Computer System Architecture Notes

Introduction

Today, we often take for granted the impressive array of computing machinery that surrounds us and helps us manage our daily lives. Because you are studying computer architecture and digital hardware, you no doubt have a good understanding of these machines, and you've probably written countless programs on your PCs and workstations. However, it is very easy to become jaded and forget the evolution of the technology that has led us to the point where every Nintendo Game Boy has 100 times the computing power of the computer systems on the first Mercury space missions.

A digital computer consists of an interconnected system of processors, memories, and input/output devices. This chapter is an introduction to these three components and to the methods by which computers are interconnected.

Digital Computer

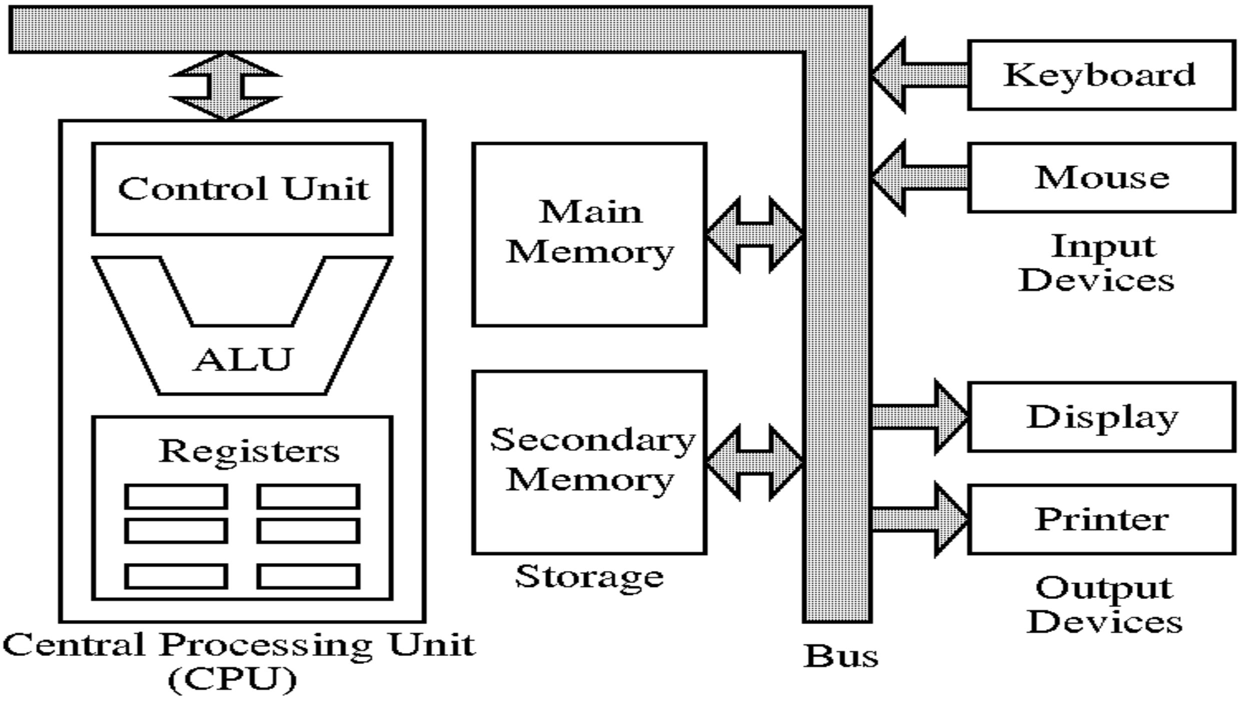
A digital computer is a device for storing and manipulating numbers using a small set of discrete states of a given physical system. It is the opposite of an analog computer, in which quantities are represented using a continuous variable. A watch, with blades rotating smoothly and continuously around the clock face, is a good example of an analog system: any position of the blades is allowed. In contrast, counting using our fingers is an example of a digital process. Only full values are meaningful and we don't count using "half a finger". In fact, the word "digital" comes from "digits". Digital computers count using "digits", i.e. a discrete representation of numbers.

The earliest example of a digital computer is the Analytical Engine, designed by Charles Babbage in England in the nineteenth century, but never completed. The Analytical Engine, stored numbers using rotating wheels. Each of them could represent ten digits, from zero to nine, depending on its angle relative to the rest position. This was the method used to store numbers in mechanical calculators until they were replaced by electronic computers and electronic calculators in the twentieth century.

Modern digital computers operate using the binary system. The only digits needed are "0" and "1", i.e. the binary digits or "bits". Binary components are usually cheaper and easier to manufacture than components for other numerical bases. However, when the first digital computers were built in the USA and in Europe, it was not yet totally clear which technology should be used for the internal design. The American ENIAC and Mark I computers used binary signals but also a decimal internal representation for numbers. They were digital computers with a hybrid numerical base.

Components of a Digital Computer

A Digital computer is a machine that can perform computation. It is difficult to give a precise definition of computation. Intuitively, a computation involves the following components:



1) PROCESSOR (Central processing unit).

It is the “brain” of a computer. The CPU consists of a CU and an ALU. CU - Control Unit: It directs and coordinates the operations of the entire computer. CU fetches the instructions from RAM and stores it in the instruction register.

- ALU - Arithmetic Logic Unit: It performs mathematical operations.
- MICROPROCESSOR: In mainframe computers CU and ALU are both separate units but since 1971 both units have existed on the same chip. This was the beginning of the microprocessor era and PC computers. The quality of the microprocessor performance is measured by several parameters. Two of them are: Clock rate and word size.
- CLOCK RATE: The computer has a master clock that determines the speed at which the microprocessor can execute an instruction. The speed at which the microprocessor completes an instruction execution cycle is measured in MHz - megahertz (Millions of cycles per second) or GHz - gigahertz (Billions of cycles per second).
- WORD SIZE: The number of bits the microprocessor can manipulate at one time varies from computer to computer. A microprocessor with a large word size can process more data in each instruction cycle. (a 32 bits word size computer can process 32 bit data at a time.)

2) MEMORY

Memory is one of the most important elements of every computer. The computer memory is electronic circuitry that holds data and program instructions until it is their turn to be processed. There are three types of computer memory: RAM, CMOS, and ROM.

- RAM - Random Access Memory: It is a temporary holding area for data before and after they are processed. It is like scratch paper - volatile (after the computer is turned off all data is gone). It consists of thousands of circuits that each hold one bit of data. The computer holds (stores) data in RAM and copies it to the CPU and from the CPU (processor) back to RAM. RAM is a holder of data and also instructions. When the processor turns into a "word processor" the instructions for this activity are copied from the disk to RAM. 32 MB RAM means that the RAM can hold 32 million bytes of data.

- CMOS - Complementary Metal Oxide Semiconductor: This is a battery-powered chip that retains data about the computer configuration when the computer is turned off.

- ROM - Read Only Memory: A set of chips containing permanent instructions about the "boot process." Because RAM is empty at the beginning of the processing, the computer must have a separate system for loading all the instructions to RAM. All this is done through the "BOOT PROCESS." The processor performs step by step instructions that are given by ROM. The boot process is therefore a series of instructions from ROM that are preparing the computer to perform its work.

3) BUS:

It is an electronic path that connects the main parts of the computer. The system bus transports data back and forth between the processor and RAM or hard disk drive. Data bus transports data between the processor and parts of the computer. A 32-bit bus transports 32 bits of data at a time. It works like a "normal" bus with 32 seats. (Data bus is a sub-unit of system bus).

4) STORAGE:

None of the computer operations would be useful for practical life if we did not have a place to store it. The main storage computer device is the hard disk drive. Hard disk drive: It consists of one or more platters (flat and rigid disks made of aluminum or glass that are coated with magnetic oxide.) Each platter has a read-write head associated with it. The hard disk drive is formatted into the tracks, cylinders, sectors and groups of sectors. Tracks are concentric circles in a disk that hold the data. Cylinders are stacks of tracks in a disk. Sectors are subdivisions of the platters and tracks (9 sectors each stores 512 bytes) and 40 tracks usually.

Capacity of hard disk drive = cylinders x surfaces x sectors x 512

- Floppy diskette
- CD-ROM, CD-RW
- DVD

- Flash memory

5) IO DEVICES (Input Device & Output Device) :

Input Devices are the devices using which the user provides input instances. In a programmable computer, input devices are also used to input programs. Examples: keyboard, mouse.

Output devices notify the user about the outputs of a computation. Example: screen, printer.

Interconnection Structures

A computer consists of a set of components or modules of three basic types (processor, memory, I/O) that communicate with each other. In effect, a computer is a network of basic modules. Thus, there must be paths for connecting the modules.

The collection of paths connecting the various modules is called the interconnection structure. The design of this structure will depend on the exchanges that must be made between modules.

Figure suggests the types of exchanges that are needed by indicating the major forms of input and output for each module type:

* Memory

* Input/Output

* CPU

The interconnection structure must support the following types of transfers:

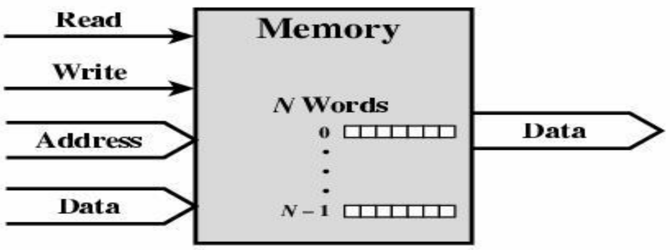
* Memory to processor: The processor reads an instruction or a unit of data from memory.

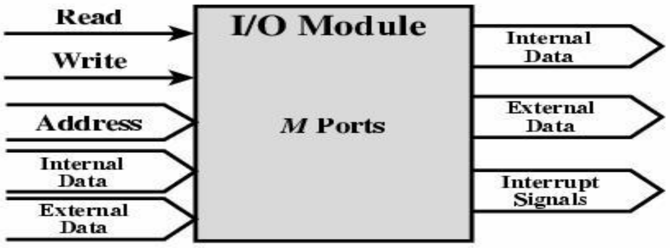
* Processor to memory: The processor writes a unit of data to memory.

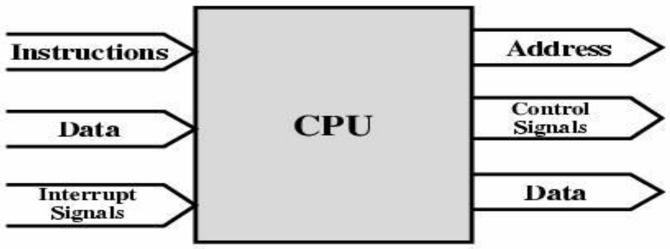
* I/O to processor: The processor reads data from an I/O device via an I/O module.

* Processor to I/O: The processor sends data to the I/O device.

* I/O to or from memory: For these two cases, an I/O module is allowed to







exchange data directly with memory, without going through the processor, using direct memory access (DMA).

Over the years, a number of interconnection structures have been tried. By far the most common is the bus and various multiple-bus structures.

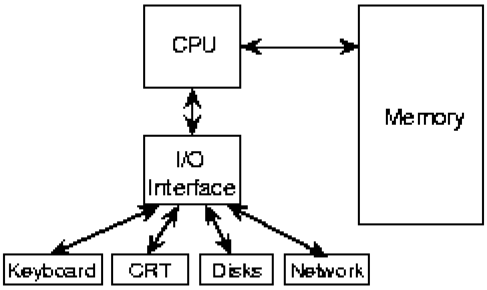
Computer Architecture

What's Computer Architecture?

Computer Architecture is the science and art of selecting and interconnecting hardware components to create computers that meet functional, performance and cost goals. Computer architecture is *not* about using computers to design buildings.

To understand digital signal processing systems, we must understand a little about how computers compute. The modern definition of a *computer* is an electronic device that performs calculations on data, presenting the results to humans or other computers in a variety of (hopefully useful) ways.

Organization of a Simple Computer



Generic computer hardware organization.

The generic computer contains *input* devices (keyboard, mouse, A/D (analog-to-digital) converter, etc.), a *computational unit*, and output devices (monitors, printers, D/A converters). The computational unit is the computer's heart, and usually consists of a *central processing unit* (CPU), a *memory*, and an input/output (I/O) interface.

Thus, computer architecture refers to those attributes of the system that are visible to a programmer Those attributes that have a direct impact on the execution of a program

- *Instruction sets*
- *Data representations*
- *Addressing*
- *I/O*

Computer Organization

Computer Organization refers to the operational units and their interconnections that realize

the architectural specifications. Examples are things that are transparent to the programmer:

- Control signals
- Interfaces between computer and peripherals
- The memory technology being used

Computer organization refers to the operational units and their interconnections that realized the architectural specifications. Examples of architectural attributes include the instruction set, the number of bits used to represent various data type(e,g, numbers and character) , I/O mechanisms, and techniques for addressing mode Organizational attributes include those hardware details transparent to the programmer, such as control signals , interfaces between the computer and peripherals and the memory technology used.

Computer Organization and Architecture

In describing computer system, a distinction is often made between computer architecture and computer organization.

Computer architecture refers to those attributes of a system visible to a programmer, or put another way, those attributes that have a direct impact on the logical execution of a program.

Computer organization refers to the operational units and their interconnection that realize the architecture specification.

Examples of architecture attributes include the instruction set, the number of bit to represent various data types (e.g.., numbers, and characters), I/O mechanisms, and technique for addressing memory.

Examples of organization attributes include those hardware details transparent to the programmer, such as control signals, interfaces between the computer and peripherals, and the memory technology used.

As an example, it is an architectural design issue whether a computer will have a multiply instruction. It is an organizational issue whether that instruction will be implemented by a special multiply unit or by a mechanism that makes repeated use of the add unit of the system. The organization decision may be bases on the anticipated frequency of use of the multiply instruction, the relative speed of the two approaches, and the cost and physical size of a special multiply unit.

Historically, and still today, the distinction between architecture and organization has been an important one. Many computer manufacturers offer a family of computer model, all with the same architecture but with differences in organization. Consequently, the different models in the family have different price and performance characteristics. Furthermore, an architecture may survive many years, but its organization changes with changing technology.

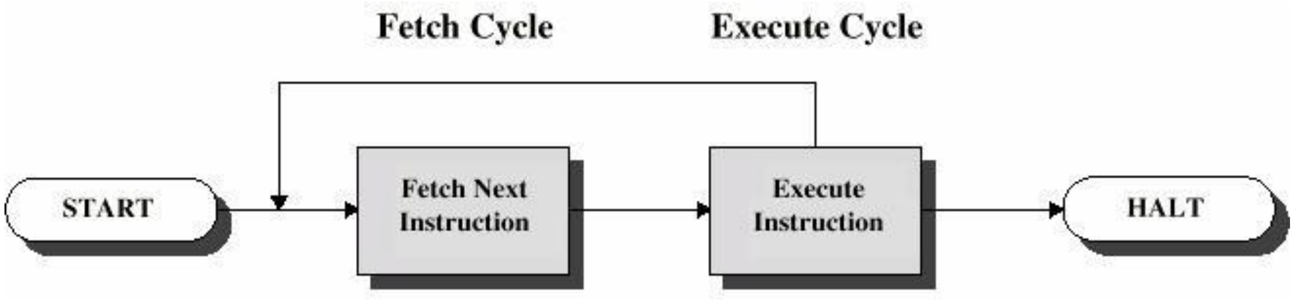
Architecture and organization are independent; you can change the organization of a computer without changing its architecture. For example, a 64-bit architecture can be internally organized as a true 64-bit machine or as a 16-bit machine that uses four cycles to handle 64-bit values.

The difference between architecture and organization is best illustrated by a noncomputer example. Is the gear lever in a car part of its architecture or organization? The architecture of a car is simple; it transports you from A to B. The gear lever belongs to the car's organization because it implements the function of a car but is not part of that function.

Function Of Computer

The basic function performed by a computer is execution of a program, which consists of a set of instructions stored in memory. The processor does the actual work by executing instructions specified in the program. In its simplest form, instruction processing consists of two steps: The processor reads (fetches) instructions from memory one at a time and executes each instruction. Program execution consists of repeating the process of instruction fetch and instruction execution. The Instruction execution may involve several operations and depends on the nature of the instruction.

The processing required for a single instruction is called an instruction cycle. Using the simplified two-step description given previously, the instruction cycle is depicted in Figure



Basic instruction cycle

The two steps are referred to as the fetch cycle and the execute cycle. Program execution halts only if the machine is turned off, some sort of unrecoverable error occurs, or a program instruction that halts the computer is encountered. Thus broadly speaking there are two cycle for execution of instruction:

1. Fetch Cycle
2. Execute Cycle

The total time taken by an instruction to be executed is known as **Instruction Cycle.** Thus it contain fetch-decode-execute sequence. Each computer's CPU can have different cycles based on different instruction sets.

1. Fetching the instruction

The CPU presents the value of the program counter (PC) on the address bus. The CPU then fetches the instruction from main memory via the data bus into the memory data register (MDR). The value from the MDR is then placed into the current instruction register (CIR), a circuit that holds the instruction temporarily so that it can be decoded and executed.

2. Decode the instruction

The instruction decoder interprets and implements the instruction. The instruction register (IR) holds the current instruction, while the program counter (PC) holds the address in memory of the next instruction to be executed.

Fetch data from main memory

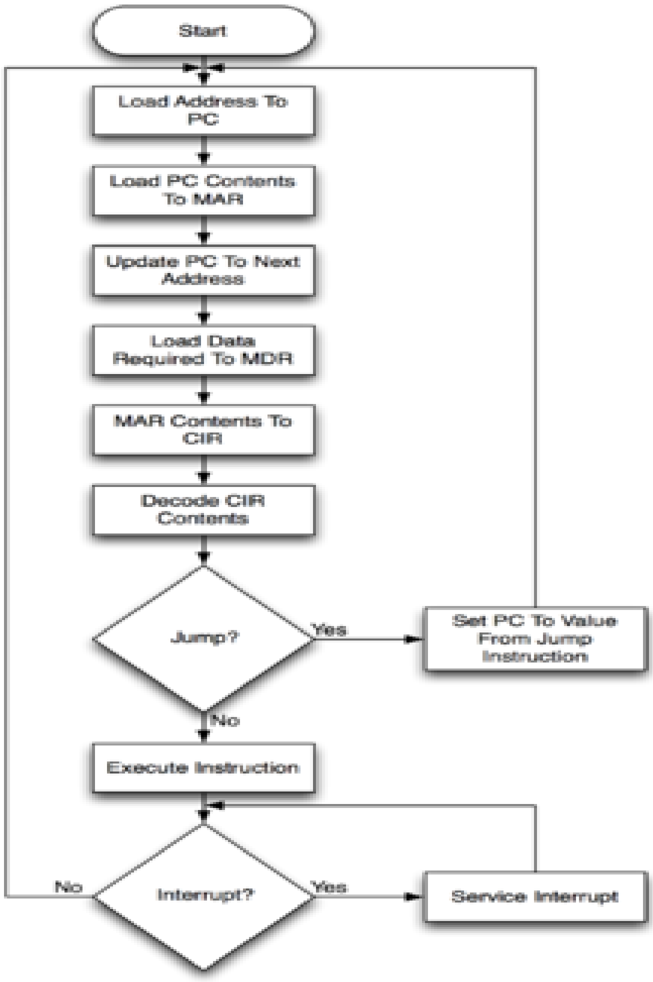
Read the effective address from main memory if the instruction has an indirect address. Fetch required data from main memory to be processed and place it into data registers.

3. Execute the instruction

From the instruction register, the data forming the instruction is decoded by the control unit. It then passes the decoded information as a sequence of control signals to the relevant function units of the CPU to perform the actions required by the instruction such as reading values from registers, passing them to the Arithmetic logic unit (ALU) to add them together and writing the result back to a register. A condition signal is sent back to the control unit by the ALU if it is involved.

4. Store results

Also called write back to memory. The result generated by the operation is stored in the main memory, or sent to an output device. Based on the condition feedback from the ALU, the PC is either incremented to address the next instruction or updated to a different address where the next instruction will be fetched. The cycle is then repeated.



Fetch cycle

Steps 1 and 2 of the Instruction Cycle are called the Fetch Cycle. These steps are the same for each instruction. The fetch cycle processes the instruction from the instruction word which contains an opcode and an operand.

Execute cycle

Steps 3 and 4 of the Instruction Cycle are part of the Execute Cycle. These steps will change with each instruction.

The first step of the execute cycle is the Process-Memory. Data is transferred between the CPU and the I/O module. Next is the Data-Processing uses mathematical operations as well as logical operations in reference to data. Central alterations is the next step, is a sequence of operations, for example a jump operation. The last step is a combined operation from all the other steps.

A diagram of the Fetch Execute Cycle.

The fetch-decode-execute cycle works as follows:

1. For starting the execution of a program, a sequence of machine instructions is copied to the instruction area of the memory. Also some global variables and input parameters are copied to the data area of the memory.
2. A particular control register, called the **program counter** (PC), is loaded with the address of the first instruction of the program.
3. The CPU fetches the instruction from that location in the memory that is currently stored in the PC register.
4. The instruction is decoded in the control unit of the CPU.
5. The instruction may require one or more operands. An operand may be either a data or a memory address. A data may be either a constant (also called an immediate operand) or a value stored in the data area of the memory or a value stored in a register. Similarly, an address may be either immediate or a resident of the main memory or available in a register.
6. An immediate operand is available from the instruction itself. The content of a register is also available at the time of the execution of the instruction. Finally, a variable value is fetched from the data part of the main memory.
7. If the instruction is a data movement operation, the corresponding movement is performed. For example, a "load" instruction copies the data fetched from memory to a register, whereas a "save" instruction sends a value from a register to the data area of the memory.
8. If the instruction is an arithmetic or logical instruction, it is executed in the ALU after all the operands are available in the CPU (in its registers). The output from the ALU is stored back in a register.
9. If the instruction is a jump instruction, the instruction must contain a memory address to jump to. The program counter (PC) is loaded with this address. A jump may be conditional, i.e., the PC is loaded with the new address if and only if some condition(s) is/are true.
10. If the instruction is not a jump instruction, the address stored in the PC is incremented by one.
11. If the end of the program is not reached, the CPU goes to Step 3 and continues its fetch-decode-execute cycle.

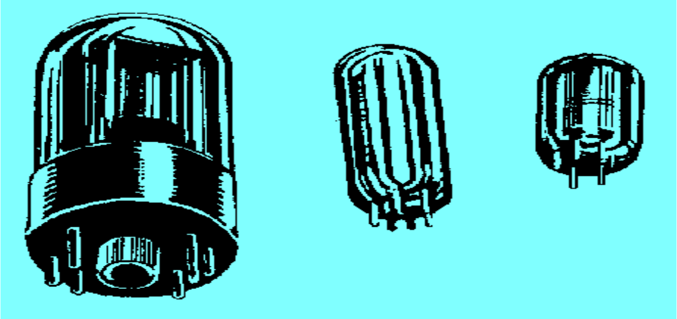
A Brief History of Computers

DIGITAL COMPUTER GENERATIONS

In the electronic computer world, we measure technological advancement by generations. A specific system is said to belong to a specific "generation." Each generation indicates a significant change in computer design. The UNIVAC I represents the first generation. Currently we are moving toward the fourth generation.

FIRST GENERATION

The computers of the first generation (1951-1958) were physically very large machines characterized by the vacuum tube. Because they used vacuum tubes, they were very unreliable, required a lot of power to run, and produced so much heat that adequate air conditioning was critical to protect the computer parts. Compared to today's computers, they had slow input and output devices, were slow in processing, and had small storage capacities. Many of the internal processing functions were measured in thousandths of a second (millisecond). The software (computer program) used on first generation computers was unsophisticated and machine oriented. This meant that the programmers had to code all computer instructions and data in actual machine language. They also had to keep track of where instructions and data were stored in memory. Using such a machine language was efficient for the computer but difficult for the programmer.

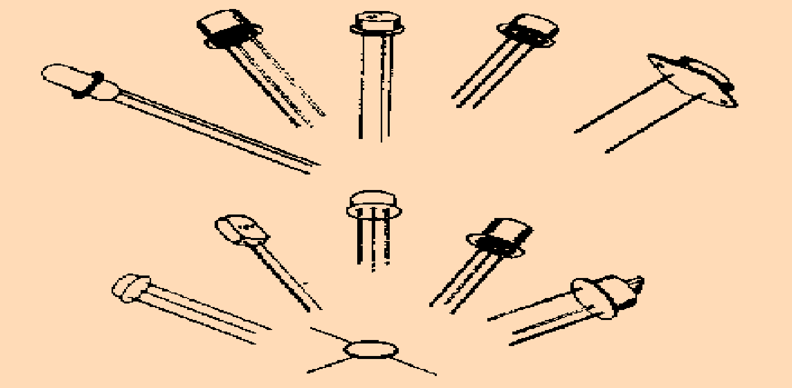


First generation computers used vacuum tubes.

SECOND GENERATION

The computers of the second generation (1959-1963), were characterized by transistors instead of vacuum tubes. Transistors were smaller, less expensive, generated almost no heat, and required very little power. Thus second generation computers were smaller, required less power, and produced a lot less heat. The use of small, long lasting transistors also increased processing speeds and reliability. Cost performance also improved. The storage capacity was greatly increased with the introduction of magnetic disk storage and the use of magnetic cores for main storage. High speed card readers, printers, and magnetic tape units were also introduced. Internal processing speeds increased. Functions were measured in millionths of a second (microseconds). Like the first generation, a particular computer of the second generation was designed to process either scientific or business oriented problems but not both. The software was also improved. Symbolic machine languages or assembly languages were used instead of actual machine languages. This allowed the programmer to use mnemonic operation codes for instruction operations and symbolic names for storage locations or stored variables.

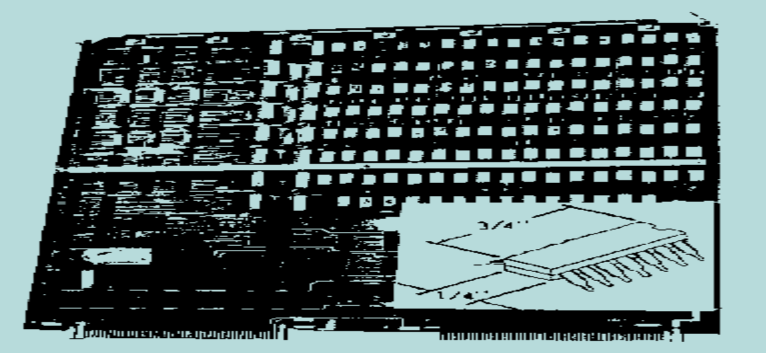
Compiler languages were also developed for the second generation computers.



Second generation computers used transistors.

THIRD GENERATION

The computers of this generation (1964-1970), many of which are still in use, are characterized by miniaturized circuits. This reduces the physical size of computers even more and increases their durability and internal processing speeds. One design employs solid-state logic microcircuits for which conductors, resistors, diodes, and transistors have been miniaturized and combined on half-inch ceramic squares. Another smaller design uses silicon wafers on which the circuit and its components are etched. The smaller circuits allow for faster internal processing speeds resulting in faster execution of instructions. Internal processing speeds are measured in billionths of a second (nanoseconds). The faster computers make it possible to run jobs that were considered impractical or impossible on first or second generation equipment. Because the miniature components are more reliable, maintenance is reduced. New mass storage, such as the data cell, was introduced during this generation, giving a storage capacity of over 100 million characters. Drum and disk capacities and speed have been increased, the portable disk pack has been developed, and faster, higher density magnetic tapes have come into use. Considerable improvements were made to card readers and printers, while the overall cost has been greatly reduced. Applications using online processing, real-time processing, time sharing, multiprogramming, multiprocessing, and teleprocessing have become widely accepted. More on this in later chapters.

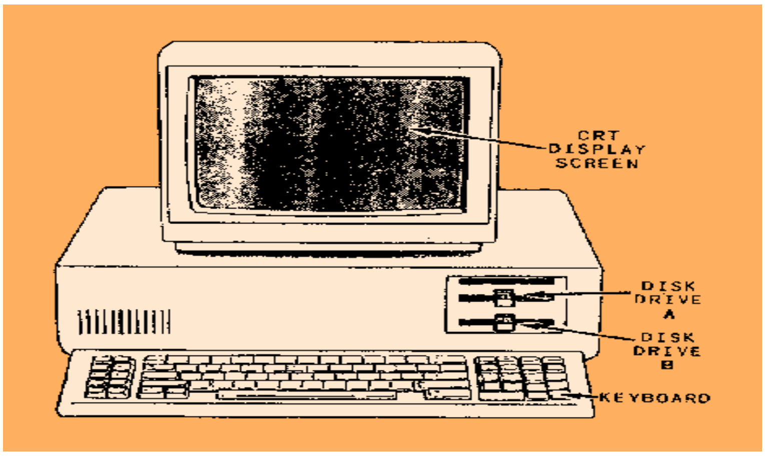


Third generation computers used microcircuits.

Manufacturers of third generation computers are producing a series of similar and compatible computers. This allows programs written for one computer model to run on most larger models of the same series. Most third generation systems are designed to handle both scientific and business data processing applications. Improved program and operating software has been designed to provide better control, resulting in faster processing. These enhancements are of significant importance to the computer operator. They simplify system initialization (booting) and minimize the need for inputs to the program from a keyboard (console intervention) by the operator.

FOURTH GENERATION AND BEYOND

The computers of the fourth generation are not easily distinguished from earlier generations, yet there are some striking and important differences. The manufacturing of integrated circuits has advanced to the point where thousands of circuits (active components) can be placed on a silicon wafer only a fraction of an inch in size (the computer on a chip). This has led to what is called large scale integration (LSI) and very large scale integration (VLSI). As a result of this technology, computers are significantly smaller in physical size and lower in cost. Yet they have retained large memory capacities and are ultra fast. Large mainframe computers are increasingly complex. Medium sized computers can perform the same tasks as large third generation computers. An entirely new breed of computers called microcomputers (fig. 1-9) and minicomputers are small and inexpensive, and yet they provide a large amount of computing power.



Fourth generation desktop (personal) computer.

What is in store for the future? The computer industry still has a long way to go in the field of miniaturization. You can expect to see the power of large mainframe computers on a single super chip. Massive data bases, such as the Navy's supply system, may be written into read-only memory (ROM) on a piece of equipment no bigger than a desktop calculator (more about ROM in chapter 2). The future challenge will not be in increasing the storage or increasing the computer's power, but rather in properly and effectively using the computing power available. This is where software (programs such as assemblers, report generators, subroutine libraries, compilers, operating systems, and applications programs) will come into play (see chapter 3). Some believe developments in software and in learning how to use these extraordinary, powerful machines we already possess will be far more important than further developments in hardware over the next 10 to 20 years. As a result, the next 20 years (during your career) may be even more interesting and surprising than the last 20 years.

The summary of Generations of Computer

* Vacuum tube - 1946-1957

* Transistor - 1958-1964

* Small scale integration: 1965

Up to 100 devices on a chip

* Medium scale integration: -1971

100-3,000 devices on a chip

* Large scale integration :1971-1977

3,000 - 100,000 devices on a chip

* Very large scale integration: 1978 -1991

100,000 - 100,000,000 devices on a chip

* Ultra large scale integration : 1991

Over 100,000,000 devices on a chip