. Changes in the 82430NX PCIset from the 82430 PCIset include:

1. Increased maximum DRAM memory size to 512 MBytes. The 82430NX PCIset increases the maximum memory array size from 192 MBytes to 512 MBytes.
2. Two additional row address lines (RAS[7:6]#) for a total of eight (RAS[7:0]#).
3. Addition of 50 MHz host-bus optimized DRAM timing sets.
4. Three additional registers are added to support the increased memory sizeDRAM Row Boundary Registers 6 and 7 (DRB[7:6]) and the DRAM Row Boundary Extension (DRBE) Register.
5. Modified MA[11:0] timing to provide more MA[11:0] setup time to CAS[7:0]# assertion.

6.2.1 DRAM ADDRESS TRANSLATION

The MA[11:0] lines are translated from the host address lines A[26:3] for all memory accesses, except those targeted to memory that has been remapped as a result of the creation of a memory space gap in the lower extended memory area. In the case of a cycle targeting remapped memory, the least significant bits come directly from the host address, while the more significant bits depend on the memory space gap start address, gap size, and the size of main memory.

2

Table 15. DRAM Address Translation

Memory Address MA[11:0]	11	10	9	8	7	6	5	4	3	2	1	0
Column Address	A25	A23	A21	A11	A10	A9	A8	A7	A6	A5	A4	A3
Row Address	A26	A24	A22	A20	A19	A18	A17	A16	A15	A14	A13	A12

6.2.2 CYCLE TIMING SUMMARY

The 82434NX PCMC DRAM performance for 50 MHz Host bus clock is summarized in Table 13 for all CPU read and write cycles. The 60/66 MHz MA[11:0] timings when in X-4-4-4 mode have one difference from the 82434LX MA[11:0] timings. The MA lines switch to the next address in the burst sequence one clock sooner than in the 82434LX, providing more MA[11:0] setup time to CAS[7:0]# assertion. The 60/66 MHz DRAM timings for write cycles have been improved by 1 clock for all leadoffs. The 50 MHz timings shown below are selected by HOF=00, DBT=11, RWS=0 and CWS=0.

Table 16. CPU to DRAM Performance Summary for 50 MHz Host Bus Clock

Cycle Type	x-3-3-3 Timing(1)
Read (Page Hit/Row Miss/ Page Miss)	6/10/12-3-3-3
Posted Write	4-1-1-1
Write (Page Hit/Row Miss/ Page Miss)	10/11/13-3-3-3
0-Active RAS# Mode Reads	9-3-3-3
0-Active RAS# Mode Writes	9-3-3-3

NOTES:
1. Single cycle timings are identical to these leadoff timings.

Table 17. Refresh Cycle Performance (Independent of CPU frequency)

Refresh Type	Hidden Refresh	RAS# Only Refresh	CAS#-Before-RAS#
Single	16	17	18
Burst of Four	64	68	72

6.2.3 CPU TO DRAM BUS CYCLES

In this section, all timing diagrams are for 50 MHz DRAM timing, 1-Active RAS mode. The 60/66 MHz MA[11:0] timings when in X-4-4-4 mode have one difference from the 82434LX MA[11:0] timings. The MA lines switch to the next address in the burst sequence one clock sooner than in the 82434LX. The write cycle leadoffs are 1 clock earlier for 82430NX than 82430 (the MIGs and CAS timings improved by 1 clock). The 0-Active RAS# modes closely resemble the row miss cases. In 0-Active RAS# mode, RAS# is asserted one clock sooner than is shown in the row miss timing diagrams.

6.2.3.1 Burst DRAM Read Page Hit

Figure 55 depicts a CPU burst read page hit to DRAM. The 82434NX decodes the CPU address as a page hit and drives the column address onto the MA[11:0] lines. CAS[7:0]# are then asserted for two CLKs and negated for one CLK. CMR (CPU Memory Read) is driven on the HIG[4:0] lines to enable the memory data to host data path through the LBXs. The PCMC advances the MA[1:0] lines through the processor burst order, negating and asserting CAS[7:0]# to read each Qword. The MD[63:0] data is sampled with HCLK in the LBXs when MDLE is asserted, and driven on the host bus the following cycle to meet the setup time of the CPU. BRDY# is then asserted. When MDLE is negated, the LBX continues to drive the latched HD[63:0] to ensure that the data hold time to CWE[7:0]# is met for standard SRAMs. The LBXs tri-state the host data bus when HIG[4:0] change to NOPC and MDLE rises. A single read page hit from DRAM is similar to the first read of this sequence. The HIG[4:0] lines are driven to NOPC when the last BRDY# is asserted.

The diagram also shows the typical control signal timing for a burst SRAM line fill operation. Note that CCS# inactive will mask any new ADS# (caused by the NA# assertion) to the burst SRAMs.

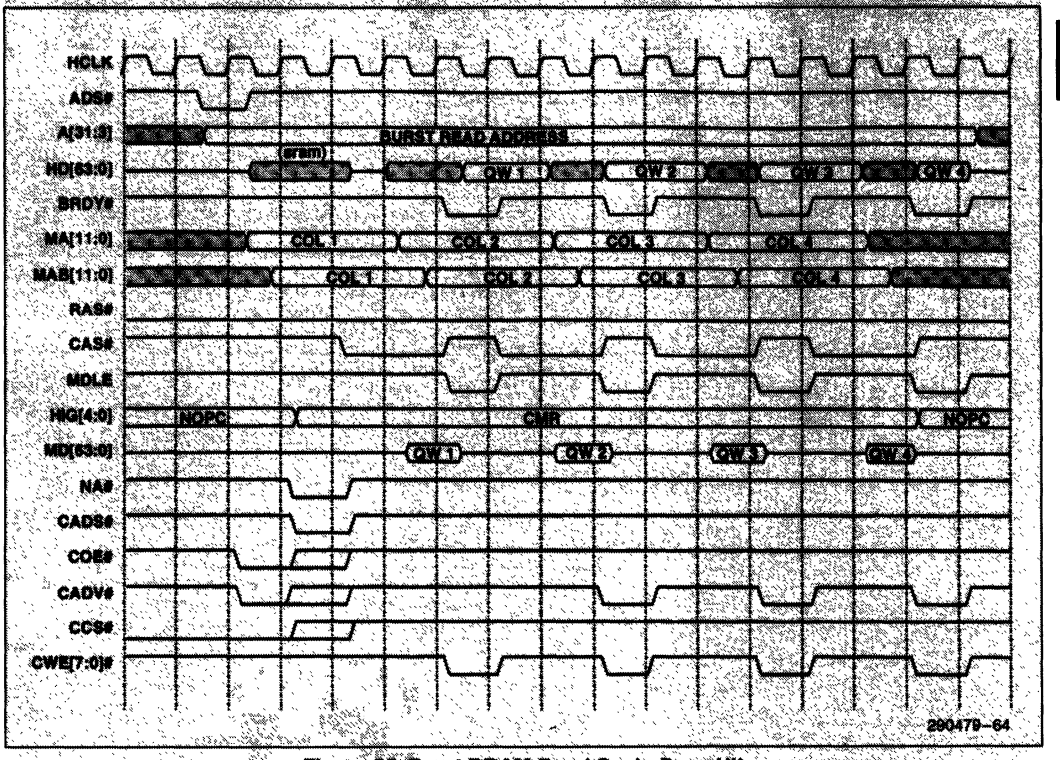


Figure 55. Burst DRAM Read Cycle-Page Hit

2

6.2.3.2 Burst DRAM Read Page Miss

Figure 56 depicts a CPU to DRAM burst read page miss cycle. The 82434NX decodes the CPU address as a page miss and switches from initially driving the column address to driving the row address on the MA[11:0] lines. RAS# is then negated to precharge the DRAMs and then asserted to latch the new DRAM row address. The PCMC then switches the MA[11:0] lines to drive the column address and asserts CAS[7:0]#. CMR (CPU Memory Read) is driven on the HIG[4:0] lines to enable the memory data to host data path through the LBXs. The PCMC advances the MA[1:0] lines through the microprocessor burst order, negating and asserting CAS[7:0]# to read each Qword. The MD[63:0] data is sampled with HCLK in the LBXs when MDLE is asserted, and driven on the host bus the following cycle to meet the setup time of the CPU. BRDY# is then asserted. When MDLE is negated, the LBX continues to drive the latched HD[63:0] to ensure that the data hold time to CWE[7:0]# is met for standard SRAMs. The LBXs tri-state the host data bus when HIG[4:0] change to NOPC and MDLE rises. The HIG[4:0] lines are driven to NOPC when the last BRDY# is asserted.

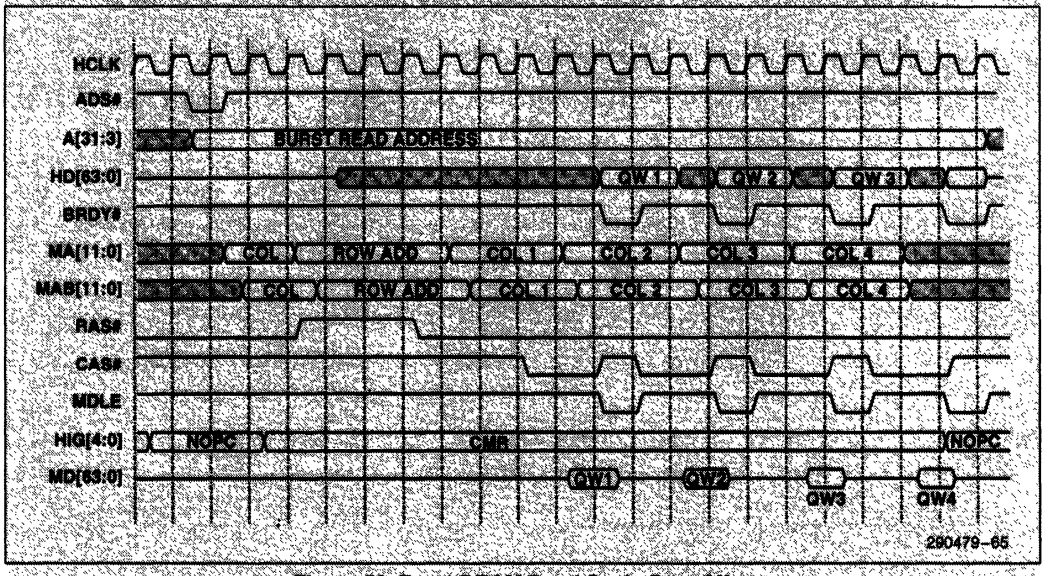


Figure 56. Burst DRAM Read Cycle-Page Miss

6.2.3.3 Burst DRAM Read Row Miss

Figure 57 depicts a CPU to DRAM burst read row miss cycle. The 82434NX decodes the CPU address as a row miss and switches from initially driving the column address to driving the row address on the MA[11:0] lines. The RAS# signal that was asserted is negated and the RAS# for the currently accessed row is asserted (RAS# is asserted 1 clock earlier in 0-Active RAS# Mode.) The PCMC then switches the MA[11:0] lines to drive the column address and asserts CAS[7:0]#. CMR (CPU Memory Read) is driven on the HIG[4:0] lines to enable the memory data to host data path through the LBXs. The PCMC advances the MA[1:0] lines through the microprocessor burst order, negating and asserting CAS[7:0]# to read each Qword. The MD[63:0] data is sampled with HCLK in the LBXs when MDLE is asserted, and driven on the host bus the following cycle to meet the setup time of the CPU. BRDY# is then asserted. When MDLE is negated, the LBX continues to drive the latched HD[63:0] to ensure that the data hold time to CWE[7:0]# is met for standard SRAMs. The LBXs tri-state the host data bus when HIG[4:0] change to NOPC and MDLE rises. A single read row miss from DRAM is similar to the first read of this sequence. The HIG[4:0] lines are driven to NOPC when the last BRDY# is asserted.

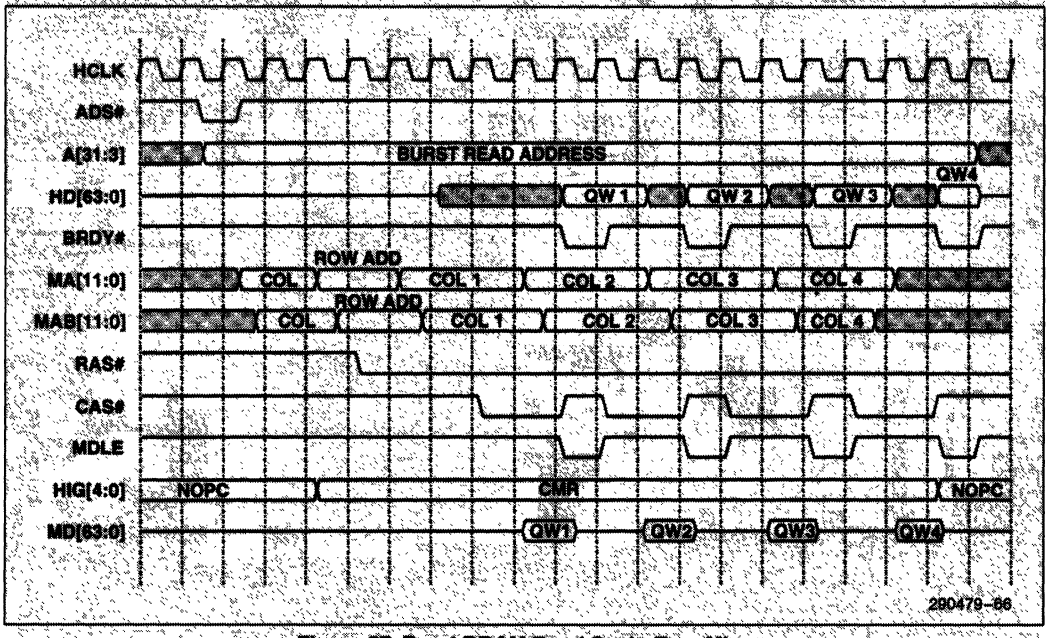


Figure 57. Burst DRAM Read Cycle-Row Miss

2

6.2.3.4 Burst DRAM Write Page Hit

Figure 58 depicts a CPU burst write page hit to DRAM. The 82434NX decodes the CPU write cycle as a DRAM page hit. The HIG[4:0] lines are driven to PCMWQ to post the write to the LBXs. In the figure, the write cycle is posted to the CPU-to-Memory Posted Write Buffer at 3-1-1-1. When the cycle is decoded as a page hit, the PCMC asserts WE# and drives the RCMWQ command on MIG[2:0] to enable the LBXs to drive the first Qword of the write onto the memory data lines. MEMDRV is then driven to cause the LBXs to continue to drive the first Qword for two more clocks. CAS[7:0]# are then negated and asserted to perform the writes to the DRAMs as the MA[1:0] lines advance through the Pentium processor burst order. A single write is similar to the first write of the burst sequence. The MIG[2:0] lines are driven to NOPM in the clock when the last CAS[7:0]# are asserted.

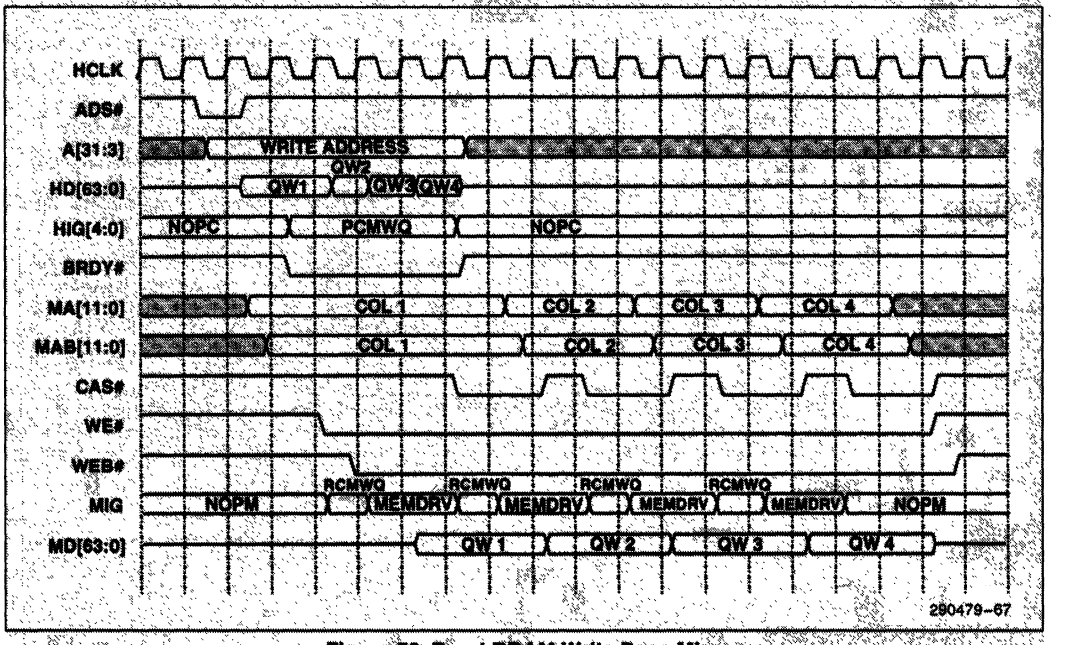


Figure 58. Burst DRAM Write Page Miss

6.2.3.5 Burst DRAM Write Page Miss

Figure 59 depicts a CPU burst write page miss to DRAM. The 82434NX decodes the CPU write cycle as a DRAM page miss and drives the PCMWQ command [HIG[4:0] lines] to post the write data to the LBXs. In the figure, the write cycle is posted to the CPU-to-Memory Posted Write Buffer at 3-1-1-1. When the cycle is decoded as a page miss, the PCMC switches the MA[11:0] lines from the column address to the row address and asserts WE# in clock 4. The PCMC drives the RCMWQ command on MIG[2:0] to enable the LBXs to drive the first Qword of the write onto the memory data lines. MEMDRV is then driven to cause the LBXs to continue to drive the first Qword. The RAS# signal for the currently decoded row is negated to precharge the DRAMs. RAS# is then asserted to cause the DRAMs to latch the row address. The PCMC then switches the MA[11:0] lines to the column address and asserts CAS[7:0]# to initiate the first write. CAS[7:0]# are then negated and asserted to perform the writes to the DRAMs as the MA[1:0] lines advance through the Pentium processor burst order. A single write is similar to the first write of the burst sequence. The MIG[2:0] lines are driven to NOPM in the clock when the last CAS[7:0]# are asserted.

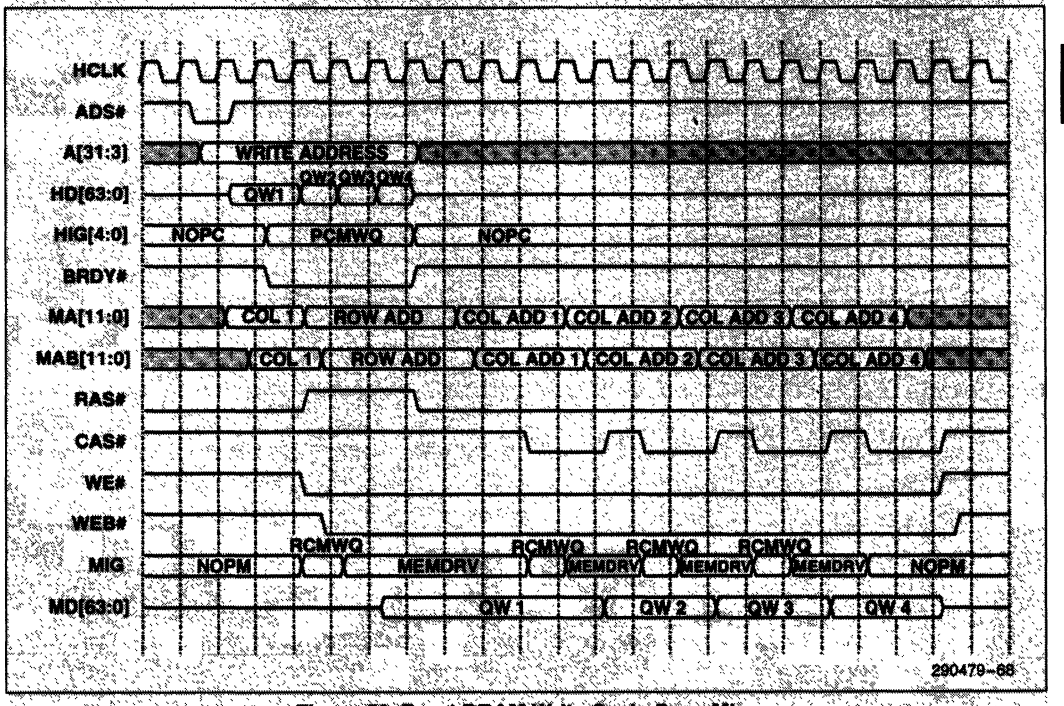


Figure 59. Burst DRAM Write Cycle-Page Miss

6.2.3.6 Burst DRAM Write Row Miss

Figure 60 depicts a CPU burst write row miss to DRAM. The 82434NX decodes the CPU write cycle as a DRAM row miss and the HIG[4:0] lines are driven to PCMWQ to post the write data into LBXs. When the cycle is decoded as a row miss, the PCMC negates the already active RAS# signal, switches the MA[11:0] lines from the column address to the row address and asserts the RAS# signal for the currently decoded row. The PCMC asserts WE# and drives the RCMWQ command on MIG[2:0] to enable the LBXs to drive the first Qword of the write onto the memory data lines. MEMDRV is then driven to cause the LBXs to continue to drive the first Qword. The PCMC then switches the MA[11:0] lines to the column address and asserts CAS[7:0]# to initiate the first write. CAS[7:0]# are then negated and asserted to perform the writes to the DRAMs as the MA[1:0] lines advance through the microprocessor burst order. A single write is similar to the first write of the burst sequence. The MIG[2:0] lines are driven to NOPM in the clock when the last CAS[7:0]# are asserted.

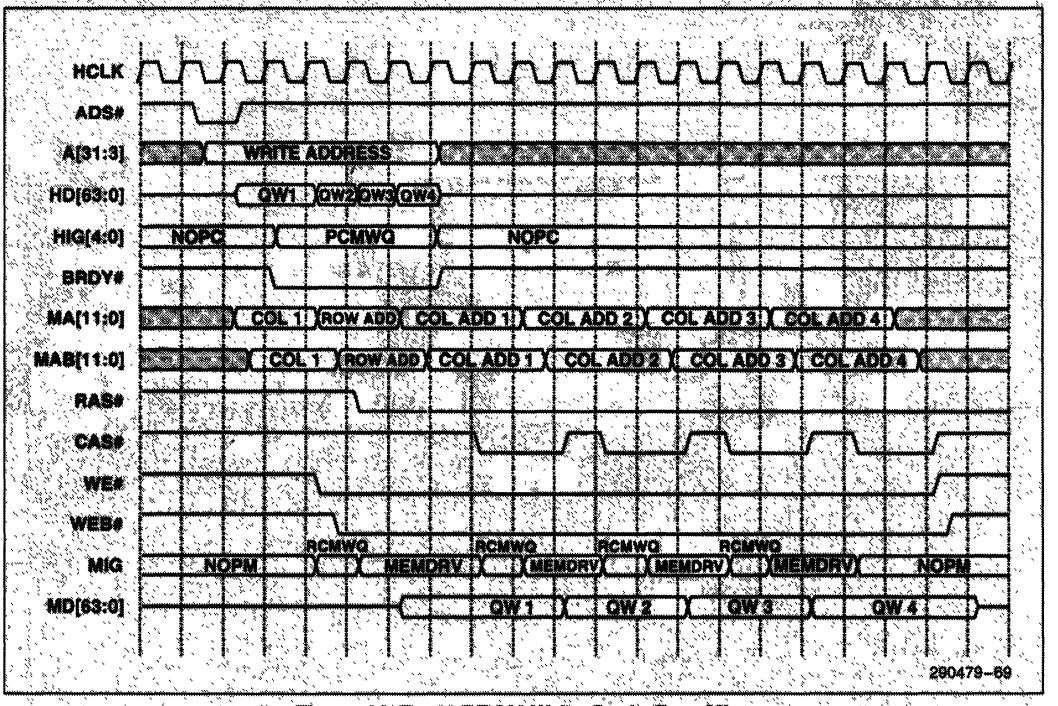


Figure 60. Burst DRAM Write Cycle-Row Miss

6.2.4 REFRESH

The refresh of the DRAM array can be performed by either using RAS#-only or CAS# before-RAS# refresh cycles. When programmed for CAS#-before-RAS# refresh, hidden refresh cycles are initiated when possible. The timing of internally generated refresh cycles is derived from HCLK and is independent of any expansion bus refresh cycles.

The DRAM controller contains an internal refresh timer which periodically requests the refresh control logic to perform either a single refresh or a burst of four refreshes. The single refresh interval is 15.6 µs. The interval for burst of four refreshes is four times the single refresh interval, or 62.4 µs. The PCMC is configured for either single or burst of four refresh and either RAS#-only or CAS#-before-RAS# refresh via the DRAM Control Register (offset 57h).

To minimize performance impact, refresh cycles are partially deferred until the DRAM interface is idle. Refresh cycles are initiated such that the RAS# maximum active time is never violated.

Hidden refresh cycles are run whenever all eight CAS# lines are active at the end of a read transaction when the refresh cycle is internally requested. Normal CAS#-before-RAS# refresh cycles are run whenever the DRAM interface is idle when the refresh is requested, or when any subset of the CAS# lines is inactive as the refresh is internally requested.

To minimize the power surge for refreshing a large DRAM array, the DRAM interface staggers the assertion and negation of the RAS# signals during both CAS#-before-RAS# and RAS#-only refresh cycles. The order of RAS# edges is dependent on which RAS# was most recently asserted prior to the refresh sequence. The RAS# that was active will be the last to be activated during the refresh sequence. All RAS[7:0]# lines are negated at the end of refresh cycles, making the first DRAM cycle after a refresh sequence a row miss.

6.2.4.1 RAS#-Only Refresh—Single

Figure 61 depicts a RAS#-only refresh cycle when the 82434NX is programmed for single refresh cycles. The diagram shows a cycle completing as the refresh timer inside the PCMC generates a refresh request. The refresh address is driven on the MA[11:0] lines. Since the cycle was to row 0, RAS0# is negated. RAS1# is the first to be asserted. RAS2# through RAS7# are then asserted sequentially while RAS0# is driven high, precharging the DRAMs in row 0. RAS0# is then asserted after RAS7#. Each RAS# line is asserted for eight host clocks.

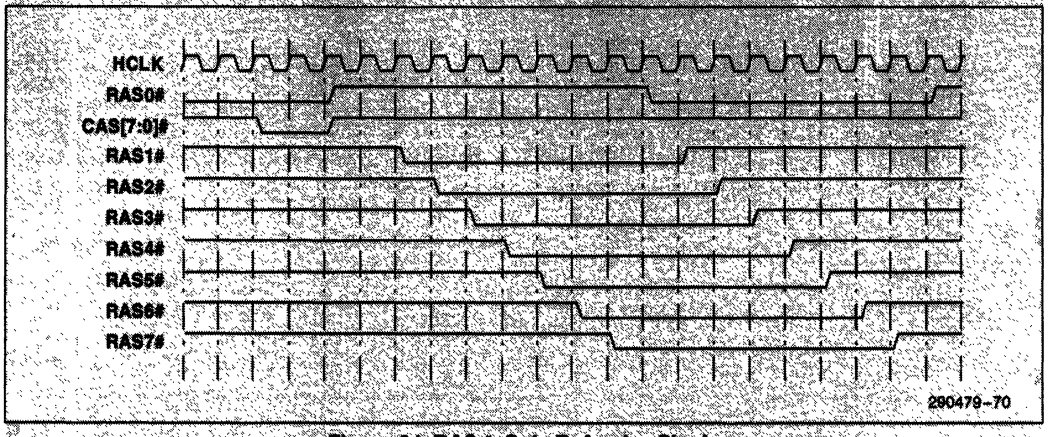


Figure 61. RAS#-Only Refresh—Single

6.2.4.2 CAS#-before-RAS# Refresh—Single

Figure 62 depicts a CAS#-before-RAS# refresh cycle when the 82434NX is programmed for single refresh cycles. The diagram shows a write cycle completing as the refresh timer inside the PCMC generates a refresh request. The cycle is less than a Qword, therefore a hidden refresh is not initiated. After the cycle completes, all of the RAS# and CAS# lines are negated. The PCMC then asserts CAS[7:0]# and then sequentially asserts the RAS# lines, starting with RAS1# since RAS0# was the last RAS# line asserted. Each RAS# line is asserted for eight clocks.

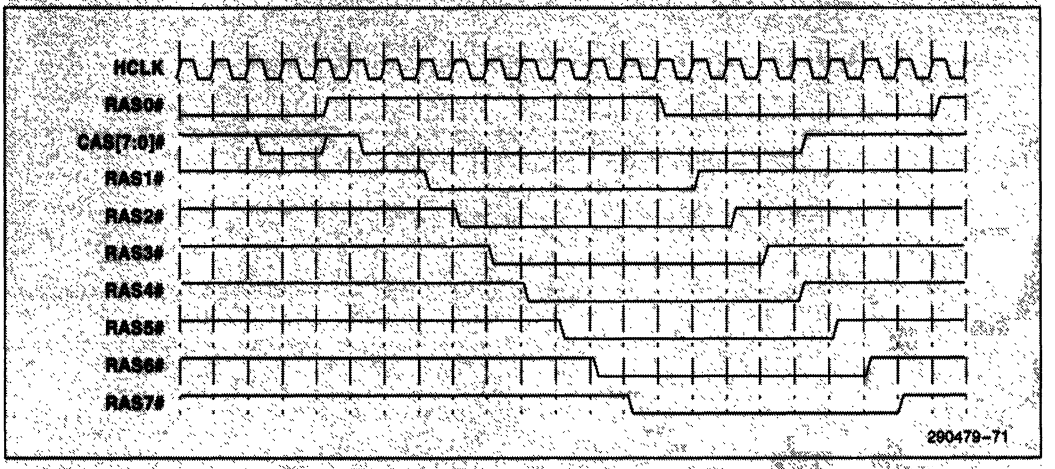


Figure 62. CAS#-Before-RAS# Refresh—Single

6.2.4.3 Hidden Refresh-Single

Figure 63 depicts a hidden refresh cycle which takes place after a DRAM read page hit cycle. The diagram shows a read cycle completing as the refresh timing inside the 82434NX PCMC generates a refresh request. The cycle is an entire Qword, therefore, a hidden refresh is initiated. After the cycle completes, RAS# is negated, but all eight CAS# lines remain asserted. The PCMC then sequentially asserts the RAS# lines, starting with RAS1# since RAS0# was the last active RAS# line. Each RAS# line is asserted for eight clocks.

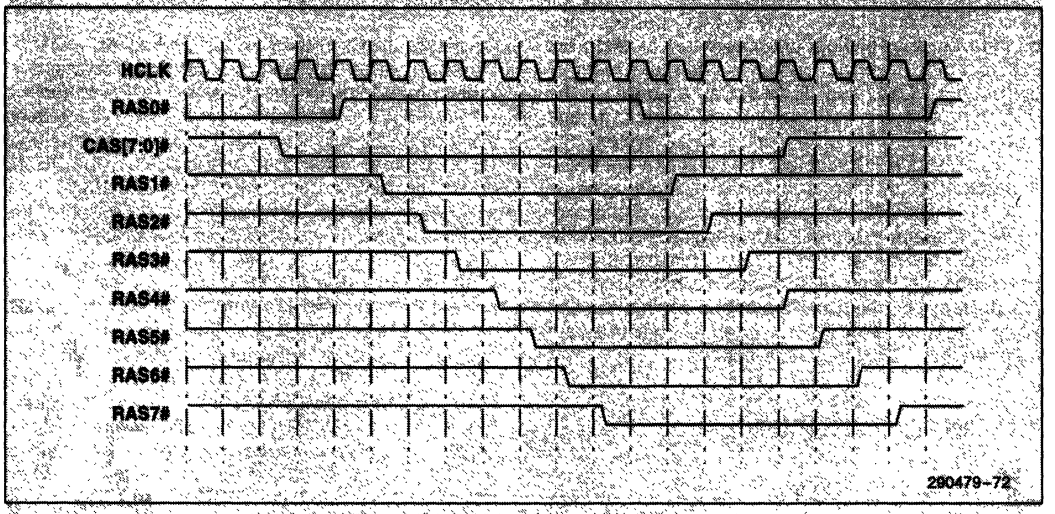


Figure 63. Hidden Refresh—Single

7.0 PCI INTERFACE

The description in this section applies to both the 82434LX and 82434NX.

7.1 PCI Interface Overview

The PCMC and LBXs form a high performance bridge from the Pentium processor to PCI and from PCI to main memory. During PCI-to-main memory cycles, the PCMC and LBXs act as a target on the PCI Bus, allowing PCI masters to read from and write to main memory. During CPU cycles, the PCMC acts as a PCI master. The CPU can then read and write I/O, memory and configuration spaces on PCI. When the CPU accesses I/O mapped and configuration space mapped PCMC registers, the PCMC intercepts the cycles and does not forward them to PCI. Although these CPU cycles do not result in a PCI bus cycle, they are described in this section since most of the PCMC internal registers are mapped into PCI configuration space.

7.2 CPU-to-PCI Cycles

7.2.1 CPU WRITE TO PCI

Figure 64 depicts a series of CPU memory writes which are posted to PCI. The CPU initiates the cycles by asserting ADS# and driving the memory address onto the host address lines. The PCMC asserts NA# in the clock after ADS# allowing the Pentium processor to drive another cycle onto the host bus two clocks later. The PCMC decodes the memory address and drives PCPWL on the HIG[4:0] lines, posting the host address bus and the low Dword of the data bus to the LBXs. The PCMC asserts BRDY#, terminating the CPU cycle with one wait state. Since NA# is asserted in the second clock of the first cycle, the Pentium processor does not insert an idle cycle after this cycle completes, but immediately drives the next cycle onto the bus. Thus, the Pentium processor maximum Dword write bandwidth of 89 MBytes/second is achieved during back-to-back Dword writes cycles. Each of the following write cycles is posted to the LBXs in three clocks.

In this example, the PCMC is parked on PCI and therefore, does not need to arbitrate for the bus. When parked, the PCMC drives the SCPA command on the PIG[3:0] lines and asserts DRVPCI, causing the host address lines to be driven on the PCI AD[31:0] lines. After the write is posted, the PCMC drives the DCPWA command on the PIG[3:0] lines to drive the previously posted address onto the AD[31:0] lines. The PCMC then drives DCPWD onto the PIG[3:0] lines, to drive the previously posted write data onto the AD[31:0] lines. As this is occurring on PCI, the second write cycle is being posted on the host bus. In this case, the second write is to a sequential and incrementing address. Thus, the PCMC leaves FRAME# asserted, converting the write cycle into a PCI burst cycle. The PCMC continues to drive the DCPWD command on the PIG[3:0] lines. The LBXs advance the posted write buffer pointer to point to the next posted Dword when DCPWD is sampled on PIG[3:0] and TRDY# is sampled asserted. Therefore, if the target inserts a wait-state by negating TRDY#, the LBXs continue to drive the data for the current transfer. The remaining writes are posted on the host bus, while the PCMC and LBXs complete the writes on PCI.

CPU I/O write cycles to PCI differ from the memory write cycle described here in that I/O writes are never posted. BRDY# is asserted to terminate the cycle only after TRDY# is sampled asserted, completing the cycle on PCI.

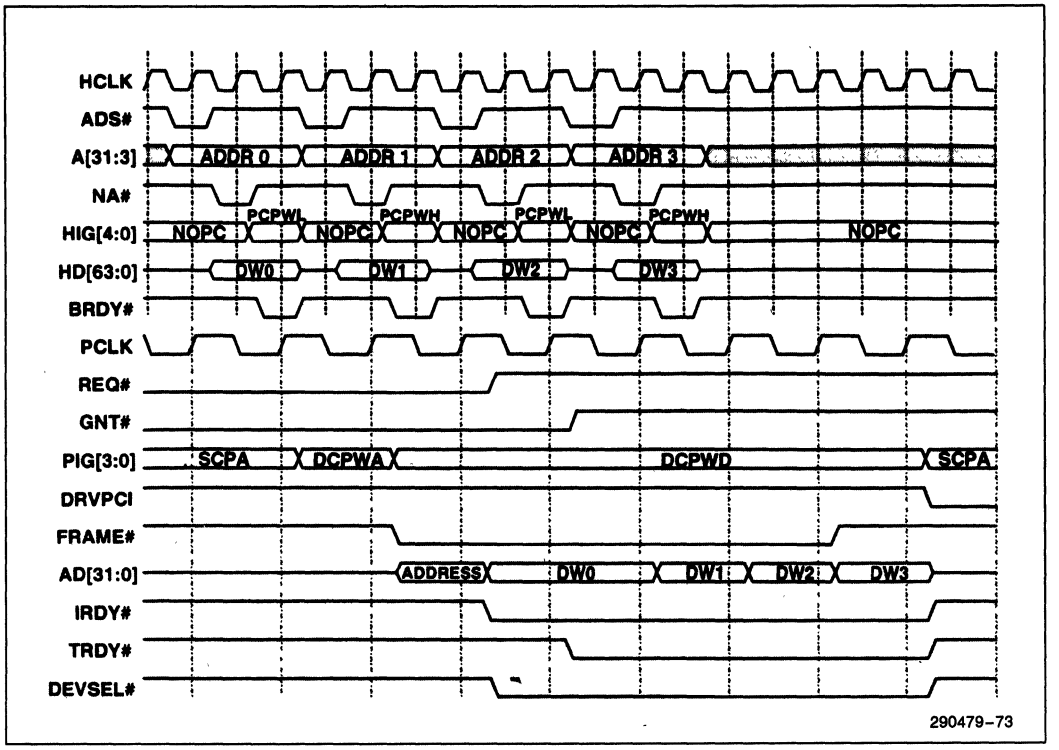


Figure 64. CPU Memory Writes to PCI

7.3 Register Access Cycles

The PCMC contains two registers which are mapped into I/O space, the Configuration Space Enable Register (I/O port CF8h) and the Turbo-Reset Control Register (I/O port CF9h). All other internal PCMC configuration registers are mapped into PCI configuration space. Configuration space must be enabled by writing a non-zero value to the Key field in the CSE Register before accesses to these registers can occur. These registers are mapped to locations C000h through C0FFh in PCI configuration space. If the Key field is programmed with 0h, CPU I/O cycles to locations C000h through CFFFh are forwarded to PCI as ordinary I/O cycles. Externally, accesses to the I/O mapped registers and the configuration space mapped registers use the same bus transfer protocol. Only the PCMC internal decode of the cycle differs. NA# is never asserted during PCMC configuration register or PCI configuration register access cycles. See Section 3.2, PCI Configuration Space Mapped Registers for details on the PCMC configuration space mapping mechanism.

7.3.1 CPU WRITE CYCLE TO PCMC INTERNAL REGISTER

A write to an internal PCMC register (either CSE Register, TRC Register or a configuration space-mapped register) is shown in Figure 65. The cycle begins with the address, byte enables and status signals (W/R#, D/C# and M/IO#) being driven to a valid state indicating an I/O write to either CF8h to access the CSE register, CF9h to access the TRC Register or C0XXh when configuration space is enabled to access a PCMC internal configuration register. The PCMC decodes the cycle and asserts AHOLD to tri-state the CPU address lines. The PCMC signals the LBXs to copy either the upper Dword or the lower Dword of the data bus onto the address lines. The PCMC makes the decision on which Dword to copy based on the BE[7:0]# lines. The HIG[4:0] lines are driven to DACPYH or DACPYL depending on whether the lower Dword of the data bus or the upper Dword of the data bus needs to be copied onto the address bus. The LBXs sample the HIG[4:0] command, and drive the data onto the address lines. The PCMC samples the A[31:0] lines on the second rising edge of HCLK after the LBXs begin driving the data. Finally, the PCMC negates AHOLD and asserts BRDY#, terminating the cycle.

If the write is to the CSE Register and the Key field is programmed to 0000b then configuration space is disabled. If the Key field is programmed to a non-zero value then configuration space is enabled.

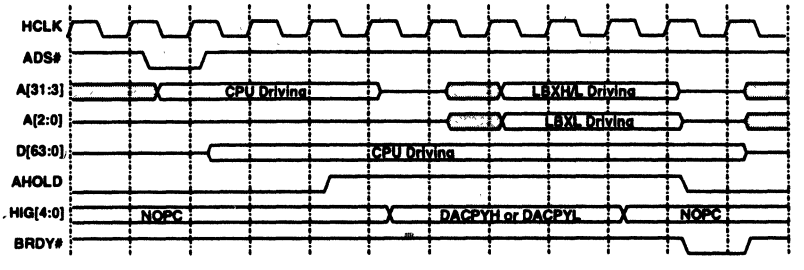


Figure 65. CPU Write to a PCMC Configuration Register

7.3.2 CPU READ FROM PCMC INTERNAL REGISTER

A read from an internal PCMC register (either CSE Register, TRC Register or a configuration space-mapped register) is shown in Figure 66. The I/O read cycle is from either CF8h to access the CSE register, CF9h to access the TRC Register or C0XXh when configuration space is enabled to access a configuration space-mapped register. The PCMC decodes the cycle and asserts AHOLD to tri-state the CPU address lines. The PCMC then drives the contents of the addressed register onto the A[31:0] lines. One byte is enabled on each rising HCLK edge for four consecutive clocks. The PCMC signals the LBXs that the current cycle is a read from an internal PCMC register by issuing the ADCPY command to the LBXs over the HIG[4:0] lines. The LBXs sample the HIG[4:0] command and copy the address lines onto the data lines. Finally, the PCMC negates AHOLD, and asserts BRDY# terminating the cycle.

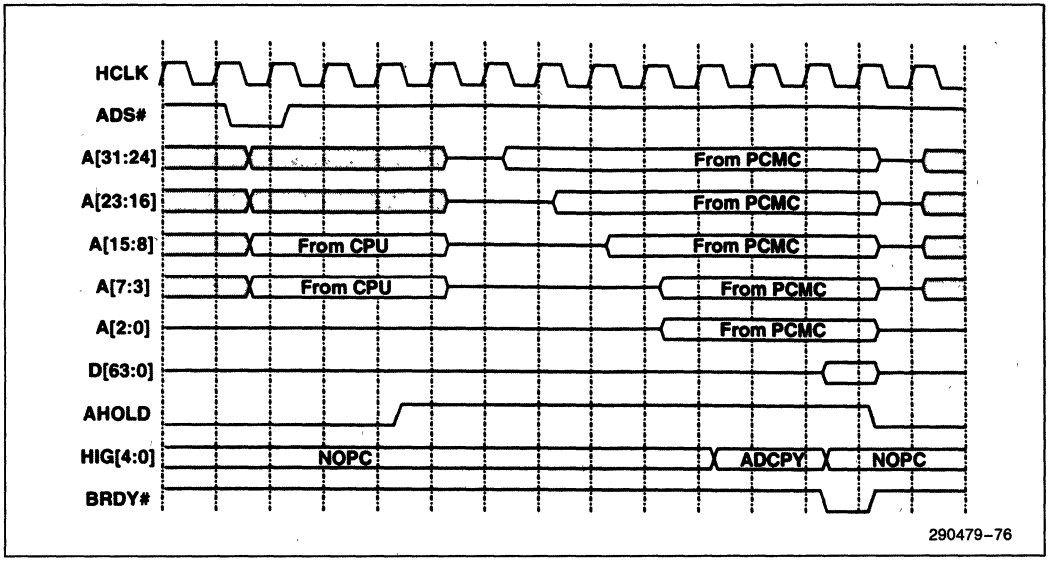


Figure 66. CPU Read from PCMC Configuration Register

7.3.3 CPU WRITE TO PCI DEVICE CONFIGURATION REGISTER

In order to write to or read from a PCI device configuration register the Key field in the CSE register must be programmed to a non-zero value, enabling configuration space. When configuration space is enabled, PCI device configuration registers are accessed by CPU I/O accesses within the range of CnXXh where each PCI device has a unique non-zero value of n. This allows a separate configuration space for each of 15 devices on PCI. Recall that when configuration space is enabled, the PCMC configuration registers are mapped into I/O ports C000h through C0FFh.

A write to a PCI device configuration register is shown in Figure 67. The PCMC internally latches the host address lines and byte enables. The PCMC asserts AHOLD to tri-state the CPU address bus and drives the address lines with the translated address for the PCI configuration cycle. The translation is described in Section 3.2, PCI Configuration Space Mapped Registers. On the HIG[4:0] lines, the PCMC signals the LBXs to latch either the upper Dword of the host data bus or the lower Dword of the host data bus to be driven onto PCI during the data phase of the PCI cycle. On the PIG[3:0] lines, the PCMC signals the LBXs to drive the latched host address lines on the PCI AD[31:0] lines. The upper two bytes of the address lines are used during configuration as IDSEL signals for the PCI devices. The IDSEL pin on each PCI device is connected to one of the AD[31:17] lines.

The PCMC drives the command for a configuration write (1011) onto the C/BE[3:0]# lines and asserts FRAME# for one PCI clock. The PCMC drives the PIG[3:0] lines signaling the LBXs to drive the contents of the PCI write buffer onto the PCI AD[31:0] lines. This command is driven for only one PCI clock before returning to the SCPA command on the PIG[3:0] lines. The LBXs continue to drive the AD[31:0] lines with the valid write data as long as DRVPCI is asserted. The PCMC then asserts IRDY# and waits until sampling the TRDY# signal active. When TRDY# is sampled asserted, the PCMC negates DRVPCI tri-stating the LBX AD[31:0] lines. BRDY# is asserted for one clock to terminate the CPU cycle.

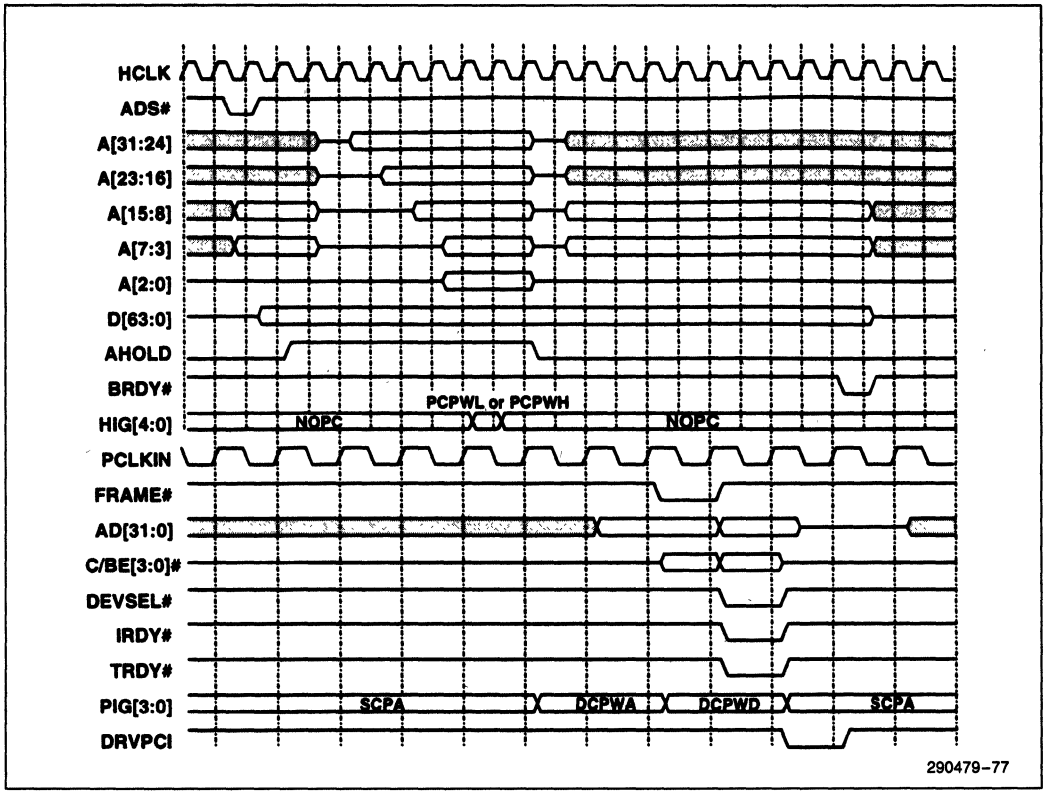


Figure 67. CPU Write to PCI Device Configuration Register

7.3.4 CPU READ FROM PCI DEVICE CONFIGURATION REGISTER

In order to write to or read from a PCI device configuration register the Key field in the CSE register must be programmed to a non-zero value, enabling configuration space. When configuration space is enabled, PCI device configuration registers are accessed by CPU I/O accesses within the range of CnXXh where each PCI device has a unique non-zero value of n. This allows a separate configuration space for each of 15 devices on PCI. Recall that when configuration space is enabled, the PCMC configuration registers occupy I/O addresses C0XXH.

A CPU read from a PCI device configuration register is shown in Figure 68. The PCMC internally latches the host address lines and byte enables. The PCMC asserts AHOLD to tri-state the CPU address bus. The PCMC drives the address lines with the translated address for the PCI configuration cycle. The translation is described in Section 3.2, PCI Configuration Space Mapped Registers. On the PIG[3:0] lines, the PCMC signals the LBXs to drive the latched host address lines on the PCI AD[31:0] lines. The upper two bytes of the address lines are used during configuration as IDSEL signals for the PCI devices. The IDSEL pin on each PCI device is connected to one of the AD[31:17] lines.

The PCMC drives the command for a configuration read (1010) onto the C/BE[3:0]# lines and asserts FRAME# for one PCI clock. The PCMC drives the PIG[3:0] lines signaling the LBXs to latch the data on the PCI AD[31:0] lines into the CPU-to-PCI first read prefetch buffer. The PCMC then drives the HIG[4:0] lines signaling the LBXs to drive the data from the buffer onto the host data lines. The PCMC asserts IRDY# and waits until sampling TRDY# active. After TRDY# is sampled active, BRDY# is asserted for one clock to terminate the CPU cycle.

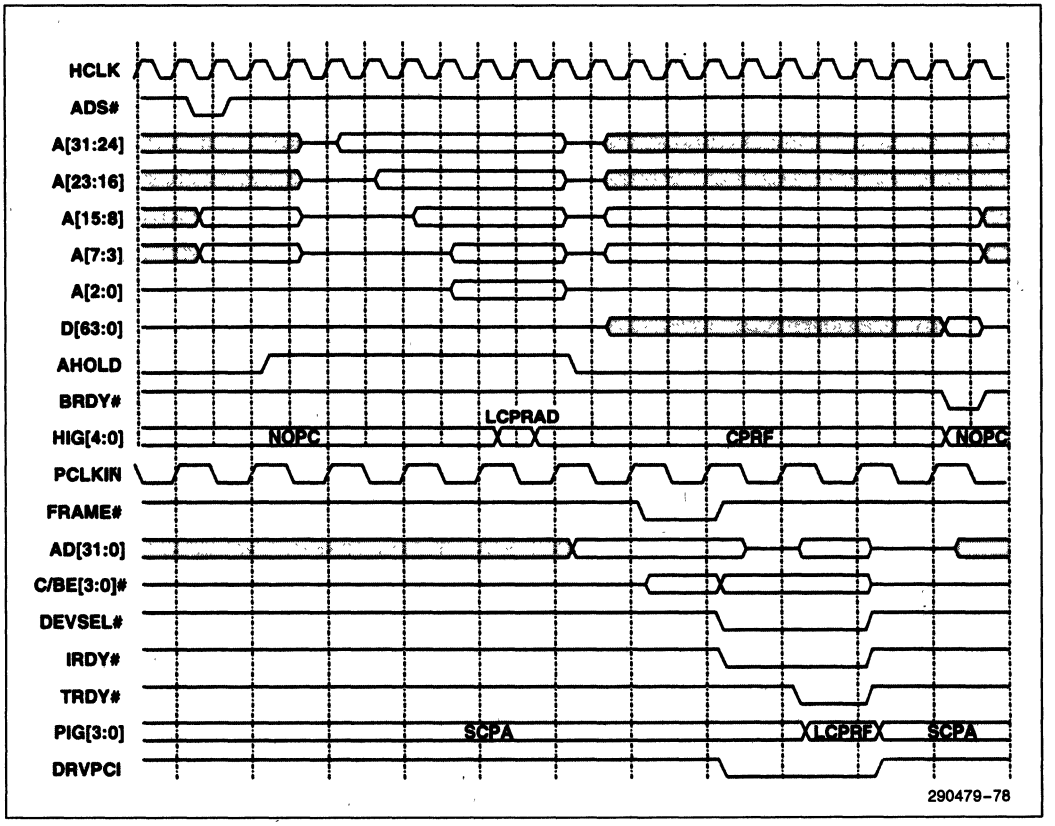


Figure 68. CPU Read from PCI Device Configuration Register

During system initialization, the CPU typically attempts to read from the configuration space of all 15 possible PCI devices to detect the presence of the devices. If no device is present, DEVSEL# is not be asserted and the cycle is terminated, returning FF ... FFh to the CPU. Figure 69 depicts an attempted read from a configuration register of a non-existent device. If no device responds then the PCMC aborts the cycle and sends the DRVFF command over the HIG[4:0] lines causing the LBXs to drive FF ... FFh onto the host data lines.

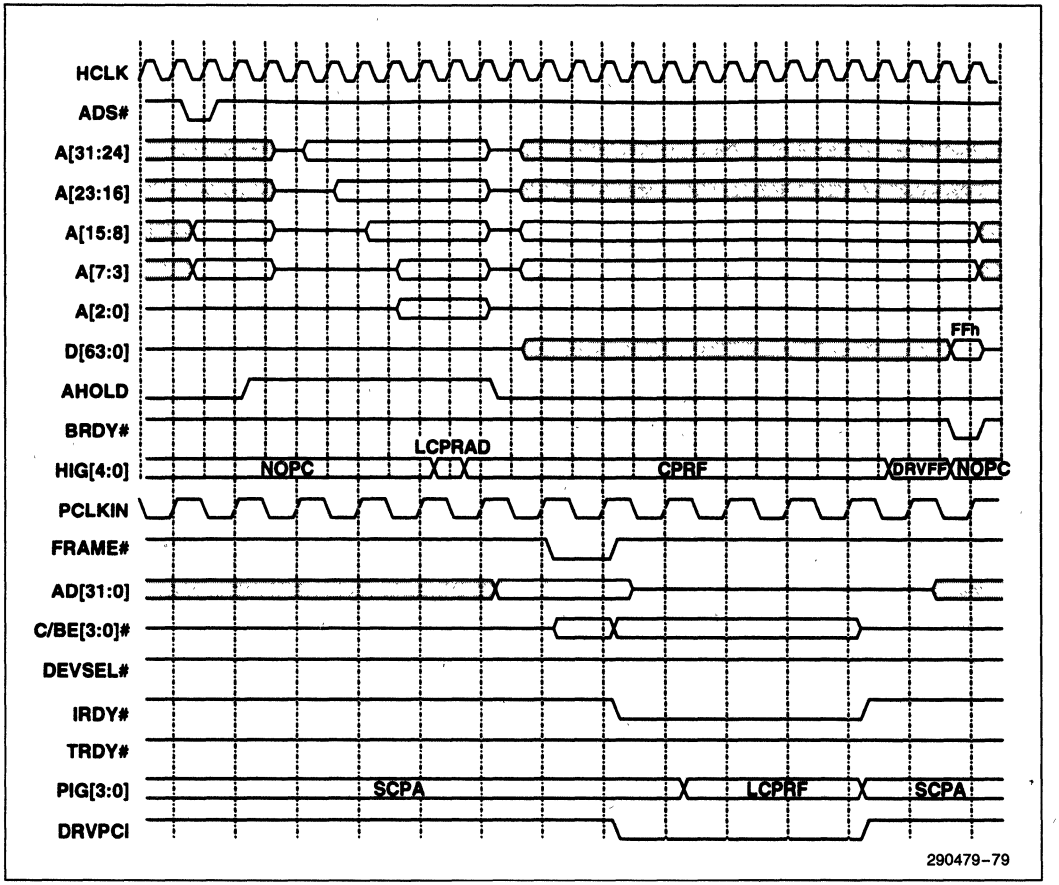


Figure 69. CPU Attempted Configuration Read from Non-Existent PCI Device

7.4 PCI-to-Main Memory Cycles

7.4.1 PCI MASTER WRITE TO MAIN MEMORY

Figure 70 depicts a PCI master burst write to main memory. The PCI master begins by driving the address on the AD[31:0] lines and asserting FRAME#. Upon sampling FRAME# active, the PCMC drives the LCPA command on the PIG[3:0] lines causing the LBXs to retain the address that was latched on the previous PCLK rising edge. The PCMC then samples MEMCS# active, indicating that the cycle is directed to main memory. The PCMC drives the PPMWA command on the PIG[3:0] lines to move the latched PCI address into the write buffer address register. The PCMC then drives the DPWA command on the HIG[4:0] lines enabling the LBXs to drive the PCI master write address onto the host address bus. The PCMC asserts EADS# to initiate a first level cache snoop cycle and simultaneously begins an internal second level cache snoop cycle. Since the snoop is a result of a PCI master write, INV is asserted with EADS#. HITM# remains negated and the snoop either hits an unmodified line or misses in the second level cache, thus no write-back cycles are required. If the snoop hit an unmodified line in either the first or second level cache, the line is invalidated. The cycle is immediately forwarded to the DRAM interface. The four posted Dwords are written to main memory as two Qwords with two CAS[7:0]# cycles. In this example, the DRAM interface is configured for X-3-3-3 write timing, thus each CAS[7:0]# low pulse is two HCLKs in length.

The PCMC disconnects the cycle by asserting STOP# when one of the two four-Dword-deep PCI-to-Memory Posted Write Buffers is full. If the master terminates the cycle before sampling STOP# asserted, then IRDY#, STOP# and DEVSEL# are negated when FRAME# is sampled negated. If the master intended to continue bursting, then the master negates FRAME# when it samples STOP# asserted. IRDY#, STOP# and DEVSEL# are then negated one clock later.

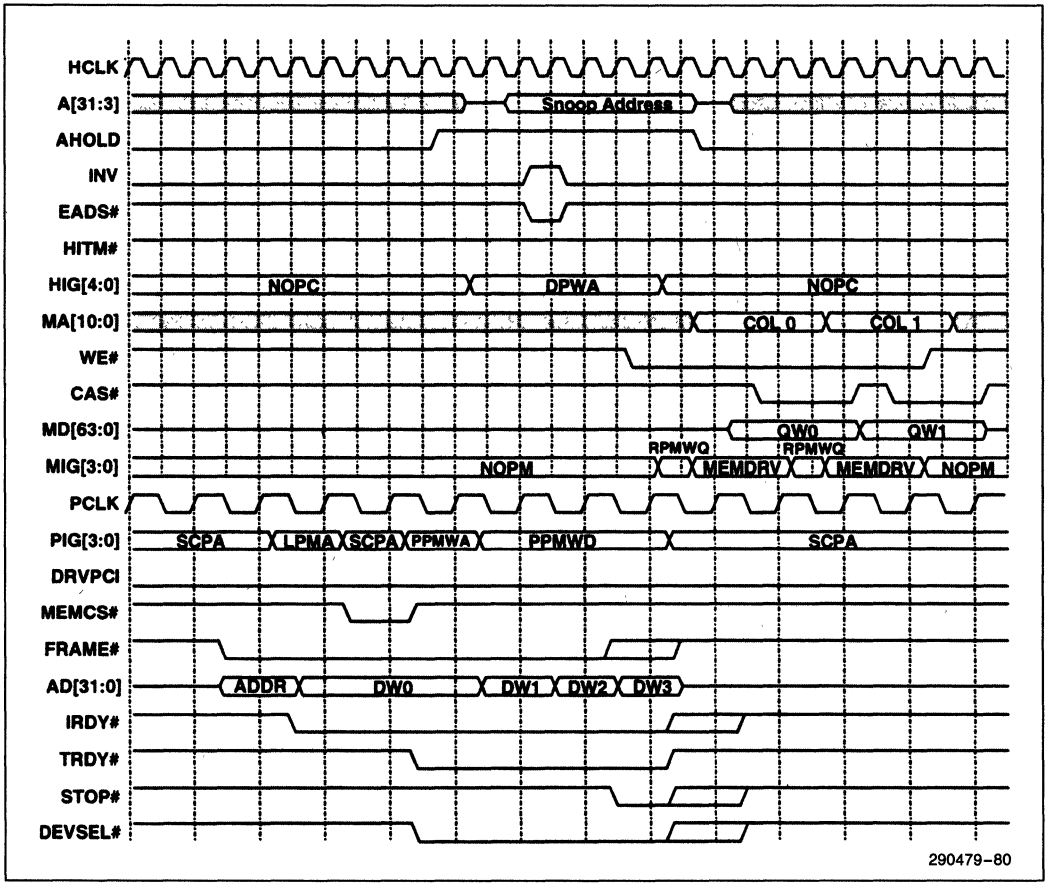


Figure 70. PCI Master Write to Main Memory-Page Hit

7.4.2 PCI MASTER READ FROM MAIN MEMORY

Figure 71 depicts a PCI master read from main memory. The PCI master initiates the cycle by driving the read address on the AD[31:0] lines and asserting FRAME#. The PCMC drives the LPMA command on the PIG[3:0] lines causing the LBXs to retain the address latched on the previous PCLK rising edge. The PCMC drives the DPRA command on the HIG[4:0] lines enabling the LBXs to drive the read address onto the host address lines. The snoop cycle misses in the second level cache and either hits an unmodified line or misses in the first level cache. The cycle is then forwarded to the DRAM interface. A read of four Qwords is performed. Each Qword is posted in the PCI-Memory Read Prefetch Buffer. The data is then driven onto PCI in an eight Dword burst cycle. If the master terminates the cycle before sampling STOP#, then IRDY#, STOP# and DEVSEL# are all negated after FRAME# is sampled inactive. If the master intended to continue bursting, then the master negates FRAME# when it samples STOP# asserted and IRDY#, STOP# and DEVSEL# are negated one clock later.

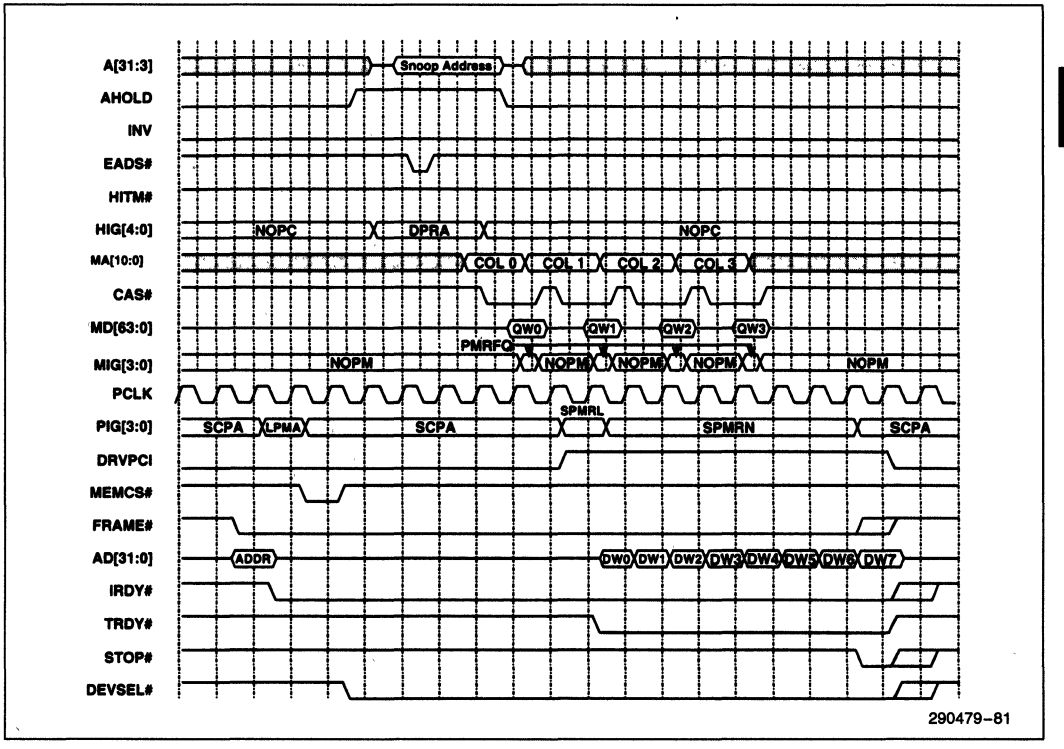


Figure 71. PCI Master Read from Main Memory-Page Hit

8.0 SYSTEM CLOCKING AND RESET

8.1 Clock Domains

The 82434LX and 82434NX PCMCs and 82433LX and 82433NX LBXs operate based on two clocks, HCLK and PCLK. The CPU, second level cache, and the DRAM interfaces operate based on HCLK. The PCI interface timing is based on PCLK.

8.2 Clock Generation and Distribution

Figure 72 shows an example of the 82434LX and 82434NX PCMC host clock distribution in the CPU, cache and memory subsystem. HCLK is distributed to the CPU, PCMC, LBXs and the second level cache SRAMs (in the case of a burst SRAM second level cache).

The host clock originates from an oscillator which is connected to the HCLKOSC input on the PCMC. The PCMC generates six low skew copies of HCLK, HCLKA–HCLKF. Figure 72 shows an example of a host clock distribution scheme for a uni-processor system. In this figure, clock loading is balanced with each HCLK output driving two loads in the system. Each clock output should drive a trace of length *k* with stubs at the end of the trace of length *l* connecting to the two loads. The *l* and *k* parameters should be matched for each of the six clock outputs to minimize overall system clock skew. One of the HCLK outputs is used to clock the PCMC and the Pentium processor. Because the clock driven to the PCMC HCLKIN input and the Pentium processor CLK input originates with the same HCLK output, clock skew between the PCMC and the CPU can be kept lower than between the PCMC and other system components. Another copy of HCLK is used to clock the LBXs. A 256 KByte burst SRAM second level cache can be implemented with eight 32 KByte x 9 synchronous SRAMs. The four remaining copies of HCLK are used to clock the SRAMs. Each HCLK output drives two SRAMs. A 512 KByte second level cache is implemented with four 64 KByte x 18 synchronous SRAMs. Two of the four extra copies are used to clock the SRAMs while the other two are unused. Any one of the HCLK outputs can be used to clock the PCMC and Pentium processor, the two LBXs or any pair of SRAMs. All six copies are identical in drive strength.

Figure 73 depicts the PCI clock distribution.

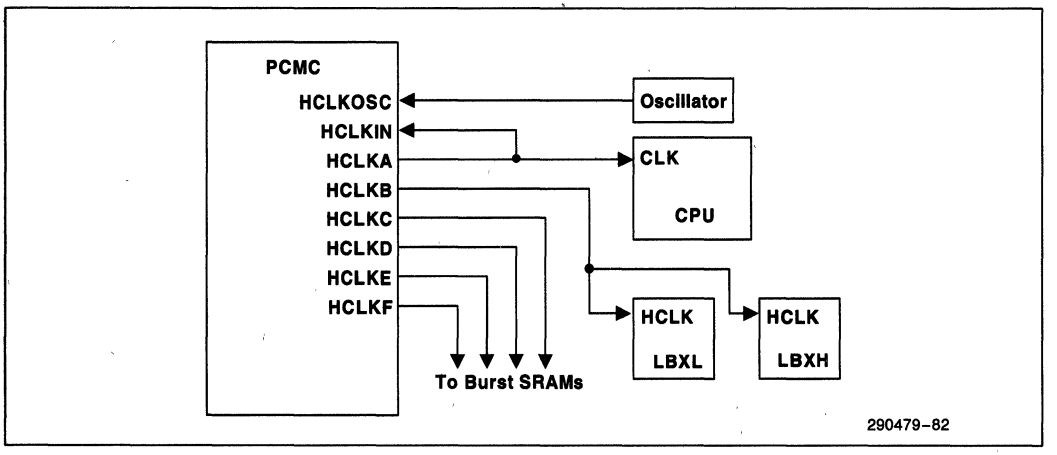


Figure 72. HCLK Distribution Example

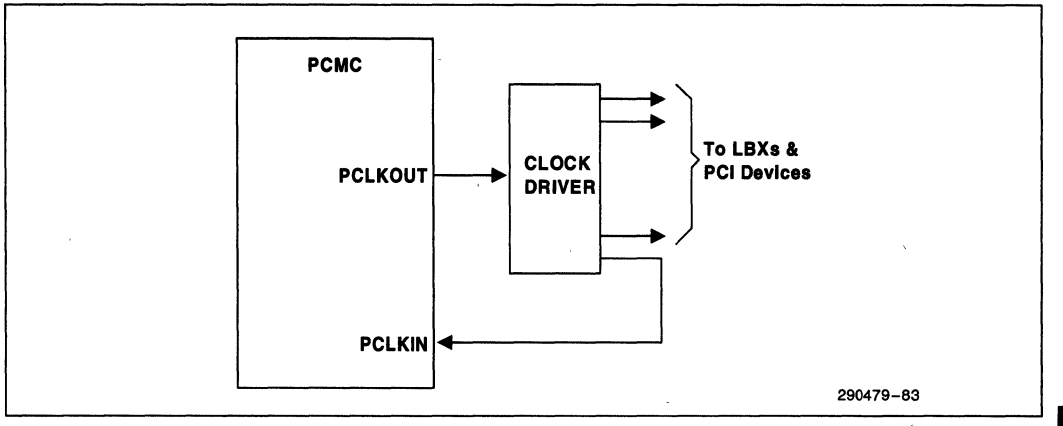


Figure 73. PCI Clock Distribution

The PCMC generates PCLKOUT with an internal Phase Locked Loop (PLL). The PCLKOUT signal is buffered using a single component to produce several low skew copies of PCLK to drive the LBXs and other devices on PCI. One of the outputs of the clock driver is directed back to the PCLKIN input on the PCMC. The PLL locks the rising edges of PCLKIN in phase with the rising edges of HCLKIN. The PLL effectively compensates for the delay of the external clock driver. The resulting PCI clock is one half the frequency of HCLK. Timing for all of the PCI interface signals is based on PCLKIN. All PCI interface inputs are sampled on PCLKIN rising edges and all outputs transition as valid delays from PCLKIN rising edges. Clock skew between the PCLKIN pin on the PCMC and the PCLK pins on the LBXs must be kept within 1.25 ns to guarantee proper operation of the LBXs.

8.3 Phase Locked Loop Circuitry

The 82434LX and 82434NX PCMCs each contain two internal Phase Locked Loops (PLLs). Loop filters and power supply decoupling circuitry must be provided externally. Figure 74 shows the PCMC connections to the external PLL circuitry.

One of the PCMC internal Phase Locked Loops (PLL) locks onto the HCLKIN input. The PLL is used by the PCMC in generating and sampling timing critical signals. An external loop filter is required. The PLLARC1 and PLLARC2 pins connect to the external HCLK loop filter. Two resistors and a capacitor form the loop filter. The loop filter circuitry should be placed as close as possible to the PCMC loop filter pins. The PLL also has dedicated power and ground pins, PLLAVDD, PLLAVSS and PLLAGND. These power pins require a low noise supply. PLLAVDD, PLLAVSS and PLLAGND must be connected to the RC network shown in Figure 74.

The second PCMC internal Phase Locked Loop (PLL) locks the PCLKIN input in phase with the HCLKIN input. The PLL is used by the PCMC to keep the PCI clock in phase with the host clock. An external loop filter is required. The PLLBRC1 and PLLBRC2 pins connect to the external PCLK loop filter. Two resistors and a capacitor form the loop filter. The loop filter circuitry should be placed as close as possible to the PCMC loop filter pins. The PLL also has dedicated power and ground pins, PLLBVDD, PLLBVSS and PLLBGND. These power pins require a low noise supply. PLLBVDD, PLLBVSS and PLLBGND must be connected to the RC network shown in Figure 74.

The resistance and capacitance values for the external PLL circuitry are listed below.

R1 = 10 KΩ ± 5%

R2 = 150Ω ± 5%

R3 = 33Ω ± 5%

C1 = 0.01 μF ± 10%

C2 = 0.47 μF ± 10%

An additional 0.01 μF capacitor in parallel with C2 will help to improve noise immunity.

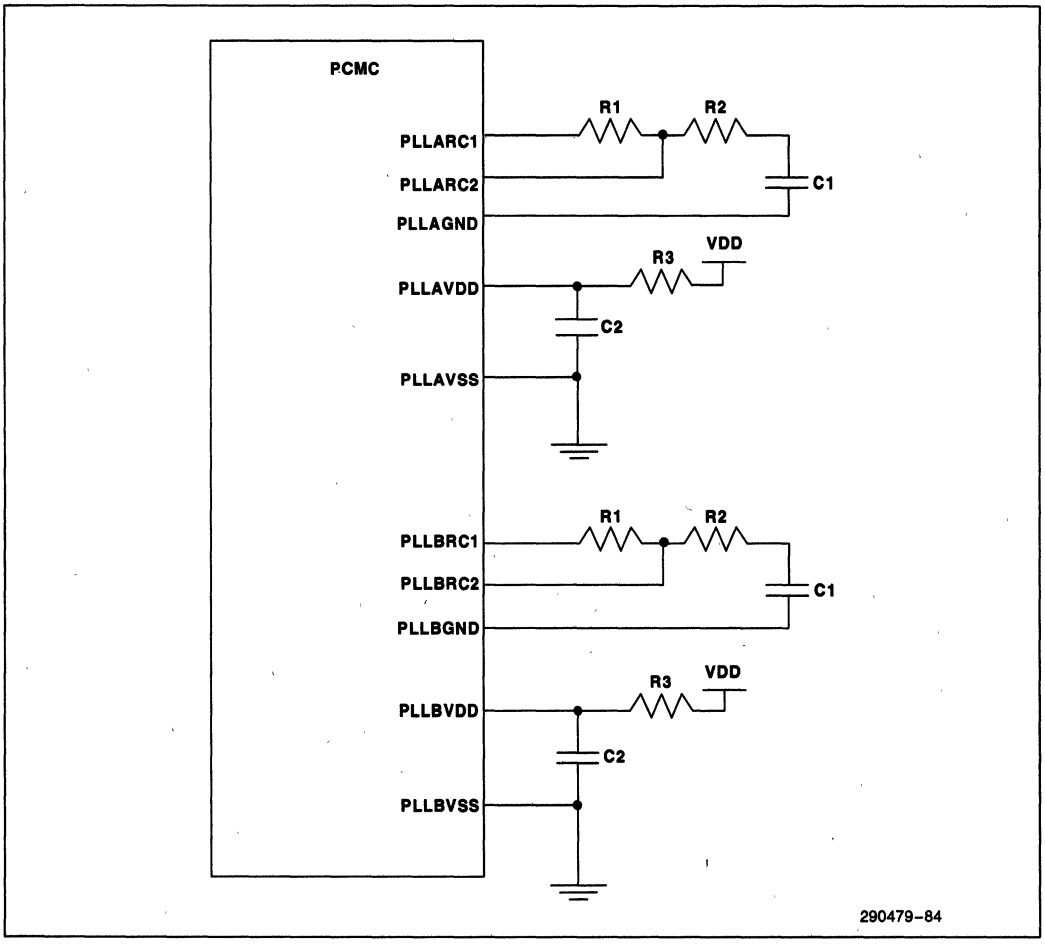


Figure 74. PCMC PLL Circuitry Connections

8.4 System Reset

Figure 75 shows the 82434LX and 82434NX PCMC system reset connections. The 82434LX and 82434NX PCMC reset logic monitors PWROK and generates CPURST, PCIRST# and INIT.

When asserted, PWROK is an indicator to the PCMC that VDD and HCLK have stabilized long enough for proper system operation. CPURST is asserted to initiate hard reset. INIT is asserted to initiate soft reset. PCIRST# is asserted to reset devices on PCI.

Hard reset is initiated by the PCMC in response to one of two conditions. First, hard reset is initiated when power is first applied to the system. PWROK must be driven inactive and must not be asserted until 1 ms after VDD and HCLK have stabilized at their AC and DC specifications. While PWROK is negated, the 82434LX asserts CPURST and PCIRST#. PWROK can be asserted asynchronously. When PWROK is asserted, the 82434LX first ensures that it has been completely initialized before negating CPURST and PCIRST#. CPURST is negated synchronously to the rising edge of HCLK. PCIRST# is negated asynchronously.

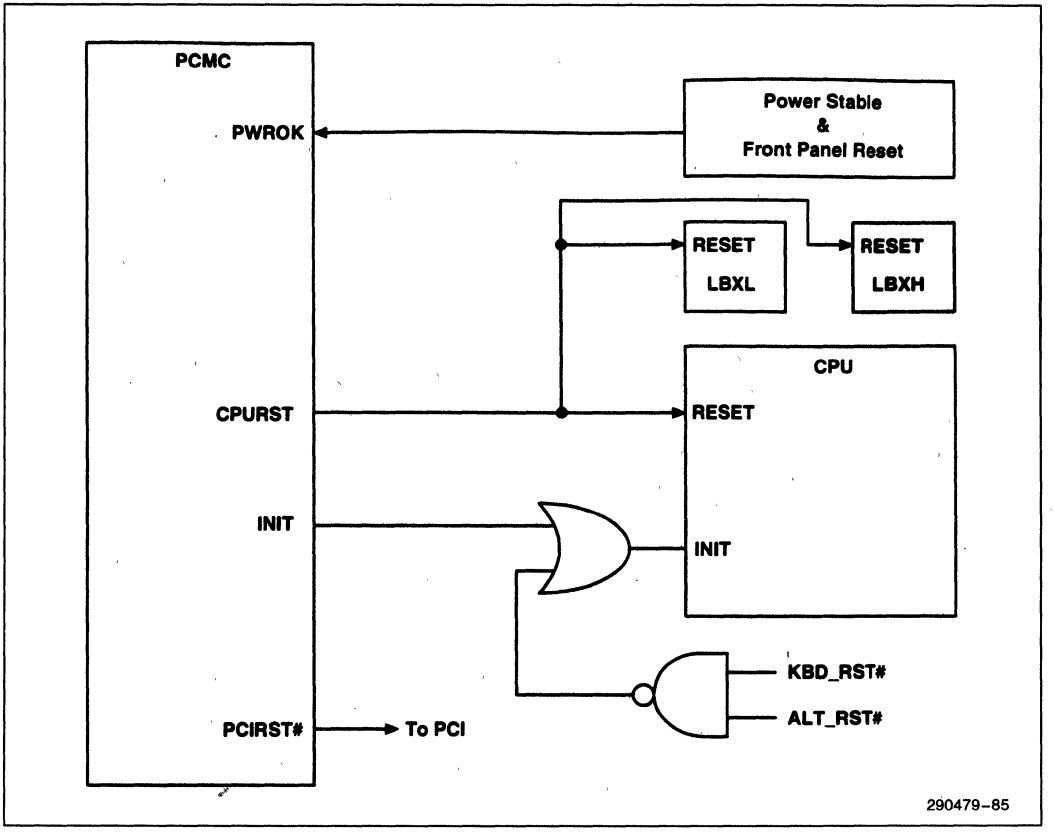


Figure 75. PCMC System Reset Logic

When PWROK is negated, the PCMC asserts AHOLD causing the CPU to tri-state the host address lines. Address lines A[31:29] are sampled by the PCMC 1 ms after the rising edge of PWROK. The values sampled on A[31:30] are inverted inside the PCMC and then stored in Configuration Register 52h bits 7 and 6. The A[31:30] strapping options are depicted in Table 18.

Table 18. A[31:30] Strapping Options

A[31:30]	Configuration Register 52h, Bits[7:6]	Secondary Cache Size
11	00	Not Populated
10	01	Reserved
01	10	256 KByte Cache
00	11	512 KByte Cache

The value sampled on A29 is inverted inside the PCMC and stored in the SRAM Type Bit (bit 5) in the SCC Register. A28 is required to be pulled high for compatibility with future versions of the PCMC.

The PCMC also initiates hard reset when the System Hard Reset Enable bit in the Turbo-Reset Control Register (I/O address CF9h) is set to 1 and the Reset CPU bit toggles from 0 to 1. The PCMC drives CPURST and PCIRST# active for a minimum of 1 ms.

Table 19 shows the state of all 82434LX PCMC output and bi-directional signals during hard reset. During hard reset both CPURST and PCIRST# are asserted. When the hard reset is due to PWROK negation, AHOLD is asserted. The PCMC samples the strapping options on the A[31:29] lines 1 ms after the rising edge of PWROK. When hard reset is initiated via a write to the Turbo-Reset Control Register (I/O port CF9h) AHOLD remains negated throughout the hard reset. Table 19 also applies to the 82434NX, with the exception of the signals listed in Section 8.5, 82434NX Reset Sequencing.

Table 19. 82434LX Output and I/O Signal States During Hard Reset

Signal	State	Signal	State
A[31:0]	Input	IRDY#	Input
AHOLD	High/Low	KEN#	Undefined
BOFF#	High	MA[10:0]	Undefined
BRDY#	High	MDLE	High
CAA[6:3]	Undefined	MEMACK#	High-Z
CAB[6:3]	Undefined	MIG[2:0]	Low
CADS[1:0]#	High	NA#	High
CADV[1:0]#	High	PAR	Input
CALE	High	PEN#	High
CAS[7:0]#	High	PERR#	Input
COE[1:0]#	High	PLOCK#	Input
CWE[7:0]#	High	PIG3	Low
C/BE[3:0]#	Input	PIG[2:0]	High
DEVSEL#	Input	RAS[5:0]#	High
DRVPCI	Low	REQ#	High-Z
EADS#	High	SERR#	Input
FRAME#	Input	STOP#	Input
HIG[4:0]	Low	TRDY#	Input
INIT	Low	WE#	High
INV	Low		

Soft reset is initiated by the PCMC in response to one of two conditions. First, when the System Hard Reset Enable bit in the TRC Register is reset to 0, and the Reset CPU bit toggles from 0 to 1, the PCMC initiates soft reset by asserting INIT for a minimum of 2 HCLKs. Second, the PCMC initiates a soft reset upon detecting a shutdown cycle from the CPU. In this case, the PCMC first broadcasts a shutdown special cycle on PCI and then asserts INIT for a minimum of 2 HCLKs.

8.5 82434NX Reset Sequencing

When PWROK is negated, the 82434NX PCMC drives the following signals low—BRDY#, NA#, AHOLD, EADS#, INV, BOFF#, KEN#, PEN#, CPURST, INIT, CALE, CADS[1:0]#, CADV[1:0]#, CAA[6:3], CAB[6:3], COE[1:0]#, CWE[7:0]#. HCLK[A:F] are driven as soon as the 3.3V supply is active. Note that CWE[7:0]# low prevents the second level cache data RAMs from driving the data bus, even though COE[1:0]# are also driven low. Also, note that BOFF# driven low causes the CPU to tri-state all outputs to the 82434NX PCMC and 82433NX LBX, except HITM#, SMIACT#, and PCHK#. This minimizes the number of signals that the CPU may drive to the PCMC when the 3.3V supply is active and the 5V supply is not active.

Figure 76 shows how the 82434NX sequences CPURST and PCIRST# in response to PWROK assertion.

Some PCI devices may drive 3.3V friendly signals directly to 3.3V devices that are not 5V tolerant. If such signals are powered from the 5V supply they must be driven low when PCIRST# is asserted. Some of these signals may need to be driven high before CPURST is negated. PCIRST# is negated 1 ms before CPURST to allow time for this to occur.

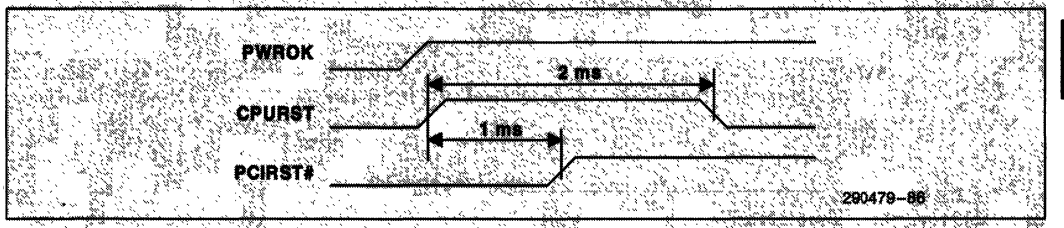


Figure 76. 82434NX Reset Sequencing at Power-Up

2

9.0 ELECTRICAL CHARACTERISTICS

NOTICE: This data sheet contains information on products in the sampling and initial production phases of development. The specifications are subject to change without notice. Verify with your local Intel Sales office that you have the latest data sheet before finalizing a design.

9.1 Absolute Maximum Ratings

Case Temperature under Bias .....0°C to +85°C

Storage Temperature .....−55°C to +150°C

Voltage on Any Pin with Respect to Ground.....−0.3 to V_{CC} + 0.3V

Supply Voltage with Respect to V_{SS} .....−0.3 to +6.5V

Maximum Total Power Dissipation 2.0W

Maximum Power Dissipation, $V_{CC}3$ .....470 mW

The Maximum total power dissipation in the 82434NX on the V_{CC} and $V_{CC}3$ pins is 2.0W. The $V_{CC}3$ pins may draw as much as 470 mW, however, total power will not exceed 2.0W.

**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.*

9.2 Thermal Characteristics

The 82434LX and 82434NX PCMCs are designed for operation at case temperatures between 0°C and 85°C. The thermal resistances of the package are given in Table 20.

Table 20. PCMC Package Thermal Resistance

Parameter	Air Flow Meters/Second (Linear Feet per Minute)				
	0 (0)	0.5 (98.4)	1.0 (196.9)	2.0 (393.7)	5.0 (984.3)
θ_{JA} (°C/Watt)	31	27	24.5	23	19
θ_{JC} (°C/Watt)	8.6				

9.3 82434LX DC Characteristics

Functional Operating Range (V_{CC} = 5V ±5%; T_{CASE} = 0°C to +85°C)

Symbol	Parameter	Min	Max	Unit	Test Conditions
V_{IL1}	Input Low Voltage	−0.3	0.8	V	Note 1, V_{CC} = 4.75V
V_{IH1}	Input High Voltage	2.2	V_{CC} + 0.3	V	Note 1, V_{CC} = 5.25V
V_{IL2}	Input Low Voltage	−0.3	1.35	V	Note 2, V_{CC} = 4.75V
V_{IH2}	Input High Voltage	3.85	V_{CC} + 0.3	V	Note 2, V_{CC} = 5.25V
V_{T1}	Schmitt Trigger Threshold Voltage, Falling Edge	0.7	1.35	V	Note 3, V_{CC} = 5.0V
V_{T1+}	Schmitt Trigger Threshold Voltage, Falling Edge	1.4	2.2	V	Note 3, V_{CC} = 5.0V
V_{H1}	Hysteresis Voltage	0.3	1.2	V	Note 3, V_{CC} = 5.0V
V_{T2-}	Schmitt Trigger Threshold Voltage, Falling Edge	1.25	2.3	V	Note 3, V_{CC} = 5.0V
V_{T2+}	Schmitt Trigger Threshold Voltage, Rising Edge	2.3	3.7	V	Note 3, V_{CC} = 5.0V
V_{H2}	Hysterersis Voltage	0.3	1.2	V	Note 3, V_{CC} = 5.0V

Functional Operating Range (V_{CC} = 5V ±5%; T_{CASE} = 0°C to +85°C) (Continued)

Symbol	Parameter	Min	Max	Unit	Test Conditions
V_{OL1}	Output Low Voltage		0.5	V	Note 4
V_{OH1}	Output High Voltage	V_{CC} − 0.5		V	Note 4
V_{OL2}	Output Low Voltage		0.4	V	Note 5
V_{OH2}	Output High Voltage	2.4		V	Note 5
I_{OL1}	Output Low Current		1	mA	Note 6
I_{OH1}	Output High Current	−1		mA	Note 6
I_{OL2}	Output Low Current		3	mA	Note 7
I_{OH2}	Output High Current	−2		mA	Note 7
I_{OL3}	Output Low Current		6	mA	Note 8
I_{OH3}	Output High Current	−2		mA	Note 8
I_{OL4}	Output Low Current		3	mA	Note 9
I_{OH4}	Output High Current	−1		mA	Note 9
I_{IH}	Input Leakage Current		+10	uA	
I_{IL}	Input Leakage Current		−10	uA	
C_{IN}	Input Capacitance		12	pF	F_C = 1 MHz
C_{OUT}	Output Capacitance		12	pF	F_C = 1 MHz
$C_{I/O}$	I/O Capacitance		12	pF	F_C = 1 MHz

NOTES:
1. V_{IL1} and V_{IH1} apply to the following signals: A[31:0], BE[7:0]#, D/C#, W/R#, M/IO#, HLOCK#, ADS#, PCHK#, HITM#, CACHE#, SMIACT#, PCLKIN, HCLKIN, HCLKOSC, FLSHBUF#, MEMCS#, SERR#, PERR#, MEMREQ#, GNT#, PLOCK#, STOP#, IRDY#, TRDY#, FRAME#, C/BE[3:0]#.
2. V_{IL2} and V_{IH2} apply to the following signals: PPOUT[1:0], EOL.
3. V_{T1-}, V_{T1+} and V_{H1} apply to PWROK. V_{T2-}, V_{T2+} and V_{H2} apply to TESTEN.
4. V_{OL1} and V_{OH1} apply to the following signals: HIG[4:0], MIG[2:0], PIG[3:0], DRVPCI, MDLE, PCIRST#.
5. V_{OL2} and V_{OH2} apply to the following signals: REQ#, MEMACK#, FRAME#, C/BE[3:0]#, TRDY#, IRDY#, STOP#, PLOCK#, DEVSEL#, PAR, PERR#, SERR#, BOFF#, AHOLD, BRDY#, NA#, EADS#, KEN#, INV, A[31:0], PCLKOUT, HCLKA–HCLKF, CALE, COE[1:0]#, CWE[7:0]#, CADV[1:0]#, CADS[1:0]#, CAA[6:3], CAB[6:3], RAS[5:0]#, CAS[7:0]#, MA[10:0], WE#.
6. I_{OL1} and I_{OH1} apply to the following signals: HIG[4:0], MIG[2:0], PIG[3:0], DRVPCI, MDLE, PCIRST#.
7. I_{OL2} and I_{OH2} apply to the following signals: C/BE[3:0]#, REQ#, MEMACK#, MA[10:0], WE#.
8. I_{OL3} and I_{OH3} apply to the following signals: FRAME#, TRDY#, IRDY#, STOP#, PLOCK#, DEVSEL#, PAR, PERR#, SERR#.
9. I_{OL4} and I_{OH4} apply to the following signals: BOFF#, AHOLD, BRDY#, NA#, EADS#, KEN#, INV, CPURST, INIT, A[31:0], PCLKOUT, CALE, COE[1:0]#, CADS[1:0]#, CADV[1:0]#, CWE[7:0]#, CAA[6:3], CAB[6:3], RAS[5:0]#, CAS[7:0]#.

9.4 82434NX DC Characteristics

Functional Operating Range (V_{CC} = 5V ±5%; $V_{CC}3$ = 3.135 to 3.465 V; T_{CASE} = 0°C to +85°C)

Symbol	Parameter	Min	Max	Unit	Test Conditions
V_{IL1}	Input Low Voltage	−0.3	0.8	V	Note 1, V_{CC} = 4.75V
V_{IH1}	Input High Voltage	2.2	V_{CC} + 0.3	V	Note 1, V_{CC} = 5.25V
V_{IL2}	Input Low Voltage	−0.3	1.35	V	Note 2, V_{CC} = 4.75V
V_{IH2}	Input High Voltage	3.65	V_{CC} + 0.3	V	Note 2, V_{CC} = 5.25V
V_{IL3}	Input Low Voltage	−0.3	0.8	V	Note 3, V_{ccq} = 3.135V
V_{IH3}	Input High Voltage	2.2	V_{CC} + 0.3	V	Note 3, V_{ccq} = 3.465V
V_{T1}	Schmitt Trigger Threshold Voltage, Falling Edge	0.7	1.35	V	Note 4, V_{CC} = 5.0V
V_{T1+}	Schmitt Trigger Threshold Voltage, Rising Edge	1.4	2.2	V	Note 4, V_{CC} = 5.0V
V_{H1}	Hysteresis Voltage	0.3	1.2	V	Note 4, V_{CC} = 5.0V
V_{T2-}	Schmitt Trigger Threshold Voltage, Falling Edge	1.25	2.3	V	Note 4, V_{CC} = 5.0V
V_{T2+}	Schmitt Trigger Threshold Voltage, Rising Edge	2.3	3.7	V	Note 4, V_{CC} = 5.0V
V_{H2}	Hysterersis Voltage	0.3	1.2	V	Note 4, V_{CC} = 5.0V
V_{OL1}	Output Low Voltage		0.5	V	Note 5
V_{OH1}	Output High Voltage	V_{CC} − 0.5		V	Note 5
V_{OL2}	Output Low Voltage		0.4	V	Note 6
V_{OH2}	Output High Voltage	2.4		V	Note 6
I_{OL1}	Output Low Current		1	mA	Note 7
I_{OH1}	Output High Current	−1		mA	Note 7
I_{OL2}	Output Low Current		3	mA	Note 8

Functional Operating Range (V_{CC} = 5V ±5%; $V_{CC}3$ = 3.135 to 3.465 V; T_{CASE} = 0°C to +85°C) (Continued)

Symbol	Parameter	Min	Max	Unit	Test Conditions
I_{OH2}	Output High Current	−2		mA	Note 8
I_{OL3}	Output Low Current		6	mA	Note 9
I_{OH3}	Output High Current	−2		mA	Note 9
I_{OL4}	Output Low Current		3	mA	Note 10
I_{OH4}	Output High Current	−1		mA	Note 10
I_{IH}	Input Leakage Current		+10	uA	
I_{IL}	Input Leakage Current		−10	uA	
C_{IN}	Input Capacitance		12	pF	F_C = 1 MHz
C_{OUT}	Output Capacitance		12	pF	F_C = 1 MHz
$C_{I/O}$	I/O Capacitance		12	pF	F_C = 1 MHz

NOTES:
1. V_{IL1} and V_{IH1} apply to the following signals: BE[7:0]#, D/C#, W/R#, M/IO#, HLOCK#, ADS#, PCHK#, HITM#, CACHE#, SMIACT#, PCLKIN, HCLKOSC, FLSHBUF#, MEMCS#, SERR#, PERR#, MEMREQ#, GNT#, PLOCK#, STOP#, IRDY#, TRDY#, FRAME#, C/BE[3:0]#.
2. V_{IL2} and V_{IH2} apply to the following signals: PPOUT[1:0], EOL.
3. VIL3 and VIH3 apply to the following signals: A[31:0], HCLKIN.
4. V_{T1-}, V_{T1+} and V_{H1} apply to PWROK. V_{T2-}, V_{T2+} and V_{H2} apply to TESTEN.
5. V_{OL1} and V_{OH1} apply to the following signals: HIG[4:0], MIG[2:0], PIG[3:0], DRVPCI, MDLE, PCIRST#.
6. V_{OL2} and V_{OH2} apply to the following signals: REQ#, MEMACK#, FRAME#, C/BE[3:0]#, TRDY#, IRDY#, STOP#, PLOCK#, DEVSEL#, PAR, PERR#, SERR#, BOFF#, AHOLD, BRDY#, NA#, EADS#, KEN#, INV, A[31:0], PCLKOUT, HCLKA–HCLKF, CALE, COE[1:0]#, CWE[7:0]#, CADV[1:0]#, CADS[1:0]#, CAA[6:3], CAB[6:3], RAS[7:0]#, CAS[7:0]#, MA[11:0], WE#.
7. I_{OL1} and I_{OH1} apply to the following signals: HIG[4:0], MIG[2:0], PIG[3:0], DRVPCI, MDLE, A[31:3], A[2:0], PCIRST#.
8. I_{OL2} and I_{OH2} apply to the following signals: C/BE[3:0]#, REQ#, MEMACK#, MA[11:0], WE#.
9. I_{OL3} and I_{OH3} apply to the following signals: FRAME#, TRDY#, IRDY#, STOP#, PLOCK#, DEVSEL#, PAR, PERR#, SERR#.
10. I_{OL4} and I_{OH4} apply to the following signals: BOFF#, AHOLD, BRDY#, NA#, EADS#, KEN#, INV, CPURST, INIT, A[7:3], PCLKOUT, CALE, COE[1:0]#, CADS[1:0]#, CADV[1:0]#, CWE[7:0]#, CAA[6:3], CAB[6:3], RAS[7:0]#, CAS[7:0]#.
11. The output buffers for BRDY#, NA#, AHOLD, EADS#, INV, BOFF#, KEN#, PEN#, CPURST, INIT, CALE, CADS[1:0], CADV[1:0]#, CAA[6:3], CAB[6:3], COE[1:0]#, CWE[7:0]#, A[31:3] AND HCLK[A:F] are powered with $V_{CC}3$ and therefore drive 3.3V signal levels.

9.5 82434LX AC Characteristics

The AC characteristics given in this section consist of propagation delays, valid delays, input setup requirements, input hold requirements, output float delays, output enable delays, output-to-output delays, pulse widths, clock high and low times and clock period specifications. Figure 77 through Figure 85 define these specifications. Section 9.5 lists the 82434LX AC Characteristics. Output test loads are listed in the right column.

In Figure 77 through Figure 85, VT = 1.5V for the following signals:

A[31:0], BE[7:0]#, PEN#, D/C#, W/R#, M/IO#, HLOCK#, ADS#, PCHK#, HITM#, EADS#, BRDY#, BOFF#, AHOLD, NA#, KEN#, INV, CACHE#, SMIACT#, INIT, CPURST, CALE, CADV[1:0]#, COE[1:0]#, CWE[7:0]#, CADS[1:0]#, CAA[6:3], CAB[6:3], WE#, RAS[5:0]#, CAS[7:0]#, MA[10:0], C/BE[3:0]#, FRAME#, TRDY#, IRDY#, STOP#, PLOCK#, GNT#, DEVSEL#, MEMREQ#, PAR, PERR#, SERR#, REQ#, MEMCS#, FLSHBUF#,MEMACK#, PWROK, HCLKIN, HCLKA–HCLKF, PCLKIN, PCLKOUT.

VT = 2.5V for the following signals:

PPOUT[1:0], EOL, HIG[4:0], PIG[3:0], MIG[2:0], DRVPCI, MDLE, PCIRST#.

9.5.1 HOST CLOCK TIMING, 66 MHz (82434LX)

Functional Operating Range (V_{CC} = 4.9V to 5.25V; T_{CASE} = 0°C to +70°C)

Symbol	Parameter	Min	Max	Figure	Notes
t1a	HCLKOSC High Time	6.0		82	
t1b	HCLKOSC Low Time	5.0		82	
t2a	HCLKIN Period	15	20	82	
t2b	HCLKIN Period Stability		±100		ps(1)
t2c	HCLKIN High Time	4		82	
t2d	HCLKIN Low Time	4		82	
t2e	HCLKIN Rise Time		1.5	83	
t2f	HCLKIN Fall Time		1.5	83	
t3a	HCLKA–HCLKF Output-to-Output Skew		0.5	85	0 pF
t3b	HCLKA–HCLKF High Time	5.0		85	0 pF
t3c	HCLKA–HCLKF Low Time	5.0		85	0 pF

NOTE:
1. Measured on rising edge of adjacent clocks at 1.5V.

9.5.2 CPU INTERFACE TIMING, 66 MHz (82434LX)

Functional Operating Range (V_{CC} = 4.9V to 5.25V; T_{CASE} = 0°C to +70°C)

Symbol	Parameter	Min	Max	Fig	Notes
t10a	ADS#, HITM#, W/R#, M/IO#, D/C#, HLOCK#, CACHE#, BE[7:0]#, SMIACT# Setup Time to HCLKIN Rising	4.6		79	
t10b	ADS#, HITM#, W/R#, M/IO#, D/C#, HLOCK#, CACHE#, BE[7:0]#, SMIACT# Hold Time from HCLKIN Rising	0.8		79	
t11a	PCHK# Setup Time to HCLKIN Rising	4.3		79	
t11b	PCHK# Hold Time from HCLKIN Rising	1.1		79	
t12a	A[18:3] Rising Edge Setup Time to HCLKIN Rising	4.5		79	Setup to HCLKIN rising when ADS# is sampled active by PCMC.
t12aa	A[18:3] Falling Edge Setup Time to HCLKIN Rising	3.2		79	Setup to HCLKIN Rising when ADS# is Sampled Active by PCMC.
t12ab	A[18:3] Rising Edge Setup Time to HCLKIN Rising	4.7			Setup to HCLKIN Rising when ADS# is Sampled Active by PCMC.
t12ac	A[18:3] Falling Edge Setup Time to HCLKIN Rising	4.1			Setup to HCLKIN Rising when ADS# is Sampled Active by PCMC.
t12b	A[31:0] Hold Time from HCLKIN Rising	0.5		79	Hold from HCLKIN rising two clocks after ADS# is sampled active by PCMC.
t12c	A[31:0] Setup Time to HCLKIN Rising	6.5		79	Setup to HCLKIN rising when EADS# is sampled active by the CPU.
t12d	A[31:0] Hold Time from HCLKIN Rising	1.5		79	Hold from HCLKIN rising when EADS# is sampled active by the CPU.

Functional Operating Range (V_{CC} = 4.9V to 5.25V; T_{CASE} = 0°C to +70°C) (Continued)

Symbol	Parameter	Min	Max	Fig	Notes
t12e	A[31:0] Output Enable from HCLKIN Rising	0	13	81	
t12f	A[31:0] Valid Delay from HCLKIN Rising	1.3	13	78	0 pF
t12g	A[31:0] Float Delay from HCLKIN Rising	0	13	80	
t12h	A[2:0] Propagation Delay from BE[7:0]#	1	16	77	0 pF
t13a	BRDY# Rising Edge Valid Delay from HCLKIN Rising	1.7	7.8	78	0 pF
t13b	BRDY# Falling Edge Valid Delay from HCLKIN Rising	1.7	7.6	78	0 pF
t14	NA# Valid Delay from HCLKIN Rising	1.3	7.8	78	0 pF
t15a	AHOLD Valid Delay from HCLKIN Rising	1.3	7.1	78	0 pF
t15b	BOFF# Valid Delay from HCLKIN Rising	1.8	7.1	78	
t16a	EADS#, INV, PEN# Valid Delay from HCLKIN Rising	1.3	7.4	78	0 pF
t16b	CPURST Rising Edge Valid Delay from HCLKIN Rising	0.9	7.5	78	
t16c	CPURST Falling Edge Valid Delay from HCLKIN Rising	0.9	7.0	78	
t16d	KEN# Valid delay from HCLKIN Rising	1.3	7.6	78	
t17	INIT High Pulse Width	2 HCLKs		84	Soft reset via TRC register or CPU shutdown special cycle
t18	CPURST High Pulse Width	1 ms		84	Hard reset via TRC register, 0 pF

9.5.3 SECOND LEVEL CACHE STANDARD SRAM TIMING, 66 MHz (82434LX)

Functional Operating Range (V_{CC} = 4.9V to 5.25V; T_{CASE} = 0°C to +70°C)

Symbol	Parameter	Min	Max	Fig	Notes
t20a	CAA[6:3]/CAB[6:3] Propagation Delay from A[6:3]	0	8.5	77	0 pF
t20b	CAA[6:3]/CAB[6:3] Valid Delay from HCLKIN Rising	0	7.2	78	0 pF
t21a	COE[1:0]# Falling Edge Valid Delay from HCLKIN Rising	0	9	78	0 pF
t21b	COE[1:0]# Rising Edge Valid Delay from HCLKIN Rising	0	5.5	78	0 pF
t22a	CWE[7:0]#/CBS[7:0]# Falling Edge Valid Delay from HCLKIN Rising	2	14	78	CPU burst or single write to second level cache, 0 pF
t22b	CWE[7:0]#/CBS[7:0]# Rising Edge Valid Delay from HCLKIN Rising	3	14	78	CPU burst or single write to second level cache, 0 pF
t22c	CWE[7:0]#/CBS[7:0]# Valid Delay from HCLKIN Rising	1.4	7.7	78	Cache line Fill, 0 pF
t22d	CWE[7:0]#/CBS[7:0]# Low Pulse Width	1 HCLK		84	0 pF
t22e	CWE[7:0]#/CBS[7:0]# Driven High before CALE Driven High	-1		85	Last write to second level cache during cache line fill, 0 pF
t22f	CAA[4:3]/CAB[4:3] Valid before CWE[7:0]# Falling	1.5		85	CPU burst write to second level cache, 0 pF
t23	CALE Valid Delay from HCLKIN Rising	0	7.5	78	0 pF
t24	CR/W[1:0]# Valid Delay from HCLKIN Rising	1.5	7.6	78	0 pF
t25	CBS[1:0]# Valid Delay from HCLKIN Rising; Reads from Cache SRAMs	1.0	12.0	78	0 pF

9.5.4 SECOND LEVEL CACHE BURST SRAM TIMING, 66 MHz (82434LX)

Functional Operating Range (V_{CC} = 4.9V to 5.25V; T_{CASE} = 0°C to +70°C)

Symbol	Parameter	Min	Max	Fig	Notes
t30a	CAA[6:3]/CAB[6:3] Propagation Delay from A[6:3]	0	8.5	77	0 pF
t30b	CAA[6:3]/CAB[6:3] Valid Delay from HCLKIN Rising	0	7.0	78	0 pF
t31	CADS[1:0]# Valid Delay from HCLKIN Rising	1.5	7.7	78	0 pF
t32	CADV[1:0]# Valid Delay from HCLKIN Rising	1.5	7.1	78	0 pF
t33	CWE[7:0]# Valid Delay from HCLKIN Rising	1.0	9.0	78	0 pF
t34a	COE[1:0]# Falling Edge Valid Delay from HCLKIN Rising	0	9.0	78	0 pF
t34b	COE[1:0]# Rising Edge Valid Delay from HCLKIN Rising	0	5.5	78	0 pF
t35	CALE Valid Delay from HCLKIN Rising	0	7.5	78	0 pF

9.5.5 DRAM INTERFACE TIMING, 66 MHz (82434LX)

Functional Operating Range (V_{CC} = 4.9V to 5.25V; T_{CASE} = 0°C to +70°C)

Symbol	Parameter	Min	Max	Fig	Notes
t40a	RAS[5:0]# Valid Delay from HCLKIN Rising	0	7.5	78	50 pF
t40b	RAS[5:0]# Pulse Width High	4 HCLKs − 5		84	RAS# precharge at beginning of page miss cycle, 50 pF
t41a	CAS[7:0]# Valid Delay from HCLKIN Rising	0	7.5	78	50 pF
t41b	CAS[7:0]# Pulse Width High	1 HCLKIN − 5		84	CAS# precharge during burst cycles, 50 pF
t42	WE# Valid Delay from HCLKIN Rising	0	21	78	50 pF
t43a	MA[10:0] Propagation Delay from A[23:3]	0	23	77	50 pF
t43b	MA[10:0] Valid Delay from HCLKIN Rising	0	10.1	78	50 pF

9.5.6 PCI CLOCK TIMING, 66 MHz (82434LX)

Functional Operating Range (V_{CC} = 4.9V to 5.25V; T_{CASE} = 0°C to +70°C)

Symbol	Parameter	Min	Max	Fig	Notes
t50a	PCLKOUT High Time	13		82	20 pF
t50b	PCLKOUT Low Time	13		82	20 pF
t51a	PCLKIN High Time	12		82	
t51b	PCLKIN Low Time	12		82	
t51c	PCLKIN Rise Time		3	83	
t51d	PCLKIN Fall Time		3	83	

9.5.7 PCI INTERFACE TIMING, 66 MHz (82434LX)

Functional Operating Range (V_{CC} = 4.9V to 5.25V; T_{CASE} = 0°C to +70°C)

Symbol	Parameter	Min	Max	Fig	Notes
t60a	C/BE[3:0]#, FRAME#, TRDY#, IRDY#, STOP#, PLOCK#, PAR, PERR#, SERR#, DEVSEL# Valid Delay from PCLKIN Rising	2	11	78	Min: 0 pF Max: 50 pF
t60b	C/BE[3:0]#, FRAME#, TRDY#, IRDY#, STOP#, PLOCK#, PAR, PERR#, SERR#, DEVSEL# Output Enable Delay from PCLKIN Rising	2		81	
t60c	C/BE[3:0]#, FRAME#, TRDY#, IRDY#, STOP#, PLOCK#, PAR, PERR#, SERR#, DEVSEL# Float Delay from PCLKIN Rising	2	28	80	
t60d	C/BE[3:0]#, FRAME#, PLOCK#, PAR, PERR#, SERR#, Setup Time to PCLKIN Rising	7		79	
t60da	TRDY#, IRDY# Setup Time to PCLKIN Rising	8.1		77	
t60db	STOP#, DEVSEL# Setup Time to PCLKIN Rising	8.5		77	
t60e	C/BE[3:0]#, FRAME#, PLOCK#, PAR, PERR#, SERR# Hold Time from PCLKIN Rising	0		77	
t61a	REQ#, MEMACK# Valid Delay from PCLKIN Rising	2	12	78	Min: 0 pF Max: 50 pF
t61b	REQ#, MEMACK# Output Enable Delay from PCLKIN Rising	2		81	
t61c	REQ#, MEMACK# Float Delay from PCLKIN Rising	2	28	80	
t62a	FLSHREQ#, MEMREQ# Setup Time to PCLKIN Rising	12		79	
t62b	FLSHREQ#, MEMREQ# Hold Time from PCLKIN Rising	0		79	
t63a	GNT# Setup Time to PCLKIN Rising	10		79	
t63b	GNT# Hold Time from PCLKIN Rising	0		79	
t64a	MEMCS# Setup Time to PCLKIN Rising	7		79	
t64b	MEMCS# Hold Time from PCLKIN Rising	0		79	
t65	PCIRST# Low Pulse Width	1 ms		84	Hard Reset via TRC Register, 0 pF

9.5.8 LBX INTERFACE TIMING, 66 MHz (82434LX)

Functional Operating Range (V_{CC} = 4.9V to 5.25V; T_{CASE} = 0°C to +70°C)

Symbol	Parameter	Min	Max	Fig	Notes
t70	HIG[4:0] Valid Delay from HCLKIN Rising	0.8	6.5	78	0 pF
t71	MIG[2:0] Valid Delay from HCLKIN Rising	0.9	6.5	78	0 pF
t72	PIG[3:0] Valid Delay from PCLKIN Rising	0.7	10.9	78	0 pF
t73	PCIDRV Valid Delay from PCLKIN Rising	1	13.5	78	0 pF
t74a	MDLE Falling Edge Valid Delay from HCLKIN Rising	0.6	5.6	78	0 pF
t74b	MDLE Rising Edge Valid Delay from HCLKIN Rising	0.6	6.8	85	0 pF
t75a	EOL, PPOUT[1:0] Setup Time to PCLKIN Rising	7.7		79	
t75b	EOL, PPOUT[1:0] Hold Time from PCLKIN Rising	1.0		79	

9.5.9 HOST CLOCK TIMING, 60 MHz (82434LX)

Functional Operating Range (V_{CC} = 4.75V to 5.25V; T_{CASE} = 0°C to +85°C)

Symbol	Parameter	Min	Max	Fig	Notes
t1a	HCLKOSC High Time	6.0		82	
t1b	HCLKOSC Low Time	5.0		82	
t2a	HCLKIN Period	16.66	20	82	
t2b	HCLKIN Period Stability		±100		ps(1)
t2c	HCLKIN High Time	4		82	
t2d	HCLKIN Low Time	4		82	
t2e	HCLKIN Rise Time		1.5	83	
t2f	HCLKIN Fall Time		1.5	83	
t3a	HCLKA–HCLKF Output-to-Output Skew		0.5	85	0 pF
t3b	HCLKA–HCLKF High Time	5.0		82	0 pF
t3c	HCLKA–HCLKF Low Time	5.0		82	0 pF

NOTE:
1. Measured on rising edge of adjacent clocks at 1.5V.

9.5.10 CPU INTERFACE TIMING, 60 MHz (82434LX)

Functional Operating Range (V_{CC} = 4.75V to 5.25V; T_{CASE} = 0°C to +85°C)

Symbol	Parameter	Min	Max	Fig	Notes
t10a	ADS#, HITM#, W/R#, M/IO#, D/C#, HLOCK#, CACHE#, BE[7:0]#, SMIACT# Setup Time to HCLKIN Rising	4.6		79	
t10b	ADS#, HITM#, W/R#, M/IO#, D/C#, HLOCK#, CACHE#, BE[7:0]#, SMIACT# Hold Time from HCLKIN Rising	1.1		79	
t11a	PCHK# Setup Time to HCLKIN Rising	4.3		79	
t11b	PCHK# Hold Time from HCLKIN Rising	1.1		79	
t12a	A[18:3] Rising Edge Setup Time to HCLKIN Rising	4.5		79	Setup to HCLKIN rising when ADS# is sampled active by PCMC.
t12aa	A[18:3] Falling Edge Setup Time to HCLKIN Rising	3.2		79	Setup to HCLKIN Rising when ADS# is Sampled Active by PCMC.
t12ab	A[18:3] Rising Edge Setup Time to HCLKIN Rising	4.7		79	Setup to HCLKIN Rising when ADS# is Sampled Active by PCMC.
t12ac	A[18:3] Falling Edge Setup Time to HCLKIN Rising	4.1		79	Setup to HCLKIN Rising when ADS# is Sampled Active by PCMC.
t12b	A[31:0] Hold Time from HCLKIN Rising	0.5		79	Hold from HCLKIN rising two clocks after ADS# is sampled active by PCMC.
t12c	A[31:0] Setup Time to HCLKIN Rising	6.5		79	Setup to HCLKIN rising when EADS# is sampled active by the CPU.
t12d	A[31:0] Hold Time from HCLKIN Rising	1.5		79	Hold from HCLKIN rising when EADS# is sampled active by the CPU.
t12e	A[31:0] Output Enable from HCLKIN Rising	0	13	81	
t12f	A[31:0] Valid Delay from HCLKIN Rising	1.3	13	78	0 pF
t12g	A[31:0] Float Delay from HCLKIN Rising	0	13	80	

Functional Operating Range (V_{CC} = 4.75V to 5.25V; T_{CASE} = 0°C to +85°C) (Continued)

Symbol	Parameter	Min	Max	Fig	Notes
t12h	A[2:0] Propagation Delay from BE[7:0]#	1	16	77	0 pF
t13a	BRDY# Rising Edge Valid Delay from HCLKIN Rising	2.1	7.9	78	0 pF
t13b	BRDY# Falling Edge Valid Delay from HCLKIN Rising	2.1	7.9	78	0 pF
t14	NA# Valid Delay from HCLKIN Rising	1.4	8.4	78	0 pF
t15a	AHOLD Valid Delay from HCLKIN Rising	2.0	7.6	78	0 pF
t15b	BOFF# Valid Delay from HCLKIN Rising	2.0	7.6	78	
t16a	EADS#, INV, PEN# Valid Delay from HCLKIN Rising	2.0	8.0	78	0 pF
t16b	CPURST Rising Edge Valid Delay from HCLKIN Rising	1.2	7.5	78	
t16c	CPURST Falling Edge Valid Delay from HCLKIN Rising	1.2	7.5	78	
t16d	KEN# Valid delay from HCLKIN Rising	1.7	8.2	78	
t17	INIT High Pulse Width	2 HCLKs		84	Soft reset via TRC register or CPU shutdown special cycle
t18	CPURST High Pulse Width	1 ms		84	Hard reset via TRC register, 0 pF

9.5.11 SECOND LEVEL CACHE STANDARD SRAM TIMING, 60 MHz (82434LX)

Functional Operating Range (V_{CC} = 4.75V to 5.25V; T_{CASE} = 0°C to +85°C)

Symbol	Parameter	Min	Max	Fig	Notes
t20a	CAA[6:3]/CAB[6:3] Propagation Delay from A[6:3]	0	8.5	77	0 pF
t20b	CAA[6:3]/CAB[6:3] Valid Delay from HCLKIN Rising	0	7.2	78	0 pF
t21a	COE[1:0]# Falling Edge Valid Delay from HCLKIN Rising	0	9	78	0 pF
t21b	COE[1:0]# Rising Edge Valid Delay from HCLKIN Rising	0	5.5	78	0 pF
t22a	CWE[7:0]#/CBS[7:0]# Falling Edge Valid Delay from HCLKIN Rising	2	14	78	CPU burst or single write to second level cache, 0 pF
t22b	CWE[7:0]#/CBS[7:0]# Rising Edge Valid Delay from HCLKIN Rising	3	15	78	CPU burst or single write to second level cache, 0 pF
t22c	CWE[7:0]#/CBS[7:0]# Valid Delay from HCLKIN Rising	1.4	7.7	78	Cache line Fill, 0 pF
t22d	CWE[7:0]#/CBS[7:0]# Low Pulse Width	1 HCLK		84	0 pF
t22e	CWE[7:0]#/CBS[7:0]# Driven High before CALE Driven High	−1		85	Last write to second level cache during cache line fill, 0 pF
t22f	CAA[4:3]/CAB[4:3] Valid before CWE[7:0]# Falling	1.5		85	CPU burst write to second level cache, 0 pF
t23	CALE Valid Delay from HCLKIN Rising	0	8	78	0 pF
t24	CR/W[1:0]# Valid Delay from HCLKIN Rising	1.5	8.2	78	0 pF
t25	CBS[1:0]# Valid Delay from HCLKIN Rising; Reads from Cache SRAMs	1.0	12.0	78	0 pF

9.5.12 SECOND LEVEL CACHE BURST SRAM TIMING, 60 MHz (82434LX)

Functional Operating Range (V_{CC} = 4.75V to 5.25V; T_{CASE} = 0°C to +85°C)

Symbol	Parameter	Min	Max	Fig	Notes
t30a	CAA[6:3]/CAB[6:3] Propagation Delay from A[6:3]	0	8.5	77	0 pF
t30b	CAA[6:3]/CAB[6:3] Valid Delay from HCLKIN Rising	0	8.2	78	0 pF
t31	CADS[1:0]# Valid Delay from HCLKIN Rising	1.5	8.2	78	0 pF
t32	CADV[1:0]# Valid Delay from HCLKIN Rising	1.5	8.2	78	0 pF
t33	CWE[7:0]# Valid Delay from HCLKIN Rising	1.0	10.5	78	0 pF
t34a	COE[1:0]# Falling Edge Valid Delay from HCLKIN Rising	0	9.5	78	0 pF
t34b	COE[1:0]# Rising Edge Valid Delay from HCLKIN Rising	0	6.0	78	0 pF
t35	CALE Valid Delay from HCLKIN Rising	0	8.5	78	0 pF

9.5.13 DRAM INTERFACE TIMING, 60 MHz (82434LX)

Functional Operating Range (V_{CC} = 4.75V to 5.25V; T_{CASE} = 0°C to +85°C)

Symbol	Parameter	Min	Max	Fig	Notes
t40a	RAS[5:0]# Valid Delay from HCLKIN Rising	0	8.0	78	50 pF
t40b	RAS[5:0]# Pulse Width High	4 HCLKs−5		84	RAS# precharge at beginning of page miss cycle, 50 pF
t41a	CAS[7:0]# Valid Delay from HCLKIN Rising	0	8.0	78	50 pF
t41b	CAS[7:0]# Pulse Width High	1 HCLK−5		84	CAS# precharge during burst cycles, 50 pF
t42	WE# Valid Delay from HCLKIN Rising	0	21	78	50 pF
t43a	MA[10:0] Propagation Delay from A[23:3]	0	23	77	50 pF
t43b	MA[10:0] Valid Delay from HCLKIN Rising	0	10.7	78	50 pF

9.5.14 PCI CLOCK TIMING, 60 MHz (82434LX)

Functional Operating Range (V_{CC} = 4.75V to 5.25V; T_{CASE} = 0°C to +85°C)

Symbol	Parameter	Min	Max	Fig	Notes
t50a	PCLKOUT High Time	13		82	20 pF
t50b	PCLKOUT Low Time	13		82	20 pF
t51a	PCLKIN High Time	12		82	
t51b	PCLKIN Low Time	12		82	
t51c	PCLKIN Rise Time		3	83	
t51d	PCLKIN Fall Time		3	83	

9.5.15 PCI INTERFACE TIMING, 60 MHz (82434LX)

Functional Operating Range (V_{CC} = 4.75V to 5.25V; T_{CASE} = 0°C to +85°C)

Symbol	Parameter	Min	Max	Fig	Notes
t60a	C/BE[3:0]#, FRAME#, TRDY#, IRDY#, STOP#, PLOCK#, PAR, PERR#, SERR#, DEVSEL# Valid Delay from PCLKIN Rising	2	11	78	Min: 0 pF Max: 50 pF
t60b	C/BE[3:0]#, FRAME#, TRDY#, IRDY#, STOP#, PLOCK#, PAR, PERR#, SERR#, DEVSEL# Output Enable Delay from PCLKIN Rising	2		81	
t60c	C/BE[3:0]#, FRAME#, TRDY#, IRDY#, STOP#, PLOCK#, PAR, PERR#, SERR#, DEVSEL# Float Delay from PCLKIN Rising	2	28	80	
t60d	C/BE[3:0]#, FRAME#, TRDY#, IRDY#, STOP#, PLOCK#, PAR, PERR#, SERR#, DEVSEL# Setup Time to PCLKIN Rising	9		79	
t60e	C/BE[3:0]#, FRAME#, TRDY#, IRDY#, STOP#, PLOCK#, PAR, PERR#, SERR#, DEVSEL# Hold Time from PCLKIN Rising	0		79	
t61a	REQ#, MEMACK# Valid Delay from PCLKIN Rising	2	12	78	Min: 0 pF Max: 50 pF
t61b	REQ#, MEMACK# Output Enable Delay from PCLKIN Rising	2		81	
t61c	REQ#, MEMACK# Float Delay from PCLKIN Rising	2	28	80	
t62a	FLSHREQ#, MEMREQ# Setup Time to PCLKIN Rising	12		79	

Functional Operating Range (V_{CC} = 4.75V to 5.25V; T_{CASE} = 0°C to +85°C) (Continued)

Symbol	Parameter	Min	Max	Fig	Notes
t62b	FLSHREQ#, MEMREQ# Hold Time from PCLKIN Rising	0		79	
t63a	GNT# Setup Time to PCLKIN Rising	10		79	
t63b	GNT# Hold Time from PCLKIN Rising	0		79	
t64a	MEMCS# Setup Time to PCLKIN Rising	7		79	
t64b	MEMCS# Hold Time from PCLKIN Rising	0		79	
t65	PCIRST# Low Pulse Width	1 ms		84	Hard Reset via TRC Register, 0 pF

9.5.16 LBX INTERFACE TIMING, 60 MHz (82434LX)

Functional Operating Range (V_{CC} = 4.75V to 5.25V; T_{CASE} = 0°C to +85°C)

Symbol	Parameter	Min	Max	Fig	Notes
t70	HIG[4:0] Valid Delay from HCLKIN Rising	0.8	6.7	78	0 pF
t71	MIG[2:0] Valid Delay from HCLKIN Rising	0.9	6.5	78	0 pF
t72	PIG[3:0] Valid Delay from PCLKIN Rising	1.5	12	78	0 pF
t73	PCIDRV Valid Delay from PCLKIN Rising	1	13	78	0 pF
t74a	MDLE Falling Edge Valid Delay from HCLKIN Rising	0.6	6.8	78	0 pF
t74b	MDLE Rising Edge Valid Delay from HCLKIN Rising	0.6	6.8	85	0 pF
t75a	EOL, PPOUT[1:0] Setup Time to PCLKIN Rising	7.7		79	
t75b	EOL, PPOUT[1:0] Hold Time from PCLKIN Rising	1.0		79	

9.6 82434NX AC Characteristics

The AC characteristics given in this section consist of propagation delays, valid delays, input setup requirements, input hold requirements, output float delays, output enable delays, output-to-output delays, pulse widths, clock high and low times and clock period specifications. Figure 77 through Figure 85 define these specifications. Output test loads are listed in the right column.

In Figure 77 through Figure 85, VT = 1.5V for the following signals:

A[31:0], BE[7:0]#, PEN#, D/C#, W/R#, M/IO#, HLOCK#, ADS#, PCHK#, HITM#, EADS#, BRDY#, BOFF#, AHOLD, NA#, KEN#, INV, CACHE#, SMIACT#, INIT, CPURST, CALE, CADV[1:0]#, COE[1:0]#, CWE[7:0]#, CADS[1:0]#, CAA[6:3], CAB[6:3], WE#, RAS[5:0]#, CAS[7:0]#, MA[10:0], C/BE[3:0]#, FRAME#, TRDY#, IRDY#, STOP#, PLOCK#, GNT#, DEVSEL#, MEMREQ#, PAR, PERR#, SERR#, REQ#, MEMCS#, FLSHBUF#, MEMACK#, PWROK, HCLKIN, HCLKA–HCLKF, PCLKIN, PCLKOUT.

VT = 2.5V for the following signals:

PPOUT[1:0], EOL, HIG[4:0], PIG[3:0], MIG[2:0], DRVPCI, MDLE, PCIRST#

2

9.6.1 HOST CLOCK TIMING, 66 MHz (82434NX), PRELIMINARY

Functional Operating Range (V_{CC} = 4.75V to 5.25V; $V_{CC}3$ = 3.135V to 3.465V; T_{CASE} = 0°C to +85°C)

Symbol	Parameter	Min	Max	Fig	Notes
t1a	HCLKOSC High Time	6.0		82	
t1b	HCLKOSC Low Time	5.0		82	
t2a	HCLKIN Period	15	20	82	
t2b	HCLKIN Period Stability		±100		ps(1)
t2c	HCLKIN High Time	4		82	
t2d	HCLKIN Low Time	4		82	
t2e	HCLKIN Rise Time		1.5	83	
t2f	HCLKIN Fall Time		1.5	83	
t3a	HCLKA–HCLKF Output-to-Output Skew		0.5	85	0 pF
t3b	HCLKA–HCLKF High Time	5.0		82	0 pF
t3c	HCLKA–HCLKF Low Time	5.0		82	0 pF

NOTES:
1. Measured on rising edge of adjacent clocks at 1.5V.

9.6.2 CPU INTERFACE TIMING, 66 MHz (82434NX), PRELIMINARY

Functional Operating Range (V_{CC} = 4.75V to 5.25V; $V_{CC}3$ = 3.135V to 3.465V; T_{CASE} = 0°C to +85°C)

Symbol	Parameter	Min	Max	Fig	Notes
t10a	ADS#, W/R#, Setup Time to HCLKIN Rising	4.6		79	
t10b	BE[7:0]# Setup Time to HCLKIN Rising	4.6		79	
t10c	HITM# Setup Time to HCLKIN Rising	6.4		79	
t10d	CACHE#, M/IO# Setup Time to HCLKIN Rising	4.6		79	
t10e	D/C# Setup Time to HCLKIN Rising	4.0		79	
t10f	HLOCK#, SMIACT#, Setup Time to HCLKIN Rising	4.0		79	
t10g	HITM#, M/IO#, D/C#, Hold Time from HCLKIN Rising	0.7		79	
t10h	W/R#, HLOCK#, Hold Time from HCLKIN Rising	0.8		79	
t10i	ADS#, BE[7:0]# Hold Time from HCLKIN Rising	1.1		79	
t10j	CACHE#, SMIACT# Hold Time from HCLKIN Rising	1.1		79	
t11a	PCHK# Setup Time to HCLKIN Rising	4.3		79	
t11b	PCHK# Hold Time from HCLKIN Rising	1.1		79	
t12a	A[31:0] Setup Time to HCLKIN Rising	2.7		79	Setup to HCLKIN rising when ADS# is sampled active by PCMC.
t12b	A[31:0] Hold Time from HCLKIN Rising	0.5			HOLD from HCLKIN Rising two clocks after ADS# is sampled active by PCMC.
t12c	A[31:0] Setup Time to HCLKIN Rising	6.0		79	Setup to HCLKIN rising when EADS# is sampled active by the CPU.

Functional Operating Range (V_{CC} = 4.75V to 5.25V; $V_{CC}3$ = 3.135V to 3.465V; T_{CASE} = 0°C to +85°C) (Continued)

Symbol	Parameter	Min	Max	Fig	Notes
t12d	A[31:0] Hold Time from HCLKIN Rising	1.5		79	Hold from HCLKIN rising when EADS# is sampled active by the CPU.
t12e	A[31:0] Output Enable from HCLKIN Rising	0	13	81	
t12f	A[31:0] Valid Delay from HCLKIN Rising	1.3	13	78	0 pF
t12g	A[31:0] Float Delay from HCLKIN Rising	0	13	80	
t12h	A[2:0] Propagation Delay from BE[7:0]#	1.0	16	77	0 pF
t13a	BRDY# Rising Edge Valid Delay from HCLKIN Rising	1.6	7.5	78	0 pF
t13b	BRDY# Falling Edge Valid Delay from HCLKIN Rising	1.6	7.5	78	0 pF
t14	NA# Valid Delay from HCLKIN Rising	.9	7.6	78	0 pF
t15a	AHOLD Valid Delay from HCLKIN Rising	1.5	7.0	78	0 pF
t15b	BOFF# Valid Delay from HCLKIN Rising	1.5	7.0	78	0 pF
t16a	EADS#, INV, PEN# Valid Delay from HCLKIN Rising	1.5	7.5	78	0 pF
t16b	CPURST Rising Edge Valid Delay from HCLKIN Rising	1.2	7.0	78	0 pF
t16c	CPURST Falling Edge Valid Delay from HCLKIN Rising	1.2	7.0	78	0 pF
t16d	KEN# Valid delay from HCLKIN Rising	1.5	7.5	78	0 pF
t17	INIT High Pulse Width	2 HCLKs		84	0 pF
t18	CPURST High Pulse Width	1 ms		84	0 pF; Hard reset via TRC register

9.6.3 SECOND LEVEL CACHE STANDARD SRAM TIMING, 66 MHz (82434NX), PRELIMINARY

Functional Operating Range (V_{CC} = 4.75V to 5.25V; $V_{CC}3$ = 3.135V to 3.465V; T_{CASE} = 0°C to +85°C)

Symbol	Parameter	Min	Max	Fig	Notes
t20a	CAA[6:3]/CAB[6:3] Propagation Delay from A[6:3]	0	8.5	77	0 pF
t20b	CAA[6:3]/CAB[6:3] Valid Delay from HCLKIN Rising	0	7.2	78	0 pF
t21a	COE[1:0]# Falling Edge Valid Delay from HCLKIN Rising	0	9	78	0 pF
t21b	COE[1:0]# Rising Edge Valid Delay from HCLKIN Rising	0	5.5	78	0 pF
t22a	CWE[7:0]#/CBS[7:0]# Falling Edge Valid Delay from HCLKIN Rising	2	14	78	CPU burst or single write to second level cache, 0 pF
t22b	CWE[7:0]#/CBS[7:0]# Rising Edge Valid Delay from HCLKIN Rising	3	14	78	CPU burst or single write to second level cache, 0 pF
t22c	CWE[7:0]#/CBS[7:0]# Valid Delay from HCLKIN Rising	1.0	7.7	78	Cache line Fill, 0 pF
t22d	CWE[7:0]#/CBS[7:0]# Low Pulse Width	1 HCLK		84	0 pF
t22e	CWE[7:0]#/CBS[7:0]# Driven High before CALE Driven High	−1		85	Last write to second level cache during cache line fill, 0 pF
t22f	CAA[4:3]/CAB[4:3] Valid before CWE[7:0]# Falling	1.5		85	CPU burst write to second level cache, 0 pF
t23	CALE Valid Delay from HCLKIN Rising	0	8.0	78	0 pF
t24	CR/W[1:0]# Valid Delay from HCLKIN Rising	1.5	8.2	78	0 pF
t25	CBS[1:0]# Valid Delay from HCLKIN Rising; Reads from Cache SRAMs	1.0	12.0	78	0 pF
t26a	CCS[1:0]# Propagation Delay from ADS# Falling		7.0	77	0 pF; First access after powerdown
t26b	CCS[1:0]# Valid Delay from HCLKIN Rising	1.5	8.2	78	0 pF; Entering powerdown

9.6.4 SECOND LEVEL CACHE BURST SRAM TIMING, 66 MHz (82434NX), PRELIMINARY

Functional Operating Range (V_{CC} = 4.75V to 5.25V; $V_{CC}3$ = 3.135V to 3.465V; T_{CASE} = 0°C to +85°C)

Symbol	Parameter	Min	Max	Fig	Notes
t30a	CAA[6:3]/CAB[6:3] Propagation Delay from A[6:3]	0	8.5	77	0 pF
t30b	CAA[6:3]/CAB[6:3] Valid Delay from HCLKIN Rising	0	8.2	78	0 pF
t31	CADS[1:0]# Valid Delay from HCLKIN Rising	1.5	8.0	78	0 pF
t32	CADV[1:0]# Valid Delay from HCLKIN Rising	1.5	8.0	78	0 pF
t33	CWE[7:0]# Valid Delay from HCLKIN Rising	1.5	9.0	78	0 pF
t34a	COE[1:0]# Falling Edge Valid Delay from HCLKIN Rising	0.5	9.0	78	0 pF
t34b	COE[1:0]# Rising Edge Valid Delay from HCLKIN Rising	0.5	6.0	78	0 pF
t35	CALE Valid Delay from HCLKIN Rising	0	8.0	78	0 pF

2

9.6.5 DRAM INTERFACE TIMING, 66 MHz (82434NX), PRELIMINARY

Functional Operating Range (V_{CC} = 4.75V to 5.25V; $V_{CC}3$ = 3.135V to 3.465V; T_{CASE} = 0°C to +85°C)

Symbol	Parameter	Min	Max	Fig	Notes
t40a	RAS[7:0]# Valid Delay from HCLKIN Rising	0	8.0	78	50 pF
t40b	RAS[7:0]# Pulse Width High	4 HCLKs−5		84	RAS# precharge at beginning of page miss cycle, 50 pF
t41a	CAS[7:0]# Valid Delay from HCLKIN Rising	0	8.0	78	50 pF
t41b	CAS[7:0]# Pulse Width High	1 HCLKIN−5		84	CAS# precharge during burst cycles, 50 pF
t42	WE# Valid Delay from HCLKIN Rising	0	21	78	50 pF
t43a	MA[10:0] Propagation Delay from A[23:3]	0	23	77	50 pF
t43b	MA[10:0] Valid Delay from HCLKIN Rising	0	10.7	78	50 pF
t43c	MA11 Propagation Delay from A[25:24]	0	28.0	77	50 pF
t43d	MA11 Valid Delay from HCLKIN Rising	0	12	78	50 pF

9.6.6 PCI CLOCK TIMING, 66 MHz (82434NX), PRELIMINARY

Functional Operating Range (V_{CC} = 4.75V to 5.25V; $V_{CC}3$ = 3.135V to 3.465V; T_{CASE} = 0°C to +85°C)

Symbol	Parameter	Min	Max	Fig	Notes
t50a	PCLKOUT High Time	13		82	20 pF
t50b	PCLKOUT Low Time	13		82	20 pF
t51a	PCLKIN High Time	12		82	
t51b	PCLKIN Low Time	12		82	
t51c	PCLKIN Rise Time		3	83	
t51d	PCLKIN Fall Time		3	83	

9.6.7 PCI INTERFACE TIMING, 66 MHz (82434NX), PRELIMINARY

Functional Operating Range (V_{CC} = 4.75V to 5.25V; $V_{CC}3$ = 3.135V to 3.465V; T_{CASE} = 0°C to +85°C)

Symbol	Parameter	Min	Max	Fig	Notes
t60a	C/BE[3:0]#, FRAME#, TRDY#, IRDY#, STOP#, PLOCK#, PAR, PERR#, SERR#, DEVSEL# Valid Delay from PCLKIN Rising	2	11	78	Min: 0 pF Max: 50 pF
t60b	C/BE[3:0]#, FRAME#, TRDY#, IRDY#, STOP#, PLOCK#, PAR, PERR#, SERR#, DEVSEL# Output Enable Delay from PCLKIN Rising	2		81	
t60c	C/BE[3:0]#, FRAME#, TRDY#, IRDY#, STOP#, PLOCK#, PAR, PERR#, SERR#, DEVSEL# Float Delay from PCLKIN Rising	2	28	80	
t60d	C/BE[3:0]#, FRAME#, TRDY#, IRDY#, STOP#, PLOCK#, PAR, PERR#, SERR#, DEVSEL# Setup Time to PCLKIN Rising	7		79	
t60e	C/BE[3:0]#, FRAME#, TRDY#, IRDY#, STOP#, PLOCK#, PAR, PERR#, SERR#, DEVSEL# Hold Time from PCLKIN Rising	0		79	
t61a	REQ#, MEMACK# Valid Delay from PCLKIN Rising	2	12	78	Min: 0 pF Max: 50 pF
t61b	REQ#, MEMACK# Output Enable Delay from PCLKIN Rising	2		81	
t61c	REQ#, MEMACK# Float Delay from PCLKIN Rising	2	28	80	
t62a	FLSHREQ#, MEMREQ# Setup Time to PCLKIN Rising	12		79	
t62b	FLSHREQ#, MEMREQ# Hold Time from PCLKIN Rising	0		79	

Functional Operating Range (V_{CC} = 4.75V to 5.25V; $V_{CC}3$ = 3.135V to 3.465V; T_{CASE} = 0°C to +85°C) (Continued)

Symbol	Parameter	Min	Max	Fig	Notes
t63a	GNT# Setup Time to PCLKIN Rising	10		79	
t63b	GNT# Hold Time from PCLKIN Rising	0		79	
t64a	MEMCS# Setup Time to PCLKIN Rising	7		79	
t64b	MEMCS# Hold Time from PCLKIN Rising	0		79	
t65	PCIRST# Low Pulse Width	1 ms		84	Hard Reset via TRC Register,0 pF

9.6.8 LBX INTERFACE TIMING, 66 MHz (82434NX), PRELIMINARY

Functional Operating Range (V_{CC} = 4.75V to 5.25V; $V_{CC}3$ = 3.135V to 3.465V; T_{CASE} = 0°C to +85°C)

Symbol	Parameter	Min	Max	Fig	Notes
t70	HIG[4:0] Valid Delay from HCLKIN Rising	0.8	6.5	78	0 pF
t71	MIG[2:0] Valid Delay from HCLKIN Rising	0.9	6.5	78	0 pF
t72	PIG[3:0] Valid Delay from PCLKIN Rising	1.5	12	78	0 pF
t73	PCIDRV Valid Delay from PCLKIN Rising	1	13	78	0 pF
t74a	MDLE Falling Edge Valid Delay from HCLKIN Rising	0.6	6.0	78	0 pF
t74b	MDLE Rising Edge Valid from HCLKIN Rising	0.6	6.0	85	0 pF
t75a	EOL, PPOUT[1:0] Setup Time to PCLKIN Rising	7.7		79	
t75b	EOL, PPOUT[1:0] Hold Time from PCLKIN Rising	1.0		79	

9.6.9 HOST CLOCK TIMING, 50 and 60 MHz (82434NX)

Functional Operating Range (V_{CC} = 4.75V to 5.25V; $V_{CC}3$ = 3.135V to 3.465V; T_{CASE} = 0°C to +85°C)

Symbol	Parameter	Min	Max	Fig	Notes
t1a	HCLKOSC High Time	6.0		82	
t1b	HCLKOSC Low Time	5.0		82	
t2a	HCLKIN Period	16.66	20	82	
t2b	HCLKIN Period Stability		±100		ps(1)
t2c	HCLKIN High Time	4		82	
t2d	HCLKIN Low Time	4		82	
t2e	HCLKIN Rise Time		1.5	83	
t2f	HCLKIN Fall Time		1.5	83	
t3a	HCLKA–HCLKF Output-to-Output Skew		0.5	85	0 pF
t3b	HCLKA–HCLKF High Time	5.0		82	0 pF
t3c	HCLKA–HCLKF Low Time	5.0		82	0 pF

NOTES:
1. Measured on rising edge of adjacent clocks at 1.5V.

9.6.10 CPU INTERFACE TIMING, 50 AND 60 MHz (82434NX)

Functional Operating Range (V_{CC} = 4.75V to 5.25V; $V_{CC}3$ = 3.135V to 3.465V; T_{CASE} = 0°C to +85°C)

Symbol	Parameter	Min	Max	Fig	Notes
t10a	ADS#, W/R#, Setup Time to HCLKIN Rising	4.6		79	
t10b	BE[7:0]# Setup Time to HCLKIN Rising	4.6		79	
t10c	HITM# Setup Time to HCLKIN Rising	6.8		79	
t10d	CACHE#, M/IO# Setup Time to HCLKIN Rising	4.6		79	
t10e	D/C# Setup Time to HCLKIN Rising	4.6		79	
t10f	HLOCK#, SMIACT#, Setup Time to HCLKIN Rising	4.6		79	
t10g	HITM#, M/IO#, D/C#, Hold Time from HCLKIN Rising	0.7		79	
t10h	W/R#, HLOCK# Hold from HCLKIN Rising	0.8		79	
t10i	ADS#, BE[7:0]# Hold Time from HCLKIN Rising	0.9		79	
t10j	CACHE#, SMIACT# Hold Time from HCLKIN Rising	1.1		79	
t11a	PCHK# Setup Time to HCLKIN Rising	4.3		79	
t11b	PCHK# Hold Time from HCLKIN Rising	1.1		79	
t12a	A[31:0] Setup Time to HCLKIN Rising	3.0		79	Setup to HCLKIN rising when ADS# is sampled active by PCMC.
t12b	A[31:0] Hold Time from HCLKIN Rising	0.5		79	HOLD from HCLKIN Rising two clocks after ADS# is sampled active by PCMC
t12c	A[31:0] Setup Time to HCLKIN Rising	6.5		79	Setup to HCLKIN rising when EADS# is sampled active by the CPU.
t12d	A[31:0] Hold Time from HCLKIN Rising	1.5		79	Hold from HCLKIN rising when EADS# is sampled active by the CPU.

Functional Operating Range (V_{CC} = 4.75V to 5.25V; $V_{CC}3$ = 3.135V to 3.465V; T_{CASE} = 0°C to +85°C)
(Continued)

Symbol	Parameter	Min	Max	Fig	Notes
t12e	A[31:0] Output Enable from HCLKIN Rising	0	13	81	
t12f	A[31:0] Valid Delay from HCLKIN Rising	1.3	13	78	0 pF
t12g	A[31:0] Float Delay from HCLKIN Rising	0	13	80	
t12h	A[2:0] Propagation Delay from BE[7:0]#	1.0	16	77	0 pF
t13a	BRDY# Rising Edge Valid Delay from HCLKIN Rising	2.1	7.9	78	0 pF
t13b	BRDY# Falling Edge Valid Delay from HCLKIN Rising	2.1	7.9	78	0 pF
t14	NA# Valid Delay from HCLKIN Rising	1.4	8.4	78	0 pF
t15a	AHOLD Valid Delay from HCLKIN Rising	2.0	7.6	78	0 pF
t15b	BOFF# Valid Delay from HCLKIN Rising	2.0	7.6	78	0 pF
t16a	EADS#, INV, PEN# Valid Delay from HCLKIN Rising	2.0	8.0	78	0 pF
t16b	CPURST Rising Edge Valid Delay from HCLKIN Rising	1.2	7.5	78	0 pF
t16c	CPURST Falling Edge Valid Delay from HCLKIN Rising	1.2	7.5	78	0 pF
t16d	KEN# Valid delay from HCLKIN Rising	1.7	8.2	78	0 pF
t17	INIT High Pulse Width	2 HCLKs		84	0 pF
t18	CPURST High Pulse Width	1 ms		84	0 pF; Hard reset via TRC register

9.6.11 SECOND LEVEL CACHE STANDARD SRAM TIMING, 50 AND 60 MHz (82434NX)

Functional Operating Range (V_{CC} = 4.75V to 5.25V; $V_{CC}3$ = 3.135V to 3.465V; T_{CASE} = 0°C to +85°C)

Symbol	Parameter	Min	Max	Fig	Notes
t20a	CAA[6:3]/CAB[6:3] Propagation Delay from A[6:3]	0	8.5	77	0 pF
t20b	CAA[6:3]/CAB[6:3] Valid Delay from HCLKIN Rising	0	7.2	78	0 pF
t21a	COE[1:0]# Falling Edge Valid Delay from HCLKIN Rising	0	9.0	78	0 pF
t21b	COE[1:0]# Rising Edge Valid Delay from HCLKIN Rising	0	5.5	78	0 pF
t22a	CWE[7:0]#/CBS[7:0]# Falling Edge Valid Delay from HCLKIN Rising	2	14	78	CPU burst or single write to second level cache, 0 pF
t22b	CWE[7:0]#/CBS[7:0]# Rising Edge Valid Delay from HCLKIN Rising	3	15	78	CPU burst or single write to second level cache, 0 pF
t22c	CWE[7:0]#/CBS[7:0]# Valid Delay from HCLKIN Rising	1.4	7.7	78	Cache line Fill, 0 pF
t22d	CWE[7:0]#/CBS[7:0]# Low Pulse Width	14		84	0 pF
t22e	CWE[7:0]#/CBS[7:0]# Driven High before CALE Driven High	-1		85	Last write to second level cache during cache line fill, 0 pF
t22f	CAA[4:3]/CAB[4:3] Valid before CWE[7:0]# Falling	1.5		85	CPU burst write to second level cache, 0 pF
t23	CALE Valid Delay from HCLKIN Rising	0	8	78	0 pF
t24	CR/W[1:0]# Valid Delay from HCLKIN Rising	1.5	8.2	78	0 pF
t25	CBS[1:0]# Valid Delay from HCLKIN Rising; Reads from Cache SRAMs	1.0	12.0	78	0 pF
t26a	CCS[1:0]# Propagation Delay from ADS# Falling		7.0	77	0 pF; First access after powerdown
t26b	CCS[1:0]# Valid Delay from HCLKIN Rising	1.5	8.2	78	0 pF; Entering powerdown

9.6.12 SECOND LEVEL CACHE BURST SRAM TIMING, 50 AND 60 MHz (82434NX)

Functional Operating Range (V_{CC} = 4.75V to 5.25V; $V_{CC}3$ = 3.135V to 3.465V; T_{CASE} = 0°C to +85°C)

Symbol	Parameter	Min	Max	Fig	Notes
t30a	CAA[6:3]/CAB[6:3] Propagation Delay from A[6:3]	0	8.5	77	0 pF
t30b	CAA[6:3]/CAB[6:3] Valid Delay from HCLKIN Rising	0	8.2	78	0 pF
t31	CADS[1:0]# Valid Delay from HCLKIN Rising	1.5	8.2	78	0 pF
t32	CADV[1:0]# Valid Delay from HCLKIN Rising	1.5	8.2	78	0 pF
t33	CWE[7:0]# Valid Delay from HCLKIN Rising	1.0	10.5	78	0 pF
t34a	COE[1:0]# Falling Edge Valid Delay from HCLKIN Rising	0	9.5	78	0 pF
t34b	COE[1:0]# Rising Edge Valid Delay from HCLKIN Rising	0	6.0	78	0 pF
t35	CALE Valid Delay from HCLKIN Rising	0	8.5	78	0 pF

2

9.6.13 DRAM INTERFACE TIMING, 50 AND 60 MHz (82434NX)

Functional Operating Range (V_{CC} = 4.75V to 5.25V; $V_{CC}3$ = 3.135V to 3.465V; T_{CASE} = 0°C to +85°C)

Symbol	Parameter	Min	Max	Fig	Notes
t40a	RAS[7:0]# Valid Delay from HCLKIN Rising	0	8.0	78	50 pF
t40b	RAS[7:0]# Pulse Width High	4 HCLKs–5		84	RAS# precharge at beginning of page miss cycle, 50 pF
t41a	CAS[7:0]# Valid Delay from HCLKIN Rising	0	8.0	78	50 pF
t41b	CAS[7:0]# Pulse Width High	1 HCLK–5		84	CAS# precharge during burst cycles, 50 pF
t42	WE# Valid Delay from HCLKIN Rising	0	21	78	50 pF
t43a	MA[10:0] Propagation Delay from A[23:8]	0	23	77	50 pF
t43b	MA[10:0] Valid Delay from HCLKIN Rising	0	10.7	78	50 pF
t43c	MA11 Propagation Delay from A[25:24]	0	24.3	77	50 pF
t43d	MA11 Valid Delay from HCLKIN Rising	0	12	78	50 pF

9.6.14 PCI CLOCK TIMING, 50 AND 60 MHz (82434NX)

Functional Operating Range (V_{CC} = 4.75V to 5.25V; $V_{CC}3$ = 3.135V to 3.465V; T_{CASE} = 0°C to +85°C)

Symbol	Parameter	Min	Max	Fig	Notes
t50a	PCLKOUT High Time	13		82	20 pF
t50b	PCLKOUT Low Time	13		82	20 pF
t51a	PCLKIN High Time	12		82	
t51b	PCLKIN Low Time	12		82	
t51c	PCLKIN Rise Time		3	83	
t51d	PCLKIN Fall Time		3	83	

9.6.15 PCI INTERFACE TIMING, 50 AND 60 MHz (82434NX)

Functional Operating Range (V_{CC} = 4.75V to 5.25V; $V_{CC}3$ = 3.135V to 3.465V; T_{CASE} = 0°C to +85°C)

Symbol	Parameter	Min	Max	Fig	Notes
t60a	C/BE[3:0]#, FRAME#, TRDY#, IRDY#, STOP#, PLOCK#, PAR, PERR#, SERR#, DEVSEL# Valid Delay from PCLKIN Rising	2	11	78	Min: 0 pF Max: 50 pF
t60b	C/BE[3:0]#, FRAME#, TRDY#, IRDY#, STOP#, PLOCK#, PAR, PERR#, SERR#, DEVSEL# Output Enable Delay from PCLKIN Rising	2		81	
t60c	C/BE[3:0]#, FRAME#, TRDY#, IRDY#, STOP#, PLOCK#, PAR, PERR#, SERR#, DEVSEL# Float Delay from PCLKIN Rising	2	28	80	
t60d	C/BE[3:0]#, FRAME#, TRDY#, IRDY#, STOP#, PLOCK#, PAR, PERR#, SERR#, DEVSEL# Setup Time to PCLKIN Rising	9		79	
t60e	C/BE[3:0]#, FRAME#, TRDY#, IRDY#, STOP#, PLOCK#, PAR, PERR#, SERR#, DEVSEL# Hold Time from PCLKIN Rising	0		79	
t61a	REQ#, MEMACK# Valid Delay from PCLKIN Rising	2	12	78	Min: 0 pF Max: 50 pF
t61b	REQ#, MEMACK# Output Enable Delay from PCLKIN Rising	2		81	
t61c	REQ#, MEMACK# Float Delay from PCLKIN Rising	2	28	80	
t62a	FLSHREQ#, MEMREQ# Setup Time to PCLKIN Rising	12		79	
t62b	FLSHREQ#, MEMREQ# Hold Time from PCLKIN Rising	0		79	
t63a	GNT# Setup Time to PCLKIN Rising	10		79	
t63b	GNT# Hold Time from PCLKIN Rising	0		79	
t64a	MEMCS# Setup Time to PCLKIN Rising	7		79	
t64b	MEMCS# Hold Time from PCLKIN Rising	0		79	
t65	PCIRST# Low Pulse Width	1 ms		84	Hard Reset via TRC Register, 0 pF

9.6.16 LBX INTERFACE TIMING, 50 AND 60 MHz (82434NX)

Functional Operating Range (V_{CC} = 4.75V to 5.25V; $V_{CC}3$ = 3.135V to 3.465V; T_{CASE} = 0°C to +85°C)

Symbol	Parameter	Min	Max	Fig	Notes
t70	HIG[4:0] Valid Delay from HCLKIN Rising	0.8	6.7	78	0 pF
t71	MIG[2:0] Valid Delay from HCLKIN Rising	0.9	6.5	78	0 pF
t72	PIG[3:0] Valid Delay from PCLKIN Rising	1.5	12	78	0 pF
t73	PCIDRV Valid Delay from PCLKIN Rising	1	13	78	0 pF
t74a	MDLE Falling Edge Valid Delay from HCLKIN Rising	0.6	6.8	78	0 pF
t74b	MDLE Rising Edge Valid Delay from HCLKIN Rising	0.6	6.8	85	0 pF
t75a	EOL, PPOUT[1:0] Setup Time to PCLKIN Rising	7.7		79	
t75b	EOL, PPOUT[1:0] Hold Time from PCLKIN Rising	1.0		79	

2

9.6.17 TIMING DIAGRAMS

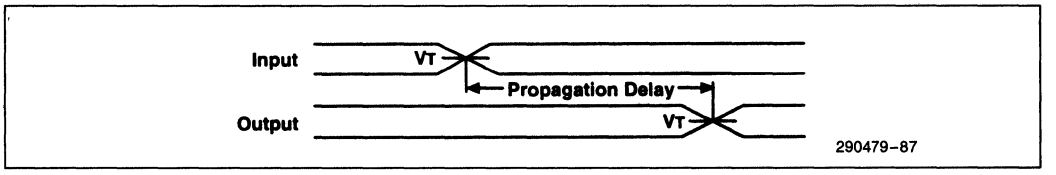


Figure 77. Propagation Delay

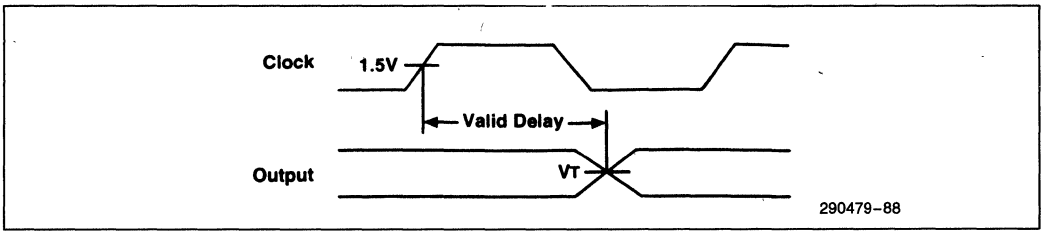


Figure 78. Valid Delay from Rising Clock Edge

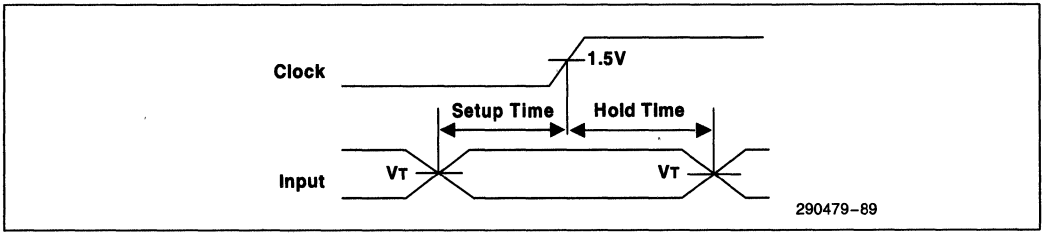


Figure 79. Setup and Hold Times

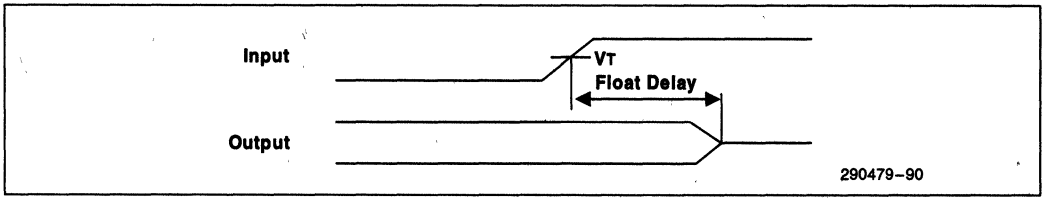


Figure 80. Float Delay

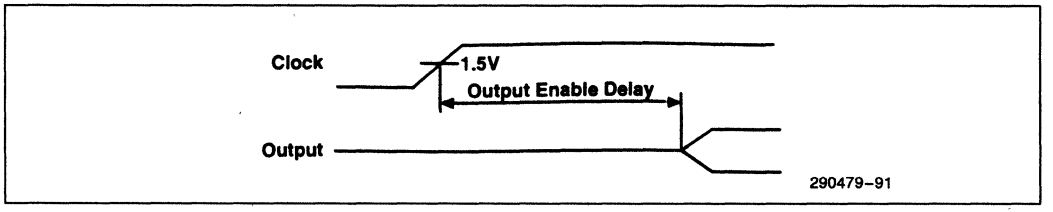


Figure 81. Output Enable Delay

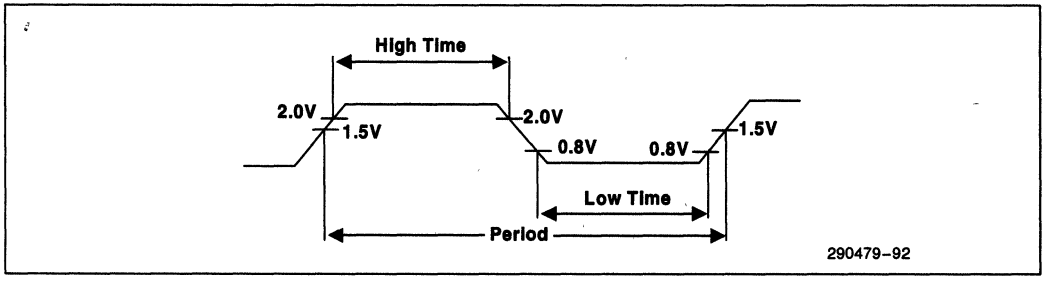


Figure 82. Clock High and Low Times and Period

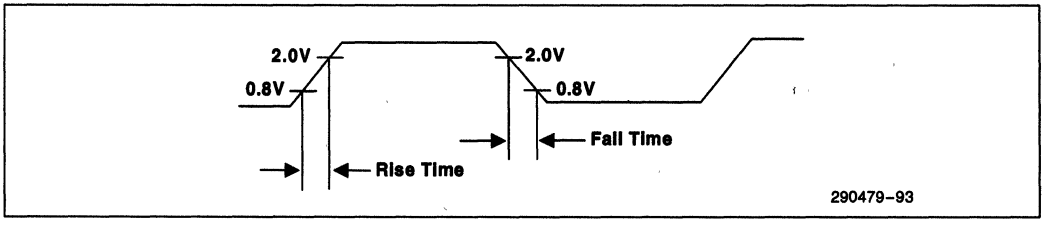


Figure 83. Clock Rise and Fall Times

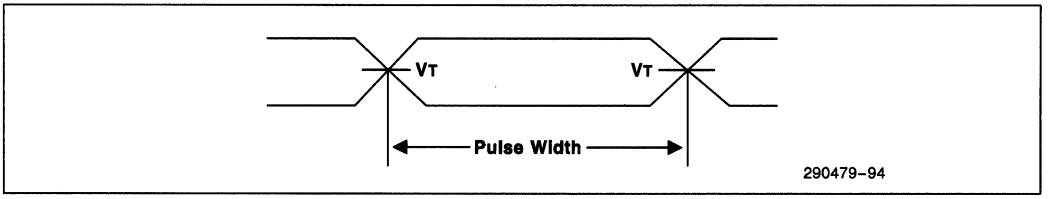


Figure 84. Pulse Width

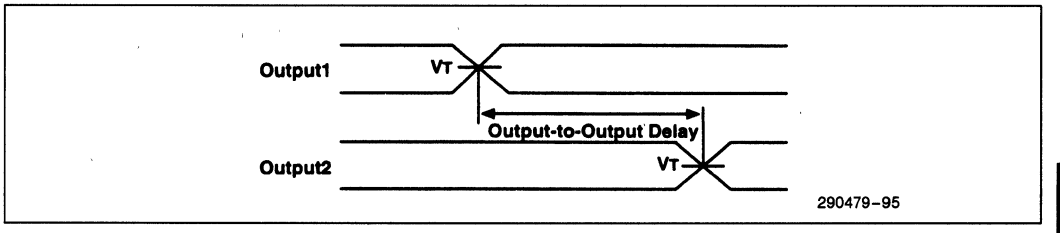
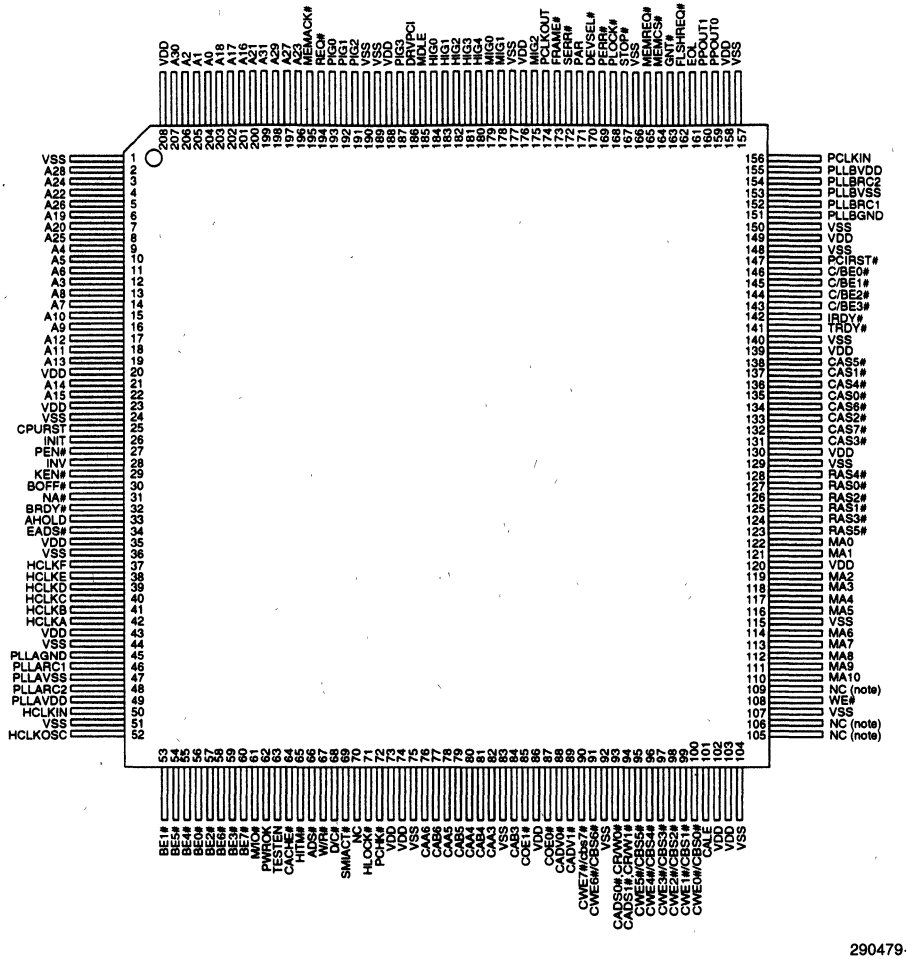


Figure 85. Output-to-Output Delay

10.0 PINOUT AND PACKAGE INFORMATION

10.1 Pin Assignment

Except for the pins listed in Figure 86 notes, the pin assignment for the 82434LX and 82434NX are the same.



290479-96

NOTES:

1. For the 82434NX, pin 105 = RAS6#, 106 = RAS7#, and 109 = MA11. These pins are no connects for the 82434LX and are signal connections for the 82434NX.
2. For the 82434NX, pins 23, 35, 43, 74, 86, and 102 are 3.3V VDD pins (i.e., VDD3). These pins are VDD pins for the 82434LX.

Figure 86. PCMC Pin Assignment

Table 21. 82434LX Alphabetical Pin Assignment

Pin Name	Pin #	Type
A0	204	t/s
A1	205	t/s
A2	206	t/s
A3	12	t/s
A4	9	t/s
A5	10	t/s
A6	11	t/s
A7	14	t/s
A8	13	t/s
A9	16	t/s
A10	15	t/s
A11	18	t/s
A12	17	t/s
A13	19	t/s
A14	21	t/s
A15	22	t/s
A16	201	t/s
A17	202	t/s
A18	203	t/s
A19	6	t/s
A20	7	t/s
A21	200	t/s
A22	4	t/s
A23	196	t/s
A24	3	t/s
A25	8	t/s
A26	5	t/s
A27	197	t/s
A28	2	t/s
A29	198	t/s
A30	207	t/s
A31	199	t/s
ADS#	66	in
AHOLD	33	out
BE0#	56	in
BE1#	53	in
BE2#	57	in
BE3#	59	in
BE4#	55	in
BE5#	54	in
BE6#	58	in
BE7#	60	in
BOFF#	30	out
BRDY#	32	out
CAA3	82	out
CAA4	80	out
CAA5	78	out
CAA6	76	out
CAB3	84	out
CAB4	81	out
CAB5	79	out
CAB6	77	out
CACHE#	64	in
CADS0#,CR/W0#	93	out
CADS1#,CR/W1#	94	out
CADV0# (82434LX) CADV0#/CCS0# (82434NX)	88	out
CADV1# (82434LX) CADV1#/CCS1# (82434NX)	89	out
CALE	101	out
CAS0#	135	out
CAS1#	137	out
CAS2#	133	out
CAS3#	131	out
CAS4#	136	out
CAS5#	138	out
CAS6#	134	out
CAS7#	132	out
CBE0#	146	t/s
CBE1#	145	t/s
CBE2#	144	t/s
CBE3#	143	t/s
COE0#	87	out
COE1#	85	out
CPURST	25	out
CWE0#/CBS0#	100	out
CWE1#/CBS1#	99	out
CWE2#/CBS2#	98	out
CWE3#/CBS3#	97	out
CWE4#/CBS4#	96	out
CWE5#/CBS5#	95	out
CWE6#/CBS6#	91	out
CWE7#/CBS7#	90	out
D/C#	68	in
DEVSEL#	170	s/t/s
DRVPCI	186	out
EADS#	34	out
EOL	161	in
FLSHREQ#	162	in
FRAME#	173	s/t/s
GNT#	163	in
HCLKA	42	out
HCLKB	41	out
HCLKC	40	out
HCLKD	39	out
HCLKE	38	out
HCLKF	37	out
HCLKIN	50	in

Table 21. 82434LX Alphabetical Pin Assignment (Continued)

Pin Name	Pin #	Type
HCLKOSC	52	in
HIG0	184	out
HIG1	183	out
HIG2	182	out
HIG3	181	out
HIG4	180	out
HITM#	65	in
HLOCK#	71	in
INIT	26	out
INV	28	out
IRDY#	142	s/t/s
KEN#	29	out
M/IO#	61	in
MA0	122	out
MA1	121	out
MA2	119	out
MA3	118	out
MA4	117	out
MA5	116	out
MA6	114	out
MA7	113	out
MA8	112	out
MA9	111	out
MA10	110	out
MA11 (82434NX only)	109	out
MDLE	185	out
MEMACK#	195	out
MEMCS#	164	in
MEMREQ#	165	in
MIG0	179	out
MIG1	178	out
MIG2	175	out
NA#	31	out
NC	70	NC
NC (82434LX only)	105	NC
NC (82434LX only)	106	NC
NC (82434LX only)	109	NC
PAR	171	t/s
PCHK#	72	in
PCIRST#	147	out
PCLKIN	156	in
PCLKOUT	174	out
PEN#	27	out
PERR#	169	s/o/d
PIG0	193	out
PIG1	192	out
PIG2	191	out
PIG3	187	out
PLLAGND	45	V
PLLARC1	46	in
PLLARC2	48	in
PLLAVDD	49	V
PLLAVSS	47	V
PLLBGND	151	V
PLLBRC1	152	in
PLLBRC2	154	in
PLLBVDD	155	V
PLLBVSS	153	V
PLOCK#	168	s/t/s
PPOUT0	159	in
PPOUT1	160	in
PWROK	62	in
RAS0#	127	out
RAS1#	125	out
RAS2#	126	out
RAS3#	124	out
RAS4#	128	out
RAS5#	123	out
RAS6# (82434NX only)	105	out
RAS7# (82434NX only)	106	out

Table 21. 82434LX Alphabetical Pin Assignment (Continued)

Pin Name	Pin #	Type
REQ#	194	out
SERR#	172	s/o/d
SMIACT#	69	in
STOP#	167	s/t/s
TESTEN	63	in
TRDY#	141	s/t/s
V_{DD}	20	V
V_{DD} (82434LX) V_{DD3} (82434NX)	23	V
V_{DD} (82434LX) V_{DD3} (82434NX)	35	V
V_{DD} (82434LX) V_{DD3} (82434NX)	43	V
V_{DD}	73	V
V_{DD} (82434LX) V_{DD3} (82434NX)	74	V
V_{DD} (82434LX) V_{DD3} (82434NX)	86	V
V_{DD} (82434LX) V_{DD3} (82434NX)	102	V

Pin Name	Pin #	Type
V_{DD}	103	V
V_{DD}	120	V
V_{DD}	130	V
V_{DD}	139	V
V_{DD}	149	V
V_{DD}	158	V
V_{DD}	176	V
V_{DD}	188	V
V_{DD}	208	V
V_{SS}	1	V
V_{SS}	24	V
V_{SS}	36	V
V_{SS}	44	V
V_{SS}	51	V
V_{SS}	75	V
V_{SS}	83	V

Pin Name	Pin #	Type
V_{SS}	92	V
V_{SS}	104	V
V_{SS}	107	V
V_{SS}	115	V
V_{SS}	129	V
V_{SS}	140	V
V_{SS}	148	V
V_{SS}	150	V
V_{SS}	157	V
V_{SS}	166	V
V_{SS}	177	V
V_{SS}	189	V
V_{SS}	190	V
W/R#	67	in
WE#	108	out

Table 22. Numerical Pin Assignment

Pin #	Pin Name	Type
1	V_{SS}	V
2	A28	t/s
3	A24	t/s
4	A22	t/s
5	A26	t/s
6	A19	t/s
7	A20	t/s
8	A25	t/s
9	A4	t/s
10	A5	t/s
11	A6	t/s
12	A3	t/s
13	A8	t/s
14	A7	t/s
15	A10	t/s
16	A9	t/s
17	A12	t/s
18	A11	t/s
19	A13	t/s
20	V_{DD}	V
21	A14	t/s
22	A15	t/s
23	V_{DD} (82434LX) V_{DD3} (82434NX)	V
24	V_{SS}	V
25	CPURST	out
26	INIT	out
27	PEN#	out
28	INV	out
29	KEN#	out
30	BOFF#	out
31	NA#	out
32	BRDY#	out
33	AHOLD	out
34	EADS#	out
35	V_{DD} (82434LX) V_{DD3} (82434NX)	V
36	V_{SS}	V
37	HCLKF	out
38	HCLKE	out
39	HCLKD	out
40	HCLKC	out
41	HCLKB	out
42	HCLKA	out
43	V_{DD} (82434LX) V_{DD3} (82434NX)	V
44	V_{SS}	V
45	PLLAGND	V
46	PLLARC1	in
47	PLLAVSS	V
48	PLLARC2	in
49	PLLAVDD	V
50	HCLKIN	in
51	V_{SS}	V
52	HCLKOSC	in
53	BE1#	in
54	BE5#	in
55	BE4#	in
56	BE0#	in
57	BE2#	in
58	BE6#	in
59	BE3#	in
60	BE7#	in
61	M/IO#	in
62	PWROK	in
63	TESTEN	in
64	CACHE#	in
65	HITM#	in
66	ADS#	in
67	W/R#	in
68	D/C#	in
69	SMIACT#	in
70	NC	NC
71	HLOCK#	in
72	PCHK#	in
73	V_{DD}	V
74	V_{DD} (82434LX) V_{DD3} (82434NX)	V
75	V_{SS}	V
76	CAA6	out
77	CAB6	out
78	CAA5	out
79	CAB5	out
80	CAA4	out
81	CAB4	out
82	CAA3	out
83	V_{SS}	V
84	CAB3	out
85	COE1#	out
86	V_{DD} (82434LX) V_{DD3} (82434NX)	V
87	COE0#	out
88	CADV0# (82434LX) CADV0#/CCS0# (82434NX)	out
89	CADV1# (82434LX) CADV1#/CCS1# (82434NX)	out

Table 22. Numerical Pin Assignment (Continued)

Pin #	Pin Name	Type
90	CWE7#/CBS7#	out
91	CWE6#/CBS6#	out
92	V_{SS}	V
93	CADS0#,CR/W0#	out
94	CADS1#,CR/W1#	out
95	CWE5#/CBS5#	out
96	CWE4#/CBS4#	out
97	CWE3#/CBS3#	out
98	CWE2#/CBS2#	out
99	CWE1#/CBS1#	out
100	CWE0#/CBS0#	out
101	CALE	out
102	V_{DD} (82434LX) V_{DD3} (82434NX)	V
103	V_{DD}	V
104	V_{SS}	V
105	NC (82434LX) RAS6# (82434NX)	NC out
106	NC (82434LX) RAS7# (82434NX)	NC out
107	V_{SS}	V
108	WE#	out
109	NC (82434LX) MA11 (82434NX)	NC out
110	MA10	out
111	MA9	out
112	MA8	out
113	MA7	out
114	MA6	out
115	V_{SS}	V
116	MA5	out
117	MA4	out
118	MA3	out

Pin #	Pin Name	Type
119	MA2	out
120	V_{DD}	V
121	MA1	out
122	MA0	out
123	RAS5#	out
124	RAS3#	out
125	RAS1#	out
126	RAS2#	out
127	RAS0#	out
128	RAS4#	out
129	V_{SS}	V
130	V_{DD}	V
131	CAS3#	out
132	CAS7#	out
133	CAS2#	out
134	CAS6#	out
135	CAS0#	out
136	CAS4#	out
137	CAS1#	out
138	CAS5#	out
139	V_{DD}	V
140	V_{SS}	V
141	TRDY#	s/t/s
142	IRDY#	s/t/s
143	CBE3#	t/s
144	CBE2#	t/s
145	CBE1#	t/s
146	CBE0#	t/s
147	PCIRST#	out
148	V_{SS}	V
149	V_{DD}	V
150	V_{SS}	V

Pin #	Pin Name	Type
151	PLLBGND	V
152	PLLBRC1	in
153	PLLBVSS	V
154	PLLBRC2	in
155	PLLBVDD	V
156	PCLKIN	in
157	V_{SS}	V
158	V_{DD}	V
159	PPOUT0	in
160	PPOUT1	in
161	EOL	in
162	FLSHREQ#	in
163	GNT#	in
164	MEMCS#	in
165	MEMREQ#	in
166	V_{SS}	V
167	STOP#	s/t/s
168	PLOCK#	s/t/s
169	PERR#	s/o/d
170	DEVSEL#	s/t/s
171	PAR	t/s
172	SERR#	s/o/d
173	FRAME#	s/t/s
174	PCLKOUT	out
175	MIG2	out
176	V_{DD}	V
177	V_{SS}	V
178	MIG1	out
179	MIG0	out
180	HIG4	out
181	HIG3	out
182	HIG2	out

Table 22. Numerical Pin Assignment (Continued)

Pin #	Pin Name	Type
183	HIG1	out
184	HIG0	out
185	MDLE	out
186	DRVPCI	out
187	PIG3	out
188	V_{DD}	V
189	V_{SS}	V
190	V_{SS}	V
191	PIG2	out

Pin #	Pin Name	Type
192	PIG1	out
193	PIG0	out
194	REQ#	out
195	MEMACK#	out
196	A23	t/s
197	A27	t/s
198	A29	t/s
199	A31	t/s
200	A21	t/s

Pin #	Pin Name	Type
201	A16	t/s
202	A17	t/s
203	A18	t/s
204	A0	t/s
205	A1	t/s
206	A2	t/s
207	A30	t/s
208	V_{DD}	V

10.2 Package Characteristics

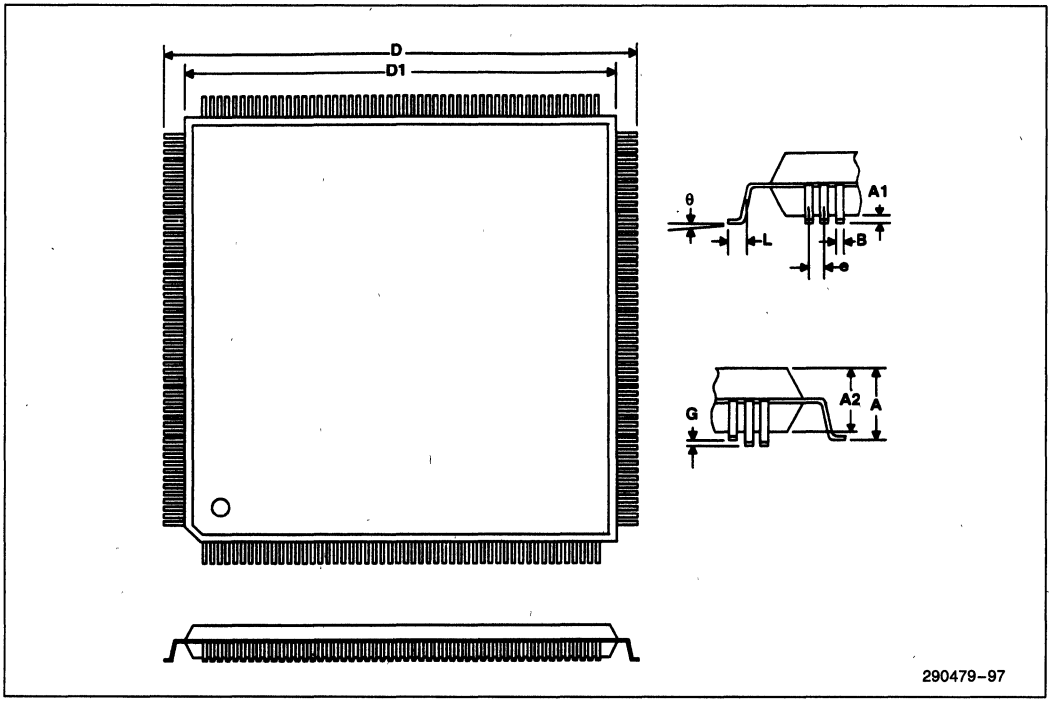


Figure 87. 208-Pin Quad Flatpack (QFP) Dimensions

Table 23. 82434LX Package Dimensions

Symbol	Description	Value (mm)
A	Seating Height	3.5 (max)
A1	Stand-Off Height	0.20–0.50
A2	Package Height	3.0 (nominal)
B	Lead Width	0.18 +0.1/−0.05
D	Package Length and Width, Including Pins	30.6 ± 0.3
D1	Package Length and Width, Excluding Pins	28 ± 0.1
e	Linear Lead Pitch	0.5 ± 0.1
G	Lead Coplanarity	0.1 (max)
L	Lead Length	0.5 ± 0.2
θ	Lead Angle	0°–10°

Table 24. 82434NX Package Dimensions

Symbol	Description	Value (mm)
A	Seating Height	3.7 (max)
A1	Stand-Off Height	0.05–0.50
A2	Package Height	3.45 (max)
B	Lead Width	0.13–0.27
D	Package Length and Width, Including Pins	30.6 ± 0.3
D1	Package Length and Width, Excluding Pins	28 ± 0.1
e	Linear Lead Pitch	0.5 (nominal)
G	Lead Coplanarity	0.1 (max)
L	Lead Length	0.5 ± 0.2
θ	Lead Angle	0°–10°

11.0 TESTABILITY

A NAND tree is provided in the 82434LX and 82434NX PCMCs for Automated Test Equipment (ATE) board level testing. The NAND tree allows the tester to test the connectivity of a subset of the PCMC signal pins.

For the 82434LX, the output of the NAND tree is driven on pin 109. The NAND tree is enabled when A24 = 1, A25 = 0, A26 = 1, and TESTEN = 1 at the rising edge of PWROK. PLL Bypass mode is enabled when A24 = 1, and TESTEN = 1 at the rising edge of PWROK. In PLL Bypass mode, the 82434LX and 82434NX PCMC AC specifications are affected as follows:

1. Output valid delays increase by 20 ns.
2. All hold times are 20 ns.
3. Setup times and propagation delays are unaffected.
4. Input clock high and low times are 100 ns.

In both the NAND tree test mode and PLL Bypass mode, TESTEN must remain asserted throughout the testing. A[28:24] should be set up at least 1 HCLK before the rising edge of PWROK and held at least 3 HCLKs after PWROK. Table 11 shows the order of the NAND tree inside the PCMC.

When not in NAND Tree test mode, the 82434LX drives the output of the host clock PLL onto pin 109.

82434NX Test Modes

The state of A[28:24], TESTEN, CPURST, and PWROK can place the 82434NX PCMC into two test modes. When PWROK is low, A[27:24] and TESTEN directly control the mode of operation of the PCMC. When PWROK is high, the state of A[27:24] and TESTEN are latched and the PCMC remains in the indicated mode until PWROK is again negated. The high order LBX samples the state of A27 on the falling edge of CPURST.

When PWROK is low and both TESTEN and A27 are low, the 82434NX drives MA11 onto pin 109. If both TESTEN and A27 are low when PWROK transitions from low to high, the PCMC continues to drive MA11 onto pin 109. If the high order LBX samples A27 low on the falling edge of CPURST, it will tri-state pin 123.

When PWROK is low, TESTEN is low, and A27 is high the PCMC drives the output of the host clock PLL onto pin 109. Observing pin 109 when in this mode indicates if the host clock PLL has locked onto the correct frequency. If TESTEN is low and A27 is high when PWROK transitions from low to high the PCMC continues to drive the output of the host clock PLL onto pin 109, regardless of the values of TESTEN and A27. If the high order LBX samples A27 high on the falling edge of CPURST, it drives the output of its host clock PLL onto pin 123. No phase delay information can be inferred from these outputs.

When PWROK is low, TESTEN is high, A26 is high, A25 is low, A28 is high and A24 is high, the PCMC will drive the output of the NAND tree onto pin 109. If TESTEN is high, A26 is high, and A25 is low when PWROK transitions from low to high, the PCMC continues to drive the output of the NAND tree onto pin 109.

A27 must be pulled low via a pulldown resistor to ground for normal operation.

Table 25. NAND Tree Order

Order	Pin #	Signal
1	141	TRDY#
2	142	IRDY#
3	143	CBE3#
4	144	CBE2#
5	145	CBE1#
6	146	CBE0#
7	159	PPOUT0
8	160	PPOUT1
9	161	EOL
10	162	FLSHBUF#
11	163	GNT#
12	164	MEMCS#
13	165	MEMREQ#
14	167	STOP#
15	168	PLOCK#
16	169	PERR#
17	170	DEVSEL#
18	171	PAR
19	172	SERR#
20	173	FRAME#
21	194	REQ#
22	196	A23
23	197	A27
24	198	A29
25	199	A31
26	200	A21
27	201	A16
28	202	A17
29	203	A18
30	204	A0
31	205	A1
32	206	A2
33	207	A30
34	2	A28
35	3	A24
36	4	A22
37	5	A26
38	6	A19
39	7	A20
40	8	A25
41	9	A4
42	10	A5
43	11	A6
44	12	A3
45	13	A8
46	14	A7
47	15	A10
48	16	A9
49	17	A12
50	18	A11
51	19	A13
52	21	A14
53	22	A15
54	53	BE1#
55	54	BE5#
56	55	BE4#
57	56	BE0#
58	57	BE2#
59	58	BE6#
60	59	BE3#
61	60	BE7#
62	61	M/IO#
63	64	CACHE#
64	65	HITM#
65	66	ADS#
66	67	W/R#
67	68	D/C#
68	69	SMIACT#
69	71	HLOCK#
70	72	PCHK#
71	63	TESTEN

ADDITIONAL TESTING NOTES:

1. HCLKOUT[6:1] can be toggled via HCLKIN.
2. CAx[6:3] are flow through outputs via A[6:3] after PWROK transitions high.
3. MA[10:0] are flow through outputs via A[13:3] after PWROK transitions high.
4. CAS[7:0]# outputs can be tested by performing a DRAM read cycle.
5. PCLKOUT can be tested in PLL bypass mode, frequency is HCLK/2.
6. PCIRST is the NAND Tree output of Tree Cell 6.
7. INIT is the NAND Tree output of Tree Cell 53.

82420/82430
PCIset BRIDGE COMPONENT

82378ZB (SIO), 82379AB (SIO.A) FOR ISA BUSES

- Provides the Bridge between the PCI Bus and ISA Bus
- 100% PCI and ISA Compatible
- Enhanced DMA Functions (82378ZB Only)
- Integrated Data Buffers to Improve Performance
- Integrated 16-bit BIOS Timer
- Arbitration for PCI Devices
- Arbitration for ISA Devices
- Integrates the Functionality of One 82C54 Timer
- Integrates the Functionality of Two 82C59 Interrupt Controllers
- Non-Maskable Interrupts (NMI)
- Four Dedicated PCI Interrupts
- Complete Support for SL Enhanced Intel486™ CPU's
- Integrated Power Management Support
  - System Management Interrupts
  - Fast Off Timer
  - STPCLK# Signal to Throttle CPU Clock
  - APM Port
- Provides I/O APIC for Dual-Processor (DP) Support

82374EB/SB (ESC), 82375EB/SB (PCEB) FOR EISA BUSES

- Provides the Bridge between the PCI Bus and EISA Bus
- 100% PCI and EISA Compatible
- Data Buffers Improve Performance
- Data Buffer Management Ensures Data Coherency
- Burst Transfers on both the PCI and EISA Buses
- 32-Bit Data Paths
- PCI and EISA Address Decoding and Mapping
- Programmable Main Memory Address Decoding
- Integrated EISA Compatible Bus Controller
- Supports Eight EISA Slots
- Provides Enhanced DMA Controller
- Provides High Performance Arbitration
- Integrates Support Logic for X-Bus Peripheral and more
- Integrates the Functionality of Two 82C59 Interrupt Controllers and Two 82C54 Timers
- Generates Non-Maskable Interrupts
- Provides BIOS Interface

The 82420/82430 PCIset Bridge components provide a bridge between the PCI to either EISA or ISA buses. The 82378 provides the bridge between PCI bus and the ISA bus while the 82374 and 83275 together provide the bridge between the PCI bus and the EISA bus.

The SIO integrates many of the common I/O functions found in todays ISA based PC systems. The SIO incorporates the logic for a PCI interface (master and slave), ISA interface (master and slave), enhanced seven channel DMA controller and support for other decode logic. The 82379AB adds an APIC for dual-processing Pentium™ Processor systems.

November 1994
Order Number: 290509-002

The 82374 EISA System Component (ESC) and 83275 PCI-EISA Bridge (PCEB) together provide the EISA system compatible master/slave functions on both the PCI Local Bus and the EISA Bus and the common I/O functions found in today's EISA systems. The ESC incorporates the logic for an EISA (master and slave) interface, EISA bus controller, enhanced seven channel DMA controller with Scatter-Gather support, EISA arbitration, 14 channel interrupt controller, five programmable timer/counters and non-maskable control logic. The ESC also integrates support logic to decode peripheral devices such as the Flash BIOS, Real Time Clock, Keyboard/Mouse Controller, Floppy Controller, two Serial Ports, one Parallel Port, and IDE Hard Disk Drive. The PCEB provides the address and data paths, bus controls, and bus protocol translation for PCI-to-EISA and EISA-to-PCI transfers. Extensive data buffering in both directions increases system performance by maximizing PCI and EISA Bus efficiency and allowing concurrency on the two buses. The PCEB integrates central bus control functions, PCI parity generation, system error reporting, and programmable PCI and EISA memory and I/O address space mapping and decoding.

82374EB/82374SB EISA SYSTEM COMPONENT (ESC)

- Integrates EISA Compatible Bus Controller
  - Translates Cycles between EISA and ISA Bus
  - Supports EISA Burst and Standard Cycles
  - Supports ISA Zero Wait-State Cycles
  - Supports Byte Assembly/ Disassembly for 8-, 16- and 32-Bit Transfers
  - Supports EISA Bus Frequency of Up to 8.33 MHz
- Supports Eight EISA Slots
  - Directly Drives Address, Data and Control Signals for Eight Slots
  - Decodes Address for Eight Slot Specific AENs
- Provides Enhanced DMA Controller
  - Provides Scatter-Gather Function
  - Supports Type A, Type B, Type C (Burst), and Compatible DMA Transfer
  - Provides Seven Independently Programmable Channels
  - Integrates Two 82C37A Compatible DMA Controllers
- Integrates the Functionality of Two 82C59 Interrupt Controllers and Two 82C54 Timers
  - Provides 14 Programmable Channels for Edge or Level Interrupts
  - Provides 4 PCI Interrupts Routable to any of 11 Interrupt Channels
  - Supports Timer Function for Refresh Request, System Timer, Speaker Tone, Fail Safe Timer, and CPU Speed Control
- Advanced Programmable Interrupt Controller (APIC)
  - Multiprocessor Interrupt Management
  - Separate Bus for Interrupt Messages
- 5V CMOS Technology
- Provides High Performance Arbitration
  - Supports Eight EISA Masters and PCEB
  - Supports ISA Masters, DMA Channels, and Refresh
  - Provides Programmable Arbitration Scheme for Fixed, Rotating, or Combination Priority
- Integrates Support Logic for X-Bus Peripherals
  - Generates Chip Selects/Encoded Chip Selects for Floppy and Keyboard Controller, IDE, Parallel/ Serial Ports, and General Purpose Peripherals
  - Provides Interface for Real Time Clock
  - Generates Control Signals for X-Bus Data Transceiver
  - Integrates Port 92, Mouse Interrupt, and Coprocessor Error Reporting
- Generates Non-Maskable Interrupts (NMI)
  - PCI System Errors
  - PCI Parity Errors
  - EISA Bus Parity Errors
  - Fail Safe Timer
  - Bus Timeout
  - Via Software Control
- Provides BIOS Interface
  - Supports 512 KBytes of Flash or EPROM BIOS on the X-Bus
  - Allows BIOS on PCI
  - Supports Integrated VGA BIOS
- 82374SB System Power Management (Intel SMM Support)
  - Fast On/Off Support via SMI Generation—Hardware Events, Software Events, EXTSMI#, Fast Off Timer, System Events
  - Programmable CPU Clock Control
  - Enables Energy Efficient Desktop Systems
- 208-Pin QFP Package

November 1994
Order Number: 290476-004

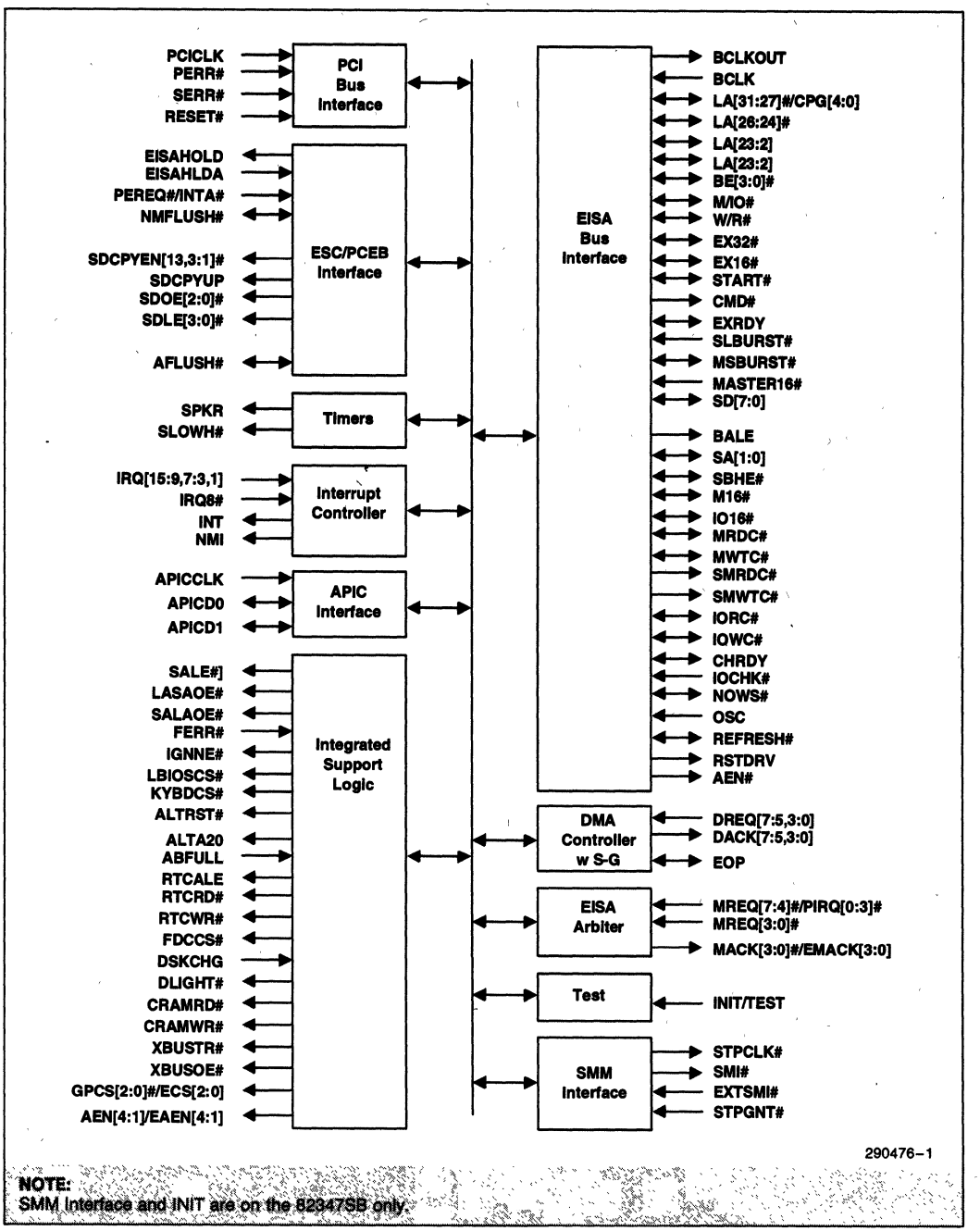
IMPORTANT—READ THIS SECTION BEFORE READING THE REST OF THE DATA SHEET.

This data sheet describes the 82374EB and 82374SB components. All normal text describes the functionality for both components. All features that exist on the 82374SB are shaded as shown below.

This is an example of what the shaded sections that apply only to the 82374SB component look like.

The 82374EB/SB EISA System Component (ESC) provides all the EISA system compatible functions. The ESC with the PCEB provide all the functions to implement an EISA-to-PCI bridge and EISA I/O subsystem. The ESC integrates the common I/O functions found in today's EISA-based PC systems. The ESC incorporates the logic for an EISA (master and slave) interface, EISA bus controller, enhanced seven channel DMA controller with scatter-gather support, EISA arbitration, 14 channel interrupt controller, Advanced Programmable Interrupt Controller (APIC), five programmable timer/counters, and non-maskable-interrupt (NMI) control logic. The ESC also integrates support logic to decode peripheral devices such as the Flash BIOS, real time clock, keyboard/mouse controller, floppy controller, two serial ports, one parallel port, and IDE hard disk drive.

The 82374SB also contains support for SMM power management.



NOTE:
SMM Interface and INIT are on the 82347SB only.

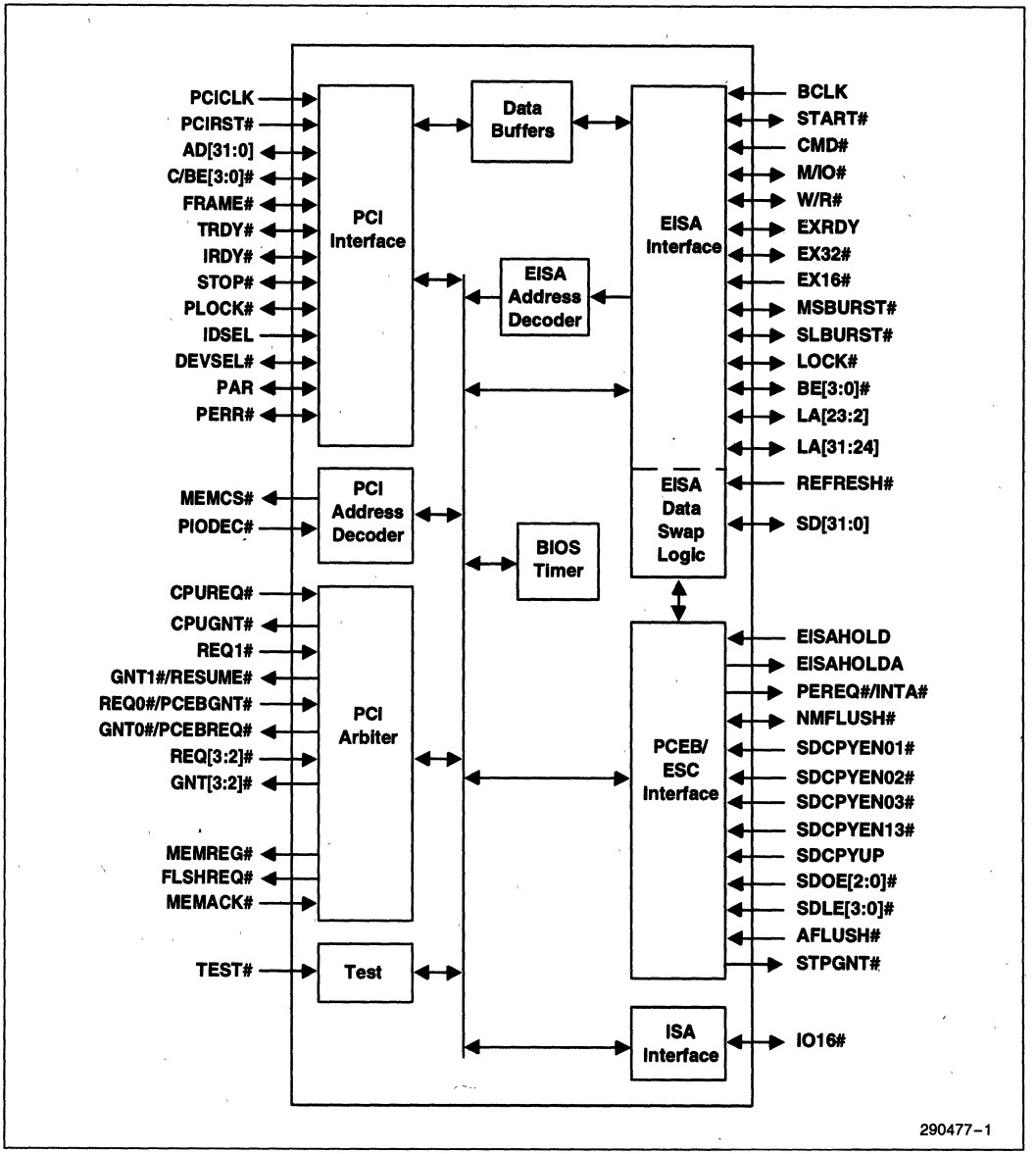
Simplified ESC Block Diagram

82375EB/82375SB PCI-EISA BRIDGE (PCEB)

- Provides the Bridge Between the PCI Local Bus and EISA Bus
- 100% PCI and EISA Compatible
  - PCI and EISA Master/Slave Interface
  - Directly Drives 10 PCI Loads and 8 EISA Slots
  - Supports PCI from 25 to 33 MHz
- Data Buffers Improve Performance
  - Four 32-Bit PCI-to-EISA Posted Write Buffers
  - Four 16-Byte EISA-to-PCI Read/Write Line Buffers
  - EISA-to-PCI Read Prefetch
  - EISA-to-PCI and PCI-to-EISA Write Posting
- Data Buffer Management Ensures Data Coherency
  - Flush Posted Write Buffers
  - Flush or Invalidate Line Buffers
  - System-Wide Data Buffer Coherency Control
- Burst Transfers on Both the PCI and EISA Buses
- 32-Bit Data Paths
- Integrated EISA Data Swap Buffers
- Arbitration for PCI Devices
  - Supports Six PCI Masters
  - Fixed, Rotating, or a Combination of the Two
  - Supports External PCI Arbiter and Arbiter Cascading
- PCI and EISA Address Decoding and Mapping
  - Positive Decode of Main Memory Areas (MEMCS# Generation)
  - Four Programmable PCI Memory Space Regions
  - Four Programmable PCI I/O Space Regions
- Programmable Main Memory Address Decoding
  - Main Memory Sizes Up to 512 MBytes
  - Access Attributes for 15 Memory Segments in First 1 MByte of Main Memory
  - Programmable Main Memory Hole
- Integrated 16-Bit BIOS Timer

The 82375EB/SB PCI-EISA Bridge (PCEB) provides the master/slave functions on both the PCI Local Bus and the EISA Bus. Functioning as a bridge between the PCI and EISA buses, the PCEB provides the address and data paths, bus controls, and bus protocol translation for PCI-to-EISA and EISA-to-PCI transfers. Extensive data buffering in both directions increases system performance by maximizing PCI and EISA Bus efficiency and allowing concurrency on the two buses. The PCEB's buffer management mechanism ensures data coherency. The PCEB integrates central bus control functions including a programmable bus arbiter for the PCI Bus and EISA data swap buffers for the EISA Bus. Integrated system functions include PCI parity generation, system error reporting, and programmable PCI and EISA memory and I/O address space mapping and decoding. The PCEB also contains a BIOS Timer that can be used to implement timing loops. The PCEB is intended to be used with the EISA System Component (ESC) to provide an EISA I/O subsystem interface.

November 1994
Order Number: 290477-004



PCEB Simplified Block Diagram

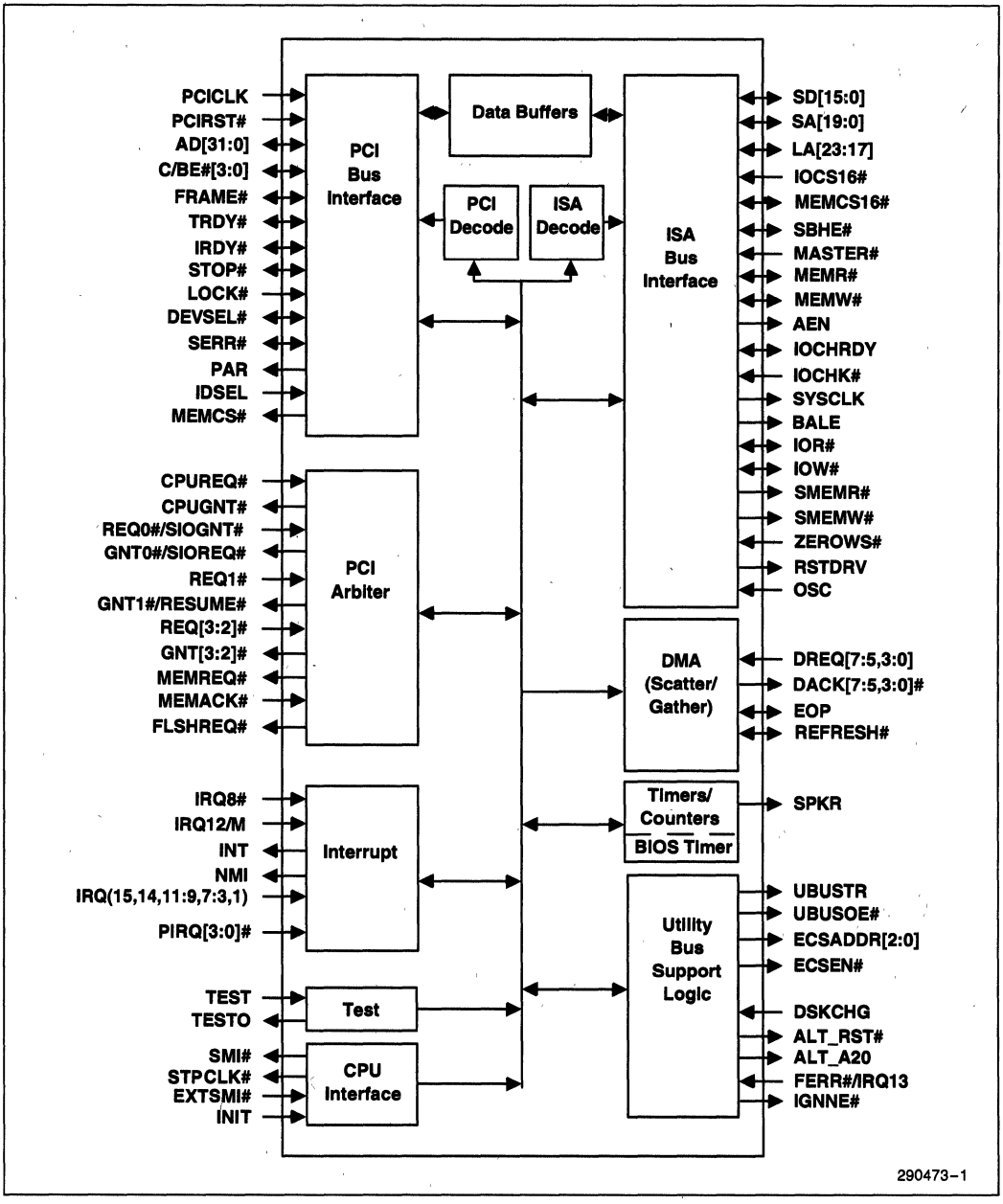
82378 SYSTEM I/O (SIO)

- **Provides the Bridge Between the PCI Bus and ISA Bus**
- **100% PCI and ISA Compatible**
  - **— PCI and ISA Master/Slave Interface**
  - **— Directly Drives 10 PCI Loads and 6 ISA Slots**
  - **— Supports PCI at 25 MHz and 33 MHz**
  - **— Supports ISA from 6 MHz to 8.33 MHz**
- **Enhanced DMA Functions**
  - **— Scatter/Gather**
  - **— Fast DMA Type A, B and F**
  - **— Compatible DMA Transfers**
  - **— 32-bit Addressability**
  - **— Seven Independently Programmable Channels**
  - **— Functionality of Two 82C37A DMA Controllers**
- **Integrated Data Buffers to Improve Performance**
  - **— 8-Byte DMA/ISA Master Line Buffer**
  - **— 32-bit Posted Memory Write Buffer to ISA**
- **Integrated 16-bit BIOS Timer**
- **Non-Maskable Interrupts (NMI)**
  - **— PCI System Errors**
  - **— ISA Parity Errors**
- **Arbitration for ISA Devices**
  - **— ISA Masters**
  - **— DMA and Refresh**
- **Four Dedicated PCI Interrupts**
  - **— Level Sensitive**
  - **— Can be Mapped to Any Unused Interrupt**
- **Arbitration for PCI Devices**
  - **— Six PCI Masters Supported**
  - **— Fixed, Rotating, or a Combination of the Two**
- **Utility Bus (X-Bus) Peripheral Support**
  - **— Provides Chip Select Decode**
  - **— Controls Lower X-Bus Data Byte Transceiver**
- **Integrates the Functionality of One 82C54 Timer**
  - **— System Timer**
  - **— Refresh Request**
  - **— Speaker Tone Output**
- **Integrates the Functionality of Two 82C59 Interrupt Controllers**
  - **— 14 Interrupts Supported**
  - **— Edge/Level Selectable Interrupts: Each Interrupt Individually Programmable**
- **Complete Support for SL Enhanced Intel486 CPU's**
  - **— SMI# Generation Based on System Hardware Events**
  - **— STPCLK# Generation to Power Down the CPU**

The 82378 System I/O (SIO) component provides the bridge between the PCI bus and the ISA expansion bus. The SIO also integrates many of the common I/O functions found in today's ISA based PC systems. The SIO incorporates the logic for a PCI interface (master and slave), ISA interface (master and slave), enhanced seven channel DMA controller that supports fast DMA transfers and Scatter/Gather, data buffers to isolate the PCI bus from the ISA bus and to enhance performance, PCI and ISA arbitration, 14 level interrupt controller, a 16-bit BIOS timer, three programmable timer/counters, and Non-Maskable Interrupt (NMI) Control Logic. The SIO also provides decode for peripheral devices such as the Flash BIOS, Real Time Clock, Keyboard/Mouse Controller, Floppy Controller, two Serial Ports, one Parallel Port, and IDE Hard Disk Drive.

The 82378 also supports several Advanced Power Management features such as SMI#, APM Register, Fast On and Fast Off Event Timers, Clock Throttling, and support for an external SMI# Interrupt. The 82378 also supports a total of 6 PCI Masters, and can support up to 4 PCI Interrupts.

December 1994
Order Number: 290473-004



SIO Component Block Diagram

82378 SYSTEM I/O (SIO)

CONTENTS

CONTENTS

CONTENTS

CONTENTS

PAGE

CONTENTS

CONTENTS

CONTENTS

1.0 ARCHITECTURAL OVERVIEW

The major functions of the SIO component are broken up into blocks as shown in the SIO Component Block Diagram. A description of each block is provided below.

PCI Bus Interface

The PCI Bus Interface provides the interface between the SIO and the PCI bus. The SIO provides both a master and slave interface to the PCI bus. As a PCI master, the SIO runs cycles on behalf of DMA, ISA masters, and the internal data buffer management logic when buffer flushing is required. The SIO will burst a maximum of two Dwords when reading from PCI memory, and one Dword when writing to PCI memory. The SIO does not generate PCI I/O cycles as a master. As a PCI slave, the SIO accepts cycles initiated by PCI masters targeted for the SIO's internal register set or the ISA bus. The SIO will accept a maximum of one data transaction before terminating the transaction. This supports the Incremental Latency Mechanism as defined in the Peripheral Component Interconnect (PCI) Specification.

As a master, the SIO generates address and command signal (C/BE#) parity for read and write cycles, and data parity for write cycles. As a slave, the SIO generates data parity for read cycles. Parity checking is not supported. The SIO also provides support for system error reporting by generating a Non-Maskable-Interrupt (NMI) when SERR# is driven active.

The SIO, as a resource, can be locked by any PCI master. In the context of locked cycles, the entire SIO subsystem (including the ISA bus) is considered a single resource.

The SIO directly supports the PCI Interface running at either 25 MHz or 33 MHz. If a frequency of less than 33 MHz is required (not including 25 MHz), a SYSCLK divisor value (as indicated in the ISA Clock Divisor Register) must be selected that guarantees that the ISA bus frequency does not violate the 6 MHz to 8.33 MHz SYSCLK range.

PCI Arbiter

The PCI arbiter provides support for six PCI masters; the Host Bridge, SIO, and four PCI masters. The arbiter can be programmed for a purely rotating scheme, fixed, or a combination of the two. The Arbiter can also be programmed to support bus parking. This gives the Host Bridge default access to the PCI bus when no other device is requesting service. The arbiter can be disabled if an external arbiter is used.

PCI Decode/ISA Decode

The SIO contains two address decoders; one to decode PCI initiated cycles and one to decode ISA master and DMA initiated cycles. Two decoders are used to allow the PCI and ISA buses to run concurrently.

The SIO is also programmable to provide address decode on behalf of the Host Bridge. When programmed, the SIO monitors the PCI and ISA address buses, and generates a memory chip select signal (MEMCS#) indicating that the current cycle is targeted for system memory residing behind the Host Bridge. This feature can be disabled through software.

Data Buffers

To isolate the slower ISA bus from the PCI bus, the SIO provides two types of data buffers. One Dword deep posted write buffer is provided for the posting of PCI initiated memory write cycles to the ISA bus. The second buffer is a bi-directional, 8-byte line buffer used for ISA master and DMA accesses to the PCI bus. All DMA and ISA master read and write cycles go through the 8-byte line buffer.

The data buffers also provide the data assembly or disassembly when needed for transactions between the PCI and ISA buses.

Buffering is programmable and can be enabled or disabled through software.

ISA Bus Interface

The SIO incorporates a fully ISA-bus compatible master and slave interface. The SIO directly drives six ISA slots without external data or address buffering. The ISA interface also provides byte swap logic, I/O recovery support, wait-state generation, and SYSCLK generation. The SIO supports ISA bus frequencies from 6 MHz to 8.33 MHz.

As an ISA master, the SIO generates cycles on behalf of DMA, Refresh, and PCI master initiated cycles. The SIO supports compressed cycles when accessing ISA slaves (i.e. ZEROWS# asserted). As an ISA slave, the SIO accepts ISA master accesses targeted for the SIO's internal register set or ISA master memory cycles targeted for the PCI bus. The SIO does not support ISA master initiated I/O cycles targeted for the PCI bus.

The SIO also monitors ISA master to ISA slave cycles to generate SMEMR# or SMEMW#, and to support data byte swapping, if necessary.

DMA

The DMA controller incorporates the functionality of two 82C37 DMA controllers with seven independently programmable channels. Each channel can be programmed for 8- or 16-bit DMA device size, and ISA-compatible or fast DMA type "A", type "B", or type "F" timings. Full 32-bit addressing is supported as an extension of the ISA-compatible specification. The DMA controller is also responsible for generating ISA refresh cycles.

The DMA controller supports an enhanced feature called Scatter/Gather. This feature provides the capability of transferring multiple buffers between memory and I/O without CPU intervention. In Scatter/Gather mode, the DMA can read the memory address and word count from an array of buffer descriptors, located in system memory, called the Scatter/Gather Descriptor (SGD) Table. This allows the DMA controller to sustain DMA transfers until all of the buffers in the SGD table are read.

Timer Block

The timer block contains three counters that are equivalent in function to those found in one 82C54 programmable interval timer. These three counters are combined to provide the System Timer function, Refresh Request, and speaker tone. The three counters use the 14.31818 MHz OSC input for a clock source.

In addition to the three counters, the SIO provides a programmable 16-bit BIOS timer. This timer can be used by BIOS software to implement timing loops. The timer uses the ISA system clock (SYSCLK) divided by 8 as a clock source. An 8:1 ratio between the SYSCLK and the BIOS timer clock is always maintained. The accuracy of the BIOS timer is ±1 ms.

Utility Bus (X-Bus) Logic

The SIO provides four encoded chip selects that are decoded externally to provide chip selects for Flash BIOS, Real Time Clock, Keyboard/Mouse Controller, Floppy Controller, two Serial Ports, one Parallel Port, and an IDE Hard Disk Drive. The SIO provides the control for the buffer that isolates the lower eight bits of the Utility Bus from the lower 8 bits of the ISA bus.

In addition to providing the encoded chip selects and Utility Bus buffer control, the SIO also provides Port 92 functions (Alternate Reset and Alternate A20), Coprocessor error reporting, the Floppy DSKCHG function, and a mouse interrupt input.

Interrupt Controller Block

The SIO provides an ISA compatible interrupt controller that incorporates the functionality of two 82C59 interrupt controllers. The two interrupt controllers are cascaded so that 14 external and 2 internal interrupts are possible.

Test

The test block provides the interface to the test circuitry within the SIO. The test input can be used to tri-state all of the SIO outputs.

2.0 PIN ASSIGNMENT

The SIO package is a 208-pin Quad Flatpack (QFP). The package signals are listed in Table 1. The following notations are used to describe pin types.

Signal Type	Description
I	**Input** is a standard input-only signal.
O	**Totem Pole Output** is a standard active driver.
OD	**Open Drain** Input/Output
IO	**Input/Output** is a bidirectional, tri-state pin.
s/t/s	**Sustained Tri-State** is an active low tri-state signal owned and driven by one and only one agent at a time. The agent that drives a s/t/s pin low must drive it high for at least one clock before letting it float. A new agent can not start driving a s/t/s signal any sooner than one clock after the previous owner tri-states it. A pull-up sustains the inactive state until another agent drives it and is provided by the central resource.
t/s/o	**Tri-State Output**

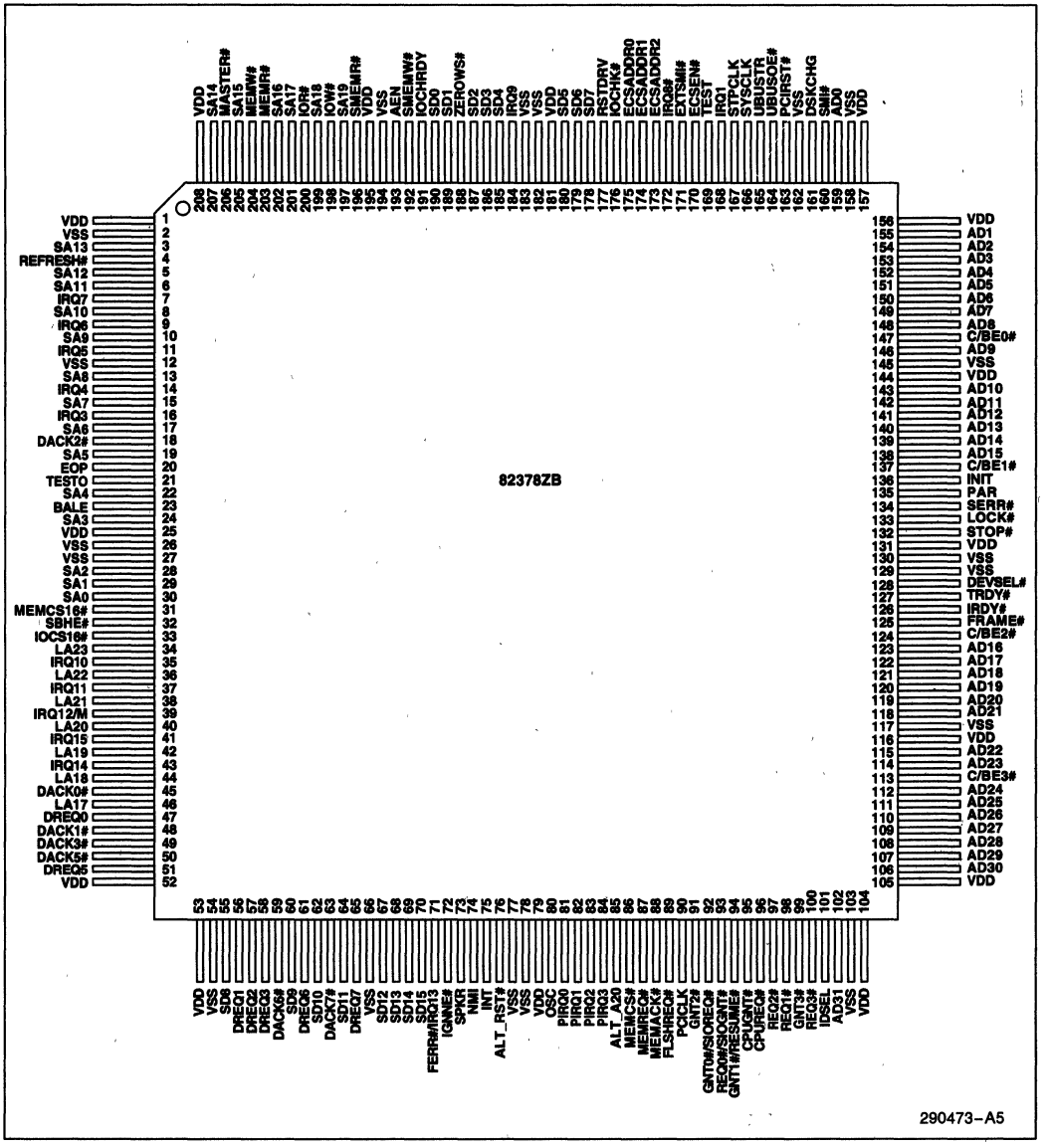


Figure 1. SIO Package Pinout Diagram

Table 1. Alphabetical Pin Assignment

Pin Name	Pin #	Type
AD0	159	I/O
AD1	155	I/O
AD2	154	I/O
AD3	153	I/O
AD4	152	I/O
AD5	151	I/O
AD6	150	I/O
AD7	149	I/O
AD8	148	I/O
AD9	146	I/O
AD10	143	I/O
AD11	142	I/O
AD12	141	I/O
AD13	140	I/O
AD14	139	I/O
AD15	138	I/O
AD16	123	I/O
AD17	122	I/O
AD18	121	I/O
AD19	120	I/O
AD20	119	I/O
AD21	118	I/O
AD22	115	I/O
AD23	114	I/O
AD24	112	I/O
AD25	111	I/O
AD26	109	I/O
AD27	109	I/O
AD28	108	I/O
AD29	107	I/O
AD30	106	I/O
AD31	102	I/O
AEN	193	O
ALT_A20	85	O
ALT_RST#	76	O

Pin Name	Pin #	Type
BALE	23	O
C/BE0#	147	I/O
C/BE1#	137	I/O
C/BE2#	124	I/O
C/BE3#	113	I/O
CPUGNT#	95	t/s/o
CPUREQ#	96	I
DACK0#	45	O
DACK1#	48	O
DACK2#	18	O
DACK3#	49	O
DACK5#	50	O
DACK6#	59	O
DACK7#	63	O
DEVSEL#	128	I/O (s/t/s)
DREQ0	47	I
DREQ1	56	I
DREQ2	57	I
DREQ3	58	I
DREQ5	51	I
DREQ6	61	I
DREQ7	65	I
DSKCHG	161	I
ECSADDR0	175	O
ECSADDR1	174	O
ECSADDR2	173	O
ECSEN#	170	O
EOP	20	I/O
EXTSMI#	171	I
FERR#/IRQ13	71	I
FLSHREQ#	89	t/s/o
FRAME#	125	I/O (s/t/s)
GNT0#/SIOREQ#	92	t/s/o
GNT1#/RESUME#	94	t/s/o
GNT2#	91	t/s/o

Table 1. Alphabetical Pin Assignment (Continued)

Pin Name	Pin #	Type
GNT3#	99	t/s/o
IDSEL	101	I
IGNNE#	72	O
INIT	136	I
INT	75	O
IOCHK#	176	I
IOCHRDY	191	I/O
IOCS16#	33	I
IOR#	200	I/O
IOW#	198	I/O
IRDY#	126	I/O (s/t/s)
IRQ1	168	I
IRQ3	16	I
IRQ4	14	I
IRQ5	11	I
IRQ6	9	I
IRQ7	7	I
IRQ8#	172	I
IRQ9	184	I
IRQ10	35	I
IRQ11	37	I
IRQ12/M	39	I
IRQ14	43	I
IRQ15	41	I
LA17	46	I/O
LA18	44	I/O
LA19	42	I/O
LA20	40	I/O
LA21	38	I/O
LA22	36	I/O
LA23	34	I/O
LOCK#	133	I (s/t/s)
MASTER#	206	I
MEMACK#	88	I
MEMCS#	86	O

Pin Name	Pin #	Type
MEMCS16#	31	I/O (o/d)
MEMR#	203	I/O
MEMREQ#	87	t/s/o
MEMW#	204	I/O
NMI	74	O
OSC	80	I
PAR	135	O
PCICLK	90	I
PCIRST#	163	I
PIRQ0#	81	I
PIRQ1#	82	I
PIRQ2#	83	I
PIRQ3#	84	I
REFRESH#	4	I/O
REQ0#/SIOGNT#	93	I
REQ1#	98	I
REQ2#	97	I
REQ3#	100	I
RSTDRV	177	O
SA0	30	I/O
SA1	29	I/O
SA2	28	I/O
SA3	24	I/O
SA4	22	I/O
SA5	19	I/O
SA6	17	I/O
SA7	15	I/O
SA8	13	I/O
SA9	10	I/O
SA10	8	I/O
SA11	6	I/O
SA12	5	I/O
SA13	3	I/O
SA14	207	I/O
SA15	205	I/O

Table 1. Alphabetical Pin Assignment (Continued)

Pin Name	Pin #	Type
SA16	202	I/O
SA17	201	I/O
SA18	199	I/O
SA19	197	I/O
SBHE#	32	I/O
SD0	190	I/O
SD1	189	I/O
SD2	187	I/O
SD3	186	I/O
SD4	185	I/O
SD5	180	I/O
SD6	179	I/O
SD7	178	I/O
SD8	55	I/O
SD9	60	I/O
SD10	62	I/O
SD11	64	I/O
SD12	67	I/O
SD13	68	I/O
SD14	69	I/O
SD15	70	I/O
SERR#	134	I
SMEMR#	196	O
SMEMW#	192	O
SMI#	160	O
SPKR	73	O
STOP#	132	I/O (s/t/s)
STPCLK#	167	O
SYSCLK	166	O
TEST	169	I
TEST0	21	O
TRDY#	127	I/O (s/t/s)
UBUSOE#	164	O
UBUSTR	165	O

Pin Name	Pin #	Type
ZEROWS#	188	I
V_{DD}	1	V
V_{DD}	79	V
V_{DD}	104	V
V_{DD}	105	V
V_{DD}	116	V
V_{DD}	131	V
V_{DD}	144	V
V_{DD}	156	V
V_{DD}	157	V
V_{DD}	181	V
V_{DD}	25	V
V_{DD}	52	V
V_{DD}	53	V
V_{DD}	195	V
V_{DD}	208	V
V_{SS}	2	V
V_{SS}	12	V
V_{SS}	26	V
V_{SS}	27	V
V_{SS}	54	V
V_{SS}	66	V
V_{SS}	77	V
V_{SS}	78	V
V_{SS}	103	V
V_{SS}	117	V
V_{SS}	129	V
V_{SS}	130	V
V_{SS}	145	V
V_{SS}	158	V
V_{SS}	162	V
V_{SS}	182	V
V_{SS}	183	V
V_{SS}	194	V

Table 2. Numerical Pin Assignment

Pin Name	Pin #	Type
V_{DD}	1	V
V_{SS}	2	V
SA13	3	I/O
REFRESH#	4	I/O
SA12	5	I/O
SA11	6	I/O
IRQ7	7	I
SA10	8	I/O
IRQ6	9	I
SA9	10	I/O
IRQ5	11	I
V_{SS}	12	V
SA8	13	I/O
IRQ4	14	I
SA7	15	I/O
IRQ3	16	I
SA6	17	I/O
DACK2#	18	O
SA5	19	I/O
EOP	20	I/O
TEST0	21	O
SA4	22	I/O
BALE	23	O
SA3	24	I/O
V_{DD}	25	V
V_{SS}	26	V
V_{SS}	27	V
SA2	28	I/O
SA1	29	I/O
SA0	30	I/O
MEMCS16#	31	I/O (o/d)
SBHE#	32	I/O
IOCS16#	33	I
LA23	34	I/O
IRQ10	35	I

Pin Name	Pin #	Type
LA22	36	I/O
IRQ11	37	I
LA21	38	I/O
IRQ12/M	39	I
LA20	40	I/O
IRQ15	41	I
LA19	42	I/O
IRQ14	43	I
LA18	44	I/O
DACK0#	45	O
LA17	46	I/O
DREQ0	47	I
DACK1#	48	O
DACK3#	49	O
DACK5#	50	O
DREQ5	51	I
V_{DD}	52	V
V_{DD}	53	V
V_{SS}	54	V
SD8	55	I/O
DREQ1	56	I
DREQ2	57	I
DREQ3	58	I
DACK6#	59	O
SD9	60	I/O
DREQ6	61	I
SD10	62	I/O
DACK7#	63	O
SD11	64	I/O
DREQ7	65	I
V_{SS}	66	V
SD12	67	I/O
SD13	68	I/O
SD14	69	I/O
SD15	70	I/O

Table 2. Numerical Pin Assignment (Continued)

Pin Name	Pin #	Type
FERR#/IRQ13	71	I
IGNNE#	72	O
SPKR	73	O
NMI	74	O
INT	75	O
ALT_RST#	76	O
V_{SS}	77	V
V_{SS}	78	V
V_{DD}	79	V
OSC	80	I
PIRQ0#	81	I
PIRQ1#	82	I
PIRQ2#	83	I
PIRQ3#	84	I
ALT_A20	85	O
MEMCS#	86	O
MEMREQ#	87	t/s/o
MEMACK#	88	I
FLSHREQ#	89	t/s/o
PCICLK	90	I
GNT2#	91	t/s/o
GNT0#/SIOREQ#	92	t/s/o
REQ0#/SIOGNT#	93	I
GNT1#/RESUME#	94	t/s/o
CPUGNT#	95	t/s/o
CPUREQ#	96	I
REQ2#	97	I
REQ1#	98	I
GNT3#	99	I
REQ3#	100	I
IDSEL	101	I
AD31	102	I/O
V_{SS}	103	V
V_{DD}	104	V
V_{DD}	105	V

Pin Name	Pin #	Type
AD30	106	I/O
AD29	107	I/O
AD28	108	I/O
AD27	109	I/O
AD26	110	I/O
AD25	111	I/O
AD24	112	I/O
C/BE3#	113	I/O
AD23	114	I/O
AD22	115	I/O
V_{DD}	116	V
V_{SS}	117	V
AD21	118	I/O
AD20	119	I/O
AD19	120	I/O
AD18	121	I/O
AD17	122	I/O
AD16	123	I/O
C/BE2#	124	I/O
FRAME#	125	I/O (s/t/s)
IRDY#	126	I/O (s/t/s)
TRDY#	127	I/O (s/t/s)
DEVSEL#	128	I/O (s/t/s)
V_{SS}	129	V
V_{SS}	130	V
V_{DD}	131	V
STOP#	132	I/O (s/t/s)
LOCK#	133	I (s/t/s)
SERR#	134	I
PAR	135	O
INIT	136	I
C/BE1#	137	I/O
AD15	138	I/O
AD14	139	I/O
AD13	140	I/O

Table 2. Numerical Pin Assignment (Continued)

Pin Name	Pin #	Type
AD12	141	I/O
AD11	142	I/O
AD10	143	I/O
V_{DD}	144	V
V_{SS}	145	V
AD9	146	I/O
C/BE0#	147	I/O
AD8	148	I/O
AD7	149	I/O
AD6	150	I/O
AD5	151	I/O
AD4	152	I/O
AD3	153	I/O
AD2	154	I/O
AD1	155	I/O
V_{DD}	156	V
V_{DD}	157	V
V_{SS}	158	V
AD0	159	I/O
SMI#	160	O
DSKCHG	161	I
V_{SS}	162	V
PCIRST#	163	I
UBUSOE#	164	O
UBUSTR	165	O
SYSCLK	166	O
STPCLK#	167	O
IRQ1	168	I
TEST	169	I
ECSEN#	170	O
EXTSMI#	171	I
IRQ8#	172	I
ECSADDR2	173	O
ECSADDR1	174	O

Pin Name	Pin #	Type
ECSADDR0	175	O
IOCHK#	176	I
RSTDRV	177	O
SD7	178	I/O
SD6	179	I/O
SD5	180	I/O
V_{DD}	181	V
V_{SS}	182	V
V_{SS}	183	V
IRQ9	184	I
SD4	185	I/O
SD3	186	I/O
SD2	187	I/O
ZEROWS#	188	I
SD1	189	I/O
SD0	190	I/O
IOCHRDY	191	I/O
SMEMW#	192	O
AEN	193	O
V_{SS}	194	V
V_{DD}	195	V
SMEMR#	196	O
SA19	197	I/O
IOW#	198	I/O
SA18	199	I/O
IOR#	200	I/O
SA17	201	I/O
SA16	202	I/O
MEMR#	203	I/O
MEMW#	204	I/O
SA15	205	I/O
MASTER#	206	I
SA14	207	I/O
V_{DD}	208	V

3.0 SIGNAL DESCRIPTION

This section contains a detailed description of each signal. The signals are arranged in functional groups according to the interface.

Note that the "#" symbol at the end of a signal name indicates that the active, or asserted state occurs when the signal is at a low voltage level. When "#" is not present after the signal name, the signal is asserted when at the high voltage level.

The terms assertion and negation are used extensively. This is done to avoid confusion when working with a mixture of "active-low" and "active-high" signals. The term **assert**, or **assertion** indicates that a signal is active, independent of whether that level is represented by a high or low voltage. The term **negate**, or **negation** indicates that a signal is inactive.

3.1 PCI Bus Interface Signals

Signal Name	Type	Description
PCICLK	I	**PCI CLOCK:** PCICLK provides timing for all transactions on the PCI Bus. All other PCI signals are sampled on the rising edge of PCICLK, and all timing parameters are defined with respect to this edge. Frequencies supported by the SIO include 25 MHz and 33 MHz.
PCIRST#	I	**PCI RESET:** PCIRST# forces the SIO to a known state. AD[31:0], C/BE[3:0]#, and PAR are always driven low by the SIO synchronously from the leading edge of PCIRST#. The SIO always tri-states these signals from the trailing edge of PCIRST#. If the internal arbiter is enabled (CPUREQ# sampled high on the trailing edge of PCIRST#), the SIO will drive these signals low again (synchronously 2–5 PCICLKs later) until the bus is given to another master. If the internal arbiter is disabled (CPUREQ# sampled low on the trailing edge of PCIRST#), these signals remain tri-stated until the SIO is required to drive them valid as a master or slave. FRAME#, IRDY#, TRDY#, STOP#, DEVSEL#, MEMREQ#, FLSHREQ#, CPUGNT#, GNT0#/SIOREQ#, and GNT1#/RESUME# are tri-stated from the leading edge of PCIRST#. FRAME#, IRDY#, TRDY#, STOP#, and DEVSEL# remain tri-stated until driven by the SIO as either a master or a slave. MEMREQ#, FLSHREQ#, CPUGNT#, GNT0#/SIOREQ#, and GNT1#/RESUME# are tri-stated until driven by the SIO. After PCIRST#, MEMREQ# and FLSHREQ# are driven inactive asynchronously from PCIRST# inactive. CPUGNT#, GNT0#/SIOREQ#, and GNT1#/RESUME# are driven based on the arbitration scheme and the asserted REQx#'s. All registers are set to their default values. PCIRST# may be asynchronous to PCICLK when asserted or negated. Although asynchronous, negation must be a clean, bounce-free edge. Note that PCIRST# must be asserted for more than 1 μs.

3.1 PCI Bus Interface Signals (Continued)

Signal Name	Type	Description
AD[31:0]	I/O	**PCI ADDRESS/DATA.** AD[31:0] is a multiplexed address and data bus. During the first clock of a transaction, AD[31:0] contain a physical byte address (32 bits). During subsequent clocks, AD[31:0] contain data. A SIO Bus transaction consists of an address phase followed by one or more data phases. Little-endian byte ordering is used. AD[7:0] define the least significant byte (LSB) and AD[31:24] the most significant byte (MSB). When the SIO is a target, AD[31:0] are inputs during the address phase of a transaction. During the following data phase(s), the SIO may be asked to supply data on AD[31:0] for a PCI read, or accept data for a PCI write. As a master, the SIO drives a valid address on AD[31:2] during the address phase, and drives write or latches read data on AD[31:0] during the data phase. The SIO always drives AD[1:0] low as a master. AD[31:0] are always driven low by the SIO synchronously from the leading edge of PCIRST#. The SIO always tri-states AD[31:0] from the trailing edge of PCIRST#. If the internal arbiter is enabled (CPUREQ# sampled high on the trailing edge of PCIRST#), the SIO drives AD[31:0] low again (synchronously 2–5 PCICLKs later) until the bus is given to another master. If the internal arbiter is disabled (CPUREQ# sampled low on the trailing edge of PCIRST#), AD[31:0] remain tri-stated until the SIO is required to drive them valid as a master or slave. When the internal arbiter is enabled, the SIO acts as the central resource responsible for driving the AD[31:0] signals when no one is granted the PCI Bus and the bus is idle. When the internal arbiter is disabled, the SIO does not drive AD[31:0] as the central resource. The SIO is always responsible for driving AD[31:0] when it is granted the bus (SIOGNT# and idle bus) and as appropriate when it is the master of a transaction.
C/BE[3:0]#	I/O	**BUS COMMAND AND BYTE ENABLES:** The command and byte enable signals are multiplexed on the same PCI pins. During the address phase of a transaction, C/BE[3:0]# define the bus command. During the data phase C/BE[3:0]# are used as Byte Enables. The Byte Enables determine which byte lanes carry meaningful data. C/BE#[0] applies to byte 0, C/BE[1] to byte 1, C/BE[2] to byte 2, and C/BE#[3] to byte 3. The SIO drives C/BE[3:0]# as an initiator of a PCI Bus cycle and monitors C/BE[3:0]# as a Target. C/BE[3:0]# are always driven low by the SIO synchronously from the leading edge of PCIRST#. The SIO always tri-states C/BE[3:0]# from the trailing edge of PCIRST#. If the internal arbiter is enabled (CPUREQ# sampled high on the trailing edge of PCIRST#), the SIO drives C/BE[3:0]# low again (synchronously 2-5 PCICLKs later) until the bus is given to another master. If the internal arbiter is disabled (CPUREQ# sampled low on the trailing edge of PCIRST#), C/BE[3:0]# remain tri-stated until the SIO is required to drive them valid as a master or slave. When the internal arbiter is enabled, the SIO acts as the central resource responsible for driving the C/BE[3:0]# signals when no one is granted the PCI Bus and the bus is idle. When the internal arbiter is disabled, the SIO does not drive C/BE[3:0]# as the central resource. The SIO is always responsible for driving C/BE[3:0]# when it is granted the bus (SIOGNT# and idle bus) and as appropriate when it is the master of a transaction.

3.1 PCI Bus Interface Signals (Continued)

Signal Name	Type	Description
FRAME#	I/O (s/t/s)	**CYCLE FRAME:** FRAME# is driven by the current master to indicate the beginning and duration of an access. FRAME# is asserted to indicate a bus transaction is beginning. While FRAME# is asserted data transfers continue. When FRAME# is negated the transaction is in the final data phase. FRAME# is an input to the SIO when the SIO is the target. FRAME# is an output when the SIO is the initiator. FRAME# is tri-stated from the leading edge of PCIRST#. FRAME# remains tri-stated until driven by the SIO as either a master or a slave.
TRDY#	I/O (s/t/s)	**TARGET READY:** TRDY# indicates the SIO's ability to complete the current data phase of the transaction. TRDY# is used in conjunction with IRDY#. A data phase is completed when both TRDY# and IRDY# are sampled asserted. During a read, TRDY# indicates that the SIO, as a target, has placed valid data on AD[31:0]. During a write, it indicates the SIO, as a target is prepared to latch data. TRDY is an input to the SIO when the SIO is the initiator and an output when the SIO is a target. TRDY# is tri-stated from the leading edge of PCIRST#. TRDY# remains tri-stated until driven by the SIO as either a master or a slave.
IRDY#	I/O (s/t/s)	**INITIATOR READY:** IRDY# indicates the SIO's ability, as an initiator, to complete the current data phase of the transaction. It is used in conjunction with TRDY#. A data phase is completed on any clock that both IRDY# and TRDY# are sampled asserted. During a write, IRDY# indicates the SIO has valid data present on AD[31:0]. During a read, it indicates the SIO is prepared to latch data. IRDY is an input to the SIO when the SIO is the target and an output when the SIO is an initiator. IRDY# is tri-stated from the leading edge of PCIRST#. IRDY# remains tri-stated until driven by the SIO as either a master or a slave.
STOP#	I/O (s/t/s)	**STOP:** STOP# indicates that the SIO, as a target, is requesting a master to stop the current transaction. As a master, STOP# causes the SIO to stop the current transaction. STOP# is an output when the SIO is a target and an input when the SIO is an initiator. STOP# is tri-stated from the leading edge of PCIRST#. STOP# remains tri-stated until driven by the SIO as either a master or a slave.
LOCK#	I	**LOCK:** LOCK# indicates an atomic operation that may require multiple transactions to complete. LOCK# is always an input to the SIO. When the SIO is the target of a transaction and samples LOCK# negated during the address phase of a transaction, the SIO considers itself a locked resource until it samples LOCK# and FRAME# negated. When other masters attempt accesses while the SIO is locked, the SIO responds with a retry termination. LOCK# is tri-stated during reset.
IDSEL	I	**INITIALIZATION DEVICE SELECT:** IDSEL is used as a chip select during configuration read and write transactions. The SIO samples IDSEL during the address phase of a transaction. If IDSEL is sampled active, and the bus command is a configuration read or write, the SIO responds by asserting DEVSEL# on the next cycle.
DEVSEL#	I/O (s/t/s)	**DEVICE SELECT:** The SIO asserts DEVSEL# to claim a PCI transaction through positive or subtractive decoding. As an output, the SIO asserts DEVSEL# when it samples IDSEL active in configuration cycles to SIO configuration registers. The SIO also asserts DEVSEL# when an internal SIO address is decoded or when the SIO subtractively decodes a cycle. As an input, DEVSEL# indicates the response to a SIO master-initiated transaction. The SIO also samples this signal for all PCI transactions to decide to subtractively decode the cycle. DEVSEL# is tri-stated from the leading edge of PCIRST#. DEVSEL# remains tri-stated until driven by the SIO as either a master or a slave.

3.1 PCI Bus Interface Signals (Continued)

Signal Name	Type	Description
PIRQ[3:0]#	I	**PCI INTERRUPT REQUEST:** PIRQ#s are used to generate asynchronous interrupts to the CPU via the Programmable Interrupt Controllers (82C59s) integrated in the SIO. These signals are defined as level sensitive and are asserted low. The PIRQx# interrupts can be steered into any unused IRQ interrupt. The PIRQx# Route Control Register determines which IRQ interrupt each PCI interrupt is steered into. These pins include a weak internal pull-up resistor.
PAR	O	**CALCULATED PARITY SIGNAL:** PAR is "even" parity and is calculated on 36 bits—AD[31:0] plus C/BE[3:0]#. "Even" parity means that the number of "1"s within the 36 bits plus PAR are counted and the sum is always even. PAR is always calculated on 36 bits regardless of the valid byte enables. PAR is generated for address and data phases and is only guaranteed to be valid one PCI clock after the corresponding address or data phase. PAR is driven and tri-stated identically to the AD[31:0] lines except that PAR is delayed by exactly one PCI clock. PAR is an output during the address phase (delayed one clock) for all SIO master transactions. It is also an output during the data phase (delayed one clock) when the SIO is the master of a PCI write transaction, and when it is the target of a read transaction. PAR is always driven low by the SIO synchronously from the leading edge of PCIRST#. The SIO always tri-states PAR from the trailing edge of PCIRST#. If the internal arbiter is enabled (CPUREQ# sampled high on the trailing edge of PCIRST#), the SIO drives PAR low again (synchronously 2-5 PCICLKs later) until the bus is given to another master. If the internal arbiter is disabled (CPUREQ# sampled low on the trailing edge of PCIRST#), PAR remains tri-stated until the SIO is required to drive them valid as a master or slave. When the internal arbiter is enabled, the SIO acts as the central resource responsible for driving PAR when no device is granted the PCI Bus and the bus is idle. When the internal arbiter is disabled, the SIO does not drive PAR as the central resource. The SIO is always responsible for driving PAR when it is granted the bus (SIOGNT# and idle bus) and as appropriate when it is the master of a transaction.
SERR#	I	**SYSTEM ERROR:** SERR# can be pulsed active by any PCI device that detects a system error condition. Upon sampling SERR# active, the SIO generates a non-maskable interrupt (NMI) to the CPU.

3.2 PCI Arbiter Signals

Signal Name	Type	Description
CPUREQ#	I	**CPU REQUEST:** This signal provides the following functions: 1. If CPUREQ# is sampled high on the trailing edge of PCIRST#, the internal arbiter is enabled. If CPUREQ# is sampled low on the trailing edge of PCIRST#, the internal arbiter is disabled. This requires that the host bridge drive CPUREQ# high during PCIRST#. 2. If the SIO's internal arbiter is enabled, this pin is configured as CPUREQ#. An active low assertion indicates that the CPU initiator desires the use of the PCI Bus. If the internal arbiter is disabled, this pin is meaningless after reset. This pin has a weak internal pull-up resistor.
REQ0#/ SIOGNT#	I	**REQUEST 0/SIO GRANT:** If the SIO's internal arbiter is enabled, this pin is configured as REQ0#. An active low assertion indicates that Initiator0 desires the use of the PCI Bus. If the internal arbiter is disabled, this pin is configured as SIOGNT#. When asserted, SIOGNT# indicates that the external PCI arbiter has granted use of the bus to the SIO. This pin has a weak internal pull-up resistor.
REQ1#	I	**REQUEST 1:** If the SIO's internal arbiter is enabled through the Arbiter Configuration Register, then this signal is configured as REQ1#. An active low assertion indicates that Initiator1 desires the use of the PCI Bus. If the internal arbiter is disabled, the SIO ignores REQ1# after reset. This pin has a weak internal pull-up resistor.
CPUGNT#	t/s/o	**CPU GRANT:** If the SIO's internal arbiter is enabled, this pin is configured as CPUGNT#. The SIO's internal arbiter asserts CPUGNT# to indicate that the CPU initiator has been granted the PCI Bus. If the internal arbiter is disabled, this signal is meaningless. CPUGNT# is tri-stated from the leading edge of PCIRST#. CPUGNT# is tri-stated until driven by the SIO. CPUGNT# is driven based on the arbitration scheme and the asserted REQx#'s.
GNT0#/ SIOREQ#	t/s/o	GRANT 0/SIO REQUEST: If the SIO's internal arbiter is enabled, this pin is configured as GNT0#. The SIO's internal arbiter asserts GNT0# to indicate that Initiator0 has been granted the PCI Bus. If the internal arbiter is disabled, this pin is configured as SIOREQ#. The SIO asserts SIOREQ# to request the PCI Bus. GNT0#/SIOREQ# is tri-stated from the leading edge of PCIRST#. GNT0#/SIOREQ# is tri-stated until driven by the SIO. GNT0#/SIOREQ# is driven based on the arbitration scheme and the asserted REQx#'s.
GNT1#/ RESUME#	t/s/o	**GRANT 1/RESUME:** If the SIO's internal arbiter is enabled, this pin is configured as GNT1#. The SIO's internal arbiter asserts GNT1# to indicate that Initiator1 has been granted the PCI Bus. If the internal arbiter is disabled, this pin is configured as RESUME#. The SIO asserts RESUME# to indicate that the conditions causing the SIO to retry the cycle has passed. GNT1#/RESUME# is tri-stated from the leading edge of PCIRST#. GNT1#/RESUME# is tri-stated until driven by the SIO. GNT1#/RESUME# is driven based on the arbitration scheme and the asserted REQx#'s.
REQ2#	I	**REQUEST 2:** This pin is an active low signal that indicates that Initiator2 desires the use of the PCI Bus. This signal has a weak internal pull-up resistor.

3.2 PCI Arbiter Signals (Continued)

Signal Name	Type	Description
REQ3#	I	**REQUEST 3:** This pin is an active low signal that indicates that Initiator3 desires the use of the PCI Bus. This signal has a weak internal pull-up resistor.
GNT2#	t/s/o	**GRANT 2:** This pin is configured as GNT2#. The SIO's internal arbiter asserts GNT2# to indicate that Initiator2 has been granted the PCI Bus. GNT2# is high upon reset.
GNT3#	t/s/o	**GRANT 3:** This pin is configured as GNT3#. The SIO's internal arbiter asserts GNT3# to indicate that Initiator3 has been granted the PCI Bus. GNT3# is high upon reset.
MEMREQ#	t/s/o	**MEMORY REQUEST:** If the SIO is configured in Guaranteed Access Time (GAT) Mode, MEMREQ# will be asserted when an ISA master or DMA is requesting the ISA Bus (along with FLSHREQ#) to indicate that the SIO requires ownership of the main memory. MEMREQ# is tri-stated from the leading edge of PCIRST#. MEMREQ# remains tri-stated until driven by the SIO. After PCIRST, MEMREQ# is driven inactive asynchronously from PCIRST# inactive. The SIO asserts FLSHREQ# concurrently with asserting MEMREQ#. **FLSHREQ# / MEMREQ# / Meaning** 1 / 1 / Idle 0 / 1 / Flush buffers pointing towards PCI to avoid ISA deadlock 1 / 0 / Reserved 0 / 0 / GAT mode. Guarantee PCI Bus immediate access to main memory (this may or may not require the PCI-to-main memory buffers to be flushed first depending on the number of buffers).
FLSHREQ#	t/s/o	**FLUSH REQUEST:** FLSHREQ# is generated by the SIO to command all of the system's posted write buffers pointing towards the PCI Bus to be flushed. This is required before granting the ISA Bus to an ISA master or the DMA. FLSHREQ# is tri-stated from the leading edge of PCIRST#. FLSHREQ# remains tri-stated until driven by the SIO. After PCIRST, FLSHREQ# is driven inactive asynchronously from PCIRST# inactive.
MEMACK#	I	**MEMORY ACKNOWLEDGE:** MEMACK# is the response handshake that indicates to the SIO that the function requested over the MEMREQ# and/or FLSHREQ# signals has been completed. In GAT mode (MEMREQ# and FLSHREQ# asserted), the main memory bus is dedicated to the PCI Bus and the system's posted write buffers pointing towards the PCI Bus have been flushed and are disabled. In non-GAT mode (FLSHREQ# asserted alone), this means the system's posted write buffers have been flushed and are disabled. In either case, the SIO can now grant the ISA Bus to the requester.

3.3 Address Decoder Signal

Signal Name	Type	Description
MEMCS#	O	**MEMORY CHIP SELECT:** MEMCS# is a programmable address decode signal provided to a Host CPU bridge. A CPU bridge can use MEMCS# to forward a PCI cycle to main memory behind the bridge. MEMCS# is driven one PCI clock after FRAME# is sampled active (address phase) and is valid for one clock cycle before going inactive. MEMCS# is high upon reset.

3.4 Power Management Signals

Signal Name	Type	Description
SMI#	O	**SYSTEM MANAGEMENT INTERRUPT:** SMI# is an active low output that is asserted by the SIO in response to one of many enableable hardware or software events. SMI# connects directly to the CPU. The SMI# signal is an asychronous input to the CPU. The CPU recognizes the falling edge of SMI# as the highest priority interrupt in the system. The CPU responds by entering SMM (System Management Mode). SMI# is deasserted during and following reset.
STPCLK#	O	**STOP CLOCK:** STPCLK# is an active low output that is asserted by the SIO in response to one of many enableable hardware or software events. STPCLK# connects directly to the CPU. The STPCLK# signal is an asynchronous input to the CPU. When the CPU samples STPCLK# asserted it responds by stopping its internal clock. STPCLK# is deasserted during and following reset.
EXTSMI#	I	**EXTERNAL SYSTEM MANAGEMENT INTERRUPT:** EXTSMI# is a falling edge triggered input to the SIO indicating that an external device is requesting the system to enter SMM mode. When enabled, a falling edge on EXTSMI# will result in the assertion of the SMI# signal to the CPU. EXTSMI# is an asychronous input to the SIO. However, when the setup and hold times are met, it is only required to be asserted for one PCICLK. Once deasserted, it must remain deasserted for at least four PCICLKs in order to allow the edge detect logic to reset. This pin includes a weak internal pull-up resistor.
INIT	I	**INIT:** INIT is an input to the SIO indicating that the CPU is actually being soft reset. It is connected to the INIT pin of the CPU. This pin includes a weak internal pull-up resistor.

3.5 ISA Interface Signals

Signal Name	Type	Description
AEN	O	**ADDRESS ENABLE:** AEN is asserted during DMA cycles to prevent I/O slaves from misinterpreting DMA cycles as valid I/O cycles. When negated, AEN indicates that an I/O slave may respond to address and I/O commands. When asserted, AEN informs I/O resources on the ISA Bus that a DMA transfer is occurring. This signal is also driven high during refresh cycles. AEN is driven low upon reset.

3.5 ISA Interface Signals (Continued)

Signal Name	Type	Description
BALE	O	**BUS ADDRESS LATCH ENABLE:** BALE is an active high signal asserted by the SIO to indicate that the address (SA[19:0], LA[23:17]), AEN and SBHE# signal lines are valid. The LA[23:17] address lines are latched on the trailing edge of BALE. BALE remains asserted throughout DMA and ISA master cycles. BALE is driven low upon reset.
SYSCLK	O	**SYSTEM CLOCK:** SYSCLK is an output of the SIO component. The frequencies supported are 6 MHz to 8.33 MHz.
IOCHRDY	I/O	**I/O CHANNEL READY:** Resources on the ISA Bus assert IOCHRDY to indicate that additional time (wait-states) is required to complete the cycle. This signal is normally high on the ISA Bus. IOCHRDY is an input when the SIO owns the ISA Bus and a PCI agent is accessing an ISA slave or during compatible DMA transfers (compatible cycles only). IOCHRDY is output when an external ISA Bus Master owns the ISA Bus and is accessing a PCI slave or an SIO register. As an SIO output, IOCHRDY is driven inactive (low) from the falling edge of the ISA commands. After data is available for an ISA master read or the SIO latches the data for a write cycle, IOCHRDY is asserted for 70 ns. After 70 ns, the SIO floats IOCHRDY. The 70 ns includes both the drive time and the time it takes the SIO to float IOCHRDY. The SIO does not drive this signal when an ISA Bus master is accessing an ISA Bus slave. IOCHRDY is tri-stated upon reset.
IOCS16#	I	**16-BIT I/O CHIP SELECT:** This signal is driven by I/O devices on the ISA Bus to indicate that they support 16-bit I/O bus cycles.
IOCHK#	I	**I/O CHANNEL CHECK:** IOCHK# can be driven by any resource on the ISA Bus. When asserted, it indicates that a parity or an un-correctable error has occurred for a device or memory on the ISA Bus. A NMI will be generated to the CPU if the NMI generation is enabled.
IOR#	I/O	**I/O READ:** IOR# is the command to an ISA I/O slave device that the slave may drive data on to the ISA data bus (SD[15:0]). The I/O slave device must hold the data valid until after IOR# is negated. IOR# is an output when the SIO owns the ISA Bus. IOR# is an input when an external ISA master owns the ISA Bus. IOR# is driven high upon reset.
IOW#	I/O	**I/O WRITE:** IOW# is the command to an ISA I/O slave device that the slave may latch data from the ISA data bus (SD[15:0]). IOW# is an output when the SIO owns the ISA Bus. IOW# is an input when an external ISA master owns the ISA Bus. IOW# is driven high upon reset.
LA[23:17]	I/O	**UNLATCHED ADDRESS:** The LA[23:17] address lines are bi-directional. These address lines allow accesses to physical memory on the ISA Bus up to 16 MBytes. LA[23:17] are outputs when the SIO owns the ISA Bus. The LA[23:17] lines become inputs whenever an ISA master owns the ISA Bus. These signals are undefined during DMA type "A", "B", and "F" cycles. The LA[23:17] signals are at an unknown state upon reset.
SA[19:0]	I/O	**SYSTEM ADDRESS BUS:** These bi-directional address lines define the selection with the granularity of one byte within the one Megabyte section of memory defined by the LA[23:17] address lines. The address lines SA[19:17] that are coincident with LA[19:17] are defined to have the same values as LA[19:17] for all memory cycles. For I/O accesses, only SA[15:0] are used. SA[19:0] are outputs when the SIO owns the ISA Bus. SA[19:0] are inputs when an external ISA Master owns the ISA Bus. SA[19:0] are undefined during DMA type "A", 'B", or "F" cycles. SA[19:0] are at an unknown state upon reset.

3.5 ISA Interface Signals (Continued)

Signal Name	Type	Description
SBHE#	I/O	**SYSTEM BYTE HIGH ENABLE:** SBHE# indicates, when asserted, that a byte is being transferred on the upper byte (SD[15:8]) of the data bus. SBHE# is negated during refresh cycles. SBHE# is an output when the SIO owns the ISA Bus. SBHE# is an input when an external ISA master owns the ISA Bus. SBHE# is at an unknown state upon reset.
MEMCS16#	OD	**MEMORY CHIP SELECT 16:** MEMCS16# is a decode of LA[23:17] without any qualification of the command signal lines. ISA slaves that are 16-bit memory devices drive this signal low. The SIO ignores MEMCS16# during I/O access cycles and refresh cycles. During DMA cycles, this signal is only used by the byte swap logic. MEMCS16# is an input when the SIO owns the ISA Bus. MEMCS16# is an output when an ISA Bus master owns the ISA Bus. The SIO drives this signal low during ISA master to PCI memory cycles. MEMCS16# is at an unknown state upon reset.
MASTER#	I	**MASTER:** An ISA Bus master asserts MASTER# to indicate that it has control of the ISA Bus. Before the ISA master can assert MASTER#, it must first sample DACK# active. Once MASTER# is asserted, the ISA master has control of the ISA Bus until it negates MASTER#.
MEMR#	I/O	**MEMORY READ:** MEMR# is the command to a memory slave that it may drive data onto the ISA data bus. MEMR# is an output when the SIO is a master on the ISA Bus. MEMR# is an input when an ISA master, other than the SIO, owns the ISA Bus. This signal is also driven by the SIO during refresh cycles. For compatible timing mode DMA cycles, the SIO, as a master, asserts MEMR# if the address is less than 16 MBytes. This signal is not generated for accesses to addresses greater than 16 MByte. MEMR# is not driven active during DMA type "A", "B", or "F" cycles.
MEMW#	I/O	**MEMORY WRITE:** MEMW# is the command to a memory slave that it may latch data from the ISA data bus. MEMW# is an output when the SIO owns the ISA Bus. MEMW# is an input when an ISA master, other than the SIO, owns the ISA Bus. For compatible timing mode DMA cycles, the SIO, as a master, asserts MEMW# if the address is less than 16 MBytes. This signal is not generated for accesses to addresses greater than 16 MByte. MEMW# is not driven active during DMA type "A", "B", or "F" cycles.
SMEMW#	O	**SYSTEM MEMORY WRITE:** The SIO asserts SMEMW# to request a memory slave to accept data from the data lines. If the access is below the 1 MByte range (00000000h–000FFFFFh) during DMA compatible, SIO master, or ISA master cycles, the SIO asserts SMEMW#. SMEMW# is a delayed version of MEMW#. SMEMW# is driven high upon reset.
SMEMR#	O	**SYSTEM MEMORY READ:** The SIO asserts SMEMR# to request a memory slave to accept data from the data lines. If the access is below the 1 MByte range (00000000h–000FFFFFh) during DMA compatible, SIO master, or ISA master cycles, the SIO asserts SMEMR#. SMEMR# is a delay version of MEMR#. Upon PCIRST# this signal is low. SMEMR# is driven high upon reset.

3.5 ISA Interface Signals (Continued)

Signal Name	Type	Description
ZEROWS#	I	**ZERO WAIT-STATES:** An ISA slave asserts ZEROWS# after its address and command signals have been decoded to indicate that the current cycle can be shortened. A 16-bit ISA memory cycle can be reduced to two SYSCLKs. An 8-bit memory or I/O cycle can be reduced to three SYSCLKs. ZEROWS# has no effect during 16-bit I/O cycles. If IOCHRDY and ZEROWS# are both asserted during the same clock, then ZEROWS# is ignored and wait states are added as a function of IOCHRDY (i.e., IOCHRDY has precedence over ZEROWS#).
OSC	I	**OSCILLATOR:** OSC is the 14.31818 MHz ISA clock signal. It is used by the internal 8254 Timer, counters 0, 1, and 2.
RSTDRV	O	**RESET DRIVE:** The SIO asserts RSTDRV to reset devices that reside on the ISA Bus. The SIO asserts this signal when PCIRST# (PCI Reset) is asserted. In addition, the SIO can be programmed to assert RSTDRV by writing to the ISA Clock Divisor Register. Software should assert the RSTDRV during configuration to reset the ISA Bus when changing the clock divisor. Note that when RSTDRV is generated via the ISA Clock Divisor Register, software must ensure that RSTDRV is driven active for a minimum of 1 μs.
SD[15:0]	I/O	**System DATA:** SD[15:0] provide the 16-bit data path for devices residing on the ISA Bus. SD[15:8] correspond to the high order byte and SD[7:0] correspond to the low order byte. SD[15:0] are undefined during refresh. The SIO tri-states SD[15:0] during reset.

3.6 DMA Signals

Signal Name	Type	Description
DREQ [3:0,7:5]	I	**DMA REQUEST:** The DREQ lines are used to request DMA service from the SIO's DMA controller or for a 16-bit master to gain control of the ISA expansion bus. The active level (high or low) is programmed via the DMA Command Register (bit 6). When the bit 6 = 0, DREQ[3:0,7:5] are active high and when bit 6 = 1, the signals are active low. All inactive to active edges of DREQ are assumed to be asynchronous. The request must remain active until the appropriate DACK signal is asserted.
DACK# [3:0,7:5]	O	**DMA ACKNOWLEDGE:** The DACK output lines indicate that a request for DMA service has been granted by the SIO or that a 16-bit master has been granted the bus. The active level (high or low) is programmed via the DMA Command Register (bit 7). When bit 7 = 0, DACK# [3:0,7:5] are active low and when bit 7 = 1, the signals are active high. These lines should be used to decode the DMA slave device with the IOR# or IOW# line to indicate selection. If used to signal acceptance of a bus master request, this signal indicates when it is legal to assert MASTER#. If the DMA controller has been programmed for a timing mode other than compatible mode, and another device has requested the bus, and a 4 μs time has elapsed, this line will be negated and the transfer stopped before the transfer is complete. In this case, the transfer is re-started at the next arbitration period that the channel wins the bus. Upon PCIRST#, these lines are set inactive (high).

3.6 DMA Signals (Continued)

Signal Name	Type	Description
EOP	I/O	**END OF PROCESS:** EOP is bi-directional, acting in one of two modes, and is directly connected to the TC line of the ISA Bus. DMA slaves assert EOP to the SIO to terminate DMA cycles. The SIO asserts EOP to DMA slaves as a terminal count indicator. **EOP-IN MODE:** For all transfer types during DMA, the SIO samples EOP. If it is sampled asserted, the transfer is terminated. **TC-OUT MODE:** The SIO asserts EOP after a new address has been output, if the byte count expires with that transfer. The EOP (TC) remains asserted until AEN is negated, unless AEN is negated during an autoinitialization. EOP (TC) is negated before AEN is negated during an autoinitialization. When all the DMA channels are not in use, the EOP signal is in output mode and negated (low). After PCIRST#, EOP is in output mode and inactive.
REFRESH#	I/O	**REFRESH:** As an output, REFRESH# is used by the SIO to indicate when a refresh cycle is in progress. It should be used to enable the SA[15:0] address to the row address inputs of all banks of dynamic memory on the ISA Bus. Thus, when MEMR# is asserted, the entire expansion bus dynamic memory is refreshed. Memory slaves must not drive any data onto the bus during refresh. As an output, this signal is driven directly onto the ISA Bus. This signal is an output only when the SIO DMA refresh is a master on the bus responding to an internally generated request for refresh. As an input, REFRESH# is driven by 16-bit ISA Bus masters to initiate refresh cycles. Upon PCIRST#, this signal is tri-stated.

3.7 Timer Signal

Signal Name	Type	Description
SPKR	O	**SPEAKER DRIVE:** The SPKR signal is the output of counter 2 and is "ANDed" with Port 061h bit 1 to provide Speaker Data Enable. This signal drives an external speaker driver device, which in turn drives the ISA system speaker. SPKR has a 24 mA drive capability. Upon reset, its output state is 0.

3.8 Interrupt Controller Signals

Signal Name	Type	Description
IRQ[15,14, 11:9, 7:3,1]	I	**INTERRUPT REQUEST:** The IRQ signals provide both system board components and ISA Bus I/O devices with a mechanism for asynchronously interrupting the CPU. The assertion mode of these inputs depends on the programming of LTIM, bit 3 of ICW1 on both Controller-1 and Controller-2. When LTIM is programmed to a 0, a low-to-high transition on any of that controller's IRQ lines is recognized as an interrupt request. This is "edge-triggered" mode. Edge-triggered mode is the SIO default. When LTIM is programmed to a 1, a high level on any of that controller's IRQ lines is recognized as an interrupt request. This mode is "level-triggered" mode. Upon PCIRST#, the IRQ lines are placed in edge-triggered mode. An active IRQ input must remain asserted until after the interrupt is acknowledged. If the input goes inactive before this time, a DEFAULT IRQ7 occurs when the CPU acknowledges the interrupt. **NOTE:** Refer to the Utility Bus Signal descriptions for IRQ12 and IRQ13 signal descriptions.
IRQ8#	I	**INTERRUPT REQUEST EIGHT SIGNAL:** IRQ8# is an active low interrupt input. The assertion mode of this input depends on the programming of the LTIM bit of ICW1 on both Controller-1 and Controller-2. When the LTIM = 0, a high-to-low transition on IRQ8# is recognized as an interrupt request. This is "edge-triggered" mode. Edge triggered mode is the SIO default. When the LTIM = 1, a low level on IRQ8# is recognized as an interrupt request. This mode is "level-triggered" mode. Upon PCIRST#, IRQ8# will be placed in edge-triggered mode. IRQ8# must remain asserted until after the interrupt is acknowledged. If the input goes inactive before this time, a DEFAULT IRQ7 will occur when the CPU acknowledges the interrupt.
INT	O	**CPU INTERRUPT:** INT is driven by the SIO to signal the CPU that an interrupt request is pending and needs to be serviced. It is asynchronous with respect to SYSCLK or PCICLK and is always an output. The interrupt controller must be programmed following a reset to ensure that INT is at a known state. Upon PCIRST#, INT is driven low.
NMI	O	**NON-MASKABLE INTERRUPT:** NMI is used to force a non-maskable interrupt to the CPU. The SIO generates an NMI when either SERR# or IOCHK# is asserted, depending on how the NMI Status and Control Register is programmed. The CPU detects an NMI when it detects a rising edge on NMI. After the NMI interrupt routine processes the interrupt, the NMI status bits in the NMI Status and Control Register are cleared by software. The NMI interrupt routine must read this register to determine the source of the interrupt. The NMI is reset by setting the corresponding NMI source enable/disable bit in the NMI Status and Control Register. To enable NMI interrupts, the two NMI enable/disable bits in the register must be set to 0, and the NMI mask bit in the NMI Enable/Disable and Real-Time Clock Address Register must be set to 0. Upon PCIRST#, this signal is driven low.

3.9 Utility Bus Signals

Signal Name	Type	Description
UBUSTR	O	**UTILITY DATA BUS TRANSMIT/RECEIVE:** UBUSTR is tied directly to the direction control of a 74F245 that buffers the utility data bus, UD[7:0]. UBUSTR is asserted for all I/O read cycles (regardless if a Utility Bus device has been decoded). UBUSTR is asserted for memory cycles only if BIOS space has been decoded. For PCI and ISA master-initiated read cycles, UBUSTR is asserted from the falling edge of either IOR# or MEMR#, depending on the cycle type (driven from MEMR# only if BIOS space has been decoded). When the rising edge of IOR# or MEMR# occurs, the SIO negates UBUSTR. For DMA read cycles from the Utility Bus, UBUSTR is asserted when DACKx# is asserted and negated when DACKx# is negated. At all other times, UBUSTR is negated. Upon PCIRST#, this signal is driven low.
UBUSOE#	O	**UTILITY DATA BUS OUTPUT ENABLE:** UBUSOE# is tied directly to the output enable of a 74F245 that buffers the utility data bus, UD[7:0], from the system data bus, SD[7:0]. UBUSOE# is asserted anytime a SIO supported Utility Bus device is decoded, and the devices decode is enabled in the Utility Bus Chip Select Enable Registers. UBUSOE# is asserted from the falling edge of the ISA commands (IOR#, IOW#, MEMR#, or MEMW#) for PCI and ISA master-initiated cycles. UBUSOE# is negated from the rising edge of the ISA command signals for SIO-initiated cycles and the SA[16:0] and LA[23:17] address for ISA master-initiated cycles. For DMA cycles, UBUSOE# is asserted when DACK2# is asserted and negated when DACK2# negated. UBUSOE# is not driven active under the following conditions: **NOTES:** 1. During an I/O access to the floppy controller, if DSKCHG is sampled low at reset. 2. If the Digital Output Register is programmed to ignore DACK2#. 3. During an I/O read access to floppy location 3F7h (primary) or 377h (secondary), if the IDE decode space is disabled (i.e. IDE is not resident on the Utility Bus). 4. During any access to a utility bus peripheral in which its decode space has been disabled. Upon a PCIRST#, this signal is driven inactive (high).
ECSADDR [2:0]	O	**ENCODED CHIP SELECTS:** ECSADDR[2:0] are the encoded chip selects and/or control signals for the Utility Bus peripherals supported by the SIO. The binary code formed by the three signals indicates which Utility Bus device is selected. These signals tie to the address inputs of two external 74F138 decoder chips and are driven valid/invalid from the SA[16:0] and LA[23:17] address lines. Upon PCIRST#, these signals are driven high.
ECSEN#	O	**ENCODED CHIP SELECT ENABLE:** ECSEN# is used to determine which of the two external 74F138 decoders is to be selected. ECSEN# is driven low to select decoder 1 and driven high to select decoder 2. This signal is driven valid/invalid from the SA[16:0] and LA[23:17] address lines (except for the generation of RTCALE#, in which case, ECSEN# is driven active based on IOW# falling, and remains active for two SYSCLKs). During a non-valid address or during an access not targeted for the Utility Bus, this signal is driven high. Upon PCIRST#, this signal is driven high.

3.9 Utility Bus Signals (Continued)

Signal Name	Type	Description
ALT_RST#	O	**ALTERNATE RESET:** ALT_RST# is used to reset the CPU under program control. This signal is AND'ed together externally with the reset signal (KBDRST#) from the keyboard controller to provide a software means of resetting the CPU. This provides a faster means of reset than is provided by the keyboard controller. Writing a 1 to bit 0 in the Port 92 Register causes this signal to pulse low for approximately 4 SYSCLKs. Before another ALT_RST# pulse can be generated, bit 0 must be set to 0. Upon PCIRST#, this signal is driven inactive high (bit 0 in the Port 92 Register is set to 0).
ALT_A20	O	**ALTERNATE A20:** ALT_A20 is used to force A20M# to the CPU low for support of real mode compatible software. This signal is externally OR'ed with the A20GATE signal from the keyboard controller and CPURST to control the A20M# input of the CPU. Writing a 0 to bit 1 of the Port 92 Register forces ALT_A20 low. ALT_A20 low drives A20M# to the CPU low, if A20GATE from the keyboard controller is also low. Writing a 1 to bit 1 of the Port 92 Register force ALT_A20 high. ALT_A20 high drives A20M# to the CPU high, regardless of the state of A20GATE from the keyboard controller. Upon reset, this signal is driven low.
DSKCHG	I	**DISK CHANGE:** DSKCHG is tied directly to the DSKCHG signal of the floppy controller. This signal is inverted and driven on SD7 during I/O read cycles to floppy address locations 3F7h (primary) or 377h (secondary) as shown in the table below. Note that the primary and secondary locations are programmed in the Utility Bus Address Decode Enable/Disable Register "A". **FLOPPYCS# Decode** / **IDECSx# Decode** / **State of SD7 (output)** / **State of UBUSOE#** Enabled / Enabled / Tri-stated / Enabled Enabled / Disabled / Driven via DSKCHG / Disabled Disabled / Enabled / Tri-stated / Enabled (note) Disabled / Disabled / Tri-stated / Disabled **NOTE:** For this mode to be supported, extra logic is required to disable the U-bus transceiver for accesses to 3F7/377. This is necessary because of potential contention between the Utility Bus buffer and a floppy on the ISA Bus driving the system bus at the same time during shared I/O accesses. This signal is also used to determine if the floppy controller is present on the Utility Bus. It is sampled on the trailing edge of PCIRST#, and if high, the Floppy is present. For systems that do not support a Floppy via the SIO, this pin should be strapped low. If sampled low, the SD7 function, UBUSOE#, and ECSADDR[2:0] signals will not be enabled for DMA or programmed I/O accesses to the floppy disk controller. This condition overrides the floppy decode enable bits in the Utility Bus Chip Select A.

3.9 Utility Bus Signals (Continued)

Signal Name	Type	Description
FERR#/ IRQ13	I	**NUMERIC COPROCESSOR ERROR/IRQ13:** This signal has two separate functions, depending on bit 5 in the ISA Clock Divisor Register. This pin functions as a FERR# signal supporting coprocessor errors, if this function is enabled (bit 5 = 1), or as an external IRQ13, if the coprocessor error function is disabled (bit 5 = 0). If programmed to support coprocessor error reporting, this signal is tied to the coprocessor error signal on the CPU. If FERR# is asserted by the coprocessor inside the CPU, the SIO generates an internal IRQ13 to its interrupt controller unit. The SIO then asserts the INT output to the CPU. Also, in this mode, FERR# gates the IGNNE# signal to ensure that IGNNE# is not asserted to the CPU unless FERR# is active. When FERR# is asserted, the SIO asserts INT to the CPU as an IRQ13. IRQ13 continues to be asserted until a write to F0h has been detected. If the Coprocessor error reporting is disabled, FERR# can be used by the system as IRQ13. Upon PCIRST#, this signal provides the standard IRQ13 function. This signal should be pulled high with an external 8.2 KΩ pull-up resistor if the IRQ13 mode is used or the pin is left floating.
IGNNE#	O	**IGNORE ERROR:** This signal is connected to the ignore error pin of the CPU. IGNNE# is only used if the SIO coprocessor error reporting function is enabled in the ISA Clock Divisor Register (bit 5 = 1). If FERR# is active, indicating a coprocessor error, a write to the Coprocessor Error Register (F0h) causes the IGNNE# to be asserted. IGNNE# remains asserted until FERR# is negated. If FERR# is not asserted when the Coprocessor Error Register is written, the IGNNE# is not asserted. IGNNE# is driven high upon a reset.
IRQ12/M	I	**INTERRUPT REQUEST/MOUSE INTERRUPT:** In addition to providing the standard interrupt function as described in the pin description for IRQ[15,14, 11:9, 7:3, 1], this pin also provides a mouse interrupt function. Bit 4 in the ISA Clock Divisor Register determines the functionality of IRQ12/M. When bit 4 = 0, the standard interrupt function is provided and this pin can be tied to the ISA connector. When bit 4 = 1, the mouse interrupt function is provided and this pin can be tied to the DIRQ12 output of the keyboard controller. When the mouse interrupt function is selected, a low to high transition on this signal is latched by the SIO and an INT is generated to the CPU as IRQ12. An interrupt will continue to be generated until a PCIRST# or an I/O read access to address 60h (falling edge of IOR#) is detected. After a PCIRST#, this pin provides the standard IRQ12 function.

3.10 Test Signals

Signal Name	Type	Description
TEST	I	**TEST:** The TEST signal is used to tri-state all of the SIO outputs. During normal operation, this input should be tied to ground.
TESTO	O	**TEST OUTPUT:** This is the output pin used during NAND tree testing.

4.0 REGISTER DESCRIPTION

The SIO contains both PCI configuration registers and non-configuration registers. The configuration registers (Table 3) are located in PCI configuration space and are only accessible from the PCI Bus. Addresses for configuration registers are offset values that appear on AD[7:2] and C/BE#[3:0]. The configuration registers (Section 4.1) can be accessed as Byte, Word (16-bit), or Dword (32-bit) quantities. All multi-byte numeric fields use "little-endian" ordering (i.e., lower addresses contain the least significant parts of the fields).

The non-configuration registers (Table 4) include DMA Registers (Section 4.2), Timer Registers (Section 4.3), Interrupt Controller Registers (Section 4.4), and Control Registers (Section 4.5). All of these registers are accessible from the PCI Bus. In addition, some of the registers are accessible from the ISA Bus. Table 4 indicates the bus access for each register. Except for the DMA scatter/gather registers and the BIOS timer registers, the non-configuration registers can only be accessed as byte quantities. If a PCI master attempts a multi-byte access (i.e., more than one Byte Enable signal asserted), the SIO responds with a target-abort. The scatter/gather registers and BIOS timer registers can be accessed as Byte, Word, or Dword quantities.

Some of the SIO configuration and non-configuration registers contain reserved bits. These bits are labeled "Reserved". Software must take care to deal correctly with bit-encoded fields that are reserved. On reads, software must use appropriate masks to extract the defined bits and not rely on reserved bits being any particular value. On writes, software must ensure that the values of reserved bit positions are preserved. That is, the values of reserved bit positions must first be read, merged with the new values for other bit positions, and the data then written back.

In addition to reserved bits within a register, the SIO contains address locations in the PCI configuration space that are marked "Reserved" (Table 3). The SIO responds to accesses to these address locations by completing the PCI cycle. However, reads of reserved address locations yield all zeroes and writes have no affect on the SIO.

The SIO, upon receiving a hard reset (PCIRST# signal), sets its internal registers to pre-determined **default** states. The default values are indicated in the individual register descriptions.

Table 3. Configuration Registers

Configuration Offset	Register	Register Access	Bus Access
00h–01h	Vendor Identification	RO	PCI Only
02h–03h	Device Identification	RO	PCI Only
04h–05h	Command	R/W	PCI Only
06h–07h	Device Status	R/W	PCI Only
08h	Revision Identification	RO	PCI Only
09h–3Fh	Reserved	—	PCI Only
40h	PCI Control	R/W	PCI Only
41h	PCI Arbiter Control	R/W	PCI Only
42h	PCI Arbiter Priority Control	R/W	PCI Only
43h	PCI Arbiter Priority Control Extension Register	R/W	PCI Only
44h	MEMCS# Control	R/W	PCI Only
45h	MEMCS# Bottom of Hole	R/W	PCI Only
46h	MEMCS# Top of Hole	R/W	PCI Only
47h	MEMCS# Top of Memory	R/W	PCI Only
48h	ISA Address Decoder Control	R/W	PCI Only
49h	ISA Address Decoder ROM Block Enable	R/W	PCI Only
4Ah	ISA Address Decoder Bottom of Hole	R/W	PCI Only
4Bh	ISA Address Decoder Top of Hole	R/W	PCI Only
4Ch	ISA Controller Recovery Timer	R/W	PCI Only
4Dh	ISA Clock Divisor	R/W	PCI Only
4Eh	Utility Bus Chip Select Enable A	R/W	PCI Only
4Fh	Utility Bus Chip Select Enable B	R/W	PCI Only
50h–53h	Reserved	—	PCI Only
54h	MEMCS# Attribute Register #1	R/W	PCI Only
55h	MEMCS# Attribute Register #2	R/W	PCI Only
56h	MEMCS# Attribute Register #3	R/W	PCI Only
57h	Scatter/Gather Relocation Base Address	R/W	PCI Only
58h–5Fh	Reserved	—	PCI Only

Table 3. Configuration Registers (Continued)

Configuration Offset	Register	Register Access	Bus Access
60h	PIRQ0# Route Control	R/W	PCI Only
61h	PIRQ1# Route Control	R/W	PCI Only
62h	PIRQ2# Route Control	R/W	PCI Only
63h	PIRQ3# Route Control	R/W	PCI Only
64h–7Fh	Reserved	—	PCI Only
80h–81h	BIOS Timer Base Address	R/W	PCI Only
82h–9Fh	Reserved	—	PCI Only
A0h	SMI Control (SMICNTL)	R/W	PCI Only
A1h	Reserved	—	PCI Only
A2h–A3h	SMI Enable (SMIEN)	R/W	PCI Only
A4h–A7h	System Event Enable (SEE)	R/W	PCI Only
A8h	Fast Off Timer (FTMR)	R/W	PCI Only
A9h	Reserved	—	PCI Only
AAh–ABh	SMI Request (SMIREQ)	R/W	PCI Only
ACh	Clock Throttle STPCLK# Low Timer (CTLTMRL)	R/W	PCI Only
ADh	Reserved	—	PCI Only
AEh	Clock Throttle STPCLK# High Timer (CTLTMRH)	R/W	PCI Only
AFh–FFh	Reserved	—	PCI Only

Table 4. Non-Configuration Registers

Address	Function Unit	Register	Register Access	Bus Access
0000h	DMA	DMA1 CH0 Base and Current Address	R/W	PCI Only
0001h	DMA	DMA1 CH0 Base and Current Count	R/W	PCI Only
0002h	DMA	DMA1 CH1 Base and Current Address	R/W	PCI Only
0003h	DMA	DMA1 CH1 Base and Current Count	R/W	PCI Only
0004h	DMA	DMA1 CH2 Base and Current Address	R/W	PCI Only
0005h	DMA	DMA1 CH2 Base and Current Count	R/W	PCI Only
0006h	DMA	DMA1 CH3 Base and Current Address	R/W	PCI Only
0007h	DMA	DMA1 CH3 Base and Current Count	R/W	PCI Only
0008h	DMA	DMA1 Status(R) Command(W)	R/W	PCI Only
0009h	DMA	DMA1 Write Request	WO	PCI Only
000Ah	DMA	DMA1 Write Single Mask Bit	WO	PCI Only
000Bh	DMA	DMA1 Write Mode	WO	PCI Only
000Ch	DMA	DMA1 Clear Byte Pointer	WO	PCI Only
000Dh	DMA	DMA1 Master Clear	WO	PCI Only
000Eh	DMA	DMA1 Clear Mask	WO	PCI Only
000Fh	DMA	DMA1 Read/Write All Mask Register Bits	R/W	PCI Only
0020h	Interrupt	INT 1 Control	R/W	PCI/ISA
0021h	Interrupt	INT 1 Mask	R/W	PCI/ISA
0040h	Timer	Timer Counter 1–Counter 0 Count	R/W	PCI/ISA
0041h	Timer	Timer Counter 1–Counter 1 Count	R/W	PCI/ISA
0042h	Timer	Timer Counter 1–Counter 2 Count	R/W	PCI/ISA
0043h	Timer	Timer Counter 1 Command Mode	WO	PCI/ISA
0060h(2)	Control	Reset UBus IRQ12	RO	PCI/ISA
0061h	Control	NMI Status and Control	R/W	PCI/ISA
0070h(2)	Control	CMOS RAM Address and NMI Mask	WO	PCI/ISA
0078h-007Bh(3,4,5)	Timer	BIOS Timer	R/W	PCI Only
0080h(1)	DMA	DMA Page Register Reserved	R/W	PCI/ISA
0081h	DMA	DMA Channel 2 Page Register	R/W	PCI/ISA
0082h	DMA	DMA Channel 3 Page Register	R/W	PCI/ISA
0083h	DMA	DMA Channel 1 Page Register	R/W	PCI/ISA

Table 4. Non-Configuration Registers (Continued)

Address	Function Unit	Register	Register Access	Bus Access
0084h(1)	DMA	DMA Page Register Reserved	R/W	PCI/ISA
0085h(1)	DMA	DMA Page Register Reserved	R/W	PCI/ISA
0086h(1)	DMA	DMA Page Register Reserved	R/W	PCI/ISA
0087h	DMA	DMA Channel 0 Page Register	R/W	PCI/ISA
0088h(1)	DMA	DMA Page Register Reserved	R/W	PCI/ISA
0089h	DMA	DMA Channel 6 Page Register	R/W	PCI/ISA
008Ah	DMA	DMA Channel 7 Page Register	R/W	PCI/ISA
008Bh	DMA	DMA Channel 5 Page Register	R/W	PCI/ISA
008Ch(1)	DMA	DMA Page Register Reserved	R/W	PCI/ISA
008Dh(1)	DMA	DMA Page Register Reserved	R/W	PCI/ISA
008Eh(1)	DMA	DMA Page Register Reserved	R/W	PCI/ISA
008Fh	DMA	DMA Low Page Register Refresh	R/W	PCI/ISA
0090h(6)	DMA	DMA Page Register Reserved	R/W	PCI/ISA
0092h(2)	Control	Port 92 Register	R/W	PCI/ISA
0094h(6)	DMA	DMA Page Register Reserved	R/W	PCI/ISA
0095h(6)	DMA	DMA Page Register Reserved	R/W	PCI/ISA
0096h(6)	DMA	DMA Page Register Reserved	R/W	PCI/ISA
0098h(6)	DMA	DMA Page Register Reserved	R/W	PCI/ISA
009Ch(6)	DMA	DMA Page Register Reserved	R/W	PCI/ISA
009Dh(6)	DMA	DMA Page Register Reserved	R/W	PCI/ISA
009Eh(6)	DMA	DMA Page Register Reserved	R/W	PCI/ISA
009Fh	DMA	DMA Low Page Register Refresh	R/W	PCI/ISA
00A0h	Interrupt	INT 2 Control Register	R/W	PCI/ISA
00A1h	Interrupt	INT 2 Mask Register	R/W	PCI/ISA
00B2h	P.M.	Advanced Power Management Control Port	R/W	PCI Only
00B3h	P.M.	Advanced Power Management Status Port	R/W	PCI Only
00C0h	DMA	DMA2 CH0 Base and Current Address	R/W	PCI Only
00C2h	DMA	DMA2 CH0 Base and Current Count	R/W	PCI Only
00C4h	DMA	DMA2 CH1 Base and Current Address	R/W	PCI Only
00C6h	DMA	DMA2 CH1 Base and Current Count	R/W	PCI Only
00C8h	DMA	DMA2 CH2 Base and Current Address	R/W	PCI Only

Table 4. Non-Configuration Registers (Continued)

Address	Function Unit	Register	Register Access	Bus Access
00CAh	DMA	DMA2 CH2 Base and Current Count	R/W	PCI Only
00CCh	DMA	DMA2 CH3 Base and Current Address	R/W	PCI Only
00CEh	DMA	DMA2 CH3 Base and Current Count	R/W	PCI Only
00D0h	DMA	DMA2 Status(r) Command(w) Register	R/W	PCI Only
00D2h	DMA	DMA2 Write Request Register	WO	PCI Only
00D4h	DMA	DMA2 Write Single Mask Bit Register	WO	PCI Only
00D6h	DMA	DMA2 Write Mode Register	WO	PCI Only
00D8h	DMA	DMA2 Clear Byte Pointer Register	WO	PCI Only
00DAh	DMA	DMA2 Master Clear Register	WO	PCI Only
00DCh	DMA	DMA2 Clear Mask Register	WO	PCI Only
00DEh	DMA	DMA2 Read/Write All Mask Register Bits	R/W	PCI Only
00F0h(2)	Control	Coprocessor Error Register	WO	PCI/ISA
0372h(2)	Control	Secondary Floppy Disk Digital Output Register	WO	PCI/ISA
03F2h(2)	Control	Primary Floppy Disk Digital Output Register	WO	PCI/ISA
040Ah(3)	DMA	Scatter/Gather Interrupt Status Register	RO	PCI Only
040Bh	DMA	DMA1 Extended Mode Register	WO	PCI/ISA
0410h(3,4)	DMA	CH0 Scatter/Gather Command	WO	PCI Only
0411h(3,4)	DMA	CH1 Scatter/Gather Command	WO	PCI Only
0412h(3,4)	DMA	CH2 Scatter/Gather Command	WO	PCI Only
0413h(3,4)	DMA	CH3 Scatter/Gather Command	WO	PCI Only
0415h(3,4)	DMA	CH5 Scatter/Gather Command	WO	PCI Only
0416h(3,4)	DMA	CH6 Scatter/Gather Command	WO	PCI Only
0417h(3,4)	DMA	CH7 Scatter/Gather Command	WO	PCI Only
0418h(3,4)	DMA	CH0 Scatter/Gather Status	RO	PCI Only
0419h(3,4)	DMA	CH1 Scatter/Gather Status	RO	PCI Only
041Ah(3,4)	DMA	CH2 Scatter/Gather Status	RO	PCI Only
041Bh(3,4)	DMA	CH3 Scatter/Gather Status	RO	PCI Only
041Dh(3,4)	DMA	CH5 Scatter/Gather Status	RO	PCI Only
041Eh(3,4)	DMA	CH6 Scatter/Gather Status	RO	PCI Only
041Fh(3,4)	DMA	CH7 Scatter/Gather Status	RO	PCI Only
0420h–0423h(3,4)	DMA	CH0 Scatter/Gather Descriptor Table Pointer	R/W	PCI Only

Table 4. Non-Configuration Registers (Continued)

Address	Function Unit	Register	Register Access	Bus Access
0424h–0427h(3,4)	DMA	CH1 Scatter/Gather Descriptor Table Pointer	R/W	PCI Only
0428h–042Bh(3,4)	DMA	CH2 Scatter/Gather Descriptor Table Pointer	R/W	PCI Only
042Ch–042Fh(3,4)	DMA	CH3 Scatter/Gather Descriptor Table Pointer	R/W	PCI Only
0434h–0437h(3,4)	DMA	CH5 Scatter/Gather Descriptor Table Pointer	R/W	PCI Only
0438h–043Bh(3,4)	DMA	CH6 Scatter/Gather Descriptor Table Pointer	R/W	PCI Only
043Ch–043Fh(3,4)	DMA	CH7 Scatter/Gather Descriptor Table Pointer	R/W	PCI Only
0481h	DMA	DMA CH2 High Page Register	R/W	PCI/ISA
0482h	DMA	DMA CH3 High Page Register	R/W	PCI/ISA
0483h	DMA	DMA CH1 High Page Register	R/W	PCI/ISA
0487h	DMA	DMA CH0 High Page Register	R/W	PCI/ISA
0489h	DMA	DMA CH6 High Page Register	R/W	PCI/ISA
048Ah	DMA	DMA CH7 High Page Register	R/W	PCI/ISA
048Bh	DMA	DMA CH5 High Page Register	R/W	PCI/ISA
04D0h	Interrupt	Edge/Level Control Register—INT CNTRL 1	R/W	PCI Only
04D1h	Interrupt	Edge/Level Control Register—INT CNTRL 2	R/W	PCI Only
04D6h	DMA	DMA2 Extended Mode Register	WO	PCI/ISA

NOTES:
1. PCI write cycles to these address locations flow through to the ISA Bus. PCI read cycles to these address locations do not flow through to the ISA Bus.
2. PCI read and write cycles to these address locations flow through to the ISA Bus.
3. The I/O address of this register is relocatable. The value shown in this table is the default address location.
4. This register can be accessed as a Byte, Word, or Dword quantity.
5. If this register location is enabled, PCI accesses to the BIOS Timer Register do not flow through to the ISA Bus. If disabled, accesses to this address location flow through to the ISA Bus.
6. When the DMAAC bit in the PCI Control Register is '0', the 82378 will alias I/O accesses in the 80h–8Fh range to the 90h–9Fh range. Write accesses to these address locations flow through to the ISA Bus. Read cycles to these address locations do not flow through to the ISA Bus. When DMAAC = 1, the SIO will only respond to the 80h-8Fh range and read and write accesses to these addresses in the 90h–9Fh range will be forwarded from the PCI bus to the ISA Bus (I/O port 92h is always a distinct register in the 90h–9Fh range and is always fully decoded, regardless of the setting of the DMAAC bit).

4.1 SIO Configuration Register Description

This section describes the SIO configuration registers. These registers include the Mandatory Header Registers (located in the first 64 bytes of configuration space) and the SIO specific registers (located from configuration offset 40h–56h).

4.1.1 VID—VENDOR IDENTIFICATION REGISTER

Address Offset: 00h, 01h
Default Value: 8086h
Attribute: Read Only
Size: 16 bits

The VID Register contains the vendor identification number. This 16-bit register combined with the Device Identification Register uniquely identifies any PCI device. Writes to this register have no effect.

Bits[15:0]: Vendor Identification Number
This is a 16-bit value assigned to Intel.

4.1.2 DID—DEVICE IDENTIFICATION REGISTER

Address Offset: 02h, 03h
Default Value: 0484h
Attribute: Read Only
Size: 16 bits

The DID Register contains the device identification number. This register, along with the Vendor ID, uniquely identifies the SIO. Writes to this register have no effect.

Bits[15:0]: Device Identification Number
This is a 16-bit value assigned to the SIO.

4.1.3 COM—COMMAND REGISTER

Address Offset: 04h–05h
Default Value: 0007h
Attribute: Read/Write
Size: 16 bits

Bits[15:5]: Reserved
Read 0.

Bit 4: PMWE (Postable Memory Write Enable)
Enable Postable memory write, memory write and invalidate, and memory read Pre-fetch commands. The SIO does not support these commands as a master or slave so this bit is not implemented. This bit will always be read as a 0.

Bit 3: SCE (Special Cycle Enable)
When this bit is set to a "1", the SIO will recognize PCI Special Cycles. When set to "0", the SIO will ignore all PCI Special Cycles. This bit MUST be enabled in the 82378ZB if the STPCLK feature is being used.

Bit 2: BME (Bus Master Enable)
Since the SIO always requests the PCI Bus on behalf of ISA masters, DMA, or line buffer PCI requests, this bit is hardwired to a 1 and will always be read as a 1.

Bit 1: MSE (Memory Space Enable)
Enables SIO to accept a PCI-originated memory cycle. Since the SIO always responds to PCI-originated memory cycles (and ISA-bound cycles) by asserting DEVSEL#, this bit is hardwired to a 1 and will always be read as a 1.

Bit 0: IOSE (I/O Space Enable)
Enable SIO to accept a PCI-originated I/O cycle. Since the SIO always responds to a master I/O cycle, this bit is hardwired to a 1 and will always be read as a 1.

4.1.4 DS—DEVICE STATUS REGISTER

Address Offset: 06h, 07h
Default Value: 0200h
Attribute: Read/Write
Size: 16 bits

DSR is a 16-bit status register that reports the occurrence of a PCI master-abort by the SIO or a PCI target-abort when the SIO is a master. The register also indicates the SIO DEVSEL# signal timing that is hardwired in the SIO.

Bit 15: Reserved
Read as 0.

Bit 14: SERRS (SERR# Status)
This bit is set by the PCI devices that assert the SERR# signal. Since SERR# is only an input to the SIO, this bit is not implemented and will always be read as 0.

Bit 13: MA (Master-Abort Status)
When the SIO, as a master, generates a master-abort, MA is set to a 1. Software sets MA to 0 by writing a 1 to this bit location.

Bit 12: RTA (Received Target-Abort Status)
When the SIO is a master on the PCI Bus and receives a target-abort, this bit is set to a 1. Software sets RTA to 0 by writing a 1 to this bit location.

Bit 11: STA (Signaled Target-Abort Status)
This bit is set to a 1 by the SIO when it generates a target-abort.

Bits[10:9]: DEVT (SIO DEVSEL# Timing Status)
This 2-bit field defines the timing for DEVSEL# assertion. These read only bits indicate the SIO's DEVSEL# timing when performing a positive decode. Since the SIO always generates DEVSEL# with medium timing, DEVT = 01. This DEVSEL# timing does not include Configuration cycles.

Bits[8:0]: Reserved
Read as 0's.

4.1.5 RID—REVISION IDENTIFICATION REGISTER

Address Offset: 08h
Default Value: xxh (dependent on Part Revision)
Attribute: Read Only
Size: 4 bits (upper nibble reserved)

This 4-bit register contains the revision number for the SIO. This number indicates the stepping number of the component. Additionally, the upper nibble of the value is reserved. BIOS should mask the upper nibble when reading this register. These bits are read only. Writes to this register have no effect.

Bits[7:4]: Reserved
These 4 bits are reserved.

Bits[3:0]: Revision Identification Number
This is an 4-bit value that indicates the revision identification number for the SIO. Numbers used so far include:
0h: 82378IB A0-Stepping
1h: 82378IB B0-Stepping
WAS NOT IMPLEMENTED. B0 steppings read 0h also. Read the BIOS Timer Base Address Configuration Register to identify between A0 and B0 steppings.
A0 = 0000h
B0 = 0078h
3h: 82378ZB A0-Stepping

4.1.6 PCICON—PCI CONTROL REGISTER

Address Offset: 40h
Default Value: 20h
Attribute: Read/Write
Size: 8 bits

This 8-bit register controls the Line Buffer operation, the SIO's PCI Posted Write Buffer enabling, and the DEVSEL# signal sampling point. The PCICON Register also controls how the SIO responds to INTA cycles on the PCI Bus and if the reserved DMA page registers are aliased from 80h–8Fh to 90h–9Fh.

Bit 7: Reserved
Read as 0.

Bit 6: DMAAC (DMA Reserved Page Register Aliasing Control)
These register bits control whether the SIO will alias I/O accesses in the 80h–8Fh to the 90h–9Fh range. When DMAAC = 0, the SIO will alias I/O accesses in the 80h–8Fh to the 90h–9Fh range (AD4 is not used for decoding the DMA reserved page registers). When DMAAC = 1, the SIO will only respond to the 80h–8Fh range (AD4 is used for decoding the DMA reserved page registers). Read and write accesses to the 90h–9Fh range will be forwarded from the PCI bus to the ISA bus.

NOTE:
I/O port 92h is always a distinct register in the 90h–9Fh range and is always fully decoded, regardless of the setting of this bit.

Bit 5: IAE (Interrupt Acknowledge Enable)
When IAE = 0, the SIO ignores INTA cycles generated on the PCI Bus. However, when disabled, the SIO still responds to accesses to the 8259's register set and allows poll mode functions. When IAE = 1, the SIO responds to INTA cycles in the normal fashion. This bit defaults to a 1 (respond to INTA cycles).

Bits[4:3]: SDSP (Subtractive Decoding Sample Point)
The SDSP field determines the DEVSEL# sample point, after which an inactive DEVSEL# results in the SIO forwarding the unclaimed PCI cycle to the ISA Bus (subtractive decoding). This setting should match the slowest device in the system.

Bit	4	3	Operation
	0	0	Slow sample point
	0	1	Typical sample point
	1	0	Fast sample point
	1	1	Reserved

Bit 2: PPBE (PCI Posted Write Buffer Enable)
When PPBE = 0, the PCI posted write buffer is disabled. When PPBE = 1, the PCI posted write buffer is enabled. This bit defaults to disabled mode (PPBE = 0).

Bit 1: ILBC (ISA Master Line Buffer Configuration)
When ILBC = 0, the Line Buffer is in single transaction mode. When ILBC = 1, the Line Buffer is in 8-byte mode. This bit applies only to ISA Master transfers. This bit defaults to single transaction mode (ILBC = 0).

Bit 0: DLBC (DMA Line Buffer Configuration)
When DLBC = 0, the Line Buffer is in single transaction mode. When DLBC = 1, the Line Buffer is in 8-byte mode. This bit applies only to DMA transfers. This bit defaults to single transaction mode (DLBC = 0).

4.1.7 PAC—PCI ARBITER CONTROL REGISTER

Address Offset:	41h
Default Value:	00h
Attribute:	Read/Write
Size:	8 bits

This 8-bit register controls the operation of the PCI arbiter. The PAC register enables/disables the guaranteed access time mode, controls bus lock cycles, enables/disables CPU bus parking, and controls the master retry timer.

Bits[7:5]: Reserved
Read as 0's.

Bits[4:3]: MRT (Master Retry Timer)
This 2-bit field determines the number of PCICLKs after the first retry that a PCI initiator's Bus request will be unmasked.

Bit	4	3	Operation
	0	0	Timer disabled, retries never masked.
	0	1	Retries unmasked after 16 PCICLK's.
	1	0	Retries unmasked after 32 PCICLK's.
	1	1	Retries unmasked after 64 PCICLK's.

Bit 2: BP (Bus Park)
Set to a 1 the SIO will park CPUREQ# on the PCI bus when it detects the PCI bus idle. If Bus Park is disabled, the SIO takes responsibility for driving AD, C/BE# and PAR upon detection of bus idle state if the internal arbiter is enabled.

Bit 1: BL (Bus Lock)
This bit selects between bus lock and resource lock. When BL = 1, Bus Lock is selected. The arbiter considers the entire PCI bus locked upon initiation of any locked transaction. When BL = 0, resource lock is enabled. A locked agent is considered a locked resource and other agents may continue normal PCI transactions.

Bit 0: GAT (Guaranteed Access Time)
This bit enables/disables the guaranteed access time mode. When GAT = 1, the SIO is configured for Guaranteed Access Time mode. This mode is available in order to guarantee the 2.5 μs CHRDY time-out specification for the ISA Bus. When the SIO is an Initiator on behalf of an ISA master, the PCI and memory busses are arbitrated for in serial and must be owned before the ISA master is given ownership of the ISA Bus. When GAT = 0, the guaranteed access time mode is disabled. When guaranteed access time mode is disabled, the ISA master is first granted the ISA Bus and then the SIO arbitrates for the PCI Bus.

4.1.8 PAPC—PCI ARBITER PRIORITY CONTROL REGISTER

Address Offset: 42h
Default Value: 04h
Attribute: Read/Write
Size: 8 bits

This register controls the PCI arbiter priority scheme. The arbiter supports six masters arranged through four switching banks. This permits the six masters to be arranged in a purely rotating priority scheme, one of twenty-four fixed priority schemes, or a hybrid combination of the fixed and rotating priority schemes. Bits[4:7] enable/disable rotate priority for the four banks. For each bit, a 1 enables the mode and a 0 disables the mode. If both fixed and rotate modes are enabled for the same bank, the bank will be in rotate mode. For example, if both bits 0 and 4 are set to a 1, bank 0 will be in rotate mode.

Bit 7: Bank 3 Rotate Control

Bit 6: Bank 2 Rotate Control

Bit 5: Bank 1 Rotate Control

Bit 4: Bank 0 Rotate Control

Bit 3: Bank 2 Fixed Priority Mode Select B

Bit 2: Bank 2 Fixed Priority Mode Select A

Bit 1: Bank 1 Fixed Priority Mode Select

Bit 0: Bank 0 Fixed Priority Mode Select

Fixed Priority Mode

The fixed bank control bits select which requester is the highest priority device within that particular bank.

Table 5. Fixed Mode Bank Control Bits

Mode	Bank					Priority					
	3	2b	2a	1	0	Highest					Lowest
00	0	0	0	0	0	SIOREQ#	REQ0#	REQ2#	REQ3#	CPUREQ#	REQ1#
01	0	0	0	0	1	REQ0#	SIOREQ#	REQ2#	REQ3#	CPUREQ#	REQ1#
02	0	0	0	1	0	SIOREQ#	REQ0#	REQ2#	REQ3#	REQ1#	CPUREQ#
03	0	0	0	1	1	REQ0#	SIOREQ#	REQ2#	REQ3#	REQ1#	CPUREQ#
04	0	0	1	0	0	CPUREQ#	REQ1#	SIOREQ#	REQ0#	REQ2#	REQ3#
05	0	0	1	0	1	CPUREQ#	REQ1#	REQ0#	SIOREQ#	REQ2#	REQ3#
06	0	0	1	1	0	REQ1#	CPUREQ#	SIOREQ#	REQ0#	REQ2#	REQ3#
07	0	0	1	1	1	REQ1#	CPUREQ#	REQ0#	SIOREQ#	REQ2#	REQ3#
08	0	1	0	0	0	REQ2#	REQ3#	CPUREQ#	REQ1#	SIOREQ#	REQ0#
09	0	1	0	0	1	REQ2#	REQ3#	CPUREQ#	REQ1#	REQ0#	SIOREQ#
0A	0	1	0	1	0	REQ2#	REQ3#	REQ1#	CPUREQ#	SIOREQ#	REQ0#
0B	0	1	0	1	1	REQ2#	REQ3#	REQ1#	CPUREQ#	REQ0#	SIOREQ#
0C–0F	0	1	1	x	x	Reserved					
10	1	0	0	0	0	SIOREQ#	REQ0#	REQ3#	REQ2#	CPUREQ#	REQ1#
11	1	0	0	0	1	REQ0#	SIOREQ#	REQ3#	REQ2#	CPUREQ#	REQ1#
12	1	0	0	1	0	SIOREQ#	REQ0#	REQ3#	REQ2#	REQ1#	CPUREQ#
13	1	0	0	1	1	REQ0#	SIOREQ#	REQ3#	REQ2#	REQ1#	CPUREQ#
14	1	0	1	0	0	CPUREQ#	REQ1#	SIOREQ#	REQ0#	REQ3#	REQ2#
15	1	0	1	0	1	CPUREQ#	REQ1#	REQ0#	SIOREQ#	REQ3#	REQ2#
16	1	0	1	1	0	REQ1#	CPUREQ#	SIOREQ#	REQ0#	REQ3#	REQ2#
17	1	0	1	1	1	REQ1#	CPUREQ#	REQ0#	SIOREQ#	REQ3#	REQ2#
18	1	1	0	0	0	REQ3#	REQ2#	CPUREQ#	REQ1#	SIOREQ#	REQ0#
19	1	1	0	0	1	REQ3#	REQ2#	CPUREQ#	REQ1#	REQ0#	SIOREQ#
1A	1	1	0	1	0	REQ3#	REQ2#	REQ1#	CPUREQ#	SIOREQ#	REQ0#
1B		1	0	1	1	REQ3#	REQ2#	REQ1#	CPUREQ#	REQ0#	SIOREQ#
1C–1F	1	1	1	x	x	Reserved					

Rotating Priority Mode
When any Bank Rotate Control bit is set to a one, that particular bank rotates between the two requesting inputs. Any or all banks can be set in rotate mode. If, within a rotating bank, the highest priority input does not have an active request, then the lower priority input will be granted the bus. However, this does not change the rotation scheme. When the bank toggles, the previously lowest priority input will become the highest priority input. Because of this, the maximum latency a device may encounter would be two complete rotations.

4.1.9 ARBPRIX—PCI ARBITER PRIORITY CONTROL EXTENSION REGISTER

Address Offset: 43h
Default Value: 00h
Attribute: Read/Write
Size: 8 bits

This register provides the Fixed Priority Mode select for Bank 3 of the arbiter. The ARBPRIX Register fields are shown.

Bits[7:1]: Reserved
Read as 0.

Bit 0: Bank 3 Fixed Priority Mode Select

0 = REQ2# higher priority
1 = REQ3# higher priority

4.1.10 MEMCSCON—MEMCS# CONTROL REGISTER

Address Offset: 44h
Default value: 00h
Attribute: Read/Write
Size: 8 bits

Bits 0-2 of this register enable MEMCS# blocks. PCI addresses within the enabled blocks result in the generation of MEMCS#. Note that the 0–512 KByte segment does not have RE and WE bits. The 0–512 KByte segment can only be turned off with the MEMCS# Master Enable bit (bit 4). Note also, that when the RE and WE bits are both 0 for a particular segment, the PCI master can not access the segment.

Bits[7:5]: Reserved
Read as 0's.

Bit 4: MEMCS# Master Enable
When the MEMCS# master enable bit is set to a 1, the SIO asserts MEMCS# for all accesses to the defined MEMCS# region (that have been programmed in this register and the MAR1, MAR2, and MAR3 Registers). Also, when this bit is a 1, the positive decoding functions enabled by having the ISA Clock Divisor Register bit 6 = 1 and the Utility Bus Chip Select Register "A" bit 6 = 1 are ignored. Subtractive decoding is provided for these memory areas, instead. When the MEMCS# master enable bit is set to a 0, the entire MEMCS# function is disabled. When this bit is 0, MEMCS# will never be asserted.

Bit 3: Write Enable For 0F0000h–0FFFFFh (Upper 64 KByte BIOS)
When this bit is set to a 1, the SIO generates MEMCS# for PCI master memory write accesses to the address range 0F0000h–0FFFFFh. When this bit is set to a 0, the SIO does not generate MEMCS# for PCI master memory write accesses to the address range 0F0000h–0FFFFFh.

Bit 2: Read Enable For 0F0000h–0FFFFFh (Upper 64 KByte BIOS)
When this bit is set to a 1, the SIO generates MEMCS# for PCI master memory read accesses to the address range 0F0000h–0FFFFFh. When this bit is set to a 0, the SIO does not generate MEMCS# for PCI master memory read accesses to the address range 0F0000h–0FFFFFh.

Bit 1: Write Enable For 080000h–09FFFFh (512 KByte–640 KByte)
When this bit is set to a 1, the SIO generates MEMCS# for PCI master memory write accesses to the address range 080000h–09FFFFh. When this bit is set to a 0, the SIO does not generate MEMCS# for PCI master memory write accesses to the address range 080000h–09FFFFh.

Bit 0: Read Enable For 080000h–09FFFFh (512 KByte–640 KByte)
When this bit is set to a 1, the SIO generates MEMCS# for PCI master memory read accesses to the address range 080000h–09FFFFh. When this bit is set to a 0, the SIO does not generate MEMCS# for PCI master memory read accesses to the address range 080000h–09FFFFh.

4.1.11 MEMCSBOH—MEMCS# BOTTOM OF HOLE REGISTER

Address Offset: 45h
Default value: 10h
Attribute: Read/Write
Size: 8 bits

This register defines the bottom of the MEMCS# hole. MEMCS# is not generated for accesses to addresses within the hole defined by this register and the MCSTOH Register. The hole is defined by the following equation: TOH ≥ address ≥ BOH. TOH is the top of the MEMCS# hole defined by the MCSTOH Register and BOH is the bottom of the MEMCS# hole defined by this register.

For example, to program the BOH at 1 MByte, the value of 10h should be written to this register. To program the BOH at 2 MByte + 64 KByte this register should be programmed to 21h. To program the BOH at 8 MByte this register should be programmed to 80h.

When the TOH < BOH the hole is effectively disabled. It is the responsibility of the programmer to guarantee that the BOH is at or above 1 MB. AD[31:24] must be 0's for the hole, meaning the hole is restricted to be under the 16 MByte boundary. The default value for the BOH and TOH effectively disables the hole.

Bit 7: AD23

Bit 6: AD22

Bit 5: AD21

Bit 4: AD20

Bit 3: AD19

Bit 2: AD18

Bit 1: AD17

Bit 0: AD16

4.1.12 MEMCSTOH—MEMCS# TOP OF HOLE REGISTER

Address Offset: 46h
Default value: 0Fh
Attribute: Read/Write
Size: 8 bits

This register defines the top of the MEMCS# hole. MEMCS# is not generated for accesses to addresses within the hole defined by this register and the MCSBOH Register. The hole is defined by the following equation: TOH ≥ address ≥ BOH. TOH is the top of the MEMCS# hole defined by this register and BOH is the bottom of the MEMCS# hole defined by the MCSBOH Register.

For example, to program the TOH at 1 MByte + 64 KByte, this register should be programmed to 10h. To program the TOH at 2 MByte + 128 KByte this register should be programmed to 21h. To program the TOH at 12 MByte this register should be programmed to BFh.

When the TOH < BOH the hole is effectively disabled. It is the responsibility of the programmer to guarantee that the TOH is above 1 MByte. AD[31:24] must be 0's for the hole, meaning the hole is restricted to be under the 16 MByte boundary. The default value for the BOH and TOH effectively disables the hole.

Bit 7: AD23

Bit 6: AD22

Bit 5: AD21

Bit 4: AD20

Bit 3: AD19

Bit 2: AD18

Bit 1: AD17

Bit 0: AD16

4.1.13 MEMCSTOM—MEMCS# TOP OF MEMORY REGISTER

Address Offset: 47h
Default value: 00h
Attribute: Read/Write
Size: 8 bits

This register determines MEMCS# top of memory boundary. The top of memory boundary ranges up to 512 MBytes, in 2 MByte increments. This register is typically set to the top of main memory. Accesses ≥ 2 MByte and ≤ top of memory boundary results in the assertion of the MEMCS# signal (unless the address resides in the hole programmed by the MCSBOH and MCSTOH Registers). A value of 00h disables this 2 MByte-to-top of memory region. A value of 00h assigns the top of memory to include 2 MByte − 1. A value of FFh assigns the top of memory to include 512 MByte − 1.

Bit 7: AD28

Bit 6: AD27

Bit 5: AD26

Bit 4: AD25

Bit 3: AD24

Bit 2: AD23

Bit 1: AD22

Bit 0: AD21

4.1.14 IADCON—ISA ADDRESS DECODER CONTROL REGISTER

Address Offset: 48h
Default value: 01h
Attribute: Read/Write
Size: 8 bits

This register enables the forwarding of ISA or DMA memory cycles to the PCI Bus. In addition, this register sets the top of the "1 MByte to top of main memory" region.

Bits[7:4]:
The top can be assigned in 1 MByte increments from 1 MByte up to 16 MByte. ISA master or DMA accesses within this region are forwarded to PCI unless they are within the hole.

Bits	7	6	5	4	Top of Memory
	0	0	0	0	1 MByte
	0	0	0	1	2 MByte
	0	0	1	0	3 MByte
	0	0	1	1	4 MByte
	0	1	0	0	5 MByte
	0	1	0	1	6 MByte
	0	1	1	0	7 MByte
	0	1	1	1	8 MByte
	1	0	0	0	9 MByte
	1	0	0	1	10 MByte
	1	0	1	0	11 MByte
	1	0	1	1	12 MByte
	1	1	0	0	13 MByte
	1	1	0	1	14 MByte
	1	1	1	0	15 MByte
	1	1	1	1	16 MByte

Bits[3:0]:
ISA and DMA Memory Cycle To PCI Bus Enables. The memory block is enabled by writing a 1 to the corresponding bit position. Setting the bit to 0 disables the corresponding block. ISA or DMA memory cycles to the enabled blocks result in the ISA cycle being forwarded to the PCI Bus. The BIOSCS# enable bit (bit 6 in the UBCSA Register) for the 896K–960K region overrides the function of bit 3 of this register. If the BIOSCS# bit is set to a 1, the ISA or DMA memory cycle is always contained to ISA, regardless of the setting of bit 3 in this register. If the BIOSCS# bit is disabled, the cycle is forwarded to the PCI bus if bit 3 in this register is enabled. Refer to Section 5.5.1.2 for a complete description of BIOS decoding.

Bit	Memory Block
0	0–512 KByte Memory
1	512–640 KByte Memory
2	640–768 KByte VGA Memory
3	896–960 KByte Low BIOS

4.1.15 IADRBE—ISA ADDRESS DECODER ROM BLOCK ENABLE REGISTER

Address Offset: 49h
Default value: 00h
Attribute: Read/Write
Size: 8 bits

ISA addresses within the enabled ranges result in the ISA memory cycle being forwarded to the PCI Bus. For each bit position, the memory block is enabled if the bit is set to 1 and is disabled if the bit is set to 0. If the memory block is disabled, the ISA cycle is not forwarded to the PCI Bus.

Bit 7: 880–896K Memory Enable

Bit 6: 864–880K Memory Enable

Bit 5: 848–864K Memory Enable

Bit 4: 832–848K Memory Enable

Bit 3: 816–832K Memory Enable

Bit 2: 800–816K Memory Enable

Bit 1: 784–800K Memory Enable

Bit 0: 768–784K Memory Enable

4.1.16 IADBOH—ISA ADDRESS DECODER BOTTOM OF HOLE REGISTER

Address Offset: 4Ah
Default value: 10h
Attribute: Read/Write
Size: 8 bits

This register defines the bottom of the ISA Address Decoder hole. The hole is defined by the following equation: TOH ≥ address ≥ BOH, where BOH is the bottom of the hole address programmed into this register and TOH is the top of the hole address programmed into the IADTOH Register. ISA master or DMA addresses falling within the hole will not be forwarded to the PCI Bus. The hole can be sized in 64 KByte increments and placed anywhere between 1 MByte and 16 MByte on any 64 KByte boundary. It is the responsibility of the programmer to guarantee that the BOH is at or above 1 MByte. A[31:24] must be 0's for the hole, meaning the hole is restricted to be under the 16 MByte boundary. When TOH < BOH, the hole is effectively disabled. The default value for the BOH and TOH disables the hole.

For example, to program the BOH at 1 MByte, this register should be set to 10h. To program the BOH at 2 MBytes, this register should be set to 20h. To program the BOH at 8 MBytes, this register should be set to 80h. These settings are shown in Figure 2.

Bit 7: A23

Bit 6: A22

Bit 5: A21

Bit 4: A20

Bit 3: A19

Bit 2: A18

Bit 1: A17

Bit 0: A16

4.1.17 IADTOH—ISA ADDRESS DECODER TOP OF HOLE REGISTER

Address Offset: 4Bh
Default value: 0Fh
Attribute: Read/Write
Size: 8 bits

This register defines the top of the ISA Address Decoder hole. The hole is defined by the following equation: TOH ≥ address ≥ BOH, where BOH is the bottom of the hole address programmed into the LADBOH Register and TOH is the top of the hole address programmed into this Register. ISA master or DMA addresses falling within the hole will not be forwarded to the PCI Bus. The hole can be sized in 64 KByte increments and placed anywhere between 1 MByte and 16 MByte on any 64 KByte boundary. It is the responsibility of the programmer to guarantee that the TOH is at or above 1 MByte. A[31:24] must be 0's for the hole, meaning the hole is restricted to be under the 16 MByte boundary. When TOH < BOH, the hole is disabled. The default value for the BOH and TOH disables the hole.

For example, to program the TOH at 1 MByte + 64 KByte, this register should be set to 10h. To program the TOH at 2 MByte + 128 KByte, this register should be set to 21h. To program the TOH at 12 MByte, this register should be set to BFh. These settings are shown in Figure 2.

Bit 7: A23

Bit 6: A22

Bit 5: A21

Bit 4: A20

Bit 3: A19

Bit 2: A18

Bit 1: A17

Bit 0: A16

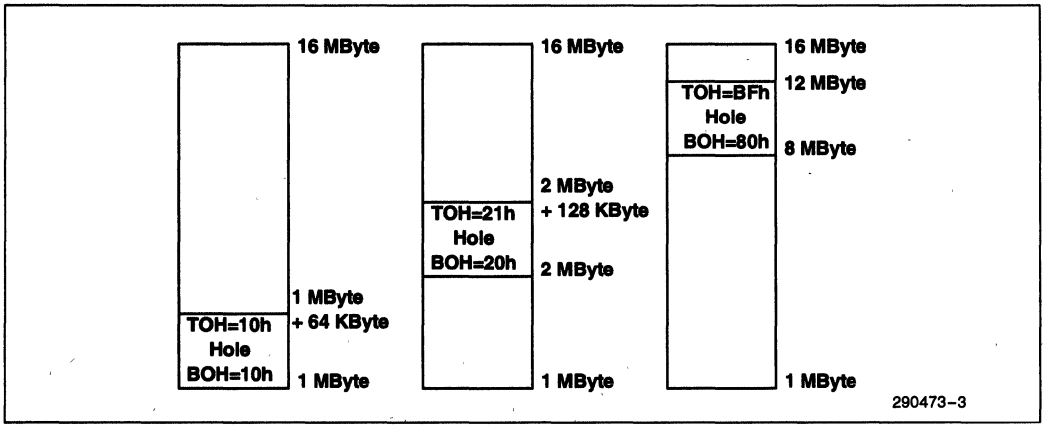


Figure 2. ISA Address Decoder Hole Examples

Table 6. Examples of ISA Decoding

Test Case Description	TOM (48h)	TOH (4Bh)	BOH (4Ah)	Address (hex)	Address	Result
8MB TOM, no hole @ 1M	7xh	0Fh	10h	01000000h 00FFFFFFh 00800000h 007FFFFFh 00100000h 000FFFFFh	16MB 16MB-1 8MB 8MB-1 1MB 1MB-1	To PCI ISA ISA To PCI To PCI ISA (BIOS)
4MB TOM, no hole @ 2M	3xh	1Fh	20h	01000000h 00FFFFFFh 00400000h 003FFFFFh 00200000h 001FFFFFh 00100000h	16MB 16MB-1 4MB 4MB-1 2MB 2MB-1 1MB	To PCI ISA ISA To PCI To PCI To PCI To PCI
1MB TOM, no hole @ 1M	0xh	0Fh	10h	01000000h 00FFFFFFh 00100000h 000FFFFFh	16MB 16MB-1 1MB 1MB-1	To PCI ISA ISA ISA (BIOS)
16MB TOM, 64KB hole @ 15MB	Fxh	F0h	F0h	01000000h 00FFFFFFh 00F10000h 00F0FFFFh 00F00000h 00EFFFFFh 00E10000h 00E0FFFFh 00E00000h 00DFFFFFh	16MB 16MB-1 15MB + 64KB 15MB + 64KB-1 15MB 15MB-1 14MB + 64KB 14MB + 64KB-1 14MB 14MB-1	To PCI To PCI To PCI ISA ISA To PCI To PCI To PCI To PCI To PCI
12MB TOM, 2MB + 128KB hole @ 2MB	Bxh	21h	20h	01000000h 00FFFFFFh 00C00000h 00BFFFFFh 00220000h 0021FFFFh 00210000h 0020FFFFh 00200000h 001FFFFFh 00100000h	16MB 16MB-1 12MB 12MB-1 2MB + 128KB 2MB + 128KB-1 2MB + 64KB 2MB + 64KB-1 2MB 2MB-1 1MB	To PCI ISA ISA To PCI To PCI ISA ISA ISA ISA To PCI To PCI

Table 6. Examples of ISA Decoding (Continued)

Test Case Description	TOM (48h)	TOH (4Bh)	BOH (4Ah)	Address (hex)	Address	Result
5MB TOM, 3MB hole @	4xh	47h	18h	01000000h	16MB	To PCI
				00FFFFFFh	16MB-1	ISA
				00500000h	5MB	ISA
				004FFFFFh	5MB-1	To PCI
				00480000h	4.5MB	To PCI
				0047FFFFh	4.5MB-1	ISA
				00180000h	1.5MB	ISA
				0017FFFFh	1.5MB-1	To PCI
				00100000h	1MB	To PCI

4.1.18 ICRT—ISA CONTROLLER RECOVERY TIMER REGISTER

Address Offset: 4Ch
Default Value: 56h
Attribute: Read/Write
Size: 8 bits

The I/O recovery mechanism in the SIO is used to add additional recovery delay between PCI originated 8-bit and 16-bit I/O cycles to the ISA bus. The SIO automatically forces a minimum delay of five SYSCLKs between back-to-back 8- and 16-bit I/O cycles to the ISA bus. The delay is measured from the rising edge of the I/O command (IOR# or IOW#) to the falling edge of the next BALE. If a delay of greater than five SYSCLKs is required, the ISA I/O Recovery Time Register can be programmed to increase the delay in increments of SYSCLKs. Note that no additional delay is inserted for back-to-back I/O "sub cycles" generated as a result of byte assembly or disassembly. This register defaults to 8- and 16-bit recovery enabled with two clocks added to the standard I/O recovery.

Bit 7: Reserved
Read as 0.

Bit 6: 8-Bit I/O Recovery Enable
This bit enables the recovery times programmed into bits 0 and 1 of this register. When this bit is set to 1, the recovery times shown for bits[5:3] are enabled. When this bit is set to 0, recovery times are disabled.

Bits[5:3]: 8-Bit I/O Recovery Times
This 3-bit field defines the recovery times for 8-bit I/O. Programmable delays between back-to-back 8-bit PCI cycles to ISA I/O slaves is shown in terms of ISA clock cycles (SYSCLK) added to the five minimum. The selected delay programmed into this field is enabled/disabled via bit 6 of this register.

Bit 5	4	3	SYSCLK Added	Total SYSCLKs
0	0	1	+1	6
0	1	0	+2	7
0	1	1	+3	8
1	0	0	+4	9
1	0	1	+5	10
1	1	0	+6	11
1	1	1	+7	12
0	0	0	+8	13

Bit 2: 16-Bit I/O Recovery Enable
This bit enables the recovery times programmed into bits 0 and 1 of this register. When this bit is set to 1, the recovery times shown for bits 0 and 1 are enabled. When this bit is set to 0, recovery times are disabled.

Bits[1:0]: 16-Bit I/O Recovery Times
This 2-bit field defines the recovery time for 16-bit I/O. Programmable delays between back-to-back 16-bit PCI cycles to ISA I/O slaves is shown in terms of ISA clock cycles (SYSCLK) added to the five minimum. The selected delay programmed into this field is enabled/disabled via bit 2 of this register.

Bit 1	Bit 0	SYSCLK Added	Total SYSCLKs
0	1	+1	6
1	0	+2	7
1	1	+3	8
0	0	+4	9

4.1.19 ICD—ISA CLOCK DIVISOR REGISTER

Address Offset: 4Dh
Default Value: 40h
Attribute: Read/Write
Size: 8 bits

This register selects the integer value used to divide the PCI clock (PCICLK) to generate the ISA clock (SYSCLK). In addition, this register provides an ISA Reset bit to software control RSTDRV, a bit to enable/disable the MOUSE function, a bit to enable/disable the coprocessor error support, and a bit to disable the positive decode for the upper 64 KBytes of BIOS at the top of 1 MByte (F0000h–FFFFFh) and aliased regions.

Bit 7: Reserved

Bit 6: Positive Decode of Upper 64 KByte BIOS Enable
This bit enables (bit 6 = 1) and disables (bit 6 = 0) the positive decode of the upper 64 KBytes of BIOS area at the top of 1 MByte (F0000h–FFFFFh) and the aliased regions at the top of 4 GBytes (FFFF0000h–FFFFFFFFh) and 4 GByte-1 MByte (FFEF0000–FFEFFFFFh). When bit 6 = 1, these address regions are positively decoded, unless bit 4 in the MEMCS# Control Register is set to a 1 in which case these regions are subtractively decoded. When bit 6 = 0, these address regions are subtractively decoded. The encoded chip selects for BIOSCS# and the UBUSOE# signal will always be generated when these locations are accessed, regardless of the state of this bit. A reset sets this bit to a 1 (positive decode enabled).

Bit 5: Coprocessor Error Enable
This bit is used to enable and disable the Coprocessor error support. When enabled (bit 5 = 1), the FERR# input, when driven active, triggers an IRQ13 to the SIO's interrupt controller. FERR# is also used to gate the IGNNE# output. When disabled (bit 5 = 0), the FERR# signal can be used as IRQ13 and the coprocessor support is disabled. A reset sets this bit to 0 (coprocessor support disabled).

Bit 4: IRQ12/M Mouse Function Enable
When this bit is set to 1, IRQ12/M provides the mouse function. When this bit is set to 0, IRQ12/M provides the standard IRQ12 interrupt function. A hard reset sets this bit to 0.

Bit 3: RSTDRV Enable
This bit is used to enable RSTDRV on the ISA Bus. When this bit is set to 1, RSTDRV is asserted and remains asserted until this bit is set to a 0. When set to 0, normal operation of RSTDRV is provided. This bit should be used during configuration to reset the ISA Bus when changing the clock divisor. For a reset, this bit defaults to 0. Note that the software must ensure that RSTDRV is asserted for a minimum of 1 μs.

Bit[2:0]: PCICLK-to-ISA SYSCLK Divisor
These bits are used to select the integer that is used to divide the PCICLK down to generate the ISA SYSCLK. Upon reset, these bits are set to 000 (divisor of 4 selected). For PCI frequencies less than 33 MHz (not including 25 MHz), a clock divisor value must be selected that ensures that the ISA Bus frequency does not violate the 6 MHz to 8.33 MHz SYSCLK specification.

Bit 2	Bit 1	Bit 0	Divisor	SYSCLK
0	0	0	4 (33 MHz)	8.33 MHz
0	0	1	3 (25 MHz)	8.33 MHz
0	1	0	Reserved	
0	1	1	Reserved	
1	0	0	Reserved	
1	0	1	Reserved	
1	1	0	Reserved	
1	1	1	Reserved	

4.1.20 UBCSA—UTILITY BUS CHIP SELECT A REGISTER

Address Offset: 4Eh
Default Value: 07h
Attribute: Read/Write
Size: 8 bits

This register enables/disables accesses to the RTC, keyboard controller, Floppy Disk controller, IDE, and BIOS locations E0000h–EFFFFh and FFF80000h–FFFDFFFFh. Disabling any of these bits prevents the encoded chip select bits (ECSADDR[2:0]) and utility bus transceiver control signal (UBUSOE#) for that device from being generated.

This register is also used to select which address range (primary or secondary) will be decoded for the resident floppy controller and IDE. This ensures that there is no contention with the Utility bus transceiver driving the system data bus during read accesses to these devices.

Bit 7: Extended BIOS Enable
When bit 7 = 1 (enabled), PCI accesses to locations FFF80000h–FFFDFFFFh result in the generation of the encoded signals (ECSADDR[2:0]) for BIOS. When enabled, PCI master accesses to this area are positively decoded and UBUSOE# is generated. When this bit is disabled (bit 7 = 0), the SIO does not generate the encoded (ECSADDR[2:0]) signals or UBUSOE#.

Bit 6: Lower BIOS Enable
When bit 6 = 1 (enabled), PCI or ISA accesses to the lower 64 KByte BIOS block (E0000h–EFFFFh) at the top of 1 MByte, or the aliases at the top of 4 GByte and 4 GByte–1 MByte results in the generation of the encoded (ECSADDR[2:0]) signals for BIOS. When enabled, PCI master accesses to this area are positively decoded to the ISA Bus, unless bit 4 in the MEMCS# Control Register is set to a 1 in which case these regions are subtractively decoded. Also, when enabled, ISA master or DMA master accesses to this region are not forwarded to the PCI Bus. When this bit is disabled (bit 6 = 0), the SIO does not generate the encoded (ECSADDR[2:0]) signals. Also, when this bit is disabled, ISA master or DMA accesses to this region are forwarded to PCI, if bit 3 in the IADCON Register is set to 1.

Bit 4: IDE Decode Enable
Bit 4 enables/disables IDE locations 1F0h–1F7h (primary) or 170h–177h (secondary) and 3F6h, 3F7h (primary) or 376h, 377h (secondary). When bit 4 = 1, the IDE encoded chip select signals and the Utility Bus transceiver signal (UBUSOE#) are generated for these addresses. When bit 4 = 0, the IDE encoded chip select signals and the Utility Bus transceiver signal (UBUSOE#) are not generated for these addresses.

Bit [5, 3:2]: Floppy Disk Address Locations Enable
Bits 2 and 3 are used to enable or disable the floppy locations as indicated below. A PCIRST# sets bit 2 to 1 and bit 3 to 0. Bit 5 is used to select between the primary and secondary address range used by the Floppy Controller and the IDE. Only primary or only secondary can be programmed at any one time. A PCIRST# sets this bit to 0 (primary).

The following table shows how these bits are used to select the floppy controller:

Address	Bit 5	Bit 3	Bit 2	DSKCHG	ECSADDR[2:0]	FLOPPYCS#
X	X	X	X	0	1 1 1	1
3F0h, 3F1h	0	1	X	1	1 0 0	0
3F2h–3F7h	0	X	1	1	1 0 0	0 (note)
370h, 371h	1	1	X	1	1 0 0	0
372h–37Fh	1	X	1	1	1 0 0	0 (note)

NOTE:
If IDE decode is enabled (bit 4 = 1), all accesses to locations 03F6h and 03F7h (primary) or 0376h and 0377h (secondary) result in the ECSADDR[2:0] signals generating a decode for IDECS1# (FLOPPYCS# is not generated). An external AND gate can be used to tie IDECS1# and FLOPPYCS# together to insure that the floppy is enabled for these accesses. If IDE decode is disabled (bit 4 = 0), and the decode for the floppy is enabled, then the encoded chip selects for the floppy locations are generated.

Bit 1: Keyboard Controller Address Location Enable
Enables (1) or disables (0) the Keyboard controller address locations 60h, 62h, 64h, and 66h. When this bit is set to 0, the Keyboard Controller encoded chip select signals (ECSADDR[2:0]) and the Utility Bus transceiver signal (UBUSOE#) are not generated for these locations.

Bit 0: RTC Address Location Enable
Enables (1) or disables (0) the RTC address locations 70h–77h. When this bit is set to 0, the RTC encoded chip select signals (ECSADDR[2:0]), RTCALE#, RTCCS#, and UBUSOE# signals are not generated for these addresses.

4.1.21 UBCSB—UTILITY BUS CHIP SELECT B REGISTER

Address Offset: 4Fh
Default Value: 4Fh
Attribute: Read/Write
Size: 8 bits

This register is used to enable/disable accesses to the serial ports and parallel port locations supported by the SIO. When disabled, the ECSADDR(2:0) encoded chip select bits and Utility Bus Transceiver control signal (UBUSOE#), for that device, are not generated. This register is also used to disable accesses to port 92 and enable or disable configuration RAM decode.

Bit 7: Configuration RAM Decode Enable
This bit is used to enable (bit 7 = 1) or disable (bit 7 = 0) I/O write accesses to location 0C00h and I/O read/write accesses to locations 0800h–08FFh. When enabled, the encoded chip select signals for generating an external configuration page chip select (CPAGECS#) are generated for accesses to 0C00h. The encoded chip select signals for generating an external configuration memory chip select (CFIGMEMCS#) are generated for accesses to 0800h–08FFh. When bit 7 = 0, configuration RAM decode is disabled and the CPAGECS# and CFIGMEMCS# are not generated for the corresponding accesses.

Bit 6: Port 92 Enable
This bit is used to enable/disable access to Port 92. When bit 6 = 1, Port 92 is enabled. When bit 6 = 0, Port 92 is disabled. When a PCIRST# occurs, this bit is set to 1 (enable).

Bits[5:4]: Parallel Port Enable
These bits are used to select the parallel port address range: (LPT1, LPT2, LPT3, or disable). When a PCIRST# occurs, this field is set to 00 (LPT1).

Bit 5	Bit 4	Function
0	0	3BCh–3BFh (LPT1)
0	1	378h–37Fh (LP2)
1	0	278h–27Fh (LPT3)
1	1	Disabled

Bits[3:2]: Serial Port B Enable
These bits are used to assign serial port B address range: (COM1, COM2, or disable). If either COM1 or COM2 address ranges are selected, the encoded chip select signals [ECSADDR(2:0)] for Port B will be generated. A PCIRST# sets bits[3:2] to 11 (Port B disabled).

Bit 3	2	Function
0	0	3F8h–3FFh (COM1)
0	1	2F8h–2FFh (COM2)
1	0	Reserved
1	1	Port B Disabled

NOTE:
If Serial port A and B are programmed for the same I/O address, the encoded chip select signals, ECSADDR(2:0), for port B are disabled.

Bits[1:0]: Serial Port A Enable
These bits are used to assign serial port A address range: (COM1, COM2, or disable). If either COM1 or COM2 address ranges are selected, the encoded chip select signals (ECSADDR[2:0]) for Port A will be generated. A PCIRST# sets bits[1:0] to 11 (port A disabled).

Bit 1	0	Function
0	0	3F8h–3FFh (COM1)
0	1	2F8h–2FFh (COM2)
1	0	Reserved
1	1	Port A disabled

NOTE:
If Serial port A and B are programmed for the same I/O address, the encoded chip select signals, ECSADDR[2:0], for port B are disabled.

4.1.22 MAR1—MEMCS# ATTRIBUTE REGISTER #1

Address Offset: 54h
Default Value: 00h
Attribute: Read/Write
Size: 8 bits

RE—Read Enable. When the RE bit (bit 6, 4, 2, 0) is set to a 1, the SIO generates MEMCS# for PCI master, DMA, or ISA master memory read accesses to the corresponding segment. When the RE bit is set to a 0, the SIO does not generate MEMCS# for PCI master memory read accesses to the corresponding segment. When the RE and WE bits are both 0 (or bit 4 in the MEMCS# Control Register is set to a 0–disabled), the PCI master, DMA, or ISA master can not access the corresponding segment.

WE—Write Enable. When the WE bit (bit 7, 5, 3, 1) is set to a 1, the SIO generates MEMCS# for PCI master, DMA, or ISA master memory write accesses to the corresponding segment. When this bit is set to a 0, the SIO does not generate MEMCS# for PCI master memory write accesses to the corresponding segment. When the RE and WE bits are both 0 (or bit 4 in the MEMCS# Control Register is set to a 0–disabled), the PCI master, DMA, or ISA master can not access the corresponding segment.

Bit 7: 0CC000h–0CFFFFh Exp. ROM: WE

Bit 6: 0CC000h–0CFFFFh Exp. ROM: RE

Bit 5: 0C8000h–0CBFFFh Exp. ROM: WE

Bit 4: 0C8000h–0CBFFFh Exp. ROM: RE

Bit 3: 0C4000h–0C7FFFh Exp. ROM: WE

Bit 2: 0C4000h–0C7FFFh Exp. ROM: RE

Bit 1: 0C0000h–0C3FFFh Exp. ROM: WE

Bit 0: 0C0000h–0C3FFFh Exp. ROM: RE

4.1.23 MAR2—MEMCS# ATTRIBUTE REGISTER #2

Address Offset: 55h
Default Value: 00h
Attribute: Read/Write
Size: 8 bits

RE—Read Enable. When the RE bit (bit 6, 4, 2, 0) is set to a 1, the SIO generates MEMCS# for PCI master, DMA, or ISA master memory read accesses to the corresponding segment. When this bit is set to a 0, the SIO does not generate MEMCS# for PCI master memory read accesses to the corresponding segment. When the RE and WE bits are both 0 (or bit 4 in the MEMCS# Control Register is set to a 0–disabled), the PCI master, DMA, or ISA master can not access the corresponding segment.

WE—Write Enable. When the WE bit (bit 7, 5, 3, 1) is set to a 1, the SIO generates MEMCS# for PCI master, DMA, or ISA master memory write accesses to the corresponding segment. When this bit is set to a 0, the SIO does not generate MEMCS# for PCI master memory write accesses to the corresponding segment. When the RE and WE bits are both 0 (or bit 4 in the MEMCS# Control Register is set to a 0–disabled), the PCI master, DMA, or ISA master can not access the corresponding segment.

Bit 7: 0DC000h–0DFFFFh Exp. ROM : WE

Bit 6: 0DC000h–0DFFFFh Exp. ROM : RE

Bit 5: 0D8000h–0DBFFFh Exp. ROM : WE

Bit 4: 0D8000h–0DBFFFh Exp. ROM : RE

Bit 3: 0D4000h–0D7FFFh Exp. ROM : WE

Bit 2: 0D4000h–0D7FFFh Exp. ROM : RE

Bit 1: 0D0000h–0D3FFFh Exp. ROM : WE

Bit 0: 0D0000h–0D3FFFh Exp. ROM : RE

4.1.24 MAR3—MEMCS# ATTRIBUTE REGISTER #3

Address Offset: 56h
Default Value: 00h
Attribute: Read/Write
Size: 8 bits

RE—Read Enable. When the RE bit (bit 6, 4, 2, 0) is set to a 1, the SIO generates MEMCS# for PCI master, DMA, ISA master memory read accesses to the corresponding segment. When this bit is set to a 0, the SIO does not generate MEMCS# for PCI master memory read accesses to the corresponding segment. When the RE and WE bits are both 0 (or bit 4 in the MEMCS# Control Register is set to a 0–disabled), the PCI master can not access the corresponding segment.

WE—Write Enable. When the WE bit (bit 7, 5, 3, 1) is set to a 1, the SIO generates MEMCS# for PCI master, DMA, ISA master memory write accesses to the corresponding segment. When this bit is set to a 0, the SIO does not generate MEMCS# for PCI master memory write accesses to the corresponding segment. When the RE and WE bits are both 0 (or bit 4 in the MEMCS# Control Register is set to a 0–disabled), the PCI master can not access the corresponding segment.

Bit 7: 0EC000h–0EFFFFh Lower 64 KByte BIOS: WE

Bit 6: 0EC000h–0EFFFFh Lower 64 KByte BIOS: RE

Bit 5: 0E8000h–0EBFFFh Lower 64 KByte BIOS WE

Bit 4: 0E8000h–0EBFFFh Lower 64 KByte BIOS: RE

Bit 3: 0E4000h–0E7FFFh Lower 64 KByte BIOS: WE

Bit 2: 0E4000h–0E7FFFh Lower 64 KByte BIOS: RE

Bit 1: 0E0000h–0E3FFFh Lower 64 KByte BIOS: WE

Bit 0: 0E0000h–0E3FFFh Lower 64 KByte BIOS: RE

4.1.25 DMA SCATTER/GATHER RELOCATION BASE ADDRESS REGISTER

Address Offset: 57h
Default Value: 04h
Attribute: Read/Write
Size: 8 bits

The value programmed into this register determines the high order I/O address of the Scatter/Gather Command Registers, Scatter/Gather Status Registers, and the Scatter/Gather Descriptor Table Registers. The default value is 04h so the first S/G register default address is at 0410h.

Bit 7: A15

Bit 6: A14

Bit 5: A13

Bit 4: A12

Bit 3: A11

Bit 2: A10

Bit 1: A9

Bit 0: A8

4.1.26 PIRQ[3:0]#—PIRQ ROUTE CONTROL REGISTERS

Register Name: PIRQ0#, PIRQ1#, PIRQ2#, PIRQ3# Route Control

Address Offset: 60h, 61h, 62h, 63h
Default Value: 80h
Attribute: Read/Write
Size: 8 bits

These registers control the routing of PCI Interrupts (PIRQ[0:3]#) to the PC compatible Interrupts. Each PCI interrupt can be independently routed to 1 of 11 compatible interrupts. Note that two or more PCI interrupts (PIRQ[3:0]#) can be steered into the same IRQ signal (the interrupts are level sensitive and can be shared).

Each IRQ to which a PCI Interrupt is steered into must have its interrupt set to level sensitive in the Edge/Level Control Register.

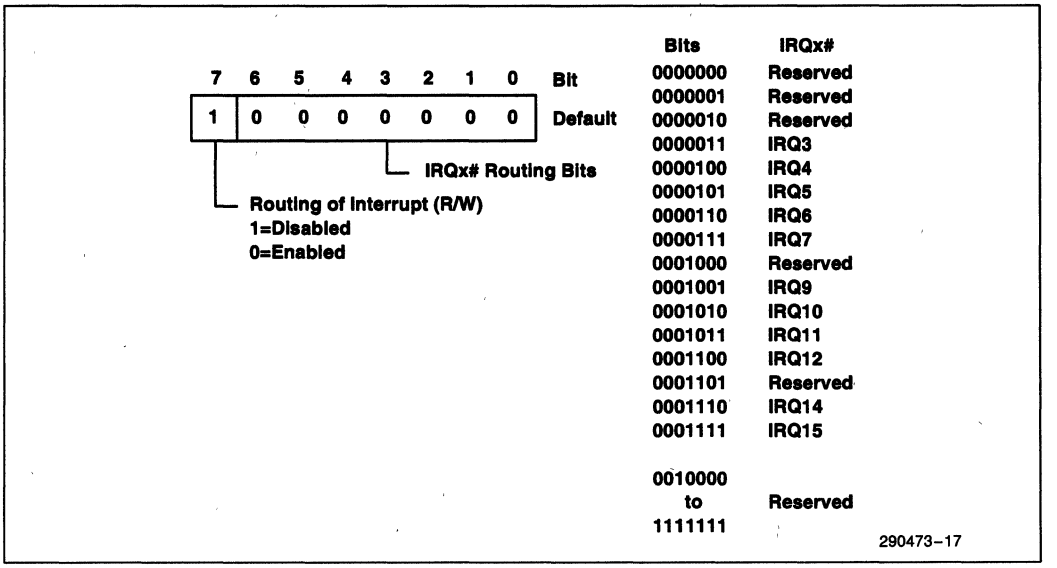


Figure 3. PIRQ Route Control Registers

Bit 7: Routing of Interrupts
When enabled, this bit routes the PCI Interrupt signal to the PC compatible interrupt signal specified in bits[3:0]. At reset, this bit is disabled (set to 1).

Bits[6:4]: Reserved
Read as 0's.

Bits[3:0]: IRQx# Routing Bits
These bits specify which IRQ signal to generate.

4.1.27 BIOS TIMER BASE ADDRESS REGISTER

Address Offset: 80h–81h
Default value: 0078h
Attribute: Read/Write
Size: 16 bits

This register determines the base address for the BIOS Timer Register located in the I/O space. The base address can be set at Dword boundary anywhere in the 64 KByte I/O space. This register also provides the BIOS Timer access enable/disable control bit.

Bits[15:2]: BIOS Timer Base Address
Bits[15:2] correspond to PCI address lines A[15:2].

Bit 1: Reserved

Bit 0: BIOS Timer Access Enable
When bit 0 = 1, access to the BIOS Timer is enabled. When bit 0 = 0, access to the BIOS Timer is disabled. The default value is 0 (disabled).

4.1.28 SMICNTL—SMI CONTROL REGISTER

Address Offset: A0h
Default value: 08h
Attribute: Read/Write
Size: 8 bits

Bit 7: Reserved
Reserved for future Intel use.

Bit 6: Reserved
Reserved for future Intel use.

Bits[5:4]: Reserved
Reserved for future Intel use.

Bit 3: CTMRFRZ
Used to freeze the timers when in SMM. When this bit is set, the Fast Off timer will stop counting. This prevents time-outs from occurring while executing SMM code.

Bit 2: CSTPCLKTH
When set, the STPCLK# throttle is enabled.

Bit 1: CSTPCLKEN
When set, a read from the APMC register will cause STPCLK# to be asserted. CSTPCLKEN will be cleared by writing it to 0 or by any write to the APMC register. Enables SW to put the CPU into a low power state.

Bit 0: CSMIGATE
When this bit is written to "0" SMI# will be deasserted. When this bit is written to a "1", SMI# will be asserted if any SMIs are pending.

4.1.29 SMIEN—SMI ENABLE REGISTER

Address Offset: A2h–A3h
Default value: 0000h
Attribute: Read/Write
Size: 16 bits

The following bits control the enabling of associated hardware events that will generate an SMI. When set to a "1", SMI# will be asserted when the associated event occurs. When bit 7 is set, writes to the APM Control port (APMC) will generate an SMI.

Bits[15:8]: Reserved
Will be read as 0. Writes have no effect.

Bit 7: SAPMCEN
Write to APM Control Port.

Bit 6: SEXTSMIEN
EXTSMI# input asserted.

Bit 5: SFOFFTMREN
Fast Off (Idle) Timer Enable.

Bit 4: SIRQ12EN
PS/2 Mouse Interrupt.

Bit 3: SIRQ8EN
RTC Alarm Interrupt.

Bit 2: SIRQ4EN
COM2/COM4 Interrupt (Mouse).

Bit 1: SIRQ3EN
COM1/COM3 Interrupt (Mouse).

Bit 0: SIRQ1EN
Keyboard Interrupt.

4.1.30 SEE—SYSTEM EVENT ENABLE REGISTER

Address Offset:	A4h, A5h, A6h, A7h
Default value:	00000000h
Attribute:	Read/Write
Size:	32 bits

These bits are used to enable the corresponding hardware events as system events. When set to a "1", anytime the associated hardware event occurs, the Fast Off Timer is reloaded with its initial count. Also, when enabled the associated hardware system event is recognized as a Break Event causing STPCLK# to be deasserted.

Bit 31: FSMIEN
Prevents the system from entering Fast Off and causes STPCLK# to be deasserted when an SMI occurs.

Bit 30: Reserved
Will be read as 0. Writes have no effect.

Bit 29: FNMIEN
Prevents the system from entering Fast Off and causes STPCLK# to be deasserted when an NMI occurs (parity error for example).

Bits[28:16]: Reserved
Will be read as 0. Writes have no effect.

Bits[15:3]: FIRQ[15:3]EN
This prevents the system from entering Fast Off and causes STPCLK# to be deasserted when selected hardware interrupts occur.

Bit 2: Reserved
Will be read as 0. Writes have no effect.

Bits[1:0]: FIRQ[1:0]EN
Prevents the system from entering Fast Off and causes STPCLK# to be deasserted when selected hardware interrupts occur.

4.1.31 FTMR—FAST OFF TIMER

Address Offset:	A8h
Default value:	0Fh
Attribute:	Read/Write
Size:	8 bits

The Fast Off Timer is used to indicate (through an SMI) that the system has been idle for a pre-programmed period of time. The Fast Off Timer consists of a count down timer that is decremented every minute. The value programmed into this register gets loaded into the Fast Off Timer when an enabled system event occurs. Each count represents one minute. When the timer expires, an SMI Special Cycle is generated. Writes to the FTMRLD register cause the Fast Off Timer to be loaded. When this register is read, the value last written to this register is returned.

PROGRAMMER'S NOTE:
Before writing to the FTMRLD register the Fast Off Timer must be stopped by writing a "1" to the CTMRFRZ bit. The Fast Off Timer will begin decrementing when the CTMRFRZ bit is subsequently set to "0".

Bits[7:0]: FTMRLD[7:0]
A write to the FTMRLD register when the Fast Off Timer is stopped (CTMRFRZ = 1) will load the Fast Off Timer with the value being written to FTMRLD register. When the Fast Off Timer is enabled (CTMRFRZ = 0) it counts down from the value loaded into it. When the Fast Off Timer reaches 00h it will trigger an SMI. If an enabled system event occurs before the Fast Off Timer reaches 00h the Fast Off Timer is reloaded with the value stored in the FTMRLD register. A read from the FTMRLD register will return the value last written to this register.

4.1.32 SMIREQ-SMI REQUEST REGISTER

Address Offset: AAh, ABh
Default value: 00h
Attribute: Read/Write
Size: 16 bits

These bits are status bits indicating the cause of the SMI. When an enabled event causes an SMI, the hardware automatically sets the corresponding event's request bit. The request bits are cleared by writing a "0" to them. Only the hardware can set request bits to a "1". In the event that the hardware is trying to set the bit to a "1" at the same time that it is being cleared, the hardware set to "1" will dominate.

Bits[15:8]: Reserved
Reserved for future Intel use.

Bit 7: RAPMC
When set to a "1" indicates that a write to the APM Control Port caused an SMI#.

Bit 6: REXT
When set to a "1", indicates that EXTSMI# was sampled asserted, causing an SMI#.

Bit 5: RFOFFTMR
Fast Off Timer expired causing an SMI#.

Bit 4: RIRQ12
IRQ12 was asserted causing an SMI#.

Bit 3: RIRQ8
IRQ8 was asserted causing an SMI#.

Bit 2: RIRQ4
IRQ4 was asserted causing an SMI#.

Bit 1: RIRQ3
IRQ3 was asserted causing an SMI#.

Bit 0: RIRQ1
IRQ1 was asserted causing an SMI#.

4.1.33 CTLTMRL—CLOCK THROTTLE STPCLK# LOW TIMER

Address Offset: ACh
Default Value: 00h
Attribute: Read/Write
Size: 8 bits

The value in this register defines the duration of the STPCLK# asserted period when the CSTPCLKTH bit is set. The value in this register is loaded into the STPCLK# Timer when STPCLK# is asserted. The STPCLK# timer runs off a 32 μs clock. Note that the timer does not begin to count until the Stop Grant Special Cycle is received.

Bits[7:0]: KSTPLOLD[7:0]
The value in this register defines the duration of the STPCLK# asserted period when the CSTPCLKTH bit is set.

4.1.34 CTLTMRH—CLOCK THROTTLE STPCLK# HIGHTIMER

Address Offset: AEh
Default value: 00h
Attribute: Read/Write
Size: 8 bits

The value in this register defines the duration of the STPCLK# deasserted period when the CSTPCLKTH bit is set. The value in this register is loaded into the STPCLK# Timer when STPCLK# is deasserted. The STPCLK# timer runs off a 32 μs clock.

Bits[7:0]: KSTPHILD[7:0]
The value in this register defines the duration of the STPCLK# deasserted period when the CSTPCLKTH bit is set.

4.2 DMA Register Description

The SIO contains DMA circuitry that incorporates the functionality of two 82C37 DMA controllers (DMA1 and DMA2). The DMA registers control the operation of the DMA controllers and are all accessible from the PCI Bus via PCI I/O space. In addition, some of the registers are accessed from the ISA Bus via ISA I/O space. Table 4, at the beginning of Section 4.0 lists the bus access for each register.

This section describes the DMA registers. Unless otherwise stated, a PCIRST# sets each register to its default value. The operation of the DMA is further described in Section 5.4, DMA Controller.

4.2.1 DCOM—DMA COMMAND REGISTER

Address Offset: Channels 0-3–08h
Channels 4–7–0D0h
Default Value: 00h
Attribute: Write Only
Size: 8 bits

This 8-bit register controls the configuration of the DMA. It is programmed by the microprocessor in the Program Condition and is cleared by PCIRST# or a Master Clear instruction. Note that disabling channels 4–7 also disables channels 0–3, since channels 0–3 are cascaded onto channel 4. The DREQ and DACK# channel assertion sensitivity is assigned by channel group, not per individual channel. For priority resolution, the DMA consists of two logical channel groups—channels 0–3 (Controller 1-DMA1) and channels 4–7 (Controller 2-DMA2). Each group can be assigned fixed or rotating priority. Both groups can be assigned fixed priority, one group can be assigned fixed priority and the second rotating priority, or both groups can be assigned rotating priority. Following a PCIRST# or DMA Master Clear, both DMA1 and DMA2 are enabled in fixed priority, the DREQ sense level is active high, and the DACK# assertion level is active low.

Bit 7: DACK# Assert Level (DACK#[3:0], [7:5])
Bit 7 controls the DMA channel request acknowledge (DACK#) assertion level. Following PCIRST#, the DACK# assertion level is active low. The low level indicates recognition and acknowledgment of the DMA request to the DMA slave requesting service. Writing a 0 to bit 7 assigns active low as the assertion level. When a 1 is written to this bit, a high level on the DACK# line indicates acknowledgment of the request for DMA service to the DMA slave.

Bit 6: DREQ Sense Assert Level (DREQ[3:0], [7:5])
Bit 6 controls the DMA channel request (DREQ) assertion detect level. Following PCIRST#, the DREQ sense assert level is active high. In this condition, an active high level sampled on DREQ is decoded as an active DMA channel request. Writing a 0 to bit 6 assigns active high as the sense assert level. When a 1 is written to this bit, a low level on the DREQ line is decoded as an active DMA channel request.

Bit 5: Reserved
Must be 0.

Bit 4: DMA Group Arbitration Priority
Each channel group is individually assigned either fixed or rotating arbitration priority. At PCIRST#, each group is initialized in fixed priority. Writing a 0 to bit 4 assigns fixed priority to the channel group, while writing a 1 assigns rotating priority to the group.

Bit 3: Reserved
Must be 0.

Bit 2: DMA Channel Group Enable
Writing a 1 to this bit disables the DMA channel group, while writing a 0 to this bit enables the DMA channel group. Both channel groups are enabled following PCIRST#. Disabling channel group 4–7 also disables channel group 0–3, which is cascaded through channel 4.

Bits[1:0]: Reserved
Must be 0.

4.2.2 DCM—DMA CHANNEL MODE REGISTER

Register Name: DMA Channel Mode
Address Offset: Channels 0-3–0Bh
Channels 4-7–0D6h
Default Value: Bits[7:2] = 0,
Bits[1:0] = undefined
Attribute: Write Only
Size: 6 bits

Each channel has a 6-bit DMA Channel Mode Register. The Channel Mode Registers provide control over DMA Transfer type, transfer mode, address increment/decrement, and autoinitialization. Bits[1:0] select the appropriate Channel Mode Register and are not stored. Only bits[7:2] are stored in the register. This register is set to its default value upon

PCIRST# or Master Clear. Its default value is Verify transfer, Autoinitialize disable, Address increment, and Demand mode. Channel 4 defaults to cascade mode and cannot be programmed for any mode other than cascade mode.

Bits[7:6]: DMA Transfer Mode
Each DMA channel can be programmed in one of four different modes: single transfer, block transfer, demand transfer and cascade.

Bits	7	6	Transfer Mode
	0	0	Demand mode
	0	1	Single mode
	1	0	Block mode
	1	1	Cascade mode

Bit 5: Address Increment/Decrement Select
Bit 5 controls address increment/decrement during multi-byte DMA transfers. When bit 5 = 0, address increment is selected. When bit 5 = 1, address decrement is selected. Address increment is the default after a PCIRST# cycle or Master Clear command.

Bit 4: Autoinitialize Enable
When bit 4 = 1, the DMA restores the Base Page, Address, and Word count information to their respective current registers following a terminal count (TC). When bit 4 = 0, the autoinitialize feature is disabled and the DMA does not restore the above mentioned registers. A PCIRST# or Master Clear disables autoinitialization (sets bit 4 to 0).

Bits[3:2]: DMA Transfer Type
Verify, write and read transfer types are available. Verify transfer is the default transfer type upon PCIRST# or Master Clear. Write transfers move data from an I/O device to memory. Read transfers move data from memory to an I/O device. Verify transfers are pseudo transfers; addresses are generated as in a normal read or write transfer and the device responds to EOP etc. However, with Verify transfers, the ISA memory and I/O cycle lines are not driven. Bit combination 11 is illegal. When the channel is programmed for cascade ([7:6] = 11) the transfer type bits are irrelevant.

Bits	3	2	Transfer Mode
	0	0	Verify transfer
	0	1	Write transfer
	1	0	Read Transfer
	1	1	Illegal

Bits[1:0]: DMA Channel Select
Bits[1:0] select the DMA Channel Mode Register that will be written by bits[7:2].

Bits	1	0	Channel
	0	0	Channel 0 (4)
	0	1	Channel 1 (5)
	1	0	Channel 2 (6)
	1	1	Channel 3 (7)

4.2.3 DCEM—DMA CHANNEL EXTENDED MODE REGISTER

Address Offset: Channels 0-3—040Bh
Channels 4-7—04D6h
Default Value: Bits[1:0] = undefined,
Bits[3:2] = 00 for DMA1,
Bits[3:2] = 01 for DMA2,
Bits[7:4] = 0
Attribute: Write Only
Size: 6 bits

Each channel has a 6-bit Extended Mode Register. The register is used to program the DMA device data size, timing mode, EOP input/output selection, and Stop Register selection. Bits[1:0] select the appropriate Channel Extend Mode Register and are not stored. Only bits[7:2] are stored in the register. Four timing modes are available: ISA-compatible, "A", "B", and "F". Timings "A", "B", and "F" are extended timing modes and can only be run to main memory. DMA cycles to ISA expansion bus memory defaults to compatible timing if the channel is programmed in an extended timing mode.

The default bit values for each DMA group are selected upon PCIRST#. A Master Clear or any other programming sequence will not set the default register settings. The default programmed values for DMA1 channels 0-3 are 8-bit I/O count by bytes, compatible timing, and EOP output. The default values for DMA2 channels 4-7 are 16-bit I/O count by words with shifted address, compatible timing, and EOP output.

Bit 7: Reserved
Must be 0.

Bit 6: EOP Input/Output Selection
Bit 6 selects whether the EOP signal is to be used as an output during DMA transfers on this channel or an input. EOP is typically used as an output, as was available on the PC/AT. The input function was added to support data communication and other devices that would like to trigger an autoinitialize when a collision or some other event occurs. The direction of EOP is switched when DACK is changed (when a different channel is granted the bus). There may be some overlap of the SIO driving the EOP signal along with the DMA slave. However, during this overlap, both devices drive the signal to a low level (inactive). For example, assume channel 2 is about to go inactive (DACK negating) and channel 1 is about to go active. In addition, assume that channel 2 is programmed for "EOP OUT" and channel 1 is programmed for "EOP IN". When channel 2's DACK is negated and channel 1's DACK is asserted, the SIO may be driving EOP to a low value on behalf of channel 2. At the same time the device connected to channel 1 is driving EOP in to the SIO, also at an inactive level. This overlap only lasts until the SIO EOP output buffer is tri-stated, and does not effect the DMA operation. Upon PCIRST#, bit 6 is set to 0-EOP output selected.

Bits[5:4]: DMA Cycle Timing Mode
The SIO supports four DMA transfer timings: ISA-compatible, type "A", "B", and "F". Each timing and its corresponding code are described below. Upon PCIRST#, compatible timing is selected and the value of these bits is "00". The cycle timings noted below are for a SYSCLK (8.33 MHz, maximum SYSCLK frequency). DMA cycles to ISA expansion bus memory defaults to compatible timing if the channel is programmed in one of the performance timing modes (type "A", "B", or "F").

Bits[5:4] = 00: Compatible Timing
Compatible timing is provided for DMA slave devices, that, due to some design limitation, cannot support one of the faster timings. Compatible timing runs at 9 SYSCLKs (1080 nsec/single cycle) and 8 SYSCLKs (960 nsec/cycle) during the repeated portion of a BLOCK or DEMAND mode transfer.

Bits[5:4] = 01: Type "A" Timing
Type "A" timing is provided to allow shorter cycles to main memory (via the PCI Bus). Type "A" timing runs at 6 SYSCLKs (720 nsec/cycle) during the repeated portion of a BLOCK or DEMAND mode transfer. Type "A" timing varies from compatible timing primarily in shortening the memory operation to the minimum allowed main memory. The I/O portion of the cycle (data setup on write, I/O read access time) is the same as with compatible cycles. The actual active command time is shorter. However, it is expected that the DMA devices that provide the data access time or write data setup time should not require excess IOR# or IOW# command active time. Because of this, most ISA DMA devices should be able to use type "A" timing.

Bits[5:4] = 10: Type "B" Timing
Type "B" timing is provided for 8/16-bit ISA DMA devices that can accept faster I/O timing. Type "B" only works with fast main memory. Type "B" timing runs at 5 SYSCLKs (600 nsec/cycle) during the repeated portion of a BLOCK or DEMAND mode transfer. Type "B" timing requires faster DMA slave devices than compatible timing. In Type "B" timing the cycles are shortened so that the data setup time on I/O write cycles is shortened and the I/O read access time is required to be faster.

Bits[5:4] = 11: Type "F" Timing
Type "F" timing provides high performance DMA transfer capability. Type "F" timing runs at 3 SYSCLKs (360 nsec/single cycle) during the repeated portion of a BLOCK or DEMAND mode transfer, resulting in a maximum data transfer rate of 8.33 MBytes/second.

Bits[3:2]: Addressing Mode
The SIO supports both 8- and 16-bit DMA device data sizes. Three data size options are programmable with bits[3:2]. Both the 8-bit I/O, "count by bytes" mode and the 16-bit I/O, "count by words" (address shifted) mode are ISA compatible. The 16-bit I/O, "count by bytes" mode is offered as an extension of the ISA compatible modes. Bits[3:2] = 10 is reserved. Byte assembly/disassembly is performed by the ISA control unit. Each of the data transfer size modes is discussed below.

Bits[3:2] = 00: 8-bit I/O, "Count By Bytes" Mode
In 8-bit I/O, "count by bytes" mode, the Current Address Register can be programmed to any address. The Current Byte/Word Count Register is programmed with the "number of bytes minus 1" to transfer.

Bits[3:2] = 01: 16-bit I/O, "Count By Words" (Address Shifted) Mode
In "count by words" mode (address shifted), the Current Address Register can be programmed to any even address, but must be programmed with the address value shifted right by one bit. The Low Page and High Page Registers are not shifted during DMA transfers. Thus, the least significant bit of the Low Page register is ignored when the address is driven out onto the bus. The Current Byte/Word Count Register is programmed with the number of words minus 1 to be transferred.

Bits[3:2] = 10: Reserved

Bits[3:2] = 11: 16-Bit I/O, "Count By Bytes" Mode
In 16-bit "count by bytes" mode, the Current Address Register can be programmed to any byte address. For most DMA devices, however, it should be programmed only to even addresses. If the address is programmed to an odd address, the DMA controller does a partial word transfer during the first and last transfer, if necessary. The bus controller does the Byte/Word assembly necessary to write any size memory device. In this mode, the Current Address Register is incremented or decremented by two and the byte count is decremented by the number of bytes transferred during each bus cycle. The Current Byte/Word Count Register is programmed with the "number of bytes minus 1" to be transferred. This mode should only be programmed for 16-bit ISA DMA slaves.

Bits[1:0]: DMA Channel Select
Bits[1:0] select the particular channel that will have its DMA Channel Extend Mode Register programmed with bits[7:2].

Bits	1	0	Channel
	0	0	Channel 0 (4)
	0	1	Channel 1 (5)
	1	0	Channel 2 (6)
	1	1	Channel 3 (7)

4.2.4 DR—DMA REQUEST REGISTER

Address Offset:	Channels 0-3–09h Channels 4-7–0D2h
Default Value:	Bits[1:0] = undefined, Bits[7:2] = 0
Attribute:	Write Only
Size:	4 bits

Each channel has a request bit in one of the two 4-bit DMA Request Registers. The Request Register is used by software to initiate a DMA request. The DMA responds to the software request as though DREQ[x] is asserted. These requests are non-maskable and subject to prioritization by the priority encoder network. Each register bit is set to 1 or 0 separately under software control or is set to 0 upon generation of a TC. The entire register is set to 0 upon PCIRST# or a Master Clear. It is not affected upon a RSTDRV output. To program a bit, the software loads the proper form of the data word. Bits[1:0] determine which channel Request Register will be written. In order to make a software request, the channel must be in Block Mode. The Request Register status for DMA1 and DMA2 is output on bits[7:4] of a Status Register read to the appropriate port.

Bits[7:3]: Reserved
Must be 0.

Bit 2: DMA Channel Service Request
Writing a 0 to bit 2 resets the individual software DMA channel request bit. Writing a 1 to bit 2 sets the request bit. The request bit for each DMA channel is reset to 0 upon a PCIRST# or a Master Clear.

Bits[1:0]: DMA Channel Select
Bits[1:0] select the DMA channel mode register to program with bit 2.

Bits	1	0	Channel
	0	0	Channel 0
	0	1	Channel 1 (5)
	1	0	Channel 2 (6)
	1	1	Channel 3 (7)

4.2.5 MASK REGISTER—WRITE SINGLE MASK BIT

Address Offset: Channels 0-3–0Ah
Channels 4-7–0D4h
Default Value: Bits[1:0] = undefined, Bit 2 = 1, Bits[7:3] = 0
Attribute: Write Only
Size: 1 bit/channel

Each DMA channel has a mask bit that enables/disables an incoming DMA channel service request DREQ[x]. Two 4-bit registers store the current mask status for DMA1 and DMA2. Setting the mask bit disables the incoming DREQ[x] for that channel. Clearing the mask bit enables the incoming DREQ[x]. A channel's mask bit is automatically set when the Current Byte/Word Count register reaches terminal count (unless the channel is programmed for autoinitialization). Each mask bit may also be set or cleared under software control. The entire register is also set by a PCIRST# or a Master Clear. Setting the entire register disables all DMA requests until a clear mask register instruction allows them to occur. This instruction format is similar to the format used with the DMA Request Register.

Individually masking DMA channel 4 (DMA controller 2, channel 0) will automatically mask DMA channels [3:0], as this channel group is logically cascaded onto channel 4. Setting this mask bit disables the incoming DREQ's for channels [3:0].

Bits[7:3]: Reserved
Must be 0.

Bit 2: Channel Mask Select
When bit 2 is set to a 1, DREQ is disabled for the selected channel. When bit 2 is set to a 0, DREQ is enabled for the selected channel.

Bit[1:0]: DMA Channel Select
Bits[1:0] select the DMA Channel Mode Register to program with bit 2.

Bits	1	0	Channel
	0	0	Channel 0 (4)
	0	1	Channel 1 (5)
	1	0	Channel 2 (6)
	1	1	Channel 3 (7)

4.2.6 MASK REGISTER—WRITE ALL MASK BITS

Address Offset: Channels 0-3–0Fh
Channels 4-7–0DEh
Default Value: Bit[3:0] = 1, Bit[7:4] = 0
Attribute: Read/Write
Size: 4 bits

Writing to this register enables/disables incoming DREQ assertions. There are four mask bits per register, one for each channel. This permits all four channels to be simultaneously enabled/disabled instead of enabling/disabling each channel individually, as is the case with the Mask Register—Write Single Mask Bit.

Two 4-bit registers store the current mask status for DMA1 and DMA2. Unlike the Mask Register—Write Single Mask Bit, this register and includes a status read to check the current mask status of the selected DMA channel group. A channel's mask bit is automatically set to 1 when the Current Byte/Word Count Register reaches terminal count (unless the channel is programmed for autoinitialization). Bits[3:0] are set to 1 by a PCIRST# or a Master Clear. Setting bits[3:0] to 1 disables all DMA requests until a clear mask register instruction enables the requests.

Two important points should be taken into consideration when programming the mask registers. First, individually masking DMA channel 4 (DMA controller 2, channel 0) will automatically mask DMA channels [3:0], as this channel group is logically cascaded onto channel 4. Second, masking DMA controller 2 with a write to port 0DEh will also mask DREQ assertions from DMA controller 1 for the same reason. When DMA channel 4 is masked, so are DMA channels 0-3.

Bits[7:4]: Reserved
Must be 0.

Bits[3:0]: Channel Mask Bits
Setting the bit(s) to a 1 disables the corresponding DREQ(s). Setting the bit(s) to a 0 enables the corresponding DREQ(s). Bits[3:0] are set to 1 upon PCIRST# or Master Clear. When read, bits[3:0] indicate the DMA channel [3:0] ([7:4]) mask status.

Bit	Channel
0	0 (4)
1	1 (5)
2	2 (6)
3	3 (7)

NOTE:
Disabling channel 4 also disables channels 0-3 due to the cascade of DMA1 through channel 4 of DMA2.

4.2.7 DS—DMA STATUS REGISTER

Address Offset: Channels 0-3–08h
Channels 4-7–0D0h
Default Value: 00h
Attribute: Read Only
Size: 8 bits

Each DMA controller has a read-only DMA Status Register. This register indicates which channels have reached terminal count and which channels have a pending DMA request. Bits[3:0] are set every time the corresponding TC is reached by that channel. Bits[3:0] are set to 0 upon PCIRST# and on each status read. Bits[7:4] are set whenever their corresponding channel is requesting service.

Bits[7:4]: Channel Request Status
When a valid DMA request is pending for a channel (on its DREQ signal line), the corresponding bit is set to 1. When a DMA request is not pending for a particular channel, the corresponding bit is set to 0. The source of the DREQ may be hardware, a timed-out block transfer, or a software request. Note that channel 4 does not have DREQ or DACK lines, so the response for a read of DMA2 status for channel 4 is irrelevant.

Bit	Channel
4	0
5	1 (5)
6	2 (6)
7	3 (7)

Bits[3:0]: Channel Terminal Count Status
When a channel reaches terminal count (TC), its status bit is set to 1. If TC has not been reached, the status bit is set to 0. Note that channel 4 is programmed for cascade, and is not used for a DMA transfer. Therefore, the TC bit response for a status read on DMA2 for channel 4 is irrelevant.

Bit	Channel
0	0
1	1 (5)
2	2 (6)
3	3 (7)

4.2.8 DMA BASE AND CURRENT ADDRESS REGISTERS (8237 COMPATIBLE SEGMENT)

Address Offset: DMA Channel 0–000h
DMA Channel 1–002h
DMA Channel 2–004h
DMA Channel 3–006h
DMA Channel 4–0C0h
DMA Channel 5–0C4h
DMA Channel 6–0C8h
DMA Channel 7–0CCh
Default Value: All bits undefined
Attribute: Read/Write
Size: 16 bits per channel

Each channel has a 16-bit Current Address Register. This register contains the value of the 16 least significant bits of the full 32-bit address used during DMA transfers. The address is automatically incremented or decremented after each transfer and the intermediate values of the address are stored in the Current Address Register during the transfer. This register is written to or read from by the PCI Bus or ISA Bus master in successive 8-bit bytes. The programmer must issue the "Clear Byte Pointer Flip-Flop" command to reset the internal byte pointer and correctly align the write prior to programming the Current Address Register. After clearing the Byte Pointer Flip-Flop, the first write to the Current Address Register programs the low byte, bits[7:0], and the second write programs the high byte, bits[15:8]. This procedure also applies to read cycles. It may also be reinitialized by an Autoinitialize back to its original value. Autoinitialize takes place only after a TC or EOP.

Each channel has a Base Address Register located at the same port address as the corresponding Current Address Register. These registers store the original value of their associated Current Address Registers. During autoinitialize these values are used to restore the Current Address Registers to their original values. The Base Registers are written simultaneously with their corresponding Current Address Register in successive 8-bit bytes. The Base Registers are write-only.

In Scatter/gather mode, these registers store the lowest 16 bits of the current memory address. During a Scatter/gather transfer, the DMA will load a reserve buffer into the base memory address register.

Bits[15:0]: Base and Current Address [15:0]
These bits represent the 16 least significant address bits used during DMA transfers. Together with the DMA Low Page Register, they form the ISA-compatible 24-bit DMA address. As an extension of the ISA compatible functionality, the DMA High Page Register completes the 32-bit address needed when implementing SIO extensions such as DMA to the PCI Bus slaves that can take advantage of full 32-bit addressability. Upon PCIRST# or Master Clear, the value of these bits is 0000h.

4.2.9 DMA BASE AND CURRENT BYTE/WORD COUNT REGISTERS (8237 COMPATIBLE SEGMENT)

Address Offset:	DMA Channel 0–001h
	DMA Channel 1–003h
	DMA Channel 2–005h
	DMA Channel 3–007h
	DMA Channel 4–0C2h
	DMA Channel 5–0C6h
	DMA Channel 6–0CAh
	DMA Channel 7–0CEh
Default Value:	All bits undefined
Attribute:	Read/Write
Size:	16 bits per channel

Each channel has a 16-bit Current Byte/Word Count Register. This register determines the number of transfers to be performed. The actual number of transfers is one more than the number programmed in the Current Byte/Word Count Register (i.e., programming a count of 100 results in 101 transfers). The Byte/Word count is decremented after each transfer. The intermediate value of the Byte/Word count is stored in the register during the transfer. When the value in the register goes from zero to 0FFFFh, a TC is generated.

Following the end of a DMA service the register may also be re-initialized by an autoinitialization back to its original value. Autoinitialize can only occur when a TC occurs. If it is not autoinitialized, this register has a count of FFFFh after TC.

When the Extended Mode Register is programmed for, or defaulted to, transfers to/from an 8-bit I/O, the Byte/Word count indicates the number of bytes to be transferred.

When the Extended Mode Register is programmed for, or defaulted to, transfers to/from a 16-bit I/O, with shifted address, the Byte/Word count indicates the number of 16-bit words to be transferred.

When the Extended Mode Register is programmed for transfers to/from a 16-bit I/O, the Byte/Word Count indicates the number of bytes to be transferred. The number of bytes does not need to be a multiple of two or four in this case.

Each channel has a Base Byte/Word Count Register located at the same port address as the corresponding Current Byte/Word Count Register. These registers store the original value of their associated Current Byte/Word Count Registers. During Autoinitialize these values are used to restore the Current registers to their original values. The Base registers are written simultaneously with their corresponding Current register in successive 8-bit bytes. The Base registers cannot be read by any external agents.

In Scatter/gather mode, these registers store the 16 bits of the current Byte/Word count. During Scatter/gather transfer, the DMA will load a reserve buffer into the base Byte/Word Count register.

Bits[15:0]: Base and Current Byte/Word Count
These bits represent the 16 byte/word count bits used when counting down a DMA transfer. Upon PCIRST# or Master Clear, the value of these bits is 0000h.

4.2.10 DMA MEMORY BASE LOW PAGE AND CURRENT LOW PAGE REGISTERS

Register Name:	DMA Memory Current Low Page Register (Read/Write) DMA Memory Base Low Page Register (Write Only)
Address Offset:	DMA Channel 0–087h DMA Channel 1–083h DMA Channel 2–081h DMA Channel 3–082h DMA Channel 5–08Bh DMA Channel 6–089h DMA Channel 7–08Ah
Default Value:	All bits undefined
Size:	8 bits per channel

Each channel has an 8-bit Low Page Register. The DMA memory Low Page Register contains the eight second most-significant bits of the 32-bit address. The register works in conjunction with the DMA controller's High Page Register and Current Address Register to define the complete (32-bit) address for the DMA channel. This 8-bit register is read or written directly. It may also be re-initialized by an autoinitialize back to its original value. Autoinitialize takes place only after a TC or EOP.

Each channel has a Base Low Page Address Register located at the same port address as the corresponding Current Low Page Register. These registers store the original value of their associated Current Low Page Registers. During autoinitialization, these values are used to restore the Current Low Page Registers to their original values. The 8-bit Base Low Page Registers are written simultaneously with their corresponding Current Low Page Register by the microprocessor. The Base Low Page registers are write only.

During Scatter/gather, these registers store the 8 bits from the third byte of the current memory address. During a Scater-Gather transfer, the DMA will load a reserve buffer into the base memory address register.

Bits[7:0]: DMA Low Page and Base Low Page [23:16]
These bits represent the eight second most significant address bits when forming the full 32-bit address for a DMA transfer. Upon PCIRST# or Master Clear, the value of these bits is 00h.

4.2.11 DMA MEMORY BASE HIGH PAGE AND CURRENT HIGH PAGE REGISTERS

Register Name:	DMA Memory Current High Page Register (Read/Write) DMA Memory Base High Page Register (Write Only)
Address Offset:	DMA Channel 0–0487h DMA Channel 1–0483h DMA Channel 2–0481h DMA Channel 3–0482h DMA Channel 5–048Bh DMA Channel 6–0489h DMA Channel 7–048Ah
Default Value:	All bits undefined
Size:	8 bits per channel

Each channel has an 8-bit Current High Page Register. The DMA memory Current High Page Register contains the eight most significant bits of the 32-bit address. The register works in conjunction with the DMA controller's Current Low Page Register and Current Address Register to define the complete (32-bit) address for the DMA channels and corresponds to the Current Address Register for each channel. This 8-bit register is read or written directly. It may also be autoinitialized back to its original value. Autoinitialize takes place only after a TC or EOP.

This register is set to 0 during the programming of both the Current Low Page Register and the Current Address Register. Thus, if this register is not programmed after the other address and Low Page Registers are programmed, then its value is 00h. In this case, the DMA channel operates the same as an 82C37 (from an addressing standpoint). This is the address compatibility mode.

If the high 8 bits of the address are programmed after the other addresses, then the channel modifies its operation to increment (or decrement) the entire 32-bit address. This is unlike the 82C37 "Page" register in the original PCs which could only increment to a 64 KByte boundary for 8-bit channels or 128 KByte boundary for 16-bit channels. This is extended address mode. In this mode, the ISA Bus controller generates the signals MEMR# and MEMW# only for addresses below 16 MBytes.

Each channel has a Base High Page Register located at the same port address as the corresponding Current High Page Register. These registers store the original value of their associated Current High Page Registers. During autoinitialize, these values are used to restore the Current High Page Registers to their original values. The 8-bit Base High Page Registers are written simultaneously with their corresponding Current High Page Register. The Base High Page Registers are write only.

During Scatter/Gather, these registers store the 8 bits from the highest byte of the current memory address. During a Scatter/Gather transfer, the DMA will load a reserve buffer into the base memory address register.

Bits[7:0]: DMA High Page and Base High Page [31:24]
These bits represent the eight most-significant address bits when forming the full 32-bit address for a DMA transfer. Upon PCIRST# or Master Clear, the value of these bits is 00h.

4.2.12 DMA CLEAR BYTE POINTER REGISTER

Address Offset:	Channels 0–3–00Ch
	Channels 4–7–0D8h
Default Value:	All bits undefined
Attribute:	Write Only
Size:	8 bits

Writing to this register executes the clear byte pointer command. This command is executed prior to writing or reading new address or word count information to the DMA. This command initializes the byte pointer flip-flop to a known state so that subsequent accesses to register contents will address upper and lower bytes in the correct sequence.

The clear byte pointer command clears the internal latch used to address the upper or lower byte of the 16-bit Address and Word Count Registers. The latch is also cleared at power on by PCIRST# and by the Master Clear command. The Host CPU may read or write a 16-bit DMA controller register by performing two consecutive accesses to the I/O port. The Clear Byte Pointer command precedes the first access. The first I/O write to a register port loads the least significant byte, and the second access automatically accesses the most significant byte.

When DMA registers are being read or written, two Byte Pointer flip-flops are used. One flip-flop is for Channels 0–3 and one for Channels 4–7. Both of these act independently. There are separate software commands for clearing each of them (0Ch for Channels 0–3, 0D8h for Channels 4–7).

Bits[7:0]: Clear Byte Pointer
No specific pattern. Command enabled with a write to the I/O port address.

4.2.13 DMC—DMA MASTER CLEAR REGISTER

Address Offset:	Channel 0–3–00Dh
	Channel 4–7–0DAh
Default Value:	All bits undefined
Attribute:	Write Only
Size:	8 bit

This software instruction has the same effect as the hardware Reset. The Command, Status, Request, and Internal First/Last Flip-Flop registers are cleared and the Mask Register is set. The DMA controller enters the idle cycle. There are two independent Master Clear Commands; 0Dh acts on Channels 0–3, and 0DAh acts on Channels 4–7.

Bits[7:0]: Master Clear
No specific pattern. Command enabled with a write to the I/O port address.

4.2.14 DCM—DMA CLEAR MASK REGISTER

Address Offset:	Channel 0–3–00Eh
	Channel 4–7–0DCh
Default Value:	All bits undefined
Attribute:	Write Only
Size:	8 bit

This command clears the mask bits of all four channels, enabling them to accept DMA requests. I/O port 0Eh is used for Channels 0–3 and I/O port 0DCh is used for Channels 4–7.

Bits[7:0]: Clear Mask Register
No specific pattern. Command enabled with a write to the I/O port address.

4.2.15 SCATTER/GATHER COMMAND REGISTER

Register Name: DMA Scatter Gather Command

Address Offset: Channels 0 default address—0410h
Channels 1 default address—0411h
Channels 2 default address—0412h
Channels 3 default address—0413h
Channels 5 default address—0415h
Channels 6 default address—0416h
Channels 7 default address—0417h

Default Value: 00h
Attribute: Write Only, Relocatable
Size: 8 bits

The Scatter/Gather Command Register controls operation of the descriptor table aspect of scatter/gather transfers. This register can be used to start and stop a scatter/gather transfer. The register can also be used to select between IRQ13 and EOP to be asserted following a terminal count. The current scatter/gather transfer status can be read in the scatter/gather channel's corresponding Scatter/Gather Status Register. After a PCIRST# or Master Clear, IRQ13 is disabled and EOP is enabled.

Bit 7: IRQ13/EOP Select
Bit 7, if enabled via bit 6 of this register, selects whether EOP or IRQ13 is asserted at termination caused by a last buffer expiring. The last buffer can be either the last buffer in the list or the last buffer loaded in the DMA while it is suspended. If bit 7 = 1 (and bit 6 = 1), EOP is asserted when the last buffer is completed. If bit 7 = 0 (and bit 6 = 1), IRQ13 is asserted when the last buffer is completed.

EOP can be used to alert an expansion bus I/O device that a scatter/gather termination condition was reached. The I/O device, in turn, can assert its own interrupt request line to invoke a dedicated interrupt handling routine. IRQ13 should be used when the CPU needs to be notified directly.

Following PCIRST#, or Master Clear, the value stored for this bit is "1", and EOP is selected. Bit-6 must be set to a "1" to enable this bit during a S/G Command register write. When bit 6 is a "0" during the write, bit 7 will not have any effect on the current EOP/IRQ13 selection.

Bit 6: IRQ13/EOP Programming Enable
Enabling IRQ13/EOP programming allows initialization or modification of the S/G termination handling bits. When bit 6 = 0, bit 7 does not affect the state of IRQ13 or EOP assertion. When bit 6 = 1, bit 7 determines the termination handling following a terminal count.

Bits[5:2]: Reserved
Must be 0.

Bits[1:0]: Scatter/Gather Commands
This 2-bit field is used to start and stop scatter/gather.

Bits[1:0] = 00: No S/G operation
No S/G command operation is performed. Bits[7:6] may still be used to program IRQ13/EOP selection.

Bits[1:0] = 01: Start S/G Command
The Start command initiates the scatter/gather process. Immediately after the start command is issued (setting bits[1:0] to 01), a request is issued to fetch the initial buffer from the descriptor table to fill the Base Register set in preparation for performing a transfer. The buffer prefetch request has the same priority with respect to other channels as the DREQ it is associated with. Within the channel, DREQ is higher in priority than a prefetch request.

The Start command assumes the Base and Current registers are both empty and will request a prefetch automatically. Note that this command also sets the Scatter/Gather Status Register to S/G Active, Base Empty, Current Empty, not Terminated, and Next Null Indicator to 0. The EOP/IRQ13 bit will still reflect the last value programmed.

Bits[1:0] = 10: Stop S/G Command
The Stop command halts a Scatter/gather transfer immediately. When a Stop command is given, the Terminate bit in the S/G Status register and the DMA channel mask bit are both set.

Bits[1:0] = 11: Reserved

4.2.16 SCATTER/GATHER STATUS REGISTER

Address Offset:	Channels 0 default address—0418h Channels 1 default address—0419h Channels 2 default address—041Ah Channels 3 default address—041Bh Channels 5 default address—041Dh Channels 6 default address—041Eh Channels 7 default address—041Fh
Default Value:	00h
Attribute:	Read Only, Relocatable
Size:	8-bits

The Scatter/Gather Status Register contains information on the scatter/gather transfer status. This register provides dynamic status information on S/G transfer activity, the current and base buffer state, S/G transfer termination, and the End of the List indicator.

An Active bit is set to "1" after the S/G Start command is issued. The Active bit will be "0" before the initial Start command, following a terminal count, and after a S/G Stop command is issued. The Current Register and Base Register Status bits indicate whether the corresponding register has a buffer loaded. It is possible for the Base Register Status to be set while the Current Register Status is cleared. When the Current Register transfer is complete, the Base Register will not be moved into the Current Register until the start of the next data transfer. Thus, the Current Register State is empty (cleared), while the Base Register State is full (set). The Terminate bit is set active after a Stop command, after TC for the last buffer in the list, and both Base and Current Registers have expired. The EOP and IRQ13 bits indicate which end of process indicator will be used to alert the system of an S/G process termination. The EOL status bit is set if the DMA controller has loaded the last buffer of the Link List. Following PCIRST#, or Master Clear, each bit in this register is reset to "0".

Bit 7: Next Link Null Indicator
If the next scatter/gather descriptor fetched from memory during a fetch operation has the EOL value set to 1, the current value of the Next Link Register is not overwritten. Instead, bit 7 of the channel's Scatter/Gather Status Register is set to a 1. If the fetch returns a EOL value set to 0, this bit is set to 0. This status bit is written after every fetch operation. Following PCIRST#, or Master Clear, this bit is set to 0. This bit is also cleared by an S/G Start Command write to the Scatter/Gather Command Register.

Bit 6: Reserved

Bit 5: Issue IRQ13/EOP on Last Buffer
When bit 5 = 0, EOP was either defaulted to at reset or selected through the Scatter/Gather Command Register as the S/G process termination indicator. EOP is issued when a terminal count occurs or following the Stop Command. When bit 5 = 1, an IRQ13 is issued to alert the CPU of this same status.

Bit 4: Reserved

Bit 3: Scatter/Gather Base Register Status
When bit 3 = 0, the Base Register is empty. When bit 3 = 1, the Base Register has a buffer link loaded. Note that the Base Register State may be set while the Current Register state is cleared. This condition occurs when the Current Register expires following a transfer. The Base Register will not be moved into the Current Register until the start of the next DMA transfer.

Bit 2: Scatter/Gather Current Register Status
When bit 2 = 0, the Current Register is empty. When bit 2 = 1, the Current Register has a buffer link loaded and is considered full. Following PCIRST#, bit 2 is set to 0.

Bit 1: Reserved

Bit 0: Scatter/Gather Active
The Scatter/gather Active bit indicates the current S/G transfer status. Bit 0 is set to a 1 after an S/G Start Command is issued. Bit 0 is set to 0 before the Start Command is issued. Bit 0 is 0 after terminal count on the last buffer on the channel is reached. Bit 0 is also 0 after an S/G Stop Command has been issued. Following a PCIRST# or Master Clear, this bit is 0.

4.2.17 SCATTER/GATHER DESCRIPTOR TABLE POINTER REGISTER

Address Offset: Channel 0 default address—0420h–0423h
Channel 1 default address—0424h–0424h
Channel 2 default address—0428h–042Bh
Channel 3 default address—042Ch–042Ch
Channel 5 default address—0434h–0437h
Channel 6 default address—0438h–043Bh
Channel 7 default address—043Ch–043Fh
Default Value: All bits undefined
Attribute: Read/Write, Relocatable
Size: 32 bits

The Scatter/Gather Descriptor Table Pointer Register contains the 32-bit pointer address to the first scatter/gather descriptor entry in the descriptor table in memory. Before the start of a S/G transfer, this register should be programmed to point to the first descriptor in the Scatter/Gather Descriptor Table. Following a S/G Start command, the SIO reads the first SGD entry. Subsequently, at the end of the each buffer block transfer, the contents of the SGD Table pointer registers are incremented by 8 until the end of the SGD Table is reached.

The Scatter/Gather Descriptor Table Pointer Registers can be programmed with a single 32-bit PCI write.

Following a prefetch to the address pointed to by the channel's Scatter/Gather Descriptor Table Pointer Register, the new memory address is loaded into the Base Address Register, the new Byte Count is loaded into the Base Byte Count Register, and the newly fetched next scatter/gather descriptor replaces the current next scatter/gather value.

The end of the Scatter/Gather Descriptor Table is indicated by an End of Table field having a MSB equal to 1. When this value is read during a scatter/gather descriptor fetch, the current scatter/gather descriptor value is not replaced. Instead, bit 7 of the channel's Status Register is set to a 1, when the EOL is read from memory.

Bits[31:0]:
The Scatter/Gather Descriptor Table Pointer Register contains a 32-bit pointer address to the main memory location where the software maintains the Scatter Gather Descriptors for the linked-list buffers. Bits[31:0] correspond to A[31:0] on the PCI.

4.2.18 SCATTER/GATHER INTERRUPT STATUS REGISTER

Address Offset: 040Ah
Default Value: 00h
Attribute: Read Only, Relocatable
Size: 8 bits

The Scatter/Gather Interrupt Status Register is a read only register and is used to indicate the source (channel) of a DMA Scatter/Gather interrupt on IRQ13. The DMA controller drives IRQ13 active after reaching terminal count during a Scatter/Gather transfer. It does not drive IRQ13 active during the initial programming sequence that loads the Base registers.

Bit 7: Channel 7 Interrupt Status
When this bit is set to a 1, Channel 7 has an interrupt due to a Scatter/Gather Transfer; otherwise this bit is set to a 0.

Bit 6: Channel 6 Interrupt Status
When this bit is set to a 1, Channel 6 has an interrupt due to a Scatter/Gather Transfer; otherwise this bit is set to a 0.

Bit 5: Channel 5 Interrupt Status
When this bit is set to a 1, Channel 5 has an interrupt due to a Scatter/Gather Transfer; otherwise this bit is set to a 0.

Bit 4: Reserved
Read as 0.

Bit 3: Channel 3 Interrupt Status
When this bit is set to a 1, Channel 3 has an interrupt due to a Scatter/Gather Transfer; otherwise this bit is set to a 0.

Bit 2: Channel 2 Interrupt Status
When this bit is set to a 1, Channel 2 has an interrupt due to a Scatter/Gather Transfer; otherwise this bit is set to a 0.

Bit 1: Channel 1 Interrupt Status
When this bit is set to a 1, Channel 1 has an interrupt due to a Scatter/Gather Transfer; otherwise this bit is set to a 0.

Bit 0: Channel 0 Interrupt Status
When this bit is set to a 1, Channel 0 has an interrupt due to a Scatter/Gather Transfer; otherwise this bit is set to a 0.

4.3 Timer Register Description

The SIO contains three counters that are equivalent to those found in the 82C54 Programmable Interval Timer. The Timer registers control these counters and can be accessed from either the ISA Bus via ISA I/O space or the PCI Bus via PCI I/O space.

This section describes the counter/timer registers on the SIO. The counter/timer operations are further described in Section 5.7, Timer Unit.

4.3.1 TCW—TIMER CONTROL WORD REGISTER

Address Offset: 043h
Default Value: All bits undefined
Attribute: Write Only
Size: 8 bits

The Timer Control Word Register specifies the counter selection, the operating mode, the counter byte programming order and size of the count value, and whether the counter counts down in a 16-bit or binary-coded decimal (BCD) format. After writing the control word, a new count can be written at any time. The new value will take effect according to the programmed mode.

There are six programmable counting modes. Typically, the SIO Timer Counters 0 and 2 are programmed for Mode 3, the Square Wave Mode, while Counter 1 is programmed in Mode 2, the Rate Generator Mode.

Two special commands are selected through the Timer Control Word Register. The Read Back Command (see Section 4.3.1.1) is selected when bits[7:6] are both 1 and the Counter Latch Command (see Section 4.3.1.2) is selected when bits[5:4] are both 0. When either of these two commands are selected, the meaning of the other bits in the register changes.

Bits 4 and 5 are also used to select the count register programming mode. The read/write selection chosen with the control word indicates the programming sequence that must follow when initializing the specified counter. If a counter is programmed to read/write two byte counts, note that a program must not transfer control between writing the first and second byte to another routine that also writes into that same counter. Otherwise, the counter will be loaded with an incorrect count. The count must always be completely loaded with both bytes.

Bits 6 and 7 are also used to select the counter for the control word being written.

Following PCIRST#, the control words for each register are undefined. Each timer must be programmed to bring it into a known state. However, each counter OUT signal is set to 0 following PCIRST#. The SPKR output, interrupt controller input IRQ0 (internal), bit 5 of port 061h, and the internally generated refresh request are each set to 0 following PCIRST#.

Bits[7:6]: Counter Select
The Counter Selection bits select the counter the control word acts upon as shown below. The Read Back Command is selected when bits[7:6] are both 1.

Bit 7	Bit 6	Function
0	0	Counter 0 select
0	1	Counter 1 select
1	0	Counter 2 select
1	1	Read Back Command (see Section 4.3.1.1)

Bits[5:4]: Read/Write Select
Bits[5:4] are the read/write control bits. The Counter Latch Command is selected when bits[5:4] are both 0. The read/write options include r/w least significant byte, r/w most significant byte, or r/w the LSB and then the MSB. The actual counter programming is done through the counter I/O port (040h, 041h, and 042h for counters 0, 1, and 2, respectively).

Bit 5	Bit 4	Function
0	0	Counter Latch Command (see Section 4.3.1.2)
0	1	R/W Least Significant Byte (LSB)
1	0	R/W Most Significant Byte (MSB)
1	1	R/W LSB then MSB

Bits[3:1]: Counter Mode Selection
Bits[3:1] select one of six possible modes of operation for the counter as shown below.

Bit 3	2	1	Mode	Function
0	0	0	0	Out signal on end of count (= 0)
0	0	1	1	Hardware retriggerable one-shot
X	1	0	2	Rate generator (divide by n counter)
X	1	1	3	Square wave output
1	0	0	4	Software triggered strobe
1	0	1	5	Hardware triggered strobe

Bit 0: Binary/BCD Countdown Select
When bit 0 = 0, a binary countdown is used. The largest possible binary count is 2^{16}. When bit 0 = 1, a binary coded decimal (BCD) count is used. The largest BCD count allowed is 10^4.

4.3.1.1 Read Back Command

The Read Back Command is used to determine the count value, programmed mode, and current states of the OUT pin and Null count flag of the selected counter or counters. The Read Back Command is written to the Timer Control Word Register which latches the current states of the above mentioned variables. The value of the counter and its status may then be read by I/O access to the counter address.

Status and/or count may be latched on one, two, or all three of the counters by selecting the counter during the register write. The count latched remains latched until read, regardless of further latch commands. The count must be read before newer latch commands latch a new count. The status latched by the Read Back Command also remains latched until after a read to the counter's I/O port by reading the Counter Access Ports Register. Thus, the status and count are unlatched only after a counter read of the Timer Status Byte Format Register, the Counter Access Ports Register, or the Timer Status Byte Register and Counter Access Ports Register in succession.

Both count and status of the selected counter(s) may be latched simultaneously by setting both bit 5 and bit 4 to 0. This is functionally the same as issuing two consecutive, separate Read Back Commands. As mentioned above, if multiple count and/or status Read Back Commands are issued to the same counter(s) without any intervening reads, all but the first are ignored.

If both count and status of a counter are latched, the first read operation from that counter returns the latched status, regardless of which was latched first. The next one or two reads (depending on whether the counter is programmed for one or two byte counts) returns the latched count. Subsequent reads return an unlatched count.

NOTE:
The Timer Counter Register bit definitions are different during the Read Back Command than for a normal Timer Counter Register write.

Bits[7:6]: Read Back Command
When bits[7:6] are both 1, the Read Back Command is selected during a write to the Timer Control Word Register. As noted above, the normal meanings (mode, countdown, r/w select) of the bits in the control register at I/O address 043h change when the Read Back Command is selected. Following the Read Back Command, I/O reads from the selected counter's I/O addresses produce the current latch status, the current latched count, or both if bits 4 and 5 are both 0.

Bit 5: Latch Count of Selected Counters
When bit 5 = 1, the current count value of the selected counters will be latched. When bit 4 = 0, the status will not be latched.

Bit 4: Latch Status of Selected Counters
When bit 4 = 1, the status of the selected counters will be latched. When bit 4 = 0, the status will not be latched. The status byte format is described in Section 4.3.2, Interval Timer Status Byte Format Register.

Bit 3: Counter 2 Select
When bit 3 = 1, Counter 2 is selected for the latch command selected with bits 4 and 5. When bit 3 = 0, status and/or count will not be latched.

Bit 2: Counter 1 Select
When bit 2 = 1, Counter 1 is selected for the latch command selected with bits 4 and 5. When bit 2 = 0, status and/or count will not be latched.

Bit 1: Counter 0 Select
When bit 1 = 1, Counter 0 is selected for the latch command selected with bits 4 and 5. When bit 1 = 0, status and/or count will not be latched.

Bit 0: Reserved
Must be 0.

4.3.1.2 Counter Latch Command

The Counter Latch Command latches the current count value at the time the command is received. This command is used to insure that the count read from the counter is accurate (particularly when reading a two-byte count). The count value is then read from each counter's count register (via the Counter Access Ports Register). One, two or all three counters may be latched with one Counter Latch Command.

If a Counter is latched once and then, some time later, latched again before the count is read, the second Counter Latch Command is ignored. The count read will be the count at the time the first Counter Latch Command was issued.

The count must be read according to the programmed format. Specifically, if the Counter is programmed for two byte counts, two bytes must be read. The two bytes do not have to be read one right after the other (read, write, or programming operations for other counters may be inserted between the reads).

NOTES:

1. If a counter is programmed to read/write two-byte counts, a program must not transfer control between reading the first and second byte to another routine that also reads from that same counter. Otherwise, an incorrect count will be read. Finish reading the latched two-byte count before transferring control to another routine.
2. The Timer Counter Register bit definitions are different during the Counter Latch Command than for a normal Timer Counter Register write.

Bits[7:6]: Counter Selection
Bits 6 and 7 are used to select the counter for latching.

Bit 7	6	Function
0	0	latch counter 0 select
0	1	latch counter 1 select
1	0	latch counter 2 select
1	1	Read Back Command select

Bits[5:4]: Counter Latch Command
When bits[5:4] are both 0, the Counter Latch Command is selected during a write to the Timer Control Word Register. As noted above, the normal meanings (mode, countdown, r/w select) of the bits in the control register at I/O address 043h change when the Counter Latch Command is selected. Following the Counter Latch Command, I/O reads from the selected counter's I/O addresses produce the current latched count.

Bits[3:0]: Reserved
Must be 0.

4.3.2 INTERVAL TIMER STATUS BYTE FORMAT REGISTER

Address Offset:	Counter 0–040h Counter 1–041h Counter 2–042h
Default Value:	Bits[6:0] = X, Bit 7 = 0
Attribute:	Read Only
Size:	8 bits per counter

Each counter's status byte can be read following an Interval Timer Read Back Command. The Read Back Command is programmed through the Timer Control Word Register. If latch status is chosen (bit 4 = 0, Read Back Command) as a read back option for a given counter, the next read from the counter's Counter Access Ports Register returns the status byte. The status byte returns the countdown type, either BCD or binary; the counter operational mode; the read/write selection status; the Null count, also referred to as the count register status; and the current state of the counter OUT pin.

Bit 7: Counter OUT Pin State
When this bit is a 1, the OUT pin of the counter is also a 1. When this bit is a 0, the OUT pin of the counter is also a 0.

Bit 6: Count Register Status
Null Count, also referred to as the Count Status Register, indicates when the last count written to the Count Register (CR) has been loaded into the Counting Element (CE). The exact time this happens depends on the counter mode, but until the count is loaded into the counting element (CE), it can't be read from the counter. If the count is latched or read before the load time, the count value returned will not reflect the new count written to the register. When bit 6 = 0, the count has been transferred from CR to CE and is available for reading. When bit 6 = 1, the Null count condition exists. The count has not been transferred from CR to CE and is not yet available for reading.

Bits[5:4]: Read/Write Selection Status
Bits[5:4] reflect the read/write selection made through bits[5:4] of the control register. The binary codes returned during the status read match the codes used to program the counter read/write selection.

Bit	5	4	Function
	0	0	Counter Latch Command
	0	1	R/W Least Significant Byte (LSB)
	1	0	R/W Most Significant Byte (MSB)
	1	1	R/W LSB then MSB

Bits[3:1]: Mode Selection Status
Bits[3:1] return the counter mode programming. The binary code returned matches the code used to program the counter mode, as listed under the bit function above.

Bit	3	2	1	Mode Selected
	0	0	0	0
	0	0	1	1
	X	1	0	2
	X	1	1	3
	1	0	0	4
	1	0	1	5

Bit 0: Countdown Type Status
Bit reflects the current countdown type; either 0 for binary countdown or a 1 for binary coded decimal (BCD) countdown.

4.3.3 COUNTER ACCESS PORTS REGISTER

Address Offset:	Counter 0, System Timer-040h Counter 1, Refresh Request-041h Counter 2, Speaker Tone-042h
Default Value:	All bits undefined
Attribute:	Read/Write
Size:	8 bits per counter

Each of these I/O ports is used for writing count values to the Count Registers; reading the current count value from the counter by either an I/O read, after a counter-latch command, or after a Read Back Command; and reading the status byte following a Read Back Command.

Bits[7:0]: Counter Port Bit[x]
Each counter I/O port address is used to program the 16-bit Count Register. The order of programming, either LSB only, MSB only, or LSB then MSB, is defined with the Interval Counter Control Register at I/O port address 043h. The counter I/O port is also used to read the current count from the Count Register, and return the status of the counter programming following a Read Back Command.

4.3.4 BIOS TIMER REGISTER

Register Location:	Default = 78h–7Bh (Dword aligned)
Default Value:	0000xxxxh
Attribute:	Read/Write, Programmable
Size:	32 bit

A write to the BIOS Timer initiates a counting sequence. The timer can be initiated by writing either a 16-bit data portion or the entire 32-bit register (the upper 16 bits are don't cares). Bits[15:0] can be written with the initial count value to start the timer or read to check the current count value. It is the programmer's responsiblity to ensure that all 16 bits are written at the same time. After data is written into BIOS timer, the timer will start decrementing until it reaches zero. It will "freeze" at zero until the new count value is written.

The BIOS Timer consists of a single 32-bit register mapped in the I/O space on the location determined by the value written into the BIOS Timer Base Address Register. Bit 0 of the BIOS Timer Base Address Register enables/disables accesses to the BIOS Timer and must be 1 to enable access to the BIOS Timer Register. When the BIOS Timer is enabled, PCI accesses to the BIOS Timer Register do not flow through to the ISA Bus. If the BIOS Timer is disabled, accesses to the addresses assigned to the BIOS Timer Register flow through to the ISA bus. Note, however, that the counter continues to count normally.

Bits[31:16]: Reserved
Read as 0.

Bits[15:0]:
Timer count value.

4.4 Interrupt Controller Register Description

The SIO contains an ISA compatible interrupt controller that incorporates the functionality of two 82C59 interrupt controllers. The interrupt registers control the operation of the interrupt controller and can be accessed from the PCI Bus via PCI I/O space. In addition, some of the registers can be accessed from the ISA Bus via ISA I/O space. The bus access for each register is listed in Table 4.

4.4.1 ICW1—INITIALIZATION COMMAND WORD 1 REGISTER

Register Location:	INT CNTRL-1–020h INT CNTRL-2–0A0h
Default Value:	All bits undefined
Attribute:	Write Only
Size:	8 bits per controller

A write to Initialization Command Word 1 starts the interrupt controller initialization sequence. Addresses 020h and 0A0h are referred to as the base addresses of CNTRL-1 and CNTRL-2, respectively.

An I/O write to the CNTRL-1 or CNTRL-2 base address with bit 4 equal to 1 is interpreted as ICW1. For SIO-based ISA systems, three I/O writes to "base address + 1" must follow the ICW1. The first write to "base address + 1" performs ICW2, the second write performs ICW3, and the third write performs ICW4.

ICW1 starts the initialization sequence during which the following automatically occur:

a. The edge sense circuit is reset. This means that following initialization, an interrupt request (IRQ) input must make a low-to-high transition to generate an interrupt.

b. The Interrupt Mask register is cleared.

c. IRQ7 input is assigned priority 7.

d. The slave mode address is set to 7.

e. Special Mask Mode is cleared and Status Read is set to IRR.

f. If IC4 was set to 0, then all functions selected by ICW4 are set to 0. However, ICW4 must be programmed in the SIO implementation of this interrupt controller, and IC4 must be set to a 1.

ICW1 has three significant functions within the SIO interrupt controller configuration. ICW4 is needed, so bit 0 must be programmed to a 1. There are two interrupt controllers in the system, so bit 1, SNGL, must be programmed to a 0 on both CNTRL-1 and CNTRL-2, to indicate a cascade configuration. LTIM, the interrupt controller IRQ edge/level detection control bit, defines the IRQ sensing mode for each controller. When bit 3 is a 0, each IRQ line on the selected controller is programmed for edge-triggered mode. This mode is signified by a low-to-high transition on an IRQ input line. When bit 3 is a 1, the controller is programmed in level-triggered mode, where a high level on an IRQ input indicates the presence of an interrupt request. LTIM is global for each controller. The incoming IRQs are either all edge-triggered or all level-triggered. Bit D4 must be a 1 when programming ICW1. OCW2 and OCW3 are also addressed at the same port as ICW1. This bit indicates that ICW1, and not OCW2 or OCW3, will be programmed during the write to this port.

Bit 2, ADI, and bits[7:5], A7-A5, are specific to an MSC-85 implementation. These bits are not used by the SIO interrupt controllers. Bits[7:5,2] should each be initialized to 0.

In the 82378ZB, bit 3, the LTIM bit, is not used by the interrupt controller and is always read as a 1.

Bits[7:5]: ICW/OCW Select
A7–A5 are MCS-85 implementation specific bits. They are not needed by the SIO. These bits should be 000 when programming the SIO.

Bit 4: ICW/OCW Select
Bit 4 must be a 1 to select ICW1. After the fixed initialization sequence to ICW1, ICW2, ICW3, and ICW4, the controller base address is used to write to OCW2 and OCW3. Bit 4 is a 0 on writes to these registers. A 1 on this bit at any time will force the interrupt controller to interpret the write as an ICW1. The controller will then expect to see ICW2, ICW3, and ICW4.

Bit 3: LTIM (Edge/Level Bank Select)
Ignored for the SIO.

Bit 2: ADI
Ignored for the SIO.

Bit 1: SNGL (Single or Cascade)
SNGL must be programmed to a 0 to indicate that two interrupt controllers are operating in cascade mode on the SIO.

Bit 0: IC4 (ICW4 Write Required)
This bit must be set to a 1. IC4 indicates that ICW4 needs to be programmed. The SIO requires that ICW4 be programmed to indicate that the controllers are operating in an 80x86 type system.

4.4.2 ICW2—INITIALIZATION COMMAND WORD 2 REGISTER

Address Offset:	INT CNTRL-1–021h INT CNTRL-2–0A1h
Default Value:	All bits undefined
Attribute:	Write Only
Size:	8 bits per controller

ICW2 is used to initialize the interrupt controller with the five most significant bits of the interrupt vector address. The value programmed for bits[7:3] is used by the Host CPU to define the base address in the interrupt vector table for the interrupt routines associated with each IRQ on the controller. Typical ISA ICW2 values are 04h for CNTRL-1 and 70h for CNTRL-2.

Bits[7:3]: Interrupt Vector Base Address
Bits[7:3] define the base address in the interrupt vector table for the interrupt routines associated with each interrupt request level input. For CNTRL-1, a typical value is 00001, and for CNTRL-2, 10000.

The interrupt controller combines a binary code representing the interrupt level to receive service with this base address to form the interrupt vector that is driven out onto the bus. For example, the complete interrupt vector for IRQ[0] (CNTRL-1), would be 0000 1000b (CNTRL-1 [7:3] = 00001b and 000b representing IRQ[0]). This vector is used by the CPU to point to the address information that defines the start of the interrupt routine.

Bits[2:0]: Interrupt Request Level
When writing ICW2, these bits should all be 0. During an interrupt acknowledge cycle, these bits are programmed by the interrupt controller with the interrupt code representing the interrupt level to be serviced. This interrupt code is combined with bits[7:3] to form the complete interrupt vector driven onto the data bus during the second INTA# cycle. Section 5.0, Detailed Function Description, outlines each of these codes. The code is a simple three bit binary code: 000 represents IRQ0 (IRQ8), 001 IRQ1 (IRQ9), 010 IRQ2 (IRQ10), and so on until 111 IRQ7 (IRQ15).

4.4.3 ICW3—INITIALIZATION COMMAND WORD 3 REGISTER

Register Name:	Initialization Command Word 3 (Controller 1-Master Unit)
Address Offset:	INT CNTRL-1–021h
Default Value:	All bits undefined
Attribute:	Write Only
Size:	8 bits

The meaning of ICW3 differs between CNTRL-1 and CNTRL-2. On CNTRL-1, the master controller, ICW3 indicates which CNTRL-1 IRQ line physically connects the INT output of CNTRL-2 to CNTRL-1. ICW3 must be programmed to 04h, indicating the cascade of the CNTRL-2 INT output to the IRQ[2] input of CNTRL-1.

An interrupt request on IRQ2 causes CNTRL-1 to enable CNTRL-2 to present the interrupt vector address during the second interrupt acknowledge cycle.

Bits[7:3]:
These bits must be programmed to zero.

Bit 2: Cascaded Interrupt Controller IRQ Connection
Bit 2 must always be programmed to a 1. This bit indicates that CNTRL-2, the slave controller, is cascaded on interrupt request line two (IRQ[2]). When an interrupt request is asserted to CNTRL-2, the IRQ goes through the priority resolver. After the slave

controller priority resolution is finished, the INT output of CNTRL-2 is asserted. However, this INT assertion does not go directly to the CPU. Instead, the INT assertion cascades into IRQ[2] on CNTRL-1. IRQ[2] must go through the priority resolution process on CNTRL-1. If it wins the priority resolution on CNTRL-1 and the CNTRL-1 INT signal is asserted to the CPU, the returning interrupt acknowledge cycle is really destined for CNTRL-2. The interrupt was originally requested at CNTRL-2, so the interrupt acknowledge is destined for CNTRL-2, and not a response for IRQ[2] on CNTRL-1.

When an interrupt request from IRQ[2] wins the priority arbitration, in reality an interrupt from CNTRL-2 has won the arbitration. Because bit 2 of ICW3 on the master is set to 1, the master knows which identification code to broadcast on the internal cascade lines, alerting the slave controller that it is responsible for driving the interrupt vector during the second INTA# pulse.

Bits[1:0]:
These bits must be programmed to zero.

4.4.4 ICW3—INITIALIZATION COMMAND WORD 3 REGISTER

Register Name:	Initialization Command Word 3 (Controller 2-Slave Unit)
Address Offset:	INT CNTRL-2–0A1h
Default Value:	All bits undefined
Attribute:	Write Only
Size:	8 bits

On CNTRL-2 (the slave controller), ICW3 is the slave identification code broadcast by CNTRL-1 from the trailing edge of the first INTA# pulse to the trailing edge of the second INTA# pulse. CNTRL-2 compares the value programmed in ICW3 with the incoming identification code. The code is broadcast over three SIO internal cascade lines. ICW3 must be programmed to 02h for CNTRL-2. When 010b is broadcast by CNTRL-1 during the INTA# sequence, CNTRL-2 assumes responsibility for broadcasting the interrupt vector during the second interrupt acknowledge cycle.

As an illustration, consider an interrupt request on IRQ[2] of CNTRL-1. By definition, a request on IRQ[2] must have been asserted by CNTRL-2. If IRQ[2] wins the priority resolution on CNTRL-1, the interrupt acknowledge cycle returned by the CPU following the interrupt is destined for CNTRL-2, not CNTRL-1. CNTRL-1 will see the INTA# signal, and knowing that the actual destination is CNTRL-2, will broadcast a slave identification code across the internal cascade lines. CNTRL-2 will compare this incoming value with the 010b stored in ICW3. Following a positive decode of the incoming message from CNTRL-1, CNTRL-2 will drive the appropriate interrupt vector onto the data bus during the second interrupt acknowledge cycle.

Bits[7:3]: Reserved
Must be 0.

Bits[2:0]: Slave Identification Code
The Slave Identification code must be programmed to 010b during the initialization sequence. The code stored in ICW3 is compared to the incoming slave identification code broadcast by the master controller during interrupt acknowledge cycles.

4.4.5 ICW4—INITIALIZATION COMMAND WORD 4 REGISTER

Address Offset:	INT CNTRL-1–021h INT CNTRL-2–0A1h
Default Value:	01h
Attribute:	Write Only
Size:	8 bits

Both SIO interrupt controllers must have ICW4 programmed as part of their initialization sequence. Minimally, the microprocessor mode bit, bit 0, must be set to a 1 to indicate to the controller that it is operating in an 80x86 based system. Failure to program this bit will result in improper controller operation during interrupt acknowledge cycles. Additionally, the Automatic End of Interrupt (AEOI) may be selected, as well as the Special Fully Nested Mode (SFNM) of operation.

The default programming for ICW4 is 01h, which selects 80x86 mode, normal EOI, buffered mode, and special fully nested mode disabled.

Bits 2 and 3 must be programmed to 0 for the SIO interrupt controller to function correctly.

Both bit 1, AEOI, and bit 4, SFNM, can be programmed if the system developer chooses to invoke either mode.

Bits[7:5]: Reserved
Must be 0.

Bit 4: SFNM (Special Fully Nested Mode)
Bit 4, SFNM, should normally be disabled by writing a 0 to this bit. If SFNM = 1, the special fully nested mode is programmed.

Bit 3: BUF (Buffered Mode)
Bit 3, BUF, must be programmed to 0 for the SIO. This is non-buffered mode. As illustrated above under bit functionality, different programming options are offered for bits 2 and 3. However, within the SIO interrupt unit, bits 2 and 3 must always be programmed to 00b.

Bit 2: Master/Slave in Buffered Mode
This bit is not used by the SIO interrupt unit. Bit 2 should always be programmed to 0.

Bit 1: AEOI (Automatic End of Interrupt)
This bit should normally be programmed to 0. This is the normal end of interrupt. If this bit is 1, the automatic end of interrupt mode is programmed.

Bit 0: Microprocessor Mode
The Microprocessor Mode bit must be programmed to 1 to indicate that the interrupt controller is operating in an 80x86-based system. Never program this bit to 0.

4.4.6 OCW1—OPERATIONAL CONTROL WORD 1 REGISTER

Address Offset:	INT CNTRL-1-021h INT CNTRL-2-0A1h
Default Value:	00h
Attribute:	Read/Write
Size:	8 bits

OCW1 sets and clears the mask bits in the Interrupt Mask Register (IMR). Each interrupt request line may be selectively masked or unmasked any time after initialization. A single byte is written to this register. Each bit position in the byte represents the same-numbered channel: bit 0 = IRQ[0], bit 1 = IRQ[1] and so on. Setting the bit to a 1 sets the mask, and clearing the bit to a 0 clears the mask. Note that masking IRQ[2] on CNTRL-1 will also mask all of controller 2's interrupt requests (IRQ8–IRQ15). Reading OCW1 returns the controller's mask register status.

The IMR stores the bits which mask the interrupt lines to be masked. The IMR operates on the IRR. Masking of a higher priority input will not affect the interrupt request lines of lower priority.

Unlike status reads of the ISR and IRR, for reading the IMR, no OCW3 is needed. The output data bus will contain the IMR whenever I/O read is active and the I/O port address is 021h or 0A1h (OCW1).

All writes to OCW1 must occur following the ICW1–ICW4 initialization sequence, since the same I/O ports are used for OCW1, ICW2, ICW3 and ICW4.

Bits[7:0]: Interrupt Request Mask (Mask [7:0])
When a 1 is written to any bit in this register, the corresponding IRQ[x] line is masked. For example, if bit 4 is set to a 1, then IRQ[4] will be masked. Interrupt requests on IRQ[4] will not set channel 4's interrupt request register (IRR) bit as long is the channel is masked.

When a 0 is written to any bit in this register, the corresponding IRQ[x] mask bit is cleared, and interrupt requests will again be accepted by the controller.

NOTE:
Masking IRQ[2] on CNTRL-1 will also mask the interrupt requests from CNTRL-2, which is physically cascaded to IRQ[2].

4.4.7 OCW2—OPERATIONAL CONTROL WORD 2 REGISTER

Address Offset: INT CNTRL-1-020h
INT CNTRL-2-0A0h
Default Value: Bit[4:0] = undefined,
Bit[7:5] = 001
Attribute: Write Only
Size: 8 bits

OCW2 controls both the Rotate Mode and the End of Interrupt Mode, and combinations of the two. The three high order bits in an OCW2 write represent the encoded command. The three low order bits are used to select individual interrupt channels during three of the seven commands. The three low order bits (labeled L2, L1 and L0) are used when bit 6 is set to a 1 during the command.

Following a PCIRST# and ICW initialization, the controller enters the fully nested mode of operation. Non-specific EOI without rotation is the default. Both rotation mode and specific EOI mode are disabled following initialization.

Bits[7:5]: Rotate and EOI Codes
R, SL, EOI-These three bits control the Rotate and End of Interrupt modes and combinations of the two. A chart of these combinations is listed above under the bit definition.

Bits 7	6	5	Function
0	0	1	Non-Specific EOI Command
0	1	1	Specific EOI Command
1	0	1	Rotate on Non-Specific EOI Command
1	0	0	Rotate in Auto EOI Mode (Set)
0	0	0	Rotate in Auto EOI Mode (Clear)
1	1	1	*Rotate on Specific EOI Command
1	1	0	*Set Priority Command
0	1	0	No Operation

NOTE:
* L0–L2 Are Used

Bits[4:3]: OCW2 Select
When selecting OCW2, bits 3 and 4 must both be 0. If bit 4 is a 1, the interrupt controller interprets the write to this port as an ICW1. Therefore, always ensure that these bits are both 0 when writing an OCW2.

Bits[2:0]: Interrupt Level Select (L2, L1, L0)
L2, L1, and L0 determine the interrupt level acted upon when the SL bit is active. A simple binary code, outlined above, selects the channel for the command to act upon. When the SL bit is inactive, these bits do not have a defined function; programming L2, L1 and L0 to 0 is sufficient in this case.

Bit 2	1	0	Interrupt Level
0	0	0	IRQ 0(8)
0	0	1	IRQ 1(9)
0	1	0	IRQ 2(10)
0	1	1	IRQ 3(11)
1	0	0	IRQ 4(12)
1	0	1	IRQ 5(13)
1	1	0	IRQ 6(14)
1	1	1	IRQ 7(15)

4.4.8 OCW3—OPERATIONAL CONTROL WORD 3 REGISTER

Address Offset: INT CNTRL-1-020h
INT CNTRL-2-0A0h
Default Value: Bit[6,0] = 0,
Bit[7,4:2] = undefined,
Bit[5,1] = 1
Attribute: Read/Write
Size: 8 bits

OCW3 serves three important functions: Enable Special Mask Mode, Poll Mode control, and IRR/ISR register read control.

First, OCW3 is used to set or reset the Special Mask Mode (SMM). The Special Mask Mode can be used by an interrupt service routine to dynamically alter the system priority structure while the routine is executing, through selective enabling/disabling of the other channel's mask bits.

Second, the Poll Mode is enabled when a write to OCW3 is issued with bit 2 equal to 1. The next I/O read to the interrupt controller is treated like an interrupt acknowledge; a binary code representing the highest priority level interrupt request is released onto the bus.

Third, OCW3 provides control for reading the In-Service Register (ISR) and the Interrupt Request Register (IRR). Either the ISR or IRR is selected for reading with a write to OCW3. Bits 0 and 1 carry the encoded command to select either register. The next I/O read to the OCW3 port address will return the register status specified during the previous write. The register specified for a status read is retained by the interrupt controller. Therefore, a write to OCW3 prior to every status read command is unnecessary, provided the status read desired is from the register selected with the last OCW3 write.

Bit 7: Reserved
Must be 0.

Bit 6: SMM (Special Mask Mode)
If ESMM = 1 and SMM = 1 the interrupt controller enters Special Mask Mode. If ESMM = 1 and SMM = 0, the interrupt controller is in normal mask mode. When ESMM = 0, SMM has no effect.

Bit 5: ESMM (Enable Special Mask Mode)
When ESMM = 1, the SMM bit is enabled to set or reset the Special Mask Mode. When ESMM = 0, the SMM bit becomes a "don't care".

Bits[4:3]: OCW3 Select
When selecting OCW3, bit 3 must be a 1 and bit 4 must be 0. If bit 4 = 1, the interrupt controller interprets the write to this port as an ICW1. Therefore, always ensure that bits[4:3] = 01 when writing an OCW3.

Bit 2: Poll Mode Command
When bit 2 = 0, the Poll command is not issued. When bit 2 = 1, the next I/O read to the interrupt controller is treated as an interrupt acknowledge cycle. An encoded byte is driven onto the data bus, representing the highest priority level requesting service.

Bits[1:0]: Register Read Command
Bits[1:0] provide control for reading the In-Service Register (ISR) and the Interrupt Request Register (IRR). When bit 1 = 0, bit 0 will not affect the register read selection. When bit 1 = 1, bit 0 selects the register status returned following an OCW3 read. If bit 0 = 0, the IRR will be read. If bit 0 = 1, the ISR will be read. Following ICW initialization, the default OCW3 port address read will be "read IRR". To retain the current selection (read ISR or read IRR), always write a 0 to bit 1 when programming this register. The selected register can be read repeatedly without reprogramming OCW3. To select a new status register, OCW3 must be reprogrammed prior to attempting the read.

Bit	1	0	Function
	0	0	No Action
	0	1	No Action
	1	0	Read IRQ Register
	1	1	Read IS Register

4.5 Control Registers

This section contains NMI registers, a real-time clock register, Port 92 Register, and the Digital Output Register.

4.5.1 NMISC—NMI STATUS AND CONTROL REGISTER

Address Offset: 061h
Default Value: 00h
Attribute: Read/Write
Size: 8 bits

This register is used to check the status of different system components, control the output of the speaker counter (Counter 2), and gate the counter output that drives the SPKR signal.

Bits 4, 5, 6, and 7 are read-only. When writing to this port, these bits must be written as 0's. Bit 6 returns the IOCHK# NMI status. This input signal comes from the ISA Bus. It is used for parity errors on memory cards plugged into the bus, and for other high priority interrupts. The current status of bit 3 enables or disables this NMI source. Bit 5 is the current state of the OUT pin of interval Timer 1, Counter 2. Bit 4 toggles from 1-0 or from 0-1 after every Refresh cycle. Following PCIRST#, bits 4 and 6 are both 0. Bit 5 is undetermined until Counter 2 is properly programmed. Bit 7 returns the PCI System Error status (SERR#). If 0, bit 7 indicates that SERR# was not pulsed active by a PCI agent. If 1, bit 7 indicates that SERR# was pulsed active by a PCI agent and that an NMI will be issued to the Host CPU. This NMI can be disabled with bit 2 of this register.

Bits 0–3 are both read and write. Bit 0 is the GATE input signal for Timer 1, Counter 2. The GATE input is used to disable counting in Counter 2. The Counter 2 output is ANDed with bit 1 to form the SPKR output signal. Bit 1 gates the Counter 2 OUT value. When bit 1 is disabled, the SPKR signal is disabled; when bit 1 is enabled, the SPKR output follows the value at the OUT pin of Counter 2. The Counter 2 OUT pin status can be checked by reading port 061h and checking bit 5. Bit 2 is used to enable the System Error (SERR#) signal. Bit 3 enables or disables the incoming IOCHK# NMI signal from the expansion bus. Each of these bits is reset to 0 following PCIRST#.

Bit 7: SERR# Status
Bit 7 is set if a system board agent (PCI devices or main memory) detects a system board error and pulses the PCI SERR# line. This interrupt is enabled by setting bit 2 to 0. To reset the interrupt, set bit 2 to 0 and then set it to 1. This bit is read-only. When writing to port 061h, bit 6 must be a 0.

Bit 6: IOCHK# NMI Source Status
Bit 6 is set if an expansion board asserts IOCHK# on the ISA/SIO bus. This interrupt is enabled by setting bit 3 to 0. To reset the interrupt, set bit 3 to 0 and then set it to 1. This bit is read-only. When writing to port 061h, bit 6 must be a 0.

Bit 5: Timer Counter 2 OUT Status
The Counter 2 OUT signal state is reflected in bit 5. The value on this bit following a read is the current state of the Counter 2 OUT signal. Counter 2 must be programmed following a PCIRST# for this bit to have a determinate value. Bit 5 is read-only. When writing to port 061h, bit 5 must be a 0.

Bit 4: Refresh Cycle Toggle
The Refresh Cycle Toggle signal toggles from either 0 to 1 or 1 to 0 following every refresh cycle. This read-only bit is a 0 following PCIRST#. When writing to port 061h, bit 4 must be a 0.

Bit 3: IOCHK# NMI Enable
When bit 3 = 1, IOCHK# NMI's are disabled and cleared. When bit 3 = 0, IOCHK# NMI's are enabled. Following PCIRST#, bit 3 is reset to 0.

Bit 2: PCI SERR# Enable
When bit 2 = 1, the PCI System Error (SERR#) is disabled and cleared. When bit 2 = 0, SERR# is enabled. Following PCIRST#, bit 2 is a 0.

Bit 1: Speaker Data Enable
Speaker Data Enable is ANDed with the Counter 2 OUT signal to drive the SPKR output signal. When bit 1 = 0, the result of the AND is always 0 and the SPKR output is always 0. When bit 1 = 1, the SPKR output is equivalent to the Counter 2 OUT signal value. Following PCIRST#, bit 1 is a 0.

Bit 0: Timer Counter 2 Enable
When bit 0 = 0, Counter 2 counting is disabled. Counting is enabled when bit 0 = 1. This bit controls the GATE input to Counter 2. Following PCIRST#, the value of this bit is 0.

4.5.2 NMI ENABLE AND REAL-TIME CLOCK ADDRESS REGISTER

Address Offset: 070h
Default Value: Bit[6:0] = undefined, Bit 7 = 1
Attribute: Write Only
Size: 8 bits

The Mask register for the NMI interrupt is at I/O address 070h shown below. The most significant bit enables or disables all NMI sources including IOCHK# and the NMI Port. Write an 80h to port 70h to mask the NMI signal. This port is shared with the real-time clock. The real-time-clock uses the lower six bits of this port to address memory locations. Writing to port 70h sets both the enable/disable bit and the memory address pointer. Do not modify the contents of this register without considering the effects on the state of the other bits. Reads and writes to this register address flow through to the ISA Bus.

Bit 7: NMI Enable
Setting bit 7 to a 1 disables all NMI sources. Setting the bit to a 0 enables the NMI interrupt. Following PCIRST#, this bit is a 1.

Bits[6:0]: Real Time Clock Address
Used by the Real Time Clock on the Base I/O component to address memory locations. Not used for NMI enabling/disabling.

4.5.3 PORT 92 REGISTER

Address Offset: 92h
Default Value: 24h
Attribute: Read/Write
Size: 8 bits

This register is used to support the alternate reset (ALT__RST#) and alternate A20 (ALT__A20) functions. This register is only accessible if bit 6 in the Utility Bus Chip Select B Register is set to a 1. Reads and writes to this register location flow through to the ISA Bus.

Bits[7:6]: Reserved
Returns 00 when read.

Bit 5: Reserved
Returns a 1 when read.

Bit 4: Reserved
Returns a 0 when read.

Bit 3: Reserved
Returns a 0 when read.

Bit 2: Reserved
Returns a 1 when read.

Bit 1: ALT__A20 Signal Control
Writing a 0 to this bit causes the ALT__A20 signal to be driven low. Writing a 1 to this bit causes the ALT__A20 signal to be driven high.

Bit 0: Alternate System Reset
This read/write bit provides an alternate system reset function. This function provides an alternate means to reset the system CPU to effect a mode switch from Protected Virtual Address Mode to the Real Address Mode. This provides a faster means of reset than is provided by the Keyboard controller. This bit is set to a 0 by a system reset. Writing a 1 to this bit will cause the ALT__RST# signal to pulse active (low) for approximately 4 SYSCLK's. Before another ALT__RST# pulse can be generated, this bit must be written back to a 0.

4.5.4 DIGITAL OUTPUT REGISTER

Address Offset: 03F2h (Primary), 0372h (Secondary)
Default Value: Bit[7:4,2:0] = undefined, Bit 3 = 0
Attribute: Write only
Size: 8 bits

This register is used to prevent UBUSOE# from responding to DACK2# during a DMA read access to a floppy controller on the ISA Bus. If a second floppy (residing on the ISA Bus) is using DACK2# in conjunction with a floppy on the utility bus, this prevents the floppy on the utility bus and the utility bus transceiver from responding to an access targeted for the floppy on the ISA Bus. This register is also located in the floppy controller device. Reads and writes to this register location flow through to the ISA Bus.

Bits[7:4]: Not Used
These bits exist in the floppy controller.

Bit 3: DMA Enable
When this bit is a 1, the assertion of DACK# will result in UBUSOE# being asserted. If this bit is 0, DACK2# has no effect on UBUSOE#. This port bit also exists on the floppy controller. This bit defaults to disable (0).

Bits[2:0]: Not Used
These bits exist in the floppy controller.

4.5.5 RESET UBUS IRQ12 REGISTER

Address Offset: 60h
Default Value: N/A
Attribute: Read only
Size: 8 bits

This address location (60h) is used to clear the mouse interrupt function to the CPU. Reads to this address are monitored by the SIO. When the mouse interrupt function is enabled (bit 4 of the ISA Clock Divisor Register is 1), the mouse interrupt function is provided on the IRQ12/M input signal. In this mode, a mouse interrupt generates an interrupt through IRQ13 to the Host CPU. A read of 60h releases IRQ12. If bit 4 = 0 in the ISA Clock Divisor Register, a read of address 60h has no effect on IRQ12/M. Reads and writes to this register flow through to the ISA Bus. For additional information, see the IRQ12/M description in Section 3.0, Signal Description.

Bits[7:0]: Reset IRQ12
No specific pattern. A read of address 60h executes the command.

4.5.6 COPROCESSOR ERROR REGISTER

Address Offset: F0h
Default Value: N/A
Attribute: Write only
Size: 8 bits

This address location (F0h) is used when the SIO is programmed for coprocessor error reporting (bit 5 of the ISA Clock Divisor Register is 1). Writes to this address are monitored by the SIO. In this mode, the SIO generates an interrupt (INT) to the CPU when it receives an error signal (FERR# asserted) from the CPU's coprocessor. Writing address F0h, when FERR# is asserted, causes the SIO to assert IGNNE# and negate IRQ13. IGNNE# remains asserted until FERR# is negated. If FERR# is not asserted, writing to address F0h does not effect IGNNE#. Reads and writes to this register flow through to the ISA Bus. For additional information, see the IGNNE# description in Section 3.0, Signal Description.

Bits[7:0]: Reset IRQ12
No specific pattern. A write to address F0h executes the command.

4.5.7 ELCR—EDGE/LEVEL CONTROL REGISTER

Address Offset: INT CNTRL-1–04D0h
INT CNTRL-2–04D1h
Default Value: 00h
Attribute: Read/Write
Size: 8 bits

The Edge/Level Control Register is used to set the interrupts to be triggered by either the signal edge or the logic level. INT0, INT1, INT2, INT8, INT13 must be set to edge sensitive. After a reset, all the INT signals are set to edge sensitive. Figure 4 shows which bit numbers represent the various INT signals.

Each IRQ to which a PCI interrupt is steered into (see the PIRQ Route Control Register) must have its interrupt set to level sensitive.

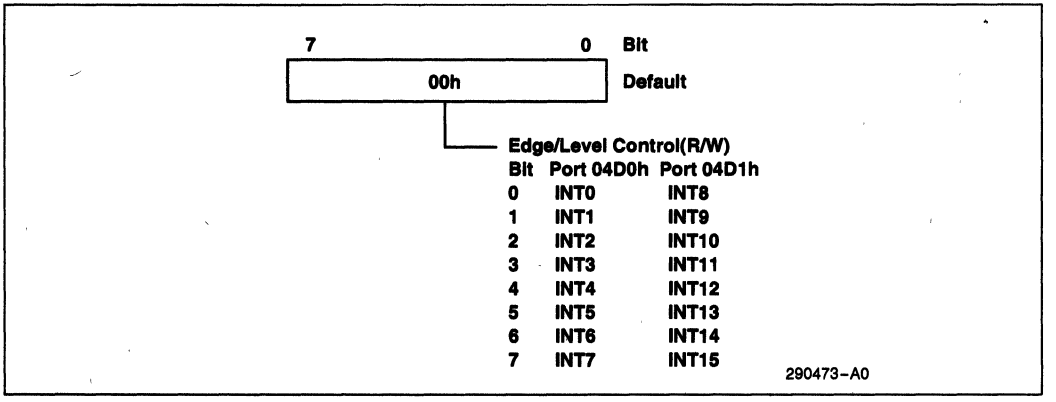


Figure 4. Edge/Level Select Register

Register Location: 04D0h-INT CNTRL-1
04D1h-INT CNTRL-2

04D0h-INT CNTRL-1 Register

Bit[7:0]: Edge/Level Select
These bits select if the interrupts are triggered by either the signal edge or the logic level. A 0 bit represents an edge sensitive interrupt, and a 1 is for level sensitive. The following bits MUST be set to 0:

Port 04D0h (INT-CNTRL-1)

0–INT0	0	Reserved. Read as zero.
1–INT1	0	Reserved. Read as zero.
2–INT2	0	Reserved. Read as zero.
3–INT3	x	
4–INT4	x	
5–INT5	x	
6–INT6	x	
7–INT7	x	

x = selectable to either a 0 or a 1,
0 = edge sensitive, 1 = level sensitive

After reset, this register is set to 00h.

04D1h-INT CNTRL-2 Register

Bit[7:0]: Edge/Level Select
These bits select if the interrupts are triggered by either the signal edge or the logic level. A 0 bit represents an edge sensitive interrupt, and a 1 is for level sensitive. The following bits MUST be set to 0:

Port 04D1h (INT-CNTRL-2)

0–INT8	0	Reserved. Read as zero.
1–INT9	x	
2–INT10	x	
3–INT11	x	
4–INT12	x	
5–INT13	0	Reserved. Read as zero.
6–INT14	x	
7–INT15	x	

x = selectable to either a 0 or a 1,
0 = edge sensitive, 1 = level sensitive

After reset, this register is set to 00h.

4.6. Power Management Registers

This section contains the Power Management Registers located in non-configuration space.

4.6.1 APMC—ADVANCED POWER MANAGEMENT CONTROL PORT

Address Offset: 0B2h
Default Value: 00h
Attribute: Read/Write
Size: 8 bits

Bits[7:0]: APMC[7:0]
APM Control Port. Readable/writeable at system I/O address 0B2h. Used to pass an APM command between the OS and the SMI handler. Writes to this port not only store data in the APMC register, but also generate an SMI when the SAPMCEN bit is set. Reads to this port will not generate an SMI. If CSTPCLKEN is set, a read from the APMC will cause STPCLK# to be asserted.

4.6.2 APMS—ADVANCED POWER MANAGEMENT STATUS PORT

Address Offset: 0B3h
Default Value: 00h
Attribute: Read/Write
Size: 8 bits

Bits[7:0]: APMS[7:0]
Readable/writeable at system address 0B3h. Used to pass data between the OS and the SMI handler.

5.0 DETAILED FUNCTIONAL DESCRIPTION

5.1 PCI Interface

5.1.1 PCI COMMAND SET

Bus commands indicate to the slave the type of transaction the master is requesting. Bus Commands are encoded on the C/BE[3:0]# lines during the address phase of a PCI cycle.

5.1.2 PCI BUS TRANSFER BASICS

Details of PCI Bus operations can be found in the *Peripheral Component Interconnect (PCI) Specification*. Only details of the PCI Bus unique to the SIO are included in this data sheet.

Table 7. PCI Commands

C/BE[3:0]#	Command Type As Slave	Supported As Slave	Supported As Master
0000	Interrupt Acknowledge	Yes	No
0001	Special Cycle(4)	No/Yes	No
0010	I/O Read	Yes	No
0011	I/O Write	Yes	No
0100	Reserved(3)	No	No
0101	Reserved(3)	No	No
0110	Memory Read	Yes	Yes
0111	Memory Write	Yes	Yes
1000	Reserved(3)	No	No
1001	Reserved(3)	No	No
1010	Configuration Read	Yes	No
1011	Configuration Write	Yes	No
1100	Memory Read Multiple	No(2)	No
1101	Reserved(3)	No	No
1110	Memory Read Line	No(2)	No
1111	Memory Write and Invalidate	No(1)	No

NOTES:
1. Treated as Memory Write.
2. Treated as Memory Read.
3. Reserved Cycles are considered invalid by the SIO and are to be competely ignored. All internal address decoding is ignored and DEVSEL# is never to be asserted.
4. The 82378 responds to a Stop Grant Special Cycle.

5.1.2.1 PCI Addressing

PCI address decoding uses the AD[31:0] signals. AD[31:2] are always used for address decoding while the information contained in the two low order bits AD[1:0] varies for memory, I/O, and configuration cycles.

For I/O cycles, AD[31:0] are decoded to provide a byte address. AD[1:0] are used for generation of DEVSEL# only and indicate the least significant valid byte involved in the transfer. For example, if only BE0# is asserted, AD[1:0] are 00. If only BE3# is asserted, then AD[1:0] are 11. If BE3# and BE2# are asserted, AD[1:0] are 10. If all BEx#'s are asserted, then AD[1:0] are 00. The byte enables determine which byte lanes contain valid data. The SIO requires that PCI accesses to byte-wide internal registers must assert only one byte enable.

When the SIO is the target of any PCI transaction in which BE[3:0]# = 1111, the SIO terminates the cycle normally by asserting TRDY#. No data is written into the SIO during write cycles and the data driven by the SIO during read cycles is indeterminate.

For memory cycles, accesses are decoded as Dword accesses. This means that AD[1:0] are ignored for decoding memory cycles. The byte enables determine which byte lanes contain valid data. When the SIO is a PCI master, it drives 00 on AD[1:0] for all memory cycles.

For configuration cycles, DEVSEL# is a function of IDSEL and AD[1:0]. DEVSEL# is selected during a configuration cycle only if IDSEL is active and both AD[1:0] = 00. The cycle is ignored by the SIO if either AD1 or AD0 is non-zero. Configuration registers are selected as Dwords using AD[7:2]. The byte enables determine which byte lanes contain valid data.

5.1.2.2 DEVSEL# Generation

As a PCI slave, the SIO asserts the DEVSEL# signal to indicate it is the slave of the PCI transaction. DEVSEL# is asserted when the SIO positively or subtractively decodes the PCI transaction. The SIO asserts DEVSEL# (claim the transaction) before it issues any other slave response, i.e., TRDY#, STOP#, etc. After the SIO asserts DEVSEL#, it does not negate DEVSEL# until the same edge that the master uses to negate the final IRDY#.

It is expected that most (perhaps all) PCI target devices will be able to complete a decode and assert DEVSEL# within 1 or 2 clocks of FRAME#. Since the SIO subtractively decodes all unclaimed PCI cycles (except configuration cycles), it provides a configuration option to pull in (by 1 or 2 clocks) the edge when the SIO samples DEVSEL#. This allows faster access to the expansion bus. Use of such an option is limited by the slowest positive decode agent on the bus. This is described in more detail in Section 5.5.1.4, Subtractively Decoded Cycles to ISA.

As a PCI master, the SIO waits for 5 PCICLKs after the assertion of FRAME# for a slave to assert DEVSEL#. If the SIO does not receive DEVSEL# in this time, it will master-abort the cycle. See Section 5.1.3.1, SIO as MasterMaster-Initiated Termination, for further details.

5.1.2.3 Basic PCI Read Cycles (I/O and Memory)

As a PCI master, the SIO only performs memory read transfers (i.e. I/O read transfers are not supported). When reading data from PCI memory, the SIO requests a maximum of 8 bytes via a two data phase burst read cycle to fill its internal 8 byte line buffer. If the line buffer is programmed for single transaction mode, fewer bytes are requested (refer to Section 5.6.1, DMA/ISA Master Line Buffer). Read cycles from PCI memory are generated on behalf of ISA masters and DMA devices.

As a PCI slave, the SIO responds to both I/O read and memory read transfers. For multiple read transactions, the SIO always target-terminates after the first data read transaction by asserting STOP# and TRDY# at the end of the first data phase. For single read transactions, the SIO finishes the cycle in a normal fashion, by asserting TRDY# without asserting STOP#.

5.1.2.4 Basic PCI Write Cycles (I/O and Memory)

As a PCI master, the SIO generates a PCI memory write cycle when it decodes an ISA-originated/PCI-bound memory write cycle. I/O write cycles are never initiated by the SIO. When writing data to PCI memory, the SIO writes a maximum of 4 bytes via a single data transaction write cycle. If the SIO's internal ISA master/DMA line buffer is programmed for single transaction mode, fewer bytes will be generated (refer to Section 5.6.1, DMA/ISA Master Line Buffer). In either case, only one data transaction will be performed. Cycles to PCI memory are generated on behalf of ISA masters, DMA devices, and the SIO when the SIO needs to flush the ISA master/DMA line buffer.

As a PCI master, the SIO drives the AD0 and AD1 signals low during the address phase of the cycle. This is done to indicate to the slave that the address will increment during the transfer. If there is no response on the PCI Bus, the SIO will master-abort due to the DEVSEL# time out.

As a PCI slave, the SIO will respond to both I/O write and memory write transfers. For multiple write transactions, the SIO will always target-terminate after the first data write transaction by asserting STOP# and TRDY# at the end of the first data phase. For single write transactions, the SIO will finish the cycle normally by asserting TRDY# without asserting STOP#.

5.1.2.5 Configuration Cycles

The configuration read or write command defined by the bus control signals C/BE[3:0]# is used to configure the SIO. During the address phase of the configuration read or write command, the SIO will sample its IDSEL (ID select). If IDSEL is active and AD[1:0] are both zero, the SIO generates DEVSEL#. Otherwise, the cycle is ignored by the SIO. During the configuration cycle address phase, bits AD[7:2] and C/BE[3:0]# are used to select particular bytes within a configuration register. Note that IDSEL is normally a "don't care" except during the address phase of a transaction.

NOTE:
An unclaimed configuration cycle is never forwarded to the ISA Bus.

5.1.2.6 Interrupt Acknowledge Cycle

The interrupt acknowledge command is a single byte read implicitly addressed to the SIO's interrupt controller. The address bits are logical "don't cares" during the address phase and the byte enables will indicate to the SIO an 8-bit interrupt vector is to be returned on AD[7:0]. The SIO converts this single cycle transfer into two cycles that the internal 8259 pair can respond to (see Section 5.8, Interrupt Controller). The SIO will hold the PCI Bus in wait states until the 8 bit interrupt vector is returned.

SIO responses to an interrupt acknowledge cycle can be disabled by setting bit 5 in the PCI Control Register to a 0. However, if disabled, the SIO will still respond to accesses to the interrupt register set and allow poll mode functions.

5.1.2.7 Exclusive Access

The SIO marks itself locked anytime it is the slave of the access and LOCK# is sampled negated during the address phase. As a locked slave, the SIO responds to a master only when it samples LOCK# negated and FRAME# asserted. The locking master may negate LOCK# at the end of the last data phase. The SIO unlocks itself when FRAME# and LOCK# are both negated. The SIO will respond by asserting STOP# with TRDY# negated (retry) to all transactions when LOCK# is asserted during the address phase.

Locked cycles are never generated by the SIO.

5.1.2.8 PCI Special Cycle

When the SCE bit (bit 3) in the COM PCI configuration register (configuration offset 04h) is set to a "0", the SIO will ignore all PCI Special Cycles. When the SCE bit is set to a "1", the SIO will recognize PCI Special Cycles.

The only PCI Special Cycle currently recognized is the Stop Grant Special Cycle which is broadcast onto the PCI bus when an S-series processor enters the Stop Grant State. The SCE bit must be set to a "1" when the Stop Clock feature is being used.

5.1.3 TRANSACTION TERMINATION

The SIO supports both Master-initiated Termination as well as Target-initiated Termination.

5.1.3.1 SIO As Master—Master-Initiated Termination

The SIO supports two forms of master-initiated termination:

1. Normal termination of a completed transaction.
2. Abnormal termination due to no slave responding to the transaction (Abort).

Figure 5 shows the SIO performing master-abort termination. This occurs when no slave responds to the SIO's master transaction by asserting DEVSEL# within 5 PCICLK's after FRAME# assertion. This master-abort condition is abnormal and it indicates an error condition. The SIO will not retry the cycle. The Received Master-abort Status bit in the PCI Status Register will be set indicating that the SIO experienced a master-abort condition.

If an ISA master or the DMA is waiting for the PCI cycle to terminate (CHRDY negated), the master-abort condition will cause the SIO to assert CHRDY to terminate the ISA cycle. Note that write data will be lost and the read data will be all 1's at the end of the cycle. This is identical to the way an unclaimed cycle is handled on the "normally ready" ISA Bus. If the line buffer is the requester of the PCI transaction, the master-abort mechanism will end the PCI cycle, but no data will be transferred into or out of the line buffer. The line buffer will not be allowed to retry the cycle.

5.1.3.2 SIO As A Master—Response To Target-Initiated Termination

SIO's response as a master to target-termination:

1. For a target-abort, the SIO will not retry the cycle. If an ISA master or the DMA is waiting for the PCI cycle to complete (CHRDY negated), the target-abort condition will cause the SIO to assert CHRDY and end the cycle on the ISA Bus. If the ISA master or DMA device was reading from PCI memory, the SIO will drive all 1's on the data lines of the ISA Bus. The Received Target-abort Status bit in the PCI Status Register will be set indicating that the SIO experienced a target-abort condition.
2. If the SIO is retried as a master on the PCI Bus, it will remove it's request for 2 PCI clocks before asserting it again to retry the cycle.
3. If the SIO is disconnected as a master on the PCI Bus, it will respond very much as if it had been retried. The difference between retry and disconnect is that the SIO did not see any data phase for the retry. Disconnect may be generated by a PCI slave when the SIO is running a burst memory read cycle to fill it's 8-byte Line Buffer. In this case, the SIO may need to finish a multi-data phase transfer, and thus, must recycle through arbitration as required for a retry. An example of this is when the on-board DMA requests an 8-byte Line Buffer transfer and the SIO is disconnected before the Line Buffer is completely filled.

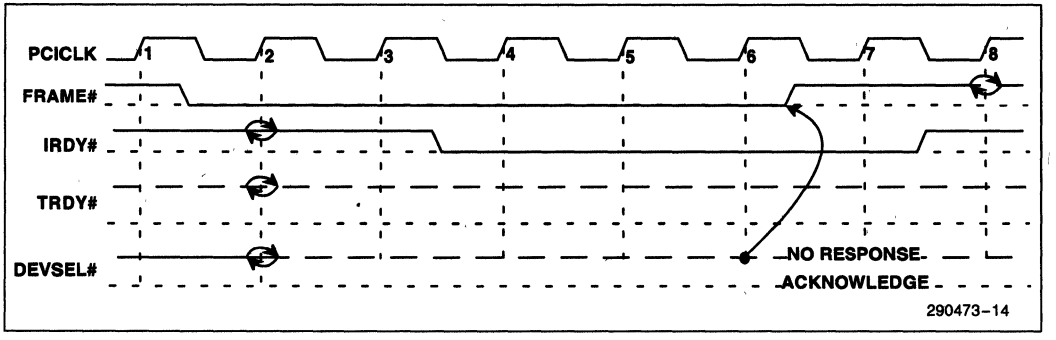


Figure 5. Master—Initiated Termination (Master-Abort)

5.1.3.3 SIO As A Target—Target-Initiated Termination

The SIO supports three forms of Target-initiated Termination:

Disconnect	Disconnect refers to termination requested because the SIO is unable to respond within the latency guidelines of the PCI specification. Note that this is not usually done on the first data phase.
Retry	Retry refers to termination requested because the target is currently in a state which makes it unable to process the transaction.
Abort	Abort refers to termination requested because the target will never be able to respond to the transaction.

The SIO will initiate Disconnect for PCI-originated/ISA-bound cycles after the first data phase due to incremental latency requirements. Since the SIO has only one Posted Write Buffer and every PCI to ISA incremental data phase will take longer than the specified 8 clocks, the SIO will always terminate burst cycles with a disconnect protocol. An example of this is when the SIO receives a burst memory write. Since the SIO only has one Posted Write Buffer, the transaction will automatically be disconnected after the first data phase.

The SIO will retry PCI masters:

1. For memory write cycles when the posted write buffer is full.
2. When the pending PCI cycle initiates some type of buffer management activity.
3. When the SIO is locked as a resource and a PCI master tries to access the SIO without negating the LOCK# signal in the address phase.
4. When the ISA Bus is occupied by an ISA master or DMA.

Target-abort is issued by the SIO when the internal SIO registers are the target of a PCI master I/O cycle and more than one byte enable is active. Accesses to the BIOS Timer Register and the Scatter/Gather Descriptor Table Pointer Registers are exceptions to this rule. Accesses to the Scatter/Gather Descriptor Table Pointer Register must be 32-bits wide and accesses to the BIOS Timer Register must be 16- or 32-bits wide. These accesses will not result in a SIO target-abort. The SIO responds with a target-abort since the registers must be accessed as 8-bit quantities. Target-abort resembles a retry, although the SIO also negates DEVSEL# along with the assertion of STOP#. Bit 11 in the Device Status Register is set to a 1 when the SIO target-aborts.

5.1.4 BUS LATENCY TIME-OUT

5.1.4.1 Master Latency Timer

Because the SIO only bursts a maximum of two Dwords, the PCI master latency timer is not implemented.

5.1.4.2 Target Incremental Latency Mechanism

As a slave, the SIO supports the Incremental Latency Mechanism for PCI to ISA cycles. The PCI specification states that for multi-data phase PCI cycles, if the incremental latency from current data phase (N) to the next data phase (N+1) is greater than 8 PCICLK's, then the slave must manipulate TRDY# and STOP# to stop the transaction upon completion of the current data phase (N). Since all PCI-originated (SIO is a slave)/ISA-bound cycles will require greater than the stated 8 PCICLK's, the SIO will automatically terminate these cycles after the first data phase. Note that latency to the first data phase is not restricted by this mechanism.

5.1.5 PARITY SUPPORT

As a master, the SIO generates address parity for read and write cycles, and data parity for write cycles. As a slave, the SIO generates data parity for read cycles. The SIO does not check parity and does not generate SERR#.

PAR is the calculated parity signal. PAR is "even" parity and is calculated on 36 bits; the 32 AD[31:0] signals plus the 4 C/BE[3:0]# signals. "Even" parity means that the number of 1's within the 36 bits plus PAR are counted and the sum is always even. PAR is always calculated on 36 bits, regardless of the valid byte enables. PAR is only guaranteed to be valid one PCI clock after the corresponding address or data phase.

5.1.6 RESET SUPPORT

The PCIRST# pin acts as the SIO hardware reset pin.

During Reset

AD[31:0], C/BE[3:0]#, and PAR are always driven low by the SIO from the leading edge of PCIRST#. FRAME#, IRDY#, TRDY#, STOP#, DEVSEL#, MEMREQ#, FLSHREQ#, CPUGNT#, GNT0#/ SIOREQ#, and GNT1#/ RESUME# are tri-stated from the leading edge of PCIRST#.

GNT2# and GNT3# are tri-stated from the leading edge of PCIRST#.

After Reset

AD[31:0], C/BE[3:0]#, and PAR are always tri-stated from the trailing edge of PCIRST#. If the internal arbiter is enabled (CPUREQ# sampled high on the trailing edge of PCIRST#), the SIO will drive these signals low again (synchronously 2-5 PCICLKs later) until the bus is given to another master. If the internal arbiter is disabled (CPUREQ# sampled low on the trailing edge of PCIRST#), these signals remain tri-stated until the SIO is required to drive them valid as a master or slave.

FRAME#, IRDY#, TRDY#, STOP#, and DEVSEL# remain tri-stated until driven by the SIO as either a master or a slave. MEMREQ#, FLSHREQ#, CPUGNT#, GNT0#/SIOREQ#, and GNT1#/RESUME# are tri-stated until driven by the SIO.

GNT2# and GNT3# are tri-stated until driven by the SIO.

After PCIRST, MEMREQ# and FLSHREQ# are driven inactive asynchronously from PCIRST# inactive. CPUGNT#, GNT0#/SIOREQ#, and GNT1#/ RESUME# are driven based on the arbitration scheme and the asserted REQx#'s.

GNT2# and GNT3# are also driven based on the arbitration scheme and the asserted REQx#'s.

5.1.7 DATA STEERING

Data steering logic internal to the SIO provides the assembly/disassembly, copy up/copy down mechanism for cycles between the 32-bit PCI data bus and the 16-bit ISA Bus. The steering logic ensures that the correct bytes are steered to the correct byte lane and that multiple cycles are run where applicable.

5.2 PCI Arbitration Controller

The 82378 contains a PCI Bus arbiter that supports six PCI masters; the Host Bridge, SIO, and four other masters. The SIO's REQ#/GNT# lines are internal. The integrated arbiter can be disabled by asserting CPUREQ# during PCIRST# (see Section 5.2.7, Power-up Configuration). When disabled, the SIO's REQ#, GNT#, and RESUME# signals become visible for an external arbiter. The internal arbiter is enabled upon power-up.

The internal arbiter contains several features that contribute to system efficiency:

- Use of a RESUME# signal to re-enable a backed-off initiator in order to minimize PCI Bus thrashing when the SIO generates a retry (Section 5.2.4.1).
- A programmable timer to re-enable retried initiators after a programmable number of PCICLK's (Section 5.2.4.2).
- The CPU (host bridge) can be optionally parked on the PCI Bus (Section 5.2.5).
- A programmable PCI Bus lock or PCI resource lock function (Section 5.2.6).

The PCI arbiter is also responsible for control of the Guaranteed Access Time (GAT) mode signals (Section 5.2.3.2).

5.2.1 ARBITRATION SIGNAL PROTOCOL

The internal arbiter follows the PCI arbitration method as outlined in the *Peripheral Component Interconnect (PCI) Specification*. The SIO's arbiter is discussed in this section.

5.2.1.1 Back-To-Back Transactions

The SIO as a master does not generate fast back-to-back accesses since it does not know if it is accessing the same target.

The SIO as a target supports fast back-to-back transactions. Note that for back-to-back cycles, the SIO treats positively decoded accesses and subtractively decoded accesses as different targets. Therefore, masters can only run fast back-to-back cycles to positively decoded addresses or to subtractively decoded addresses. Fast back-to-back cycles must not mix positive and subtractive decoded addresses. See the address decoding section to determine what addresses the SIO positively decodes and subtractively decodes.

5.2.2 PRIORITY SCHEME

The PCI arbitration priority scheme is programmable through the PCI Arbiter Priority Control and Arbiter Priority Control Extension Register. The arbiter consists of four banks that can be configured for the six masters to be arranged in a purely rotating priority scheme, one of twenty-four fixed priority schemes, or a hybrid combination (Figure 6).

Note that SIOREQ#/SIOGNT# are SIO internal signals.

The PCI Arbiter Priority Control (PAPC) and PCI Arbiter Priority Control Extension Register bits are shown below:

PCI Arbiter Priority Control Register Bits (PAPC)

Bit	Description
7	Bank 3 Rotate Control
6	Bank 2 Rotate Control
5	Bank 1 Rotate Control
4	Bank 0 Rotate Control
3	Bank 2 Fixed Priority Mode select B
2	Bank 2 Fixed Priority Mode select A
1	Bank 1 Fixed Priority Mode select
0	Bank 0 Fixed Priority Mode select

PCI Arbiter Priority Control Extension Register Bits (ARBPRIX)

Bit	Description
7:1	Reserved. Read as 0
0	Bank 3 Fixed Priority Mode select

PAPC defaults to 04h and ARBPRIX to 00h at reset selecting fixed mode #10 (Table 8) with the CPU the highest priority device guaranteeing access to BIOS.

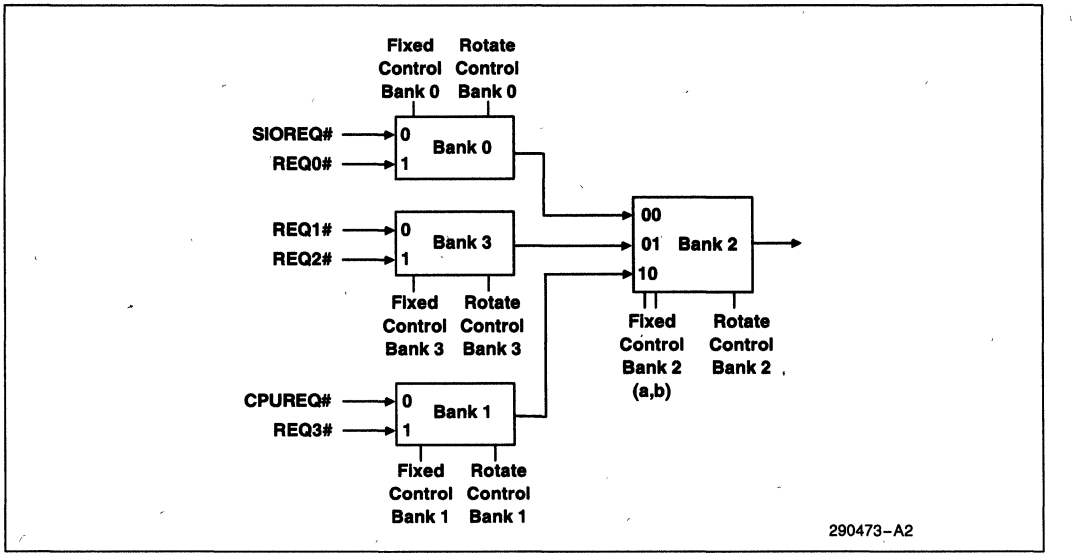


Figure 6. Arbiter Configuration Diagram for 82378ZB

5.2.2.1 Fixed Priority Mode

The 24 selectable fixed priority schemes are listed in Table 8.

Table 8. Fixed Priority Mode Bank Control Bits

Mode	Bank					Priority					
	3	2b	2a	1	0	Highest					Lowest
00	0	0	0	0	0	SIOREQ#	REQ0#	REQ2#	REQ3#	CPUREQ#	REQ1#
01	0	0	0	0	1	REQ0#	SIOREQ#	REQ2#	REQ3#	CPUREQ#	REQ#
02	0	0	0	1	0	SIOREQ#	REQ0#	REQ2#	REQ3#	REQ1#	CPUREQ#
03	0	0	0	1	1	REQ0#	SIOREQ#	REQ2#	REQ3#	REQ1#	CPUREQ#
04	0	0	1	0	0	CPUREQ#	REQ1#	SIOREQ#	REQ0#	REQ2#	REQ3#
05	0	0	1	0	1	CPUREQ#	REQ1#	REQ0#	SIOREQ#	REQ2#	REQ3#
06	0	0	1	1	0	REQ1#	CPUREQ#	SIOREQ#	REQ0#	REQ2#	REQ3#
07	0	0	1	1	1	REQ1#	CPUREQ#	REQ0#	SIOREQ#	REQ2#	REQ3#
08	0	1	0	0	0	REQ2#	REQ3#	CPUREQ#	REQ1#	SIOREQ#	REQ0#
09	0	1	0	0	1	REQ2#	REQ3#	CPUREQ#	REQ1#	REQ0#	SIOREQ#
0A	0	1	0	1	0	REQ2#	REQ3#	REQ1#	CPUREQ#	SIOREQ#	REQ0#
0B	0	1	0	1	1	REQ2#	REQ3#	REQ1#	CPUREQ#	REQ0#	SIOREQ#
0C–0F	0	1	1	x	x	Reserved					
10	1	0	0	0	0	SIOREQ#	REQ0#	REQ3#	REQ2#	CPUREQ#	REQ1#
11	1	0	0	0	1	REQ0#	SIOREQ#	REQ3#	REQ2#	CPUREQ#	REQ1#
12	1	0	0	1	0	SIOREQ#	REQ0#	REQ3#	REQ2#	REQ1#	CPUREQ#
13	1	0	0	1	1	REQ0#	SIOREQ#	REQ3#	REQ2#	REQ1#	CPUREQ#
14	1	0	1	0	0	CPUREQ#	REQ1#	SIOREQ#	REQ0#	REQ3#	REQ2#
15	1	0	1	0	1	CPUREQ#	REQ1#	REQ0#	SIOREQ#	REQ3#	REQ2#
16	1	0	1	1	0	REQ1#	SPUREQ#	SIOREQ#	REQ0#	REQ3#	REQ2#
17	1	0	1	1	1	REQ1#	CPUREQ#	REQ0#	SIOREQ#	REQ3#	REQ2#
18	1	1	0	0	0	REQ3#	REQ2#	CPUREQ#	REQ1#	SIOREQ#	REQ0#
19	1	1	0	0	1	REQ3#	REQ2#	CPUREQ#	REQ1#	REQ0#	SIOREQ#
1A	1	1	0	1	0	REQ3#	REQ2#	REQ1#	CPUREQ#	SIOREQ#	REQ0#
1B		1	0	1	1	REQ3#	REQ2#	REQ1#	CPUREQ#	REQ0#	SIOREQ#
1C–1F	1	1	1	x	x	Reserved					

The fixed bank control bit(s) selects which requester is the highest priority device within that particular bank. For fixed priority mode, bits[7:4] of the PAPC Register and bit zero of ARBPRIX must be 0's (rotate mode disabled).

The selectable fixed priority schemes provide 24 of the 64 possible fixed mode permutations possible for the six masters.

5.2.2.2 Rotating Priority Mode

When any bank rotate control bit is set to a one, that particular bank rotates between the requesting inputs. Any or all banks can be set in rotate mode. If all four banks are set in rotate mode, the six supported masters are all rotated and the arbiter is in a pure rotating priority mode. If, within a rotating bank, the highest priority device (a) does not have an active request, the lower priority device (b or c) will be granted the bus. However, this does not change the rotation scheme. When the bank toggles, device b is the highest priority. Because of this, the maximum latency a device can encounter is two complete rotations.

5.2.2.3 Mixed Priority Mode

Any combination of fixed priority and rotate priority modes can be used in different arbitration banks to achieve a specific arbitration scheme.

5.2.2.4 Locking Masters

When a master acquires the LOCK# signal, the arbiter gives that master highest priority until the LOCK# signal is negated and FRAME# is negated. This ensures that a master that locked a resource will eventually be able to unlock that same resource.

5.2.3 MEMREQ#, FLSHREQ#, AND MEMACK# PROTOCOL

Before an ISA master or the DMA can be granted the PCI Bus, it is necessary that all PCI system posted write buffers be flushed (including the SIO's Posted Write Buffer). Also, since the ISA originated cycle could access memory on the host bridge, it's possible that the ISA master or the DMA could be held in wait states (via IOCHRDY) waiting for the host bridge arbitration for longer than the 2.5 μs ISA specification. The SIO has an optional mode called the Guaranteed Access Time Mode (GAT) that ensures that this timing specification is not violated. This is accomplished by delaying the ISA REQ# signal to the requesting master or DMA until the ISA Bus, PCI Bus, and the System Memory Bus are arbitrated for and owned.

Three PCI sideband signals, MEMREQ#, FLSHREQ#, and MEMACK# are used to support the System Posted Write Buffer Flushing and Guaranteed Access Time mechanisms. The MEMACK# signal is the common acknowledge signal for both mechanisms. Note that when MEMREQ# is asserted, FLSHREQ# is also asserted. Table 9 shows the relationship between MEMREQ# and FLSHREQ#:

Table 9. FLSHREQ#, MEMREQ#

FLSHREQ#	MEMREQ#	Meaning
1	1	Idle
0	1	Flush buffers pointing towards PCI to avoid ISA deadlock
1	0	Reserved
0	0	GAT mode, Guarantee PCI Bus immediate access to main memory

5.2.3.1 Flushing the System Posted Write Buffers

Once an ISA master or the DMA begins a cycle on the ISA Bus, the cycle can not be backed-off. It can only be held in wait states via IOCHRDY. In order to know the destination of ISA master cycles, the cycle needs to begin. However, after the cycle has started, no other device can intervene and gain ownership of the ISA Bus until the cycle has completed and arbitration is performed. A potential deadlock condition exists when an ISA originated cycle to the PCI Bus finds the PCI target inaccessible due to an interacting event that also requires the ISA Bus. To avoid this potential deadlock, all PCI posted write buffers in the system must be disabled and flushed before DACK can be returned. The buffers must remain disabled while the ISA Bus is occupied by an ISA master or the DMA.

When an ISA master or the DMA requests the ISA Bus, the SIO asserts FLSHREQ#. FLSHREQ# is an indication to the system to flush all posted write buffers pointing towards the PCI Bus. The SIO also flushes it's own Posted Write Buffer. Once the posted write buffers have been flushed and disabled, the system asserts MEMACK#. Once the SIO receives the MEMACK# acknowledgment signal, it asserts the DACK signal giving the requesting master the bus. FLSHREQ# stays active as long as the ISA master or DMA owns the ISA Bus.

5.2.3.2 Guaranteed Access Time Mode

Guaranteed Access Time (GAT) Mode is enabled/disabled via the PCI Arbiter Control Register. When this mode is enabled, the MEMREQ# and MEMACK# signals are used to guarantee that the ISA 2.5 μs IOCHRDY specification is not violated.

When an ISA master or DMA slave requests the ISA Bus (DREQ# active), the ISA Bus, the PCI Bus, and the memory bus must be arbitrated for and all three must be owned before the ISA master or DMA slave is granted the ISA Bus. After receiving the DREQ# signal from the ISA master or DMA slave, MEMREQ# and FLSHREQ# are asserted (FLSHREQ# is driven active, regardless of GAT mode being enabled or disabled). MEMREQ# is a request for direct access to main memory. MEMREQ# and FLSHREQ# will be asserted as long as the ISA master or the DMA owns the ISA Bus. When MEMACK# is received by the SIO (all posted write buffers are flushed and the memory bus is dedicated to the PCI interface), it will request the PCI Bus. When it is granted the PCI Bus, it asserts the DACK signal releasing the ISA Bus to the requesting master or the DMA.

The use of MEMREQ#, FLSHREQ#, and MEMACK# does not guarantee functionality with ISA masters that don't acknowledge IOCHRDY. These signals just guarantee the IOCHRDY inactive specification.

NOTE:
Usage of an external arbiter in GAT mode will require special logic in the arbiter.

2

5.2.4 RETRY THRASHING RESOLVE

When a PCI initiator's access is retried, the initiator releases the PCI Bus for a minimum of two PCI clocks and will then normally request the PCI Bus again. To avoid thrashing the bus with retry after retry, the PCI arbiter provides REQ# masking. The REQ# masking mechanism differentiates between SIO target retries and all other retries.

For initiators which were retried by the SIO as a target, the masked REQ# is flagged to be cleared upon RESUME# active. All other retries trigger the Master Retry Timer, if enabled. Upon expiration of this timer, the mask is cleared.

5.2.4.1 Resume Function (RESUME#)

The conditions under which the SIO forces a retry to a PCI master and will mask the REQ# are:

1. Any required buffer management
2. ISA Bus occupied by ISA master or DMA
3. The PCI to ISA Posted Write Buffer is full
4. The SIO is locked as a resource and LOCK# is asserted during the address process.

The RESUME# signal is pulsed whenever the SIO has retried a PCI cycle for one of the above reasons and that condition has passed. When RESUME# is asserted, the SIO will unmask the REQ#'s that are masked and flagged to be cleared by RESUME#.

If the internal arbiter is enabled, RESUME# is an internal signal. The RESUME# signal becomes visible as an output when the internal arbiter is disabled. This allows an external arbiter to optionally avoid retry thrashing associated with the SIO as a target. The RESUME# signal is asserted for one PCI clock.

5.2.4.2 Master Retry Timer

To re-enable a PCI master's REQ# which resulted in a retry to a slave other than the SIO, a SIO programmable Master Retry Timer has been provided. This timer can be programmed for 0 (disabled), 16, 32, or 64 PCICLKs. Once the SIO has detected that a PCI slave has forced a retry, the timer will be triggered and the corresponding master's REQ# will be masked. All subsequent PCI retries by this REQ# signal will by masked by the SIO. Expiration of this timer will unmask all of the masked requests. This timer has no effect on the request lines that have been masked due to a SIO retry.

If no other PCI masters are requesting the PCI Bus, all of the REQ#'s masked for the timer will be cleared and the timer will be reset. This is necessary to assist the host bridge in determining when to re-enable any disabled posted write buffers.

5.2.5 BUS PARKING

The SIO arbitration logic supplies a mechanism for PCI Bus parking. Parking is only allowed for the device which is tied to CPUREQ# (typically the system CPU). When bus parking is enabled, CPUGNT# will be asserted when no other agent is currently using or requesting the bus. This achieves the minimum PCI arbitration latency possible. Enabling of bus parking is achieved by programming the Arbiter Control Register. REQ0#, REQ1#, and the internal SIOREQ# are not allowed to park on the PCI Bus.

Upon assertion of CPUGNT# due to bus parking enabled and the PCI Bus idle, the CPU (or the parked agent) must ensure that AD[31:0], C/BE[3:0], and (one PCICLK later) PAR are driven. If bus parking is disabled, the SIO takes responsibility for driving the bus when it is idle.

5.2.6 BUS LOCK MODE

As an option, the SIO arbiter can be configured to run in Bus Lock Mode or Resource Lock Mode. The Bus Lock Mode is used to lock the entire PCI Bus. This may improve performance in some systems that frequently run quick read-modify-write cycles. Bus Lock Mode emulates the LOCK environment found in today's PC by restricting bus ownership when the PCI Bus is locked. With Bus Lock enabled, the arbiter recognizes a LOCK# being driven by any initiator and does not allow any other PCI initiator to be granted the PCI Bus until LOCK# and FRAME# are both negated indicating the master released lock. When Bus Lock is disabled, the default resource lock mechanism is implemented (normal resource lock) and a higher priority PCI initiator could intervene between the read and write cycles and run non-exclusive accesses to any unlocked resource.

5.2.7 POWER-UP CONFIGURATION

The SIO's arbiter is enabled if CPUREQ# is sampled high on the trailing edge of PCIRST#. When enabled, the arbiter is set in fixed priority mode 4 with CPU bus parking turned off. Fixed mode 4 guarantees that the CPU will be able to run accesses to the BIOS in order to configure the system, regardless of the state of the other REQ#'s. Note that the Host Bridge should drive CPUREQ# high during the trailing edge of PCIRST#. When the arbiter is enabled, the SIO acts as the central resource responsible for driving the AD[31:0], C/BE[3:0]#, and PAR signals when no one is granted the PCI Bus and the bus is idle. The SIO is always responsible for driving AD[31:0], C/BE[3:0]#, and PAR when it is granted the bus and as appropriate when it is the master of a transaction. After reset, if the arbiter is enabled, CPUGNT#, GNT0#, GNT1#, and the internal SIOGNT# will be driven based on the arbitration scheme and the asserted REQ#'s.

If an external arbiter is present in the system, the CPUREQ# signal should be tied low. When CPUREQ# is sampled low on the trailing edge of PCIRST#, the internal arbiter is disabled. When the internal arbiter is disabled, the SIO does not drive AD[31:0], C/BE[3:0]#, and PAR as the central resource. In this case, the SIO is only responsible for driving AD[31:0], C/BE[3:0]#, and PAR when it is granted the bus. If the SIO's arbiter is disabled, GNT0# becomes SIOREQ#, GNT1# becomes RESUME#, and REQ0# becomes SIOGNT#. This exposes the normally embedded SIO arbitration signals.

NOTE:
Usage of an external arbiter in GAT mode will require special logic in the arbiter.

5.3 ISA Interface

5.3.1 ISA INTERFACE OVERVIEW

The SIO incorporates a fully ISA Bus compatible master and slave interface. The SIO directly drives six ISA slots without external data or address buffers. The ISA interface also provides byte swap logic, I/O recovery support, wait-state generation, and SYSCLK generation.

The ISA interface supports the following types of cycles:

- PCI-initiated I/O and memory cycles to the ISA Bus.
- DMA compatible cycles between PCI memory and ISA I/O and between ISA I/O and ISA memory, DMA type "A", type "B", and type "F" cycles between PCI memory and ISA I/O.
- ISA Refresh cycles initiated by either the SIO or an external ISA master.
- ISA master-initiated memory cycles to the PCI Bus and ISA master-initiated I/O cycles to the internal SIO registers.

The refresh and DMA cycles are shown and described in Section 5.4.

5.3.2 SIO AS AN ISA MASTER

The SIO executes ISA cycles as an ISA master whenever a PCI initiated cycle is forwarded to the ISA Bus. The SIO also acts as an ISA master on behalf of DMA and refresh.

ISYSCLK is an internal 8 MHz clock.

5.3.3 SIO AS AN ISA SLAVE

The SIO operates as an ISA slave when:

- An ISA master accesses SIO internal registers.
- An ISA master accesses PCI memory on the PCI Bus.

5.3.3.1 ISA Master Accesses To SIO Registers

An ISA Bus master has access to SIO internal registers as shown in Table 19. An ISA master to SIO register cycle will always run as an 8-bit extended cycle (IOCHRDY will be held inactive until the cycle is completed).

Table 10. Arbitration Latency

Bus Condition	Arbitration Latency
Parked	0 PCICLKs for Agent 0, 2 PCICLKs for All Other
Not Parked	1 PCICLK for All Agents

5.3.3.2 ISA Master Accesses to PCI Resource

An ISA master can access PCI memory, but not I/O devices residing on the PCI Bus. The ISA/DMA address decoder determines which memory cycles should be directed towards the PCI Bus. During ISA master read cycles to the PCI Bus, the SIO will return all 1's if the PCI cycle is target-aborted or does not respond.

If the SIO is programmed for GAT mode, the SIO arbiter will not grant the ISA Bus before gaining ownership of both the PCI Bus and system memory. However, if the SIO is not programmed in this mode, the SIO does not need to arbitrate for the PCI Bus before granting the ISA Bus to the ISA master. For more details on the arbitration, refer to Section 5.2.2.

All cycles forwarded to a PCI resource will run as 16-bit extended cycles (i.e. IOCHRDY will be held inactive until the cycle is completed).

Because the ISA bus size is different from the PCI bus size, the data steering logic inside the SIO is responsible for steering the data to the correct byte lanes on both buses, and assembling/disassembling the data as necessary.

5.3.4 ISA MASTER TO ISA SLAVE SUPPORT

During ISA master cycles to ISA slaves, the SIO drives several signals to support the transfer:

BALE:

This signal is driven high while the ISA master owns the ISA Bus.

AEN:

This signal is driven low while the ISA master owns the ISA Bus.

SMEMR# and SMEMW#:

These signals are driven active by the SIO whenever the ISA master drives a memory cycle to an address below 1 Mb.

Utility Bus Buffer Control Signals and Chip Select Signals:

These signals are driven active as appropriate whenever an ISA master accesses devices on the Utility Bus. For more details, see Section 5.9.

Data Swap Logic:

The data swap logic inside the SIO is activated as appropriate to swap data between the even and odd byte lanes. This is discussed in further detail in Section 5.3.5.

5.3.5 DATA BYTE SWAPPING

The data swap logic is integrated in the SIO. For slaves that reside on the ISA Bus, data swapping is performed if the slave (I/O or memory) and ISA bus master (or DMA) sizes differ and the upper (odd) byte of data is being accessed. Table 11 shows when data swapping is provided during DMA. Table 12 shows when data swapping is provided during ISA master cycles to 8-bit ISA slaves.

Table 11. DMA Data Swap

DMA I/O Device Size	ISA Memory Slave Size	Swap	Comments I/O	⟷	Memory
8-Bit	8-Bit	No	SD[7:0]	⟷	SD[7:0]
8-Bit	16-Bit	Yes	SD[7:0]	⟷	SD[7:0]
			SD[7:0]	⟷	SD[15:8]
16-Bit	8-Bit	No	Not Supported		
16-Bit	16-Bit	No	SD[15:0]	⟷	SD[15:0]

The SIO monitors the SBHE# and SA0 signals to determine when to swap the data. The SIO ensures that the data is placed on the appropriate byte lane.

Table 12. 16-Bit Master to 8-Bit Slave Data Swap

SBHE#	SA0	SD[15:8]	SD[7:0]	Comments
0	0	Odd	Even	Word Transfer (data swapping not required)
0	1	Odd	Even	Byte Swap(1,2)
1	0	—	Even	Byte Transfer (data swapping not required)
1	1	—	—	Not Allowed

NOTES:
1. For ISA master read cycles, the SIO swaps the data from the lower byte to the upper byte.
2. For ISA master write cycles, the SIO swaps the data from the upper byte to the lower byte.

5.3.6 ISA CLOCK GENERATION

The SIO generates the ISA system clock (SYSCLK). SYSCLK is a divided down version of the PCICLK (see Table 13). The clock divisor value is programmed through the ISA Clock Divisor Register.

Table 13. SYSCLK Generation from PCICLK

PCICLK (MHz)	Divisor (Programmable)	SYSCLK (MHz)
25	3	8.33
33	4 (default)	8.33

NOTE:
For PCI frequencies less than 33 MHz (not including 25 MHz), a clock divisor value must be selected that ensures that the ISA Bus frequency does not violate the 6 MHz to 8.33 MHz SYSCLK specification.

5.3.7 WAIT STATE GENERATION

The SIO will add wait states to the following cycles, if IOCHRDY is sampled active low. Wait states will be added as long as IOCHRDY is low.

- During Refresh and SIO master cycles (not including DMA) to the ISA Bus.
- During DMA compatible transfers between ISA I/O and ISA memory only.

For ISA master cycles targeted for the SIO's internal registers or PCI memory, the SIO will always extend the cycle by driving IOCHRDY low until the transaction is complete.

The SIO will shorten the following cycles, if ZEROWS# is sampled active.

- During SIO master cycles (not including DMA) to 8-bit and 16-bit ISA memory.
- During SIO master cycles (not including DMA) to 8-bit ISA I/O only.

For ISA master cycles targeted for the SIO's internal registers or PCI memory, the SIO will not assert ZEROWS#.

NOTE:
If IOCHRDY and ZEROWS# are sampled active at the same time, IOCHRDY will take precedence and wait-states will be added.

5.3.8 I/O RECOVERY

The I/O recovery mechanism in the SIO is used to add additional recovery delay between PCI originated 8-bit and 16-bit I/O cycles to the ISA Bus. The SIO automatically forces a minimum delay of four SYSCLKs between back-to-back 8- and 16-bit I/O cycles to the ISA Bus. This delay is measured from the rising edge of the I/O command (IOR# or IOW#) to the falling edge of the next BALE. If a delay of greater than four SYSCLKs is required, the ISA I/O Recovery Time Register can be programmed to increase the delay in increments of SYSCLKs. No additional delay is inserted for back-to-back I/O "sub cycles" generated as a result of byte assembly or disassembly .

5.4 DMA Controller

5.4.1 DMA CONTROLLER OVERVIEW

The DMA circuitry incorporates the functionality of two 82C37 DMA controllers with seven independently programmable channels (Channels 0–3 and Channels 5–7). DMA Channel 4 is used to cascade the two controllers and will default to cascade mode in the DMA Channel Mode (DCM) Register. In addition to accepting requests from DMA slaves, the DMA controller also responds to requests that are initiated by software. Software may initiate a DMA service request by setting any bit in the DMA Channel Request Register to a 1. The DMA controller for Channels 0–3 is referred to as "DMA-1" and the controller for Channels 4–7 is referred to as "DMA-2".

Each DMA channel may be programmed for 8- or 16-bit DMA device size and ISA-compatible, Type "A", Type "B", or Type "F" transfer timing. Each DMA channel defaults to the compatible settings for DMA device size: channels [3:0] default to 8-bit, count-by-bytes transfers, and channels [7:5] default to 16-bit, count-by-words (address shifted) transfers. The SIO provides the timing control and data size translation necessary for the DMA transfer between the PCI and the ISA Bus. ISA-compatible is the default transfer timing.

Full 32-bit addressing is supported as an extension of the ISA-compatible specification. Each channel includes a 16-bit ISA compatible Current Register which holds the 16 least-significant bits of the 32-bit address, and an ISA compatible Low Page Register which contains the eight second most significant bits. An additional High Page Register contains the eight most significant bits of the 32-bit address.

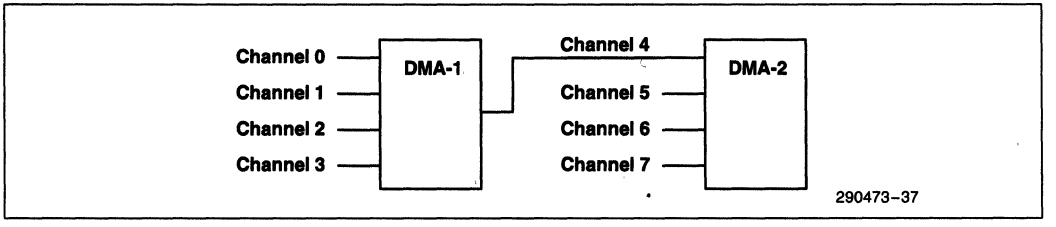


Figure 7. Internal DMA Controller

The DMA controller also features refresh address generation, and auto-initialization following a DMA termination.

The DMA controller receives commands from the ISA Bus arbiter to perform either DMA cycles or refresh cycles. The arbiter determines which requester from among the requesting DMA slaves, the PCI Bus, and refresh should have the bus.

The DMA controller is at any time either in master mode or slave mode. In master mode, the DMA controller is either servicing a DMA slave's request for DMA cycles, or allowing a 16-bit ISA master to use the bus via a cascaded DREQ signal. In slave mode, the SIO monitors both the ISA Bus and the PCI, decoding and responding to I/O read and write commands that address its registers.

Note that a DMA device (I/O device) is always on the ISA Bus, but the memory device is either on the ISA or PCI Bus. If the memory is decoded to be on the ISA Bus, then the DMA cycle will run as a compatible cycle. If the memory is decoded to be on the PCI Bus, the cycle can run as compatible, "A", "B", or "F" type. The ISA controller will not drive a valid address for type "A", "B", and "F" DMA transfers on the ISA Bus.

When the SIO is running a DMA cycle in compatible timing mode, the SIO will drive the MEMR# or MEMW# strobes if the address is less than 16 MBytes (00000000h–00FFFFFFh). These memory strobes will be generated regardless of whether the cycle is decoded for PCI or ISA memory. The SMEMR# and SMEMW# will be generated if the address is less than 1 MBytes (00000000h–000FFFFFh). If the address is greater than 16 MBytes (01000000h–FFFFFFFFh) the MEMR# or MEMW# strobe will not be generated in order to avoid aliasing problems. For type "A", "B", and "F" timing mode DMA cycles, the SIO will only generate the MEMR# or MEMW# strobe when the address is decoded for ISA memory. When this occurs, the cycle converts to compatible mode timing.

During DMA cycles, the ISA controller drives AEN high to prevent the I/O devices from misinterpreting the DMA cycle as a valid I/O cycle. The BALE signal is also driven high during DMA cycles. Also, during DMA memory read cycles to the PCI Bus, the SIO will return all 1's to the ISA Bus if the PCI cycle is either target-aborted or does not respond.

Further details can be found in the 82C37 data sheet.

5.4.2 DMA TIMINGS

ISA Compatible timing is provided for DMA slave devices. Three additional timings are provided for I/O slaves capable of running at faster speeds. These timings are referred to as Type "A", Type "B", and Type "F".

2

5.4.2.1 Compatible Timing

Compatible timing runs at 8 SYSCLKs during the repeated portion of a Block or Demand mode transfer.

5.4.2.2 Type "A" Timing

Type "A" timing is provided to allow shorter cycles to PCI memory. Type "A" timing runs at 6 SYSCLKs (720 ns/cycle) during the repeated portion of a block or demand mode transfer. This timing assumes an 8.33 MHz SYSCLK. Type "A" timing varies from compatible timing primarily in shortening the memory operation to the minimum allowed by system memory. The I/O portion of the cycle (data setup on write, I/O read access time) is the same as with compatible cycles. The actual active command time is shorter, but it is expected that the DMA devices which provide the data access time or write data setup time should not require excess IOR# or IOW# command active time. Because of this, most ISA DMA devices should be able to use type "A" timing.

5.4.2.3 Type "B" Timing

Type "B" timing is provided for 8-/16-bit ISA DMA devices which can accept faster I/O timing. Type "B" only works with PCI memory. Type "B" timing runs at 5 SYSCLKs (600 ns/cycle) during the repeated portion of a Block or Demand mode transfer. This timing assumes an 8.33 MHz SYSCLK. Type "B" timing requires faster DMA slave devices than compatible timing in that the cycles are shortened so that the data setup time on I/O write cycles is shortened and the I/O read access time is required to be faster. Some of the current ISA devices should be able to support type "B" timing, but these will probably be more recent designs using relatively fast technology.

5.4.2.4 Type "F" Timing

Type "F" timing provides high performance DMA transfer capability. These transfers are mainly for fast I/O devices (i.e. IDE devices). Type "F" timing runs at 3 SYSCLKs (360 ns/cycle) during the repeated portion of a Block or Demand mode transfer.

5.4.2.5 DREQ and DACK# Latency Control

The SIO DMA arbiter maintains a minimum DREQ to DACK# latency on DMA channels programmed to operate in compatible timing mode. This is to support older devices such as the 8272A. The DREQs are delayed by eight SYSCLKs prior to being seen by the arbiter logic. Software requests will not have this minimum request to DACK# latency.

5.4.3 ISA BUS/DMA ARBITRATION

The ISA Bus arbiter evaluates requests for the ISA Bus coming from several different sources. The DMA unit, the refresh counter, and the PCI Bus (primarily the Host CPU) may all request access to the ISA Bus. Additionally, 16-bit ISA masters may request the bus through a cascaded DMA channel.

The SIO ISA arbiter uses a three-way rotating priority arbitration method. At each level, the devices which are considered equal are given a rotating priority. On a fully loaded bus, the order in which the devices are granted bus access is independent of the order in which they assert a bus request. This is because devices are serviced based on their position in the rotation. The arbitration scheme assures that DMA channels access the bus with minimal latency.

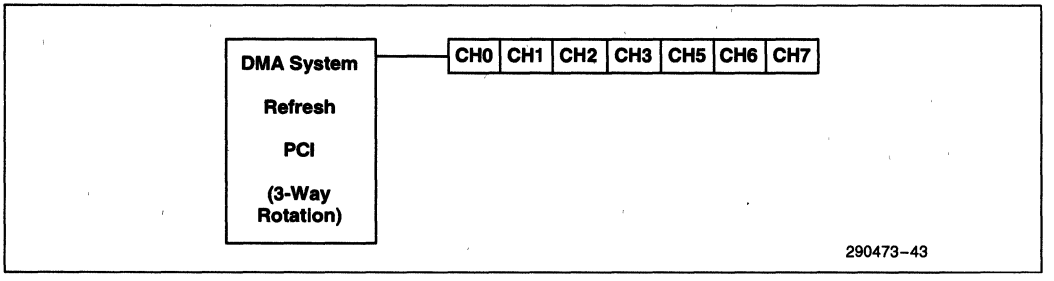


Figure 8. ISA Arbiter with DMA in Fixed Priority

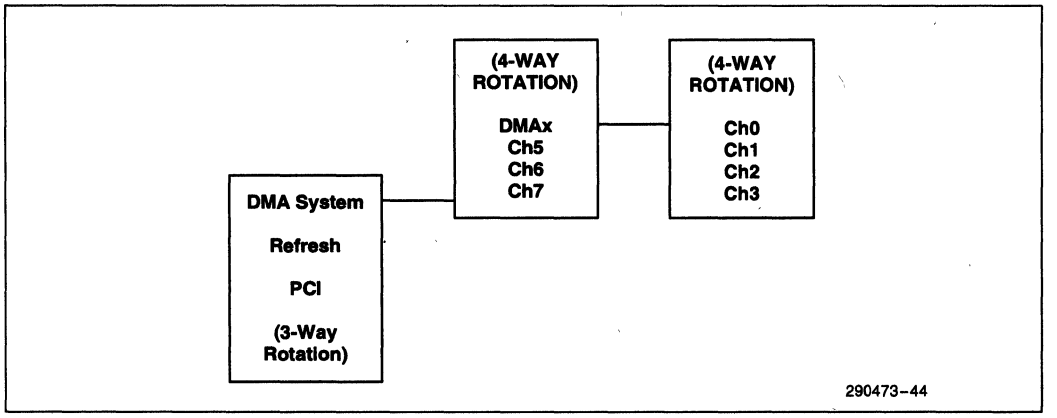


Figure 9. ISA Arbiter with DMA in Rotating Priority

2

5.4.3.1 Channel Priority

For priority resolution the DMA consists of two logical channel groups: channels 0–3 and channels 4–7 (see Figure 7). Each group may be in either fixed or rotate mode, as determined by the DMA Command Register.

For prioritization purposes, the source of the DMA request is transparent. DMA I/O slaves normally assert their DREQ line to arbitrate for DMA service. However, a software request for DMA service can be presented through each channel's DMA Request Register. A software request is subject to the same prioritization as any hardware request. Please see the detailed register description in Section 4.2.4 for Request Register programming information.

Fixed Priority

The initial fixed priority structure is as follows:

Table 14. Initial Fixed Priority Structure

High Priority . . . . . Low Priority
(0, 1, 2, 3), 5, 6, 7

The fixed priority ordering is 0, 1, 2, 3, 5, 6, and 7. In this scheme, Channel 0 has the highest priority, and channel 7 has the lowest priority. Channels [3:0] of DMA-1 assume the priority position of Channel 4 in DMA-2, thus taking priority over channels 5, 6, and 7.

Rotating Priority

Rotation allows for "fairness" in priority resolution. The priority chain rotates so that the last channel serviced is assigned the lowest priority in the channel group (0–3, 5–7).

Channels 0–3 rotate as a group of 4. They are always placed between Channel 5 and Channel 7 in the priority list.

Channel 5–7 rotate as part of a group of 4. That is, channels (5–7) form the first three positions in the rotation, while channel group (0–3) comprises the fourth position in the arbitration.

Table 15 demonstrates rotation priority.

Table 15. Rotating Priority Example

Programmed Mode	Action	Priority High .....Low
Group (0–3) is in Rotation Mode Group (4–7) is in Fixed Mode	1) Initial Setting	(0, 1, 2, 3), 5, 6, 7
	2) After Servicing Channel 2	(3, 0, 1, 2), 5, 6, 7
	3) After Servicing Channel 3	(0, 1, 2, 3), 5, 6, 7
Group (0–3) is in Rotation Mode Group (4–7) is in Rotation Mode	1) Initial Setting	(0, 1, 2, 3), 5, 6, 7
	2) After Servicing Channel 0	5, 6, 7, (1, 2, 3, 0)
	3) After Servicing Channel 5	6, 7, (1, 2, 3, 0), 5
	4) After servicing Channel 6	7, (1, 2, 3, 0), 5, 6
	5) After servicing Channel 7	(1, 2, 3, 0), 5, 6, 7

NOTE:
The first servicing of channel 0 caused double rotation.

5.4.3.2 DMA Preemption In Performance Timing Modes

A DMA slave device that is not programmed for compatible timing will be preempted from the bus by another device that requests use of the bus. This will occur, regardless of the priority of the pending request. For DMA devices not using compatible timing mode, the DMA controller stops the DMA transfer and releases the bus within 32 BCLK (4 μs) of a preemption. Upon the expiration of the 4 μs timer, the DACK will be inactivated after the current DMA cycle has completed. The bus will then be arbitrated for and granted to the highest priority requester. This feature allows flexibility in programming the DMA for long transfer sequences in a performance timing mode while guaranteeing that vital system services such as refresh are allowed access to the ISA Bus.

The 4 μs timer is not used in compatible timing mode. It is only used for DMA channels programmed for Type "A", Type "B", or Type "F" timing. It is also not used for 16-bit ISA masters cascaded through the DMA DREQ lines.

If the DMA channel that was preempted by the 4 μs timer was operating in Block Mode, an internal bit will be set so that the channel will be arbitrated for again, independent of the state of DREQ.

5.4.3.3. Arbitration during Non-Maskable Interrupts

If a non-maskable interrupt (NMI) is pending, and the CPU is requesting the bus, then the DMA controller will be bypassed each time it comes up for rotation. This will give the CPU the bus bandwidth it requires to process the interrupt as fast as possible.

5.4.3.4 Programmable Guaranteed Access Time Mode (GAT Mode)

The PCI Arbiter Register contains a bit for configuring the SIO in "Guaranteed Access Time Mode" (GAT Mode). This mode guarantees that the 2.5 μs CHRDY time-out specification for ISA masters running cycles to PCI will not be exceeded. When an ISA master or DMA slave arbitrates for the ISA Bus, and the SIO is configured in Guaranteed Access Time Mode, the MEMREQ# pin will be asserted by the PCI arbiter in order to gain ownership of main memory. The arbitration for the PCI and then the main memory bus must be completed prior to granting the DACK# to the ISA master or DMA slave. A MEMACK# signal to the SIO indicates that the SIO now owns main memory and can grant the DACK# to the ISA master or DMA slave. A detailed description is contained in Section 5.2.3.2.

5.4.4 REGISTER FUNCTIONALITY

Please see Section 4.2 for detailed information on register programming, bit definitions, and default values/functions of the DMA registers after a PCIRST#.

DMA Channel 4 is used to cascade the two DMA controllers together and should not be programmed for any mode other than cascade. The DMA Channel Mode Register for channel 4 will default to cascade mode. Special attention should also be take when programming the Command and Mask Registers as related to channel 4.

5.4.4.1 Address Compatibility Mode

Whenever the DMA is operating in address compatibility mode, the addresses do not increment or decrement through the High and Low Page Registers, and the High Page Register is set to 00h. This is compatible with the 82C37 and Low Page Register implementation used in the PC/AT. This mode is set when any of the lower three address bytes of a channel are programmed. If the upper byte of a channel's address is programmed last, the channel will go into extended address mode. In this mode, the high byte may be any value and the address will increment or decrement through the entire 32-bit address.

After PCIRST# is negated, all channels will be set to address compatibility mode. The DMA Master Clear command will also reset the proper channels to address compatibility mode.

5.4.4.2 Summary of DMA Transfer Sizes

Table 16 lists each of the DMA device transfer sizes. The column labeled "Current Byte/Word Count Register" indicates that the register contents represents either the number of bytes to transfer or the number of 16-bit words to transfer. The column labeled "Current Address Increment/Decrement" indicates the number added to or taken from the Current Address register after each DMA transfer cycle. The DMA Channel Mode Register determines if the Current Address Register will be incremented or decremented.

5.4.4.3 Address Shifting when Programmed for 16-Bit I/O Count by Words

To maintain compatibility with the implementation of the DMA in the PC/AT which used the 82C37, the DMA will shift the addresses when the DMA Channel Extended Mode Register is programmed for, or defaulted to, transfers to/from a 16-bit device count-by-words. Note that the least significant bit of the Low Page Register is dropped in 16-bit shifted mode. When programming the Current Address Register when the DMA channel is in this mode, the Current Address must be programmed to an even address with the address value shifted right by one bit. The address shifting is shown in Table 17.

5.4.4.4 Autoinitialize

By programming a bit in the DMA Channel Mode Register, a channel may be set up as an autoinitialize channel. During Autoinitialize initialization, the original values of the Current Page, Current Address and Current Byte/Word Count Registers are automatically restored from the Base Page, Address, and Byte/Word Count Registers of that channel following TC. The Base Registers are loaded simultaneously with the Current Registers by the microprocessor and remain unchanged throughout the DMA service. The mask bit is not set when the channel is in autoinitialize. Following Autoinitialize, the channel is ready to perform another DMA service, without CPU intervention, as soon as a valid DREQ is detected.

Table 16. DMA Transfer Size

DMA Device Date Size and Word Count	Current Byte/Word Count Register	Current Address Increment/Decrement
8-Bit I/O, Count by Bytes	Bytes	1
16-Bit I/O, Count by Words (Address Shifted)	Words	1
16-Bit I/O, Count by Bytes	Bytes	2

Table 17. Address Shifting in 16-Bit I/O DMA Transfers

Output Address	8-Bit I/O Programmed Address	16-Bit I/O Programmed Address (Shifted)	16-Bit I/O Programmed Address (No Shift)
A0	A0	0	A0
A[16:1]	A[16:1]	A[15:0]	A[16:1]
A[31:17]	A[31:17]	A[31:17]	A[31:17]

NOTE:
The least significant bit of the Low Page Register is dropped in 16-bit shifted mode.

5.4.5 SOFTWARE COMMANDS

There are three additional special software commands which can be executed by the DMA controller. The three software commands are:

1. Clear Byte Pointer Flip-Flop
2. Master Clear
3. Clear Mask Register

They do not depend on any specific bit pattern on the data bus.

5.4.5.1 Clear Byte Pointer Flip-Flop

This command is executed prior to writing or reading new address or word count information to/from the DMA controller. This initializes the flip-flop to a known state so that subsequent accesses to register contents by the microprocessor will address upper and lower bytes in the correct sequence.

When the Host CPU is reading or writing DMA registers, two Byte Pointer Flip-Flops are used; one for channels 0–3 and one for channels 4–7. Both of these act independently. There are separate software commands for clearing each of them (0Ch for channels 0–3, 0D8h for channels 4–7).

5.4.5.2 DMA Master Clear

This software instruction has the same effect as the hardware reset. The Command, Status, Request, and Internal First/Last Flip-Flop Registers are cleared and the Mask Register is set. The DMA controller will enter the idle cycle.

There are two independent master clear commands, 0Dh which acts on channels 0–3, and 0DAh which acts on channels 4–7.

5.4.5.3 Clear Mask Register

This command clears the mask bits of all four channels, enabling them to accept DMA requests. I/O port 00Eh is used for channels 0–3 and I/O port 0DCh is used for channels 4–7.

5.4.6 TERMINAL COUNT/EOP SUMMARY

This is a summary of the events that will happen as a result of a terminal count or external EOP when running DMA in various modes. (See Table 18.)

5.4.7 ISA REFRESH CYCLES

Refresh cycles are generated by two sources: the refresh controller inside the SIO component or by ISA bus masters other than the SIO. The ISA bus controller will enable the address lines SA[15:0] so that when MEMR# goes active, the entire ISA system memory is refreshed at one time. Memory slaves on the ISA Bus must not drive any data onto the data bus during the refresh cycle.

Counter 1 in the timer register set should be programmed to provide a request for refresh about every 15 μs.

Table 18. Terminal Count/EOP Summary Table

Conditions AUTOINIT	No		Yes	
Event				
Word Counter Expired	Yes	X	Yes	X
EOP Input	X	Asserted	X	Asserted
Result				
Status TC	set	set	set	set
Mask	set	set	—	—
SW Request	clr	clr	clr	clr
Current Register	—	—	load	load

NOTES:
load = load current from base
— = no change
X = don't care
clr = clear

5.4.8 SCATTER/GATHER DESCRIPTION

Scatter/Gather (S/G) provides the capability of transferring multiple buffers between memory and I/O without CPU intervention. In Scatter/Gather, the DMA can read the memory address and word count from an array of buffer descriptors, located in system memory (ISA or PCI), called the Scatter/Gather Descriptor (SGD) Table. This allows the DMA controller to sustain DMA transfers until all of the buffers in the SGD Table are transferred.

The S/G Command and Status Registers are used to control the operational aspect of S/G transfers. The SGD Table Pointer Register holds the address of the next buffer descriptor in the SGD Table.

The next buffer descriptor is fetched from the SGD Table by a DMA read transfer. DACK# will not be asserted for this transfer because the IO device is the SIO itself. The SIO will fetch the next buffer descriptor from either PCI memory or ISA memory, depending on where the SGD Table is located. If the SGD table is located in PCI memory, the memory read will use the line buffer to temporarily store the PCI read before loading it into the DMA S/G registers. The line buffer mode (8 byte or single transaction) for the S/G fetch operation will be the same as what is set for all DMA operations. If set in 8 byte mode, the SGD Table fetches will be PCI burst memory reads. The SGD Table PCI cycle fetches are subject to all types of PCI cycle termination (retry, disconnect, target-abort, master-abort). The fetched SGD Table data is subject to normal line buffer coherency management and invalidation. EOP will be asserted at the end of the complete link transfer.

To initiate a typical DMA Scatter/Gather transfer between memory and an I/O device, the following steps are required:

1. Software prepares a SGD Table in system memory. Each SGD is 8 bytes long and consists of an address pointer to the starting address and the transfer count of the memory buffer to be transferred. In any given SGD Table, two consecutive SGDs are offset by 8 bytes and are aligned on a 4-byte boundary.

   Each Scatter/Gather Descriptor for the linked list contains the following information:

   a. Memory Address (buffer start) 4 bytes

   b. Transfer Size (buffer size) 2 bytes

   c. End of Link List 1 bit (MSB)
2. Initialize the DMA Channel Mode and DMA Channel Extended Mode Registers with transfer specific information like 8-/16-bit I/O device, Transfer Mode, Transfer Type, etc.
3. Software provides the starting address of the SGD Table by loading the SGD Table Pointer Register.
4. Engage the Scatter/Gather function by writing a Start command to the S/G Command Register.
5. The Mask register should be cleared as the last step of programming the DMA register set. This is to prevent the DMA from starting a transfer with a partially loaded command description.
6. Once the register set is loaded and the channel is unmasked, the DMA will generate an internal request to fetch the first buffer from the SGD Table.

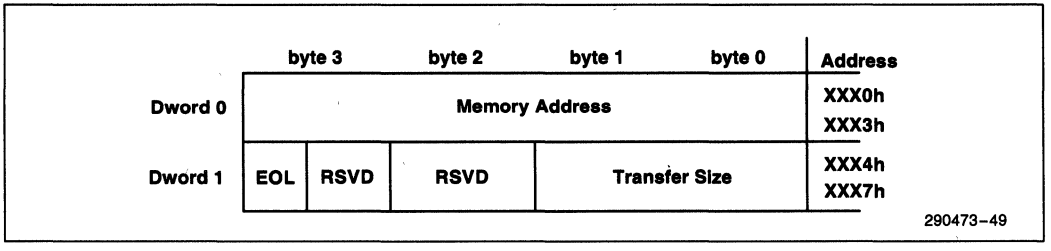


Figure 10. SGD Format

After the above steps are finished, the DMA will then respond to DREQ or software requests. The first transfer from the first buffer moves the memory address and word count from the Base register set to the Current register set. As long as S/G is active and the Base register set is not loaded and the last buffer has not been fetched, the channel will generate a request to fetch a reserve buffer into the Base register set. The reserve buffer is loaded to minimize latency problems going from one buffer to another. Fetching a reserve buffer has a lower priority than completing DMA transfers for the channel.

The DMA controller will terminate a Scatter/Gather cycle by detecting an End of List (EOL) bit in the SGD Table. After the EOL bit is detected, the channel transfers the buffers in the Base and Current register sets, if they are loaded. At terminal count the channel asserts EOP or IRQ13, depending on its programming and set the terminate bit in the S/G Status Register. If the channel asserted IRQ13, then the appropriate bit is set in the S/G Interrupt Status Register. The active bit in the S/G Status Register will be reset and the channel's Mask bit will be set.

5.5 Address Decoding

The SIO contains two address decoders; one to decode PCI master cycles and one to decode DMA/ISA master cycles. Two decoders are required to support the PCI and ISA Buses running concurrently. The PCI address decoder decodes the address from the multiplexed PCI address/data bus. The DMA/ISA master address decoder decodes the address from the ISA address bus for DMA and ISA master cycles. The address decoders determine how the cycle is handled.

5.5.1 PCI ADDRESS DECODER

PCI address decoding is always a function of AD[31:2]. The information contained in the two low order bits (AD[1:0]) varies for memory, I/O, and configuration cycles.

2

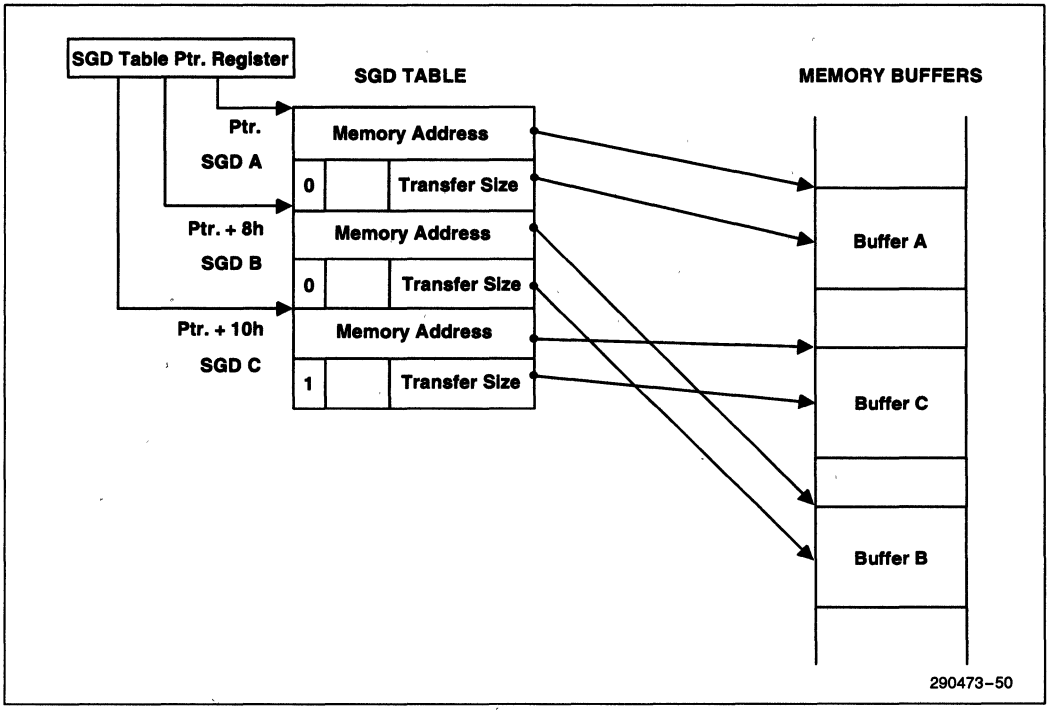


Figure 11. Link List Example

For I/O cycles, AD[31:0] are all decoded to provide a byte address. The byte enables determine which byte lanes contain valid data. The SIO requires that PCI accesses to byte-wide internal registers must assert only one byte enable.

For memory cycles, accesses are decoded as Dword accesses. This means that AD[1:0] are ignored for decoding memory cycles. The byte enables are used only to determine which byte lanes contain valid data.

For configuration cycles, DEVSEL# is a function of IDSEL# and AD[1:0]. DEVSEL# is generated only when AD[1:0] are both zero. If either AD[1:0] are non-zero, the cycle is ignored by the SIO. Individual bytes of a configuration register can be accessed with the byte enables. A particular configuration register is selected using AD[7:2]. Again, the byte enables determine which byte lanes contain valid data.

All PCI cycles decoded in one of the following ways result in the SIO generating DEVSEL#. The PCI master cycle decoder decodes the following addresses based on the settings of the relevant configuration registers:

SIO I/O Addresses: Positively decodes I/O addresses for registers contained within the SIO (exceptions: 60h, 92h, 3F2h, 372h, and F0h).

BIOS Memory Space: Positively decodes BIOS memory space.

MEMCS# Address Decoding: Decodes memory addresses that reside on the other side of the Host bridge and generates the MEMCS# signal. (SIO does not generate DEVSEL# in this case). The address range(s) used for this decoding is selected via the MEMCSCON, MEMCSBOH, MEMCSTOH, MEMCSTOM, MAR1, MAR2, and MAR3 Registers (see Section 4.1).

Subtractively Decoding Cycles to ISA: Subtractively decodes cycles to the ISA Bus. Accesses to registers 60h, 92h, 3F2h, 372h, and F0h are also subtractively decoded to the ISA Bus.

One of the PCI requirements is that, upon power-up, PCI agents do not respond to any address. Typically, the only access to a PCI agent is through the IDSEL configuration mechanism until it is enabled through configuration. The SIO is an exception to this, since it controls access to the BIOS boot code. All addresses decoded by the PCI address decoder, that are enabled after chip reset, are accessible immediately after power-up.

5.5.1.1 SIO I/O Addresses

These addresses are the internal, non-configuration SIO register locations and are shown in the SIO Address Decoding Table, Table 19. These addresses are fixed. Note that the Configuration Registers, listed in Table 3, are accessed with PCI configuration cycles as described in Section 5.1.2.5

In general, PCI accesses to the internal SIO registers will not be broadcast to the ISA Bus. However, PCI accesses to addresses 70h, 60h, 92h, 3F2h, 372h, and F0h are exceptions. Read and write accesses to these SIO locations are broadcast onto the ISA Bus. PCI master accesses to SIO registers will be retried if the ISA Bus is owned by an ISA master or the DMA controller. All of the registers are 8 bit registers. Accesses to these registers must be 8 bit accesses. Target-abort is issued by the SIO when the internal SIO non-configuration registers are the target of a PCI master I/O cycle and more than one byte enable is active. Refer to Table 19 for the SIO Address Decoding Map.

Accesses to the BIOS Timer Register (78h–7Bh) are broadcast to the ISA bus only if this register is disabled. If this register is enabled, the cycle is not broadcast to the ISA bus.

The address decoding logic includes the read/write cycle type. For example, read cycles to write only registers are not positively decoded and get forwarded to the ISA bus via subtractive decoding.

Table 19. SIO Address Decoding

Address	Address				Type	Name	Block
	FEDC	BA98	7654	3210			
0000h	0000	0000	000x	0000	r/w	DMA1 CH0 Base and Current Address	DMA
0001h	0000	0000	000x	0001	r/w	DMA1 CH0 Base and Current Count	DMA
0002h	0000	0000	000x	0010	r/w	DMA1 CH1 Base and Current Address	DMA
0003h	0000	0000	000x	0011	r/w	DMA1 CH1 Base and Current Count	DMA
0004h	0000	0000	000x	0100	r/w	DMA1 CH2 Base and Current Address	DMA
0005h	0000	0000	000x	0101	r/w	DMA1 CH2 Base and Current Count	DMA
0006h	0000	0000	000x	0110	r/w	DMA1 CH3 Base and Current Address	DMA
0007h	0000	0000	000x	0111	r/w	DMA1 CH3 Base and Current Count	DMA
0008h	0000	0000	000x	1000	r/w	DMA1 Status(r) Command(w) Register	DMA
0009h	0000	0000	000x	1001	wo	DMA1 Write Request Register	DMA
000Ah	0000	0000	000x	1010	wo	DMA1 Write Single Mask Bit	DMA
000Bh	0000	0000	000x	1011	wo	DMA1 Write Mode Register	DMA
000Ch	0000	0000	000x	1100	wo	DMA1 Clear Byte Pointer	DMA
000Dh	0000	0000	000x	1101	wo	DMA1 Master Clear	DMA
000Eh	0000	0000	000x	1110	wo	DMA1 Clear Mask Register	DMA
000Fh	0000	0000	000x	1111	r/w	DMA1 Read/Write All Mask Register Bits	DMA
0020h	0000	0000	001x	xx00	r/w	INT 1 Control Register	PIC
0021h	0000	0000	001x	xx01	r/w	INT 1 Mask Register	PIC
0040h	0000	0000	010x	0000	r/w	Timer Counter 1—Counter 0 Count	TC
0041h	0000	0000	010x	0001	r/w	Timer Counter 1—Counter 1 Count	TC
0042h	0000	0000	010x	0010	r/w	Timer Counter 1—Counter 2 Count	TC
0043h	0000	0000	010x	0011	wo	Timer Counter 1 Command Mode Register	TC
0060h	0000	0000	0110	0000	ro	Reset UBus IRQ12	Control
0061h	0000	0000	0110	0xx1	r/w	NMI Status and Control	Control
0070h	0000	0000	0111	0xx0	wo	CMOS RAM Address and NMI Mask Register	Control
0078h(1)	0000	0000	0111	10xx	r/w	BIOS Timer	TC

Table 19. SIO Address Decoding (Continued)

Address	Address				Type	Name	Block
	FEDC	BA98	7654	3210			
0080h	0000	0000	100x	0000	r/w	DMA Page Register (Reserved)	DMA
0081h	0000	0000	100x	0001	r/w	DMA Channel 2 Page Register	DMA
0082h	0000	0000	1000	0010	r/w	DMA Channel 3 Page Register	DMA
0083h	0000	0000	100x	0011	r/w	DMA Channel 1 Page Register	DMA
0084h	0000	0000	100x	0100	r/w	DMA Page Register (Reserved)	DMA
0085h	0000	0000	100x	0101	r/w	DMA Page Register (Reserved)	DMA
0086h	0000	0000	100x	0110	r/w	DMA Page Register (Reserved)	DMA
0087h	0000	0000	100x	0111	r/w	DMA Channel 0 Page Register	DMA
0088h	0000	0000	100x	0100	r/w	DMA Page Register (Reserved)	DMA
0089h	0000	0000	100x	1001	r/w	DMA Channel 6 Page Register	DMA
008Ah	0000	0000	100x	1010	r/w	DMA Channel 7 Page Register	DMA
008Bh	0000	0000	100x	1011	r/w	DMA Channel 5 Page Register	DMA
008Ch	0000	0000	100x	1100	r/w	DMA Page Register (Reserved)	DMA
008Dh	0000	0000	100x	1101	r/w	DMA Page Register (Reserved)	DMA
008Eh	0000	0000	100x	1110	r/w	DMA Page Register (Reserved)	DMA
008Fh	0000	0000	100x	1111	r/w	DMA Low Page Register Refresh	DMA
0090h	0000	0000	100x	0000	r/w	DMA Page Register (Reserved)	DMA
0092h	0000	0000	1001	0010	r/w	System Control Port	Control
0094h	0000	0000	100x	0100	r/w	DMA Page Register (Reserved)	DMA
0095h	0000	0000	100x	0101	r/w	DMA Page Register (Reserved)	DMA
0096h	0000	0000	100x	0110	r/w	DMA Page Register (Reserved)	DMA
0098h	0000	0000	100x	1000	r/w	DMA Page Register (Reserved)	DMA
009Ch	0000	0000	100x	1100	r/w	DMA Page Register (Reserved)	DMA
009Dh	0000	0000	100x	1101	r/w	DMA Page Register (Reserved)	DMA
009Eh	0000	0000	100x	1110	r/w	DMA Page Register (Reserved)	DMA
009Fh	0000	0000	100x	1111	r/w	DMA low page Register Refresh	DMA
00A0h	0000	0000	101x	xx00	r/w	INT 2 Control Register	PIC
00A1h	0000	0000	101x	xx01	r/w	INT 2 Mask Register	PIC
00B2h	0000	0000	1011	0010	r/w	Advanced Power Management Control Port	PM
00B3h	0000	0000	1011	0011	r/w	Advanced Power Management Status Port	PM
00C0h	0000	0000	1100	000x	r/w	DMA2 CH0 Base and Current Address	DMA

Table 19. SIO Address Decoding (Continued)

Address	Address				Type	Name	Block
	FEDC	BA98	7654	3210			
00C2h	0000	0000	1100	001x	r/w	DMA2 CH0 Base and Current Count	DMA
00C4h	0000	0000	1100	010x	r/w	DMA2 CH1 Base and Current Address	DMA
00C6h	0000	0000	1100	011x	r/w	DMA2 CH1 Base and Current Count	DMA
00C8h	0000	0000	1100	100x	r/w	DMA2 CH2 Base and Current Address	DMA
00CAh	0000	0000	1100	101x	r/w	DMA2 CH2 Base and Current Count	DMA
00CCh	0000	0000	1100	110x	r/w	DMA2 CH3 Base and Current Address	DMA
00CEh	0000	0000	1100	111x	r/w	DMA2 CH3 Base and Current Count	DMA
00D0h	0000	0000	1101	000x	r/w	DMA2 Status(r) Command(w) Register	DMA
00D2h	0000	0000	1101	001x	wo	DMA2 Write Request Register	DMA
00D4h	0000	0000	1101	010x	wo	DMA2 Write Single Mask Bit	DMA
00D6h	0000	0000	1101	011x	wo	DMA2 Write Mode Register	DMA
00D8h	0000	0000	1101	100x	wo	DMA2 Clear Byte Pointer	DMA
00DAh	0000	0000	1101	101x	wo	DMA2 Master Clear	DMA
00DCh	0000	0000	1101	110x	wo	DMA2 Clear Mask Register	DMA
00DEh	0000	0000	1101	111x	r/w	DMA2 Read/Write All Mask Register Bits	DMA
00F0h	0000	0000	1111	0000	wo	Coprocessor Error	Control
0372h	0000	0011	0111	0010	wo	Secondary Floppy Disk Digital Output Reg.	Control
03F2h	0000	0011	1111	0001	wo	Primary Floppy Disk Digital Output Reg.	Control
040Ah	0000	0100	0000	1010	ro	Scatter/Gather Interrupt Status Register	DMA
040Bh	0000	0100	0000	1011	wo	DMA1 Extended Mode register	DMA
0410h(1)	0000	0100	0001	0000	wo	CH0 Scatter/Gather Command	DMA
0411h(1)	0000	0100	0001	0001	wo	CH1 Scatter/Gather Command	DMA
0412h(1)	0000	0100	0001	0010	wo	CH2 Scatter/Gather Command	DMA
0413h(1)	0000	0100	0001	0011	wo	CH3 Scatter/Gather Command	DMA
0415h(1)	0000	0100	0001	0101	wo	CH5 Scatter/Gather Command	DMA
0416h(1)	0000	0100	0001	0110	wo	CH6 Scatter/Gather Command	DMA
0417h(1)	0000	0100	0001	0111	wo	CH7 Scatter/Gather Command	DMA
0418h(1)	0000	0100	0001	1000	ro	CH0 Scatter/Gather Status	DMA
0419h(1)	0000	0100	0001	1001	ro	CH1 Scatter/Gather Status	DMA
041Ah(1)	0000	0100	0001	1010	ro	CH2 Scatter/Gather Status	DMA
041Bh(1)	0000	0100	0001	1011	ro	CH3 Scatter/Gather Status	DMA

Table 19. SIO Address Decoding (Continued)

Address	Address				Type	Name	Block
	FEDC	BA98	7654	3210			
041Dh(1)	0000	0100	0001	1101	ro	CH5 Scatter/Gather Status	DMA
041Eh(1)	0000	0100	0001	1110	ro	CH6 Scatter/Gather Status	DMA
041Fh(1)	0000	0100	0001	1111	ro	CH7 Scatter/Gather Status	DMA
0420h(1)	0000	0100	0010	00xx	r/w	CH0 Scatter/Gather Descriptor Table Pointer	DMA
0424h(1)	0000	0100	0010	01xx	r/w	CH1 Scatter/Gather Descriptor Table Pointer	DMA
0428h(1)	0000	0100	0010	10xx	r/w	CH2 Scatter/Gather Descriptor Table Pointer	DMA
042Ch(1)	0000	0100	0010	11xx	r/w	CH3 Scatter/Gather Descriptor Table Pointer	DMA
0434h(1)	0000	0100	0011	01xx	r/w	CH5 Scatter/Gather Descriptor Table Pointer	DMA
0438h(1)	0000	0100	0011	10xx	r/w	CH6 Scatter/Gather Descriptor Table Pointer	DMA
043Ch(1)	0000	0100	0011	11xx	r/w	CH7 Scatter/Gather Descriptor Table Pointer	DMA
0481h	0000	0100	1000	0001	r/w	DMA CH2 High Page Register	DMA
0482h	0000	0100	1000	0010	r/w	DMA CH3 High Page Register	DMA
0483h	0000	0100	1000	0011	r/w	DMA CH1 High Page Register	DMA
0487h	0000	0100	1000	0111	r/w	DMA CH0 High Page Register	DMA
0489h	0000	0100	1000	1001	r/w	DMA CH6 High Page Register	DMA
048Ah	0000	0100	1000	1010	r/w	DMA CH7 High Page Register	DMA
048Bh	0000	0100	1000	1011	r/w	DMA CH5 High Page Register	DMA
04D0	0000	0100	1101	0000	r/w	INT CNTRL-1 Edge Level Control Register	Control
04D1	0000	0100	1101	0001	r/w	INT CNTRL-2 Edge Level Control Register	Control
04D6h	0000	0100	1101	0010	wo	DMA2 Extended Mode Register	DMA

NOTE:
1. The I/O address of this register is relocatable. The value shown in this table is the default address location.

5.5.1.2 BIOS Memory Space

The 128 Kb BIOS memory space is located at 000E0000h to 000FFFFFh (top of 1 Mb), and is aliased at FFFE0000h to FFFFFFFFh (top of 4 Gb) and FFEE0000h to FFEFFFFFh (top of 4 Gb-1 Mb). The aliased regions account for the CPU reset vector and the uncertainty of the state of A20GATE when a software reset occurs. This 128 Kb block is split into two 64 Kb blocks. The top 64 Kb is always enabled while the bottom 64 Kb can be enabled or disabled (the aliases automatically match). Enabling the lower 64 Kb BIOS space (000E0000h to 000EFFFFh, 896 Kb-960 Kb) results in positively decoding this region and enables the BIOSCS# signal generation. The upper 64 Kb is positively decoded only if bit 6 = 1 in the ISA Clock Divisor Register. Otherwise this region is subtractively decoded. Positively decoding these cycles expedites BIOS cycles to the ISA Bus. Note that both of these regions are subtractively decoded if bit 4 in the MEMCS# Control Register is set to a 1.

When PCI master accesses to the 128 Kb BIOS space at 4 Gb-1 Mb are forwarded to the ISA Bus, the LA20 line is driven to a 1 to avoid aliasing at the 15 Mb area. The 4 Gb-1 Mb BIOS decode area accounts for the condition when A20M# is asserted and an ALT-CTRL-DEL reset is generated. The CPU's reset vector will access 4 Gb-1 Mb. When this gets forwarded to ISA, AD[32:24] are truncated and the access is aliased to 16 Mb-1 Mb = 15 Mb space. If ISA memory is present at 15 Mb, there will be contention. Forcing LA20 high aliases this region to 16 Mb. The alias here is permissible since this is the 80286 reset vector location.

In addition to the normal 128 Kb BIOS space, the SIO supports an additional 384 Kb BIOS space. The SIO can support a total of 512 Kb BIOS space. The additional 384 Kb region can only be accessed by PCI masters and resides at FFF80000h to FFFDFFFFh. When enabled via the UBCSA Register, memory accesses within this region will be positively decoded, forwarded to the ISA Bus, and encoded BIOSCS# will be generated. When forwarded to the ISA Bus, the PCI AD[23:20] signals will be propagated to the ISA LA[23:20] lines as all 1's which will result in aliasing this 512 Kb region at the top of the 16 Mb space. To avoid contention, ISA add-in memory must not be present in this space.

All PCI cycles positively decoded in the enabled BIOS space will be broadcast to the ISA Bus. Since the BIOS device is 8 or 16 bits wide and typically has very long access times, PCI burst reads from the BIOS space will invoke "disconnect target termination" semantics after the first data transaction in order to meet the PCI incremental latency guidelines.

The following tables and diagrams describe the operation of the SIO in response to PCI BIOS space accesses.

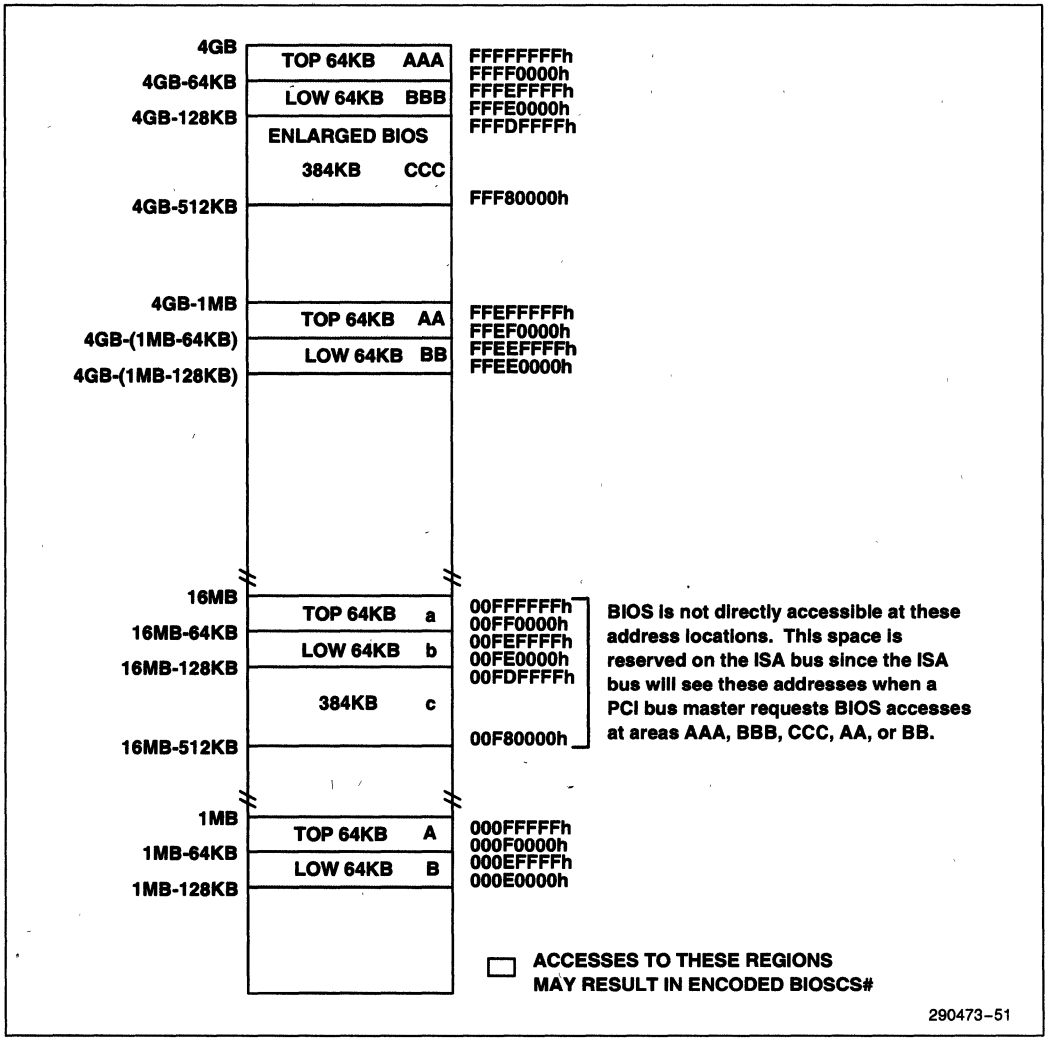


Figure 12. BIOS Space Decode Map

The BIOS space decode map, Figure 12, shows the possible BIOS spaces and the aliases throughout the memory space. The various regions are designated with code letters; "a's" for the top 64 Kb, "b's" for the low 64 Kb, and "c's" for the enlarged space.

Table 20 indicates the SIO's response to PCI BIOS space accesses based on its configuration state.

Table 20. PCI Master BIOS Space Decoding

Master	Region	Top 64 Kb BIOS Positive Decode Enabled(1)	Low 64 Kb BIOS Enabled(2)	Enlarged BIOS Enabled(3)	Encoded BIOSCS# Generated	LA20	Positive PCI Decode	Subtractive PCI Decode
PCI	A	0	x	x	Yes	Pass (0)	No	Yes
PCI	A	1	x	x	Yes	Pass (0)	Yes(5)	No(5)
PCI	B	x	0	x	No	Pass (0)	No	Yes
PCI	B	x	1	x	Yes	Pass (0)	Yes(5)	No(5)
PCI	a	1	x	x	No	Pass (1)	No	Yes
PCI	b	x	0	x	No	Pass (1)	No	Yes
PCI	c	x	x	0	No	Pass (1)	No	Yes
PCI	AA	0	x	x	Yes	1	No	Yes(4)
PCI	AA	1	x	x	Yes	1	Yes(4,5)	No(5)
PCI	BB	x	0	x	No	x	No	No
PCI	BB	x	1	x	Yes	1	Yes(4,5)	No(5)
PCI	AAA	0	x	x	Yes	Pass (1)	No	Yes(4)
PCI	AAA	1	x	x	Yes	Pass (1)	Yes(4,5)	No(5)
PCI	BBB	x	0	x	No	x	No	No
PCI	BBB	x	1	x	Yes	Pass (1)	Yes(4,5)	No(5)
PCI	CCC	x	x	0	No	x	No	No
PCI	CCC	x	x	1	Yes	Pass (1)	Yes(4)	No

NOTES:
1. The column labeled "Top 64 Kb BIOS Positive Decode Enable" shows the value of the ISA Clock Register bit 6. This bit determines the decoding for memory regions A, AA, and AAA (1 = positive, 0 = negative decoding).
   Note that if bit 4 in the MEMCS# Control Register is set to a 1 (Global MEMCS# decode enabled), the positive decoding function enabled by having ISA Clock Register bit 6 = 1 is ignored. Subtractive decoding is provided, instead.
2. The column labeled "Low 64 Kb BIOS Enable" shows the value of the Utility Bus Chip Select Enable A Bit 6. This bit determines whether memory regions B, BB, and BBB are enabled (bit = 1) or disabled (bit = 0).
3. The column labeled "Enlarged BIOS Enabled" shows the value of the Utility Bus Chip Select Enable A Bit 7. This bit determines whether memory region CCC is enabled (bit = 1) or disabled (bit = 0).
4. ISA memory is not allowed to be enabled at the corresponding aliased areas or contention will result.
5. When bit 4 in the MEMCS# Control Register is set to a 1 (Global MEMCS# decode enabled), positive decoding for these areas will be disabled. The SIO will only provide subtractive decoding in this case.

5.5.1.3 MEMCS# Decoding

For MEMCS# decoding, the SIO decodes sixteen ranges. Fourteen of these ranges can be enabled or disabled independently for both read and write cycles. The fifteenth range (0 KB–512 KB) and sixteenth range (programmable from 1 MB up to 512 MB in 2 MB increments) can be enabled or disabled only. Addresses within these enabled regions generate a MEMCS# signal that can be used by the host bridge to know when to forward PCI cycles to main memory. A seventeenth range is available that can be used to identify a "memory hole". Addresses within this hole will not generate a MEMCS#. The address regions are summarized:

- 0 KB to 512 KB Memory (can only be disabled if MEMCS# is completely disabled)
- 512 KB to 640 KB Memory
- (1 MB–64 KB) to 1 MB Memory (BIOS Area)
- 768 KB to 918 KB in 16 KB sections (total of 8 sections)
- 918 KB to 983 KB in 16 KB sections (total of 4 sections)
- 1M-to-programmable boundary on 2 MB increments from 2 MB up to 512 MB
- programmable "memory hole" in 64 KB increments between 1 MB and 16 MB

Table 21 and Figure 13 show the registers and decode areas for MEMCS#.

Table 21. MEMCS# Decoding Register Summary

MAR Registers	Attribute	Memory Segments	Comments
MCSCON[4] = 0	Disable	Disable MEMCS# Function	Enable/Disable MEMCS# Function
MCSCON[4] = 1	Enable	Enable MEMCS# Function	When Enabled, 0 KB to 512 KB Range is also Automatically Enabled (RE/WE)
MCSTOH/ MCSBOH	MEMCS# Hole	100000h–0FFFFFFh	1 MB to 16 MB Hole in MEMCS# Region
MCSTOM	MEMCS# Top	200000h–1FFFFFFFh	2 MB to 512 MB Top of MEMCS# Region
MCSCON[1:0]	[0] = RE[1] = WE	080000h–09FFFFh	512 KB to 640 KB R/W Enable
MCSCON[3:2]	[2] = RE[3] = WE	0F0000h–0FFFFFh	BIOS Area R/W Enable
MAR1[1:0]	[0] = RE[1] = WE	0C0000h–0C3FFFh	ISA Add-On BIOS R/W Enable
MAR1[3:2]	[2] = RE[3] = WE	0C4000h–0C7FFFh	ISA Add-On BIOS R/W Enable
MAR1[5:4]	[4] = RE[5] = WE	0C8000h–0CBFFFh	ISA Add-On BIOS R/W Enable
MAR1[7:6]	[6] = RE[7] = WE	0CC000h–0CFFFFh	ISA Add-On BIOS R/W Enable
MAR2[1:0]	[0] = RE[1] = WE	0D0000h–0D3FFFh	ISA Add-On BIOS R/W Enable
MAR2[3:2]	[2] = RE[3] = WE	0D4000h–0D7FFFh	ISA Add-On BIOS R/W Enable
MAR2[5:4]	[4] = RE[5] = WE	0D8000h–0DBFFFh	ISA Add-On BIOS R/W Enable
MAR2[7:6]	[6] = RE[7] = WE	0DC000h–0DFFFFh	ISA Add-On BIOS R/W Enable
MAR3[1:0]	[0] = RE[1] = WE	0E0000h–0E3FFFh	BIOS Extension R/W Enable
MAR3[3:2]	[2] = RE[3] = WE	0E4000h–0E7FFFh	BIOS Extension R/W Enable
MAR3[5:4]	[4] = RE[5] = WE	0E8000h–0EBFFFh	BIOS Extension R/W Enable
MAR3[7:6]	[6] = RE[7] = WE	0EC000h–0EFFFFh	BIOS Extension R/W Enable

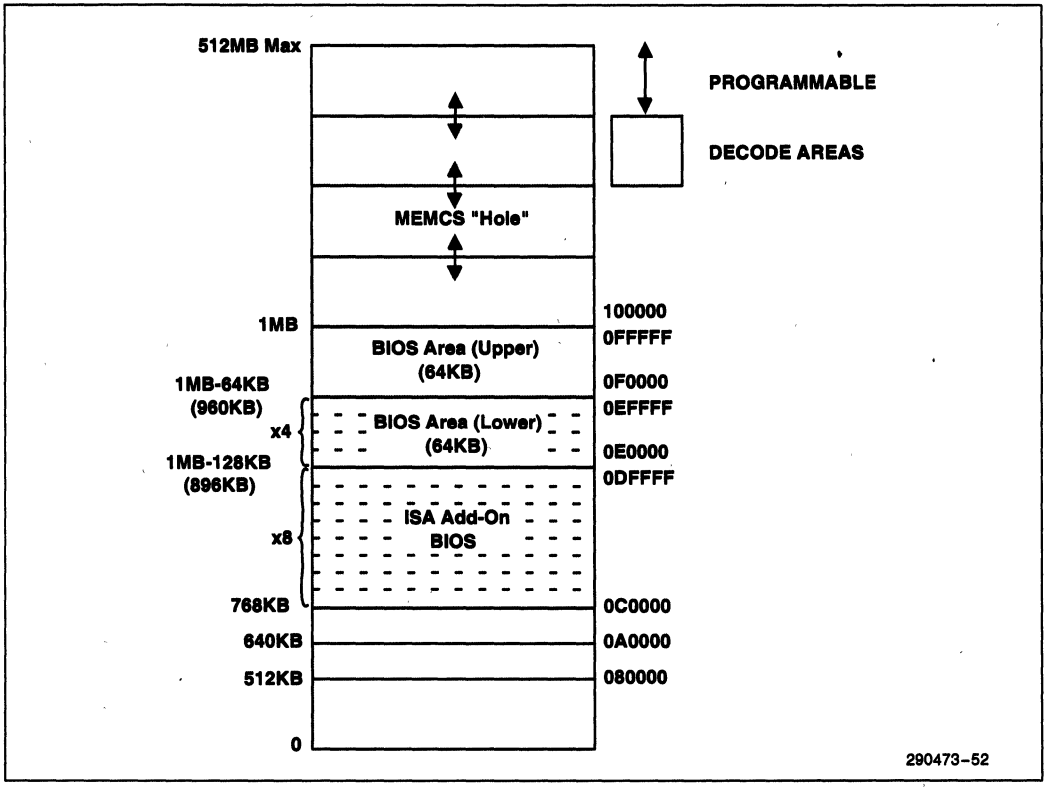


Figure 13. MEMCS# Decode Areas

The SIO generates MEMCS# from the PCI address. MEMCS# is generated from the clock edge after FRAME# is sampled active. MEMCS# will only go active for one PCI clock period. The SIO does not take any other action as a result of this decode other than generating MEMCS#. It is the responsibility of the device using the MEMCS# signal to generate DEVSEL#, TRDY# and any other cycle response. The device using MEMCS# will always generate DEVSEL# on the next clock. This fact can be used to avoid an extra clock delay in the subtractive decoder described in the next section.

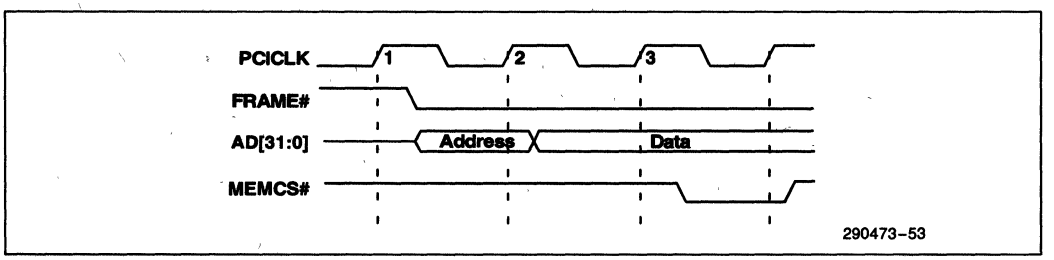


Figure 14. MEMCS# Generation

5.5.1.4 Subtractively Decoded Cycles to ISA

The addresses that reside on the ISA Bus could be highly fragmented. For this reason, subtractive decoding is used to forward PCI cycles to the ISA Bus. An inactive DEVSEL# will cause the SIO to forward the PCI cycle to the ISA Bus. The DEVSEL# sample point can be configured for three different settings. If the "fast" point is selected, the cycle will be forwarded to ISA when DEVSEL# is inactive at the F sample point as shown in Figure 15. If the "typical" point is selected, DEVSEL# will be sampled on both F and T, and if inactive, will be forwarded to the ISA Bus. Likewise, if the "slow" point is selected, DEVSEL# will be sampled at F, T, and S. The sample point should be configured to match the slowest PCI device in the system. This capability reduces the latency to ISA slaves when all PCI devices are "fast" and also allows for devices with slow decoding. Note that when these unclaimed cycles are forwarded to the ISA Bus, the SIO will drive the DEVSEL# active.

Since an active MEMCS# will always result in an active DEVSEL# at the "Slow" sample point, MEMCS# is used as an early indication of DEVSEL#. In this case, if the device using MEMCS# is the only "slow" agent in the system, the sample point can be moved in to the "typical" edge.

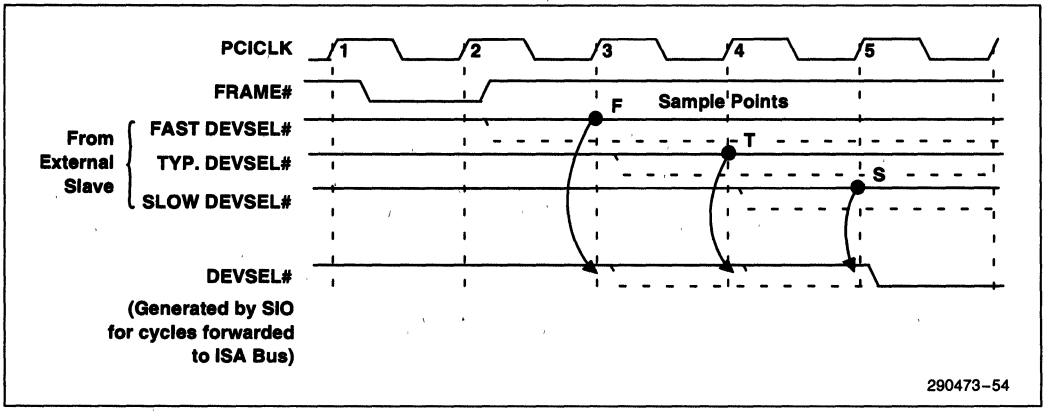


Figure 15. DEVSEL# Generation

Unclaimed PCI cycles with memory addresses above 16M and I/O addresses above 64K will not be forwarded to the ISA Bus. The SIO will not respond with DEVSEL# (BIOS accesses are an exception to this). This is required to avoid the possibility of aliasing. Under this condition, these unclaimed cycles will be recognized as such by the originating master and the master will use "master-abort" semantics to terminate the PCI cycle.

5.5.2 DMA/ISA MASTER CYCLE ADDRESS DECODER

The SIO also contains a decoder which is used to determine the destination of ISA master and DMA master cycles. This decoder provides:

Positive Decode to PCI: Positively decodes addresses to be forwarded to the PCI Bus. This includes addresses residing directly on PCI as well as addresses that reside on the back side of PCI bridges (Host Bridges).

Access to SIO Internal Registers: Positively decodes addresses to registers within the SIO.

BIOS Accesses: Positively decodes BIOS memory accesses and generates encoded BIOSCS#.

Utility Bus Chip Selects: Positively decodes utility bus chip selects.

Subtractive Decode: Subtractively decodes cycles to be contained to the ISA Bus.

5.5.2.1 Positive Decode to PCI

ISA master or DMA addresses that are positively decoded by this decoder will be propagated to the PCI Bus. This is the only way to forward a cycle from an ISA master or the DMA to the PCI Bus. If the cycle is not decoded by this decoder it will *not* be forwarded to the PCI Bus.

This decoder has several memory address regions to positively decode cycles that should be forwarded to the PCI Bus. These regions are listed below. Regions "a" through "e" are fixed and can be enabled or disabled independently. Region "f" defines a space starting at 1M with a programmable upper boundary up to 16 MB. Within this region a hole can be opened. Its size and location are programmable to allow a hole to be opened in the memory space. A memory address above 16 MB will be forwarded to the PCI Bus automatically. This is possible only during DMA cycles in which the DMA has been programmed for 32-bit addressing above 16 MB.

a. Memory: 0 KB–512 KB

b. Memory: 512 KB–640 KB

c. Memory: 640 KB–768 KB (Video buffer)

d. Memory: 768 KB–896 KB in eight 16K sections (Expansion ROM)

e. Memory: 896 KB–960 KB (lower BIOS area)

f. Memory: 1 MB-to-X MB (up to 16 MB) within which a hole can be opened. Accesses to the hole are not forwarded to PCI. The top of the region can be programmed on 64 KByte boundaries up to 16 MB. The hole can be between 64 KB and 8 MB in size in 64 KB increments located on any 64 KB boundary. (Refer to the ISA Address Decoder Register in the register description section, Section 5.5.2)

g. Memory: > 16 MB automatically forwarded to PCI

Figure 16 shows a map of the ISA master/DMA decode regions and Table 22 summarizes the registers used to configure the decoder.

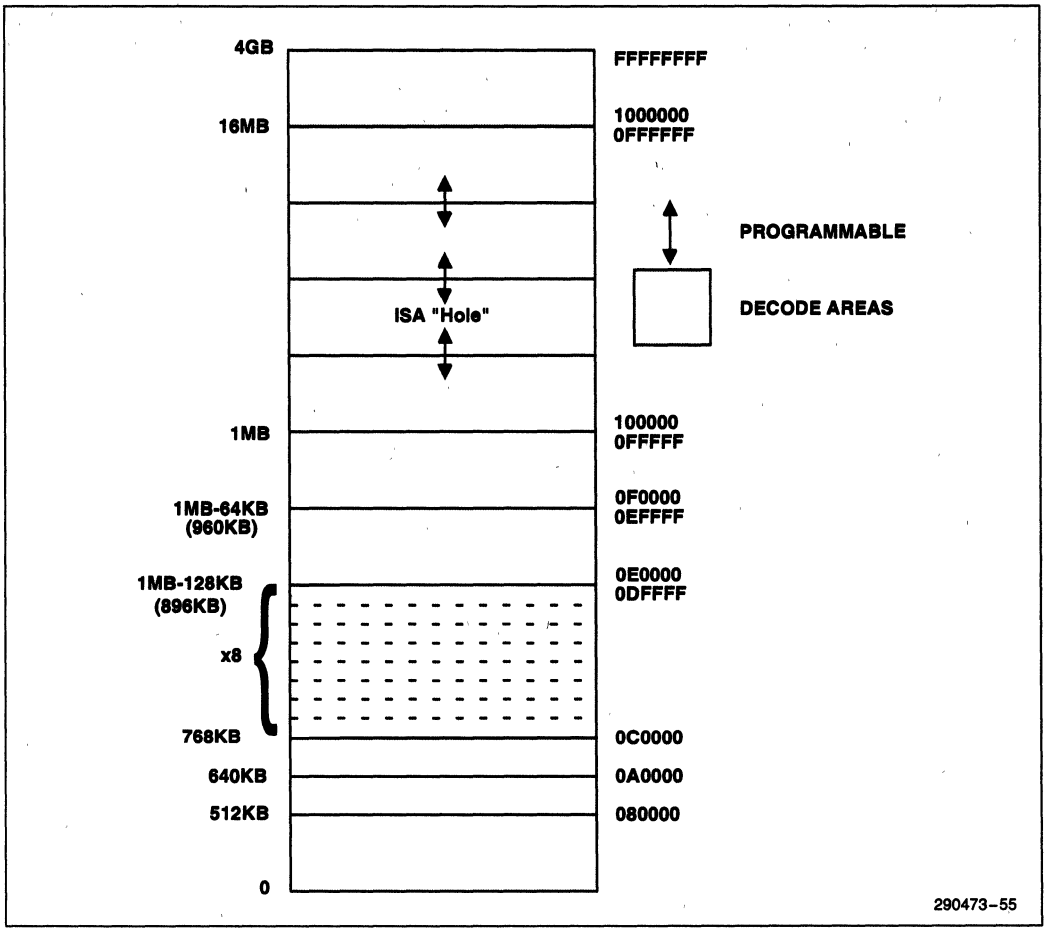


Figure 16. ISA Master/DMA to PCI Bus Decoder Regions

Table 22. ISA Master/DMA to PCI Bus Decoding Register Summary

MAR Registers	Attribute	Memory Segments	Comments
IADCON[7:4]	ISA Memory Top	100000h–0FFFFFFh	1 MB to 16 MB Top of ISA Region
IADTOH/IADBOH	ISA Hole	100000h–0FFFFFFh	1 MB to 16 MB Hole in ISA Region
IADCON[0]	Enable/Disable	000000h–07FFFFh	0 to 512 KB Enable/Disable
IADCON[1]	Enable/Disable	080000h–09FFFFh	512 KB to 640 KB Enable/Disable
IADCON[2]	Enable/Disable	0A0000h–0BFFFFh	640 KB to 768 KB Enable/Disable
IADCON[3]*	Enable/Disable	0E0000h–0EFFFFh	896 KB to 960 KB Lower BIOS Enable/Disable
IADRBE[0]	Enable/Disable	0C0000h–0C3FFFh	ISA Add-On BIOS (Expansion ROM) Enable
IADRBE[1]	Enable/Disable	0C4000h–0C7FFFh	ISA Add-On BIOS (Expansion ROM) Enable
IADRBE[2]	Enable/Disable	0C8000h–0CBFFFh	ISA Add-On BIOS (Expansion ROM) Enable
IADRBE[3]	Enable/Disable	0CC000h–0CFFFFh	ISA Add-On BIOS (Expansion ROM) Enable
IADRBE[4]	Enable/Disable	0D0000h–0D3FFFh	ISA Add-On BIOS (Expansion ROM) Enable
IADRBE[5]	Enable/Disable	0D4000h–0D7FFFh	ISA Add-On BIOS (Expansion ROM) Enable
IADRBE[6]	Enable/Disable	0D8000h–0DBFFFh	ISA Add-On BIOS (Expansion ROM) Enable
IADRBE[7]	Enable/Disable	0DC000h–0DFFFFh	ISA Add-On BIOS (Expansion ROM) Enable

NOTE:
* This can be overridden by bit 6 of the UBCSA Register being set to a 1.

5.5.2.2 SIO Internal Registers

Most of the internal SIO registers are accessible by ISA masters. Table 19 lists the registers that are not accessible by ISA masters. Registers accessed by ISA masters are run as 8-bit extended I/O cycles.

5.5.2.3 BIOS Accesses

The 128K BIOS memory space is located at 000E0000h to 000FFFFFh, and is aliased at FFFE0000h to FFFFFFFFh (top of 4 GB) and FFEE0000h to FFEFFFFFh (top of 4 GB–1 MB). The aliased regions account for the CPU reset vector and the uncertainty of the state of A20GATE when a software reset occurs. This 128K block is split into two 64K blocks. The top 64K is always enabled while the bottom 64K can be enabled or disabled (the aliases automatically match). ISA masters can only access BIOS in the 000E0000 to 000FFFFFh region.

ISA originated accesses to the enabled 64K sections of the BIOS space (000E0000h–000FFFFFh) will activate the encoded BIOSCS# signal. ISA originated cycles will not be forwarded to the PCI Bus. Encoded BIOSCS# is combinatorially generated from the ISA, SA, and LA address bus. Encoded BIOSCS# is disabled during refresh and DMA cycles. The ISA Master/DMA BIOS Decoding Table indicates the SIO's response to BIOS accesses based on the configuration state.

Table 23. ISA Master/DMA BIOS Decoding

Cycle		SIO Configuration			SIO Response		
Master	Region(1)	Top 64 KB PCI Positive Decode Enabled(2)	Low 64 KB BIOS Enabled(3)	Forward Low 64 KB to PCI Enabled(4)	Encoded BIOSCS# Generated	Forward to PCI	Contain to ISA
ISA/DMA	A	x	x	x	Yes	No	Yes
ISA/DMA	B	x	0(5)	0	No	No	Yes
ISA/DMA	B	x	0(5)	1	No	Yes	No
ISA/DMA	B	x	1	x	Yes	No	Yes
ISA/DMA	a	These cycles will be forwarded to PCI dependent on the state of the ISA Address Decoder Configuration Registers. Encoded BIOSCS# will not be generated for any of these cycles.					
ISA/DMA	b						
ISA/DMA	c						

NOTES:
1. The memory sections referenced can be found in Figure 12.
2. The column labeled "Top 64 KB BIOS Positive Decode Enabled" shows the value of the ISA Clock Divisor Configuration Register bit 6. This bit determines how the memory region is decoded (0 = subtractively decoded, 1 = positively decoded).
3. The column labeled "Low 64 KB BIOS Enable" shows the value of the Utility Bus Chip Select Enable A Configuration Register bit 6. This bit determines if the memory region is enabled (bit = 1) or disabled (bit = 0).
4. The column labeled "Forward Low 64 KB to PCI Enables" shows the value of the ISA Address Decoder Control Configuration Register Bit 3. This bit determines whether PCI Bus forwarding is enabled (bit = 1) or disabled (bit = 0).
5. Forward to PCI if IADCON Bit 6 = 1.

5.5.2.4 Utility Bus Encoded Chip Selects

The SIO generates encoded chip selects for certain functions that are located on the utility bus (formerly X-Bus). The encoded chip selects are generated combinatorially from the ISA SA[15:0] address bus. The encoded chip selects are decoded externally (see Figure 19).

The encoded chip select table (Table 24) shows the addresses that result in encoded chip select generation. Chip selects can be enabled or disabled via configuration registers. In general, the addresses shown in Table 24 do not reside in the SIO itself. Write only addresses 70h, 372h, 3F2h are exceptions since particular bits from these registers reside in the SIO. For ISA master cycles, the SIO will respond to writes to address 70h, 372h, and 3F2h by generating IOCHRDY and writing to the appropriate bits.

Note that the SIO monitors read accesses to address 60h to support the mouse function. In this case, IOCHRDY is not generated.

Table 24. Encoded Chip Select Table

Address	Address				Type	Name	Encoded Chip Select
	FEDC	BA98	7654	3210			
0060h	0000	0000	0110	00x0	r/w	Keyboard Controller	KEYBRDCS#
0064h	0000	0000	0110	01x0	r/w	Keyboard Controller	KEYBRDCS#
0070h	0000	0000	0111	0xx0	w	Real Time Clock Address	RTCALE#

Table 24. Encoded Chip Select Table (Continued)

Address	Address				Type	Name	Encoded Chip Select
	FEDC	BA98	7654	3210			
0071h	0000	0000	0111	0xx1	r/w	Real Time Clock Data	RTCCS#
0170h	0000	0001	0111	0000	r/w	Secondary Data Register	IDECS0#
0171h	0000	0001	0111	0001	r/w	Secondary Error Register	IDECS0#
0172h	0000	0001	0111	0010	r/w	Secondary Sector Count Register	IDECS0#
0173h	0000	0001	0111	0011	r/w	Secondary Sector Number Register	IDECS0#
0174h	0000	0001	0111	0100	r/w	Secondary Cylinder Low Register	IDECS0#
0175h	0000	0001	0111	0101	r/w	Secondary Cylinder High Register	IDECS0#
0176h	0000	0001	0111	0110	r/w	Secondary Drive/Head Register	IDECS0#
0177h	0000	0001	0111	0111	r/w	Secondary Status Register	IDECS0#
01F0h	0000	0001	1111	0000	r/w	Primary Data Register	IDECS0#
01F1h	0000	0001	1111	0001	r/w	Primary Error Register	IDECS0#
01F2h	0000	0001	1111	0010	r/w	Primary Sector Count Register	IDECS0#
01F3h	0000	0001	1111	0011	r/w	Primary Sector Number Register	IDECS0#
01F4h	0000	0001	1111	0100	r/w	Primary Cylinder Low Register	IDECS0#
01F5h	0000	0001	1111	0101	r/w	Primary Cylinder High Register	IDECS0#
01F6h	0000	0001	1111	0110	r/w	Primary Drive/Head Register	IDECS0#
01F7h	0000	0001	1111	0111	r/w	Primary Status Register	IDECS0#
0278h	0000	0010	0111	1x00	r/w	LPT3 PP Data Latch	LPTCS#
0279h	0000	0010	0111	1x01	r	LPT3 PP Status	LPTCS#
027Ah	0000	0010	0111	1x10	r/w	LPT3 PP Control	LPTCS#
027Bh	0000	0010	0111	1x11	r/w		LPTCS#
02F8h	0000	0010	1111	1000	r/w	COM2 SP Transmit/Receive Register	COM2CS#
02F9h	0000	0010	1111	1001	r/w	COM2 SP Interrupt Enable Register	COM2CS#
02FAh	0000	0010	1111	1010	r	COM2 SP Interrupt Identification Register	COM2CS#
02FBh	0000	0010	1111	1011	r/w	COM2 SP Line Control Register	COM2CS#
02FCh	0000	0010	1111	1100	r/w	COM2 SP Modem Control Register	COM2CS#
02FDh	0000	0010	1111	1101	r	COM2 SP Line Status Register	COM2CS#
02FEh	0000	0010	1111	1110	r	COM2 SP Modem Status Register	COM2CS#
02FFh	0000	0010	1111	1111	r/w	COM2 SP Scratch Register	COM2CS#

Table 24. Encoded Chip Select Table (Continued)

Address	Address				Type	Name	Encoded Chip Select
	FEDC	BA98	7654	3210			
0370h	0000	0011	0111	0000	r/w	Secondary Floppy Disk Extended Mode Register	FLOPPYCS#
0371h	0000	0011	0111	0001	r/w	Secondary Floppy Disk Extended Mode Register	FLOPPYCS#
0372	0000	0011	0111	0010	w	Secondary Floppy Disk Digital Output Register	FLOPPYCS#
0373h	0000	0011	0111	0011	r/w	Reserved	FLOPPYCS#
0374h	0000	0011	0111	0100	r/w	Secondary Floppy Disk Status Register	FLOPPYCS#
0375h	0000	0011	0111	0101	r/w	Secondary Floppy Disk Data Register	FLOPPYCS#
0376h	0000	0011	0111	0110	r/w	Secondary Alternate Status Register	IDECS1#
0377h	0000	0011	0111	0111	r	Secondary Drive Address Register	IDECS1#
0377h*	0000	0011	0111	011x	r/w	Secondary Floppy Disk Digital Input Register	FLOPPYCS#
0378h	0000	0011	0111	1x00	r/w	LPT2 PP Data Latch	LPTCS#
0379h	0000	0011	0111	1x01	r	LPT2 PP Status	LPTCS#
037Ah	0000	0011	0111	1x10	r/w	LPT2 PP Control	LPTCS#
037Bh	0000	0011	0111	1x11	r/w		LPTCS#
03BCh	0000	0011	1011	1100	r/w	LPT1 PP Data Latch	LPTCS#
03BDh	0000	0011	1011	1101	r	LPT1 PP Status	LPTCS#
03BEh	0000	0011	1011	1110	r/w	LPT1 PP Control	LPTCS#
03BFh	0000	0011	1011	1111	r/w		LPTCS#
03F0h	0000	0011	1111	0000	r/w	Primary Floppy Disk Extended Mode Register	FLOPPYCS#
03F1h	0000	0011	1111	0001	r/w	Primary Floppy Disk Extended Mode Register	FLOPPYCS#
03F2h	0000	0011	1111	0010	w	Primary Floppy Disk Digital Output Register	FLOPPYCS#
03F3h	0000	0011	1111	0011	r/w	Reserved	FLOPPYCS#
03F4h	0000	0011	1111	0100	r/w	Primary Floppy Disk Status Register	FLOPPYCS#
03F5h	0000	0011	1111	0101	r/w	Primary Floppy Disk Data Register	FLOPPYCS#
03F6h	0000	0011	1111	0110	r/w	Primary Drive Alternate Status Register	IDECS1#
03F7h	0000	0011	1111	0111	r	Primary Drive Address Register	IDECS1#
03F7h*	0000	0011	1111	011x	r/w	Primary Floppy Disk Digital Input Register	FLOPPYCS#

Table 24. Encoded Chip Select Table (Continued)

Address	Address				Type	Name	Encoded Chip Select
	FEDC	BA98	7654	3210			
03F8h	0000	0011	1111	1000	r/w	COM1 SP Transmit/Receive Register	COM1CS#
03F9h	0000	0011	1111	1001	r/w	COM1 SP Interrupt Enable Register	COM1CS#
03FAh	0000	0011	1111	1010	r	COM1 SP Interrupt Identification Register	COM1CS#
03FBh	0000	0011	1111	1011	r/w	COM1 SP Line Control Register	COM1CS#
03FCh	0000	0011	1111	1100	r/w	COM1 SP Modem Control Register	COM1CS#
03FDh	0000	0011	1111	1101	r	COM1 SP Line Status Register	COM1CS#
03FEh	0000	0011	1111	1110	r	COM1 SP Modem Status Register	COM1CS#
03FFh	0000	0011	1111	1111	r/w	COM1 SP Scratch Register	COM1CS#
0800h–08FFh	0000	1000	xxxx	xxxx	r/w		CFIGMEMCS#
0C00h	0000	1100	0000	0000	r/w		CPAGECS#

NOTE:
*If both the IDE and Floppy Drive are located on the UBUS, FLOPPYCS# will not be generated, IDECS1# will be generated.

5.5.2.5 Subtractive Decode to ISA

ISA master and DMA cycles not positively decoded by the ISA decoder are contained to the ISA Bus.

5.6 Data Buffering

The SIO contains data buffers to isolate the PCI Bus from the ISA Bus. The buffering is described from two perspectives: PCI master accesses to the ISA Bus (Posted Write Buffer) and DMA/ISA master accesses to the PCI Bus (Line Buffer). Temporarily buffering the data requires buffer management logic to ensure that the data buffers remain coherent.

5.6.1 DMA/ISA MASTER LINE BUFFER

An 8-byte Line Buffer is used to isolate the ISA Bus's slower I/O devices from the PCI Bus. The Line Buffer is bi-directional and is used by ISA masters and the DMA controller to assemble and disassemble data. Only memory data written to or read from the PCI Bus by an ISA master or DMA is assembled/disassembled using this 8 byte line buffer. I/O cycles do not use the buffer.

Bits 0 and 1 of the PCI Control Register set the buffer to operate in either single transaction mode (bit = 0) or 8-byte mode (bit = 1). Note that ISA masters and DMA controllers can have their buffer modes configured separately.

In single transaction mode, the buffer will store only one transaction. For DMA/ISA master writes, this single transaction buffer looks like a posted write buffer. As soon as the ISA cycle is complete, a PCI cycle is scheduled. Subsequent DMA/ISA master writes are held off in wait-states until the buffer is empty. For DMA/ISA master reads, only the data requested is read over the PCI Bus. For instance, if the DMA channel is programmed in 16-bit mode, 16 bits of data will be read from PCI. As soon as the requested data is valid on the PCI bus, it is latched into the Line Buffer and the ISA cycle is then completed, as timing allows. Single transaction mode will guarantee strong read and write ordering through the buffers.

In 8 byte mode, for write data assembly, the Line Buffer acts as two individual 4 byte buffers working in ping pong fashion. For read data disassembly, the Line Buffer acts as one 8 byte buffer.

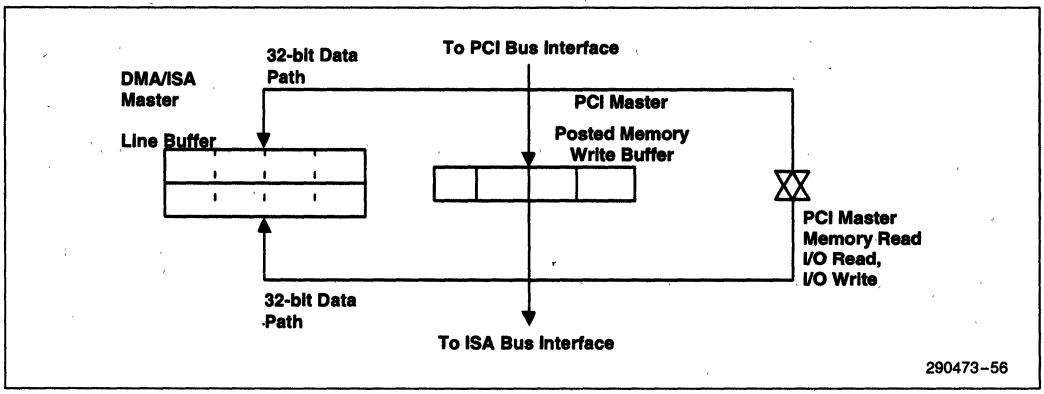


Figure 17. SIO Buffer Diagram

5.6.2 PCI MASTER POSTED WRITE BUFFER

PCI master memory write cycles destined to ISA memory are buffered in a 32-bit Posted Write Buffer. The PCI Memory Write and Memory Write and Invalidate commands are all treated as a memory write and can be posted, subject to the Posted Write Buffer status. The Posted Write Buffer has an address associated with it. A PCI master memory write can be posted any time the posted write buffer is empty and write posting is enabled (bit 2 of the PCI Control Configuration Register is set to a 1). Also, the ISA Bus must not be occupied. If the posted write buffer contains data, the PCI master write cycle is retried. If the posted write buffer is disabled, the SIO's response to a PCI master memory write is dependent on the state of the ISA Bus. If the ISA Bus is available and the posted write buffer is disabled, the cycle will immediately be forwarded to the ISA Bus (TRDY# will not be asserted until the ISA cycle has completed). If the ISA Bus is busy and the posted write buffer is disabled, the cycle is retried.

Memory read and I/O read and I/O write cycles do not use the 32-bit Posted Write Buffer.

5.6.3 BUFFER MANAGEMENT

Any time data is temporarily stored in the buffers between the ISA Bus and the PCI Bus, there are potential data coherency problems.

The SIO contains buffer management circuitry which guarantees data coherency by intercepting synchronization protocol between the buses and managing the buffers before synchronization communication between the buses is complete. The buffers are flushed or invalidated as appropriate before a bus cycle is allowed to occur in cases where data coherency could be lost.

5.6.3.1 DMA/ISA Master Line Buffer—Write State

When the DMA/ISA Master Line Buffer contains data that is to be written to the PCI Bus, it is in the Write State. The 8-byte line buffer is flushed when the line becomes full, when a subsequent write is a line miss, when a subsequent write would overwrite an already valid byte, or when a subsequent cycle is a read. The ISA master or DMA cycle that triggers the buffer flush will be held in wait-states until the flush is complete. The buffer is also flushed whenever there is a change in ISA Bus ownership as indicated by any DACK# signal going inactive.

Once the buffer is scheduled to be flushed to PCI, any PCI cycle to the SIO or ISA Bus will get retried by the SIO.

5.6.3.2 DMA/ISA Master Line Buffer—Read State

When the DMA/ISA Master Line Buffer contains data that has been read from the PCI Bus, it is in the Read State. The data in the buffer will be invalidated when the SIO accepts a PCI memory or I/O write cycle. The line buffer in the read state is also invalidated when a subsequent read is a line miss, or when a subsequent cycle is a write. The line buffer in the read state is not invalidated on a change of ISA ownership. Note that as bytes are disassembled from the line buffer, they are invalidated so that subsequent reads to the same byte will cause a line buffer miss.

5.6.3.3 PCI Master Posted Write Buffer

As soon as a PCI master has posted a memory write into the posted write buffer, the buffer is scheduled to be written to the ISA Bus. Any subsequent PCI master cycles to the SIO (including ISA Bus) will be retried until the posted write buffer is empty.

Prior to granting the ISA Bus to an ISA master or the DMA, the PCI master posted memory write buffer is flushed. Also, as long as the ISA master or DMA owns the ISA Bus, the posted write buffer is disabled. A PCI master write can not be posted while an ISA master or the DMA owns the ISA Bus.

5.7 SIO Timers

5.7.1 INTERVAL TIMERS

The SIO contains three counters that are equivalent to those found in the 82C54 programmable interval timer. The three counters are contained in one SIO timer unit, referred to as Timer-1. Each counter output provides a key system function. Counter 0 is connected to interrupt controller IRQ0 and provides a system timer interrupt for a time-of-day, diskette time-out, or other system timing functions. Counter 1 generates a refresh request signal and Counter 2 generates the tone for the speaker. Note that the 14.31818 MHz counters use OSC for a clock source.

Full details of this counter can be found in the 82C54 data sheet.

2

Table 25. Interval Timer Functions Table

Interval Timer Functions	
Function	**Counter 0—System Timer**
Gate	Always On
Clock In	1.193 MHz (OSC/12)
Out	INT-1 IRQ0
Function	**Counter 1—Refresh Request**
Gate	Always On
Clock In	1.193 MHz (OSC/12)
Out	Refresh Request
Function	**Counter 2—Speaker Tone**
Gate	Programmable-Port 61h
Clock In	1.193 MHz (OSC/12)
Out	Speaker

5.7.1.1 Interval Timer Address Map

Table 26 shows the I/O address map of the interval timer counters.

Table 26. Interval Timer Counters I/O Address Map

I/O Address	Register Description
040h	System Timer (Counter 0)
041h	Refresh Request (Counter 1)
042h	Speaker Tone (Counter 2)
043h	Control Word Register

Counter 0, System Timer

This counter functions as the system timer by controlling the state of IRQ0 and is typically programmed for Mode 3 operation. The counter produces a square wave with a period equal to the product of the counter period (838 ns) and the initial count value. The counter loads the initial count value one counter period after software writes the count value to the counter I/O address. The counter initially asserts IRQ0 and decrements the count value by two each counter period. The counter negates IRQ0 when the count value reaches 0. It then reloads the initial count value and again decrements the initial count value by two each counter period. The counter then asserts IRQ0 when the count value reaches 0, reloads the initial count value, and repeats the cycle, alternately asserting and negating IRQ0.

Counter 1, Refresh Request Signal

This counter provides the refresh request signal and is typically programmed for Mode 2 operation. The counter negates refresh request for one counter period (833 ns) during each count cycle. The initial count value is loaded one counter period after being written to the counter I/O address. The counter initially asserts refresh request, and negates it for 1 counter period when the count value reaches 1. The counter then asserts refresh request and continues counting from the initial count value.

Counter 2, Speaker Tone

This counter provides the speaker tone and is typically programmed for Mode 3 operation. The counter provides a speaker frequency equal to the counter clock frequency (1.193 MHz) divided by the initial count value. The speaker must be enabled by a write to port 061h (see Section 4.5.1 on the NMI Status and Control Register).

5.7.2 BIOS TIMER

5.7.2.1 Overview

The SIO provides a system BIOS Timer that decrements at each edge of its 1.04 MHz clock (derived by dividing the 8.33 MHz SYSCLK by 8). Since the state of the counter is undefined at power-up, it must be programmed before it can be used. Accesses to the BIOS Timer are enabled and disabled through the BIOS Timer Base Address Register. The timer continues to count even if accesses are disabled.

A BIOS Timer Register is provided to start the timer counter by writing an initial clock value. The BIOS Timer Register can be accessed as a single 16-bit I/O port or as a 32-bit port with the upper 16-bits being "don't care" (reserved). It is up to the software to access the I/O register in the most convenient way. The I/O address of the BIOS Timer Register is software relocatable. The I/O address is determined by the value programmed into the BIOS Timer Base Address Register.

The BIOS Timer clock has a value of 1.04 MHz using an 8.33 MHz SYSCLK input (an 8 to 1 ratio will always exist between SYSCLK and the timer clock). This allows the counting of time intervals from 0 ms to approximately 65 ms. Because of the PCI clock rate, it is possible to start the counter and read the value back in less than 1 μs. The expected value of the expired interval is 0, but depending on the state of the internal clock divisor, the BIOS Timer might indicate that 1 ms has expired. Therefore, accuracy of the counter is $\pm$ 1 μs.

5.7.2.2 BIOS Timer Operations

A write operation to the BIOS Timer Register will initiate the counting sequence. The timer can be initiated by writing either the 16-bit data portion or the whole 32-bit register (upper 16 bits are "don't care"). After initialization, the BIOS timer will start decrementing until it reaches zero. Then it will stop decrementing (and hold a zero value) until initialized again.

After the timer is initialized, the current value can be read at any time and the timer can be reprogrammed (new initial value written), even before it reaches zero.

All write and read operations to the BIOS timer Register should include all 16 counter bits. Separate accesses to the individual bytes of the counter must be avoided since this can cause unexpected results (wrong count intervals).

5.8 Interrupt Controller

The SIO provides an ISA compatible interrupt controller which incorporates the functionality of two 82C59 interrupt controllers. The two controllers are cascaded so that 14 external and two internal interrupts are possible. The master interrupt controller provides IRQ[7:0] and the slave interrupt controller provides IRQ [15:8] (see Figure 18). The two internal interrupts are used for internal functions only and are not available to the user. IRQ2 is used to cascade the two controllers together and IRQ0 is used as a system timer interrupt and is tied to Interval Timer 1, Counter 0. The remaining 14 interrupt lines (IRQ1, IRQ3–IRQ15) are available for external system interrupts. Edge or level sense selection is programmable on a by-controller basis.

The Interrupt Controller consists of two separate 82C59 cores. Interrupt Controller 1 (CNTRL-1) and Interrupt Controller 2 (CNTRL-2) are initialized separately and can be programmed to operate in different modes. The default settings are: 80x86 Mode, Edge Sensitive (IRQ0-15) Detection, Normal EOI, Non-Buffered Mode, Special Fully Nested Mode disabled, and Cascade Mode. CNTRL-1 is connected as the Master Interrupt Controller and CNTRL-2 is connected as the Slave Interrupt Controller.

Note that IRQ13 is generated internally (as part of the coprocessor error support) by the SIO when bit 5 in the ISA Clock Divisor Register is set to a 1. When this bit is set to a 0, then the FERR#/IRQ13 signal is used as an external IRQ13 signal and has the same functionality as the normal IRQ13 signal. IRQ12/M is generated internally (as part of the mouse support) by the SIO when bit 4 in the ISA Clock Divisor Register is set to a 1. When set to a 0, the standard IRQ12 function is provided.

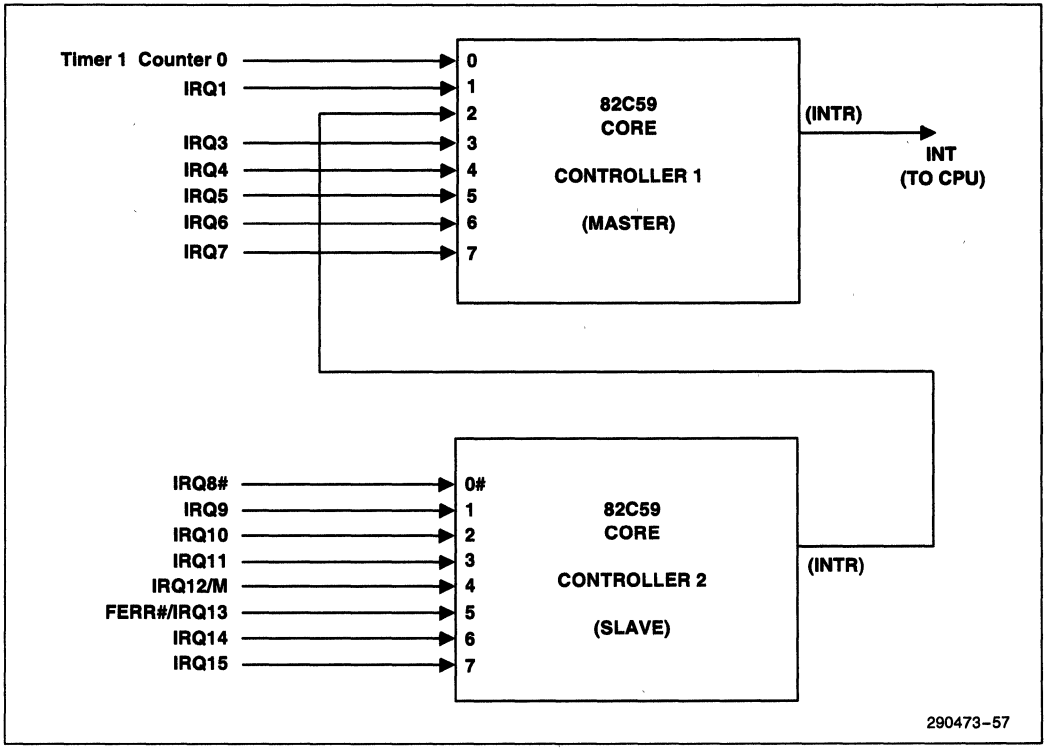


Figure 18. Block Diagram of the Interrupt Controller

Table 27 lists the I/O port address map for the interrupt registers:

Table 27. Interrupt Registers I/O Port Address Map

Interrupts	I/O Address	# of Bits	Register
IRQ[7:0]	0020h	8	CNTRL-1 Control Register
IRQ[7:0]	0021h	8	CNTRL-1 Mask Register
IRQ[15:8]	00A0h	8	CNTRL-2 Control Register
IRQ[15:8]	00A1h	8	CNTRL-2 Mask Register

IRQ0, IRQ2, (and possibly IRQ13 and IRQ12 if the "mouse" or floating point error logic is disabled in the ISA Clock Divisor Register), are connected to the interrupt controllers internally. The other interrupts are always generated internally and their typical functions are shown in Table 28:

Table 28. Typical Interrupt Functions

Priority	Label	Controller	Typical Interrupt Source
1	IRQ0	1	Interval timer 1, Counter 0 OUT
2	IRQ1	1	Keyboard
3–10	IRQ2	1	Interrupt from Controller 2
3	IRQ8#	2	Real Time Clock
4	IRQ9	2	Expansion Bus Pin B04
5	IRQ10	2	Expansion Bus Pin D03
6	IRQ11	2	Expansion Bus Pin D04
7	IRQ12/M	2	Mouse Interrupt
8	FERR#/IRQ13	2	Coprocessor Error
9	IRQ14	2	Fixed Disk Drive Controller Expansion Bus Pin D07
10	IRQ15	2	Expansion Bus Pin D06
11	IRQ3	1	Serial Port 2, Expansion Bus B25
12	IRQ4	1	Serial Port 1, Expansion Bus B24
13	IRQ5	1	Parallel Port 2, Expansion Bus B23
14	IRQ6	1	Diskette Controller, Expansion Bus B22
15	IRQ7	1	Parallel Port 1, Expansion Bus B21

5.8.1 EDGE AND LEVEL TRIGGERED MODES

There are two ELCR registers, one for each 82C59 bank. They are located at I/O ports 04D0h (for the Master Bank, IRQ[0:1,3:7]#) and 04D1h (for the Slave Bank, IRQ[8:15])#. They allow the edge and level sense selection to be made on an interrupt by interrupt basis instead of on a complete bank. Interrupts reserved for ISA use MUST be programmed for edge sensitivity (to ensure ISA compatibility). That is, IRQ (0,1,2,8#,13) must be programmed for edge sensitive operation. The LTIM bit (Edge/Level Bank select, offsets 20h, A0h) is disabled in the SIO. The default programming is equivalent to programming the LTIM bit (ICW1 bit 3) to a 0.

If an ELCR bit is equal to "0", an interrupt request will be recognized by a low to high transition on the corresponding IRQ input. The IRQ input can remain high without generating another interrupt.

If an ELCR bit is equal to "1", an interrupt request will be recognized by a "low" level on the corresponding IRQ input, and there is no need for an edge detection. For level triggered interrupt mode, the interrupt request signal must be removed before the EOI command is issued or the CPU interrupt must be disabled. This is necessary to prevent a second interrupt from occurring.

In both the edge and level triggered modes the IRQ inputs must remain active until after the falling edge of the first INTA#. If the IRQ input goes inactive before this time a DEFAULT IRQ7 will occur when the CPU acknowledges the interrupt. This can be a useful safeguard for detecting interrupts caused by spurious noise glitches on the IRQ inputs. To implement this feature the IRQ7 routine is used for "clean up" simply executing a return instruction, thus ignoring the interrupt. If IRQ7 is needed for other purposes a default IRQ7 can still be detected by reading the ISR. A normal IRQ7 interrupt will set the corresponding ISR bit, a default IRQ7 won't. If a default IRQ7 routine occurs during a normal IRQ7 routine, however, the ISR will remain set. In this case it is necessary to keep track of whether or not the IRQ7 routine was previously entered. If another IRQ7 occurs it is a default.

5.8.2 REGISTER FUNCTIONALITY

For a detailed description of the Interrupt Controller register set, please see Section 4.4, Interrupt Controller Register Description.

5.8.3 NON-MASKABLE INTERRUPT (NMI)

An NMI is an interrupt requiring immediate attention and has priority over the normal interrupt lines (IRQx). The SIO indicates error conditions by generating a non-maskable interrupt.

NMI interrupts are caused by the following conditions:

1. System Errors on the PCI Bus. SERR# is driven low by a PCI resource when this error occurs.
2. Parity errors on the add-in memory boards on the ISA expansion bus. IOCHK# is driven low when this error occurs.

The NMI logic incorporates two different 8-bit registers. These registers are addressed at locations 061h and 070h. The status of Port (061h) is read by the CPU to determine which source caused the NMI. Bits set to 1 in these ports show which device requested an NMI interrupt. After the NMI interrupt routine processes the interrupt, the NMI status bits are cleared by the software. This is done by setting the corresponding enable/disable bit high. Port (070H) is the mask register for the NMI interrupts. This register can mask the NMI signal and also disable or enable all NMI sources.

The individual enable/disable bits clear the NMI detect flip-flops when disabled.

All NMI sources can be enabled or disabled by setting Port 070h bit 7 to a 0 or 1. This disable function does not clear the NMI detect flip-flops. This means, if NMI is disabled then enabled via Port 070h, then an NMI will occur when Port 070h is re-enabled if one of the NMI detect flip-flops had been previously set.

To ensure that all NMI requests are serviced, the NMI service routine software needs to incorporate a few very specific requirements. These requirements are due to the edge detect circuitry of the host microprocessor, 80386 or 80486. The software flow would need to be the following:

1. NMI is detected by the processor on the rising edge of the NMI input.
2. The processor will read the status stored in port 061h to determine what sources caused the NMI. The processor may then set to 0 the register bits controlling the sources that it has determined to be active. Between the time the processor reads

the NMI sources and sets them to a 0, an NMI may have been generated by another source. The level of NMI will then remain active. This new NMI source will not be recognized by the processor because there was no edge on NMI.

3. The processor must then disable all NMI's by setting bit 7 of port 070H to a 1 and then enable all NMI's by setting bit 7 of port 070H to a 0. This will cause the NMI output to transition low then high if there are any pending NMI sources. The CPU's NMI input logic will then register a new NMI.

Section 4.5 Control Registers, contains a detailed description of the NMI Status and Control Register (port 061h) and the NMI Enable and Real-Time Clock Address Register at port 070h.

5.9 Utility Bus Peripheral Support

The Utility Bus is a secondary bus buffered from the ISA Bus used to interface with peripheral devices that do not require a high speed interface. The buffer control for the lower 8 data signals is provided by the SIO via two control signals; UBUSOE# and UBUSTR. Figure 19 shows a block diagram of the external logic required as part of the decode and Utility Bus buffer control.

The SIO provides the address decode and three encoded chip selects to support:

1. Floppy Controller
2. Keyboard Controller
3. Real Time Clock
4. IDE Drive
5. 2 Serial Ports (COM1 and COM2)
6. 1 Parallel Port (LPT1, 2, or 3)
7. BIOS Memory
8. Configuration Memory (8 Kbyte I/O Mapped)

The SIO also supports the following functions:

1. Floppy DSKCHG Function
2. Port 92 Function (Alternate A20 and Alternate Reset)
3. Coprocessor Logic (FERR# and IGNNE# Function)

The binary code formed by the three Encoded Chip Selects determines which Utility Bus device is selected. The SIO also provides an Encoded Chip Select Enable signal (ECSEN#) that is used to select between the two external decoders. A zero selects decoder 1 and a one selects decoder 2. The table below shows the address decode for each of the Utility Bus devices.

Table 29. NMI Source Enable/Disable and Status Port Bits

NMI Source	I/O Port Bit for Status Reads	I/O Port Bit for Enable/Disable
IOCHK#	Port 061h, Bit 6	Port 061h, Bit 3
SERR#	Port 061h, Bit 7	Port 061h, Bit 2

Table 30. Encoded Chip Select Summary Table

ECSADDR2	ECSADDR1	ECSADDR0	ECSEN#	Address Decoded	External Chip Select	Note	Cycle Type
Decoder 1							
0	0	0	0	70h, 72h, 74, 76h	RTCALE#		I/O W
0	0	1	0	71h, 73h, 75h, 77h	RTCCS#		I/O R/W
0	1	0	0	60h, 62h, 64h, 66h	KEYBRDCS#		I/O R/W
0	1	1	0	000E0000h–000FFFFFh FFFE0000h–FFFFFFFFh FFF80000h–FFFDFFFFh	BIOSCS#	1	MEM R/W
1	0	0	0	3F0h–3F7h (primary) 370h–377h (secondary)	FLOPPYCS#	2	I/O R/W
1	0	1	0	1F0h–1F7h (primary) 170h–177h (secondary)	IDECS0#	2	I/O R/W
1	1	0	0	3F6h–3F7h (primary) 376h–377h (secondary)	IDECS1#	2	I/O R/W
1	1	1	0	Reserved			
Decoder 2							
0	0	0	1	Reserved			
0	0	1	1	0C00h	CPAGECS#	3	I/O R/W
0	1	0	1	0800h–08FFh	CFIGMEMCS#	3	I/O R/W
0	1	1	1	3F8h–3FFh (COM1) -or- 2F8h–37Fh (COM2)	COMACS#	4	I/O R/W
1	0	0	1	3F8h–3FFh (COM1) -or- 2F8h–37Fh (COM2)	COMBCS#	4	I/O R/W
1	0	1	1	3BCh–3BFh (LPT1) 378h–37Fh (LPT2) 278h–27Fh (LPT3)	LPTCS#	5	I/O R/W
1	1	0	1	Reserved			
1	1	1	1	Idle State			

NOTES:
1. The encoded chip select signals for BIOSCS# will always be generated for accesses to the upper 64 KB at the top of 1 MByte (F0000h–FFFFFh) and its aliases at the top of the 4 GB and 4 GB-1 MByte. Access to the lower 64 KByte (E0000h–EFFFFh) and its aliases at the top of 4 GB and 4GB–1MB can be enabled or disabled through the SIO. An additional 384 KB of BIOS memory at the top of 4 GB (FFFD0000h–FFFDFFFFh) can be enabled for BIOS use.
2. The primary and secondary locations are programmable through the SIO. Only one location range can be enabled at any one time. The floppy and IDE share the same enable and disable bit (i.e. if the floppy is set for primary, the IDE is also set for primary).
3. These signals can be used to select additional configuration RAM.
4. COM1 and COM2 address ranges can be programmed for either port A (COMACS#) or port B (COMBCS#).
5. Only one address range (LPT1, LPT2, or LPT3) can be programmed at any one time.

Port 92h Function

The SIO integrates the Port 92h Register. This register provides the alternate reset (ALTRST) and alternate A20 (ALT_A20) functions. Figure 19 shows how these functions are tied into the system.

DSKCHG Function

DSKCHG is tied directly to the DSKCHG signal of the floppy controller. This signal is inverted and driven onto system data line 7 (SD7) during I/O read cycles to floppy address locations 3F7h (primary) or 377 (secondary) as indicated by Table 31.

Table 31. DSKCHG Summary Table

FLOPPYCS# Decode	IDECSx# Decode	State of SD7 (Output)	State of UBUSOE#
Enabled	Enabled	Tri-stated	Enabled
Enabled	Disabled	Driven via DSKCHG	Disabled
Disabled	Enabled	Tri-stated	Enabled(1)
Disabled	Disabled	Tri-stated	Disabled

NOTE:
1. For this mode to be supported, extra logic is required to disable the U-bus transceiver for accesses to 3F7/377. This is necessary because of potential contention between the Utility bus buffer and a floppy on the ISA Bus driving the system bus at the same time during shared I/O accesses.

Coprocessor Error Support

If bit 5 in the ISA Clock Divisor Register is set to a one, the SIO will support coprocessor error reporting through the FERR#/IRQ13 signal.

FERR# is tied directly to the Coprocessor error signal of the CPU. If FERR# is driven active in this mode (coprocessor error detected by the CPU), an internal IRQ13 is generated and the INT output from the SIO is driven active. When a write to I/O location F0h is detected, the SIO negates IRQ13 and drives IGNNE# active. IGNNE# remains active until FERR# is driven inactive. Note that IGNNE# is not generated unless FERR# is active.

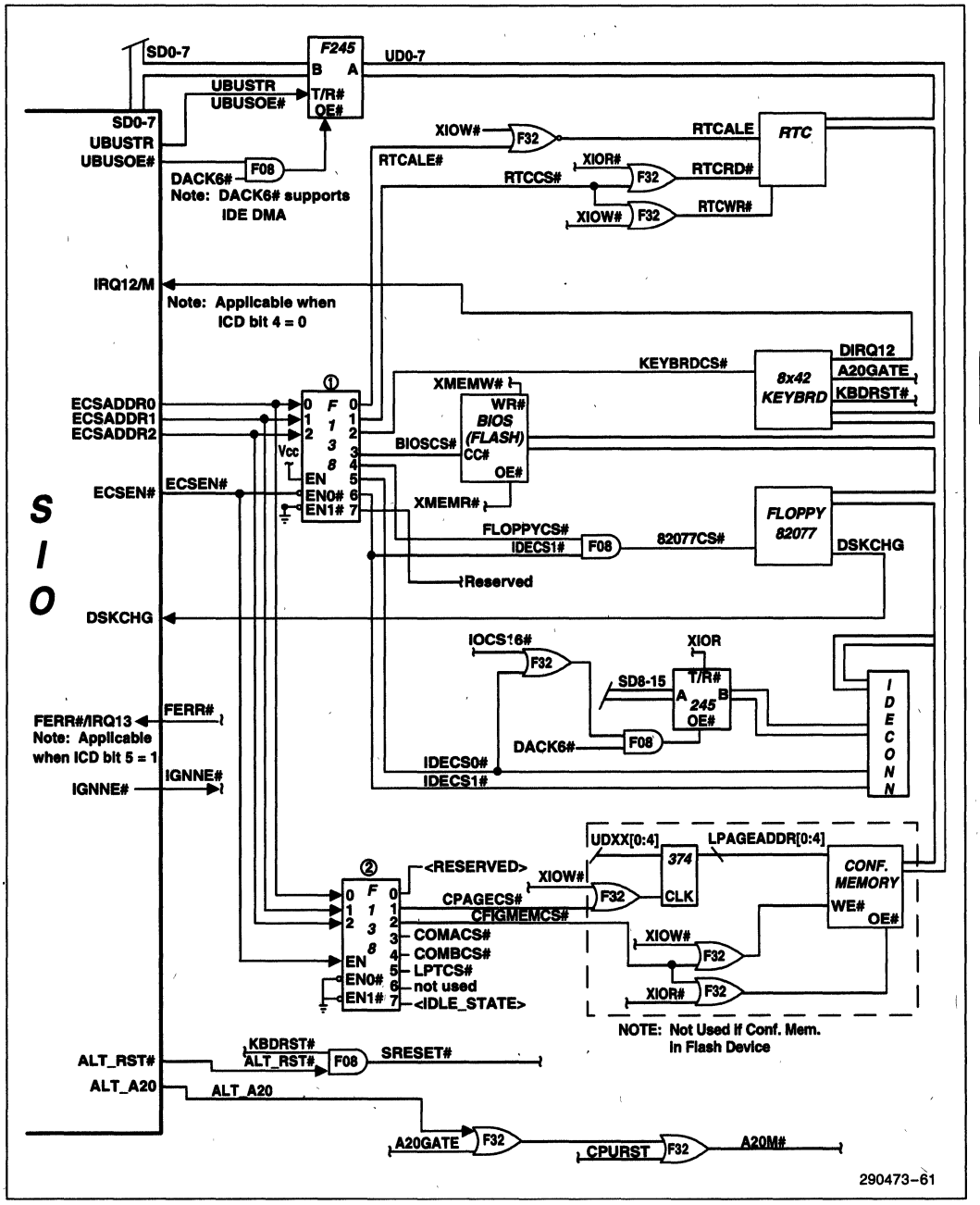


Figure 19. Utility Bus External Support Logic

Utility Bus accesses by the SIO, by an ISA master, and by the DMA is shown in Figure 20 and Figure 21. UBUSOE# and UBUSTR are driven differently for DMA cycles as shown in Figure 21.

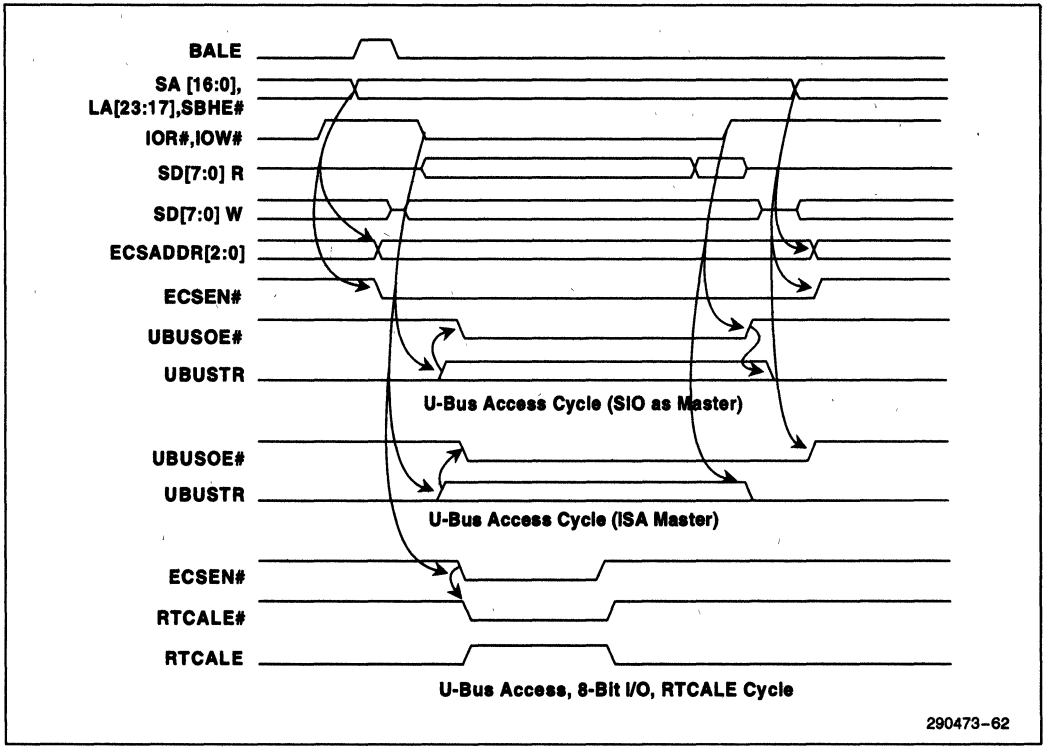


Figure 20. Utility Bus Access (SIO and ISA Master)

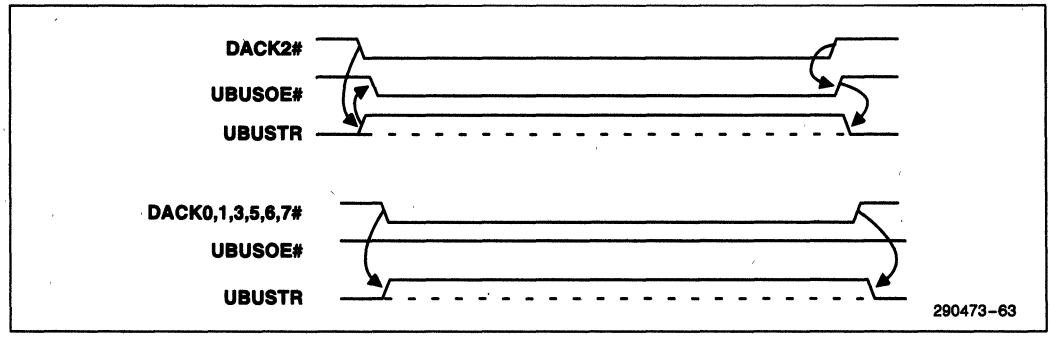


Figure 21. Utility Bus Access (DMA)

5.10 Power Management

The SIO's power management architecture is based around three core functions:

1. SMM (System Management Mode)
2. Clock Throttling
3. APM (Advanced Power Management Interface)

SMM is a mode during which an S Series Processor is executing SMM code from a secure memory space (SMRAM). SMM is invoked through the assertion of an SMI (System Management Interrupt). Physically, this is signaled over the SMI# pin. SMI's are triggered by various hardware and software events. SMRAM is used to store the SMM code which is really the SMI interrupt handler routine.

Clock Throttling will be used to reduce the power consumption of the CPU. STPCLK# is the physical signal used to control the CPU's clock.

APM creates an interface to allow the Operating System to communicate with the SMM code.

Figure 22 shows how the power management signals are connected in a Saturn based system with an S-series CPU.

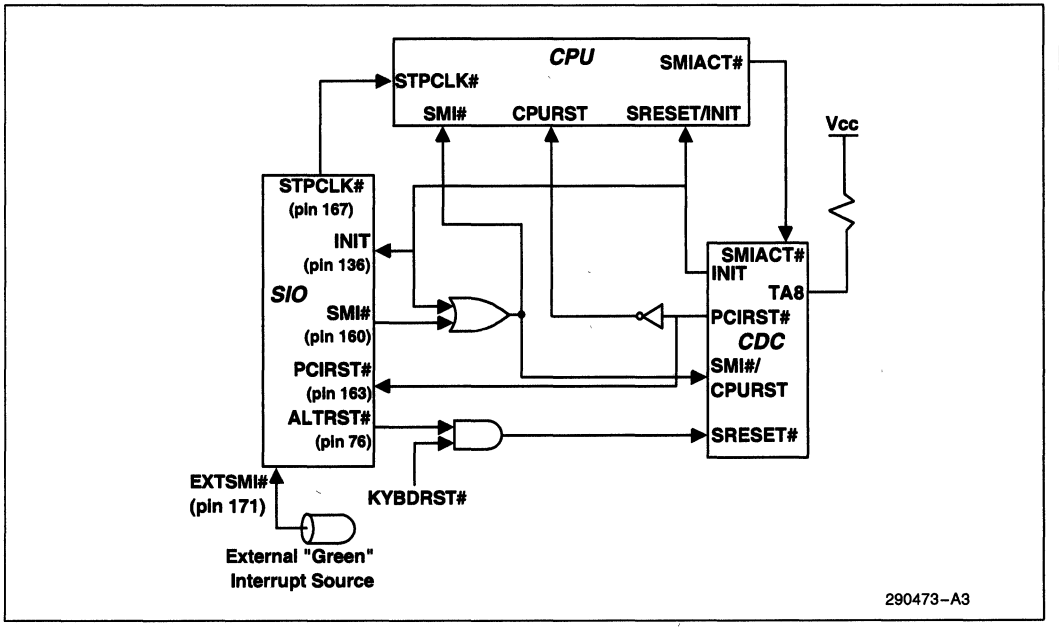


Figure 22. Power Management

6.0 ELECTRICAL CHARACTERISTICS

6.1 Absolute Maximum Ratings*

Case Temperature under Bias . . . −65°C to +110°C
Storage Temperature . . . . . −65°C to +150°C
Supply Voltages
with Respect to Ground . . . −0.5V to V_{CC} + 0.5V
Voltage On Any Pin . . . . . −0.5V to V_{CC} + 0.5V

** 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.*

7.0 MECHANICAL SPECIFICATIONS

7.1 Package Diagram

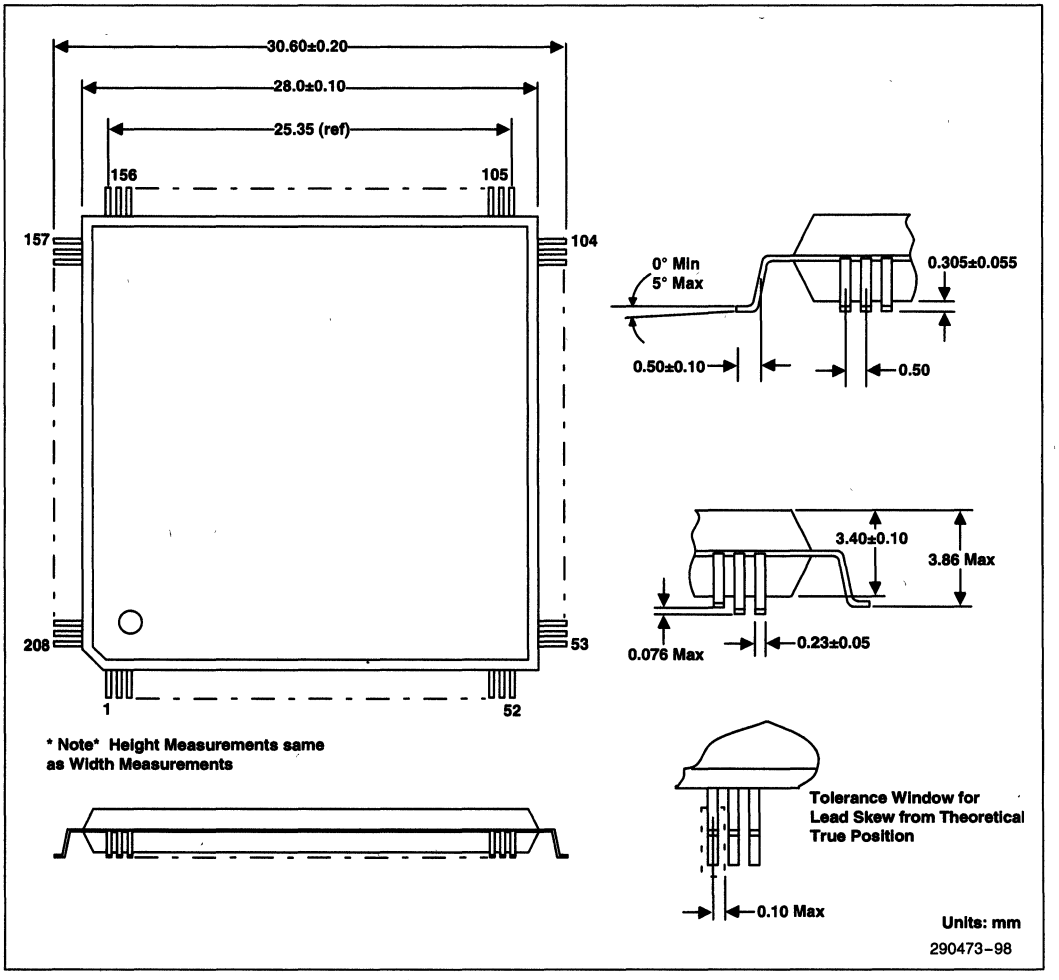


Figure 23. 208-Pin Quad Flat Pack (QFP) Package Dimensions

7.2 Thermal Specifications

Table 32. 82378 QFP Package Thermal Characteristics

Thermal Resistance-°C/Watt			
	Air Flow Rate (Ft./Min)		
Parameter	**0**	**200**	**400**
$\theta_{Junction\ to\ Case}$	6.6	6.6	6.6
$\theta_{Case\ to\ Ambient}$	36.6	27.4	24

8.0 TESTABILITY

The TEST and TESTO pins are used to test the SIO. During normal operations, the TEST pin must be grounded. The test output TESTO may be left as a no-connect (NC).

8.1 Global Tri-State

The TEST pin and IRQ3 are used to provide a high-impedance tri-state test mode. When the following input combination occurs, all outputs and bi-directional pins are tri-stated, with the exception of TESTO:

TEST = "1"
IRQ3 = "1"

The SIO must be reset after the bi-directional and output pins have been tri-stated in this manner.

8.2 NAND Tree

A NAND Tree is provided primarily for V_{IL}/V_{IH} testing. The NAND Tree is also useful for ATE at board level testing. The NAND Tree allows the tester to test the solder connections for each individual signal pin.

The TEST pin, along with IRQ5 or IRQ6, activates the NAND Tree. All bi-directional pins, and certain pure output pins using bi-directional buffers for performance reasons, are tri-stated when the following input combinations occur:

TEST = "1"
IRQ5 = "1"
- or -
TEST = "1"
IRQ6 = "0"

In the 82378, the output pulse train is observed at the TESTO test output. Pure output pins are not included directly in the NAND Tree. As noted in Section 8.3, each output can be expected to toggle after the corresponding node noted next to the pin name toggles from a "1" to a "0".

The sequence of the ATE test is as follows:

1. Drive TEST and IRQ5 high or TEST high and IRQ6 low.
2. Drive each input and bi-directional pin noted in Section 8.3 high.
3. Starting with the pin farthest from TESTO (SA8), individually drive each pin low. Expect TESTO to toggle with each pin. Expect each pure output noted in Section 8.3 to toggle after each corresponding input pin has been driven low.
4. Turn off tester drivers before driving TEST low.
5. Reset the SIO prior to proceeding with further testing.

8.3 NAND Tree Cell Order

Table 33. NAND Tree Cell Order

Tree Output #	Pin #	Pin Name	Notes
	14	IRQ4	Reserved
	21	TESTO	Test Mode Output
1	11	IRQ5	Cell Closest to TESTO
2	10	SA9	
3	9	IRQ6	
4	8	SA10	
5	7	IRQ7	
6	6	SA11	
7	5	SA12	
8	4	REFRESH#	
9	3	SA13	
10	207	SA14	
11	206	MASTER#	
12	205	SA15	
13	204	MEMW#	
14	203	MEMR#	
15	202	SA16	
16	201	SA17	
17	200	IOR#	
18	199	SA18	
19	198	IOW#	
20	197	SA19	
21	196	SMEMR#	
22	193	AEN	
23	192	SMEMW#	
24	191	IOCHRDY	
25	190	SD0	
26	189	SD1	

Table 33. NAND Tree Cell Order (Continued)

Tree Output#	Pin #	Pin Name	Notes
27	188	ZEROWS#	
28	187	SD2	
29	186	SD3	
30	185	SD4	
31	184	IRQ9	
32	180	SD5	
33	179	SD6	
34	178	SD7	
35	177	RSTDRV	
36	176	IOCHK#	
	175	ECSADDR0	NAND Tree Output of Tree Cell 28
	174	ECSADDR1	NAND Tree Output of Tree Cell 29
	173	ECSADDR2	NAND Tree Output of Tree Cell 30
37	172	IRQ8#	
38	171	EXTSMI#	
	170	ECSEN#	NAND Tree Output of Tree Cell 32
	169	TEST	PI = > VCC, TEST must be '1'
39	168	IRQ1	
	167	STPCLK#	
40	166	SYSCLK	
	165	UBUSTR	NAND Tree Output of Tree Cell 33
	164	UBUSOE#	NAND Tree Output of Tree Cell 34
41	163	PCIRST#	
42	161	DSKCHG	
	160	SMI#	
43	159	AD0	
44	155	AD1	
45	154	AD2	
46	153	AD3	

Table 33. NAND Tree Cell Order (Continued)

Tree Output#	Pin #	Pin Name	Notes
47	152	AD4	
48	151	AD5	
49	150	AD6	
50	149	AD7	
51	148	AD8	
52	147	C/BE0#	
53	146	AD9	
54	143	AD10	
55	142	AD11	
56	141	AD12	
57	140	AD13	
58	139	AD14	
59	138	AD15	
60	137	C/BE1#	
61	136	INIT	
62	135	PAR	
63	134	SERR#	
64	133	LOCK#	
65	132	STOP#	
66	128	DEVSEL#	
67	127	TRDY#	
68	126	IRDY#	
69	125	FRAME#	
70	124	C/BE2#	
71	123	AD16	
72	122	AD17	
73	121	AD18	
74	120	AD19	

Table 33. NAND Tree Cell Order (Continued)

Tree Output #	Pin #	Pin Name	Notes
75	119	AD20	
76	118	AD21	
77	115	AD22	
78	114	AD23	
79	113	C/BE3#	
80	112	AD24	
81	111	AD25	
82	110	AD26	
83	109	AD27	
84	108	AD28	
85	107	AD29	
86	106	AD30	
87	102	AD31	
88	101	IDSEL	
89	100	REQ3#	
90	98	REQ1#	
91	97	REQ2#	
92	96	CPUREQ#	
	95	CPUGNT#	NAND Tree Output of Tree Cell 93
	94	GNT1#	NAND Tree Output of Tree Cell 95
93	93	REQ0#	
	92	GNT0#	NAND Tree Output of Tree Cell 100
94	90	PCICLK	
	89	FLSHREQ#	NAND Tree Output of Tree Cell 102
95	88	MEMACK#	
	87	MEMREQ#	NAND Tree Output of Tree Cell 103

Table 33. NAND Tree Cell Order (Continued)

Tree Output#	Pin #	Pin Name	Notes
	86	MEMCS#	NAND Tree Output of Tree Cell 104
	85	ALT_A20	NAND Tree Output of Tree Cell 105
96	84	PIRQ[3]#	
97	83	PIRQ[2]#	
98	82	PIRQ[1]#	
99	81	PIRQ[0]#	
100	80	OSC	
	76	ALT_RST#	NAND Tree Output of Tree Cell 23
	75	INT	NAND Tree Output of Tree Cell 24
	74	NMI	NAND Tree Output of Tree Cell 25
101	73	SPKR	
	72	IGNNE#	NAND Tree Output of Tree Cell 26
102	71	FERR#	
103	70	SD15	
104	69	SD14	
105	68	SD13	
106	67	SD12	
107	65	DREQ7	
108	64	SD11	
109	63	DACK7#	
110	62	SD10	
111	61	DREQ6	
112	60	SD9	
113	59	DACK6#	
114	58	DREQ3	
115	57	DREQ2	
116	56	DREQ1	
117	55	SD8	
118	51	DREQ5	

Table 33. NAND Tree Cell Order (Continued)

Tree Output#	Pin #	Pin Name	Notes
119	50	DACK5#	
120	49	DACK3#	
121	48	DACK1#	
122	47	DREQ0	
123	46	LA17	
124	45	DACK0#	
125	44	LA18	
126	43	IRQ14	
127	42	LA19	
128	41	IRQ15	
129	40	LA20	
130	39	IRQ12/M	
131	38	LA21	
132	37	IRQ11	
133	36	LA22	
134	35	IRQ10	
135	34	LA23	
136	33	IOCS16#	
137	32	SBHE#	
138	31	MEMCS16#	
139	30	SA0	
140	29	SA1	
141	28	SA2	
142	24	SA3	
143	23	BALE	
144	22	SA4	
145	20	EOP	
146	19	SA5	

Table 33. NAND Tree Cell Order (Continued)

Tree Output #	Pin #	Pin Name	Notes
147	18	DACK2#	
148	17	SA6	
149	16	IRQ3	Output signals will transition from high-impedance state to driving state after this pin is driven low.
150	15	SA7	
151	13	SA8	Cell furthest from TESTO Start of NAND Tree

8.4 NAND Tree Diagram

Figure 24 shows the NAND Tree Diagram.

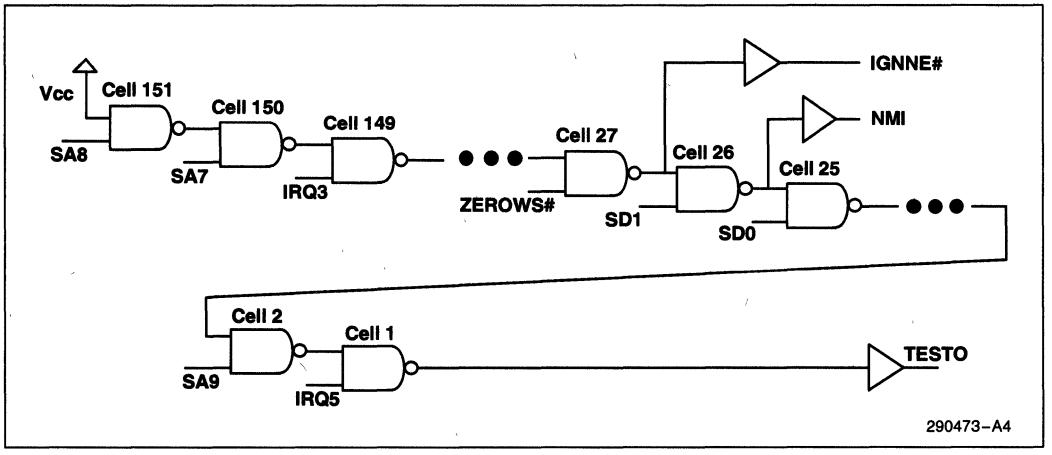


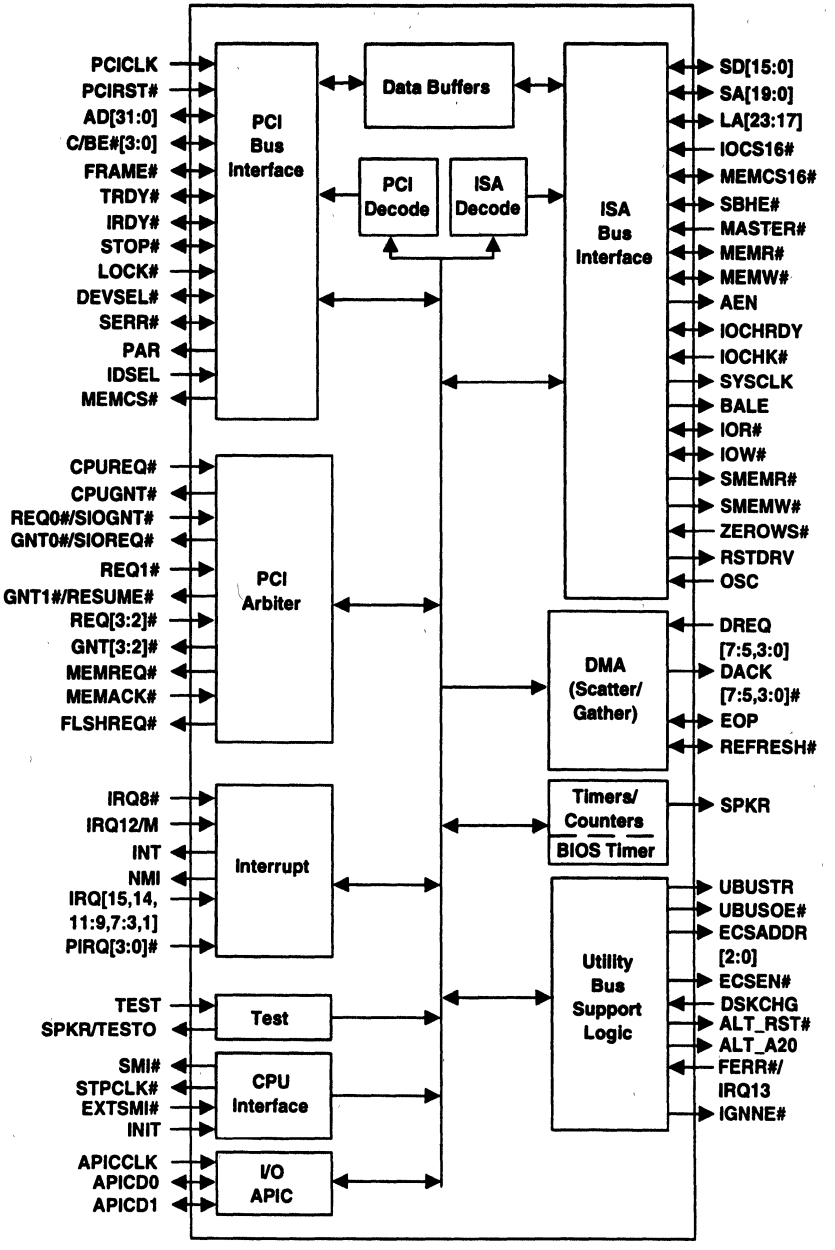
Figure 24. NAND Tree Diagram for 82378

82379AB SYSTEM I/O-APIC (SIO.A)

- **Provides the Bridge between the PCI Bus and ISA Bus**
- **100% PCI and ISA Compatible**
  - **— PCI and ISA Master/Slave Interface**
  - **— Directly Drives 10 PCI Loads and 6 ISA Slots**
  - **— Supports PCI at 25 MHz and 33 MHz**
  - **— Supports ISA from 6 MHz to 8.33 MHz**
- **Enhanced DMA Functions**
  - **— Compatible DMA Transfers**
  - **— 27-Bit Addressability**
  - **— Seven Independently Programmable Channels**
  - **— Functionality of Two 82C37A DMA Controllers**
- **Integrated Data Buffers to Improve Performance**
  - **— 8-Byte DMA/ISA Master Line Buffer**
  - **— 32-Bit Posted Memory Write Buffer to ISA**
- **Integrated 16-Bit BIOS Timer**
- **Non-Maskable Interrupts (NMI)**
  - **— PCI System Errors**
  - **— ISA Parity Errors**
- **Four Dedicated PCI Interrupts**
  - **— Level Sensitive**
  - **— Can be Mapped to Any Unused Interrupt**
- **Arbitration for ISA Devices**
  - **— ISA Masters**
  - **— DMA and Refresh**
- **Arbitration for PCI Devices**
  - **— Six PCI Masters Are Supported**
  - **— Fixed, Rotating, or a Combination of the Two**
- **Utility Bus (X-Bus) Peripheral Support**
  - **— Provides Chip Select Decode**
  - **— Controls Lower X-Bus Data Byte Transceiver**
- **Integrates the Functionality of One 82C54 Timer**
  - **— System Timer**
  - **— Refresh Request**
  - **— Speaker Tone Output**
- **Integrates the Functionality of Two 82C59 Interrupt Controllers**
  - **— 14 Interrupts Supported**
  - **— Edge/Level Selectable Interrupts: Each Interrupt Individually Programmable**
- **Complete Support for SL Enhanced Intel486™ CPU's**
  - **— SMI# Generation Based on System Hardware Events**
  - **— STPCLK# Generation to Power Down the CPU**
- **Integrated I/O Advanced Programmable Interrupt Controller (APIC)**

The 82379AB System I/O-APIC (SIO.A) component provides the bridge between the PCI bus and the ISA expansion bus. The 82379AB also integrates many of the common I/O functions found in today's ISA based PC systems. The 82379AB incorporates the logic for a PCI interface (master and slave), ISA interface (master and slave), enhanced seven channel DMA controller that supports data buffers to isolate the PCI bus from the ISA bus and to enhance performance, PCI and ISA arbitration, 14 level interrupt controller, a 16-bit BIOS timer, three programmable timer/counters, and Non-Maskable Interrupt (NMI) Control Logic. The 82379AB also provides decode for peripheral devices such as the Flash BIOS, Real Time Clock, Keyboard/Mouse Controller, Floppy Controller, two Serial Ports, one Parallel Port, and IDE Hard Disk Drive. The 82379AB supports several Advanced Power Management features such as SMI# Interrupt. The 82379AB also supports a total of 6 PCI Masters, and can support up to 4 PCI Interrupts. The 82379AB incorporates an Advanced Programmable Interrupt Controller (APIC) that communicates with the processor via a dedicated two data bit bus.

December 1994
Order Number: 290520-001



290520-1

82379AB Component Block Diagram

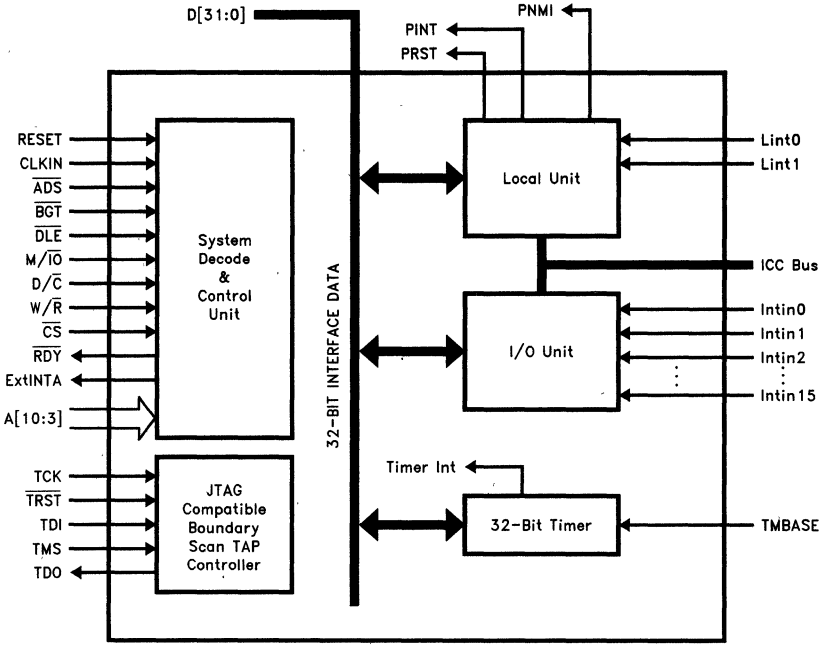
82489DX ADVANCED PROGRAMMABLE INTERRUPT CONTROLLER

82489DX FEATURES OVERVIEW

- Advanced Interrupt Controller for 32-Bit Operating Systems
- Solution for Multiprocessor Interrupt Management
- Dynamic Interrupt Distribution for Load Balancing in MP Systems
- Separate Nibble Bus (Interrupt Controller Communications (ICC) Bus) for Interrupt Messages
- Inter-Processor Interrupts
- Various Addressing Schemes—Broadcast, Fixed, Lowest Priority, etc.
- Compatibility Mode with 8259A
- 32-Bit Internal Registers
- Integrated Timer Support
- 33 MHz Operation
- 132-Lead PQFP Package, Package Type KU
  (See Packaging Specification. Order Number: 240800)

2

82489DX Block Diagram



290446-1

Refer to Application Note AP-388: 82489DX User's Manual (Order Number 292116) when evaluating your design needs.

The complete document for this product is available on Intel's "Data-on-Demand" CD-ROM product. Contact your local Intel field sales office, Intel technical distributor, or call 1-800-548-4725.

December 1994
Order Number: 290446-002

1.0 INTRODUCTION

The 82489DX Advanced Programmable Interrupt Controller provides multiprocessor interrupt management, providing both static and dynamic symmetrical Interrupt distribution across all processors.

The main function of the 82489DX is to provide interrupt management across all processors. This dynamic interrupt distribution includes routing of the interrupt to the lowest-priority processor. The 82489DX works in systems with multiple I/O subsystems, where each subsystem can have its own set of interrupts. This chip also provides inter-processor interrupts, allowing any processor to interrupt any processor or set of processors. Each 82489DX I/O unit Interrupt Input pin is individually programmable by software as either edge or level triggered. The interrupt vector and interrupt steering information can be specified per pin. A 32-bit wide timer is provided that can be programmed to interrupt the local processor. The timer can be used as a counter to provide a time base to software running on the processor, or to generate time slice interrupts locally to that processor. The 82489DX provides 32-bit software access to its internal registers. Since no 82489DX register reads have any side effects, the 82489DX registers can be aliased to a user read-only page for fast user access (e.g., performance monitoring timers).

The 82489DX supports a generalized naming/addressing scheme that can be tailored by software to fit a variety of system architectures and usage models. It also supports 8259A compatibility by becoming virtually transparent with regard to an externally connected 8259A style controller, making the 8259A visible to software.

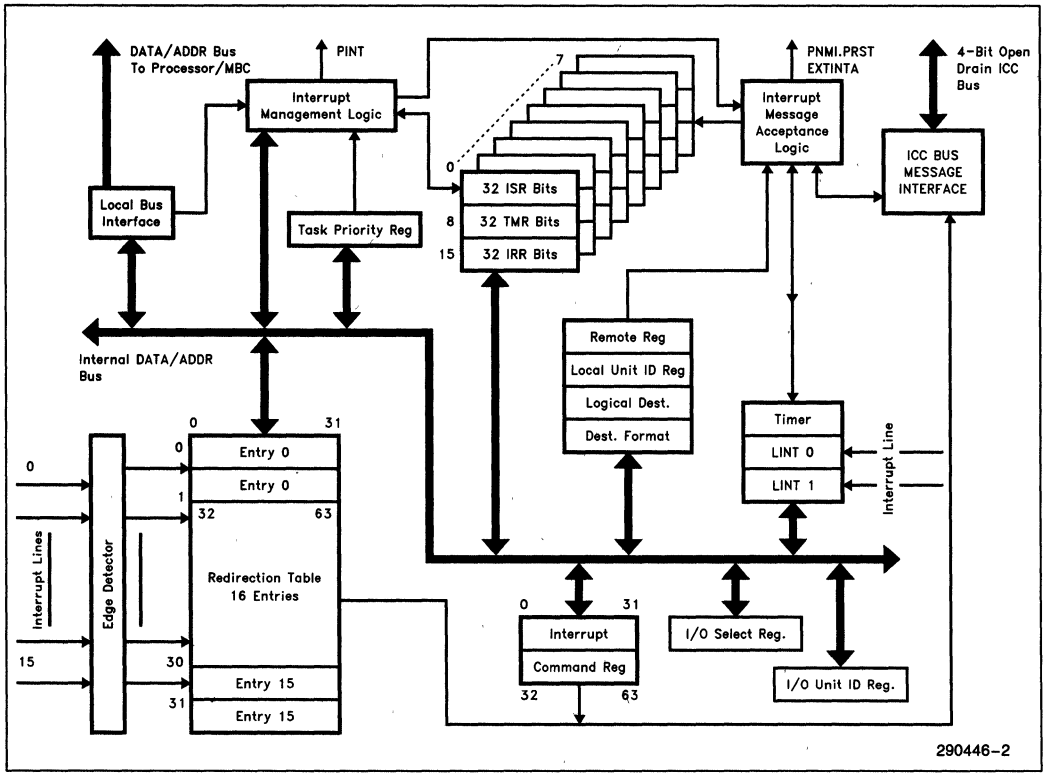


Figure 1. 82489DX Architecture

2.0 FUNCTIONAL OVERVIEW

82489DX Functional Blocks

82489DX contains one Local Unit, one I/O unit and a timer. The ICC bus is used to pass interrupt messages.

ICC BUS

The ICC bus is a 5-wire synchronous bus connecting all 82489DXs (all I/O Units and all Local Units). The Local Units and I/O Units communicate over this ICC bus. Four of these five wires are used for data transmissions and arbitration, and one wire is a clock.

LOCAL UNIT

The Local Unit contains the necessary intelligence to determine whether or not its processor should accept interrupt messages sent on the ICC bus by other Local Units and I/O Units. The Local Unit also provides local pending of interrupts, nesting and masking of interrupts, and handles all interactions with its local processor such as the INT/INTA/EOI protocol. The Local Unit further provides inter-processor interrupt functionality and a timer to its local processor. The interface of a processor to its 82489DX Local Unit is identical for every processor.

I/O UNIT

The I/O Unit provides the interrupt input pins on which I/O devices inject interrupts into the system in the form of an edge or a level. The I/O unit also contains a Redirection Table for the interrupt input pins. Each entry in the Redirection Table can be individually programmed to indicate whether an interrupt on the pin is recognized as either an edge or a level; what vector and also what priority the interrupt has; and which of all possible processors should service the interrupt and how to select that processor (statically or dynamically). The information in the table is used to send interrupt messages to all 82489DX Units via the ICC bus.

TIMER

The 82489DX provides a 32-bit wide timer that can be programmed to interrupt the local processor. The timer can be used as a counter to provide a time-base to software running on the processor, or to generate time-slice interrupts local to that processor.

2

3.0 PIN DESCRIPTION

The 82489DX pin description is organized in a small number of functional groups. Pin definitions and protocols have been designed to minimize interface issues. In particular, they support the notion of independently controlled address and data phases. The primary host interface is synchronous in nature.

In the following pin definition table if the signal name has (__) over it, the signal is in its active state when it has a low level. The signal direction column identifies output only signals as a continuous drive (O), tristate (T/S), or open drain (O/D). All bi-directional (BI-D) signals have tri-stating outputs.

Pin Definition Table

Symbol	Pin No.	Type	Function
SYSTEM PINS			
RESET	65	I	The **RESET INPUT** forces 82489DX to enter its initial state. The 82489DX Local Unit in turn asserts it PRST (Processor Reset) output. All tri-state outputs remain in high impedance until explicitly enabled.
ExtINTA	41	O	The **EXTERNAL INTERRUPT ACKNOWLEDGE** output is asserted (high) when an external interrupt controller (e.g., 8259) is expected to respond to the current INTA cycle. If deasserted (low), 82489DX will respond, and the INTA cycle must not be delivered to the external controller.
CLKIN	57	I	**CLOCK INPUT** provides reference timing for most of the bus signals.
$\overline{\text{TRST}}$	56	I	**TEST RESET** is the JTAG compatible boundary scan TAP controller reset pin. A weak pull-up keeps the pin high if not driven.
TCK	55	I	**TEST CLOCK** is the clock input for the JTAG compatible boundary scan controller and latches.
TDI	53	I	**TEST DATA INPUT** is the test data input pin for the JTAG compatible boundary scan chain and TAP controller. A weak pull-up keeps this pin high if not driven.
TDO	52	O	**TEST DATA OUTPUT** is the test data output for the JTAG compatible boundary scan chain.
TMS	54	I	**TEST MODE SELECT** is the test mode select pin for the JTAG boundary scan TAP controller. A weak pull-up keeps this pin high if not driven.
TIMER PIN			
TMBASE	59	I	The **TIME BASE** input provides a standard frequency that is only used by the 82489DX timer and that is independent of the system clock.
INTERRUPT PINS			
INTIN[15:0]	82–97	I	These 16 **INTERRUPT INPUT** pins accept edge or level sensitive interrupt requests from I/O or other devices. The pin numbers are specified respectively. INTIN15 corresponds to pin number 82, INTIN14 corresponds to pin number 83 etc., and INTIN0 corresponds to pin number 97. These pins are active high.
LINTIN[1] LINTIN[0]	80 81	I I	Two **LOCAL INTERRUPT INPUT** pins accept edge or level sensitive interrupt requests that can only be delivered to the connected processor. These pins are active high.
REGISTER ACCESS PINS			
$\overline{\text{ADS}}$	64	I	**ADDRESS STROBE** signal indicating the start of a bus cycle. 82489DX does not commit to start the cycle internally until BUS GRANT is detected active.

Pin Definition Table (Continued)

Symbol	Pin No.	Type	Function
REGISTER ACCESS PINS (Continued)			
M/$\overline{IO}$, D/$\overline{C}$, W/$\overline{R}$	63 61 62	I I I	Bus cycle definition signals. Note that since the 82489DX registers can be mapped in either memory or I/O space, the M/$\overline{IO}$ pin is not used for register access cycles; it is only used to decode interrupt acknowledge cycles. 82489DX does not respond to code read cycles.
$\overline{BGT}$	66	I	The **BUS GRANT** input is optional and is used to indicate the address phase of a bus cycle in configurations where address timing cannot be inferred from $\overline{ADS}$. This signal is really used as an address latch enable, but is named as it is to indicate that it can normally be connected to the Intel Cache Controller generated signal of the same name. Must be tied low if not used.
$\overline{CS}$	74	I	The **CHIP SELECT** input indicates that the 82489DX registers are being addressed.
A3 A4 A5 A6 A7 A8 A9 A10	31 29 28 27 26 24 22 21	BI-D BI-D BI-D BI-D BI-D BI-D BI-D BI-D	The address pins are used as inputs in addressing internal register space. Output function is reserved. They are also used to latch local unit ID on reset.
$\overline{DLE}$	73	I	**DATA LATCH/ENABLE** is optional and is used to indicate committing the data phase of a bus cycle in configurations where data timing cannot be inferred from other cycle timings. Must be tied low if not used.
D31 D30 D29 D28 D27 D26 D25 D24 D23 D22 D21 D20 D19 D18 D17 D16 D15 D14 D13 D12 D11	105 107 109 110 111 112 114 115 116 118 119 121 122 123 124 125 128 129 130 131 2	BI-D BI-D BI-D BI-D BI-D BI-D BI-D BI-D BI-D BI-D BI-D BI-D BI-D BI-D BI-D BI-D BI-D BI-D BI-D BI-D BI-D	The DATA BUS is for all register accesses and interrupt vectoring.

2

Pin Definition Table (Continued)

Symbol	Pin No.	Type	Function
REGISTER ACCESS PINS (Continued)			
D10 D9 D8 D7 D6 D5 D4 D3 D2 D1 D0	3 4 7 8 9 11 12 13 14 16 18	BI-D BI-D BI-D BI-D BI-D BI-D BI-D BI-D BI-D BI-D BI-D	
DP3 DP2 DP1 DP0	101 102 103 104	BI-D BI-D BI-D BI-D	One Data Parity pin for each byte on the data bus. EVEN parity is generated any time the data bus is driven by the 82489DX.
$\overline{RDY}$	43	O	**READY** output indicates that the current bus cycle is complete. In the case of a read cycle, valid data and the return to inactive state after going active low may be delayed till $\overline{DLE}$ goes active.
PROCESSOR PINS			
PINT	35	T/S	The **PROCESSOR INTERRUPT OUTPUT** indicates to the processor that one or more maskable interrupts are pending. This pin is tri-stated at reset, and has an internal pull-down resistor to prevent false signaling to the processor until the 82489DX Local Unit is enabled and this pin is actively driven.
PRST	38	O	The **PROCESSOR RESET OUTPUT** is asserted/de-asserted upon 82489DX reset, and also in response to ICC bus messages with "RESET" delivery mode. This pin should be used with care.
PNMI	37	T/S	The **NON-MASKABLE INTERRUPT** output is signaled in respone to ICC bus messages with "NMI" delivery mode. This pin is tri-stated at reset, and has an internal pull-down resistor to prevent false signaling to the processor until the Local Unit is enabled and this pin is actively driven.
ICC BUS PINS			
ICLK	60	I	The **ICC BUS CLOCK** input provides synchronous operation of the ICC bus.
MBI[3:0]	76–79	I	The four **ICC BUS IN** inputs are used for incoming ICC bus messages. In smaller configurations the ICC bus input and outputs may be tied directly together at the pins. Pin number for MBI3 is 76, MBI2 is 77, MBI1 is 78 and MBI0 is 79.
MBO3 MBO2 MBO1 MBO0	45 48 49 51	O/D	The four **ICC BUS OUT** outputs are used for outgoing ICC bus messages. The current capacity is only 4 mA. So external buffers will be needed.

Pin Definition Table (Continued)

Symbol	Pin No.	Type	Function
RESERVED PINS			
Reserved	34, 42	NC	These pins **MUST BE LEFT OPEN.**
Reserved	70, 72, 75		**Reserved** by Intel. **These pins should be strapped to V_{CC}.**
Reserved	71, 19, 20		**Reserved** by Intel. **These pins should be strapped to GND.**
POWER AND GROUND PINS			
V_{CC}	1, 32, 69, 98	POWER	Nominally +5V. These pins along with V_{SS} and V_{SSI} should be separately bypassed.
V_{CCP}	6, 15, 25,100, 108, 117, 126	POWER	Nominally +5V. These pins along with V_{SSP} should be separately bypassed.
V_{CCPO}	39, 46	POWER	Nominally +5V. These pins along with V_{SSPO} should be separately bypassed.
V_{SS}	5, 33, 67, 68, 99	GND	Nominally 0V. These pins along with V_{CC} should be separately bypassed.
V_{SSP}	10, 17, 23, 30, 106, 113, 120, 127, 132,	GND	Nominally 0V. These pins along with V_{CCP} should be separately bypassed.
V_{SSPO}	36, 40, 44, 47, 50	GND	Nominally 0V. These pins along with V_{CCPO} should be separately bypassed.
V_{SSI}	58	GND	Nominally 0V. These pins along with V_{CC} should be separately bypassed.

NOTE:
V_{CC}, V_{CCP} and V_{CCPO} should be of same voltage. V_{SS}, V_{SSP}, V_{SSPO} and V_{SSI} should be 0V.

4.0 FUNCTIONAL DESCRIPTION

As far as interrupt management is concerned, the 82489DX's interrupt control function spans over two functional units, the I/O Unit of which there is one per I/O subsystem, and the Local Unit of which there is one per processor. 82489DX has one I/O unit and one Local Unit in a single package. This section takes a detailed look at both local and I/O Units.

I/O Unit

The I/O Unit consists of a set of Interrupt Input pins, an Interrupt Redirection Table, and a message unit for sending and receiving messages from the ICC bus. The I/O Unit is where I/O devices inject their interrupts, the I/O Unit selects the corresponding entry in the Redirection Table and uses the information in that entry to format an interrupt request message. The message unit then broadcasts this message over the ICC bus. The content of the Redirection Table is under software control and is assigned benign defaults upon reset. The masks in the Redirectional Table entries are set to 1 at *hardware reset* to disable the interrupts.

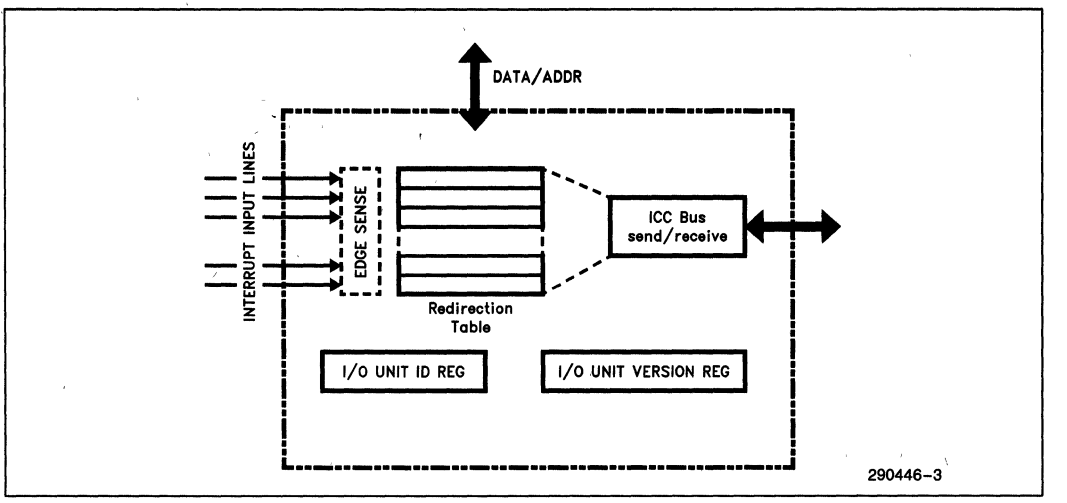


Figure 2. 82489DX I/O Unit Block Diagram

Local Unit

Interrupt Management of the Local Unit is responsible for local interrupt sources, interrupt acceptance, dispensing interrupts to the processor, and sending inter-processor interrupts. Depending on the delivery mode of the interrupt, zero, one or more units can accept an interrupt. A Local Unit accepts an interrupt only if it will deliver the interrupt to its processor. Accepting an interrupt is purely an inter-82489DX matter; dispensing an interrupt to the local processor only involves a 82489DX and its local processor.

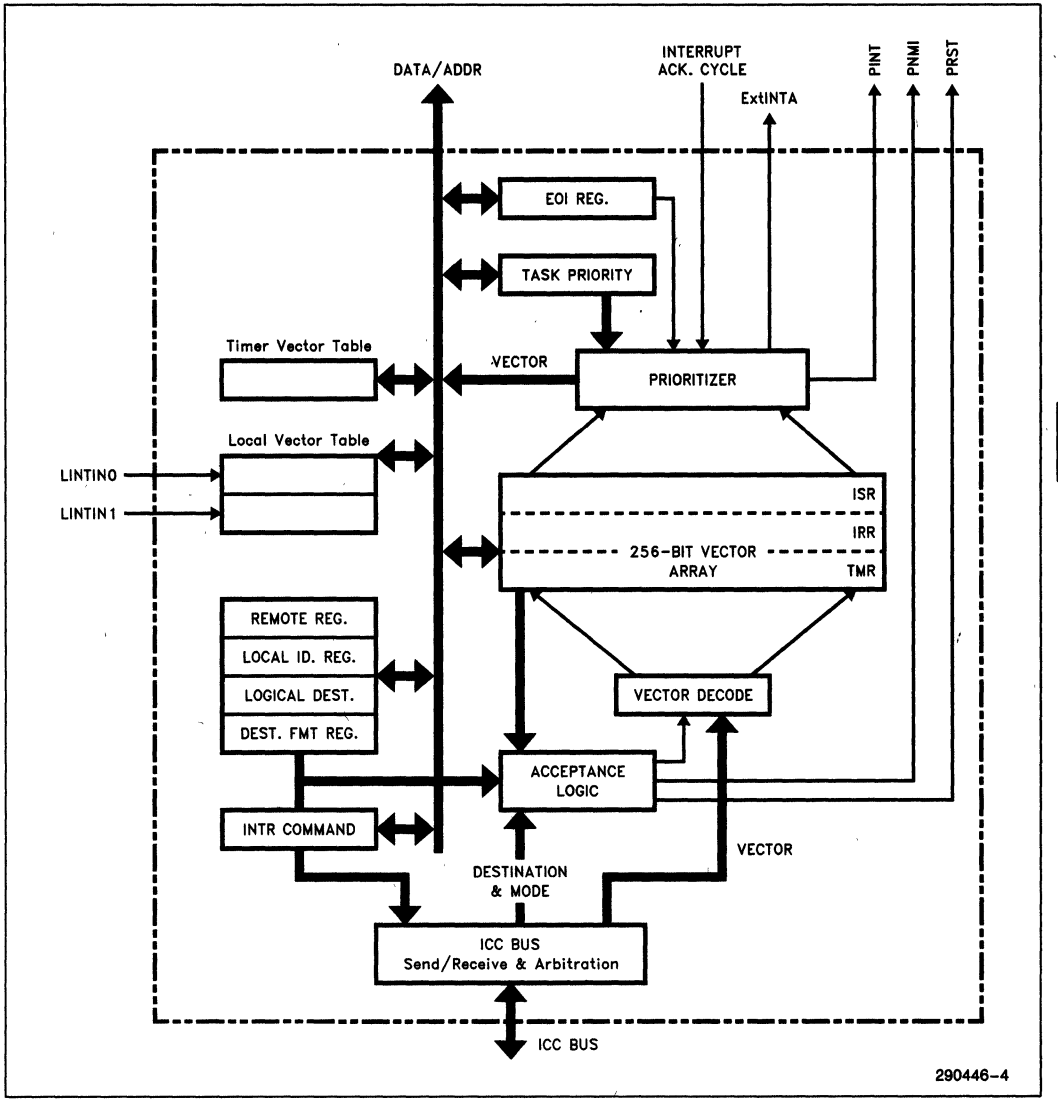


Figure 3. 82489DX Local Unit Block Diagram

5.0 INTERRUPT CONTROL MECHANISM

This section describes briefly the interrupt control mechanism in the 82489DX.

5.1 Interrupts

The interrupt control function of all 82489DXs are collectively responsible for delivering interrupts from interrupt sources to interrupt destinations in the multiprocessor system. When a processor accepts an interrupt, it uses the vector to locate the entry point of the handler in its interrupt table. The 82489DX architecture allows for 16 possible interrupt priorities; zero being the lowest priority and 15 being the highest. Priority of interrupt A "is higher than" the priority of interrupt B if servicing A is more urgent than servicing B. An interrupt's priority is implied by its vector; namely priority = vector/16.

With 256 vectors and 16 different priorities, this implies that 16 different interrupt vectors can share a single interrupt priority.

TOTAL ALLOWED INTERRUPT VECTORS

Out of 256 vectors, interrupt vectors 0 to 15 should not be used in the 82489DX. Only 240 interrupt vectors (vectors from 16 to 255) are supported in the 82489DX.

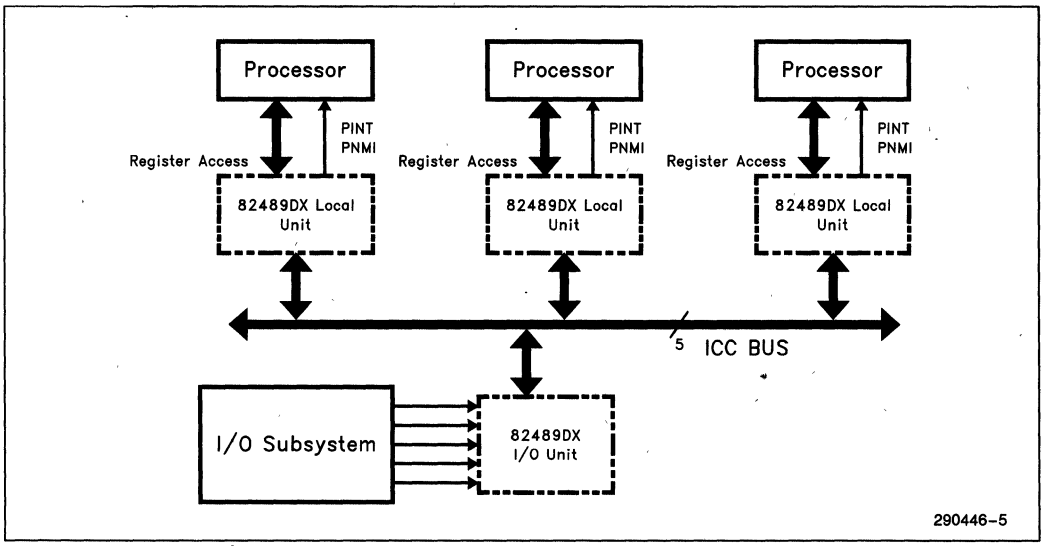


Figure 4. I/O Units and Local Units

INTERRUPT SOURCES

Interrupts are generated by a number of different interrupt sources in the system.

Possible interrupt sources are:

- Externally connected (I/O) devices. Interrupts from these external sources manifest themselves as edges or levels on interrupt input pins and can be redirected to any processor.
- Locally connected devices. These originate as edges or levels on interrupt pins, but they are always directed to the local processor only.
- 82489DX timer generated interrupts. Like locally connected devices, 82489DX timer can only interrupt its local processor.
- Processors. A processor can interrupt any individual processor or sets of processors. This supports software self-interrupts, preemptive scheduling, TLB flushing, and interrupt forwarding. A processor generates interrupts by writing to the interrupt command register in its Local Unit.

INTERRUPT DESTINATIONS

I/O Units can only source interrupts whereas Local Units can both source and accept interrupts, so whenever "interrupt destination" is discussed, it is implied that the Local Unit is the destination of the interrupt. In physical mode the destination processor is specified by a unique 8-bit 82489DX local ID. Only a single destination or a broadcast to all (LOCAL ID of all ones) can be specified in physical destination mode.

In logical mode destinations are specified using a 32-bit destination field. All Local Units contain a 32-bit Logical Destination register against which the destination field of the interrupt is matched to determine if the receiver is being targeted by the interrupt. An additional 32-bit Destination Format register in each Local Unit enables the logical mode addressing.

INTERRUPT DELIVERY

The description of interrupt delivery makes frequent use of the following terms:

- Each processor has a processor priority that reflects the relative importance of the code the processor is currently executing. This code can be part of a process or thread, or can be an interrupt handler. A processor's priority fluctuates as a processor switches threads, a thread or handler raises and lowers its priority level to mask out interrupt, and the processor enters an interrupt handler and returns from an interrupt handler to previously interrupted activity.
- A processor is lowest priority within a given group of processors if its processor priority is the lowest of all processors in the group. Note that more than one processor can be the lowest priority in a given group.
- A processor is the focus of an interrupt if it is currently servicing that interrupt, or if it currently has a request pending for the interrupt.

Interrupt delivery begins with an interrupt source injecting its interrupt into the interrupt system at one of the 82489DX. Delivery is complete only when the servicing processor tells its 82489DX Local Unit it is complete by issuing an end-of-interrupt (EOI) command to its 82489DX Local Unit. Only then has all (relevant) internal state regarding that occurrence of the interrupt been erased. The interrupt system guarantees exactly-once delivery semantics of interrupts to the specified destinations. Exactly-once guaranteed delivery implies a number of things:

- The interrupt system never rejects interrupts; it never NAKs interrupt injection, interrupts are never lost, and the same interrupt (occurrence) is never delivered more than once.

Clearly a single edge interrupt or level interrupt counts as a single occurrence of an interrupt. In uniprocessor systems, an occurrence of an interrupt that is already pending (IRR) cannot be distinguished from the previous occurrence. All occurrences are recorded in the same IRR bit. They are therefore treated as "the same" interrupt occurrence.

For lowest-priority delivery mode, by delivering an interrupt first to its focus processor (if it currently has one), the identical behavior can be achieved in a MP (Multiprocessor) system. If an interrupt has a focus processor then the interrupt will be delivered to the interrupt's focus processor independent of priority information. This means that even if there is a lower priority processor compared to the focus processor, the interrupt still gets delivered to the focus processor.

Each edge occurring on an edge triggered interrupt input pin is clearly a one-shot event; each occurrence of an edge is delivered. An active level on a level triggered interrupt input pin represents more of a "continuous event". Repeatedly broadcasting an interrupt message while the level is active would cause flooding of the ICC bus, and in effect transmits very little useful information since the same processor (the focus) would have to be the target.

Instead, for level triggered interrupts the 82489DX merely recreate the state of the interrupt input pin at the destination . The source 82489DX accomplishes this by tracking the state of the appropriate destination 82489DX's Interrupt Request Register (or pending bit) and only sending inter-82489DX messages when the state of the interrupt input pin and the destination's interrupt request enter a disagreement. Unlike edge triggered interrupts, when a level interrupt goes into service, the interrupt request at the servicing 82489DX is not automatically removed. If the handler of a level sensitive interrupt executes an EOI then that interrupt will immediately be raised to the processor again, unless the processor has explicitly raised its task priority, or the source of that interrupt has been removed.

5.2 Interrupt Redirection

This section specifically talks about how a processor is picked during interrupt delivery. The 82489DX supports two modes for selecting the destination processor: Fixed and Lowest Priority.

- *Fixed Delivery Mode*
  In fixed delivery mode, the interrupt is unconditionally delivered to all local 82489DXs that match in the destination information supplied with the interrupt. Note that for I/O device interrupts typically only a single 82489DX would be listed in the destination. Priority and focus information are ignored. If the priority of a destination processor equal to or higher than the priority of the interrupt, then the interrupt is held pending locally in the destination processor's Local Unit, until the processor priority becomes low enough at which time the interrupt is dispensed to the processor. More than one processor can be the destination in fixed-delivery mode.
- *Lowest Priority Delivery Mode*
  Under the lowest priority delivery mode, the processor to handle the interrupt is the one in the specified destination with the lowest processor priority value. If more than one processor is at the lowest priority, then a unique arbitration ID is used to break ties. For lowest priority dynamic delivery, the interrupt will always be taken by its focus processor if it has one. The lowest priority delivery method assures minimum interruption of high priority tasks. Since each Local Unit only knows its own processor priority, determining the lowest priority processor is done by arbitration on the ICC bus. Only one processor can be the destination in lowest-priority delivery mode.

INTER-82489DX COMMUNICATION

All I/O and Local Units communicate during interrupt delivery. Interrupt information is exchanged between different units on a dedicated five wire ICC bus in the form of broadcast messages. A 82489DX Unit's 8-bit ID is used as its name for the purpose of using the ICC bus, and all 82489DX units using one ICC bus should be assigned a different ID. The Arbitration ID of the Local Units used to resolve ties during lowest priority arbitration is also derived from the Local Unit's ID.

16.0 GUIDELINES FOR 82489DX USERS

16.1 Initialization

This section outlines one possible initialization scenario. Other scenarios are certainly possible, and one would be selected as part of a platform standard initialization scheme. The intent of this section is to illustrate that the initialization support provided by the 82489DX is adequate to support MP (Multiprocessor) system initialization.

Each 82489DX has a RESET input pin connected to a common Reset line. Upon system reset, this common reset line is activated, causing all the 82489DXs to go through reset. All 82489DX local units (note: only local units and not I/O units) latch their ID from their address bus on reset. The ID can be provided by the bus control agent based on slot number.

The local units next assert their processor's Reset pin, holding the processor in reset, and next perform their internal reset, setting all registers to their initial state. The initial state of all 82489DX Units (both local and I/O units) is "all masks set" and all Local

Units disabled; registers are otherwise initialized to zero. Note that the PINT and PNMI output pins are in tri-state mode when the local unit is disabled. After this, each 82489DX local unit will deassert its processor's Reset pin, allowing the processors to come out of reset and perform self test and start executing initialization code.

Note that while connecting PRST pin it should be noted that whenever PRST pin is activated by 82489DX either because of software reset message or hardware reset, the 82489DX itself is reset. It should be taken care in the cases of Warm reset where only processors need to be reset and not the interrupt controller. In brief, the usage of PRST depends upon the system requirement on various reset.

Somewhere in this code sequence, the processors that are "alive" will enable their 82489DX local units, and attempt to force all the other processors back into Reset. Forcing the other processors into reset is performed by sending them the inter-processor interrupt with Destination Mode = "Physical", Delivery Mode = "Reset", Trigger Mode = "Level", Level = "1", and Destination Shorthand = "All Excl Self". Only the first processor to get the ICC bus will succeed in sending this signal and reset all other 82489DXs and their processors. The other processors are kept in reset until such time that an MP operating system decides they can become active again. The only running processor next performs the rest of system initialization.

Eventually, an MP operating system will be booted at which time the operating system would send "deassert reset" interprocessor signals to activate the other processors in the system. A mechanism must be provided by the platform that allows the added processors to differentiate the very first reset from a subsequent one.

16.2 Compatibility

COMPATIBILITY LEVELS

The 82489DX can be used in conjunction with standard 8259A-style interrupt controllers to provide a range of compatibility levels.

At the lowest level we have "PC shrink-wrap" compatibility. This level effectively creates a uniprocessor hardware environment within the MP platform capable of booting/running DOS shrinkwrap software. In this mode, only the 8259A generates interrupts and the 82489DX becomes a virtual wire. The interrupt latency can be minimized by connecting the 8259A interrupt to local unit directly.

The next level preserves the software compatible view of an 8259A but it allows more than one processor to be active in the system. This results in an asymmetrical arrangement, with one processor fielding all 8259A interrupts but with added inter-processor interrupt capability. In this mode, 82489DX "merges" 8259A interrupts with inter-processor interrupts. Existing I/O drivers would be bound to the compatible CPU and interface directly with the 8259A.

At the next compatibility level, 8259A compatible drivers can be mixed with native 82489DX drivers. Devices can generate interrupts at either 8259A or an 82489DX. This provides for partial symmetry as individual drivers migrate from the 8259A to native 82489DXs.

Another 8259A compatible point can be defined for MP systems. Each processor could have its own compatible 8259A controllers, allowing multiple processors to run compatible I/O drivers, but statically spreading the load across the available processors.

82489DX/8259A INTERACTION

The principle of compatible operation is very straightforward; the 82489DX(s) become a virtual wire connecting the 8259A's INT output through to the processor, while at the same time making 8259A visible to the processor.

The two connection schemes described only differ in the number of 82489DX(s) (one or two) that are located in the path from the 8259A to the processor. In the one 82489DX example illustrated in Figure 37, the INT output of the 8259A connects to one of the Interrupt Input pins of the 82489DX through an edge generation logic. This could be an interrupt pin on the 82489DX's I/O unit or local unit; assume a local interrupt input is used. The Local Vector Table entry for the interrupt pin that connects to the 8259A is set up with a Delivery Mode of "ExtINT" and edge trigger mode. This indicates that the interrupt is generated by an external controller. The processor's INT pin connects to the 82489DX PINT pin.

This setup enables the 82489DX local unit to detect assertions (up-edges) of the 8259A's INT output pin and pass this on to the processor's INT input. 82489DX asserts ExtINTA pin along with (one clock prior to) PINT pin to indicate "8259" interrupt. When the processor performs its INTA cycle the 82489DX itself does not respond other than deasserting PINT to the processor. At the third clock after $\overline{ADS}$ in the second bus cycle of INTA cycle ExtINTA is deasserted. External logic should make use of the ExtINTA signal to make the INTA cycle visible to the 8259A and the 8259A should provide the vector. At the same time, the local unit considers the external request as delivered, and need not wait for the external 8259A's INT to be deasserted. *A new up-edge must be generated on the 8259A INT pin before the local unit will assert the processor's INT pin on behalf of the 8259A. External edge generation logic should be used for this.* Compatible software interacts directly with the 8259A.

The mechanism is essentially the same in the two-82489DX scheme. The difference is that the 8259A connects to an interrupt input pin of the 82489DX I/O unit in the I/O system. The Redirection Table entry for this pin is again programmed with an "ExtINT" Delivery Mode, and the (single) 82489DX destination local ID corresponding to the compatible DOS processor. Capturing the up-edges of the 8259A's INT pin by the 82489DX local unit now involves sending messages from the 82489DX I/O unit to the 82489DX local unit via the ICC bus. The "virtual wire" now includes messages over the ICC bus.

Adding inter-processor ICC interrupts (or any other 82489DX generated interrupts) to the compatible operation is accomplished by having the 82489DX internally OR the 8259A's INT request with any 82489DX interrupt request.

Before the 82489DX actually sends the interrupt signal to the processor, the 82489DX decides whether it does this for an 82489DX interrupt or whether it does this on behalf of the external controller. When the processor performs the corresponding INTA cycle, only the 82489DX knows whether it should respond with a vector, or whether the external 8259A should.

If the 82489DX needs to respond, then it will enable an externally implemented trap that prevents the 8259A from seeing the INTA cycle. If the 8259A needs to respond, then the 82489DX will not enable the INTA trap, and the INTA will be allowed to reach the 8259A. 82489DX implements this by asserting its EXTINTA pin to indicate external 8259A should respond with the vector. The 82489DX local unit controls the INTA trap via its "ExtINTA" output pin; the 82489DX does not actually provide the trap itself.

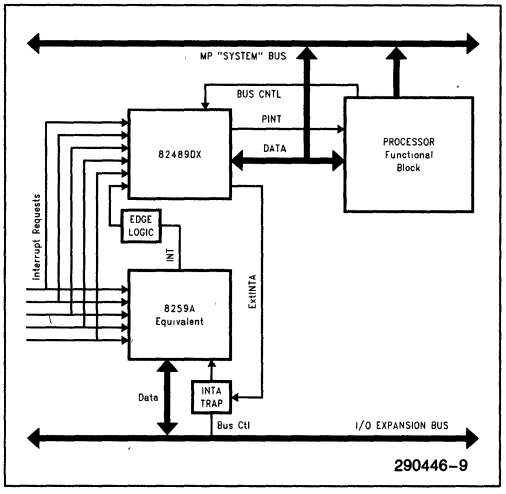


Figure 37. Edge Logic

82489DX/8259A DUAL MODE CONNECTION

In systems that can be booted either as a configuration with compatible 8259A or without, device interrupt lines are connected to both the Interrupt Request pins of the 8259A and Interrupt Input pins of the 82489DX with all interrupts either masked at the 82489DX or at the 8259A. Some EISA and Micro-Channel chip sets that include on-chip 8259As also have internally connected interrupt requests. For example, the 82357 (the ISP of the EISA chipset) generates timer and DMA chaining interrupts internally. These are not available as separate interrupts outside the ISP. In non-compatible mode the ISP timers are not used, since each local 82489DX unit provides its own timer. Therefore, the ISP's 8259A is configured to mask out all interrupts except the DMA chaining interrupt which is configured in level-sensitive, auto EOI mode. This causes the 8259A's INT output to track the state of the internal DMA interrupt request. The 8259A's INT output is then connected to one of the 82489DX interrupt input pins programmed to generate a regular (i.e., not "ExtINT") level-sensitive interrupt. The ISP 8259A then no longer functions as an external interrupt controller; it has been logically disabled, and it needs no interrupt acknowledge or EOI. The INTA and EOI cycles occur only at the 82489DX. It should be noted that 82489DX accepts only active high level/edge interrupt inputs. External programmable logic should take care of polarity reversal that may be needed in EISA system for sharing of interrupts.

intel®

AP-388

APPLICATION NOTE

2

82489DX User's Manual

M. JAYAKUMAR
MULTIPROCESSING TECHOLOGY GROUP

November 1994

PRELIMINARY
Order Number: 292116-002

82489DX User's Manual

CONTENTS

CONTENTS

INTRODUCTION

82489DX is the new interrupt controller for high performance systems and 32-bit OS. Some important considerations for hardware designers are given. This application note will provide information of all registers in 82489DX and their bits and bytes organization. The control word for various programming options are given in a tabular format. Some programming hints are given to facilitate a quick understanding of the interrupt architecture and the priority model in 82489DX.

The programming model discusses the registers, their data structure like fields, bits, bytes and default register values. The system considerations and key points to be noted while programming 82489DX are discussed next. Typical examples of initialization, interrupt service routine and Spl() routines are given. The notes discuss important hardware design considerations.

Related Reference Materials

1) 82489DX Data Book, Order Number 290446.
2) An APIC based Symmetric Multiprocessor System Design AP-474, Order Number 241521.

REGISTER ORGANIZATION

The 82489DX contains both the local unit and I/O unit. I/O unit has its own Unit ID and local unit has its own Unit ID. Both units are operational at all times once they are enabled and the access can be done to both units. It should be noted that the local unit has its own version register, and I/O unit has its own version register, namely, I/O version register. The unit enable bit is provided for local unit and it is not provided for I/O unit. However, I/O unit has mask bit for each redirection table entry to mask the interrupts. Functionally I/O unit can only transmit interrupt messages whereas local unit can both transmit and receive interrupt messages. In summary, 82489DX should be viewed as an integrated chip having a local unit and an I/O unit both capable of operating at the same time.

INITIAL REGISTER VALUES AFTER HARDWARE RESET

The local unit ID register latches the value on the address pins A3 to A10 after hardware reset whereas the I/O unit ID register gets cleared to 0 after hardware reset. The local unit Version Register is cleared to 0 whereas the I/O unit Version Register contains 1111 in its Max Redir Entry field. The interrupt masks in the local timer vector table register and in the I/O redirection table entry(31:0) registers are set so that after reset all the interrupts are masked. The spurious vector register's unit enable bit is cleared so that local unit is disabled after hardware reset. Since all the interrupts are masked after hardware reset, the I/O unit will not transmit any interrupt after hardware reset until mask is cleared specifically by software and the interrupt is active.

All other registers are cleared to 0 after hardware reset.

SYSTEM CONSIDERATIONS WHILE PROGRAMMING THE 82489DX

The 82489DX register data structure contains different fields to specify the mode of operations and the options available within each mode. Since certain options are applicable to specific modes only (for example "Remote Read" mode applies only to Interrupt Command Register, it does not have any relevance to I/O unit's redirection tables) the following programming hints are provided.

82489DX and Memory Mapping

The 82489DX is a 32-bit high performance interrupt controller. It allows the CPU to do 32-bit read and write to it. By memory mapping the 82489DX, system performance can be enhanced. Even though the 82489DX can be memory mapped, its functionality as an interrupt controller should be kept in mind while programming the virtual memory management control data structure. The caching policy for the page where an 82489DX is mapped should also be done with the functionality of the 82489DX in mind. For example, the reads to an 82489DX should not be cached and writes should be write-through. Since 82489DX registers are aligned at 128-bit boundaries, memory mapping the 82489DX with interleaved memory system should not be a problem. However, it should be noted that the 82489DX does not support pipelining.

Unique ID Requirement

All the local units and I/O units hooked on an I_{CC} bus should have an unique ID before they can use the bus. This should be ensured by the programmer, since for I_{CC} bus arbitration the units (whether it is local unit or I/O unit) arbitrate with their unit ID.

PROGRAMMING THE LOCAL UNIT

Dos and Don'ts

1. The local interrupt vector table entry (and the I/O unit redirection table entry) should not be programmed for "Remote Read" Delivery mode. In other words, only Interrupt Command register supports "Remote Read" Delivery mode.
2. Local Interrupts should not be programmed with "Lowest Priority" Delivery mode.
3. Local Interrupts should not be programmed with "Reset" Delivery mode.
4. It is not recommended to use level triggered mode except for "Reset Deassert" messages.

Atomic Write Read to Task Priority Register

This section discusses issues regarding write buffer flushing and necessity of atomicity of task priority register programming.

Typically, the task priority register is written with higher priority to mask certain low level interrupts before entering into a critical section code. In a system where an 82489DX is memory mapped the CPU may buffer this task priority register write to its on chip write buffer. The following scenario can happen in such situation: CPU posts task priority register write to its on chip write buffer and enters into the critical code. A lower priority interrupt (which should not enter the critical code) interrupts the CPU before the write buffer gets flushed into task priority register). The CPU now erraneously accepts the lower priority interrupt. To avoid the situation, atomic write and read to task priority register should be done. The read following write ensures that the write buffer is flushed to task priority register and the atomicity ensures that no interrupt will be accepted by the CPU during its write to task priority. In case if the CPU itself takes care of flushing its write buffers before INTA cycle, there is no problem. However, if there are system posted write buffers then external logic should make sure to flush the system write buffers before INTA-cycle.

Task Priority Register and Total Usable Vectors

Task priority register is used to specify the priority of the task the processor is executing. In 8259 the priority is defined only among the interrupts that it handles. 82489DX goes further ahead in handling priority. In multitasking system, in addition to device interrupts, various tasks have different priority and 82489DX allows consideration of the priority at system level. The processor specifies the priority of the task it executes by writing to task priority register. Now any interrupts *at or below* the task priority will be masked until the task priority gets lowered. The masking granularity is at priority level. Out of 256 interrupt vectors 16 priority levels are specified and 16 vectors share one priority level. Since the masking granularity by the task priority register is at priority level, group of 16 vectors get masked when a local unit increases its task priority by one level.

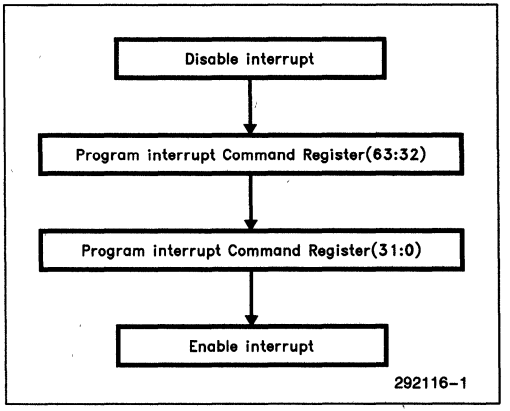
When task priority register is at its minimum level of 0, interrupt vectors having level 1 to 16 are passed to CPU. Stated in other words, even when the task priority register is at its minimum (level of 0), interrupt vectors at level 0 will be masked. This means that the interrupts should not be programmed with vectors 0 to 15. So out of 256 interrupt vectors, only 240 interrupt vectors (vector 16 to 255) can be used in 82489DX.

ISR/IRR/TMR

1. Bits 0–15 of IRR/ISR/TMR do not track interrupt. No interrupt of vector number from 0–15 can be posted. The total interrupts supported are 240. This can be easily explained by the way the priority mechanism is defined. When reading the lowest 32 bits of this register, 0 will always be returned for the lower 16 bits.

Interrupt Command Register Programming Considerations

The interrupt command register (31:0) has the side effect of sending interrupt once it is written. There is no mask bit associated with Interrupt Command Register. Once interrupt command register (31:0) is written, the interrupt is sent from the local unit. The interrupt destination is provided in the interrupt command register (63:32). So, the interrupt command register (63:32) should always be programmed before the interrupt command register (31:0) is programmed.



CRITICAL REGIONS AND MUTUAL EXCLUSION

This section discusses the reasons for mutual exclusion to be exercised when writing to interrupt command register. Each 82489DX has a single Interrupt Command Register that is used to send interrupts to other processors. The programmer should make sure to synchronize access to this register. Specifically, (1) writing all fields of the register (MSB), (2) Sending the interrupt message (by writing the LSB register), and (3) waiting for Delivery State to become Idle again, should occur as a single atomic operation. For example, if interrupt handlers are also allowed to send inter processor interrupts, then interrupt dispensing to the processor must be disabled for the duration of these activities so that interrupt handlers are excluded from accessing the ICR. This is explained as follows. Let us assume in a typical MP system preemptive scheduling (on another processor) is implemented by sending inter processor interrupts (IPIs). IPI can be also used for clock distribution in an asymmetric system where the timer interrupts only one processor and that processor notifies all other processors in the system through IPI. Inter processor interrupts are implemented by using interrupt command register. If we allow interrupts during writing interrupt command register the following erraneous operation may result. If interrupts are enabled (they should not be) during writing to interrupt command register, interrupts can come after writing to MSB portion and before writing to LSB portion of interrupt command register. Now in the interrupt service routine, if ICR is used (for distribution of interrupt to other processor(s), for example) then this ISR also starts writing to the Interrupt Command Register. That means the ISR will overwrite the MSB portion just written by the previous IPI. After returning from the ISR when the previous IPI continous writing to the remaining LSB portion, the message will be delivered to wrong address since MSB is modified by the module which interrupted. **The inference is that while accessing ICR interrupts should be disabled.** Also it should be noted that except for "Reassert Deassert Messages", IPI should only use edge triggered mode.

BUFFERING IN INTERRUPT COMMAND REGISTER

The Interrupt Command Register provides one level of buffering which should be kept in mind while programming an 82489DX. The ICR (Interrupt Command Register) becomes busy as soon as inter processor message is written into it. It hands the message over to I_{CC} bus transmit unit which in turn tries to send through I_{CC} bus. Since the ICR has passed the command to transmit Unit (whose responsibility is to send it through I_{CC} bus) it becomes free. The software before writing next inter processor message reads the flag to be free and writes next message. Thus there is a possibility of next message being written into the 82489DX before the first message is really sent out. The programmer should be aware of this.

INTERRUPT COMMAND REGISTER DO'S AND DON'TS

1. "ExtINTA" delivery mode should not be used for all destination shorthand.
2. "Remote Read" should always be programmed as "Edge" triggered interrupt.
3. "Remote Read" should always be programmed with physical Destination mode (and not with Logical Destination mode).
3. Only Fixed Delivery Mode should be used for "Self" destination shorthand. Stated otherwise, "lowest priority", "Remote Read", "Reset", "NMI" delivery modes do not apply for "Self".
4. For "All incl. self" and "All excl Self" destination shorthands, "Remote Read" delivery mode should not be used.
5. For "All incl. self" and "Self" destination shorthands "Reset" Assert mode should not be used.
6. For "All exclusive self" destination shorthand if "Reset ASSERT" delivery mode is used, it should be ensured at system level that only one processor executes this instruction at any time. To explain this, let us consider the following situation. Let us assume that two CPUs, CPU A and CPU B are executing "Reset ASSERT, All Exclusive Self". The message of CPU A puts every CPU except CPU A in reset state. After the message is written by CPU A it typically takes 2.9 μs for the message to flow through the I_{CC} bus to reach other local units to reset all other processors. Before this message resets, let us assume another processor also, say CPU B, issues the "Reset ASSERT, All Exclusive Self" message. The following CPU B message (which was sent out before CPU B itself got reset because of CPU A reset message) will reset every CPU, which will include CPU A, except CPU B. But CPU B will eventually get reset by the message sent by CPU A and CPU A will also get reset by the message sent by CPU B. Thus all the CPUs in the system goes into reset state and this is an irrecoverable state. To avoid this, only one processor should execute this instruction at any time. This can be achieved, for example, by spinlock or mutex implemented as shared variables between multiprocessors.
7. Messages could be sent out in "Logical" or "Physical" mode with destination ID of all 1's depending on the way Destination Mode entry is programmed. In brief, "All incl. self" and "All excl. self" supports both "Logical" and "Physical" addressing mode.
8. When destination shorthand (*i.e.*, broadcast) is used with "lowest priority" destination mode, then even though all participates in arbitrating for destination, only the lowest priority gets the message. So even though the addressing is broadcast since the destination mode is lowest priority only one gets the message.

9. When destination shorthand (*i.e.*, broadcast) is used with "Fixed" destination mode, then all the units get the message. So to send messages to all units Fixed destination mode should be used in addition to using destination shorthand.
10. It is recommended that all IPI messages, except for "Reset Deassert", use only edge triggered interrupt mode.

IPI THROUGH INTERRUPT COMMAND REGISTER

Interrupt command register can be used to send inter processor interrupt. Inter processor interrupts can be used for preemptive scheduling, TLB flushing, clock distribution, etc. IPIs can also used in asymmetric systems to pass certain work to another processor who has exclusive access to certain piece of hardware. Let us consider a dual processor system which uses only two 82489DX in the whole system. Since the local units should be accessible only by their respective processor a local unit should be selected only when the arbitrator grants the bus to its processor. Since 82489DX has common chip select for its local unit and I/O unit, for logical simplicity, system hardware may select a 82489DX when the corresponding processor is granted bus. Because of this, processor A can access only I/O unit and local unit that are available in its 82489DX. It is not possible to access the I/O unit of the other 82489DX. The same thing holds good for the other processor. Since I/O unit should be globally visible to both processors, there may be situations when a processor may want to access the other I/O unit. This is typically the case for enabling and disabling the I/O interrupt. IPI can be used to pass that task to the other processor which can access that I/O unit. This is just one example for using IPI.

ExtINTA INTERRUPT POSTING

ExtINTA interrupts are used to support 8259 in a 82489DX based system. The external interrupts (ExtINTA) are specific in their characteristics in that they do not have any priority relationship with rest of the interrupt structure. But when posting an interrupt to the processor, if both an external interrupt and a 82489DX interrupt are pending, 82489DX could post either one to the processor. In 82489DX implementation, it would post external interrupt whenever there is no other 82489DX interrupt that can be posted to the processor. It should be also noted that External Interrupts can not be masked by raising task priority. However, they can be masked by the mask bit in the table entry for that (ExtINTA) interrupt.

Since ExtINTA interrupts do not have any priority relationship, ISR and IRR bits are not maintained for external interrupts. As far as interrupt acceptance is concerned, if more than one ExtINTA interrupts are directed towards a local unit, that local unit treats all the ExtINTA interrupts directed to it as only one ExtINTA interrupt. This leads to an important point that in a system no more than one interrupt should be programmed as ExtINTA interrupt type with the **same destination**. However, it should be noted that there can be more than one ExtINTA' type of interrupt in a system with each having different local unit as destination.

LOWEST PRIORITY

Under the lowest priority delivery method, the processor to handle the interrupt is the *one* in the specified destination with the lowest processor priority value. If more than one processor is at the lowest priority, then a unique arbitration ID is used to break ties. To have unique arbitration ID in the system (which is mandatory for the lowest priority algorithm to work) all the arbitration ID of local 82489DXs in the system should be in sync. On reset, arbitration ID is reset to zero by the hardware. Hence all the local units in the system after reset will have same arbitration ID (namely zero). For lowest priority arbitration to work properly we need to have unique arbitration ID in the system. This means after local unit IDs are written in all local units (obviously, each unit ID should be different from other IDs) a RESET DEASSERT message should be sent in ALL INCLUSIVE mode. The important side effect of RESET DEASSERT message is that it copies the unitID into the respective arbitration ID. Since unit IDs are unique, the RESET DEASSERT message ensures that the arbitration ID also are unique in the system. This RESET DEASSERT message should be sent before system is used for lowest priority arbitration.

The RESET DEASSERT message, if not sent, only once delivery semantics may not be guaranteed. If RESET DEASSERT message is not sent then all the arbID in the system will be same. When a message is sent in the lowest priority arbitration, the participating local units use their processor priority concatenated with arbitration ID to decide the destination. Processor priority is derived from the task priority. There is a chance that two local units can have same task priority depending on the code they are executing and thereby same processor priority. In addition since arbID are also same if RESET DEASSERT message WAS NOT sent, all the processors in the same priority may accept the message in lowest priority arbitration. This violates the only once delivery semantics. The inference is that RESET DEASSERT message in ALL INCLUSIVE SELF mode should be sent as part of initialization before enabling interrupt in the lowest priority destination scheme.

It should be noted that only once delivery semantics for a group destination is guaranteed only if multiple fixed delivery of the same interrupt vector are not mixed.

DISABLING LOCAL UNIT

Once the 82489DX is enabled by setting bit 8 of spurious vector register to 1, the user should not disable the local unit by resetting the bit to 0. The result will put the local unit in an inconsistent state. However, a local unit can be disabled by getting "reset" interrupt message from any other local unit across the I_{CC} bus.

ISSUING EOI

EOI, End of Interrupt issuing indicates end of service routine to 82489DX. Always the highest priority ISR bit which is set during INTA cycle gets cleared by EOI. This section discusses the relevance of EOI to the specific types of interrupts and its timing related to interrupt deassertion.

EXTERNAL INTERRUPTS AND EOI

External Interrupts (ExtINTA) should be programmed as edge type. INTA cycles to external interrupts are taken automatically as EOI by 82489DX. This is similar to AEOI, Automatic End of Interrupt of 8259A. So EOI should not be issued to 82489DX for ExtINTA interrupt servicing. For ExtINTA type of interrupts, there is no need to have interrupt service routines since the main purpose of ExtINTA interrupt itself is to have software transparency in the compatible mode. The existing interrupt service routines written for 8259 will be executed by the processor for ExtINTA interrupts.

SPURIOUS INTERRUPTS AND EOI

Spurious Interrupts do not have any priority relationship to other interrupts in the system. So IRR is not set for spurious interrupts. EOI should not be issued for spurious interrupts. It is advisable not to share the spurious interrupt vector with any interrupt.

If spurious interrupt vector is shared with some other interrupt then the following guidelines should be followed. If the source is spurious interrupt (for which the corresponding ISR is **not** set) then EOI should **not** be issued. If the source is a valid interrupt sharing the spurious interrupt vector (for which the corresponding ISR is set) then EOI should be issued.

NMI AND EOI

For NMI type of interrupt no IRR bit is set. So, obviously EOI should not be issued while servicing NMI type of interrupts.

PROGRAMMING I/O UNIT

Interrupt Sharing Considerations

Two different interrupts should not be programmed with the same interrupt vector. This means that each redirection table in a system should have unique vector. Interrupt sharing can be done electrically. Interrupts connected at different interrupt input pins of 82489DX CAN NOT share interrupt by having same vector. 82489DX does not support active low interrupts. So sharing interrupts should have polarity logic support externally.

I/O Unit and Priority

The 82489DX partitions its interrupt control function among two different units:

1. I/O unit
2. local unit

The priority resolving is done at local unit. The I/O unit does not involve itself in the priority mechanism. The I/O unit takes a snapshot of interrupts pending at the INTIN interrupt input pins. If interrupts are active, it starts sending the interrupt messages over I_{CC} bus. It starts sending the lowest numbered interrupt input first. That is if INTIN0 and INTIN5 are found active in a snapshot, interrupt message corresponding to INTIN0 is sent first regardless of the priority of the vectors that are associated with these interrupts. It sequentially sends all the interrupts found active in a snapshot. Before sending, it checks whether the corresponding INTIN is still active. **This is the reason why interrupts, both edge and level triggered, should be kept active until CPU acknowledges it.** The difference between edge triggered and level triggered interrupt is that edge triggered interrupts ensure only one activation of interrupt per low to high edge whereas the level triggered interrupt allows to have multiple interrupts as long as the interrupt is held high. It should be noted that both edge and level triggered interrupts are active high.

MP SYSTEM

Initialization Sequence

This section assumes the system with multiple CPUs with each CPU having its own 82489DX local units and local interrupts (like local secondary cache data parity interrupt, coprocessor interrupt etc.,) connected to the respective local units. The system additionally assumes symmetric multiprocessing in the sense that I/O system is symmetric and it can be initialized by any CPU in the system.

Section A: Code Executed by all CPUs in the system

Section B: Synchronization to indicate Section A is completed

Section C: Only one CPU need to execute this Code

Each local unit is visible (through address mapping) only to that CPU to which local unit is attached. So each local unit will be programmed by its own CPU. Thus the code specified as section A will be executed by all CPUs in the system.

Section B of the initialization code is also executed by all the CPUs. This section of the code ensures that all the CPUs have completed execution of their "Section A" so that all Local units are properly initialized with different IDs, the system is in a consistent state, etc.,

Section C initializes system wide I/O unit and enables the interrupt mechanism to start functioning. Since the I/O unit is system wide, only one CPU need to program the I/O unit part of the 82489DX.

Section A

Write the local Unit ID (if needed)

Write all Ones to Destination Format Register

Write Logical Destination Register

Raise the Task Priority

Program the Spurious Interrupt Vector
Vector and Enable the Local Unit

Program the Vectors for Local Interrupts and Timer

Program the Timer Control Registers

Clear the mask for Local Interrupts and Timer

Initialize the local Interrupt sources

Broadcast ALL INCL. SELF
Reset DEASSERT message

Lower the Task Priority

It should be noted that the interrupt descriptors, interrupt service routine, spurious interrupt service routine and other interrupt related structure should have been initialized before the Section A. This is because Section A code enables respective local interrupts and timer interrupt vectors and when the interrupts arrive from these devices Section A ensures that 82489DX will provide the vector. But the code executed before Section A should ensure the interrupt structure is initialized. Spurious interrupts are bound to occur because of the asynchronous interaction between interrupts and software writing to task priority register. So spurious interrupt service routine has to be initialized before Section A.

WRITE THE LOCAL UNIT ID (IF NEEDED)

Each local unit can get the ID latched by reset from 82489DX address pins A3–A10. If the hardware ensures that during reset each local unit in the system gets different pattern on the address pins A3–A10 then all the local units are initialized automatically with different IDs. In that case writing to the local unit ID by software is not mandatory. If software writes the local unit ID then it should be read from some address space which is same for all CPUs but have different IDs for different CPUs. This will ensure that the same code when executed by different CPUs will initialize respective local unit with different ID.

WRITE ALL ONES TO DESTINATION FORMAT REGISTER

All the 32 bits of Destination Format Register are written with 1. This is to support single level logical addressing mode. This mode is explained in the following paragraph.

WRITE TO LOGICAL DESTINATION REGISTER

The logical Destination Register should be written with the logical destination address. It should be noted that since each CPU needs to assign a different logical Destination address to its own 82489DX local unit and since this code is executed by all CPUs the logical Destination address should be read from some address space which is the same for all CPUs but contains different Destination address values. Since logical Destination address is in bit decoding format, typically this can be achieved by shifting the CPUID.

The logical destination register with Destination format register can be used to support flat model. In this model, bits 24 through 31 of the destination address of the interrupt message vector are interpreted as decoded field. Intel strongly recommends for future compatibility to use only bits 31 to 24 of logical destination register. To have binary compatibility with future APIC implementations, any code written for 82489DX should not use bits 0 to 23 of logical destination register. This field is compared against the logical destination register of the local unit. If there is a bit match (i.e., if at least one of the corresponding pair of bits of the destination field and logical destination register match) this local unit is selected for interrupt delivery. Each bit position in the destination field corresponds to an individual local unit. For future compatibility, only bits 0 to 23 of logical destination register should be zero. This scheme allows the specification of arbitrary groups of 82489DXs simply by setting the member's bit to one, but allows a maximum of 8 local units in the system since bits 0 to 23 of the logical destination register is zero. Broadcast to all is achieved by setting all 8 bits of destination to ones. This selects all 82489DXs in the system.

If more than 8 units are to be addressed in the system (and if future compatibility is not a major issue) then all the bits of the logical distination register can be used as a bit map thereby increasing the number of CPUs addressable in logical addressing to 32.

RAISE THE TASK PRIORITY

Before enabling the local interrupts and timer interrupts the task priority is raised to maximum priority in the system so that these interrupts are masked temporarily.

PROGRAM THE SPURIOUS INTERRUPT VECTOR AND ENABLE THE LOCAL UNIT

The spurious interrupt vector register is programmed with the corresponding vector. This vector will be pointing to a dummy routine with just an IRET. The unit is enabled so that the tristate pin PINT can come out of tristate state to pass the interrupts.

PROGRAM THE VECTORS FOR LOCAL INTERRUPTS AND TIMER

The interrupt vectors for Local Interrupts and timer are initialized with corresponding vector.

PROGRAM THE TIMER CONTROL REGISTERS

The timer registers such as divider configuration register, initial count, mode of operation and source of the timer clock are programmed.

CLEAR THE INTERRUPT MASK FOR TIMER AND LOCAL INTERRUPT

The interrupt mask is cleared for timer and local interrupts by clearing the interrupt mask bit in their respective interrupt vector register.

INITIALIZE THE LOCAL INTERRUPT SOURCE

The local interrupt sources are also programmed for proper system operation. This involves enabling the interrupt from the sources. The order of enabling the interrupt is very important. First the 82489DX entries should be cleared and then the sources connected to the pins should be enabled. If done the other way, interrupts may get lost. This is true particularly in edge triggered interrupt inputs where if 82489DX mask is cleared after enabling the source interrupt, 82489DX may not have a chance to capture the low to high edge which might have produced immediately after the source interrupt is enabled and before 82489DX mask is cleared.

BROADCAST ALL INCL. SELF RESET DEASSERT MESSAGE

This is done so that all the local unit's ArbIDs are in sync. It should be noted that for breaking the tie during lowest priority arbitration ArbID is used. ArbID is copied from local unit ID during reset. Since local unit IDs can be written through software and at that time ArbID is not updated there may be a case where all ArbID in a system to have same value. To avoid such situation Reset Deassert message is sent to ALL INCL. SELF so that the ArbIDs are different in the system.

LOWER THE TASK PRIORITY

Task priority is lowered so that the interrupts can be armed to the CPU.

Section B

Synchronization

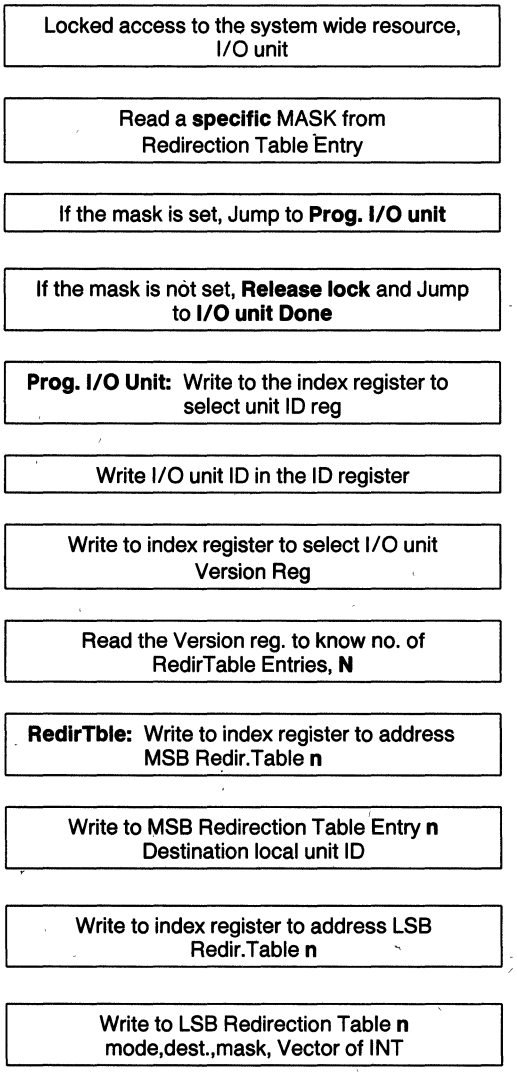
There are many methods available for synchronization. **Test - and - set** is a simple primitive, for example, available for synchronization. **Counting semaphores** can be built using this test - and - set primitive and synchronization can be achieved.

The main idea is to achieve global synchronization among the processors to indicate the local unit portion is programmed.

Section C

SYSTEM WIDE RESOURCES PROGRAMMING

This portion needs to be programmed by one CPU only. It should be noted that since the system environment we are assuming is shared memory symmetric MP system, CPU specific coding is not possible. System wide resource programming can be achieved by many ways depending on simplicity and performance (since this is only initialization routines, performance should not matter much) tradeoff. The following sequence illustrates a simple approach to program. The assumption here is that 82489DX will get reset both during cold reset and warm reset.

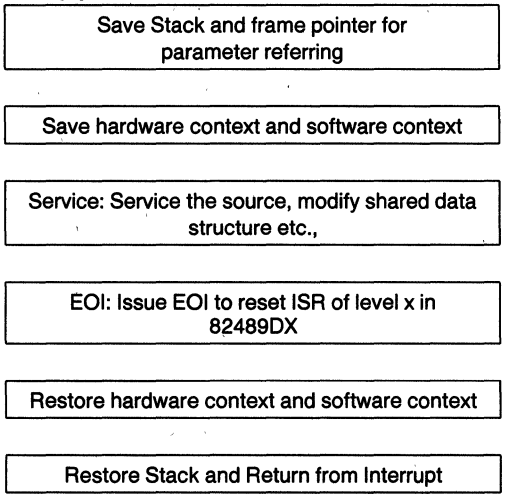


The first CPU getting the lock will find the mask to be set (since after reset, 82489DX mask is set). It locks the I/O unit for programming. The mask selected is specific in that the system initializes such that the mask is cleared during initialization. So by reading that mask mutual exclusion is achieved. If the system requirement is such that no mask can be cleared during system initialization some other register can be read. For example, the I/O unit IDs are reset to 0 on reset. Since the system initialization will have all the local unit IDs starting from 0 and I/O unit will be initialized by the system to non Zero ID, I/O unit can be read and if 0 can be assumed that programming is not yet done (so that it can gain control of lock and start programming) and if found non Zero, then that CPU can skip programming the I/O units by jumping to I/O unit done.

The I/O unit registers are organized as index register and data register. Other portions of section C are self explanatory. After I/O unit done, the I/O system initialization can be started so that the interrupts can start flowing in the system.

INTERRUPT SERVICE ROUTINE

ISR (x)



The above ISR x shows the interrupt service routine of interrupt level x. The stack, hardware context like CPU registers and software context like task specific variables are saved. The Servicing is done as specific to the interrupting source. This may involve reading a status register or initiating a thread to read a "full" buffer, initiating a thread to write data to some "empty" register, or acknowledging an interrupt from another CPU. This is the point at which the interrupting source is supposed to deactivate its request. Next EOI is issued to reset the ISR bit corresponding to the interrupt level x. Till now the interrupts from same level and lower levels were masked. Once EOI is written, interrupts from all the levels can start coming. The hardware context and software context are restored followed by stack cleaning and a proper Return from Interrupt is executed.

There are couple of timing issues that can be considered here. The time delay between **Service** and **EOI** is referred here. This timing and its relevance to edge/level triggered interrupt is discussed as follows: In the case of edge triggered interrupts, for each edge one Interrupt message is sent by I/O unit to local unit over I_{CC} bus whereas for level triggered interrupts there are two interrupt messages sent, one during assertion of level interrupt, and another during deassertion of level interrupt. In edge triggered interrupts, since the deassertion of interrupt does not result in any interrupt message, there are not many issues with the timing delay between **Service** and **EOI**, even though in general delaying EOI means interrupts from the same interrupt source are kept pending from interrupting CPU. In level triggered interrupts after **service** the I/O device starts deasserting its interrupt request. This results in an interrupt message to clear IRR bit in the local unit. This may take some time because the minimum possible time in I_{CC} bus is 2.3 μs (10 MHz I_{CC} clock assumed). If the I_{CC} bus is occupied by some other messages already then this IRR clearing message has to wait to get its turn which means additional delay. If EOI is issued before this happens then ISR gets cleared and IRR for this "done interrupt" is still alive to erroneously set ISR again. This will result in another interrupt. So "Early Servicing" is advisable in level triggered interrupts.

DOS Environment

In the DOS environment the initialization portion is the only routine to be coded since the 82489DX acts as a virtual wire once initialized and needs no more programming. Since it is uniprocessor environment there is no need for synchronization.

The interrupt from 8259 is programmed as type ExtINTA and other redirection table entries are not accessed since their masks are set by reset and hence disabled.

In the Interrupt service routine, since EOI is not needed for ExtINTA type of interrupts, no programming is needed for 82489DX. Since ExtINTA type of interrupts do not have any relationship to task priority, Spl routines do not apply for DOS configurations.

Transition from 8259 to 82489DX

Typically, platforms with 82489DX will support 82489DX in virtual wire mode. The BIOS in the EPROM will program the 82489DX in "Virtual Wire" mode. Typically systems boot DOS and then the 32 bit high performance OS is given control. There are also situations where after BIOS code is executed the high performance OS is given control. In both the situations, the 8259 will be operational during the DOS or BIOS portion of the code and interrupts will be flowing in the system. When the high performance OS is given control, it may want to disable the interrupt during initialization. This will involve disabling 8259. After disabling 8259, the 32 bit OS initializes and then it may want to enable interrupt mechanism which involves enabling 82489DX. The sequence we are encountering here is 8259 (and one input of 82489DX enabled in "ExtINTA" mode) enabled, 8259 disabled and then 82489DX enabled. When 82489DX is enabled in 32 bit OS all the interrupt inputs are enabled as opposed to the only one interrupt enabled in "Virtual Wire" mode. The additional difference is that 82489DX is no more a "virtual wire" but it is functioning as an interrupt controller.

In the above situation, consider the following scenario. The 82489DX is functioning as "virtual wire" and passing the 8259 interrupts as "ExtINTA" mode to the local unit. When interrupt mechanism is disabled by CLI (Clear interrupt) or masking the 8259 interrupt, there may be a possibility that already 8259 originated interrupt may be pending at the local unit asserting interrupt to the CPU. Now since the CPU has executed CLI, the interrupt is not serviced and the interrupt is kept pending. It should be noted that the pending interrupt is of type "ExtINTA". After this, 32 bit OS gets loaded which configures 82489DX redirection tables and interrupt is enabled. Now the "old pending" interrupt is delivered and since it is "ExtINTA" the external hardware will typically pass the interrupt acknowledge cycle to 8259. But at this point of time 8259 has been masked by 32 bit OS. Hence the "masked" 8259 responds with IR7 vector. So the 32 bit OS should reserve IR7 vector for both master and slave 8259 for "emptying" the old pending interrupt since the "Virtual Wire" remembers the previous interrupt.

Sequence of Enabling: In the case of enabling interrupt controllers in "ExtINTA" mode the 82489DX should be enabled before the 8259 interrupt Controller is enabled. This is because ExtINTA is "edge triggered" and if 8259 is enabled before 82489DX, 8259 might have

given an interrupt request by activating its interrupt output while 82489DX is still not enabled. When 82489DX is enabled the interrupt input has a high level and 82489DX had no chance of capturing the low to high edge of 8259.

Spurious Interrupt Service Routine

It is advisable not to share spurious interrupt vector with any genuine interrupt source. This section assumes that spurious interrupt vector is not shared with any other interrupt.

82489DX does not set ISR in response to spurious interrupt, NMI type of interrupt, Reset type of interrupt and ExtINTA type of interrupts. For all these interrupts EOI should not be issued.

Some systems have a variable count in the supervisor data structure to count number of spurious interrupts raised in the system. This can be used to study the reliability and "noise level" of the system. But in 82489DX architecture, spurious interrupt can occur even by a dynamic write to task priority register, frequency of spurious interrupt does not mean anything related to "noise level".

Return from interrupt

Spl(x) Routines

The processor handles the I/O system through device driver interface. The device driver consists of two entries to access the I/O system: 1) Call entry and 2) Interrupt entry. The interrupt entry is the one that we have been discussing for a while, i.e., interrupt service routine. The Call entry is the way the I/O system is accessed to initiate and service devices. The call entry has its own task priority and interrupt entry has the priority that is associated with the device interrupt level. The call entry and interrupt entry processes have I/O data structure like linked list, buffer pointers in common which they share. Mutual exclusion is needed to ensure the integrity of I/O system.

To ensure the mutual exclusion between these two processes running in the same processor, Spl(x) routine is used. The call entry routine (which is normally at a lower priority than the interrupt entry routine) calls Spl(x) routine to elevate its own priority above (or equal to) that of the corresponding device's interrupt priority. At this priority the interrupts from the device are masked out and the shared data structures can be accessed (exclusively).

Once this is done the priority is restored back to original value so that other interrupts won't suffer for relatively long time.

Spl() is used to save the current task priority and Spl(x) is used to elevate the task priority.

Spl()

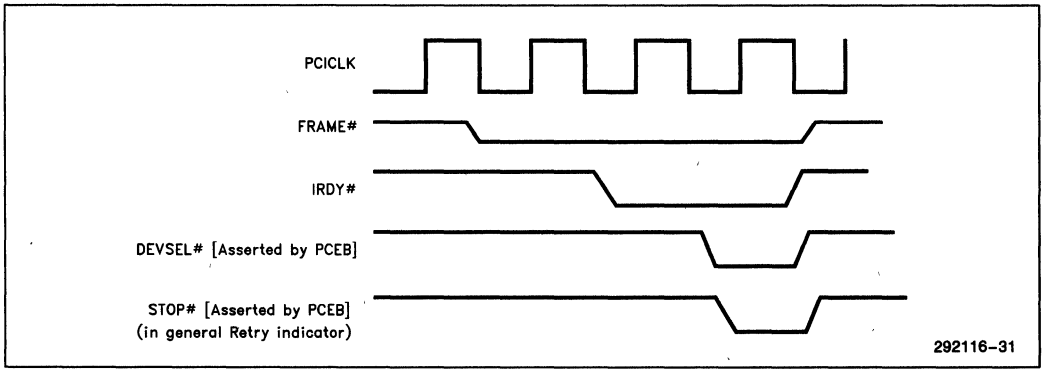
Read and return the 82489DX local unit task priority register

Spl(x)

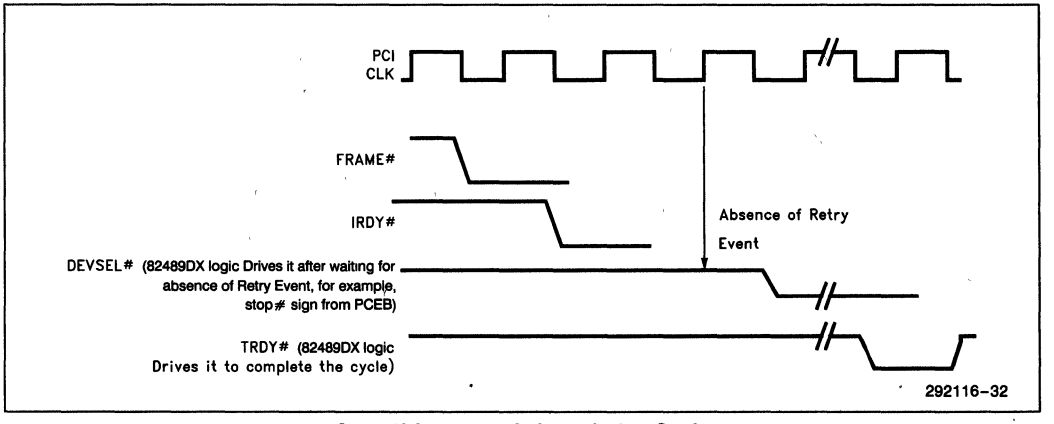
Write x to task priority register to raise priority to x

Typical usage of these routines

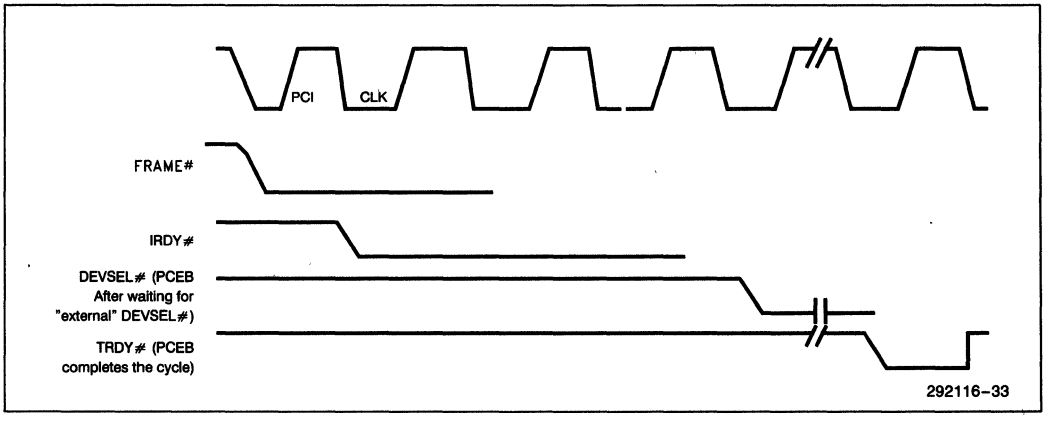
```
y = spl()    //Save the current task priority register value//
spl(x)       //Raise the task priority value                 //
:
:
;            // Access the shared data structure             //
spl(y)       // Restore the task priority register           //
```



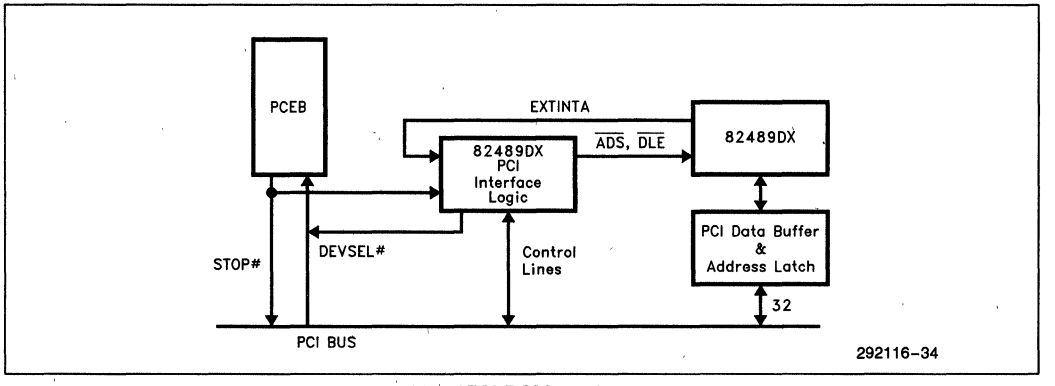
Case 1) Any INTA# Cycle Buffer Management and Retry



Case 2) Interrupt Acknowledge Cycle
Source of the Interrupt and Vector is 82489DX



Case 3) Interrupt Acknowledge Cycle
Source of the Interrupt and Vector is ESC 8259



82489DX-PCI Interface

82489DX AND PCI-EISA BRIDGE INTEROPERABILITY

82489DX gives performance benefits in multithreaded operating systems. Thus in both uniprocessor and multiprocessor systems 82489DX enhances the system level performance. This is because of its advantage in task priority management. Intel's PCIset EISA bridge component PCEB (B-stepping onwards) can interoperate with 82489DX. In this section, we will go over the "hook" provided by PCEB to connect 82489DX in a PCI system with some external glue logic.

From the Interrupt acknowledge timings of PCEB (B-stepping onwards) it can be inferred that **whenever the internal data buffers are empty**, on an interrupt acknowledge cycle, PCEB waits one clock cycle so that PCI interface logic for 82489DX can activate DEVSEL#. But, if the internal data buffers are not empty, then PCEB drives STOP# to retry the INTA cycle so that it can flush the buffers.

If DEVSEL# is seen active and if the PCEB's internal data buffers are empty, then PCEB allows 82489DX to own the INTA cycle. Thus, the external 82489DX glue logic, on an INTA cycle, should first sample STOP#. If STOP# is driven, then the glue logic should ignore the cycle. Because PCEB has some data in the buffers it wants to flush them before INTA cycle is run. So, the 82489DX glue logic should not start the cycle to 82489DX. If STOP# is not active and if the "ExtINTA" pin from 82489DX is inactive (to indicate that the cycle is for 82489DX) then it should drive DEVSEL# immediately to own the INTA cycle. At the same time it can start the cycle to 82489DX. It should be noted that 82489DX needs two INTA cycles whereas PCI bus has only one INTA cycle. So, the external logic is responsible for splitting one PCI INTA cycle into two 82489DX INTA cycles back to back but pass only one READY to the system.

During INTA cycle on PCI bus if "ExtINTA" pin is active (to indicate that it is 8259 INTA cycle), then the 82489DX glue logic should not drive DEVSEL#. Thus by finding DEVSEL# inactive, the PCEB will respond to the INTA cycle.

Thus with minimal external glue logic, it is possible to design an APIC based PCI system. Since PCI local bus will improve the I/O performance of the system, APIC will enhance the improvement of the overall system performance in a multithreaded environment.

HARDWARE DESIGN CONSIDERATIONS

Design Consideration 0

Any edge triggered interrupt creating an active edge while the interrupt is masked at 82489DX is lost. The 82489DX samples the edge triggered interrupt input only when it is unmasked. If an edge occurs while the interrupt is masked, that interrupt is lost. The software should always unmask the interrupt at 82489DX and then enable at the device. By this, it is made sure that 82489DX will have a chance to find the active going edge.

Design Consideration 1

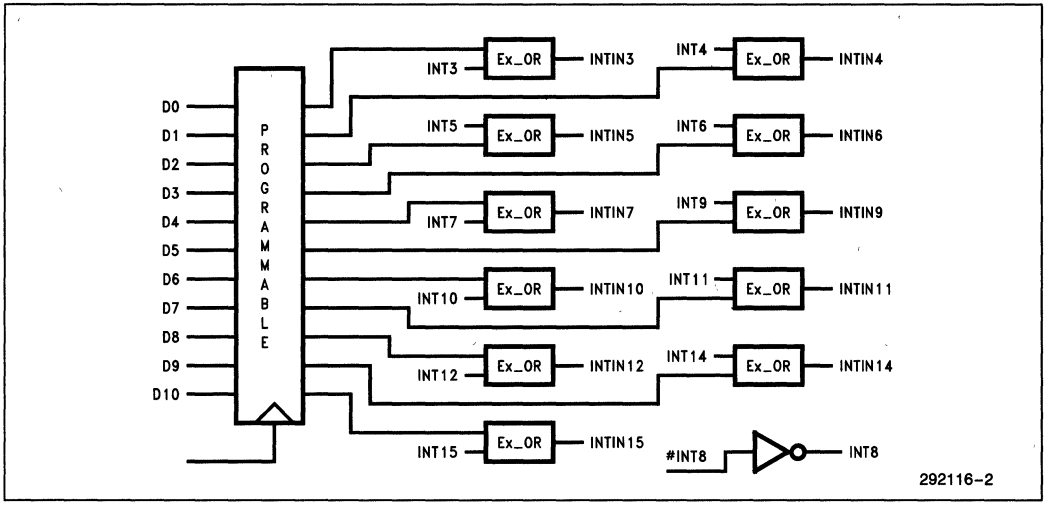
Description: The following design consideration has to be taken care of when using ISP (82357) as external interrupt controller. 82489DX allows connecting external 8259 type interrupt controller at one of its inputs. The mode associated with the interrupt input which has 8259 connected to it is called ExtINTA mode. 82489DX allows only EDGE TRIGGERED program-

ming option for ExtINTA mode. But in the case of 82357, the INT output from ISP stays high in case more than one interrupt is pending at its inputs. It does not always inactivate its INT output after INTA cycle. This will lead to a situation where ISP keeps the interrupt at high level continously and waits for INTA cycle. But since 82489DX expects an edge for interrupt sensing (for ExtINTA interrupts) it does not pass the interrupt to CPU and further interrupts are lost. So External circuitry should monitor the end of SECOND CYCLE of INTA cycle and force an inactive state at 82489DX's input. This can be done by ANDing ISP's output with a forced brief low going pulse at the end of second INTA cycle. This will generate an edge for each interrupt at 82489DX's input. For more refined edge generating logic, refer to data book, Order Number 290446.

Design Consideration 2

Description: The following design consideration has to be taken care of when using 82489DX in EISA systems. EISA ISP(82357) chip integrates 8259A. It additionally allows sharing of interrupts. To facilitate this sharing it has a programmable register, ELCR (Edge / Level trigger control register) by which certain interrupt inputs can be programmed as edge (low to high except for RTC) or level (the level is active low). The determination of edge or level is done during initial configuration of EISA system by reading EISA add in boards from the interrupt description data structures. The solution is to have programmable logic at the interrupt inputs so that 82489DX is compatible with EISA ISP. This will introduce one more register and logic to support this. This should be an 11 bit programmable register and an array of ExOR logic (12 ExOR gates or equivalent PLD). The ISP allows programmability of the following interrupts. It is highly recommended to use the same address of ELCR (and also bit definitions) for this polarity register, if possible, so that when ELCR is written this register will also be written. By this there is no separate programming needed for this polarity register. This will help to maintain compatibility with future APIC implementations which may use the existing ELCR register itself for polarity control. This is true for integrated APIC.

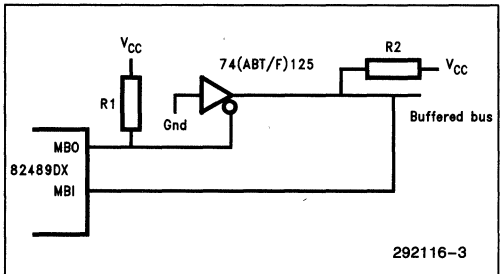
INT3 INT4 INT5 INT6 INT7 INT9 INT10 INT11 INT12 INT14 INT15. In addition to the above 11 interrupts, it fixes **INT8** to be active low edge triggered interrupt. INT8 is the only case where it is active low edge triggered type. So the following logic can be used to add programmability in 82489DX based EISA system. Before connecting these 11 interrupt lines directly (#INT8 which is from Real Time Clock is always active low edge triggered. #INT8 can be passed through an inverter since there is no need for programmability) to the 82489DX they should pass through an array of 11 Ex__OR gates. One input of Ex__OR gate connects to the corresponding INT pin and other input connects to a bit of programmable register. The output of Ex__OR gate is connected to 82489DX. The idea of Ex__OR is to use as a controlled inverter.



INTIN are the interrupt inputs to the 82489DX and INT are the system interrupt. The Ex_OR gating register is programmed after EISA configuration is found from add in boards as how these interrupt lines are going to be used in that particular configuration. If a particular input is edge triggered, then the corresponding bit in the register is written with 0. If a particular input is level triggered, then the corresponding bit in the register is written with 1.

Design Consideration 3

I_{CC} bus drive is an open drain bus with drive capacity of 4 mA only. Since data is transmitted at each I_{CC} clock, the "charging" of I_{CC} bus should be fast enough to ensure proper logic level at each clock edge. The I_{CC} bus needs pull up resistors since it is open drain bus. Since the drive is only 4 mA, the pull up resistor value can not be less than 5V/4mA. This being the limit of the resistor value, the length and the characteristics of the I_{CC} trace forces a capacitance value. Both the resistor and capacitance brings a RC time constant to the I_{CC} bus waveform. So, Electrical consideration has to be given to and practice of controlled impedence should be exercised for layout of the I_{CC} bus. The length of the trace should be kept as minimum as possible. If the length of the I_{CC} bus can't be kept less, than say 6 inch, because of mechanical design of the system, the external line drivers should be added to I_{CC} bus and I_{CC} bus should be simulated with the added driver characteristics.



NOTE:
R1 can be typically 1K.
R2 is designed from the simulation results.

Design Consideration 4

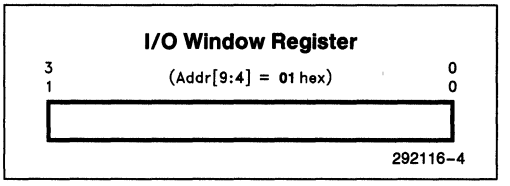
This is related to ADS#, BGT# and CS# timings. For bus cycles not intended for 82489DX, (CS# = 1 where 82489DX is supposed to sample it), any change in CS# line while the ADS# is still active, may erroneously cause a RDY# returned from 82489DX. Anomolous behavior may result if for BGT# ties low cases

a) BGT# goes away just one clock after ADS# or

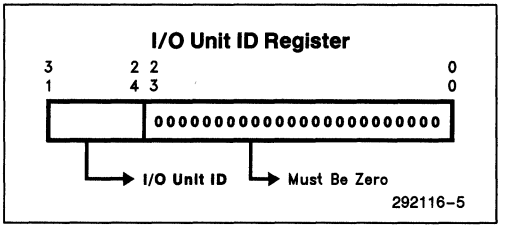
b) ADS# is still active, and CS# changes during this period.

For other cases anomolous behavior results if CS# changes when ADS# is still active. The following considerations are important from timing point of view. Always limit the pulse width of 82489DX ADS# to one CLKIN. Also avoid changing levels on BGT#/CS# line, when ADS# is active for cases being identified as BGT# tied low (BGT# sampled low when ADS# goes active). Also avoid changing levels on CS# line when BGT# is active.

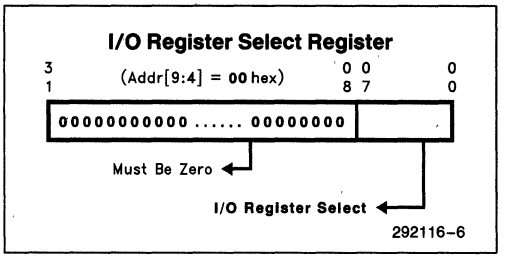
REGISTER PROGRAMMING DETAILS



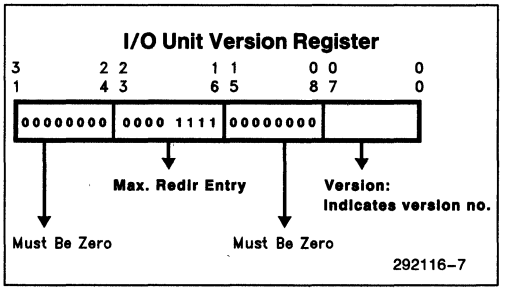
Data access to the register selected by I/O register select Register.



For example, for a Unit ID of 0A hex, the I/O unit ID register should be written with **0A00 0000.**



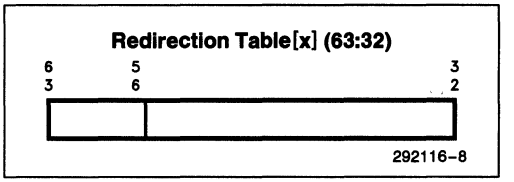
I/O Register Select	Register Selected
00 hex	I/O Unit ID Register
01 hex	I/O Unit Version Register
10 hex	Redirection Table[0] (31:0)
11 hex	Redirection Table[0] (63:32)
12 hex	Redirection Table[1] (31:0)
13 hex	Redirection Table[1] (63:32)
14 hex	Redirection Table[2] (31:0)
15 hex	Redirection Table[2] (63:32)
•	•
•	•
1E hex	Redirection Table[7] (31:0)
1F hex	Redirection Table[7] (63:32)
20 hex	Redirection Table[8] (31:0)
21 hex	Redirection Table[8] (63:32)
•	•
•	•
2E hex	Redirection Table[15] (31:0)
2F hex	Redirection Table[15] (63:32)



I/O Unit Version Register is read only register. It reads as **000F 00XX** where XX is version.

Max.Redir Entry: This is equal to the number of interrupt input pins minus 1 of this I/O unit. Read as 15 in 82489DX.

Version: The version number that identifies this version.



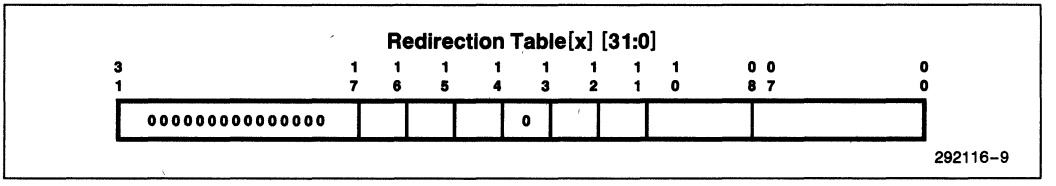
Destination: If the destination mode of this entry is "Physical Mode", then the 8 MSB (bits 56 through 63) contain an 82489DX local unit ID.

If logical mode, then all the 32 bits (bits 63 through 32) of the Destination field potentially defines a set of processors.

NOTE:
The same format holds good for Redirection Tables 0 to 15 (x = 0 to 15) for bits 63 to 32.

If the destination is to a local unit with ID, say, 05 in physical mode, then the redirection table [63:32] should be programmed as hex **0500 0000.**

Redirection Table [63:32] should be programmed first before programming Redirection Table [31:0].



Bits [31:17]: Reserved. Should be written 0.

Bit 16: MASK
0 — Not masked
1 — Masked

Bit [15]: TRIGGER MODE
0 — Edge Triggered
1 — Level Triggered

Bit 14: Remote IRR Status (Read only)
0 — Remote IRR is clear
1 — Remote IRR is set

Bit [13]: Reserved. Should be written 0.

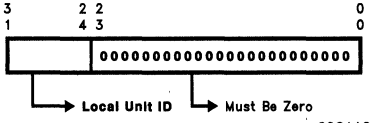
Bit 12: Delivery Status (Read only)
0 — Idle
1 — Send Pending

Bit [11]: Destination Mode
0 — Physical
1 — Logical

Bits [10:8] Delivery Mode
000: Fixed
001: Lowest Priority
100: NMI
101: Reset
111: ExtINTA

Bits [7:0] Vector
Vector for this interrupt

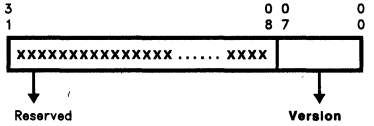
Local Unit ID Register: (Addr (9:4) = hex **02**)



292116-10

For example, for a local Unit ID of 0A hex, the local unit ID register should be written with **0A00 0000.**

Local Unit Version Register
(Addr (9:4) = hex **03**)

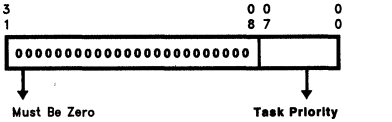


292116-11

Local Unit Version Register is **read only register.** It reads as **0000 00YY** where YY is version number.

Bits [7:0] Version: The version number that identifies this version.

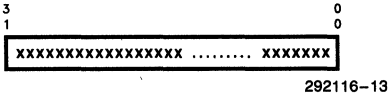
Task Priority Register (Addr (9:4) = hex **08**)



292116-12

Bits [0:7] Task Priority: Should be written with task priority.

End Of Interrupt (EOI) Register
(Addr (9:4) = hex **0B**)

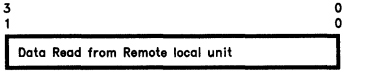


292116-13

Bits [31:0]: Data written to EOI is don't care.

Before returning from the interrupt handler, software must issue an End-Of-Interrupt (EOI) command to the 82489DX local unit. For NMI and ExtINTA and Spurious interrupts EOI SHOULD **NOT** be issued.

Remote Read Register: (Addr (9:4) = hex **0C**)

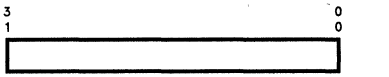


292116-14

The data read from remote local unit is latched in Remote Read Register. The software should qualify this data with "Remote Read Status bit" in the ICR register.

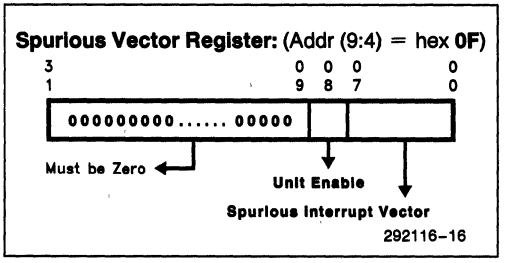
Interrupt Status Register [ISR]:

Register	Address[9:4]
ISR[31:0]	hex 10
ISR[63:32]	hex 11
ISR[95:64]	hex 12
ISR[127:96]	hex 13
ISR[159:128]	hex 14
ISR[191:160]	hex 15
ISR[223:192]	hex 16
ISR[255:224]	hex 17



292116-15

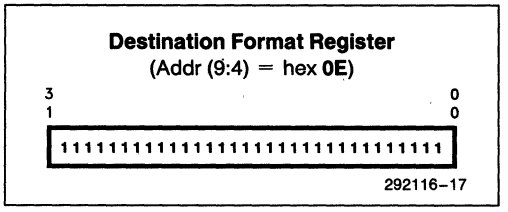
Interrupt Status Register is read only. It marks the interrupts that have been delivered to the processor and waiting for EOI.



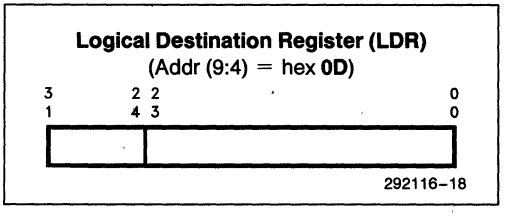
Bits [31:09]: Reserved bits. Must be zero.

Bit 8: Unit Enable: When this bit is 0, the local unit is disabled with regard to transmit and responding messages on I_{CC} bus. It only responds to messages with delivery mode set to "Reset". Reading a 0 at this bit indicates that the unit is disabled. When a 1 is written to the bit, the local unit is enabled for both transmitting and receiving messages. **Once enabled, it should not be disabled by software. Only further resets can take the unit into disabled condition.**

Bits [7:0] Spurious Interrupt Vector: For future compatibility, the bits [3:0] should be written with 1111. A spurious interrupt service routine should be existing in the address corresponding to the spurious interrupt vector.



The destination format register enables logical addressing by specifying the bit map in logical destination register. For future compatibility, all the 32 bits of Destination Format Register should be 1.



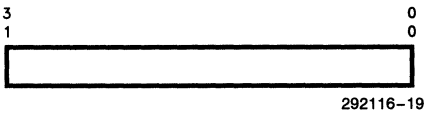
Each local unit can be addressed either physically using physical ID or logically using logical destination register. In physical addressing, either only one local unit can be addressed at a time or broadcast to all local units can be done. In logical addressing, a group of local units can be addressed through bit mapping in destination addressing and the logical destination register.

For future compatibility, bits [0:23] of the logical destination register and bits [0:23] of the destination address in the message should be zero. Bits 24 through 31 of destination information in the interrupt message received are interpreted as decoded field. This field is compared against the logical destination register of the local unit. If there is a bit match (i.e., at least one of the corresponding pair of bits of the destination field and LDR match) that unit is selected for interrupt delivery. Each bit position in the destination field corresponds to an individual Local unit. This scheme allows the specification of arbitrary groups of local units by setting the member's bits to 1, but allows a maximum of 8 local units in a system since only bits 24 through 31 (of the Logical Destination Register and logical destination address in interrupt message) are used. Broadcast to all is achieved by setting all 8 bits of destination to ones. This selects all local units in the system.

In a very large multiprocessor system where future compatibility is not a main problem, all the 32 bits of the Logical Destination Register and all the 32 bits of the destination address in the interrupt message can be used as a bit map to address the processors. When message addresses the destination using logical addressing scheme, the local unit compares the logical address in the interrupt message with its own logical Destination Register. Thus it is possible to support 32 processors in logical addressing mode.

Trigger Mode Register [TMR]:

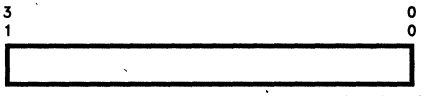
Register	Address[9:4]
TMR[31:0]	hex 18
TMR[63:32]	hex 19
TMR[95:64]	hex 1A
TMR[127:96]	hex 1B
TMR[159:128]	hex 1C
TMR[191:160]	hex 1D
TMR[223:192]	hex 1E
TMR[255:224]	hex 1F



If a bit corresponding to an interrpt vector number is 0, then it is assumed as edge triggered interrupt. For edge triggered interrupt, the corresponding IRR bit is automatically cleared when interrupt service starts. If 1 (level triggered) this is not the case. Instead, the source 82489DX (source I/O unit or Source Local Unit) must explicitly request the IRR bit be cleared (upon deassert of the interrupt input pin or upon sending an appropriate interprocessor interrupt). Upon acceptance of interrupt, the TMR bit is cleared for edge triggered interrupts and set for level triggered interrupts. This information was carried in the accepted interrupt message. The source 82489DX I/O unit also tracks the state of the destination unit's IRR bit (Remote IRR bit in the redirection table). When a level triggered interrupt input is deasserted, the source 82489DX I/O unit detects the discrepancy between the input pin state and the Remote IRR, and automatically sends a message telling destination 82489DX to clear IRR for the interrupt.

Interrupt Request Register [IRR]:

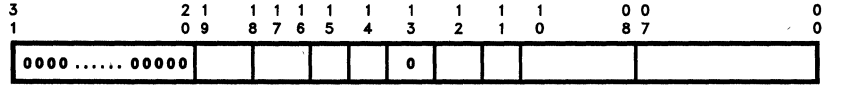
Register	Address [9:4]
IRR[31:0]	hex 20
IRR[63:32]	hex 21
IRR[95:64]	hex 22
IRR[127:96]	hex 23
IRR[159:128]	hex 24
IRR[191:160]	hex 25
IRR[223:192]	hex 26
IRR[255:224]	hex 27



292116-20

It contains the active interrupt requests that have been accepted, but not yet dispensed by this 82489DX local unit. A bit in IRR is set when 82489DX local unit accepts the interrupt. When TMR is 0, it is cleared when the interrupt is serviced; when TMR is 1, it is cleared when the 82489DX local unit receives a message to clear it.

Interrupt Command Register [31:0] (Addr [9:4] = **30** hex)



292116-21

Bits [31:20]: Reserved. Should be written 0.

Bits [19:18]: Destination Shorthand. This field indicates whether a shorthand notation is used to specify the destination of the interrupt and if so, which shorthand is used. Destination shorthands do not use the 32-bit Destination field, and can be sent by software with a single 32-bit write to the 82489DX's interrupt command register. Shorthands are defined for the following cases: Software self interrupt, interrupt to all processors in the system including the sender, interrupts to all processors in the system excluding the sender.

00: (dest field) means that no shorthand is used. The destination is specified in the 32-bit Destination field in the second word (bits 32 to 63) of the interrupt control register.

01: (self) means that the current local unit is the single destination of the interrupt. This is useful for software interrupts. The destination field in the interrupt command register is ignored. RESET assert Delivery mode should not be used with self destination. Only FIXED delivery mode should be used with SELF.

10: (all incl. self) means that the interrupt is to be sent to "all" processors in the system including the processor sending the interrupt. The 82489DX will broadcast a message with destination unit ID field set to all ones. RESET assert Delivery mode should not be used with "all incl. self" destination.

11: (all excl. self) means that the interrupt is to be sent to all processors in the system excluding the processor sending the interrupt. The 82489DX will broadcast a message with destination unit ID field set to all ones.

Bits [17:16]: Remote Read Status. This field indicates the status of the data contained in the Remote Read register. This field is read only to software. Whenever software writes to the interrupt command register using Delivery mode "Remote Read" the Remote Read Status becomes "in progress" (waiting for the remote data to arrive). The remote 82489DX local unit is expected to respond in a fixed amount of time. If the remote 82489DX local unit is unable to do so, then the remote read status becomes "invalid". If successful, the Remote Read status resolves to "Valid". Software should poll this field to determine completion and success of the Remote Read command.

00: (invalid): The content of the Remote Read register is invalid. This is the case when after a Remote Read command is issued and the remote 82489DX Local unit was unable to deliver the Register content in time.

01: (in progress): a remote read command has been issued and this 82489DX is waiting for the data to arrive from remote 82489DX local unit

10: (valid): the most recent Remote Read command has completed and the remote read register content is valid.

11: reserved.

Bit [15]: TRIGGER MODE

0 — Edge Triggered

1 — Level Triggered

Software should use this bit in conjunction with Level Assert/Deassert to generate interrupts that behave as edges or levels. **For future compatibility, send ICR messages only in edge triggered mode.**

Bit [14]: LEVEL. Software should use this bit in conjunction with the Trigger mode bit when issuing an inter-processor interrupt to simulate assertion/deassertion of level sensitive interrupts.

To assert: Trigger mode = 1 and Level = 1.

To deassert: Trigger mode = 1 and Level = 0.

For example, a message with Delivery mode of "Reset", a trigger mode of "Level", and Level bit of 0 deasserts reset to the processor of the addressed 82489DX Local unit(s). As a side effect, this will also cause all 82489DX to reset their Arbitration ID to their unit ID. (The Arb ID is used for tie breaking in lowest priority arbitration.) **For future compatibility, only edge triggering should be used in ICR.**

Bit [13]: Reserved. Should be written 0.

Bit [12]: Delivery Status (Read only)

0 — Idle

1 — Send pending

Delivery status is software read-only. Software can read to find out if the current interrupt has been sent, and the Interrupt command register is available to send the next interrupt. If the interrupt command register is overwritten before the Delivery status is "idle", then the destiny of that interrupt is undefined; the interrupt may have been lost.

Bit [11]: Destination Mode

0 — Physical

1 — Logical

In physical mode, a destination 82489DX is identified by its Local Unit ID. Bits 56 through 63 (8 MSB of the destination field) specify the 8-bit 82489DX Local unit ID.

In logical mode, destinations are identified by matching on Logical Destination under the control of the Destination Format Register in each Local 82489DX. The 32-bit Destination field is the logical destination. **For future compatibility, use only bits [31:24] of the logical destination address.** Bits [23:0] should be zero.

Bits [10:8]: Delivery Mode

000: (Fixed) means deliver the signal on the INT pin of all processors listed in the destination. Trigger mode for "fixed" Delivery Mode can be edge or level.

001: (Lowest Priority) means deliver the signal on the INT pin of the processor that is executing at the lowest priority among all the processors listed in the specified destination; Trigger mode for "lowest priority" Delivery mode can be edge or level.

011: (Remote Read) is a request to a remote 82489DX local unit to send the value of one of its registers over the I_{CC} bus. The register is selected by providing its address in the vector field. The register value is latched by the requesting 82489DX and stored in the Remote Register where it can be read by the local processor. A Delivery Mode of "Remote Read" requires an "Edge" Triggered mode.

100: (NMI) means deliver the signal on the NMI pin of all processors listed in the destination. Vector information is ignored. A delivery mode equal to "NMI" requires a "LEVEL" Trigger mode.

101: (Reset) means deliver the signal to all processors listed in the destination by asserting/deasserting the 82489DX local unit's PRST output pin. All addressed 82489DX local units will assume their reset state but preserve their ID. One side effect of a message with Delivery mode equal to "Reset" that results in a deassert of reset is that all Local Units (whether listed in the destination or not) will reset their lowest-priority tie breaker arbitration ID to their Local unit ID. A delivery mode of "Reset" requires a "level" Trigger mode. "Reset" should not be used with "Self" or "all incl.Self" Shorthand mode since it will leave the system in non-recoverable reset state. If "RESET" is used with "all exc.Self" mode, software should make sure that only one CPU executes this instruction in an MP system.

Delivery mode options **010,110,111** are Intel reserved. They should not be used.

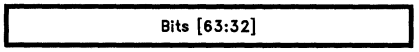
Bits [7:0] Vector. The vector identifies the interrupt being sent. If the Delivery mode is "Remote Read", then the Vector field contains the address of the register to be read in the remote 82489DX's Local unit.

NOTE:
In cases where Destination field in Interrupt Command Register [63:32] is used, Interrupt Command Register [31:0] should be programmed only **AFTER** programming Interrupt Command Register [63:32], since writing to [31:0] will start sending the message.

The following are the control words for interrupt command register [31:0] for different modes. The interrupt vector, for example, is illustrated with **AA hex**. In the remote Read request command **RR in the vector field** specify address of the register to be read. The **XX** in the vector field means the vector is don't care.

CONTROL WORD	PHYSICAL Destination Mode	LOGICAL Destination Mode
Fixed INT, Edge triggered int, *dest. field specified*	0000 00AA hex	0000 08AA hex
Lowest priority INT, Edge trigg. int, *dest. field specified*	0000 01AA hex	0000 09AA hex
Remote Read (only Edge Triggered), *dest. field specified*	0000 03RR hex	NOT SUPPORTED
NMI (Only Level) Level ASSERT, *dest. field specified*	0000 C4XX hex	0000 CCXX hex
NMI (Only Level) Level DEASSERT, *dest. field specified*	0000 84XX hex	0000 8CXX hex
Reset (Only Level) Level ASSERT, *dest. field specified*	0000 C5XX hex	0000 CDXX hex
Reset (Only Level) Level DEASSERT, *dest. field specified*	0000 85XX hex	0000 8DXX hex
Fixed INT, Edge triggered int, *Self*	0004 00AA hex	0004 08AA hex
Fixed INT, Edge trigg. int, *All inclusive Self*	0008 00AA hex	0008 08AA hex
Lowest priority INT, Edge trigg. int, *All inclusive Self*	0008 01AA hex	0008 09AA hex
NMI, Level ASSERT, *All inclusive Self*	0008 C4XX hex	0008 CCXX hex
NMI, Level DEASSERT, *All inclusive Self*	0008 84XX hex	0008 8CXX hex
Reset, Level DEASSERT, *All inclusive Self*	0008 85XX hex	0008 8DXX hex
Fixed INT, Edge trigg. int, *All exclusive self*	000C 00AA hex	000C 08AA hex
Lowest priority INT, Edge trigg. int, *All exclusive self*	000C 01AA hex	000C 09AA hex
NMI, Level ASSERT, *All exclusive Self*	000C C4XX hex	000C CCXX hex
NMI, Level DEASSERT, *All exclusive Self*	000C 84XX hex	000C 8CXX hex
Reset, Level ASSERT, *All exclusive Self*	000C C5XX hex	000C CDXX hex
Reset, Level DEASSERT, *All exclusive Self*	000C 85XX hex	000C 8DXX hex

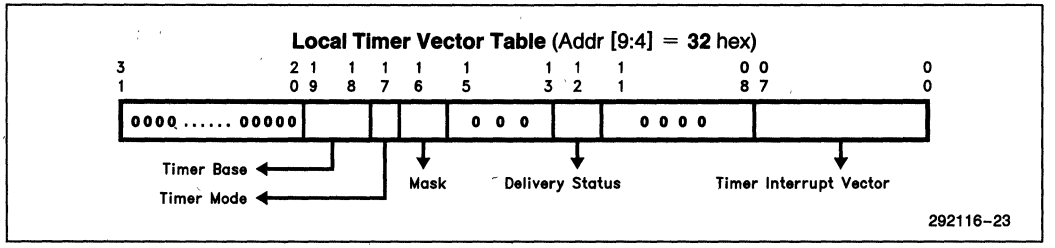
Interrupt Command Register [63:32]
(Addr [9:4] = **31** hex)



292116-22

Bits [63:32] Destination

This field is only used when the Destination Shorthand field is set to "Destination Field". If Destination field is physical mode, **then the 8 MSB contain an Destination Unit ID.** If logical mode, the full 32-bit Destination field contains the logical address. This register should be programmed for proper destination before programming Interrupt Command Register [31:0]. If the destination to a local unit with ID, say, 05 in physical mode, then the Interrupt Command Register [63:32] should be programmed as hex **0500 0000.**



Bits [31:20] Reserved. Should be written Zero.

Bits [19:18] Timer Base: This field selects the time base input to be used by timer.

00: (Base 0): Uses "CLKIN" as input.

01: (Base 1): Uses "TMBASE".

10: (Base 2): Uses the output of the divider (Base 2).

Bit 17: Timer Mode: This field indicates the operation mode of timer.

0 — ONE- SHOT;

1 — PERIODIC

In *ONE-SHOT*, the current count register remains at Zero after the timer reaches zero and software needs to reassign the timer's initial count register to rearm the timer.

In *PERIODIC* mode, when the timer reaches zero, the Current Count Register is automatically reloaded with the value in the initial Count Register, and the timer counts down again.

Bit 16: Mask: This bit serves to mask timer interrupt generation.

0 — Not masked;

1 — Masked.

Bits [15:13] Reserved. Should be written Zero.

Bit [12] Delivery Status: Delivery status indicates the current status of the delivery status of this interrupt.

0 — IDLE means that there is currently no activity for this interrupt.

1 — SEND PENDING indicates that the interrupt has been injected, but its delivery is temporarily held up by other recently injected interrupts that are in the process of being delivered; Delivery status is software read only.

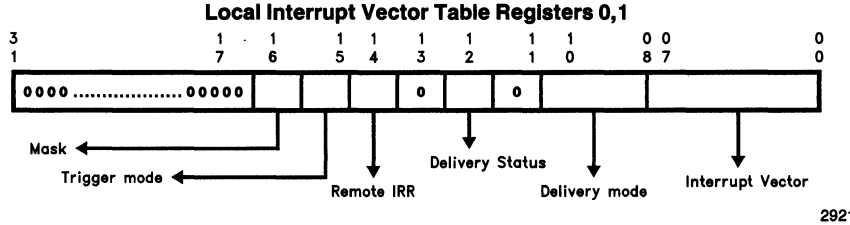
Bits [11:8] Reserved. Should be written Zero.

Bits [7:0] Timer Interrupt vector: This is the 8-bit interrupt vector to be used when timer generates an interrupt.

NOTE:
TIMER interrupts are always treated as EDGE triggered interrupts.

The following is the control word for various modes to be used in Local Timer Vector Table. For illustration purpose, the interrupt vector for Timer is shown as AA hex.

Control Word	CLKIN Input (Base 0)	TMBASE Input (Base 1)	Divider Input (Base 2)
PERIODIC timer, MASK cleared	0002 00AA hex	0006 00AA hex	000A 00AA hex
PERIODIC timer, MASK set	0003 00AA hex	0007 00AA hex	000B 00AA hex
ONE SHOT timer, MASK cleared	0000 00AA hex	0004 00AA hex	0008 00AA hex
ONE SHOT timer, MASK set	0001 00AA hex	0005 00AA hex	0009 00AA hex



Register	Address [9:4]
Local Int0 Vector table register	**35** hex
Local Int1 Vector table register	**36** hex

The same format applies to both Local Int0 and Local Int1 registers.

Bits [31:17]: Reserved: Must be Zero.

Bit 16: MASK:

0 — enables interrupt by clearing mask

1 — masks the interrupt.

Bit 15: Trigger mode:

0 — Edge Triggered

1 — Level Triggered

Bit 14: Remote IRR: This bit is used for level triggered local interrupts. Its meaning is undefined for edge triggered interrupts. Remote IRR mirrors the interrupt's IRR bit of this local unit. Remote IRR is software read only.

Bit 13: Reserved. Must be Zero.

Bit 12: Delivery Status: Software read only. Indicates the current status of the delivery of this interrupt.

0 — **IDLE** means that there is currently no activity for this interrupt.

1 — **Send Pending** indicates that the interrupt has been injected, but its delivery is temporarily held up by the recently injected interrupts that are in the process of being delivered.

Bit 11: Reserved. Must be Zero.

Bits [10:8]: Delivery mode

000 — Fixed INT

100 — NMI

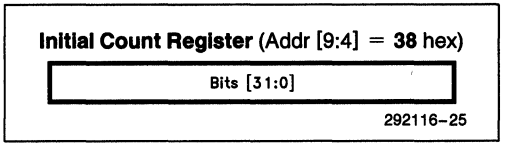
111 — ExtINTA

All other options of Bits [10:8] are reserved. Should not be used.

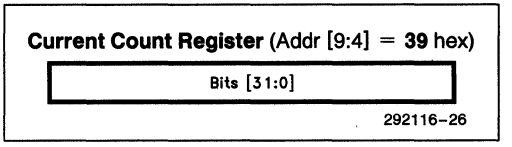
Bits [7:0]: Vector: This is the interrupt vector to use when generating interrupt for this entry.

The following are the control words for local interrupt [0 as well as 1] vector tables for different modes. The interrupt vector, for example, is illustrated with **AA hex**. The XX in the vector field means the vector is don't care.

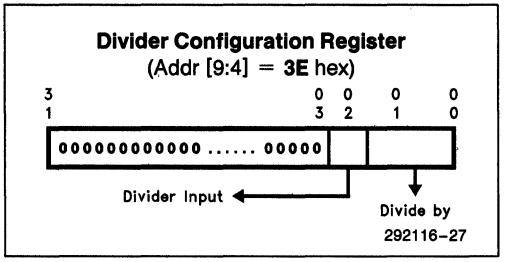
Interrupt Option	Control Word
Fixed INT, Edge triggered	0000 00AA hex
Fixed INT, Level trigg. int	0000 80AA hex
NMI (Only Level)	0000 84XX hex
ExtINTA (Only Edge)	0000 07XX hex



Bits [31:0] Initial Count: Software writes to this register to set the initial count for timer. This register can be written at any time. When written, the value is copied to the current count Register and countdown starts or continues from there. The initial count register is read-write by software.



Bits [31:0] Current Count: This is the current count of timer. It is read only by software and can be read any time.



Configuration	Control Word
Divide CLKIN by 2	0000 0000 hex
Divide CLKIN by 4	0000 0001 hex
Divide CLKIN by 8	0000 0002 hex
Divide CLKIN by 16	0000 0003 hex
Divide TMBASE by 2	0000 0004 hex
Divide TMBASE by 4	0000 0005 hex
Divide TMBASE by 8	0000 0006 hex
Divide TMBASE by 16	0000 0007 hex

Bits [31:3] Reserved. Must be Zero.

Bit [2]: Divider Input: Selects whether divider's input connects to the 82489DX local unit's CLKIN pin or TMBASE.

0 — means the divider takes its input signal from CLKIN.

1 — means use TMBASE

Bits [1:0]: Divide by: Selects by how much the divider divides.

00 — divide by 2

01 — divide by 4

10 — divide by 8

11 — divide by 16

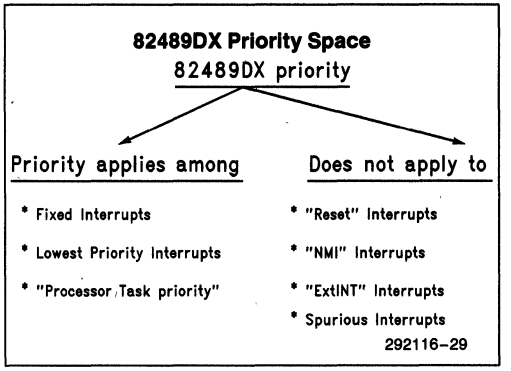
Programming Guidelines

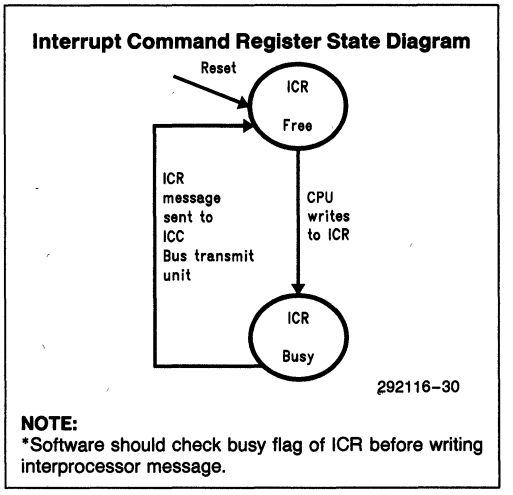
A) Modes of Interrupt in 82489DX:

Trigger Mode	Delivery Mode				
	Fixed Destination	Lowest Priority Delivery	NMI	Reset	ExtINT
Edge	✓	✓			✓
Level	✓	✓	✓	✓	

NOTE:

- RESET delivery mode should not be used for Local Interrupts.
- EOI should not be issued for NMI and ExtINT delivery mode.





CONCLUSION

82489DX has simple and powerful programming model. It has programmable priority and it supports task priority in the light of interrupt priority. It reduces the SPL() overhead which is very useful in uniprocessor system. The system performance is improved by using interrupt priority model to prioritize interrupts and by using task priority register for SPL() calls. It provides an easy migration path from 8259 by providing ExtINTA mode for DOS compatibility.

intel®

AP-479

APPLICATION NOTE

Pentium™ Processor Clock Design

2

DERRICK LIN

JIM REILLY

October 1993

PRELIMINARY

Order Number: 241574-001

Pentium™ Processor Clock Design

CONTENTS

CONTENTS

PAGE

FIGURES

Pentium™ Processor Clock Design

CONTENTS

PAGE

FIGURES

CONTENTS

PAGE

TABLES

1.0 INTRODUCTION

Today's high speed microprocessors place a heavy demand on clock generation and distribution. To maintain a synchronous system, well-controlled and precise clocking solutions are required. Pentium™ processor, with operating frequencies of 60 MHz and 66 MHz, has tight system clock specifications. In order to bring clock signals of acceptable quality and minimal skew to the Pentium processor and the rest of the system, system designers have to contend with high speed issues for clock distribution and limited number of precise clock driver devices. In this application note, the key issues in the design of a 60 MHz or 66 MHz clock for a Pentium processor-based system will be discussed, available clock drivers will be listed and discussed, and detailed design examples of a clock solution for the Pentium processor with 256K second-level cache subsystem, using the 82496 Cache Controller and the 82491 Cache SRAMs, are provided.

The Pentium processor, 82496 Cache Controller, and 82491 Cache SRAM form a CPU-Cache core or chip set. Along with a memory bus controller (MBC), the chip set provides a CPU-like interface for many types of memory buses.

This application note is intended for system designers concerned with clock generation and distribution for the Pentium processor and CPU-Cache chip set based systems. It reflects data collected from several quarters of characterization of the Pentium processor and experience with some of the clock driver devices, as well. This application note gives readers a good understanding of the issues and solutions of high speed clocking, particularly that for the Pentium processor. The reader should be familiar with the Pentium processor and CPU-Cache chip set electrical and mechanical specifications, *Clock Design in 50 MHz Intel486™Systems*, and transmission line theory. If not, please read materials listed in Section 9.0 before proceeding.

1.1 General Clocking Issues

There are two major problems with distributing clock signals at 66 MHz: clock signal quality and clock skew. At high speed, one set of effects which has been minor in slower designs is now significant—the effects of transmission line. At high frequencies and fast edge rates, long traces behave like transmission lines. The "lumped" circuit assumption which assumes instantaneous signal transmission is no longer valid. Instead, signals travel in a finite time. When a transmission line is not properly terminated, one can observe severe overshoot, undershoot and ringback, all of which degrade logical signals. Bad signal quality can cause false switching or multiple switching, and can in extreme cases damage the devices. To maintain a clean clock signal, designers must consider clock driver characteristics, signal routing, load characteristics, and transmission line termination.

There are four basic ways to terminate a transmission line, series, parallel, Thevenin, and AC terminations (Figure 1). Series termination is recommended when driver output impedance is less than the transmission line characteristic impedance (true for most TTL drivers) and the line is driving a small number of devices. Series termination consumes low power and uses only one device; however, the termination method increases signal rise and fall times. Series termination ensures good signal quality by eliminating secondary reflection off the driver end. The rest of the termination methods eliminate reflection at the load end. All of the termination methods can provide good, clean clock signals at the load. Both parallel and Thevenin terminations consume a large amount of power. Thevenin termination consumes less power than parallel but requires one more device. AC termination consumes low power but adds capacitive load to the driver and delay due to RC time constant. Design examples provided with this application note use series termination. For more information on transmission line effects and design issues, please refer to [ref. 3, ref. 4, ref. 5]

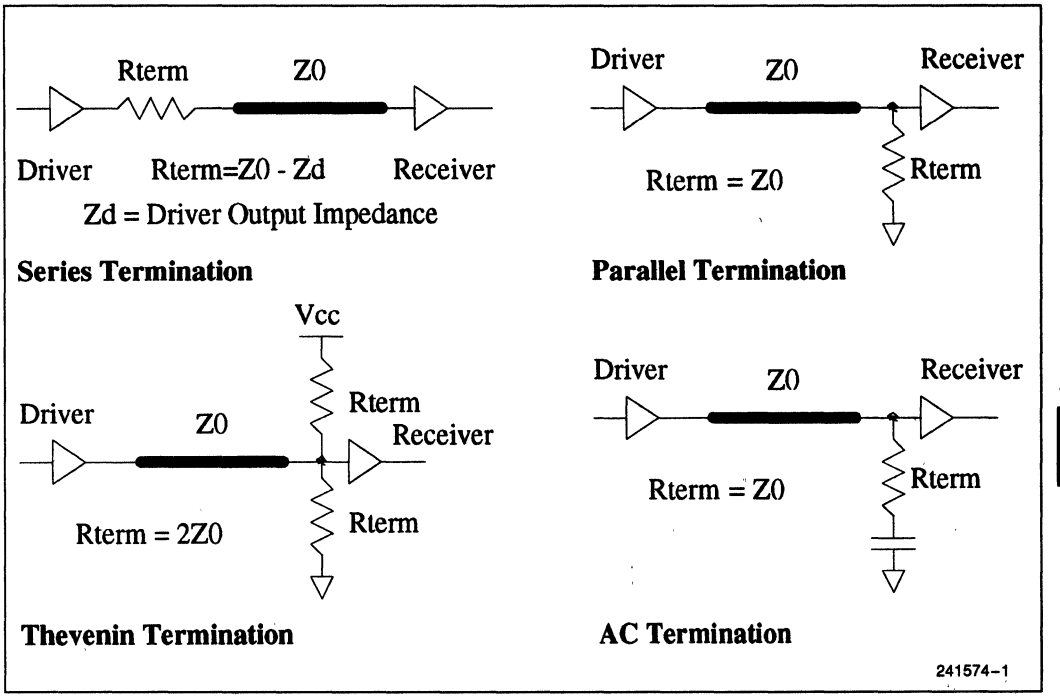


Figure 1. Common Termination Techniques

Skew is defined as the time difference between when the clock signal reaches each component. As frequency increases, there is less and less time for computation in a given clock period for a synchronous design. For a typical design, the time from one rising edge to the next is composed of the largest path-delay, setup time, propagational delay through logic elements, and skew. Clock skew then, takes away from the time available for propagational delay, thereby restricting the amount of logic done in a clock cycle. For high speed designs, skew must be minimized.

To minimize skew, designers must tune clock traces so that the propagational delay from driver through each trace to load is the same for each load. For balanced loads, tuned traces have same lengths. For unbalanced loads, trace lengths can be adjusted to make up for loading differences. If possible, designers should try to keep the loading on each clock line the same.

2.0 Pentium™ PROCESSOR, 82496 AND 82491 SYSTEM CLOCK SPECIFICATIONS

System clock specifications can be divided into 2 categories: signal quality requirements and skew specifications. Clock signal quality requirements are the same for the Pentium processor and CPU-Cache chip set. Skew specifications are only required for CPU-Cache chip set.

Signal quality requirements define boundaries for acceptable signal shapes and levels. There are two parts to signal quality requirements: signal quality specifications (Table 1) and guidelines (Table 2). Please refer to the latest revision of the Pentium processor and CPU-Cache chip set specifications for more details and for the most up-to-date information.

Table 1. Clock Signal Quality Specifications

Symbol (5)	Parameter	Minimum	Maximum	Unit	Notes
	CLK Frequency	33.33	66.66	MHz	(1)
t2	CLK Period	15		ns	
t3	CLK High Time	4		ns	(2)
t4	CLK Low Time	4		ns	(3)
t5	CLK Rise Time	0.15	1.5	ns	(4)
t6	CLK Fall Time	0.15	1.5	ns	(4)
	CLK Stability		±250	ps	(6), (7), (8), (9)
	V_{IH}	2	V_{CC} + 0.3	V	
	V_{IL}	−0.3	0.8	V	

NOTES:
1. Below 66 MHz only functionality is guaranteed.
2. High times are measured between 2.0V crossing points.
3. Low times are measured between 0.8V crossing points.
4. Rise and fall times are measured between 0.8V and 2.0V.
5. Symbols in Figure 2.
6. Functionality is guaranteed by design/characterization.
7. Measured on rising edge of adjacent CLKs at 1.5V.
8. To ensure a 1:1 relationship between the amplitude of the input jitter and the internal and external clocks, the jitter frequency spectrum should not have any power spectrum peaking between 500 KHz and 1/3 of the CLK operating frequency.
9. The amount of jitter present must be accounted for as a component of CLK skew between devices.

Table 2. Clock Signal Quality Guidelines

Parameter	Maximum	Unit	Notes
Overshoot	1.6	V	(1)
Undershoot	1.6	V	(1)
Ringback	0.8	V	(2)

NOTES:
1. Overshoot (undershoot) is the absolute value of the maximum voltage above V_{CC} (or below V_{SS}). The guideline assumes the absence of diodes on the input.
2. Ringback is the absolute value of the maximum voltage at the receiving pin below V_{CC} (or above V_{SS}) relative to V_{CC} (or V_{SS}) level after the signal has reached its maximum voltage level. The input diodes are assumed present.

The overshoot guideline should be used in simulations, without diodes present, to ensure overshoot (undershoot) is within the acceptable range. The ringback guideline is provided for verification in an actual system. System designers do not have to worry about ringback if the signal does not overshoot or undershoot, respectively. Figure 2 summarizes clock waveform requirements listed in Table 1.

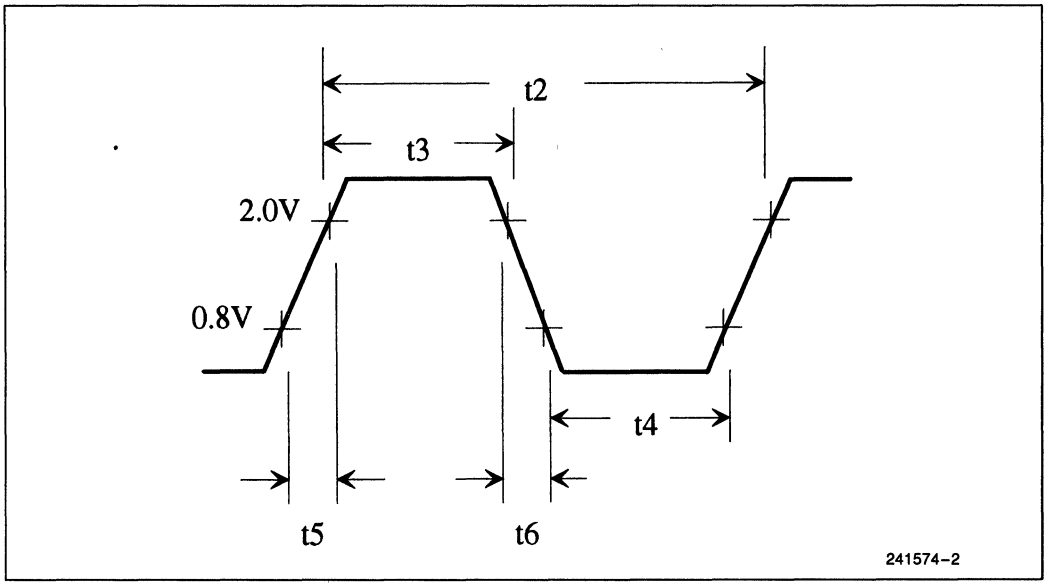


Figure 2. Clock Requirements for the Pentium™ Processor and CPU-Cache Chip Set

Figure 3 to Figure 5 illustrates examples of acceptable and unacceptable clock waveforms. Waveform in Figure 3 is for an input model without diodes. Waveform in Figure 4 is for an input model with diodes. The diodes clamp the voltage and prevent it from going more than a diode drop above V_{CC} or below V_{SS}. Waveform in Figure 5 is for an input model without diodes. The waveform is not acceptable for several reasons. It violates the minimum low time specification (4 ns), the maximum fall time specification (1.5 ns), and it does not follow the maximum undershoot guideline (1.6V).

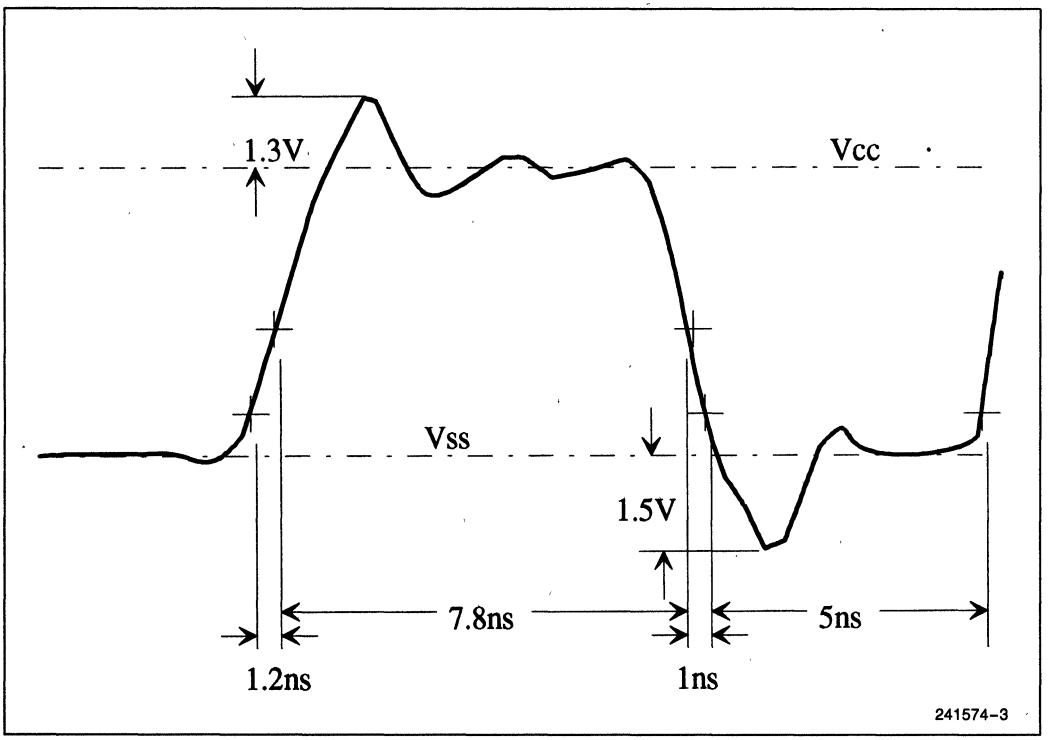


Figure 3. An Example of an Acceptable Clock Waveform (Diodes are Absent from the Input Model)

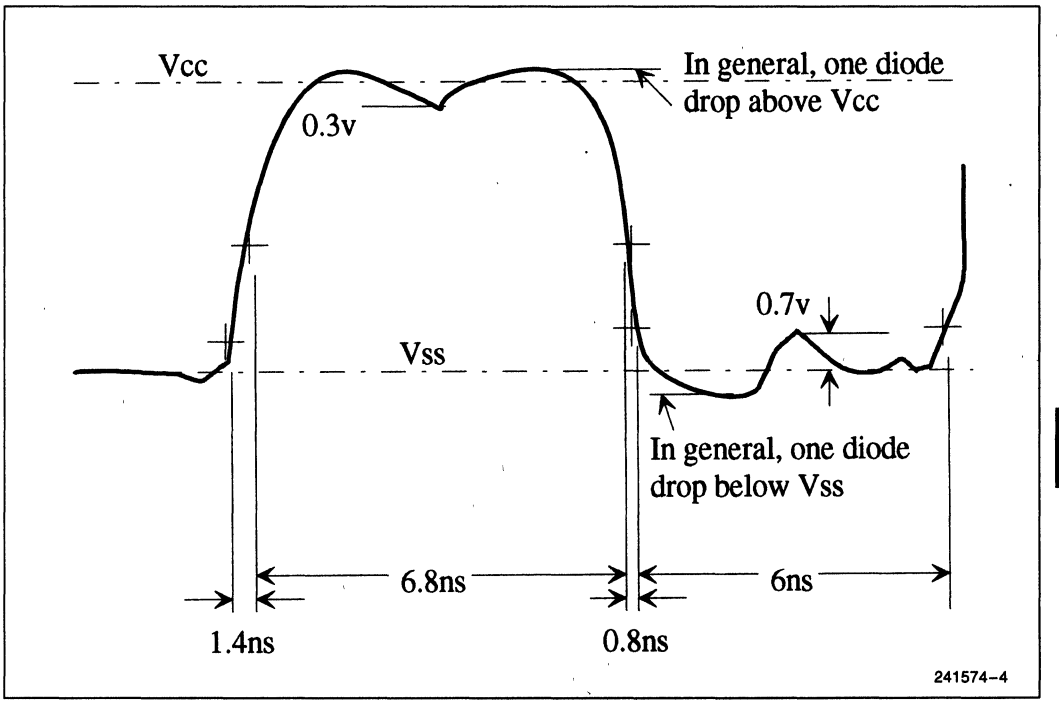


Figure 4. An Example of an Acceptable Clock Waveform (Diodes are Present in the Input Model)

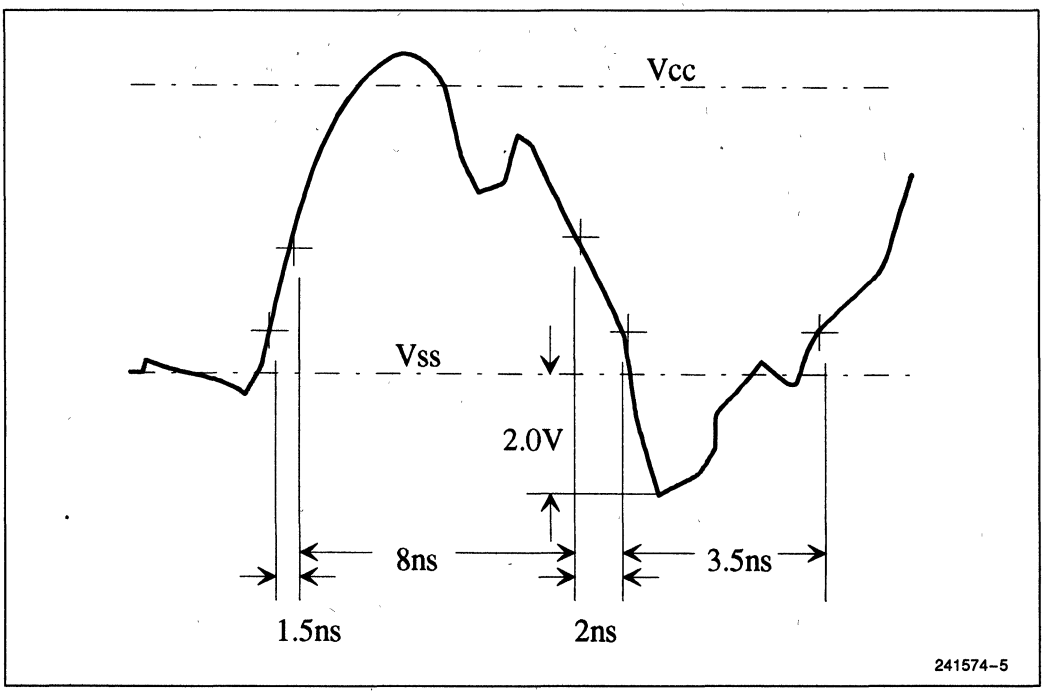


Figure 5. An Example of an Unacceptable Clock Waveform (Diodes Are Absent from the Input Model)

Clock skews for the CPU-Cache chip set are measured at 0.8V, 1.5V, and 2.0V on the rising edge. Worst case skew between the Pentium processor and the 82496 is 0.2 ns, and worst case skew between any 82491 and either the Pentium processor or the 82496 is 0.7 ns.

3.0 AVAILABLE CLOCK DRIVERS

Intel has held discussions with many clock driver component companies. The intent has been to enable these companies to offer clock driver solutions that meet the Pentium processor specifications. It has also been to ensure that the super set of these companies can provide support and distribution worldwide on a schedule that closely matches the Pentium processor's availability. Based on information available, Table 3 lists a number of companies who are planning to offer solutions to meet these requirements. All the clock drivers listed in Table 3 have maximum output frequency equal or above 66 MHz. Preliminary data sheets show that solutions listed in Table 3 meet the CPU or CPU-Cache chip set requirements. The specifications listed are based on preliminary data provided by each company and may be subject to change. Designers should contact each company for the latest specifications and availability. Some evaluation has been done by simulating an example clock layout using output models supplied by a subset of the companies listed, along with interconnect models and preliminary clock input model of the CPU-Cache chip set. For more detail on the simulations and example routing, please see Section 5.0. Intel has been and will be working closely with the listed companies to ensure they have the latest specifications for the Pentium processor. With published preliminary data sheets, all the listed parts meet either the CPU or the chip set clock specifications (including the signal quality and skew specifications). Please contact individual manufacturers for data sheets and sample availability.

Table 3. Clock Driver Options

Mfgr	Part #	Driver Type	Level In/Out	Pin-to-Pin Skew (ns)	Part-to-Part Skew (ns)	t_r/t_f (0.8V–2.0V) (ns)	Clock Stability	# of Outputs (per pkg)	Spec'd Loading	Max. Freq.
Intel Spec			TTL inputs	(1)		1.5/1.5	±250 ps (2)			66 MHz and 60 MHz
AMCC	SC35XX-1	Buffer	PECL or TTL/TTL	0.5	1.0	1.5/1.5		20 Outputs Which Vary with Part #	10 pF	80 MHz
	SC44XX-80	PLL	TTL/TTL	±0.2	1.0 (9)	1.5/1.5		6–12 Outputs	35 pF	80 MHz
AT&T (7)	DA400	PLL	PECL or TTL/TTL	0.2	0.5	1.5/1.5		8@1, 0.5X (Prog Shift)	50 pF	100 MHz
Cypress	CY7B991	PLL	TTL/TTL	0.5	1.2 (6)	1.5/1.5	0.5%	4@1X 4@1, 0.5, 0.25X	50Ω/ 30 pF	80 MHz
ICS	ICS2686	PLL		0.5	0.6	1.5/1.5		5@1, 0.5X	20 pF	
Intel	85C224-100	Divider/ Buffer/PLD	TTL/CMOS	Divider 0.4 Buffer 0.5	NA	Divider 1.2/1.1 Buffer 1.4/1.1		8 @ + 1X, −1X, 0.5X	70Ω/ 50 pF	Divider 100/50 Buffer 133
Intel	85C224-10	Buffer/ Divider/PLD	TTL/CMOS	Divider 0.4 Buffer 0.5	NA	Divider 12./1.1 Buffer 1.4/1.1		8 @+1X, −1X, 0.5X	70Ω/ 50 pF	Divider 58/29 Buffer 100
Motorola	MC10H646	Buffer	PECL or TTL/TTL	0.5	1.0	1.2/1.2	NA	8	50Ω/ 50 pF	100 MHz
	88915	PLL	TTL/CMOS	0.5	NA	2.5/2.5(11)	NA	5@1X 1@2X 1@.5X 1 Inverted X	50Ω	66 MHz

Table 3. Clock Driver Options (Continued)

Mfgr	Part #	Driver Type	Level In/Out	Pin-to-Pin Skew (ns)	Part-to-Part Skew (ns)	t_r/t_f (0.8V–2.0V) (ns)	Clock Stability	# of Outputs — (per pkg)	Spec'd Loading	Max. Freq.
National	CGS74CT2524	Buffer	TTL/CMOS	0.45	NA	1.5/1.5		4	50 pF	100 MHz
	CGS74CT2527	Buffer	TTL/CMOS	0.45	NA	1.5/1.5		8	50 pF	100 MHz
	CGS74B2528	Buffer	TTL/TTL	0.55	NA	1.5/1.5		10	50 pF	70 MHz
Pioneer	PI6B2407	PLL	TTL/TTL	±0.25	NA	1.5/1.5	100 ps	12@1, 0.5, 2X (Prog Shift)		80 MHz
TI (8)	CDC328	Buffer	TTL/TTL	0.7	NA	1.2/0.5	NA	6	500Ω/ 50 pF	Not Spec'd.
Triquint (10)	GA1085	PLL	TTL/TTL	0.25	NA	1.4/1.4	75 ps (typ.)	5@1X 4@0.5X 2@0.5X (Prog Shift)	50Ω	66 MHz
	GA1086	PLL	TTL/TTL	0.25	NA	1.4/1.4	75 ps (typ.)	9@1X 1@0.5X	50Ω	66 MHz
	GA1087	PLL	TTL/TTL	0.25	NA	1.4/1.4	75 ps (typ.)	6@1X 4@0.5X	50Ω	66 MHz
Vitesse	VSL4485	PLL	TTL/TTL	0.5	NA	1.5/1.5		6@1X 2@1, 2, 4X	50 pF	70 MHz
	VSL4586	PLL	TTL/TTL	0.5	NA	1.5/1.5		2@1X 6@1, 2, 4X	50 pF	70 MHz

NOTES:
1. 0.7 ns between Pentium processor-82491, 82496-82491, 82491-82491. 0.2 ns between Pentium processor-82496. Assumed 0.5 ns between clock driver outputs, leaving 0.2 ns for routing or trace skew.
2. See complete specification in Table 1 or the data book.
3. Manufacturers listed in alphabetical order.
4. Contact manufacturers for price and availability information.
5. Intel does not guarantee specifications for other manufacturer's devices. All clock driver specifications listed were provided by the manufacturer and are subject to change. Designers should contact the manufacturer for the latest specification/data sheet information.
6. As low as 0.75 ns in some configurations.
7. First samples in March '93. Specifications may improve during characterization.
8. Other Solutions are under development. Contact TI for preliminary details.
9. Maximum phase erro quoted in the manufacturer's data sheet for the entire frequency range.
10. Other configurations available. Contact Triquint for details.
11. Between 0.2 V_{CC} and 0.8 V_{CC}. Contact Motorola for details between 0.8 and 2.0V.

AMCC offers the SC35XX-1 series of buffered clock drivers and the SC44XX-80 series of PLL based clock drivers. The SC35XX-1 series must be driven with a TTL or PECL 2X frequency input. Each member of the series provides 20 outputs. Depending on the specific part within the series, these 20 outputs can be configured to provide the primary frequency, 1/2, or 1/4 the primary frequency. The SC3502-1 even provides 5 inverted outputs of the primary frequency. The SC44XX-80 series must be driven with a TTL input. The PLL design allows for very low skew (±200 ps) between the outputs.Different members of the series offer different numbers and configurations of outputs. Between 4 and 8 outputs are available at the primary frequency. These devices also allow a subset of the outputs to be configured for 1/2X or 2X the primary frequency. In addition, the PLL allows the outputs to be skewed in phase from one another.

AT&T DA400 is a PLL clock driver. Its inputs can be driven by TTL or PECL levels. Eight outputs are provided. They can be configured for the primary frequency or 1/2X the primary frequency. In addition each output has a programmable delay line which allows 1/32 or 1/64 increments of the clock period of delay between outputs.

Cypress's CY7B991 is a PLL clock driver. It requires a TTL input and is able to drive 8 outputs. A subset of the outputs can be configured as 1/2X, 1/4X, or inverted outputs. As with other PLL solutions, the skew between outputs is small and the outputs can be configured for a fixed amount of delay or skew between outputs.

ICS's ICS2686 is a PLL clock driver. Five outputs are available. Both primary and 1/2X frequencies are available. The ICS2686 has been designed to work with the 74ABT240 type buffer to provide more than 5 outputs. A unique feature of the ICS2686 is the multiple feedback inputs. This feature allows synchronizing multiple outputs at their destination or load with the input clock.

Intel's 85C224-100 is a "20V8" architecture programmable logic device. From its TTL inputs it provides 8 TTL outputs which can be configured to provide 1X, 1X inverted, and 1/2X versions of the primary frequency, in any combination. When programmed to function as a frequency divider, the primary frequency can be as high as 100 MHz and the 1/2X frequency outputs will maintain output skew below 400 ps. When programmed to operate as a straight 1X buffer, it supports frequencies of up to 133 MHz with less than 500 ps of output skew. The 85C224-100 provides a combination of superior output signal quality including fast rise and fall times and low output skew. A particularly unique feature of the 85C224-100 is in its programmable logic circuitry. Its flexibility satisfies programmable logic needs such as control line signals and widespread glue logic. With this minimized output skew PLD, a single 28-pin PLCC can provide low output skew clock distribution, frequency division, and programmable logic; for the low price of a 20V8 PLD.

Motorola offers both a buffered and a PLL clock solution. Motorola's 10H646 is a buffered clock driver. It offers both TTL and ECL inputs which supports backplane routing using ECL levels. The clock driver's outputs are clamped to 3V, not V_{CC}. 10H646's output stage has similar rise and fall output resistances. Similar rise and fall output resistances makes series termination easier since the termination resistance is the difference between the characteristic impedance of the transmission line connecting the output to the load and the driver's output impedance. 10H646 has 8 1x outputs. As a straight buffer, 10H646 does not offer any multiples of the input besides 1x. The Motorola 88915 is a PLL clock driver. It provides a 0.5 ns skew between outputs. The 88915 provides 5 1X outputs along with 1 2X, 1 0.5X, and 1 inverted X outputs.

National's clock buffers are packaged to function reliably at high frequencies. Their output rise and fall resistances are approximately equal. The CGS74CT2524 and 2527 provide 0.45 ns of output skew. The CGS74CT2528's output skew, 0.55 ns, allows for only 0.15 ns skew due to board traces or any unbalanced loading effects when using the 82496/82491 cache, however this amount may be sufficient for other cache solutions. These parts offer a range of 4, 8, and 10 outputs. The CGS74CT2524 and 2527 have CMOS level outputs, which transition from rail to rail.

Pioneer's PI6B2407 is a PLL clock driver. From its TTL input, it provides twelve TTL outputs, which can be configured to operate at 1X or 2X the input frequency. In addition, the outputs can be phase adjusted from the input clock. The PI6B2407 is able to provide ±0.25 ns of skew between outputs while maintaining the fast 1.5 ns rise and fall times.

Texas Instruments' ABT328 driver provides six outputs with an output skew of 0.7 ns. Please contact Texas Instruments for the availability of 0.5 ns output skew parts. 0.7 ns output skew is too large for the chip set application. In the design example on Section 5.0, 0.5 ns output skew is assumed. As a buffered driver, the ABT328 offers only 1x outputs.

TriQuint's GA1086 is a Gallium Arsenide-based product. It takes a 66 MHz input and produces nine 66 MHz outputs and one 33 MHz output. The availability of a low skew 33 MHz output facilitates clock distribution for systems that have synchronous 33 MHz memory buses. Since the part is phase-lock-loop based, one of the outputs can be fed back to the input so that all the outputs are synchronized with the input clock. Such a set up is ideal for cascading clock drivers to achieve maximum fanout. The specified output skew of the GA1086 is 0.25 ps, the smallest skew number available. Triquint also offers the GA1085 and GA1087. These products are similar to the GA1086, however, they offer different combinations of outputs between 1X and 0.5X.

Vitesse's VSL4485 is also a Gallium Arsenide-based product. It offers 1x, 2x, and 4x options on two of its eight outputs. Thus, to obtain both 33 MHz and 66 MHz signals with low skew, for example, the clock input frequency of the VSL4485 can be 33 MHz. For the chip set application, two 66 MHz outputs are not enough, and thus cascading another driver is necessary. Alternatively, the input can be 66 MHz and all of its outputs can be at 66 MHz. It offers 0.5 ns output skew, and a low effective delay. In addition, VSL4485 can generate programmable, multiple phase relationships among its outputs.

4.0 CLOCK GENERATION FOR THE Pentium™ PROCESSOR AND THE CPU-CACHE CHIP SET

Clock generation is the generation of copies of clock signals from a signal oscillator or any other source which then are distributed to the various loads. The function of a clock driver is to generate multiple copies of clocks from a single source. In general, Pentium processor-based systems have three types of memory interface: fully synchronous, divided synchronous, and asynchronous. Each interface requires different methods of clock generation. The basic setup of a processor card is illustrated symbolically in Figure 6. Depending on the configuration, the Clock In signal can come from the memory bus or a separate oscillator.

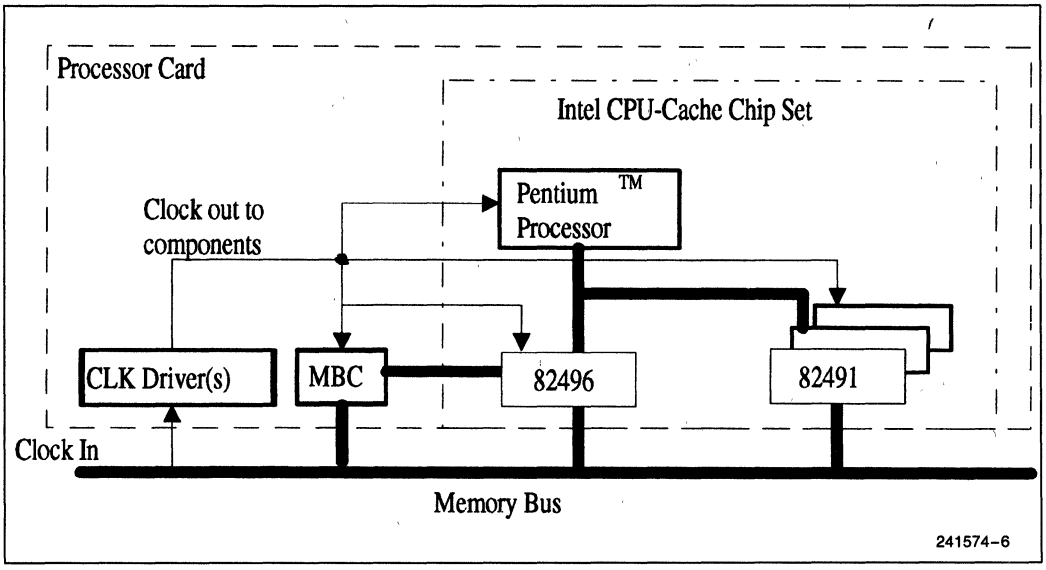


Figure 6. A CPU Module with the Pentium™ Processor, 82496 and 82491 CPU-Cache Chip Set

4.1 Clock Generation for Fully Synchronous Systems

A fully synchronous system is one which everything in the system runs synchronous to the CPU. In particular, the memory bus interface is synchronous to the CPU. In Figure 6, the memory bus is at 66 MHz, synchronous to the CPU module. Clock In signal must be synchronous to the memory bus. Clocking for this case involves the generation of tightly controlled copies of clock signals that are distributed to all the clocked parts. The task of clock generation and distribution is the most difficult for this type of set up. All copies of clock signals must come from a single source, and must be deskewed appropriately. For Pentium processor-based systems that run at 66 MHz, the most critical parameter in choosing a clock driver is its output skew, as well as its part-to-part skew if more than one driver is needed. Since all the clock signals are at 66 MHz, only 1x outputs are needed. All of the drivers listed in Section 3.0 can be used here.

For a fully synchronous configuration, it is likely that a single clock driver cannot provide enough copies of clock signals. Then, some kind of cascading of drivers is necessary. Figure 7 shows two ways of clock generation by cascading drivers. Tskew is the total worst case skew at outputs of CD2 and CD3. Tpp23 is the worst case part-to-part skew between CD2 and CD3. Tos2 is the worst case output skew of CD2, assuming the worst case output skew of CD3 is the same as Tos2. Tos1 is the worst case output skew of CD1. Ttol2 is the feedback tolerance of CD2. Feedback tolerance is the phase tolerance between the feedback input and the reference clock. Typically, Ttol2 is a small number. For the examples in Figure 7, it is assumed that only the second level drivers feed the clock signals to the loads. Otherwise, for part a, signals from CD2 will be later than signals from CD1 by the propagational delay of CD2 which is typically between 6 ns to 8 ns.

For the examples in Figure 7 clock signals for the CPU-Cache chip set must be derived from one clock driver outputs only so that the 0.2 ns and 0.7 ns skew specifications can be met. In part a, Tskew, the sum of Tpp23, Tos2, and Tos1 is the worst case skew which is the skew between an output of CD2, and an output of CD3. The output skew of CD1 (Tos1) causes the inputs to CD2 and CD3 to arrive at different times. The difference in propagational delay which is Tpp23, further skews the outputs of CD2 and CD3. If the part-to-part skew does not include output skew, different outputs from CD2 and CD3 can also be skewed by the output skew. For part b, Tskew, the sum of Ttol2, Tos2, and Tos1, is also the worst case skew between the outputs of CD2 and the outputs of CD3. Once again, Tos1 causes the inputs to CD2 and CD3 to arrive at different times. The feedback in CD2 synchronizes all its outputs in the input. The feedback output of CD2 is different from the input reference clock only by Ttol2. All the other outputs are further skewed from the feedback output by Tos2. The analysis for CD3 is the same.

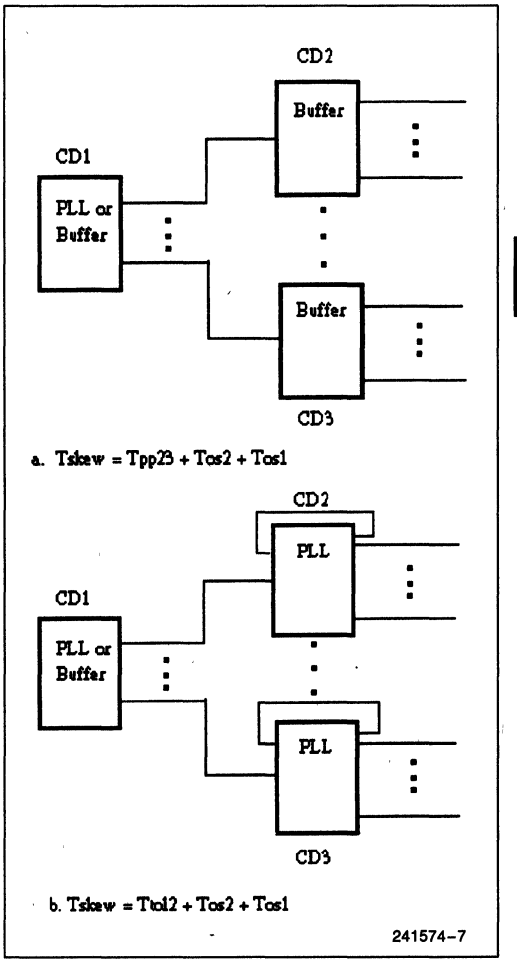


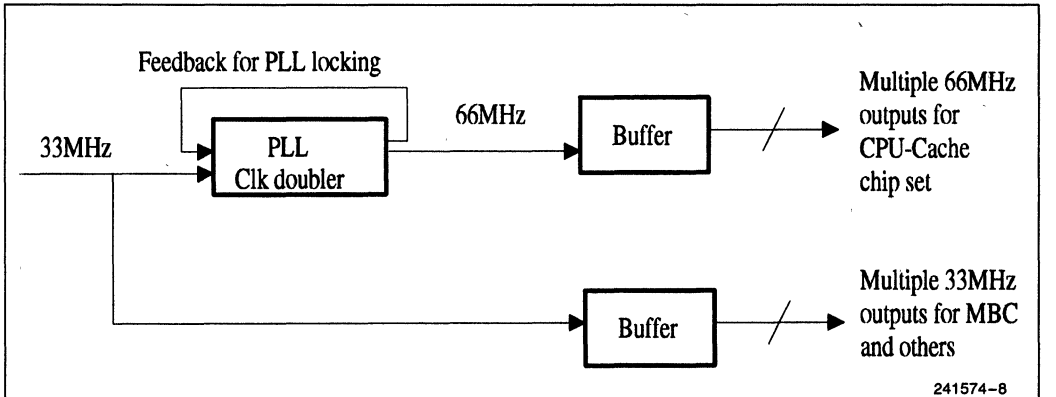
Figure 7. Examples of Clock Generation

4.2 Clock Generation for Divided Synchronous Systems

For a divided synchronous system, the memory bus is at half the speed of the CPU-Cache chip set; i.e.,

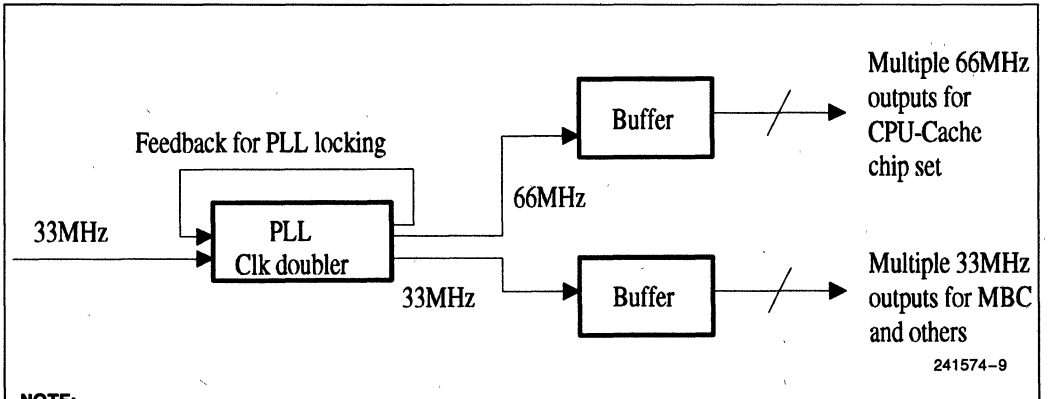
the memory bus runs at 33 MHz for the Pentium processor or the CPU-Cache chip set based systems. A 33 MHz reference clock (Clock In) can come from the backplane from which all the clocks serving the CPU-cache module (Figure 6) must be synchronized. The memory bus controller (MBC) itself requires both 33 MHz and 66 MHz clocks. For this configuration, clock drivers that can provide both 33 MHz and 66 MHz outputs are needed.

There are several ways of providing the two frequencies. They are shown in Figure 8 through Figure 12. Tskew is the worst skew between 33 MHz signals and 66 MHz signals. The skews among 66 MHz signals or among 33 MHz signals are simply the output skew of the driving devices. Ttolpll is the PLL CLK doubler or PLL CLK divider's feedback tolerance. Tospll is the PLL CLK doubler or PLL CLK divider's worst case output skew. Tppbuf is the worst case part-to-part skew of the second level buffers. Those buffers can be phase-lock-loops also in which case Tppbuf is the feedback tolerance of the PLLs if feedback is used. Tosbuf is the worst case output skew of the second level buffers. Tos1 is the output skew of CD1, Ttol2 is the feedback tolerance of CD2, and Tos2 is the output skew of CD2.



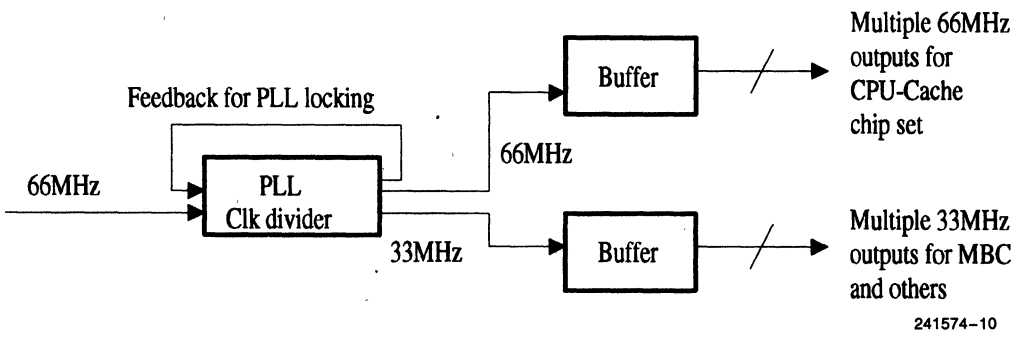
NOTE:
Tskew (between 33 and 66) = Ttolpll + Tospll + Tppbuf + Tosbuf

Figure 8. Clock Generation Using Clock Doubler



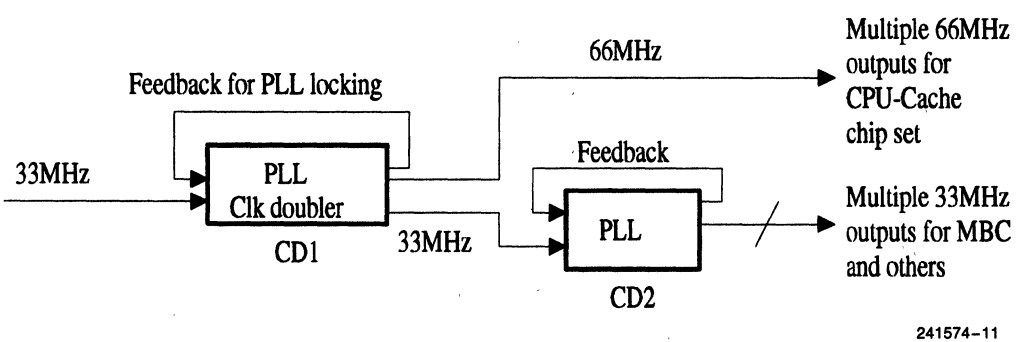
NOTE:
Tskew (between 33 and 66) = Tospll + Tppbuf + Tosbuf

Figure 9. Clock Generation Using Clock Doubler



NOTE:
Tskew (between 33 and 66) = Tospll + Tppbuf + Tosbuf

Figure 10. Clock Generation Using Clock Divider



NOTE:
Tskew (between 33 and 66) ~ Tos1 + Ttol2 + Tos2

Figure 11. Clock Generation Using Two PLLs

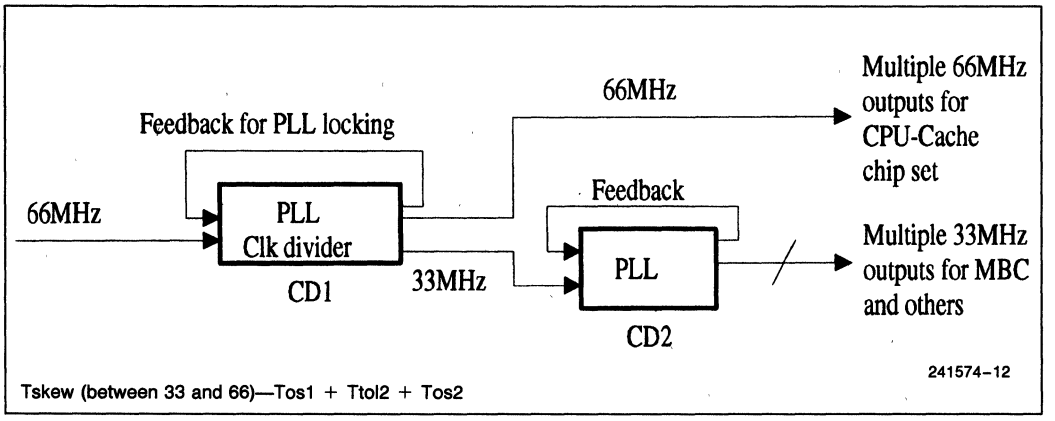


Figure 12. Clock Generation Using Two PLLs

Since the outputs of the first level PLL CLK doublers and dividers go directly to inputs of another clock driver, the signal quality requirements of these outputs are not as stringent as if the outputs drive the loads of the Pentium processor and others. One of the functions of a clock driver is to buffer and clean up a clock signal in addition to generating multiple copies of the same. However, the output skew of the PLL used for the first level is very important. Depending if feedback is used, the feedback tolerance is of importance. When choosing a clock driver, also be sure that its maximum output frequency is greater than 66 MHz for 66 MHz outputs and 33 MHz for 33 MHz outputs. The parts listed in Table 4 and Table 5 are examples of devices that can be used as first level drivers as illustrated in Figure 8 through Figure 12.

Table 4. List of Clock Doubler Parts

Manufacturer	Part #	Driver Type	# of Outputs (per pkg)
Motorola	88915	PLL	1 @ 2x 6 @ 1x (1) 1 @ 0.5x
Motorola	88916	PLL	1 @ 2x 4 @ 1x (1) 1 @ 0.5x
TI	ABT338	PLL	1 @ 2x 4 @ 1x 1 @ 0.5x
Vitesse	VSL4485	PLL	6 @ 1x 2 @ 1, 2, or 4x

NOTES:
1. One of the outputs is inverted.
2. This list is not meant to be complete. Other solutions may be available.

The phase-lock-loop drivers listed in Table 4 can be used to drive the Pentium processor loads directly if only one copy of 66 MHz clock signal is needed. In this case, the second level buffers are not necessary if the driver used can provide enough 33 MHz copies. Intel has not done formal analysis on these parts.

Table 5. List of Clock Divider Parts

Manufacturer	Part #	Driver Type	# of Outputs (per pkg)
Motorola	88915	PLL	1 @ 2x 6 @ 1x (1) 1 @ 0.5x
Motorola	88916	PLL	1 @ 2x 4 @ 1x (1) 1 @ 0.5x
Texas Instruments	ABT338	PLL	1 @ 2x 4 @ 1x 1 @ 0.5x
Texas Instruments	ABT337	Buffer	4 @ 1x 4 @ 0.5x
Texas Instruments	ABT339	Buffer	4 @ 1x 4 @ 0.5x
TriQuint	GA1086	PLL	9 @ 1x 1 @ 0.5x

NOTES:
1. One of the outputs is inverted.
2. This list is not meant to be complete. Other solutions may be available.

Table 5 lists examples of clock drivers that offer divided by 2 outputs. These devices can be used as the first level drivers illustrated in Figure 8 through Figure 12. Depending on the number of 66 MHz copies and 33 MHz copies needed, the second level buffers may not be necessary. Again, Intel has not performed any formal analysis on these parts.

4.3 Clock Generation for Asynchronous Systems

If the memory bus is not synchronized with the CPU or CPU-cache module, clock generation for the system is easier compared with the two configurations above. However, clock synchronization for the Pentium processor, 82496, and 82491, as well as the clocks for the MBC is still a concern. In order for the MBC to communicate properly with the CPU-Cache chip set, some synchronized clocks at 66 MHz are needed. Since the system is asynchronous to the CPU-Cache chip set, the number of synchronous MBC clock signals is less than the synchronous case. The examples in Section 5.0 illustrate how the synchronization is done. Since the system is asynchronous, one can use a different clock source for the CPU-Cache chip set from the rest of the system.

5.0 Pentium™ PROCESSOR WITH 256K 82496/82491 SECOND LEVEL CACHE CLOCK DISTRIBUTION DESIGN EXAMPLES

After a clock generation scheme is determined, careful analysis must be done on clock distribution to ensure minimal skew and proper rise and fall times. Clock distribution is the connection between clock driver outputs and clock inputs of the components that need clocking. Preliminary analysis has been done on several of the drivers listed in Section 3.0. The following examples show in detail how to terminate transmission lines properly, tune clock traces to minimize board trace skew, and validate the usefulness of the drivers to the CPU-Cache chip set using models from the manufacturers. The examples have been done using preliminary or typical models for the devices involved. They are meant as an example of the process designers can use when selecting and routing a clock circuit. Although the examples only show the clock distribution for the CPU-Cache chip set, the same principles can be applied to distribution to the memory bus controller (MBC) and other parts.

5.1 Clock Routing for the 256K CPU-Cache Chip Set

Analysis for CPU-Cache chip set clock routing is done using first order input models for the three chips, shown in Figure 13. Specific to CLK inputs, the models shown are **typical models** based on on-going simulation efforts, and are subject to change. Refer to the Intel data books or contact Intel for the latest models (including minimum and maximum conditions). The models include package inductance, package capacitance, and input buffer capacitance of the clock pins.

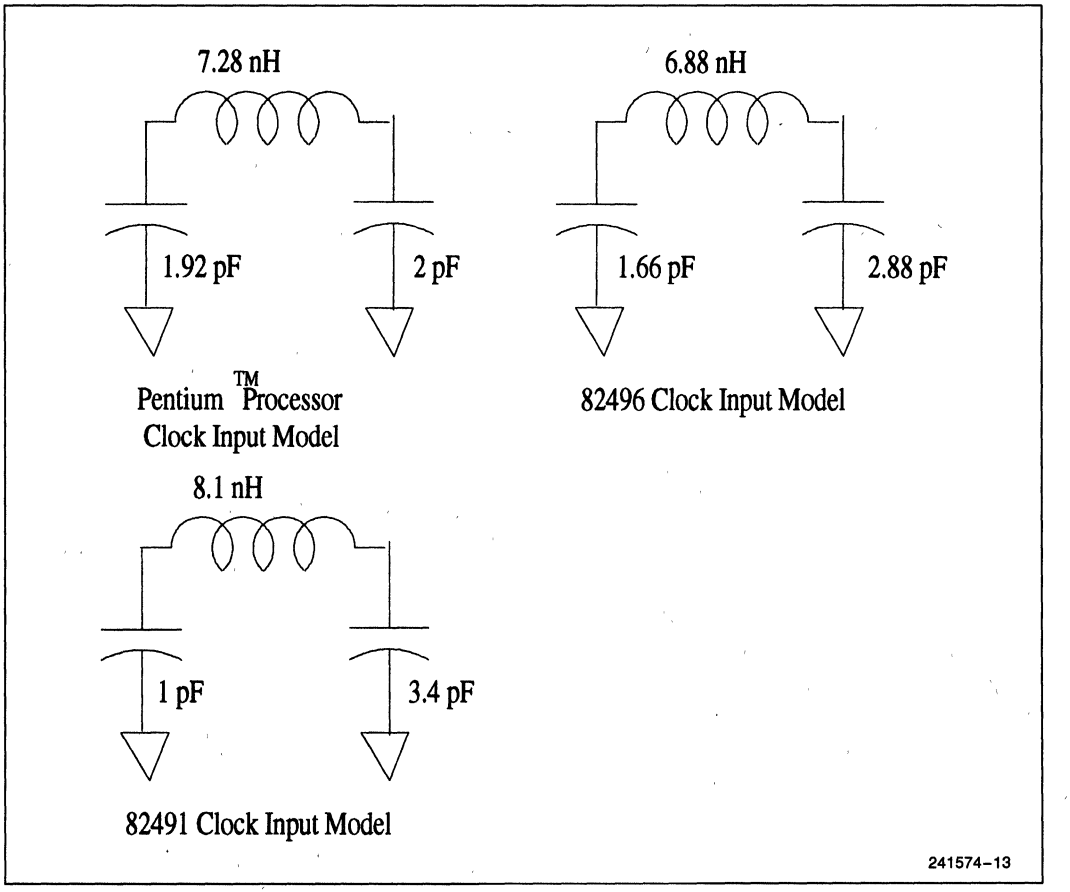
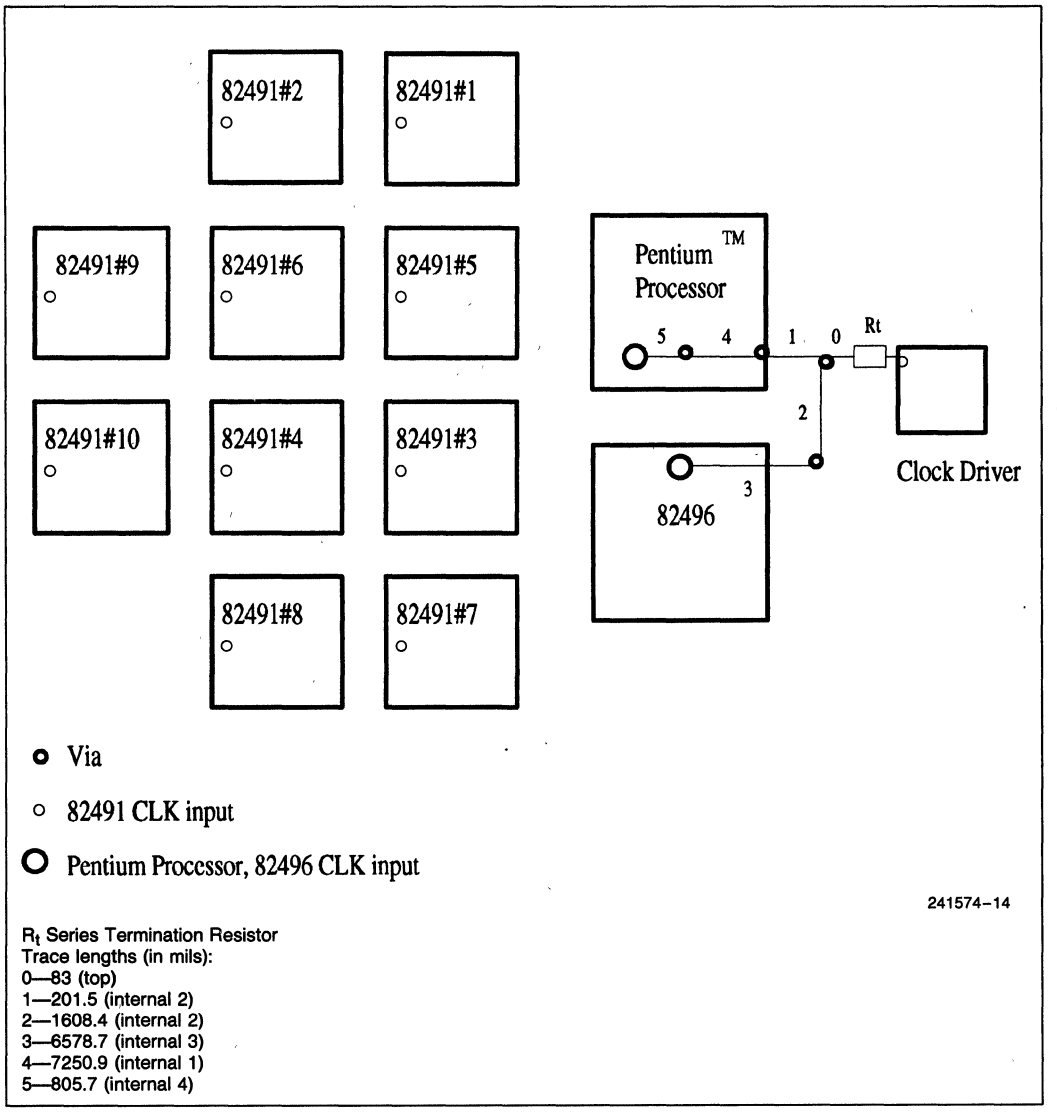
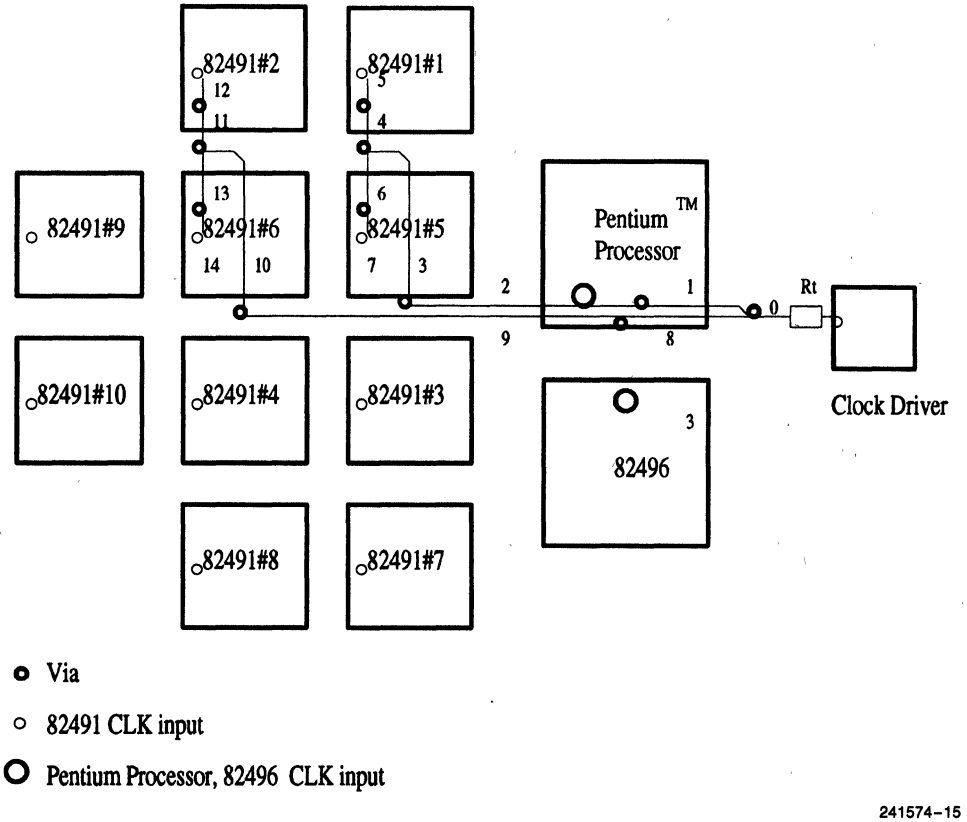


Figure 13. Pentium™ Processor, 82496 and 82491 Clock Input Models



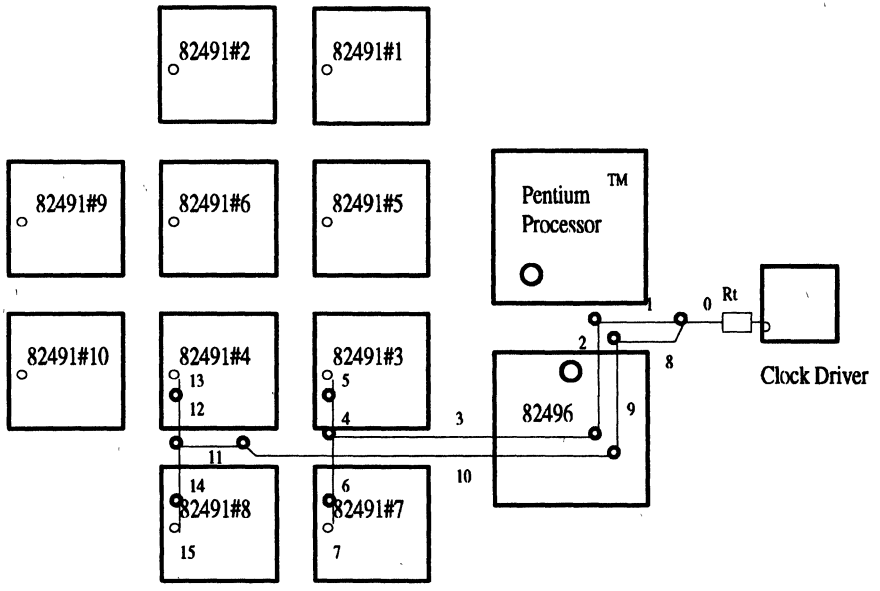
R_t Series Termination Resistor
Trace lengths (in mils):
0—83 (top)
1—201.5 (internal 2)
2—1608.4 (internal 2)
3—6578.7 (internal 3)
4—7250.9 (internal 1)
5—805.7 (internal 4)

Figure 14. CLK0 Layout for 256K Chip Set with Parity



R_t Series Termination Resistor
Trace lengths (in mils):
0—78 (top)
1—455.3 (internal 2)
2—3639.3 (internal 1)
3—2180.7 (internal 4)
4—530.7 (internal 4)
5—75 (top)
6—727.2 (internal 4)
7—75 (top)
8—462.4 (internal 2)
9—4650 (internal 1)
10—1156.7 (internal 4)
11—526.5 (internal 4)
12—75 (top)
13—520.7 (internal 4)
14—75 (top)

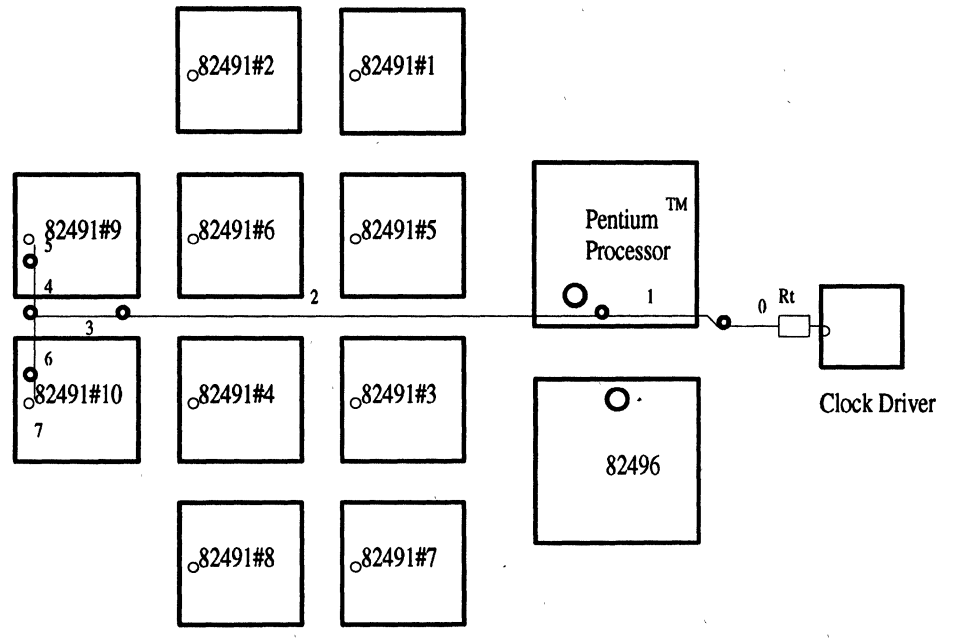
Figure 15. CLK1 Layout for 256K Chip Set with Parity



241574-16

R_t Series Termination Resistor
Trace lengths (in mils):
0—78 (top)
1—206.5 (internal 2)
2—3480.5 (internal 1)
3—2550.9 (internal 4)
4—560.2 (internal 4)
5—75 (top)
6—527.2 (internal 4)
7—75 (top)
8—160 (internal 2)
9—1236.0 (internal 2)
10—4290.9 (internal 3)
11—631.0 (internal 1)
12—530.7 (internal 4)
13—75 (top)
14—534.3 (internal 4)
15—75 (top)

Figure 16. CLK2 Layout for 256K Chip Set with Parity



R_t Series Termination Resistor
Trace lengths (in mils):
0—75 (top)
1—562.4 (internal 2)
2—5897.2 (internal 3); 7147.2 for National 74CT2527 as clock driver
3—631.0 (internal 4)
4—526.5 (internal 4)
5—75 (top)
6—520.7 (internal 4)
7—75 (top)

Figure 17. CLK3 Layout for 256K Chip Set with Parity

Figure 14 through Figure 17 show how clock signals are distributed from the clock driver to each component in the CPU-Cache chip set. Each clock line is tuned to minimize skew. Series termination is used on each line. Since the 82491, numbers 9 and 10, are parity chips, they are grouped onto the same driver output. Since the skew requirement between the Pentium processor and the 82496 is very tight (0.2 ns), they are also on the same driver output. All the loads on the same driver output can be tuned to have close to zero skew. Loads on different outputs, however, must contend with the output skew of the driver. CLK0 through CLK2 are layed out such that they branch off very close to the driver. For these lines, the transmission line characteristic impedance that the clock driver output sees can be treated as two resistors in parallel. In other words, the driver output sees half the impedance for CLK0, CLK1, and CLK2. The advantage of this scheme is that the value of the termination resistor is reduced dramatically. A smaller termination resistor helps faster rise and fall times. For CLK3, the branch off is at the end of the line, and thus the driver output sees the full characteristic impedance.

To achieve minimal skew, all the loads should be balanced, i.e., all the loads should be the same. For this design example, CLK1 and CLK2 have about twice the loads of CLK0 and CLK3. The load imbalance will add to the output skew quoted by manufacturers since on data sheets, manufacturers generally quote output skew for balanced loads. Since heavier loads see the transmitted signal later than lighter loads assuming the transmission lines are of the same length, traces for the lighter loads can be made longer to compensate for the discrepancy. CLK0 and CLK3 have longer traces from driver output to load than CLK1 and CLK2 traces. Since heavier loads (higher capacitance) have a longer rise time, and since for the CPU-Cache chip set skew measurements are taken at 0.8V, 1.5V, and 2.0V, to minimize skew at all free points, the termination resistors for CLK1 and CLK2 should be smaller than the termination resistors on CLK0. However, a smaller termination resistor than the value needed to perfectly terminate the line will result in a larger undershoot. When choosing termination values, it is a trade off among rise/fall times, skew, and undershoot.

When choosing a termination value, it is important to know the output impedance of the driver. For many TTL drivers, output rise impedance is different from output fall impedance. [Refrence 3, Section 9] shows how to measure output impedance, or the driver manufacturer can be contacted for the information. Typically, output fall impedance is 5Ω–10Ω, and rise impedance is 5Ω–50Ω.

Figure 14 through Figure 17 are extensions to the layout topologies in the *Pentium™ Processor, 82496, and 82491 256K CPU-Cache Chip Set Layout Example*. The routing topologies 15–18 shown in the example route the clock signals from the Pentium processor, 82496, and all the 82491's to the outside. The layout examples shown in this application note (Figure 14 through Figure 17) take the layout all the way to the clock driver, complete with termination. Although in the example, clock signals are routed toward the bottom and in the examples here, the clock signals are routed toward the side, the same principles apply. If routing toward the bottom is preferred, the same layouts as illustrated in Figure 14 through Figure 17 can be used with little or no modification.

5.2 Analysis of Drivers Used in Examples

Output models for MC10H646, CGS74CT2527, GA1086, and VSL4485 are used to drive the clock network described in Section 5.1. The clock networks shown in Figure 14 through Figure 17 were used. The simulations assume no variation in characteristic impedance and propagation speed for the board traces. Fast and slow simulations were performed. Three sigma clock driver models are used when available. Board traces are assumed to have plus/minus 10% variation in characteristic impedance and propagational speed. Table 6 shows the range of trace characteristics. Slow simulations assume the highest operating temperature the drivers expect to see, and slow interconnect characteristics. Fast simulations assume operating temperature to be zero and fast interconnect characteristics.

Table 6. Interconnect Characteristics

Corner	Trace Type	Z0(1) (Ω)	TD(2) (ns/ft)
Slow	Inner	58.5	2.41
Fast	Inner	71.5	1.85
Slow	Surface	72	2.05
Fast	Surface	108	1.35

NOTES:
1. Characteristic Impedance
2. Propagational Speed

Since simulation can only account for skew due to board trace and load imbalance, total skew is assumed to be the sum of the worst case output skew published by driver manufacturers and skew from simulation. Skew from simulation is derived by using identical driver models for each driver output, thus assuming zero

output skew. The board traces and termination resistances are tuned with 0.5 ns output skew in mind which leaves 0.2 ns for trace skew. For TriQuint's GA1086, the output skew is 0.25 ns; thus, there is a larger window for trace skew. Table 7 summarizes simulation results of the tightest parameters for each driver. All of the drivers can meet the 4 ns minimum high and low times easily. Most clock drivers guarantee a 45/55 duty cycle, which exceeds Intel's requirement.

Series termination resistors are chosen to minimize skew and undershoot. To achieve similar rise time for each load, termination resistance values are smaller for heavier loaded lines such as CLK1 and CLK2 compared to the resistance values for CLK0. Since CLK3 splits off at the end of the line, its termination value is about twice as CLK0's. Table 8 lists the termination values for each line and for each driver. Waveforms for each driver are attached. Notice the signals at the CLK input for each load is relatively clean whereas the signals at the driver side are not. Since the clock signals are only important to the component receiving the signal, how dirty the signal is at the driver end is not important, providing that the signal does not cause any damage or other ill effects on the driver.

Figures attached in this section show some waveforms from the simulations. V(201) is the voltage at the Pentium processor clock input, V(202) is the voltage at the 82496 clock input, and V(213), V(214), V(217), and V(218) are voltages at the 82491 clock inputs for the 82491s on CLK1 line. For 74CT2527, V(8) is the voltage at driver output, V(100) is the voltage at the junction of the series termination resistor and the beginning of board trace. For 10H646, V(9) is the voltage at driver output, V(20) is the voltage at the junction of the series termination resistor and the beginning of board trace.

Table 7. Compilation of Simulation Data

Mfr.	Clock Driver	Worst Skew P5-C5C (ns)(1)			Worst Skew C8C-Others (ns)			Worst Skew C8C (No Parity) (ns)(2)			Undershoot (−mV)(3)			Tr/Tf (ns)(4)		
		Slow	Fast	Spec	Slow	Fast	Spec	Slow	Fast	Spec	Slow	Fast	Spec	Slow	Fast	Spec
Motorola	10H646	0.021	0.023	0.2	0.65	0.67	0.7	0.65	0.67	0.7	468	816	1600	0.90/ 1.13	0.74/ 0.67	1.5/ 1.5
National	74CT2527	0.0071	(5)	0.2	0.67	(5)	0.7	0.61	(5)	0.7	285	(5)	1600	1.14/ 0.42	(5)	1.5/ 1.5
TriQuint	GA1086	0	0	0.2	0.55	0.45	0.7	0.45	0.45	0.7	150	400	1600	0.9/ 1.9 (6)	0.6/ 1.2	1.5/ 1.5
Vitesse	VSL4485	0.02	0.05	0.2	0.7	0.57	0.7	0.66	0.57	0.7	275	800	1600	0.95/ 0.78	0.78/ 0.6	1.5/ 1.5

NOTES:
1. All Skews are worst case numbers
2. Not using the parity chips
3. Worst Undershoot of all the CLK nodes
4. Slowest rise and fall times of all the CLK nodes
5. Only typical model at 25°C is available. Thus, only simulation performed is with slow interconnect corner
6. Simulation done on driver slow corner. Device specification for t_f is 1.4 ns worst case. Device was still under development when simulation was done. Please contact TriQuint for more information.

Clock distribution method for the memory bus controller (MBC) is very similar to that of the chip set. When distributing clocks for the MBC, be sure to load each driver output with similar loads as for the chip set, and route clock traces with similar lengths as for the chip set. For example, CLK1 and CLK2 have an aggregate load of about 20 pF, and the total clock trace length is about 7″ from driver output to a load. To minimize the clock skew of the MBC clock from loads on CLK1 and CLK2 lines, the clock lines should fan out 2.9 pF per inch. Also, be sure to terminate the line properly. It is important to keep the loading similar to the loadings on clock lines of the chip set if skew is to be kept close to 0.7 ns. Adjusting trace lengths and termination resistance can compensate for load imbalance to a degree, but not perfectly and not always.

Simulations results provided here are based on best available models at the time. Some models were for parts still under development at the time of simulation. Therefore, the simulation results are subject to change.

Table 8. Series Termination Resistor Values for Each Line

Manufacturer	Clock	CLK Line	R_t (Ω)
Motorola	10H646	CLK0	26
		CLK1	20
		CLK2	20
		CLK3	59
National	74CT2527	CLK0	20
		CLK1	15
		CLK2	15
		CLK3	45
TriQuint	GA1086	CLK0	30
		CLK1	30
		CLK2	30
		CLK3	50
Vitesse	VSL4485	CLK0	32
		CLK1	23
		CLK2	23
		CLK3	48

2

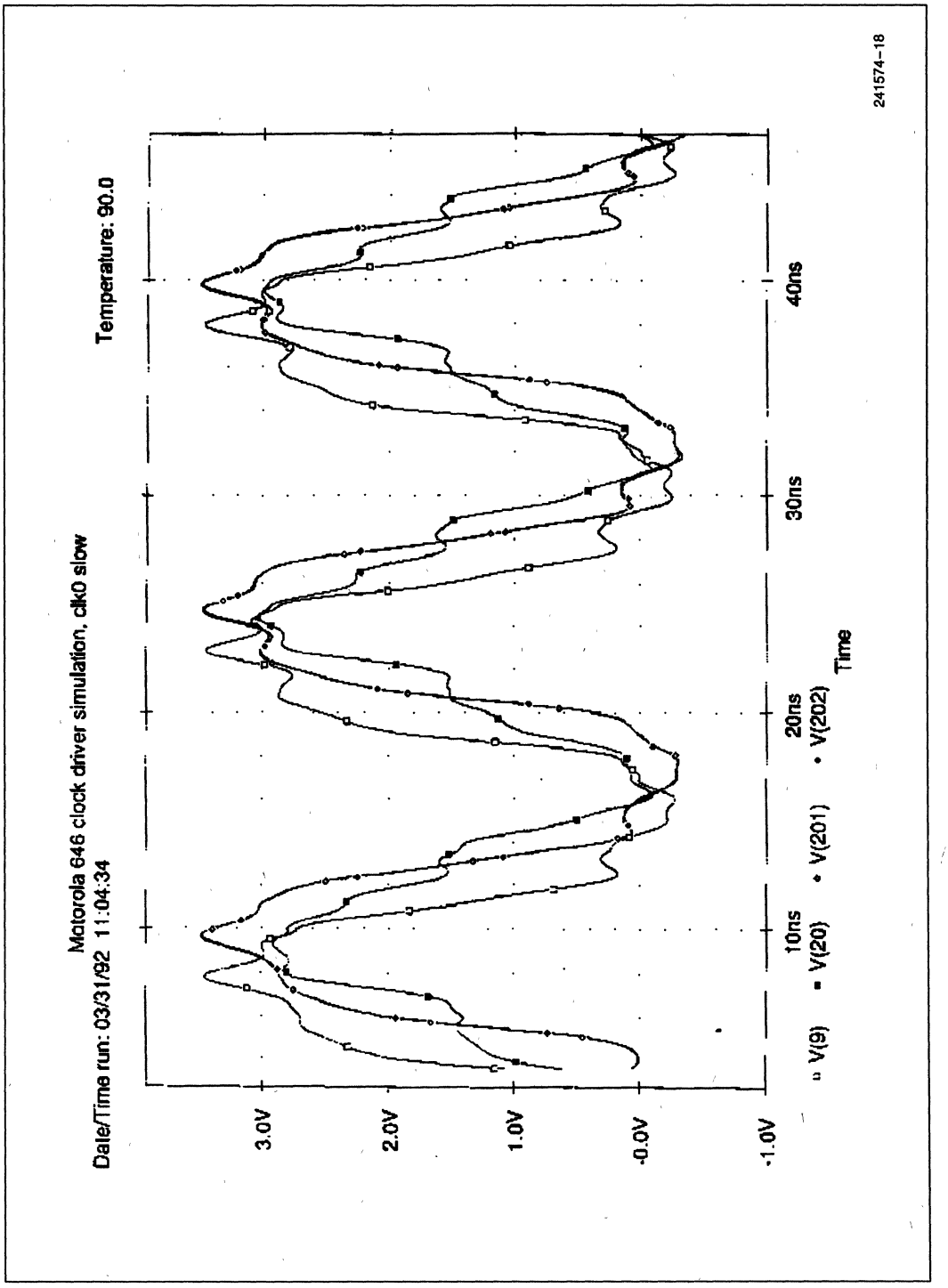


Figure 18. Motorola Waveform

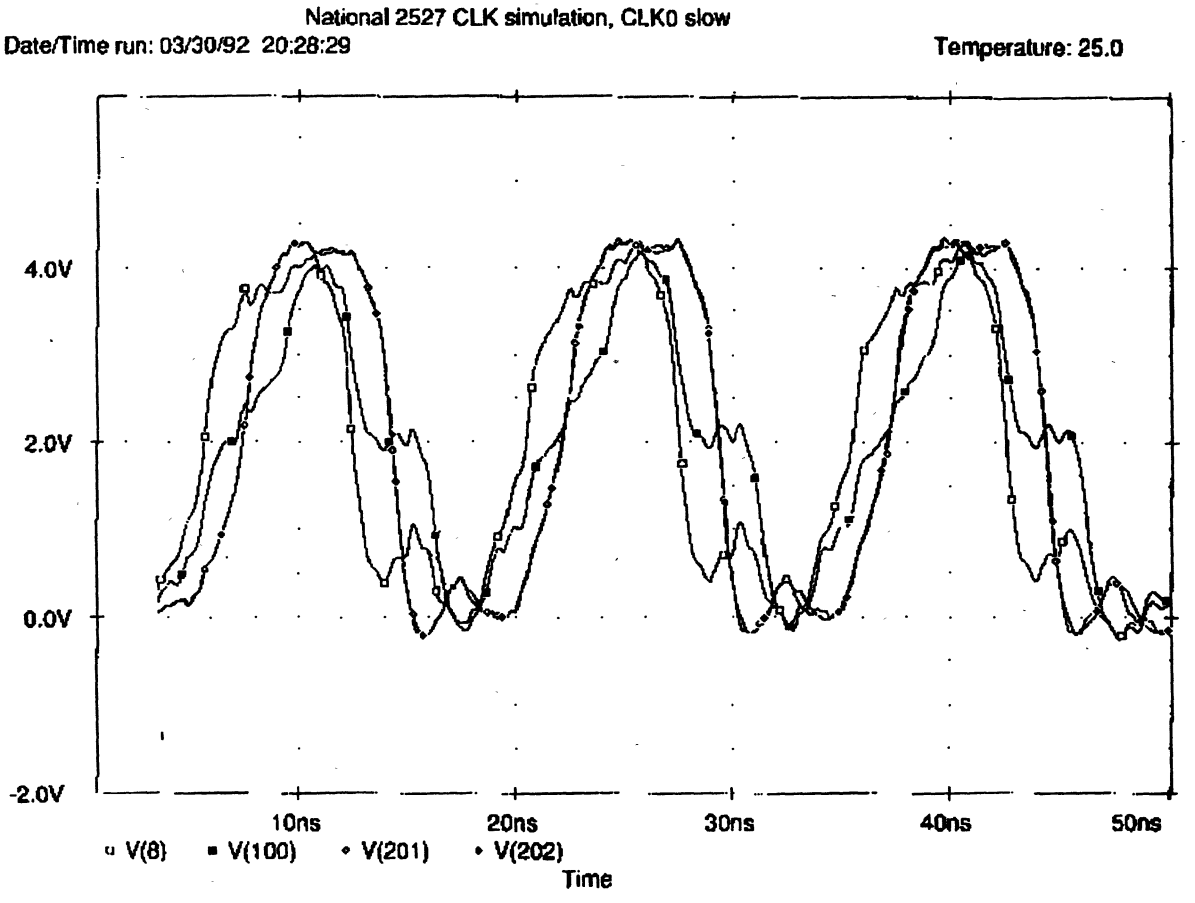


Figure 19. National Waveform

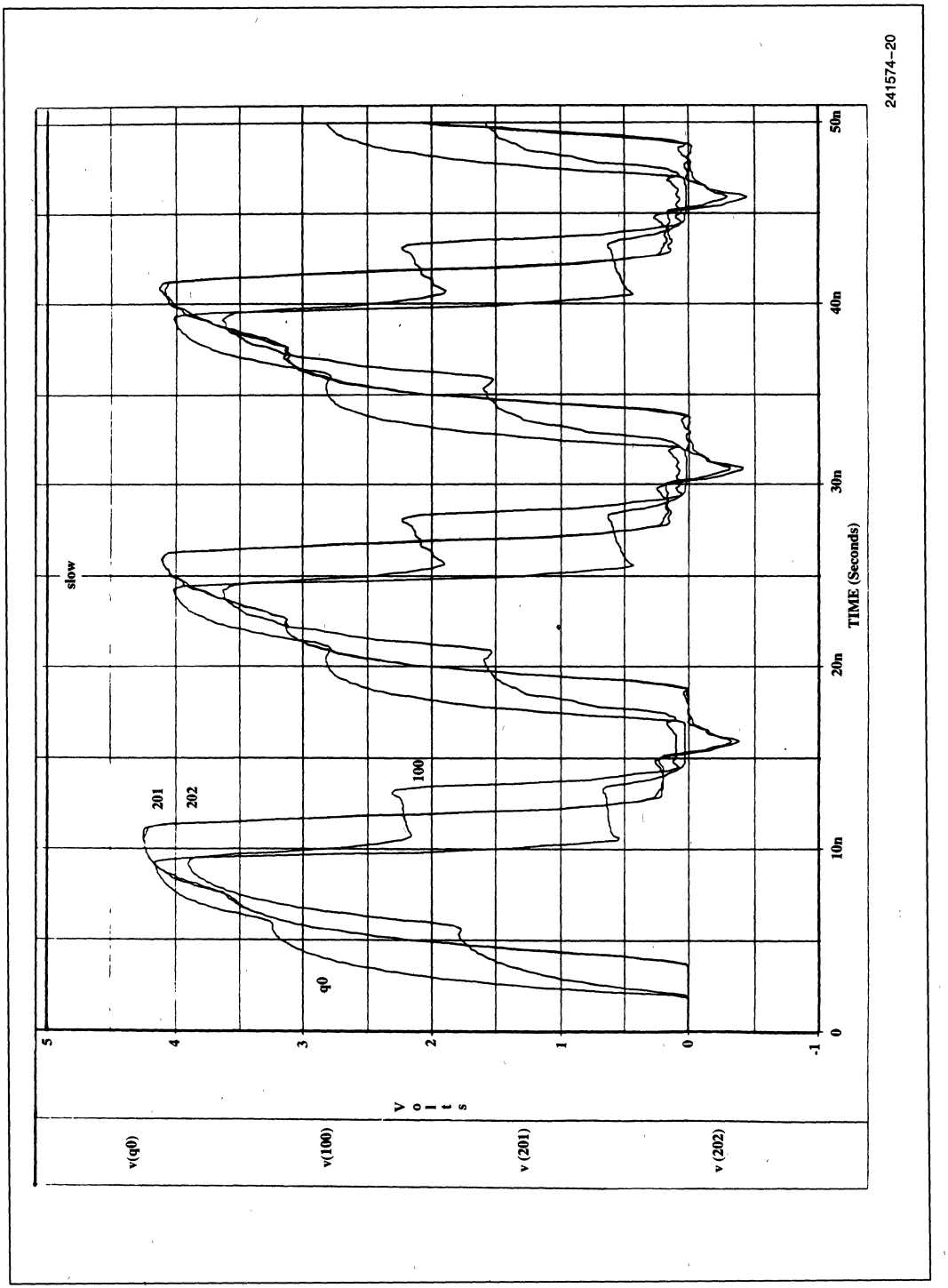


Figure 20. Vitesse (Slow) Waveform

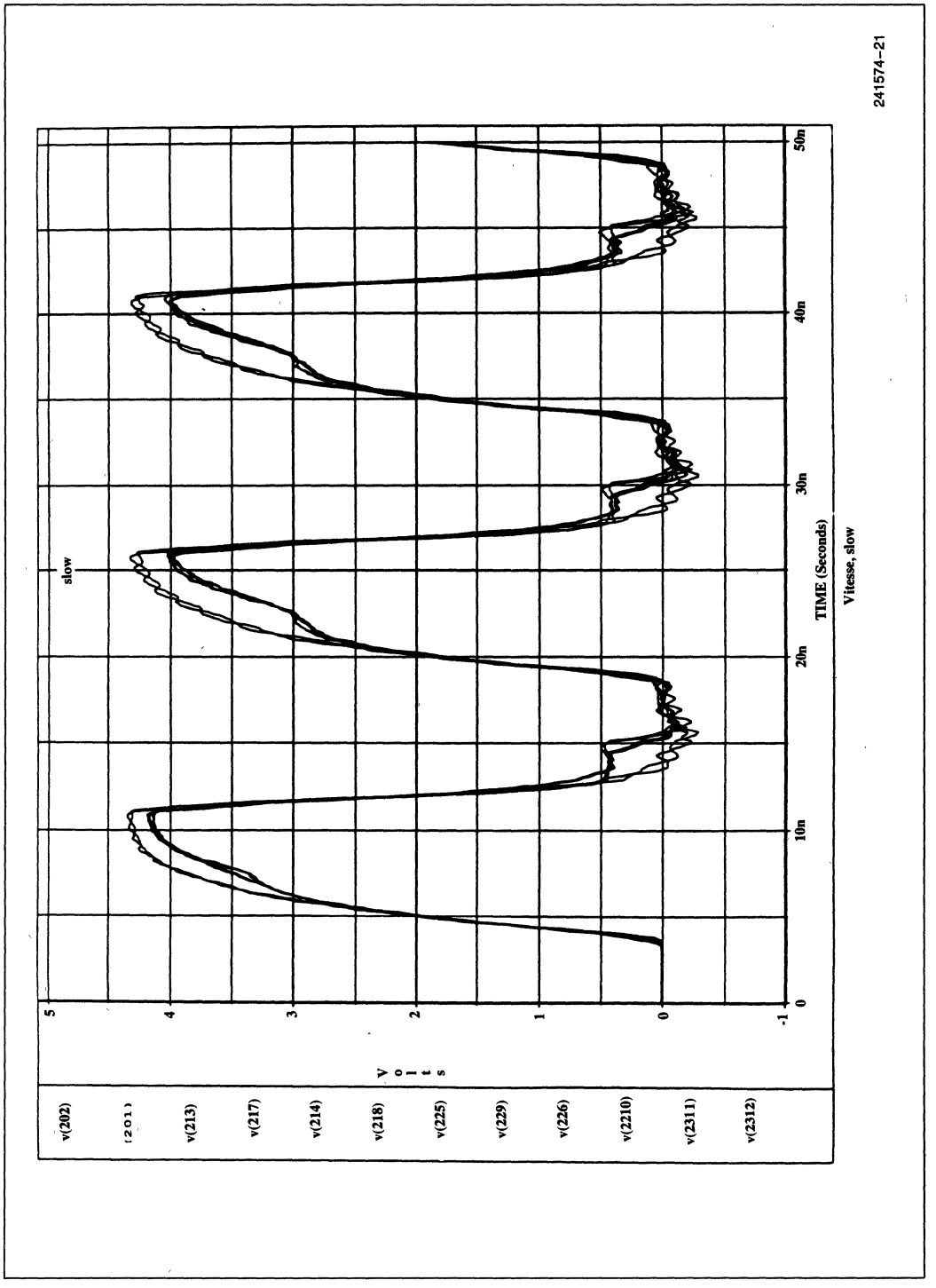


Figure 21. Vitesse (Slow) Waveform (Continued)

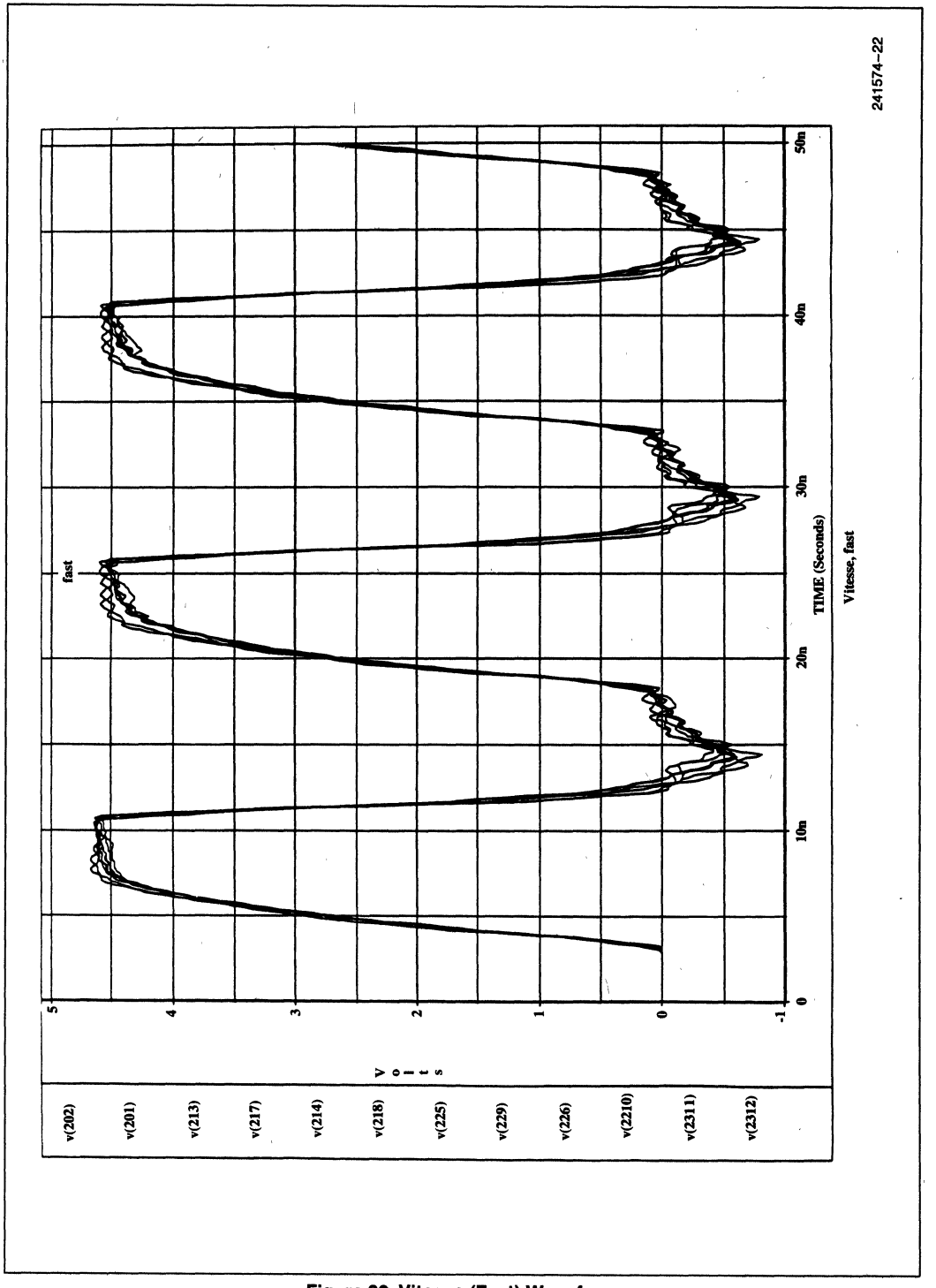
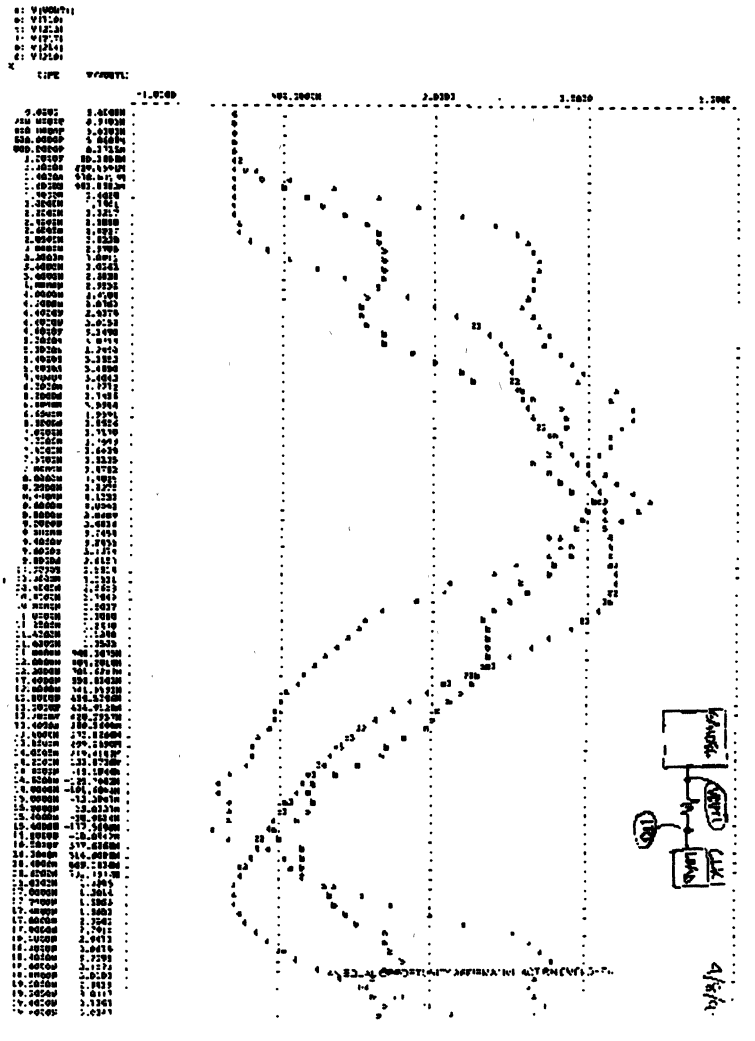
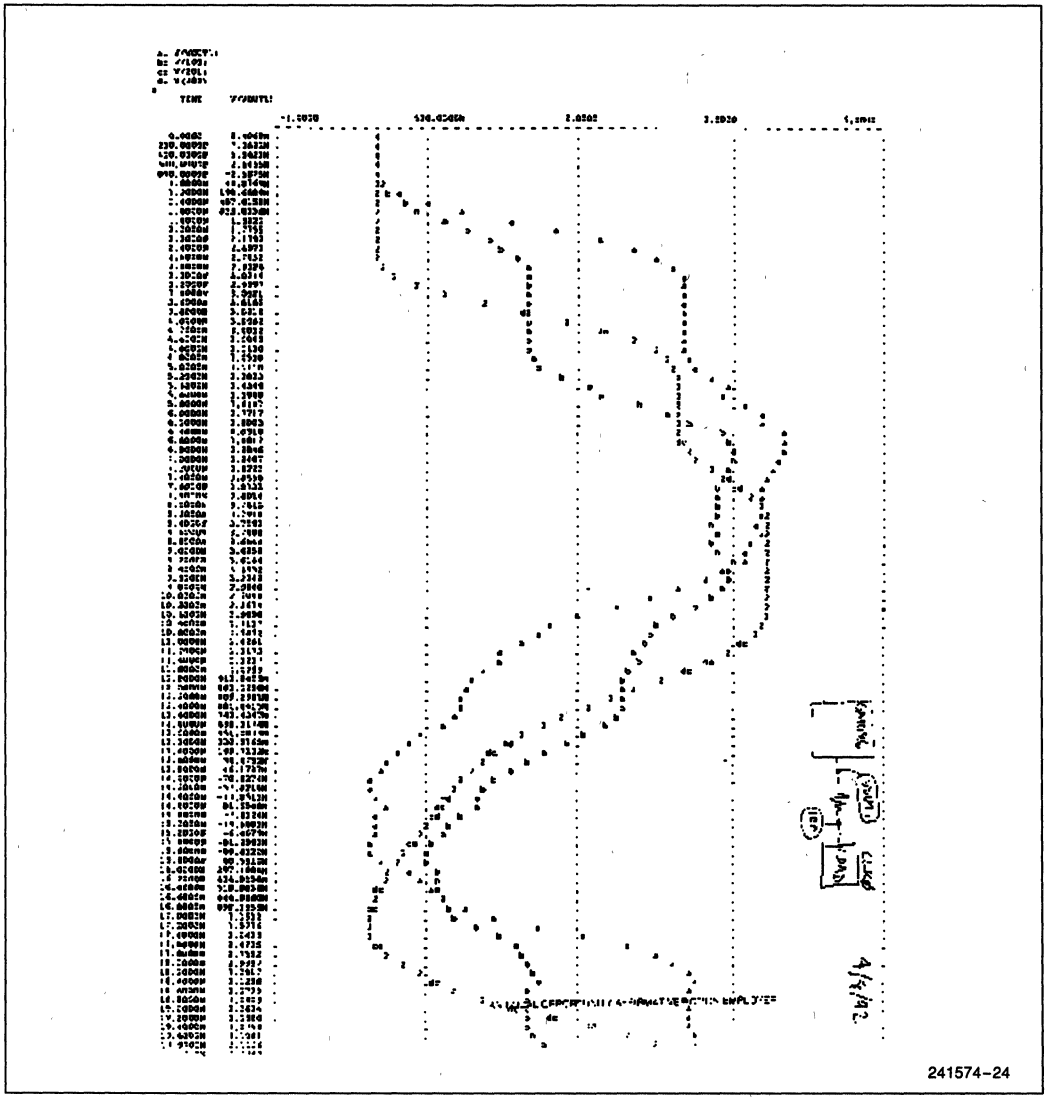


Figure 22. Vitesse (Fast) Waveform



241574-23

Figure 23. Triquint Waveform



241574-24

Figure 24. Triquint Waveform (Continued)

6.0 Pentium™ PROCESSOR WITH 512K 82496/82491 SECOND LEVEL CACHE CLOCK DISTRIBUTION ISSUES

Clock distribution for 512K CPU-Cache chip set can be done in the same way as for the 256K chip set. Since there are more SRAM chips for the 512K cache, there are more loads that need clocking. Including parity, there are 18 82491s. Once again, the same principles apply. Keep the driver loading as close to balanced as possible. Tune traces and adjust termination resistance so that skew is minimized.

7.0 CLOCK DISTRIBUTION FOR THE Pentium™ PROCESSOR WITH OTHER SECOND LEVEL CACHES

The Pentium processor can be used with cache configurations other than with the 82496 and 82491, as well as, without a second level cache. With other caches, the first thing that must be done is to decide how much skew is tolerable. Then, decide on which clock driver to use and carefully layout clock signals for distribution. If skew requirements do not exceed the CPU-Cache chip set requirements, the same drivers and the same distribution can be used. Design examples in Section 5.0 serve as a guide to how to distribute clocks for Pentium processor systems with tight skew.

If the Pentium processor is used without a second level cache, and only a small number of 66 MHz signals are needed, there are a few more options for clock drivers. For example, Motorola's 88915 has one 2x output that can run to maximum 70 MHz. Texas Instruments has the ABT337, 338, and 339 that can provide four copies of 66 MHz signals.

8.0 SUMMARY

At high speeds, clock synchronization becomes a difficult problem. Clock traces must be treated as transmission lines. Proper termination must be given to the lines to ensure good signal quality. The Pentium processor, with operating frequencies of 60 MHz and 66 MHz, has tight clock requirements. Together with the 82496 Cache Controller and 82491 Cache SRAM, the CPU-Cache chip set must be synchronized with minimal skew.

For the Pentium processor clocking, the most critical parameters are skew and rise and fall times. Depending on the memory interface to the CPU-Cache chip set, there are many ways of generating multiple copies of clock signals.

Fully synchronous designs need to route 66 MHz only, but with minimal skew for all of them. Divided synchronous designs require both 66 MHz and 33 MHz signals. Asynchronous designs need to worry about the CPU-Cache chip set clock generation and distribution as well as the MBC.

Several clock drivers have been analyzed in detail with carefully tuned clock routing and the proper termination such that the clock signals transmitted to the Pentium processor, 82496, and 82491 meet all the timing requirements of the Intel chip set parts. Loading on a clock driver should be as balanced as possible. Clock traces should have equivalent length from driver output to load. The clock lines should be terminated properly to minimize reflections.

The same design principles used in the 256K CPU-Cache chip set clocking example can be applied to other CPU-cache configurations, or to a cacheless interface.

This application note has listed a number of devices from several different manufacturers. The purpose of this list is to supply a starting point for finding a clocking solution that meets each system's specific requirements. The lists provided are not meant as an endorsement or guarantee of the devices listed. In addition, these lists are not a complete listing of devices. These or other manufacturers may offer additional devices that meet the clock specifications for the Pentium processor.

9.0 REFERENCES

1. Intel Corporation, *Pentium™ Processor User's Manual*, Order Number: 241563.
2. Intel Corporation, *Designing with the Pentium™ Processor, 82496, and 82491 256K CPU-Cache Chip Set*, Order Number: 241576.
3. Jolly, Rich, *Clock Design in 50 MHz Intel486™ Systems*, Application Note AP-453, 1991, Intel Corporation, Santa Clara, CA.
4. Blood, William R., Jr., *MECL System Design Handbook*, 1988, Motorola Inc.
5. Hanke, Chris and Tharalson, Gary, *Low Skew Clock Drivers and their System Design Considerations*, Application Note AP-1091, Motorola Inc., 1990.

APPENDIX A
CLOCK DRIVER MANUFACTURERS

The following is a list of contacts for the clock driver manufacturers listed in this application note. It is not meant to be an exhaustive list of all possible solutions. It is meant as a starting point for system designers to assist in finding a clock solution that meets their system requirements.

AMCC

United States:
Headquarters
6195 Lusk Boulevard
San Diego, CA 92121-2793
Ph: 619-450-9333 or 800-PLL-AMCC (755-2622)
FAX: 619-450-9885

Europe:

Amega Electronics
Basingstoke, RG24OPF, U.K.
Ph: 011/44-256-843166

Japan:

Teksel Co., Ltd.
Kawasaki 213
Tokyo, Japan
Ph: 011/81-448127430

Israel:

EIM
Petach Tiqva, Israel
Ph: 011/972-3-9233257

AT&T Microelectronics

AT&T Customer Response Center
Ph: 800-372-2447 x773

Danny George
555 Union Blvd.
Allentown, PA 18103
50N2G2100
Ph: 215-439-6697

Cypress

Sean Dingman
3901 N. 1st St.
San Jose, CA 95134
408-943-2743

ICS

Bruce Rogers
Technical Marketing Manager
2626 Van Buren Ave.
P.O. Box 968
Valley Forge, PA 19482
215-666-1900

Intel PLD BU International Contact List

United States:

John Van Sack
Intel Corporation
FM4-42
1900 Prairie City Road
Folsom, CA 95630
Ph: (916) 356-3964
FAX: (916) 356-6949

Europe:

Tony O'Sullivan
Intel Corporation GmbH
Dornacher Str. 1
PostFach 213
D-8016 FeldKirchen/Munchen
Germany
Ph: (49) 89/90992-340
FAX: (49)89/9043948

Japan:

Norikazu Aoki
5-6 Tokodia, Tsukuba-shi
Ibaraki-Ken 300-26
Japan
Ph: 0298-47-0721
FAX: 0298-47-8819

APAC:

Eric Chan
Intel Technology SDN BHD
Bayan Lepas Free Trade Zone,
Box 121
11900 Penang
Malaysia
Ph: 604-820-7271
FAX: 604-836-405

Motorola Inc.

Todd Pearson
Motorola Inc
2200 W. Broadway Rd.
Mesa, Arizona 85202
USA
Ph: (602) 962-3410

Masanori Matsubara
Nippon Motorola LTD
3-20-1, Minami-Azabu
Minato Ku, Tokyo 106
Japan
Ph: 81-33-280-8383

Axel Krepil
Motorola GMBH
Schatzbogen 7
8000 Munchen 81
Germany
Ph: 49-89-92103-167

Derek Leung
Motorola Hong Kong LTD
Silicon Harbour Center
2 Dai King Street
Taipo Industrial Estate
Taipo N. T. Hong Kong
Ph: 852-666-8194

National Semiconductor

National Semiconductor
Santa Clara, CA
Tony Ochoa
Ph: 408-721-6804
Ph: 800-272-9959

Pioneer Semiconductor

Joe Kraus
2343 Bering Dr.
San Jose, CA 95131
Ph: 408-435-0800
FAX: 408-435-1100

Texas Instruments

United States:

Steve Plote
Program Manager
CLOCK DRIVERS
8330 LBJ Freeway, Center 3
P.O. Box 655303
Dallas, Texas 75265
Ph: 214-997-5214

Brett Clark
Applications Engineer
Ph: 903-868-5836

Japan:

Mich Komatsu
Texas Instruments Japan LTD.
M.S. Shibaura Bldg. 13-23
Shibaura 4-Chome
Minato-ku Tokyo, 108 Japan
Ph: 033-769-8717

Asia Pacific Region:

Eric Wey
Texas Instruments Taiwan LTD.
Taipei Branch
10F Bank Tower, 205 Tung Hua N.
Taipei, Taiwan ROC
Ph: 886-2-713-9311

Europe:

Lothar Katz
Texas Instruments
8050 Freising, Fed. Rep. of Ger.
Deutschland GMBH
Haggertystr. 1
Ph: 49-816-80314

TriQuint Semiconductor

United States:

Marketing, Sunil Sanghavi
(408) 982-0900 x142, FAX (408) 982-0222
Western Sales, Mark Wu
(408) 982-0900 x113, FAX (408) 982-0222
Central Sales, John Watson
(214) 422-2532, FAX (214) 423-4947
East Sales, Mike Zyla
(215) 493-6944, FAX (215) 493-7418
International, Mike Kilgore
(503) 644-3535 x228, FAX (503) 644-3198

Europe - GiGA A/S

Fin Helmer, President
45-4-343-1588, FAX 45-4-343-5967

Japan - Japan Macnica Corp.

Shin Ishikawa, Product Manager
045-939-9140, FAX 045-939-6141

Vitesse Semiconductor

United States:

Corporate Headquarters
Vitesse Semiconductor Corporation
741 Calle Plano
Camarillo, CA 93012
Ph: 805-388-7501
FAX: 805-388-7565

Europe:

Thomson Composants Micronodes
Route Departementale 128 B.P. 48
91401 Orsay Cedex France
Ph: 33-1-60-19-7000
FAX: 33-1-60-19-7140

Japan:

H.Y. Associates Co., LTD.
3-1-10, Sekimachikita, Nerima-Ku
Tokyo, 177 Japan
Ph: 81-33-929-7111
FAX: 81-33-928-0301

Korea:

Beaver International, Inc.
3601 Deauville Court
Calabasas, CA 91302
Ph: 818-591-0356
FAX: 818-591-0753

Taiwan:

Tamarack Microelectronics
16 Fl., No. 1, Fu-Hsing N. Road
Taipei, Taiwan, ROC
Ph: 886-2-772-7400
FAX: 886-2-776-0545

intel®

AP-480

APPLICATION NOTE

Pentium™ PROCESSOR THERMAL DESIGN GUIDELINES REV. 2.0

DAN McCUTCHAN

January 1994

PRELIMINARY
Order Number: 241575-002

Pentium™ PROCESSOR THERMAL DESIGN GUIDELINES REV. 2.0

CONTENTS

CONTENTS

1.0 INTRODUCTION

In a system environment, the Pentium™ processor's temperature is a function of both the system and component thermal characteristics. The system level thermal constraints imposed on the package are local ambient temperature and thermal conductivity (i.e., airflow over the device). The Pentium processor thermal characteristics depend on the package (size and material), the type of interconnection to the printed circuit board (PCB), the presence of a heat sink, and the thermal conductivity and the power density of the PCB.

All of these parameters are aggravated by the continued push of technology to increase the operating speeds and the packaging density. As operating frequencies increase and packaging size decreases the power density increases and the heat sink size and airflow become more constrained. The result is an increased importance on system design to ensure that thermal design requirements are met for each component in the system.

In addition to heat sinks and fans, there are other solutions for cooling integrated circuit devices. A few of these solutions are: fan mounted on heat sink, heat pipes, thermoelectric (peltier) cooling, liquid cooling, etc. While these alternatives are capable of dissipating additional heat, they have disadvantages in terms of system cost, complexity, reliability, and efficiency. These techniques are more expensive than a passive heat sink and fan. The introduction of active devices can also decrease reliability. Finally, the power efficiency of some of these techniques is poor, and gets worse as the amount of power being dissipated increases. Despite these disadvantages, each of these solutions may be the right one for particular system implementations.

However, for the purpose of this application note, Intel has focused its efforts on describing solutions using passive heat sinks and fans.

1.1 Document Goal

The goal of this document is to provide thermal performance information for the Pentium processor and recommendations for meeting the thermal requirements imposed on systems. This application note attempts to provide an understanding of the thermal characteristics of the Pentium processor and some examples of how the thermal requirements can be met.

2.0 IMPORTANCE OF THERMAL MANAGEMENT

Thermal management of an electronic system encompasses all of the thermal processes and technologies that must be employed to remove and transfer heat from individual components to the system's thermal sink in a controlled manner.

The objective of thermal management is to ensure that the temperature of all components is maintained within functional and absolute maximum limits. The functional temperature limit is the range within which the electrical circuits can be expected to meet their specified performance requirements. Operation outside the functional limit can degrade system performance or cause logic errors. The absolute maximum temperature limit is the highest temperature that a portion of the component may be safely exposed. Temperatures exceeding the limit can cause physical destruction or may result in irreversible changes in operating characteristics. Higher temperatures result in earlier failure of the devices in the system. Every 10°C rise above the operating range means a halving of the mean time between failures.

3.0 Pentium™ PROCESSOR POWER SPECIFICATIONS

The Pentium processor's power dissipation and case temperature specs for 60 MHz and 66 MHz are shown in Table 1.

To ensure functionality and reliability of the Pentium processor, maximum device junction temperature must remain below 90°C. Considering the power dissipation levels and typical ambient environments of 40°C to 45°C, the Pentium processor's junction temperatures cannot be maintained below 90°C without additional thermal enhancement to dissipate the heat generated by this level of power consumption.

The thermal characterization data described in Table 2 illustrates that both a heat sink and airflow are needed. The size of heat sink and the amount of airflow are interrelated and can be traded off against each other. For example, an increase in heat sink size decreases the amount of airflow required. In a typical system, heat sink size is limited by board layout, spacing, and component placement. Airflow is limited by the size and number of fans along with their placement in relation to the components and the airflow channels. In addition, acoustic noise constraints may limit the size or types of fans limiting the airflow.

To develop a reliable thermal solution, all of the above variables must be considered. Thermal characterization and simulation should be carried out at the entire system level accounting for the thermal requirements of each component.

4.0 THERMAL PARAMETERS

Component power dissipation results in a rise in temperature relative to the temperature of a reference point. The amount of rise in temperature depends on the net thermal resistance between the junction and the reference point. Thermal resistance is the key factor in determining the power handling capability of any electronic package.

Thermal resistance from junction to case (θ_{JC}), and from junction to ambient (θ_{JA}) are the two most often specified thermal parameters for integrated circuit packages.

4.1 Ambient Temperature

Ambient temperature is the temperature of the undistributed ambient air surrounding the package. Denoted T_A, ambient temperature is usually measured at a specified distance away from the package. In the laboratory test environment, ambient temperature is measured 12 inches upstream from the package under investigation. In a system environment, ambient temperature is the temperature of the air upstream to the package and in its close vicinity.

Table 1. Pentium™ Processor Power Dissipation

	Package Type	Total Pins	Pin Array	Package Size	Power (Typical)	Power (Max)	Max Case Temp (°C)
Pentium Processor 60 MHz	PGA	273	21 x 21	2.16" x 2.16"	11.9W	14.6W	80
Pentium Processor 66 MHZ	PGA	273	21 x 21	2.16" x 2.16"	13W	16W	70

4.2 Case Temperature

Case temperature, denoted T_C, is measured at the center of the top surface (on top of the heat spreader, see Figure 1) of the package, typically the hottest point on the package case. Special care is required when measuring the case temperature to ensure an accurate temperature measurement. Thermocouples are often used to measure T_C. Before any temperature measurements, the thermocouples have to be calibrated. When measuring the temperature of a surface which is at a different temperature from the surrounding ambient air, errors could be introduced in the measurements. The measurement errors could be due to having a poor thermal contact between the thermocouple junction and the surface, heat loss by radiation or by conduction through thermocouple leads. To minimize the measurement errors, it is recommended to use the following approach:

- Use 36 gauge or finer diameter K, T, or J type thermocouples. The laboratory testing was done using a thermocouple made by Omega (part number: 5TC-TTK-36-36).
- Attach the thermocouple bead or junction to the center of the package top surface using high thermal conductivity cements. The laboratory testing was done by using Omega Bond (part number: OB-100).
- The thermocouple should be attached at a 90° angle as shown in Figure 1. When a heat sink is attached a hole (no larger than 0.15") should be drilled through the heat sink to allow probing the center of the package as shown in Figure 1.
- If the case temperature is measured with a heat sink attached to the package, drill a hole through the heat sink to route the thermocouple wire out.

4.3 Junction Temperature

Junction temperature, denoted T_J, is the average temperature of the die within the package.

The junction temperature for a given junction-to-ambient thermal resistance, power dissipation, and ambient temperature is given by the following formula:

$$T_J = P_D * \theta_{JA} + T_A$$

If a heat sink with thermal resistance of θ_{SA} (sink-to-ambient) is used, then the thermal resistance from the junction-to-case, θ_{JC}, is given by the following formula:

$$T_J = P_D * (\theta_{JC} + \theta_{CS} + \theta_{SA}) + T_A$$

where:

θ_{CS} is the thermal resistance from the component (case) to the heat sink.

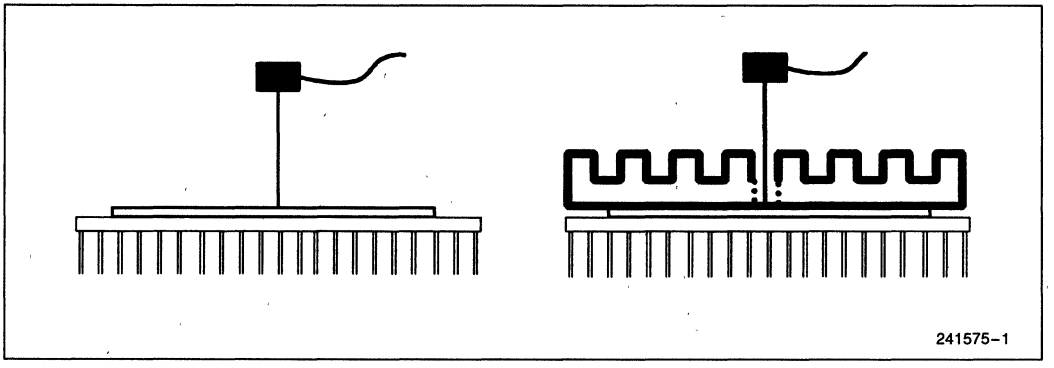


Figure 1. Thermocouple Attachment

4.4 Thermal Resistance

Thermal resistance (Figure 2) values for junction-to-ambient, θ_{JA} and junction-to-case, θ_{JC}, are used as measures of IC package thermal performance. θ_{JC} is a measure of the package's internal thermal resistance along the major heat flow path from silicon die to package exterior. This value is strongly dependent on the material, thermal conductivity, and geometry of the package. θ_{JC} values also depend on the location of the reference point (in this case center of the package top surface), the external cooling configurations and the heat flow paths from the package to the ambient. For example, if a heat sink is attached to the package top surface or more heat is pulled into the board through the pins, the θ_{JC} values measured with reference to the center of the package top surface will change. θ_{JA} values include not only internal thermal resistance, but also the radiative and convective thermal resistance from the package exterior to ambient air. θ_{JA} values depend on the material, thermal conductivity, and geometry of the package and also on ambient conditions such as airflow rates and coolant physical properties.

In order to obtain thermal resistance values, junction temperature is measured using the temperature sensitive parameter (TSP) method. With this method, special design thermal test structures are used which are approximately the same size as the Pentium processor die. The test structure consists of resistors and diodes. Resistors are used to simulate the Pentium processor power dissipation and thereby heat up the package. Diodes, which are located at the center of the thermal test die, are used to measure the die temperature. The measurements are carried out in a wind tunnel environment. The air flow rate and the ambient temperature are measured 12 inches away from the package in the upstream air.

The parameters are defined by the following relationships:

$$\theta_{JA} = (T_J - T_A)/P_D$$
$$\theta_{JC} = (T_J - T_C)/P_D$$
$$\theta_{JA} = \theta_{JC} + \theta_{CA}$$

where:

θ_{JA} = junction-to-ambient thermal resistance (°C/W)
θ_{JC} = junction-to-case thermal resistance (°C/W)
θ_{CA} = case-to-ambient thermal resistance (°C/W)
T_J = average die (junction) temperature (°C)
T_C = case temperature at a pre-defined location (°C)
T_A = ambient temperature (°C)
P_D = device power dissipation (W)

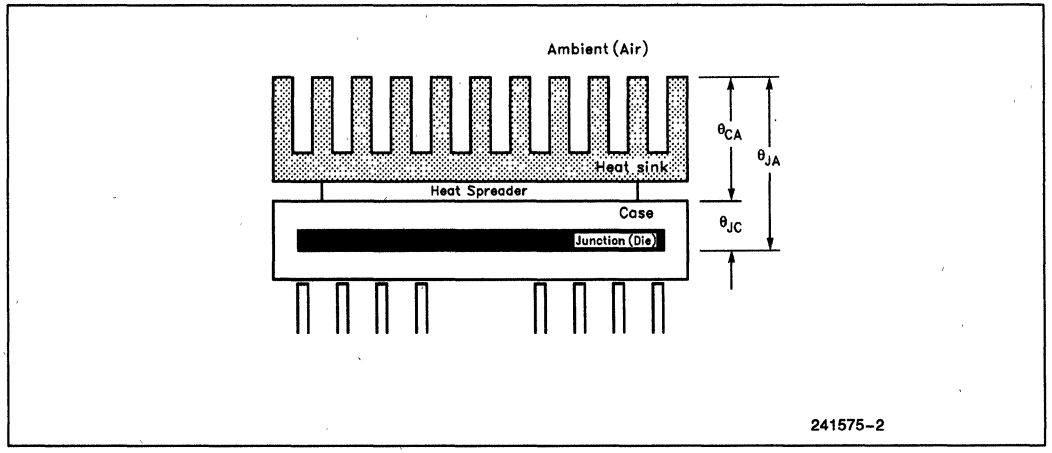


Figure 2. Thermal Resistance Parameters

Table 2 lists the junction-to-case and case-to-ambient thermal resistances for the Pentium processor (with and without a heat sink).

Table 2. Thermal Characterization Data

	θ_{JC}	θ_{CA} **vs Airflow (ft/min.)					
		0	200	400	600	800	1000
With 0.25" Heat Sink	0.6	8.3	5.4	3.5	2.6	2.1	1.8
With 0.35" Heat Sink	0.6	7.4	4.5	3.0	2.2	1.8	1.6
With 0.65" Heat Sink	0.6	5.9	3.0	1.9	1.5	1.2	1.1
Without Heat Sink	1.2	10.5	7.9	5.5	3.8	2.8	2.4

NOTE:
Heat Sink: 2.1 in^2 base, omni-directional pin Al heat sink with 0.050 in. pin width, 0.143 in. pin-to-pin center spacing and 0.150 in. base thickness. Heat sinks are attached to the package with a 2 to 4 mil thick layer of typical thermal grease. The thermal conductivity of this grease is about 1.2 w/m c.
** θ_{CA} values shown in this table are typical values. The actual θ_{CA} values depend on the air flow in the system (which is typically unsteady, non-uniform and turbulent) and thermal interactions between Pentium CPU and surrounding components through PCB and the ambient.

2

5.0 DESIGNING FOR THERMAL PERFORMANCE

At this point the application note turns from describing the characteristics that define thermal performance to describing how designers should use these characteristics to assess thermal requirements of PC system designs. The Pentium processor specifies a maximum case temperature, T_C, of 70°C @ 66 MHz. This case temperature limit along with the Pentium processor's power and thermal resistance characteristics can be used to determine the ambient temperature required to keep the Pentium processor operating within its specified limits. Using these parameters in the following equations:

$$T_A = T_C - (P^* \theta_{CA})$$

$$T_A = 70°C - (16W^* 10.5°C/W)$$

$$T_A = -98°C$$

The maximum ambient temperature required in a Pentium processor system without any additional thermal enhancement is −98°C at 66 MHz. Obviously, this ambient temperature is impractical and unachievable in a PC system. In order to be able to maintain the case temperature at 70°C in a typical system ambient with air temperature of 40°C to 45°C, the thermal resistance between the case and the ambient must be reduced.

5.1 Heat Sinks

The most common way to improve the package thermal performance is to increase the surface area of the device by attaching a large piece of metal (a heat sink) to the package. The heat sink is usually made of Aluminum and is chosen for its price/thermal-performance ratio. There are materials that offer higher conductivity such as copper, but cost becomes prohibitive. To maximize the flow of heat for a given junction temperature rise over the ambient temperature, the thermal resistance from heat sink to air can be reduced by a) maximizing the surface area, and b) maximizing the air flow across the surface area (maximizing air flow **through** heat sink fins in most cases).

Intel has used test data to determine what size of heat sink and airflow is needed to properly cool a Pentium processor system. The data was derived assuming an adhesive attach process that offers thermal resistance of about 0.2°C/W.

The testing was done in a wind tuunel in the configuration (in Figure 3) where the heat sink was mounted on a real Pentium processor package with a thermal die mounted inside to generate the 16W of power. The package is then mounted in a socket which is soldered to a 2-layer PCB that brings power to the die.

Based on these tests, three specific heat sink and airflow combinations have been identified that properly dissipate the Pentium processor's 16W and maintains a case temperature below 70°C @ 66 MHz. The three heat sinks are shown in Figure 4.

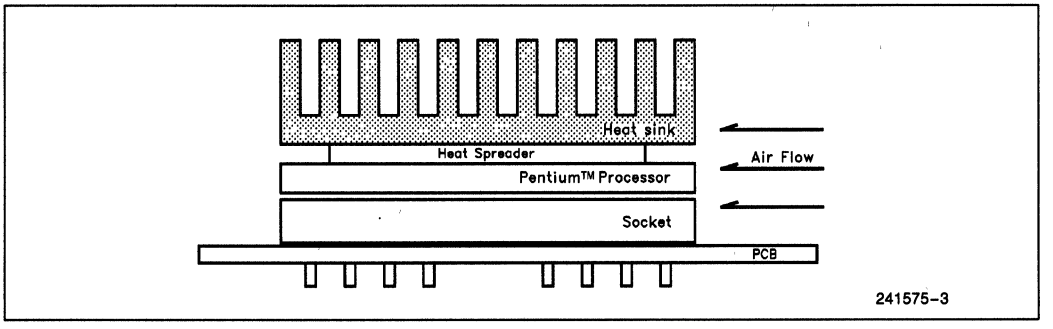


Figure 3. Improving Thermal Performance

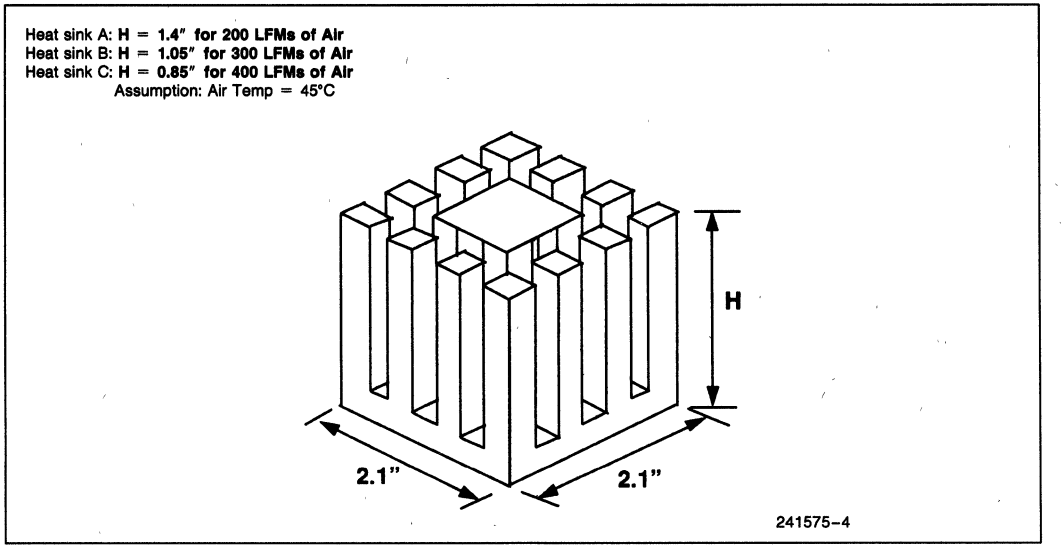


Figure 4. Recommended Combinations

In addition, testing has been done to provide more general guidelines which allow deviating from the above conditions. These guidelines allow systems to derive various combinations of heat sink size and airflow that ensure the Pentium processor thermal specifications are met. For example, by increasing the heat sink x-y dimensions and extending it over the package footprint, the heat sink height can be reduced while maintaining the same thermal performance as the taller heat sink with the same footprint as that of the package. The first three charts (Figures 5, 6, 7) show the thermal resistance as a function of heat sink size and airflow. The last three charts (Figures 8, 9, 10) show the power dissipation achievable with a given heat sink size and airflow. The power dissipation calculations assume T_C = 70°C @ 66 MHz, T_A = 45°C, and θ_{JC} = 0.6°C/W.

$$P_{max} = (T_C - T_A)/\theta_{CA} = 25/\theta_{CA}$$

A key assumption in all of these calculations is that a perfect thermal connection can be achieved between the case and the heat sink. One ban extrapolate the heat sink solutions by adding the additional thermal resistance of any chosen heat sink attach process. See Appendix B for case to heat sink thermal interface options.

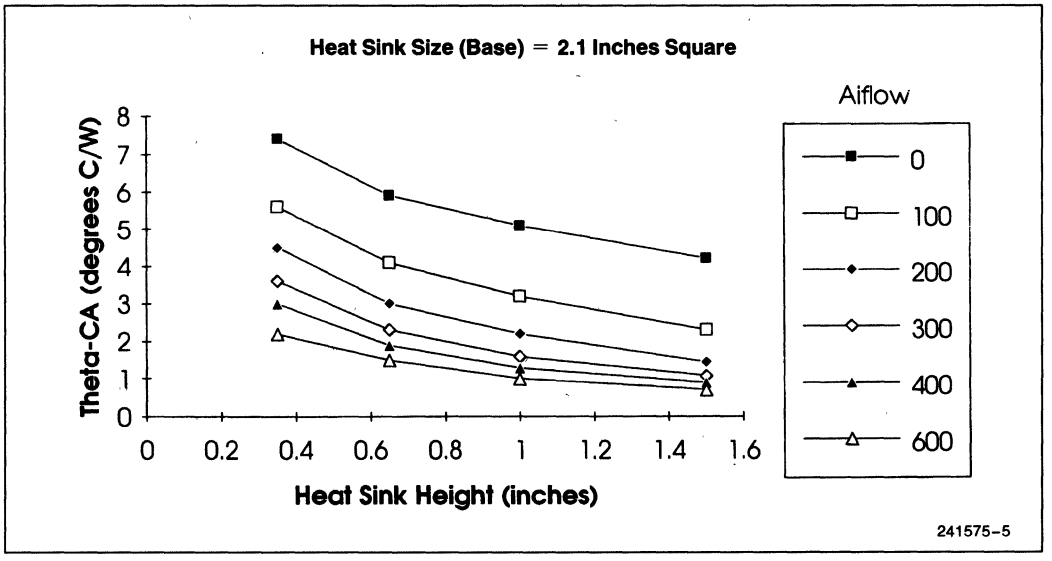


Figure 5. Thermal Resistance

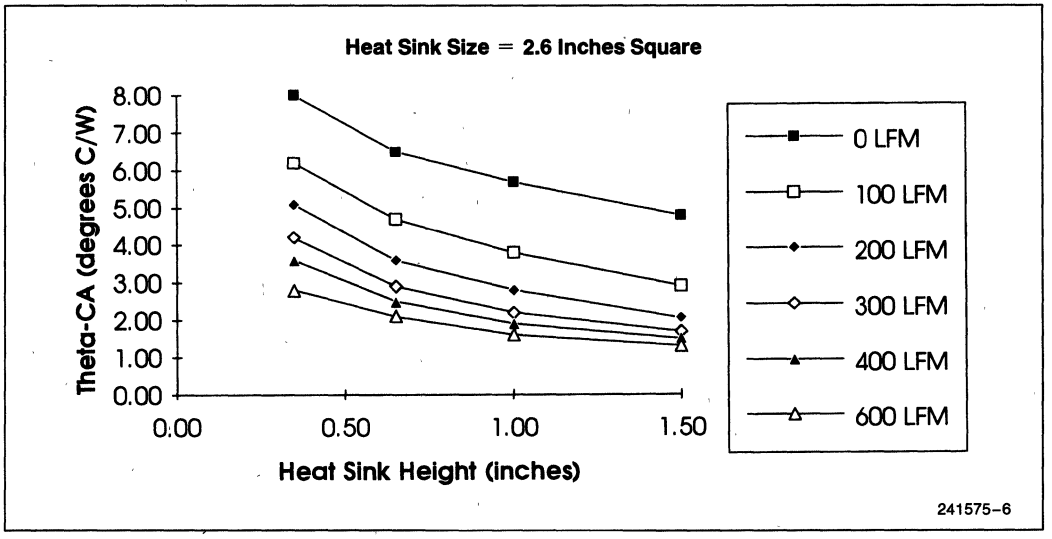


Figure 6. Thermal Resistance

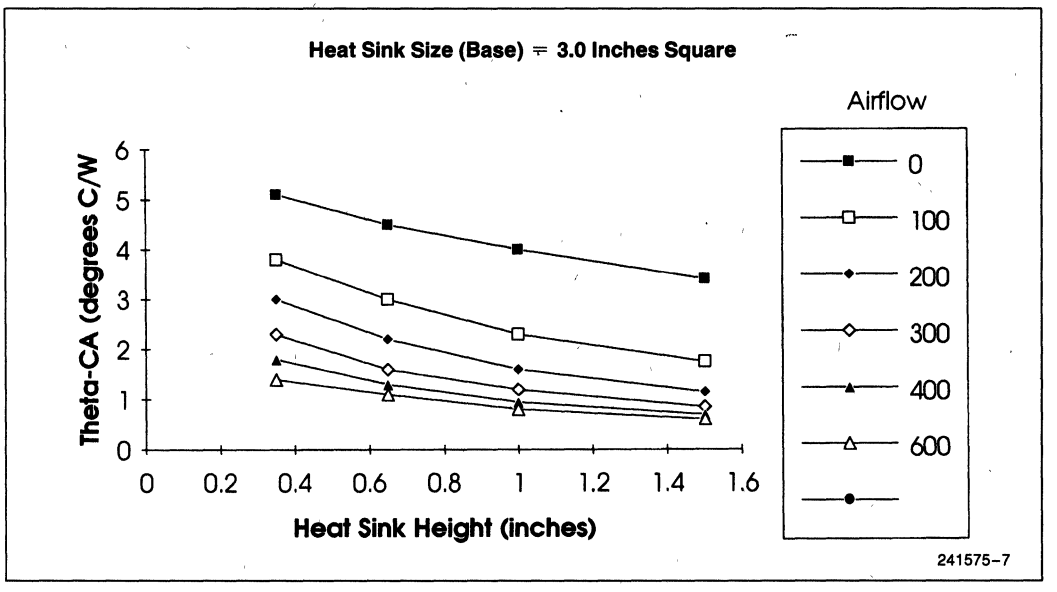


Figure 7. Thermal Resistance

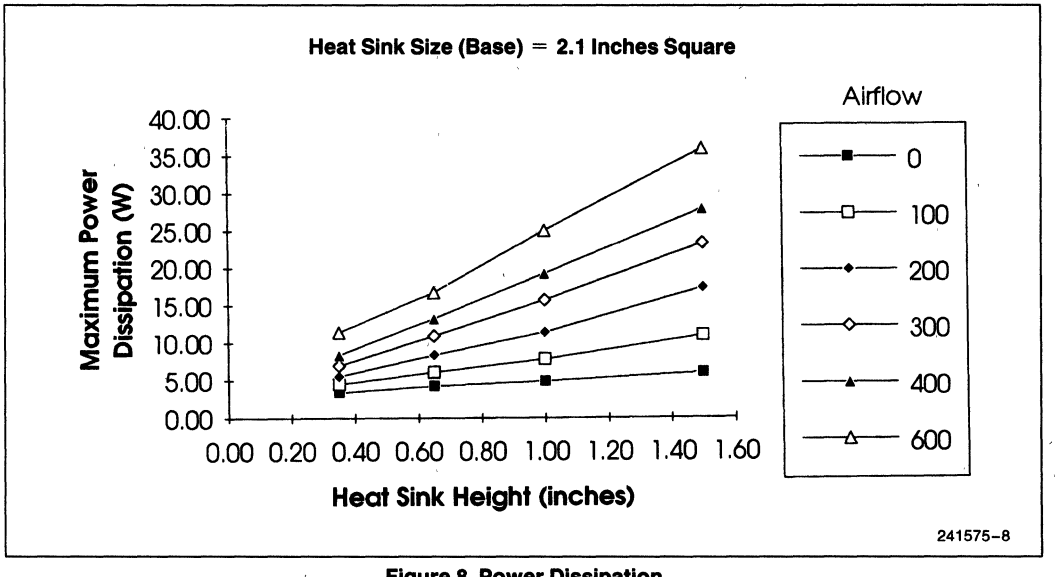


Figure 8. Power Dissipation

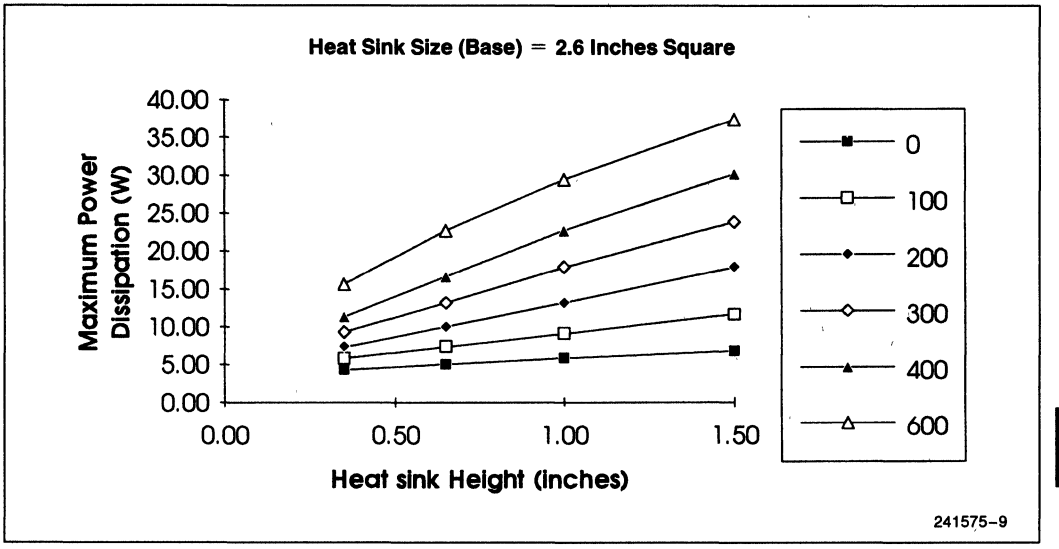


Figure 9. Power Dissipation

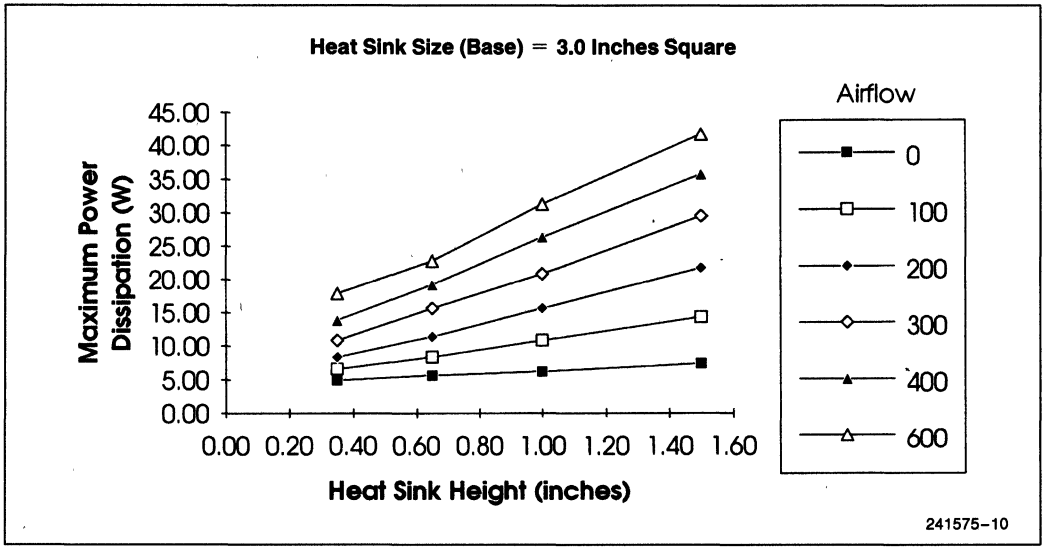


Figure 10. Power Dissipation

5.2 Airflow

To improve the effectiveness of heat sinks it is important to manage the airflow so as to maximize the amount of air that flows over the device or heat sink's surface area. In the system, the air flow around the processor can be increased by providing an additional fan or increasing the output of existing fan. If this is not possible, baffling the airflow to direct it across the device may help. This means the addition of sheet metal or objects to guide the air to the target device. Often the addition of simple baffles can eliminate the need for an extra fan. In addition, the order in which air passes over devices can impact the amount of heat dissipated.

5.3 Fans

Fans are often needed to assist in moving the air inside a chassis. A typical variable speed fan capable of up to 100 CFM of air can be found for approximately $10.

The airflow rate is usually directly related to the acoustic noise level of the fan and system. Therefore maximum acceptable noise levels may limit the fan output or the number of fans selected for a system.

A fan may be placed at the top of a heat sink to produce direct air impingement on the heat sink for efficient heat removal. A key issue with fans is their reliability. Although many fans are rated for approximately 50,000 hours of operation, operating conditions such as operating temperature, pressure drop across the fan and the particles in the air can significantly reduce the fan's useful life.

5.4 Thermal Performance Validation

The recommended thermal solutions in Section 5.1–5.3 are only design guidelines. Since there are many variables that can effect thermal performance in an actual system, thermal performance should always be validated experimentally, following the case temperature measurement technique described in Section 4.2.

6.0 CONCLUSION

As the complexity of today's microprocessors continues to increase so do the power dissipation requirements. Care must be taken to ensure the additional heat resulting from the power is properly dissipated. As documented, the heat can be dissipated using passive heat sinks, fans and/or active cooling devices.

The simplest and probably most cost effective method is to use a heat sink and a fan. The size of the heat sink and the output of the fan can be varied to balance the tradeoffs between size and space constraints versus noise. For example, if space is available a 1.4″ high heat sink can be used with only 200 LFM to cool the 66 MHz Pentium processor and the 16W it dissipates. Another example in which space is restricted shows a 0.7″ high heat sink can be used if approximately 500 LFM of airflow is provided.

These are not the only valid solutions, but both provide adequate cooling to maintain the Pentium processor case temperature at or below the 70°C @ 66 MHz specified limit. By maintaining this specification, the system can guarantee proper functionality and reliability of the Pentium processor.

APPENDIX A
EXAMPLES

Thermal management examples, designing with the Pentium processor. Using the best known methods.

Appendix Goal

The goal of this appendix is to measure the operating temperatures in a real system versus the wind tunnel laboratory measurements. These experiments are done with heat sinks that are similar to the ones suggested in Section 5.1 of the main document. The thermocouples and attachment methods suggested in Section 4.2 of the main document are also used. The appendix begins by reviewing the variables that the system designer has control over and uses tables to describe thermal resistance in the context of where the system designer can have the most effect The importance of the case to heat sink thermal interface and correct attachment methods are reviewed and different options given. The appendix proceeds to describe the system used for these tests and the tools and equipment needed. The lab set up procedures are discussed in detail and the measurements are presented with comments at the conclusion.

WHAT ARE THE VARIABLES?

Table A-1 shows the cooling options that customers can control when designing a system. From Table A-1 it is obvious that changing the heat sink and air flow are the two most effective ways for a system designer to affect the thermal performance of a system.

Table A-1. Variables

COOLING OPTIONS UNDER CUSTOMER CONTROL	
Variables	**Options for Cooling**
Device	• Use Power Management SW in the System • Clock the Device at a Lower Speed
Heat Sink	• Increase Heat Sink Area, Width or Height • Use Interface Materials with Lower Thermal Resistance • Use Active Cooling Devices • Use a Combination of Active and Passive Cooling Devices
Air Flow	• Increase Air Flow • Manage Air Flow • Place Hottest IC's near Highest Airflow
Ambient Temperature	• Place System in a Controlled Climate

Figure A-1 sums up the thermal tradeoff picture succinctly. Looking at Figure A-1 reiterates the previous statement that increasing the heat sink size and air flow rate provide the largest thermal performance improvements. In addition it shows the variables that are constant. Note that the θ_{JC} (junction-to-case thermal resistance) of the Pentium processor is fixed and a system designer can have no effect on this parameter. Also note that the θ_{CS} (case-to-heat sink thermal resistance) is a constant. Even though θ_{CS} is shown as a constant in Figure A-1 it can move up and down the Y axis depending on the interface material chosen. The case to heat sink interface is critical to the overall success of the thermal solution and cannot be overlooked. The next section will go into detail on this subject.

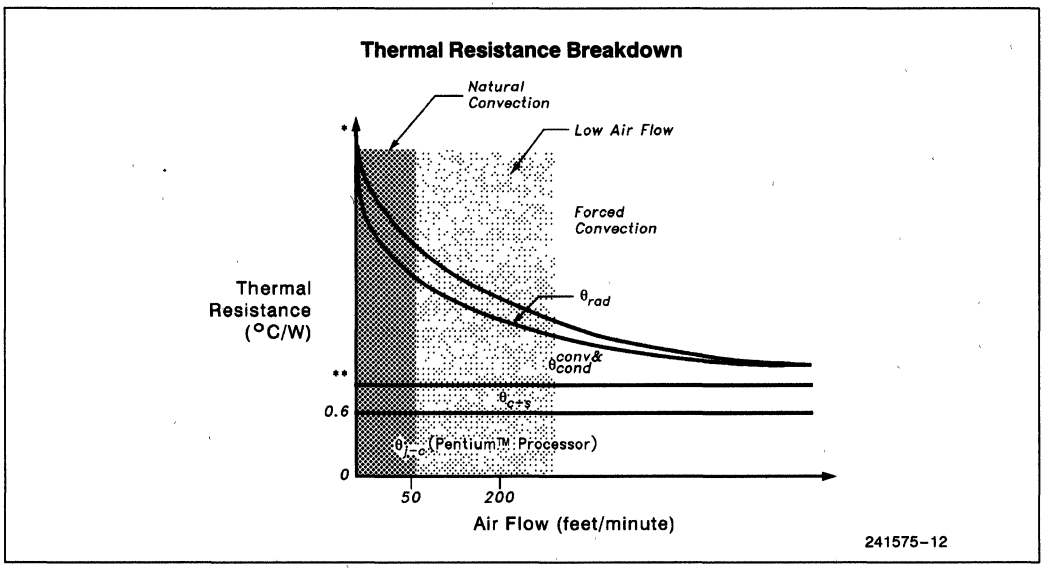


Figure A-1. Thermal Resistance

The main of purpose Figure A-2 is to show that packages and heat sinks are not perfectly flat and this requires that the air gap be filled with an interface material that has a lower thermal resistance than air. The whole point is to try and minimize the contact thermal resistance. The different types of thermal interface materials are listed to show the wide array of materials available to the system designer. Intel's data books have a mechanical section that list the flatness of the package. Heat sink vendors should be able to provide specifications for their heat sink offerings.

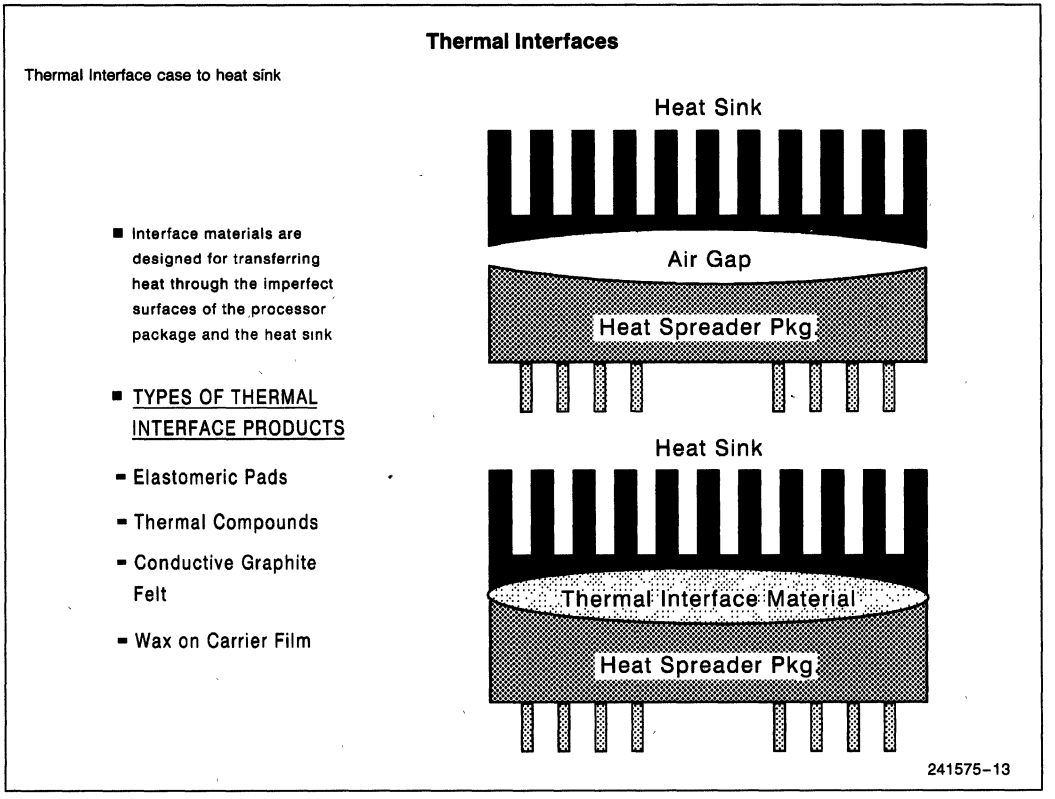


Figure A-2. Thermal Interfaces

The next section (Figures A-3 through A-5) covers attachment methods which generally fall into the categories shown; epoxies, double sided tapes or manual clips to either chip or socket.

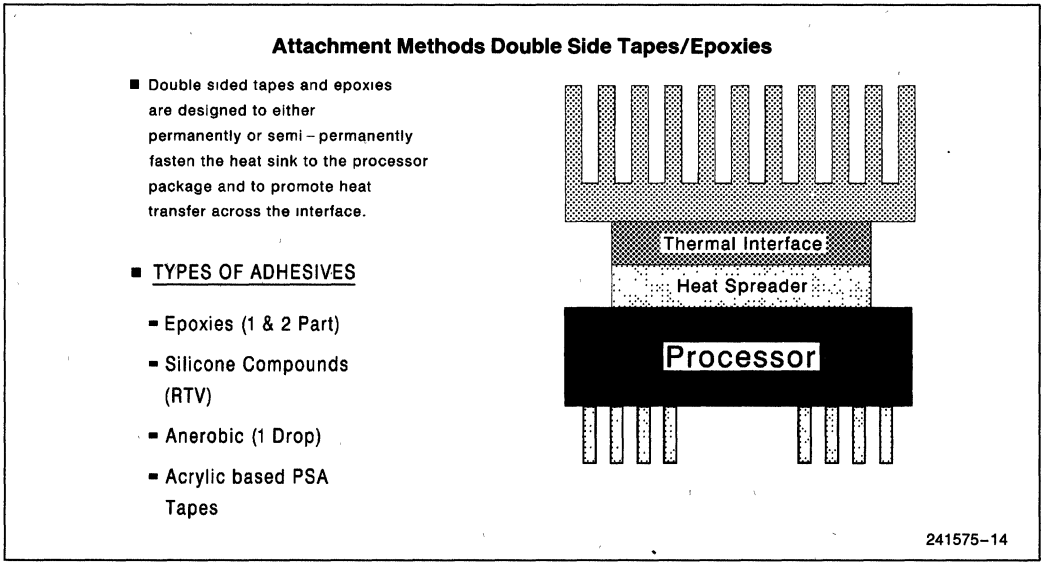


Figure A-3. Attachment Methods

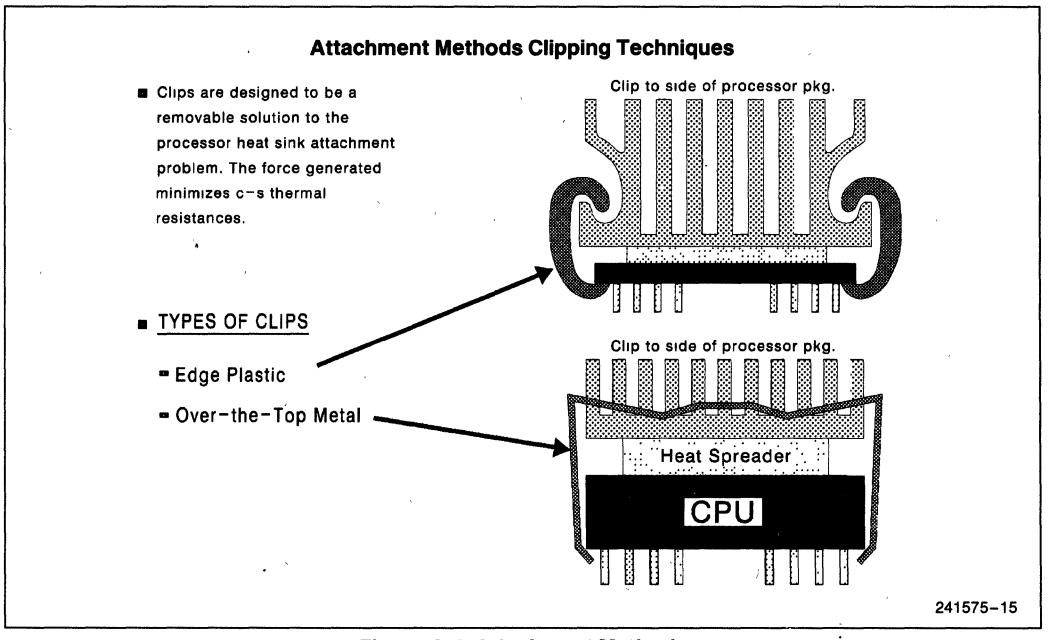


Figure A-4. Attachment Methods

Note that some clips don't allow the package to be pushed all the way into the socket and this could be a problem with short lead packages. The main advantage of this type of system is that a low profile socket can be used to lower the height of the processor heat sink assembly.

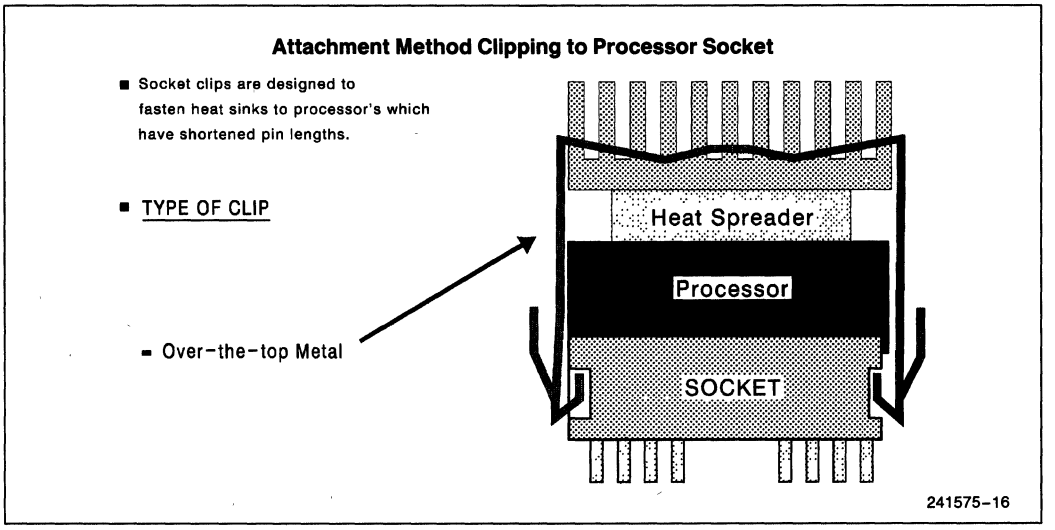


Figure A-5. Attachment Methods

Table A-2 lists pros and cons of the different attachment methods covered.

Table A-2. Attachment Methods

ATTACHMENT METHODS	ADVANTAGES	DISADVANTAGES
DOUBLE SIDED TAPES	• Quick to Use • Low Installed Cost • Compliant	• High Thermal Resistance • Requires Flat Interfaces • Assembly Contact Pressure
EPOXIES	• Low Potential Thermal Resistance • Low Contact Pressure	• Mixing, Curing, Messy • Timing Consuming (if not automated) • CTE Stress, High Rigidity • Variable Thickness (theta)
CLIPS	• Centralized Pressure Points • Removable • Easily Installed • Solution to Upgrade • Accommodates Wide Tolerance	• Removable • Force Limits vs Assembly • Insufficient Shock and Vibration Data • Potential for Loss of Pressure

The materials chart Figure A-6 shows the performance of each type of thermal interface material. Note that even though thermal grease has a deserved reputation for being messy and harder to control it still performs well as a thermal interface. All the examples that are shown in Appendix A use thermal grease as the case to heat sink interface.

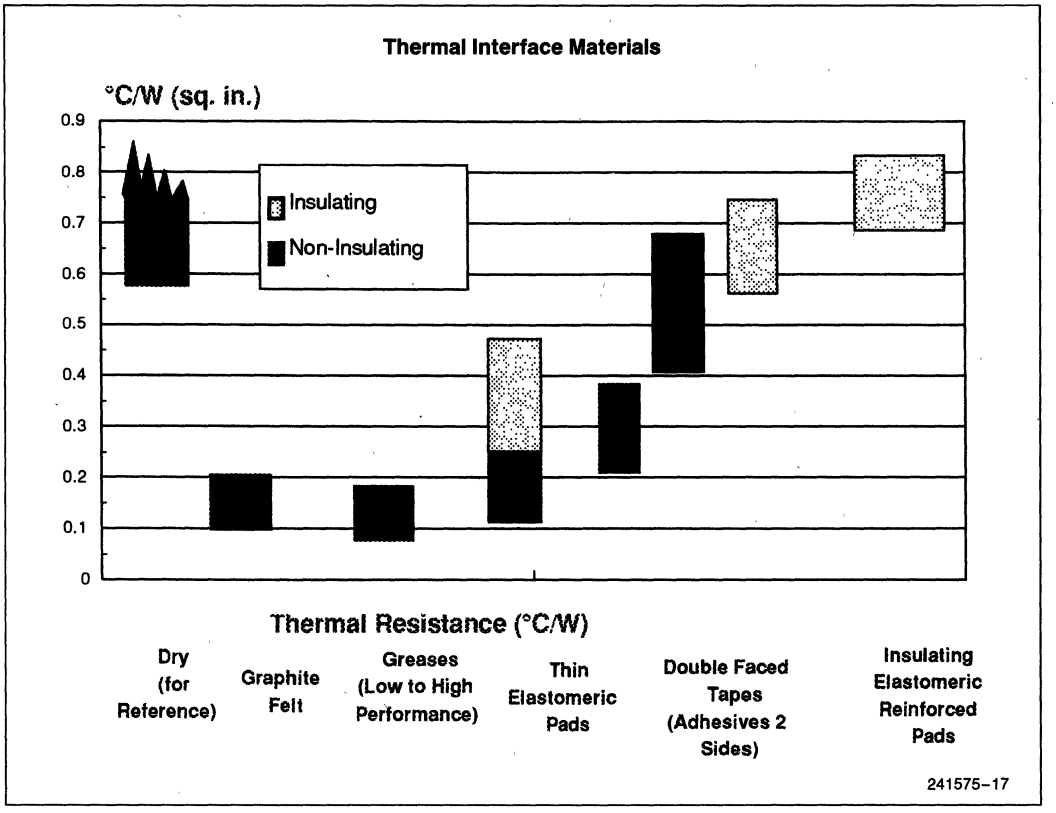


Figure A-6. Materials

The next step is to chose a heat sink. Figure A-7 shows the wide range of choices and the cost associated with each technology.

Now that all the variables and options are known for this problem we can proceed on to do some real system measurements using the recommendations and data shown in the first part of this application note.

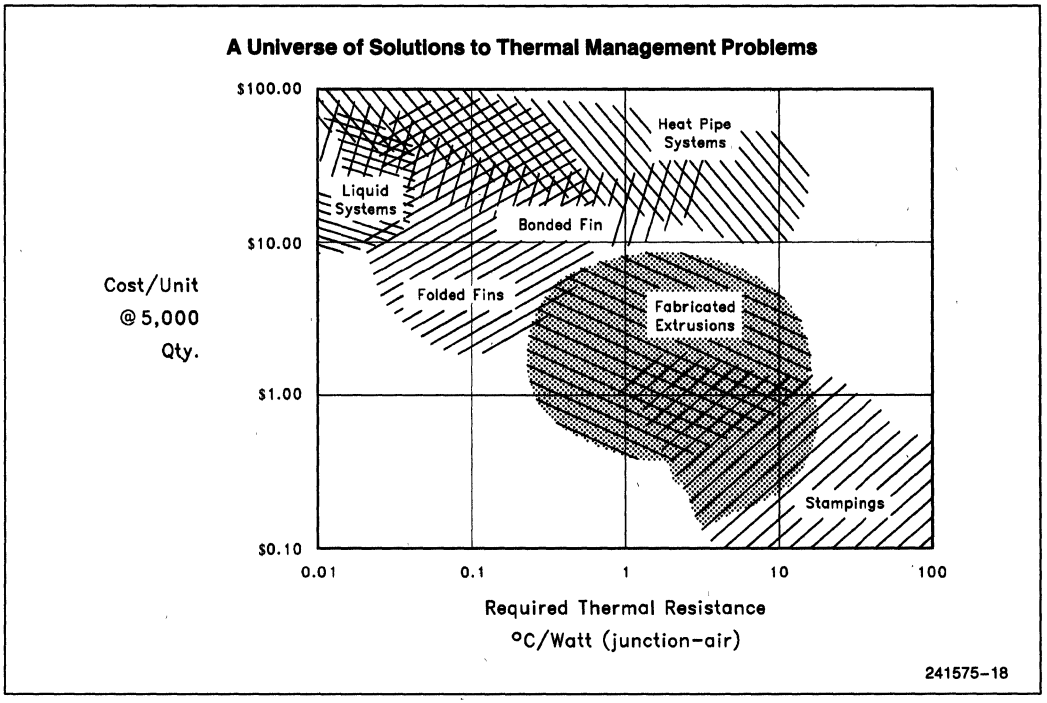


Figure A-7. Solutions

Examples

For all the examples in this section we used a 40 MHz system with a Pentium processor and 256K cache. A picture of the system under test is shown in Figure A-8 with the covers off to show the placement of the Pentium processor and the associated cache components. A 40 MHz system was used because it was the only one available at the time the testing was done.

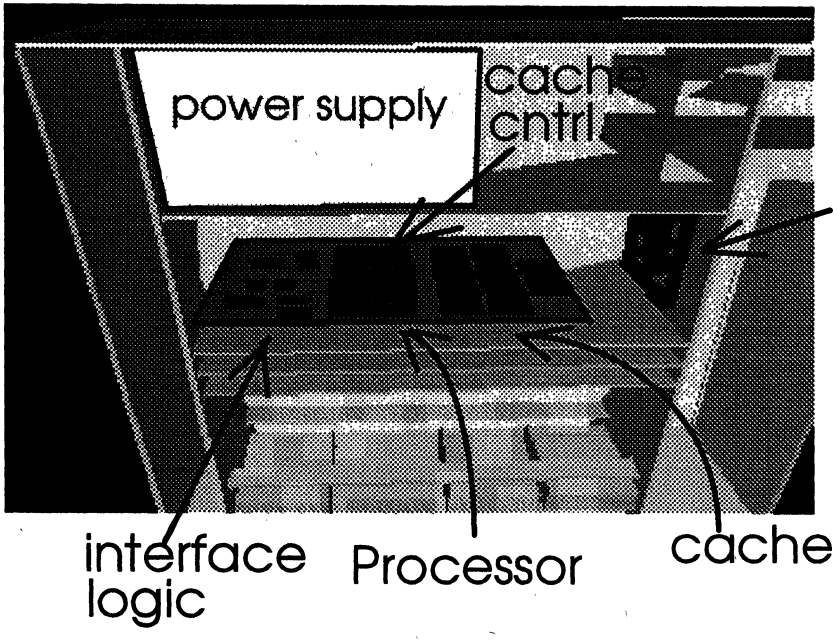


Figure A-8. Pentium™ Processor System

Objectives

- To measure a Pentium processor system operating under real working conditions.
- To compare the measured results to the predicted results shown in the beginning of this application note. The reader should always keep the main goal in mind; **the main goal is always to meet the case temperature specification for the Pentium processor.** Any combination of heat sink and air flow rate is fine as long as the case temperature specification is met. The heat sinks used in test #1 thru #4 will match the suggested heat sinks as close as possible to accurately correlate with the wind tunnel data. This is meant to illustrate how a system designer might start by using the suggested heat sinks and air flow rates as starting points to thermally tune their particular system. Test #5 uses a heat sink and a fan combination. The fan heat sink is best described as a fan attached directly to the heat sink on the Pentium processor. It is an active device used for spot cooling ICs. We will concentrate on traditional passive heat sink solutions with only one set of measurements being done for a fan heat sink assembly.

Tools and Equipment

1. Pentium processor-based system running at 40 MHz.
2. Hot wire anemometer to measure airflow rate.
3. Thermocouples and high thermal conductivity cement as recommended in the application note.
4. Homemade jig for accurate and repeatable attachment of the thermocouples to the package.
5. Homemade power supply isolation socket for setting the V_{CC} and reading the I_{CC} of the processor independently of the rest of the system.
6. Adjustable power supply with adequate current capabilities and both current and voltage read out.
7. Multimeter to read the voltage and current.
8. Cables to connect everything up.
9. Software test suite that simulates "worst case conditions for a typical real application." In this case it was Microsoft Excel and Word for Windows test suites.
10. Drill and drill bits.
11. Thermal grease.

The lab procedure was as follows:

Preparing the System

1. Load the test software on the system disk (or floppy) and make sure everything runs correctly before you start. After everything works satisfactorily proceed to the next step.
2. Remove the covers, choose several places (random) around the processor to measure the air flow of the system. Then drill holes large enough to allow the anemometer to be inserted. Five holes were drilled in the system cover.
3. In this case we had a 12" long ¼" diameter directional anemometer. To get more repeatable measurements the shaft of the probe was marked with a pencil to get the same depth, into the box, for each measurement
4. We then removed the processor card from the chassis (use anti-static procedures to prevent IC damage).
5. Remove the Pentium processor from the card and install the isolation socket.

Preparing the Pentium Processor for Testing

1. Using the jig carefully attach the thermocouple to the center of the processor package using cement and let it cure as recommended by the manufacturer of the cement.
2. Drill holes no larger than 0.125" in the centers of the heat sinks to be tested just large enough to get the thermocouple wires through the hole. In the case of the fan heat sink, the fan was removed and the heat sink was drilled the same as the others and then re-assembled. Each of the holes were counter sunk on the bottom to better conform to the tear drop shape the thermocouple and cement naturally forms into. The idea is to not disturb or break the contact between the cement and the package. If it is broken or cracked the measurements will be incorrect
3. Apply the thermal grease (less than 0.004" thick) evenly, with no voids, to the processor package.
4. Slide the heat sink down the thermocouple wires being careful not to disturb the thermocouple while at the same time firmly seating the heat sink to the package. Attach the plug for the temperature meter to the other end of the thermocouple wire terminals.
5. Re-install the processor/thermocouple/heat sink assembly into the isolation socket on the processor board, again being careful not to disturb the thermocouple connection.

Preparing for Measurements

1. Re-install the processor card into the system.
2. Connect the power supply wires to the power supply and the isolation socket.
3. Connect the multimeter to the the power supply to monitor the V_{CC} and set the power supply meter to measure I_{CC}.
4. Connect the thermocouple to the meter.
5. Turn on the processor power supply and then the system supply.
6. Wait for the system to boot and then run the test software.

Thermal Measurements

The next step was to determine the baseline airflow in the system without a heat sink attached to the processor. Measure the airflow at several locations using the access holes in the system and the marks on the probe to ensure accurate placement of the probe and repeatability of the measurements. Table A-3 shows the results. Be cautious when placing the fan in a system relative to the processor. All fans have a dead spot (low airflow) in the center of the fan. Avoid the dead spot. Even several inches away from the fan the dead spot can influence airflow considerably.

Test #1

The next step is to compare how close the suggested values and tables are to the measured results. Use the formulas described in the beginning of the application note and the values from Table A-4.

$P_D = V_{CC} * I_{CC} = (1.82*4.89) = 8.827W$

$\theta_{JC} = (T_J - T_C)/P_D = 0.6$

$\theta_{JA} = (T_J - T_A)/P_D$

$\theta_{CA} = \theta_{JA} - \theta_{JC} = [(T_J - T_A) - (T_J - T_C)]/P_D$
$= [T_C - T_A]/P_D$

$\theta_{CA} = (55.3 - 29)/(1.8*4.85) = 24/8.827 = 2.97$

θ_{CA} = 2.7°C/W is the measured value in the system for this configuration.

Table A-3. Baseline Airflow

Airflow Measured, LFM	120–160
Location of probe	~ 2 inches (upstream from the fan) @ center of the processor (above the heat sink)

Table A-4. Test Conditions Test #1

Heat Sink Size, Inches H x W	Temperature, degrees C			I_{CC} Amps	V_{CC} Volts	Air Flow LFM
	Room T_A	System T_A	Case T_C			
1.2 x 2.1 sq.	23	29	55.3	1.82	4.85	100–150

The graph (Figure A-9) from the application note, for heat sink size size of 2.1" x 2.1", is used to compare the predicted θ_{CA}, for a 1.2" tall heat sink, to the measured value of θ_{CA}.

The predictions from the graph (Figure A-9) are:

θ_{CA} = 2.9°C/W @ 100 LFM

θ_{CA} = 2.4°C/W @ 150 LFM

θ_{CA} = 1.9°C/W @ 200 LFM

And the measured value is:

θ_{CA} = 2.97°C/W @ 100–150 LFM

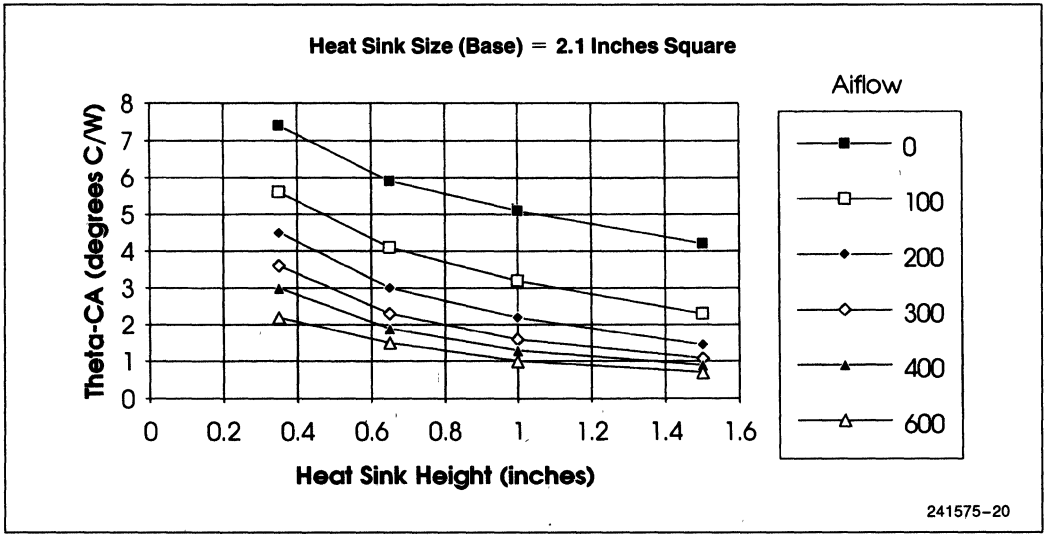


Figure A-9. Thermal Resistance

Table A-5. Test Conditions Test #2

Heat Sink Size, Inches H x W	Temperature, degrees C			I_{CC} Amps	V_{CC} Volts	Air Flow LFM
	Room T_A	System T_A	Case T_C			
0.5 x 2.1 sq.	22	29	58.3	1.81	4.85	100–150

Test #2

The same test only the heat sink height was reduced to 0.5 inch height.

$\theta_{CA} = (T_C - T_A)/P_D$

$\theta_{CA} = 58.3 - 29/8.79 = 3.33°C/W$

The 2. 1" x 2.1" graph (Figure A-9) from test #1 is used again and it predicts:

$\theta_{CA} = 4.9°C/W$ @ 100 LFM

$\theta_{CA} = 4.3°C/W$ @ 150 LFM

$\theta_{CA} = 3.8°C/W$ @ 200 LFM

And the measured value is:

$\theta_{CA} = 3.33°C/W$ @ 100–150 LFM

Test #3

This test is the same as test #2 except that processor board to board spacing was reduced to 0.6 inches using a cardboard baffle to simulate a system with very tight board spacing. An existing system that is upgrading from an Intel 486 processor to the Pentium processor might have this type of spacing. Note that this particular configuration actually has more airflow than test #2. It could have just as easily been lower. It all depends on the particular system being measured.

$\theta_{CA} = (T_C - T_A)/P_D$

$\theta_{CA} = 70.3 - 27/8.79 = 4.9°C/W$

The 2.1 " x 2.1" graph (Figure A-9) from test #1 is used again to predict the θ_{CA}:

$\theta_{CA} = 4.3°C/W$ @ 150 LFM

$\theta_{CA} = 3.8°C/W$ @ 200 LFM

$\theta_{CA} = 3.0°C/W$ @ 300 LFM

And the measured value is:

$\theta_{CA} = 4.9°C/W$ @ 175–200 LFM

Table A-6. Test Conditions Test #3

Heat Sink Size, Inches H x W	Temperature, degrees C			I_{CC} Amps	V_{CC} Volts	Air Flow LFM
	Room T_A	System T_A	Case T_C			
0.5 x 2.1 sq.	23	27	70.3	1.81	4.85	175–200

Table A-7. Test Conditions Test #4

Heat Sink Size, Inches H x W	Temperature, degrees C			I_{CC} Amps	V_{CC} Volts	Air Flow LFM
	Room T_A	System T_A	Case T_C			
0.65 x 3.1 sq.	23	29	55.3	1.8	4.85	100–140

Test #4

This test uses a 0.65″ tall heat sink that is 3.1″ sq. This type of heat sink might be used when height is limited and there is room to spread out by adding more area to the heat sink base.

$\theta_{CA} = (T_C - T_A)/P_D$

$\theta_{CA} = (55.3 - 29)/8.73 = 3.0$

The 3.0″ x 3.0″ graph (Figure A-10) from the application note is used since it is similar to the heat sink used. The 3.0″ x 3.0″ graph predicts:

θ_{CA} = 3.0°C/W @ 100 LFM

θ_{CA} = 2.6°C/W @ 150 LFM

And the measured value is:

θ_{CA} = 3.0°C/W @ 100–140 LFM

Test #5

The last test was done using a fan/heat sink assembly that has become popular for prototyping, debug and spot cooling in some situations. We were not able to measure the airflow on the processor with this configuration because the air flow is not directional enough to get a reading with the probe available. The case temperature however was monitored by mounting a thermocouple in the same manner used above. We did modify the setup by bringing the thermocouple wires out the side to clear the fan. This will change the measurements the thermocouple produces and should be factored into any data. We do not have any wind tunnel data on the fan/heat sink combination. Note that the case temperature is within specification.

2

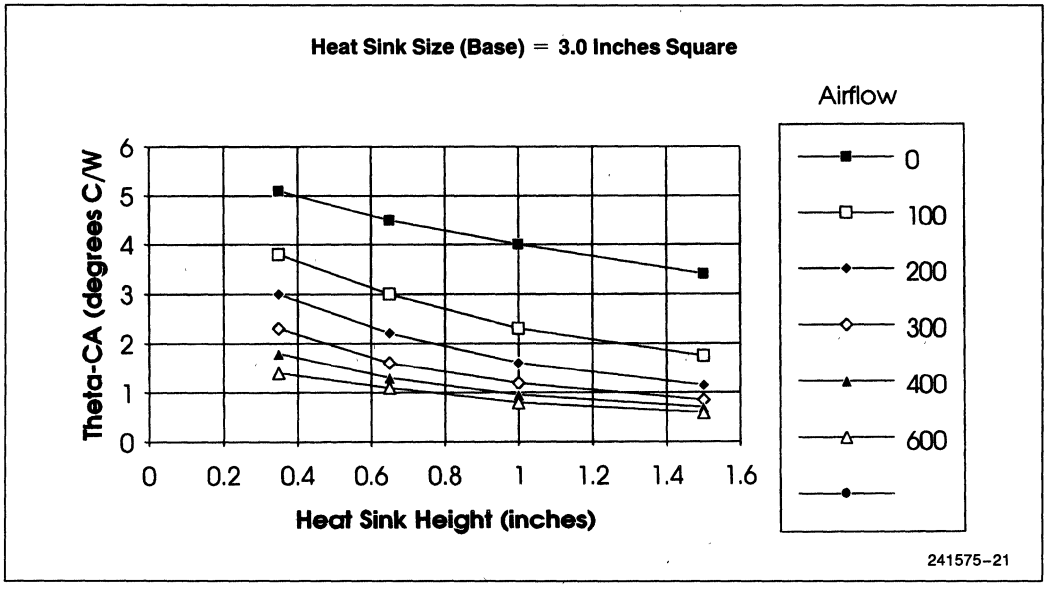


Figure A-10. Thermal Resistance

Table A-8. Test Conditions Test #5

Heat Sink Size, Inches H x W	Temperature, degrees C			I_{CC} Amps	V_{CC} Volts	Air Flow LFM
	Room T_A	System T_A	Case T_C			
HS/Fan	23	29	46	1.8	4.85	120–160

Conclusion

Table A-9 shows all the tests in one table. The data shows that the suggestions in the application note are a very good starting point to begin tuning any Pentium processor system and that there is no one cookbook answer that fits all systems because of the complexity of air flow and variations from each type of system. Indeed the results show that airflow can be changed dramatically even in the same system by changing one variable. For example test #2 and #3 are exactly the same except that board to board spacing was reduced significantly. Note that case temperature rose significantly even though the airflow sensor was reading a higher value. This suggests that the airflow through the heat sink was lower even though the anemometer, 2 inches away, was reading higher airflow at its position. Note also that test #2 more closely approximates the wind tunnel test setup because it has open space above the board instead of a board nearby. This is also why the predicted data versus the measured data is so far off for test #3, while test #2 is very close to the predicted results.

Test #1 and #4 demonstrate a fundamental principle of the physics involved. If you have the same airflow and must reduce the height of the heat sink, you have to spread out the area of the heat sink to compensate for the reduced height. Test #1 uses a 1.2" height heat sink that is the same size as the package. Test #4 was able to produce the same case temperature with a shorter heat sink and more area.

Test #5 demonstrates that a fan/heat sink assembly can spot cool effectively if you have enough space above and around it to allow the required back pressure. This is the only active device tested. If you look back at the "A Universe of Solutions to Thermal Management Problems" (Figure A-7) chart you will see the reason why. While the Pentium processor is at the outer envelope of passive cooling, this method of cooling still offers lower cost, power usage and reliability in most cases.

Most of all the system designer should **never lose sight of the real goal which is to keep the junction temperature within the operating limit**. Since the designer cannot measure junction temperature they must use the case temperature, which can be measured to ensure proper operation for the component.

Table A-9

Test #	Heat Sink Size, Inches H x W	Temperature, degrees C			I_{CC} Amps	Air Flow LFM
		Room T_A	System T_A	Case T_C		
1	1.2 x 2.1 sq.	23	29	55.3	1.82	100–150
2	0.5 x 2.1 sq.	22	29	58.3	1.81	100–150
3	0.5 x 2.1 sq.	23	27	70.3	1.81	175–200
4	0.65 x 3.1 sq.	23	29	55.3	1.8	100–140
5	HS/Fan	23	29	46	1.8	120–160

V_{CC} = 4.85V for all tests.

APPENDIX B
HEAT SINK VENDORS

Aavid Engineering

One Kool Path
P.O. Box 400
Laconia, NH 03247
(603) 528-3400
(603) 525-1478 (FAX)

Contact: Gary F. Kuzmin (Product Marketing Manager)

EG&G Wakefield Engineering

60 Audubon Rd.
Wakefield, MA 01880
(617) 245-5900
(617) 246-0874 (FAX)

Contact: David Saums (Marketing Manager)

IERC

135 W. Magnolia Blvd.
Burbank, CA 91502
(818) 842-7277
(818) 848-8872 (FAX)

Contact: Guy R. Addis (Western Region Applications Engineer)

Thermalloy

2021 W. Valley View Lane
Dallas, TX 75234-8993
(214) 243-4321

Contact: Larry Tucker (VP of Sales and Marketing)

intel®

AP-481

APPLICATION NOTE

Designing with the Pentium™ Processor, 82496 Cache Controller and 82491 Cache SRAM CPU-Cache Chip Set

JIM REILLY
TECHNICAL MARKETING

October 1993

PRELIMINARY

Order Number: 241576-001

Designing with the Pentium™ Processor, 82496 Cache Controller and 82491 Cache SRAM CPU-Cache Chip Set

CONTENTS

CONTENTS

Designing with the Pentium™ Processor, 82496 Cache Controller and 82491 Cache SRAM CPU-Cache Chip Set

CONTENTS

CONTENTS

ACKNOWLEDGEMENTS

I would like to thank those who have provided their time and expertise to ensure that this document is accurate and complete.

Rob Aslett, Gary Dudeck, Scott Huck, Chris Lane, Dan McCutchan, Prakash Narayaranan, Tim Schreyer, and Tawfik Arabi

1.0 INTRODUCTION

The Pentium™ processor, 82496 cache controller, and 82491 cache SRAM CPU-Cache Chip Set has been designed to take advantage of the high performance available from the 66-MHz operating frequency. In addition, it has been designed to obtain this performance without severely adding to the complexity of the system design. These benefits are accomplished by dividing the chip set into two interfaces. The first is the External Interface which is the interface between the CPU-Cache core and the rest of the system. It consists of the memory bus and the memory bus controller. This interface has been designed to operate at a fraction of the CPU's frequency or asynchronous to the CPU. These options simplify the system design by minimizing the portions that must deal with the high frequency signals. The second interface is the Optimized Interface which is between the Pentium processor and the 82496 cache controller and 82491 cache SRAM. This interface is tuned for the known configuration options of the chip set and includes specially designed input and output buffers optimized for the defined electrical environment of each signal path. The specification of the optimized interface is defined to accommodate the tuning involved.

The purpose of this application note is to provide additional explanation of the steps involved in completing a chip set design. It includes:

1. Describing the specifications and how they should be used.
2. Describing Intel's I/O buffer models.
3. Describing the characteristics of printed circuit boards and transmission lines.
4. Stepping through an example of using the information and tools to complete the layout of one signal and verify that it meets the published chip set specifications.

In addition, all of the information associated with a completed chip set optimized interface design example is included. This information allows system designers to minimize their effort in implementing the same or similar design.

2.0 A/C SPECIFICATIONS OF THE CHIP SET

The AC/DC specifications for the chip set are published in the latest revision of the *Pentium™ Processor User's Manual Vol. 2: 82496 Cache Controller and 82491 Cache SRAM Data Book*. It includes the specifications for both the optimized and external interfaces.

2.1 Optimized Interface

The optimized interface is the high-performance interconnect between the Pentium processor and the 82496 cache controller and 82491 cache SRAM. The input and output buffers have been tuned for the defined configuration and electrical environment (loading, etc.). This tuning is what allows the chip set's CPU-Cache core to operate synchronously at 66 MHz.

There are two types of specifications for signals in the optimized interface. The first is Flight Time which is used to guarantee that signal timings are met. The second is Signal Quality to guarantee reliable operation. I/O buffer models have been provided as a tool to ensure these specifications are met. In this section, flight time and signal quality will be discussed. I/O buffer models will be left for the next section.

2

2.1.1 FLIGHT TIME SPECIFICATION

2.1.1.1 Purpose of Flight Time Specification

The purpose of the flight time specification is to guarantee that a signal supplied by a driving component is available at the receiving component for sampling.

It replaces the output valid delay and input setup time. The two methods are analogous, except that flight time allows the input and output buffer behavior to be matched without major impact to designers or the documentation. In other words, if component A's output is slower than expected, but component B's input is faster than expected, these two can be traded off without having to change the flight time specification. However, if output valid delay and input setup time specifications had been used the specifications would have to be changed. Note that in both cases the time available to the system designer to move the signal from one component to the other is the same.

2.1.1.2 Definition of Flight Time

Flight Time is the propagation delay of a signal from a driving component to any receiving component. It is defined as the time difference between the Vcc/2 (50%) level of an unloaded output signal and the Vcc/2 (50%) level of a receiving signal whose 50% Vcc to 65% Vcc rise time is greater than or equal to 1V/ns. Figure 1 shows the flight time measurement between the 50% Vcc points on the unloaded driver and receiver waveforms.

If the rise time between the 50% Vcc and 65% Vcc points is less than 1V/ns, the determination of flight time is slightly more difficult and requires more calculation. In this case the 65% Vcc point is extrapolated back to the 50% Vcc point using a 1V/ns reference slope (i.e., subtract 0.75 ns when Vcc = 5V). Figure 2 shows the extrapolation from the 65% Vcc point and the resulting flight time measurement.

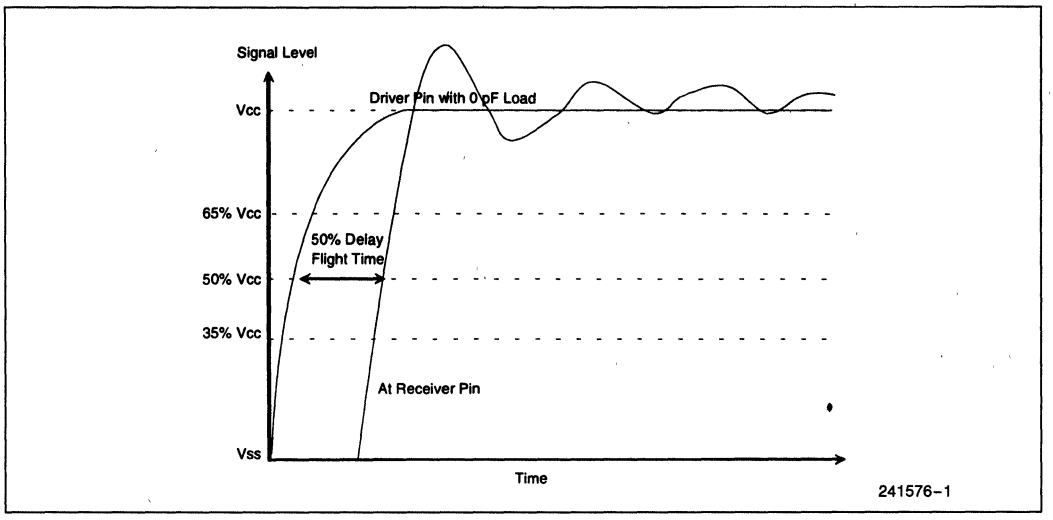


Figure 1. Flight Time Measurement

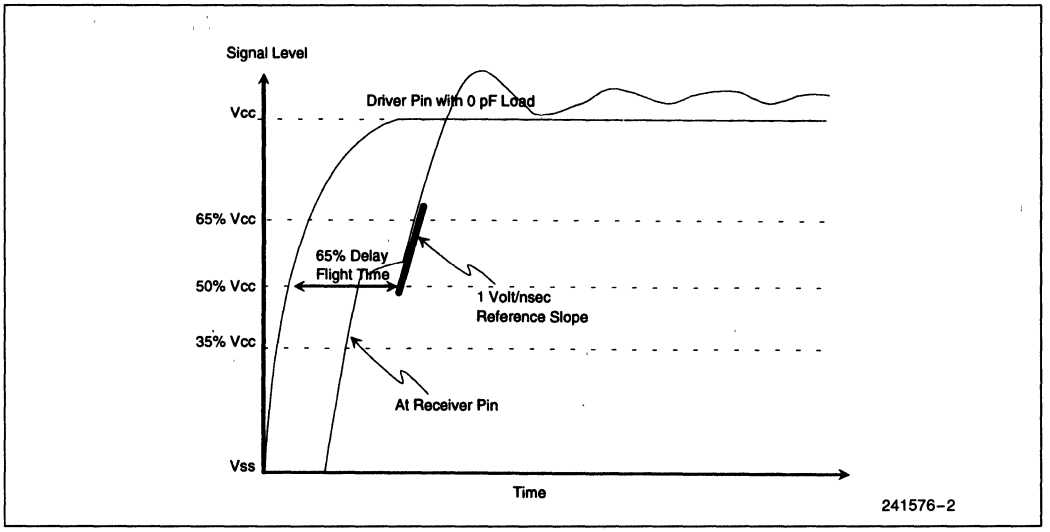


Figure 2. Flight Time Extrapolated from 65% Point

Thus flight time is the longer of T50-50 or Textrapolated. Although Figure 1 and 2 only show low-to-high transitions, flight time is the worst case of low-to-high and high-to-low transitions. Note on high-to-low transitions, 65% Vcc is replaced with 35% Vcc.

In a system environment it will not usually be possible to measure the delay of an unloaded driver. Figure 3 shows the method for measuring flight time in a system environment. As shown, the voltage measured at the pin of the loaded driver will have a ledge near the center of the transition. According to Transmission Line Theory, the time required to reach half the voltage level of the ledge is equivalent to the time required for an unloaded driver to reach the 50% Vcc level. The oscillation (if any) seen at the ledge defines the measurement uncertainty for this technique.

To measure flight time via this technique, first measure the maximum and minimum voltages of the ledge and take the average of these two values, (Vmax + Vmin)/2, to arrive at the ledge voltage. Finally, divide the ledge voltage by two, (Vmax + Vmin)/4. The result is the voltage level that approximately corresponds to the point in time at which an unloaded driver's signal would reach the 50% Vcc level. The flight time is determined by measuring the difference in time between the (Vmax + Vmin)/4 point and the extrapolated 50% point on the receiver, Textrapolated. The uncertainty of this technique is the time difference between the Vmin/2 and Vmax/2 points.

NOTE:

Figure 3 uses Textrapolated. This assumes the rise time between 50% Vcc and 65% Vcc is less than 1V/ns. If the rise time is equal to or greater than 1V/ns, the flight time should be measured to the 50% Vcc point, T0-50.

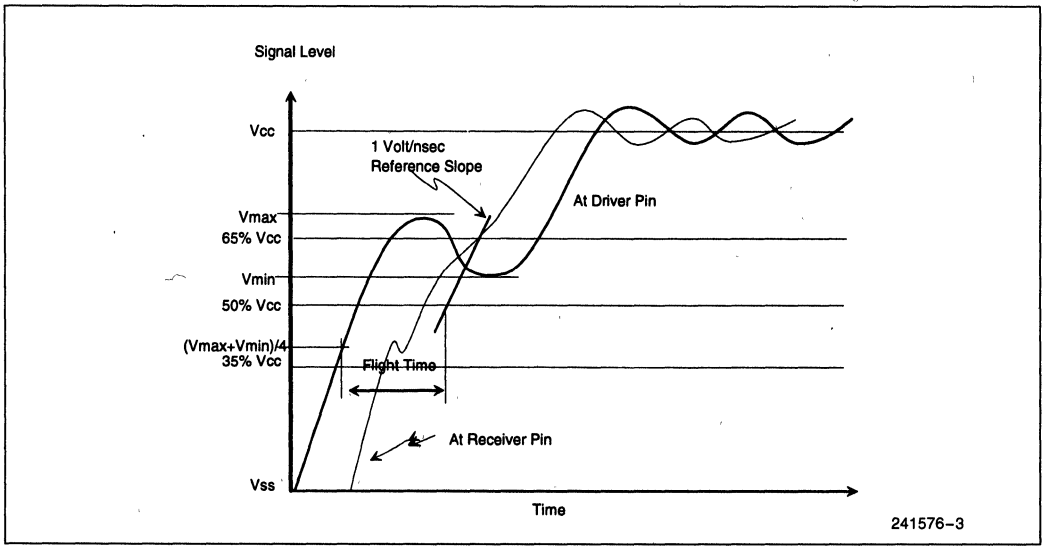


Figure 3. In-System Measurement of Flight Time

2.1.1.3 Clock Skew

Clock skew has generally been included in system design timing analysis. However, as frequencies increase, controlling clock skew becomes more important. Clock skew is the difference in time of the clock signal arriving at different components. It is measured at 0.8V, 1.5V, and 2.0V.

In synchronous devices, the clock signal defines the point in time in which signals are driven or sampled. It is important that all devices have a common reference. If the reference varies from component to component, the difference must be accounted for to ensure the devices function properly. In other words, clock skew must be subtracted from the clock period when performing a timing analysis of a system.

In the CPU-Cache Chip Set, the maximum clock skew between components in the optimized interface is a specification. This specification is required as a complement of flight time to ensure proper functionality. If clock skew exceeds the specified limit, the excess must be subtracted from the available flight time or the clock period must be increased.

2.1.2 SIGNAL QUALITY SPECIFICATIONS

Acceptable signal quality must be maintained over the entire operating range to insure reliable operation of the chip set. Signal quality consists of four parameters: Over/Undershoot, Time Beyond Supply, Ringback, and Settling Time. Figure 4 illustrates these signal quality parameters and how each is measured for a low-to-high transition. In addition to the absolute maximums associated with individual signals, each of the signal groups defined in the specification must meet maximum group average of Over/Undershoot and Time Beyond Supply. The following sections explain each of these in more detail.

Reliable operation means the signals are sampled correctly, do not exhibit false transitions, and that the long-term reliability of the component is not effected by overdriving the inputs.

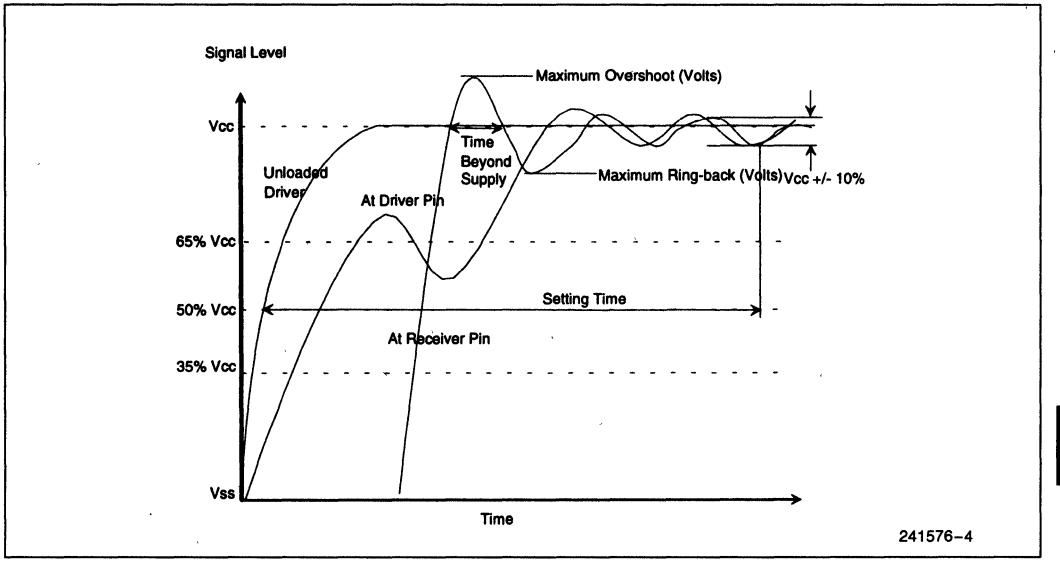


Figure 4. Signal Quality Parameter Measured for Low-to-High Transitions

2.1.2.1 Overshoot

Overshoot is the maximum absolute voltage a signal extends above Vcc or below Vss at the pin of the receiving component. Figure 4 shows the above Vcc case of overshoot. The overshoot specification is defined for use in simulation and assumes that input diodes are not included in the input model during simulation.

The overshoot specification maintains signal reliability by limiting the amount of energy that is injected into the component. Excessive energy being driven into the component can cause both short-and long-term reliability problems. They include Vcc or Vss plane shifts, electromigration, excessive ringback when the input diodes turn off, etc.

2.1.2.2 Time Beyond Supply

Time beyond supply is the maximum time a signal exceeds Vcc or Vss at the pin of the receiving component as shown in Figure 4. If the overshoot voltage is less than or equal to 0.5V, time beyond supply can be ignored. When time beyond supply is being ignored, a value of 0 ns should be used for that signal in calculating the group average time beyond supply.

Time beyond supply is a complement to overshoot when looking at signal quality. The time the signal is beyond the supply is a second factor in limiting the energy driven into the component. By limiting the time beyond supply, system designers are avoiding or minimizing the risk of the same reliability issues as described in the section on overshoot.

2.1.2.3 Ringback

Ringback is the maximum absolute voltage at the pin of the receiving component below Vcc or above Vss relative to Vcc or Vss after the signal has reached its maximum voltage level. Figure 4 illustrates how to measure ringback.

Eliminating ringback maintains signal quality by preventing a signal from re-crossing the threshold, causing a false transition to be detected.

2.1.2.4 Settling Time

Settling Time is the time required for a signal to settle within 10% of its final value referenced from the time unloaded driver's initial crossing of the 50% Vcc threshold. Figure 4 shows how settling time is measured on both low-to-high and high-to-low transitions.

The settling time specification is defined to ensure that a signal transition has completed and is no longer oscillating prior to the next transition. This is important to avoid forcing a signal to transition a distance significantly greater than Vcc/2. For example, if a signal is still not settled at the time of its next transition, it may be at a voltage above Vcc such as 6.5V. Assuming Vcc = 5V, the transition requires the voltage to swing from 6.5V (instead of 5V) to 2.5V. This added voltage distance translates into added flight time.

2.1.2.5 Group Averages

A Maximum Group Average specification is defined for Overshoot and Settling Time for each signal group. The group average is calculated by summing the maximum overshoot level or settling time for each signal in the group and dividing by the number of signals in the group.

The purpose of the group average specifications is to limit the amount of energy driven into the device. While each input can handle a certain amount of energy as limited by the Overshoot and Time Beyond Supply specifications, the overall part or portions of the part also have limits. These limits are maintained by ensuring the energy driven into these portions does not exceed a certain level.

2.2 External Interface

The External Interface is the interface between the CPU-Cache core and the rest of the system. It consists of the memory bus and the memory bus controller. This interface has been designed so that it can operate at a fraction of the CPU's frequency or asynchronous to the CPU. These options simplify the system design by minimizing the portions that must deal with the high frequency signals.

The specifications for the external interface are defined to allow system designers to connect the CPU and Cache components to other devices (ASICs, PLDs, memory, etc.). The external interface signal specifications are the more traditional output valid delay and float time and input setup and hold time. In addition, I/O buffer models have been provided as a tool to assist system designers.

2.2.1 OUTPUT VALID DELAY AND FLOAT TIME

Output Valid Delay is the amount of time it takes the signal to transition referenced from the clock edge. The maximum output valid delay is the earliest point in time that the signal can be assumed valid at the pin of the device. This timing is referenced from the clock edge and is measured at 1.5V as illustrated in Figure 5.

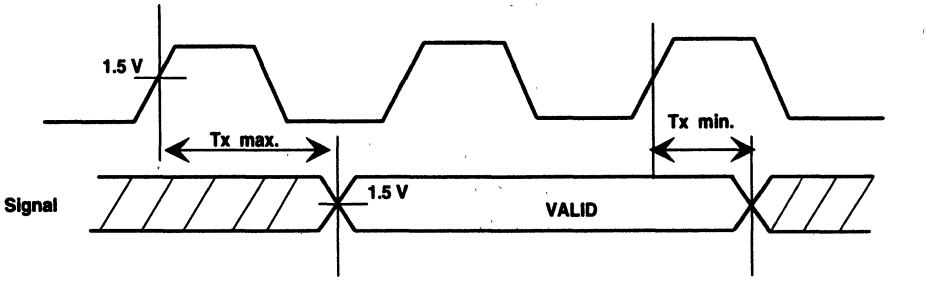
NOTE:
This specification is defined assuming C_L = 0 pF; therefore, system designers must account for all delay added by the signal traveling to the receiving device.

The maximum output valid delay is used by system designers to perform a worst case timing analysis to ensure that setup times are met at receiving devices.

The minimum output valid delay is the earliest valid data from the previous clock will begin transition after the clock edge. This timing is also referenced from the clock edge and is measured at 1.5V as illustrated in Figure 5.

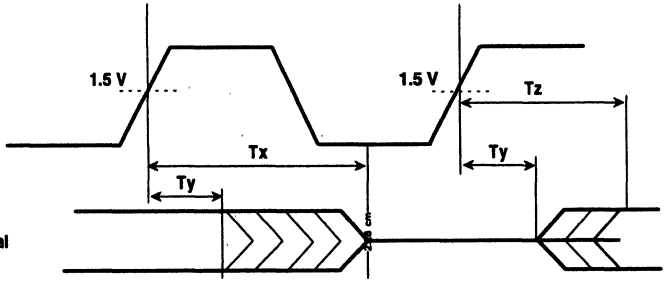
The minimum output valid delay is used by system designers to perform a worst case timing analysis to ensure that hold times are met at receiving devices. In addition, on I/O or tristate pins, it is used to ensure no bus contention exists due to multiple devices driving the bus simultaneously.

The maximum Output Float Time is the amount of time it takes to float a signal that was driven in the previous clock. This timing is also referenced from the clock edge and is illustrated in Figure 6. Note that the minimum valid delay determines how long data from the previous clock remains valid as shown in Figure 6.



Tx = valid delay

Figure 5. Output Valid Delay



Tx = max. float delay
Ty = min. valid delay
Tz = max. valid delay

Figure 6. Output Float Time

2.2.2 INPUT SETUP AND HOLD TIME

Input Setup Time is the amount of time a signal must be valid at the component's input pin prior to the clock edge it is sampled. The minimum input setup time is the latest point in time that the signal can be assumed valid at the pin of the device. This timing is referenced to the clock edge and is measured at 1.5V as illustrated in Figure 7.

The input setup time is used by system designers to perform a worst case timing analysis to verify that fast enough drivers have been chosen for ASICs or other devices and that the signals are able to travel across the board layout in the allotted amount of time.

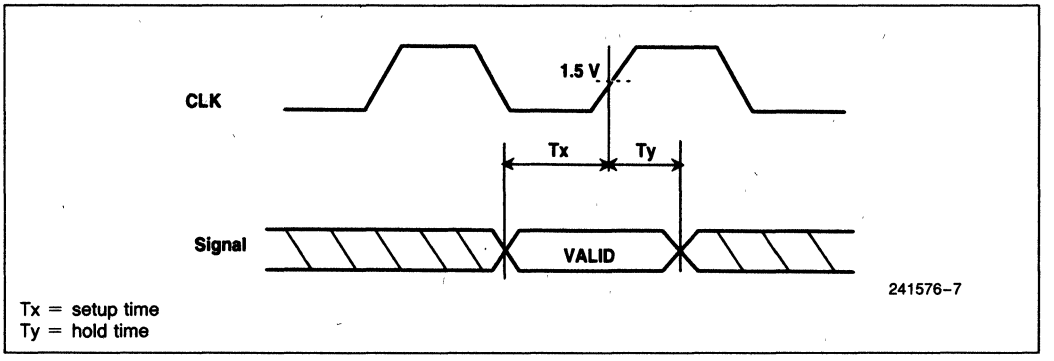


Figure 7. Input Setup and Hold Time

Input Hold Time is the amount of time a signal must be valid at the component's input pin after the clock edge is sampled. The minimum input hold time is the earliest point in time that the signal can start its next transition at the pin of the device. This timing is referenced to the clock edge and is measured at 1.5V as illustrated in Figure 7.

3.0 I/O BUFFER MODELS

3.1 Description of the First Order I/O Buffer Model

The first order I/O buffer model is a simplified representation of the complex input and output buffers used in the Pentium processor, 82496 cache controller, and 82491 cache SRAM. Figure 8 shows the structure of the input buffer model and Figure 9 shows the output buffer model. Tables 1 and 2 show the parameters used to specify these models.

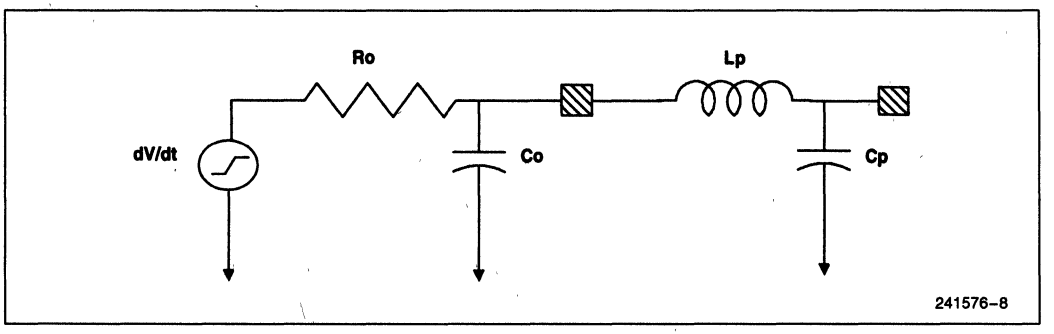


Figure 8. First Order Input Buffer

Table 1. The Parameters Used in the Specification of the First Order Input Buffer Model

Parameter	Description
Cin	Minimum and maximum value of the capacitance of the input buffer model
Lp	Minimum and maximum value of the package inductance
Cp	Minimum and maximum value of the package capacitance

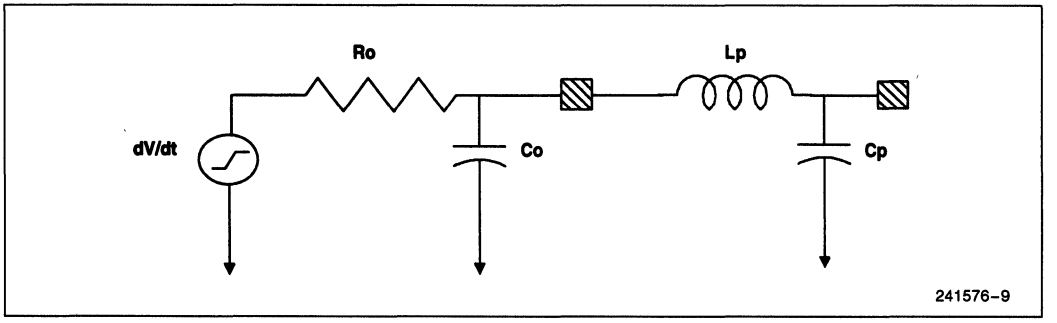


Figure 9. First Order Output Buffer

Table 2. The Parameters Used in the Specification of the First Order Output Buffer Model

Parameter	Description
dV/dt	Minimum and maximum value of the rate of change of the open circuit voltage source used in the output buffer model
Ro	Minimum and maximum value of the output impedance of the output buffer model
Co	Minimum and maximum value of the capacitance of the output buffer model
Lp	Minimum and maximum value of the package inductance
Cp	Minimum and maximum value of the package capacitance

The first order I/O buffer parameters for the chip set are published in the latest revision of the *Pentium™ Processor Data Book*. It includes the minimum and maximum values for each parameter allowing simulations for both the fast and slow condition to be performed.

The key to the first order model is that it is a simplistic representation of the input and output buffers. The parameters are easy to use in a variety of simulators and provide the needed accuracy to complete a chip set design. In addition, the simplicity greatly reduces the compute time required to perform simulations as compared to full transistor and process models.

4.0 HIGH FREQUENCY DESIGN CONSIDERATIONS

Any board interconnection is a transmission line by definition. However, as a general rule, the effects of modeling an interconnect/trace as a transmission line are negligible at low frequencies. This is because the reflections get masked since the propagation is short with respect to the signal rise time. In this case, system interconnects can be modelled as lumped loads with no sacrifice to accuracy. As the frequency at which signals change increases, the rise time becomes shorter and transmission line properties become significant and must be considered. Reflections, interference, and noise cause measureable changes in the appearance or behavior of signals at the higher frequencies. They can slow the transition time or cause signal quality violations. Figures 10 and 11 illustrate the effect of modeling traces as lumped loads or as transmission lines. Figure 10 shows a block diagram of a lumped load model and the resulting simulation and Figure 11 shows a block diagram of a transmission line model and the resulting waveform. The transmission line case shows the effects of reflections and how the signal varies at each component. These details are not shown in the lumped load case.

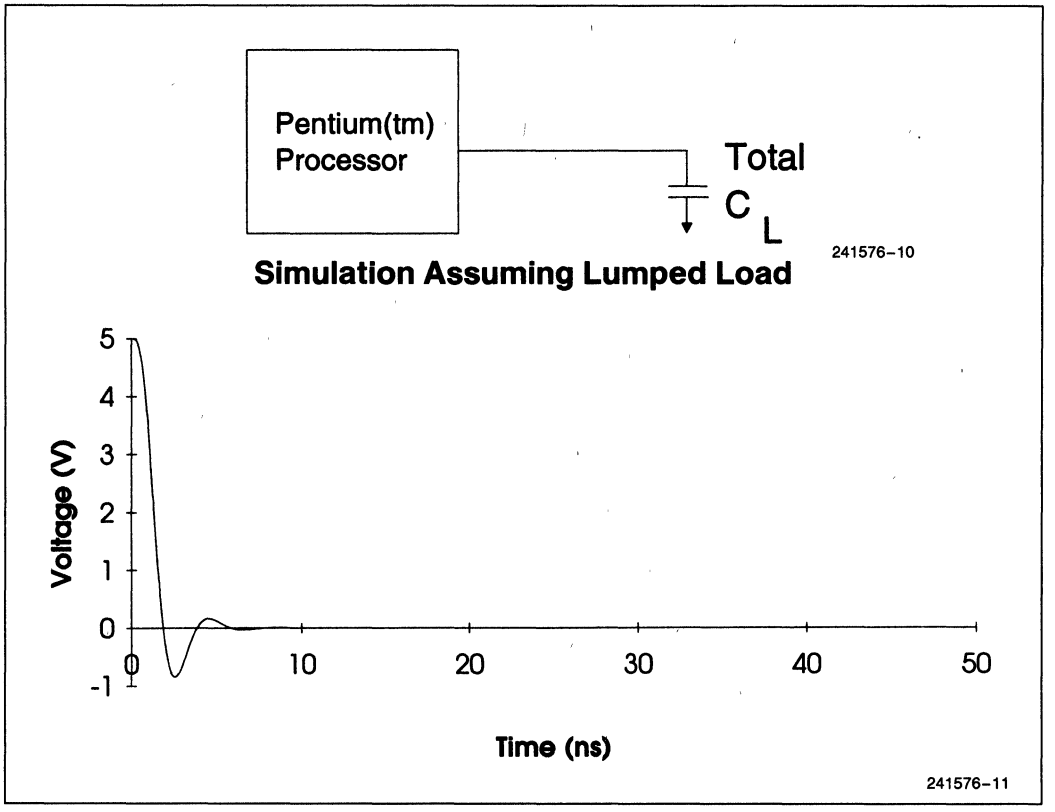


Figure 10. Lumped Load Simulations

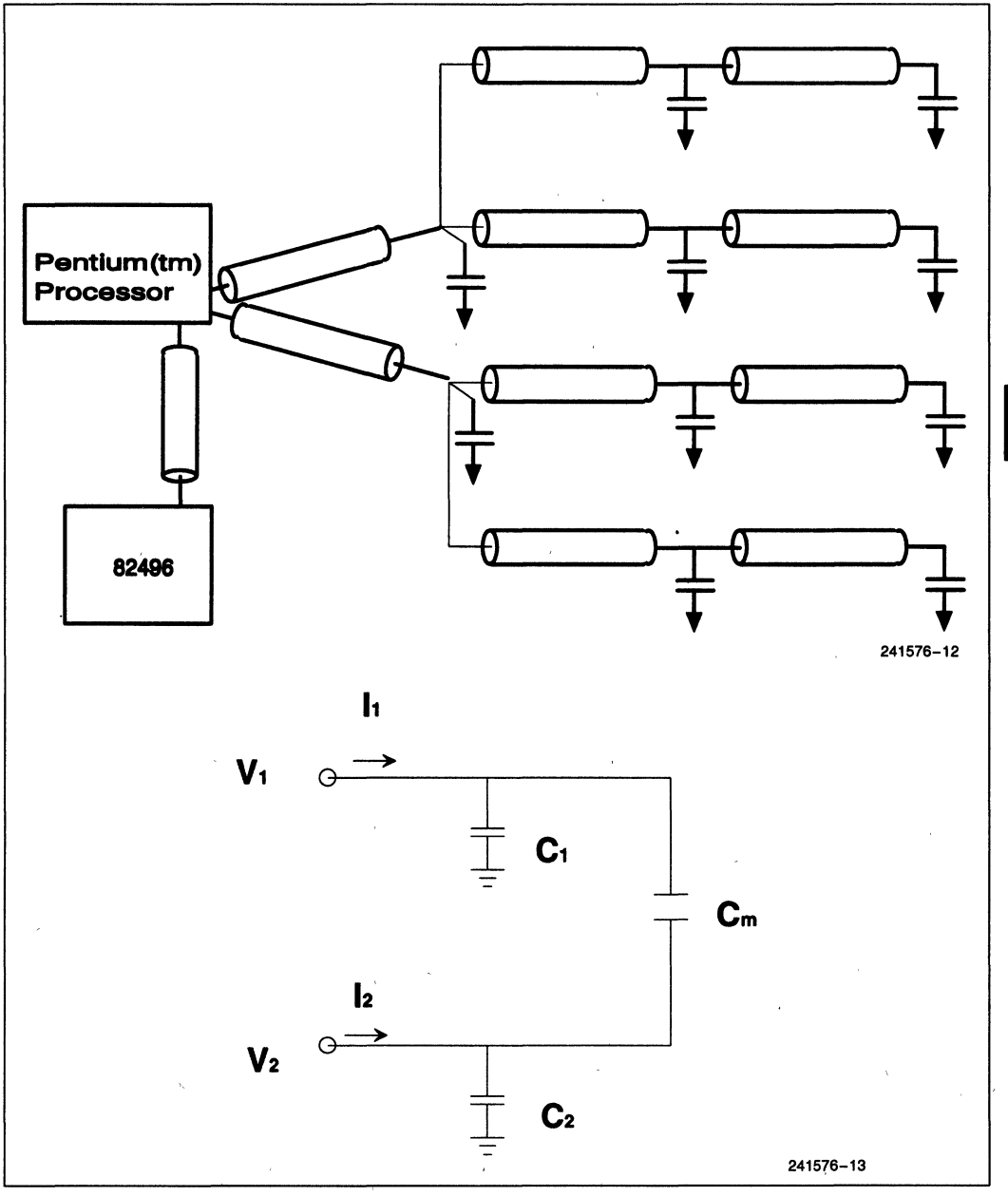


Figure 11. Transmission Line Simulation

To predict a trace's behavior and eliminate the negative effects, it is important to properly model board interconnects as transmission lines. A model consists of the driving component's output buffer model, the characteristics of the trace, and the receiving component's input buffer model. The I/O buffers are modelled using

the parameters published in the latest revision of the *Pentium™ Processor User's Manual, Vol. 2: 82496 Cache Controller and 82491 Cache SRAM Data Book.* The trace characteristics are a function of the actual board and the material used in its construction. Characteristic impedance (Zo) and propagation delay (Tpd) are the primary characteristics needed. These two parameters, along with length, can be used in many simulators to represent the transmission line as shown in Figure 12(a).

Some simulators may not have a single representation for a transmission line. These simulators require the user to model the transmission line in a more basic form. In addition to characteristic impedance and propagation delay, transmission lines can be characterized by their characteristic inductance (Lo) and characteristic capacitance (Co). The following equations can be used to convert between these four parameters:

$$Lo = Zo * Tpd$$

$$Co = Tpd/Zo.$$

Figure 12(b) illustrates the model of a transmission line using Co and Lo. The number of L-C links/inch in the chain should be chosen such that $(LC)^{1/2} \ll Tr$. As a good rule of thumb, two to four links per inch is a good starting point. The value of L and C is determined by dividing the characteristic impedance by the number of links/inch. A transmission line also has a resistive component, Ro. However, the Ro value is usually assumed negligible and omitted from the model. Its only effect is to cause a voltage loss between the driver and the receiver. Since Ro's value is negligible, the voltage loss is very small and can also be ignored.

$$L = Lo/n$$

$$C = Co/n$$

$$n = \text{number of links/inch}$$

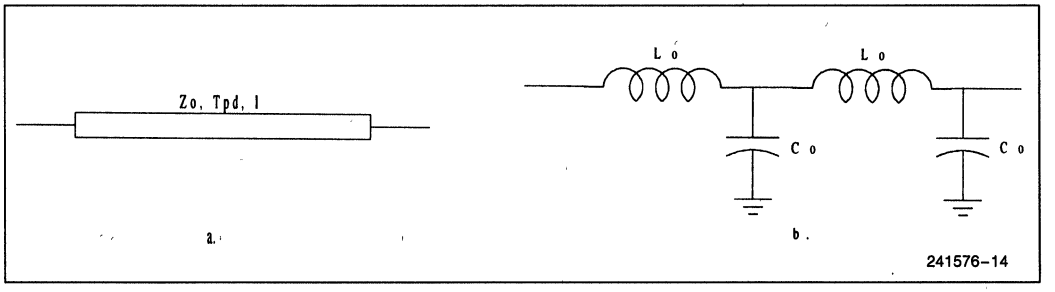


Figure 12. Representation of a Transmission Line

In order to complete the model, the actual parameter values must be determined. That means the next step is to determine the values of Zo and Tpd. To do this the designer must understand the construction of the transmission line. In particular that means understanding how the printed circuit board is constructed and the geometry of the various layers and traces.

4.1 Printed Circuit Board

A printed circuit board consists of some number of signal layers and power and ground planes separated by a dielectric material. An example is illustrated in Figure 13.

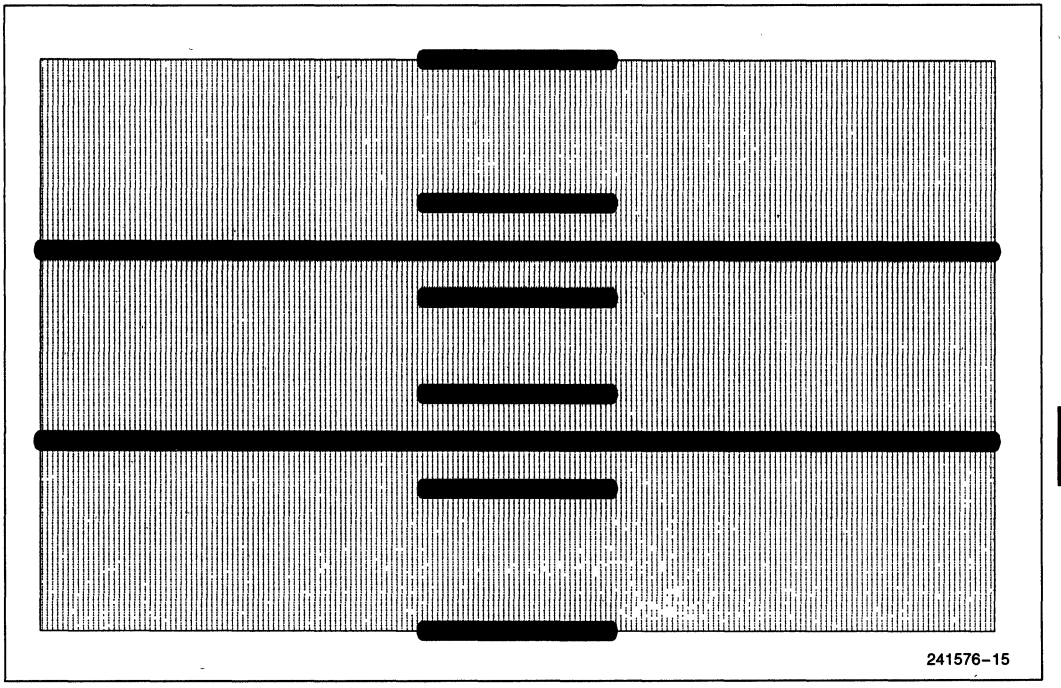


Figure 13. Printed Circuit Board

Although there are many types of transmission lines, those most commonly used on printed circuit boards are microstrip lines and striplines. Microstrip lines consist of a signal trace that is separated from a power or ground plane by a dielectric as shown in Figure 14.

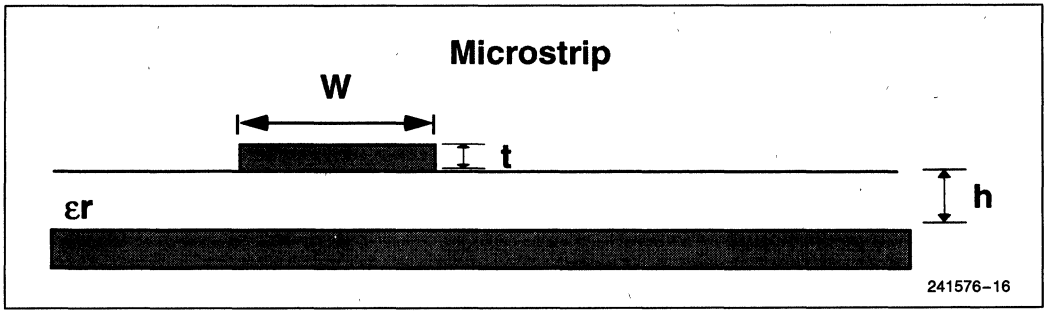


Figure 14. Microstrip Trace

The characteristic impedance is a function of the dielectric constant and the board geometry. For a microstrip trace this is given by:

$$Zo = \frac{87}{(\epsilon_r + 1.41)^{1/2}} \ln\left(\frac{5.98h}{0.8W + t}\right) \text{ Ohms}$$

where e_ϵ is the relative dielectric constant of the board material. The propagation delay is a function of the dielectric constant only. For a microstrip trace this is given by:

$$tpd = 1.017\ (0.475\ \epsilon_r + 0.67)^{1/2}\ ns / ft$$

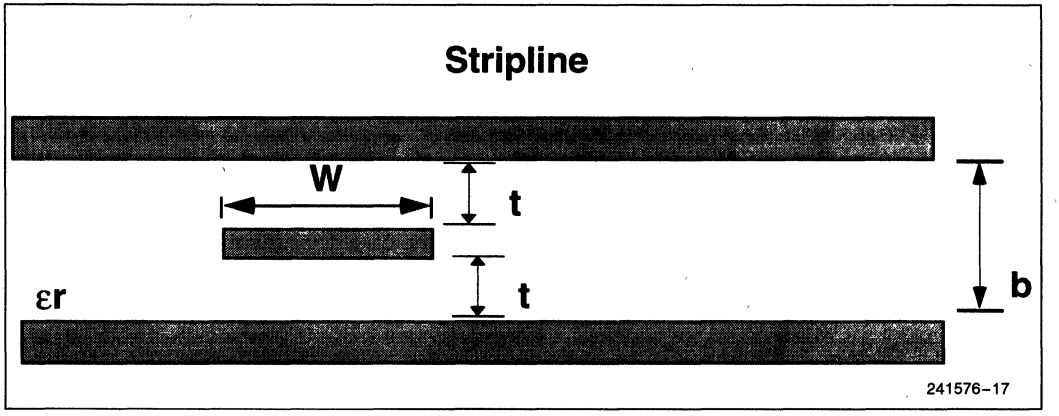


Figure 15. Stripline Trace

The characteristic impedance is a function of the dielectric constant and the board geometry. For a stripline trace this is given by:

$$Zo = \frac{60}{(\epsilon_r)^{1/2}} \ln\left(\frac{4b}{0.67\,\pi(t + 0.8W)}\right) \text{ Ohms}$$

where e_r is the relative dielectric constant of the board material. The propagation delay is a function of the dielectric constant only. For a stripline trace this is given by:

$$tpd = 1.017\ (\epsilon_r)^{1/2}\ ns/ft$$

Signal traces that are located on the external layers of the board can be modelled using the microstrip characteristics. Signal traces that are located on internal layers between the power and ground planes can be modelled by using the stripline characteristics.

4.2 Transmission Line Behavior

Now that the parameters needed to represent a transmission line are complete, the next step is to look at how the transmission line behaves or effects signals that are transmitted on it. The four primary effects are signal propagation and reflection, crosstalk, and skew.

Striplines consist of a signal trace that is located in a dielectric material between two power or ground planes as shown in Figure 15.

4.2.1 SIGNAL PROPAGATION AND REFLECTION

Ideally, a signal that is driven by one device to another across a trace will reach the receiving device instantly and without any distortion. In reality, this is not the case. Because of the resistive, capacitive, and inductive components of a transmission line and the attached loads, the voltage and current of a signal may change as the signal travels along the trace and may vary over time.

For simplicity, the examples that follow will assume lossless transmission lines. In other words, the transmission line itself does not cause an attenuation of the voltage as it travels along the trace. The voltage loss seen at the receiving end is a function of the resistive component of the transmission line which is negligible.

An example of a signal propagating along a transmission line is shown in Figure 16. The voltage source at the left launches a wave onto the transmission line. The voltage of the wave is equal to the voltage division of the voltage source resistance and the line inductance.

$$Vi = Vin * Zo/(Zo + Rs)$$

This waveform is propagated down the transmission line without any voltage loss or change in the waveform. The only effect of the transmission line is to delay the signal from reaching the receiving end. When the wave reaches the receiving end, it is reflected back towards the source. The reflection is proportional to the initial wave by an amount called the Reflective Coefficient. A reflective coefficient exists for both the source and the load. The values are a function of the source or load resistance and the lines characteristic impedance.

$$\rho_L = (R_L - Zo)/(R_L + Zo)$$
$$\rho_s = (R_s - Zo)/(R_s + Zo)$$

The value of the initial reflection at the load is:

$$Vr = Vi * \rho_L$$

The reflections can be caused by any discontinuity on the line. The discontinuity can be caused by a mismatch in impedance between the source or load and the characteristic impedance of the trace, branches in the trace, vias, or bends and angles in the trace. Here the discontinuity between the source and load are used as an example because they are probably the most prominent. Each reflection can attenuate or reinforce the wave depending on the phase of the reflection. The reflections continue indefinitely; however, with each reflection the magnitude of the voltage decreases and the line approaches a steady state value. A rule of thumb is that by the third or fourth reflection the value is negligible.

2

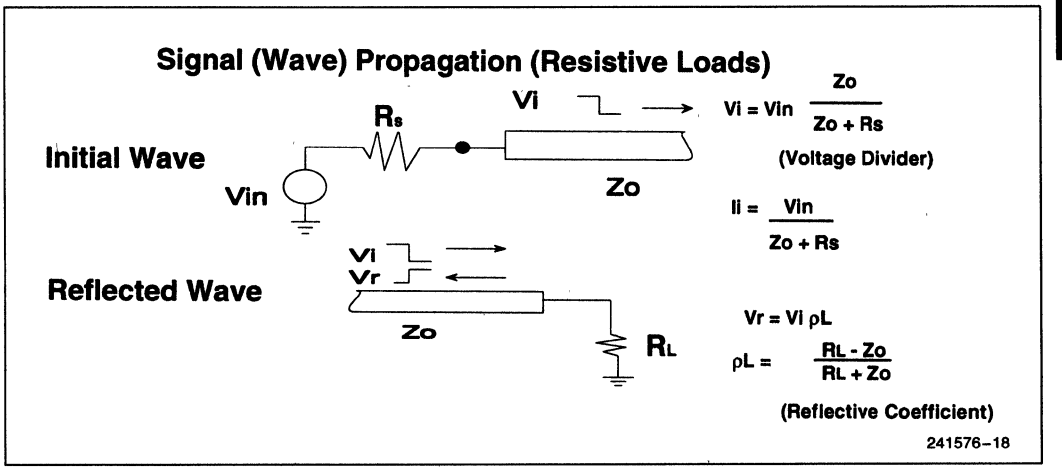


Figure 16. Signal Propagation Along a Transmission Line with Resistive Loads

Now that the voltage of each reflected waveform can be calculated, the next step is to sum these values to determine the voltage measured on the line at any point in time. The superposition principle comes into play. It states that the voltage/current at any point on a transmission line equals the sum of the voltages/currents of all the signals (waves) that have passed that point. In other words, as each reflection passes the point of measurement, it is added to the previous voltage seen at that point. Figure 17 illustrates the voltage seen at the midpoint of the circuit shown in Figure 16 over time.

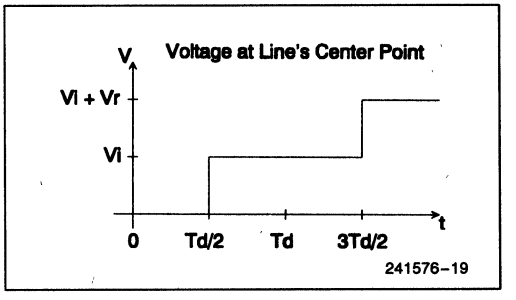


Figure 17. Voltage at the Midpoint of Circuit in Figure 16

From time 0 until Td/2 the voltage is 0V at the midpoint because the initial wave has not yet reached the midpoint. At time Td/2, the initial wave reaches the midpoint and the voltage is Vi. This wave travels down the trace until it reaches the load and a reflection occurs. The reflection begins traveling back towards the source, but does not reach the midpoint until time 3Td/2. At that time the voltage increases by Vr, the reflections voltage as shown in Figure 17. The following equation determines the voltage at any given point on a trace at the given time.

$$
\begin{aligned}
V(x,t) = [Zo / (Zo + Zs)] * \{ & Vin\,[t - (t - tpd * x)] * U(t - tpd * x) \\
& + \rho_L * Vin\,[t - (t - tpd * (2L - x))] * U(t - tpd * (2L - x)) \\
& + \rho_L * \rho_S * Vin\,[t - (t - tpd * (2L + x))] * U(t - tpd * (2L + x)) \\
& + \rho_L 2 * \rho_S * Vin\,[t - (t - tpd * (4L - x))] * U(t - tpd * (4L x)) \\
& + \rho_L 2 * \rho_S 2 * Vin\,[t - (t - tpd * (4L + x))] * U(t - tpd * (4L + x)) \\
& + \ldots\}
\end{aligned}
$$

U(x) = unit step function
Tpd = propagation delay of signal traveling along the transmission line (ns/ft)

As reflections occur on the line they can cause slower signal transitions, overshoot, undershoot, ringing, and other undesirable effects. Although many of the effects of reflections are negative, sometimes designers take advantage of constructive reflections to decrease the time it takes for the voltage to reach its final value at the destination. In general, designers try to minimize the magnitude of reflections.

System designers can do several things to reduce or minimize reflections: reduce angles (specifically 90° angles in traces, minimize the number of vias, and use termination when necessary).

Figure 18 illustrates how angles can be reduced by using 135° bends instead of 90° bends. The 135° bend approximates a smooth curve more closely than the 90° bend. The discontinuity occurs in the 90° bend because the trace is wider through the bend and therefore the impedance is altered by the geometry of the trace.

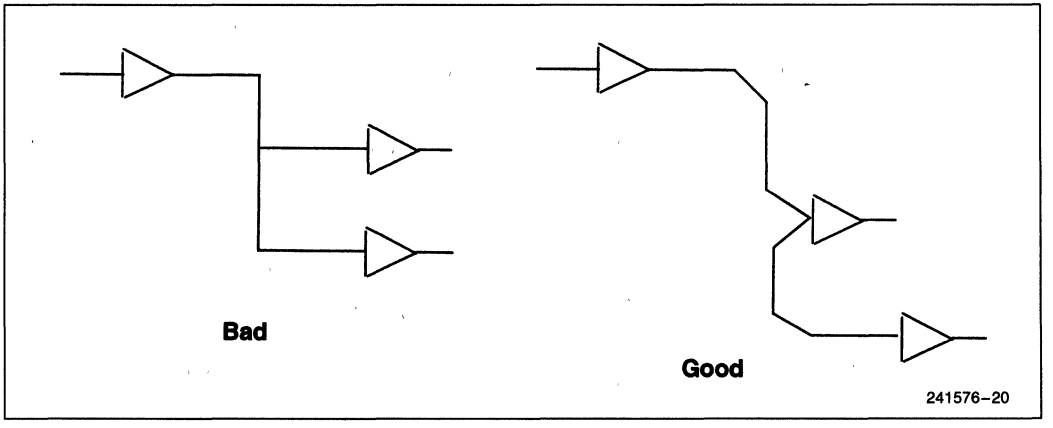


Figure 18. Eliminate 90° Angles

Figure 19 illustrates a way to reduce the number of discontinuities by minimizing the number of vias. Once again, a via causes a change in the path of a signal much like the 90° angles do. In addition, the geometry of a via is generally wider or thicker than the rest of the trace resulting in a different impedance for that portion of the interconnect. The change of impedance in the path causes the discontinuity and the resulting reflections.

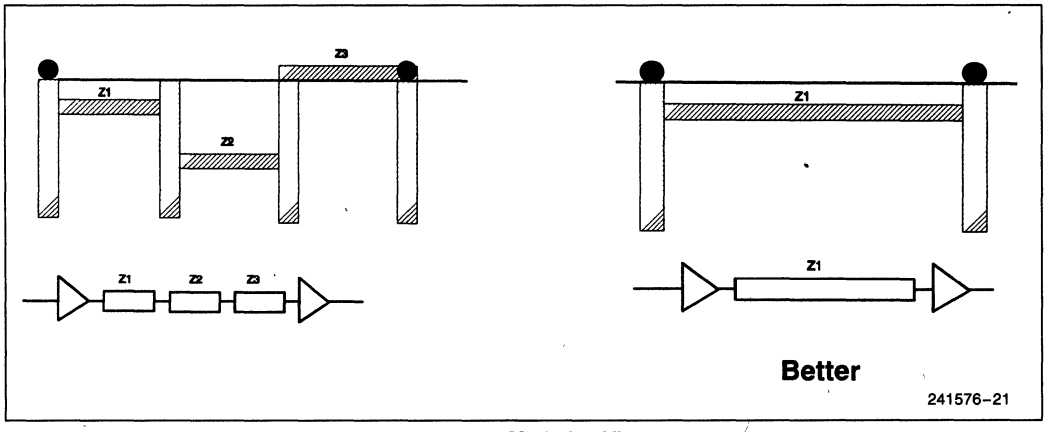


Figure 19. Minimize Vias

It is not always possible to eliminate all the discontinuities or mismatches in impedance. When this is the case, it is sometimes necessary to use a technique called *Termination* to artificially make the mismatched impedances appear matched. This technique is normally used to match a traces characteristic impedance with either the load or sources impedance.

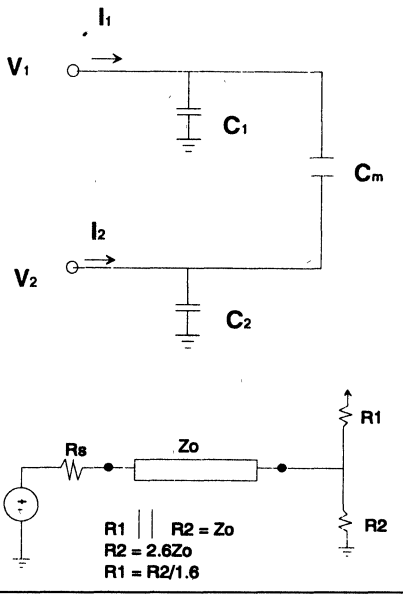
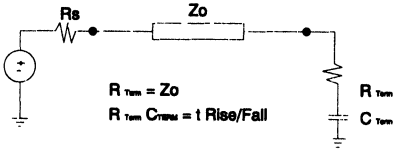
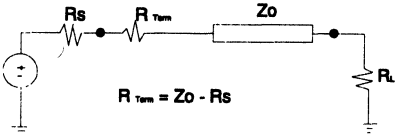
$$Z_o = Z_{term} + Z_L$$

and

$$Z_o = Z_{term} + Z_S$$

Several techniques of termination exist. They are Parallel, AC or RC, and Series termination. Each technique has its own advantages and disadvantages as summarized in Table 3.

Table 3. Termination Techniques

Type of Termination	Figure	Advantages	Disadvantages
Parallel		• eliminates reflections at receiver • good overshoot suppression • no added delay	• high power dissipation • requires Zo > 100Ω to avoid exceeding dc current limit • reduced voltage swing
AC or RC		• low power consumption • full voltage swing • eliminates initial reflection at receiver	• Cterm adds capacitive Load to driver • added delay due to RC time constant • component size and count
Series		• no additional loading on driver • no additional charging time • low power consumption • eliminates secondary reflection at source	• added delay

4.2.2 CROSSTALK

Crosstalk is another side effect of transmission line interconnnects. Crosstalk is the result of fields from adjacent traces interacting with each other. The interaction can alter the characteristics of a driven line or cause noise to be coupled into passive lines.

Crosstalk can be characterized by two parameters: Mutual Inductance, L_m, and Mutual Capacitance, C_m, as shown in Figure 20 and 21, respectively. These two parameters represent the inductive and capacitive values that exist between two adjacent lines. The inductance allows a current in one line to induce a voltage in a second line.

$$V_{m2} = L_m * \Delta I1/\Delta t$$

The capacitance allows a voltage on one line to induce a current in the second.

$$I_{M2} = C_m * \Delta(V1V2)/\Delta t$$

These mutual components have an additive effect to the L and C used to characterize each transmission line. To see what effect this has, examine the two components separately. First, Figure 20 illustrates two parallel traces with their inductive components and a mutual inductance between the two.

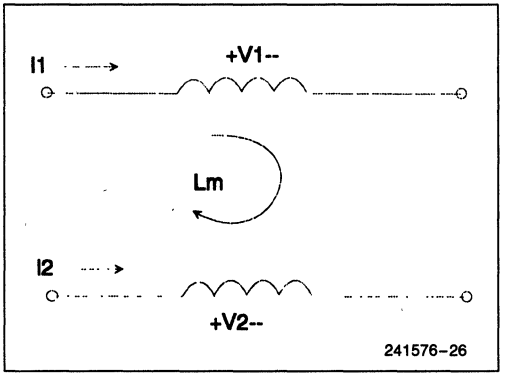


Figure 20. Inductive Components of Two Parallel Traces

The voltage seen on each line is given by:

$$V_1 = V_{L1} + V_m = (L_1 * \Delta I_1/\Delta t) + (L_m * \Delta I_2/\Delta t)$$
$$V_2 = V_{L2} + V_m = (L_2 * \Delta I_2/\Delta t) + (L_m * \Delta I_1/\Delta t)$$

To see what effect this has assume $L_1 = L_2$ and the magnitude of $\Delta I_1/\Delta t$ = the magnitude of $\Delta I_2/\Delta t$. This allows the above equations to be simplified to:

$$V_1 = V_2 = (L_1 + L_m) * \Delta I_1/\Delta t$$

(Current in same direction)

and

$$V_1 = -V_2 = (L_1 - L_m) * \Delta I_1/\Delta t$$

(Current in opposite direction).

From these equations the effective inductance seen on either trace is:

$$L_{eff} = L_1 + L_m$$

(Current in same direction)

and

$$L_{eff} = L_1 - L_m$$

(Current in opposite direction).

Therefore, if the currents are flowing in the same direction the effective inductance of each trace is increased. If the currents are in opposite directions the effective inductance of each trace is decreased.

Secondly, Figure 21 illustrates two parallel traces with their capacitive components and a mutual capacitance between the two.

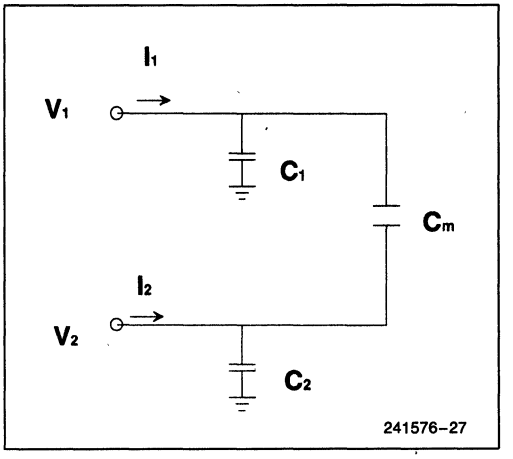


Figure 21. Capacitive Components of Two Parallel Traces

The current seen in each line is given by:

$$I_1 = I_{C1} + I_M$$
$$= (C_1 * \Delta V_1/\Delta t) + (C_m * \Delta(V_1 - V_2)/\Delta t)$$
$$= ((C_1 + C_m) * \Delta V_1/\Delta t) - (C_m * \Delta V_2/\Delta t)$$

$$I_2 = I_{C2} + I_M$$
$$= (C_2 * \Delta V_2/\Delta t) + (C_m * \Delta(V_2 - V_1)/\Delta t)$$
$$= ((C_2 + C_m) * \Delta V_2/\Delta t) - (C_m * \Delta V_1/\Delta t)$$

Using the same assumptions that $C_1 = C_2$ and that the magnitude of $\Delta V_1/\Delta t$ = the magnitude of $\Delta V_2/\Delta t$ allows the equations to be simplified to:

$I_1 = I_2 = C_1 * \Delta V_1/\Delta t$ (Voltage change in the same direction on both traces)

and

$I_1 = -I_2 = (C_1 + 2C_m) * \Delta V_1/\Delta t$ (Voltage change in the opposite direction on each trace).

From these equations the effective capacitance seen on either trace is:

$$C_{eff} = C_1 \quad \text{(Voltage change in same direction)}$$

and

$$C_{eff} = C_1 + 2C_m \quad \text{(Voltage change in opposite directions).}$$

Therefore, if the voltages are changing in the same direction the effective capacitance of each trace is unchanged or decreases. If the voltages are changing in opposite directions the effective capacitance of each trace is increased. See Figure 22.

If L_{eff} and C_{eff} are used to determine Zo and Tpd, the following results:

$$Zo = (Leff/Ceff)^{1/2} \quad Tpd = (Leff * Ceff)^{1/2}$$

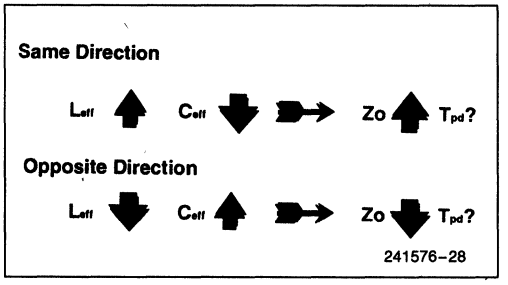


Figure 22. Effect of Changing Voltages in the Same or Opposite Directions

Electrons travel at the speed of light, so Tpd can never decrease. Therefore, Tpd either remains constant or increases.

This altering of Zo and Tpd by crosstalk explains why termination is never 100% effective. The crosstalk leads to a variation between the targeted Zo and the actual Zo. Termination is usually defined to match the targeted Zo's. The result is an interconnect that is not perfectly matched via termination.

5.0 CHIP SET DESIGN

As simulation tools have improved it has become easier to test design assumptions before actually spending the time or money to build a printed circuit board. In addition, the frequencies at which signals switch have also increased and complicated the process of designing a board that ensures signals reach their destination at the correct point in time and maintain a reasonable level of signal quality. To better predict signal behavior and minimize the need for board rework or revision, many designers are simulating their board layouts before building a board. The complexity of the optimized interface of the CPU-Cache Chip Set is a prime candidate for this type of approach. In this interface designers must ensure that the signals accurately travel along the interconnects at very high frequencies (i.e., 66 MHz). As discussed, transmission line effects become a more dominant influence on signals switching at these frequencies. It is important to take these effects into account to ensure that no specification violations occur.

A possible scenario for designing the optimized interface is shown in Figure 23. As always the first step is to understand the specifications. This document along with the published specifications should help complete this step. Based on these specifications, system geometry requirements, and an understanding of the board's basic electrical characteristics, a first pass component placement and routing can be completed. Once the routing is complete or possibly as part of the routing, individual traces should be simulated to determine their electrical behavior. This includes examining both flight time and signal quality for each signal and determining if it meets the specification. Any signals that violate the specification must be modified. Portions of this document will provide some information and guidelines on how to modify or route the traces to meet the specifications. With each change, the routing should be re-simulated to ensure the specifications are still met.

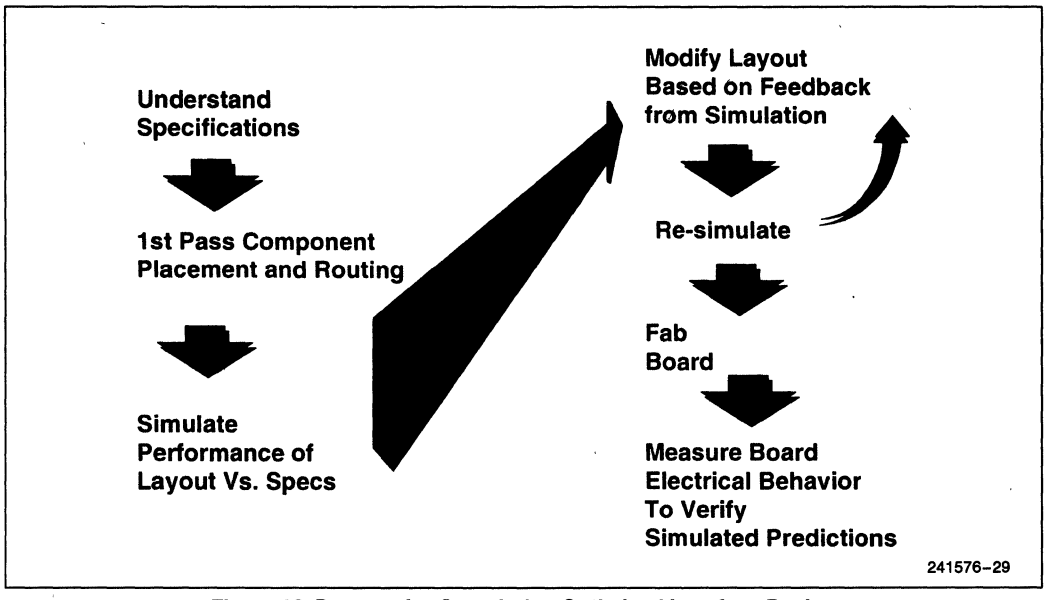


Figure 23. Process for Completing Optimized Interface Design

Once all the specifications are met, it is time to build the board. The goal is that once this board is manufactured and the components are installed, it will meet specification without any changes. However, this must be verified by making actual measurements on the board to verify all of the flight time and signal quality specifications are met. It is also beneficial to make sure that the actual measurements correlate to the predicted results from simulation. This is especially helpful if any corrections are required to bring the board within specification.

The next couple of sections will describe the requirements and guidelines that should be followed while making these simulations and measurements.

5.1 Simulation Environment

The environment chosen to simulate the optimized interface is very critical. A number of different options are available on the market today. It is the system designer's responsibility to select the option best suited for their design requirements. These requirements will include the accuracy of the results; as well as, how easy it is to import schematics, layout routing, or modeling parameters.

5.1.1 SIMULATION REQUIREMENTS

When simulating the optimized interface to determine flight time and signal quality, it is important that the appropriate modeling parameters are used. The I/O models are provided with minimum and maximum values for each parameter. Using these values the fast and slow corners of the buffer's behavior can be modeled. In addition, the printed circuit board can be modeled for its fast and slow corners. Table 4 restates the characteristics of a printed circuit board.

Flight time is determined by simulating with the slow corner used for all parameters. In this corner signals require the longest amount of time to transition and reach their destination. The fast corner is used to simulate signal quality. In the fast corner, signals transition their fastest and are therefore their noisiest. Table 5 summarizes the parameter values used to simulate for flight time and signal quality.

Table 4. Parameters Used to Specify Printed Circuit Board Characteristics

Parameter	Symbol	Description
Characteristic Impedance	Zo (Ω)	Minimum and maximum impedance for signal traces on each layer
Propagation Delay	S (ns/ft)	Minimum and maximum propagation delay for signal traces on each layer
Via Capacitance	Cvia (pF)	Minimum and maximum capacitance of a via used to pass a signal from one layer to another of the PC board

Table 5. Parameter Values Used to Simulate Flight Time and Signal Quality

Device	Modeling Parameter	Flight Time	Signal Quality
Input Buffer	Cp	Max	Min
	Lp	Max	Min
	Cin	Max	Min
Output Buffer	dV/dt	Min	Max
	Ro	Max	Min
	Co	Max	Min
	Lp	Max	Min
	Cp	Max	Min
Printed Circuit Board	Zo	Min	Max
	S	Max	Min
	Cvia	Max	Min
Other	Temperature	Max	Min
	Vcc	Min	Max

These values should be used to define the simulation model files used to simulate for flight time and signal quality.

While simulating the two corners it should become obvious that there will be trade-offs in optimizing for one or the other. Some sacrifices in signal quality may be required to ensure flight time specifications are met, or vice versa.

5.2 Routing Signal Traces for Their Optimal Performance

Priority should be given to optimizing the performance of the signals in the optimized interface. For the 256K byte layout example that Intel completed, the signals have been divided into the categories listed in Table 6. These categories are based on fanout and connectivity characteristics.

Table 6. Optimized Interface Signal Categories

Category	Signal
Low Address (connected to PP, CC, and CS)	A3–A16, HITM#, W/R#
High Address (connected to PP and CC)	A17–31, BT0–3
PP-CC Control (connected to PP and CC)	Driven by PP: ADSC#, AP, CACHE#, D/C#, LOCK#, M/IO#, PCD, PWT, SCYC Driven by CC: AHOLD, BRDYC1#, EADS#, INV, KEN#, NA#, WB/WT#
PP-CS Control (connected to PP and CSs)	ADS#
CC-CS Control (connected to CC and CSs)	BLAST#, BRDYC2#, BUS#, MAWEA#, MCYC#, WBA, WBTYP, WBWE#, WRARR#, WAY
Other CC Control	BLEC#, BOFF#
Byte Enables	BE0#–BE7#
CPU Data and Parity	CD0–CD63, CP0–CP7

PP = Pentium processor
CC = 82496 cache controller
CS = 82491 cache SRAM

Within each category the routing or topology should be defined to minimize delay while maintaining acceptable signal quality. To do this and maintain the manufacturability of the board, rules were defined to govern the line lengths for each segment of a topology. To develop these rules some analysis of board characteristics and signal behavior is necessary.

5.2.1 RULES FOR OPTIMIZING SIGNAL ROUTING

Both the fast and slow corners must be considered to ensure both flight time and signal quality are met by optimizing a signal's routing.

Flight time is minimized by optimizing each interconnect to minimize the distance the signal must travel and the loading presented to the driver. The dominant opposition to minimizing these factors is the printed circuit board's geometry requirements (i.e., physical distance between components and component placement) and electrical characteristics (propagation delay and characteristic impedance).

The strategy used to optimize each interconnect for signal quality is to make each net's routing electrically symmetric. This is especially important on heavily loaded nets.

Electrically symmetric means the delays of each branch within the net are equal when viewed from the driver. Figure 24 shows a topology from the 256 Kbyte layout example that illustrates this principle. For this topology with the Pentium processor driving, the symmetry is best when the delay from the Pentium processor to the 82496 cache controller is equal to the delay from the Pentium processor to the farthest 82491 cache SRAM. By making these delays equal, the round-trip delays are also equal, and therefore any reflections return to the Pentium processor simultaneously. By returning simultaneously, the reflections can rapidly cancel each other, resulting in the waveform settling quickly.

If these two delays are not equal, asymmetric reflections return to the Pentium processor at different times, and do not cancel each other. The result is a complex interference pattern that generates considerable ringing. In some cases this ringing can last for more than one clock cycle.

5.2.2 DETERMING THE OPTIMAL NET

There are two methods of optimizing the line lengths and relationships of traces within a net. One uses an asymmetry factor [5] to identify the optimal relationship. The other uses settling time to find this relationship.

Using the asymmetry factor to optimize the symmetry of the net in Figure 24, the input impedance (frequency-domain) of each branch was calculated from the driver's point of view. The branch impedances were compared to each other and the difference was integrated over all frequencies, resulting in a symmetry energy factor which quantifies the amount of symmetry in the topology. Figure 25 also shows a plot of this factor as a function of the lengths of segments L__a and L__b.

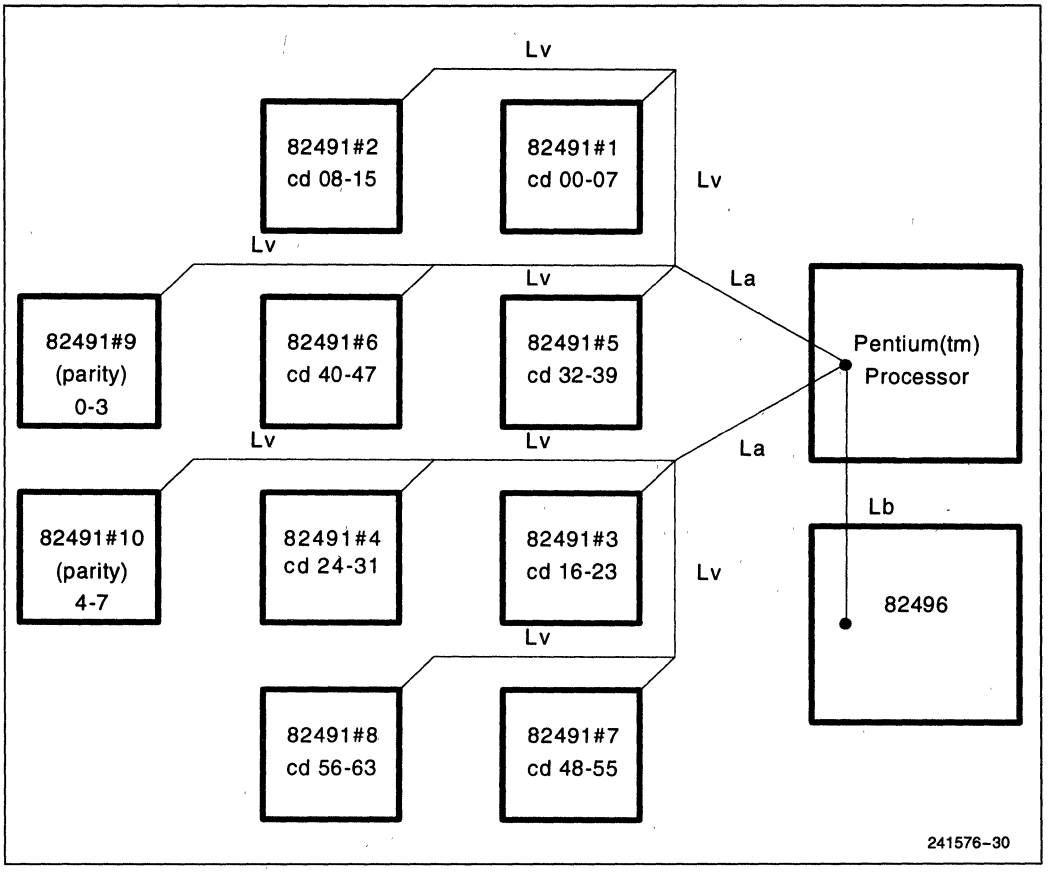


Figure 24. Energy Minimization for a Given Topology

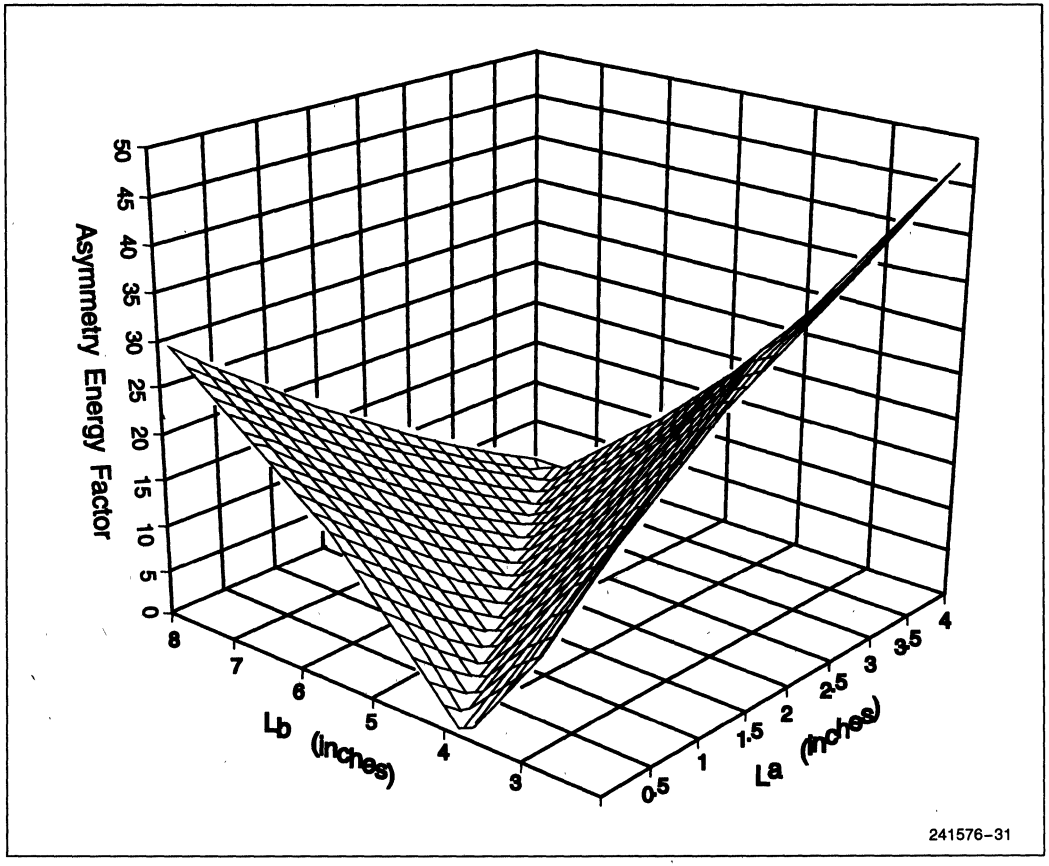


Figure 25. Energy Minimization for a Given Topology

There is a strong correlation between the asymmetry factor and the net's signal quality. This is reinforced by simulating the topology using values for L__a and L__b from the plot in Figure 25. Figure 26 shows the waveforms obtained by simulating the topology with symmetric values for L__a and L__b from along the energy minimum. The line of points in the plot of Figure 25 where the energy minimum occurs corresponds to topologies that are electrically symmetric. An asymmetric topology is obtained by using values for L__a and L__b that lie away from the energy minimum. The waveform obtained by simulating this asymmetric case is also shown in Figure 26. Notice the difference in signal quality in the two plots. The symmetric case is much better than the asymmetric.

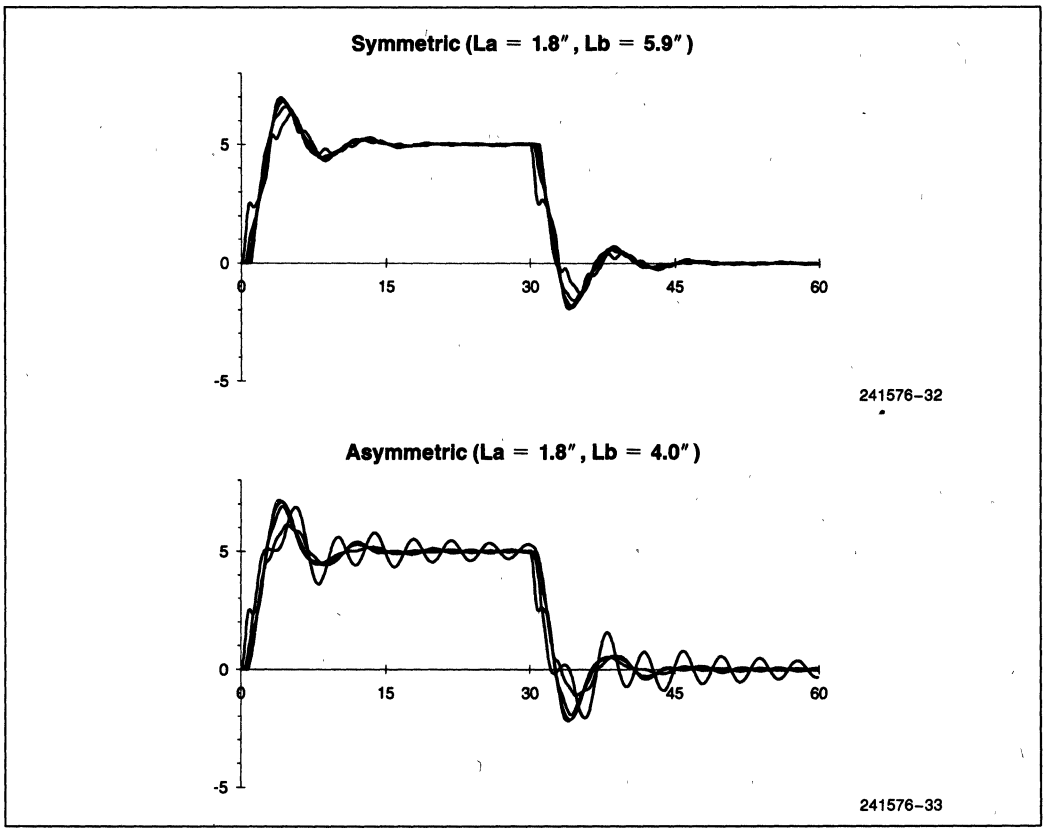


Figure 26. Symmetric Versus Asymmetric Values for La and Lb

This technique can be used to optimize the routing of all heavily loaded signals in a chip set design. From the energy factor plot rules can be defined to govern the segment lengths needed to minimize the energy factor and obtain the specified signal quality.

The 256K layout example that follows used this technique extensively to route the heavily loaded signals. For each signal group or topology, the asymmetry energy factor was calculated as a function of the topology's segment lengths and a set of rules defined to govern the segment lengths required to provide a routing that meets the signal quality specifications.

Similar results can be determined by using settling time to the optimal routing. To optimize the symmetry of a net, the settling time is plotted against line length. The minimum settling time occurs at the point where the net is balanced. Figure 27 shows a settling time plot for the net in Figure 24. Settling time is plotted against the true length for the segment between the Pentium processor and the 82496 for a given length between the Pentium processor and 82491. For $La = 1.8$ inches the settling time approach recommends $L_b = 5.8-6.0$ inches.

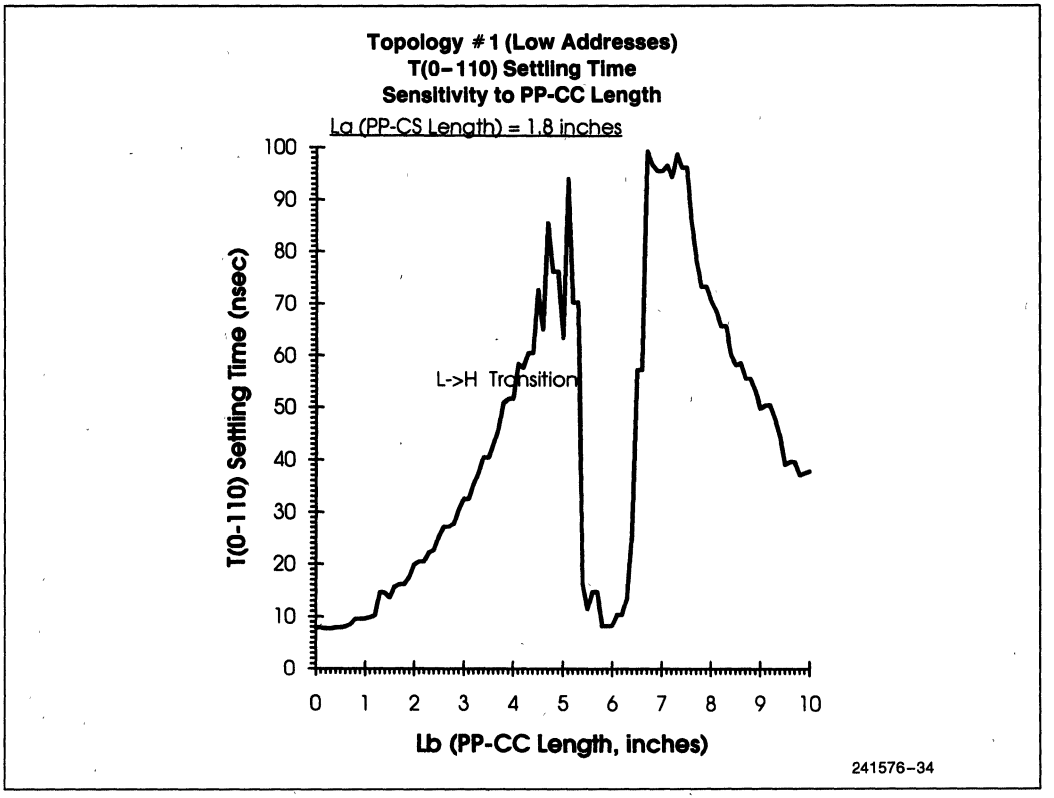


Figure 27. Settling Time Versus Line Length

5.2.3 SERPENTINE STRUCTURES

Serpentine structures are one design technique that can be used to assist in balancing the interconnect delays between the Pentium processor, 82496 cache controller, and 82491 cache SRAM components. The structure is used to add length to specific traces within the nets. The goal of adding this length is to make the net "balanced" or electrically symmetrical. In particular, this technique has been used to add length to the trace between the Pentium processor and 82496 cache controller so that it is electrically symmetric with the traces between the Pentium processor and 82491s of the same net as shown in Figure 28.

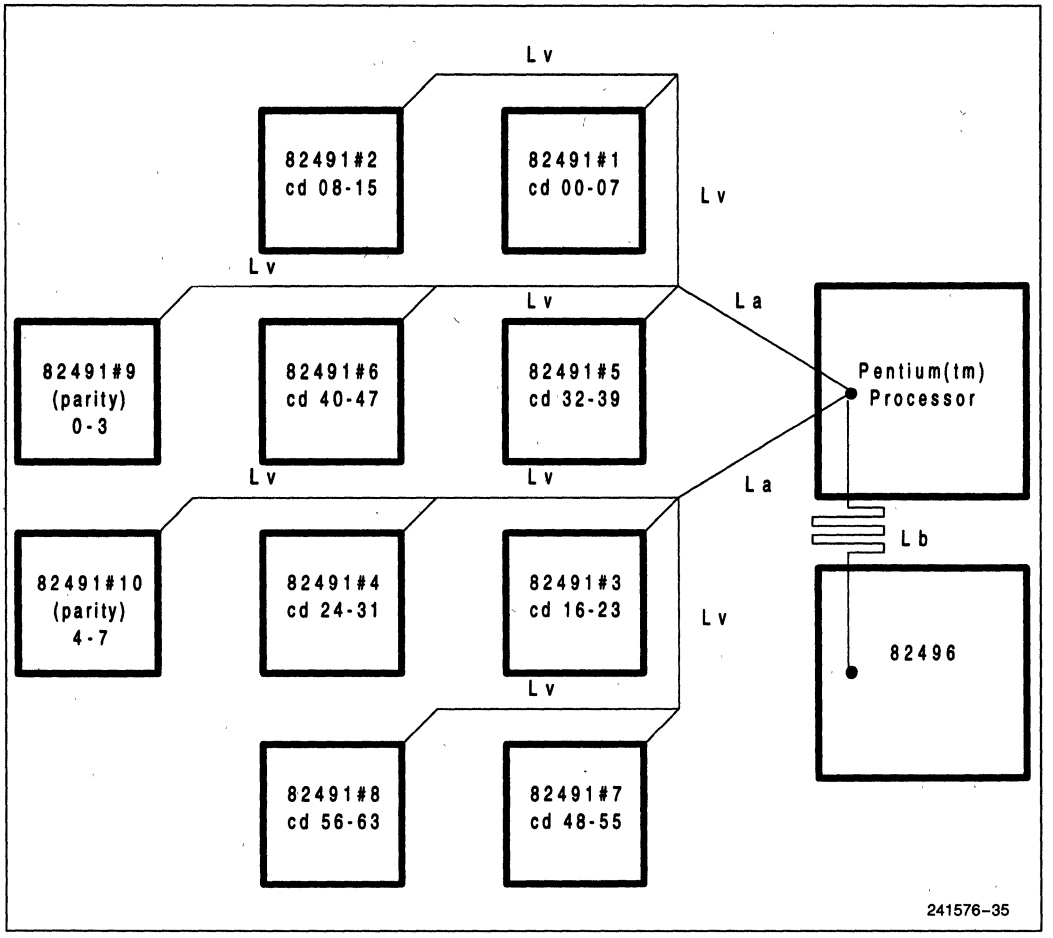


Figure 28. Balancing Nets Using Serpentine Structures

Due to the parallel traces that make up the serpentine, cross-coupling may occur between the individual portions of the serpentine. The cross-coupling may cause the propagation velocity and characteristic impedance of the serpentine to differ from those of a straight line of equivalent length. In general, the propagation velocity may be greater and the characteristic impedance less for the serpentine structure. To simplify the simulation environment for the CPU-cache-chip set design example, the added trace length was assumed to be equal to the length of trace used to make the serpentine. See Figure 29.

Experiments were performed to confirm that this assumption was valid. The experiments involved Time Domain Reflectometry (TDR) and Time Domain Transmission (TDT) measurements on various serpentine configurations. The height of the serpentine, h, and the separation, s, were varied.

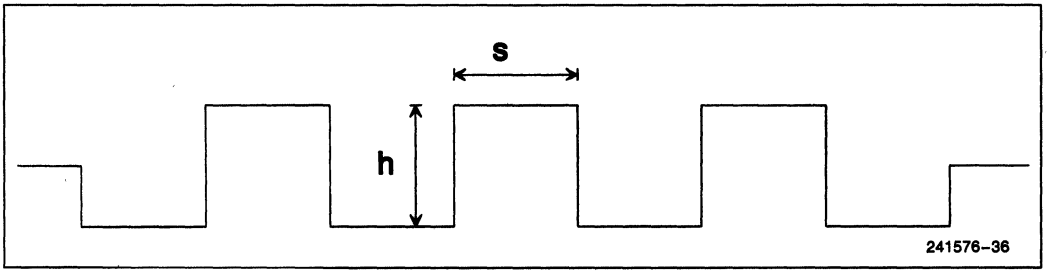


Figure 29. Parameters of a Serpentine Structure

It was found that as the separation was increased or the height decreased the propagation velocity increased. Amount of increase varied from being almost negligible to being approximately 40% for short, closely spaced serpentines. Also, it was observed that as the height decreases the magnitude of the decrease in impedance gets smaller, with the largest decrease in impedance, approximately 12%, seen when the height and separation are at their smallest. The serpentines used in the CPU-cache-chip set design example were not the worst case configuration. Based on the experiments, the serpentines should cause less than 10% decrease in the characteristic impedance and less than 30% decrease in the propagation delay.

Both of these variations appear considerable at first glance. However, serpentines, as used in the CPU-cache-chip set design example, account for only a small percentage of the entire trace length of a net. For example, if the serpentine is only 25% of the total trace length and the total propagation delay is 2 ns, representing the trace as a straight line length only introduces a maximum error of about 150 ps. This is determined by assuming the 25% of trace accounts for 0.5 ns delay and a 30% decrease is 150 ps. Based on the small amount of error introduced, it was decided that a straight line representation was accurate enough and simplified the simulations.

Note the variation in effects caused by the serpentines. Each design should perform similar analysis if a different serpentine structure than that used in the CPU-cache-chip set design example is used.

If the designer chooses to include the propagation velocity and characteristic impedance variations in the simulations, the only change is to represent the serpentine length of trace as a separate length with the different characteristics.

6.0 EXAMPLE: DESIGNING THE A12 NET FOR THE CPU-CACHE CHIP SET

The A12 net is one of the more complex nets in the optimized interface of the CPU-Cache Chip Set. The signal is driven by the Pentium processor to the 82496 cache controller and all the 82491 cache SRAMs during memory reads. In addition, the 82496 cache controller drives this signal to the Pentium processor during inquire cycles.

In routing A12 net all of the guidelines and techniques described were used. An initial routing of the A12 net was made using an H-type routing and attempting to make all of the interconnects as short as possible. The resulting topology is shown in Figure 30.

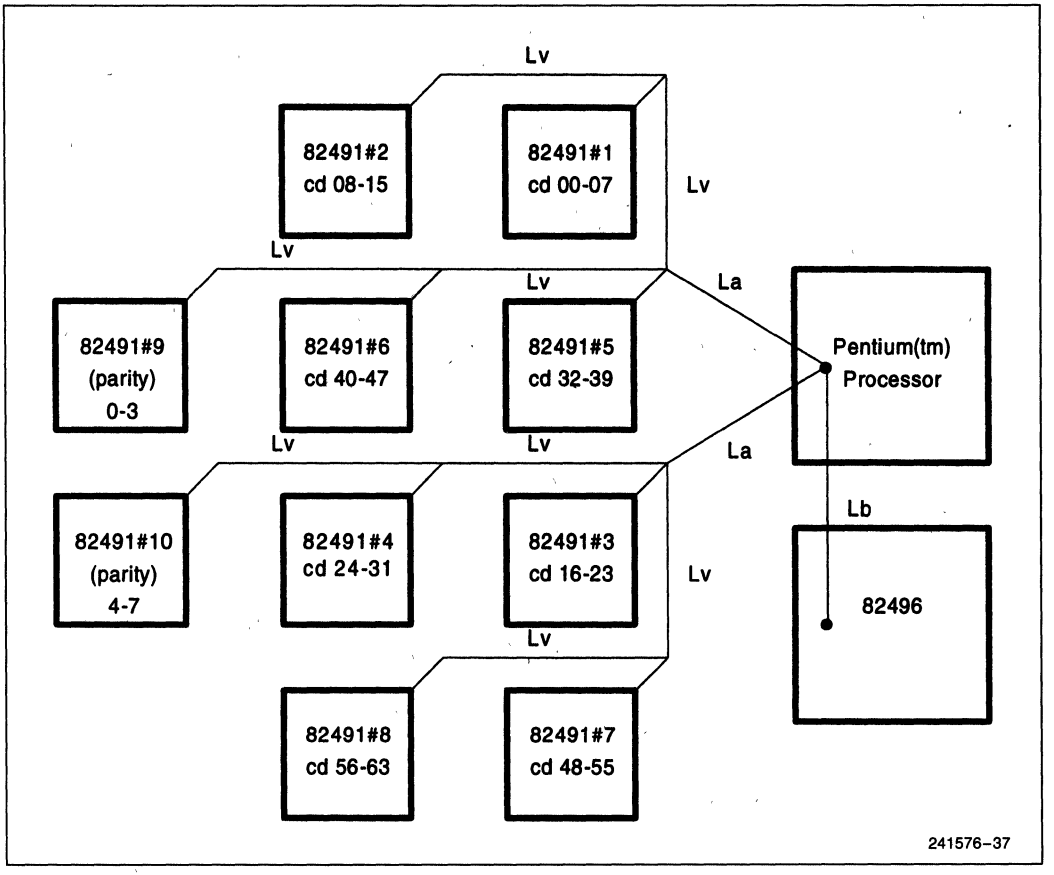


Figure 30. Initial Topology for A12 Net

The A12 net was simulated assuming the Pentium processor is driving the net. Quad Design's TLC was used to simulate the A12 net. For flight time the slow corner is used. The slow corner uses the model parameters as defined in Table 5. The actual values can be obtained from the *Pentium™ Processor User's Manual.* A TLC control file calls the appropriate model and topology files along with setting the needed measurement points to complete the flight time simulations.

Initially, the line lengths or segments between components was assumed to be the straight line distance. In other words, the initial routing conserved space and used the shortest line possible to connect the components. The rising and falling waveforms resulting from the TLC simulations of this routing are shown in Figure 31.

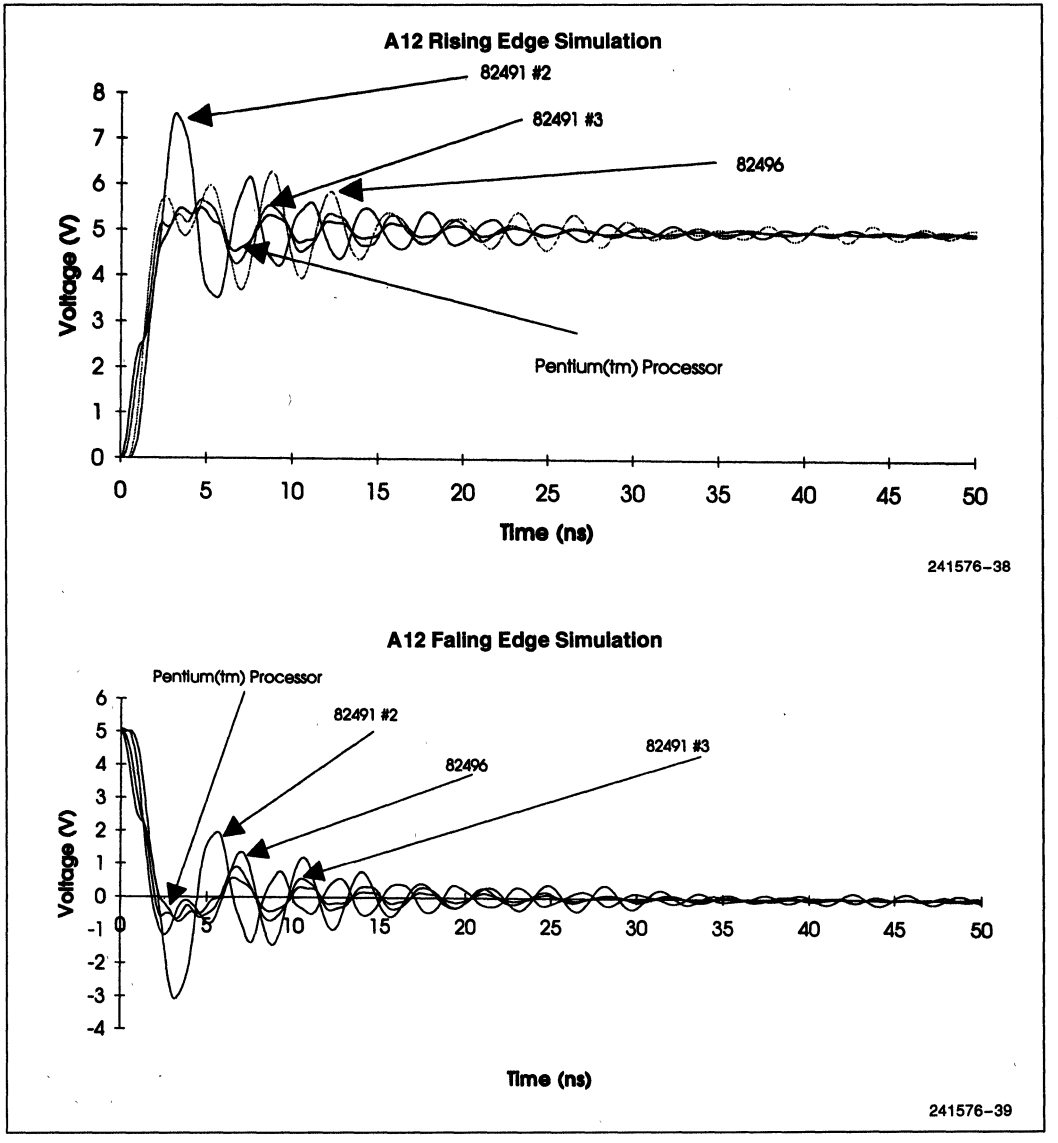


Figure 31. TLC Simulation of A12 Net Using Straight Line Lengths

Notice the large amount of ringing that occurs in this routing of the net. The excessive ringing can cause failures in both the signal quality and flight specifications. To bring the net within specification the routing must be improved to better "balance" the net. At first glance, the loading on the 82496 cache controller branch is much less than the 82491 cache SRAM branch. Splitting the 82491 cache SRAM branch into two branches and continuing the H-type routing along those branches and lengthening the 82496 cache controller branch should improve the "balance." The asymmetry energy factor, described in Section 5.2, was used to derive the relationship between individual trace segments needed to balance the net. The relationship is that:

$$Lb = 2 * La + Lc + 800 \text{ mils} ; La \leq 2000 \text{ mils}$$
$$1.6 * La + Lc + 1500 \text{ mils} ; La > 2000 \text{ mils}$$

Following these rules ensures that the A12 net is electrically symmetric. Figure 32 illustrates this improved routing.

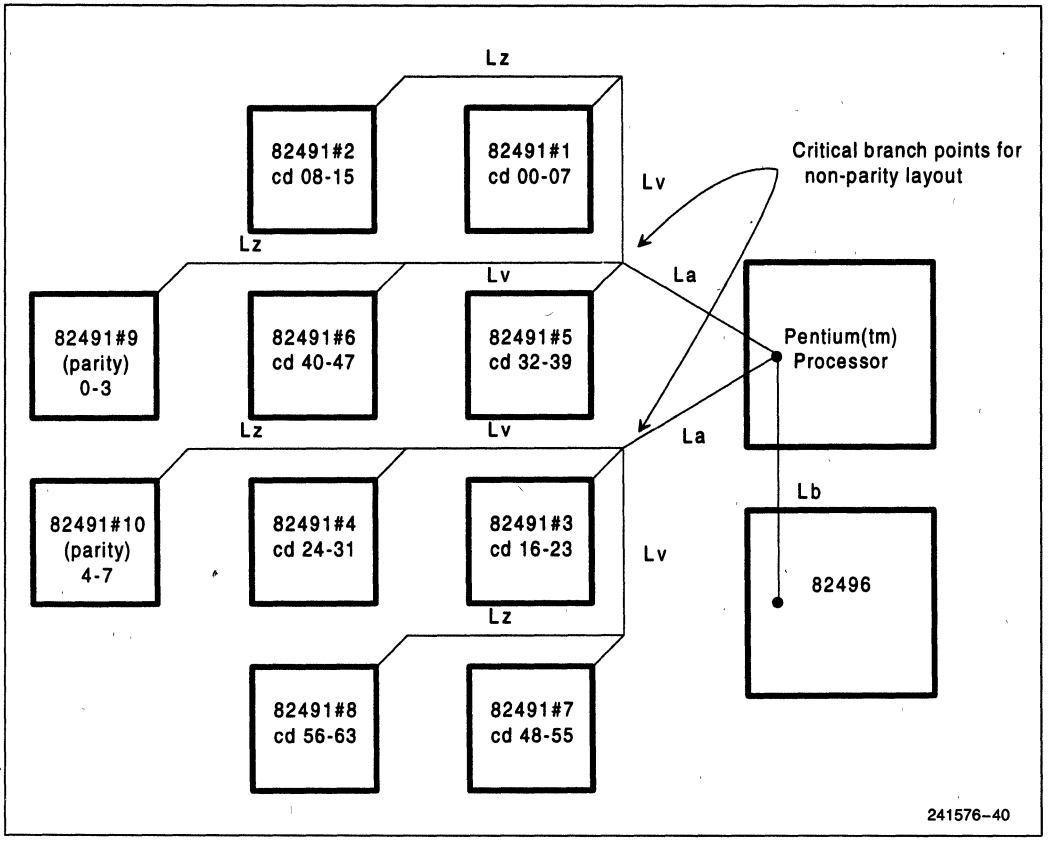


Figure 32. Improved Topology of A12 Net

The improved net was also simulated using TLC and the resulting waveforms are shown in Figure 33 through Figure 35. Notice the reduction in ringing. With the "balanced" or electrically symmetric routing the net exhibits better flight time and signal quality. In fact these parameters are now within specification as summarized in Table 7.

The technique for measuring flight time and signal quality are described in detail in Section 2.0. To measure flight time one must first determine the 50% point of the unloaded Pentium processor driver as shown in Figure 33. In the example this occurs at 2.26 ns. Next one must determine where the waveform crosses the 50% Vcc point at the receiver as shown in Figure 34. This crossing occurs at 4.54 ns. The time difference between these two points is the 50–50 flight time. The 50–50 flight time for the A12 net example is 2.28 ns.

Flight time also assumes the waveform continues through the 50% Vcc point with a slope of at least 1 V/ns through the 65% Vcc point. To ensure this, first determine the 65% point on the A12 flight time simulation as shown in Figure 35. The 65% Vcc point is 5.32 ns. Next extrapolate using the 1V/ns line to find where it crosses the 50% voltage level by subtracting 0.68 ns from the 65% Vcc number. The extrapolated 50% Vcc point for the example is 4.64 ns. The time difference between the unloaded buffer's 50% point and the extrapolated crossing of the 50% point is the 50–65 flight time. The 50–65 flight time is 2.38 ns.

The greater of the 50–50 and 50–65 flight times is the flight time for the net. In this case, the flight time is 2.38 ns, the 50–65 flight time.

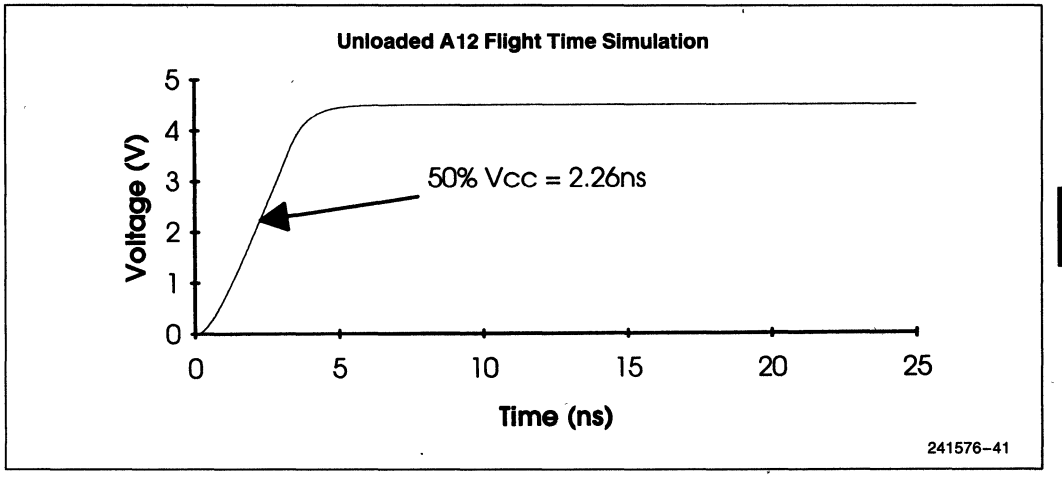


Figure 33. Measuring the 50% Vcc Point of the Unloaded Output

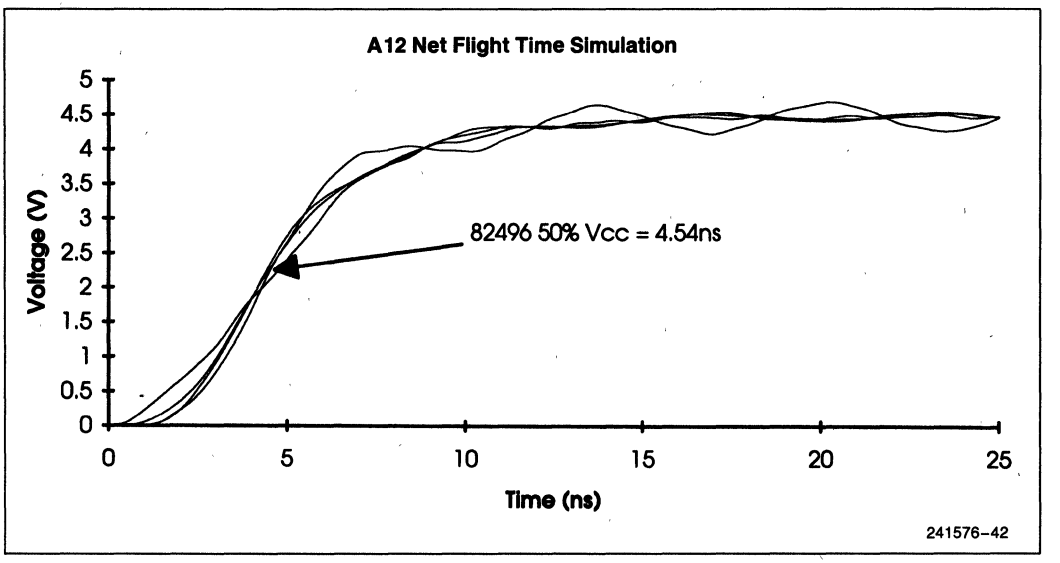


Figure 34. Measuring the 50% Vcc Point at the A12 Input of the 82496

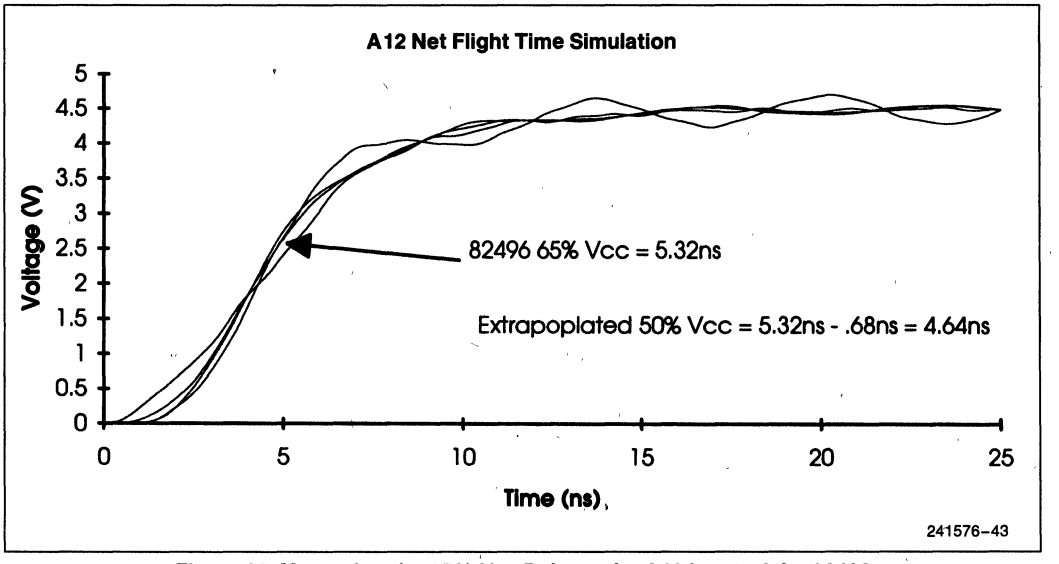


Figure 35. Measuring the 65% Vcc Point at the A12 Input of the 82496

Figure 36 and 37 illustrate the measurement of the signal quality parameters for the A12 net. Overshoot is the maximum voltage above Vcc. Time beyond supply is measured between points A and B. Ringback is the maximum voltage amount that the signal cross back across Vcc. Settling time is measured from point C and D.

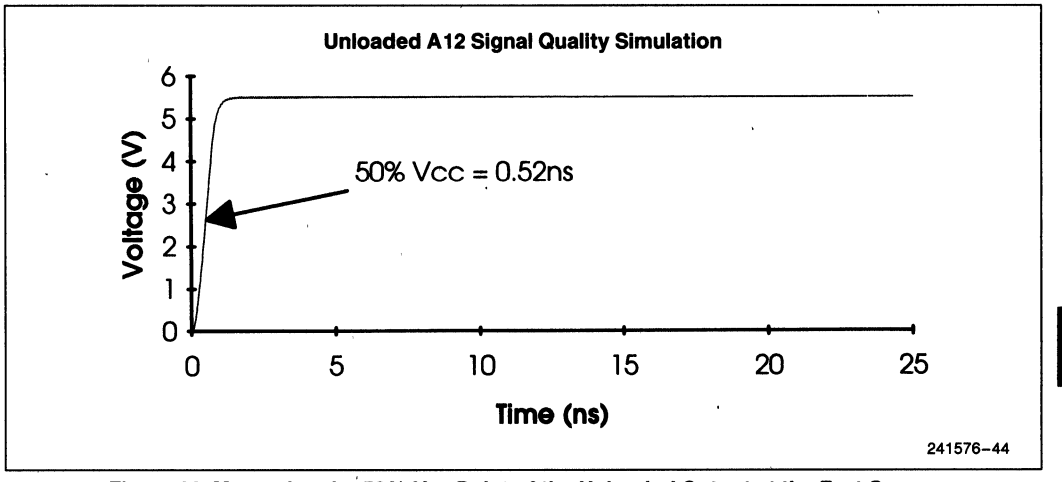


Figure 36. Measuring the 50% Vcc Point of the Unloaded Output at the Fast Corner

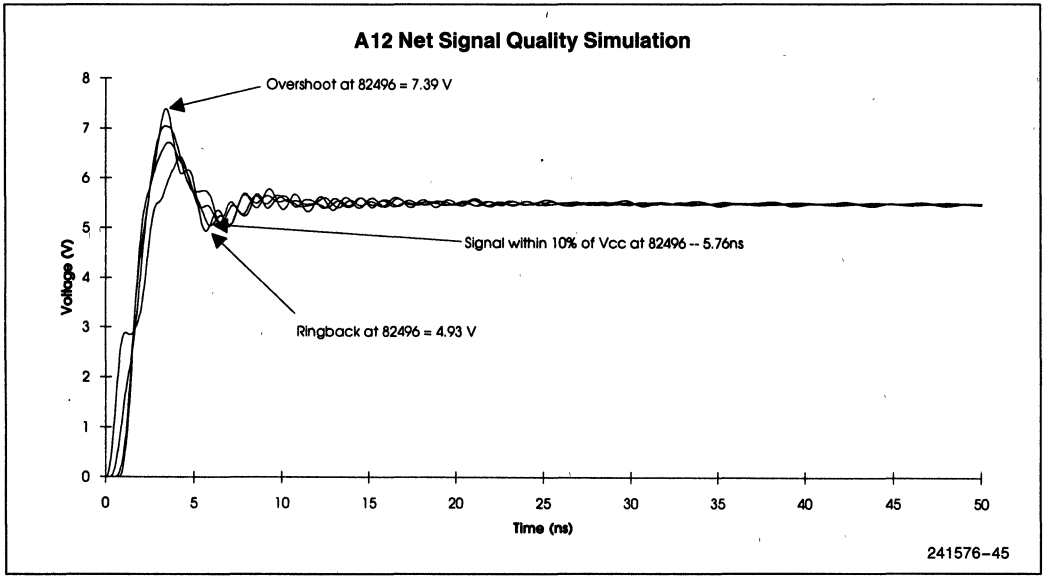


Figure 37. Measuring the Signal Quality of the 82496

The flight time and signal quality specifications for this net are listed in Table 7 along with the values measured in the simulation of the net.

Table 7. Flight Time and Signal Quality Simulated Values

Signal	Flight Time		Overshoot		Ringback		Settling Time	
	Spec	TLC	Spec	TLC	Spec	TLC	Spec	TLC
A12	28 ns	2.38 ns	3.0V	1.89V	35% Vcc	0.57V	12.5 ns	5.76 ns

7.0 256K CPU-CACHE CHIP SET OPTIMIZED INTERFACE LAYOUT DESIGN EXAMPLE

This chapter contains an example layout design for Intel's 256 Kbyte CPU-Cache Chip Set's optimized interface. Intel has simulated and verified the example layout using the latest information. Work is currently underway to validate the design by measuring the flight time and signal quality parameters on boards based on the design example. As updated information becomes available on the components and the boards, Intel plans to update this example accordingly.

The intent of the design example is to provide system designers a starting point. It provides one solution of how the Pentium processor, 82496 cache controller, and 82491 cache SRAM components can be placed and routed to ensure flight time and signal quality specifications are met. It is not the only solution. System designers can alter the layout to meet their system requirements as long as the flight time and signal quality specifications are met.

7.1 Layout Objectives

The 256K layout is an example of a CPU-Cache chip set arrangement that meets Intel's chip set specifications. The layout consists of 1 Pentium processor, 1 82496 cache controller, and 10 82491 cache SRAMs for a 256K second-level cache with parity. Although the layout is specifically designed for a chip set with parity, we will also discuss conversion to a non-parity layout.

This example layout follows the chip set's flight time and signal quality specifications. In addition to meeting those specifications, we had the following objectives:

1. To design the optimized portion of the interface so that the layout is not limited by interconnect performance. By not artificially creating any critical paths, the interface can yield maximum performance of the chip set.
2. To be consistent with EMI and thermal requirements.
3. To have the layout be used as a validation and correction vehicle by Intel. Intel will use the layout to validate the optimized interface of the chip set, measure flight times and signal quality, and tune input and output buffers.

Provided are complete specifications for a board layout: part lists, board layer plots, and the electronic files in Gerber format. Also provided are a set of topologies and line lengths so it will be easy to understand how the layout was generated.

7.2 Component Placement

To meet flight time with clock skew restrictions we placed the parts in relative proximity to each other. At the same time, we ensured that the layout's Memory Bus Controller (MBC) interface signals are routable. Figure 38 illustrates how the chip set components are placed in the layout example. The dot indicates the location of pin 1. Figure 38 component side view of the layout.

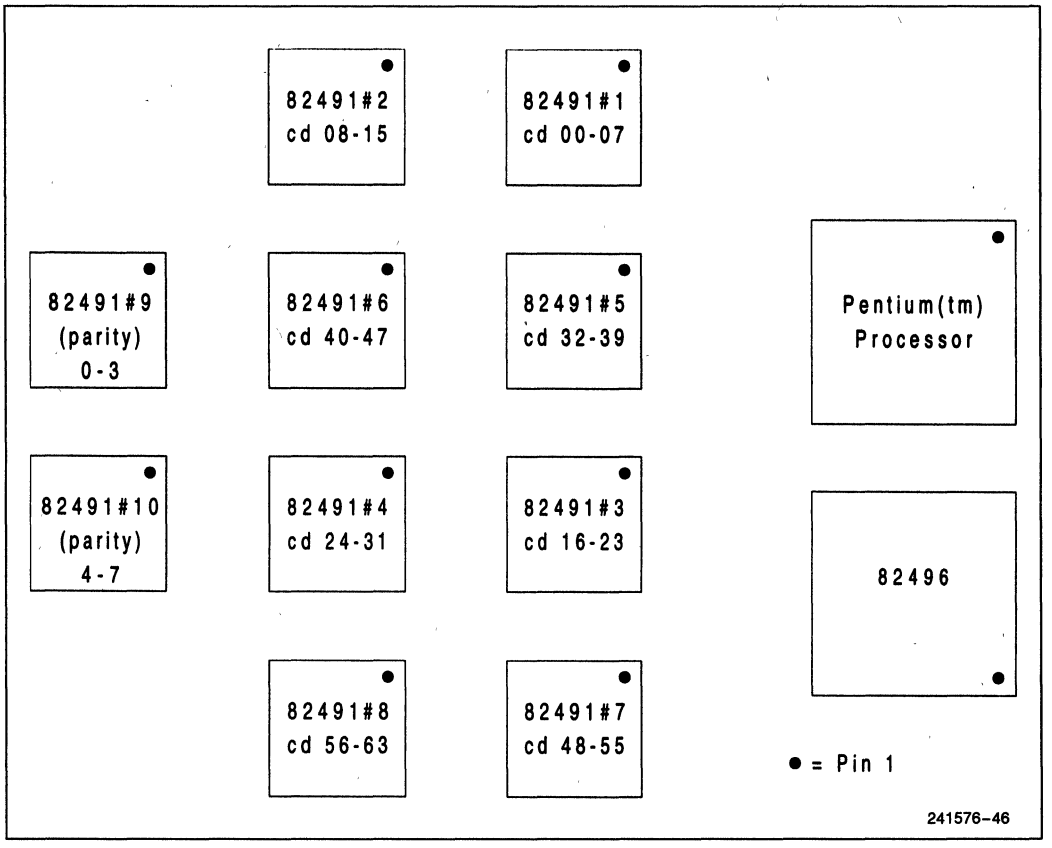


Figure 38. Component Placement

7.3 Signal Routing/Topologies

Tables 8 and 9 list the signal nets and their corresponding topologies for the optimized and external interfaces of the CPU-Cache Chip Set.

All chip set signals in the optimized interface fall into six groups: low addresses, high addresses, Pentium processor control, 82496 control, CPU data, and byte enables. Within each group are subsets of signals that share common origination and destination points. Each subset has a unique routing called a "topology." Groups, subsets, and topologies are listed in Table 8.

Topologies are given only for signals that are routed to multiple chips. It is the system designer's responsibility for routing the "point-to-point" signals such as CADS#.

Topologies are also supplied for the external interface. These topolgies provide channels for routing signals from the chip set components to the periphery where they can be connected to the memory bus and memory bus controller (MBC). However, topologies are not supplied for point to point signals in the MBC interface (e.g. CRDY#). Instead, the system designer must optimize these for the particular application.

Table 9 lists the topologies provided for the MBC interface signals which are not point to point.

Table 8. Optimized Interface Signal Net/Topology Assignments

Grouping	Routing Requirements	Topology
Low Addresses		
(PA3–PA16)	Bused to all core components. Must be routed to optimize delay and signal quality at all points.	1
High Addresses		
(PA17–PA31, PBT0–PBT3)	Point to point links. Must be kept as short as possible.	4
Pentium™ Processor Control		
(HITM#, W/R#)	Same as low addresses.	1
(ADS#)		3b
(ADSC#, AP, CACHE#, D/C#, LOCK#, M/IO#, PCD, PWT, SCYC)	Same as high addresses.	4
CC Control		
(BRDYC2#, WRARR#, MCYC#, WAY, BUS#, MAWEA#, WBWE#, WBTYP#, WBA, BLAST#)	Must be routed to optimize delay and signal quality at the CS.	3
(BLEC#)	Not routed to parity CSs.	3a
(BOFF#)		1b
(AHOLD, EADS#, KEN#, BRDYC1#, INV, EWBE#, NA#, WB/WT#)	Same as high addresses.	4
CPU Data		
(CD0–CD63)	Point to point signals. Keep as short as possible. Keep the total length of each trace within 1/2" of each other to minimize skew.	4
Byte Enables		
(CBE0#–CBE7#)		5

Table 9. External Interface Signal Net/Topology Assignments

Signal	Topology
RESETC50, CRDY0#	10
CRDY#, RESETC51	11
MBRDY#, MOCLK, MDOE#	12, 13
MFRZ#, MSEL#, MZBT#, MCLK	14
BRDY#, CLK0	15
CLK1, BRDY1#, MEOC1#	16
CLK2, BRDY2#, MEOC2#	17
CLK3, BRDY3#, MEOC3#	18
MDATA0–63	19

Figures 39 through Figure 58 are the topologies which are described in Tables 8 and 9. A topology is a graphical representation how specific sets of signals are routed. A topology shows the components that share a specific signal and the relative lengths of the traces between components.

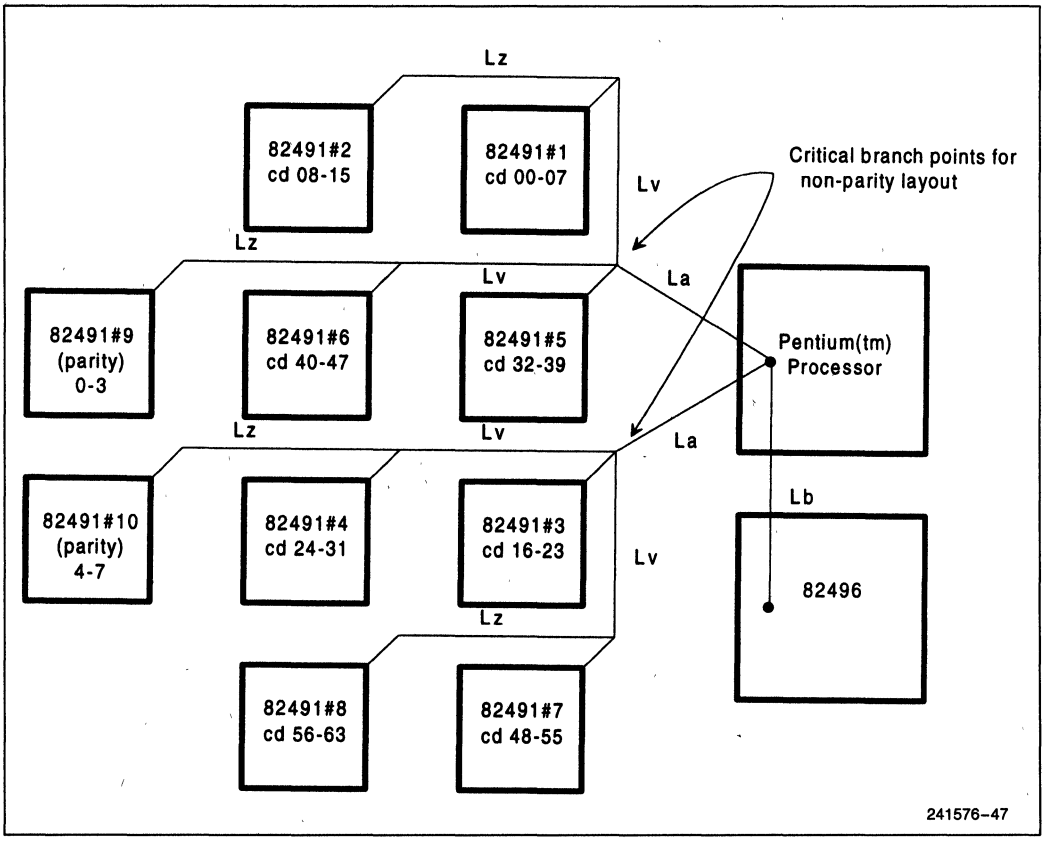


Figure 39. Topology 1

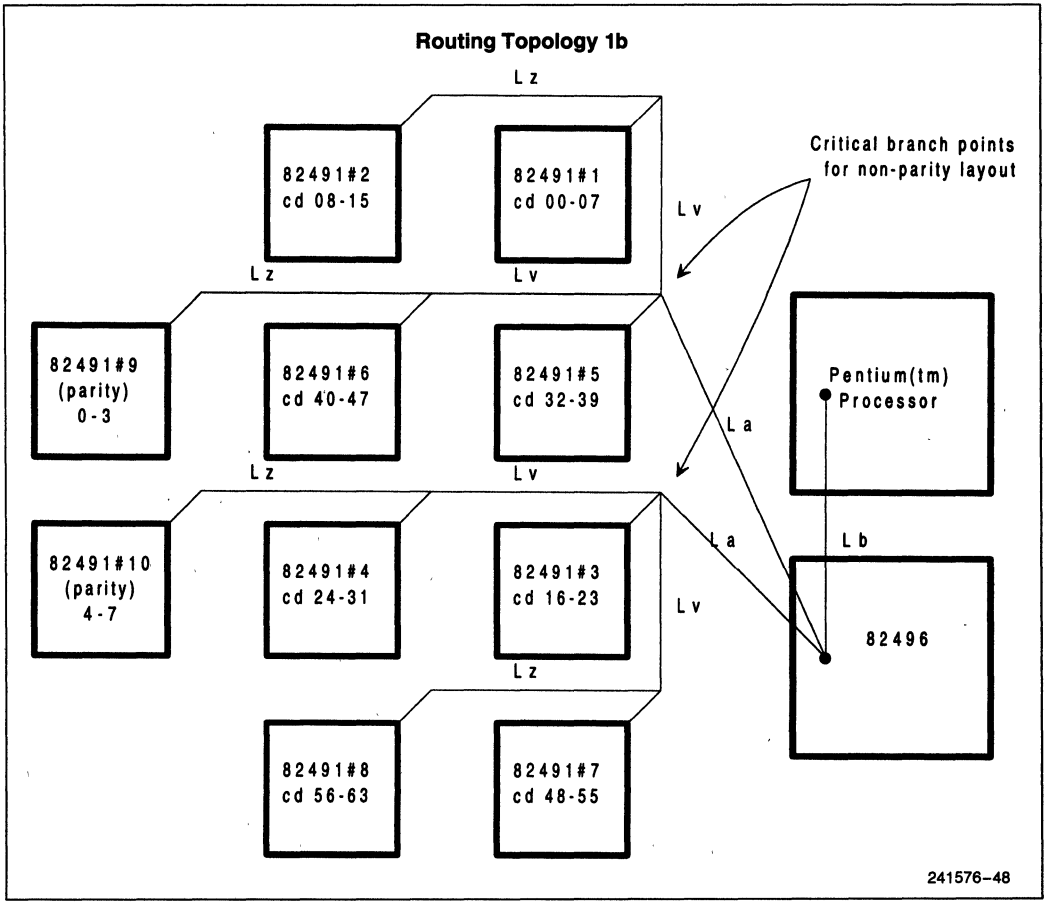


Figure 40. Topology 1b

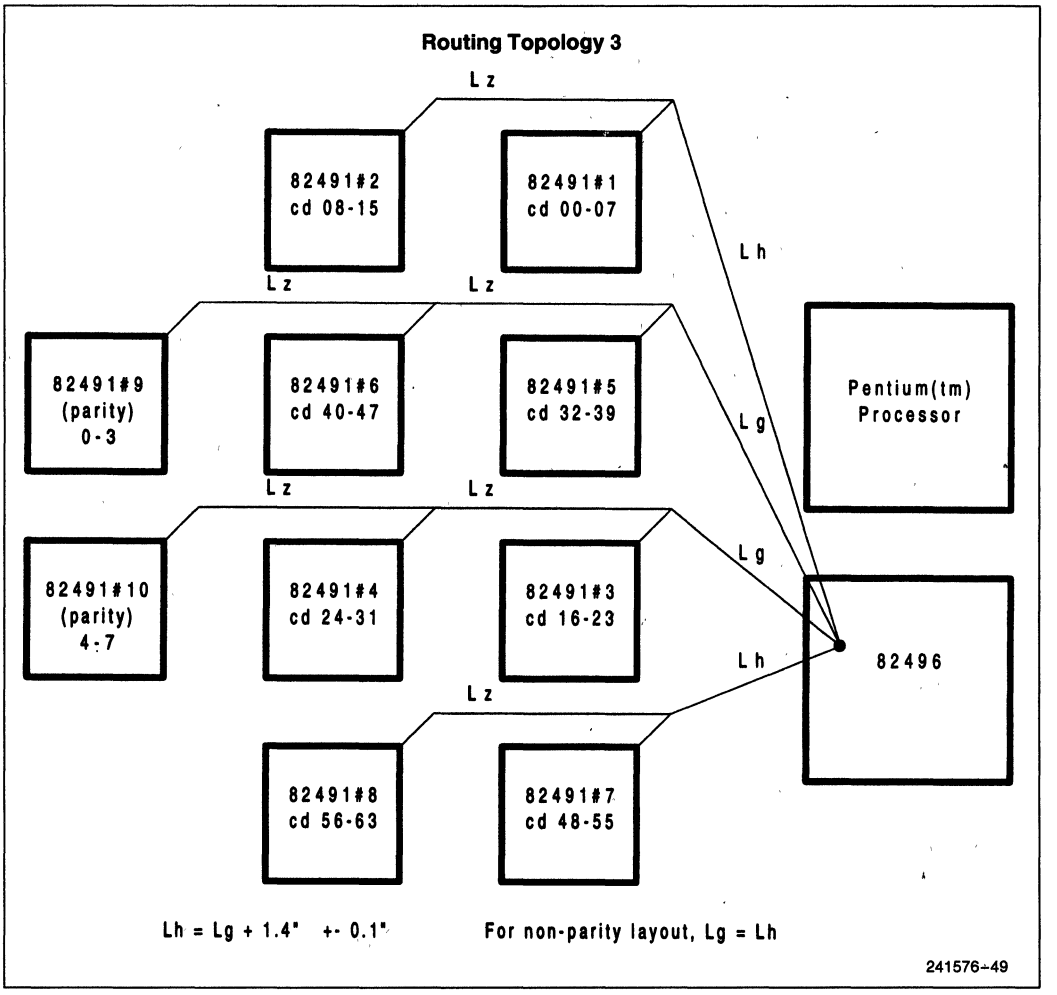


Figure 41. Topology 3

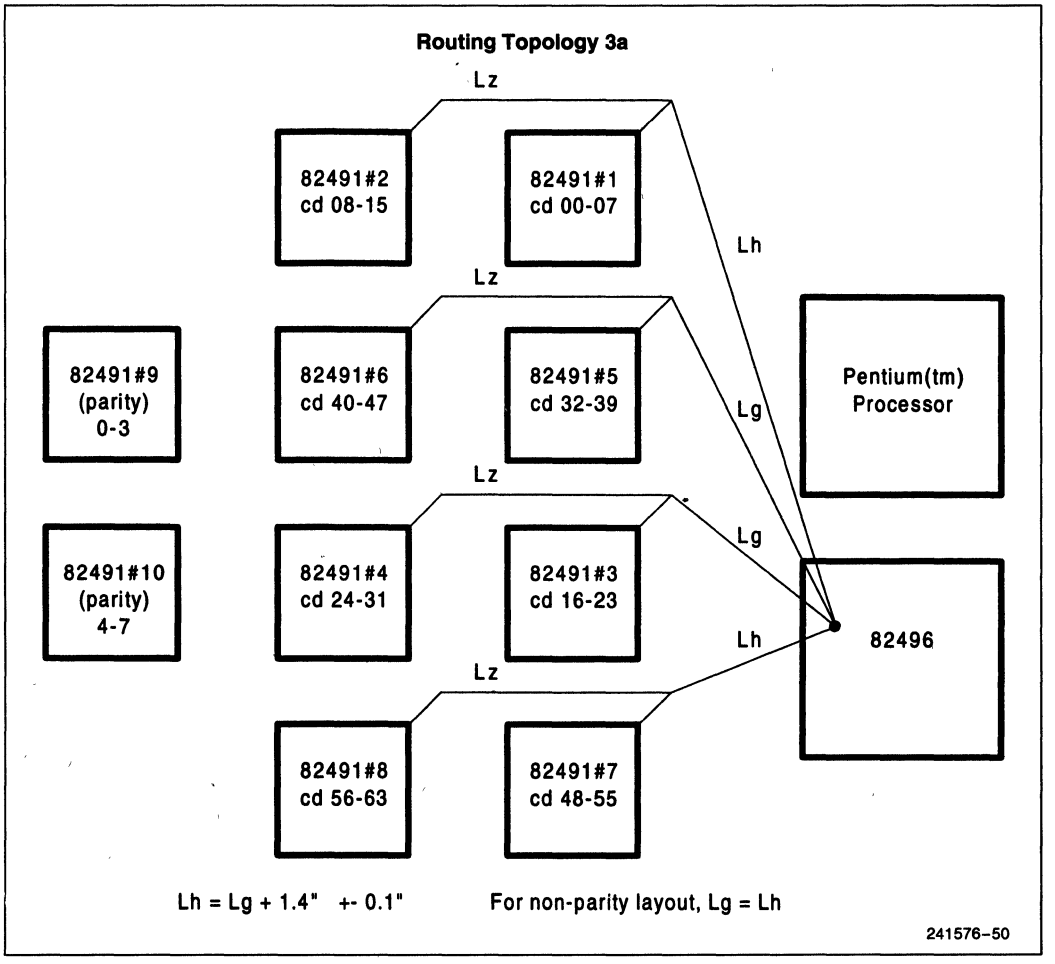


Figure 42. Topology 3a

2

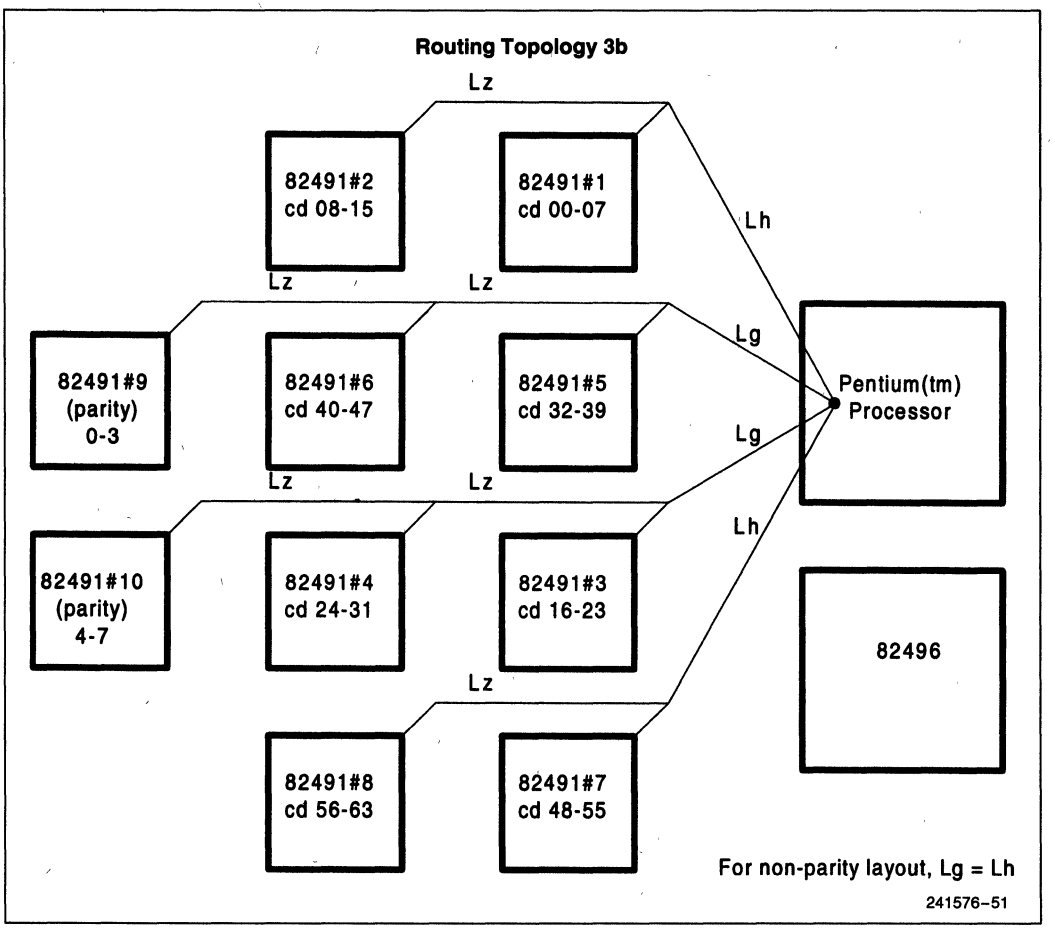


Figure 43. Topology 3b

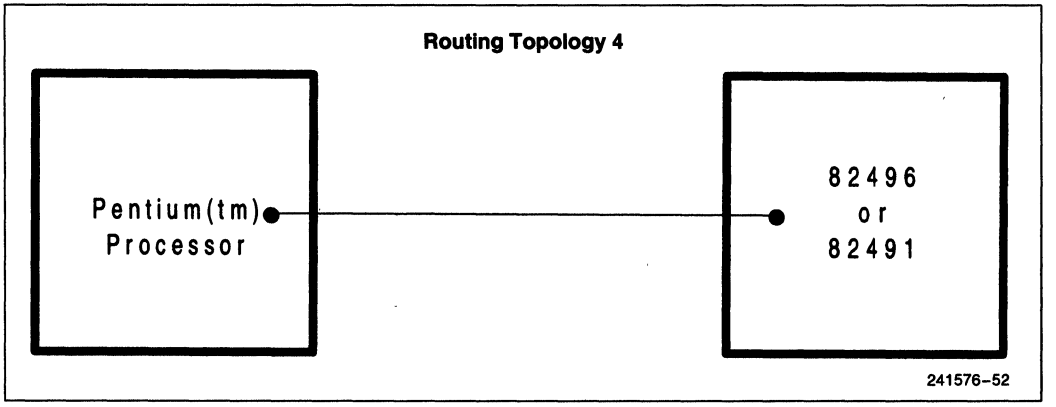


Figure 44. Topology 4

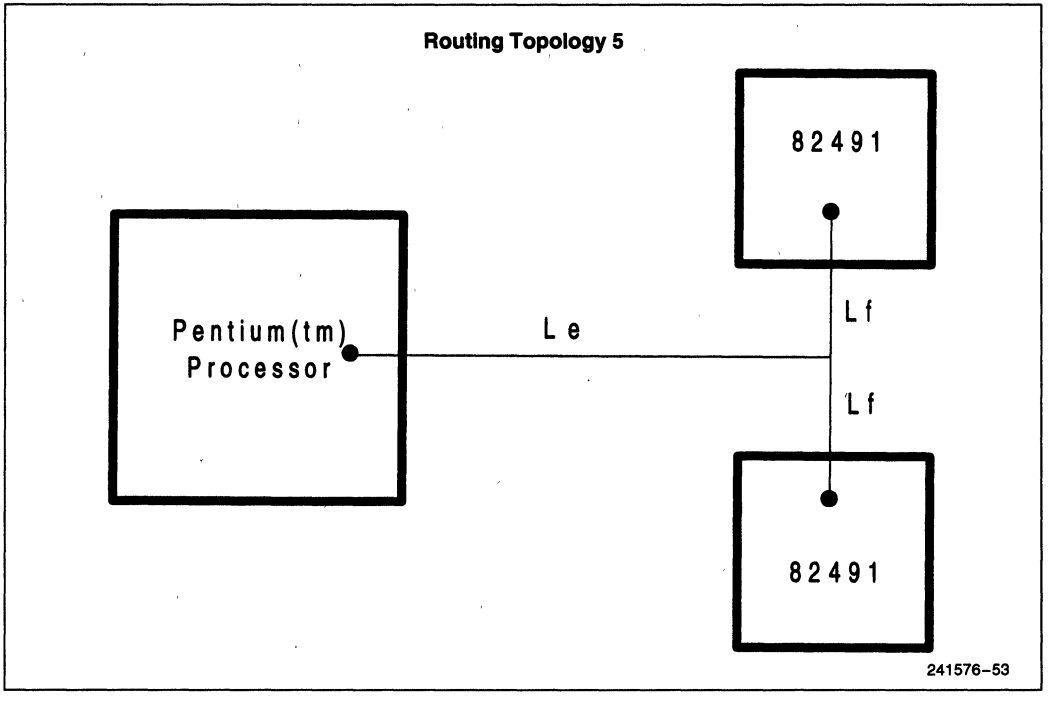


Figure 45. Topology 5

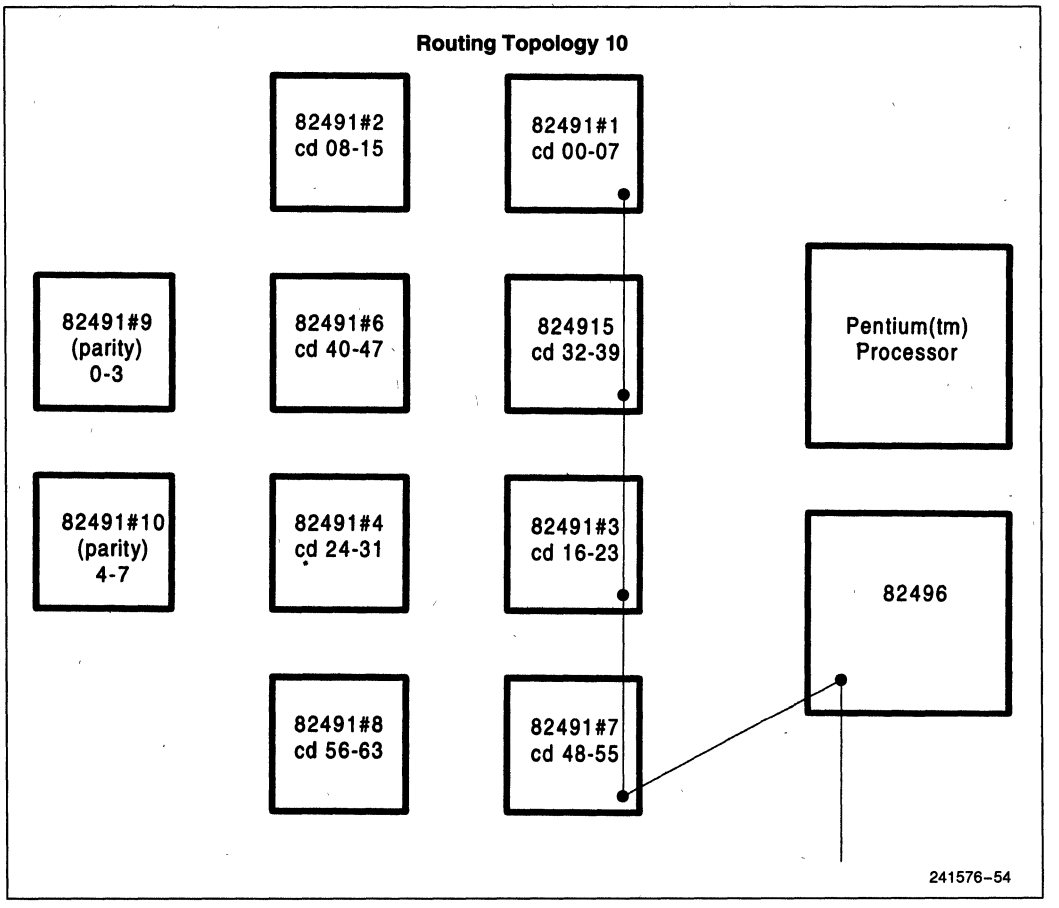


Figure 46. Topology 10

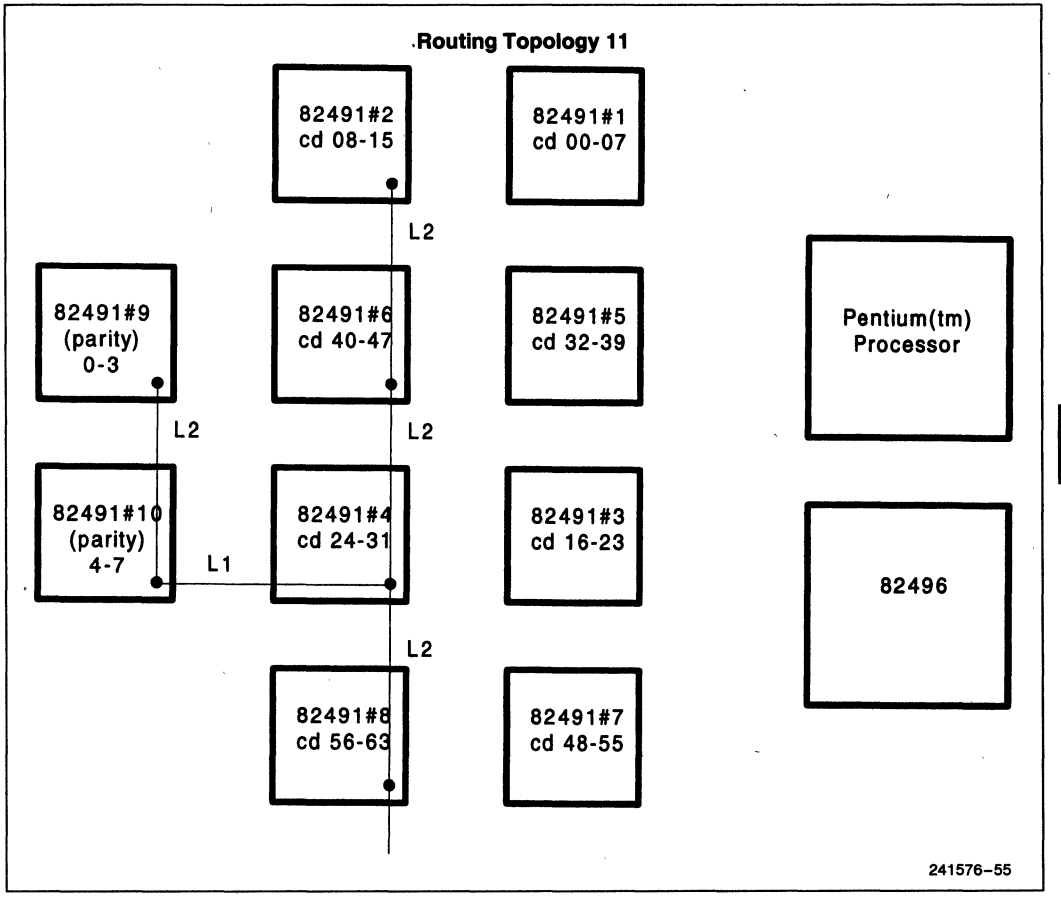


Figure 47. Topology 11

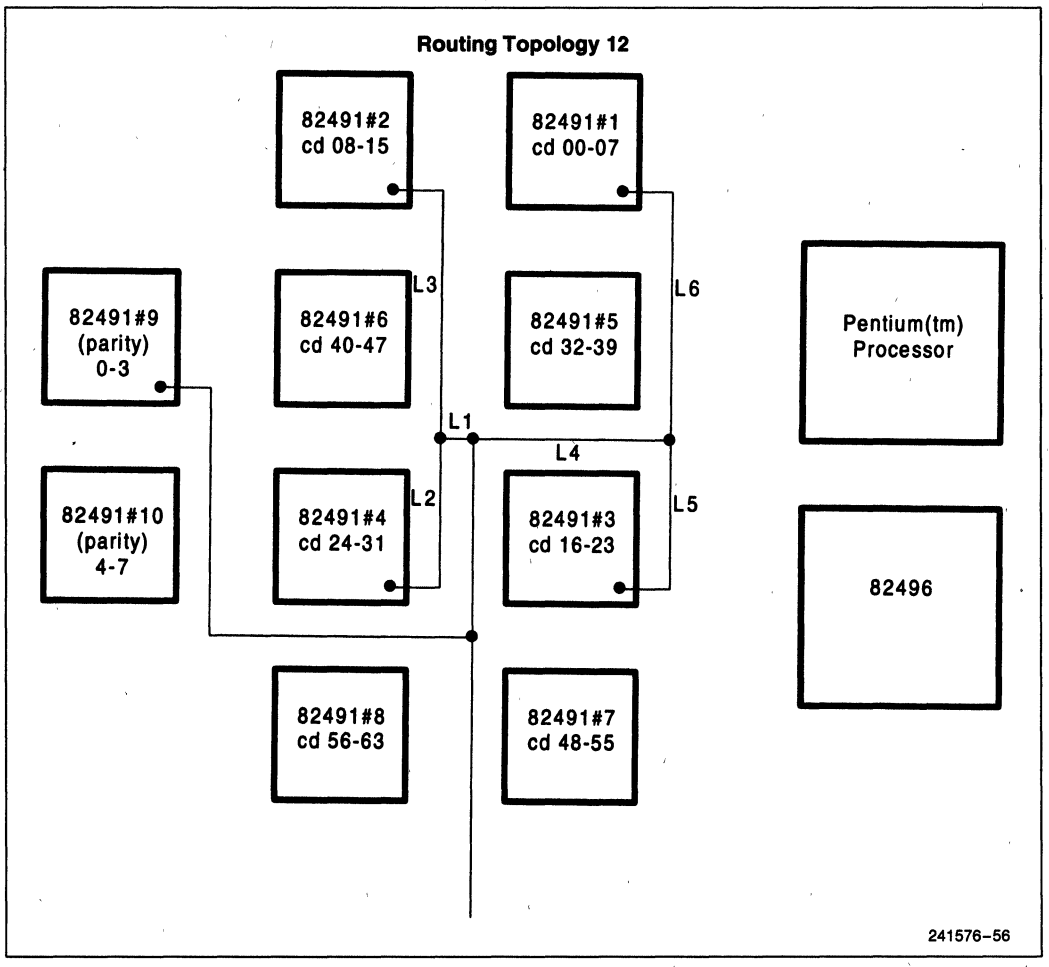


Figure 48. Topology 12

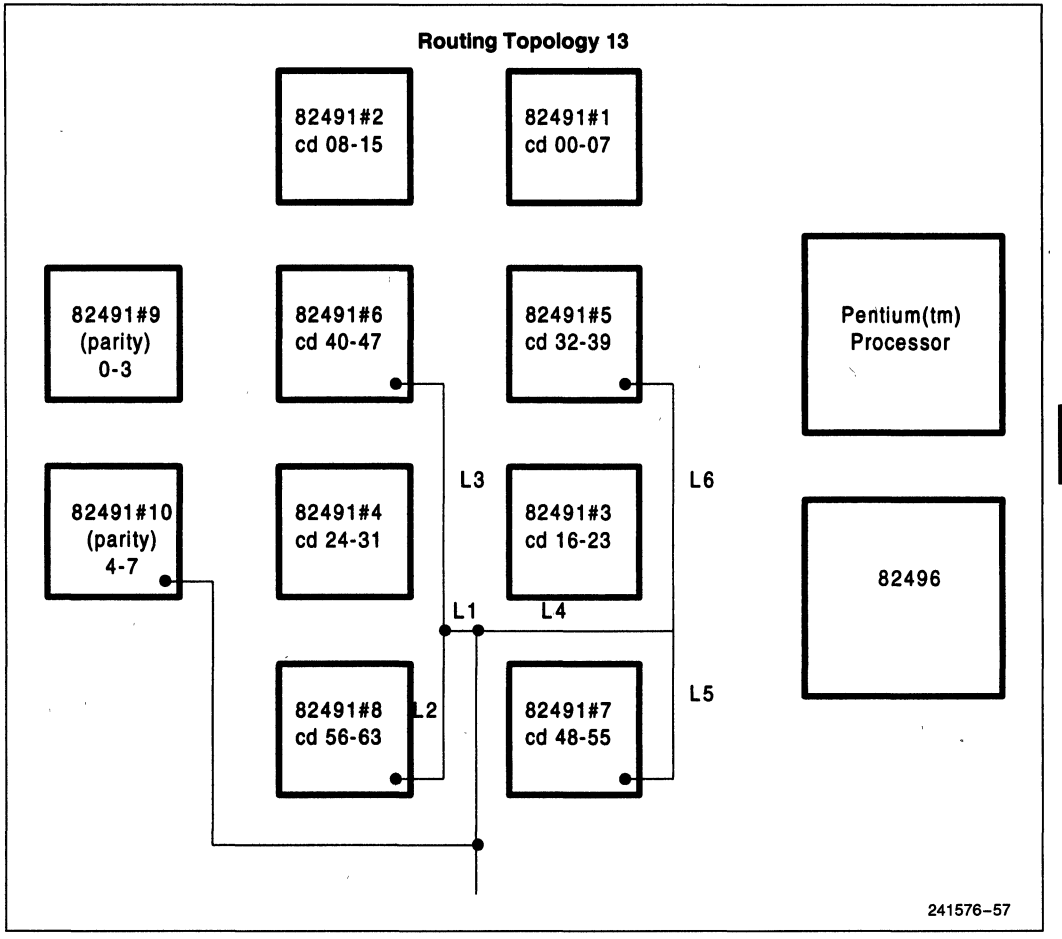


Figure 49. Topology 13

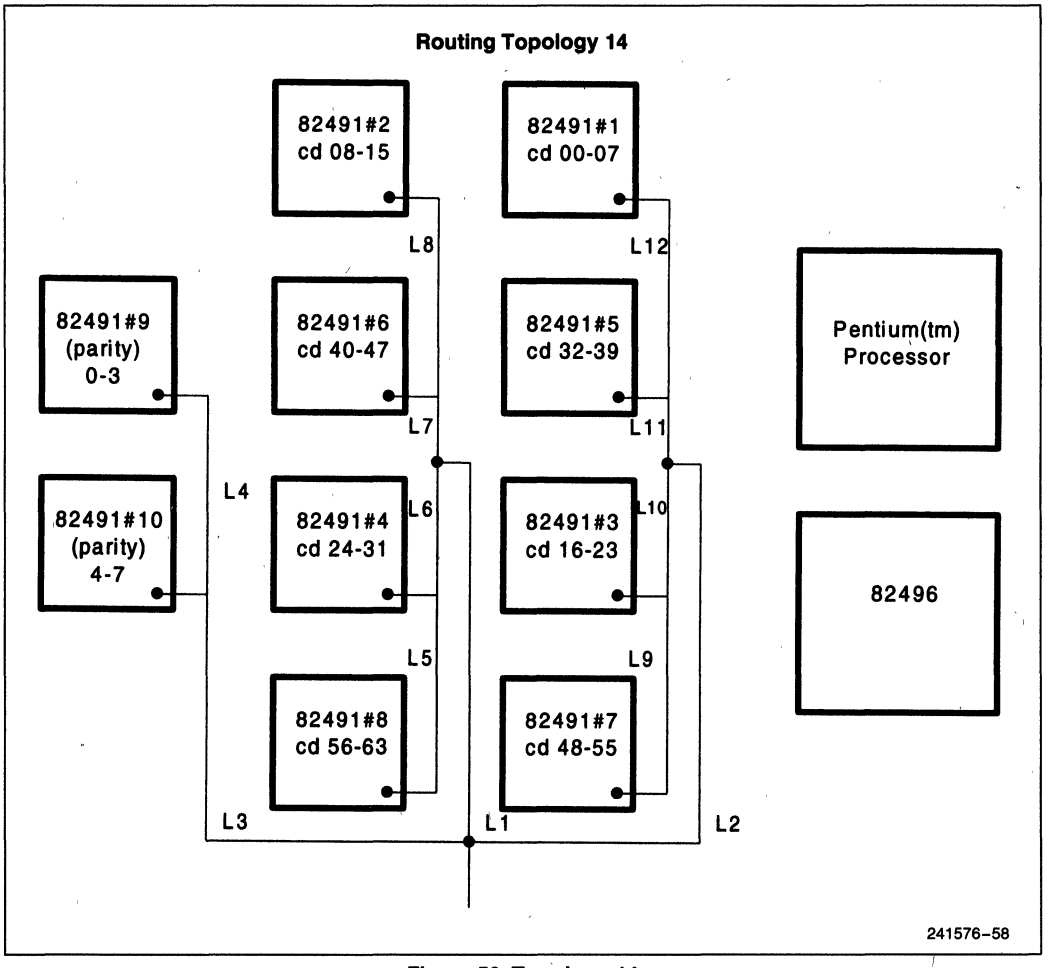


Figure 50. Topology 14

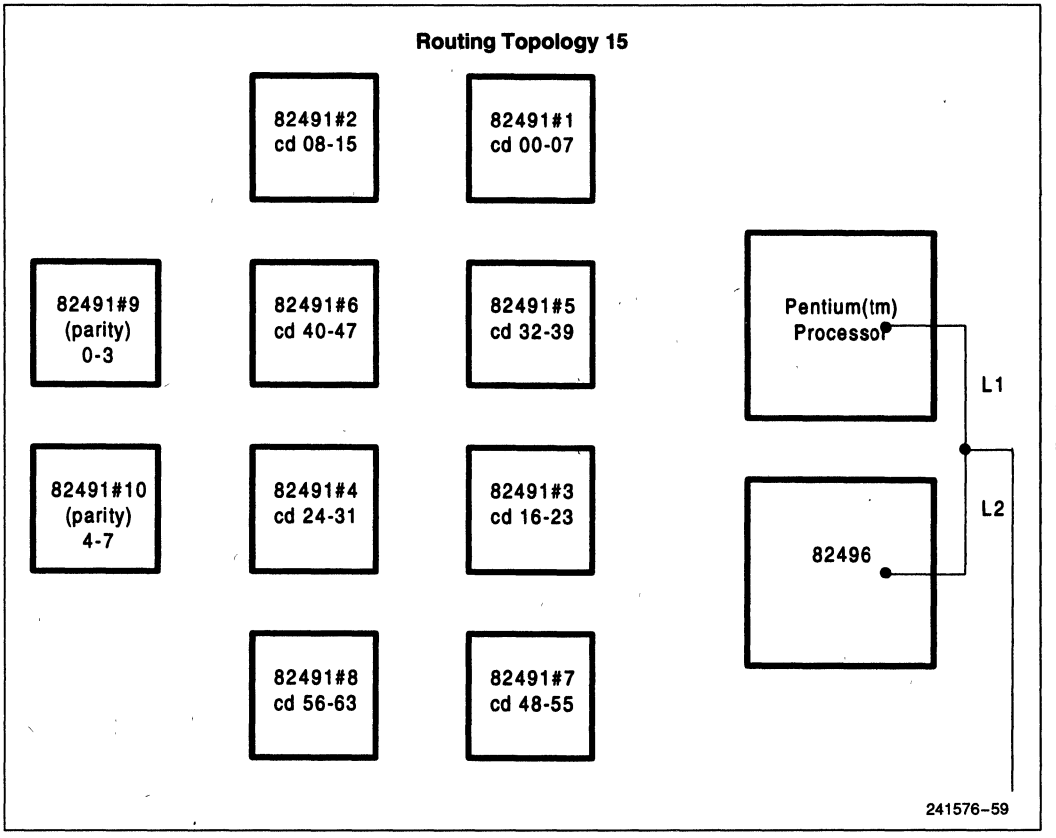


Figure 51. Topology 15

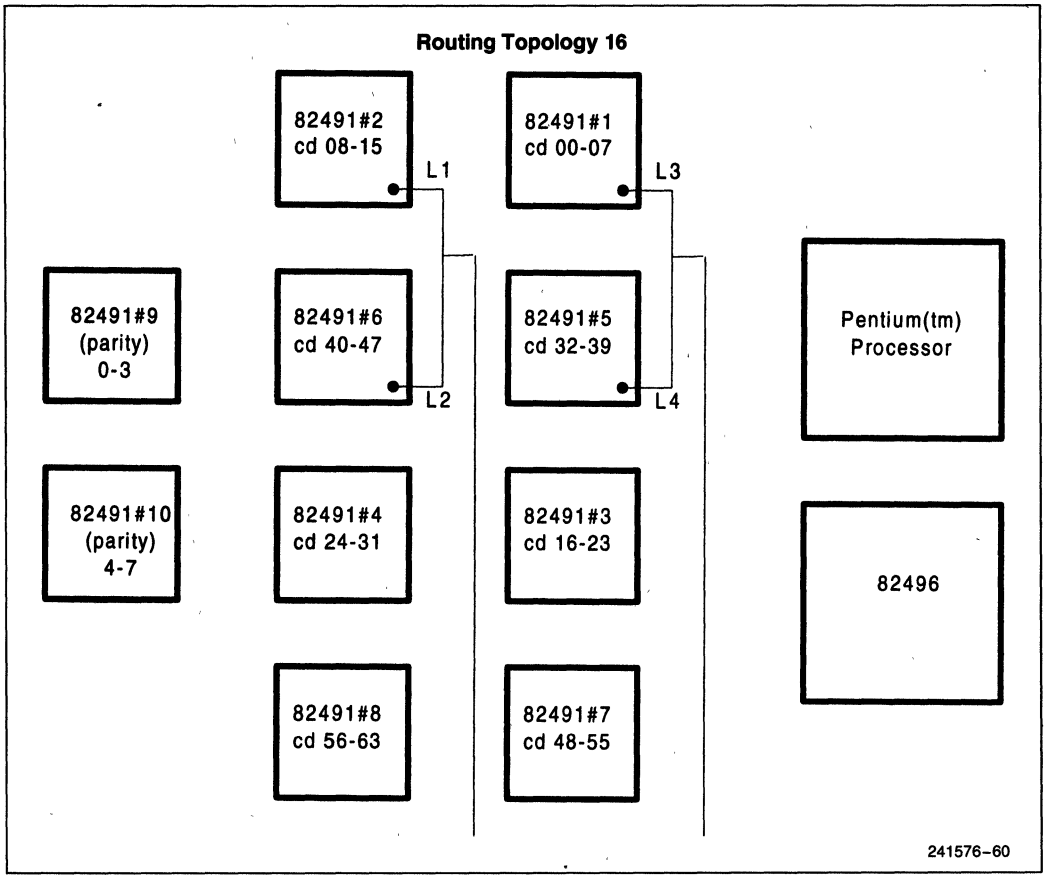


Figure 52. Topology 16

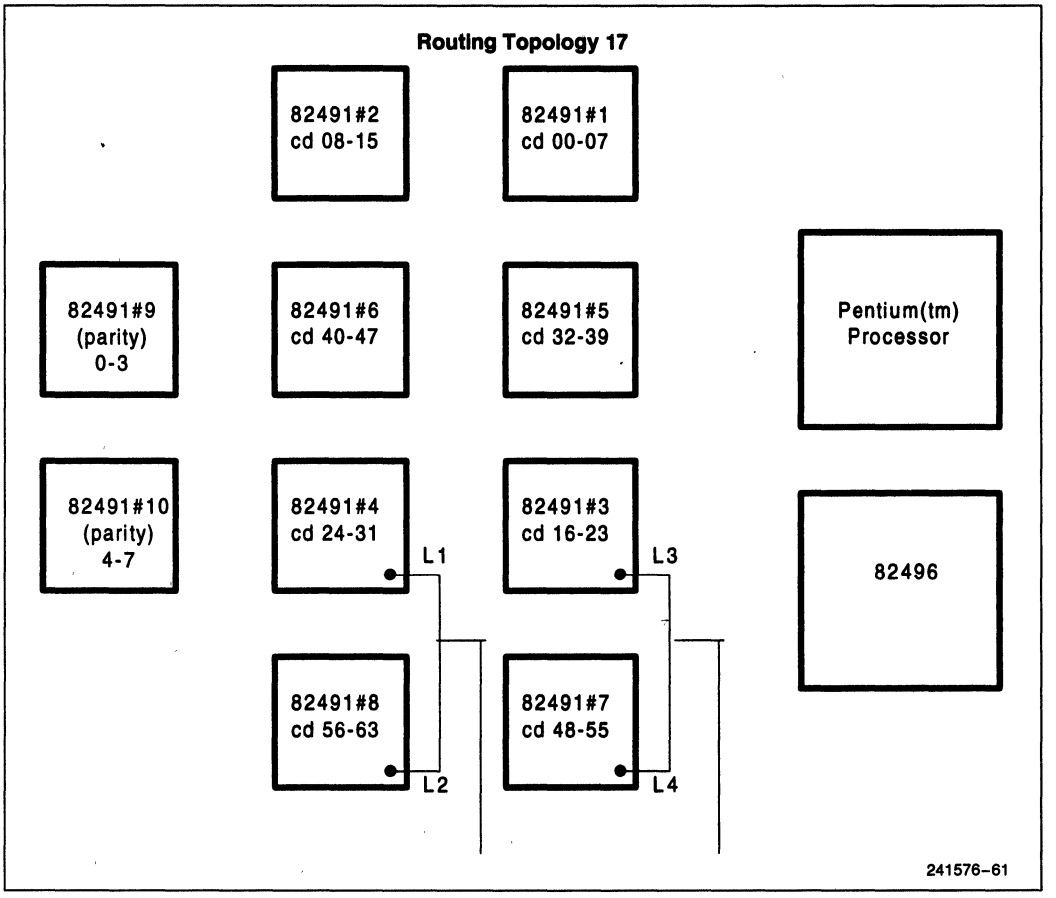


Figure 53. Topology 17

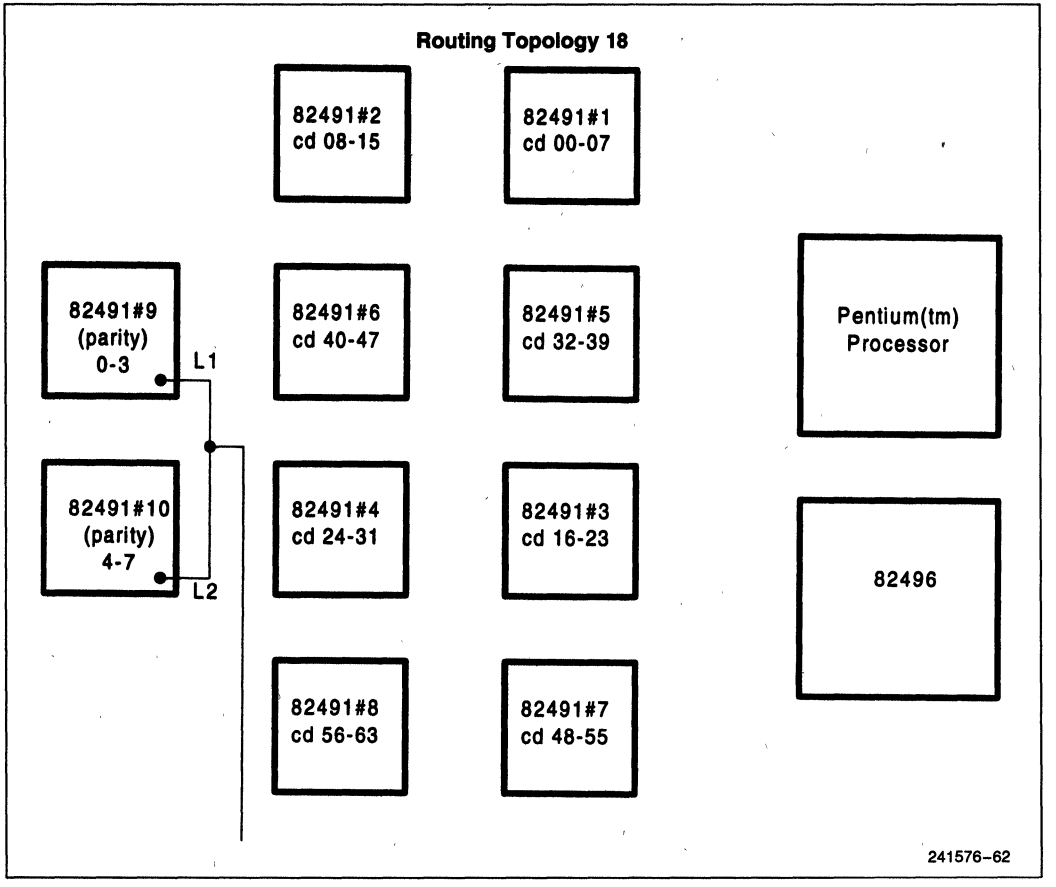


Figure 54. Topology 18

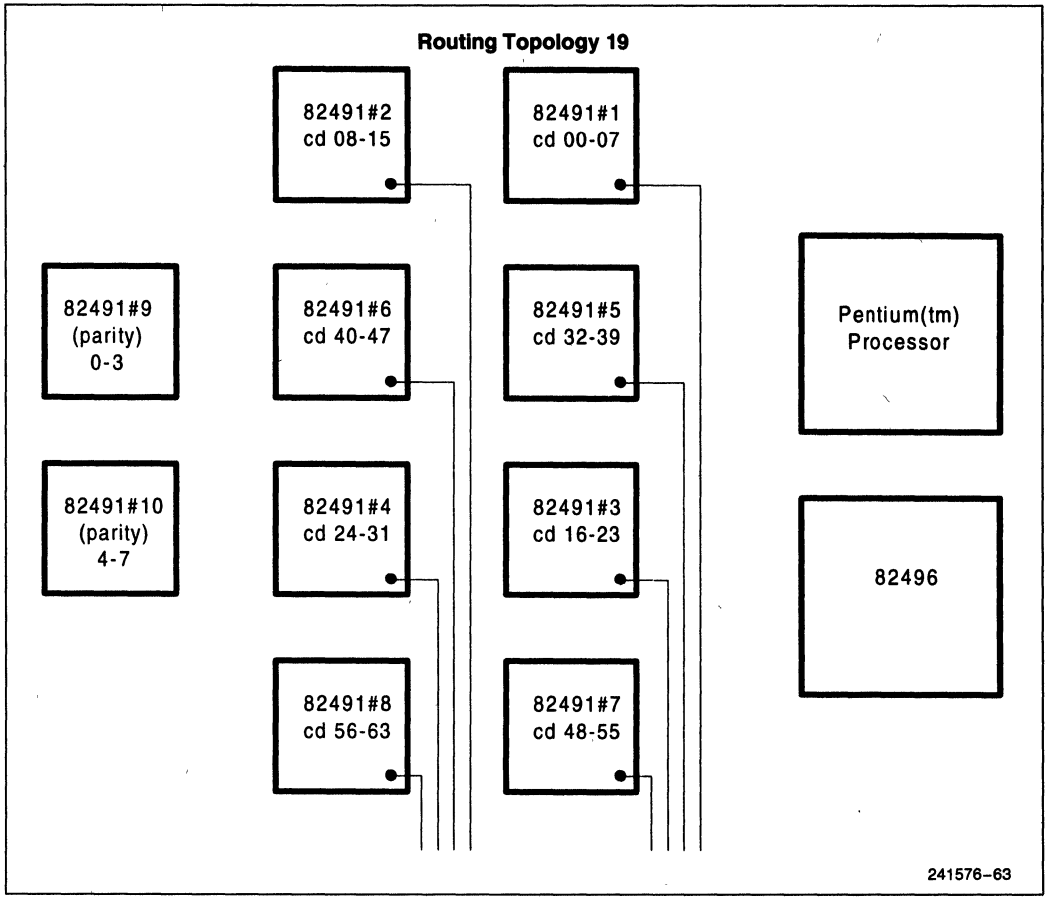


Figure 55. Topology 19

Figures 56 and 57 provide topologies for the non-parity configuration of the 256 Kbyte CPU-Cache Chip Set. Refer to Section 7.7.1 for more details on the non-parity configuration.

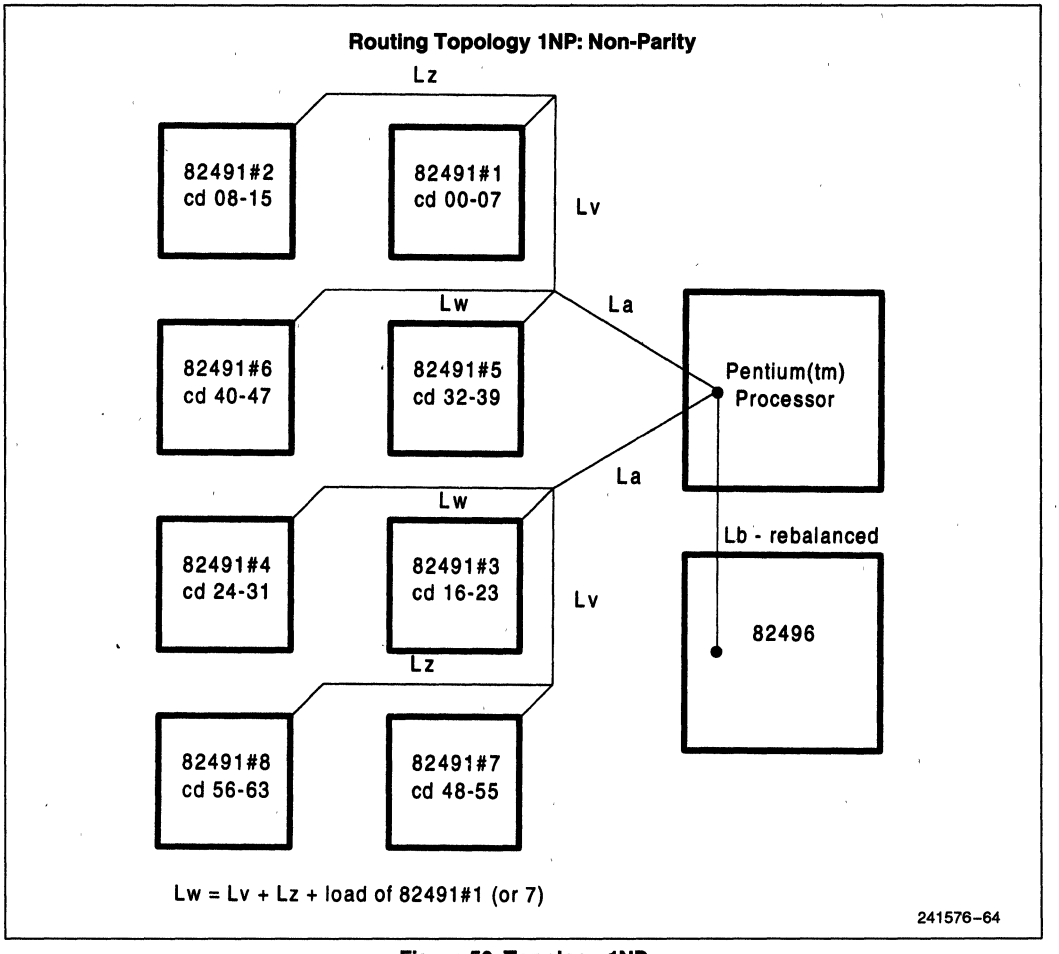


Figure 56. Topology 1NP

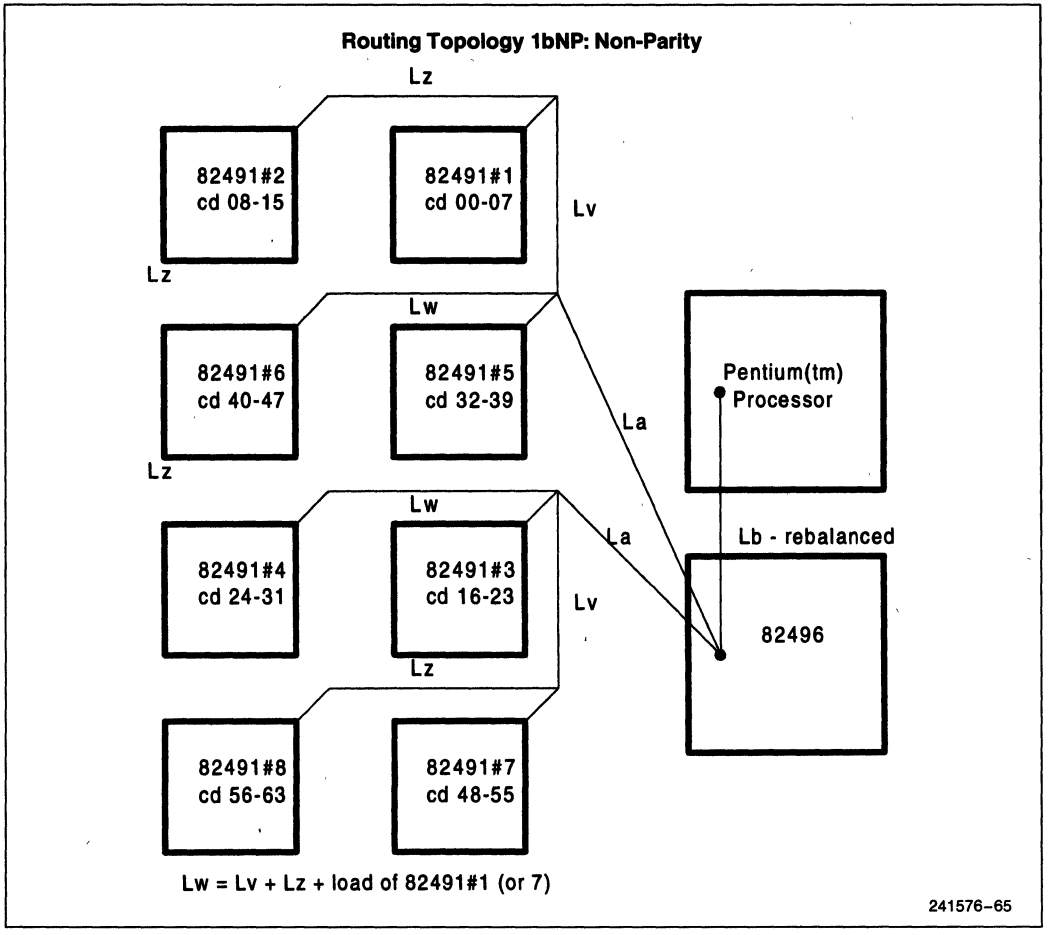


Figure 57. Topology 1bNP

7.4 Board/Trace Properties

Specific board and trace properties were assumed while performing the simulations to optimize the chip set layout. These properties were used as the specification or guideline the board manufacturer was to use in building boards. Figure 58 provides the board layer stackup.

Table 10 lists the minimum and maximum trace characteristics. These parameters along with the board material determine the spacing between layers and the total board thickness. See Table 11.

Only the inner layers of the board are impedance controlled. The top and bottom layers are not impedance controlled.

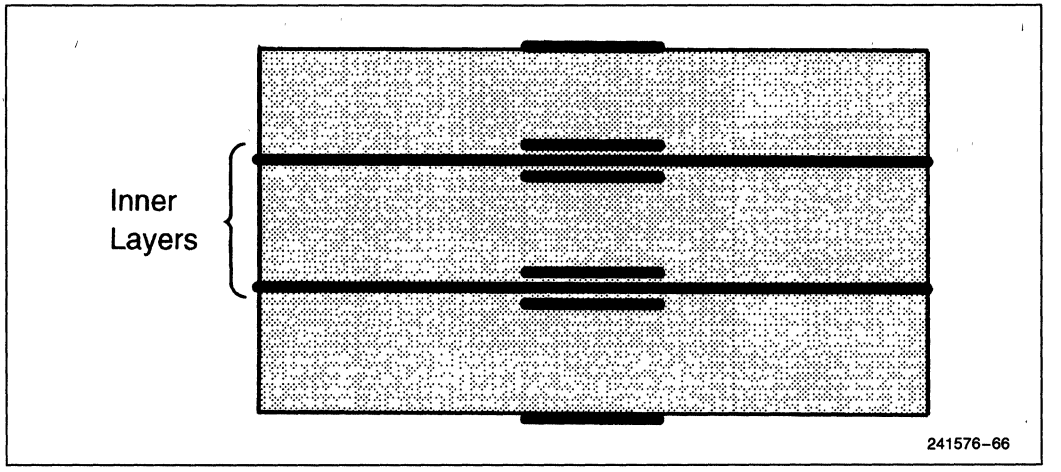


Figure 58. Board Layer Stackup

Table 10. Trace Characteristics

	4 Inner Layers	2 Outer Layers
Width/Space	5/5 Mils	8/8.5 Mils
Z_0	65Ω ± 10%	90Ω ± 20%
Velocity	1.85 to 2.41 ns/ft	1.35 to 2.05 ns/ft

Table 11. Other Printed Circuit Board Geometries

Via Pad	25 Mils
Via Hole	10 Mils
PGA Pad	55 Mils
PGA Hole	38 Mils
Layout Grid	5 Mils

7.5 Design Notes

The following design notes accompany this layout example:

1. The layout did not specifically address heat dissipation except to allow space for heat sinks to be attached. Please see the *Pentium™ Processor User's Manual* for the devices' thermal specifications. The *Pentium Processor Thermal Design Guidelines* application note provides some examples of possible thermal solutions.
2. All fast-switching signals are routed near the power and ground planes on inner layers of the board to minimize EMI effects. However, two sets of signals are routed on the top layer of the board: BRDYC1#, and JTAG signals. BRDYC1# is routed on top to take advantage of the higher trace velocity there. JTAG signals are routed on the top layer because they are low-speed signals and will probably be re-routed by each customer to suit individual needs.

3. Resistor R1 (0) is used to set the Pentium processor configurable output buffers (A3–A20, ADS#, W/R#, and HITM#). When the resistor is included the buffers are set to the Extra Large size. When it is not included (BUSCHK# internally pulled high) the buffers are set to Large size. Intel currently recommends the large buffers be used for the 256K layout example. The 0Ω resistor should be designed into your design as Intel may change the recommended buffer size once silicon and the system design have been characterized.
4. The 82496 output buffers that drive the 82491 inputs must also be configured to be Large. This is done by driving 82496 CLDRV[BGT#] (pin N04) high during reset. 82496 and 82491 Memory Bus buffer sizes must be controlled by the Memory Bus Controller.
5. Series termination resistors were added to the nets PA17, PA18, PA19, and PA20 to control overshoot. A value of 24Ω is currently recommended, but that value may change when overshoot is measured on an actual board.

7.6 Explanation of Information Provided

The following sections outline the design files associated with the 256Kbyte CPU-Cache Chip Set design example that are available from Intel. These files are provided to simplify the task of porting the design example into a specific design. By using these files, designers may eliminate or minimize the amount of duplicate effort when using the design example as the basis for their design. The following items are available:

- Schematics
- I/O Model Files
- Board Files
- Bill of Materials
- Photoplot Log
- Netlist Report
- Placed Component Report
- Artwork for Each Board Layer
- Trace Segment Line Lengths

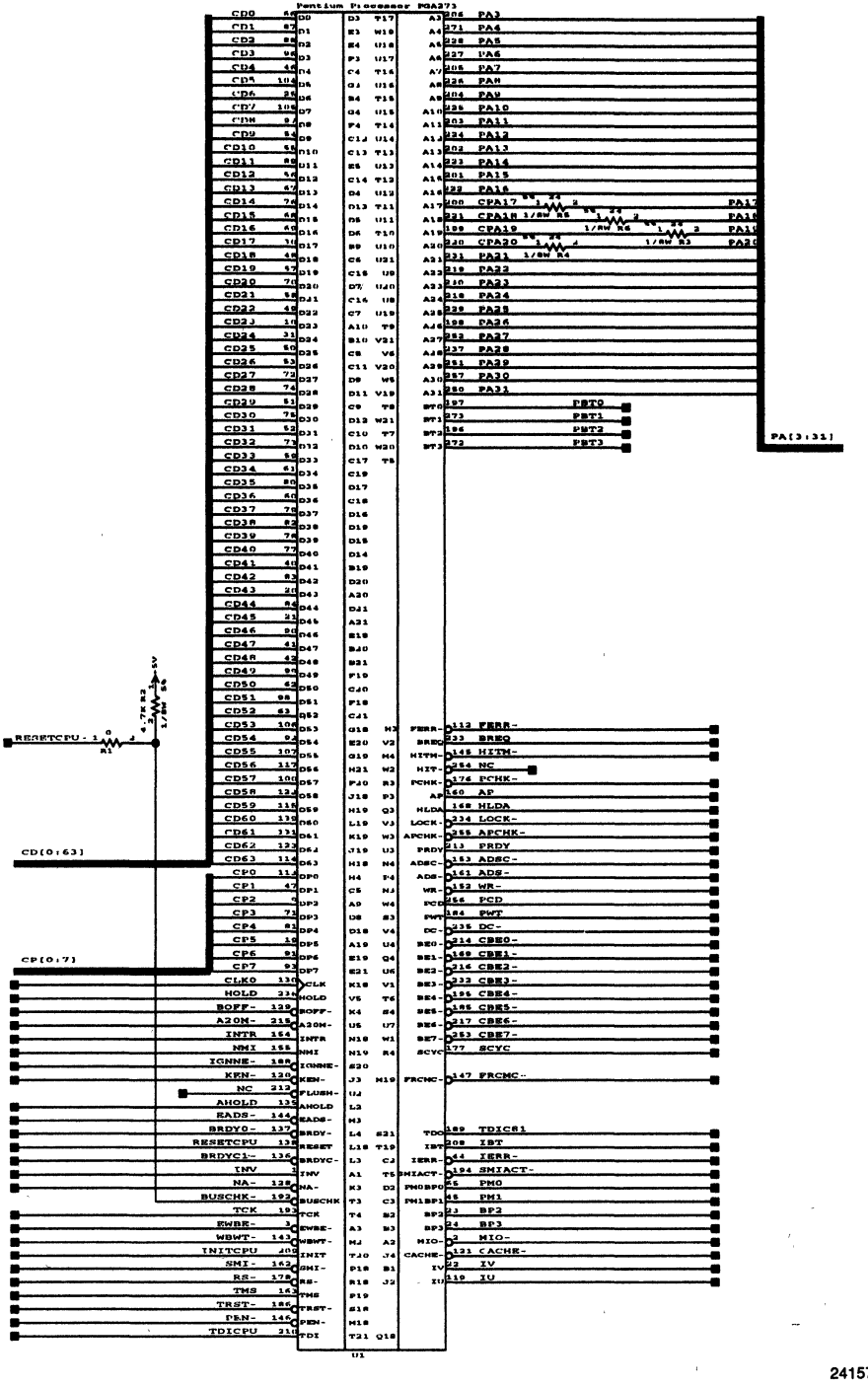
Hard copies of the schematics and trace segment line lengths are provided in the following sections. ASCII or soft copies of all the information are available from Intel by requesting order number 241663, *AP-481 Design Diskettes.*

7.6.1 SCHEMATICS

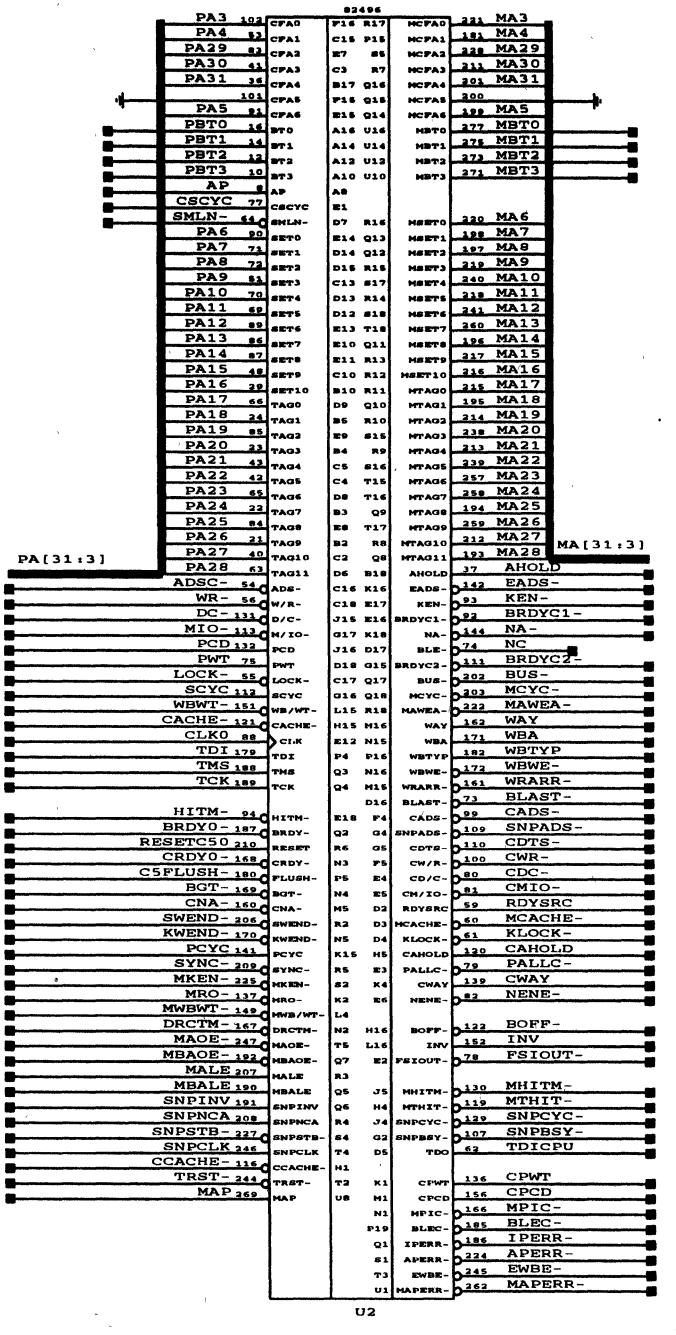
Schematics for the 256 Kbyte CPU-Cache Chip Set design example were created using ViewLogics's Workview V4.1. The schematics are 14 pages long. Both the Workview and the postscript files are available from Intel as described above.

Pentium™ Processor/82496/82491 256 KB OPTIMIZED INTERFACE
1-20-92
REV 2.1

NOTES: (Unless Otherwise Specified)
1. Capacitor values are in microfarads.
2. Resistor values are in ohms.
3. An "-" following a signal name denotes negation.
4. VCC = +5V.
5. This document also exists on electronic media.

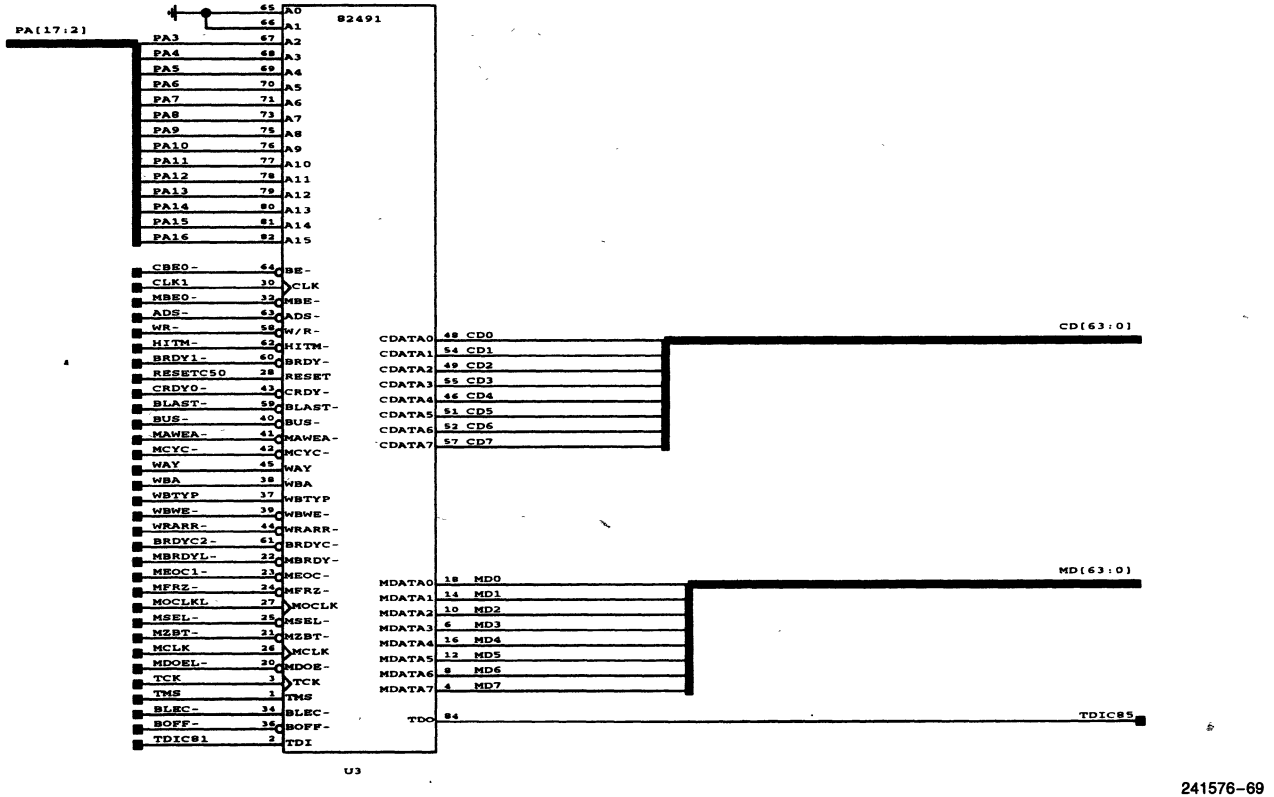

Pentium Processor PGA273
CD0
CD1
CD2
CD3
CD4
CD5
CD6
CD7
CD8
CD9
CD10
CD11
CD12
CD13
CD14
CD15
CD16
CD17
CD18
CD19
CD20
CD21
CD22
CD23
CD24
CD25
CD26
CD27
CD28
CD29
CD30
CD31
CD32
CD33
CD34
CD35
CD36
CD37
CD38
CD39
CD40
CD41
CD42
CD43
CD44
CD45
CD46
CD47
CD48
CD49
CD50
CD51
CD52
CD53
CD54
CD55
CD56
CD57
CD58
CD59
CD60
CD61
CD62
CD63
CP0
CP1
CP2
CP3
CP4
CP5
CP6
CP7
CLK0
HOLD
BOFF-
A20M-
INTR
NMI
IGNNE-
KEN-
NC
AHOLD
EADS-
BRDY0-
RESETCPU
BRDYC1-
INV
NA-
BUSCHK-
TCK
EWBE-
WBWT-
INITCPU
SMI-
RS-
TMS
TRST-
PEN-
TDICPU
RESETCPU-
CD[0:63]
CP[0:7]
PA[3:31]
PBT0
PBT1
PBT2
PBT3
FERR-
BREQ
HITM-
HIT-
PCHK-
AP
HLDA
LOCK-
APCHK-
PRDY
ADSC-
ADS-
WR-
PCD
PWT
DC-
BE0-
BE1-
BE2-
BE3-
BE4-
BE5-
BE6-
BE7-
SCYC
FRCMC-
TDIC81
IBT
IERR-
SMIACT-
PM0
PM1
BP2
BP3
MIO-
CACHE-
IV
IU
U1

241576-67


82496
PA[31:3]
MA[31:3]
U2
PA3 103 CFA0
PA4 83 CFA1
PA29 82 CFA2
PA30 41 CFA3
PA31 36 CFA4
101 CFA5
PA5 81 CFA6
PBT0 16 BT0
PBT1 14 BT1
PBT2 12 BT2
PBT3 10 BT3
AP 8 AP
CSCYC 77 CSCYC
SMLN- 44 SMLN-
PA6 80 SET0
PA7 73 SET1
PA8 72 SET2
PA9 51 SET3
PA10 70 SET4
PA11 69 SET5
PA12 89 SET6
PA13 86 SET7
PA14 87 SET8
PA15 49 SET9
PA16 29 SET10
PA17 66 TAG0
PA18 24 TAG1
PA19 85 TAG2
PA20 23 TAG3
PA21 43 TAG4
PA22 42 TAG5
PA23 65 TAG6
PA24 22 TAG7
PA25 84 TAG8
PA26 21 TAG9
PA27 40 TAG10
PA28 63 TAG11
ADSC- 54 ADS-
WR- 56 W/R-
DC- 131 D/C-
MIO- 113 M/IO-
PCD 132 PCD
PWT 75 PWT
LOCK- 55 LOCK-
SCYC 112 SCYC
WBWT- 151 WB/WT-
CACHE- 121 CACHE-
CLK0 88 CLK
TDI 179 TDI
TMS 188 TMS
TCK 189 TCK
HITM- 94 HITM-
BRDY0- 187 BRDY-
RESETC50 210 RESET
CRDY0- 168 CRDY-
C5FLUSH- 180 FLUSH-
BGT- 169 BGT-
CNA- 160 CNA-
SWEND- 206 SWEND-
KWEND- 170 KWEND-
PCYC 141 PCYC
SYNC- 209 SYNC-
MKEN- 225 MKEN-
MRO- 137 MRO-
MWBWT- 149 MWB/WT-
DRCTM- 167 DRCTM-
MAOE- 247 MAOE-
MBAOE- 192 MBAOE-
MALE 207 MALE
MBALE 190 MBALE
SNPINV 191 SNPINV
SNPNCA 208 SNPNCA
SNPSTB- 227 SNPSTB-
SNPCLK 246 SNPCLK
CCACHE- 116 CCACHE-
TRST- 244 TRST-
MAP 269 MAP
F16 R17 MCFA0 221 MA3
C15 P15 MCFA1 181 MA4
E7 S5 MCFA2 228 MA29
C3 R7 MCFA3 211 MA30
B17 Q16 MCFA4 201 MA31
F18 Q18 MCFA5 200
E15 Q14 MCFA6 198 MA5
A16 U16 MBT0 277 MBT0
A14 U14 MBT1 275 MBT1
A12 U12 MBT2 273 MBT2
A10 U10 MBT3 271 MBT3
A8
E1
D7 R16 MSET0 220 MA6
E14 Q13 MSET1 198 MA7
D14 Q12 MSET2 197 MA8
D15 R15 MSET3 219 MA9
C13 S17 MSET4 240 MA10
D13 R14 MSET5 218 MA11
D12 S18 MSET6 241 MA12
E13 T18 MSET7 260 MA13
E10 Q11 MSET8 196 MA14
E11 R13 MSET9 237 MA15
C10 R12 MSET10 216 MA16
B10 R11 MTAG0 215 MA17
D9 Q10 MTAG1 195 MA18
B5 R10 MTAG2 214 MA19
E9 S15 MTAG3 238 MA20
B4 R9 MTAG4 213 MA21
C5 S16 MTAG5 239 MA22
C4 T15 MTAG6 257 MA23
D8 T16 MTAG7 258 MA24
B3 Q9 MTAG8 194 MA25
E8 T17 MTAG9 259 MA26
B2 R8 MTAG10 212 MA27
C2 Q8 MTAG11 193 MA28
D6 B18 AHOLD 37 AHOLD
C16 K16 EADS- 142 EADS-
C18 E17 KEN- 93 KEN-
J15 E16 BRDYC1- 92 BRDYC1-
G17 K18 NA- 144 NA-
J16 D17 BLE- 74 NC
D18 G15 BRDYC2- 111 BRDYC2-
C17 Q17 BUS- 202 BUS-
G16 Q18 MCYC- 203 MCYC-
L15 R18 MAWEA- 222 MAWEA-
H15 M16 WAY 162 WAY
E12 N15 WBA 171 WBA
P4 P16 WBTYP 182 WBTYP
Q3 N16 WBWE- 172 WBWE-
Q4 M15 WRARR- 161 WRARR-
D16 BLAST- 73 BLAST-
E18 F4 CADS- 99 CADS-
Q2 G4 SNPADS- 109 SNPADS-
R6 G5 CDTS- 110 CDTS-
N3 F5 CW/R- 100 CWR-
P5 E4 CD/C- 80 CDC-
N4 E5 CM/IO- 81 CMIO-
M5 D2 RDYSRC 69 RDYSRC
R2 D3 MCACHE- 60 MCACHE-
N5 D4 KLOCK- 61 KLOCK-
K15 H5 CAHOLD 120 CAHOLD
R5 E3 PALLC- 79 PALLC-
S2 K4 CWAY 139 CWAY
K2 E6 NENE- 82 NENE-
L4
N2 H16 BOFF- 122 BOFF-
T5 L16 INV 152 INV
Q7 E2 FSIOUT- 78 FSIOUT-
R3
Q5 J5 MHITM- 130 MHITM-
Q6 H4 MTHIT- 119 MTHIT-
R4 J4 SNPCYC- 129 SNPCYC-
S4 G2 SNPBSY- 107 SNPBSY-
T4 D5 TDO 62 TDICPU
H1
T2 K1 CPWT 136 CPWT
U8 M1 CPCD 156 CPCD
N1 MPIC- 166 MPIC-
P19 BLEC- 185 BLEC-
Q1 IPERR- 186 IPERR-
S1 APERR- 224 APERR-
T3 EWBE- 245 EWBE-
U1 MAPERR- 262 MAPERR-

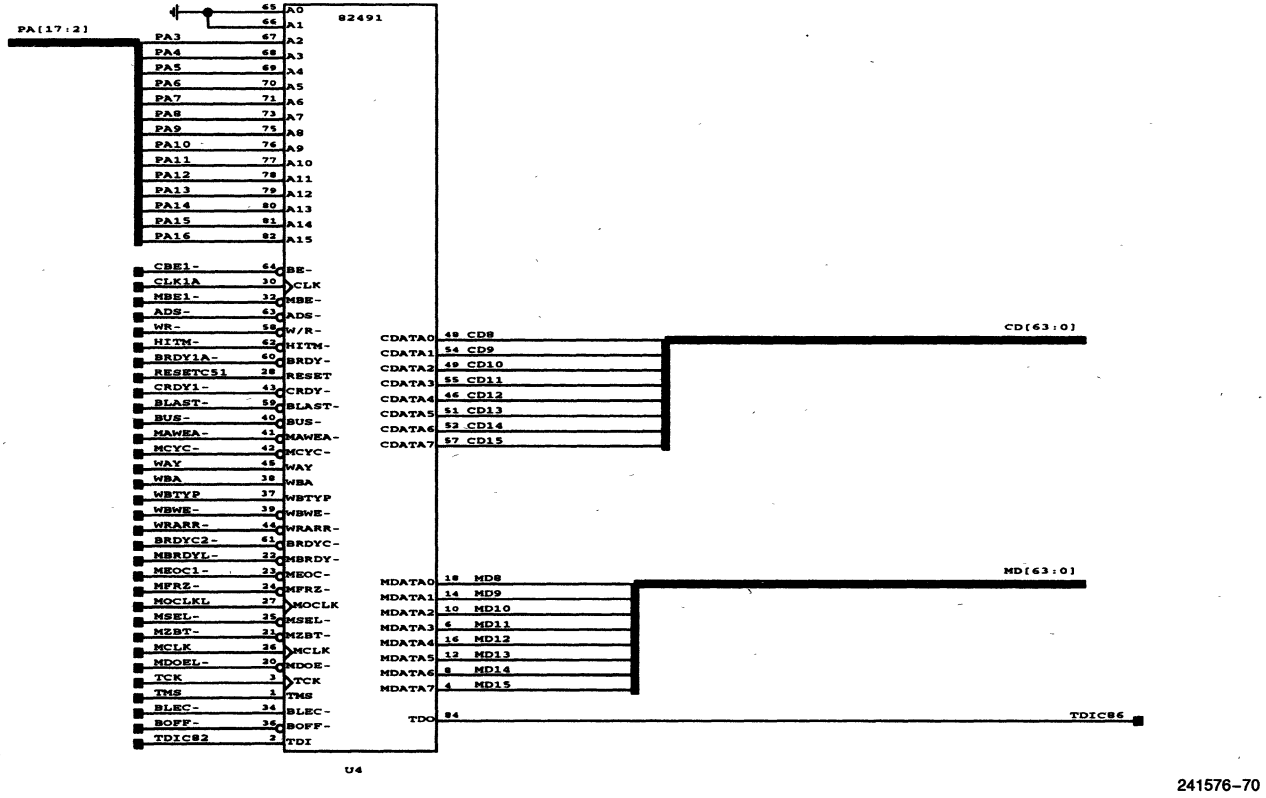
241576-68

82491 Byte 0



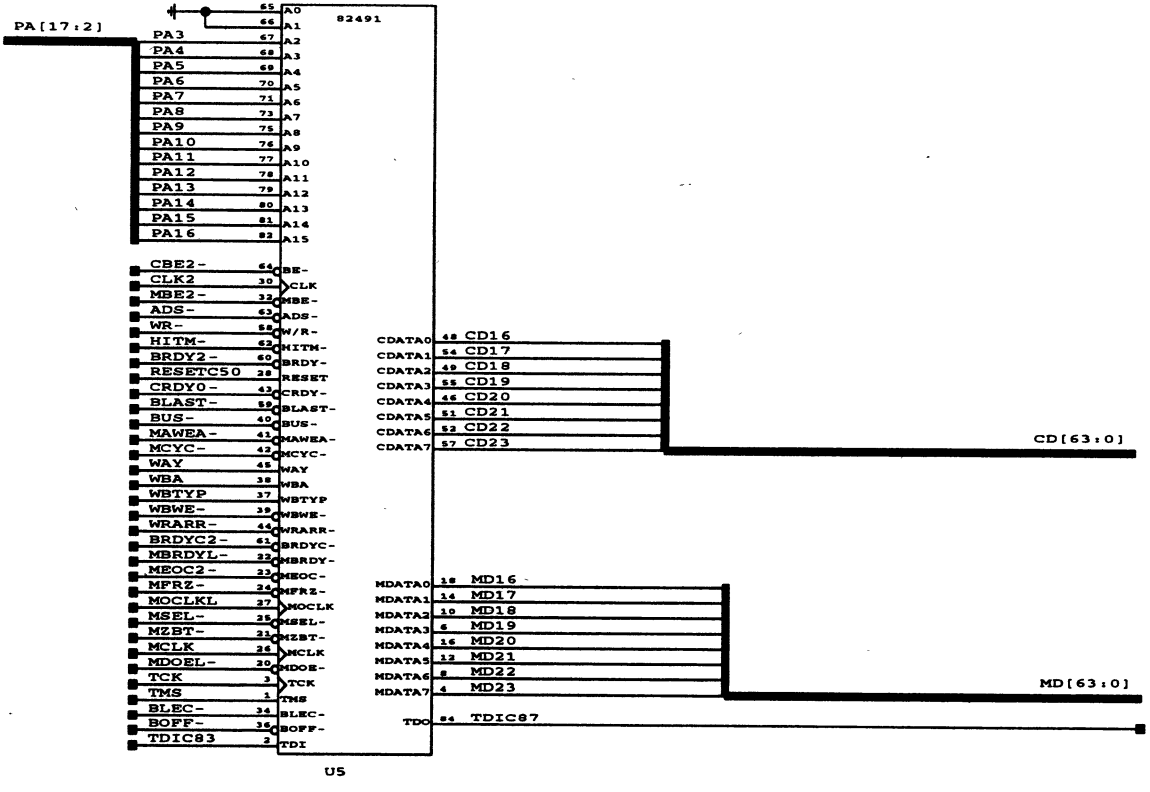
241576-69

82491 Byte 1



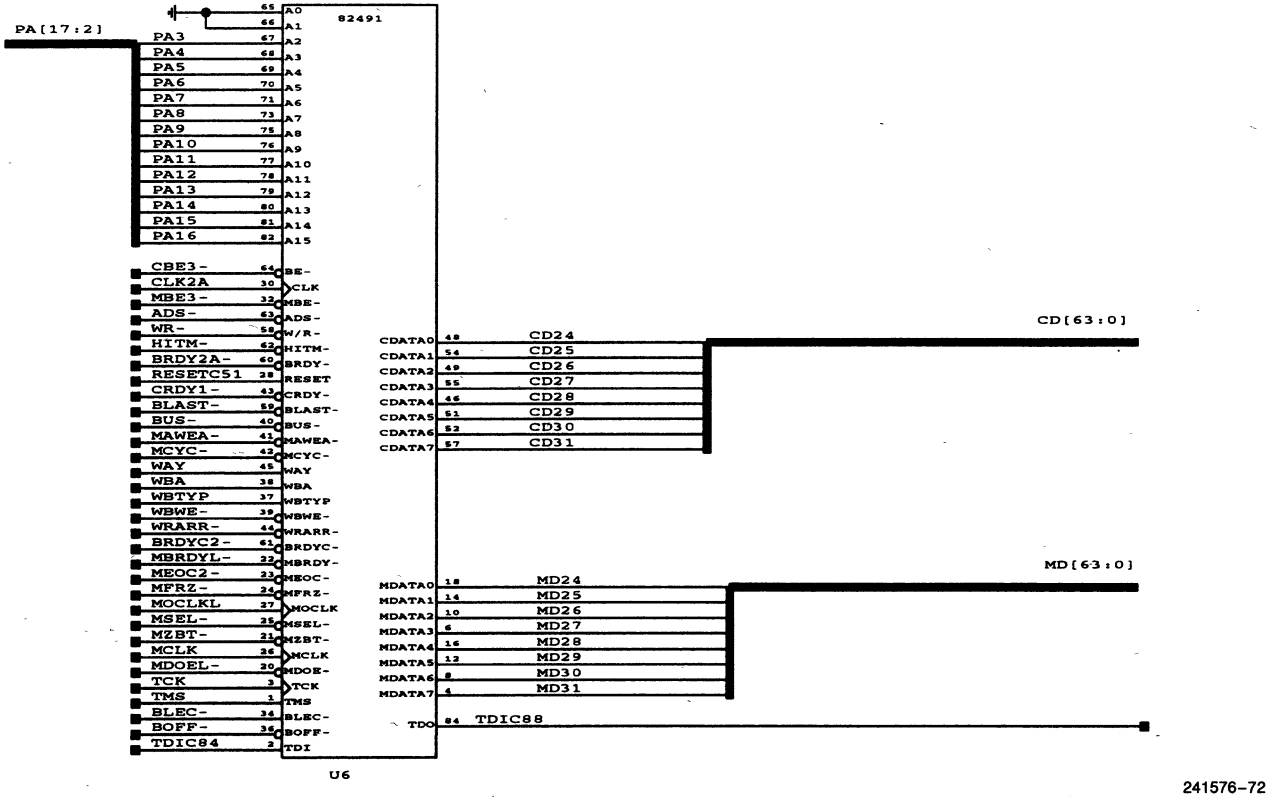
241576-70

82491 Byte 2



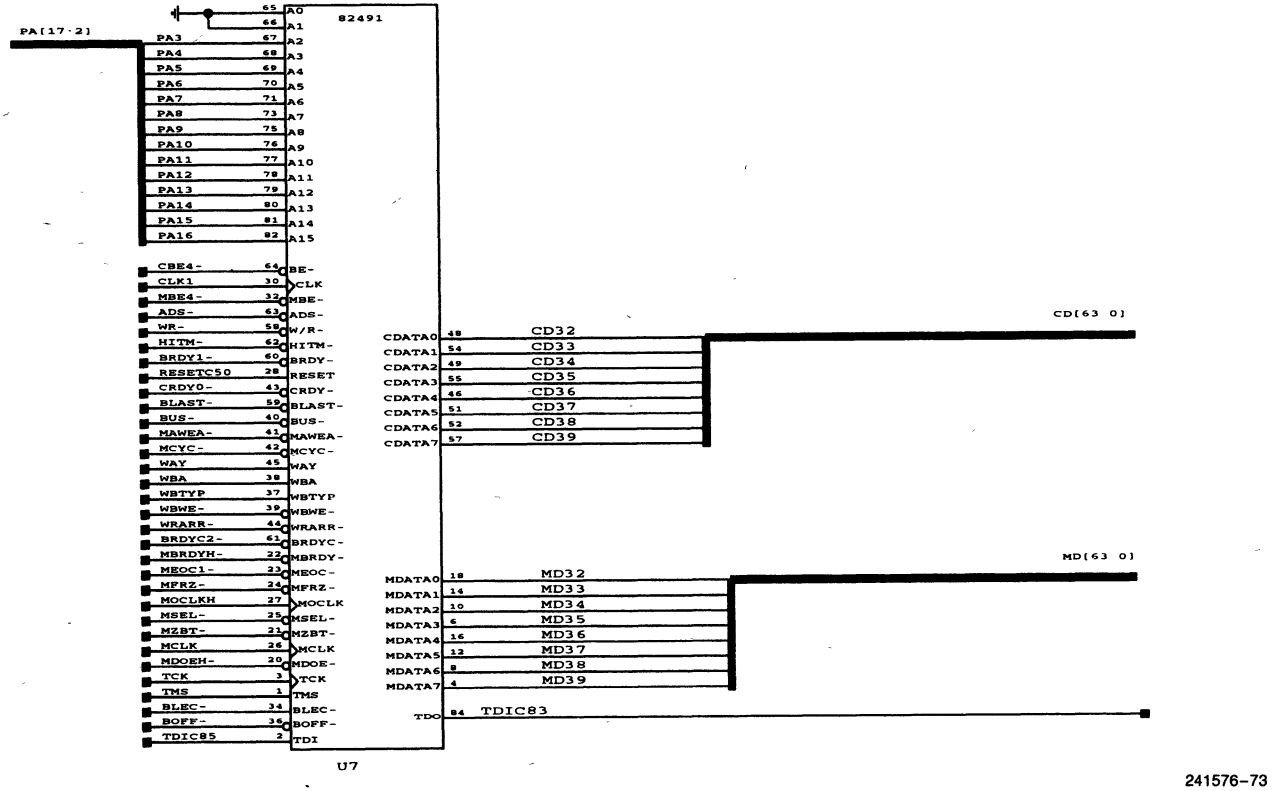
241576-71

82491 Byte 3



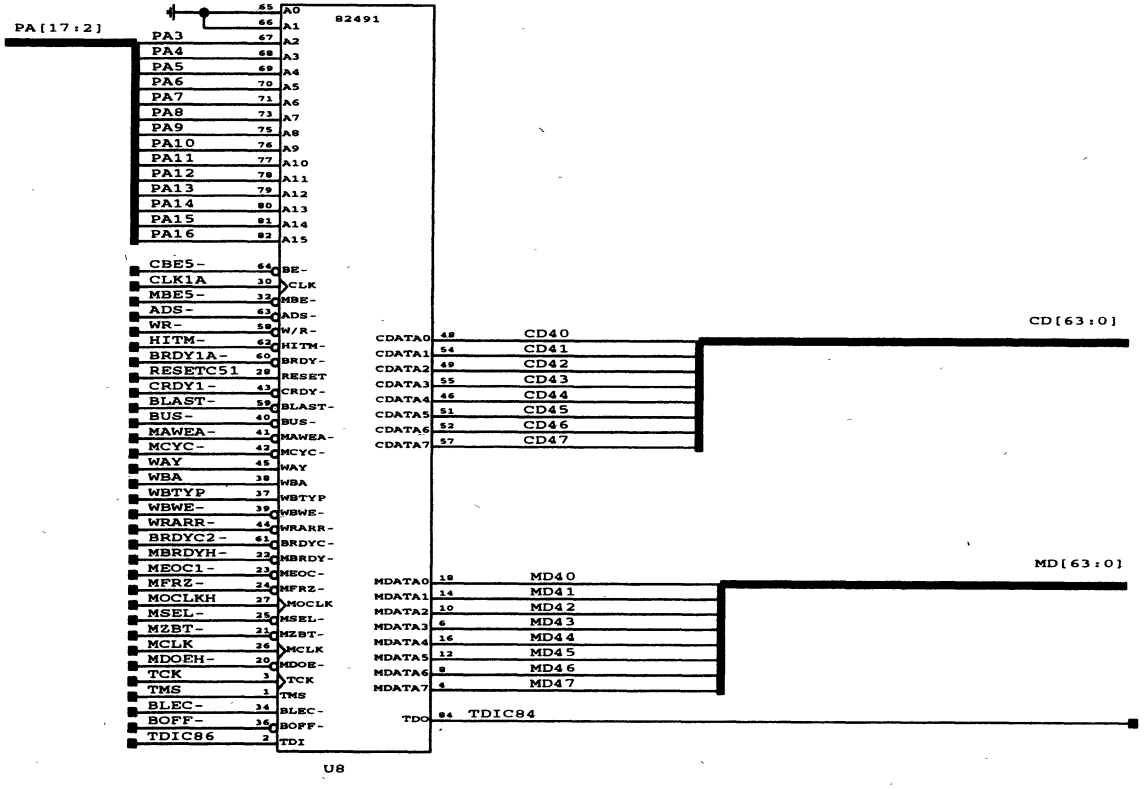
241576-72

82491 Byte 4



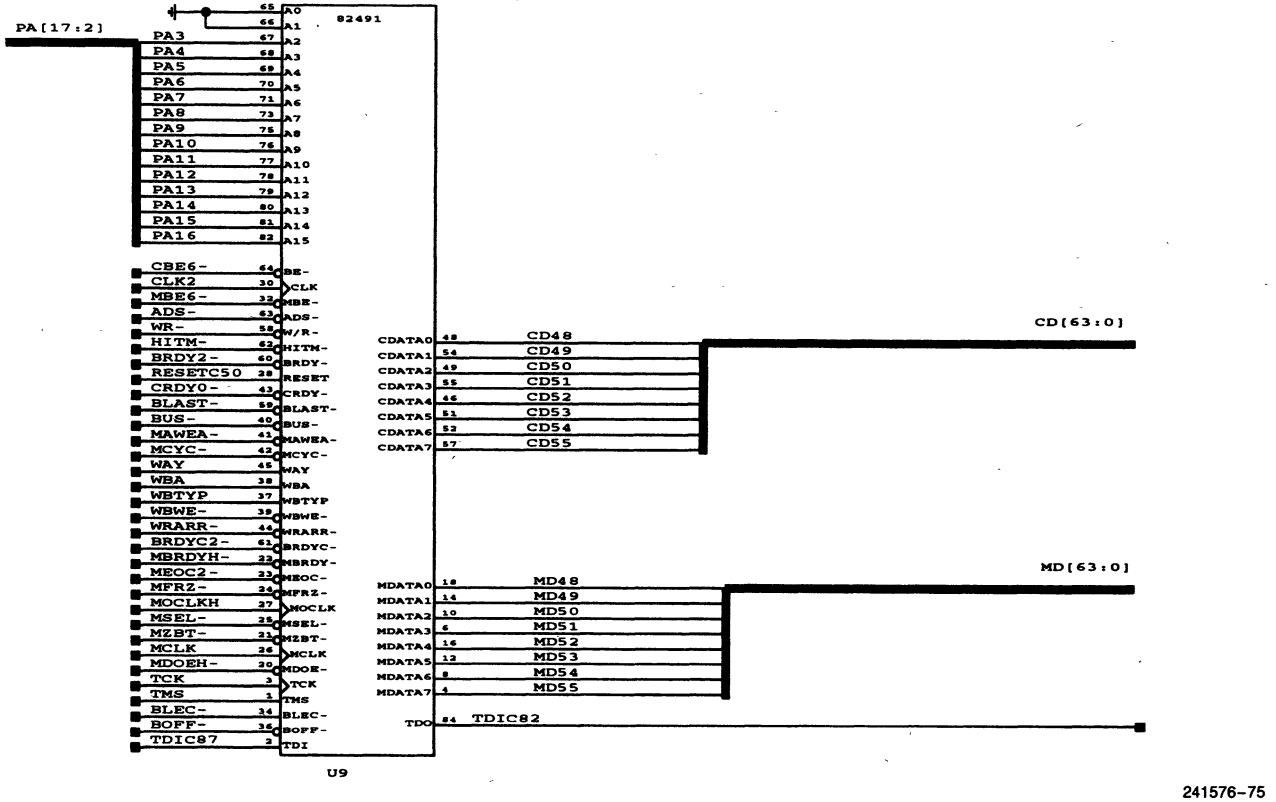
241576-73

82491 Byte 5



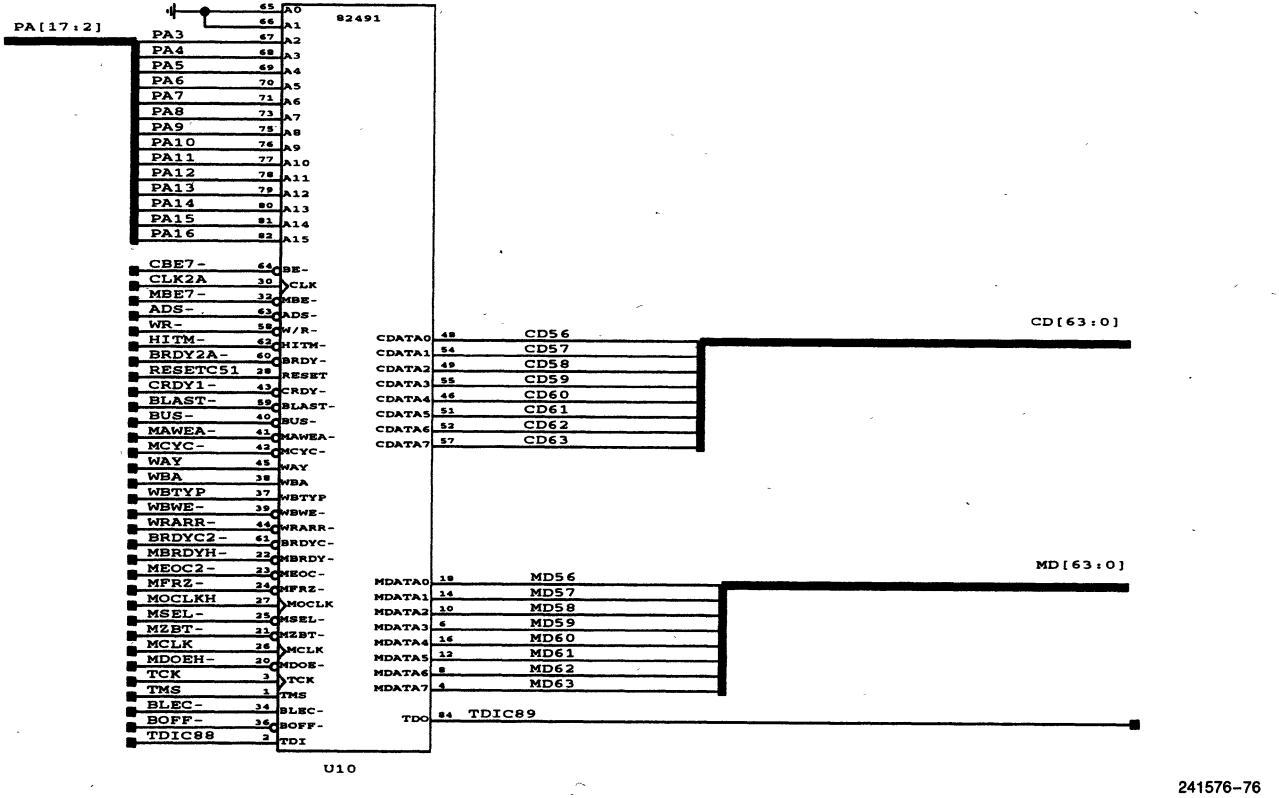
241576-74

82491 Byte 6

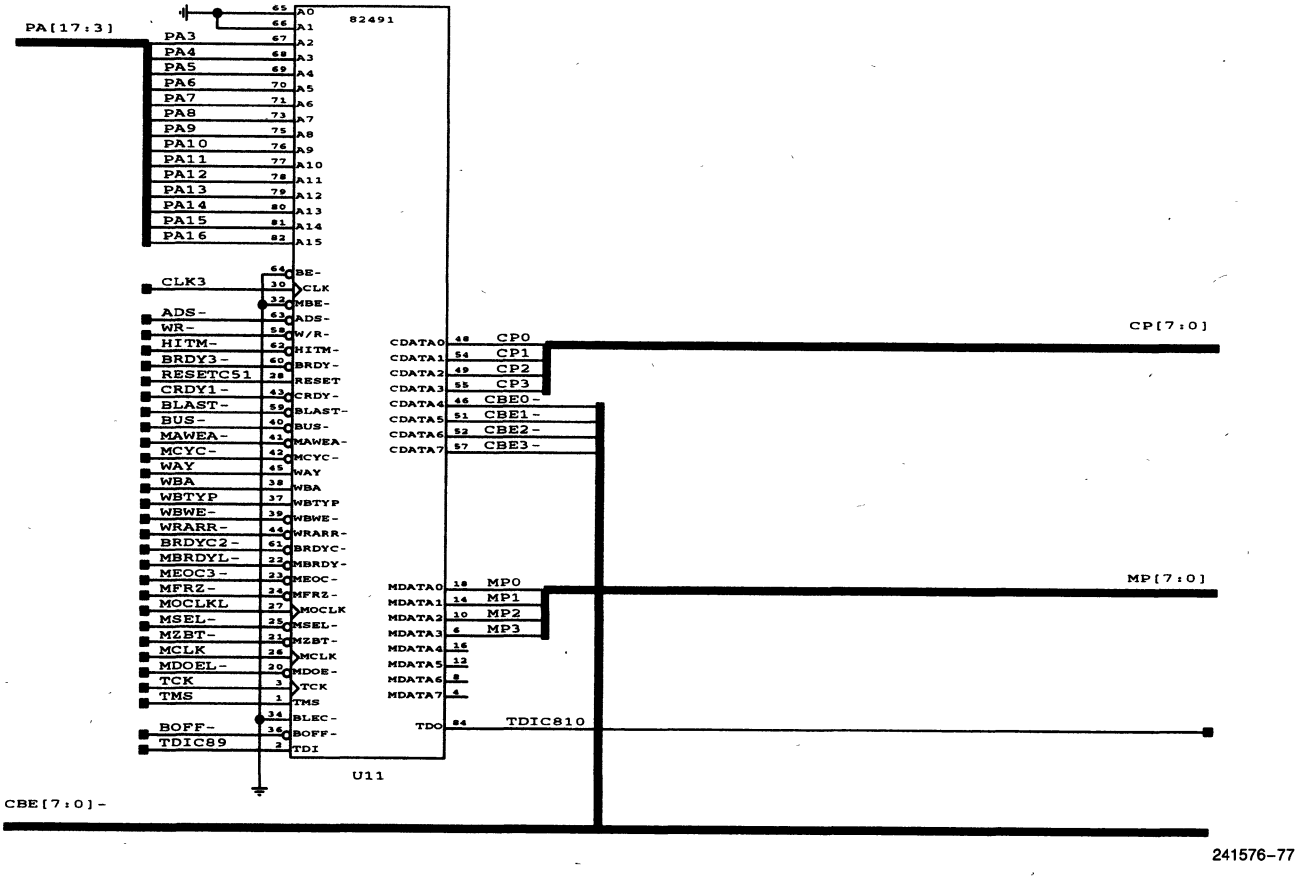


241576-75

82491 Byte 7

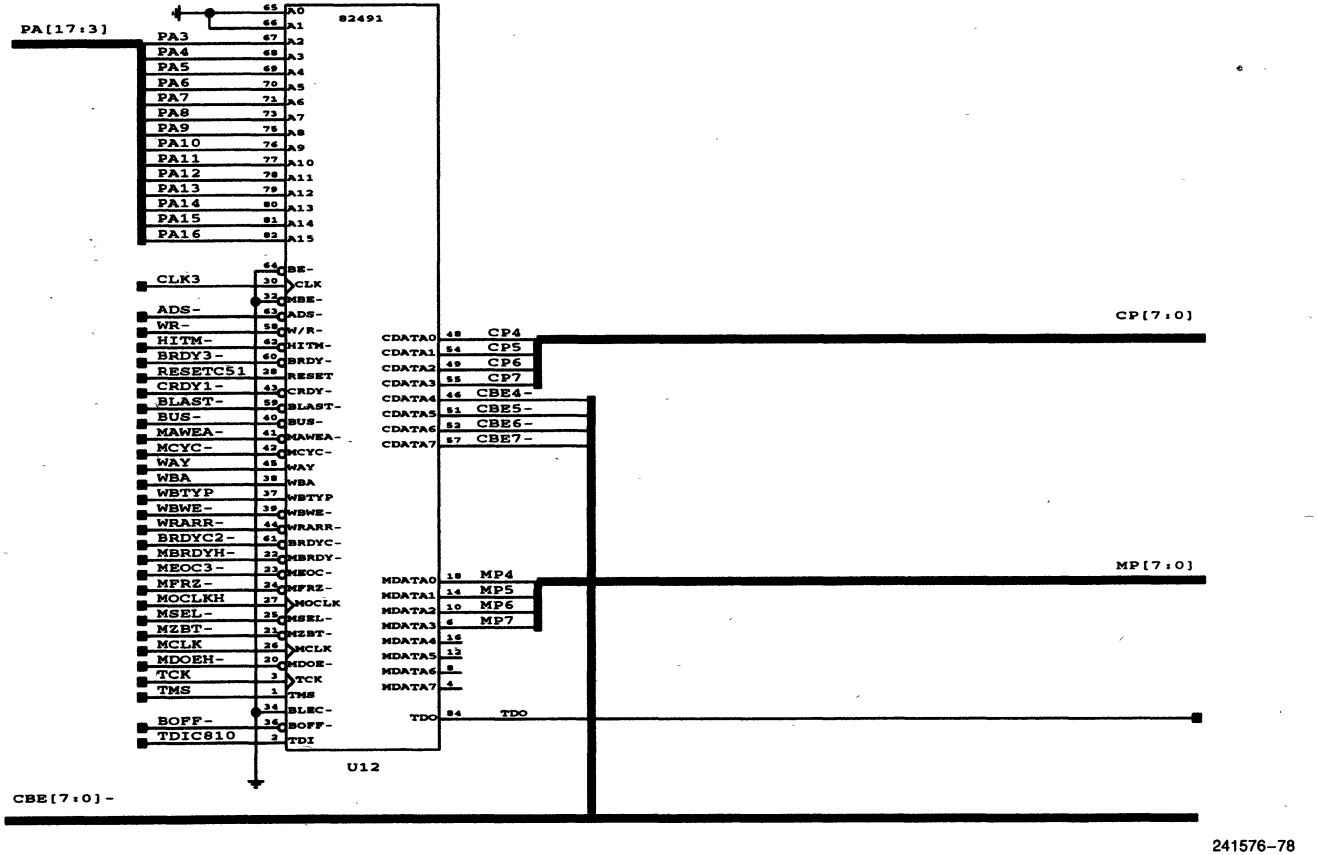


82491 Parity 0–3

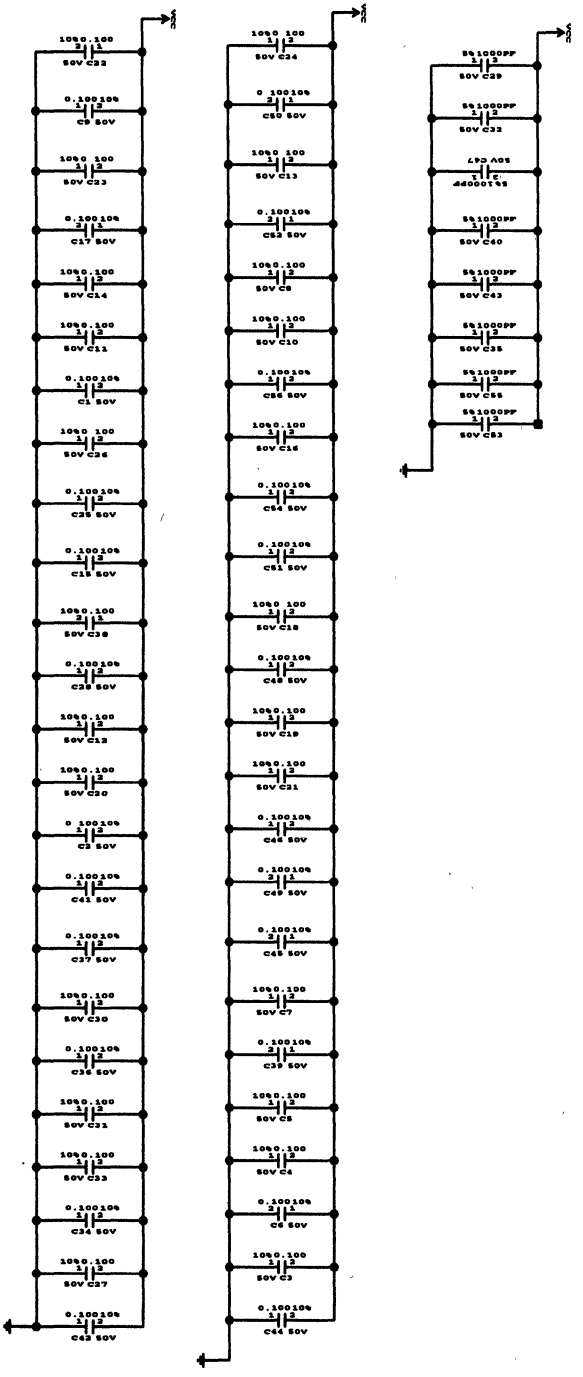


241576-77

82491 Parity 4-7



241576-78


0.100 10%
50V C22
0.100 10%
C9 50V
0.100 10%
50V C23
0.100 10%
C17 50V
0.100 10%
50V C14
0.100 10%
50V C11
0.100 10%
C1 50V
0.100 10%
50V C26
0.100 10%
C25 50V
0.100 10%
C15 50V
0.100 10%
50V C28
0.100 10%
C28 50V
0.100 10%
50V C12
0.100 10%
50V C20
0.100 10%
C2 50V
0.100 10%
C41 50V
0.100 10%
C37 50V
0.100 10%
50V C30
0.100 10%
C36 50V
0.100 10%
50V C31
0.100 10%
50V C33
0.100 10%
C34 50V
0.100 10%
50V C27
0.100 10%
C42 50V
0.100 10%
50V C24
0.100 10%
C50 50V
0.100 10%
50V C13
0.100 10%
C52 50V
0.100 10%
50V C8
0.100 10%
50V C10
0.100 10%
C56 50V
0.100 10%
50V C16
0.100 10%
C54 50V
0.100 10%
C51 50V
0.100 10%
50V C18
0.100 10%
C48 50V
0.100 10%
50V C19
0.100 10%
50V C21
0.100 10%
C46 50V
0.100 10%
C49 50V
0.100 10%
C45 50V
0.100 10%
50V C7
0.100 10%
C39 50V
0.100 10%
50V C5
0.100 10%
50V C4
0.100 10%
C6 50V
0.100 10%
50V C3
0.100 10%
C44 50V
1000PF
50V C29
1000PF
50V C32
50V C47
1000PF
50V C40
1000PF
50V C43
1000PF
50V C35
1000PF
50V C55
1000PF
50V C53

241576-79

7.6.2 I/O MODEL FILES

All electrical I/O simulations were performed using TLC V4.1.13 from Quad Design Technology, Inc. The simulations were performed at the fast and slow corners to verify all signal quality and flight time specifications are met. The files used for these simulations are available from Intel as described above. These files include the topology, model, and control files needed to run the simulations for all nets in the optimized interface.

7.6.3 BOARD FILES

The board files for the design example were created using Allegro V4.2 from Cadence Design Systems, Inc. The files are available from Intel as described above. These files may be used to import the design example into a specific system design. Note: some changes to the layout and nets may be necessary to complete importing these files into a specific system design.

7.6.4 BILL OF MATERIALS

The bill of materials file was created using Allegro V4.2 from Cadence Design Systems, Inc. The file is available from Intel as described above.

7.6.5 PHOTOPLOT LOG

The photoplot log file was created using Allegro V4.2 from Cadence Design Systems, Inc. The file is available from Intel as described above.

7.6.6 NETLIST REPORT

The netlist report was created using Allegro V4.2 from Cadence Design Systems, Inc. The file is available from Intel as described above.

7.6.7 PLACED COMPONENT REPORT

The placed component report was created using Allegro V4.2 from Cadence Design Systems, Inc. The file is available from Intel as described above.

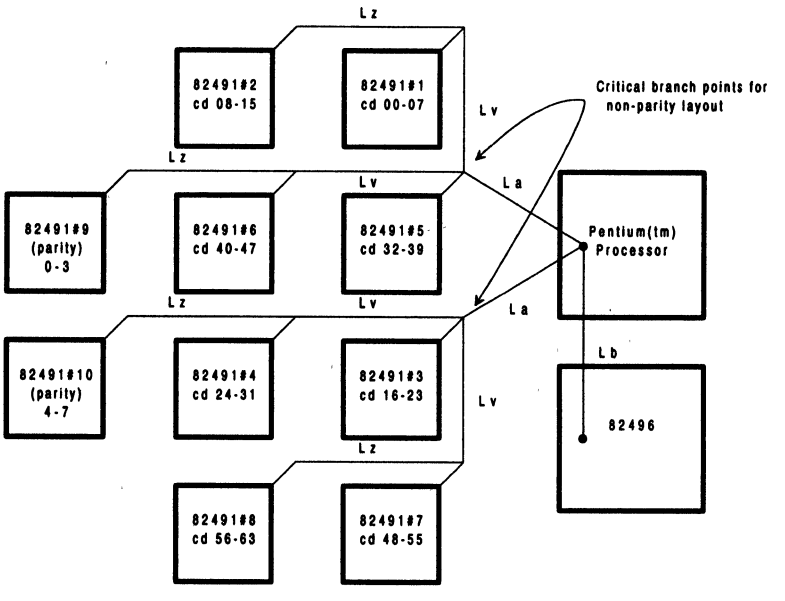
7.6.8 ARTWORK FOR EACH BOARD LAYER

The artwork for the six board layers were created using Allegro V4.2 from Cadence Design Systems, Inc. The files are available from Intel in a Gerber format as described above.

7.6.9 TRACE SEGMENT LINE LENGTHS

Sections 7.6.9.1 to 7.6.9.10 list the segment line lengths for each net of the optimized interface. All lengths are provide in mils (1/1000 inch). The stubs listed in the following tables are associated with the pin escapes required for the 82491s.

7.6.9.1 Low Addresses (Topology 1)



NET	PP-CS#5	PP-CS#3	PP-CC	CS#5-CS#1	CS#1-CS#2	CS#5-CS#6	CS#6-CS#9
PA3	1761.4	1768.6	5513	1184.6	936.5	1186.5	936.5
PA4	1259.6	1278.4	4673.2	1113.9	936.5	1113.6	936.5
PA5	1553.6	1573.9	5193.7	1164.6	936.5	1176.5	936.5
PA6	1543.1	1540	5123.2	1152.9	936.5	1156.5	936.5
PA7	1691.9	1692.4	5367.6	1152.9	936.5	1156.5	936.5
PA8	1590.7	1590.2	5243.7	1215.3	936.5	1216.5	936.5
PA9	1660.7	1663.1	5365.2	1210.5	936.5	1206.5	936.5
PA10	1543.6	1543.1	5233	1264.1	936.5	1266.5	936.5
PA11	1701.9	1695.5	5474.2	1264.1	936.5	1266.5	936.5
PA12	1586.5	1594.3	5324.1	1292.4	936.5	1296.5	936.5
PA13	1741.9	1745.5	5497.9	1295.3	936.5	1296.5	936.5
PA14	1633.6	1633.1	5403.8	1317.8	936.5	1316.5	936.5
PA15	1791.9	1792.6	5500.4	1314.8	936.5	1316.5	936.5
PA16	1623.6	1619	5359.1	1288.0	936.5	1286.5	936.5

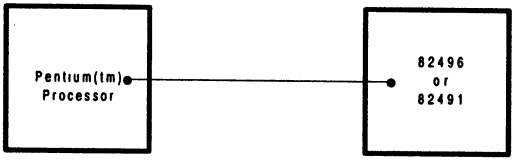
PP = Pentium processor
CC = 82496 cache controller
CS = 82491 cache SRAM

NET	CS#3-CS#4	CS#4-CS#10	CS#3-CS#7	CS#7-CS#8	Stubs
PA3 (cont)	1181.4	936.5	1184.6	936.5	135.3–135.3
PA4 (cont)	1111.4	936.5	1113.9	936.5	75.0–75.0
PA5 (cont)	1166.5	936.5	1170.5	936.5	135.3–135.3
PA6 (cont)	1156.5	936.5	1158.8	936.5	75.0–75.0
PA7 (cont)	1156.5	936.5	1158.8	936.5	135.3–135.3
PA8 (cont)	1216.5	936.5	1212.4	936.5	135.3–135.3
PA9 (cont)	1206.5	936.5	1207.6	936.5	135.3–135.3
PA10 (cont)	1266.5	936.5	1261.2	936.5	75.0–75.0
PA11 (cont)	1266.5	936.5	1261.2	936.5	135.3–135.3
PA12 (cont)	1296.5	936.5	1292.4	936.5	75.0–75.0
PA13 (cont)	1296.5	936.5	1295.3	936.5	135.3–135.3
PA14 (cont)	1316.5	936.5	1317.8	936.5	75.0–75.0
PA15 (cont)	1316.5	936.5	1314.8	936.5	135.3–135.3
PA16 (cont)	1286.5	936.5	1288.0	936.5	75.0–75.0

PP = Pentium processor
CC = 82496 cache controller
CS = 82491 cache SRAM

7.6.9.2 High Addresses (Topology 4, Point-to-Point)

Routing Topology 4



241576-81

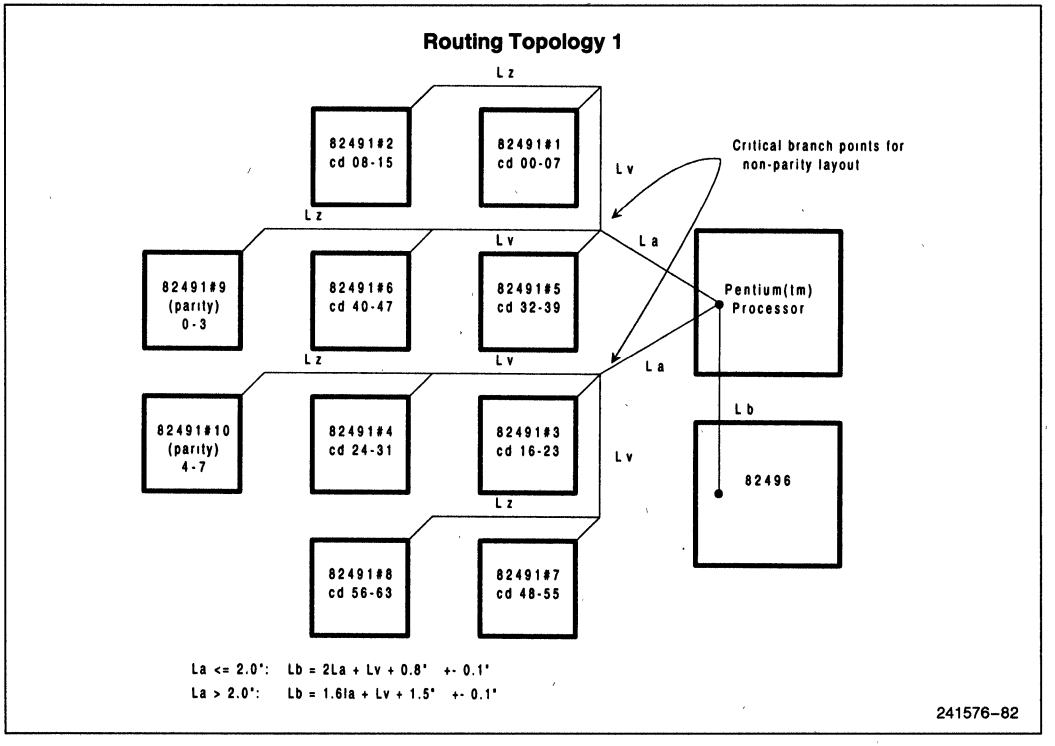
NET	PP-CC
PA17*	689.3 + 2841.7
PA18*	647.6 + 3683.1
PA19*	731.0 + 2763.3
PA20*	601.7 + 718.6
PA21	3376.6
PA22	4347.5
PA23	3111.5
PA24	4661.2
PA25	3029.7

NET	PP-CC
PA26	4495.1
PA27	3885.5
PA28	4689.3
PA29	3136.1
PA30	5177.1
PA31	2518.5
PA32	3604.4
PA33	2686.8
PA34	3952
PA35	3189.9

NET	PP-CC
ADSC#	3791.7
AP	4531.6
CACHE#	3719.8
DC#	3613.1
LOCK#	4710.4
MIO#	5062
PCD	3461.3
PWT	4295.6
SCYC	3848.2
WBWT#	3493.3

***NOTE:**
24Ω resistor included on PA[17–20].
Lengths are Pentium processor-resistor + resistor-82486, respectively.
PP = Pentium processor
CC = 82496 cache controller
CS = 82491 cache SRAM

7.6.9.3 Pentium™ Processor Control (Topology 1)

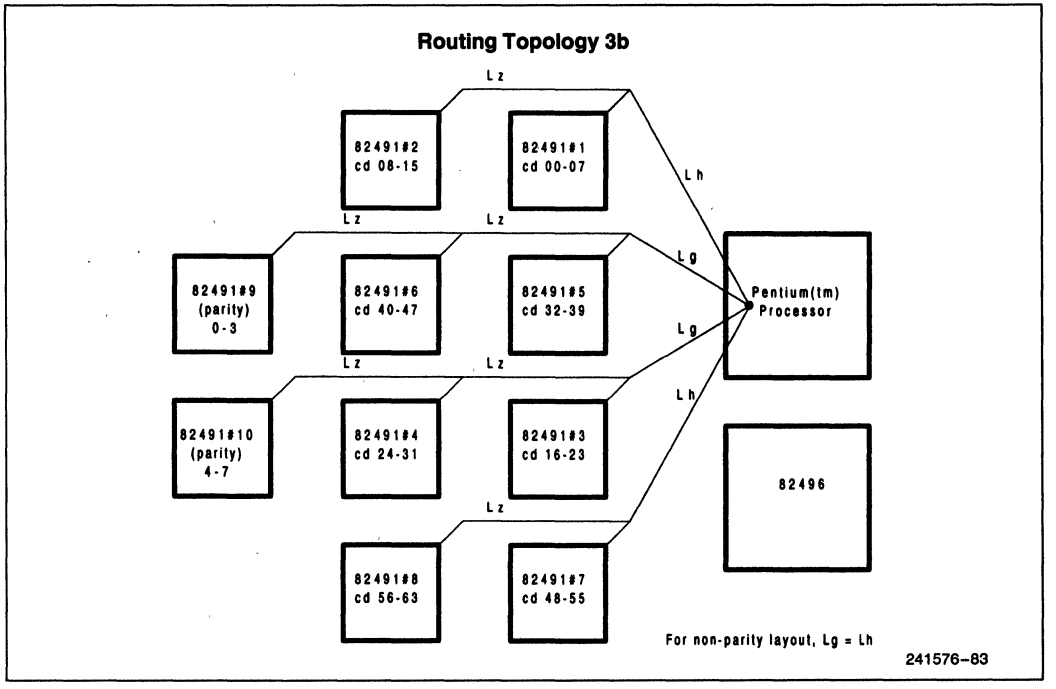


NET	PP-CS#5	PP-CS#3	PP-CC	CS#5-CS#1	CS#1-CS#2	CS#5-CS#6	CS#6-CS#9
HITM#	4205.2	4199.7	9147.1	1141.3	936.5	1146.5	936.5
WR#	4091.1	4088.2	9149.4	1193.0	936.5	1192.1	936.5

NET	CS#3-CS#4	CS#4-CS#10	CS#3-CS#7	CS#7-CS#8	Stubs
HITM# (cont)	1146.2	944.8	1146.6	944.8	95.3-95.3
WR# (cont)	1196.7	953.1	1193.0	953.1	95.3-95.3

PP = Pentium processor
CC = 82496 cache controller
CS = 82491 cache SRAM

7.6.9.4 Pentium™ Processor Control (Topology 3b No 82496)

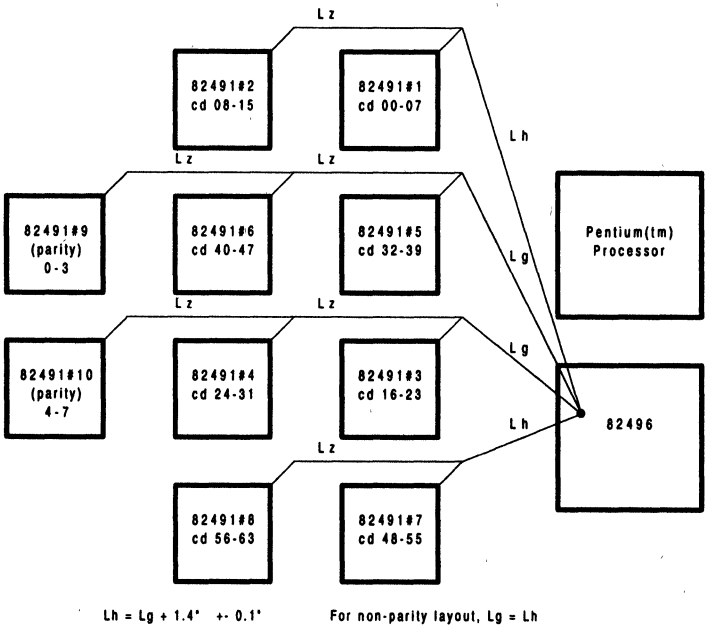


NET	PP-CS#1	PP-CS#5	PP-CS#3	PP-CS#7	CS#1-CS#2	CS#5-CS#6	CS#6-CS#9
ADS#	5113.0	3728.2	3738.5	5102.0	936.5	964.8	983.8

NET	CS#3-CS#4	CS#4-CS#10	CS#7-CS#8	Stubs
ADS# (cont)	936.5	936.5	936.5	75.0–75.0

PP = Pentium processor
CC = 82496 cache controller
CS = 82491 cache SRAM

7.6.9.5 82496 Control (Topology 3)

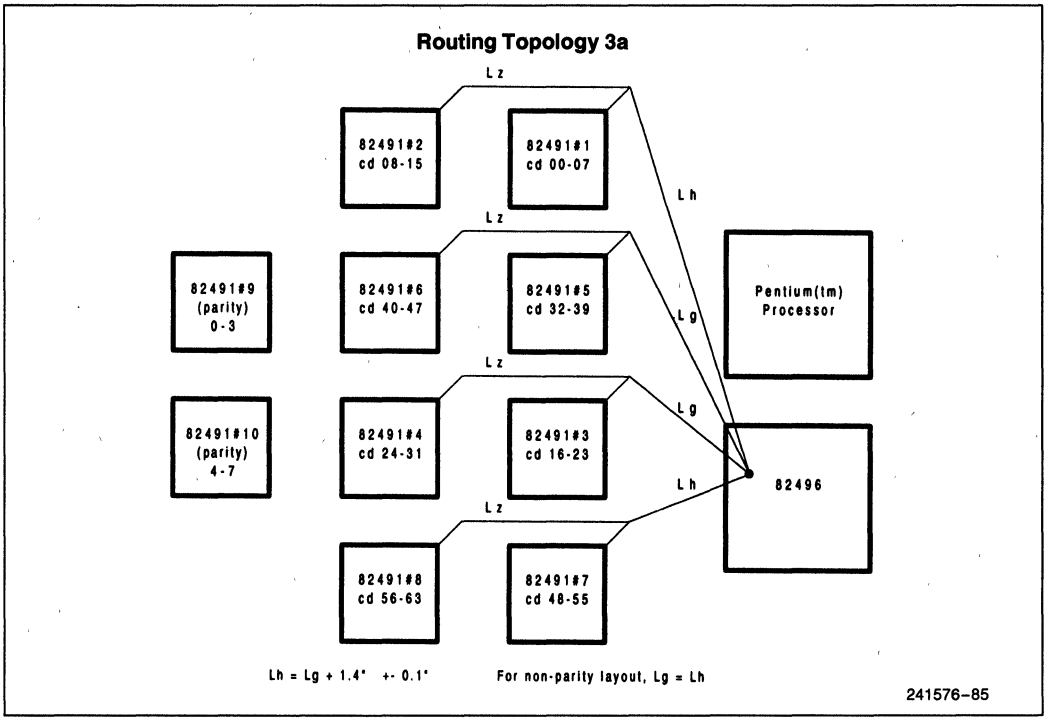


NET	CC-CS#1	CC-CS#5	CC-CS#3	CC-CS#7	CS#1-CS#2	CS#5-CS#6
BLAST#	4222.1	2820.0	2819.8	4219.2	986.7	1034.5
BRDYC2#	4186.4	2802.4	2775.0	4171.6	969.7	1037.4
BUS#	4607.6	3215.7	3210.0	4611.3	936.5	936.5
MAWEA#	4476.4	3058.9	3088.8	4481.3	936.5	936.5
MCYC#	4913.9	3507.0	3523.8	4921.5	936.5	961.4
WBA	5114.2	3747.8	3719.5	5119.3	936.5	936.5
WBTYP	4365.4	2986.2	2973.5	4376.0	936.5	936.5
WBWE#	4502.4	3109.1	3113.0	4511.7	936.5	936.5
WRARR#	4198.9	2735.0	2802.1	4199.8	936.5	969.7
WAY	4818.9	3348.4	3417.4	4816.3	936.5	936.5

NET	CS#6–CS#9	CS#3–CS#4	CS#4–CS#10	CS#7–CS#8	Stubs
BLAST# (cont)	965.5	936.5	936.5	936.5	85.0–85.0
BRDYC2# (cont) BUS# (cont)	965.5 936.5	936.5 936.5	936.5 936.5	936.5 936.5	85.0–85.0 75.0–75.0
MAWEA# (cont)	936.5	936.5	936.5	936.5	135.3–135.3
MCYC# (cont)	936.5	936.5	936.5	936.5	75.0–75.0
WBA (cont)	936.5	936.5	936.5	936.5	75.0–75.0
WBTYP (cont)	936.5	936.5	936.5	936.5	135.3–135.3
WBWE# (cont)	936.5	936.5	936.5	936.5	135.3–135.3
WRARR# (cont)	936.5	936.5	936.5	936.5	135.3–135.3
WAY (cont)	936.5	936.5	936.5	936.5	75.0–75.0

PP = Pentium processor
CC = 82496 cache controller
CS = 82491 cache SRAM

7.6.9.6 82496 Control (Topology 3a Not Connected to Parity 82491's)

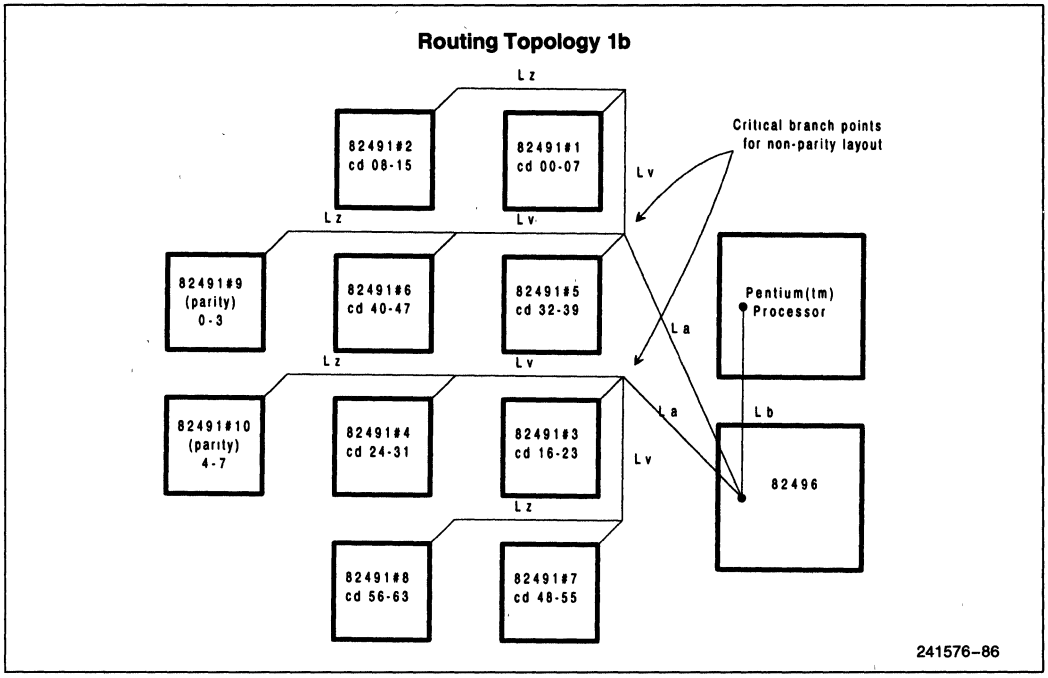


NET	CC-CS#1	CC-CS#5	CC-CS#3	CC-CS#7	CS#1-CS#2	CS#5-CS#6
BLEC#	4222.7	4219.0	4205.8	4206.4	936.5	936.5

NET	CS#6-CS#9	CS#3-CS#4	CS#4-CS#10	CS#7-CS#8	Stubs
BLEC# (cont)	n/a	936.5	n/a	936.5	75.0-75.0

PP=Pentium processor
CC=82496 cache controller
CS=82491 cache SRAM

7.6.9.7 82496 Control (Topology 1b Pentium™ Processor and 82496 Switch Positions)

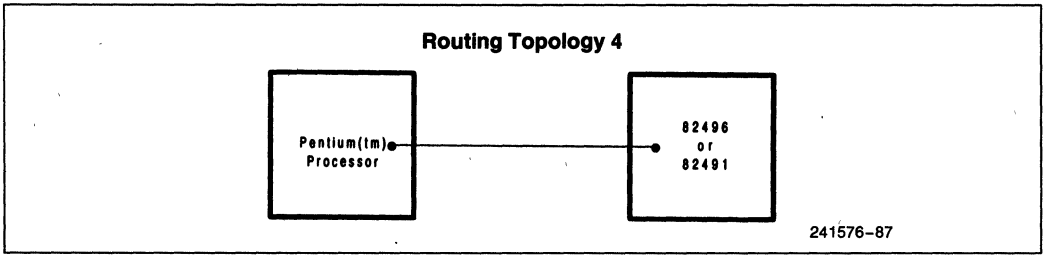


NET	CC–CS#5	CC–CS#3	CC–PP	CS#5–CS#1	CS#1–CS#2	CS#5–CS#6
BOFF#	3612.7	3596.1	7605.8	936.5	936.5	936.5

NET	CS#6–CS#9	CS#3–CS#4	CS#4–CS#10	CS#3–CS#7	CS#7–CS#8	Stubs
BOFF# (cont)	936.5	936.5	936.5	944.8	936.5	75.0–75.0

PP = Pentium processor
CC = 82496 cache controller
CS = 82491 cache SRAM

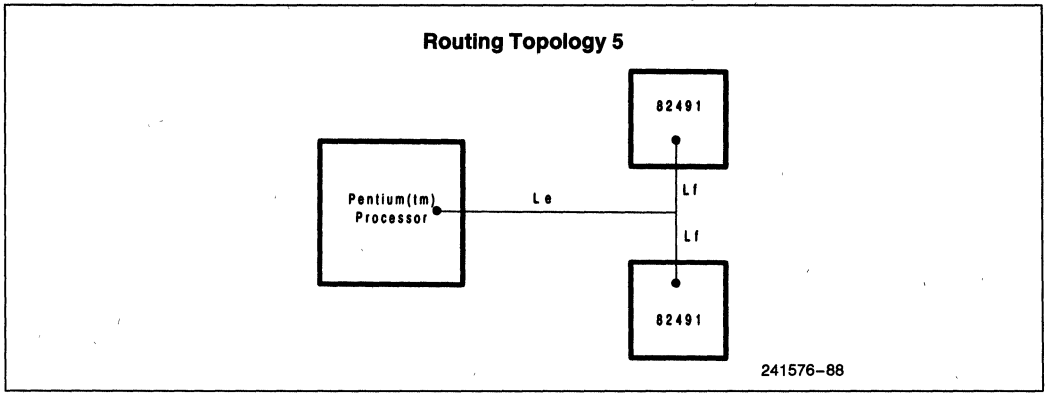
7.6.9.8 82496 Control (Topology 4, Point-to-Point)



NET	CC-PP
AHOLD	4549
BRDYC1#	3648.5
EADS#	3656.4
INV	4603.5
KEN#	4136.9
NA#	3770.3

PP=Pentium processor
CC=82496 cache controller
CS=82491 cache SRAM

7.6.9.9 Byte Enables (Topology 5)



NET	PP–Tee	Tee–CS#1	Tee–CS#9	Stubs
CBE0#	3035.4	1634.4	1633.9	112.4 135.3

NET	PP–Tee	Tee–CS#2	Tee–CS#9	Stubs
CBE1#	4098.7	1294.8	1293.1	75.0–121.0

NET	PP–Tee	Tee–CS#3	Tee–CS#9	Stubs
CBE2#	3412.9	1732.0	1682.3	75.0–95.3

NET	PP–Tee	Tee–CS#4	Tee–CS#9	Stubs
CBE3#	3547.8	1194.8	1192.1	75.0–75.0

NET	PP–Tee	Tee–CS#5	Tee–CS#10	Stubs
CBE4#	3600.9	2243.5	2242.0	75.0–135.3

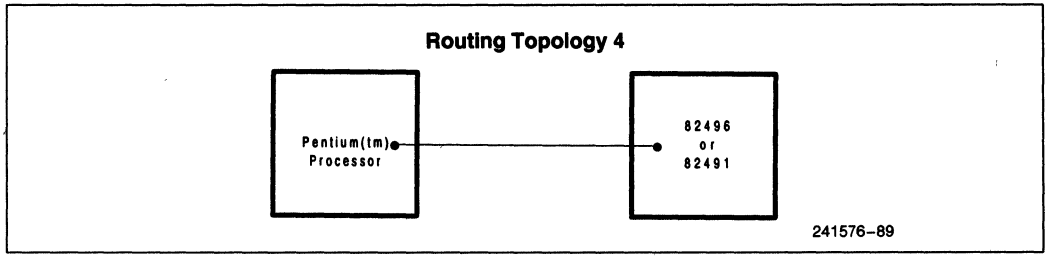
NET	PP–Tee	Tee–CS#6	Tee–CS#10	Stubs
CBE5#	4811.9	1339.7	1338.9	75.0–75.0

NET	PP–Tee	Tee–CS#7	Tee–CS#10	Stubs
CBE6#	4662.0	1663.0	1662.6	75.0–135.3

NET	PP–Tee	Tee–CS#8	Tee–CS#10	Stubs
CBE7#	4230.0	1167.7	1165.5	75.0–75.0

PP = Pentium processor
CC = 82496 cache controller
CS = 82491 cache SRAM

7.6.9.10 CDATA and Parity (Point-to-Point)



Net	PP-CS#9	Stub
CP0	5722.4	135.3
CP1	5842.1	135.3
CP2	5854.7	75.0
CP3	5712.4	75.0

NET	PP-CS#10	Stub
CP4	5831.1	135.3
CP5	5849.8	135.3
CP6	5563.8	75.0
CP7	5558.6	75.0

NET	PP-CS#1	Stub	NET	PP-CS#3	Stub	NET	PP-CS#5	Stub
CD0	5523.3	135.3	CD16	5541.4	135.3	CD32	5507.8	135.3
CD1	5376.2	135.3	CD17	5781.0	135.3	CD33	5419.5	135.3
CD2	5476.9	75.0	CD18	5550.7	75.0	CD34	5433.2	75.0
CD3	5363.3	75.0	CD19	5558.2	75.0	CD35	5756.3	75.0
CD4	5547.9	135.3	CD20	5449.7	135.3	CD36	5654.1	135.3
CD5	5393.5	75.0	CD21	5545.2	75.0	CD37	5551.6	75.0
CD6	5357.9	135.3	CD22	5410.4	135.3	CD38	5357.3	135.3
CD7	5582.3	75.0	CD23	5533.2	75.0	CD39	5451.6	75.0

NET	PP-CS#2	Stub	NET	PP-CS#4	Stub	NET	PP-CS#6	Stub
CD8	5448.4	135.3	CD24	5804.5	135.3	CD40	5430.0	135.3
CD9	5516.1	135.3	CD25	5594.4	135.3	CD41	5757.8	135.3
CD10	5629.1	75.0	CD26	5754.8	75.0	CD42	5378.1	75.0
CD11	5468.9	75.0	CD27	5705.4	75.0	CD43	5703.8	75.0
CD12	5451.5	135.3	CD28	5774.5	135.3	CD44	5462.3	135.3
CD13	5543.2	75.0	CD29	5737.7	75.0	CD45	5755.1	75.0
CD14	5707.3	135.3	CD30	5380.7	135.3	CD46	5568.6	135.3
CD15	5420.3	75.0	CD31	5608.3	75.0	CD47	5757.2	75.0

NET	PP-CS#7	Stub
CD48	5488.3	135.3
CD49	5551.8	135.3
CD50	5697.9	75.0
CD51	5581.1	75.0
CD52	5499.5	135.3
CD53	5704.4	75.0
CD54	5493.6	135.3
CD55	5627.9	75.0

NET	PP-CS#8	Stub
CD56	5553.9	135.3
CD57	5663.3	135.3
CD58	5602.7	75.0
CD59	5718.3	75.0
CD60	5451.6	135.3
CD61	5533.9	75.0
CD62	5550.4	135.3
CD63	5766.2	75.0

PP = Pentium processor
CC = 82496 cache controller
CS = 82491 cache SRAM

7.6.10 Pentium™ PROCESSOR TO 82496 SEGMENT LENGTH AND ROUTING CHANGES

The example layout described in this application note was completed using early revisions to the I/O buffer models. This process was necessary to ensure that a board was available for the arrival of first silicon. After the models were improved based on the model validation and silicon characterization, the board layout was resimulated. These simulations have resulted in the recommendation to change the line length between the Pentium processor and 82496 for several nets. These changes result in a better tuned routing that meets the specifications. In particular, these changes reduce the amount of ringback and the ringing that leads to long settling times. Table 12 summarizes the recommended segment length changes.

Table 12. Summary of Segment Lengths

Net/Signal Name	Segment	Original Length (in.)	Recommended Length (in.)
WRARR#	CC-CS#3	2.802	2.9
WRARR#	CC-CS#5	2.735	2.9
PA4	PP-CC	4.673	4.3
PA6	PP-CC	5.123	4.9
PA7	PP-CC	5.368	5.1
PA10	PP-CC	5.233	4.9
PA12	PP-CC	5.324	5.0
PA16	PP-CC	5.359	4.9

PP = Pentium processor
CC = 82496 cache controller
CS = 82491 cache SRAM

Actual system measurements have shown that the original segment lengths do not violate the specifications. The reduction in ringing is probably due to transmission line losses which are not accounted for in the simulation. Therefore for completed designs using the example layout these changes are not necessary; however, Intel does recommend that all future designs that use the layout example use these new lengths.

In addition, a layer change to the BRDYC1# routing is recommended. Section 7.5 Design Notes, describes that BRDYC1# was routed on an outer layer to reduce the propagation delay; however, this resulted in a signal quality violation. Since the original routing of the board, the flight time specification was relaxed and BRDYC1# can now be routed on an inner layer which allows it to meet both signal quality and flight time specifications.

7.6.11 I/O SIMULATION RESULTS FOR EACH NET

Electrical simulations were performed on each net within the optimized interface of the 256 Kbyte CPU-Cache Chip Set design example. The simulations were done at the fast and slow corners to verify that signal quality and flight time specifications are met. The simulations were done using TLC V4.1.13 from Quad Design Technology, Inc. using the files described in Section 7.6.2. Table 13 summarizes the simulation results assuming all the segment length changes listed in Section 7.6.10 have been implemented along with the layer change to the BRDYC1# routing.

Table 13. Summary of Simulation Results

Net	Flight Time (ns)	Signal Quality			
		Over/ Undershoot (V)	Ringback (V)	Settling Time (ns)	Time Beyond Supply (ns)
Specs.	**Vary by Pin**	**3.0**	**1.75**	**12.5**	**6.0**
82496 Driving					
A3–16 at CPU	7.2	2.1	1.0	16.5	7.4
A3–16 at SRAM	7.0	2.1	1.0	16.5	7.4
A17–31 at CPU	2.6	3.0	1.75	10.3	3.6
BT0–3	2.6	3.0	1.75	10.3	3.6
AHOLD	1.1	3.0	1.75	9.0	2.1
AP	1.4	3.0	1.75	9.0	2.1
BRDYC1#	0.9	2.9	1.7	8.3	1.8
EADS#	0.9	2.9	1.7	7.8	1.8
EWBE#	0.7	2.5	1.4	6.1	1.5
INV	1.6	2.8	1.5	12.4	2.9
KEN#	1.0	2.9	1.75	8.5	2.0
NA#	0.9	2.9	1.7	7.9	1.9
WB/WT#	0.9	2.8	1.7	7.5	1.8
BLAST#	2.5	2.2	1.0	6.2	3.3
BLEC#	2.2	2.1	0.9	5.9	3.1
BOFF# at CPU	2.5	2.6	1.1	12.1	3.0
BOFF# at SRAM	2.9	2.6	1.1	12.1	3.0
BRDYC2#	2.5	2.2	1.1	6.2	3.7

Table 13. Summary of Simulation Results (Continued)

Net	Flight Time (ns)	Signal Quality			
		Over/ Undershoot (V)	Ringback (V)	Settling Time (ns)	Time Beyond Supply (ns)
BUS#	2.5	2.1	0.9	6.5	3.3
MAWEA#	2.6	2.2	1.0	8.1	3.4
MCYC#	2.6	2.1	0.8	6.5	3.3
WAY	2.5	2.1	0.9	6.4	3.6
WBA	2.7	2.2	1.0	6.7	3.5
WBTYP	2.6	2.2	1.0	6.1	3.6
WBWE#	2.6	2.1	0.9	6.2	3.3
WRARR#	2.5	2.4	1.1	8.1	3.3
Pentium™ Processor Driving					
A3–16 at Controller	2.5	2.6	1.2	12.8	3.2
A3–16 at SRAM	2.8	2.6	1.2	12.8	3.2
A17–31	1.5	3.0	1.8	10.6	2.3
BT0–3	1.5	3.0	1.8	10.6	2.3
D0–63, DP0–7	1.2 min. 1.4 max.	3.0	1.8	11.5	2.5
ADS#	2.6	2.2	1.0	7.3	3.7
HITM# at Controller	3.0	3.2	1.6	20.3	4.1
HITM# at SRAM	3.2	3.2	1.6	20.3	4.1
W/R# at Controller	3.0	3.1	1.6	20.7	4.1
W/R# at SRAM	3.2	3.1	1.6	20.7	4.1
ADSC#	1.2	3.0	1.75	7.3	1.7
AP	1.3	3.0	1.75	8.8	2.1
CACHE#	1.1	2.9	1.75	7.2	1.7
D/C#	1.1	2.9	1.75	7.1	1.7
LOCK#	1.3	3.0	1.8	8.6	2.0
M/IO#	1.4	3.0	1.8	9.1	2.1

Table 13. Summary of Simulation Results (Continued)

Net	Flight Time (ns)	Signal Quality			
		Over/ Undershoot (V)	Ringback (V)	Settling Time (ns)	Time Beyond Supply (ns)
PCD	1.1	2.9	1.7	6.9	1.6
PWT	1.2	3.0	1.8	8.7	1.9
SCYC	1.2	3.0	1.75	7.4	1.7
BE0–7#	1.9	3.5	2.0	14.7	3.3
82491 Driving					
D0–63, DP0–7	1.7 min. 1.9 max.	3.0	1.9	12.1	2.7

Shading indicates a spec. violation.

The shaded entries indicate specification violations with the example design layout. Intel is continuing to work on addressing these violations by either relaxing specifications or improving the layout design. Note that no violations have been measured on the actual board. The violations have all been in simulation.

7.7 Possible Modification to the Layout

7.7.1 NON-PARITY LAYOUT

Intel has not simulated a non-parity layout example. The following suggestions will assist in modifying the design example for non-parity implementations. You must simulate all paths that are altered when the parity components are removed to ensure that flight time and signal quality specifications are still met.

Modify the following aspects of the layout example:

1. Remove the two leftmost 82491 components, U9 and U10. These are the parity components.
2. Rework Topologies 1 and 1b. Balance the array so that the two critical branch points branch out to electrically equivalent traces, i.e., adjust Lw to be electrically equivalent to Lv + Lz. Keep the trace leading to the Pentium processor length La. Topologies 1NP and 1bNP illustrate this (NP = Non-Parity). Also, retune Lb to be electrically equivalent with these new trace lengths. Topologies 1 and 1b indicate exactly where the critical branch points are.
3. Rework topologies 3 and 3b. Make the four traces branching from the 82496 electrically equivalent. This may be accomplished by making Lg = Lh for these topologies.
4. Remove the Byte Enable traces that connect to the parity chips.

Making traces electrically equivalent means that reflections from all branches return to the source at the same point in time. In simple cases, electrically equivalent traces are the same length. In all cases, simulate the effects of changing trace lengths to find the proper trace length and routing.

8.0 512K CPU-CACHE CHIP SET OPTIMIZED INTERFACE LAYOUT DESIGN EXAMPLE

This chapter contains an example layout design for Intel's 512 Kbyte CPU-Cache Chip Set's optimized interface. Intel has simulated and verified the example layout using the latest information. Work is currently underway to validate the design by measuring the flight time and signal quality parameters on boards based on the design example. As updated information becomes available on the components and the boards, Intel plans to update this example accordingly.

The intent of the design example is to provide system designers a starting point. It provides one solution of how the Pentium processor, 82496, and 82491 components can be placed and routed to ensure flight time and signal quality specifications are met. It is not the only solution. System designers can alter the layout to meet their system requirements as long as the flight time and signal quality specifications are met.

8.1 Layout Objectives

The 512K layout is an example of a CPU-Cache chip set arrangement that meets Intel's chip set specifications. The layout consists of 1 Pentium processor, 1 82496 cache controller, and 18 82491 cache SRAMs for a 512K second-level cache with parity. Although the layout is specifically designed for a chip set with parity, we will also discuss conversion to a non-parity layout.

This example layout targets the chip set's flight time and signal quality specifications. In addition to meeting those specifications, we had the following objectives:

1. To design the optimized portion of the interface so that the layout is not limited by interconnect performance. By not artificially creating any critical paths, the interface can yield maximum performance of the chip set.
2. To be consistent with EMI and thermal requirements.
3. To have the layout be used as a validation and correction vehicle by Intel. Intel will use the layout to validate the optimized interface of the chip set, measure flight times and signal quality, and tune input and output buffers.

Provided are complete specifications for a board layout: part lists, board layer plots, and the electronic files in Gerber format. Also provided are a set of topologies and line lengths so it will be easy to understand how the layout was generated.

8.2 Component Placement

To meet flight time with clock skew restrictions we placed the parts in relative proximity to each other. At the same time, we ensured that the layout's Memory Bus Controller (MBC) interface signals are routable. Figure 59 illustrates how the chip set components are placed in the layout example. The dot indicates the location of pin 1. Figure 59 shows a component side view of the layout.

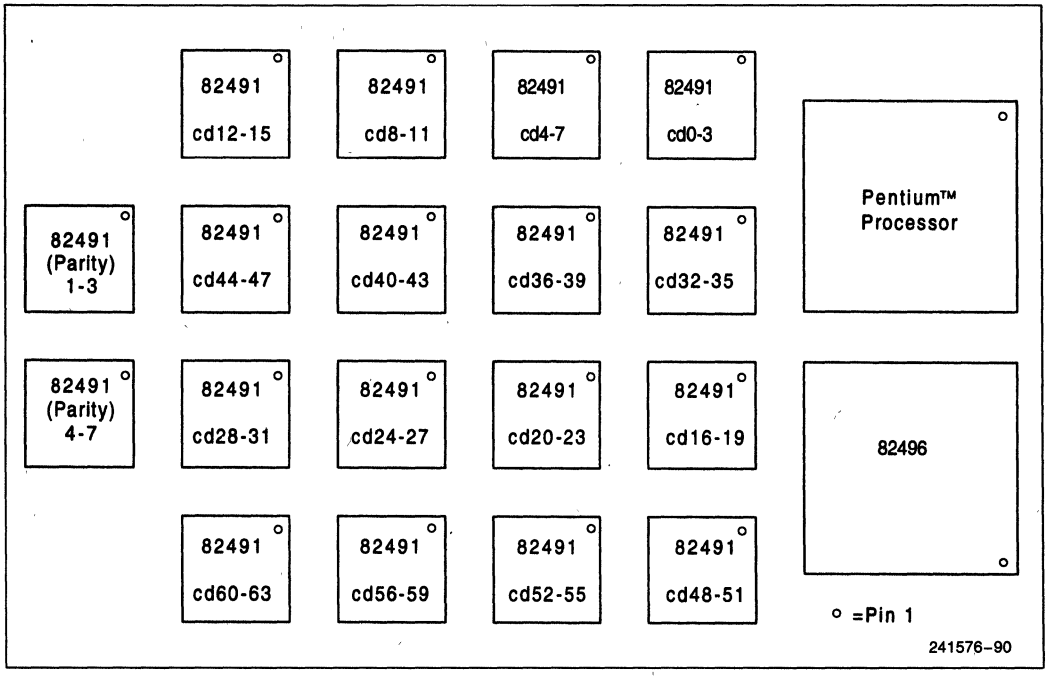


Figure 59. Component Placement

8.3 Signal Routing/Topologies

Table 14 and Table 15 list the signal nets and their corresponding topologies for the optimized and external interfaces of the CPU-Cache Chip Set.

All chip set signals in the optimized interface fall into six groups: low addresses, high addresses, Pentium processor control, 82496 control, CPU data, and byte enables. Within each group are subsets of signals that share common origination and destination points. Each subset has a unique routing called a "topology." Groups, subsets, and topologies are listed in Table 14.

Topologies are given only for signals that are routed to multiple chips. It is the system designer's responsibility for routing the "point-to-point" signals such as CADS#.

Topologies are also supplied for the external interface. These topolgies provide channels for routing signals from the chip set components to the periphery where they can be connected to the memory bus and memory bus controller (MBC). However, topologies are not supplied for point to point signals in the MBC interface (e.g. CRDY#). Instead, the system designer must optimize these for the particular application.

Table 15 lists the topologies provided for the MBC interface signals which are not point to point.

Figures 60 through 77 are the topologies which are described in Table 14 and 15. A topology is a graphical representation how specific sets of signals are routed. A topology shows the components that share a specific signal and the relative lengths of the traces between components.

Table 14. Optimized Interface Signal Net/Topology Assignments

Grouping	Routing Requirements	Topology
Low Addresses		
(PA3–PA17)	Bused to all core components. Must be routed to optimize delay and signal quality at all points.	1
High Addresses		
(PA18–PA31, PBT0–PBT3)	Point to point links. Must be kept as short as possible.	6
Pentium™ Processor Control		
(HITM#, W/R#)	Same as low addresses.	1
(ADS#)		5
(ADSC#, AP, CACHE#, D/C#, LOCK#, M/IO#, PCD, PWT, SCYC)	Same as high addresses.	6
CC Control		
(BUS#, MAWEA#, WBWE#, WBTYP#, WBA, BLAST#)	Must be routed to optimize delay and signal quality at the CS.	3
(BLEC#)	Not routed to parity CSs.	4
(BOFF#)		1
(AHOLD, EADS#, KEN#, BRDYC1#, INV, EWBE#, NA#, WB/WT#)	Same as high addresses.	6
(BRDYC2#, WRARR#, MCYC#, WAY)	Routed differently.	15
CPU Data		
(CD0–CD63, CP0–CP7)	Point to point signals. Keep as short as possible. Keep within 3" of each other to minimize skew.	6
Byte Enables		
(CBE0#–CBE7#)		7

Table 15. External Interface Signal Net/Topology Assignments

Signal	Topology
MDATA0–63, Parity0–7	8
BRDY0#, CLK0	9
CRDY#	10
RESETC	17
MBRDY#, MOCLK, MDOE#	11
BRDY1–3#	12
MEOC1–3#	18
CLK1–3	13
MFRZ#, MSEL#, MZBT#, MCLK	14
RESETCPU	16

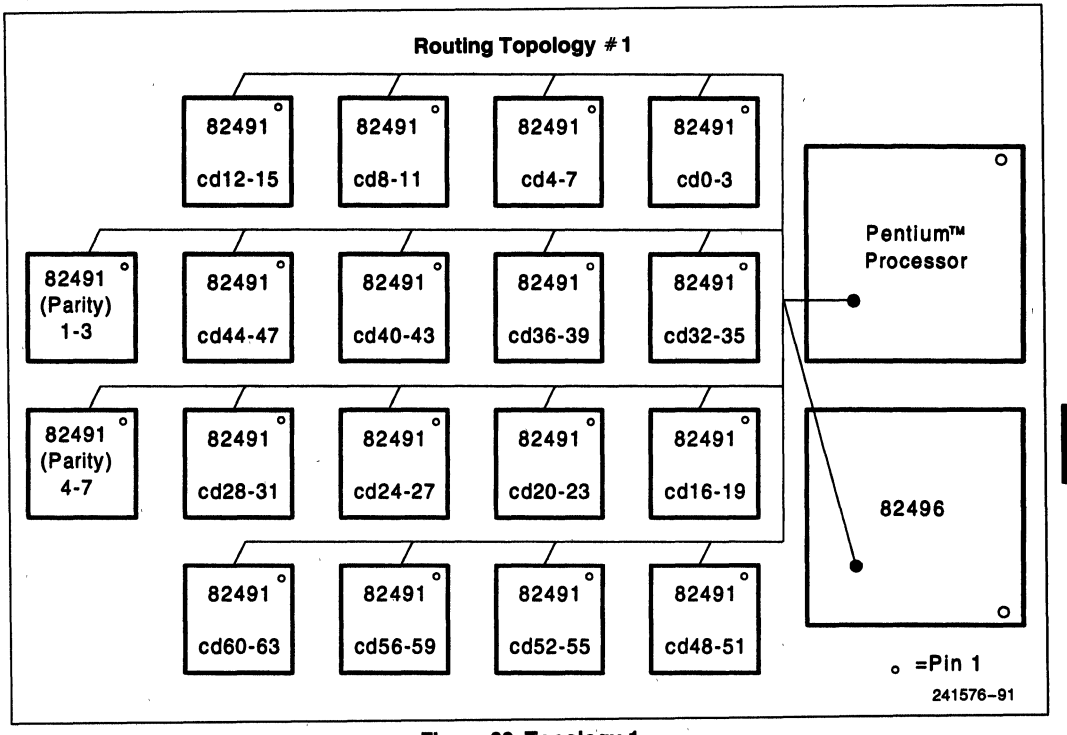


Figure 60. Topology 1

2

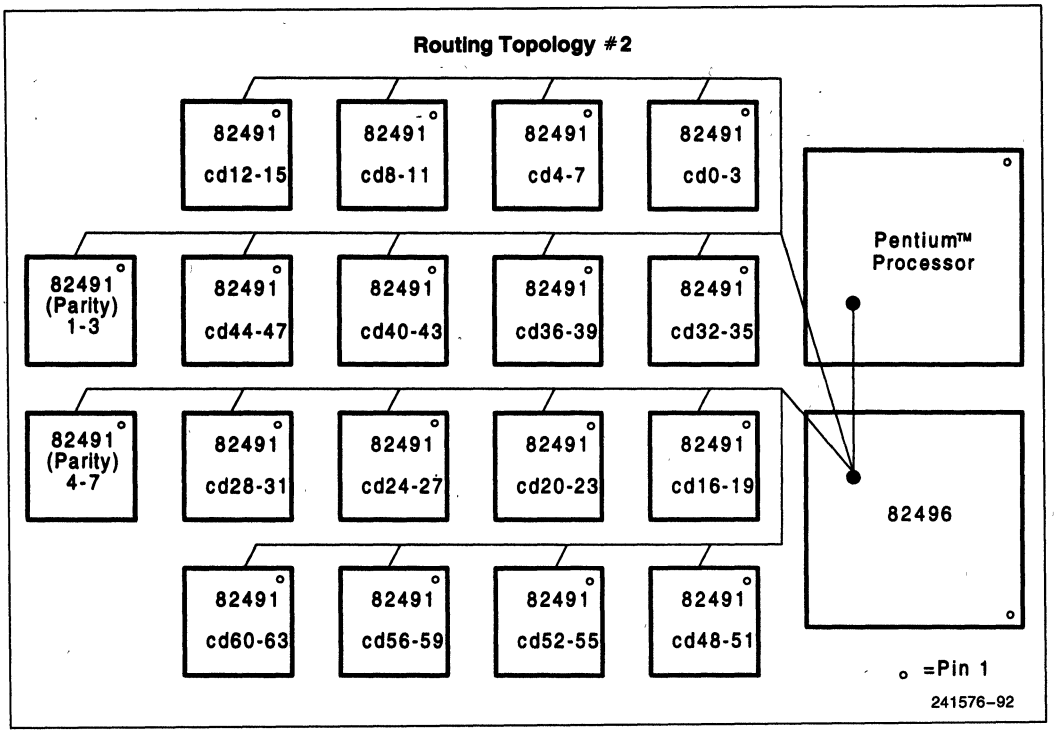


Figure 61. Topology 2

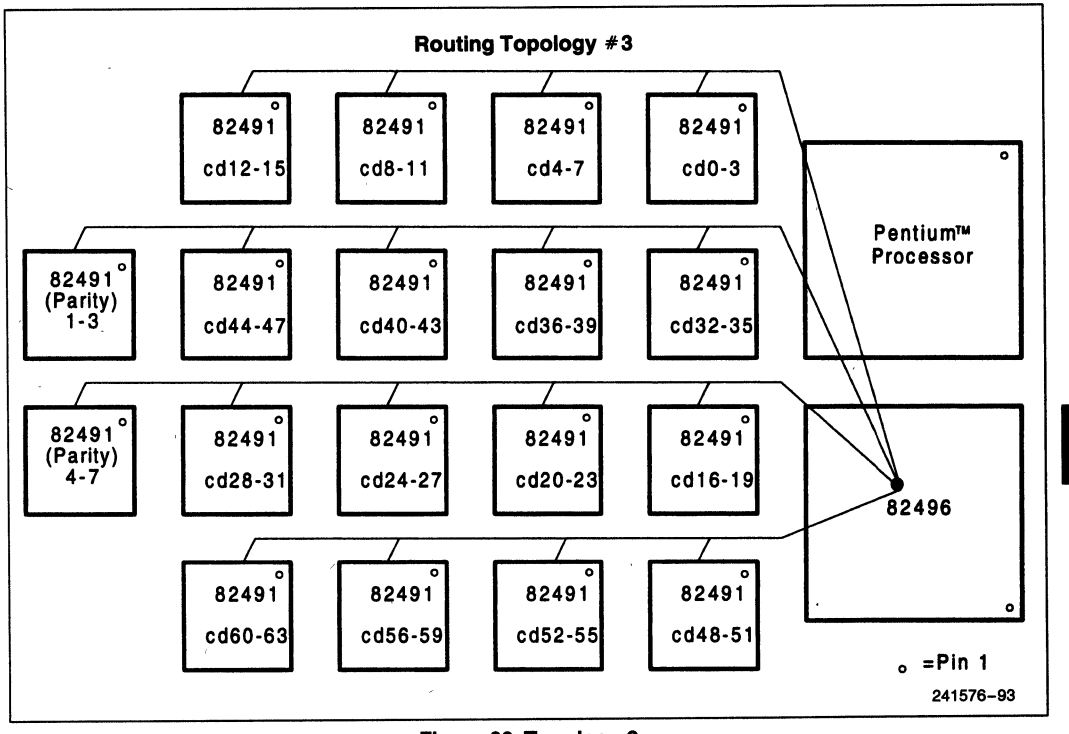


Figure 62. Topology 3

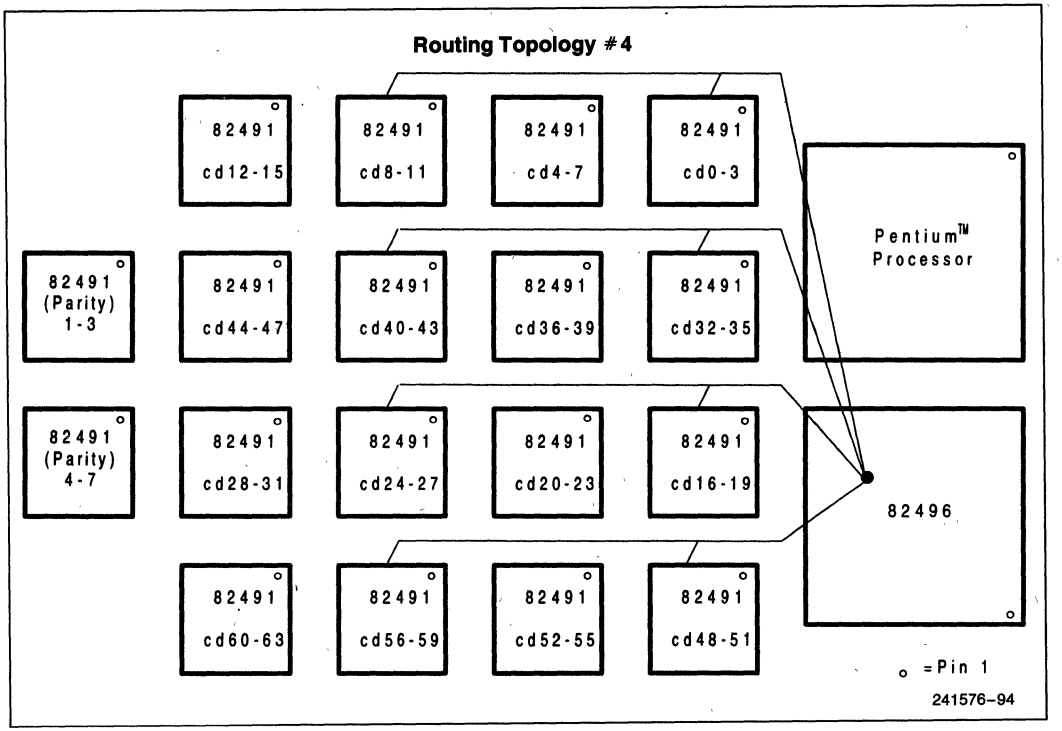


Figure 63. Topology 4

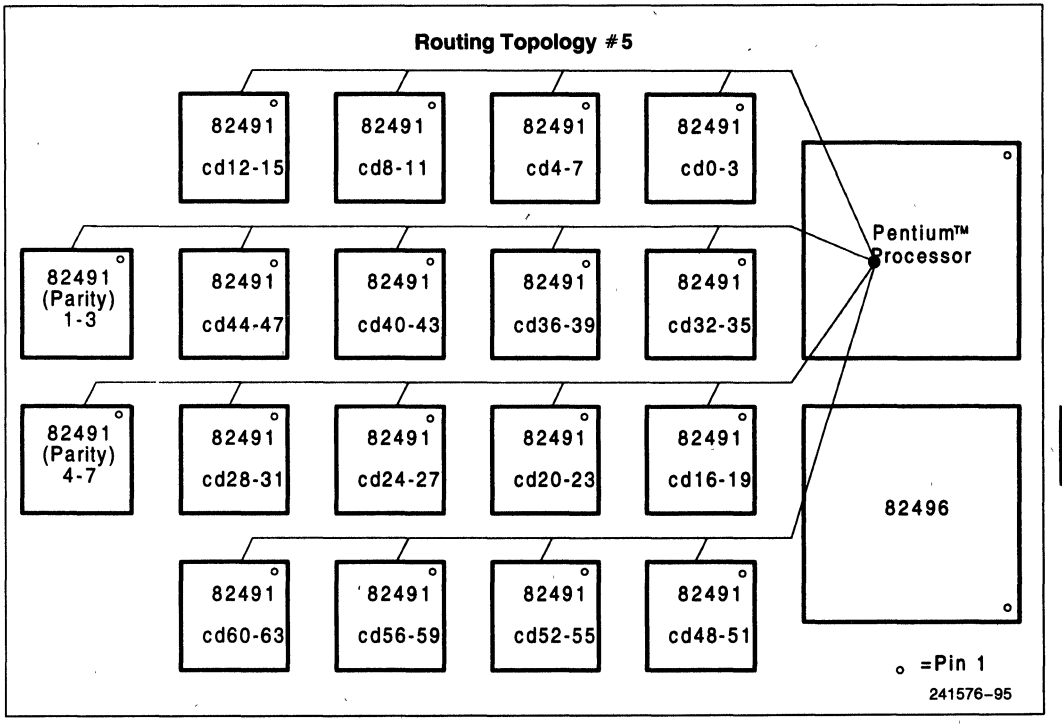


Figure 64. Topology 5

2

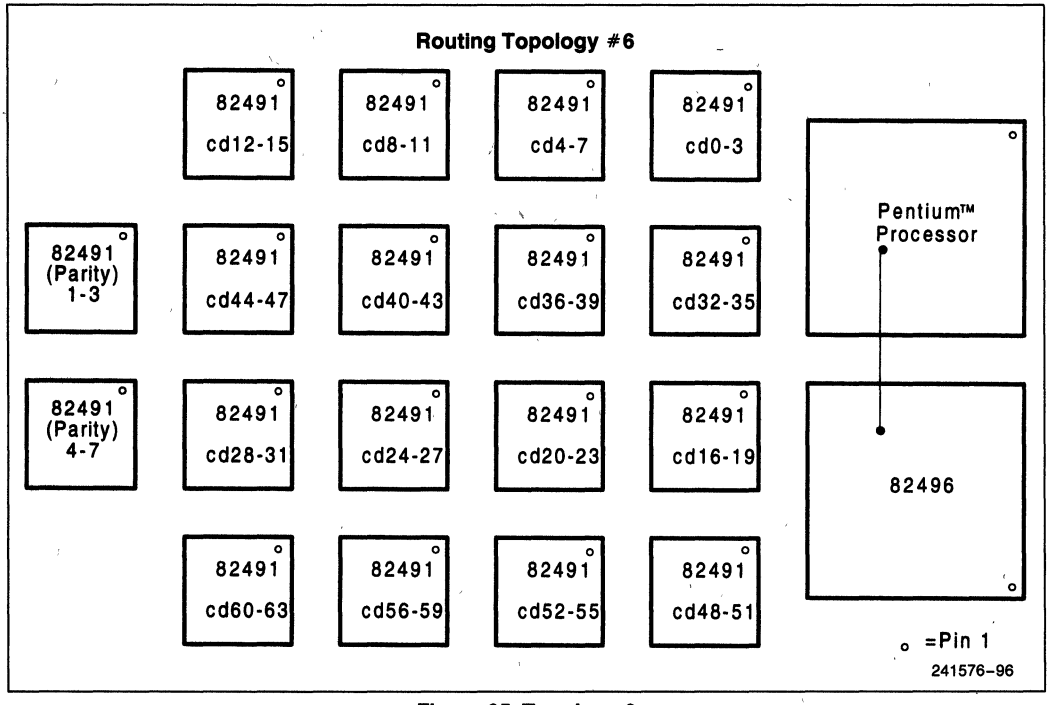


Figure 65. Topology 6

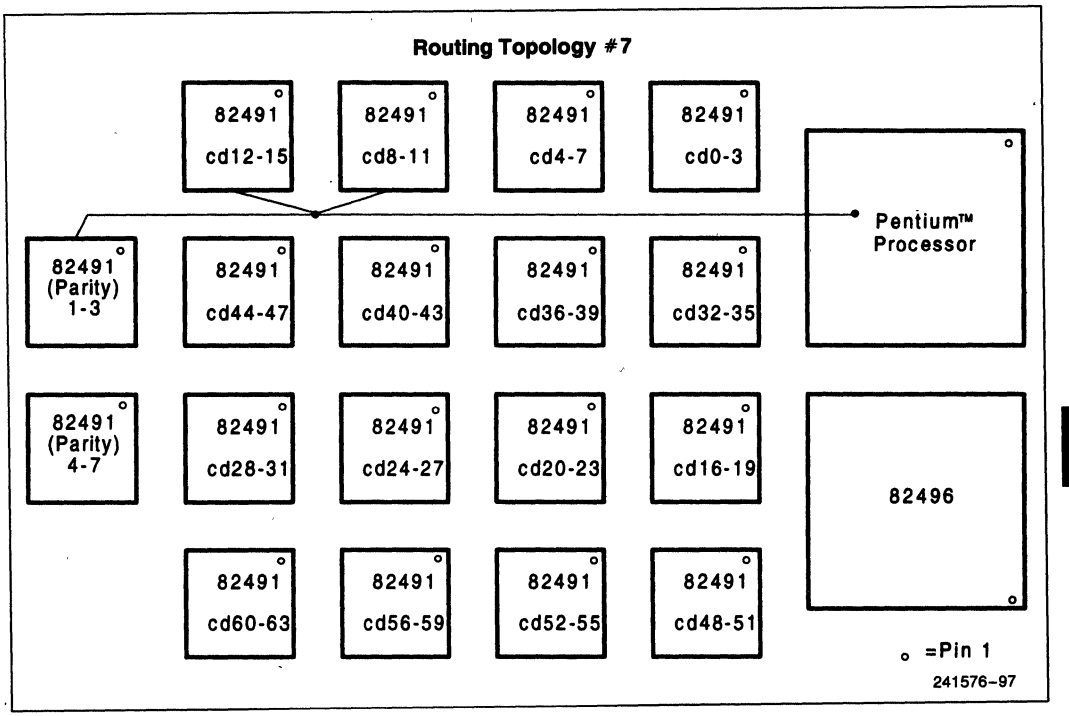


Figure 66. Topology 7

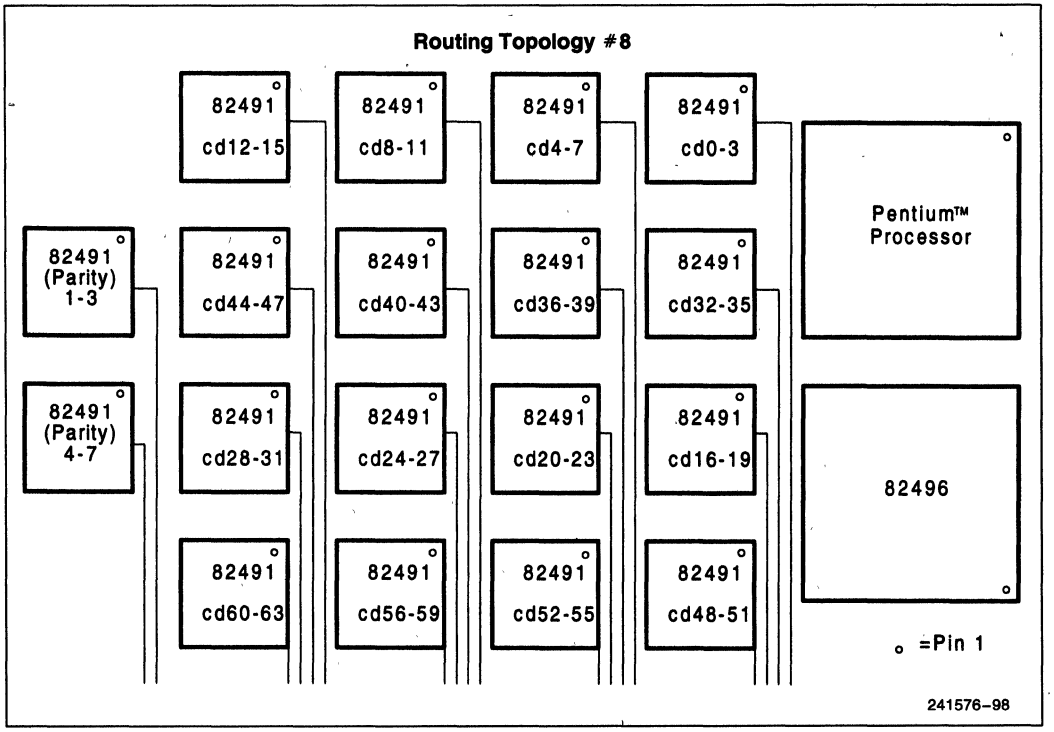


Figure 67. Topology 8

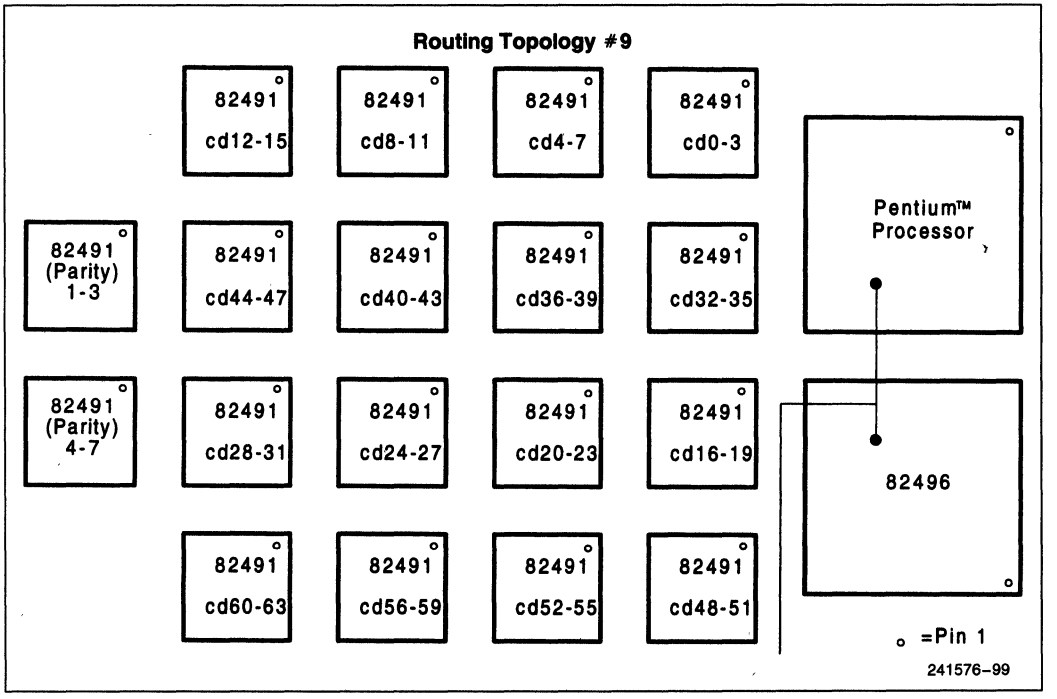


Figure 68. Topology 9

2

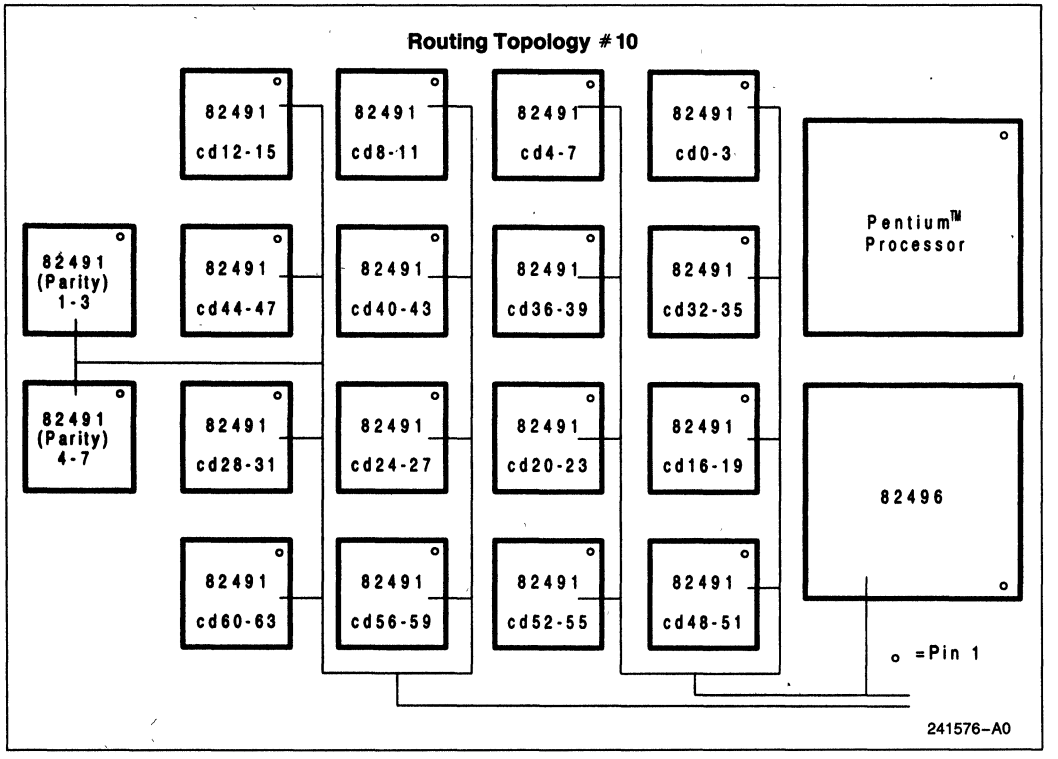


Figure 69. Topology 10

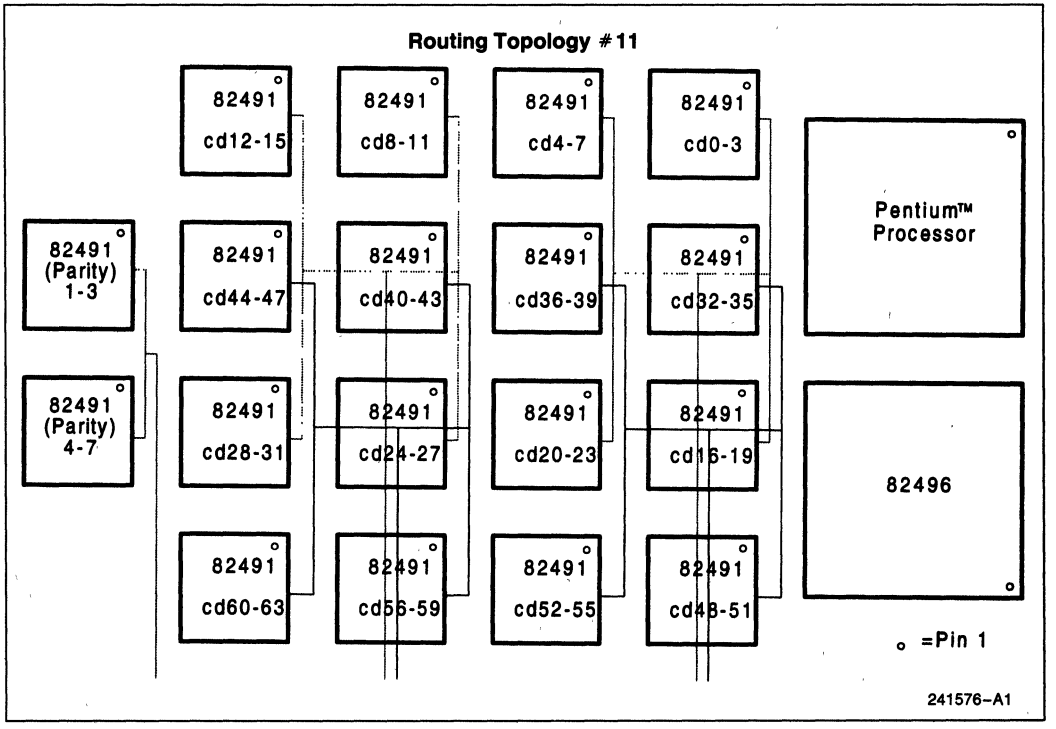


Figure 70. Topology 11

2

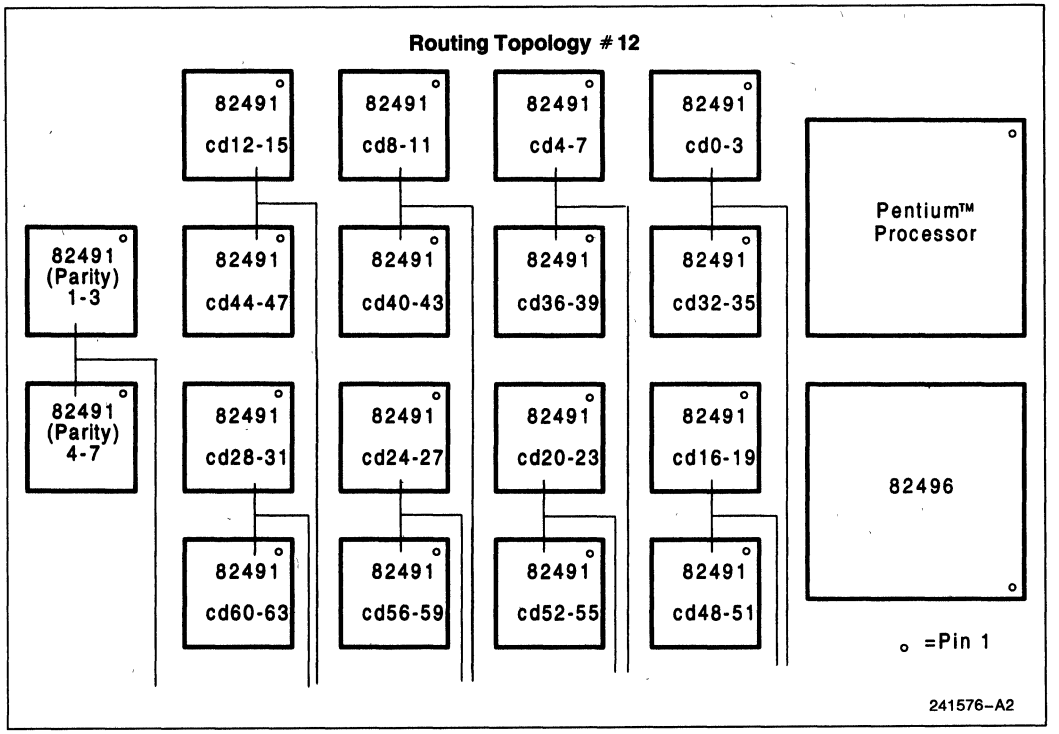


Figure 71. Topology 12

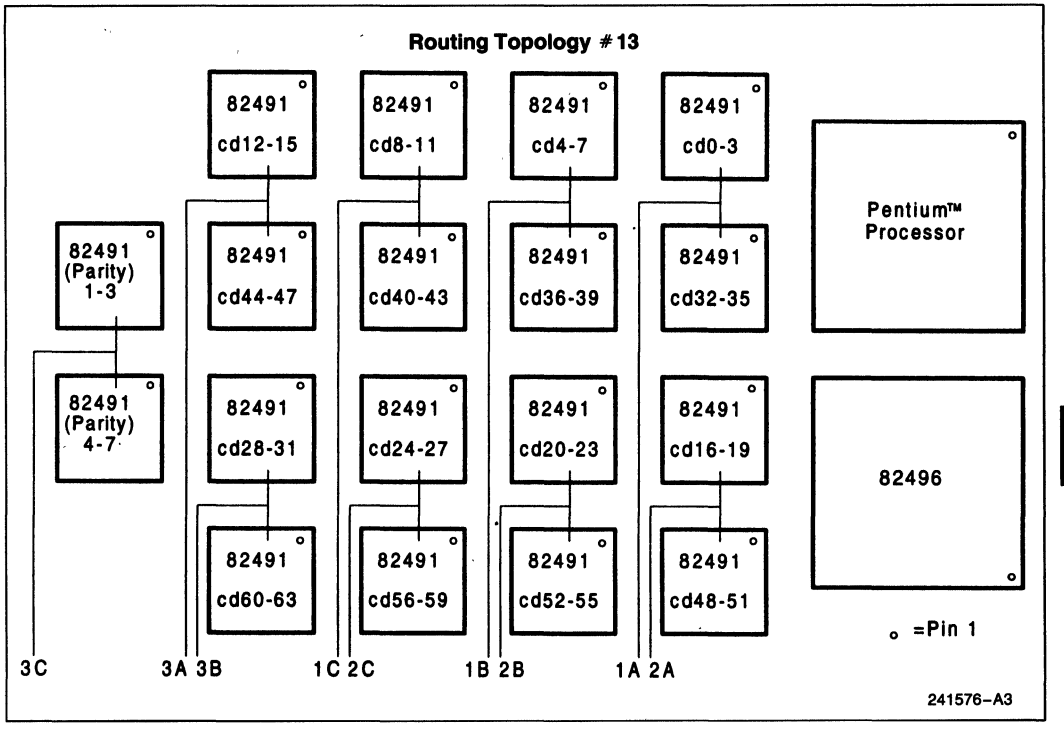


Figure 72. Topology 13

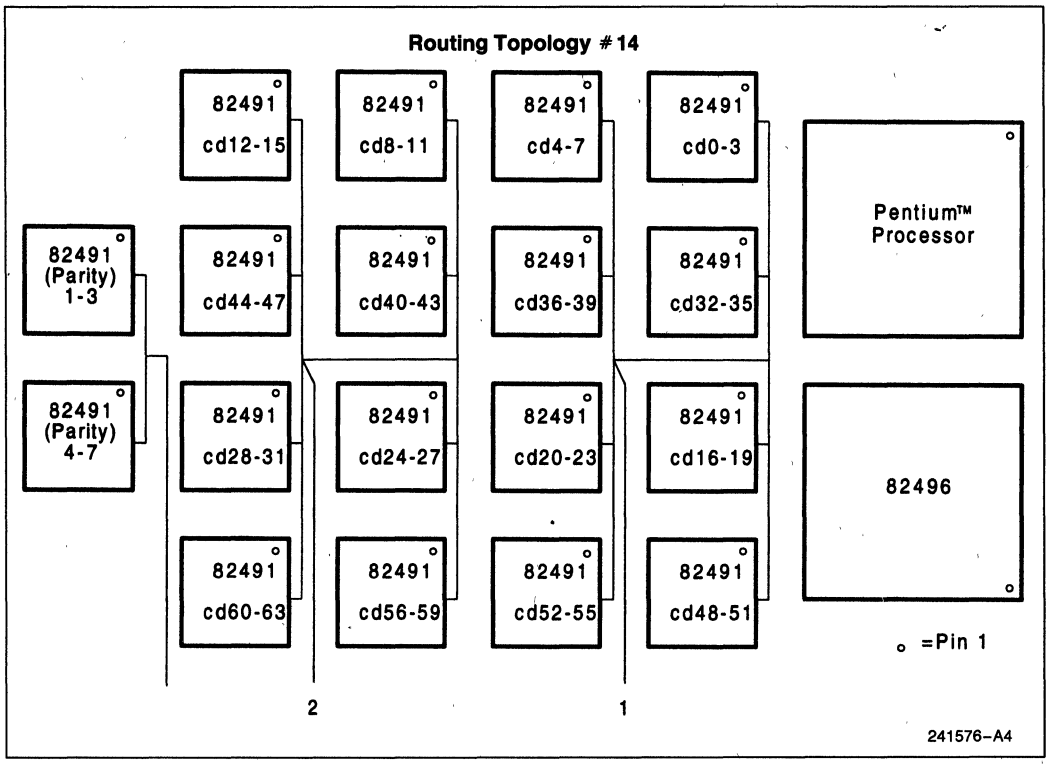


Figure 73. Topology 14

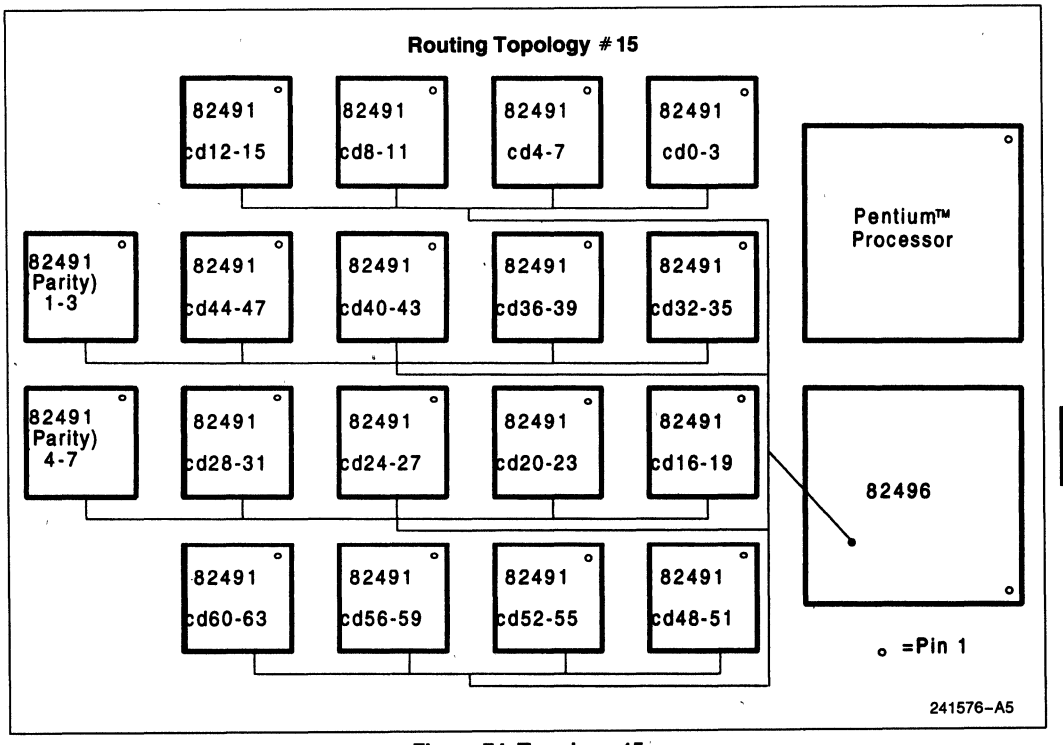


Figure 74. Topology 15

2

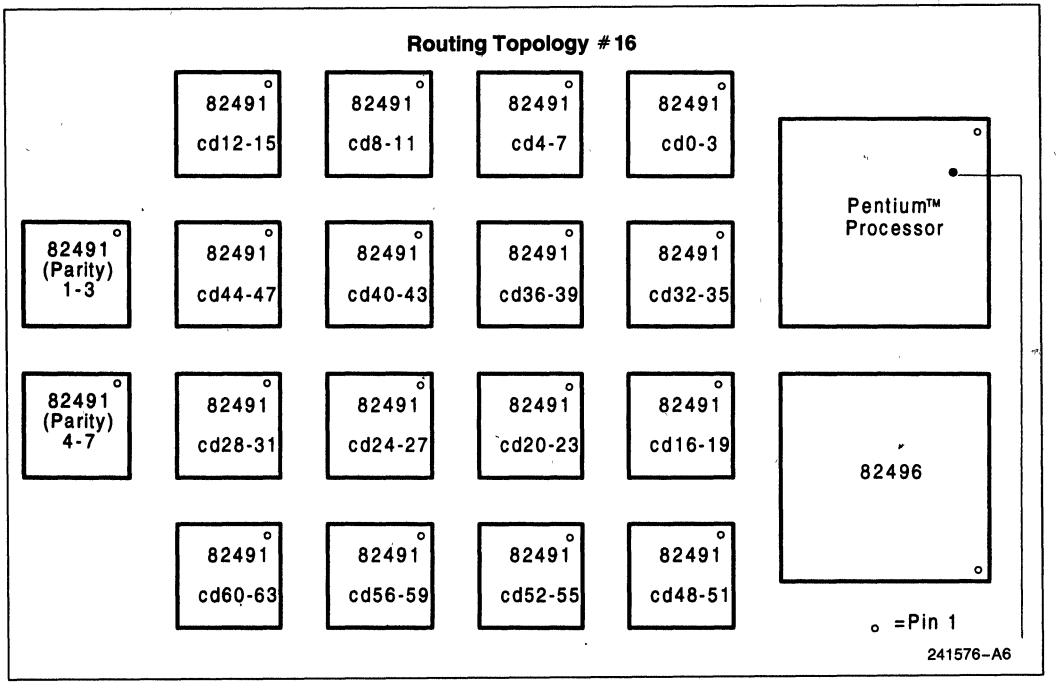


Figure 75. Topology 16

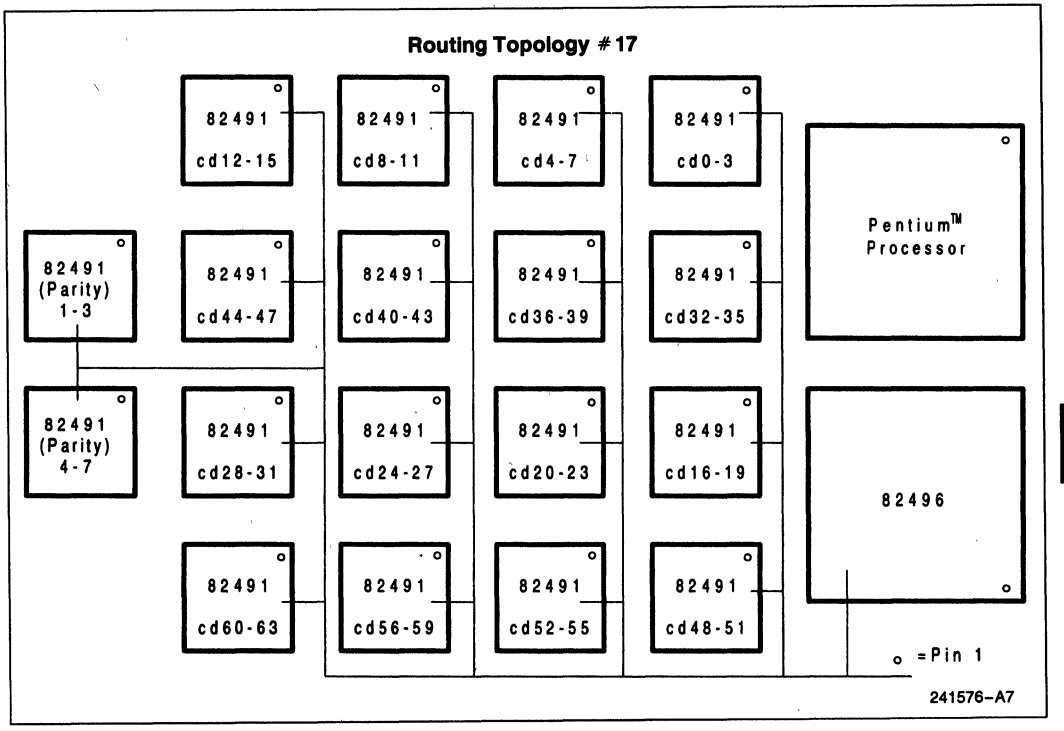


Figure 76. Topology 17

2

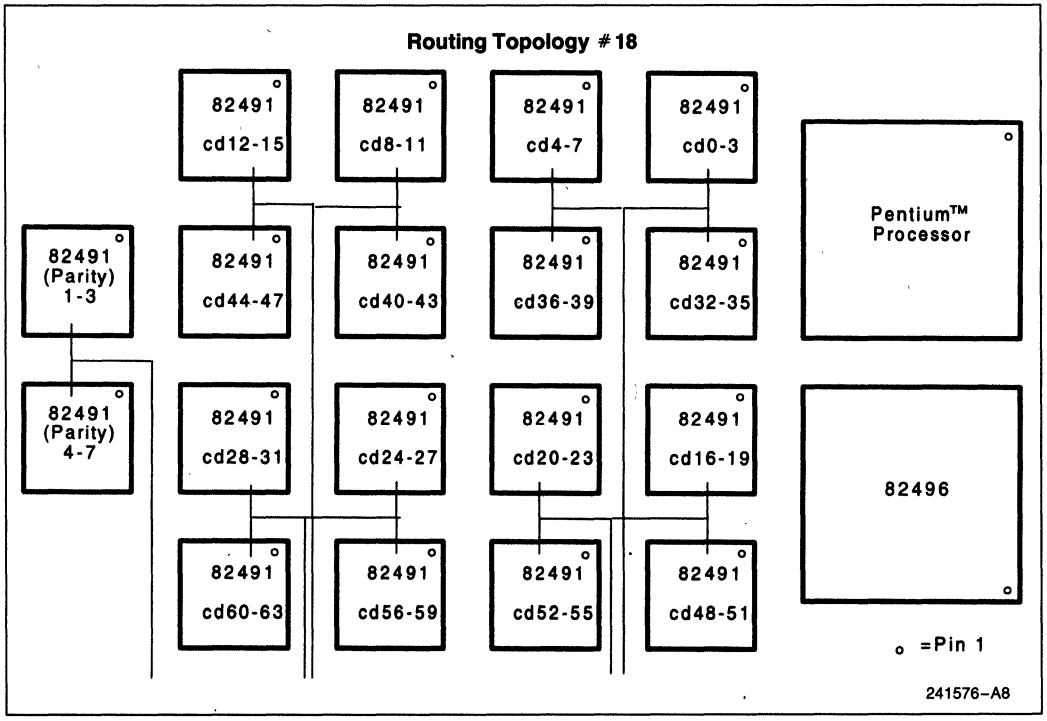


Figure 77. Topology 18

8.4 Board/Trace Properties

Specific board and trace properties were assumed while performing the simulations to optimize the chip set layout. These properties were used as the specification or guideline the board manufacturer was to use in building boards. Figure 78 provides the board layer stackup.

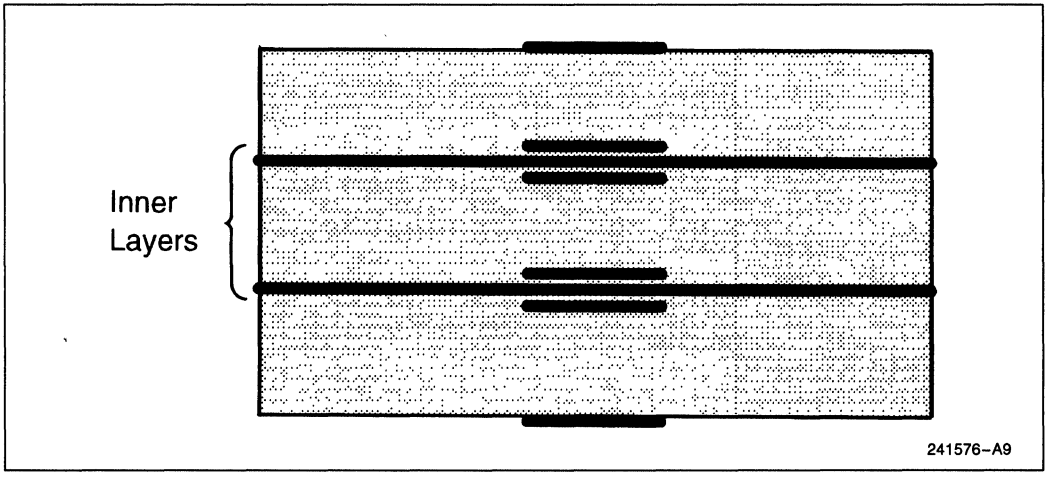


Figure 78. Board Layer Stackup

Table 16 lists the minimum and maximum trace characteristics. These parameters along with the board material determine the spacing between layers and the total board thickness. See Table 17.

Table 16. Trace Characteristics

	4 Inner Layers	2 Outer Layers
Width/Space	5/5 Mils	8/8.5 Mils
Z_0	65W ±10%	90W ±20%
Velocity	1.85 to 2.41 ns/ft	1.35 to 2.05 ns/ft

Table 17. Other Printed Circuit Board Geometries

Via Pad	25 Mils
Via Hole	10 Mils
PGA Pad	55 Mils
PGA Hole	38 Mils
Layout Grid	5 Mils

Only the inner layers of the board are impedance controlled. The top and bottom layers are not impedance controlled.

8.5 Design Notes

The following design notes accompany this layout example:

1. The layout did not specifically address heat dissipation except to allow space for heat sinks to be attached. Please see the *Pentium™ Processor User's Manual* for the devices' thermal specifications. The *Pentium™ Processor Thermal Design Guidelines* application note provides some examples of possible thermal solutions.
2. All fast-switching signals are routed near the power and ground planes on inner layers of the board to minimize EMI effects. However, two sets of signals are routed on the top layer of the board: BRDYC1#, and JTAG signals. BRDYC1# is routed on top to take advantage of the higher trace velocity there. JTAG signals are routed on the top layer because they are low-speed signals and will probably be re-routed by each customer to suit individual needs.
3. Resistor R5 (0Ω) is used to set the Pentium processor configurable output buffers (A3–A20, ADS#, W/R#, and HITM#). When the resistor is included the buffers are set to the Extra Large size. When it is not included (BUSCHK# internally pulled high) the buffers are set to Large size. Intel currently recommends the x-large buffers be used for the 512K layout example. The 0Ω resistor should be designed into your design as Intel may change the recommended buffer size once silicon and the system design have been characterized.

4. The 82496 output buffers that drive the 82491 inputs must also be configured to be x-large. This is done by using a pulldown resistor on 82496 CLDRV[BGT#] (pin N04). 82496 and 82491 Memory Bus buffer sizes must be controlled by the Memory Bus Controller.
5. Series termination resistors were added to the nets PA18, PA19, and PA20 to control overshoot. A value of 24W is currently recommended, but that value may change when overshoot is measured on an actual board.

8.6 Explanation of Information Provided

The following sections outline the design files associated with the 512 Kbyte CPU-Cache Chip Set design example that are available from Intel. These files are provided to simplify the task of porting the design example into a specific design. By using these files, designers may eliminate or minimize the amount of duplicate effort when using the design example as the basis for their design. The following items are provided:

- Schematics
- I/O Model Files
- Board Files
- Bill of Materials
- Photoplot Log
- Netlist Report
- Placed Component Report
- Artwork for Each Board Layer
- Trace Segment Line Lengths

Hard copies of the schematics and trace segment line lengths are provided in the following sections. ASCII or soft copies of all the information are available from Intel by requesting order number 241663, *AP-481 Design Diskettes.*

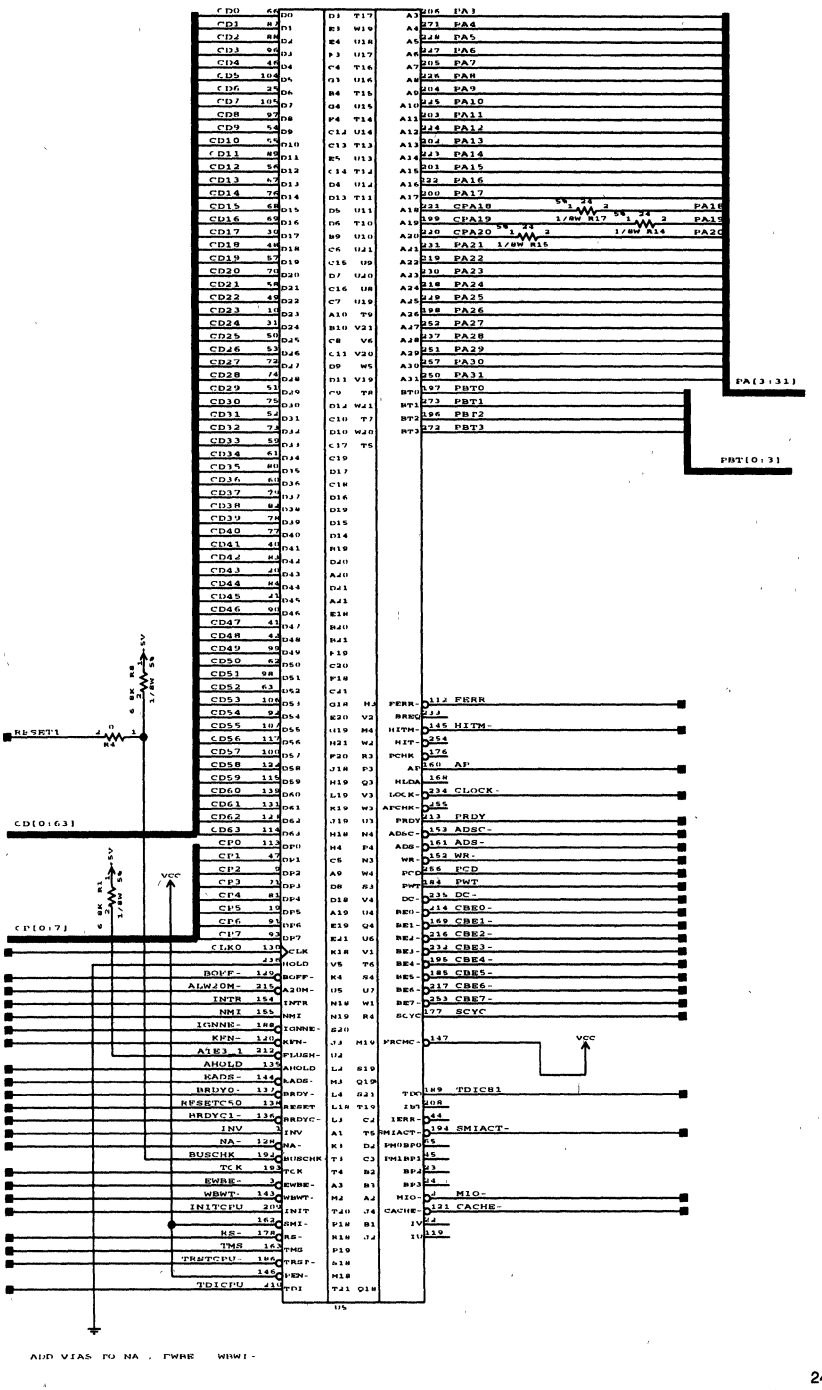
8.6.1 SCHEMATICS

Schematics for the 512 Kbyte CPU-Cache Chip Set design example were created using ViewLogics's Workview V4.1. The schematics are 13 pages long. Both the Workview and the postscript files are available from Intel as described above.

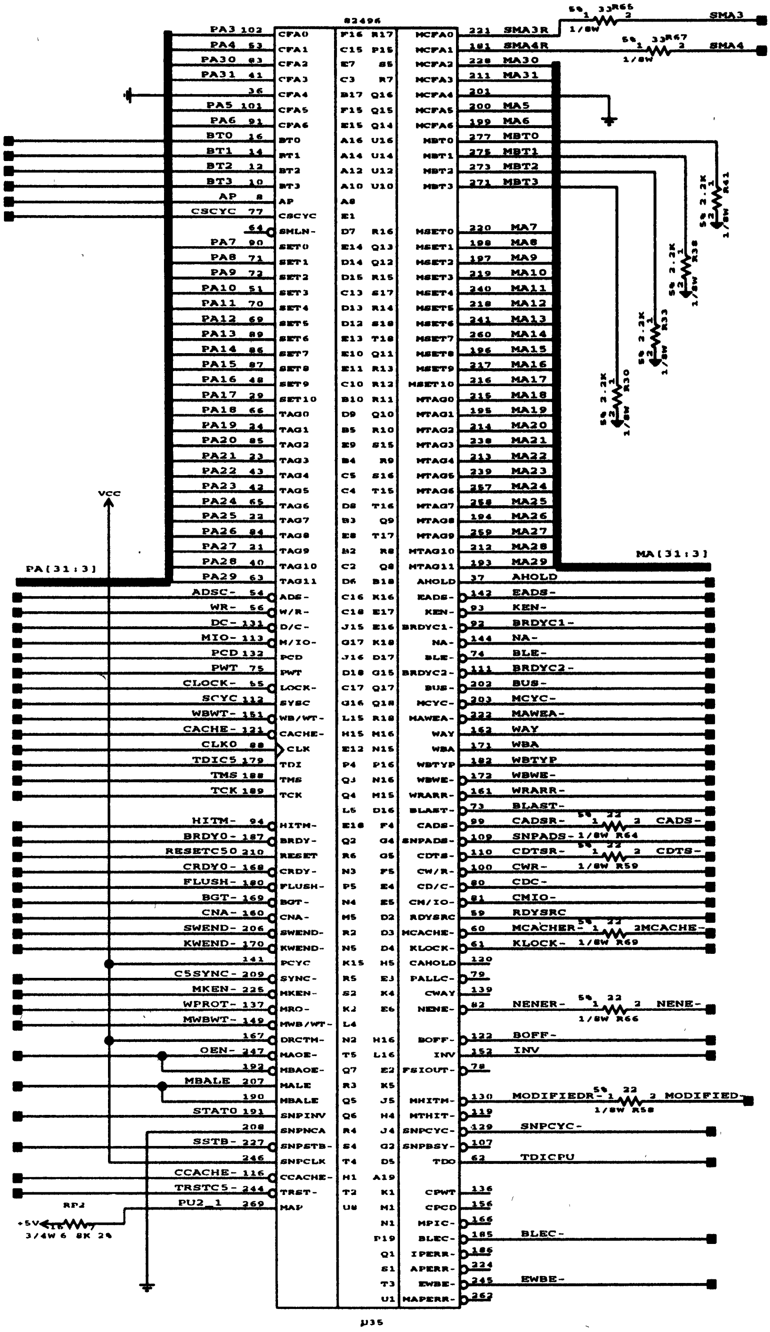
PENTIUM™ PROCESSOR/82496/82491 512KB MODULE
7-1-92
REV 2.0

(Tied MBZT- on all 82491 to Vcc)
(Removed series resistor on clk7 and added dummy load on clk7 at crdy pal, pin 27)

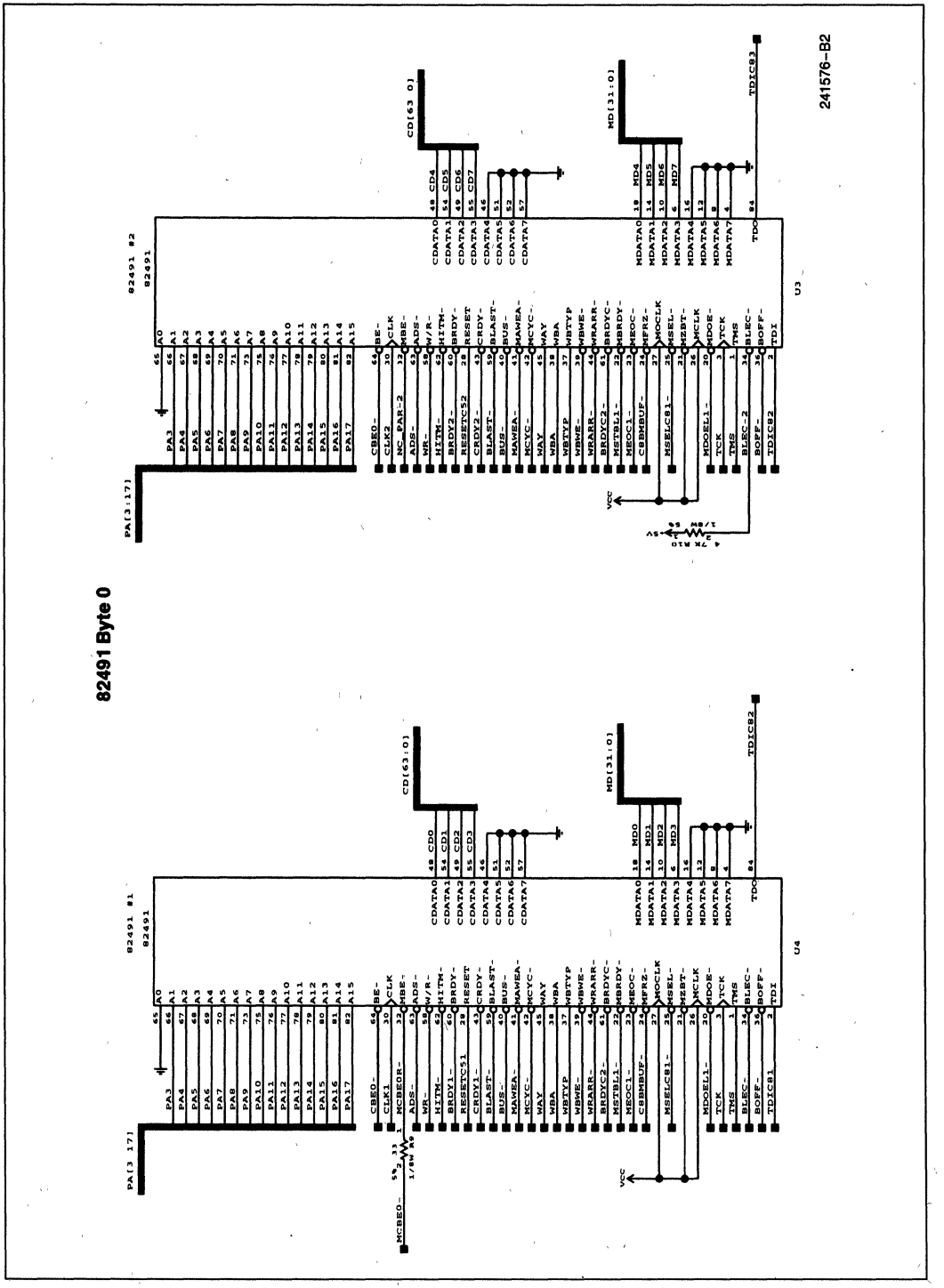
NOTES:
(Unless Otherwise Specified)
1. Capacitor values are in microfarads.
2. Resistor values are in ohms.
3. An "-" following a signal name denotes negation.
4. VCC = +5V.
5. This document also exists on electronic media.


PENTIUM PROCESSOR PGA273
CD[0:63]
CP[0:7]
RESET1
VCC
PA[3:31]
PBT[0:3]
ADD VIAS TO NA , BWBE WBWT-
241576-B0

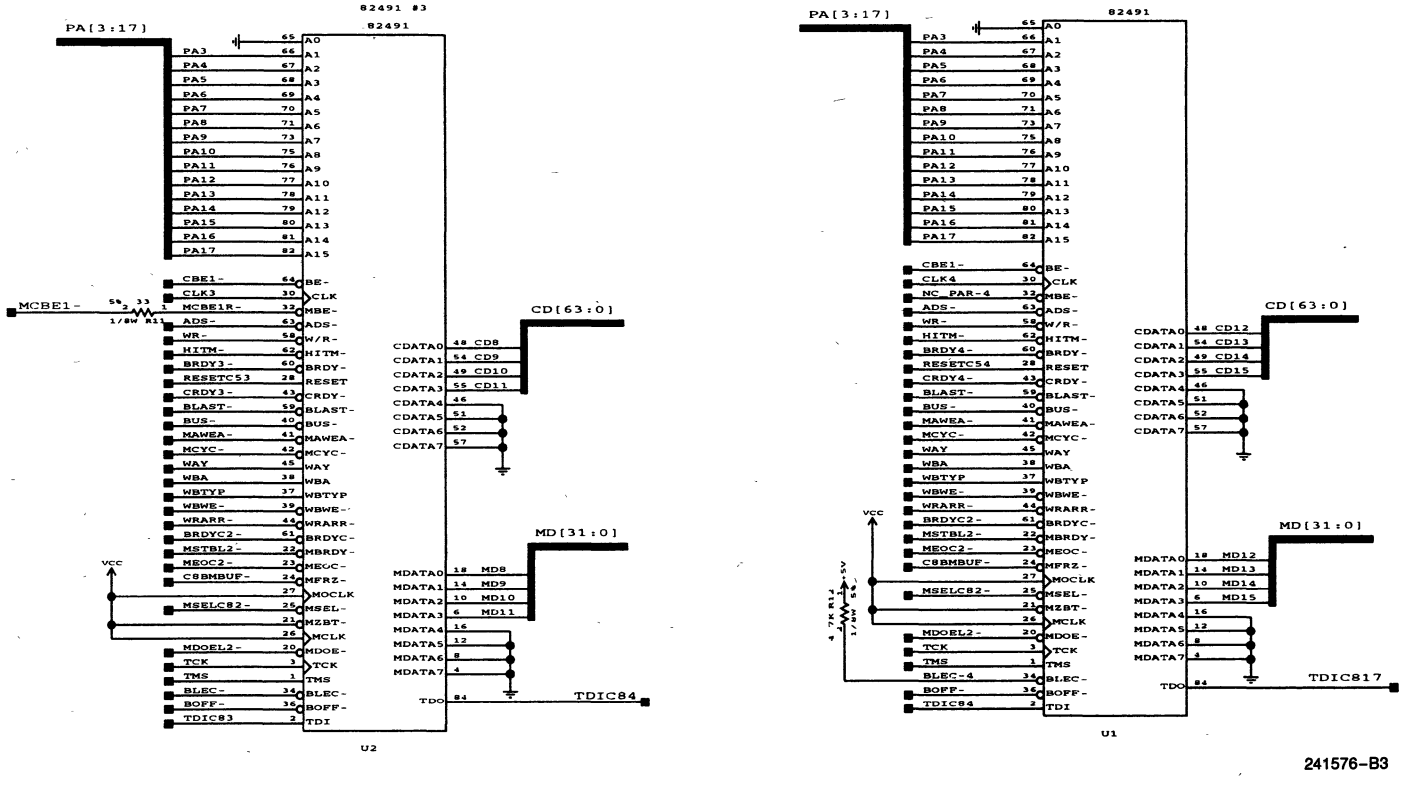
2


82496
U35
PA(31:3)
MA(31:3)
VCC
+5V
3/4W 6 8K 2%
RP2
PA3
PA4
PA30
PA31
PA5
PA6
BT0
BT1
BT2
BT3
AP
CSCYC
PA7
PA8
PA9
PA10
PA11
PA12
PA13
PA14
PA15
PA16
PA17
PA18
PA19
PA20
PA21
PA22
PA23
PA24
PA25
PA26
PA27
PA28
PA29
ADSC-
WR-
DC-
MIO-
PCD
PWT
CLOCK-
SCYC
WBWT-
CACHE-
CLKO
TDIC5
TMS
TCK
HITM-
BRDY0-
RESETC50
CRDY0-
FLUSH-
BGT-
CNA-
SWEND-
KWEND-
C5SYNC-
MKEN-
WPROT-
MWBWT-
OEN-
MBALE
STAT0
SSTB-
CCACHE-
TRSTC5-
PU2_1
CFA0
CFA1
CFA2
CFA3
CFA4
CFA5
CFA6
BT0
BT1
BT2
BT3
AP
CSCYC
SMLN-
SET0
SET1
SET2
SET3
SET4
SET5
SET6
SET7
SET8
SET9
SET10
TAG0
TAG1
TAG2
TAG3
TAG4
TAG5
TAG6
TAG7
TAG8
TAG9
TAG10
TAG11
ADS-
W/R-
D/C-
M/IO-
PCD
PWT
LOCK-
SYSC
WB/WT-
CACHE-
CLK
TDI
TMS
TCK
HITM-
BRDY-
RESET
CRDY-
FLUSH-
BGT-
CNA-
SWEND-
KWEND-
PCYC
SYNC-
MKEN-
MRO-
MWB/WT-
DRCTH-
MAOE-
MBAOE-
MALE
MBALE
SNPINV
SNPNCA
SNPSTB-
SNPCLK
CCACHE-
TRST-
MAP
MCFA0
MCFA1
MCFA2
MCFA3
MCFA4
MCFA5
MCFA6
MBT0
MBT1
MBT2
MBT3
MSET0
MSET1
MSET2
MSET3
MSET4
MSET5
MSET6
MSET7
MSET8
MSET9
MSET10
MTAG0
MTAG1
MTAG2
MTAG3
MTAG4
MTAG5
MTAG6
MTAG7
MTAG8
MTAG9
MTAG10
MTAG11
AHOLD
EADS-
KEN-
BRDYC1-
NA-
BLE-
BRDYC2-
BUS-
MCYC-
MAWEA-
WAY
WBA
WBTYP
WBWE-
WRARR-
BLAST-
CADS-
SNPADS-
CDTS-
CW/R-
CD/C-
CM/IO-
RDYSRC
MCACHE-
KLOCK-
CAHOLD
PALLC-
CWAY
NENE-
BOFF-
INV
FSIOUT-
MHITM-
MTHIT-
SNPCYC-
SNPBSY-
TDO
CPWT
CPCD
MPIC-
BLEC-
IPERR-
APERR-
EWBE-
MAPERR-
SMA3R
SMA4R
SMA3
SMA4
MA30
MA31
MA5
MA6
MBT0
MBT1
MBT2
MBT3
MA7
MA8
MA9
MA10
MA11
MA12
MA13
MA14
MA15
MA16
MA17
MA18
MA19
MA20
MA21
MA22
MA23
MA24
MA25
MA26
MA27
MA28
MA29
AHOLD
EADS-
KEN-
BRDYC1-
NA-
BLE-
BRDYC2-
BUS-
MCYC-
MAWEA-
WAY
WBA
WBTYP
WBWE-
WRARR-
BLAST-
CADSR-
CADS-
SNPADS-
CDTSR-
CDTS-
CWR-
CDC-
CMIO-
RDYSRC
MCACHER-
MCACHE-
KLOCK-
NENER-
NENE-
BOFF-
INV
MODIFIEDR-
MODIFIED-
SNPCYC-
TDICPU
BLEC-
EWBE-
R66
R67
R41
R38
R33
R30
R64
R59
R69
R58
1/8W
2.2K
33
22

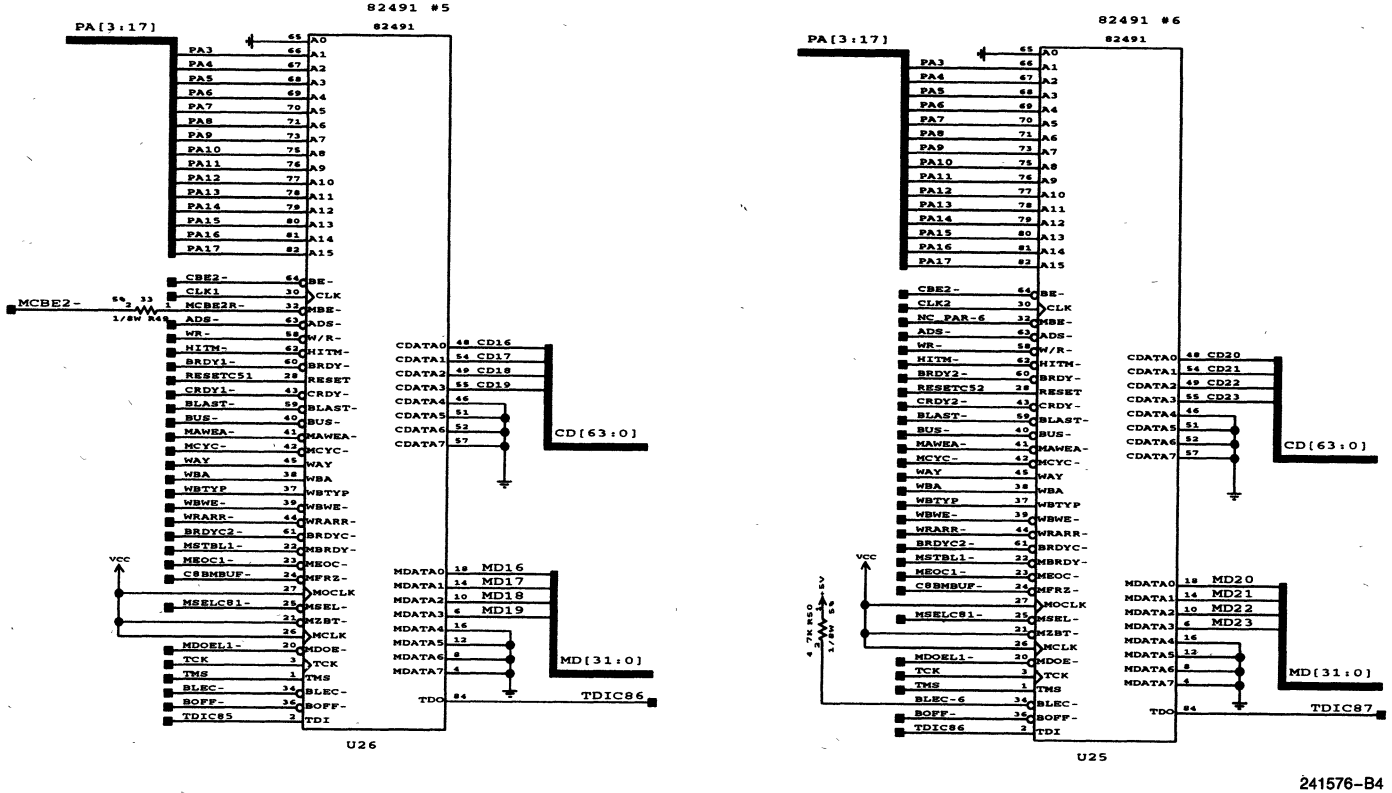
241576–B1


82491 Byte 0
241576-B2

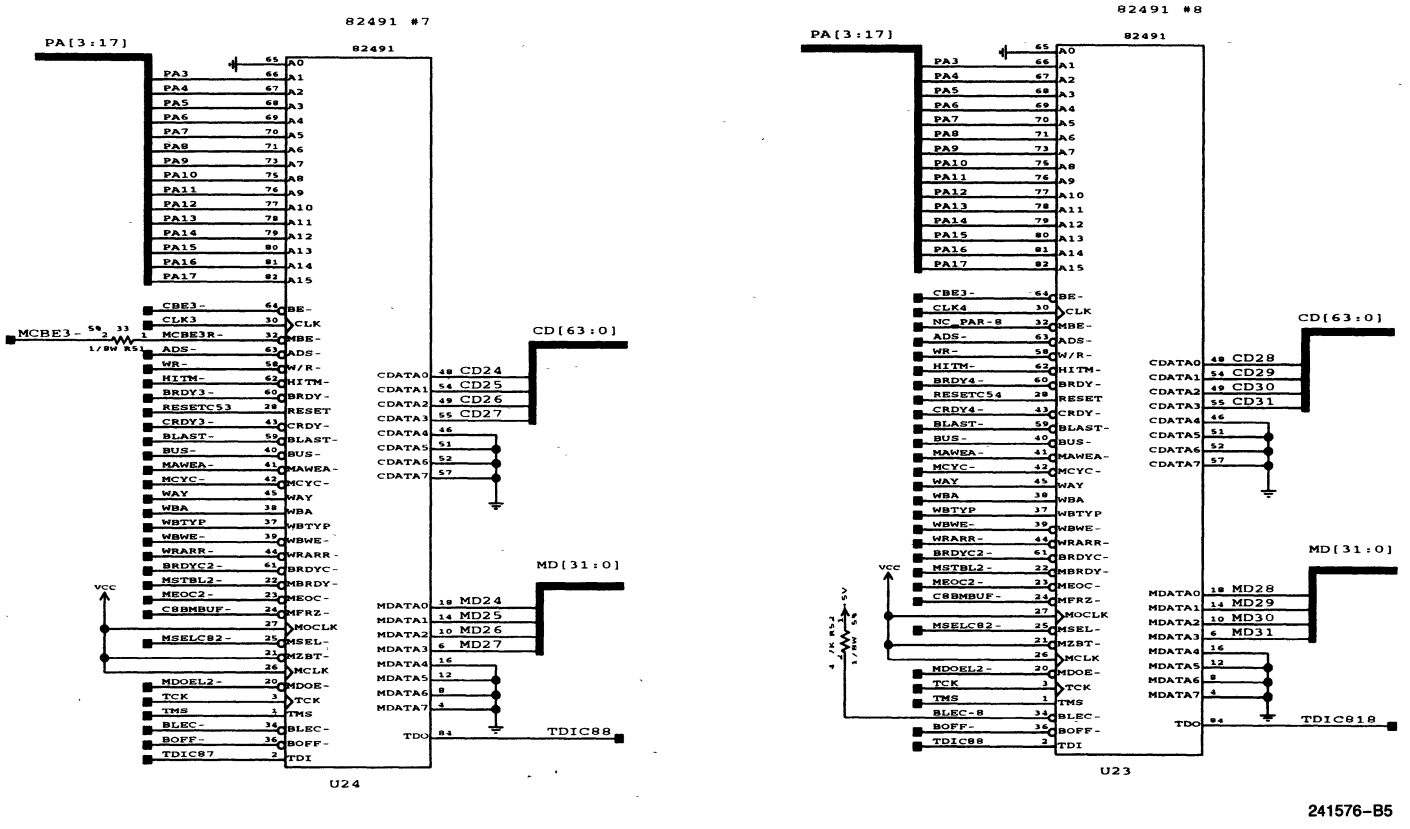
82491 Byte 1


82491 #3
82491
PA[3:17]
MCBE1-
CD[63:0]
MD[31:0]
TDIC84
U2
82491 #4
82491
PA[3:17]
CD[63:0]
MD[31:0]
TDIC817
U1
241576-B3

82491 Byte 2

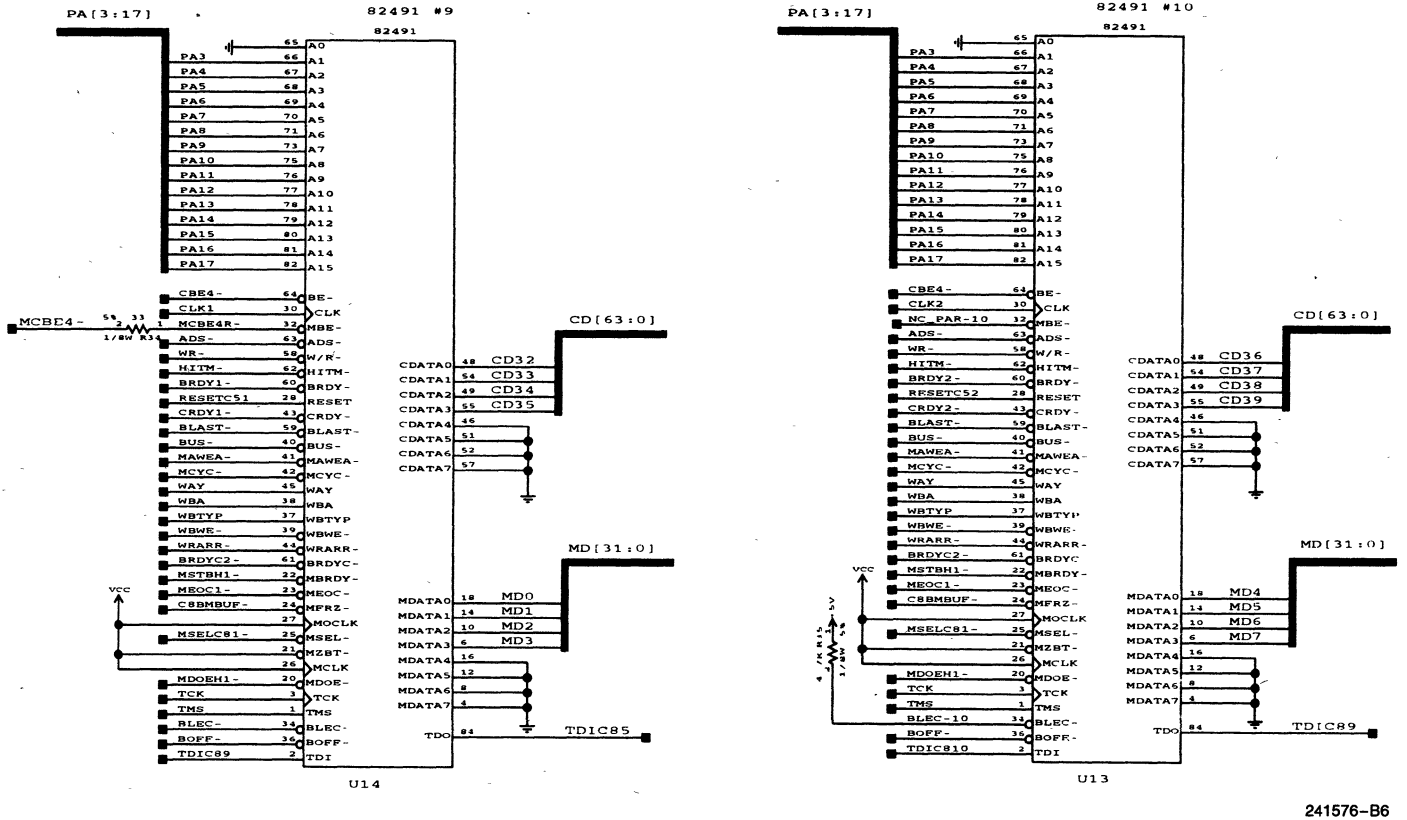

82491 #5
82491
PA[3:17]
MCBE2-
U26
CD[63:0]
MD[31:0]
TDIC86
82491 #6
82491
PA[3:17]
U25
CD[63:0]
MD[31:0]
TDIC87
241576-B4

82491 Byte 3

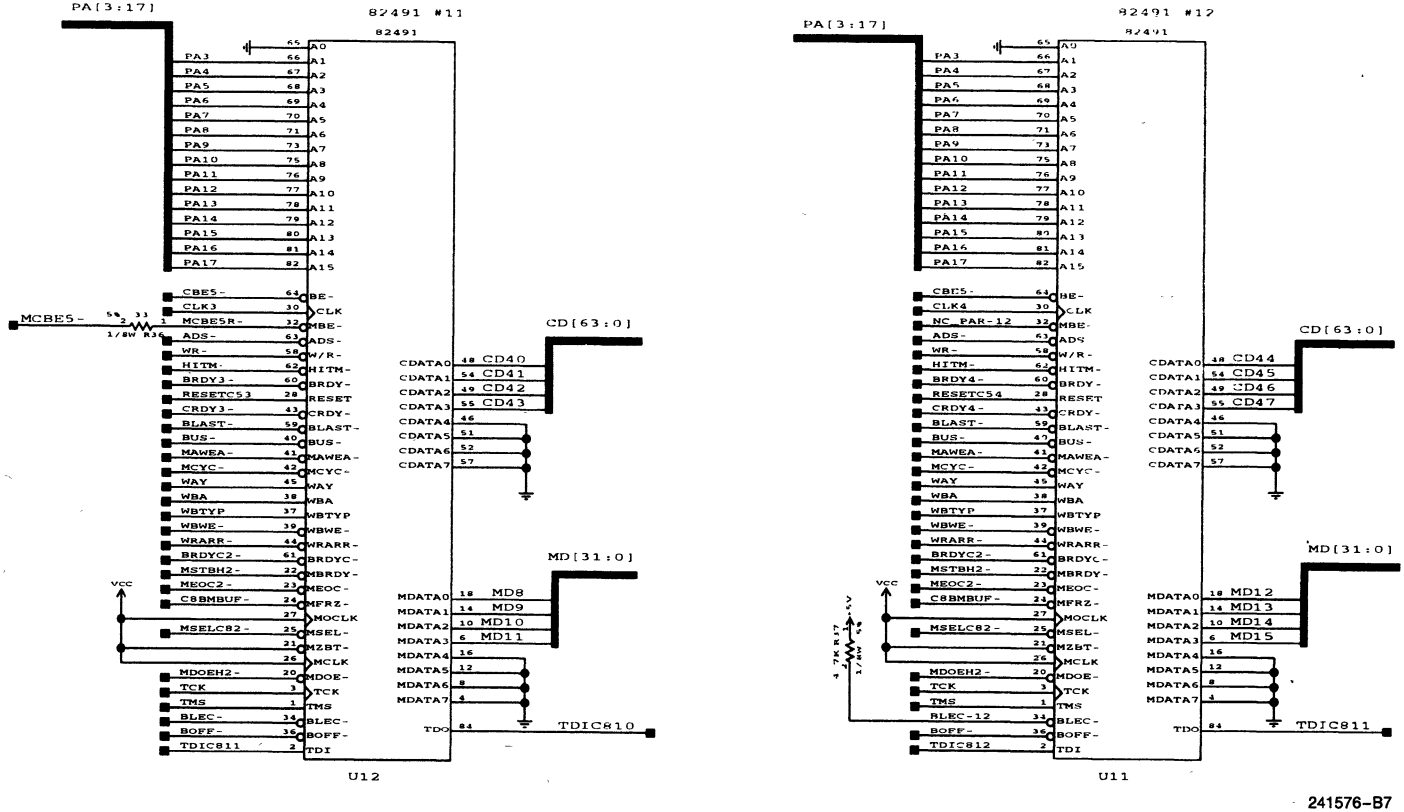


241576-B5

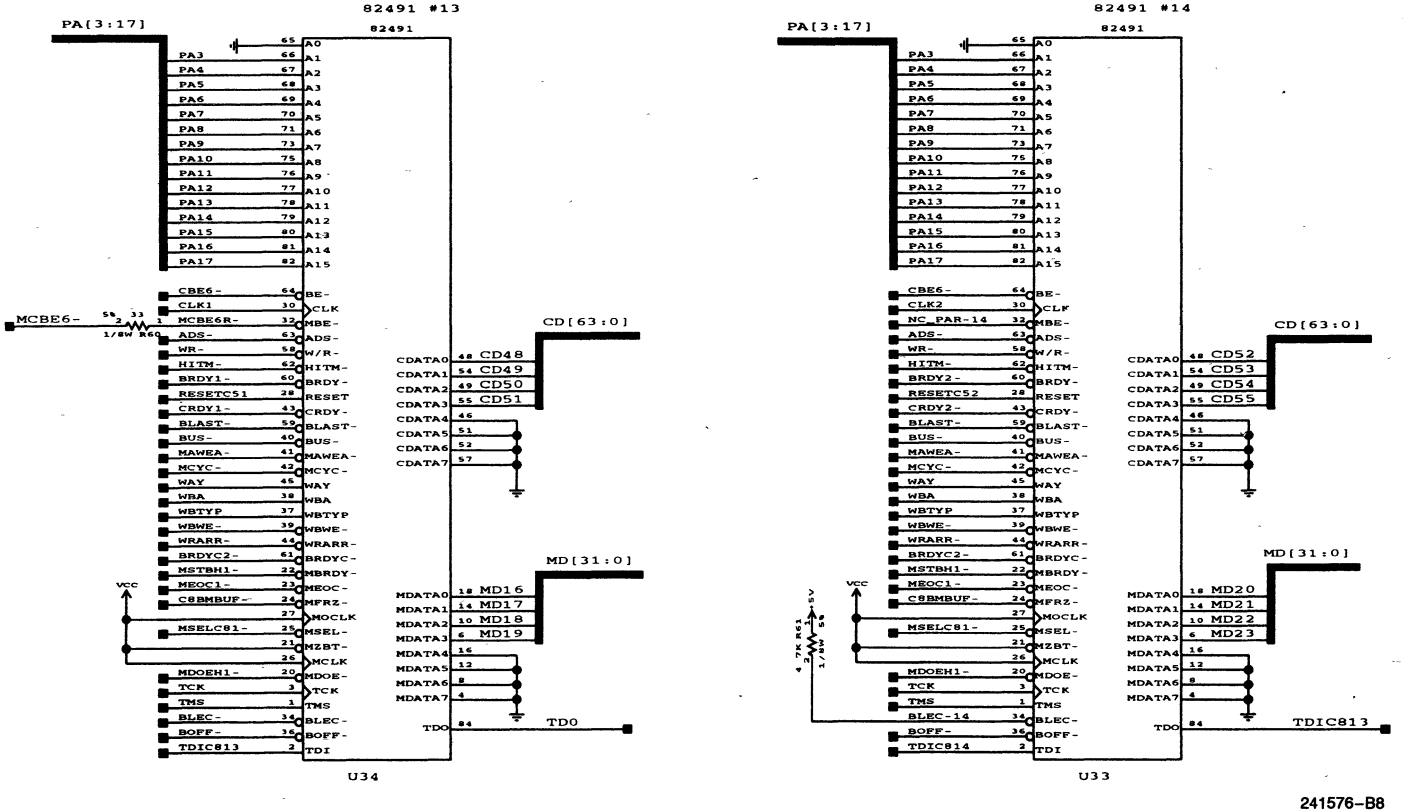
82491 Byte 4


PA[3:17]
82491 #9
82491
U14
MCBE4-
CBE4-
CLK1
MCBE4R-
ADS-
WR-
HITM-
BRDY1-
RESETC51
CRDY1-
BLAST-
BUS-
MAWEA-
MCYC-
WAY
WBA
WBTYP
WBWE-
WRARR-
BRDYC2-
MSTBH1-
MEOC1-
C8BMBUF-
MSELC81-
MDOEH1-
TCK
TMS
BLEC-
BOFF-
TDIC89
CDATA0
CDATA1
CDATA2
CDATA3
CDATA4
CDATA5
CDATA6
CDATA7
CD32
CD33
CD34
CD35
CD[63:0]
MDATA0
MDATA1
MDATA2
MDATA3
MDATA4
MDATA5
MDATA6
MDATA7
MD0
MD1
MD2
MD3
MD[31:0]
TDO
TDIC85
VCC
PA[3:17]
82491 #10
82491
U13
CBE4-
CLK2
NC_PAR-10
ADS-
WR-
HITM-
BRDY2-
RESETC52
CRDY2-
BLAST-
BUS-
MAWEA-
MCYC-
WAY
WBA
WBTYP
WBWE-
WRARR-
BRDYC2-
MSTBH1-
MEOC1-
C8BMBUF-
MSELC81-
MDOEH1-
TCK
TMS
BLEC-10
BOFF-
TDIC810
CDATA0
CDATA1
CDATA2
CDATA3
CDATA4
CDATA5
CDATA6
CDATA7
CD36
CD37
CD38
CD39
CD[63:0]
MDATA0
MDATA1
MDATA2
MDATA3
MDATA4
MDATA5
MDATA6
MDATA7
MD4
MD5
MD6
MD7
MD[31:0]
TDO
TDIC89
VCC
5V
241576-B6

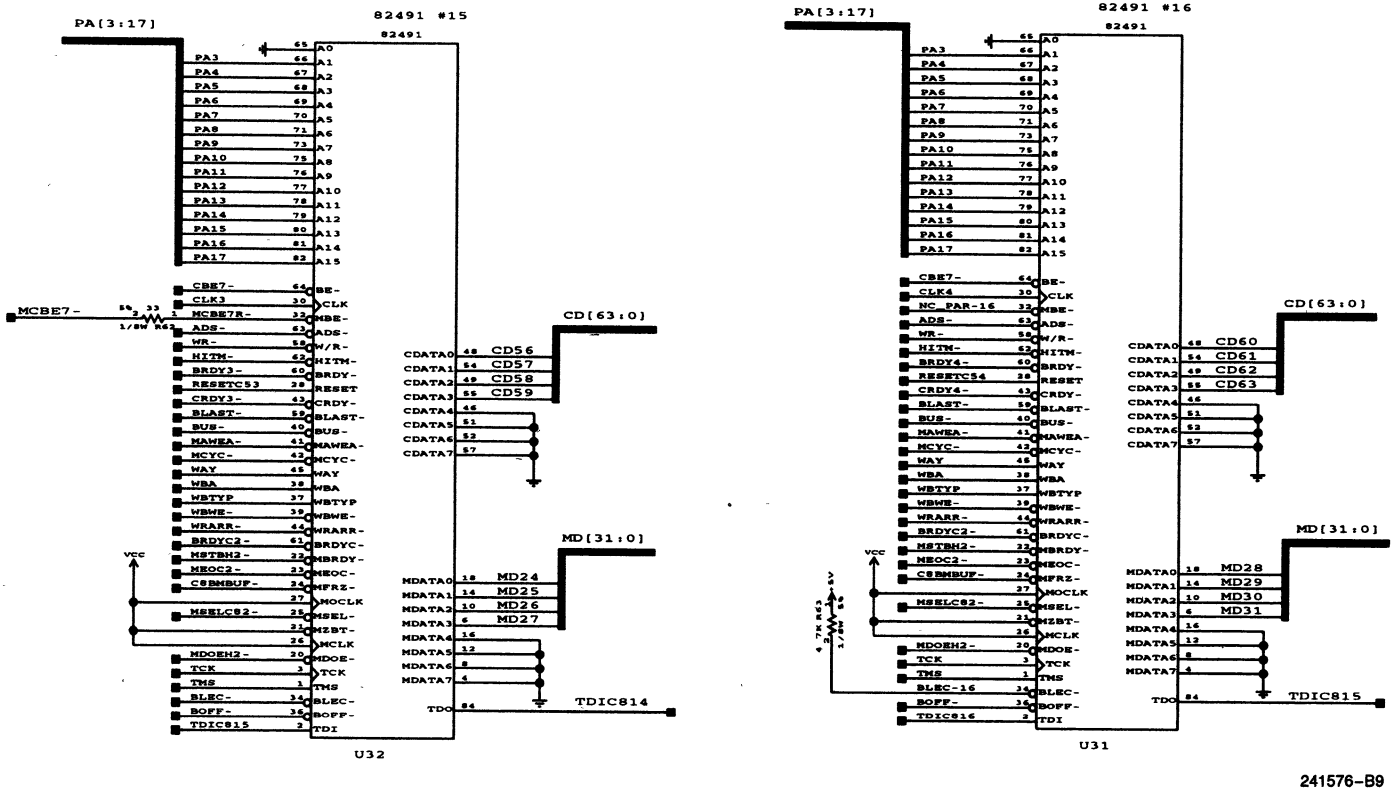
82491 Byte 5


PA[3:17]
82491 #11
82491
MCBE5-
CD[63:0]
CDATA0 CD40
CDATA1 CD41
CDATA2 CD42
CDATA3 CD43
MD[31:0]
MDATA0 MD8
MDATA1 MD9
MDATA2 MD10
MDATA3 MD11
TDIC810
U12
PA[3:17]
82491 #12
82491
CD[63:0]
CDATA0 CD44
CDATA1 CD45
CDATA2 CD46
CDATA3 CD47
MD[31:0]
MDATA0 MD12
MDATA1 MD13
MDATA2 MD14
MDATA3 MD15
TDIC811
U11
241576-B7

82491 Byte 6

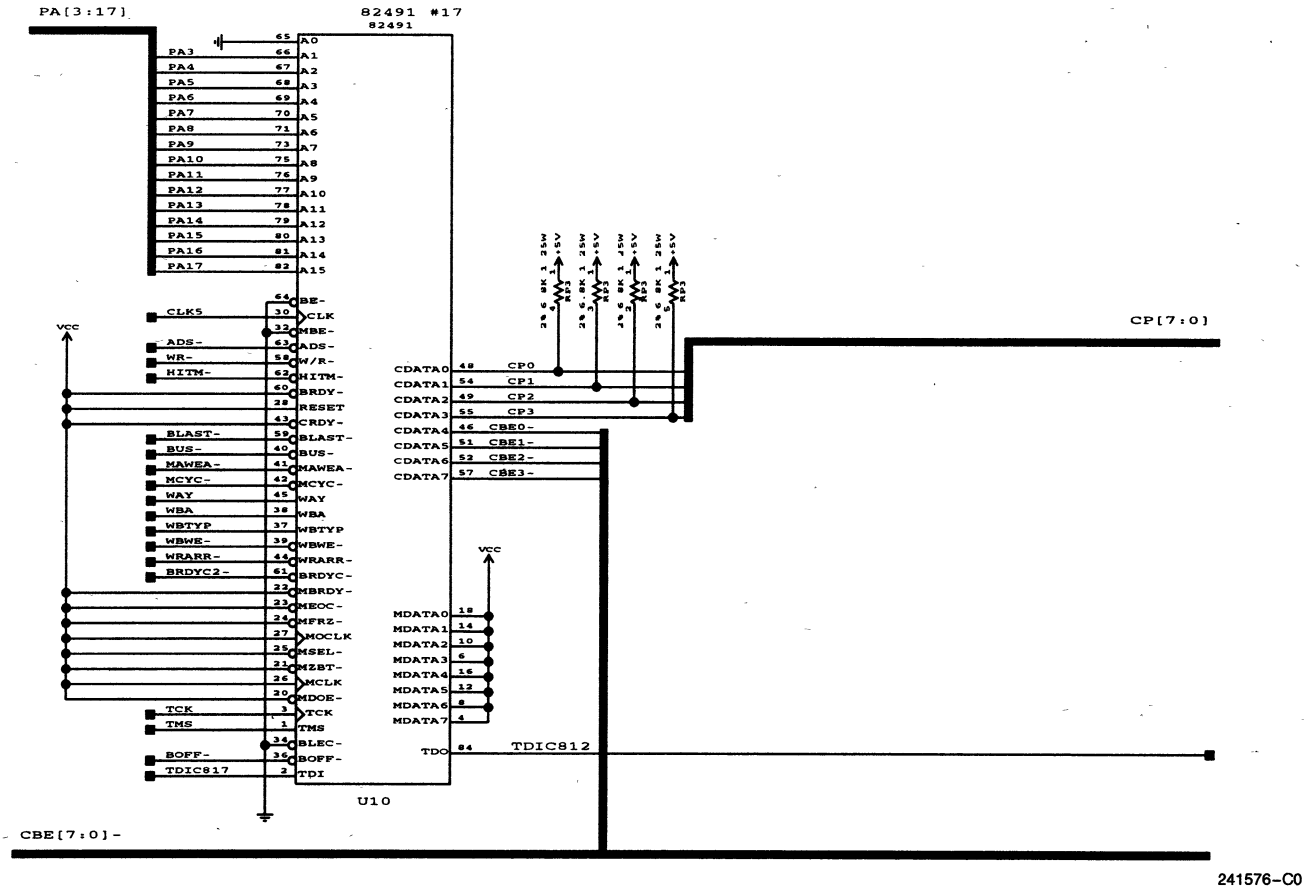


82491 Byte 7

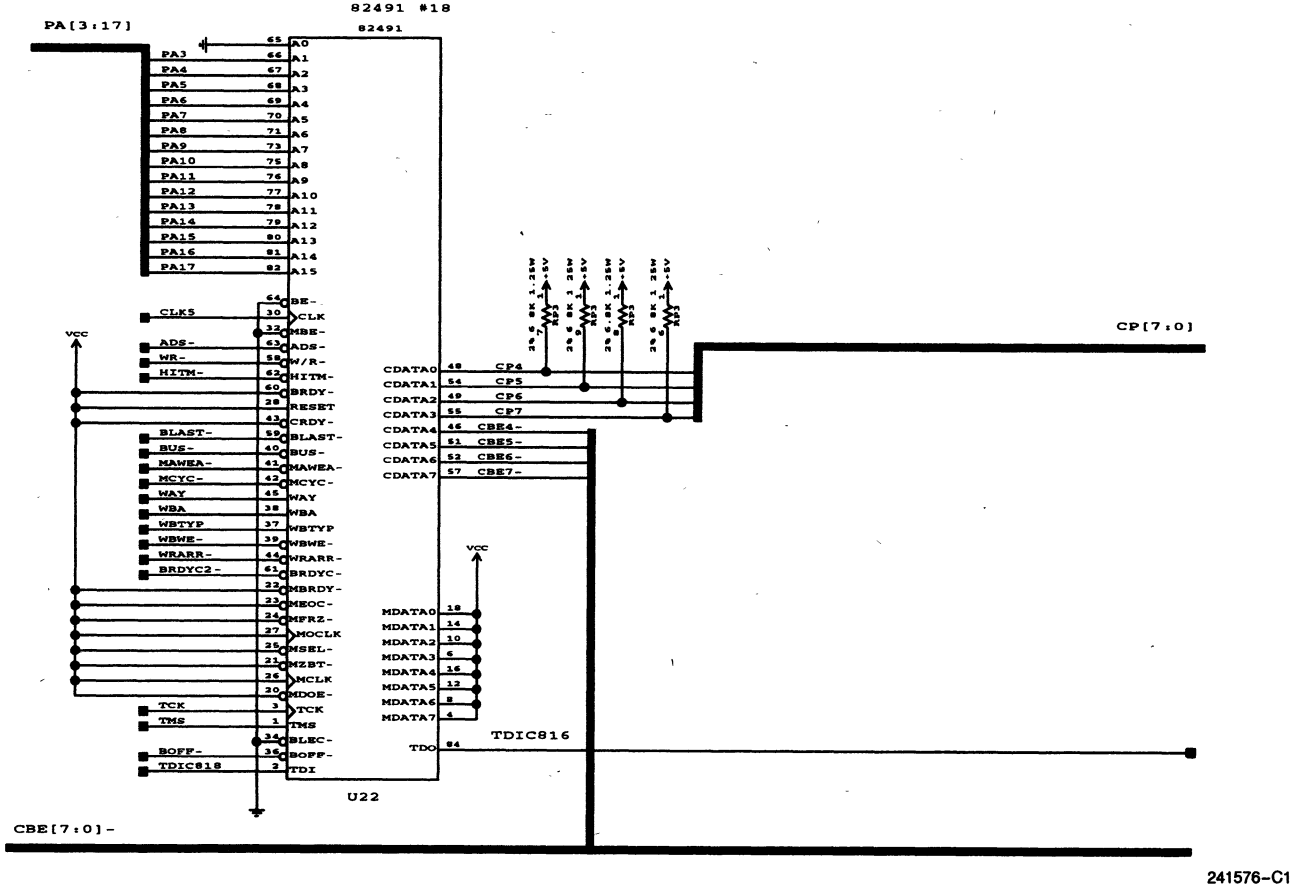


241576-B9

82491 Parity 0-3



82491 Parity 4-7



241576-C1

8.6.2 I/O MODEL FILES

All electrical I/O simulations were performed using TLC V4.1.13 from Quad Design Technology, Inc. The simulations were performed at the fast and slow corners to verify all signal quality and flight time specifications are met. The files used for these simulations are available from Intel as described above. These files include the topology, model, and control files needed to run the simulations for all nets in the optimized interface.

8.6.3 BOARD FILES

The board files for the design example were created using Allegro V4.2 from Cadence Design Systems, Inc. The files are available from Intel as described above. These files may be used to import the design example into a specific system design. Note: some changes to the layout and nets may be necessary to complete importing these files into a specific system design.

8.6.4 BILL OF MATERIALS

The bill of materials file was created using Allegro V4.2 from Cadence Design Systems, Inc. The file is available from Intel as described above.

8.6.5 PHOTOPLOT LOG

The photoplot log file was created using Allegro V4.2 from Cadence Design Systems, Inc. The file is available from Intel as described above.

8.6.6 NETLIST REPORT

The netlist report was created using Allegro V4.2 from Cadence Design Systems, Inc. The file is available from Intel as described above.

8.6.7 PLACED COMPONENT REPORT

The placed component report was created using Allegro V4.2 from Cadence Design Systems, Inc. The file is available from Intel as described above.

8.6.8 ARTWORK FOR EACH BOARD LAYER

The artwork for the six board layers were created using Allegro V4.2 from Cadence Design Systems, Inc. The files are available from Intel in a Gerber format as described above.

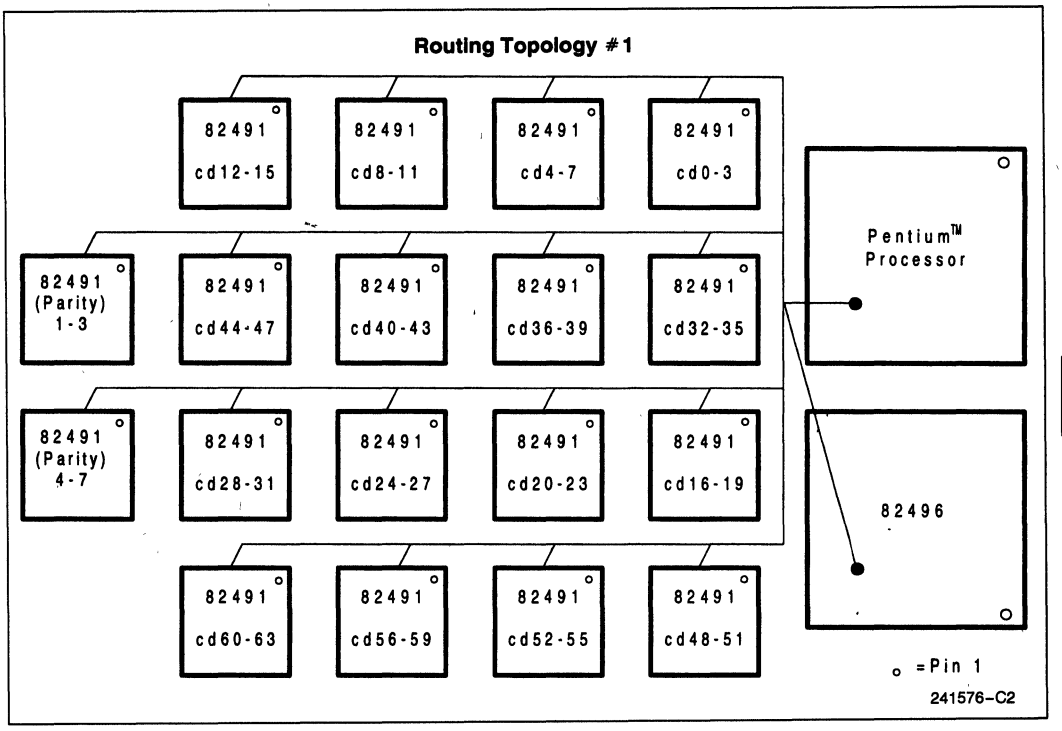
8.6.9 TRACE SEGMENT LINE LENGTHS

Sections 8.6.9.1 to 8.6.9.7 list the segment line lengths for each net of the optimized interface. All lengths are provide in mils (1/1000 inch). The stubs listed in the following tables are associated with the pin escapes required for the 82491s.

The following abbreviations have been used to represent the different devices:

PP = Pentium processor
CC = 82496 cache controller
CS = 82491 cache SRAM

8.6.9.1 Low Addresses and Pentium™ Processor Control



2

NET	PP-Tee	CC-Tee	Tee-CS#5	Tee-CS#9
PA3	1211	4398	614.9	634.9
PA4	771.3	4403	620.8	634.9
PA5	920.1	4402	577.8	637.8
PA6	1026	4405	577.8	634.9
PA7	1132	4443	569.1	642
PA8	1032	4410	563.2	640.8
PA9	1122	4449	576.7	606.6
PA10	1000	4478	582.5	597.8
PA11	1060	4332	585.8	629.9
PA12	1019	4393	585.8	635.8
PA13	1077	4434	597.8	637.8
PA14	968.3	4653	689.1	743.2
PA15	1233	4402	524.2	580.8
PA16	1092	4384	597.8	614.9
PA17	1186	4396	600.8	602.3
HITM#	3575	4409	430.7	530.7
W/R#	3913	4397	674.9	676.6

NET	CS#9–CS#1	CS#1–CS#2	CS#2–CS#3	CS#3–CS#4	CS#9–CS#10	CS#10–CS#11	CS#11–CS#12	CS#12–CS#17
PA3	1111	944.9	936.6	961.4	1124	936.6	944.9	936.6
PA4	1210	964.9	936.6	975.6	1207	936.6	961.9	936.6
PA5	1131	944.9	936.6	961.4	1104	936.6	944.9	936.6
PA6	1165	936.6	936.6	936.6	1177	936.6	936.6	936.6
PA7	1170	936.6	936.6	936.6	1157	936.6	936.6	936.6
PA8	1170	936.6	936.6	936.6	1157	936.6	936.6	936.6
PA9	1224	936.6	936.6	936.6	1217	936.6	936.6	936.6
PA10	1211	936.6	936.6	936.6	1207	936.6	936.6	936.6
PA11	1264	936.6	936.6	936.6	1267	936.6	936.6	936.6
PA12	1264	936.6	936.6	936.6	1267	936.6	936.6	936.6
PA13	1293	936.6	936.6	936.6	1297	936.6	936.6	936.6
PA14	1295	936.6	936.6	936.6	1297	936.6	936.6	936.6
PA15	1312	936.6	936.6	936.6	1317	936.6	936.6	936.6
PA16	1309	936.6	936.6	936.6	1317	936.6	936.6	936.6
PA17	1297	936.6	936.6	936.6	1287	936.6	936.6	936.6
HITM#	1141	953.1	953.1	953.1	1147	936.6	996.9	944.9
W/R#	1193	953.1	953.1	1023	1192	936.6	1003	953.1

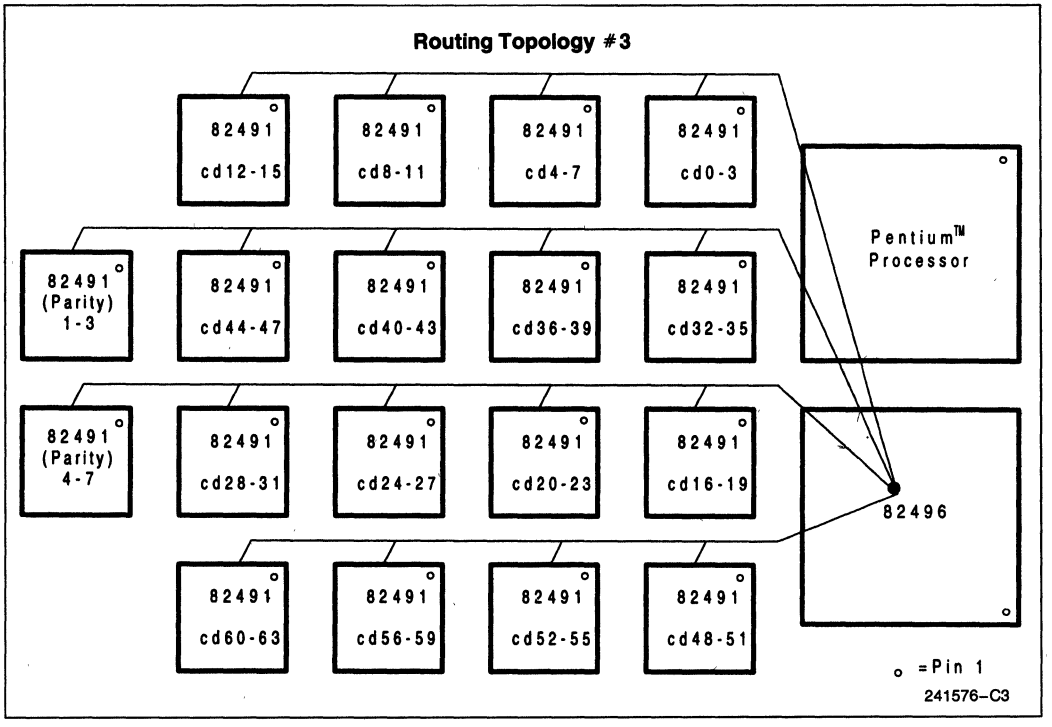
NET	CS#5-CS#6	CS#6-7 CS#	CS#7-CS#8	CS#8-CS#18	CS#5-CS#13	CS#13-CS#14	CS#14-CS#15	CS#15-CS#16	Stubs
PA3	1129	961.4	953.1	936.6	1122	944.9	936.6	944.9	75.0- 75.0
PA4	1207	969.7	967.3	944.9	1185	959	936.6	964.9	135.4- 135.4
PA5	1136	961.4	953.1	936.6	1145	944.9	936.6	944.9	75.0- 75.0
PA6	1190	936.6	936.6	936.6	1179	936.6	936.6	936.6	135.4- 135.4
PA7	1176	936.6	936.6	936.6	1179	936.6	936.6	936.6	75.0- 75.0
PA8	1177	936.6	936.6	936.6	1176	936.6	936.6	936.6	135.4- 135.4
PA9	1217	936.6	936.6	936.6	1213	936.6	936.6	936.6	135.4- 135.4
PA10	1207	936.6	936.6	936.6	1208	936.6	936.6	936.6	135.4 -135.4
PA11	1267	936.6	936.6	936.6	1261	936.6	936.6	936.6	75.0- 75.0
PA12	1267	936.6	936.6	936.6	1261	936.6	936.6	936.6	135.4 -135.4
PA13	1297	936.6	936.6	936.6	1281	936.6	936.6	936.6	75.0- 75.0
PA14	1297	936.6	936.6	936.6	1284	936.6	936.6	936.6	135.4 -135.4
PA15	1317	936.6	936.6	936.6	1318	936.6	936.6	936.6	75.0- 75.0
PA16	1317	936.6	936.6	936.6	1315	936.6	936.6	936.6	135.4 -135.4
PA17	1287	936.6	936.6	936.6	1288	936.6	936.6	936.6	75.0-75.0
HITM#	1146	944.9	944.9	964.9	1138	944.9	936.6	936.6	95.4 -95.4
W/R#	1197	975.6	961.4	953.1	1193	953.1	936.6	936.6	95.4 -95.4

NET	CC-Tee	PP-Tee	Tee-CS#5	Tee-CS#9	CS#9-CS#1	CS#1-CS#2	CS#2-CS#3	CS#3-CS#4
BOFF#	2405.6	4401.9	919.7	925.8	936.6	936.6	936.6	936.6

NET	CS#9-CS#10	CS#10-CS#11	CS#11-CS#12	CS#12-CS#17	CS#5-CS#6	CS#6-CS#	CS#7-CS#8	CS#8-CS#18
BOFF#	936.6	936.6	936.6	936.6	936.6	936.6	936.6	936.6

NET	CS#5-CS#13	CS#13-CS#14	CS#14-CS#15	CS#15-CS#16	Stubs
BOFF#	944.9	936.6	936.6	936.6	75.0-75.0

8.6.9.2 82496 Control

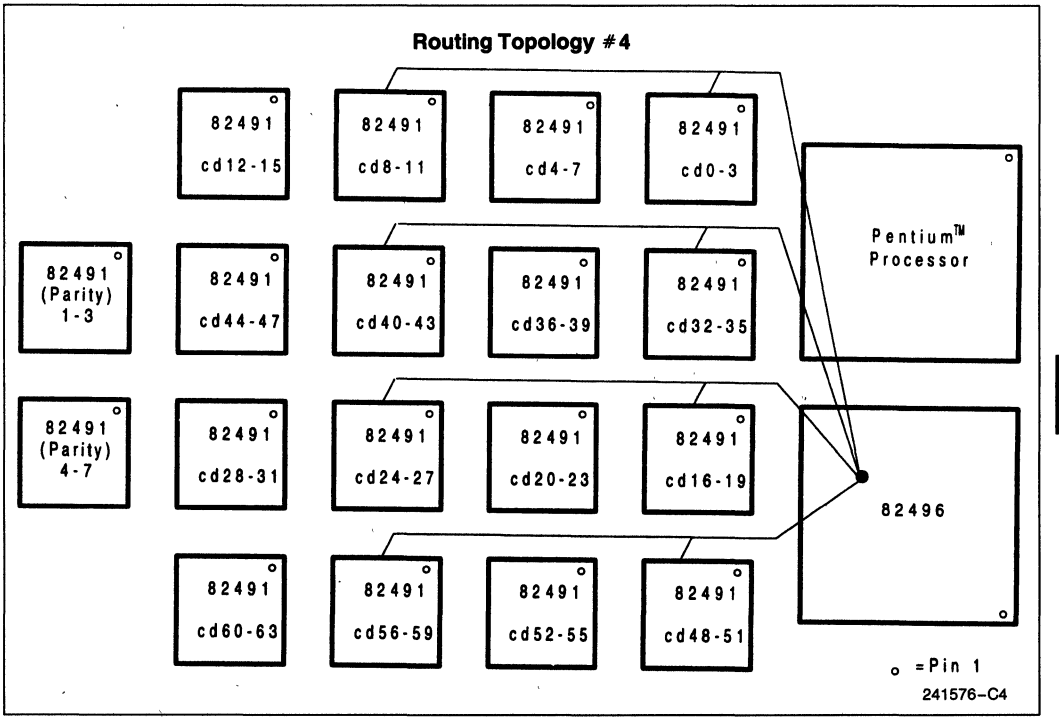


NET	CC-CS#1	CC-CS#9	CC-CS#5	CC-CS#13	CS#1-CS#2	CS#2-CS#3	CS#3-CS#4	CS#9-CS#10
BLAST#	4225	3085	3089	4186	961.4	967.3	961.4	1020
BUS#	4318	3216	3210	4311	936.6	936.6	936.6	936.6
MAWEA#	936.6	936.6	936.6	936.6	936.6	936.6	936.6	936.6
WBA	936.6	936.6	936.6	936.6	936.6	936.6	936.6	936.6
WBTYP	4087	2986	2974	4086	936.6	936.6	936.6	936.6
WBWE	4210	3109	3113	4215	936.6	936.6	936.6	936.6

NET	CS#10-CS#11	CS#11-CS#12	CS#12-CS#17	CS#5-CS#6	CS#6-CS#7	CS#7-CS#8	CS#8-CS#18	CS#13-CS#14
BLAST#	1011	969.7	969.7	964.9	936.6	961.4	944.9	936.6
BUS#	936.6	936.6	936.6	936.6	936.6	936.6	936.6	936.6
MAWEA#	4187	3059	3089	4178	936.6	936.6	936.6	936.6
WBA	4846	3748	3680	4845	936.6	936.6	936.6	936.6
WBTYP	936.6	936.6	936.6	936.6	936.6	936.6	936.6	936.6
WBWE	936.6	936.6	936.6	936.6	936.6	936.6	936.6	936.6

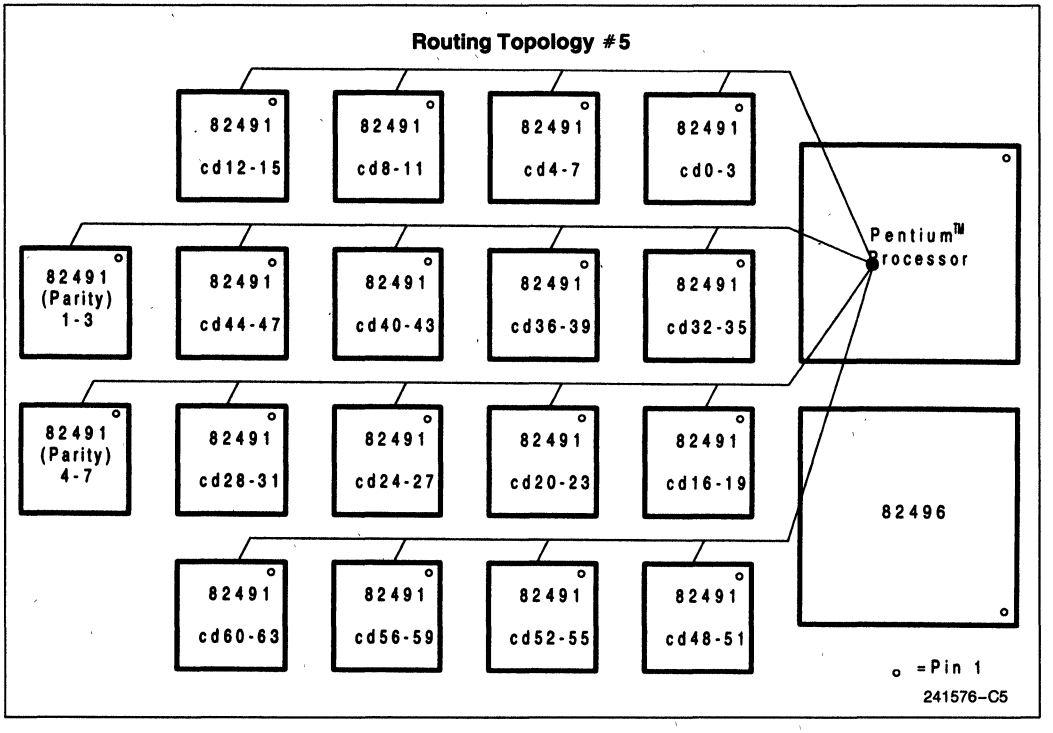
NET	CS#14-CS#15	CS#15-CS#16	Stubs
BLAST#	936.6	944.9	85.0-85.0
BUS#	936.6	936.6	75.0-75.0
MAWEA#	936.6	936.6	135.4-135.4
WBA	936.6	936.6	75.0-75.0
WBTYP	936.6	936.6	135.4-135.4
WBWE	936.6	936.6	135.4-135.4

8.6.9.3 82496 Control



NET	CC– CS#1	CC– CS#9	CC– CS#5	CC– CS#13	CS#1– CS#3	CS#9– CS#11	CS#5– CS#7	CS#13– CS#15	Stubs
BLEC–	4222.8	4219.1	4205.9	4206.5	1856.6	1856.6	1856.6	1856.6	75.0 –75.0

8.6.9.4 Pentium™ Processor Control

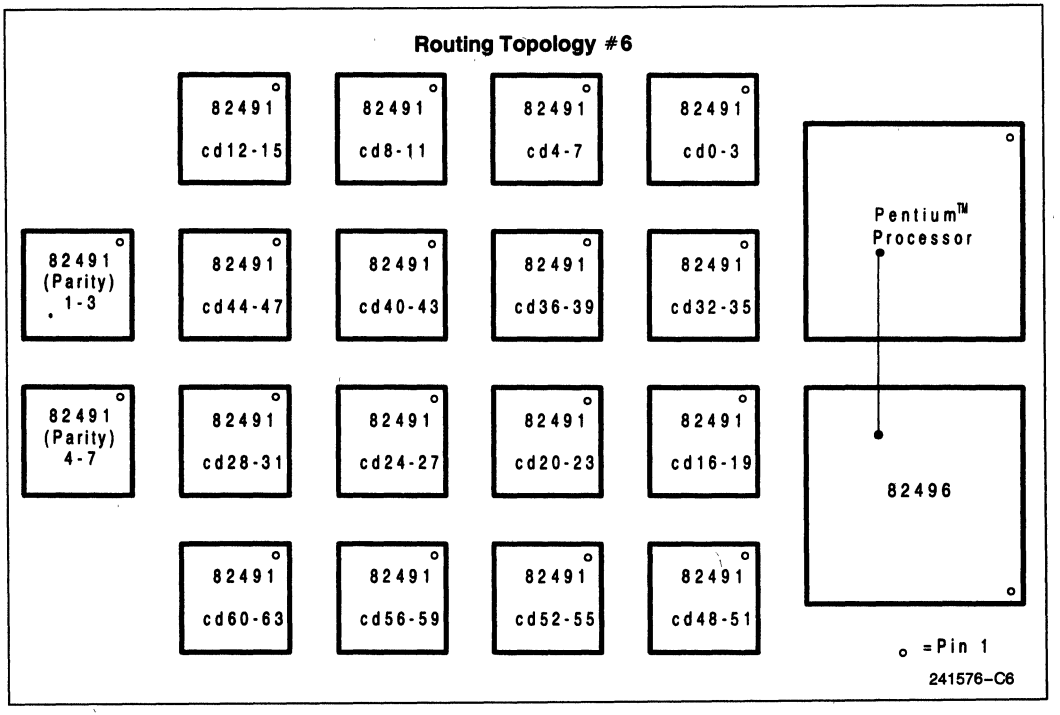


NET	PP–CS#1	PP–CS#9	PP–CS#5	PP–CS#13	CS#1–CS#2	CS#2–CS#3	CS#3–CS#4	CS#9–CS#10	CS#10–CS#11
ADS#	5213.8	4111.6	4108.4	5207.4	936.6	936.6	936.6	964.9	1003.8

NET	CS#11–CS#12	CS#12–CS#17	CS#5–CS#6	CS#6–CS#7	CS#7–CS#8	CS#8–CS#18	CS#13–CS#14	CS#14–CS#15	CS#15–CS#16
ADS#	964.9	978	936.6	936.6	936.6	969	961.9	959	961.9

NET	Stubs
ADS#	75.0–75.0

8.6.9.5 Pentium™ Processor and 82496 Control, High Addresses, Pentium™ Processor Data



2

NET	PP-res	res-CC
PA18	797.1	2907.3
PA19	824.3	3440.5
PA20	682.6	2881.1

NET	PP-CC
PA21	3505.2
PA22	4234.7
PA23	3478.8
PA24	4011.7
PA25	3756.6
PA26	3672.9
PA27	3807.5
PA28	4963.9
PA29	3318.8
PA30	4535.8
PA31	4097.2
BT0	3604.5
BT1	2677
BT2	3952.2
BT3	3185.9

NET	PP-CC
ADSC#	3880.6
AP	4556.5
CACHE#	3798.4
D/C#	3647.4
LOCK#	4721.2
M/IO#	5062.1
PCD	3366
PWT	4264.2
SCYC	3798
AHOLD	4604.8
BRDYC1#	3686
EADS#	3656.5
INV	4603.5
KEN#	4144.8
NA#	3770.4
WB/WT#	3493.3
EWBE#	2475

NET	PP-CS#17	Stubs
CP0	7558	135.4
CP1	7685.9	135.4
CP2	7675.3	75
CP3	7307.4	75

NET	PP-CS#5	Stubs
CD16	7121.5	135.4
CD17	6878.5	135.4
CD18	6974.5	75
CD19	7021.5	75

NET	PP-CS#11	Stubs
CD40	692.8	135.4
CD41	7567.4	135.4
CD42	6922.3	75
CD43	7613.3	75

NET	PP-CS#18	Stubs
CD4	7641.3	135.4
CD5	7698.3	135.4
CD6	7416.4	75
CD7	6946.9	75

NET	PP-CS#6	Stubs
CD20	7089.3	135.4
CD21	6948.1	135.4
CD22	7064.9	75
CD23	6927.4	75

NET	PP-CS#12	Stubs
CD44	7018	95.4
CD45	6997.1	135.4
CD46	6956.1	75
CD47	7491.6	75

NET	PP-CS#1	Stubs
CD0	6920.6	135.4
CD1	6964.1	135.4
CD2	7130.1	75
CD3	6910.9	75

NET	PP-CS#7	Stubs
CD24	6968.1	107.1
CD25	6877.9	135.4
CD26	7040.2	75
CD27	6891.7	75

NET	PP-CS#13	Stubs
CD48	6961	135.4
CD49	6884.2	135.4
CD50	7072.2	75
CD51	6968.3	75

NET	PP-CS#2	Stubs
CD4	7037.3	135.4
CD5	6869.4	135.4
CD6	6925	75
CD7	6886.5	75

NET	PP-CS#31	Stubs
CD28	7688.3	135.4
CD29	7872	135.4
CD30	7395.9	75
CD31	7433.3	75

NET	PP-CS#1	Stubs
CD52	7277.5	135.4
CD53	6979.9	135.4
CD54	6921.7	75
CD55	6920.2	75

NET	PP-CS#1	Stubs
CD8	6945.5	135.4
CD9	7448.5	135.4
CD10	7790.2	75
CD11	6924.1	75

NET	PP-CS#9	Stubs
CD32	7586.8	135.4
CD33	7271.6	135.4
CD34	7424.6	75
CD35	6944.1	75

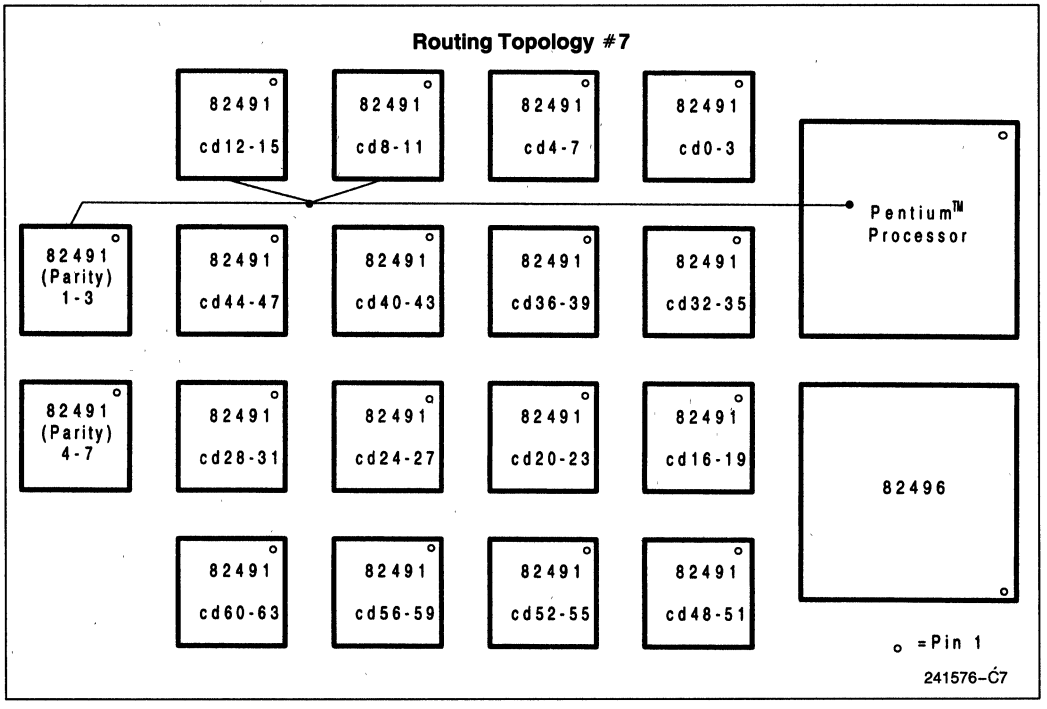
NET	PP-CS#15	Stubs
CD56	6777.1	135.4
CD57	6997.1	135.4
CD58	6881.1	75
CD59	7130.3	75

NET	PP-CS#4	Stubs
CD12	7275.6	135.4
CD13	7344.7	135.4
CD14	7692.1	75
CD15	7438.1	75

NET	PP-CS#10	Stubs
CD36	7343	135.4
CD37	6952.2	135.4
CD38	7520	75
CD39	7403	75

NET	PP-CS#16	Stubs
CD60	6919.5	135.4
CD61	6971.3	135.4
CD62	7274.5	75
CD63	7019.4	75

8.6.9.6 Byte Enables



NET	PP-Tee	Tee-CS# 1	Tee-CS# 2	Tee-CS#17	Stubs
CBE0#	1552.2	708.5	707.3	4813.3	75.0–135.4

NET	PP-Tee	Tee-CS# 3	Tee-CS# 4	Tee-CS#17	Stubs
CBE1#	4358.7	545.6	543.1	2627.4	75.0–75.0

NET	PP-Tee	Tee-CS# 5	Tee-CS# 6	Tee-CS#17	Stubs
CBE2#	3476.4	562.1	563.1	4330.5	75.0–135.4

NET	PP-Tee	Tee-CS# 7	Tee-CS# 8	Tee-CS#17	Stubs
CBE3#	5352.9	537.3	440.7	3011.3	75.0–75.0

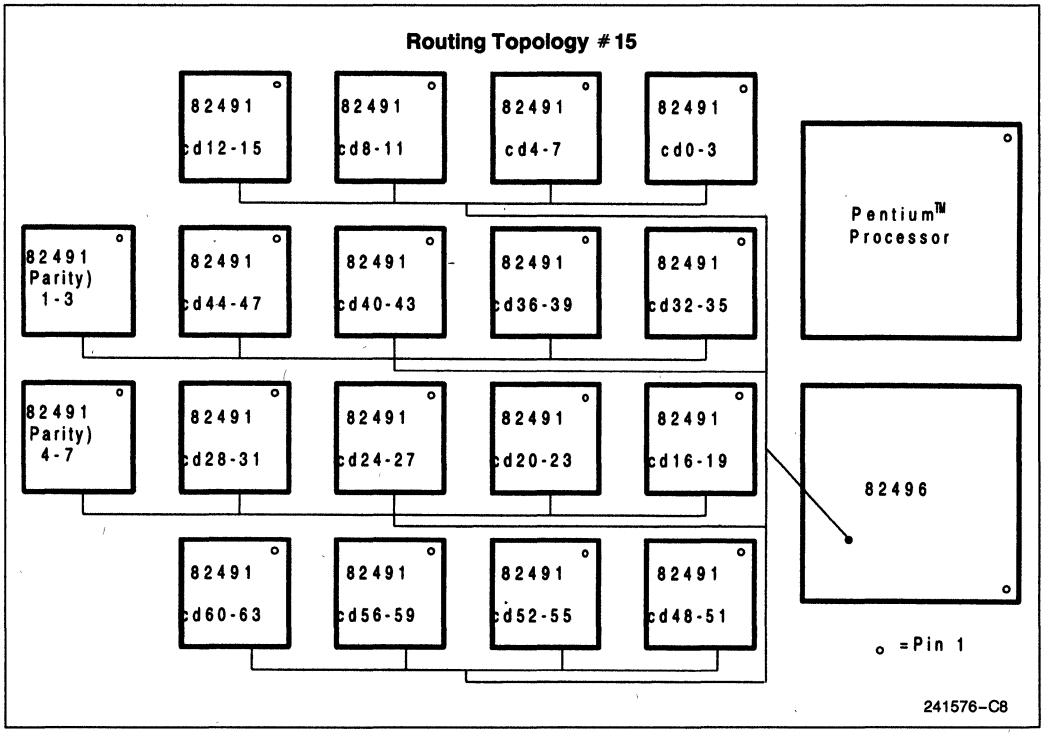
NET	PP-Tee	Tee-CS# 9	Tee-CS#10	Tee-CS#18	Stubs
CBE4#	2709.8	520.7	440.7	4762.9	75.0–135.4

NET	PP-Tee	Tee-CS#11	Tee-CS#12	Tee-CS#18	Stubs
CBE5#	5035.3	520.7	440.7	3065.4	75.0–75.0

NET	PP-Tee	Tee-CS#13	Tee-CS#14	Tee-CS#18	Stubs
CBE6#	4215.7	532.4	440.7	4338.7	75.0–95.4

NET	PP-Tee	Tee-CS#15	Tee-CS#16	Tee-CS#18	Stubs
CBE7#	6132.2	520.7	440.7	2225.6	75.0–105.7

8.6.9.7 82496 Control



NET	CC–Tee	Tee-CS#9	Tee to CS#5	CS#5 to CS#13	CS#1 to CS#9	CS#1–CS#2	CS#2–CS#3	CS#3–CS#4	CS#9–CS#10	CS#10–CS#11
BRDYC2	2077	598	595	935	932	940	940	940	540	940
MCYC	1707	540	527	1038	1041	940	940	940	1047	940
WRARR#	1820	573	576	976	976	940	940	940	940	940
WAY	1969	749	747	944	944	940	940	940	940	940

NET	CS#11–CS#12	CS#12–CS#17	CS#5–CS#6	CS#6–CS#7	CS#7–CS#8	CS#8–CS#18	CS#13–CS#14	CS#14–CS#15	CS#15–CS#16	Stubs
BRDYC2	940	940	940	940	940	940	940	940	940	85
MCYC	940	940	1047	940	940	940	940	940	940	75
WRARR#	940	940	940	940	940	940	940	940	940	135
WAY	940	940	940	940	940	940	940	940	940	75

References

1. Intel Corporation, *Pentium™ Processor User's Manual*, Order#: 241563.
2. Intel Corporation, *Pentium™ Processor Thermal Design Guide*, Application Note: AP-480, Order# 241575.
3. Intel Corporation, *Pentium Processor Clock Design*, Application Note: AP-479, Order# 241574.
4. Blood, William R., Jr., *MECL System Design Handbook*, 1988, Motorola Inc.
5. T. Rahal-Arabi and R. Suarez-Gartner, "*An Efficient Methodology for the Design of Optimum Electrical Performance of Interconnects in High Performance Systems*," IEEE Topical Meeting on Electrical Performance of Electronic Packaging, Tucson AZ, April 1992.
6. Huck, Scott, "*Simulating Intel's Optimized Interface with the Intel486™ DX CPU-Cache Chip-Set*," 1992, Intel Corp.

intel®

AP-485

APPLICATION NOTE

Intel Processor Identification with the CPUID Instruction

December 1994

Order Number: 241618-003

INTEL PROCESSOR IDENTIFICATION WITH THE CPUID INSTRUCTION

CONTENTS

CONTENTS

1.0 INTRODUCTION

As the Intel Architecture evolves, with the addition of new generations and models of processors (8086, 8088, Intel 286, Intel386™, Intel486™, and Pentium™ processors), it is essential that Intel provides an increasingly sophisticated means with which software can identify the features available on each processor. This identification mechanism has evolved in conjunction with the Intel Architecture as follows:

- Originally, Intel published code sequences that could detect minor implementation differences to identify processor generations.
- Later, with the advent of the Intel386 processor, Intel implemented processor signature identification, which provided the processor family, model, and stepping numbers to software at reset.
- As the Intel Architecture evolved, Intel extended the processor signature identification into the CPUID instruction. The CPUID instruction not only provides the processor signature, but also provides information about the features supported by and implemented on the Intel processor.

The evolution of processor identification was necessary because, as the Intel Architecture proliferates, the computing market must be able to tune processor functionality across processor generations and models that have differing sets of features. Anticipating that this trend will continue with future processor generations, the Intel Architecture implementation of the CPUID instruction is extensible.

This Application Note explains how to use the CPUID instruction in software applications, BIOS implementations, and tools. By taking advantage of the CPUID instruction, software developers can create software applications and tools that can execute compatibly across the widest range of Intel processor generations and models, past, present, and future.

1.1 Update Support

New Intel processor signature and feature bits information can be obtained from the user's manual, programmer's reference manual or appropriate documentation for a processor. In addition, Intel can provide you with updated versions of the programming examples included in this application note; contact your Intel representative for more information.

2.0 DETECTING THE CPUID INSTRUCTION

Intel has provided a straightforward method for detecting whether the CPUID instruction is available. This method uses the ID flag in bit 21 of the EFLAGS register. If software can change the value of this flag, the CPUID instruction is available. The program examples at the end of this Application Note show how to use the PUSHFD instruction to change the value of the ID flag.

3.0 OUTPUTS OF THE CPUID INSTRUCTION

Figure 1 summarizes the outputs of the CPUID instruction.

The CPUID instruction can be executed multiple times, each time with a different parameter value in the EAX register. The output depends on the value in the EAX register, as specified in Table 1. To determine the highest acceptable value in the EAX register, the program should set the EAX register parameter value to 0. In this case, the CPUID instruction returns the highest value that can be recognized in the EAX register. CPUID instruction execution should always use a parameter value that is less than or equal to this highest returned value. Currently, the highest value recognized by the CPUID instruction is 1. Future processors might recognize higher values.

The processor type, specified in bits 12 and 13, indicate whether the processor is an original OEM processor, an OverDrive™ processor, or is a dual processor (capable of being used in a dual processor system). Table 2 shows the processor type values that can be returned in bits 12 and 13 of the EAX register.

While any imitator of the Intel Architecture can provide the CPUID instruction, no imitator can legitimately claim that its part is a genuine Intel part. Therefore, the presence of the `GenuineIntel` string is an assurance that the CPUID instruction and the processor signature are implemented as described in this document.

3.1 Vendor-ID String

If the EAX register contains a value of 0, the vendor identification string is returned in the EBX, EDX, and ECX registers. These registers contain the ASCII string `GenuineIntel`.

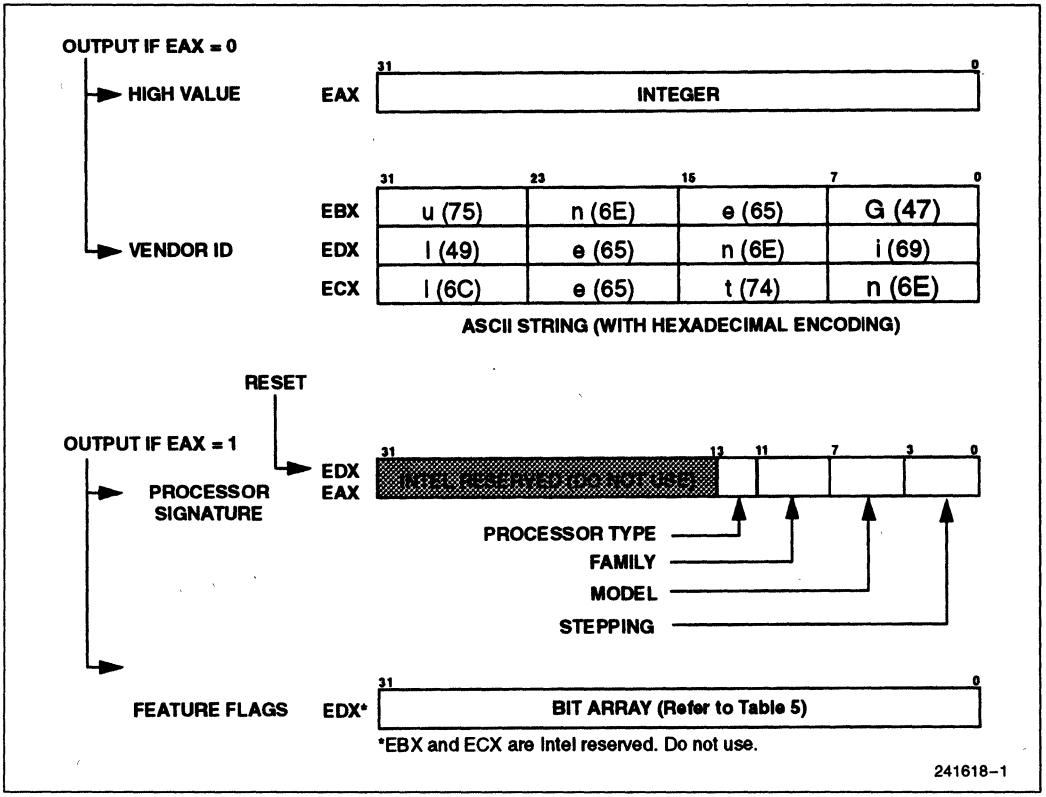


Figure 1. CPUID Instruction Outputs

Table 1. Effects of EAX Contents on CPUID Instruction Output

Parameter	Outputs of CPUID
EAX = 0	EAX ← Highest value recognized
	EBX:EDX:ECX ← Vendor identification string
EAX = 1	EAX ← Processor signature
	EDX ← Feature flags
	EBX:ECX ← Intel reserved (Do not use.)
1 < EAX ≤ highest value	Currently undefined
EAX > highest value	EAX:EBX:ECX:EDX ← Undefined (Do not use.)

Table 2. Processor Type

Bit Position	Value	Description
13,12	00	Original OEM Processor
	01	OverDrive™ Processor
	10	Dual Processor(1)
	11	Intel reserved (Do not use.)

NOTE:
1. Not applicable to Intel 386 and Intel486 processors.

3.2 Processor Signature

Beginning with the Intel386 processor family, the processor signature has been available at reset. With processors that implement the CPUID instruction, the processor signature is available both upon reset and upon execution of the CPUID instruction. Figure 1 shows the format of the signature for the Intel486 and Pentium processor families. Table 3 shows the values that are currently defined. (The high-order 18 bits are undefined and reserved.)

Older versions of Intel486 SX, Intel486 DX and IntelDX2 processors do not support the CPUID instruction. Therefore, the processor signature is only available upon reset for these processors. Refer to the programming examples at the end of this Application Note to determine which processors support the CPUID instruction.

On Intel386 processors, the format of the processor signature is somewhat different, as Figure 2 shows. Table 4 gives the current values.

Table 3. Intel486™ and Pentium™ Processor Signatures

Family	Model	Stepping(1)	Description
0100	0000 and 0001	xxxx	Intel486 DX Processors
0100	0010	xxxx	Intel486 SX Processors
0100	0011	xxxx	Intel487™ Processors(2)
0100	0011	xxxx	IntelDX2™ and Intel DX2 OverDrive™ Processors
0100	0100	xxxx	Intel486 SL Processor(2)
0100	0101	xxxx	IntelSX2™ Processors
0100	0111	xxxx	Write-Back Enhanced IntelDX2 Processors
0100	1000	xxxx	IntelDX4™ and IntelDX4 OverDrive Processors
0101	0001	xxxx	Pentium™ Processors (510\60, 567\66)
0101	0010	xxxx	Pentium Processors (735\90, 815\100)
0101	0011	xxxx	Pentium OverDrive Processors
0101	0101	xxxx	Reserved for Pentium OverDrive Processor for IntelDX4 Processor
0101	0010	xxxx	Reserved for Pentium OverDrive Processor for Pentium Processor (510\60, 567\66)
0101	0100	xxxx	Reserved for Pentium OverDrive Processor for Pentium Processor (735\90, 815\100)

NOTES:
1. Intel releases information about stepping numbers as needed.
2. This processor does not implement the CPUID instruction.

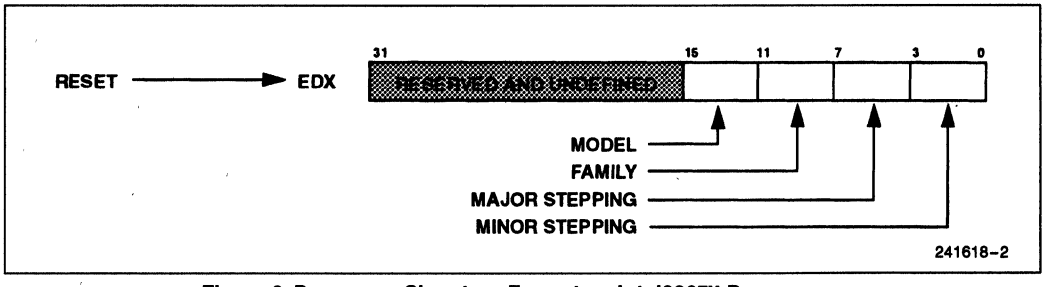


Figure 2. Processor Signature Format on Intel386™ Processors

Table 4. Intel386™ Processor Signatures

Model	Family	Major Stepping	Minor Stepping(1)	Description
0000	0011	0000	xxxx	Intel386™ DX Processor
0010	0011	0000	xxxx	Intel386 SX Processor
0010	0011	0000	xxxx	Intel386 CX Processor
0010	0011	0000	xxxx	Intel386 EX Processor
0100	0011	0000 and 0001	xxxx	Intel386 SL Processor
0000	0011	0100	xxxx	RAPIDCAD™ Processor

NOTE:
1. Intel releases information about minor stepping numbers as needed.

3.3 Feature Flags

When a value of 1 is placed in the EAX register, the CPUID instruction loads the EDX register with the feature flags. The feature flags indicate which features the processor supports. A value of 1 in a feature flag can indicate that a feature is either supported or not supported, depending on the implementation of the CPUID instruction for a specific processor. Table 5 lists the currently defined feature flag values. For future processors, refer to the programmer's reference manual, user's manual, or the appropriate documentation for the latest feature flag values.

Developers should use the feature flags in applications to determine which processor features are supported. By using the CPUID feature flags to predetermine processor features, software can detect and avoid incompatibilities that could result if the features are not present.

Table 5. Feature Flag Values

Bit	Name	Description When Flag = 1	Comments
0	FPU	Floating-Point Unit On-Chip	The processor contains an FPU that supports the Intel387 floating-point instruction set.
1	VME	Virtual Mode Extension	The processor supports extensions to virtual-8086 mode.
2(1)			(See note)
3	PSE	Page Size Extension	The processor supports 4-Mbyte pages.
4–6(1)			(See note)
7	MCE	Machine Check	Exception 18 is defined for Pentium processor style machine checks, including CR4.MCE for controlling the feature. This feature does not define the model-specific implementation of the machine-check error logging reporting and processor shutdowns. Machine-check exception handlers may have to depend on processor version to do model-specific processing of the exception or test for the presence of the standard machine-check feature.
8	CX8	CMPXCHG8B	The 8-byte (64-bit) compare and exchange instructions is supported (implicitly locked and atomic).
9	APIC	On-Chip APIC	Indicates that an integrated APIC is present and hardware enabled. (Software disabling does not affect this bit.)
10–31(1)			(See note)

NOTE:

1. Some non-essential information regarding Intel486 and Pentium processors is considered Intel confidential and proprietary and is not documented in this publication. This information is provided in the *Supplement to the Pentium™ Processor User's Manual* and is available with the appropriate non-disclosure agreements in place. Contact Intel Corporation for details.

4.0 USAGE GUIDELINES

This document presents Intel-recommended feature-detection methods. Software should not try to identify features by exploiting programming tricks, undocumented features, or otherwise deviating from the guidelines presented in this Application Note. The following is a list of guidelines that can help programmers maintain the widest range of compatibility for their software.

- Do not depend on the absence of an invalid opcode trap on the CPUID opcode to detect CPUID. Do not depend on the absence of an invalid opcode trap on the PUSHFD opcode to detect a 32-bit processor. Test the ID flag, as described in Section 2.0 and shown in Section 6.0.
- Do not assume that a given family or model has any specific feature. For example, do not assume that, because the family value is 5 (Pentium processor), there must be a floating-point unit on-chip. Use the feature flags for this determination.
- Do not assume that the features in the OverDrive processors are the same as those in the OEM version of the processor. Internal caches and instruction execution might vary.
- Do not use undocumented features of a processor to identify steppings or features. For example, the Intel386 processor A-step had bit instructions that were withdrawn with B-step. Some software attempted to execute these instructions and depended on the invalid-opcode exception as a signal that it was not running on the A-step part. This software failed to word correctly when the Intel486 processor used the same opcodes for different instructions. That software should have used the stepping information in the processor signature.
- Do not assume that a value of 1 in a feature flag ind icates that a given feature is present, even though that is the case in the first models of the Pentium processor in which the CPUID instruction is implemented. For some feature flags that might be defined in the future, a value of 1 can indicate that the corresponding feature is not present.
- Programmers should test feature flags individually and not make assumptions about undefined bits. It would be a mistake, for example, to test the FPU bit by comparing the feature register to a binary 1 with a compare instruction.
- Do not assume that the clock of a given family or model runs at a specific frequency and do not write clock-dependent code, such as timing loops. For instance, an OverDrive Processor could operate at a higher internal frequency and still report the same family and/or model. Instead, use the system's timers to measure elapsed time.
- Processor model-specific registers may differ among processors, including in various models of the Pentium processor. Do not use these registers unless identified for the installed processor.

5.0 BIOS RECOGNITION FOR INTEL OVERDRIVE™ PROCESSORS

A system's BIOS will typically identify the processor in the system and initialize the hardware accordingly. In many cases, the BIOS identifies the processor by reading the processor signature, comparing it to known signatures, and, upon finding a match, executing the corresponding hardware initialization code.

The Pentium OverDrive processor is designed to be an upgrade to any Intel486 family processor. Because there are significant operational differences between these two processor families, processor misidentification can cause system failures or diminished performance. Major differences between the Intel486 processor and the Pentium OverDrive processor include the type of on-chip cache supported (write-back or write-through), cache organization and cache size. The OverDrive processor also has an enhanced floating point unit and System Management Mode (SMM) that may not exist in the OEM processor. Inability to recognize these features causes problems like those described below.

In many BIOS implementations, the BIOS reads the processor signature at reset and compares it to known values. If the OverDrive processor's signature is not among the known values, a match will not occur and the OverDrive processor will not be identified. Often the BIOS will drop out of the search and initialize the hardware based on a default case such as initializing the chipset for an Intel486 SX processor. Following are two common examples of system failures and how to avoid them.

Example 1

If (for the Pentium OverDrive processor) the system's hardware is configured to enable the write-back cache but the BIOS fails to detect the Pentium OverDrive processor signature, the BIOS may incorrectly cause the chipset to support a write-through processor cache. This results in a data incoherency problem with the bus masters. When a bus master accesses a memory location (which was also in the processor's cache in a modified state), the processor will alert the chipset to allow it to update this data in memory. But the chipset is not programmed for such an event and the bus master instead receives stale data. This usually results in a system failure.

Example 2

If the BIOS does not recognize the OverDrive processor's signature and defaults to an Intel486 SX processor, the BIOS can incorrectly program the chipset to ignore, or improperly route, the assertion of the floating point error signaled by the processor. The result is that floating point errors will be improperly handled by the Pentium OverDrive processor. The BIOS may also completely disable math exception handling in the OverDrive processor. This can cause installation errors in applications that require hardware support for floating point instructions.

Hence, when programming or modifying a BIOS, be aware of the impact of future OverDrive processors. Intel recommends that you include processor signatures for the OverDrive processors in BIOS identification routines to eliminate diminished performance or system failures. The recommendations in this application note can help a BIOS maintain compatibility across a wide range of processor generations and models.

6.0 PROPER IDENTIFICATION SEQUENCE

The `cpuid3a.asm` program example demonstrates the correct use of the CPUID instruction. (See Example 1.) It also shows how to identify earlier processor generations that do not implement the processor signature or CPUID instruction. This program example contains the following two procedures:

- `get_cpu_type` identifies the processor type. Figure 3 illustrates the flow of this procedure.
- `get_fpu_type` determines the type of floating-point unit (FPU) or math coprocessor (MCP).

This procedure has been tested with 8086, 80286, Intel386, Intel486, and Pentium processors. This program example is written in assembly language and is suitable for inclusion in a run-time library, or as system calls in operating systems.

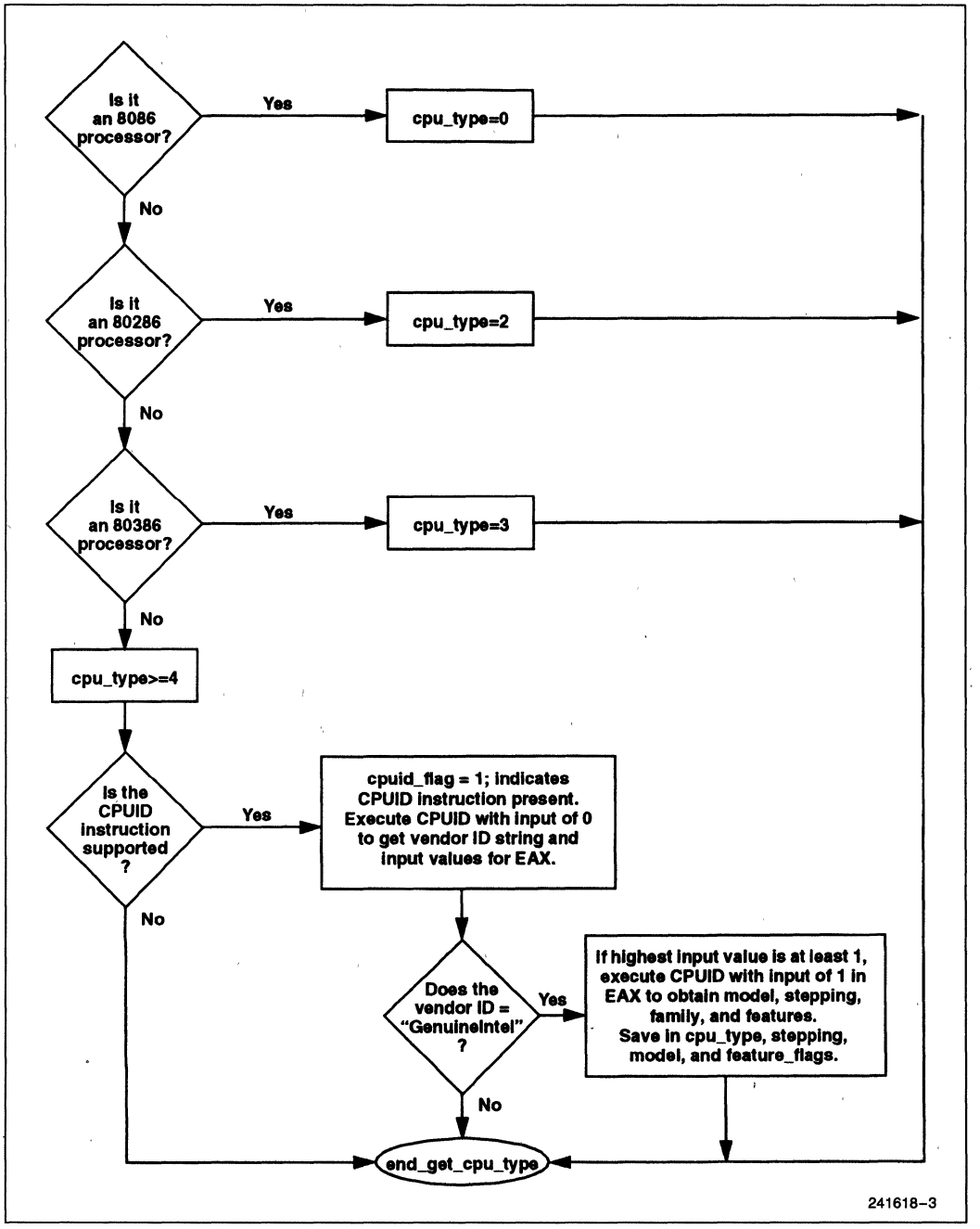


Figure 3. Flow of Processor get__cpu__type Procedure

7.0 USAGE PROGRAM EXAMPLE

The `cpuid3b.asm` and `cpuid3b.c` program examples demonstrate applications that call `get_cpu_type` and `get_fpu_type` procedures and interpret the returned information. The results, which are displayed on the monitor, identify the installed processor and features. The `cpuid3b.asm` example is written in assembly language and demonstrates an application that displays the returned information in the DOS environment. The `cpuid3b.c` example is written in the C language. (See Examples 2 and 3.)

Figure 4 presents an overview of the relationship between the three program examples.

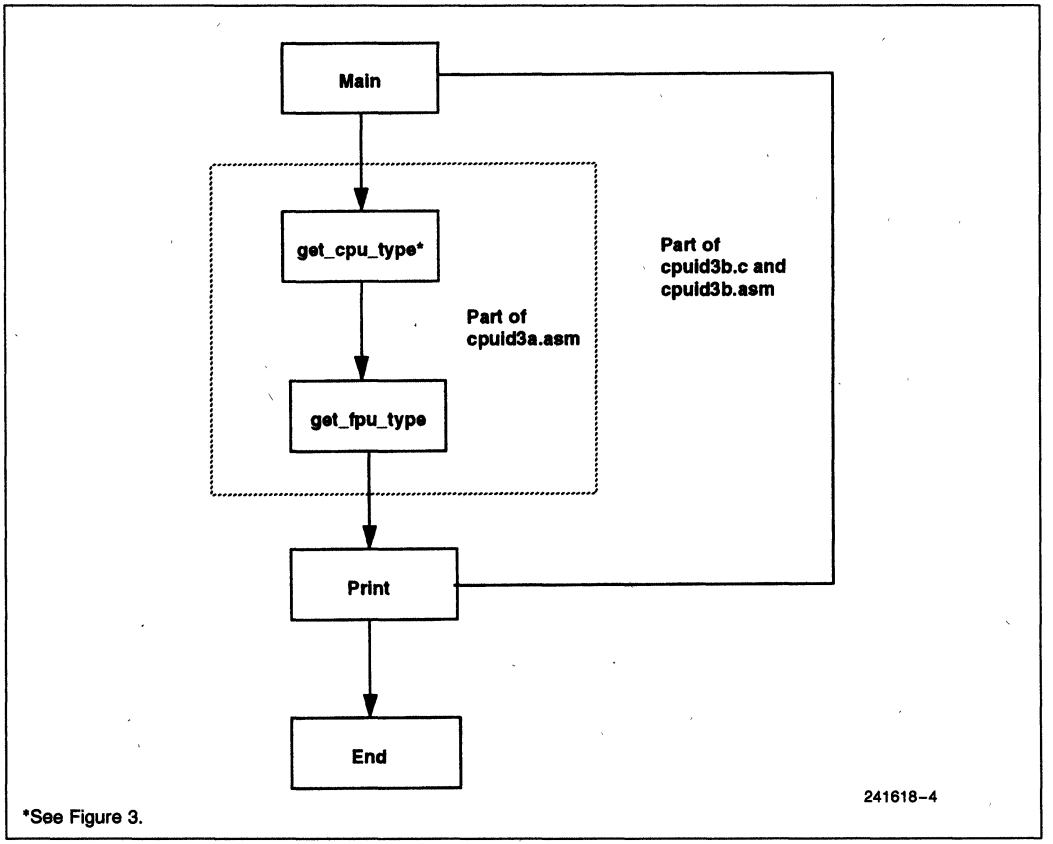


Figure 4. Flow of Processor Identification Extraction Procedures

Example 1. Processor Identification Extraction Procedure

```
;       Filename:       cpuid3a.asm
;       Copyright 1993, 1994 by Intel Corp.
;
;       This program has been developed by Intel Corporation.  You
;       have Intel's permission to incorporate this source code into
;       your product, royalty free.  Intel has intellectual property
;       rights which it may assert if another manufacturer's processor
;       mis-identifies itself as being "GenuineIntel" when the CPUID
;       instruction is executed.
;
;       Intel specifically disclaims all warranties, express or
;       implied, and all liability, including consequential and other
;       indirect damages, for the use of this code, including
;       liability for infringement of any proprietary rights, and
;       including the warranties of merchantability and fitness for a
;       particular purpose.  Intel does not assume any responsibility
;       for any errors which may appear in this code nor any
;       responsibility to update it.
;
;       This code contains two procedures:
;       _get_cpu_type: Identifies processor type in _cpu_type:
;               0=8086/8088 processor
;               2=Intel 286 processor
;               3=Intel386(TM) family processor
;               4=Intel486(TM) family processor
;               5=Pentium(TM) family processor
;
;       _get_fpu_type: Identifies FPU type in _fpu_type:
;               0=FPU not present
;               1=FPU present
;               2=287 present (only if _cpu_type=3)
;               3=387 present (only if _cpu_type=3)
;
;       This program has been tested with the MASM assembler.
;       This code correctly detects the current Intel 8086/8088,
;       80286, 80386, 80486, and Pentium(tm) processors in the
;       real-address mode.
;
;       To assemble this code with TASM, add the JUMPS directive.
;       jumps                   ; Uncomment this line for TASM


        TITLE   cpuid3a.asm
        DOSSEG
        .model  small

CPU_ID  MACRO
```

241618-5

```
        db      0fh             ; Hardcoded CPUID instruction
        db      0a2h
ENDM

        .data
        public  _cpu_type
        public  _fpu_type
        public  _cpuid_flag
        public  _intel_CPU
        public  _vendor_id
        public  _cpu_signature
        public  _features_ecx
        public  _features_edx
        public  _features_ebx
_cpu_type       db      0
_fpu_type       db      0
_cpuid_flag     db      0
_intel_CPU      db      0
_vendor_id      db      "------------"
intel_id        db      "GenuineIntel"
_cpu_signature  dd      0
_features_ecx   dd      0
_features_edx   dd      0
_features_ebx   dd      0
fp_status       dw      0

        .code
        .8086

;*************************************************************************

        public  _get_cpu_type
_get_cpu_type   proc

;       This procedure determines the type of processor in a system
;       and sets the _cpu_type variable with the appropriate
;       value.  If the CPUID instruction is available, it is used
;       to determine more specific details about the processor.
;       All registers are used by this procedure, none are preserved.
;       To avoid AC faults, the AM bit in CR0 must not be set.

;       Intel 8086 processor check
;       Bits 12-15 of the FLAGS register are always set on the
;       8086 processor.

check_8086:
        pushf                   ; push original FLAGS
        pop     ax              ; get original FLAGS
        mov     cx, ax          ; save original FLAGS
        and     ax, 0fffh       ; clear bits 12-15 in FLAGS
```

241618-6

```
        push    ax              ; save new FLAGS value on stack
        popf                    ; replace current FLAGS value
        pushf                   ; get new FLAGS
        pop     ax              ; store new FLAGS in AX
        and     ax, 0f000h      ; if bits 12-15 are set, then
        cmp     ax, 0f000h      ;   processor is an 8086/8088
        mov     _cpu_type, 0    ; turn on 8086/8088 flag
        je      end_cpu_type    ; jump if processor is 8086/8088

;       Intel 286 processor check
;       Bits 12-15 of the FLAGS register are always clear on the
;       Intel 286 processor in real-address mode.

        .286
check_80286:
        or      cx, 0f000h      ; try to set bits 12-15
        push    cx              ; save new FLAGS value on stack
        popf                    ; replace current FLAGS value
        pushf                   ; get new FLAGS
        pop     ax              ; store new FLAGS in AX
        and     ax, 0f000h      ; if bits 12-15 are clear
        mov     _cpu_type, 2    ; processor=80286, turn on 80286 flag
        jz      end_cpu_type    ; if no bits set, processor is 80286

;       Intel386 processor check
;       The AC bit, bit #18, is a new bit introduced in the EFLAGS
;       register on the Intel486 processor to generate alignment
;       faults.
;       This bit cannot be set on the Intel386 processor.

        .386                    ; it is safe to use 386 instructions
check_80386:
        pushfd                  ; push original EFLAGS
        pop     eax             ; get original EFLAGS
        mov     ecx, eax        ; save original EFLAGS
        xor     eax, 40000h     ; flip AC bit in EFLAGS
        push    eax             ; save new EFLAGS value on stack
        popfd                   ; replace current EFLAGS value
        pushfd                  ; get new EFLAGS
        pop     eax             ; store new EFLAGS in EAX
        xor     eax, ecx        ; can't toggle AC bit, processor=80386
        mov     _cpu_type, 3    ; turn on 80386 processor flag
        jz      end_cpu_type    ; jump if 80386 processor

        push    ecx
        popfd                   ; restore AC bit in EFLAGS first

;       Intel486 processor check
;       Checking for ability to set/clear ID flag (Bit 21) in EFLAGS
;       which indicates the presence of a processor with the CPUID
```

241618-7

```
;       instruction.

        .486
check_80486:
        mov     _cpu_type, 4    ; turn on 80486 processor flag
        mov     eax, ecx        ; get original EFLAGS
        xor     eax, 200000h    ; flip ID bit in EFLAGS
        push    eax             ; save new EFLAGS value on stack
        popfd                   ; replace current EFLAGS value
        pushfd                  ; get new EFLAGS
        pop     eax             ; store new EFLAGS in EAX
        xor     eax, ecx        ; can't toggle ID bit,
        je      end_cpu_type    ; processor=80486

;       Execute CPUID instruction to determine vendor, family,
;       model, stepping and features.  For the purpose of this
;       code, only the initial set of CPUID information is saved.

        mov     _cpuid_flag, 1  ; flag indicating use of CPUID inst.
        push    ebx             ; save registers
        push    esi
        push    edi
        mov     eax, 0          ; set up for CPUID instruction
        CPU_ID                  ; get and save vendor ID

        mov     dword ptr _vendor_id, ebx
        mov     dword ptr _vendor_id[+4], edx
        mov     dword ptr _vendor_id[+8], ecx

        mov     si, ds
        mov     es, si

        mov     si, offset _vendor_id
        mov     di, offset intel_id
        mov     cx, 12          ; should be length intel_id
        cld                     ; set direction flag
        repe    cmpsb           ; compare vendor ID to "GenuineIntel"
        jne     end_cpuid_type  ; if not equal, not an Intel processor

        mov     _intel_CPU, 1   ; indicate an Intel processor
        cmp     eax, 1          ; make sure 1 is valid input for CPUID
        jl      end_cpuid_type  ; if not, jump to end
        mov     eax, 1
        CPU_ID                  ; get family/model/stepping/features
        mov     _cpu_signature, eax
        mov     _features_ebx, ebx
        mov     _features_edx, edx
        mov     _features_ecx, ecx

        shr     eax, 8          ; isolate family
```

241618-8

```
        and     eax, 0fh
        mov     _cpu_type, al    ; set _cpu_type with family

end_cpuid_type:
        pop     edi              ; restore registers
        pop     esi
        pop     ebx

        .8086
end_cpu_type:
        ret
_get_cpu_type   endp

;*************************************************************************

        public  _get_fpu_type
_get_fpu_type   proc

;       This procedure determines the type of FPU in a system
;       and sets the _fpu_type variable with the appropriate value.
;       All registers are used by this procedure, none are preserved.

;       Coprocessor check
;       The algorithm is to determine whether the floating-point
;       status and control words are present.  If not, no
;       coprocessor exists.  If the status and control words can
;       be saved, the correct coprocessor is then determined
;       depending on the processor type.  The Intel386 processor can
;       work with either an Intel287 NDP or an Intel387 NDP.
;       The infinity of the coprocessor must be checked to determine
;       the correct coprocessor type.

        fninit                   ; reset FP status word
        mov     fp_status, 5a5ah; initialize temp word to non-zero
        fnstsw  fp_status        ; save FP status word
        mov     ax, fp_status    ; check FP status word
        cmp     al, 0            ; was correct status written
        mov     _fpu_type, 0     ; no FPU present
        jne     end_fpu_type

check_control_word:
        fnstcw  fp_status        ; save FP control word
        mov     ax, fp_status    ; check FP control word
        and     ax, 103fh        ; selected parts to examine
        cmp     ax, 3fh          ; was control word correct
        mov     _fpu_type, 0
        jne     end_fpu_type     ; incorrect control word, no FPU
        mov     _fpu_type, 1

        80287/80387 check for the Intel386 processor
```

241618-9

```
check_infinity:
        cmp     _cpu_type, 3
        jne     end_fpu_type
        fld1                        ; must use default control from FNINIT
        fldz                        ; form infinity
        fdiv                        ; 8087/Intel287 NDP say +inf = -inf
        fld     st                  ; form negative infinity
        fchs                        ; Intel387 NDP says +inf <> -inf
        fcompp                      ; see if they are the same
        fstsw   fp_status           ; look at status from FCOMPP
        mov     ax, fp_status
        mov     _fpu_type, 2        ; store Intel287 NDP for FPU type
        sahf                        ; see if infinities matched
        jz      end_fpu_type        ; jump if 8087 or Intel287 is present
        mov     _fpu_type, 3        ; store Intel387 NDP for FPU type
end_fpu_type:
        ret
_get_fpu_type   endp

        end
```

241618-10

Example 2. Processor Identification Procedure in Assembly Language

```
;       Filename:       cpuid3b.asm
;       Copyright 1993, 1994 by Intel Corp.
;
;       This program has been developed by Intel Corporation.  You
;       have Intel's permission to incorporate this source code into
;       your product, royalty free.  Intel has intellectual property
;       rights which it may assert if another manufacturer's processor
;       mis-identifies itself as being "GenuineIntel" when the CPUID
;       instruction is executed.
;
;       Intel specifically disclaims all warranties, express or
;       implied, and all liability, including consequential and other
;       indirect damages, for the use of this code, including
;       liability for infringement of any proprietary rights, and
;       including the warranties of merchantability and fitness for a
;       particular purpose.  Intel does not assume any responsibility
;       for any errors which may appear in this code nor any
;       responsibility to update it.
;
;       This program contains three parts:
;       Part 1: Identifies processor type in the variable _cpu_type:
;
;       Part 2: Identifies FPU type in the variable _fpu_type:
;
;       Part 3: Prints out the appropriate message.  This part is
;               specific to the DOS environment and uses the DOS
;               system calls to print out the messages.
;
;       This program has been tested with the MASM assembler.
;       If this code is assembled with no options specified and linked
;       with the cpuid3a.asm module, it correctly identifies the
;       current Intel 8086/8088, 80286, 80386, 80486, and Pentium(tm)
;       processors in the real-address mode.
;
;       To assemble this code with TASM, add the JUMPS directive.
;       jumps                   ; Uncomment this line for TASM


        TITLE   cpuid3b.asm
        DOSSEG
        .model  small
        .stack  100h

        .data
        extrn   _cpu_type: byte
        extrn   _fpu_type: byte
        extrn   _cpuid_flag: byte
```

241618-11

```
        extrn   _intel_CPU: byte
        extrn   _vendor_id: byte
        extrn   _cpu_signature: dword
        extrn   _features_ecx: dword
        extrn   _features_edx: dword
        extrn   _features_ebx: dword

;       The purpose of this code is to identify the processor and
;       coprocessor that is currently in the system.  The program
;       first determines the processor type.  Then it determines
;       whether a coprocessor exists in the system.  If a
;       coprocessor or integrated coprocessor exists, the program
;       identifies the coprocessor type.  The program then prints
;       the processor and floating point processors present and type.

        .code
        .8086
start:  mov     ax, @data
        mov     ds, ax          ; set segment register
        mov     es, ax          ; set segment register
        and     sp, not 3       ; align stack to avoid AC fault
        call    _get_cpu_type   ; determine processor type
        call    _get_fpu_type
        call    print
        mov     ax, 4c00h       ; terminate program
        int     21h

;*************************************************************************

        extrn   _get_cpu_type: proc

;*************************************************************************

        extrn   _get_fpu_type: proc

;*************************************************************************


FPU_FLAG        equ     0001h
VME_FLAG        equ     0002h
PSE_FLAG        equ     0008h
MCE_FLAG        equ     0080h
CMPXCHG8B_FLAG  equ     0100h
APIC_FLAG       equ     0200h

        .data
id_msg          db      "This system has a$"
cp_error        db      "n unknown processor$"
cp_8086         db      "n 8086/8088 processor$"
cp_286          db      "n 80286 processor$"
```

241618-12

```
cp_386          db      "n 80386 processor$"

cp_486          db      "n 80486DX, 80486DX2 processor or"
                db      " 80487SX math coprocessor$"
cp_486sx        db      "n 80486SX processor$"

fp_8087         db      " and an 8087 math coprocessor$"
fp_287          db      " and an 80287 math coprocessor$"
fp_387          db      " and an 80387 math coprocessor$"

intel486_msg    db      " Genuine Intel486(TM) processor$"
intel486dx_msg  db      " Genuine Intel486(TM) DX processor$"
intel486sx_msg  db      " Genuine Intel486(TM) SX processor$"
inteldx2_msg    db      " Genuine IntelDX2(TM) processor$"
intelsx2_msg    db      " Genuine IntelSX2(TM) processor$"
inteldx4_msg    db      " Genuine IntelDX4(TM) processor$"
inteldx2wb_msg  db      " Genuine Write-Back Enhanced"
                db      " IntelDX2(TM) processor$"
pentium_msg     db      " Genuine Intel Pentium(TM) processor$"
unknown_msg     db      "n unknown Genuine Intel processor$"

; The following 16 entries must stay intact as an array
intel_486_0     dw      offset intel486dx_msg
intel_486_1     dw      offset intel486dx_msg
intel_486_2     dw      offset intel486sx_msg
intel_486_3     dw      offset inteldx2_msg
intel_486_4     dw      offset intel486_msg
intel_486_5     dw      offset intelsx2_msg
intel_486_6     dw      offset intel486_msg
intel_486_7     dw      offset inteldx2wb_msg
intel_486_8     dw      offset inteldx4_msg
intel_486_9     dw      offset intel486_msg
intel_486_a     dw      offset intel486_msg
intel_486_b     dw      offset intel486_msg
intel_486_c     dw      offset intel486_msg
intel_486_d     dw      offset intel486_msg
intel_486_e     dw      offset intel486_msg
intel_486_f     dw      offset intel486_msg
; end of array

family_msg      db      13,10,"Processor Family:  $"
model_msg       db      13,10,"Model:             $"
stepping_msg    db      13,10,"Stepping:          "
cr_lf           db      13,10,"$"

turbo_msg       db      13,10,"The processor is an OverDrive(TM)"
                db      " processor$"
dp_msg          db      13,10,"The processor is the upgrade processor"
                db      " in a dual processor system$"
fpu_msg         db      13,10,"The processor contains an on-chip FPU$"
```

241618-13

```
mce_msg         db      13,10,"The processor supports Machine Check"
                db      " Exceptions$"
cmp_msg         db      13,10,"The processor supports the CMPXCHG8B"
                db      " instruction$"
vme_msg         db      13,10,"The processor supports Virtual Mode"
                db      " Extensions$"
pse_msg         db      13,10,"The processor supports Page Size"
                db      " Extensions$"
apic_msg        db      13,10,"The processor contains an on-chip"
                db      " APIC$"

not_intel       db      "t least an 80486 processor."
                db      13,10,"It does not contain a Genuine Intel"
                db      " part and as a result, the",13,10,"CPUID"
                db      " detection information cannot be determined"
                db      " at this time.$"

ASC_MSG MACRO   msg
        LOCAL   ascii_done              ; local label
        add     al, 30h
        cmp     al, 39h                 ; is it 0-9?
        jle     ascii_done
        add     al, 07h
ascii_done:
        mov     byte ptr msg[20], al
        mov     dx, offset msg
        mov     ah, 9h
        int     21h
ENDM

        .code
        .8086
print   proc

;       This procedure prints the appropriate cpuid string and
;       numeric processor presence status.  If the CPUID instruction
;       was used, this procedure prints out the CPUID info.
;       All registers are used by this procedure, none are preserved.

        mov     dx, offset id_msg       ; print initial message
        mov     ah, 9h
        int     21h

        cmp     _cpuid_flag, 1          ; if set to 1, processor
                                        ;   supports CPUID instruction
        je      print_cpuid_data        ; print detailed CPUID info

print_86:
        cmp     _cpu_type, 0
        jne     print_286
```

241618-14

```
        mov     dx, offset cp_8086
        mov     ah, 9h
        int     21h
        cmp     _fpu_type, 0
        je      end_print
        mov     dx, offset fp_8087
        mov     ah, 9h
        int     21h
        jmp     end_print

print_286:
        cmp     _cpu_type, 2
        jne     print_386
        mov     dx, offset cp_286
        mov     ah, 9h
        int     21h
        cmp     _fpu_type, 0
        je      end_print
print_287:
        mov     dx, offset fp_287
        mov     ah, 9h
        int     21h
        jmp     end_print

print_386:
        cmp     _cpu_type, 3
        jne     print_486
        mov     dx, offset cp_386
        mov     ah, 9h
        int     21h
        cmp     _fpu_type, 0
        je      end_print
        cmp     _fpu_type, 2
        je      print_287
        mov     dx, offset fp_387
        mov     ah, 9h
        int     21h
        jmp     end_print

print_486:
        cmp     _cpu_type, 4
        jne     print_unknown           ; Intel processors will have
        mov     dx, offset cp_486sx     ; CPUID instruction
        cmp     _fpu_type, 0
        je      print_486sx
        mov     dx, offset cp_486
print_486sx:
        mov     ah, 9h
        int     21h
        jmp     end_print
```

241618-15

```
print_unknown:
        mov     dx, offset cp_error
        jmp     print_486sx

print_cpuid_data:
        .486
        cmp     _intel_CPU, 1           ; check for genuine Intel
        jne     not_GenuineIntel        ;   processor
print_486_type:
        cmp     _cpu_type, 4            ; if 4, print 80486 processor
        jne     print_pentium_type
        mov     ax, word ptr _cpu_signature
        shr     ax, 4
        and     eax, 0fh                ; isolate model
        mov     dx, intel_486_0[eax*2]
        jmp     print_common
print_pentium_type:
        cmp     _cpu_type, 5            ; if 5, print Pentium processor
        jne     print_unknown_type
        mov     dx, offset pentium_msg
        jmp     print_common
print_unknown_type:
        mov     dx, offset unknown_msg  ; if neither, print unknown

print_common:
        mov     ah, 9h
        int     21h

; print family, model, and stepping

print_family:
        mov     al, _cpu_type
        ASC_MSG family_msg              ; print family msg

print_model:
        mov     ax, word ptr _cpu_signature
        shr     ax, 4
        and     al, 0fh
        ASC_MSG model_msg               ; print model msg

print_stepping:
        mov     ax, word ptr _cpu_signature
        and     al, 0fh
        ASC_MSG stepping_msg            ; print stepping msg

print_upgrade:
        mov     ax, word ptr _cpu_signature
        test    ax, 1000h               ; check for turbo upgrade
        jz      check_dp
```

241618-16

```
        mov     dx, offset turbo_msg
        mov     ah, 9h
        int     21h
        jmp     print_features

check_dp:
        test    ax, 2000h               ; check for dual processor
        jz      print_features
        mov     dx, offset dp_msg
        mov     ah, 9h
        int     21h

print_features:
        mov     ax, word ptr _features_edx
        and     ax, FPU_FLAG            ; check for FPU
        jz      check_MCE
        mov     dx, offset fpu_msg
        mov     ah, 9h
        int     21h

check_MCE:
        mov     ax, word ptr _features_edx
        and     ax, MCE_FLAG            ; check for MCE
        jz      check_CMPXCHG8B
        mov     dx, offset mce_msg
        mov     ah, 9h
        int     21h

check_CMPXCHG8B:
        mov     ax, word ptr _features_edx
        and     ax, CMPXCHG8B_FLAG      ; check for CMPXCHG8B
        jz      check_VME
        mov     dx, offset cmp_msg
        mov     ah, 9h
        int     21h

check_VME:
        mov     ax, word ptr _features_edx
        and     ax, VME_FLAG            ; check for VME
        jz      check_PSE
        mov     dx, offset vme_msg
        mov     ah, 9h
        int     21h

check_PSE:
        mov     ax, word ptr _features_edx
        and     ax, PSE_FLAG            ; check for PSE
        jz      check_APIC
        mov     dx, offset pse_msg
        mov     ah, 9h
```

241618-17

```
        int     21h

check_APIC:
        mov     ax, word ptr _features_edx
        and     ax, APIC_FLAG           ; check for APIC
        jz      end_print
        mov     dx, offset apic_msg
        mov     ah, 9h
        int     21h

        jmp     end_print

not_GenuineIntel:
        mov     dx, offset not_intel
        mov     ah, 9h
        int     21h

end_print:
        mov     dx, offset cr_lf
        mov     ah, 9h
        int     21h
        ret
print   endp

        end     start
```

241618-18

Example 3. Processor Identification Procedure in the C Language

```
/* Filename:    cpuid3b.c                                          */
/* Copyright 1994 by Intel Corp.                                   */
/*                                                                 */
/* This program has been developed by Intel Corporation.  You      */
/* have Intel's permission to incorporate this source code into    */
/* your product, royalty free. Intel has intellectual property     */
/* rights which it may assert if another manufacturer's processor*/
/* mis-identifies itself as being "GenuineIntel" when the CPUID    */
/* instruction is executed.                                        */
/*                                                                 */
/* Intel specifically disclaims all warranties, express or         */
/* implied, and all liability, including consequential and other */
/* indirect damages, for the use of this code, including           */
/* liability for infringement of any proprietary rights, and       */
/* including the warranties of merchantability and fitness for a */
/* particular purpose.  Intel does not assume any responsibility */
/* for any errors which may appear in this code nor any            */
/* responsibility to update it.                                    */
/*                                                                 */
/* This program contains three parts:                              */
/* Part 1: Identifies CPU type in the variable _cpu_type:          */
/*                                                                 */
/* Part 2: Identifies FPU type in the variable _fpu_type:          */
/*                                                                 */
/* Part 3: Prints out the appropriate message.                     */
/*                                                                 */
/* This program has been tested with the Microsoft C compiler.    */
/* If this code is compiled with no options specified and linked */
/* with the cpuid3a.asm module, it correctly identifies the        */
/* current Intel 8086/8088, 80286, 80386, 80486, and               */
/* Pentium(tm) processors in the real-address mode.                */

#define FPU_FLAG         0x0001
#define VME_FLAG         0x0002
#define PSE_FLAG         0x0008
#define MCE_FLAG         0x0080
#define CMPXCHG8B_FLAG   0x0100
#define APIC_FLAG        0x0200

extern char cpu_type;
extern char fpu_type;
extern char cpuid_flag;
extern char intel_CPU;
extern char vendor_id[12];
extern long cpu_signature;
extern long features_ecx;
extern long features_edx;
```

241618-19

```
extern long features_ebx;
main() {
    get_cpu_type();
    get_fpu_type();
    print();
}
print() {
    printf("This system has a");
    if (cpuid_flag == 0) {
        switch (cpu_type) {
        case 0:
            printf("n 8086/8088 processor");
            if (fpu_type) printf(" and an 8087 math coprocessor");
            break;
        case 2:
            printf("n 80286 processor");
            if (fpu_type) printf(" and an 80287 math coprocessor");
            break;
        case 3:
            printf("n 80386 processor");
            if (fpu_type == 2)
                printf(" and an 80287 math coprocessor");
            else if (fpu_type)
                printf(" and an 80387 math coprocessor");
            break;
        case 4:
            if (fpu_type) printf("n 80486DX, 80486DX2 processor or \
80487SX math coprocessor");
            else printf("n 80486SX processor");
            break;
        default:
            printf("n unknown processor");
        }
    } else {
        /* using cpuid instruction */
        if (intel_CPU) {
            if (cpu_type == 4) {
                switch ((cpu_signature>>4)&0xf) {
                case  0:
                case  1:
                    printf(" Genuine Intel486(TM) DX processor");
                    break;
                case  2:
                    printf(" Genuine Intel486(TM) SX processor");
                    break;
                case  3:
                    printf(" Genuine IntelDX2(TM) processor");
                    break;
                case  4:
                    printf(" Genuine Intel486(TM) processor");
```

241618-20

```
                    break;
                case  5:
                    printf(" Genuine IntelSX2(TM) processor");
                    break;
                case  7:
                    printf(" Genuine Write-Back Enhanced \
IntelDX2(TM) processor");
                    break;
                case  8:
                    printf(" Genuine IntelDX4(TM) processor");
                    break;
                default:
                    printf(" Genuine Intel486(TM) processor");
                }
            } else if (cpu_type == 5)
                printf(" Genuine Intel Pentium(TM) processor");
            else
                printf("n unknown Genuine Intel processor");
            printf("\nProcessor Family: %X", cpu_type);
            printf("\nModel:            %X", (cpu_signature>>4)&0xf);
            printf("\nStepping:         %X\n", cpu_signature&0xf);
            if (cpu_signature & 0x1000)
                printf("\nThe processor is an OverDrive(TM) upgrade
          \processor");
            else if (cpu_signature & 0x2000)
                printf("\nThe processor is the upgrade processor \
in a dual processor system");
            if (features_edx & FPU_FLAG)
                printf("\nThe processor contains an on-chip FPU");
            if (features_edx & MCE_FLAG)
                printf("\nThe processor supports Machine Check \
Exceptions");
            if (features_edx & CMPXCHG8B_FLAG)
                printf("\nThe processor supports the CMPXCHG8B \
instruction");
            if (features_edx & VME_FLAG)
                printf("\nThe processor supports Virtual Mode \
Extensions");
            if (features_edx & PSE_FLAG)
                printf("\nThe processor supports Page Size \
Extensions");
            if (features_edx & APIC_FLAG)
                printf("\nThe processor contains an on-chip APIC");
        } else {
            printf("t least an 80486 processor.\nIt does not \
contain a Genuine Intel part and as a result, the\nCPUID detection \
information cannot be determined at this time.");
        }
    }
    printf("\n");
}
```

241618-21

Revision	Revision History	Date
-001	Original Issue.	05/93
-002	Modified Table 2. Intel486 and Pentium Processor Signatures.	10/93
-003	Updated to accommodate new processor versions. Program examples modified for ease of use, section added discussing BIOS recognition for OverDrive processors, and feature flag information updated.	09/94

NORTH AMERICAN SALES OFFICES

ALABAMA

Intel Corp
4024 Medford Drive
Huntsville 35802
Tel: (205) 883-6137
FAX: (205) 883-4826

ARIZONA

†Intel Corp
410 North 44th Street
Suite 470
Phoenix 85008
Tel: (800) 628-8686
FAX: (602) 244-0446

CALIFORNIA

Intel Corp
3550 Watt Avenue
Suite 140
Sacramento 95821
Tel: (800) 628-8686
FAX: (916) 488-1473

†Intel Corp.
9655 Granite Ridge Drive
3rd Floor, Suite 4A
San Diego 92123
Tel: (800) 628-8686
FAX. (619) 467-2460

Intel Corp
1781 Fox Drive
San Jose 95131
Tel: (800) 628-8686
FAX. (408) 441-9540

*†Intel Corp.
1551 N. Tustin Avenue
Suite 800
Santa Ana 92701
Tel: (800) 628-8686
TWX: 910-595-1114
FAX: (714) 541-9157

†Intel Corp
15260 Ventura Boulevard
Suite 360
Sherman Oaks 91403
Tel: (800) 628-8686
FAX: (818) 995-6624

Intel Corp.
120 Birmingham
Suite 110-114
Cardiff, CA 92007
Tel: (619) 942-8938
FAX: (619) 942-2849

Intel Corp.
300 N. Continental Blvd
Suite 100
El Segundo 90245
Tel. (800) 628-8686
FAX: (310) 640-7133

COLORADO

*†Intel Corp
600 S. Cherry St
Suite 700
Denver 80222
Tel: (800) 628-8686
TWX: 910-931-2289
FAX: (303) 322-8670

CONNECTICUT

†Intel Corp
40 Old Ridgebury Road
Suite 311
Danbury 06811
Tel: (800) 628-8686
FAX: (203) 778-2168

FLORIDA

†Intel Corp.
800 Fairway Drive
Suite 160
Deerfield Beach 33441
Tel (800) 628-8686
FAX: (305) 421-244

Intel Corp.
2250 Lucien Way
Suite 100, Room 8
Maitland 32751
Tel (800) 628-8686
FAX: (407) 660-1283GEORGIA

†Intel Corp.
20 Technology Park
Suite 150
Norcross 30092
Tel: (800) 628-8686
FAX. (404) 448-0875

IDAHO

Intel Corp
9456 Fairview Ave., Suite C
Boise 83704
Tel: (800) 628-8686
FAX. (208) 377-1052

ILLINOIS

*†Intel Corp.
Woodfield Corp. Center III
300 N. Martingale Road
Suite 400
Schaumburg 60173
Tell. (800) 628-8686
FAX. (708) 605-9762

INDIANA

†Intel Corp.
8041 Knue Road
Indianapolis 46250
Tel: (800) 628-8686
FAX. (317) 577-4939

MARYLAND

*†Intel Corp.
131 National Business Parkway
Suite 200
Annapolis Junction 20701
Tel: (800) 628-8686
FAX: (301) 206-3678

MASSACHUSETTS

*†Intel Corp
Westford Corp. Center
5 Carlisle Road
2nd Floor
Westford 01886
Tel. (800) 628-8686
TWX: 710-343-6333
FAX: (508) 692-7867

MICHIGAN

†Intel Corp
7071 Orchard Lake Road
Suite 100
West Bloomfield 48322
Tel (800) 628-8686
FAX: (313) 851-8770

Intel Corp
32255 N. Western Hwy
Suite 212, Tri Atria
Farmington Hills 48334
Tel. (800) 628-8686
FAX (313) 851-8770

MINNESOTA

†Intel Corp.
3500 W 80th St
Suite 360
Bloomington 55431
Tel: (800) 628-8686
TWX: 910-576-2867
FAX: (612) 831-6497

NEW JERSEY

Intel Corp.
2001 Route 46, Suite 310
Parsippany 07054-1315
Tel: (800) 628-8686
FAX (201) 402-4893

*†Intel Corp.
Lincroft Center
125 Half Mile Road
Red Bank 07701
Tel (800) 628-8686
FAX: (908) 747-0983NEW YORK

*Intel Corp
850 Cross Keys Office Park
Fairport 14450
Tel: (800) 628-8686
TWX. 510-253-7391
FAX: (716) 223-2561

*†Intel Corp
2950 Express Dr South
Suite 130
Islandia 11722
Tel: (800) 628-8686
TWX: 510-227-6236
FAX (516) 348-7939

OHIO

*Intel Corp
56 Milford Dr., Suite 205
Hudson 44236
Tel: (800) 628-8686
FAX (216) 528-1026

*†Intel Corp
3401 Park Center Drive
Suite 220
Dayton 45414
Tel: (800) 628-8686
TWX: 810-450-2528
FAX: (513) 890-8658

OKLAHOMA

Intel Corp.
6801 N. Broadway
Suite 115
Oklahoma City 73162
Tel: (800) 628-8686
FAX (405) 840-9819

OREGON

†Intel Corp.
15254 NW Greenbrier Pkwy
Building B
Beaverton 97006
Tel: (800) 628-8686
TWX: 910-467-8741
FAX: (503) 645-8181

PENNSYLVANIA

*†Intel Corp.
925 Harvest Drive
Suite 200
Blue Bell 19422
Tel: (800) 628-8686
FAX: (215) 641-0785

SOUTH CAROLINA

Intel Corp.
7403 Parklane Rd., Suite 4
Columbia 29223
Tel (800) 628-8686
FAX: (803) 788-7999

Intel Corp.
100 Executive Center Drive
Suite 109, B183
Greenville 29615
Tel. (800) 628-8686
FAX (803) 297-3401

TEXAS

†Intel Corp
8911 N. Capital of Texas Hwy.
Suite 4230
Austin 78759
Tel. (800) 628-8686
FAX: (512) 338-9335

*†Intel Corp.
5000 Quorum Drive
Suite 750
Dallas 75240
Tel. (800) 628-8686
FAX: (214) 233-1325

*†Intel Corp.
20515 SH 249
Suite 401
Houston 77070
Tel: (800) 628-8686
TWX: 910-881-2490
FAX: (713) 376-2891

UTAH

†Intel Corp
428 East 6400 South
Suite 135
Murray 84107
Tel: (800) 628-8686
FAX: (801) 268-1457

Intel Corp.
2581 E. Cobblestone Way
Sandy, UT 84093
Tel: (801) 942-8820
FAX: (801) 942-8815

WASHINGTON

†Intel Corp.
2800 156th Avenue SE
Suite 105
Bellevue 98007
Tel: (800) 628-8686
FAX: (206) 746-4495

WISCONSIN

Intel Corp.
400 N. Executive Dr.
Suite 401
Brookfield 53005
Tel: (800) 628-8686
FAX: (414) 789-2746

CANADA

BRITISH COLUMBIA

Intel Semiconductor of
Canada, Ltd.
999 Canada Place
Suite 404, #11
Vancouver V6C 3E2
Tel: (800) 628-8686
FAX: (604) 844-2813

ONTARIO

†Intel Semiconductor of
Canada, Ltd.
2650 Queensview Drive
Suite 250
Ottawa K2B 8H6
Tel: (800) 628-8686
FAX: (613) 820-5936

†Intel Semiconductor of
Canada, Ltd.
190 Attwell Drive
Suite 500
Rexdale M9W 6H8
Tel: (800) 628-8686
FAX: (416) 675-2438

QUEBEC

†Intel Semiconductor of
Canada, Ltd.
1 Rue Holiday, Tour West
Suite 320
Pt. Claire H9R 5N3
Tel: (800) 628-8686
FAX: 514-694-0064

†Sales and Service Office
*Field Application Location

NORTH AMERICAN DISTRIBUTORS

ALABAMA

Arrow/Schweber Electronics
1015 Henderson Road
Huntsville 35806
Tel. (205) 837-6955
FAX. (205) 721-1581

Hamilton Hallmark
4890 University Square, #1
Huntsville 35816
Tel: (205) 837-8700
FAX (205) 830-2565

MTI Systems
4950 Corporate Dr , #120
Huntsville 35805
Tel: (205) 830-9526
FAX (205) 830-9557

Pioneer Technologies Group
4835 University Square, #5
Huntsville 35805
Tel. (205) 837-9300
FAX (205) 837-9358

Wyle Laboratories
7800 Governers Drive
Tower Building, 2nd Floor
Huntsville 35806
Tel (205) 830-1119
FAX (205) 830-1520

ARIZONA

Anthem Electronics
1555 W 10th Place, #101
Tempe 85281
Tel: (602) 966-6600
FAX (602) 966-4826

Arrow/Schweber Electronics
2415 W Erie Drive
Tempe 85282
Tel (602) 431-0030
FAX: (602) 252-9109

Avnet Computer
1626 S Edwards Drive
Tempe 85281
Tel (602) 902-4600
FAX (602) 902-4640

Hamilton Hallmark
4637 S. 36th Place
Phoenix 85040
Tel: (602) 437-1200
FAX: (602) 437-2348

Wyle Laboratories
4141 E Raymond
Phoenix 85040
Tel (602) 437-2088
FAX: (602) 437-2124

CALIFORNIA

Anthem Electronics
9131 Oakdale Avenue
Chatsworth 91311
Tel (818) 775-1333
FAX (818) 775-1302

Anthem Electronics
1 Oldfield Drive
Irvine 92718-2809
Tel (714) 768-4444
FAX (714) 768-6456

Anthem Electronics
580 Menlo Drive, #8
Rocklin 95677
Tel (916) 624-9744
FAX (916) 624-9750

Anthem Electronics
9369 Carroll Park Drive
San Diego 92121
Tel (619) 453-9005
FAX (619) 546-7893

Anthem Electronics
1160 Ridder Park Drive
San Jose 95131
Tel (408) 452-2219
FAX (408) 441-4504

Arrow Commercial Systems Group
1502 Crocker Avenue
Hayward 94544
Tel (510) 489-5371
FAX (510) 489-9393

Arrow Commercial Systems Group
14242 Chambers Road
Tustin 92680
Tel (714) 544-0200
FAX (714) 731-8438

Arrow/Schweber Electronics
26707 W. Agoura Road
Calabasas 91302
Tel: (818) 880-9686
FAX (818) 772-8930

Arrow/Schweber Electronics
48834 Kato Rd , Suite 103
Fremont 94538
Tel: (510) 490-9477

Arrow/Schweber Electronics
6 Cromwell, #100
Irvine 92718
Tel: (714) 838-5422
FAX (714) 454-4206

Arrow/Schweber Electronics
9511 Ridgehaven Court
San Diego 92123
Tel (619) 565-4800
FAX (619) 279-8062

Arrow/Schweber Electronics
1180 Murphy Avenue
San Jose 95131
Tel (408) 441-9700
FAX (408) 453-4810

Avnet Computer
3170 Pullman Street
Costa Mesa 92626
Tel (714) 641-4150
FAX (714) 641-4170

Avnet Computer
1361B West 190th Street
Gardena 90248
Tel. (800) 426-7999
FAX (310) 327-5389

Avnet Computer
755 Sunrise Blvd, #150
Roseville 95661
Tel (916) 781-2521
FAX (916) 781-3819

Avnet Computer
1175 Bordeaux Drive, #A
Sunnyvale 94089
Tel (408) 743-3454
FAX: (408) 743-3348

Avnet Computer
21150 Califa Street
Woodland Hills 91376
Tel (818) 594-8301
FAX (818) 594-8333

Hamilton Hallmark
3170 Pullman Street
Costa Mesa 92626
Tel: (714) 641-4100
FAX (714) 641-4122

Hamilton Hallmark
1175 Bordeaux Drive, #A
Sunnyvale 94089
Tel: (408) 435-3500
FAX (408) 745-6679

Hamilton Hallmark
4545 Viewridge Avenue
San Diego 92123
Tel (619) 571-7540
FAX (619) 277-6136

Hamilton Hallmark
21150 Califa St
Woodland Hills 91367
Tel (818) 594-0404
FAX (818) 594-8234

Hamilton Hallmark
580 Menlo Drive, #2
Rocklin 95762
Tel (916) 624-9781
FAX (916) 961-0922

Pioneer Standard
5850 Canoga Blvd , #400
Woodland Hills 91367
Tel (818) 883-4640

Pioneer Standard
217 Technology Dr , #110
Irvine 92718
Tel (714) 753-5090

Pioneer Technologies Group
134 Rio Robles
San Jose 95134
Tel (408) 954-9100
FAX (408) 954-9113

Wyle Laboratories
15370 Barranca Pkwy
Irvine 92713
Tel (714) 753-9953
FAX (714) 753-9877

Wyle Laboratories
15360 Barranca Pkwy, #200
Irvine 92713
Tel. (714) 753-9953
FAX (714) 753-9877

Wyle Laboratories
2951 Sunrise Blvd , #175
Rancho Cordova 95742
Tel. (916) 638-5282
FAX: (916) 638-1491

Wyle Laboratories
9525 Chesapeake Drive
San Diego 92123
Tel: (619) 565-9171
FAX. (619) 365-0512

Wyle Laboratories
3000 Bowers Avenue
Santa Clara 95051
Tel. (408) 727-2500
FAX (408) 727-5896

Wyle Laboratories
17872 Cowan Avenue
Irvine 92714
Tel: (714) 863-9953
FAX: (714) 263-0473

Wyle Laboratories
26010 Mureau Road, #150
Calabasas 91302
Tel: (818) 880-9000
FAX (818) 880-5510

Zeus Arrow Electronics
6276 San Ignacio Ave , #E
San Jose 95119
Tel. (408) 629-4789
FAX (408) 629-4792

Zeus Arrow Electronics
22700 Savi Ranch Pkwy.
Yorba Linda 92687-4613
Tel (714) 921-9000
FAX (714) 921-2715

COLORADO

Anthem Electronics
373 Inverness Drive South
Englewood 80112
Tel (303) 790-4500
FAX (303) 790-4532

Arrow/Schweber Electronics
61 Inverness Dr. East, #105
Englewood 80112
Tel (303) 799-0258
FAX: (303) 373-5760

Hamilton Hallmark
12503 E. Euclid Drive, #20
Englewood 80111
Tel. (303) 790-1662
FAX (303) 790-4991

Hamilton Hallmark
710 Wooten Road, #102
Colorado Springs 80915
Tel (719) 637-0055
FAX (719) 637-0088

Wyle Laboratories
451 E 124th Avenue
Thornton 80241
Tel (303) 457-9953
FAX: (303) 457-4831

CONNECTICUT

Anthem Electronics
61 Mattatuck Heights Road
Waterburg 06705
Tel (203) 575-1575
FAX (203) 596-3232

Arrow/Schweber Electronics
12 Beaumont Road
Wallingford 06492
Tel (203) 265-7741
FAX. (203) 265-7988

Avnet Computer
55 Federal Road, #103
Danbury 06810
Tel (203) 797-2880
FAX (203) 791-9050

Hamilton Hallmark
125 Commerce Court, Unit 6
Cheshire 06410
Tel: (203) 271-2844
FAX. (203) 272-1704

Pioneer Standard
2 Trap Falls Road
Shelton 06484
Tel. (203) 929-5600

FLORIDA

Anthem Electronics
598 South Northlake Blvd , #1024
Altamonte Springs 32701
Tel: (813) 797-2900
FAX: (813) 796-4880

Arrow/Schweber Electronics
400 Fairway Drive, #102
Deerfield Beach 33441
Tel: (305) 429-8200
FAX. (305) 428-3991

Arrow/Schweber Electronics
37 Skyline Drive, #3101
Lake Mary 32746
Tel: (407) 333-9300
FAX (407) 333-9320

Avnet Computer
3343 W. Commercial Boulevard
Bldg C/D, Suite 107
Ft. Lauderdale 33309
Tel (305) 730-9110
FAX: (305) 730-0368

Avnet Computer
3247 Tech Drive North
St Petersburg 33716
Tel. (813) 573-5524
FAX (813) 572-4324

Hamilton Hallmark
3350 N.W. 53rd St., #105-107
Ft. Lauderdale 33309
Tel (305) 484-5482
FAX. (305) 484-2995

Hamilton/Avnet
10491 72nd St North
Largo 34647
Tel (813) 541-7440
FAX (813) 544-4394

Hamilton/Avnet
7079 University Boulevard
Winter Park 32792
Tel (407) 657-3300
FAX (407) 678-4414

Pioneer Technologies Group
337 Northlake Blvd , #1000
Alta Monte Springs 32701
Tel (407) 834-9090
FAX. (407) 834-0865

Pioneer Technologies Group
674 S Military Trail
Deerfield Beach 33442
Tel. (305) 428-8877
FAX: (305) 481-2950

Pioneer Technologies Group
8031-2 Phillips Highway
Jacksonville 32256
Tel (904) 730-0065

Wyle Laboratories
1000 112 Circle North
St Petersburg 33716
Tel (813) 530-3400
FAX (813) 579-1518

GEORGIA

Arrow Commercial Systems Group
3400 C. Corporate Way
Duluth 30136
Tel. (404) 623-8825
FAX (404) 623-8802

Arrow/Schweber Electronics
4250 E Rivergreen Pkwy., #E
Duluth 30136
Tel (404) 497-1300
FAX (404) 476-1493

Avnet Computer
3425 Corporate Way, #G
Duluth 30136
Tel (404) 623-5452
FAX (404) 476-0125

NORTH AMERICAN DISTRIBUTORS (Contd.)

Hamilton Hallmark
3425 Corporate Way, #G & #A
Duluth 30136
Tel. (404) 623-5475
FAX: (404) 623-5490

Pioneer Technologies Group
4250 C Rivergreen Parkway
Duluth 30136
Tel: (404) 623-1003
FAX: (404) 623-0665

Wyle Laboratories
6025 The Corners Pkwy , #111
Norcross 30092
Tel. (404) 441-9045
FAX. (404) 441-9086

ILLINOIS

Anthem Electronics
1300 Remington Road, Suite A
Schaumberg 60173
Tel (708) 884-0200
FAX: (708) 885-0480

Arrow/Schweber Electronics
1140 W. Thorndale Rd.
Itasca 60143
Tel: (708) 250-0500

Avnet Computer
1124 Thorndale Avenue
Bensenville 60106
Tel: (708) 860-8572
FAX: (708) 773-7976

Hamilton/Avnet
1130 Thorndale Avenue
Bensenville 60106
Tel: (708) 860-7780
FAX: (708) 860-8530

MTI Systems
1140 W. Thorndale Avenue
Itasca 60143
Tel: (708) 250-8222
FAX: (708) 250-8275

Pioneer-Standard
2171 Executive Dr., #200
Addison 60101
Tel: (708) 495-9680
FAX: (708) 495-9831

Wyle Laboratories
2055 Army Trail Road, #140
Addison 60101
Tel: (800) 853-9953
FAX: (708) 620-1610

INDIANA

Arrow/Schweber Electronics
7108 Lakeview Parkway West Dr.
Indianapolis 46268
Tel: (317) 299-2071
FAX: (317) 299-2379

Avnet Computer
485 Gradle Drive
Carmel 46032
Tel: (317) 575-8029
FAX: (317) 844-4964

Hamilton Hallmark
4275 W. 96th
Indianapolis 46268
Tel: (317) 872-8875
FAX (317) 876-7165

Pioneer-Standard
9350 Priority Way West Dr
Indianapolis 46250
Tel: (317) 573-0880
FAX: (317) 573-0979

KANSAS

Arrow/Schweber Electronics
9801 Legler Road
Lenexa 66219
Tel. (913) 541-9542
FAX. (913) 541-0328

Avnet Computer
15313 W. 95th Street
Lenexa 61219
Tel. (913) 541-7989
FAX: (913) 541-7904

Hamilton Hallmark
10809 Lakeview Avenue
Lenexa 66215
Tel: (913) 888-4747
FAX: (913) 888-0523

KENTUCKY

Hamilton Hallmark
1847 Mercer Rd , #G
Lexington 40511
Tel: (800) 235-6039
FAX: (606) 288-4936

MARYLAND

Anthem Electronics
7168A Columbia Gateway Drive
Columbia 21046
Tel: (410) 995-6640
FAX: (410) 290-9862

Arrow Commercial Systems Group
200 Perry Parkway
Gaithersburg 20877
Tel. (301) 670-1600
FAX: (301) 670-0188

Arrow/Schweber Electronics
9800J Patuxent Woods Dr
Columbia 21046
Tel: (301) 596-7800
FAX. (301) 995-6201

Avnet Computer
7172 Columbia Gateway Dr , #G
Columbia 21045
Tel: (301) 995-3571
FAX. (301) 995-3515

Hamilton Hallmark
10240 Old Columbia Road
Columbia 21046
Tel: (410) 988-9800
FAX: (410) 381-2036

North Atlantic Industries
Systems Division
7125 River Wood Dr.
Columbia 21046
Tel (301) 312-5800
FAX: (301) 312-5850

Pioneer Technologies Group
9100 Gather Road
Gaithersburg 20877
Tel: (301) 921-0660
FAX. (301) 670-6746

Wyle Laboratories
7180 Columbia Gateway Dr
Columbia 21046
Tel: (410) 312-4844
FAX: (410) 312-4953

MASSACHUSETTS

Anthem Electronics
36 Jonspin Road
Wilmington 01887
Tel (508) 657-5170
FAX: (508) 657-6008

Arrow/Schweber Electronics
25 Upton Drive
Wilmington 01887
Tel (508) 658-0900
FAX: (508) 694-1754

Avnet Computer
10 D Centennial Drive
Peabody 01960
Tel: (508) 532-9886
FAX: (508) 532-9660

Hamilton Hallmark
10 D Centennial Drive
Peabody 01960
Tel. (508) 531-7430
FAX: (508) 532-9802

Pioneer Standard
44 Hartwell Avenue
Lexington 02173
Tel. (617) 861-9200
FAX (617) 863-1547

Wyle Laboratories
15 Third Avenue
Burlington 01803
Tel: (617) 272-7300
FAX: (617) 272-6809

MICHIGAN

Arrow/Schweber Electronics
19880 Haggerty Road
Livonia 48152
Tel (800) 231-7902
FAX. (313) 462-2686

Avnet Computer
2876 28th Street, S.W., #5
Grandville 49418
Tel (616) 531-9607
FAX. (616) 531-0059

Avnet Computer
41650 Garden Brook Rd. #120
Novi 48375
Tel. (313) 347-1820
FAX (313) 347-4067

Hamilton Hallmark
44191 Plymouth Oaks Blvd , #1300
Plymouth 48170
Tel: (313) 416-5800
FAX: (313) 416-5811

Hamilton Hallmark
41650 Garden Brook Rd., #100
Novi 49418
Tel. (313) 347-4271
FAX: (313) 347-4021

Pioneer Standard
4505 Broadmoor S.E.
Grand Rapids 49512
Tel: (616) 698-1800
FAX: (616) 698-1831

Pioneer Standard
13485 Stamford Ct
Livonia 48150
Tel: (313) 525-1800
FAX. (313) 427-3720

MINNESOTA

Anthem Electronics
7646 Golden Triangle Drive
Eden Prairie 55344
Tel: (612) 944-5454
FAX: (612) 944-3045

Arrow/Schweber Electronics
10100 Viking Drive, #100
Eden Prairie 55344
Tel: (612) 941-5280
FAX: (612) 942-7803

Avnet Computer
10000 West 76th Street
Eden Prairie 55344
Tel: (612) 829-0025
FAX. (612) 944-2781

Hamilton Hallmark
9401 James Ave. South, #140
Bloomington 55431
Tel: (612) 881-2600
FAX: (612) 881-9461

Pioneer Standard
7625 Golden Triangle Dr., #G
Eden Prairie 55344
Tel: (612) 944-3355
FAX. (612) 944-3794

Wyle Laboratories
1325 E. 79th Street, #1
Bloomington 55425
Tel: (612) 853-2280
FAX: (612) 853-2298

MISSOURI

Arrow/Schweber Electronics
2380 Schuetz Road
St Louis 63141
Tel. (314) 567-6888
FAX. (314) 567-1164

Avnet Computer
741 Goddard Avenue
Chesterfield 63005
Tel: (314) 537-2725
FAX: (314) 537-4248

Hamilton Hallmark
3783 Rider Trail South
Earth City 63045
Tel: (314) 291-5350
FAX. (314) 291-0362

NEW HAMPSHIRE

Avnet Computer
2 Executive Park Drive
Bedford 03102
Tel: (800) 442-8638
FAX: (603) 624-2402

NEW JERSEY

Anthem Electronics
26 Chapin Road, Unit K
Pine Brook 07058
Tel: (201) 227-7960
FAX. (201) 227-9246

Arrow/Schweber Electronics
4 East Stow Rd., Unit 11
Marlton 08053
Tel: (609) 596-8000
FAX. (609) 596-9632

Arrow/Schweber Electronics
43 Route 46 East
Pine Brook 07058
Tel: (201) 227-7880
FAX. (201) 538-4962

Avnet Computer
1-B Keystone Ave , Bldg. 36
Cherry Hill 08003
Tel: (609) 424-8961
FAX: (609) 751-2502

Hamilton Hallmark
1 Keystone Ave., Bldg. 36
Cherry Hill 08003
Tel. (609) 424-0110
FAX: (609) 751-2552

Hamilton Hallmark
10 Lanidex Plaza West
Parsippany 07054
Tel: (201) 515-5300
FAX. (201) 515-1601

MTI Systems
43 Route 46 East
Pinebrook 07058
Tel: (201) 882-8780
FAX: (201) 539-6430

Anthem Electronics
19017 - 120th Ave., N.E #102
Bothell 98011
Tel: (206) 483-1700
FAX. (206) 486-0571

Avnet Computer
17761 N.E 78th Place
Redmond 98052
Tel. (206) 867-0160
FAX: (206) 867-0161

Hamilton Hallmark
8630 154th Avenue
Redmond 98052
Tel: (206) 881-6697
FAX: (206) 867-0159

Wyle Laboratories
15385 N.E. 90th Street
Redmond 98052
Tel: (206) 881-1150
FAX: (206) 881-1567

PioneerStandard
14-A Madison Rd.
Fairfield 07006
Tel: (201) 575-3510
FAX: (201) 575-3454

Wyle Laboratories
20 Chapin Road, Bldg 10-13
Pinebrook 07058
Tel. (201) 882-8358
FAX: (201) 882-9109

NEW MEXICO

Alliance Electronics, Inc.
10510 Research Ave.
Albuquerque 87123
Tel: (505) 292-3360
FAX: (505) 275-6392

Avnet Computer
7801 Academy Rd.
Bldg. 1, Suite 204
Albuquerque 87109
Tel: (505) 828-9725
FAX: (505) 828-0360

NEW YORK

Anthem Electronics
47 Mall Drive
Commack 11725
Tel. (516) 864-6600
FAX: (516) 493-2244

NORTH AMERICAN DISTRIBUTORS (Contd.)

Arrow/Schweber Electronics
3375 Brighton Henrietta
Townline Rd.
Rochester 14623
Tel (716) 427-0300
FAX (716) 427-0735

Arrow/Schweber Electronics
20 Oser Avenue
Hauppauge 11788
Tel (516) 231-1000
FAX (516) 231-1072

Avnet Computer
933 Motor Parkway
Hauppauge 11788
Tel (516) 434-7443
FAX (516) 434-7426

Avnet Computer
2060 Townline Rd.
Rochester 14623
Tel (716) 272-9110
FAX (716) 272-9685

Hamilton/Avnet
933 Motor Parkway
Hauppauge 11788
Tel (516) 434-7470
FAX (516) 434-7491

Hamilton Hallmark
1057 E. Henrietta Road
Rochester 14623
Tel (716) 475-9130
FAX (716) 475-9119

Hamilton Hallmark
3075 Veterans Memorial Hwy
Ronkonkoma 11779
Tel (516) 737-0600
FAX (516) 737-0838

MTI Systems
1 Penn Plaza
250 W. 34th Street
New York 10119
Tel (212) 643-1280
FAX (212) 643-1288

Pioneer Standard
68 Corporate Drive
Binghamton 13904
Tel (607) 722-9300
FAX (607) 722-9562

Pioneer Standard
60 Crossway Park West
Woodbury, Long Island 11797
Tel (516) 921-8700
FAX (516) 921-2143

Pioneer Standard
840 Fairport Park
Fairport 14450
Tel (716) 381-7070
FAX (716) 381-5955

Zeus Arrow Electronics
100 Midland Avenue
Port Chester 10573
Tel (914) 937-7400
FAX (914) 937-2553

NORTH CAROLINA

Arrow/Schweber Electronics
5240 Greensdairy Road
Raleigh 27604
Tel (919) 876-3132
FAX (919) 878-9517

Avnet Computer
2725 Millbrook Rd , #123
Raleigh 27604
Tel (919) 790-1735
FAX (919) 872-4972

Hamilton Hallmark
5234 Greensdairy Road
Raleigh 27604
Tel. (919) 878-0819
FAX (919) 878-8729

Pioneer Technologies Group
2200 Gateway Ctr Blvd , #215
Morrisville 27560
Tel (919) 460-1530
FAX (919) 460-1540

OHIO

Arrow Commercial Systems Group
284 Cramer Creek Court
Dublin 43017
Tel (614) 889-9347
FAX (614) 889-9680

Arrow/Schweber Electronics
6573 Cochran Road, #E
Solon 44139
Tel (216) 248-3990
FAX (216) 248-1106

Arrow/Schweber Electronics
8200 Washington Village Dr.
Centerville 45458
Tel (513) 435-5563
FAX (513) 435-2049

Avnet Computer
7764 Washington Village Dr
Dayton 45459
Tel (513) 439-6756
FAX (513) 439-6719

Avnet Computer
30325 Bainbridge Rd , Bldg A
Solon 44139
Tel (216) 349-2505
FAX (216) 349-1894

Hamilton Hallmark
7760 Washington Village Dr
Dayton 45459
Tel (513) 439-6735
FAX (513) 439-6711

Hamilton Hallmark
5821 Harper Road
Solon 44139
Tel (216) 498-1100
FAX (216) 248-4803

Hamilton Hallmark
777 Dearborn Park Lane, #L
Worthington 43085
Tel (614) 888-3313
FAX (614) 888-0767

MTI Systems
23404 Commerce Park Rd.
Beachwood 44122
Tel (216) 464-6688
FAX (216) 464-3564

Pioneer Standard
4433 Interpoint Boulevard
Dayton 45424
Tel (513) 236-9900
FAX (513) 236-8133

Pioneer Standard
4800 E 131st Street
Cleveland 44105
Tel (216) 587-3600
FAX (216) 663-1004

OKLAHOMA

Arrow/Schweber Electronics
12101 East 51st Street, #106
Tulsa 74146
Tel (918) 252-7537
FAX (918) 254-0917

Hamilton Hallmark
5411 S. 125th E. Ave , #305
Tulsa 74146
Tel (918) 254-6110
FAX (918) 254-6207

Pioneer Standard
9717 E 42nd St , #105
Tulsa 74146
Tel (918) 665-7840
FAX (918) 665-1891

OREGON

AlmacArrow Electronics
1885 N W 169th Place
Beaverton 97006
Tel (503) 629-8090
FAX. (503) 645-0611

Anthem Electronics
9090 S W. Gemini Drive
Beaverton 97005
Tel. (503) 643-1114
FAX (503) 626-7928

Avnet Computer
9750 Southwest Nimbus Ave
Beaverton 97005
Tel (503) 627-0900
FAX (503) 526-6242

Hamilton Hallmark
9750 Southwest Nimbus Ave
Beaverton 97005
Tel (503) 526-6200
FAX (503) 641-5939

Wyle Laboratories
9640 Sunshine Court
Bldg. G, Suite 200
Beaverton 97005
Tel (503) 643-7900
FAX (503) 646-5466

PENNSYLVANIA

Anthem Electronics
355 Business Center Dr
Horsham 19044
Tel (215) 443-5150
FAX. (215) 675-9875

Avnet Computer
213 Executive Drive, #320
Mars 16046
Tel (412) 772-1888
FAX (412) 772-1890

Pioneer Technologies Group
259 Kappa Drive
Pittsburgh 15238
Tel (412) 782-2300
FAX (412) 963-8255

Pioneer Technologies Group
500 Enterprise Road
Keith Valley Business Center
Horsham 19044
Tel (215) 530-4700

Wyle Laboratories
1 Eves Drive, #111
Marlton 08053-3185
Tel. (609) 985-7953
FAX (609) 985-8757

TEXAS

Anthem Electronics
651 N. Plano Road, #401
Richardson 75081
Tel. (214) 238-7100
FAX (214) 238-0237

Arrow/Schweber Electronics
11500 Metric Blvd , #160
Austin 78758
Tel (512) 835-4180
FAX. (512) 832-5921

Arrow/Schweber Electronics
3220 Commander Drive
Carrollton 75006
Tel (214) 380-6464
FAX: (214) 248-7208

Arrow/Schweber Electronics
10899 Kinghurst Dr., #100
Houston 77099
Tel (713) 530-4700

Avnet Computer
4004 Beltline, Suite 200
Dallas 75244
Tel (214) 308-8181
FAX (214) 308-8129

Avnet Computer
1235 North Loop West, #525
Houston 77008
Tel (713) 867-8572
FAX (713) 861-6851

Hamilton Hallmark
12211 Technology Blvd
Austin 78727
Tel (512) 258-8848
FAX (512) 258-3777

Hamilton Hallmark
11420 Page Mill Road
Dallas 75234
Tel (214) 553-4300
FAX. (214) 553-4395

Hamilton Hallmark
8000 Westglen
Houston 77063
Tel (713) 781-6100
FAX. (713) 953-8420

Pioneer Standard
1826D Kramer Lane
Austin 78758
Tel (512) 835-4000
FAX (512) 835-9829

Pioneer Standard
13765 Beta Road
Dallas 75244
Tel. (214) 263-3168
FAX (214) 490-6419

Pioneer Standard
10530 Rockley Road, #100
Houston 77099
Tel: (713) 495-4700
FAX (713) 495-5642

Wyle Laboratories
1810 Greenville Avenue
Richardson 75081
Tel (214) 235-9953
FAX (214) 644-5064

Wyle Laboratories
4030 West Braker Lane, #330
Austin 78758
Tel. (512) 345-8853
FAX (512) 345-9330

Wyle Laboratories
11001 South Wilcrest, #100
Houston 77099
Tel (713) 879-9953
FAX (713) 879-6540

UTAH

Anthem Electronics
1279 West 2200 South
Salt Lake City 84119
Tel (801) 973-8555
FAX (801) 973-8909

Arrow/Schweber Electronics
1946 W Parkway Blvd.
Salt Lake City 84119
Tel (801) 973-6913
FAX (801) 972-0200

Avnet Computer
1100 E. 6600 South, #150
Salt Lake City 84121
Tel. (801) 266-1115
FAX (801) 266-0362

Hamilton Hallmark
1100 East 6600 South, #120
Salt Lake City 84121
Tel (801) 266-2022
FAX. (801) 263-0104

Wyle Laboratories
1325 West 2200 South, #E
West Valley 84119
Tel (801) 974-9953
FAX (801) 972-2524

WASHINGTON

AlmacArrow Electronics
14360 S E Eastgate Way
Bellevue 98007
Tel (206) 643-9992
FAX (206) 643-9709

WISCONSIN

Arrow/Schweber Electronics
200 N Patrick, #100
Brookfield 53045
Tel (414) 792-0150
FAX: (414) 792-0156

Avnet Computer
20875 Crossroads Circle, #400
Waukesha 53186
Tel (414) 784-8205
FAX (414) 784-6006

Hamilton Hallmark
2440 S. 179th Street
New Berlin 53146
Tel (414) 797-7844
FAX: (414) 797-9259

Pioneer Standard
120 Bishop Way #163
Brookfield 53005
Tel (414) 784-3480
FAX. (414) 780-3613

Wyle Laboratories
W226 N555 Eastmound Drive
Waukesha 53186
Tel. (414) 521-9333
FAX: (414) 521-9498

ALASKA

Avnet Computer
1400 West Benson Blvd , #400
Anchorage 99503
Tel. (907) 274-9899
FAX (907) 277-2639

*Self Certified Small Business per Federal Acquisition Regulations

NORTH AMERICAN DISTRIBUTORS (Contd.)

CANADA

ALBERTA

Avnet Computer
2816 21st Street, Northeast
Calgary T2E 6Z2
Tel (403) 291-3284
FAX (403) 250-1591

Zentronics
6815 8th Street N.E , #100
Calgary T2E 7H
Tel (403) 295-8838
FAX. (403) 295-8714

BRITISH COLUMBIA

AlmacArrow Electronics
8544 Baxter Place
Burnaby V5A 4T8
Tel (604) 421-2333
FAX (604) 421-5030

Hamilton Hallmark
8610 Commerce Court
Burnaby V5A 4N6
Tel (604) 420-4101
FAX (604) 420-5376

Zentronics
11400 Bridgeport Rd , #108
Richmond V6X 1T2
Tel (604) 273-5575
FAX (604) 273-2413

ONTARIO

Arrow/Schweber Electronics
1093 Meyerside, Unit 2
Mississauga L5T 1M4
Tel. (416) 670-7769
FAX. (416) 670-7781

Arrow/Schweber Electronics
36 Antares Dr , Unit 100
Nepean K2E 7W5
Tel (613) 226-6903
FAX (613) 723-2018

Avnet Computer
Canada System Engineering Group
151 Superior Blvd
Mississuaga L5T 2L1
Tel (416) 795-3835
FAX (416) 677-5091

Avnet Computer
190 Colonade Road
Nepean K2E 7J5
Tel. (613) 727-2000
FAX: (613) 226-1184

Hamilton Hallmark
151 Superior Blvd , Unit 1-6
Mississauga L5T 2L1
Tel (416) 564-6060
FAX (416) 564-6033

Hamilton Hallmark
190 Colonade Road
Nepean K2E 7J5
Tel (613) 226-1700
FAX (613) 226-1184

Zentronics
5600 Keaton Crescent, #1
Mississauga L5R 3S5
Tel (416) 507-2600
FAX (416) 507-2831

Zentronics
155 Colonnade Rd , South #17
Nepean K2E 7K1
Tel. (613) 226-8840
FAX (613) 226-6352

QUEBEC

Arrow/Schweber Electronics
1100 St Regis Blvd.
Dorval H9P 2T5
Tel: (514) 421-7411
FAX: (514) 421-7430

Arrow Schweber Electronics
500 Boul. St.-Jean-Baptiste Ave
Quebec H2E 5R9
Tel (418) 871-7500
FAX: (418) 871-6816

Avnet Computer
2795 Rue Halpern
St Laurent H4S 1P8
Tel: (514) 335-2483
FAX (514) 335-2481

Hamilton Hallmark
7575 Transcanada Highway #600
St Laurent H4T 2V6
Tel (514) 335-1000
FAX: (514) 335-2481

Zentronics
520 McCaffrey Street
St Laurent H4T 1N3
Tel: (514) 737-9700
FAX (514) 737-5212

EUROPEAN SALES OFFICES

FINLAND

Intel Finland OY
Ruosilantie 2
00390 Helsinki
Tel (358) 0 544 644
FAX (358) 0 544 030

FRANCE

Intel Corporation S.A.R L
1, Rue Edison-BP 303
78054 St Quentin-en-Yvelines
Cedex
Tel (33) (1) 30 57 70 00
FAX (33) (1) 30 64 60 32

GERMANY

Intel GmbH
Dornacher Strasse 1
85622 Feldkirchen/Muenchen
Tel (49) 089/90992-0
FAX (49) 089/9043948

ISRAEL

Intel Semiconductor Ltd.
Atidim Industrial Park-Neve Sharet
P O Box 43202
Tel-Aviv 61430
Tel (972) 03 498080
FAX (972) 03 491870

ITALY

Intel Corporation Italia S p A
Milanofiori Palazzo E
20094 Assago
Milano
Tel (39) (2) 575441
FAX (39) (2) 3498464

NETHERLANDS

Intel Semiconductor B.V.
Postbus 84130
3009 CC Rotterdam
Tel (31) 10 407 11 11
FAX. (31) 10 455 4688

RUSSIA

Intel Technologies, Inc
Krementshugskaya 6/7
121357 Moscow
Tel 007-095-4439785
FAX: 007-095-4459420
TLX 612092 smail su.

SPAIN

Intel Iberia S A.
Zubaran, 28
28010 Madrid
Tel (34) (1) 308 2552
FAX. (34) (1) 410 7570

SWEDEN

Intel Sweden A.B
Dalvagen 24
171 36 Solna
Tel: (46) 8 705 5600
FAX: (46) 8 278085

UNITED KINGDOM

Intel Corporation (U.K.) Ltd.
Pipers Way
Swindon, Wiltshire SN3 1RJ
Tel: (44) (0793) 696000
FAX (44) (0793) 641440

EUROPEAN DISTRIBUTORS/REPRESENTATIVES

AUSTRIA

†*Elbatex GmbH
Erbergasse 6
A-1231 Wien
Tel (43) 1 816 020
FAX (43) 1 816 02400

†*Spoerle Elektronik
Heiligenstadter Str. 52
A-1190 Wien
Tel (43) 1 363 6500
FAX (43) 1 369 2273

BELGIUM

†*Inelco
Avenue des Croix de Guerre 94
1120 Bruxelles
Tel (32) 2 244 2811
FAX (32) 2 216 3304

*Diode
Keiberg 2
Minervastraat, 14/B2
1930 Zaventem
Tel (32) 2 725 4660
FAX (32) 2 725 4511

CHS
Budasteenweg 2
1830 Machelen
Tel. (32) 2 255 0700
FAX (32) 2 252 5900

CZECH REPUBLIC

Elbatex
Prechodni 11
CS-14000 Praha 4
Tel (42) 2 692 8087
FAX. (42) 2 471 8203

DENMARK

*Avnet Nortec A/S
Transformervej 17
DK-2730 Herlev
Tel (45) 4284 2000
FAX (45) 4492 1552

†*Farnell Electronic Services AS
Naverland 29
DK-2600 Glostrup
Tel (45) 5254 6645
FAX. (45) 4245 7624

ESTONIA

*Avnet Baltronic AS
Akadeemia tee 21F, EE0026
Tallinn
Tel (327) 2 527 349
FAX. (372) 2 527 556

FINLAND

†*Computer 2000
Pynyntitie 3
P O Box 44
SF-02231 Espoo
Tel. (358) 0 887 331
FAX (358) 0 887 333 43

Avnet Nortec OY
Italahdenkatu 22
SF-00210 Helsinki
Tel (358) 0 670 277
FAX (358) 0 692 2326

Farnell Electronic Services O.Y
PL. 25
Tyopajakatu 5
SF-00581 Helsinki
Tel (358) 0 793 100
FAX (358) 0 701 9892

FRANCE

*Arrow Electronique
73-79 Rue des Solets
Silic 585
94663 Rungis Cedex
Tel (33) 1 4978 4978
FAX. (33) 1 4978 0596

*Avnet EMG SA
79, Rue Pierre Semard
92310 Chatillon
Tel. (33) 1 4965 2500
FAX (33) 1 4965 2769

†Metrologie
Tour d'Asnieres
4, Avenue Laurent Cely
92606 Asnieres Cedex
Tel (33) 1 4080 9000
FAX (33) 1 4791 0561

*Tekelec
Cite des Bruyeres
5, Rue Carle Vernet-BP2
92310 Sevres
Tel (33) 1 4623 2425
FAX (33) 1 4507 2191

EUROPEAN DISTRIBUTORS/REPRESENTATIVES (Cont'd)

Inelco
135 Avenue Louis Roche
92621 Gennevilliers
Tel: (33) 1 4794 7680
FAX: (33) 1 4792 3468

CHS
11 Rue de Cambrai
Bat. 28
75019 Paris
Tel: (33) 1 4005 2800
FAX: (33) 1 4034 3734

GERMANY

*Avnet E2000
Stahlgruberring 12
81829 Munchen
Tel: (49) 89 451 1001
FAX: (49) 89 45110 129

*Jermyn GmbH
Im Dachsstuck 9
65549 Limburg
Tel: (49) 6431 5080
FAX: (49) 6431 5082 89

†Metrologie GmbH
Steinerstrasse 15
81369 Munchen
Tel: (49) 89 742 170
FAX: (49) 89 7421 7111

*Proelectron Vertriebs GmbH
Max-Planck Strasse 1-3
63303 Dreieich
Tel: (49) 6103 3043 43
FAX (49) 6103 3044 25

†Raab Karcher Elektronik GmbH
Loetscher Weg 66
41334 Nettetal
Tel: (49) 2153 7330
FAX: (49) 2153 7335 13

Itesa GmbH
Beethoven Strasse 26
63526 Erlensee
Tel: (49) 6183 830
FAX: (49) 6183 8330

GREECE

†Ergodata
Aigiroupoleos 2a
176 76 Kalithea
Tel: (30) 1 951 0922
FAX: (30) 1 959 3160

*Pouliadis Associates Corp
Aristotelous St. 3
Sygrou Av. 150
17671 Athens
Tel: (30) 1 924 2072
FAX: (30) 1 924 1066

HUNGARY

Elbatex
Gabor Takacs
Vaci u. 202
H-1138 Budapest
Tel: (36) 1 140 9194
FAX: (36) 1 120 9478

IRELAND

†*Micro Marketing
Taney Hall - Eglinton Terrace
Dundrum
Dublin 14
Tel: (353) 1 298 9400
FAX: (353) 1 298 9828

Arrow Electronics
Unit 7 - Newland Business Park
Nass Road - Condalkin
Dublin 22
Tel (353) 1 627 1949
FAX. (353) 1 459 5490

ISRAEL

†*Eastronics Limited
Rozanis 11 - P.O B. 39300
Tel Baruch
Tel-Aviv 61392
Tel. (972) 3 6458 777
FAX: (972) 3 6458 666

ITALY

Avnet Adelsy SRL
Via Novara, 570
20100 Milano
Tel: (39) 2 38 103 100
FAX: (39) 2 38 002 988

Farnell Electronic Services SpA
Viale Milanofiori E/5
I-20090 Assago
Tel: (39) 2 824 701
FAX. (30) 2 824 2631

*Lasi Elettronica
P I. 00839000155
Viale Fulvio Testi, N.280
20126 Milano
Tel: (39) 2 661 431
FAX: (39) 2 6610 1385

Telcom
Via Lorenteggio 270/A
20152 Milano
Tel: (39) 2 4830 2640
FAX: (39) 2 4830 2010

Lifeboat
Via Galileo Ferraris 2
20147 Soronno
Tel: (39) 2 9670 1592
FAX (39) 2 9670 3113

LATVIA

Avnet Baltronic
Maskavas iela 40/42
New Bldg - Room 513
LV 1018 Riga
Tel: (371) 2 211 109
FAX (371) 2 211 109

NETHERLANDS

†Datelcom B.V.
Meidoornkadae 22
3993 AE Houten
Tel: (31) 3403 57222
FAX: (31) 3403 57220

*Diode Components
Coltbaan 17
3439 NG Nieuwegein
Tel: (31) 3402 9 1234
FAX. (31) 3402 3 5924

†*Koning en Hartman
Energieweg 1
2627 AP Delft
Tel. (31) 15 609 906
FAX: (31) 15 619 194

NORWAY

*Avnet Nortec A/S
Postboxs 123
N-1364 Hvalstad
Tel (47) 66 846 210
FAX. (47) 66 846 545

†Computer System Integrated A/S
Postbox 198
Hvamsvingen 24
N-2013 Skjetten
Tel. (47) 63 845 411
FAX: (47) 63 845 310

Farnell Electronic Services A/S
Postbox 120 Furuset
Karihaugvei 89
N-1001 Oslo
Tel (47) 22 321 270
FAX. (47) 66 846 545

POLAND

Elbatex
ul. Hoza 29/31 M6
PL-00-681 Warszawa
Tel: (48) 2 623 0602-09
FAX: (48) 2 623 0605

PORTUGAL

*Arrow/ATD Electronica LDA
Quinta Grande, Lote 20 - R/C DTO
2700 Amadora
Tel: (351) 1 471 4182
FAX. (351) 1 471 5886

RUSSIA

Merisel
3 Kroutitskiy Val St.
Section 2
109044 Moscow
Tel: (7) 095 276 4718
FAX: (7) 095 276 4714

SAUDI ARABIA

Hoshanco
Airport Road - PO Box 382
Riyadh 11411
Tel: (966) 1 477 2323
FAX (966) 1 479 2588

SLOVAKIA

Elbatex
Topol Cianska 23
SK-85105 Bratislava
Tel: (42) 7 831 320
FAX: (42) 7 831 320

SLOVENIA

Elbatex
Stegna 19
SLO-61117 Lubljana
Tel: (30) 61 191 126
FAX: (30) 61 192 398

SOUTH AFRICA

†*EBE
178 Erasmus Street
Meyerspark
Pretoria 0184
Tel: (27) 12 803 7680
FAX: (27) 12 803 8294

SPAIN

*Arrow/ATD Electronica
Albasanz 75-3
28037 Madrid
Tel. (34) 1 304 1534
FAX. (34) 1 327 2778

†Metrologia
Avda. Industria, 32-2
28100 Alcobendas
Madrid
Tel: (34) 1 661 1142
FAX: (34) 1 661 5755

Diode
c/Orense, 34
28080 Madrid
Tel: (34) 11 555 3686
FAX (34) 1 556 7159

SWEDEN

†Avnet Computer AB
Box 1392
Englundavagen 7
S-171 41 Solna
Tel: (46) 8 629 1400
FAX: (46) 8 627 5165

*Avnet Nortec AB
Box 1830
S-171 27 Solna
Tel: (46) 8 705 1800
FAX: (46) 8 836 918

*Farnell Electronic Services AB
Ankdammsgatan 32
Box 1330
S-171 26 Solna
Tel (46) 8 830 020
FAX. (46) 8 825 770

SWITZERLAND

†Elbatex AG
Hardstrasse 72
CH-5430 Wettingen
Tel: (41) 56 275 000
FAX: (41) 56 271 240

†Fabrimex AG
Kirchenweg 5
CH-8032 Zurich
Tel: (41) 1 386 8686
FAX (41) 1 383 2379

†IMITEC
Zurichstrasse
CH-8185 Winkel-Ruti
Tel (41) 1 862 0055
FAX. (41) 1 862 0266

†*Industrade AG
Hertistrasse 31
CH-8304 Wallisellen
Tel: (41) 1 832 8111
FAX: (41) 1 830 7550

TURKEY

*Empa Electronic
Besyol Mah E-5 Karayolu Yanı
Florya is Merkezi
No. 5 34630 Sefakoy
Istanbul
Tel (90) 212 599 3050
FAX: (90) 212 598 5353

Info
Buyukdere Cad. 107/3
Bangun Han Gayerttepe
80300 Istanbul
Tel: (902) 1 2 275 0780
FAX (902) 1 2 272 3427

UNITED KINGDOM

*Arrow/MMD
3 Bennet Court
Bennet Road
Reading RG2 OQX
Tel: (44) 734 313 232
FAX: (44) 734 313 255

*Avnet Access
Jubilee House
Jubilee Road
Letchworth SG6 1QH
Tel: (44) 462 488 500
FAX. (44) 462 488 567

†Bytech Systems
5 The Sterling Centre
Eastern Road
Bracknell RG12 2PW
Tel. (44) 344 55 333
FAX (44) 344 867 270

Bytech Electronics
12a Cedarwood
Chineham Park - Crockford Lane
Basingstoke RG12 1RW
Tel. (44) 256 707 107
FAX: (44) 256 707 162

*Datrontech PLC
42-44 Birchett Road
Aldershot
Hants GU11 1LU
Tel. (44) 252 313 155
FAX: (44) 252 341 939

†Metrologie UK Ltd.
Metrologie House
Oxford Road
High Wycombe HP11 2EE
Tel: (44) 494 526 271
FAX: (44) 494 521 860

CHS
Copse Road, St Johns, Woking
Surrey GU21 1SX
Tel: (44) 483 723 411
FAX (44) 483 729 974

UKRAINE

Kvasar Micro
Popudrenko Str. 52
253094 Kiev
Tel: (7) 044 516 8496
FAX: (7) 044 516 8608

UNITED ARAB EMIRATES

Graytech
P.O. Box 50718
Dubai
Tel: (971) 434 6952
FAX (971) 434 6546

Jumbo Electronics Co. Ltd.
Al Wahaibi Bldg.
Al Garhoud - P.O. Box 3426
Dubai
Tel. (971) 4 882 4888
FAX: (971) 4 821 839

* Technical Distributor
† VAD

intel®

INTERNATIONAL SALES OFFICES

AUSTRALIA

Intel Australia Pty Ltd
Unit 13
Allambie Grove Business Park
25 Frenchs Forest Road East
Frenchs Forest, NSW, 2086
Sydney
Tel: 61-2-975-3300
FAX: 61-2-975-3375

Intel Australia Pty. Ltd
711 High Street
1st Floor
East Kw. Vic., 3102
Melbourne
Tel: 61-3-810-2141
FAX: 61-3-819 7200

BRAZIL

Intel Semiconductores do Brasil
Rua Florida, 1703-2 and CJ 22
CEP 04565-001 Sao Paulo, SP
Tel: 55-11-530-2296
FAX. 55-11-531-5765

CHINA/HONG KONG

Intel PRC Corporation
Room 517-518
China World Tower
1 Jian Guo Men Wai Avenue
Beijing, 100004
Republic of China
Tel 861-505-0386
FAX 861-505-0383

Intel Semiconductor Ltd.*
32/F Two Pacific Place
88 Queensway
Central
Hong Kong
Tel: (852) 844-4555
FAX. (852) 868-1989

INDIA

Intel Asia Electronics, Inc
4/2, Samrah Plaza
St Mark's Road
Bangalore 560001
Tel. 91-80-215065
FAX 91-80-215067
TLX 953-845-2646 INTEL IN

JAPAN

Intel Japan K K
5-6 Tokodai, Tsukuba-shi
Ibaraki, 300-26
Tel: 0298-47-8511
FAX: 0298-47-8450

Intel Japan K K *
Hachioji ON Bldg
4-7-14 Myojin-machi
Hachioji-shi, Tokyo 192
Tel 0426-48-8770
FAX: 0426-48-8775

Intel Japan K.K.*
Kawa-asa Bldg
2-11-5 Shin-Yokohama
Kohoku-ku, Yokohama-shi
Kanagawa 222
Tel 045-474-7660
FAX: 045-471-4394

Intel Japan K K.*
Ryokuchi-Eki Bldg
2-4-1 Terauchi
Toyonaka-shi, Osaka 560
Tel. 06-863-1091
FAX 06-863-1084

Intel Japan K.K
Shinmaru Bldg.
1-5-1 Marunouchi
Chiyoda-ku, Tokyo 100
Tel: 03-3201-3621
FAX: 03-3201-6850

Intel Japan K K *
TK Gotanda Bldg. 9F
8-3-6 Nishi Gotanda
Shinagawa, Tokyo 141
Tel. 03-3493-6081
FAX: 03-3493-5951

KOREA

Intel Korea, Ltd
16th Floor, Life Bldg.
61 Yoido-dong,
Youngdeungpo-Ku
Seoul 150-010
Tel: (2) 784-8186
FAX: (2) 784-8096

MEXICO

Intel Tecnologia de Mexico
S A de C V
Av. Mexico No. 2798-9B, S.H.
44680 Guadalajara, Jal.
Tel. 523-640-1259
FAX. 523-642-7661

SINGAPORE

Intel Singapore Technology, Ltd.
101 Thomson Road
#08-05
United Square
Singapore 1130
Tel: (65) 250-7811
FAX: (65) 250-9256

TAIWAN

Intel Technology Far East Ltd.
Taiwan Branch
8th Floor, No 205
Bank Tower Bldg
Tung Hua N. Road
Taipei
Tel. 886-2-514-4200
FAX: 886-2-717-2455

INTERNATIONAL DISTRIBUTORS/REPRESENTATIVES

ARGENTINA

Dafsys Consulting S A
Chacabuco, 90-6 Piso
1069-Buenos Aires
Tel: 54-1-342-7726
FAX: 54-1-334-1871

Reycom Electronica S.A
Bernardo de Irigoyen 972-2B
1304-Buenos Aires
Tel: 541-304-2018
FAX: 541-304-2018

AUSTRALIA

NJS Electronics Australia
1A/37 Ricketts Road
Mount Waverly, VIC 3149
Tel: 61-3-558-9868
TLX: AA 30895
FAX: 61-3-558-9929

NSD-Australia
205 Middleborough Rd.
Box Hill, Victoria 3128
Tel: 03 8900970
FAX. 03 8990819

BRAZIL

Hitech
Av. Eng. Luiz Carlos Berrini
801-12 andar
Sao Paulo, SP
Cep 04571-901
Tel. 55-11-536-0355
FAX: 55-11-240-2650

Itaucom
Av Wilhelm Winter, 301
Jundai, SP Brazil
Cep 13213-000
Tel. 55-11-735-3184
FAXL 55-11-735-3004

CHILE

DTS
Rosas 1444
Santiago
Tel: 562-697-0991
FAX: 562-699-3316

CHINA/HONG KONG

Novel Precision Machinery Co., Ltd.
Room 728 Trade Square
681 Cheung Sha Wan Road
Kowloon, Hong Kong
Tel: (852) 360-8999
TWX: 32032 NVTNL HX
FAX: (852) 725-3695

GUATEMALA

Abinitio
11 Calle 2—Zona 9
Guatemala City
Tel. 5022-32-4104
FAX 5022-32-4123

INDIA

Priya International Limited
D-6, II Floor
Devatha Plaza
131/132 Residency Rd.
Bangalore 560 025
Tel. 91- 80 214027, 91-80-214395
FAX 9-80-214105

Priya International Limited
Apeejay House, 4th Floor
130 Apollo Street
Bombay 400 023
Tel. 91-22-2660949, 91-22-2665822

Priya International Limited
Flat No 8, 10th Floor
Akashdeep Building
Barakhamba Rd
New Delhi 110 001
Tel. 91-11-3314512, 91-11-3310413
FAX. 91-11-3719107

Priya International Limited
5-J, Century Plaza
560-562 Mount Road,
Teynampet
Madras 600 018
Tel 91- 44-451031, 91-44-451597

Priya International Limited
No. 10, II Floor, Minerva House,
94 Sarojini Devi Rd.
Secunderabad 500 003
Tel: 91-842-813120, 91-842-813549

Priya International Limited
Lords, III Floor
7/1 Lord Sinha Road
Calcuta 700 071
Tel 91-33-222378, 91-33-222379
FAX. 91-33-224884

SES Computers and Technologies Pvt. Ltd
11/18, SNS Chambers
239 Palace Upper Orchards
Sankey Road,
Sadashivanagar
Bangalore 560 080
Tel. 91-812-348481
FAX. 91-812-343685

SES Computers & Technologies Pvt Ltd
Arvind Chambers
194, Andheri-Kurla Road
Andheri (East)
Bombay 400 069
Tel: 91-22-6341584, 91-22-6341667
FAX 91-22-4937524

SES Computers & Technologies Pvt. Ltd
605-A Ansal Chambers II
No. 6, Bhikaji Camaplace
New Delhi 110 066
Tel 91-11-6881663
FAX 91-11-6840471

JAMAICA

MC Systems
10-12 Grenada Crescent
Kingston 5
Tel (809) 926-0104
FAX. (809) 929-5678

JAPAN

Asahi Electronics Co. Ltd
KMM Bldg 2-14-1 Asano
Kokurakita-ku
Kitakyushu-shi 802
Tel 093-511-6471
FAX 093-551-7861

Dia Semicon Systems, Inc
Flower Hill Shinmachi
Higashi-kan
1-23 Shinmachi, Setagaya-ku
Tokyo 154
Tel: 03-3439-1600
FAX: 03-3439-1601

Okaya Koki
2-4-18 Sakae
Naka-ku, Nagoya-shi 460
Tel 052-204-8315
FAX: 052-204-8380

Ryoyo Electro Corp
Konwa Bldg
1-12-22 Tsukiji
Chuo-ku, Tokyo 104
Tel 03-3546-5011
FAX. 03-3546-5044

KOREA

Samsung Electronics
Samsung Main Bldg.
150 Taepyung-Ro-2KA,
Chung-Ku
Seoul 100-102
C.P.O. Box 8780
Tel. (822) 751-3680
TWX KORSST K 27970
FAX: (822) 753-9065

Tong Baek Electronic Co., Ltd.
16-58 Hangang-ro 3-ga
Yongsan-gu, Seoul
Tel 82-2-715-6623
FAX. 82-2-715-9374

MEXICO

Mexel
Calle Diagonal No 27-3er Piso
Col. del Valle
Mexico D F , C.P. 03100
Tel. 525-682-8040
FAX 525-669-2983

Dicopel, S.A. de C.V
Fco Pimentel No. 98
Col. San Rafael
Mexico D.F , C.P 06470
Tel 525-705-7422
Fax 525-703-1772

SAUDI ARABIA

AAE Systems, Inc
642 N Pastoria Ave.
Sunnyvale, CA 94086
U.S.A
Tel: (408) 732-1710
FAX: (408) 732-3095
TLX 494-3405 AAE SYS

SINGAPORE

Electronic Resources Pte, Ltd.
17 Harvey Road
#03-01 Singapore 1336
Tel: (65) 283-0888
TWX: RS 56541 ERS
FAX: (65) 289-5327

SOUTH AFRICA

Electronic Building Elements
178 Erasmus St
(off Watermeyet St.)
Meyerspark, Pretoria, 0184
Tel 011-2712-803-7680
FAX: 011-2712-803-8294

TAIWAN

Micro Electronics Corporation
12th Floor, Section 3
285 Nanking East Road
Taipei, R.O C.
Tel: (886) 2-7198419
FAX: (886) 2-7197916

Acer Sertek Inc.
15th Floor, Section 2
Chien Kuo North Rd.
Taipei 18479 R O.C.
Tel: 886-2-501-0055
TWX: 23756 SERTEK
FAX: (886) 2-5012521

URUGUAY

Interfase S.A.
Blvr. Espana 2094
11200 Montevideo
Tel. 598-249-4600
FAX: 598-249-3040

VENEZUELA

Unixel C A.
4 Transversal de Monte Cristo
Edf AXXA, Piso 1, of. 1&2
Centro Empresarial Boleita
Caracas
Tel. 582-238-7749
FAX: 582-238-1816

*Field Application Location

intel®

NORTH AMERICAN SERVICE OFFICES

PrimeService

Intel Corporation's North American Preferred Service Provider

Central Dispatch: 1-800-876-SERV (1-800-876-7378)

ALABAMA
Birmingham
Huntsville

ALASKA
Anchorage

ARIZONA
Phoenix*
Tucson

ARKANSAS
Little Rock

CALIFORNIA
Bakersfield
Brea
Carson*
Fresno
Livermore
Mar Del Rey
Ontario*
Orange
Sacramento*
San Diego*
San Francisco*
Santa Clara*
Ventura
Sunnyvale
Walnut Creek*
Woodland Hills*

COLORADO
Colorado Springs
Denver
Englewood*

CONNECTICUT
Glastonbury*

DELAWARE
New Castle

FLORIDA
Ft Lauderdale
Heathrow
Jacksonville
Melbourne
Pensacola
Tampa
West Palm Beach

GEORGIA
Atlanta*
Savannah
West Robbins

HAWAII
Honolulu

ILLINOIS
Buffalo*
Calumet City
Chicago
Lansing
Oak Brook

INDIANA
Carmel*
Ft Wayne

KANSAS
Overland Park*
Wichita

KENTUCKY
Lexington
Louisville
Madisonville

LOUISIANA
Baton Rouge
Metarie

MAINE
Brunswick

MARYLAND
Frederick
Linthicum*
Rockville*

MASSACHUSETTS
Boston*
Natick*
Norton*
Springfield

MICHIGAN
Ann Harbor
Benton Harbor
Flint
Grand Rapids*
Leslie
Livonia*
St. Joseph
Troy*

MINNESOTA
Bloomington*
Duluth

MISSOURI
Springfield
St. Louis* NEVADA
Minden
Las Vegas
Reno

NEW HAMPSHIRE
Manchester*

NEW JERSEY
Edison*
Hamton Town*
Parsippany*

NEW MEXICO
Albuquerque

NEW YORK
Albany*
Amherst*
Dewitt*
Fairport*
Farmingdale*
New York City*

NORTH CAROLINA
Brevard
Charlotte
Greensboro
Haveluch
Raleigh
Wilmington

NORTH DAKOTA
Bismark

OHIO
Cincinnati*
Columbus
Dayton
Independence*
Middle Heights*
Toledo*

OREGON
Beaverton*

PENNSYLVANIA
Bala Cynwyd*
Camp Hill*
East Erie
Pittsburgh*
Wayne*

SOUTH CAROLINA
Charleston
Cherry Point
Columbia
Fountain Inn SOUTH DAKOTA
Sioux Falls

TENNESSEE
Bartlett
Chattanooga
Knoxville
Nashville

TEXAS
Austin
Bay City
Beaumont
Canyon
College Station
Houston*
Irving*
San Antonio
Tyler

UTAH
Salt Lake City*

VIRGINIA
Charlottesville
Glen Allen
Maclean*
Norfolk
Virginia Beach

WASHINGTON
Bellevue*
Olympia
Renton
Richland
Spokane
Verdale

WASHINGTON D.C.*

St. John WEST VIRGINIA
St. Albans

WISCONSIN
Brookfield*
Green Bay
Madison
Wausau

CANADA
Calgary*
Edmonton
Halifax
London*
Montreal*
Ottawa
Toronto*
Vancouver, BC*
Winnipeg
Regina

NORTH AMERICAN CUSTOMER TRAINING CENTERS

ARIZONA
Computervision Customer Education
2401 W Behrend Dr , Suite 17
Phoenix 85027
Tel 1-800-234-8806

ILLINOIS
Computervision Customer Education
1 Oakbrook Terrace
Suite 600
Oakbrook 60181
Tel 1-800-234-8806

MASSACHUSETTS
Computervision Customer Education
11 Oak Park Drive
Bedford 01730
Tel. 1-800-234-8806

SYSTEMS ENGINEERING OFFICES

MINNESOTA
3500 W 80th Street
Suite 360
Bloomington 55431
Tel (612) 835-6722

NEW YORK
2950 Expressway Dr , South
Islandia 11722
Tel. (506) 231-3300

*Carry-in locations

Pentium™ Processors and Related Products

In 1993, Intel introduced the Pentium™ Processor, the fifth-generation processor based on the Intel Architecture. The Pentium processor represents the next generation of power for high-end desktop and server performance. The Pentium processor is fully code-compatible with previous members of the Intel Architecture family of microprocessors, thus preserving the value of user's software investments. Up to eight times as powerful as the 33-MHz Intel486™ DX processor, the Pentium processor extends the Intel processor performance continuum while maintaining full compatibility with existing software.

The Pentium processor employs the most advanced technology and engineering innovation and is the enabling technology for today's high-end and tomorrow's emerging applications. The Pentium processor incorporates a superscalar architecture, improved floating point unit, separate on-chip code and data caches, 64-bit external data bus, and other features designed to provide an architectural platform for high-performance computing.

This databook contains information regarding the design of Pentium processor-based systems including secondary cache subsystems, local bus systems based on the PCI specification, and Advance Programmable Interrupt Controller (APIC) designs.

intel

Order Number: 241732-002
Printed in USA/0195/18K/RRD/MY
Microprocessors

Printed on Recycled Material