Reg No.										

MANIPAL INSTITUTE OF TECHNOLOGY

(Constituent Institute of MAHE- Deemed University)
MANIPAL-576104

FIFTH SEM B. E.(CSE) MAKEUP EXAMINATION JAN – 2007

OPERATING SYSTEMS AND UNIX

(CSE –307)
(10 POINT CREDIT SYSTEM)

TIME DURATION : 3 HOUR MAX.MARKS : 50

Instructions to Candidates

- Answer **ANY FIVE** full questions.

1 a) Write a note on dual mode of operation.
b) What is a system call? List out various UNIX system calls related to file management.
c) Give the sequence of steps in a typical system booting.
d) What is a Process Control Block? Explain each field in the PCB.
(3+2+2+3)

2 a) Explain the Inter_process Communication using mailboxes.
b) What are the benefits of Multithreading? Write a note on thread scheduling.
c) What is the function of dispatcher? Explain the method of predicting the next CPU burst time in case of SJF process scheduling algorithm.
d) On a system using round_robin (RR) scheduling, let ‘s’ represents the time needed to perform a process switch, ‘q’ represent the RR time quantum, and ‘r’ represent the average time a process runs before blocking on I/O. Give a formula for CPU efficiency given the following:
i) q>r ii) s<q<r (2+3+3+2)

3) a) What is meant by race condition of concurrent access ? Explain with an example.
b) Define the swap() instruction for process synchronization and give an algorithm that satisfies the mutual exclusion requirement of critical section problem using swap() instruction.

c) It is required to allocate a single resource among competing processes. Each process, when requesting an allocation of this resource, specifies the maximum time it plans to use the resource. Define a monitor to allocate the resource on priority bases where priority is inversely proportional to the time of usage of the resource.
d) What are the synchronization methods used in Linux version 2.6 onwards.
e) Give the definition of semaphore operations that encounter no busy wait loops. (2+2+2+2+2)

4 a) What are the necessary conditions for deadlocks to occur? Explain various protocols to prevent deadlocks by ensuring that the Hold and Wait condition never occur.
b) Consider the following snapshot of a system:

	Allocation				**Max**			
	A	B	C	D	A	B	C	D
P0	0	0	1	2	0	0	1	2
P1	1	0	0	0	1	7	5	0
P2	1	3	5	4	2	3	5	6
P3	0	6	3	2	0	6	5	2
P4	0	0	1	4	0	6	5	6

Available			
A	B	C	D
1	5	2	0

Using the banker's algorithm, If the request from process P1 arrives for (0,4,2,0), can the request be granted immediately ? Show all the calculations.
c) Distinguish the logical and physical address space.
d)What is external fragmentation ? discuss different ways of handling the problem of external fragmentation. (3+3+2+2)

5 a) Give the structure of a typical inverted page table and List out the advantages and disadvantages of Inverted page table structures.
b) Explain the paging in the Pentium architecture.
c) Explain the following page replacement algorithms:
i) Enhanced second chance ii) LFU
d) what is meant by thrashing ? Explain the methods of controlling thrashing. (2+3+2+3)

6 a) Write a note on tree-structured directories.
b)on a disk with 1000 cylinders numbers 0 to 999, compute the number of tracks the disk arm must move to satisfy all the requests in the disk queue. Assume the last request serviced was at track 756 and the head is moving towards track 0. The queue in FIFO order contains requests for the following tracks: 811, 348, 153, 968, 407, 500. Perform the computation for the following disk scheduling algorithm.
i) SSTF ii) SCAN
c) Write a note on program and system threats.
d) Explain the process scheduling in Linux operating system.

(2+2+3+3)

SCHEME OF VALUATION

1a) **Dual-mode** operation allows OS to protect itself and other system components.

User mode and **kernel mode**

Mode bit provided by hardware

- Provides ability to distinguish when system is running user code or kernel code.
- Some instructions designated as **privileged**, only executable in kernel mode.
- System call changes mode to kernel, return from call resets it to user. **3 Marks**

b) Programming interface to the services provided by the OS.
Typically written in a high-level language (C or C++).
Mostly accessed by programs via a high-level **Application Program Interface (API)** rather than direct system call use.

UNIX system calls related to file management………….**2 Marks**

c) *Booting* – starting a computer by loading the kernel.
Bootstrap program – code stored in ROM that is able to locate the kernel, load it into memory, and start its execution.

Operating system must be made available to hardware so hardware can start it

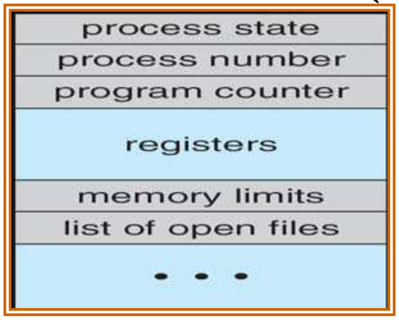
Small piece of code – **bootstrap loader**, locates the kernel, loads it into memory, and starts it.

Sometimes two-step process where **boot block** at fixed location loads bootstrap loader.

When power initialized on system, execution starts at a fixed memory location.

Firmware used to hold initial boot code. **2 Marks** d)

Process Control Block (PCB)



process state
process number
program counter
registers
memory limits
list of open files
• • •

Information associated with each process

Process state
Program counter
CPU registers
CPU scheduling information
Memory-management information
Accounting information
I/O status information **3 Marks**

2a) Messages are directed and received from mailboxes (also referred to as ports).

Each mailbox has a unique id.
Processes can communicate only if they share a mailbox.

Properties of communication link

Link established only if processes share a common mailbox.
A link may be associated with many processes.
Each pair of processes may share several communication links.
Link may be unidirectional or bi-directional.

Operations

create a new mailbox.
send and receive messages through mailbox.
destroy a mailbox.

Primitives are defined as:

send(*A*, *message*) – send a message to mailbox A.
receive(*A*, *message*) – receive a message from mailbox A.

Mailbox sharing

P1, *P2*, and *P3* share mailbox A

P1, sends; *P2* and *P3* receive

Who gets the message?

Solutions

Allow a link to be associated with at most two processes.

Allow only one process at a time to execute a receive operation.

Allow the system to select arbitrarily the receiver. Sender is notified who the receiver was. **2 Marks**

b) **Benefits**

Responsiveness

Resource Sharing

Economy

Utilization of MP Architectures

Note on thread scheduling **3 Marks**

c) Dispatcher module gives control of the CPU to the process selected by the short-term scheduler; this involves:

switching context

switching to user mode

jumping to the proper location in the user program to restart that program

Dispatch latency – time it takes for the dispatcher to stop one process and start another running.

Associate with each process the length of its next CPU burst. Use these lengths to schedule the process with the shortest time

Two schemes:

nonpreemptive – once CPU given to the process it cannot be preempted until completes its CPU burst.

preemptive – if a new process arrives with CPU burst length less than remaining time of current executing process, preempt. This scheme is know as the

Shortest-Remaining-Time-First (SRTF)

SJF is optimal – gives minimum average waiting time for a given set of processes. **3 Marks**

d) CPU efficiency formula **2 Marks**

3a) **Race Condition**

count++ could be implemented as

```
register1 = count
register1 = register1 + 1
count = register1
```

count-- could be implemented as

```
register2 = count
register2 = register2 - 1
count = register2
```

Consider this execution interleaving with “count = 5” initially:

S0: producer execute register1 = count {register1 = 5}
S1: producer execute register1 = register1 + 1 {register1 = 6}
S2: consumer execute register2 = count {register2 = 5}
S3: consumer execute register2 = register2 - 1 {register2 = 4}
S4: producer execute count = register1 {count = 6 }
S5: consumer execute count = register2 {count = 4} **2 Marks**

b) **Swap Instruction**

Definition:

```
void Swap (boolean *a, boolean *b)
 {
     boolean temp = *a;
     *a = *b;
     *b = temp:
 }
```

Shared Boolean variable lock initialized to FALSE; Each process has a local Boolean variable key.

Solution:

```
do {
     key = TRUE;
      while ( key == TRUE)
            Swap (&lock, &key );

              //    critical section

       lock = FALSE;

            //      remainder section

     } while ( TRUE);
```

2 Marks

c) problem…………………………………………….. **2 Marks**

d) Linux:

disables interrupts to implement short critical sections

Linux provides:

semaphores
spin locks

explanation………………………………………………… **2 Marks**

e) **Semaphore Implementation with no Busy waiting**

With each semaphore there is an associated waiting queue. Each entry in a waiting queue has two data items:

value (of type integer)
pointer to next record in the list

Two operations:

block – place the process invoking the operation on the appropriate waiting queue.
wakeup – remove one of processes in the waiting queue and place it in the ready queue.

Implementation of wait:

```
wait (S){
        value--;
        if (value < 0) {
              add this process to waiting queue
               block();  }
 }
```

Implementation of signal:

```
Signal (S){
          value++;
           if (value <= 0) {
                remove a process P from the waiting queue
                 wakeup(P);  }
}
```

2 Marks

4a) Deadlock can arise if four conditions hold simultaneously.

Mutual exclusion: only one process at a time can use a resource.

Hold and wait: a process holding at least one resource is waiting to acquire additional resources held by other processes.

No preemption: a resource can be released only voluntarily by the process holding it, after that process has completed its task.

Circular wait: there exists a set {*P*0, *P*1, …, *P*0} of waiting processes such that *P*0 is waiting for a resource that is held by *P*1, *P*1 is waiting for a resource that is held by

*P*2, …, *Pn*–1 is waiting for a resource that is held by

*P*n, and *P*0 is waiting for a resource that is held by *P*0. **3 Marks**

b) Problem…………………………………………………………**3 Marks**

c) **Logical vs. Physical Address Space**

The concept of a logical *address space* that is bound to a separate *physical address space* is central to proper memory management

Logical address – generated by the CPU; also referred to as *virtual address*.

Physical address – address seen by the memory unit

Logical and physical addresses are the same in compile-time and load-time address-binding schemes; logical (virtual) and physical addresses differ in execution-time address-binding scheme.

Hardware device that maps virtual to physical address.

In MMU scheme, the value in the relocation register is added to every address generated by a user process at the time it is sent to memory.

The user program deals with *logical* addresses; it never sees the *real* physical addresses. **2 Marks**

d)

External Fragmentation – total memory space exists to satisfy a request, but it is not contiguous.

Reduce external fragmentation by **compaction.**

Shuffle memory contents to place all free memory together in one large block.

Compaction is possible *only* if relocation is dynamic, and is done at execution time.

I/O problem

- Latch job in memory while it is involved in I/O.
- Do I/O only into OS buffers.

Logical address space of a process can be noncontiguous; process is allocated physical memory whenever the latter is available.

Divide physical memory into fixed-sized blocks called **frames** (size is power of 2, between 512 bytes and 8192 bytes).

Divide logical memory into blocks of same size called **pages**

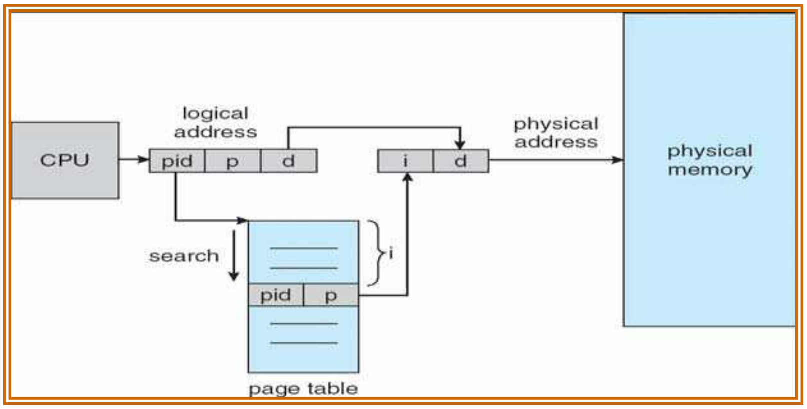
To avoid external fragmentation. **2 Marks**

5a) **Inverted Page Table**

One entry for each real page of memory.

Entry consists of the virtual address of the page stored in that real memory location, with information about the process that owns that page.

Decreases memory needed to store each page table, but increases time needed to search the table when a page reference occurs.



2 Marks

b) Paging in Pentium architecture………………………….. **3 Marks**

c) Enhanced second chance algo

LFU:

Keep a counter of the number of references that have been made to each page.

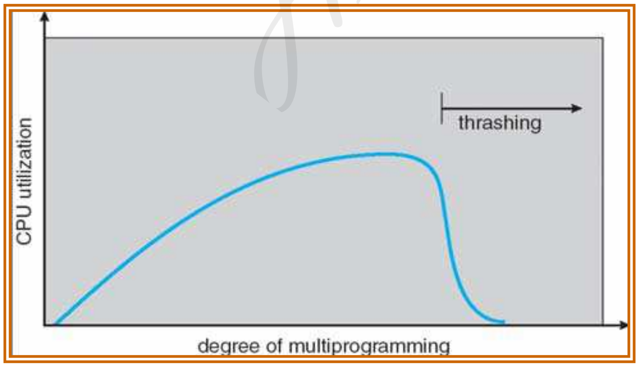
LFU Algorithm: replaces page with smallest count. **2 Marks**

d) If a process does not have "enough" pages, the page-fault rate is very high. This leads to:

low CPU utilization, operating system thinks that it needs to increase the degree of multiprogramming.

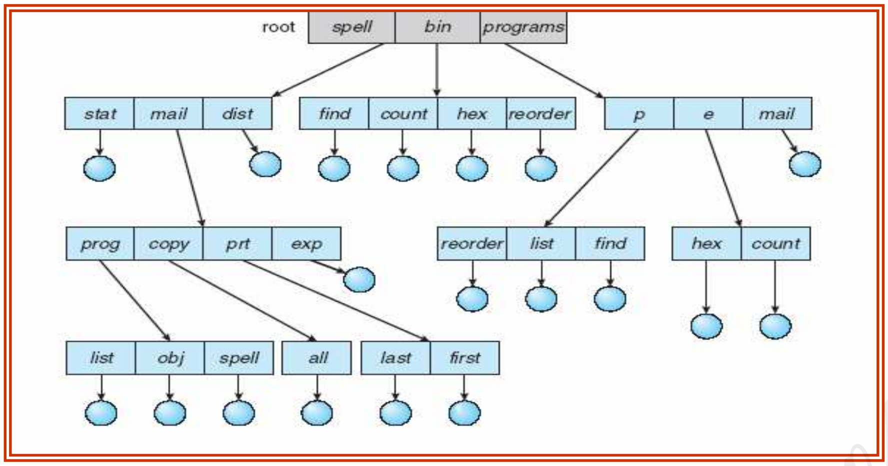
another process added to the system.

Thrashing ≡ a process is busy swapping pages in and out.



Methods to control thrashing **3 Marks**

6a)



Efficient searching.
Grouping Capability.
Current directory (working directory)
cd /spell/mail/prog
type list

Absolute or **relative** path name
Creating a new file is done in current directory.
Delete a file
rm <file-name>
Creating a new subdirectory is done in current directory
mkdir <dir-name>
Example: if in current directory /mail
mkdir count **2 Marks**

b) Disk scheduling problem……………………………. **2 Marks**

c) **Program Threats**

Viruses

Code fragment embedded in legitimate program
Very specific to CPU architecture, operating system, applications
Usually borne via email or as a macro

- Visual Basic Macro to reformat hard drive
- Sub AutoOpen()

System and Network Threats

Worms – use **spawn** mechanism; standalone program

Internet worm

Exploited UNIX networking features (remote access) and bugs in *finger* and *sendmail* programs

Grappling hook program uploaded main worm program

Port scanning

Automated attempt to connect to a range of ports on one or a range of IP addresses

Denial of Service

Overload the targeted computer preventing it from doing any useful work

Distributed denial-of-service (**DDOS**) come from multiple sites at once **3 Marks**

d) Process scheduling in Linux OS…………………………… **3 Marks**