Chapter 2

2.1 Introduction

Today's computers allow users to access the Internet, browse Web pages, display graphics and video, play music and games-and more. Personal and office computers increase productivity by managing large amounts of data. providing application- development tools and presenting an intuitive interface for authoring content. Net- works of computers coordinate to perform vast numbers of calculations and trans- actions per second. In the mobile computing market, cell phones store phone numbers, send and receive text messages and even capture photos and video. All of these computers contain various types of hardware and software, and they are all managed by operating systems.

Because the operating system is primarily a resource manager, its design must be intimately tied to the hardware and software resources that it manages. These resources include processors, memory, secondary storage (such as hard disks), other I/O devices, processes, threads, files, databases and so on. As computers evolve, operating systems must adapt to emerging hardware and software technologies and maintain compatibility with an installed base of older hardware and software. In this chapter, we introduce hardware and software concepts.

2.2 Evolution of Hardware Devices

Initially, systems programming, which entailed writing code to perform hard- ware management and provide services to programs, was relatively straightforward because the operating system managed a small number of programs and hardware resources. Operating systems facilitate applications programming, because developers can write software that requests services and resources from the operating system to perform tasks (e.g., text editing, loading Web pages or payroll processing)

2.3 Hardware Components A computer's hardware consists of its physical devices- processor(s), main mem- ory and input/output devices. The following subsections describe hardware compo- nents that an operating system manages to meet its users' computing needs.
1- CPU: Central processing Unit is the main unit of any computer system. The CPU consists of Arithmetic Logic Unit (ALU) and Control Unit (CU).the main registers included in CPU are:

•SP: Stack Pointer Register which holds an address pointing at a location in memory called

" Stack Top".

•PC : Program Counter Register which holds address of the next instruction to be executed.

•A: Accumulator Register which holds the result of any arithmetic or logic operation.

•PSW: Processor Status Word Register which holds flags showing the status of any arithmetic or logic operation (Zero, overflow, carry, etc).

•Data Registers (B, C, D,E) which act as fast storage for temporary data.

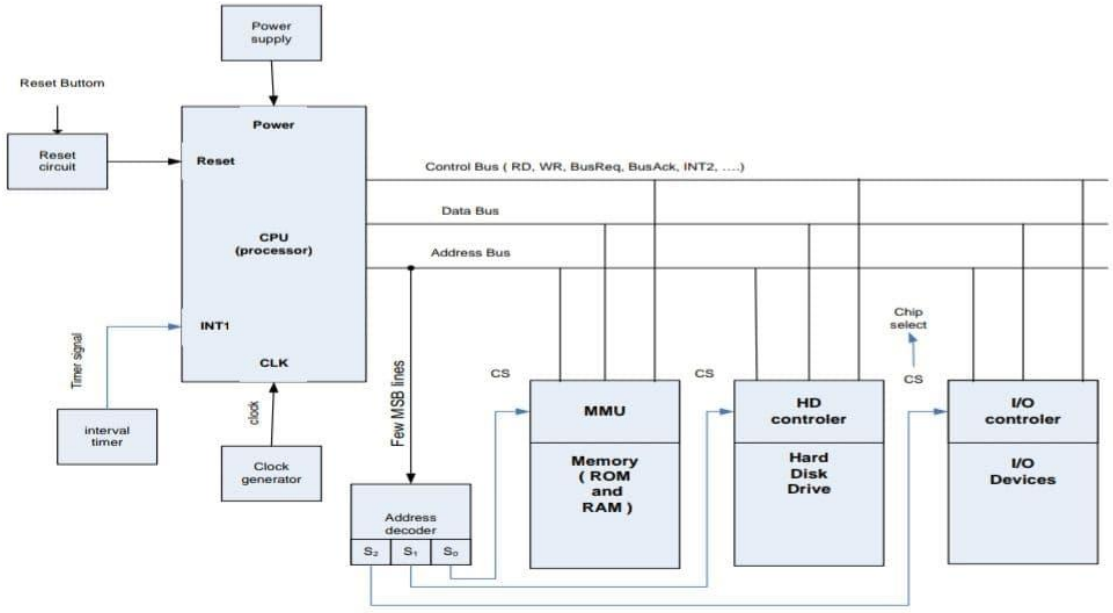

Fig 2.1 Simple Computer System.

2- Clock Generator : it is essential electronic circuit generating periodic pulse signal called "clock" as shown in fig 2.2. It is worth remembering that all actions in a computer system are timed precisely and synchronized with clock edges. The clock frequency (1/T) determines the computer speed to a large degree in addition to other factors.

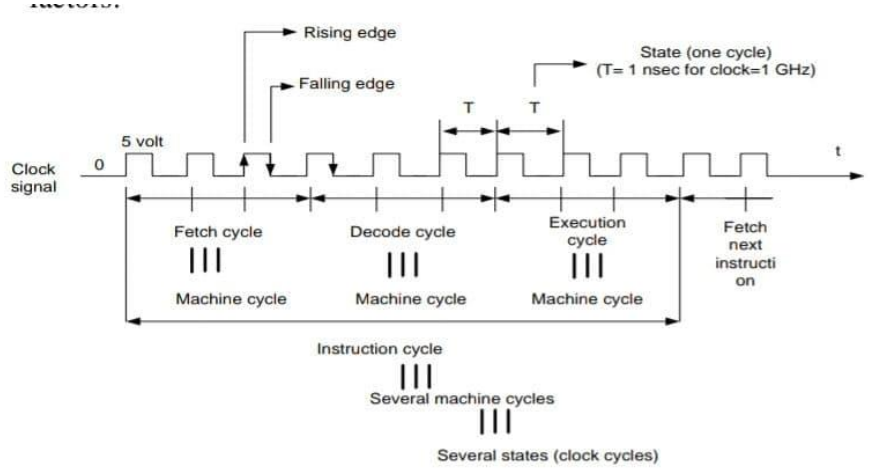

Fig 2.2 Instruction Cycle

3- Reset Circuit : when a reset button is pressed, a pulse signal is issued to CPU which forces a value of Zeros to PC and an instruction fetch cycle is started from a location in memory pointed at by pc. This means that " Reset" forces computer to start execution of a program loaded at address zero (Usually BIOS program stored in Read Only Memory ROM).

4- Interval Timer: It generates a periodic pulse but much slower than clock generator. Each Timer period is called "Time Slice" and equivalent to millions ofclock periods (states). This means that during a Time Slice, it is possible to execute part of a program. The interval timer, as will be shown later, has a big role in multiprogramming system.

5- Power Supply : It is necessary for CPU and all other components.

6- Memory : It is the second important unit in any computer system (after CPU). It is very fast addressable storage. It is used to hold programs when they are being executed in addition to other functions (e.g stack,...) there are two types of memory:

•Random Access Memory (RAM) used for storing programs and data temporarily.

•Read Only Memory (ROM) used to hold programs and data permanently (e.g BIOS).

7- Hard Disk Drive : It is a large volume of permanent storage for programs and data but it is not addressable and of slow access as well.

8- I/O Devices : the computer is not useful unless interfaced to external world such as printers, keyboards, displays, scanners, mouse, camera, etc.

9- Controllers : To connect any device to CPU, it is necessary to use a proper interface circuit (card) called " controller". The controller is usually driven by a proper program called " Driver". This means that " Controller" is a hardware (H/W) while "driver" is a software (S/W). Here, it is worth noting that a "Drive" is different from "Driver" as it represents the instrument itself e.g

disk drive which consists of disk, motor, electronic circuits, heads, etc. Also it is worth noting that a memory controller is usually called as Memory Management Unit (MMU).

10- Address Decoder: Few of most significant address signals are decoded to enable selection of one device that should respond to CPU.

11-Address Bus : Set of copper lines carrying electrical signals (voltages) which are used to select one location only in the whole computer system whether for " Read" or "Write" cycles.

12- Data Bus : several lines carrying data to or from addressed location.

13- Control Bus : Several Lines having several functions. One line, for example, is used to carry "Read" signal to declare that a cycle is a "Read" one.

14- Interrupt (INT) : It is input signal to CPU which when "active" forces certain value (address) to PC and initiates instruction cycle and hence the CPU starts executing " Interrupt Subroutine".

2-4- 1/O Data Transfer

- There are several methods for transferring data between I/O devices and memory as follows: •Programmed I/O.
- •DMA I/O.

2-4-1 Programmed I/O There are two types:

- Polling I/O
- Interrupt I/O

In polling, CPU executes a program to scan I/O device periodically to check its need for service. In interrupt I/O, there is no scan at all, however, when I/O device requires service, it activates an interrupt signal to CPU and hence interrupt subroutine will be executed which should provide the requested service.

2-4-2 Direct Memory Access (DMA) There are two useful control signals for this operation which are: BusReq and Bus Ack. BusReq is an input signal to CPU and when activated, it forces CPU to separate itself from its external "Buses" at the end of machine cycle. When separation occurs, CPU activates BusAck signal to inform I/O devices that they can use all the Buses.

Now, it is possible to explain DMA as follows:-

1- I/O device activates BusReq line and wait till BusAck is activated.
2- When Bus Ack is activated, I/O device can master the Buses and hence use them to address memory and make data transfer between Memory and I/O device.

From the above, it is clear that programmed I/O is implemented by making CPU executes a proper program while in DMA, the CPU does not interfere as it separates it self from its Buses.

2-5 CPU Modes of Operation (States) There are two main modes:

•Supervisor Mode : In this mode, the CPU can execute all instructions including user "and" privileged" instructions.

•User Mode: In this mode, the CPU can execute ”user” instruction only and can not execute privileged instructions.

The CPU modes are useful for programs protection in multi programming environment.

2-6 Bootstrapping

When a computer is powered up, or Reset activated, BIOS is started and loads " Boot Sector" from disk (Hard, CD, floppy). The Boot Sector contains a program which enables loading of main OS components into memory which then started. the above operations are called "Bootstrapping" which aims at loading OS into memory and running it.

2-7 Application Programming Interfaces (APIs)

APIS are set of subroutines that control computer resources and considered as part of OS. A user program (application) can use these APIS by calling them via special instructions called "System Calls". A system call acts as a " software interrupt" and hence changes CPU mode from " User" to " Supervisor" which in turn prevents an " address violation exception" from occurring.

Here, it should be noted that if user program tries to use normal subroutine call instruction then an address violation exception will occur as the called address is outside user program " address space" as will be explained later.

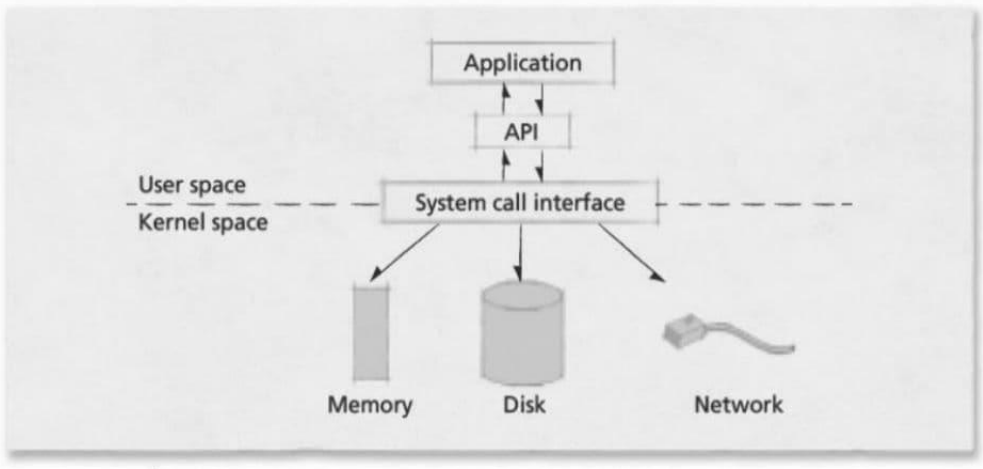

Figure 2.7 | Application programming interface (API).

2-8 Multiprocessing

It is equivalent to " multiprocessors" and not to " multi processes". Multiprocessing means that CPU consists of several processors sharing memory and controlled by single OS for the purpose of speeding up computer operation.

2-9 Buffering

A " Buffer" is an area of memory for holding data temporarily during data transfer between running program and I/O device. This technique speeds up the computer operation.

Examples of these buffers are:

a- Hard Disk Buffer :

In this case, OS writes data quickly to disk buffer (i.e to memory) and later on data will be transferred to the slow disk drive. When reading, the reverse occurs.

b- Keyboard buffer:

It is used to hold characters typed by a slow user and later the running program will process these characters.

2-10 Spooling

It is , also, a technique for speeding up computer operation. In this case, the data to be written to a very slow device (e.g printer) are written, first, to intermediate medium speed device (such as disk) and later on transferred to the slow device.

2.9 Firmware

In addition to hardware and software, most computers contain firmware, executable instructions stored in persistent, often read-only, memory attached to a device. Firmware is programmed with microprogramming, which is a layer of programming below a computer's machine language.

Microcode (i.e., microprogram instructions) typically includes simple, fundamental instructions necessary to implement all machine-language operations.62 For example, a typical machine instruction might specify that the hardware perform an addition operation. The microcode for this instruction specifies the actual primitive operations that hardware must perform, such as incrementing the pointer that references to the current machine instruction, adding each bit of the numbers, storing the result in a new register and fetching the next instruction.63,64

2. Describe the role of firmware in a computer system.

guage. 2) Firmware specifies simple, fundamental instructions necessary to implement machine-language instructions.

2.10 Middleware

Software plays an important role in distributed systems in which computers are connected across a network. Often, the computers that compose a distributed system are **heterogeneous-they use different hardware, run different operating systems and communicate across different network architectures using various network protocols. The nature of distributed stems requires middleware to enable interactions among multiple processes running on one or more computers across a net work. Middleware allows an application running on one computer to communicate with an application running communications on remote computer, enabling a between computers in distributed systems. Middleware also permits applications to run on heterogeneous computer platforms, as long as each computer has the middleware installed. Middleware simplifies application development, because developers do not need to know the details of how the middleware performs its tasks.**

Kay Terms

application programming-Software development that entails writing code that requests services and resources from the operating system to perform tasks (e.g., text editing, load- ing Web pages or payroll processing).

application programming interface (API)-Set of functions that allows an application to request services from a lower level of the system (e.g., the operating system or a library module).

basic input/output system (BIOS)-Low-level software instructions that control basic hardware initialization and management.

bootstrapping-Process of loading initial operating system components into system memory so that they can load the rest of the operating system.

buffer-Temporary storage area that holds data during I/O between devices operating at different speeds. Buffers enable a faster device to produce data at its full speed (until the buffer fills) while waiting for the slower device to consume the data.

cycle stealing-Method that gives channels priority over a processor when accessing the bus to prevent signals from channels and processors from colliding.

direct memory access (DMA)-Method of transferring data from a device to main memory via a controller that requires only interrupting the processor when the transfer completes. I/O transfer via DMA is more efficient than programmed I/O or interrupt-driven I/O because the processor does not need to supervise the transfer of each byte or word of data.

Firmware-microcode that specifies simple ,fundamental instructions necessary to implement machine-language instruction.

interval timer-Hardware that generates periodic interrupts that cause operating system code to execute, which can ensure that a processor will not be monopolized by a malicious or malfunctioning process.

mainboard-Printed circuit board that provides electrical connections between computer components such as processor, memory and peripheral devices.

main memory-Volatile memory that stores instructions and data; it is the lowest level of the memory hierarchy that can be directly referenced by a processor.

memory hierarchy-Classification of memory from fastest, lowest-capacity, most expensive memory to slowest, highest-capacity, least expensive memory.

memory protection- Mechanism that prevents processes from accessing memory used by other processes or the operating system.

privileged instruction-Instruction that can be executed only from kernel mode. Privileged instructions perform operations that access protected hardware and software resources (e.g., switching the processor between processes or issuing a command to a hard disk).

polling-Technique to discover hardware status by repeatedly testing each device. Polling can be implemented in lieu of interrupts but typically reduces performance due to increased overhead.

spool (simultaneous peripheral operations online)-Method of I/O in which processes write data to secondary storage where it is buffered before being transferred to a low- speed device.

system call-Procedure call that requests a service from an operating system. When a process issues a system call, the processor execution mode changes from user mode to kernel mode to execute operating system instructions that respond to the call. TV