S.M. Mithon Hossen
ID: 2121210031

P-01

CSI-231 (Operating System Concepts)
Final Examination - Summer-2022

#1.A

Ans to the qs no: 1.A

Valid bit:- A bit of information that indicates whether data in a block is valid (1) or not (0) valid indicates that the associated page is in the logical address space.

Invalid bit:- invalid indicates that the associted page is not in logical address space.

Ans to the qs no: 1.C

#1.C

Physical address:- physical address is the actual address of the data inside the memory. The logical address is a virtual address and the program needs physical memory for its execution. the user never deals with the physical address.

* Ans to the Q.no: 1.B *

1.B Architecture of segmentation:- (1) Logical address

Consist of a two tuple:

<segment-number, offset>.

Segment table:- maps two dimensional physical addresses: each table entry has:

(i) Base:- Contains the Starting physical Address where the segment reside in memory.

(ii) Limit:- specifies the length of the segment.

Segment table base register :- poins to the segment tables location in memory.

Segment table length register: indicates number of segment used by a program:

segment number s is legal if s < STLR.

Protection:-

≠ with each entry in segment table associate:

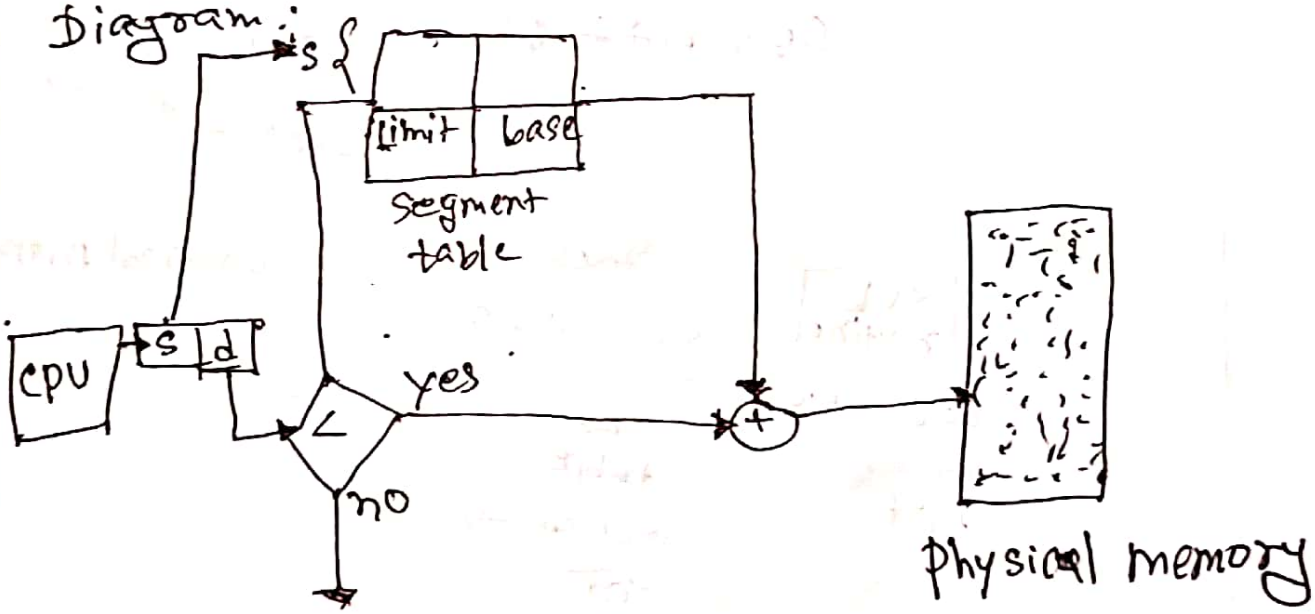
⇒ validation bit = 0 ⇒ illegal segment

≠ read/write/execute privileges.

≠ protection bits associated with segments: code sharing occurs at segment level.

≠ Since segments vary in length, memory allocation is a dynamic storage allocation problem.

⇒ A segmentation example show the following diagram:



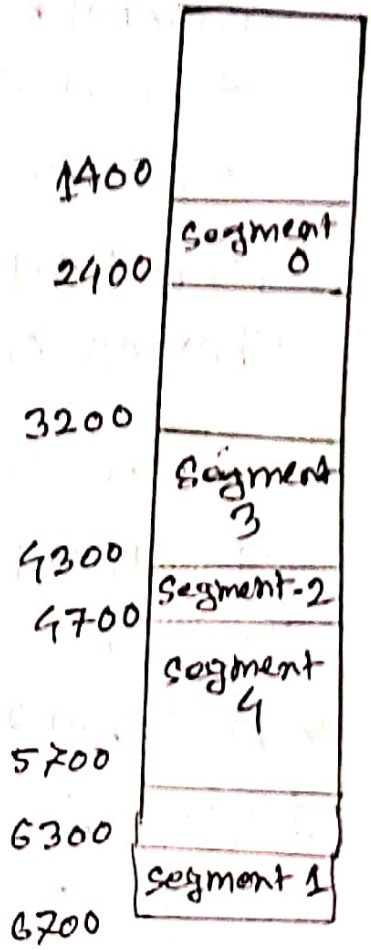
trap: addressing error

Segmentation Hardware

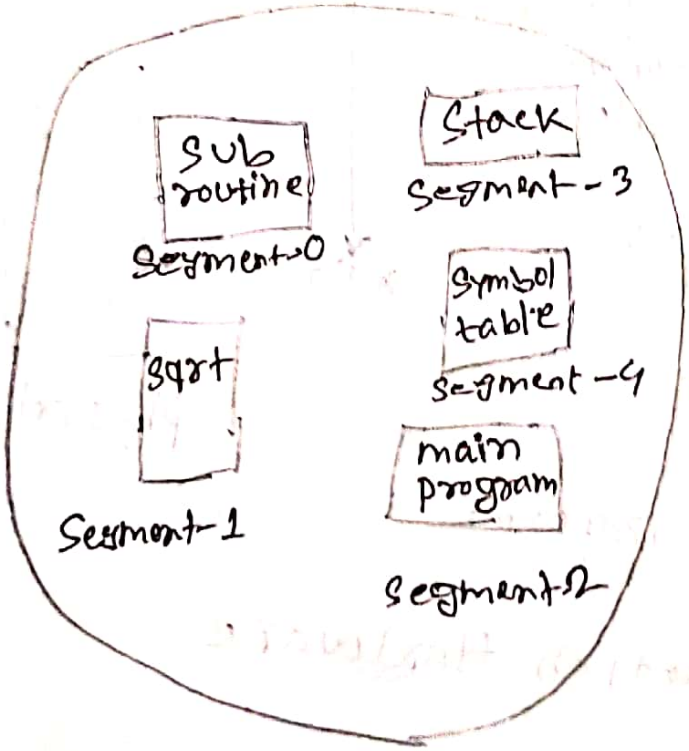
Example:-

	limit	base
0	1000	1400
1	400	6300
2	400	4300
3	1100	3200
4	1000	4200

segment table.



physical memory



Logical address space.

* Ans to the os no: 3.A

3.A

Characterization of deadlock:- Deadlock can arise if four conditions hold simultaneously

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

Hold & 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 $(P_0, P_1, \ldots P_0)$ of waiting process 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 $P_2 \ldots P_{n-1}$ is waiting for a resource that is held by P_n and P_0 is waiting for a reasource that is held by P_0.

X Ans to the Qs no: 3.B X

3.B

Steps of prevent Deadlock:- Restrain the ways request can be made.

* Mutual Exclusion:- not required for Sharable resources; must hold for nonsharable resources.

* Hold and wait:- Must guarantee that whenever a process request a resources, it does not hold any other resources.

(1) Require process to request and be allocated all its resources before it begins execution or allow process to request resources only when the process has none.

(2) Low resouree utilization, Starvation possible.

* circular wait:- impose a total ordering of all resources type and requie that each process requests resources in an increasing order of enumeration.

P.T.O

no Preemption:- (i) If a Process that is holding some resources requests another resource that cannot be immediatly allocated to it. then all resources currently being held are released.

(ii) Preempted resources are added to the list of resources for which the process is waiting.

(iii) Process will be restarted only when it can regain its old resources, as well as the new ones that it is requesting.

≠ Ans to the Q5 no: 3.c ≠

3.c **Safe State:-** Safe State in OS is State in which all resorces of System are well managed. All resources not assigned to one request, There should not be circular demand of resources in system.

when a process request an available resources; system must decide if immediate allocation leaves the system in a safe state.

① System is in safe state there exists a safe sequence of all Processes.

$\langle P_1, P_2 - P_n \rangle$ is safe if for each P_n the resoress that P_1 can still request to satisfied resorees.

* Ans to the Os no: 4.A *

4.A **Dining-philosopher problem:**

Shared data

① Bowl of rice (data set)

② Semaphore chopstick[5] initialized to 1

The Structure of philosopher i.

```
do {
    wait (chopstick [i]);
    wait (chopStick [(i+1) % 5]);
        // eat
    Signal (chopstick [i]);
    Signal (chopstick [i+1) % 5]);
        // think
} while (TRUE);
```

✱ Ans to the Q.S no: 4.B ✱

4.B Bounded-Buffer problems:- ① N buffer, each can hold one item

(II) Semaphore mutex initialized to the value 1

(III) Semaphore full initialized to the value 0

(IV) Semaphore empty initialized to the value N

The Structure of the Produer Process

```
do {
    // produce an item in nextp
    wait (empty);
    wait (mutex);
    // add the item to the buffer
    signal (mutex);
    signal (full);
} while (TRUE);
```

The structure of the consumer process

```
do {  wait (full);
      wait (mutex);
   // remove an item from buffer to nexte

      signal (mutex);
      signal (empty);

   // consume the item in nexte

} while (TRUE);
```

* Ans to the Qs no: 5.A *

5.A

Issues of Multithreading:- ① Thread

Cancellation means terminating a thread before it has finished working. There can be two approaches for this, one is Asynchonous Cancellation, which terminates the target thered immediatly. The other is Defferred Cancellation allows the target thread periodically check if it should be ~~cancelled~~ Cancelled.

Signal Handling:- Signal are used in UNIX systems to notify a process that a particular event has occurred. now in when a multithreaded process receives a signal. to which thread it must be delivered?

it can be delivered to all, or a single therad.

fork System Call :- fork() is a system call executed in the kernel through which a process creates a copy of itself. now the problem multithreaded process is, if one thread forks. will the entire Process be copied or not.)

Security Issues: yes there can be security issues because of extensive sharing of resources between multiple ~~(tasks)~~ threads. there are many other issues that you might face in a Multithreaded Process. but there are appropriate solutions available for them. pointing out some issues. for bothe side of the coin.

* Ans to the Qs no: 5.B *

5.B

Semophore:- In computer Science, Semaphore is a variable or abstract data type used to control access to a common resources by multiple threads and avoid critical section problems in a concurrent system such as a Multitasking Operating system.

Properties of Semophore:- (I) its simple and always have a non-negative integer value.

(II) Works with many process.

(III) Can have many different critical sections with different semophore.

(IV) Each critical section has unique access semaphores.

(V) Can permit mutiple Processes into the critical section at Onee, if desirable.

** Ans to the Qs no: 5.e **

5.e

Worst Fit:- Worst fit is allocates a process to the partition which is largest sufficient among the freely available partitions available in the main memory.

if a large process comes at a later stage, then memory will not have space to accommodate it.

Ans to the Qs no: 4.e

4.e

Starvation:- Starvation is the problem that occurs when high priority processes keep executing and low priority processes get blocked for indefinite time. In heavily loaded computer system,