

St. Mary's University College

Faculty of Informatics

Study Guide

For

Operating System Principles and Design

Department of Computer Science

Faculty of Informatics



Prepared by

Taye Girma (Asst.Prof)

January 2012

Summary of the Module

This study guide will direct the readers to address two parts: Operating system and computer organization and Assembly programming language. The operating system will enable to understand an overview of basic operating system, process management, memory management, I/O management and file management file system. The computer organization will guide the readers to understand about basic computer design, parallel processing, pipelining and computer set instructions. The assembly programming language part will enable the readers to address some of the basic directives, control statements, arithmetic instruction, addressing modes and string manipulation instruction.

St. Mary's University College

Faculty of Informatics

Part I

Operating System Principles and Design

Unit One

Introduction

Summary

Operating system is the system software which is used as the resource manager, where the resources are CPU, memory and I/O devices. There is no clear definition of operating system; however, the basic definition of operating system is based on the function the operating system. The common evolutions of the operating system are serial processing, batch processing, multi-programming, parallel processing and real time operating. Operating has four major functions: process management, memory management, I/O management and file management. The basic concepts of operating system include process, memory management, files, directory, system call, interrupts and deadlock.

General Objective

To understand the basic concepts; the design and working principles of operating system

Specific Objective

At the end of this unit the readers are expected to:

- Understand the software.
 - Explain what application software is.
 - Explain what system software is.
 - Identify the types of system software.
- Understand the goals and functions of operating system.
 - Explain the goals of operating system.
 - Explain the functions of operating system.
- Understand the evolution of operating system.

- Define the evolution of operating system.
 - Distinguish between serial processing and batch processing.
 - Distinguish between mono-programming and multi-programming.
 - Explain parallel processing and comparing it with multi-programming.
 - Explain the real time operating system.
- Understand the basic concepts of operating system.
- Explain the difference between process and program.
 - Explain Memory and Memory management.
 - Explain the system calls
 - Explain I/O devices and Input – output management.
 - Explain Files and File management.
 - Explain the deadlock.

Self-test question list without answer key

1. Define operating system based on the function it provides.
2. Write the difference between program and process.
3. Explain the system call.
4. List some differences between personal computer operating systems and mainframe operating systems.
5. List the goals of operating system.
6. Explain the difference between serial processing and batch processing.
7. What is the difference between parallel processing and multiprogramming?

Self-test question list with answer key

1. List the function of operating system.
2. Define the essential properties of the following types of operating systems:

a) Batch b) Interactive c) Time sharing d) Real time

e) Network f) Parallel g) Distributed h) Clustered i) Handheld

3. In a multiprogramming and time-sharing environment, several users share the system simultaneously. This situation can result in various security problems.

- a) What are two such problems?
- b) Can we ensure the same degree of security in a time-shared machine as in a dedicated machine? Explain your answer

Answer Key

1. The functions that the operating system provides are:

- Process management
- I/O management
- Memory management
- File management

2.

- a) **Batch.** Jobs with similar needs are batched together and run through the computer as a group by an operator or automatic job sequencer. Performance is increased by attempting to keep CPU and I/O devices busy at all times through buffering, off-line operation, spooling, and multiprogramming. Batch is good for executing large jobs that need little interaction; it can be submitted and picked up later.
- b) **Interactive.** This system is composed of many short transactions where the results of the next transaction may be unpredictable. Response time needs to be short (seconds) since the user submits and waits for the result.
- c) **Time sharing.** This systems uses CPU scheduling and multiprogramming to provide economical interactive use of a system. The CPU switches rapidly from one user to another. Instead of having a job defined by spooled card images, each program reads its next control card from the terminal, and output is normally printed immediately to the screen.
- d) **Real time.** Often used in a dedicated application, this system reads information from sensors and must respond within a fixed amount of time to ensure correct performance.
- e) **Network.** Provides operating system features across a network such as file sharing.

- f) **SMP.** Used in systems where there is multiple CPU's each running the same copy of the operating system. Communication takes place across the system bus.
 - g) **Distributed.** This system distributes computation among several physical processors. The processors do not share memory or a clock. Instead, each processor has its own local memory. They communicate with each other through various communication lines, such as a high-speed bus or local area network.
 - h) **Clustered.** A clustered system combines multiple computers into a single system to perform computational task distributed across the cluster.
 - i) **Handheld.** A small computer system that performs simple tasks such as calendars, email, and web browsing. Handheld systems differ from traditional desktop systems with smaller memory and display screens and slower processors.
- 3.
- a. Stealing or copying one's programs or data; using system resources (CPU, memory, disk space, peripherals) without proper accounting.
 - b. Probably not, since any protection scheme devised by humans can inevitably be broken by a human, and the more complex the scheme, the more difficult it is to feel confident of its correct implementation.

Unit Two

Process Management

Summary

Operating system executes a program in terms of process and thread. A process is the part of the program which is currently under execution. But, the thread is the light weight of the process. There are a number of operation performed on the process and during all the operation the data structure called process control block or process table will be prepared by operating system for all processes created during program execution. The content of process control block are pointer to parent process, pointer to child process, program counter register, process id, process states, memory limit and files opened related to that process. During program execution the operating system will perform the operation in process level and thread level and there are a number of algorithms that handles the inter-process communication to maintain the mutual exclusion and process scheduling.

General Objective

To understand program execution in computer system in terms of process and thread

Specific Objectives

At the end of this unit the readers are expected to:

- Understand the concept behind program and process.
 - Distinguish between program and process.
- Understand the uses of process control block or Process Table.
 - Identifying the content of process control block.
 - Identifying the reasons why PCB is used during program execution.

- Understand the process states and state diagram
 - Identify all the possible states of the process.
 - Identify the cause for the process states transition.
 - Identify the role of the scheduler during process state transition.
 - Explain process state transition using diagram.
- Understand process switching.
 - Explain how process switching is handled.
 - Identify the reasons why process switching is performed.
- Understand the concept behind threading.
 - Distinguish thread from process.
 - Identify the thread states.
 - Identify the thread functionalities.
 - Identify the reasons for thread states transitions in thread diagram.
 - Explain the multithreading and its use.
- Understand how the inter- process communication is made during program execution
 - Why inter-process communication is required?
 - How inter-process communication can be maintained?
 - Identify the ways in which mutual exclusion will be maintained.
 - Distinguish between semaphore and monitors.
- Understanding the Process scheduling?
 - Explain process scheduling.
 - Identify the CPU scheduling goals.
 - Distinguish between CPU bound and I/O bound.
 - Identify all the schedulers used during program execution

- Identify all the process scheduling criteria.
- Distinguish between the preemptive and non-preemptive scheduling.
- Identify all the scheduling algorithms.

Self-test question list without answer key

1. What are the different types of schedulers?
2. What is the difference between the role of dispatcher and short term scheduler?
3. Explain why the size of the process is greater than the size of the program from which the process is originated.
4. Explain the round robin scheduling algorithms with respect to small and large quantum time.
5. What do you mean by throughput?
6. What are the contents of process control block or process table?
7. Explain the difference between waiting time and turnaround time.
8. What advantage s there in having different time quantum sizes on different levels of multilevel queuing system?
9. Consider a system running ten I/O-bound tasks and one CPU-bound task. Assume that the I/O-bound tasks issue an I/O operation once for every millisecond of CPU computing and that each I/O operation takes 10 milliseconds to complete. Also assume that the context switching overhead is 0.1millisecond and that all processes are long-running tasks. What is the CPU utilization for a round-robin scheduler when:
 - a. The time quantum is 1 millisecond
 - b. The time quantum is 10 milliseconds

Self-test question list with answer key

1. Why is it important for the scheduler to distinguish I/O-bound programs from CPU-bound programs?
2. Discuss how the following pairs of scheduling criteria conflict in certain settings.
 - a) CPU utilization and response time
 - b) Average turnaround time and maximum waiting time
 - c) I/O device utilization and CPU utilization
3. Which of the following scheduling algorithms could result in starvation?
 - a. First-come, first-served
 - b. Shortest job first
 - c. Round robin
 - d. Priority
4. Consider a variant of the RR scheduling algorithm where the entries in the ready queue are pointers to the PCBs.
 - a) What would be the effect of putting two pointers to the same process in the ready queue?
 - b) What would be the major advantages and disadvantages of this scheme?
 - c) How would you modify the basic RR algorithm to achieve the same effect without the duplicate pointers?
5. Can a multithreaded solution using multiple user-level threads achieve better performance on a multiprocessor system than on a single-processor system?
6. Consider a system implementing multilevel queue scheduling. What strategy can a computer user employ to maximize the amount of CPU time allocated to the user's process?
7. Consider a preemptive priority scheduling algorithm based on dynamically changing priorities. Larger priority numbers imply higher priority. When a process is waiting for the CPU (in the ready queue, but not running), its priority changes at a rate α ; when it is running, its priority changes at a rate β . All processes are given a priority of 0 when they

enter the ready queue. The parameters α and β can be set to give many different scheduling algorithms.

- a. What is the algorithm that results from $\beta > \alpha > 0$?
 - b. What is the algorithm that results from $\alpha < \beta < 0$?
8. Explain the differences in the degree to which the following scheduling algorithms discriminate in favor of short processes:
- a. FCFS
 - b. RR
 - c. Multilevel feedback queues
9. Identify the process scheduling criteria.

Answer Key

1. I/O-bound programs have the property of performing only a small amount of computation before performing IO. Such programs typically do not use up their entire CPU quantum. CPU-bound programs, on the other hand, use their entire quantum without performing any blocking IO operations. Consequently, one could make better use of the computer's resources by giving higher priority to I/O-bound programs and allow them to execute ahead of the CPU-bound programs.
2.
 - CPU utilization and response time: CPU utilization is increased if the overheads associated with context switching are minimized. The context switching overheads could be lowered by performing context switches infrequently. This could however result in increasing the response time for processes.
 - Average turnaround time and maximum waiting time: Average turnaround time is minimized by executing the shortest tasks first. Such a scheduling policy could however starve long-running tasks and thereby increase their waiting time.
 - I/O device utilization and CPU utilization: CPU utilization is maximized by running long-running CPU-bound tasks without performing context switches. I/O device

utilization is maximized by scheduling I/O-bound jobs as soon as they become ready to run, thereby incurring the overheads of context switches.

3. Shortest job first and priority-based scheduling algorithms could result in starvation.
4.
 - a) In effect, that process will have increased its priority since by getting time more often it is receiving preferential treatment.
 - b) The advantage is that more important jobs could be given more time, in other words, higher priority in treatment. The consequence, of course, is that shorter jobs will suffer.
 - c) Allot a longer amount of time to processes deserving higher priority. In other words, have two or more quantum possible in the Round-Robin scheme.
5. A multithreaded system comprising of multiple user-level threads cannot make use of the different processors in a multiprocessor system simultaneously. The operating system sees only a single process and will not schedule the different threads of the process on separate processors. Consequently, there is no performance benefit associated with executing multiple user-level threads on a multiprocessor system.
6. The program could maximize the CPU time allocated to it by not fully utilizing its time quantum. It could use a large fraction of its assigned quantum, but relinquish the CPU before the end of the quantum, thereby increasing the priority associated with the process.
7. a. FCFS b. LIFO
8. a. FCFS—discriminates against short jobs since any short jobs arriving after long jobs will have a longer waiting time.
b. RR—treats all jobs equally (giving them equal bursts of CPU time) so short jobs will be able to leave the system faster since they will finish first.

c. Multilevel feedback queues—work similar to the RR algorithm— they discriminate favorably toward short jobs.
9. The Process scheduling criteria are Utilization, throughput, turnaround time, waiting time, response time and relative delay.

Unit Three

Memory Management

Summary

The idea behind memory management is very helpful to the readers and programmers so that they will write a program in a way that there will be efficient memory utilization. For there has to be an efficient memory utilization one has to understand about multiprogramming concept in which a number of processes can be executed. There are two classes of memory allocation methods: contiguous and non-contiguous. Beside these, the virtual memory technique is used to enhance the memory utilization efficiency along with swapping. However, the memory management concepts like demand paging and page fault are the concepts that will be raised with respect to virtual memory. If page fault occurred, then the lost pages will be replaced through the efficient page replacement algorithms. Page will be swapped in from hard disk whenever the page is required in the main memory.

General Objective

To understand how the memory has to be managed so as to be efficient in memory utilization.

Specific Objective

At the end of this unit the readers are expected to:

- Explain mono-programming with respect to memory allocation
- Understand the binding and linking with respect to memory management
 - Distinguish between binding and linking.
 - Distinguish between compile time and run time binding.
 - Distinguish between compile time and run time linking.
- Explain an overlay technique of memory allocation.
- Explain the memory requirements
 - Identify all the memory requirements for efficient memory management.
- Explain how the memory location address is handled.
 - Distinguish between logical address and physical address.
 - Distinguish between logical address space and physical address space.
- Explain the concept behind fragmentation
 - Explain what causes fragmentation.
 - Distinguish between internal and external fragmentation.
- Understand the concepts behind compaction or defragmentation
 - Explain defragmentation
 - Identify the effects of defragmentation
- Identify the contiguous memory allocation methods
- Explain the
 - Single Partition memory allocation method
 - Multiple Partition Memory allocation method

St. Mary's University College

Faculty of Informatics

- Fixed equal Size multiple partitions
- Fixed variable Size multiple partitions
 - FIRST – FIT Algorithm
 - BEST – FIT Algorithm
 - WORST – FIT Algorithm
- Explain the dynamic Memory allocation method
 - Explain the advantages that it provides with respect to fixed memory allocation methods
- Explain the non-contiguous memory allocation methods in terms of
 - Paging
 - Segmentation
 - Paging with segmentation
 - Paging, segmentation and translation look aside buffer
- Identify the difference between paging and segmentation.
- Distinguish between page fault and demand paging.
- Identify all the page replacement algorithms.
- Explain Belady's Anomaly with respect to page replacement algorithms.
- Identify the advantages and disadvantages of paging and segmentation.

Self-test question list without answer key

1. Explain the difference between static and dynamic loading.
2. Define the following terms:
 - a) Linking
 - b) Swapping
 - c) Logical address space
 - d) Physical address space
 - e) Paging
 - f) Fragmentation
 - g) De-fragmentation
 - h) Segmentation

3. What does mean by virtual memory?
4. What is (are) the purpose (s) of multiprogramming?
5. Explain the difference between internal and external fragmentation.
6. Compare the main memory organization schemes of contiguous memory allocation, pure segmentation, and pure paging with respect to the following issues:
 - a) external fragmentation
 - b) internal fragmentation
 - c) ability to share code across processes

Self-test question list with answer key

1. What are the advantages of virtual memory?
2. What does mean by page fault?
3. What is the main drawback of paging?
4. What are the memory management requirements?
5. Calculate the swap time of a process from hard disk to main memory if the process size is 8M bits and the speed of backing store is 5 Mbps?
6. Given memory partitions of 100KB, 500KB, 200KB, 300KB and 600KB (in order). How would each of the FIRST FIT Algorithm, BEST FIT Algorithm and the WORST FIT Algorithm place processes of 212KB, 417KB, 112KB and 426KB (in order)?
 - a) Calculate the internal and external fragmentation of each algorithm?
 - b) Which algorithm has the least internal fragmentation?
 - c) Which algorithm makes the most efficient use of the memory?

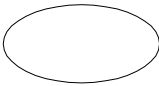
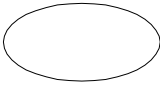
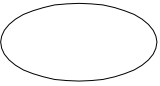










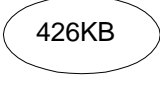
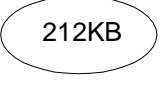
Answer Key

1. No external fragmentation, Higher degree of multiprogramming and Large virtual address space
2. If the required page is not available in main memory, we call it page fault.
3. Suffering from page break.
4. The memory management requirement includes relocation, protection, sharing, logical organization and physical organization.
5. Process size = 8×10^6 bits

Speed = 5Mbps = 5×10^6 bits/second

Swap time = $(8 \times 10^6 \text{ bits}) / (5 \times 10^6 \text{ bits/second})$

6.

	FIRST-FIT		BEST-FIT		WORST-FIT
100KB		100KB		100KB	
500KB		500KB		500KB	
200KB		200KB		200KB	
300KB		300KB		300KB	
600KB		600KB		600KB	

a) **FIRST-FIT Algorithm**

Internal fragmentation = $(500-212) + (200-112) + (600-417)$

St. Mary's University College

Faculty of Informatics

$$= 288\text{KB} + 88\text{KB} + 183\text{KB}$$

$$= 559 \text{ KB}$$

$$\text{External fragmentation} = 300\text{KB} + 100\text{KB}$$

$$= 400\text{KB}$$

BEST -FIT Algorithm

$$\text{Internal fragmentation} = (500 - 417) + (200 - 112) + (300 - 212) + (600 - 426)$$

$$= 83\text{KB} + 88\text{KB} + 88\text{KB} + 174\text{KB}$$

$$= 433\text{KB}$$

$$\text{External fragmentation} = 100\text{KB}$$

WORST-FIT Algorithm

$$\text{Internal fragmentation} = (500 - 417) + (300 - 112) + (600 - 212)$$

$$= 83\text{KB} + 188\text{KB} + 388\text{KB}$$

$$= 659 \text{ KB}$$

$$\text{External fragmentation} = 100\text{KB} + 200\text{KB}$$

$$= 300\text{KB}$$

b) BEST -FIT Algorithm

c) BEST -FIT Algorithm

Unit Four

Input – Output Management

Summary

The input – output management is achieved through the I/O subroutine in the operating system. The I/O devices are directly controlled by the respective device's controller. However, there must be the mediator software/ routine that enable the operating system to management the I/O devices. There are three techniques of I/O devices communication modes: Programmed I/O mode, Interrupt-driven mode and DMA. The speed of I/O devices is not compatible to the speed of the CPU and this will be solved by using buffering. The buffering will be attained through different types of buffers like single buffer, double buffer and circular buffer. The disk accessing is made using different disk scheduling algorithms.

General Objective

To understand how the operating system manages the input – output devices.

Specific Objective

At the end of this unit the readers are expected to:

- Explain the I/O Devices
- Explain the organization of the I/O function
 - Explain the I/O organization.
 - Identify all the possible I/O modes of communication.
- Explain Operating System Design Issues.
 - Identify all operating system design issues.
- Explain the I/O Buffering.

- Explain the uses of I/O buffering.
- Identify all types of buffers.
- Explain the Disk Scheduling.
 - Identify all disk scheduling algorithms.
 - Identify which disk scheduling algorithms are the most used in today's operating system.
- RAID
 - Identify the two fundamental different objectives served by RAID
 - Identify all the possible levels of RAID.
 - Distinguish the advantages and disadvantages of each levels of RAID.

Self-test question list without answer key

1. Explain the modes of I/O operation.
2. What is the use of interrupt?
3. Explain the DMA operation.
4. In what parameters do you think that I/O devices will differ?
5. Write the two events in which the DMAC sends the interrupt signal to the CPU.
6. List the three types of buffering.
7. Write the difference between sector sparing and sector slipping.
8. Explain the difference between low-level formatting and high- level formatting.
9. What does mean by bad sectors and how do you think it will be handled?
10. List the disk scheduling algorithms and explain them using example.
11. What is the difference between SCAN and C-SCAN disk scheduling algorithm?

Self-test question list with answer key

1. How does DMA increase system concurrency? How does it complicate hardware design?

2. Suppose that a disk drive has 5000 cylinders, numbered 0 to 4999. The drive is currently serving a request at cylinder 143, and the previous request was at cylinder 125. The queue of pending requests, in FIFO order, is

86, 1470, 913, 1774, 948, 1509, 1022, 1750, 130

Starting from the current head position, what is the total distance (in cylinders) that the disk arm moves to satisfy all the pending requests, for each of the following disk-scheduling algorithms?

a) FCFS b) SSTF c) SCAN d) LOOK e) C-SCAN

3. Compare the performance of C-SCAN and SCAN scheduling, assuming a uniform distribution of requests. Consider the average response time (the time between the arrival of a request and the completion of that request A's service), the variation in response time, and the effective bandwidth. How does performance depend on the relative sizes of seek time and rotational latency?
4. Explain why SSTF scheduling tends to favor the middle cylinders of a disk over the innermost and outermost cylinders.
5. None of the disk-scheduling disciplines, except FCFS, is truly fair (starvation may occur).
- A. Explain why this assertion is true.
 - B. Describe a way to modify algorithms such as SCAN to ensure fairness.
 - C. Explain why fairness is an important goal in a time-sharing system.
 - D. Give three or more examples of circumstances in which it is important that the operating system be unfair in serving I/O requests.

Answer Key

1. DMA increases system concurrency by allowing the CPU to perform tasks while the DMA system transfers data via the system and memory busses. Hardware design is complicated because the DMA controller must be integrated into the system, and the system must allow the DMA controller to be a bus master. Cycle stealing may also be necessary to allow the CPU and DMA controller to share use of the memory bus.

2.

- a) The FCFS schedule is 143, 86, 1470, 913, 1774, 948, 1509, 1022, 1750, 130.
The total seek distance is 7081.
- b) The SSTF schedule is 143, 130, 86, 913, 948, 1022, 1470, 1509, 1750, 1774.
The total seek distance is 1745.
- c) The SCAN schedule is 143, 913, 948, 1022, 1470, 1509, 1750, 1774, 4999, 130, 86.
The total seek distance is 9769.
- d) The LOOK schedule is 143, 913, 948, 1022, 1470, 1509, 1750, 1774, 130, 86.
The total seek distance is 3319.
- e) The C-SCAN schedule is 143, 913, 948, 1022, 1470, 1509, 1750, 1774, 4999, 86, 130.
The total seek distance is 9813.
- f) The C-LOOK schedule is 143, 913, 948, 1022, 1470, 1509, 1750, 1774, 86, 130.
The total seek distance is 3363.

3. There is no simple analytical argument to answer the first part of this question. It would make a good small simulation experiment for the students. The answer can be found in Figure 2 of Worthington et al. [1994]. (Worthington et al. studied the LOOK algorithm, but similar results obtain for SCAN. Figure 2 in Worthington et al. shows that C-LOOK has an average response time just a few percent higher than LOOK but that C-LOOK has a significantly lower variance in response time for medium and heavy workloads. The intuitive reason for the difference in variance is that LOOK (and SCAN) tends to favor requests near the middle cylinders, whereas the C-versions do not have this imbalance. The intuitive reason for the slower response time of C-LOOK is a circular seek from one end of the disk to the farthest request at the other end. This seek satisfies no requests. It only causes a small performance degradation because the square-root dependency of seek time on distance implies that a long seek isn't terribly expensive by comparison with moderate length seeks. For the second part of the question, we observe that these algorithms do not schedule to improve rotational latency; therefore, as seek times

decrease relative to rotational latency, the performance differences between the algorithms will decrease.

4. The SSTF algorithm is biased toward the middle cylinders in much the same way the SCAN algorithm is. Since, in the SSTF algorithm, the closest cylinder is always chosen, then all middle cylinder references will be serviced on the way to the end of the disk. The center of the disk is the location having the smallest average distance to all other tracks. Thus the disk head tends to move away from the edges of the disk. Here is another way to think of it. The current location of the head divides the cylinders into two groups. If the head is not in the center of the disk and a new request arrives, the new request is more likely to be in the group that includes the center of the disk; thus, the head is more likely to move in that direction.
5.
 - A. New requests for the track over which the head currently resides can theoretically arrive as quickly as these requests are being serviced.
 - B. All requests older than some predetermined age could be "forced" to the top of the queue, and an associated bit for each could be set to indicate that no new request could be moved ahead of these requests. For SSTF, the rest of the queue would have to be reorganized with respect to the last of these "old" requests.
 - C. To prevent unusually long response times.
 - D. Paging and swapping should take priority over user requests. It may be desirable for other kernel-initiated I/O, such as the writing of file system metadata, to take precedence over user I/O. If the kernel supports real-time process priorities, the I/O requests of those processes should be favored.

Unit Five

Deadlock

Summary

In multi-programming environment several processes request the resources from the operating system. If the resources are available at that time, then the operating system grants the resources to that processes. If not available, the process has to wait until the resources are released. Sometimes the processes are not released; the resources are blocked by some other processes. This situation is said to be deadlock.

General Objective

To understand the effects of inter-process communication during accessing the resources, especially the non-preemptable resources

Specific Objective

At the end of this unit the readers are expected to:

- Explain the concept of dead lock.
- Identify the type of resources that cause the deadlock to occur.
- Identify the four conditions that lead to the deadlock.
- Identify all the strategies to deal with the deadlock.
- Identify deadlock detection and recovery methods.
- Explain about the banker's algorithm.
- Explain the safety algorithm.

Self-test question list without answer key

1. Consider a system consisting of 4 resources of the same type that are shared by 3 processes, each of which needs at most two resources. Show that the system is deadlock-free.
2. N processes share M resource unit that can be reserved and released only one at a time. The maximum need of each process doesn't exceed M, and the sum of all maximum needs is less than M + N.
3. Consider the snapshot of the system.

Process	Allocation	Max	Available
	A B C D	A B C D	A B C D
P1	0 0 1 2	0 0 1 2	2 1 0 0
P2	2 0 0 0	2 7 5 0	
P3	0 0 3 4	6 6 5 6	
P4	2 3 5 4	4 3 5 6	
P5	0 3 3 2	0 6 5 2	

- a) Calculate the Need matrix.
 - b) Is the system currently in safe state or unsafe state? Why?
 - c) Is the system currently deadlocked? Why or why not?
 - d) Which process, if any, or may become deadlocked?
4. Consider the following snapshot of a system.

Process	Allocation	Max	Available
	A B C D	A B C D	A B C D
P1	0 0 1 2	0 0 1 2	1 5 2 0
P2	1 0 0 0	1 7 5 0	
P3	1 3 5 4	2 3 5 6	
P4	0 6 3 2	0 6 5 2	
P5	0 0 1 4	0 6 5 6	

- What is the content of the need matrix?
 - Is the system in safe state?
 - If a request from P1 arrives for (0, 4, 2, 0) can the request be granted immediately?
5. Explain about the deadlock prevention methods.

Self-test question list with answer key

- What are the states of the resources?
- What do you mean by safe state?
- What are the necessary conditions for a deadlock to happen?
- List the four strategies used to deal with deadlocks.
- Identify the deadlock prevention methods?

Answer Key

- Request , Use and Release

St. Mary's University College

Faculty of Informatics

2. A safe state is a state of the system in which the system can allocate resources to equal process in some order and still avoid a deadlock.
3. Mutual exclusion, No preemption, Hold and wait and Circular wait.
4. Ignoring Deadlock /Ostrich Algorithm/, Deadlock Avoidance, Deadlock Prevention and Deadlock Detection and Recovery
5. Removing the Mutual exclusion, removing the No preemption, removing the Hold and wait and removing the Circular wait conditions.

File Management

Summary

Computers can store information on different storage media, such as magnetic disks, tapes, and compact disks. The physical storage is converted into logical storage unit by the operating system. The logical storage unit is said to be file. To perform all these activities the operating system must include the file management system, which is the collection of system software that provides services to the users. The files are allocated in the storage media using contiguous method, linked method and grouped method. However, the files are accessed through Sequential Method, Direct Method and Indexed Sequential Method.

General Objective

To understand the file management system that provides services to the users.

Specific Objective

At the end of this unit the readers expected to:

- Define file and file management system.
- Identify the objectives of file management.
- Identify all the typical file attributes.
- Identify the file operation.
- Explain the file directory and directory structure.
- Identify the possible operation performed on directory.
- Identify the levels of directory.
- Identify all the file access methods and file allocation methods.

- Explain the free space management.
- Identify the free space management methods.

Self-test question list without answer key

1. Explain about the file management system.
2. Explain about the level of directory system.
3. List the file operation with example.
4. List the free space management methods.

Self-test question list with answer key

1. List out the types of file allocation methods.
2. Consider a system that supports the strategies of contiguous, linked, and indexed allocation. What criteria should be used in deciding which strategy is best utilized for a particular file?
3. What are the advantages of the variation of linked allocation that uses a FAT to chain together the blocks of a file?

Answer Key

1. Contiguous method, Linked method and Grouped method
2. Contiguous -- if file is usually accessed sequentially, if file is relatively small.

Linked -- if file is large and usually accessed sequentially

Indexed -- if file is large and usually accessed randomly

3. The advantage is that while accessing a block that is stored at the middle of a file, its location can be determined by chasing the pointers stored in the FAT as opposed to accessing all of the individual blocks of the file in a sequential manner to find the pointer to the target block. Typically, most of the FAT can be cached in memory and therefore the pointers can be determined with just memory accesses instead of having to access the disk blocks.

St. Mary's University College

Faculty of Informatics

Part II

Computer Organization & Assembly Language Programming

Unit One

Basic Computer Organization and Design

Summary

This unit introduce basic computer and shows how its operation can be specified with register transfer statements. The organization of the computer is defined by its internal registers, the timing and control structure and the set of instructions that it uses. The design of the computer s then carried out in detail. Although the basic computer presented in this unit is very small compared to commercial computers, it has the advantage of being simple enough so that it can demonstrate the design process without too many complications.

General Objective

To understand how to design the control logic circuits of registers in basic computer.

Specific Objective

At the end of this chapter the readers are expected to:

- Distinguish between instruction codes and operation codes
- Identify the basic computer registers with their respective size.
- Distinguish between direct and indirect addressing modes
- Explain the common bus system in basic computer
- Identify the possible computer instructions
- Explain the control unit of basic computer
- Explain the instruction cycle
- Distinguish between register-reference instructions and memory-reference instructions
- Explain the input-output instructions
- Explain the program interrupts

Self-test question list without answer key

1. Design the control logic circuits for memory.
2. Design the control logic circuits for accumulator (AC).
3. Design the control logic circuits for Program Counter (PC).
4. Design the control logic circuits for temporary register (TR).
5. Design the control logic circuits for data register (DR).

Self-test question list with answer key

1. Derive the Boolean expression for control logic circuits of address register (AR).
2. Derive the Boolean expression for the control logic circuits of memory read.
3. Derive the Boolean expression for the control logic circuits of memory write.
4. What are the two instructions needed in the basic computer in order to set the E flip-flop to 1?

Answer Key

1. The Boolean expression for the control logic of AR is given as

$$LD(AR) = R'T_0 + R'T_2 + D'_7IT_3$$

$$CLR(AR) = RT_0 \text{ and } INR(AR) = D_5T_4$$

2. The Boolean expression for the control logic of memory read is given as

$$Read(memory) = R'T_1 + D'_7IT_3 + (D_0 + D_1 + D_2 + D_6)T_4$$

3. The Boolean expression for the control logic of memory write is given as

$$Write(memory) = RT_1 + D_3T_4 + D_5T_4 + D_6T_6$$

4. CLE (Clear E) and CME (Complement E)

Input – Output Processor (IOP)

Summary

In computer system there are two major tasks, which are program execution and I/O operations. The program execution is the responsibility of CPU and I/O operation is the responsibility of the IOP and still the CPU is responsible for initiating the I/O devices and IOP. Input – Output processor is the co-processor that helps the CPU to handle the data transfer operation through DMA.

General Objective

To understand the co-processor that enhances the performance of the CPU during the I/O data transfers.

Specific Objective

At the end of the unit the readers are expected to:

- Identify the I/O devices.
- Explain the device controller.
- Explain the I/O controls methods.
- Identify the I/O communication modes.
- Distinguish between Programmed I/O mode, Interrupt – driven mode and Direct Memory access (DMA) mode.
- Identify the I/O addressing methods.
- Explain the DMA operation.
- Explain the input-output processor with schematic diagram.
- Explain the CPU-IOP communication with schematic diagram.

Self-test question list without answer key

1. Explain the DMA with a block diagram.
2. Write the interrupt-driven I/O communication mode.
3. List the DMA operation using example.
4. When do you think that the DMAC sends an interrupt signal to CPU?
5. Why do you think that the DMAC sends interrupt signal to CPU?

Self-test question list with answer key

1. List out the three different I/O communication modes.
2. What is the drawback of programmed I/O mode?
3. List the different types of interrupts.
4. What is the disadvantage of interrupt-driven and DMA I/O communication modes.

Answer Key

1. Three I/O communication modes are Programmed I/O mode, Interrupt – driven I/O mode and direct memory access (DMA).
2. The CPU idle time is very high which in turn reduces the throughput.
3. The interrupt exist as program interrupts, timer interrupts, I/O interrupts and hardware interrupts.
4. When the data is transferred completely (DC= 0) and when the status of I/O devices changes from ready to not ready.
5. To return the system bus back to the CPU.

Unit Three

Parallel Processing, Pipeline and Computer Set Instruction

Summary

The speed at which the operation performed in computer system depends on the speed of processor. But later the system performance of the system was not improved that much, hence the parallel processing concepts has be launched to entertains more than one processor. Michael Flynn's has come up with four different types of architecture: SISD, MISD, SIMD and MIMD. However, all the instruction were not suiting parallel processing and decomposing the sequential instruction into different independent segments to prepare the instruction that suits for parallel processing was another challenging issue. Then the pipelining has been introduced to solve this problem. The computer set instruction exists as RISC and CISC.

General Objective

To understand parallel computing, pipelining and instruction set based computer.

Specific Objective

At the end of this unit the readers are expected to:

- Distinguish between serial execution and parallel processing.
- Identify architecture of parallel computing.
- Identify various form of memory Architecture of Parallel Computing.
- Identify the factors to measure performance of parallel programs.
- Distinguish between parallel computing and pipeline.
- Identify the types of pipelining.
- Identify the types of instruction set computers.

Self-test question list without answer key

1. Explain the parallel processing?
2. Write the factors that measures the performance of parallel processing.
3. Explain the characteristics of RISC and CISC.
4. What is the difference between serial processing and parallel processing?
5. Identify which instruction set computer is available in today's computer.

Self-test question list with answer key

1. List out the Flynn's computer architecture?
2. Write two types of instruction set computer?
3. What are the two types of pipeline?
4. What is pipeline?
5. List the different memory architecture for parallel processing / computing.

Answer Key

1. SISD, SIMD, MISD and MIMD
2. RISC and CISC
3. Arithmetic pipeline and Instruction pipeline
4. It is the techniques of decomposing sequential instruction into different independent segments.
5. Shared memory, distributed memory and hybrid memory.

Assembly Language Programming

Summary

Applications that are not time-critical or only operate on standard input/output devices have minimal direct need for assembly language programming. Otherwise, assembly language is used to program time critical tasks. Moreover, some programmers must write the library routines to achieve standard interfaces and these routines are written in assembly language. Assembly language Programs consist of statement, one per line. Each statement is either an instruction, which the assembler translate into machine code, or an assembler directive, which instructs the assembler to perform some specific task, such as allocating memory space for a variable or creating a procedure. Both instruction and directive have up to four fields. Each line of source code consists of a single statement, of up to 512 characters. Statements can be entered in either uppercase or lowercase letters. Each field (except the comment field) must be separated from other fields by space or tab character.

General Objective

To understand how the system level programming and real life application is possible using low level programming languages.

Specific Objective

At the end of this unit the readers are expected to:

- Identify the components of 8086 microprocessor
- Distinguish between Execution unit and Bus interface unit of 8086 microprocessor.
- Explain the assembly language programming tokens.
- Identify all the directives.
- Identify all assembly addressing modes.
- Identify all the instruction sets of 8086/8088 microprocessor

Self-test question list without answer key

1. Write an ALP which reads a string of characters from the keyboard using the 01H function and display the total number of words in the string. Use MASM
2. Write an ALP to copy a source string defined in data segment to a destination of size 20 using MOVSB, STD and LOOP instructions. Use TASM
3. Implement the following pseudocode in assembly language. All values are unsigned:
if (BX <= CX && CX > DX)
 {
 AX = 5;
 DX = 6;
 }
4. Write the formula used to find the effective address of the following instructions?
a) MOV AX, [BX] [SI]
b) MOV BX, 50H [BX] [SI]
5. Assuming that CX contains 0055H when the instruction **DEC CH** is executed, what will CX contains and how will the flags CF, PF, SF, and ZF be set afterwards if all were Zero before execution of instruction.

Self-test questions with answer key

1. If **MSG DB 'WELCOME TO ALP EXAMINATIONS\$'** is defined in data segment, then how many bytes of memory is allocated to a data variable **MSG**?
2. Which directive declares the data variable during assembly programming?
3. Given the following assembly program

```
MOV CX, 0004H  
MOV AX, 0002H  
UP: ADD AX, CX
```

What is the content of

- a) AX register. -----
- b) CX register. -----
- c) DX register. -----
- d) BX register. -----

DEC CX

XCHG AX, CX

JCXZ DOWN

JMP UP

DOWN: XOR AX, AX

AND AX, CX

MOV DX, AX

ADD DX, CX

4. What is the content of CX after the execution of the following instruction if CX contains 38FEH? **XOR CX, CX**
5. Re-write the following array declaration in its shortest valid form?
 - a) Buffer DB 5DUP (0), 5DUP (?) b) Buffer DB 5DUP(5DUP(5DUP(4)))
6. Given,

MSG DB 'WELCOME TO ALP LAB SESSION\$'

Then, what is the purpose of the following two instructions?

- a) LEA DX, MSG b) MOV DX,OFFSET MSG
7. If CS contains $A487_{16}$ and IP contains 1436_{16} , then what is the physical address of the next instruction in memory to be executed?
8. What physical memory location is accessed by the instruction MOV [BP], AL if BP = 2C50H and SS = 5D27H?
9. Write an ALP which copies the string defined in the data segment to the destination using 'STOSB'.

Answer Key

1. One byte
2. Data directive
3. The content of
 - a) AX is Unknown
 - b) CX is Unknown
 - c) DX is Unknown
 - d) BX is Unknown

4. The content of CX is 0000H
5. a) Buffer DB 5DUP (0,?) b) Buffer DB 125DUP(5DUP(4)
6. Both instructions are used to load the offset address of variable MSG to DX register.
7. The physical address is A5CA6H
8. The physical memory location address is 5FEC0H.
9. The TASM based program for the problem is:

```
.MODEL SMALL

.STACK 100H

.DATA

    SOURCE_STRING  DB 'COLLEGE $'
    DESTINATION_STRING DB 20 DUP (?)

.CODE

.STARTUP

    MOV AX, @DATA
    MOV DS, AX
    MOV ES, AX

    LEA SI, SOURCE_STRING
    MOV CL, 07
    LEA DI, DESTINATION_STRING
    CLD
UP:  MOV AL,[SI]
    STOSB
    LOOP UP

    LEA DX, DESTINATION_STRING
    MOV AH, 09H
    INT 21H
```

St. Mary's University College

Faculty of Informatics

MOV AH, 4CH

INT 21H

END