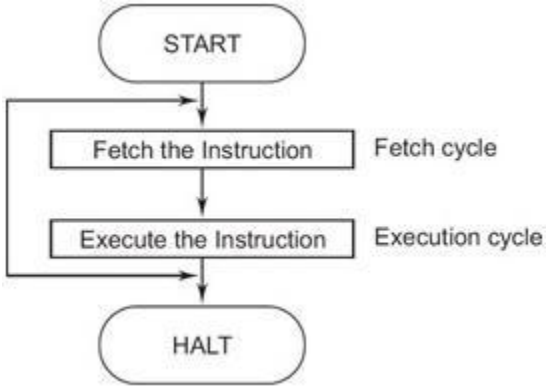
Basics of How Operating System Works

This is an article on *Basics of How Operating System Works* in *Operating System*.

In last article we went through History and Evaluation of Operating System where I gave an overview of operating System ad its evolution. Here I will first discuss about instruction and how it is executed.

The basic function performed by a computer is program execution. The program (to be executed) consists of a set of instructions stored in memory. Instruction processing consists of two steps.

- **Fetch Cycle:**
  Processor reads (fetches) instruction from memory one at a time is called "fetch cycle".



- **Execution Cycle:**
  Execution of a fetched instruction is called "execution cycle".

- **Instruction Cycle:**
  The processing time required for the execution of a single instruction is called an "instruction cycle".

  That is Instruction cycle = Fetch cycle + Execution cycle.

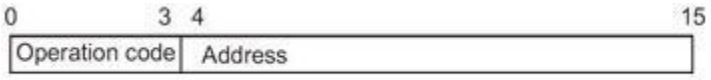
  Program execution consists of repeating the process of instruction fetch and execution. Program execution halts only if the machine is turned off. At the beginning of each instruction cycle, the processor fetches an instruction from memory.

  Program counter (PC) is used to keep track of which instruction is to be fetched next. Unless told otherwise, the processor always increments the program counter after fetching each instruction.

  The fetched instruction is loaded into a register in the processor known as the "instruction

register(IR)".

The following figure is showing the typical instruction format of 16 bits long, four bits are ment for representing 16 different operation codes (OP codes) (24 = 16) and 12 bits are meant for representing Address (i.e., 212 = 4096 (4k) words).



INTERRUPTS

Interrupts interrupt the normal processing of the processor. Interrupts are provided for improving the processing efficiency. Most external devices are much slower than the processor.

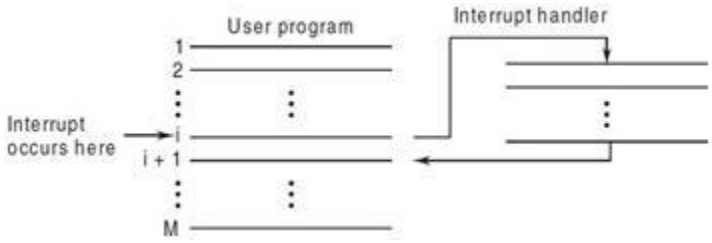
For example, if the processor is transferring data to a printer by using the instruction cycle scheme. After each WRITE operation, the processor is to pause and remain idle until the printer catches up. Here the processor time is wasting.

Classes of Interrupts

- **Program interrupts**
  Interrupt generated by some condition that occurs as a result of an instruction execution. E.g., Arithmetic overflow, division by zero, etc..
- **Timer interrupts**
  This allows the operating system to perform certain functions on a regular basis. E.g., Automatic saving of a document after certain time period.
- **I/O interrupts**
  Generated by an I/O controller to signal normal completion of an operation or to signal a variety of error conditions.
- **Hardware failure interrupts**
  Hardware failure interrupts will occur due to power failure, memory parity-error, etc.

With interrupts, the processor can be engaged in executing other instructions while an I/O operation is in progress. When an external device is ready to accept more data from the processor, the I/O module for that external device sends an interrupt request signal to the processor. It responds by suspending operation of the current program and serve the I/O device is known as an "interrupt handler", and carry out the original execution after the device is serviced. Therefore the user program does not have to contain any special code to accommodate interrupts.

This is depicted in the figure below:

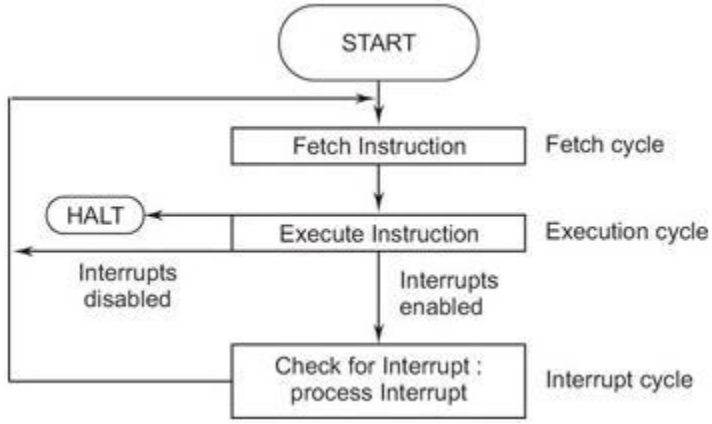


Interrupt Processing

The occurrence of an interrupt triggers a number of events, both in the processor hardware and in software.

Two approaches for dealing with multiple interrupts are:

- Disabled interrupts while an interrupt is being processed.This is shown in the figure below:



- Define priorities for interrupts: Higher priority interrupt cause a lower-priority interrupt handler to be itself interrupted.

MEMORY HIERARCHY

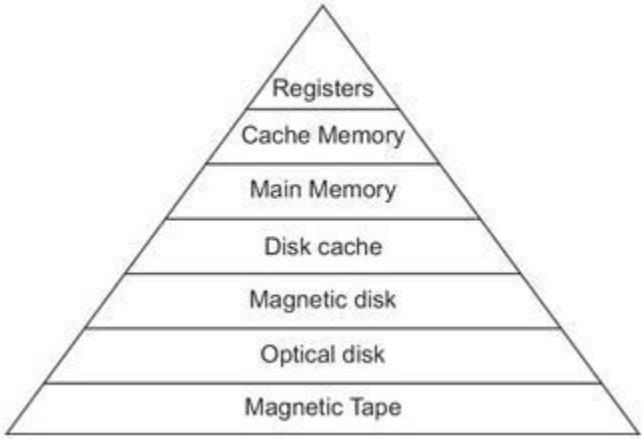
Design constraints on a computer's memory

- How much capacity?
- How fast in case of accessing time?
- How expensive?

At the time of designing we should achieve the trade-off among these three constraints.

Relationships among these constraints

- Smaller access time leads to greater cost per bit.
- Greater capacity leads to smaller cost per bit.
- Greater capacity leads to greater access time.
  The designers prefer to use expensive, smaller capacity memories with fast access times. As we go down the hierarchy in the figure below the following conditions (i.e., situations) will occur:



- Decrease cost per bit.
- Increase capacity.
- Increase (i.e., greater) access time.
- Decrease frequency of access of the memory by the Processor, is known as "locality of reference".

Disk Cache

A portion of main memory can be used as a buffer to temporarily hold data. It improves disk performance and minimizes processor involvement.

SYSTEM COMPONENTS

Create a system as large and complex as an operating system only by partitioning it into smaller pieces.

Process Management

A program does nothing unless its instructions are executed by a CPU. A process is defined as a program in execution. A process needs certain resources, including CPU time, memory, files and I/O devices to accomplish its task.

Program by itself is not a process. A program is a passive entity, such as the contents of a

file stored on disk. Process is an active entity, with a program counter specifying the next instruction to execute. The execution of a process progress in a sequential fashion. A process is the unit of work in a system. Some of which are operating system processes and remaining are user processes.

Operating system activities in connection with Process Management

- Creation and deletion of both user and system processes.
- Suspension and resumption of processes.
- Provision of mechanisms for process synchronization.
- Provision of mechanisms for process communication.
- Provision of mechanisms for deadlock handling.

Main-memory Management

Main-memory is a large array of words or bytes, ranging in size from hundreds of thousands to hundreds of millions. The central processor reads instructions from main-memory during the instruction-fetch cycle, and both reads and writes data from main memory during the data-fetch cycle. The I/O operations implemented via DMA also read and write data through main memory.

To process data from disk, first transferred to main memory. To improve both the utilization of CPU and the speed of the computer's response, keep several programs in memory. Selection of a memory-management scheme depends on many factors.

Operating system activities in memory-management

- Keep track of which parts of memory are using by whom.
- Which processes are loaded into memory and when memory space becomes available.
- Allocate and deallocate memory space as needed.

File Management

Computers can store information on several different types of physical media, such as magnetic tape, magnetic disk, optical disk, etc. These media properties include speed, capacity, data transfer rate and access method (sequential or random).

A file is a collection of related information defined by its creator. Commonly files represent programs (source and object) and data. Data files may be numeric, alphabetic or alphanumeric. Files are organized into directories to ease their use.

Operating System activities in File Management

- Creation and deletion of files.
- Creation and deletion of directories.
- Manipulating files and directories.
- Mapping of files onto secondary storage.
- Backup of files on stable (non-volatile) storage media.

Input/Output System Management

Operating system is to hide the peculiarities of specific hardware devices from the user, only the device driver knows the peculiarities of the specific device to which it is assigned.

Operating system activities with respect to I/O sub-system

- Memory management component including buffering, caching and spooling.
- A general device-driver interface.
- Drivers for specific hardware devices.

Secondary-Storage Management

The main purpose of a computer system is to execute programs. These programs, with the data they access must be in main memory during execution. Most programs including compilers, assemblers, sort routines, editors and formatters are stored on a disk until loaded into main memory.

Operating System activities in disk management

- Free-space management.
- Storage allocation.
- Disk scheduling.

Networking

A distributed system is a collection of processors that do not share memory, peripheral devices or clock. Processors communicate with one another through various communication lines.

Operating system activities in networking

- Routing and connection strategies.
- Control the problems of connection and security.

- Access to a shared resources allows computation speedup, increased data availability and enhanced reliability.

Protection System

If a computer system has multiple users and allows the concurrent execution of multiple processes, then the various processes must be protected from one another's activities.

Operating system activities in protecting systems

- Device control registers are not accessible to users.
- Controlling the access of programs, processes or users.
- To distinguish between authorized and unauthorized usage.

OPERATING SYSTEM SERVICES

Operating system services will differ from one operating system to another.

Some common classes are as follows:

- **Program creation and execution**
  The system should be able to create and load a program into memory and to run it.

- **Access to Input/Output devices**
  For Input/output operations, a running program may require Input/Output devices or a file.

- **File-system manipulation**
  As the part of the file-system manipulation, operating system can control the access to files.

- **Communications**
  There are many circumstances in which one process needs to exchange information with another process.

- **Error detection and response**
  Operating system constantly needs to be aware of possible errors. Errors may occur in the central processing unit (CPU) and in memory hardware.
  Another set of operating system functions are also there. These are not for helping the user, but for ensuring the efficient operation of the system.

These are as follows:

- **Resource allocation**
  When there are multiple users or multiple jobs running at the same time, resources must be allocated to each of them.

- **Accounting**
  A good operating system collects usage statistics for various resources and monitors performance parameters such as response time and keeps track of which users use how much memory and what kind of computer resources.

- **Protection**
  The owners of information stored in a multiuser computer system is to control its use. When several disjoint processes execute concurrently, it should not be possible for one process to interfere with the others.

SYSTEM CALLS

Provide the interface between a process and the operating system. These calls are generally available as assembly-language instructions and are listed in the manuals. Some systems may allow system calls to be made directly from a higher-level language (in-line).

E.g., C, Bliss, BCPL, PL/360, PERL and FORTRAN

To read data from one file and copy to another file. For this first input to the program is two file names (i.e., input and output files). These file names can be specified in many ways.

- Ask the user for the names of the two files.
- Specify the file names with control statements.
- Use the mouse to select the source name from a list of files and fill the destination.
  If there is already an output file with the same name as input file then the program may abort using one "system call" or delete the existing file by using another "system call" and create a new file using one more "system call". Finally after the entire file is copied, the program may close both files using another "system call". Write a message to the console (more "system calls") and terminate normally by a last "system call".

System calls occur in different ways, depending on the computer in use. System calls are roughly grouped into five major categories.

List of system calls under each category are as follows:

- **Process and Job Control**
  End (halt its execution normally)
  Abort (halt its execution abnormally)

load, execute
Create process, terminate process
Get process attributes, Set process attributes
Wait event, signal event
Allocate and free memory

- **File Manipulation**
  Create file, Delete file
  Open file, Close file
  Read file, Write file
  Get file attributes and Set file attributes

- **Device Manipulation**
  Request device, Release device
  Read device, Write device,
  Get device attributes, Set device attributes
  Logically attach or detach devices

- **Information Maintenance**
  Get time or date, Set time or date
  Get system data, Set system data
  Get process, file or device attributes
  Set process, file or device attributes.

- **Communications**
  Create communication connection, Delete communication connection
  Send messages, Receive messages
  Transfer status information
  Attach or detach remote devices.

Source: http://www.go4expert.com/articles/basics-operating-t22273/