Memory management and virtual memory

An operating system allows a user to be working on several tasks at the same time. It appears that the applications run simultaneously. Each program, open file or copied clipboard item, for example, is allocated a specific area in the memory simultaneously. The allocation of space is controlled by the operating system.

Paging and segmentation

Paging and segmentation are two techniques employed to make optimum use of the memory by splitting it into small sections.

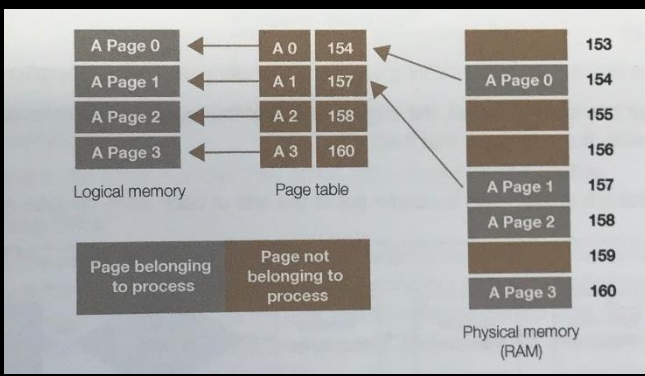
Paging

In computer operating systems, paging is a memory management scheme by which a computer stores and retrieves data from secondary storage for use in main memory. In this scheme, the operating system retrieves data from secondary storage in same-size blocks called pages.

The memory is divided up into fixed sized pages of 4Kb each. A process currently running in memory may be held in several non-contiguous pages.

Imagine a 15Kb program stored in the memory. It may be physically stored in four separate pages of the physical memory (although ordered consecutively with logical addresses).

A page table links physical memory addresses with logical address space of each process.



Memory is divided into physical units and processes may be held in non-contiguous pages.
Each program will occupy a different number of pages.
One problem with paging is that it does not consider the logic of the code being separated. It may well separate code within a loop – decreasing efficiency. The alternative strategy is segmentation that keeps logical divisions of a program together.

Segmentation

Segmentation is the logical division of address space into varying length segments which depend on the program structure.

As with paging, it is also possible to load any part of the program into the memory initially.

Segmentation allows for blocks of memory with varying sizes to be used. Segments are commonly used to store library software or other blocks of data that are shared across multiple devices.

Memory is logically divided into logical units (for groups of a process)

When the memory is full, the OS must rely on virtual memory, transferring segments or pages of a program that are not currently in use to a dedicated space on a secondary storage device. This would free up space for new pages or segments in the RAM.

Pages/segments can't be acted upon directly from the virtual memory. Instructions can't be transferred directly from a disk to the CPU.

Virtual memory

Memory has a limited capacity. As more tasks are loaded, the operating system may swap pages of temporary inactive jobs out of the disk, thus using secondary storage as an extension of memory so new processes have memory space.

When the memory is full and there is an insufficient memory, there is a slight deterioration in performance.

Pages are swapped in and out of the ram to the secondary storage across the slow system bus. When the operating system is spending most of its time swapping pages in and out, performance decreases. This is called disk thrashing.

When the computer has run out of space in the main memory, it must rely on the secondary storage as virtual memory.

Summary

- Both paging and segmentation split the memory up into smaller sections so it can be managed more easily by the OS.
- Both paging and segmentation allow programs to run despite insufficient memory, owing to the resource offered by virtual memory.
- Both pages and segments are stored on the disk (virtual memory) or on memory.
- Both pages and segments are transferred into memory when needed.

- Pages are fixed sizes and are made to fit sections of memory.
- Pages are physical divisions

- Segments are variable in size and non-contiguous .
- Segments are complete sections of programs
- Segments are logical divisions.

If physical memory is low then virtual memory has to be used. This may mean that more time is spent moving pages and segments between RAM and the secondary storage than is actually spent processing the tasks. This is known as **disk thrashing.**