Unit-II
Lecture: 6
(Process Management)
(Part-III)

Process Address Space:

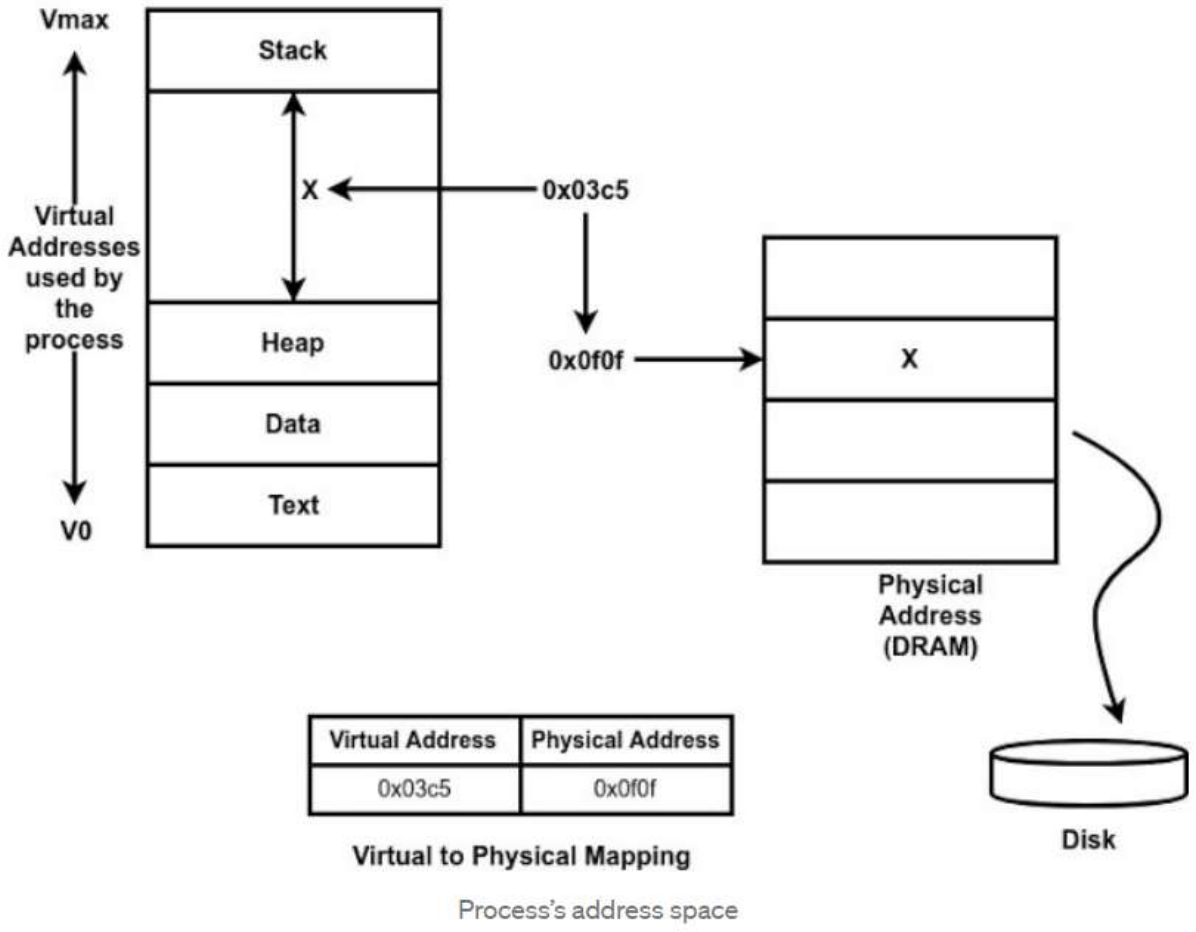
Every single element of the process has to be uniquely indentified by its address, so an OS abstraction is used to encapsulate all of the process data in an address space. The address space is defined by a range of address from V0 to some Vmax, and different types of process state will appear in different part of this address space.

The address space from V0 to Vmax does not correspond to actual address space in a physical location, instead they are virtual addresses.

The memory management hardware and operating system components responsible for memory management like page tables maps these virtual addresses to actual physical addresses.

Text and data, stack and heap are the types of state of the process.
From the last lecture (Lecture-5), we know that process stack contains temporary data(such as function parameters, return addresses and local variables) and is a dynamic part of the process state which grows and shrinks during execution in Last-in-First-out order (that's why arrows in the diagram).

- Suppose we are executing a function "A" and want to call another function "B" from this function, for this, we have to save our current state (i.e. state of function "A") to the stack and jump to execute function "B", after the execution of function "B", the state of the previous function ("A") from which the function "B" was called is restored from the top of the stack and function "A" can continue its execution form that very instruction it had left (Program counter was also saved).



All process may not need all the address space, as there can be many processes running at a time and we may not have enough space in the physical memory (i.e. in the RAM).

- In order to deal this situation, operating system dynamically decides which portion of address space will be present in physical memory.
- Multiple processes may share physical memory by temporarily swapping some portion of their address space into the disk, this portion of address space will be brought back into the memory whenever needed.
- The page table maintains the mapping from virtual address to the physical address (RAM or Disk), it also validates whether a particular memory access request by a process is allowed to perform or not.

Before the execution of a process, its source code must be compiled, the compilation of the code results in the conversion of the high-level code to binary instructions, a register in CPU is maintained which indicates the address of the next instruction to be executed for this process, we call it **Program Counter (PC).** There are some other registers in the CPU which stores

other information of the process like addresses for data or some status information. There is also a stack pointer which points to the top of the stack used by the process.

To maintain all of this process for every single process an operating system maintains a Process Control Block (PCB).

Process Control Block (PCB):

A Process Control Block is the data structure that operating system maintains for every single process. The PCB is created when the process is created and it is also initialized at that time and certain fields are updated when the process states changes.

The PCB contains process states like:

- program counter i.e. address of the next instruction to be executed.
- stack pointer, which is required to be saved when the process is switched from one state to another to retain the current position of the process.
- all the value of the CPU register
- various memory mapping form virtual to physical memory,
- a list of open files
- information which is necessary for scheduling of the process like how much time this particular process has executed on CPU in the past and
- how much time it should be allocated to execute in the future.

Process Control Block

A PCB is created at the very moment a process is created with some initializations like PC points to the first instruction that needs to be executed.