OS

Assignment-9

Answer to the question no: 1

*** Logical Processor Vs. Physical Processor**

Each core in a processor is called physical processor. Every core in a processor is capable of doing 2 or more task simultaneously. So, logical processors are the number of physical processor times the number of threads that can run on each physical process. For example-
It a processor have 2 cores and the each core can process two threads, therefore, the ~~physical processor~~ number of physical processor is 2 and the number of logical processor is $2 \times 2 = 4$.

*** Logical Address Vs Physical Address**

Logical address is the address, generated by the CPU at the time of running a program. As it does not exist physically, it is called virtual address. It is used as the reference of the physical address. Whereas, physical address is the physical location of a memory. A user program generates a logical address at the time of running and thinks as if it is running in that address. But for execution, physical address is needed. So the MMU (Memory Management Unit) maps that logical address to the physical address.

* Multi-Process Scheduling Vs Multi-Processor Scheduling:

The way of handling multiple processes running on a ~~single~~ processor is known as multi-process scheduling. There are two types of process scheduling-preemptive scheduling and non-preemptive scheduling. The Algorithms of scheduling processes are First Come First Serve (FCFS), Shortest Job First (SJF), Priority Based (PB), Round-Robin (RR) etc.

The way of handling multiple processor for running differents processes is known as multi-processor scheduling. This is done in two ways. First, all scheduling decision and I/O processing is handled by a single processor, called Master Server and the other processor executes only user code. This techniques is called Asymmetric multi-processing. Another way is, all processors use self-scheduling. Here, All processes may be in a common ready queue or each processor may have its own private queue for ready processes. This is known as Symmetric Multi-processing.

Answer to the question no:2

No, a two-level page table ~~occupy~~ occupies more space than a single-level page table. But a two-level page table works faster than a single-level page table.

Answer to the question no:3

Here, we take the address size for each page is 4B.

* For 32-bit OS having 4KB page-

Here, the address space is $= 2^{32}$ B

Each page is $= 4$ KB $= 4 \times 2^{10}$ B $= 2^{12}$ B

$\therefore$ So, the no of page is $= \frac{2^{32}}{2^{12}} = 2^{20}$

As, each page has address of size 4B, so the size of the page table will be $= 2^{20} \times 4$ B $= 4 \times 2^{10} \times 2^{10}$ B $= 4$ MB

* For 32-bit OS having 8KB page-

Here, the address space is $= 2^{32}$ B

Each page is $= 8$ KB $= 8 \times 2^{10}$ B $= 2^{13}$ B

$\therefore$ So, the no of page is $= \frac{2^{32}}{2^{13}} = 2^{19}$

As, each page has address of size 4B, so the size of the page table will be $= 2^{19} \times 4B = 2^{21} B = 2 \times 2^{10} \times 2^{10}$ B

$= 2MB$

* For 64-bit OS having 4KB page-

Here, the address space is $= 2^{64}$ B

Each page is $= 4KB = 4 \times 2^{10} B = 2^{12} B$

$\therefore$ So, the no of page is $= \dfrac{2^{64}}{2^{12}} = 2^{52}$

As, each page has address of size 4B, so the size of the page table will be $= 2^{52} \times 4B = 2^{54} B$

$= 2^4 \times 2^{10} \times 2^{10} \times 2^{10} \times 2^{10} \times 2^{10}$ B

$= 16$ PB (Petabytes)