

CS 431: Introduction to Operating System

Introduction

Chapters: 1, 2 and 3

V Kumar

Department of Computer Networking
University of Missouri-Kansas City

Q. What is an Operating System?

A. A set of specialized software modules that makes computing resources (hardware and software) available to users.

During execution a program may need a number of resources (software and hardware).

Example

```
Program example (input, output);
  var a, b, c : integer; f1, f2 : text;
  Begin
    reset (f1); rewrite (f2);
    read (f1, b); read (c);
    a := b * c;
    write (f2, a); writeln (a:5)
  end.
```

Program "example" requires: pascal compiler, files f1 and f2, a line printer, memory, etc., for execution.

It is the responsibility of the underlying operating system to find and allocate these resources to program "execution" in an appropriate time. We will discuss in detail how the O/S handle this and related problems.

We identify two types of operating systems:

- a.** No stored program systems. **b.** Stored program systems.

No stored program systems

Computer system: only hardware.

System organization: User + Operator (human) + Hardware.

User: tells the operator what to do.

Operator: Keys in the inputs using required switches on the hardware. Gets the result using different keys. Passes the results to the user. Result correct? If YES then user goes home else he/she has nightmare.

Jobs were submitted one at a time and during execution no interference was possible. So the execution steps were:

- a. Collect all the jobs to be run (batch of jobs)
- b. Give the batch to the operator
- c. The operator inputs these jobs in the system.
- d. End of execution of batch of jobs.

Stored program systems

Computer system: Hardware + Operator (non-human = a set of programs)

In this system the job of the operator is replaced by set of programs which is called Operating System. The responsibilities of the human operator is only to provide external resources (loading disk, tape etc.).

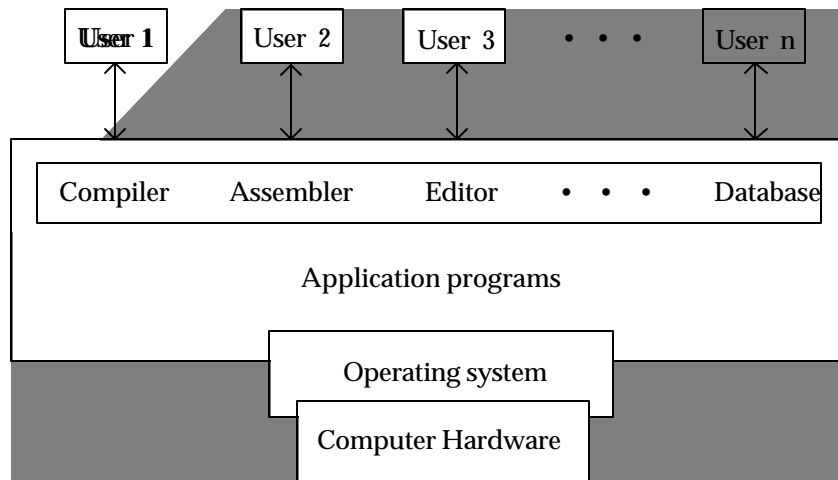
System organization: User (human) + Computer system

Execution steps:

- a. Prepare jobs on a storage medium (tape, disk etc.)
- b. Submit jobs one at a time to O/S.
- c. O/S provides necessary resources, initiates the execution and at the end of execution stores the results (if any) at a designated place.
- d. User collects results.

Systems Structure

The following diagrams gives a structure of a modern computer system:



Application programs are resources requested by users and provided by the operating systems. The operating system uses hardware to create an execution environment.

Hardware components

CPU (Central Processing Unit): Set of very fast registers (accumulators). All processing (arithmetic and boolean) are done here. CPU accesses physical memory where data and instructions are stored for processing.

Relocation register: A program can be loaded anywhere in main memory. The system must keep track of the starting address of every program. The relocation register is responsible for relocating a program anywhere in the main memory.

Storage interleaving: The entire main memory can be divided into separate blocks which are connected in such a way that data can be accesses in parallel. This reduces storage access time.

Buffer: A reserved area in the main memory. It is used to store data temporarily during program execution. A system usually provide double buffering (two buffers) to reduce waiting time in accessing data.

Peripheral devices: Line printers, disk, magnetic tape deck, card reader, card punch etc.

Timers and Clocks: The timer helps preventing a program to monopolize the entire system. It is used in implementing multiprogramming system. The clock provides real time.

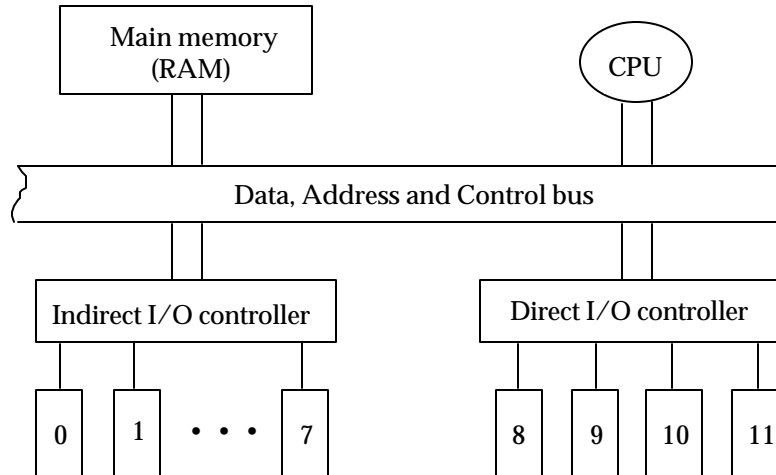
Input/Output Channels: I/O processors. Like the CPU they are specifically designed for input/output processing. Like the CPU they retrieve both their commands and data from the main memory and have their own internal registers and program counter. For example, a channel may take care of printing process relieving the CPU to do other useful things.

Types of services

Cycle stealing: There are two ways of performing input and output:

Direct I/O: Requires intensive service by the CPU in that each unit of data, usually a byte or a word, must be transferred to or from main memory by explicit actions of the CPU. This mode of data transfer is sometimes referred to as Programmed input/output.

Indirect I/O: Accomplished independently of the CPU, once initiated. In this environment the controller is sufficiently powerful to move all data between memory and the device without the aid of the CPU. Indirect input/output requires very little CPU assistance and is more efficient. The following diagram illustrates a typical single bus I/O configuration.



When read and write requests are placed on the bus for data transfer to or from memory, only one such request can be honored per memory cycle. Since high speed devices are pouring data into the controller and for efficiency these data must be accepted by the controller when available (the alternative being to stop the tape or wait an entire revolution of the disk). In other words, the controller must have access to the bus on demand.

It is possible that the CPU and the controller requests for the bus may come at the same time, the one with higher priority will be granted and the other will wait until the bus has been released. In most system, therefore, indirect I/O controller, which services high-speed devices, are located closer to memory than is the CPU (see above diagram). They thus have the higher priority and will be granted control of the bus whenever their requests are made during the same cycle as a request by the CPU. BY precedence, the

controller has stolen a cycle from the CPU; thus the term **CYCLE STEALING** is applied to high-priority controllers.

Virtual storage: It gives a programmer an illusion that he/she has memory available larger than the program size, even though the size of the real main memory may not be larger than the program size. This concept is implemented, for example, using paging, segmentation, paging and segmentation etc.

Spooling: A method to create infinite number of slow peripheral devices (line printer, magnetic tape).

Buffering: A special pre-reserved area in the main memory. The O/S stores temporary information or most frequently accessed file in the buffer. Buffers are also used during reading from a file (card reader, magnetic tape, disk, etc.) and writing to a file. To improve system performance double buffering is employed. In this arrangement two buffers are created. One buffer may be used to move data from the disk while the information stored in the other buffer could be emptied to the line printer or to the tape.

Types of Execution Environments and Operating Systems

An operating system can handle the execution of a set of jobs in a number of different ways. Each way defines an execution environment. An O/S can be classified as a general O/S or special purpose depending upon what types of environment it handle most efficiently.

Multiprogramming: It is an execution environment. The execution of a number of jobs are interleaved, i.e., a number of jobs are executed simultaneously where they share systems resources in a synchronized way. Each job gets a fair share of CPU resources. The O/S which provides multiprogramming execution environment is called a Time-Sharing O/S. A time-sharing O/S implements multiprogramming environment through time-slicing where each job is given a fixed amount of CPU resource. If a job does not finish in one time-slice then it gets the next time-slice under some scheduling policy.

Interactive: An execution environment. A user has direct communication with the operating system. The user can give commands to the O/S directly on the terminal and can expect an immediate response from the O/S. An interactive facility is only possible on an On-Line operating system. An on-line O/S provides facilities to interact with the entire system on one-to-one basis. Interactive jobs are usually multiprogrammed.

Batch: An execution environment where a number of jobs are collected together and submitted to the O/S for execution. These jobs are executed by the O/S at some predefined time. During the execution of batch jobs system does not allow a user to interact with his/her job. An O/S that can handle only batch jobs is called a Batch O/S. Batch jobs can be multiprogrammed if the batch O/S has this capability.

Real-Time: This is a type of O/S. It guarantees that a job will finished in a predefined time. Real-time jobs are highly time-dependent and they must finish within the defined time-period.

Computer System Structure

We look at inside of an O/S briefly to understand its Mode of operation.

Interrupt-Based Systems: Interrupt is a mechanism to draw system's attention to perform some specific task. Sometime it becomes necessary to divert the CPU from what it is doing to some other task. For example, if a batch job is utilizing the CPU and an important interactive job completes the data transfer from disk then the CPU must resume its execution. This type of CPU migration which is forced upon the CPU is achieved through interrupt. An interrupt mechanism gives the CPU a complete freedom to continue whatever it initiates. This improves resources utilization by decreasing CPU idle time. If there is

no interrupt then the CPU must stick to one job only and switch to other important activity after it has completed the current activity. The following steps describes the processing of an interrupt:

- a. CPU is being used by Job 1 and it generates an interrupt.
- b. The CPU leaves Job 1 to service the interrupt. But before the interrupt is serviced the address of the next instruction (link address) of Job 1 must be saved on the stack.
- c. The O/S finds the type of interrupt and loads the interrupt program from the library.
- d. The CPU begins processing the interrupt program and discovers that a data transfer has been requested. It prepares and initiates the data transfer which is done by I/O processor. The CPU is now free to do the next task decided by the scheduler.
- e. The CPU is assigned to the new task. During execution the I/O of Job 1 completes so another interrupt is generated by the I/O processor. This interrupt is processed in a similar way.
- f. Data for Job 1 is now available and its link address is loaded in the program counter from the stack and Job 1 resumes execution.

This kind of O/S is called interrupt-driven and almost all modern computers are based on this scheme. Sometime the CPU has to wait for the completion of a task. There are basically two ways to force the CPU to wait: **Busy wait** and **Special wait**. Busy wait is enforced using a simple one instruction loop which is continuously executed by the CPU. It is called busy wait since no progress (as far as a user is concerned) is made even though the CPU resource is utilized. A busy wait loop is discontinued through an interrupt. The special wait is built into hardware and keeps the CPU idle but not busy. If the wait is very small then usually busy wait is utilized.

Dual-Mode operation

During the execution of a job the system performs a number of activities. A subset of these activities are only to update systems status related to the job and the rest are exclusively for the job. This means the O/S oscillates into system mode and user mode. In system mode the system works on system data, i.e., memory management, stack management, interrupt processing, etc. These must be handled in a system mode since these are sensitive data and must be inaccessible to the user.

Operating System Services

- a. Memory management.
- b. Secondary storage management.
- c. File management
- d. Job management
- e. Process management
- f. Interrupt management
- g. Security management.