Gurleen Kaur
2301010207

Assignment - 2
Operating System
BTech CSE Core (Sec4)

Que1. Process of address Translation

1. Logical Address.
- Generated by the CPU when a process access memory.
- each process thinks its has its own Continuous memory.

2. MMU
- Translates logical address. → Physical address
- Uses Page Table that maps process page pages to frame in physical memory.

3. Physical Address.
- Actual location in RAM where data is stored.

Illustration:

$$\begin{tikzcd}
\text{CPU (Process)} \arrow[d, "\text{logical Address}"] \\
\boxed{\text{MMU + Page Table}} \arrow[d, "\text{Translated to Physical}"] \\
\text{Physical Memory (RAM)}
\end{tikzcd}$$

Que2. Memory Layout Example:
- Total Memory = 100MB.
- Allocations.
- Process A = 15MB
- Process B = 25MB
- Process C = 30MB
- Free holes remain as: 10MB + 20MB scattered → none large enough to fit a new

Mitigation Techique
- Paging → divides memory into fixed-size frames to elimate external fragmentation
- Segmentation with Paging → reduces internal waste while keeping logical views.
- Building system Allocation → splits memory in ~~bouts~~ powers of two for faster merging of free blocks.

Que 3. Paging - Based memory Allocation Model
- The OS divides physical memory into fixed-size frames & process into
- A Page table maps virtual pages to physical frames.

Trades - offs:
- Memory overhead: Page tables consume extra space, especially with large address spaces.
- Speed: Address translation adds overhead; mitigated with TLBs for faster lookups.
- Fragmentations: Eliminates external fragmentation, but introduces internal fragmentation if the last page is only partially filled

Que 4. ~~Key hardware Structure~~:
- OS Role - Manages page tables, decides which pages stay in RAM or go to disk and enforces access rights.
- Hardware Role:- Performs fast address translation & Protection checks ~~a~~ using built-in-structure.

~~Keyttte~~ ~~Key~~

Key hardware structures :

1. Memory Management Unit : Converts Virtual addresses to physical address, using page tables maintained by the OS.
2. Translation lookside Buffer : A Cache in the MMU that stores recent virtual to physical mappings for speed.
3. Page Table Base Register : Points to the Process page table in memory.
4. Projects Bits in Page table entites

Que 5. Given
- Virtual address size = 16 bites
- Page size = 1 KB = 2^{10} bytes.
- Page table entry size = 2 bytes.

a) Number of Virtual Pages
1. Total Virtual address space = 2^{16} bytes = 65,536 bytes = 64 KB.
2. Page Size = 2^{10} bytes = 1024 bytes.
3. Number of virtual Pages =

$$\frac{\text{Virtual address space}}{\text{Page size}} = \frac{2^{16}}{2^{10}} = 2^{6} = \underline{64}$$

b) Page table size
1. Each table accquires 1 entry in the page table.
2. Each entry size = 2 bytes.
3. Page table size = 64 x 2 bytes.

Que 6. First fit ~~step by ste~~

Allocate P1 = 212 → fits in first block

Memory : [P1 : 212] [Hole : 788]

2. Allocate P2 = 417 → first hole 788 → allocate
Memory : [P2 : 212] [P2 : 417] [Hole : 371]

3. Allocate P3 = 112 → first hole 371 → allocate
Memory : [P1 : 212] [P2 : 417] [P3 : 112] [Hole : 259]

Best Fit : Arrival order is the same, with a single hole at each step best-fit picks that hole and the result is identical.

After P1 → P1 : 212 Hole : 788
P2 → P1 : 212 Hole : 371
P3 → P1 : 212 Hole : 259

Worst Fit : with one initial block worst-fit also behaves the same here

After P1 → P1 : 212 Hole : 788.
P2 → P1 : 212 Hole : 371
P3 → 212 Hole : 259.

Que7: Page reference string :
7 0 1 2 0 3 0 4 2 3 0 3 2 with 3 frames
Page-fault counts.
FIFO : 10 page faults.
optimal : 7 page faults.
LRU : 9 Page faults.

#Why / ~~[illegible]~~ Belady's anomaly :
- optimal is the theoritical best (it looks ahead) so it gives the minimum faults.
- LRU performs close to optimal (9 faults because it removes the least-recently used page.

Ques. a) Additional time overhead from dirty pages.

Extras Cost per dirty page = disk write − memory write

= 10.0000 ms − 0.0001 ms = 9.9999 ms

· For 300 dirty pages.

300 X 10 ms = 3000 ms.   Ans = 3.0 sec

b) ~~One time optimization~~.

Use a background page− cleaner asynochronous write − back + clean − first replacement. policy: protectiuly flush dirty flush in the background..

Ques 9. a) The OS can apply the working set model to track the active pages needed by each task. For mission − critical tasks, it ensure their working set is always resident in memory, preventing thrashing.

b) A priority based dynamic memory allocation strategy is best allocate gaurantied minimum memory to real − time tasks, and allow flexible ~~too~~ allocation / Sharing. for non critical ones.