OPERATING SYSTEMS
Topic-Definition & Evolution

Syllabus

- **Introduction**: Definition, Design Goals, Evolution; Batch processing, Multi-programming, Time sharing; Structure and Functions of Operating System.
- **Process Management**: Process states, State Transitions, Process Control Structure, Context Switching, Process Scheduling, Threads.
- **Memory Management**: Address Binding, Dynamic Loading and Linking Concepts, Logical and Physical Addresses, Contiguous Allocation, Fragmentation, Paging, Segmentation, Combined Systems, Virtual Memory, Demand Paging, Page fault, Page replacement algorithms, Global Vs Local Allocation, Thrashing, Working Set Model.
- **Concurrent Processes**: Process Interaction, Shared Data and Critical Section, Mutual Exclusion, Busy form of waiting, Lock and unlock primitives, Synchronization, Classical Problems of Synchronization, Semaphores, Monitors, Conditional Critical Regions, System Deadlock, Wait for Graph, Deadlock Handling Techniques: Prevention, Avoidance, Detection and Recovery.
- **File and Secondary Storage Management**: File Attributes, File Types, File Access Methods, Directory Structure, Allocation Methods, Free Space management; Disk Structure, Logical and Physical View, Disk Head Scheduling.

Definitions

- The 1960’s definition of an operating system is

“the software that controls the hardware”.

- Today’s definition of an operating system is

“the programs that make the hardware usable”

In brief, an operating system is the set of programs that controls a computer.

Definitions (cont.)

- An Operating System(OS) may be viewed as an organized collection of software extensions of hardware, consisting of control routines for operating the computer and for providing an environment for execution of programs
- Other programs rely on facilities provided by the operating system to gain access to computer-system resources, such as files and input/output devices.
- Programs usually invoke services of operating system by means of operating-**system calls**
- User interact with the operating system directly by means of operating-s**ystem commands**

Major Roles of OS

- Hardware provides " raw computing power". The operating systems make this computing power conveniently available to users and manages the hardware carefully to achieve good performance.
- The operating system acts as an interface between the user and the hardware of the computer system.
- The OS takes care of all inputs and outputs in a computer system and manages users, processes, memory, printing, networking etc.

Internal & External Views

- **Internally**, an operating system acts as an a manager of resources of the computer such as processor, memory, files, I/O devices, etc.

- **Externally**, an operating system acts as an user interface.

OS Design Goals

Operating system design is a complex task

User Interface	should the interface be easy to learn by a novice user, or should it be designed for the convenience of an experienced user? (multiple user interfaces?)
Efficient System Resource Management	Unfortunately, the more complete the resource management involves the more overhead.
Security	Once again, the more secure a system is will be less efficient
Portability	Will the operating system be portable to widely varying types of hardware, or just different models of a particular class of hardware?
Backwards Compatibility and Emulation	Is it important that software that ran under previous operating system versions or under different operating systems be supported?

OS Design Goals (Cont.)

Flexibility: Most operating systems come preconfigured for many different devices. Part of the process of setting up a particular machine is to construct a version of the operating system that is tuned for the local installation. This tuning often involves setting certain limits, such as the maximum number of processes. It also involves specifying the attached hardware so that only the necessary drivers will be loaded. Some operating systems can load and unload drivers automatically at run-time.

Evolution of OS

- An OS may process its workload serially or concurrently i.e. the resources of the computer system may be dedicated to single program or dynamically reassigned among a collection of active programs in different stages of execution.
- The OS evolved through the path of serial processing, batch processing, multiprogramming etc.

Serial Processing

- Every computer was programmed in its machine language with no system software.
- Programs for bare machine(without operating syatem) can be developed by manually translating sequences of instructions into binary code.
- Instructions and data are entered into computer by means of control switches.
- Programs are started executing by loading the program counter with the address of the first instruction.
- The result of the execution are obtained by examining the relevant registers and the memory locations.
- Programming with the bare machine result in low productivity of both user and hardware.

Serial Processing (cont.)

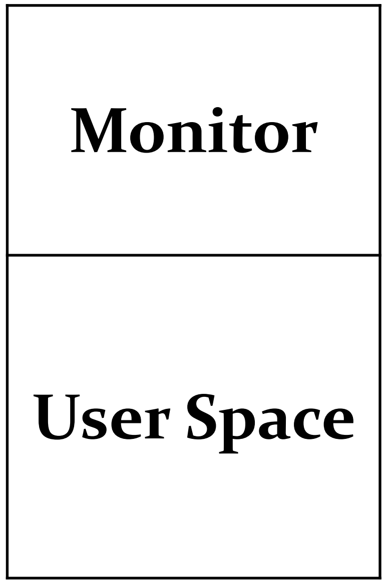
- The next stage was with the advent of input/output devices and of language translators.
- Program can be coded in a high level programming languages and translated into executable form by a compiler or an interpreter. The loader loads the executable program into memory and the program execution commences
- The executable program reads its input from the designated input devices and produce output on an output device.
- The whole process is cumbersome due serial execution and involves manual operations.

Batch Processing

- In Batch Processing
  - Firstly, user prepares his job using punch cards.
  - The user submits the job to the computer operator.
  - Operator collects the jobs from different users and sort the jobs into batches with similar needs.
  - The operator submits the batches to the processor one by one.
  - All the jobs of one particular batch are executed together.

Batch Processing (cont.)

- A batch system is one in which jobs are bundled together with the instructions necessary to allow them to be processed without intervention
- The basic physical layout of the memory of a batch job computer is shown below:
    - Monitor (permanently resident)
    - User Space (compilers, programs, data, etc.)



Advantages and Disadvantages of Batch Systems

1. Advantages of batch systems
   - ✓ Much of the manual work in serial processing is shifted from the operator to the computer
   - ✓ Increased performance as it will be possible to start a job after the previous job is finished.
2. Disadvantage
   - ✖ Turn-around time can be larger from user point of view
   - ✖ More difficult to debug program
   - ✖ One job can affect the pending jobs
     - ➢ Reading too many cards
     - ➢ A job could corrupt the monitor
     - ➢ A job could enter an infinite loop

Job Control Language for Batch System

$JOB user_spec	identify the user for the account purpose
$FORTRON	Load the FORTRON Compiler
$LOAD;	Load the compiled program
$RUN;	Run the program
Data cards	
$EOJ;	End of Job

SPOOLING

- It is a sophisticated form of I/O buffering
- It uses the disk to temporarily store input and output of jobs
- Card to disk for the subsequent program and disk to printer for the previous program are performed by the SPOOLing monitor concurrently with the execution of the current program.

Multiprogramming

Most of the programs during their execution oscillate between computational-intensive and I/O-intensive phases.

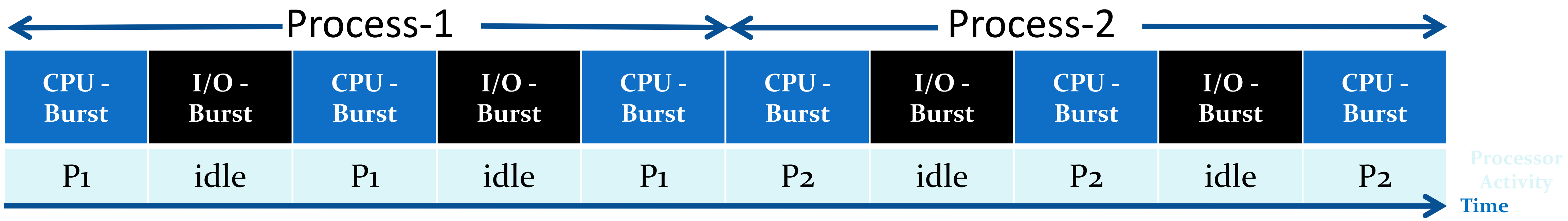


Fig. 1: Serial Processing

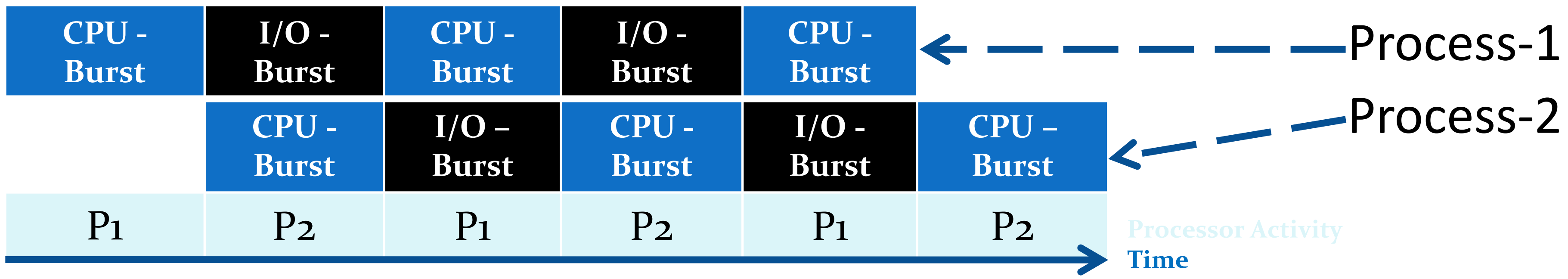


Fig.2: Multiprogramming (Interleaved Execution)

Multiprogramming (cont.)

For comparison purpose both processes P1 and P2 are assumed to have identical behaviour with regard to CPU & I/O burst times and their relative distribution

In fig.1 serial execution causes either the CPU or I/O devices to be idle at sometime even if job queue is never empty

In fig.2 whenever P1 goes for I/O completion, the CPU will be assigned to P2 and P2 goes for I/O, the CPU is assigned to P1

- As a result, significant performance gain is achieved by interleaved execution of programs- known as multiprogramming
- With single processor, parallel execution of programs is not possible and at most one program can be in control of the CPU at any time and the objective is to have some process running at all times in order to maximize CPU utilization.

Multiprogramming (cont.)

- To increase th resource utilization, actual multiprogramming system usually allow more than two programs to compete for system resources at any time
- The number of programs actively competing for resources of multiprogrammed system is called the degree of multiprogramming.
- In principle, higher degrees of multiprogramming should result in higher resource utilization.