Advances in Environmental Biology, 6(3): 1258-1265, 2012
ISSN 1995-0756

This is a refereed journal and all articles are professionally screened and reviewed **ORIGINAL ARTICLE**

Computer Science and Technology

Behzad Mahjour Shafiei, Mohammad Amin Saeed, Farshid Iranmanesh, Fariborz Iranmanesh

Kerman Branch, Islamic Azad University, Kerman, Iran

Behzad Mahjour Shafiei, Mohammad Amin Saeed, Farshid Iranmanesh, Fariborz Iranmanesh; Computer Science and Technology

ABSTRACT

Computer engineering involves the design and implementation of all aspects of computer systems. It is the practical complement to computer science, which focuses on the study of the theory of the organization and processing of information. Because hardware requires software (particularly operating systems) in order to be useful, computer engineering overlaps into software design, although the latter is usually considered to be a separate field

Key words: wimax, Microwave, Vpn, Voip, Backhaul, Wi-fi, Qos

Introduction

To get an idea of the scope of computer engineering, consider the range of components commonly found in today's desktop computers:

Processor:

The design of the microprocessor includes the number and width of registers, method of instruction processing (pipelining), the chipset (functions to be integral to the package with the microprocessor), the amount of cache, the connection to memory bus, the possible use of multiple processors, the order in which data will be moved and stored in memory (low or high-order byte first?), and the clock speed.

Memory:

The design of memory includes the type (static or dynamic) and configuration of RAM, the maximum addressable memory, and the use of parity for error detection. Besides random-access memory, other types of memory include ROM (read-only memory) and CMOS (rewritable persistent memory).

Motherboard:

The motherboard is the platform and data transfer infrastructure for the computer system. It includes the main data bus and secondary buses (such as for high-speed connection between the processor and video subsystem—see bus). The designer must also decide which components will be integral to the motherboard, and which provided as add-ons through ports of various kinds.

Peripheral Devices:

Peripheral devices include fixed and removable disk drives; CD and DVD-ROM drives, tape drives, scanners, printers, and modems.

Device Control:

Each peripheral device must have an interface circuit that receives commands from the CPU and returns data.

Input/Output And Ports:

A variety of standards exist for connecting external devices to the motherboard. Designers of devices in turn must decide which connections to support.
There are also a variety of input devices to be handled, including the keyboard, mouse, joystick, track pad, graphics tablet, and so on.

Of course this discussion isn't limited to the desktop PC; similar or analogous components are also used in larger computers.

Networking:

Networking adds another layer of complexity in controlling the transfer of data between different computer systems, using various typologies and transport mechanisms (such as Ethernet); interfaces to connect computers to the network; routers, hubs, and switches.

Corresponding Author

Behzad Mahjour Shafiei, Computer software engineering student, Islamic Azad University, Kerman Branch, Kerman, Iran.
E-mail: B_m_Shafiei@hotmail.com

Other considerations:

In designing all the subsystems of the modern computer and network, computer engineers must consider a variety of factors and tradeoffs. Hardware devices must be designed with a form factor (size and shape) that will fit efficiently into a crowded computer case. For devices that require their own source of power, the capacity of the available power supply and the likely presence of other power-consuming devices must be taken into account. Processors and other circuits generate heat, which must be dissipated. Heat and other forms of stress affect reliability. And in terms of how a device processes input data or commands, the applicable standards must be met. Finally, cost is always an issue.

Moving beyond hardware to operating system (OS) design, computer engineers must deal with many additional questions, including the file system, how the OS will communicate with devices (or device drivers), and how applications will obtain data from the OS (such as the contents of input buffers). Today's operating systems include hundreds of system functions. Since the 1980s, the provision of all the objects needed for a standard user interface (such as windows, menus, and dialog boxes) has been considered to be part of the OS design.

Trends:

In the early days of mainframe computing (and again at the beginning of microcomputing) many distinctive system architectures entered the market in rapid succession. For example, the Apple II (1977), IBM PC (1981), and Apple Macintosh (1984). Because architectures are now so complex (and so much has been invested in legacy hardware and software), wholly new architectures seldom emerge today. Because of the complexity and cost involved in creating system architectures, development tends to be incremental, such as adding PCI card slots to the IBM PC architecture while retaining older ISA slots, or replacing IDE controllers with EIDE.

The growing emphasis on networks in general and the Internet in particular has probably diverted some effort and resources from the design of stand-alone PCs to network and telecommunications engineering. At the same time, new categories of personal computing devices have emerged over the years, including the suitcase-size "transportable" PC, the laptop, the book-sized notebook PC, the handheld PDA (personal digital assistant), as well as network-oriented PCs and "appliances."

As computing capabilities are built into more traditional devices (ranging from cars to home entertainment centers), computer engineering has increasingly overlapped other fields of engineering and design. This often means thinking of devices in nontraditional ways: a car that is able to plan travel, for example, or a microwave that can keep track of nutritional information as it prepares food. The computer engineer must consider not only the required functionality but the way the user will access the functions.

Motherboard:

Large computers generally had separate large cabinets to hold the central processing unit (CPU) and memory. Personal computers, built in an era of integrated electronics, use a single large circuit board to serve as the base into which chips and expansion boards are plugged. This base is called the motherboard.

The motherboard has a special slot for the CPU. Data lines connect the CPU to RAM and various device controllers. Besides compactness, use of a motherboard minimizes the use of possibly fragile cable connections. It also provides expansion capability. Assuming its pins are compatible with the slot and it is operationally compatible, a PC user can plug a more powerful processor into the slot on the motherboard, upgrading performance. Memory expansion is also provided using a row of memory sockets. Memory, originally inserted as rows of separate chips plugged into individual sockets, is now provided in single modules called DIMMs that can be easily slid into place.

The motherboard also generally includes about six general-purpose expansion slots. These follow two different standards, ISA (industry standard architecture) and PCI (peripheral component interconnect) with PCI now predominating. These slots allow users to mix and match such accessories as graphics (video) cards, disk controllers, and network cards. Additionally, the motherboard includes a chip that stores permanent configuration settings and startup code, a battery, a system clock, and a power supply.

The most important factors in choosing a motherboard are the type and speeds of processor it can accommodate, the bus speed, the BIOS, system chipset, memory and device expansion capacity, and whether certain features (such as video) are integrated into the motherboard or provided through plug-in cards. Generally, users must work within the parameters of their system's motherboard, although knowledgeable people who like to tinker can buy a motherboard and build a system "from scratch" or keep their current peripheral components and upgrade the motherboard.

Spacer Large computers generally had separate large cabinets to hold the central processing unit (CPU) and memory. Personal computers, built in an era of integrated electronics, use a single large circuit board to serve as the base into which chips and expansion boards are plugged. This base is called the motherboard.

The motherboard has a special slot for the CPU. Data lines connect the CPU to RAM and various device controllers. Besides compactness, use of a motherboard minimizes the use of possibly fragile cable connections. It also provides expansion capability. Assuming its pins are compatible with the slot and it is operationally compatible, a PC user can plug a more powerful processor into the slot on the motherboard, upgrading performance. Memory expansion is also provided using a row of memory sockets. Memory, originally inserted as rows of separate chips plugged into individual sockets, is now provided in single modules called DIMMs that can be easily slid into place.

The motherboard also generally includes about six general-purpose expansion slots. These follow two different standards, ISA (industry standard architecture) and PCI (peripheral component interconnect) with PCI now predominating. These slots allow users to mix and match such accessories as graphics (video) cards, disk controllers, and network cards. Additionally, the motherboard includes a chip that stores permanent configuration settings and startup code, a battery, a system clock, and a power supply.

The most important factors in choosing a motherboard are the type and speeds of processor it can accommodate, the bus speed, the BIOS, system chipset, memory and device expansion capacity, and whether certain features (such as video) are integrated into the motherboard or provided through plug-in cards. Generally, users must work within the parameters of their system's motherboard, although knowledgeable people who like to tinker can buy a motherboard and build a system "from scratch" or keep their current peripheral components and upgrade the motherboard.spacer

Operating System

An operating system is an overarching program that manages the resources of the computer. It runs programs and provides them with access to memory (RAM), input/output devices, a file system, and other services. It provides application programmers with a way to invoke system services, and gives users a way to control programs and organize files.

DEVELOPMENT

The earliest computers were started with a rudimentary "loader" program that could be used to configure the system to run the main application program. Gradually, a more sophisticated way to schedule and load programs, link programs together, and assign system resources to them was developed.
As systems were developed that could run more than one program at a time, the duties of the operating systems became more complex. Programs had to be assigned individual portions of memory and prevented from accidentally overwriting another program's memory area. A technique called virtual memory was developed to enable a disk drive to be treated as an extension of the main memory, with data "swapped" to and from the disk as necessary. This enabled the computer to run more and/or larger applications. The operating system, too, became larger, amounting to millions of bytes worth of code.
During the 1960s, time sharing became popular particularly on new smaller machines such as the DEC PDP series, allowing multiple users to run programs and otherwise interact with the same computer. Operating systems such as Multics and its highly successful offshoot UNIX developed ways to assign security levels to files and access levels to users. The UNIX architecture featured a relatively small kernel that provides essential process control, memory management, and file system services, while drivers performed the necessary low-level control of devices and a shell provided user control.

Starting in the late 1970s, the development of personal computers recapitulated in many ways the earlier evolution of operating systems in the mainframe world. Early microcomputers had a program loader in read-only memory (ROM) and often rudimentary facilities for entering, running, and debugging assembly language programs.

During the 1980s, more complete operating systems appeared in the form of Apple DOS, CP/M, and MS-DOS for IBM PCs. These operating systems provided such facilities as a file system for floppy or hard disk and a command-line interface for running programs or system utilities. These systems could run only one program at a time (although exploiting a little-known feature of MS-DOS allowed additional small programs to be tucked away in memory).

As PC memory increased from 640K to multiple megabytes, operating systems became more powerful. Apple's Macintosh operating system and Microsoft Windows could manage multiple tasks. Today personal computer operating systems are comparable in sophistication and capability to those used on mainframes. Indeed, PCs can run UNIX variants such as the popular Linux.

COMPONENTS

While the architecture and features of operating systems differ considerably, there are general functions common to almost every system. The "core" functions include "booting" the system and initializing devices, process management (loading programs intro memory assigning them a share of processing time), and allowing processes to communicate with the operating system or one another. Multiprogramming systems often implement not only processes (running programs) but also threads, or sections of code within programs that can be controlled separately.

A memory management scheme is used to organize and address memory, handle requests to allocate memory, free up memory no longer being used, and rearrange memory to maximize the useful amount.

There is also a scheme for organizing data created or used by programs into files of various types. Most operating systems today have a hierarchical file system that allows for files to be organized into directories or folders that can be further subdivided if necessary. In operating systems such as UNIX, other devices such as the keyboard and screen (console) and printer are also treated like files, providing consistency in programming. The ability to redirect input and output is usually provided. Thus, the output of a program could be directed to the printer, the console, or both.

In connecting devices such as disk drives to application programs, there are often three levels of control. At the top level, the programmer uses a library function to open a file, write data to the file, and close the file. The library itself uses the operating system's lower-level input/output (I/O) calls to transfer blocks of data. These in turn are translated by a driver for the particular device into the low-level instructions needed by the processor that controls the device. Thus, the command to write data to a file is ultimately translated into commands for positioning the disk head and writing the data bytes to disk.

Operating systems, particularly those designed for multiple users, must also manage and secure user accounts. The administrator (or sometimes, ultimately, the "super user" or "root") can assign users varying levels of access to programs and files. The owners of files can in turn specify whether and how the files can be read or changed by other users.
In today's highly networked world most operating systems provide basic support for networking protocols such as TCP/IP. Applications can use this facility to establish network connections and transfer data over the local or remote network.
The operating system's functions are made available to programmers in the form of program libraries or an application programming interface (API).

The user can also interact directly with the operating system. This is done through a program called a shell that accepts and responds to user commands. Operating systems such as MS-DOS and early versions of UNIX accepted only typed-in text commands. Systems such as Microsoft Windows and UNIX (through facilities such as XWindows) allow the user to interact with the operating system through icons, menus, and mouse movements. Application programmers can also provide these interface facilities through the API. This means that programs from different developers can have a similar "look and feel," easing the learning curve for users.
ISSUES AND TRENDS

As the tasks demanded of an operating system have become more complex, designers have debated the best overall form of architecture to use. One popular approach, typified by UNIX, is to use a relatively small kernel for the core functions. A community of programmers can then write the utilities needed to manage the system, performing tasks such as listing file directories, editing text, or sending email. New releases of the operating system then incorporate the most useful of these utilities. The user also has a variety of shells (and thus interfaces) available.
The kernel approach makes it relatively easy to port the operating system to a different computer platform and then develop versions of the utilities. (Kernels were also a necessity when system memory was limited and precious, but this consideration is much less important today.)
Designers of modern operating systems face a number of continuing challenges:

* Security, in a world where nearly all computers are networked, often continuously
* The tradeoff between powerful, attractive functions such as scripting and the security vulnerabilities they tend to present

* The need to provide support for new applications such as streaming audio and video * Ease of use in installing new devices * The continuing development of new user interface concepts, including alternative interfaces for the disabled and for special applications * The growing use of multiprocessing and multiprogramming, requiring coordination of processors sharing memory and communicating with one another * Distributed systems where server programs, client programs, and data objects can be allocated among many networked computers, and allocations continually adjusted or balanced to reflect demand on the system * The spread of portable, mobile, and hand-held computers and computers embedded in devices such as engine control systems. Sometimes the choice is between devising a scaled-down version of an existing operating system and designing a new OS that is optimized for devices that may have limited memory and storage capacity.


Microprocessor:

From: Encyclopedia of Computer Science and Technology.

Spacer:

A microprocessor is an integrated circuit chip that contains all of the essential components for the central processing unit (CPU) of a microcomputer system such as a personal computer.

Microprocessor development began in the 1960s when a new company called Intel was given a contract to develop chips for programmable calculators for a Japanese firm. Marcian E. "Ted" Hoff headed the project. He decided that rather than hard-wiring most of the calculator logic into the chips, he would create a general-purpose chip that could read instructions and data, perform basic arithmetic and logical functions, and transfer data between memory and internal locations called registers.

The resulting microprocessor, when combined with some RAM (random-access memory), some preprogrammed ROM (Read Only Memory), and an input/output (I/O) chip constituted a tiny but complete CPU, soon dubbed "a computer on a chip." This first microprocessor, the Intel 4004, had only a few thousand transistors, could handle data only 4 bits at a time, and ran at 740 KHz (about one three-thousandth the speed of the latest Pentium IV chips).

Intel gradually refined the chip, giving it the logic circuits to enable it to perform additional instructions, more internal stack and register space, and 8 KB of space to store programs. The 8008 could handle 8 bits of data at a time, while the 8080 became the first microprocessor that was capable of serving as the CPU for a practical microcomputer system. Its descendants, the 8088 and 8086 (16-bit) powered industry-standard IBM-compatible PCs. Meanwhile, other companies such as Motorola (68000), Zilog (Z-80), and MOS Technology (6502) powered competing PCs from Apple, Atari, Commodore, and others.

With the dominance of the IBM PC and its clones, the Intel 80 x 86 series in turn dominated the microprocessor market. (The x refers to successive digits, as in 80286, 80386, and 80486.) At the next level this nomenclature was replaced by the Pentium series, which is up to the Pentium 4 as of 2002.

According to a famous dictum called Moore's Law, the density (number of transistors per cubic area) and speed (in terms of clock rate) of microprocessors has roughly doubled every 18 months to two years. Intel expects to be making microprocessors with 1 billion transistors by 2007.

Microprocessor And Microcomputer:

A microcomputer is a system consisting of a microprocessor and a number of auxiliary chips. The microprocessor chip serves as the central processing unit (CPU). It contains a clock that regulates the flow of data and instructions (each instruction takes a certain number of clock cycles to execute). There is also an index register that keeps track of the instruction being executed. A small number of locations called registers within the CPU allow for storing or retrieving the data being used by instructions much more quickly than retrieval from main memory (RAM).

Typically, the instruction register advances to the next instruction. The instruction is fetched, decoded, and sent to the CPU's ALU (arithmetic logic unit) for processing. Data needed to be processed by the instruction are either fetched from a register or, through an address register, fetched from RAM. (Some processors store one operand for an arithmetic operation in a special register called the accumulator.)

Floating-point operations (those involving numbers that can include decimal points) require special registers that can keep track of the decimal position. Until the mid-1990s, many systems used a separate microprocessor called a coprocessor to handle floating point operations. However, later chips such as the Pentium series integrate floating point operations into the main chip.

In order to function as the heart of a microcomputer, the CPU must communicate with a variety of other devices by interacting with special controller chips. For example, there is a bus interface chip that decodes memory addresses and routes

requests to the appropriate devices on the motherboard. When data is requested from memory, a memory controller must physically fetch the data from RAM. There is also a cache controller that interfaces with one or two levels of high-speed cache memory. The algorithms implemented in the cache controller aim to have the next instructions and the most-likely needed data already in the cache when the CPU requests them.

Other devices such as disk drives, modems, printers, and video cards are all connected to the CPU through input/output (I/O) interfaces that connect to the system bus. Most of the devices connected to the bus have their own microprocessors. Software translates high-level programming instructions (such as to open a file) to the appropriate device commands.

The CPU and many other devices also contain ROM (read only memory) chips that have permanent basic instructions stored on them. This enables the CPU and other devices to perform the necessary actions to enter into communication when the system starts up.

New Features Emerge:

Improvements in microprocessors during the 1980s included wider data paths and the ability to address a larger amount of memory. For example, the Intel 80386 was the first 32-bit processor for PCs and could address 4GB of memory. (Earlier processors such as the 80286 had to divide memory into segments or use paging to swap memory in and out of a smaller space to make it look like a larger one.) Over the years microprocessors tended to add more built-in cache memory, enabling them to have more instructions or data ready for immediate use.

Another way to get more performance out of a microprocessor is to increase the speed with which instructions can be executed. One technique, called pipelining, breaks the processor into a series of segments, each of which can execute a particular operation. Instead of waiting until an instruction has been completely executed and then turning to the next one, a pipelined microprocessor moves the instruction from segment to segment as its operations are executed, with following instructions moving into the vacated segments. As a result, two or more instructions can be undergoing execution at the same time.

In addition to pipelining, the Pentium series and other recent chips can have instructions executing simultaneously using different arithmetic logic units (ALUs) or floating-point units (FPUs).

Another way to improve instruction processing is to use a simpler set of instructions. First introduced during the 1980s for minicomputers and high-end workstations (such as the Sun SPARC series), reduced instruction set computer (RISC) chips have smaller, more uniform instructions that can be more easily pipelined, as well as many registers for holding the results of the intermediate processing. During the 1990s, RISC concepts were also adopted in PC processor designs such as the 80486 and Pentium.

In the new century, it is unclear when physical limitations will eventually slow down the tremendous rate of increase in microprocessor power. As the chips get denser and smaller, more heat is generated with less surface through which it can be removed. At still greater densities, quantum effects may also begin to be a problem. On the other hand, new technologies might take the elements of the processor down to a still smaller level.

While the stand-alone desktop, laptop, or handheld computer is the most visible manifestation of the micro- processing revolution, there are probably hundreds of "invisible" microprocessors in use for every visible computer. Today microprocessors help monitor and control everything from home appliances to cars to medical devices

Cache

A basic problem in computer design is how to optimize the fetching of instructions or data so that it will be ready when the processor (CPU) needs it. One common solution is to use a cache. A cache is an area of relatively fast-access memory into which data can be stored in anticipation of its being needed for processing. Caches are used mainly in two contexts: the processor cache and the disk cache.

CPU CACHE

The use of a processor cache is advantageous because instructions and data can be fetched more quickly from the cache (static memory chips next to or within the CPU) than they can be retrieved from the main memory (usually dynamic RAM). An algorithm analyzes the instructions currently being executed by the processor and tries to anticipate what instructions and data are likely to be needed in the near future. (For example, if the instructions call for a possible branch to one of two sets of instructions, the cache will load the set that has been used most often or most recently. Since many programs loop over and over again through the same instructions until some condition is met, the cache's prediction will be right most of the time.)

These predicted instructions and data are transferred from main memory to the cache while the processor is still executing the earlier instructions. If the cache's prediction was correct, when it is time to fetch these instructions and data they are already waiting in the high-speed cache memory. The result is an effective increase in the CPU's speed despite there being no increase in clock rate (the rate at which the processor can cycle through instructions).

The effectiveness of a processor cache depends on two things: the mix of instructions and data being processed and the location of the cache memory. If a program uses long sequences of repetitive instructions and/or data, caching will noticeably speed it up. A cache located within the CPU itself (called an L1 cache) is faster (albeit more expensive) than an L2 cache, which is a separate set of chips on the motherboard.

Changes made to data by the CPU are normally written back to the cache, not to main memory, until the cache is full. In multiprocessor systems, however, designers of processor caches must deal with the issue of cache coherency. If, for example, several processors are executing parts of the same code and are using a shared main memory to communicate, one processor may change the value of a variable in memory but not write it back immediately (since its cache is not yet full). Meanwhile, another processor may load the old value from the cache, unaware that it has been changed. This can be prevented by using special hardware that can detect such changes and automatically "write through" the new value to the memory. The processors, having received a hardware or software "signal" that data has been changed, can be directed to reread it.

DISK CACHE

A disk cache uses the same general principle as a processor cache. Here, however, it is RAM (either a part of main memory or separate memory on the disk drive) that is the faster medium and the disk drive itself that is slower. When an application starts to request data from the disk, the cache reads one or more complete blocks or sectors of data from the disk rather than just the data record being requested. Then, if the application continues to request sequential data records, these can be read from the high-speed memory on the cache rather than from the disk drive. It follows that disk caching is most effective when an application, for example, loads a database file that is stored sequentially on the disk.

Similarly, when a program writes data to the disk, the data can be accumulated in the cache and written back to the drive in whole blocks. While this increases efficiency, if a power outage or other problem erases or corrupts the cache contents, the cache will no longer be in synch with the drive. This can cause corruption in a database.

NETWORK CACHE

Caching techniques can be used in other ways. For example, most Web browsers are set to store recently read pages on disk so that if the user directs the browser to go back to such a page it can be read from disk rather than having to be retransmitted over the Internet (generally a slower process). Web servers and ISPs (such as cable services) can also cache popular pages so they can be served up quickly.

Bus:

A computer bus is a pathway for data to flow between the central processing unit (CPU), main memory (RAM), and various devices such as the keyboard, video, disk drives, and communications ports. Connecting a device to the bus allows it to communicate with the CPU and other components without there having to be a separate set of wires for each device. The bus thus provides for flexibility and simplicity in computer architecture.

Mainframe computers and large minicomputers typically have proprietary buses that provide a wide multipath connection that allows for data transfer rates from about 3 MB/sec. to 10 MB/sec or more. This is in keeping with the use of mainframes to process large amounts of data at high speeds.

Microcomputer Buses:

The bus played a key role in the development of the modern desktop computer in the later 1970s and 1980s. In the microcomputer, the bus is fitted with connectors called expansion slots, into which any expansion card that meets connection specifications can be inserted. Thus the S-100 bus made it possible for microcomputer pioneers to build a variety of systems with cards to expand the memory and add serial and parallel ports, disk controllers, and other devices. (The Apple II had a similar expansion

capability.) In 1981, when IBM announced its first PC, it also defined an 8-bit expansion bus that became known as the ISA (Industry Standard Architecture) as other companies rushed to "clone" IBM's hardware.

In the mid-1980s, IBM advanced the industry with the AT (Advanced Technology) machine, which had the 16-bit Intel 80286 chip and an expanded bus that could transmit data at up to 2 MB/sec. The clone manufacturers soon matched and exceeded these specifications, however. IBM responded by trying both to improve the microcomputer bus and to define a proprietary standard that it could control via licensing. The result was called the Micro-Channel Architecture (MCA), which increased data throughput to 20 MB/sec with full 32-bit capability. This bus had other advanced features such as a direct connection to the video system (Video Graphics Array) and the ability to configure cards in software rather than having to set physical switches. In addition, cards could now incorporate their own processors and memory in a way similar to that of their powerful mainframe counterparts (this is called bus mastering). Despite these advantages, however, the proprietary nature of the MCA and the fact that computers using this bus could not use any of the hundreds of ISA cards led to a limited market share for the new systems.

Conclusion:

Instead of paying IBM and adopting the new standard, nine major clone manufacturers joined to develop the EISA (Extended ISA) bus. EISA was also a 32-bit bus, but its maximum transfer rate of 33 MB/sec made it considerably faster than the MCA. It was tailored to the new Intel 80386 and 80486 processors, which supported the synchronous transfer of data in rapid bursts. The EISA matched and exceeded the MCA's abilities (including bus mastering and no-switch configuration), but it also retained the ability to use older ISA expansion cards. The EISA soon became the industry standard as the Pentium family of processors were introduced.

However, the endless hunger for more data-transfer capability caused by the new graphics-oriented operating systems such as Microsoft Windows led to the development of local buses. A local bus is connected to the processor's memory bus (which typically runs at half the processor's external speed rather than the much slower system bus speed), a considerable advantage in moving data (such as graphics) from main memory to the video card.

Two of these buses, the VESA (or VL) bus and the PCI bus came into widespread use in higher-end machines, with the PCI becoming dominant. The PCI bus runs at 33 MHz and supports features such as Plug and Play (the ability to automatically configure a device, supported in Windows 98 and later) and Hot Plug (the ability to connect or reconnect devices while the PC is running). The PCI retains compatibility with older 8-bit and 16-bit ISA expansion cards. At the end of the 1990s, PC makers were starting to introduce even faster buses such as the AGP (accelerated graphics port), which runs at 66 MHz.

Two important auxiliary buses are designed for the connection of peripheral devices to the main PC bus. The older SCSI (Small Computer Systems Interface) was announced in 1986 (with the expanded SCSI-2 in 1994). SCSI is primarily used to connect disk drives and other mass storage devices (such as CD-ROMs), though it can be used for scanners and other devices as well. SCSI-2 can transfer data at 20 MB/sec over a 16-bit path, and SCSI-3 (still in development) will offer a variety of high-speed capabilities. SCSI was adopted as the standard peripheral interface for many models of Apple Macintosh computers as well as UNIX workstations. On IBM architecture PCs SCSI is generally used for servers that require large amounts of mass storage. Multiple devices can be connected in series (or "chained").

The newer USB (Universal Serial Bus) is relatively slow (12 MB/sec) but convenient because devices do not need separate cards to connect to the bus. A simple plug can be inserted directly into a USB socket on the system board or the socket can be connected to a USB hub to which several devices can be connected. In 2002, USB 2.0 is gradually entering the marketplace. It offers 480 MB/sec data transfer speed.

It is uncertain whether the next advance will be the adoption of a 64-bit PCI bus or the development of an entirely different bus architecture. The latter is attractive as a way to get past certain inherent bottlenecks in the PCI design, but the desire for downward compatibility with the huge number of existing ISA, EISA, and PCI devices is also very strong

References

1. Arizona Administrative Code, Title 2, Chapter 7, “Department of Administration Finance Division, Purchasing Office.”
2. Arizona Administrative Code, Title 2, Chapter 10, “Department of Administration Risk Management Section