ABHINAV JHA
2301010211
B. Tech CSE Core - 4

Assignment - 1

1. Modern system still rely on OS because:
→ Abstraction: OS hides hardware complexity (device, CPU, memory) so apps run on many platforms.
→ Resource managment: OS schedules CPU, memory, I/O.

2. Real time embedded OS (RTOS) is used because of predictable timing, low power footprint, small memory / CPU, deterministic interrupt handling.

3. While building a new kernel we should avoid microkernel if raw performance is the only goal. Microkernels give modularity and safety but add IPC overhead because many services run in user space.

4. OS structure doesn't matter as long as process run. Refute because of structure affects :-
→ Performance
→ Reliability / security
→ Maintainability and extensibility
→ Feature support.

5. i). PCB stores saved registers, PC, stack pointer, state bits
- PCB's saved PC points to an unexpected address, the next resume will jump to wrong code.

ii). Save current CPU context into its PCB
- Change process state to waiting and enqueue on wait ~~st~~ queue.
- Update memory mapping
- Scheduler selects next ~~memor~~ process

iii). • Use asynchronous non-blocking I/O when you don't want the task to block and you want responsiveness.
• If task must wait for I/O result before proceeding, use blocking synchronous ~~cet~~ calls.

6. Save state = 2ms
Load state = 3ms
Schedular overhead = 1ms

a) Total context switching time = 2+3+1 = 6ms
b) Impact on multitasking:
→ Every switch costs 6ms of CPU. High context-switch rate reduces effective CPU for user work.
→ Frequent switching increases cache.

7. Single threaded total time = 40s
Threads per process = 2

• Estimate execution time = $\frac{40}{2}$ = 20 seconds

• Multithreading helps allows parallelism on multiple core and hides latency by overlapping I/O with computation.

8.

Process	P_1	P_2	P_3	P_4
Time(ms)	5	3	8	6

a) Gantt chart
• FCFS

P_1	P_2	P_3	P_4

0 5 8 16 22

• SJF

P_2	P_1	P_4	P_3

0 3 8 14 22

- Round Robin (Quantum = 4ms)

P1	P2	P3	P4	P1	P3	P4

0 4 7 11 15 16 20 22

b) Waiting time = Start time - arrival time
TAT - burst

Turnaround time = Finish time - arrival time

FCFS : $P_1 = 5$, $P_2 = 8$, $P_3 = 16$, $P_4 = 22$

TAT = $5 + 8 + 16 + 22 = 51$

$= \frac{51}{4} = 12.75$

Waiting time = TAT - burst

$= 0 + 5 + 8 + 16 = 29.$

Avg. WT = $\frac{29}{4} = 7.25$

SJF $P_2 = 3$, $P_1 = 8$, $P_4 = 14$, $P_3 = 22$

TAT = $8 + 3 + 22 + 14 = 47$

$= \frac{47}{4} = 11.75$

Waiting time = TAT - burst

$$\left.\begin{aligned} &= 8 - 5 = 3 \\ &3 - 3 = 0 \\ &22 - 8 = 14 \\ &14 - 6 = 8 \end{aligned}\right\} \text{Sum} = \cancel{22}\ 25$$

Avg. WT = $\frac{25}{4} = 6.25$

Round Robin $P_1 = 16$, $P_2 = 7$, $P_3 = 20$, $P_4 = 22$

TAT = $16 + 7 + 20 + 22 = 65$

Avg. TAT = $\frac{65}{4}$ = 16.25

waiting time = TAT - burst =

$$\left.\begin{array}{l} 16-5=11 \\ 7-3=4 \\ 20-8=12 \\ 22-6=16 \end{array}\right\} \text{sum} = 43$$

Avg. WT = $\frac{43}{4}$ = 10.75

c) SJF gives lowest average turnaround 11.75 and lowest avg. waiting 6.25 among the three.

9 i)a) Microkernel or modular layered architecture is preffered for scalibility and security.

- Microkernel : small trusted core, most services run in user space.
- Layered modular : It is designed also aids maintainibility and clear separation.

b). Isolation : each virtual machine has separate guest OS and virtual hardware.

- Management : Virtual machine support snapshots, cloning, living migration, templates for rapid deployment.

- Reduce optimisation : Hypervisors multiplex CPU/memory/disk across virtual machine support overcommit, dynamic scaling and scheduler - level QOS.

ii)a) Ensuring high priority tasks are handled without delay.

- Priority-based scheduling :- Assign high priority to intrusion detection tasks.

- Real-time scheduling : Use ~~Rate~~ Real time scheduler for critical tasks so deadlines are met.

- Priority inheritance: Avoid priority inversion when lower-priority tasks hold resources needed by high-priority tasks.

b) • Rate Monotonic Scheduling (RMS): Good for periodic hard-real time tasks with fixed periods.

- Earliest Deadline First (EDF): Optimal for dynamic deadlines and mixed workloads- better CPU utilization if tasks have different deadlines.

- Hybrid approach: Use real-time scheduler (RMS) for safety/critical tasks.

ABHINAV JHA
2301010211
B.Tech CSE Core - 4

Assignment - 2

1. Address translation in modern systems

- Each process generates logical (virtual) address
- MMU (Memory Managment Unit) translates these into physical address
- Translation steps :-

a) CPU generates logical address
b) MMU checks page table for corresponding frame number
c) Concatenates frame no + offset → physical address.

2. Memory layout
→ Eg. layout

Process A (100 Kb of 120 Kb block) | Free 30 Kb | Process B (200 Kb)

- Internal fragmentation = 20 Kb wasted inside A's block
- External fragmentation = 30 Kb free, but too small for 40 Kb request

→ Mitigation techniques:
- Paging (eliminates external, but many cause small internal)
- Segmentation with paging hybrid.
- Buddy system allocation.
- Slab allocators. (in Linux).

3. Paging - based allocation model for a hypothetical OS
- Memory divided into fixed - fram.
- Trade - offs:

* Overhead : Pages tables consume memory.
* Speed :
* Fragmentation

4. OS hardware interaction in virtual memory.
- Page-table in memory.
- MMU translates virtual
- TLB caches recent translations.
- Protection bits

5. 16-bit virtual address, 1KB page size
→ Virtual address = 16 bits = page no + offset
→ Page size = 1 KB = 2^{10} bytes → offset = 10 bits
→ Page no = 16 − 10 = 6 bits
→ * No. of virtual pages = 2^6 = 64
* Page table size = 64 entries × 2 bytes = 128 bytes

6.

Process	Size (KB)
P_1	212
P_2	417
P_3	112
P_4	426

- First-fit

P_1	P_2	P_3	P_4

0 212 629 741 1167

Unused memory = 259 Kb.

- Best-fit

P_1	P_2	P_3	P_4

212 417 112

P_4 still can't fit   Unused = 259 KB

- Worst-fit
  - P_1 (212) into 1000 → 788 left
  - P_2 (417) into 788 → 371 left
  - P_3 (112) into 371 → 259 left
  - P_4 (426) can't fit

Unused = 259 KB.
All three give same unused memory, but Worst-Fit may delay fragmentation buildup.

7. Page replacement reference string:
7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 2, 3

a)b) FIFO : 9 page faults
Optimal : 7 page faults
- LRU : 10 page faults

c) Best: Optimal (minimum). FIFO worse due to Belady's anomaly.

8. Disk write = 10 ms
Memory write = 100 ms
Dirty pages = 30% of 1000 = 300.

a) Overhead = 300 x 10 ms
= 3000 ms = 3 seconds

b) Optimization: Write-back caching with dirty bit tracking or pre-cleaning (background flush) reduces blocking time.

9.a) Working set model + replacement policy
- OS tracks recent active pages per task.
- For object detection: Allocate stable working set
- For infotainment: Allows flexible replacement so it adapts to available memory.

b) Memory allocation strategy.
- Use priority-based dynamic allocation.
- Real-time responsiveness ensured by working set + real time schedule