INTERDUCTION TO PERSONAL COMPUTER

Chapter 1 CPU and Memory Speed

A computer executes instructions. Each instruction tells the computer to add, subtract, multiply, divide, or compare two numbers to see which is larger. The rest is housekeeping.

Even an application that appears to do no arithmetic is still using numbers. Consider the automatic error correction in a word processor. If you type in " teh " the computer appears to recognize the common misspelling and changes it to " the ". What does this have to do with numbers? Well, every character that you type, including the space bar, transmits a code to the computer. The code is ASCII, and in that code a blank is 32, "a" is 97 and "z" is 122. So the computer sees " teh " as the sequence 32 116 101 104 32. The word processor has been programmed to check for this sequence, and when it sees it it exchanges the 101 and 104. The CPU chip doesn't know about spelling, but it is very fast and accurate handing numbers.

If everything is numbers, you might think that the speed of the computer is largely determined by how fast it can add. People expect this because adding large numbers takes us a long time. Ask someone how much is 2+2 and they will respond immediately 4. Ask how much is 154373 + 382549 and they will stop for a minute and take out a pencil. A computer adds numbers with electronic circuits that work just as fast for large or small numbers. Arithmetic is what computers do best, and they do it almost instantly.

A CPU is measured by how many instructions it can process in a second, not by how long it takes to process any single instruction. Consider a fast food counter. They have a bunch of lines, several people working the counter, and lots of people in back cooking the food. They measure themselves by how many customers they serve in any period of time. When you come to the the front of the line, the item you want may be temporarily unavailable and you have to step aside. It might take you unusually long to get your burger, but lots of other people are being served during the period. To you, the service is slow. To the business, they are moving lots of people through.

In the same way, a CPU is designed to fetch programming, fetch data, and execute instructions. Sometimes a particular instruction needs data that is not immediately available. All modern processors can push the instruction aside and have it wait while subsequent instructions are serviced. Speed is measured by the overall throughput of the chip.

1-1. Clock Speed: Tell Me When it Hertz

Computer performance is a traffic problem, moving data and instructions from memory and around inside the chip. Most people think of "traffic" in terms of cars and highways. However, there is a more relevant traffic analogy that everyone experienced before they learned to drive.

Students have been sitting in class for a long time. Finally the bell rings throughout the school signaling the end of the current period. Everyone gets up and moves through the hall to their next classroom. After a few minutes the bell rings again to signal the start of the next period. The bell has to ring everywhere in the school at the same time to coordinate movement. If each teacher decides

independently when a period is over, then students will frequently arrive at the next room and find it filled with the previous class.

The various parts of a computer hold instructions and data. Periodically they send this data along wires to the next processing station. To coordinate this activity, the computer provides a clock pulse. The clock is a regular pattern of alternating high and low voltages on a wire. The clock speed is measured in Megahertz. One Megahertz (1 MHz) is a signal that alternates between high and low values one million times a second. A 66Mh PC has a clock which "ticks" and "tocks" 66 million times each second. Each tick-tock sequence is called a cycle. The clock pulse tells some circuits when to start sending data on the wires, while it tells other circuits when the data from the previous pulse should have already arrived.

The earliest PC had one clock, and its signal applied to the CPU, memory, and all the I/O devices. A modern PC has many different clock signals for different areas of the machine.

The CPU and memory receive a clock pulse of 66, 100, or 133 MHz from the mainboard. Intel doesn't currently like the 133 MHz speed for memory and has a few motherboards that give a 133 MHz clock to the CPU and a separate 100 MHz clock to the memory. Inside the CPU a faster clock rate is generated synchronized to the external clock. For example, if the CPU runs an internal clock pulse that is 5 times as fast as a 100 MHz main clock, then the CPU rate is 500 MHz. The PCI I/O bus runs at 33 MHz. Typically this is accomplished by dividing the 66 MHz clock in half, or the 100 MHz clock by a third.

If you buy a mainboard manufactured by Intel, or a system from a well-known corporate supplier like IBM or Dell, then the clock rate will be fixed at the official standard value for that type of CPU. However, the clock rate can typically be set to other values, and most third party equipment supplies provide boards where the clock rate can be configured from the power-up setup panels. When the customer increases the clock rate beyond the standard values, this is called "overclocking".

Intel designs a particular CPU chip to run at a range of speeds. When they finish with the design and engineer the fabrication process, they have spent a couple of billion dollars. To be safe, they probably design in a slop factor just in case something is slightly wrong. Then they start to manufacture chips. A lot of the chip price goes to recover the original design and setup cost.

Intel also sets a price range for the various speeds based on what they expect the customers will be willing to pay. However, if they have done their engineering right, each chip is probably a little better than it needs to be. Sometimes Intel has a shortage of the lower priced chips and a surplus of the faster chips, and the easiest way to solve the problem is to remark the better chips to the lower price. Because of this, home users are often successful in overclocking the lowest speed chips in any given family.

The Celeron processor is supposed to run at a 66 MHz clock rate, but it can often run successfully at a 75 or 83 MHz rate. The problem is that this also speeds up the PCI bus beyond its official 33 MHz speed and some of the adapter cards won't work correctly. You also have to put faster memory in the machine so it can keep up with the overclocking.

You can spend a lot of time configuring, testing, and reconfiguring a system. If you have any real work to get done, overclocking usually isn't worth the effort. People do it because it as a hobby. If it makes you feel like an outlaw, take a walk on the wild side.

The only unambiguous measure of advanced technology is the width of circuits in the CPU chip. Smaller circuits draw less power, generate less heat, and can run at faster clock speeds. The first Pentium II processors were built on .35 micron circuit sizes and were powered by 2.8 volts. That was quickly replaced by .25 micron circuits running first at 2.2 volts and then 2.0 volts. The current state of the art is .18 micron circuits powered at 1.6 volts. The changes in size and power explain why it is not possible to simply replace an old 233 MHz Pentium II processor with a new 733 MHz Pentium III. Plug a 1.6 volt chip into a 2.8 volt socket and it will burn out.

1-2. Nanoseconds

All the ads and specifications quote clock speed in Megahertz. However, the more important number is the length of time between clock ticks (the cycle time). Such periods are usually measured in nanoseconds (billionths of a second) abbreviated "nsec."

Electricity travels through a copper wire just a bit slower than the speed of light. Normally, we can just regard the speed of light as "very fast." It becomes important when the distances are very long (astronomy) or when the times are very short (computers). A nanosecond is the amount of time that it takes light (or an electric signal) to travel about one foot.

PC clock speeds appear at first to be a strange collection of numbers. However, the corresponding cycle types display a much more regular pattern:

Clock	Cycle
66Mh	15 nsec
100Mh	10 nsec
500Mh	2 nsec

A processor with a 500 MHz clock must perform operations in less time than it takes for electricity to travel 2 feet. The chip is very small, but it has millions of circuits. All must be manufactured to a very high level of precision.

However, it is much simpler to apply quality control to a chip the size of a fingernail than to the entire mainboard. This by itself show the problems of a higher speed main clock, and the benefit of capping the I/O bus design at 33 MHz (30 light-feet of signal distance).

1-3. Instructions per Cycle: Get in Gear

To add up a column of numbers with a pocket calculator, you simply type each number in and press the "+" key (or the "=" key at the end). Most users probably think that a PC spreadsheet program does the same thing. However, the human brain has actually been doing the hard part of the operation, moving down one row in the column, focusing on the number, and recognizing it. Each PC instruction carries with it a number of additional operations that would not be obvious to the casual user.

First, the computer must locate the next instruction in memory and move it to the CPU. This instruction is coded as a number. The computer must decode the number to determine the operation (say ADD), and the size of the data (say 16-bits). Additional information is then moved and decoded to determine the location in memory (the row and column of the spreadsheet). Finally, the number is added to the running total. Although a human might take some time to add two eight-digit numbers together, the addition is the simplest part of the operation for a computer chip. Decoding the instruction and locating the data take the most time.

Each generation of Intel CPU chip has performed this operation in fewer clock cycles than the previous generation.

A 386 CPU required a minimum of 6 clock ticks to add two numbers.
A 486 CPU can generally add two numbers in two clock ticks.
A Pentium CPU can add two numbers in a single clock tick.
A Pentium II, III, Celeron, or Xeon can add two numbers in a single clock tick. If it discovers that the next instruction needs data that hasn't arrived from slow memory, it can rearrange things to execute subsequent instructions until the data arrives.

To make a car go faster, one steps on the accelerator. Extra gas makes the engine rotate faster. When RPM gets high enough, it is better to shift to a higher gear. The PC system clock (measured in MHz) is like the engine speed (measured in RPM). The CPU model selects the gear. The original 86 processor was like first gear, and the 486 is like fourth gear. So it is a mistake to compare clock speed across changes in the architecture.

1-4. High School Analogy

The first generation of PC CPU chips was like a one room schoolhouse. A class of students could enter and be seated. The first period would be English. When the bell rings, they switch books and take a period of Math. Then History, a Language, and finally Science. When all the subjects are done, they get up, leave the school, and another class can enter, sit down, and take their classes.

If you want the school to educate students more efficiently, you could try to shorten the periods (speed up the clock). However, you can also speed up things by building more classrooms. That is what happened with the 286, 386, and 486 generations of chips. In a school designed like a 486, there is one classroom for each subject. When the bell rings, the students in the English room move to Math, the Math students move to History, and so on. The students in the last class, Science, leave the school. A new class enters and sits down in the English classroom to begin their sequence of subjects.

Each new generation of chips typically triples the number of circuits of the previous generation. So the fifth generation chip, the Pentium, added a complete second set of classrooms. Now two classes would take each subject at the same time.

1-5. Dependent Instructions

If the Pentium High School has two English (first period) classrooms, then every time the bell rings two new classes of students can begin studying. In a real CPU chip, this means that with every tick of the internal clock two new instructions can begin execution. However, occasionally one computer instruction depends on the results of the previous operation. For example, to find the average of two numbers you first add them together and then divide the sum by two. If the ADD and DIVIDE instructions are both waiting to execute, they cannot both begin execution at the same time. The ADD has to start first, because only after the sum has been calculated can the DIVIDE be executed.

The Pentium chip checks each pair of instructions as they are about to execute. If the two instructions are independent (that is, if the second instruction does not use the results of the first instruction) then both can start running at the same time. If the second depends on the previous instruction it is held up, and in that clock tick only the first instruction begins execution (in the analogy, one of the English

classrooms is empty). The next time the clock ticks again, the second instruction (and maybe the one after it) will begin execution and the previous instruction will advance to the next phase of processing.

1-6. Memory Access Delay

An ADD instruction adds two numbers together. In order to execute this instruction, the CPU first has to get the instruction itself from memory and then fetch one or both numbers. To speed processing, every modern CPU chip has two types of internal memory. This Cache memory holds the most recently used sections of programs and data.

The best type of internal memory is the Level 1 (L1) cache. This memory is part of the CPU core along with the units that decode instructions and perform arithmetic. If the instruction and data are in L1 cache then the CPU can execute at full speed. The modern Intel processors have 32K of L1 internal cache. Competing processors from AMD and Via have more L1 cache.

When the instruction or data is not found in the L1 cache, modern processors have a larger amount of Level 2 cache either integrated into the CPU chip or mounted with the CPU chip inside the processor cartridge. The Pentium II had 512K of L2 cache memory operating at half the speed of the CPU. The Celeron and newer Pentium III chips have 128K and 256K respectively of L2 cache in the CPU chip operating at full processor speed.

If the processor needs an instruction or data residing in the L2 cache, then it must wait 2 to 4 cycles for the L2 cache to supply that data. Often the CPU can find other instructions pending execution that are not blocked by the missing data that can be executed while waiting.

The main memory, however, is Dynamic Random Access Memory (DRAM). DRAM is much slower than the CPU or any of its caches. A 500 MHz CPU clock has a cycle time of 2 nanoseconds, but DRAM takes a minimum of 60 nanoseconds to respond with new data. No matter how clever a Pentium III chip may be about juggling the order of execution, it will eventually have to wait for the data and miss the opportunity to execute a substantial number of the 60 instructions it could have processed during the period.

1-7. RISC Architecture

The first Intel "CPU on a chip" was the 4004 processor. It was more like a pocket calculator than a real computer. It handled ordinary base 10 digits encoded as four bits. Later chips added the ability to handle 8 bit, 16 bit, and 32 bit numbers. So on a modern Intel CPU chip there is no single Add instruction. Instead, there are separate Add operations for digits, bytes, and every other size of number. The resulting set of possible instructions is a mess. This is typical of a "Complex Instruction Set" computer chip.

In your Sunday paper, right next to the CompUSA insert there is probably something from Sears. Look at the last few pages of the ad, where they show the tools. There will almost certainly be a picture of the traditional "190 Piece Socket Wrench Set." If you purchased this item, you would always have the right tool for any job. In reality, it is almost impossible to keep all the pieces organized, and you will spends minutes searching through all the attachments to find one of the right size.

Go to a tire store. They lift your car off the floor, remove the hubcaps, and then pick up a gun shaped device connected to a hose. "Zuuurp" and each bolt comes off the wheel. You could do the same thing with the 190 Piece Socket Wrench Set, but every garage knows that automotive wheel bolts come in

only one size. So they don't have to spend time searching for the right size tool, and they can optimize the one size that they really need.

When computer designers realized the same thing, it was called Reduced Instruction Set Computers or RISC. Make all the instructions the same size. Use only one size of data. Simplify the instructions and therefore the operation decode. Then use all the room on the chip to optimize what is left, rather than filling the chip with support for instructions that are seldom executed.

Two or three years ago it was possible to argue that the future belonged to RISC computers. Given the technology available at the time, they were smaller, faster, cheaper, and easier to build than conventional computer chips. In a joint project, IBM, Apple, and Motorola developed the PowerPC chip and Apple proceeded to convert its entire Macintosh line to use it. DEC developed its family of Alpha CPUs, and Sun has its SPARC family.

Then all the other vendors sat back and waited for the Intel architecture to hit the dead end that they all predicted was inevitable. However, there is a funny thing about silicon. Technology doubles the power of chips every 18 months, and there are economies of scale when you are selling millions of chips every month.

The advantage of a Reduced Instruction Set turned out to be important in the period when chips have 2-3 million transistors (during the period of the late 486 chips and the early Pentium chips). When the PowerPC was first announced, it was billed as having "the power of a Pentium at the price of a 486." Due to software problems, IBM delayed any wide distribution of PowerPC systems, and the window of opportunity was lost. By the time that any systems other than Apple used the PowerPC, Pentium chips were selling for less than the 486 chips used to cost, and the Pentium Pro combined the best of RISC and conventional chip design.

RISC chips are still widely used in Unix systems, and they can run Windows NT. It is likely that some type of RISC multiprocessor will remain the most powerful choice for a dedicated file and database server when raw power is important and price is less important. However, at this time it appears that Intel is unstoppable and that RISC systems will never capture a significant share of the desktop or laptop market.

1-8. Superscalar, Pipeline, and Multimedia

Although a tire store may be fast at changing tires, when you really need speed look at how they do things in Indianapolis. A race car pulls into the pit for service. They jack it off the ground, and then four teams of mechanics go to work on all four wheels simultaneously. The car is back in the race in a matter of seconds. In ordinary life, such service would be prohibitively expensive. But in the world of microelectronics, transistors are cheap.

A pipeline is the sequence of processing stations that decode instructions, fetch data, perform the operation, and save the results. Inside the CPU, instructions are processed at a sequence of stations that resemble an assembly line. Memory is pipelined when the CPU can request a sequence of addresses one per cycle and then after a delay of typically four or five cycles the memory responds with the data in the order in which it was requested.

A computer is superscalar when it can execute more than one instruction per clock cycle. Since the Pentium chip, Intel processors have been able to execute two instructions per cycle overall.

A CPU is typically a general purpose device. It can perform any type of calculation equally well. There are, however, certain special calculations which occur frequently enough to justify special support.

If you buy an audio CD in a record stored, the music is encoded onto the disk as digital data. Sound is a regular vibrating difference in air pressure. The data on the CD represents a regular sample of air pressure measured one or more microphones. When it is played, a stereo system reproduces those variations in air pressure by vibrating the cones contained in the speakers.

If you listen carefully, you can pick out one instrument in an orchestra or the voice of one person from the background noise of a cocktail party. In technical jargon, this is called “signal processing”. Before computers, signal processing was done with filters that blocked certain sounds or frequencies. Today, audio and video signals can be processed by a special computation.

There is a family of applications. Used one way it can take the pops and scratches out of an old recording. In another case, it can be used to colorize old black and while movies. It is also used to compress the video signals in direct satellite broadcast.

It is possible to buy specialized computer chips that do only digital signal processing. For $15 you can get a chip that does hundreds of millions of instructions per second. Such chips are built into modems and some sound cards.

The problem isn’t the cost of the chip, having the room to mount it inside a PC. Intel has begun to add special support for the signal processing computation with a set of multimedia instructions inside their CPUs. Although such computations could be done using normal instructions, the extra hardware support produces a particularly high level of superscalar performance as many calculations can be performed at the same time.

1-9. Memory Architectures

The newspaper ad offers a computer system with a 500 MHz processor and 64 Megabytes of RAM. In other words, we are interested in how fast a CPU is, but we measure the amount of memory, not the speed.

There are lots of different CPU speeds, but DRAM is a commodity item sold in bulk and all the vendors produce essentially the same technology. Since the motherboards are all configured to support industry standard parts, there is no advantage if a vendor produces memory that is, say, 5% faster than the standard.

Modern Synchronous DRAM has two performance numbers. The first is latency, the delay between the time that a particular data item is requested and the time when the memory can reliably transmit the data back to the CPU. Modern DRAM typically has a 50-nanosecond latency. It is important to remember that latency is measured at the memory chip. Before the chip can begin, the signal has to exit the CPU, be processed by the memory controller, and run down the wires on the mainboard. Exact numbers on this additional mainboard overhead delay are not easily available, but they are substantial.

The second performance number is throughput, the rate at which SDRAM can return additional data from the same general area of memory. This is represented by a memory clock rate current SDRAM parts are PC66, PC100, and PC133 for a 66, 100, and 133MHz memory bus clock rate. The most popular current speed is PC100 memory, which at a rate of 100 MHz can return data every 10 nanoseconds.

So in a system with a 100 MHz "frontside" bus, to transfer a chunk of 32 bytes of data down the 8 bit bus, the CPU will send out the address and wait a minimum of 5 memory bus clock cycles (50 nsec latency) plus maybe an extra cycle or two for the mainbus overhead delay. Then it will receive 8 bytes of data every memory bus clock cycle for the next four cycles. That represents a minimum of 9 100 MHz clock cycles to complete the entire 32 byte transfer.

1-10. Wider Bus

For the last 10 years, the only way to increase memory speed was to widen the memory bus. Since conventional memory parts hold and transfer 8 bytes of data in a memory bus clock cycle, adding another independent row of standard memory chips creates a second memory path that can transfer another 8 bytes. This brute force solution now doubles the memory throughput to 16 bytes every 10 nanoseconds.

1-11. DDR

A more sophisticated and less expensive solution is available with something called Double Data Rate or DDR memory. DDR requires a small change to current memory design. Each memory part (each DIMM) is populated with chips that will deliver twice as many bits (16 bytes of data) in response to each request. However, some additional logic on the DIMM presents only the first 8 bytes of data and holds back the second 8 bytes of data for a half a clock tick.

Every computer clock is represented by a signal where the voltage goes to a high value for half the clock time (the “tick”) and then goes to a low value for the other half of the clock time (the “tock”). Data is normally transferred only at the moment when the voltage rises from its low to its high value (at the start of the “tick”). However, with a bit more logic, you can also transfer data when the voltage drops from the high value to the low value. This allows a 100 MHz clock to transfer data twice per clock cycle, matching the speed that traditionally required a 200 MHz clock. DDR SDRAM transfers the second 8 bytes when the clock signal drops, producing the same performance boost as adding a second memory bus but without the extra wires or sockets.

1-12. Rambus (RDRAM)

The Intel strategy backed a new, powerful, but very expensive memory technology invented by Rambus. Conventional memory up to and including SDRAM is pretty dumb. It sits there till the CPU presents an address, then after the latency period it responds with the data. There is no processing power or control logic on the memory chip.

Rambus DRAM (RDRAM) puts processing power on the memory chip and then connects the memory to the system with a more sophisticated bus. Instead of the conventional 64 bit (8 byte) path, RDRAM transfers only two bytes of data every clock tick. However, it runs the bus using a 600, 700, or 800 MHz clock. So after a typical 10 nsec period in which the SDRAM 100 MHz clock ticks once and transfers 8 bytes, the 800 MHz RDRAM clock ticks 8 times, and transfers 2 bytes per tick, for a total of 16 bytes or twice what SDRAM delivered.

RDRAM is based on the same basic memory technology used in SDRAM. The chip technology is limited to delivering a new unit of data every 10 nsec (corresponding to 100 MHz SDRAM). So how can RDRAM deliver data at 800 MHz? The trick here is that RDRAM doesn't manage all of the memory chips according to a single clock. Instead, chips are grouped into logical "devices". Each

"device" can deliver two bytes of data. The RDRAM bus staggers the processing on the chips so that each device starts 1/8 of a clock tick after the previous device and is therefore ready to deliver data 1/8 of a clock tick later. Although no single chip is delivering data more than once every 10 nsec., the entire array is delivering data eight times as fast.

It should be noted that RDRAM also transfers data on both the rise and fall of the clock signal. Therefore, although we talk about "800 MHz" as the memory speed, the actual memory bus clock is running at 400 MHz with a data transfer at every "tick" and again at every "tock".

The SDRAM bus is idle during the latency period. RDRAM is fully pipelined, overlapping the latency of the next memory request with the transfer of the previous request. It claims to keep the bus 95% busy, generating what is claimed to be 300% more data throughput than SDRAM.

The bad news is that RDRAM today costs about 300% more than the same amount of SDRAM. This is not a good idea for the casual home or business desktop user. RDRAM is likely to remain, at least for some time, a special feature of high end engineering and media workstations and database or Web application servers.

1-13. Burst

It would be inefficient to track data byte by byte. To optimize the cache, the CPU references data in 32 byte units each starting at an address that is evenly divided by 32. When an instruction or data is found to be in one of these blocks of memory, the entire 32 bytes are read by the CPU and are stored in the L1 and L2 cache.

Since the memory bus is eight bytes wide, it takes 4 memory bus clock cycles to complete the transfer. This is called a burst. Once Intel established this cache architecture, the memory vendors could optimize memory designs for the processing of the burst.

The CPU begins by presenting the address of some data to the memory controller. It must then wait for a period equal to the latency of the memory (typically more than 50 nsec) before reading any response. The latency is inherent in the way DRAM is constructed and there are no tricks that can improve it.

Back in the days of the early 486 chips, the CPU had to wait the full latency period for every memory transfer. Then a sequence of improvements (fast paged, EDO, and finally SDRAM) were able to reduce the delay for the second, third, and fourth memory value in the burst sequence. So to get 32 bytes of data from modern SDRAM, the CPU may have to present the address, wait 50 nsec (5 ticks of a 100 MHz bus) without getting any data, and then get 8 bytes of data in each of the next four 10 nsec periods.

1-14. Cache in your Chips

Just how much cache does a CPU need? Not surprisingly, the answer depends on what you use it for. Remember that the 486 was designed with only 8K of cache memory, yet that was enough so that more than 90% of potential memory references were resolved by data located in the cache. With cache, a little goes a long way.

All Intel processors have 32K of L1 cache. The L2 cache is:

Processor	L2 Size	L2 Speed compared to CPU
Celeron	128K	Full
Pentium III (Coppermine)	256K	Full
Pentium II ,other III	512K	Half
Xeon	up to 2M	Full

If you run a word processor, spreadsheet, or Web browser then even the Celeron has enough cache. Additional cache is needed for unusual computationally intensive operations, like Photoshop or other forms of media compression. Intel designed Xeon processors with large cache memory to support multiprocessor servers running database or other memory intensive operations.

All other things equal, more cache is better than less. Clearly no desktop user is going to blow $3800 to get a Xeon processor that, for ordinary applications, will be almost indistinguishable from a $150 Celeron. Just because a Pentium III system is within reach, is it really worth the money?

Vendors have an incentive to sell more expensive units. Some customers will opt for the more expensive machine because they think they're worth it. However, most casual users will get along quite nicely with a Celeron. The Pentium III is engineered best for a workstation or server with two CPUs. If it ever makes sense, the Xeon is designed for corporate servers with 4 or 8 processors.

1-15. Athlon

AMD also makes CPU chips with compatible instructions that compete with Intel. Their high end chip is called Athlon, and it competes directly with the best Intel processors. Generally, Athlon chips have more L1 and L2 cache than Intel and they can execute more instructions per clock tick. So at any given clock speed (say 800 MHz), an Athlon chip will be measurably faster than an Intel Pentium III and probably cost a little less money.

The problem with Athlon has been that mainboard logic has been unable to match the features of the CPU. For example the Athlon processor memory bus operates like DDR memory. It transfers data on both the rise and fall of the clock. Therefore, while the Athlon memory clock runs at 100 MHz, it has the data throughput of a 200 MHz system. To feed this CPU at full speed, a motherboard vendor would have had to put in a second parallel memory bus (which is complex and expensive). So the Athlon has been running with under performing memory configurations. Once DDR SDRAM memory becomes widely available, it will exactly match the Athlon design and this AMD chip may significantly out perform Pentium III systems.

1-16. Willamette

Sometime before the end of 2000, Intel will come out with its next generation 32 bit CPU chip. We are not talking about some minor update, like the transition from Pentium II to Pentium II. This is a whole new design. In addition to faster clock speeds, larger cache, and more internal parallel processing, the Willamette chip will have the equivalent of a 400 MHz memory bus clock. Thats three times the memory throughput of the best current Intel CPU chips (with a 133 MHz memory bus) and twice the speed of current Athlon chips.

The Willamette chip will have a memory bandwidth that finally matches the best current mainboard. Intel's 840 mainboard chipset supports two parallel RDRAM memory buses with an aggregate memory throughput of 3.2 gigabytes per second, and that is exactly the throughput of the Willamette chip.

1-17. Summary

Type	Width (butes)	Clock	Multiplier	Effective Clock	Bytes/10nsec	Performance
PC100 SDRAM	8	100		100	8	100%
DDR SDRAM	8	100	X2	200	16	133%
RDRAM 600	2	300	X2	600	12	
RDRAM 800	2	400	X2	800	16	300%

Why is it that DDR SDRAM appears in all but the last column to have the same performance as RDRAM, but then it only gets a 33% boost while RDRAM is 300% better? The answer is in the latency period. With ordinary and DDR SDRAM, the memory sits idle from the cycle when the CPU requests data to the end of the latency period when the data transfer can begin. Consider a simple chart where the latency is four-cycle and the data transfer (for 100 MHz SDRAM is also four cycles). Doubling the speed of data transfer drops the second part from four cycles to two, but it leaves the latency unchanged. Therefore, the burst drops from 8 cycles (4+4) to six cycles (4+2). That improves the throughput of the system to 8/6 or 133%.

However, RDRAM overlaps latency with the transfer of previous memory requests. It can use as much as 95% of the total theoretical bus bandwidth. That's 380% of PC100 SDRAM, but RDRAM advocates round that down to 300% just to not overstate the case.

Processor Memory Bus:

Type	Width	Clock	Multiplier	Effective Clock
Pentium III	8	100		100
Pentium III	8	133		133
Athlon	8	100	X2	200
Willamette	8	100	X4	400

Chapter 2 I/O Bus

A personal computer may transfer data from disk to CPU, from CPU to memory, or from memory to the display adapter. A PC cannot afford to have separate circuits between every pair of devices. A mechanical switch, like the old phone systems used, would be too slow.

The solution is a Bus. The Bus is simply a common set of wires that connect all the computer devices and chips together. Some of these wires are used to transmit data. Some send housekeeping signals, like the clock pulse. Some transmit a number (the "address") that identifies a particular device or memory location. The computer chips watch the address wires and respond when their identifying number is transmitted. They then transfer data on the other wires.

From the first IBM PC up through the first PS/2 computers (introduced in 1987) a computer had one bus and all of its devices and chips ran at the same speed. On those systems, additional computer memory was often added by plugging an adapter card into the same slots that held I/O adapters. Starting with machines that used the 386 CPU, the memory and CPU of the system ran faster than the I/O devices. The solution was to separate the CPU and memory from all the I/O. Today, memory is only added by plugging it into special sockets on the main computer board.

Whether there is more than one Bus, or one Bus with different speeds, is a matter of perspective. A car drives down the local streets at 25 miles per hour. Then it turns onto a highway ramp and accelerates to 55. Is there one road system, or two? The important thing is that there is a connection that allows a flow of traffic between the two speed zones. Within the PC, data can flow from any chip to any other chip, but different parts of the path run at different speeds.

Analogy Alert: Electricity flows through wire at the same speed everywhere (at about the speed of light). So a "faster" bus is not one where the electrons move faster, but rather one in which the time between meaningful events (the "clock speed") is faster. On a faster bus the chips have to react more quickly, and the engineering has to be more rigorous so that voltage signals can be clean and tight. If you built a road system the same way, the "speed limit" would increase by squeezing down the maximum length of each car and packing them more tightly bumper to bumper as they drove down the road.

In a modern PC, there may be half dozen different Bus areas. There is certainly a "CPU area" that still contains the CPU, memory, and basic control logic. There is a "High Speed I/O Device" area that is either a VESA Local Bus (VLB) or a PCI Bus. An very low cost home computer may have no high speed devices. A more typical desktop system connects the high-speed bus on the mainboard to the display adapter and IDE disk interface chip. Then one or two extra I/O slots may allow adapter cards to connect to the high-speed bus. The remaining I/O device slots support standard "ISA" bus cards. Some computers will also provide sockets for a number of PCMCIA "credit card" adapters commonly found in laptop computers.

2-1. History

In 1984 IBM was shipping its PC AT model. The CPU, memory, and I/O bus all shared a common 8MHz clock. This became the basis for all subsequent clone computers. The term "AT" is a registered trademark of IBM, so this I/O bus became known as the ISA (Industry Standard Architecture) bus.

Every currently marketed PC supports some ISA interface slots. The bus and matching adapter cards are simple and cheap. ISA is a 16-bit interface, which means that data can be transferred only two bytes at a time. More importantly, the ISA bus runs at only 8 MHz and it typically requires two or three clock ticks to transfer those two bytes of data. This is not a problem for devices that are inherently slow like the COM port (modem), the printer port, the sound card, or the CD-ROM. However, the ISA bus is too slow for high performance disk access and therefore is not acceptable in Servers. It is also too slow for modern Windows display adapters.

In 1987 IBM introduced a new Microchannel (MCA) bus. It had clear advantages over the previous PC bus. It's 10 MHz clock was slightly faster. The cards could be automatically configured with a utility program instead of setting physical switches and jumpers. The bus can transfer four bytes of data at a time and, in some configurations and with some cards, it can transfer data every clock tick. However, the Microchannel itself was expensive, the adapter cards were more expensive, and the technology remained encumbered by IBM licensing.

The other vendors developed an extension of the older ISA interface called EISA. An EISA slot contained the older ISA interface, and then an extra socket with additional connections. The user could plug either an old ISA card or a new EISA card into the slot. The newer cards supported a 32-bit data interface and could therefore transfer four bytes of data per operation. However, to remain compatible with the old card, EISA still ran at 8MHz. And the extra logic pushed up the cost of both the EISA system and each adapter card.

As the 486 CPU chip became popular, the idea of running I/O devices at 8 or 10 MHz collided with a mainboard that ran everything else at 33 MHz. It was also clear that it should not require an extra $500 to transfer data at 32-bits.

The first solution was the VESA Local Bus (VLB), which became popular at the start of 1993. VESA is a consortium of companies making displays and display adapters. Desktop machines began to include one or two Local Bus slots to support a high-speed video card and, perhaps, one other high-speed device. A few vendors produced VESA SCSI adapter cards, or Local Bus LAN adapters. Nevertheless, VESA remained largely a display standard.

2-2. PCI - The Current Standard

The PCI bus was developed by Intel. Although it is mostly known for its CPUs, Intel also has a historical association with Ethernet, multimedia, and some disk interfaces. So Intel was unhappy with the VLB concentration on just the video interface and wanted to develop a general-purpose bus. The objective was an interface that was fast and inexpensive. It did not have to be simple (advances in chip technology took care of that) and could achieve a low cost by high volume production.

PCI is a 64 bit interface in a 32 bit package. Figuring this out requires a bit of arithmetic. The PCI bus runs at 33 MHz and can transfer 32 bits of data (four bytes) every clock tick. That sounds like a 32-bit bus. However, a clock tick at 33 MHz is 30 nanoseconds, and memory only has a speed of 70 nanoseconds. When the CPU fetches data from RAM, it has to wait at least three clock ticks for the data. By transferring data every clock tick, the PCI bus can deliver the same throughput on a 32 bit interface that other parts of the machine deliver through a 64 bit path.

The PCI bus has all the signals of the old ISA bus. This allows a PCI adapter card to emulate older equipment. For example, a PCI disk controller can respond to the same addresses and generate the same interrupts as the older disk controllers that the BIOS understands. However, PCI devices can also be self-configuring and operate in a Plug and Play mode.

The PCI bus connects at one end to the CPU/memory bus and at the other end to a more traditional I/O bus. The PCI interface chip may support the video adapter, the EIDE disk controller chip, and maybe two external adapter cards. A desktop machine will have only one PCI chip, and so it will add a number of extra ISA only slots. A server may add additional PCI chips, and extra server slots will usually be EISA.

While ISA and EISA are exclusively PC interfaces, the PCI bus is now used in Power Macintosh systems and PowerPC machines. It may be attractive for minicomputers and other RISC workstations.

2-3. PC Card (formerly PCMCIA) and Card Bus

Laptop computers typically have two slots for "credit card" adapters. Originally this interface was called "PCMCIA" but that proved too technical for wide acceptance. Today there is an effort to rename the interface as "PC Card."

A credit card adapter is much smaller than the adapter cards that plug into the ISA or PCI slots of a laptop computer. They are also more expensive and slower. Although PC Card slots have been offered as an option in some models of desktop computers, the clear disadvantage over full sized card restricts their use to laptops.

The credit card slots are bridged to the main I/O bus of the computer. Laptops have been built to use an ISA, Microchannel, or VESA Local Bus internally. The most modern laptops now come with an internal PCI bus. Credit card adapters can be connected to any of these systems.

A full sized card can come with switches or jumpers that can be used to configure its I/O address or IRQ. More advanced cards may be configured with a utility program. Since it is small project to remove the cover from a desktop system and switch cards, a desktop adapter is designed with the assumption that it will stay put.

A PC Card adapter is sealed in a metal case. It has no configuration switches. They are easy to insert and remove, since a user may need a LAN card for use in the office and a modem card for use at home or when traveling. The PCMCIA standard was developed late enough to incorporate an early version of "plug and play."

In the laptop, each socket is itself an I/O device. A PC Card adapter can be plugged into the system at any time, even when the power is on and a system is running. The socket can query the card for identifying information, and the adapter can be configured by the operating system to use available I/O addresses, IRQs and similar resources.

Full support for PCMCIA was too complicated (and required too much memory) to easily fit into the old DOS operating system. Laptops were not an important platform for Windows NT, so it has a very limited support for this architecture. The best PCMCIA support is found in Windows 95. It is possible to plug a new adapter card into a running Windows 95 machine, have the operating system recognize it immediately, and have the system dynamically configure new driver support.

The main problem with the PCMCIA bus is performance. Current systems support only a 16 bit interface to adapter cards. Adapter cards transfer data at a rather low clock speed, and there is no provision for Busmaster data transfer. The maximum data transfer rate from a PC Card adapter to the CPU or memory is only 2 megabytes per second. A PCI adapter, in contrast, can burst data at 133 megabytes per second.

A 32 bit version of PCMCIA has been created under the label "CardBus". It is currently available only on the most powerful and expensive laptop systems. If you are looking for desktop performance in a portable system, CardBus slots are highly desirable.

2-4. Current Status

Although the ISA bus may be ten years old, it remains a perfectly reasonable option for devices that do not require highest performance. The consumer-oriented products all come with built-in video and disk adapters. Those are the two components that have the greatest impact on home computers and

probably most desktop business machines. There simply isn't any need for a higher speed to support the sound card on a multimedia system, and the CD ROM is the slowest storage device available.

The higher end of the consumer market and the machines sold directly to corporations typically have a few slots with a PCI interface. It is nice to have, but in the next year or so these slots may remain empty. The pre-installed devices cover most requirements, and storage expansion is simple and inexpensive using EIDE disks and devices. A Server comes in a full sized floor standing tower. Some servers have room for 18 disk drives. There will certainly be a PCI bus, but there will also be several standard SCSI adapters to connect all the disks, tapes, and CD-ROM units. Every adapter in a dedicated Server should be a PCI card. Some SCSI controller may be built into the mainboard. An external PCI SCSI adapter can provide additional function, cache memory, or RAID support. LAN adapters should plug into the PCI slot. Because the older ISA slots are much slower, there is a strong bias that any function that is not important enough to warrant a PCI adapter is something you probably shouldn't do on a server. Laptop computers are appearing with an internal PCI bus. This provides high speed support for the internal connection to the video and disk controller chips. It may also provide for PCI cards when the unit is connected to a docking station.

2-5. Interrupts

An I/O bus is similar to bus between the CPU, mainboard control logic, and memory. Both types of bus structure have address wires, data wires, and a similar set of housekeeping wires. Both bus structures must determine if an operation refers to memory or an I/O address. Both must distinguish between 8-bit, 16-bit, and 32-bit operations. Both must be able to introduce "Wait States" to slow down the CPU when a device needs more time to complete an operation.

The most important difference between the CPU-memory local bus and the I/O bus is the presence of Interrupt Request (IRQ) wires. The I/O bus has 15 separate IRQ wires. The CPU has only one interrupt pin. The chip set on the mainboard has to provide a translation between the two.

Without interrupts, the CPU must start an operation to a device and then spin in a loop asking, "Is it done yet? Is it done yet? Is it done yet?" After a few hundred thousand tests, the device will signal that the operation is complete. Interrupts allow the CPU (particularly on a more advanced operating system like Windows 95, OS/2, or NT) to do some other work until the operation is complete.

When a device generates an interrupt, the CPU hardware stops running an ordinary program and jumps to an interrupt handling routine in the Device Driver. The interrupt may signal that:

A previous request is complete and the device can now start a new request. Data has arrived that needs to be read (this happens mostly with the keyboard, mouse, modem, or LAN). An error has been detected on an idle device.

Any device on the I/O bus can request an interrupt by placing a signal on one of the 15 IRQ wires. If more than one IRQ signal is received at the same time, the chip set on the mainboard has to select the one with highest priority to process first. The CPU is interrupted (by sending a signal on its one wire) and the chip set then transfer the identity of the IRQ level to be processed.

Each IRQ wire goes to every slot in the I/O bus. An adapter card is configured, physically with switches or logically with a utility) to use a specific IRQ value. The I/O bus on the first PC assumed that each device would have its own IRQ line, so the circuit to drive an interrupt request was made very simple. Unfortunately, when two adapter cards are incorrectly configured with the same IRQ value, and if both try to generate IRQ's at the same time, the result is to produce a short in the I/O bus.

Usually there is no damage, but the effect can be to burn out either card or to trash the mainboard.

Later bus architectures (MCA, EISA, PCI) use safer circuitry that allows two devices to share the same interrupt. When an interrupt is shared, the system responds to an interrupt by calling the device driver for each device associated with that IRQ. The drivers poll their respective adapter cards to determine if there is any pending activity which requires a response. There is a slight loss of efficiency when interrupts are shared, but not enough to cause worry.

2-6. Plug and Play

An adapter card designed for Microchannel, EISA, or PCI can handle the IRQ problem automatically. However, the rest of the interface to advanced bus structures is expensive. The ISA bus interface is simple and cheap, but it requires the user to set the IRQ, either with physical switches on the card or through some kind of setup configuration utility. The solution to this problem is a new arrangement called "Plug and Play."

To use Plug and Play you need three things:

1.The PC has to be ready to do Plug and Play. Most new computers have this ability, and most old ones don't.
2.The adapter card has to be enabled for Plug and Play. It must minimally be able to report to the computer what I/O address and IRQ it is using or is able to use, and it must accept commands to use the values that the computer selects for it.
3.The operating system must be able to support Plug and Play. Currently, this limits you to Windows 95. Windows NT 4.0 provides limited support, based presumably on the theory that anyone running NT better be sophisticated enough that IRQ issues are no problem. OS/2 Warp doesn't support it but Merlin may.

2-7. The ISA Bus

The CPU uses the OUT instruction to send data or commands to an I/O device, and it uses the IN instruction to read data or status from the device. These instructions cause the address of the I/O device to be placed on the bus, and they flag one of the housekeeping wires in the bus to indicate that this is an I/O address and not a memory address.

Each device is configured to respond to a range of addresses (the "ports"). Generally, a device will respond to a range of eight addresses. For example, the COM1 port responds to addresses 03F8 to 03FF. When the device sees an I/O address on the bus that matches a value in its range, it responds.

A device uses each address to process a different type of command or generate a different type of status. COM1, for example, uses the first address of 03F8 to handle all the data. The remaining four addresses are used to configure the line (speed, parity), to control the phone (hang-up, begin), to check modem status, and perform other housekeeping.

Unfortunately, it is still possible to plug a card built in 1984 into a modern PC. The I/O bus cannot know how fast a device is able to operate, so it handles the worst case and slows everything down to match the speeds used ten years ago. The chip set on the main board generates a long stream of Wait State signals, forcing the CPU to wait for 250 nanoseconds. The device itself can respond to request more Wait States to give itself even longer to respond. All this time, the CPU is stopped dead waiting for the IN or OUT instruction to end.

2-8. Direct Memory Access

All PC devices support Programmed I/O (PIO). The operating system executes IN and OUT instructions to read or write data one, two, or four bytes at a time to the device. For simple devices like the keyboard or display this may be perfectly reasonable. DOS and Windows 3.x do not support multitasking, so the CPU isn't doing anything useful while any device is busy.

A smarter device can transfer data directly to memory without the use of the CPU. This is called Direct Memory Access or DMA. Some DMA capability was designed into the first IBM PC. However, DMA dropped out of favor during the 1980's. The problem is that the original DMA design required dditional bus cycles, and was therefore slower than Programmed I/O. As long as DOS was the primary operating system, it was better to disable or bypass DMA and let the CPU do the work.

One special case was the Multimedia Sound Card. A DOS or Windows game stores data representing some music or speech in a buffer. It then passes the sound card the address of the data and its length. The card uses DMA to fetch bytes of sound and play them while the CPU proceeds to show video or run the game. When the buffer is used up, the sound card generates and interrupt and the program provides a new buffer of data.

There are a set of DMA controller chips on the Mainboard. Each DMA circuit has a level number and can support one device. When installing new adapters, it is important to avoid conflicts for the DMA number just as one avoids conflicts for the I/O address or interrupt level.

A program stores into the DMA circuit a starting memory buffer address and length. When the device is ready for more data, it uses one bus cycle to send a request to the DMA chip, the chip then substitutes for the CPU in generating the next buffer address to the memory circuits to fetch the next chunk of data for the device. The CPU can be running other programs. However, that first signal from the device to the DMA chip takes one more bus cycle than ordinary Programmed I/O. Thus DMA has not been attractive for disk, LAN, and other performance critical I/O.

2-9. Busmaster

On a Microchannel, EISA, or PCI machine, the I/O bus presents a full set of 32 address wires to every adapter card. The adapter can then generate a memory address to reference any location in RAM. A Busmaster adapter card has its own Direct Memory Access control chip on the adapter board. That chip can generate a sequence of memory addresses to read data from memory to the adapter, or to write data from the adapter to memory.

Since DMA and I/O functions are combined on the same card, they can be tightly coordinated. A Busmaster EISA device can transfer data every other cycle, and PCI Busmaster cards can move data in every I/O cycle.

The software on the PC builds a high level request to READ or WRITE an entire buffer of data. It passes the address and size of the buffer to the Busmaster card. Then the PC software does something else.

Internally the Busmaster card moves data between itself and the buffer in memory. When the entire buffer has been processed, it generates an interrupt to the CPU indicating that the request is complete.

In order to make effective use of a Busmaster adapter you need two things. First, you need to have something else to do while the adapter performs the I/O. Secondly, you need an operating system that will allow I/O to run in the background. Generally Busmaster adapters make the most sense on Server machines running a multitasking operating system (Windows NT, OS/2, UNIX, or NETWARE).

Chapter 3 Video Adapters

Today, most systems are sold with a display adapter that connects to a PCI or VESA "local bus", supports some Windows accelerator, and provides SVGA resolutions. The "local bus" means that the CPU can send data to the card at high speed. The "accelerator" means that the display adapter can draw lines and boxes and can move windows and scroll text itself. Resolution and number of colors are determined by the amount of video memory, and refresh rate is determined by the quality of the components. All these items need to be explained in detail.

Display adapters are characterized by

Resolution
Color Depth
Refresh Rate
Bus interface
Accelerator

3-1. Resolution

It refers to the number of dots on the screen. It is expressed as a pair of numbers that give the number of dots on a line (horizontal) and the number of lines (vertical). Four resolutions are in common use today

640x480 (VGA)
800x600 (SVGA)
1024x768
1280x1024

A computer display is essentially a high resolution TV set. It generates colors by combining amounts of Red, Green, and Blue (an "RGB" connection). In current use, these colors are controlled by three wires in the display cable. Each has a variable amount of voltage represented by a number from 0 to 255. This produces a theoretical 16 million possible colors. Complete control of color ("Truecolor") may be needed for displaying photographs, but ordinary applications get along with far fewer.

3-2. Color Depth

Color Depth (number of colors) is determined by the number of bits assigned to hold color value.

4 bits - 16 colors
8 bits - 256 colors
16 bits - 32 or 64 thousand colors
24 bits - 16Million (Truecolor)

The display adapter stores a value (4 to 24 bits) in memory for every dot on the screen. The amount of storage needed is determined by multiplying the number of dots (resolution) by the memory required for each dot.

A 24 bit (Truecolor) depth allows a specific one byte value to be assigned to Red, Green, and Blue signals to the display. A 15 bit color value can be divided into three 5 bit fields with a value of 0 to 31 for R, G, and B. An 8 bit value, however, cannot be reasonably divided into three separate numbers. The adapter holds a table of 256 entries. Each entry has a value for R, G, and B. Different application can compete to program this table with different values. This creates a problem when the two applications appear on different windows of the same screen.

The original VGA display had a resolution of 640x480 and supported 4 bit color. This required only 256K of memory.

An SVGA adapter with 512K can generally support resolution up to 800x600 and 8 bit (1 byte per dot) color.

An SVGA adapter with 1 megabyte of video memory can support 1024x768 resolution at 8 bits, or 800x600 resolution at 16 bits. In some systems, it can also display
1280x1024 in 4 bit color.

Additional memory is required for greater resolution or more color depth.

3-3. Refresh Rate

It determines the speed that the display uses to paint the dots on the screen. The original VGA displays ran at 60Hz, but some people complained that this produced a flicker. International standards now require a rate of 70Hz. A "multisynch" monitor can adapter to refresh rates in a range, typically 60-75Hz.

Everyone knows that you cannot reliably run a CPU chip rated for 166 MHz at a speed of 200 MHz. This this called "overclocking" and though it may work for a while, you run the risk of damaging the chip. However, video systems come with a variable clock rate.

The refresh rate is defined as the number of times that the screen must be rewritten per second. The higher the resolution, the larger the number of dots that have to be written in every refresh cycle. The greater the color depth, the more work that has to be done per point. Increase any of these values and the system has to speed up the clock to support it.

The video adapter chips have a maximum speed. Separately, each display has electronics that can run at some maximum clock rate. Each version of Windows comes with display driver tables that can identify the maximum internal clock speed of each type of adapter card and display.

If you go to Control Panel - Display - Settings, Windows gives the current settings for resolution, color depth, and refresh rate. The model of display monitor establishes limits. If you increase the value of the resolution, the system may reduce the refresh rate to keep the chip speed within tolerance.

If you install Linux, the XFree86 display management software is a bit less sophisticated about identifying and protecting adapter cards and display monitors. The programmer has to configure the limits at a much more primitive level. Make an error and the display could overheat.

3-4. Accelerator

An accelerator chip on the video card can draw lines and boxes, fill in background color, scroll text, and manage the mouse pointer. These functions significantly improve the performance of Windows and OS/2. Before accelerators, a video adapter simply mapped the display memory to an area of the PC memory. The PC program would calculate the location of the line, and then would change the color dot by dot (byte by byte) in this area of memory.

With an accelerator, the CPU only has to send the video adapter a command to draw a line (and the starting point, ending point, width, and color of the line). The CPU is not required to calculate the bits in the line, and the amount of data that has to flow from the CPU through the I/O bus to the adapter card is greatly reduced.

An application program sends a sequence of requests to Window. Each request creates a window, button, box, menu, or writes some text. Some commands can be simply passed on to the accelerator chip for exection. A display adapter requires a Windows Driver routine. The Driver knows which commands the chip can handle, and which have to be turned into bits (or lines) on the CPU.

Most modern PC display adapter cards can process commands as quickly as the CPU can send them. The most common factor that limits performance is the I/O bus.

The first display adapter cards plugged into the old 8 MHz ISA bus. That was terrible. Around the time of 486 systems there was a short period in which adapter cards use the 33 MHz VESA Local Bus interface. However, as the PCI bus became a standard feature of all systems, the display adapters also switched over to PCI.

The most recent development has been the AGP port on Pentium II systems. PCI supports a 4 byte transfer transfer with every tick of the 33 MHz clock, for a total of 133 megabytes per second. AGP supports a 4 byte data transfer twice in every cycle of the 66 MHz mainboard (CPU-memory) clock. Thats four times the throughput of the PCI bus.

PCI is fast enough for Windows business applications. AGP will only produce a noticable improvement if it is used for 3D Engineering applications. That is why it is currently available only on Pentium II systems.

Chapter 4 IDE or SCSI Disk

Summary: Modern computers come with EIDE (enhanced IDE) built into the mainboard. This is perfectly adequate for personal workstations. A high performance SCSI controller can be added to a new system for an extra $220. IDE and SCSI disks operate at the same speed, but SCSI has advantages for a multitasking server because it allows many devices to be performing operations at the same time.

The hard disk has one or more metal platters coated top and bottom with a magnetic material similar to the coating on a VCR magnetic tape. In the VCR the tape moves by a fixed recording and sensing device (the "head"). With a disk, the head is connected to an "arm" which is moved in and out along a radius of the disk circle. To read or write information, the computer or disk controller must figure out where the data is on the disk. The arm is then moved the correct distance, then it waits until the

location on the disk where the data is located rotates around to the point where it passes under the head and can be transferred. The surface of the disk is preformatted into units that hold 512 byte of data (the "sectors").

In the first generation of PC's, the electronics to move the arm, position, and to control the recording or sensing was placed in a separate controller card. Advances in chip technology allow this function to be done by logic on the disk which can be more easily tuned at the factory to the special features of each type of device. Today there are two technologies, IDE and SCSI.

4-1. IDE (Standard on Desktop PCs)

IDE (Integrated Disk Electronics) is the least expensive current disk technology. IDE support is usually built into the mainboard, though it is also possible to get an interface card for the ISA bus for around $30. An IDE disk is connected to the mainboard or interface card through a flat "ribbon" cable. Rather than invent a new interface, the signals in the IDE cable simply duplicate the activity on the ISA bus itself.

After Nov, 1994 vendors started to ship systems with Enhanced IDE (EIDE). Classic IDE supported two hard disks of 528 megabyte or less. EIDE allows four devices, including a mixture of disks, tapes, and CD-ROM, and the hard disks can be larger.

An IDE interface cable has two plugs and can be attached to two devices. The first device acts as the master, and the second device acts as a slave. This interface is busy if either device is processing a request, so activity on one device blocks access to the other. It will generally be necessary when adding a new disk to a system to set a switch or connector on the disk to indicate if it is to function as master or slave.

When they designed the EIDE standard, they needed compatibility with all the existing IDE devices. So they didn't change the rules on the cable. An EIDE interface chip can support four devices, but it has two interface cables each connecting two devices. The EIDE chip looks and acts like two IDE chips. An old IDE disk can be connected to a new EIDE connector.

However, a new large EIDE disk cannot always be connected to an old PC. The original IBM programming interface limited the disk space to 528 megabytes (not a big problem when hard disks had 10 or 20 megs). Today there are 1 gig disks advertised for little more than $200. However, an old IDE disk interface chip may not support data beyond the first 528 megs. You can buy a new interface card for $40, but even then the BIOS on old systems will not support I/O to partitions that extend beyond 528 megs. You may need to load a new operating system (Windows 95, OS/2, or Windows NT) and the partitions containing the operating system files may have to reside completely within the first 528 megs of the disk.

Computers built in the last year should come with Extended IDE (EIDE). The extensions overcome limits in the original IDE design:

*IDE supports only disks. EIDE supports a mixture of disks, tapes, and CDROM drives.
*IDE supports only two devices. EIDE supports up to four devices on the same controller chip although it uses two cables.
*EIDE allows disks up to 1 gigabyte. Larger disks may also work, but that is up to the vendor. IBM, for example, doesn't officially support EIDE disks larger than one gig.

Since EIDE simulated two separate IDE interface chips, there is an optimization that many customers do not fully appreciate. Newer operating systems (OS/2, Windows NT, and even Windows 95 to some extent) permit more than one I/O request to be running at a time. When a program wants to read something from a disk, the request is given to the disk interface and another program is allowed to run while the first program waits for data. However, the IDE interface allows only one of the two disks connected to the same cable to be active at a time, and any request to use the second disk will be blocked while data is being read from the first disk. An EIDE interface duplicates this IDE restriction, but since the EIDE chip looks like two IDE devices, a request can be made through the second interface while the first interface is busy.

If you run plain old DOS and Windows 3.x, it doesn't matter. Those systems will wait for any operation to complete before running any other program. However, if you are running a new system, and if you purchase a second IDE hard disk, then there is a performance advantage to putting the second drive on the second interface cable (managed by the second simulated IDE "device") rather than connecting it to the same flat disk interface to which the first disk is connected. On separate cables, the two disks can be active at the same time.

However, if you have two hard disks and an EIDE CD-ROM, then it is best to put the two disks on the same cable and isolate the CD-ROM on the second cable. A CD-ROM is much slower than a hard disk, and it will be busy longer. If it is on the same cable with a hard disk, it will block access to that disk when any request is made. Unless it is used very infrequently, the best performance will probably be provided by isolating the slow CD-ROM on its own cable.

4-2. SCSI (For Servers and Power Users)

SCSI provides a standard interface for all types of computers. The IDE disk and the ISA bus are peculiar to IBM-compatible Intel-compatible PC machines. SCSI, however, is used by Macintosh computers, RISC workstations, minicomputers, and even some mainframes. SCSI has always supported a mixture of disks, tapes, and CD-ROM drives. While EIDE disks may go up to one gigabyte, SCSI disks are available with 4 to 9 gigabytes of storage.

SCSI is a bus. In the SCSI architecture, the PC (or more precisely, the SCSI adapter card in the PC) is just one device on the bus. Each device is a "peer" of the other devices. In theory, a tape drive could send commands to the PC. In practice, the tape drive isn't smart enough and the PC doesn't respond to commands anyway.

In the Classic SCSI bus, there are 25 signals, each represented by a pair of wires (50 wires all together). Nine of the wires hold the eight bits plus parity of a byte of data. The other wires carry control functions. Classic SCSI can transfer data up to 5 megabytes per second. The Fast SCSI option of the SCSI-2 standard allows 10 megabytes per second on the same cable. To run faster, a Fast Wide SCSI interface is defined, but it requires more than the usual 50 wire cable.

An IDE disk must be mounted inside the computer. There is no provision for the IDE ribbon cable to run to external devices. SCSI devices can also be internal. They are connected to each other and to the adapter card using a flat ribbon cable with 50 wires (OK) or a round bundled cable with 25 twisted pairs of wires (Better).

However, SCSI devices can also be external to the computer. They can be mounted in individual boxes, or can be mounted together in larger tower enclosures. The adapter card is connected to external

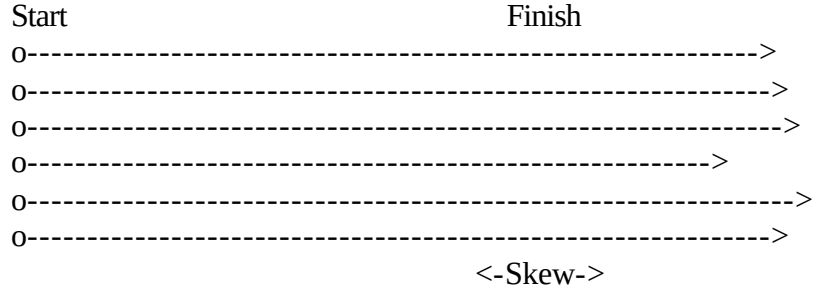
SCSI devices with a round cable containing 25 twisted pairs of wires. Four external SCSI plugs are in common use:

Apple has a cheezy, sleazy 25 pin "D Shell" connector on the back of each Macintosh. This is the same type of plug used on a modem. Obviously you cannot connect 50 wires to 25 pins. The sleazy part is that Apple takes all 25 ground wires and combines them. This eliminates the protection that would be provided by a proper cable with 25 twisted pairs. Older SCSI devices use the 50 contact Centronix interface (similar to the cable that plugs into a PC printer. This was the standard in the early years of SCSI, but it is bulky and has fallen out of favor. Newer devices use a smaller special SCSI connector with 2 rows of 25 pins. The newer SCSI-2 standard specifies this as the preferred connector. This is frequently known as a 50 pin mini-"D" shell or MDS50 connector. On serious large server machines there may be Fast-Wide SCSI cards. They can provide higher data transfer rates, but they may require special cables since they have more than 50 wires.

4-3. 1394 (Firewire)

A new standard has just been adopted which has many advantages over IDE or SCSI. It will become important in future desktop systems as soon as the vendors figure out how to manufacture adapters and devices at low cost.

In both IDE and SCSI, the adapter or controller sends a signal to the device over a cable with a separate wire for each bit of data. Data is transferred one or two bytes at a time. This is called a parallel interface. Ultimately, a parallel connection is limited by skew:



At the beginning of a 100 meter dash, the runners all line up evenly at the starting line. But they cross the finish in a ragged pattern. Electrons rush down a wire at almost the speed of light. However, when there are several wires running any distance between several devices, the signals don't all get to the end of the wire at exactly the same time. This is called skew.

Part of the problem is irregularities in the wire itself. One wire may be thicker than another, and differt batches of copper have different amounts of impurities. Part of the problem comes from differences in individual connectors every time the signal passes through an attached device.

Skew limits how tightly the bytes can be packed on the cable. All of the bits for one byte must arrive at the end of the cable before any bits for the next byte can arrive. Engineers have to design for the worst case, so they generally leave a big gap between bytes. As the cable gets longer and the number of attached devices grows, the gap has to get wider and the speed of the data goes down. Worse, because skew is cased by low tech issues (refining copper, making wire, and building cheap connectors) it is not possible to solve this problem by building better computer chips.

Advances in chip design now make it possible to pack successive bits very tightly on a pair of copper wires. The cable problems that produce skew don't affect this. Variations in the wire may cause electrons to slow down in one section of the wire, but then the next bit will also slow down at the same point. Today it is generally possible to reliably transmit more than eight bits of data on a pair of wires inside the gap required to solve the skew problem when wires run in parallel. In computer jargon, it is better to use a serial interface (one pair of wires) than a parallel interface (eight pair of wires).

Over a short distance, and in a controlled environment, the parallel signal arrangement is best. The CPU, memory, and I/O bus structures use a parallel signal. Even the cable from the EIDE controller to the devices is short and internal to the machine and works well in parallel.

The advantage of SCSI, however, has been the ability to connect to external disk and tape devices over several meters of cable. An external SCSI cable is bulky and fairly expensive. Unplugging a SCSI device when the power is on can crash the system.

Firewire (1394) is a serial alternative to the parallel SCSI connection. It was originally developed by Apple Computers, but was then proposed and accepted as a new international standard. Firewire uses two pair of wires for signals in both directions and one pair of wires to provide power to external equipment. Up to 63 devices can be connected in a loop or from the spokes of a hub. Devices can be plugged and unplugged when the system is running. Although the physical connection is different, Firewire uses the same commands as SCSI and may not require major reprogramming for existing device drivers.

Firewire is designed to be simple and low cost. In one of its more visible decisions, it connects to devices using a plug that was originally developed for the Nintendo GameBoy. Data transfer is as fast as SCSI, but the interface is much simpler.

Microsoft has proposed Firewire as part of its SIPC initiative. The SIPC computer comes in a simple basic configuration with no internal card slots or options. The system is extended by connecting external option boxes like hard disks. Unless unexpected problems develop in the engineering, computers should begin to appear with 1394 connectors sometime in early 1997.

Firewire is specifically designed for desktop and laptop systems. The same laws of physics may drive the redesign of connections for large dedicated file and disk servers, but industrial strength applications may deserve stronger engineering than the Nintendo GameBoy plug. IBM has proposed something called SSA. It has the same basic idea, but can run faster, support more devices, and provide better error recovery than Firewire. IBM uses it on its RS/6000 machines, but it is not clear at this point if it will be accepted for use by high end PC Servers.

4-4. Summary

EIDE comes standard with any modern computer. The interface is built into the mainboard and requires no slots. SCSI requires an adapter card that may cost an additional $200. EIDE disks may be slightly cheaper than 1-2 gig SCSI disks. SCSI tends to dominate the larger 4-9 gig sizes.

There is no real performance difference. EIDE can be faster on paper than the SCSI bus (because SCSI is engineered for a long external cable with lots of skew) but neither bus is likely to limit the performace of a desktop system.

EIDE is not a choice for devices that don't start out as PC's. Mac systems, RISC workstations, and minicomputers use SCSI. SCSI also provides for the external connection of devices ("Zip" removable high capacity drives, writable CD-ROMs units, backup tape units).

SCSI is worth the extra cost in a Server. EIDE supports two separate I/O operations to two disks (on the two different interface cables). SCSI allows all of the disk devices to be active simultaneously. Of course, only one device can be transferring data on the SCSI cable at any given time. However, a disk spends most of its time moving the arm to the right location and waiting for the data to rotate around to the point where it can be read or written. A SCSI controller can have all of its disks moving into position while one disk is actively transferring data.

It goes without saying that a good disk interface on a modern server will connect to the PCI bus. The ISA bus is unreasonably slow (by modern terms), the Microchannel is expensive, and EISA is now obsolete. However, there are both IDE and SCSI adapters that interface to the PCI bus.

An EIDE adapter will always be dumb and cheap. A SCSI adapter can be smart enough to Busmaster. As a Busmaster, the SCSI card can transfer data to or from buffers in memory directly. This frees up the CPU to do other things. To get the full benefit, the computer must be running an operating system (Windows NT, OS/2, Netware, or Unix) that can take advantage of the full capability of the card. Vendors may offer a slightly lower price on a SCSI adapter card that doesn't Busmaster, but the $30 savings are not worth it if the machine will act as a server or will run a multitasking operating system like Windows NT.