Samir Khadka

Operating System

Question 1

Logical addresses or virtual addresses are created during the execution of programs and processes by the CPU and pointed to the memory addresses used by the programs and operating system. These addresses are part of a process's address space; it means that programs can run at a high level of abstraction without knowing physical memory organization. With the help of logical addresses, the operations that contain capabilities such as virtual memory when the operating system gives processes a view of a large piece of continuous address space even if physical memory does not is possible.

However, the physical addresses pertain to the actual areas in the physical memory of the computer, which ranges from RAMs. These addresses are virtually addressed at the memory management unit (MMU) and the memory controller to read and write in the hardware. The address generated in the form of a logical address is converted to physical addressing by means of utilizing page tables and other structures that enable the CPU to access the correct memory locations. This translation process helps in more efficient use and protection of memory as each process runs in a separate address space thereby minimizing memory clashes and increasing system stability.

Question 2:

The page sizes used in systems for memory management are always in powers of 2 mainly due to the fact that this makes computation of areas and alignment on the memories easier. In the binary system, addresses favor breaks that are at powers of 2; and therefore, it is easier to use bitwise operations in order to arrive at the offset with the page. For instance, if a page size is 2^n bytes, bottom n bits correspond to the offset within 2^n bytes and the range of the higher order bits define a page number. This brings about faster and efficient implementation of the translation from the logical to the physical address so that the computational load is alleviated and system response time improved.

Indeed, letting the page sizes be powers of two is convenient to avoid conflicts in the MEM management unit and other related devices. Memory management in MMU also uses simple bit masks and shifts in management of memory translation with the help of linear page sizes. This uniformity follows the management of page tables concerning each entry as a fixed-size block of memory, making the management predictable. When all the types of pages follow the above said standard sizes, both the hardware and the operating systems are able to understand spatial relations in matters of memory mapping, memory access trends and everything that makes up the computing systems, reliability and performance.

Question 3:

a. Logical Address:

Given:
Number of pages: 64
Page size: 1024 words

For page number:
Number of pages = 64
Number of bits to represent 64 pages = $log_2(64) = 10\ bits$
For offset with a Page:
Page size = 1024 words
Number of bits to represent 1024 words = $log_2(1024) = 10\ bits$
Now, the total bits in logical address = 6+10 = 16 bits

b. Physical Address:

Given:
Number of frames = 32
Frame size = 1024 words
For frame number:
Number of frames = 32
Number of bits to represent 32 frames = $log_2(32) = 5\ bits$
For offset within a frame:
Frame size = 1024 words
Number of bits to represent 1024 words = $log_2(1024) = 10\ bits$
Total bits in physical address = 5+10 = 15 bits
Thus, there are 16 bits in the logical address and 15 bits in the physical address.

Question 4:

Making two entries of a page table point to the same page frame in memory results in sharing of memory. This means that there are two ways, or entrances, to access the identical physical memory; any alterations accomplished thru one address are effective in the other as well. This feature can be of great use to reduce the amount of time needed to make copy of large amount of memory. Rather than copying the information, the operating system just makes mappings to the addresses on the destination as with source, so that it involves replacing entries from page table with necessity not copying all the data.

However, such sharing of memory creates problems in maintaining the integrity of the data and hence need to be managed properly. Any change that is made on the common memory through one virtual address will be immediately seen whenever the memory is referred by the other address. This can cause problems in that if one or many processes/threads write to the shared memory at the same time then race conditions and data inconsistencies can occur. To enable such issues, however, there is a need to coordinate the access to shared memory regions using entities such as locks or semaphores.

Question 5:

To fit processes of different sizes into different memory partitions using the first-fit, best-fit and worst-fit algorithms; some guidelines have to be followed.

- **First-Fit:**
  - 115 KB -> 300 KB (185 KB left)
  - 500 KB -> 600 KB (100 KB left)
  - 358 KB -> 750 KB (392 KB left)
  - 200 KB -> 200 KB (0 KB left)
  - 375 KB -> 350 KB (0 KB left)
- **Best-Fit:**
  - 115 KB -> 125 KB (10 KB left)
  - 500 KB -> 600 KB (100 KB left)
  - 358 KB -> 750 KB (392 KB left)
  - 200 KB -> 200 KB (0 KB left)
  - 375 KB -> 350 KB (0 KB left)
- **Worst-Fit:**
  - 115 KB -> 750 KB (635 KB left)
  - 500 KB -> 600 KB (100 KB left)
  - 358 KB -> 635 KB (277 KB left)
  - 200 KB -> 350 KB (150 KB left)
  - 375 KB -> 300 KB (leftover memory not enough for process)