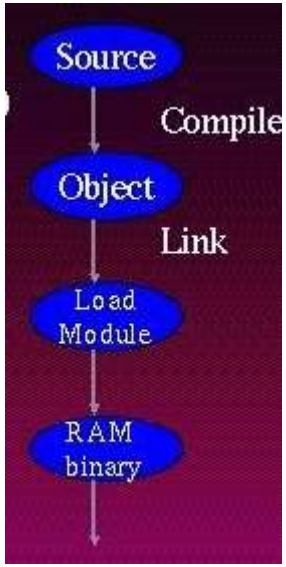
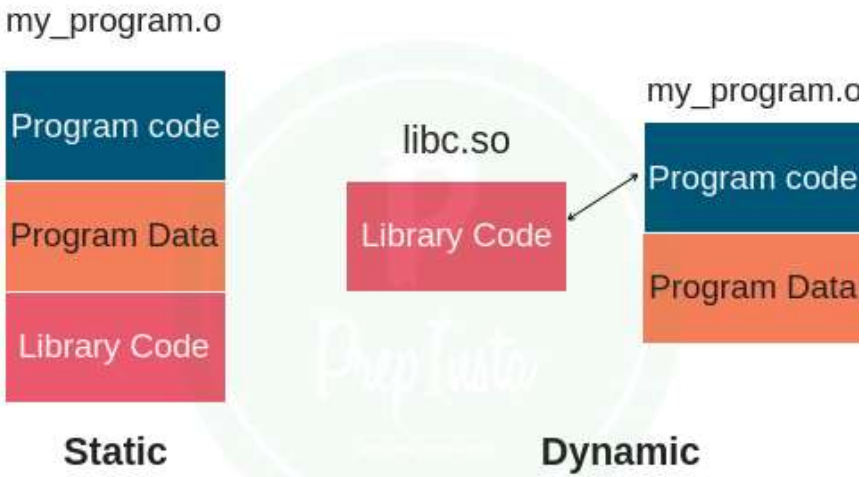
SNS COLLEGE OF TECHNOLOGY

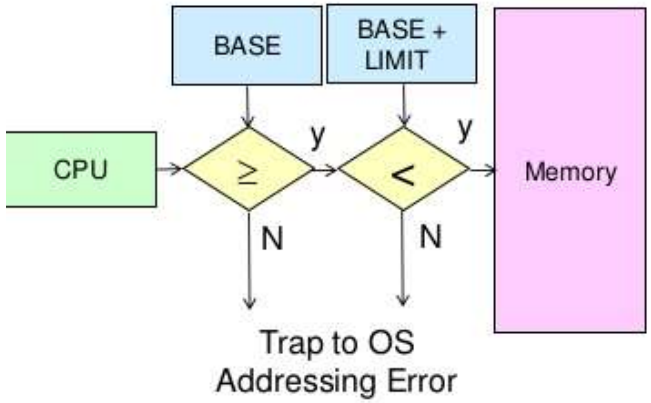
(Autonomous)

COIMBATORE-35

Memory Management Background

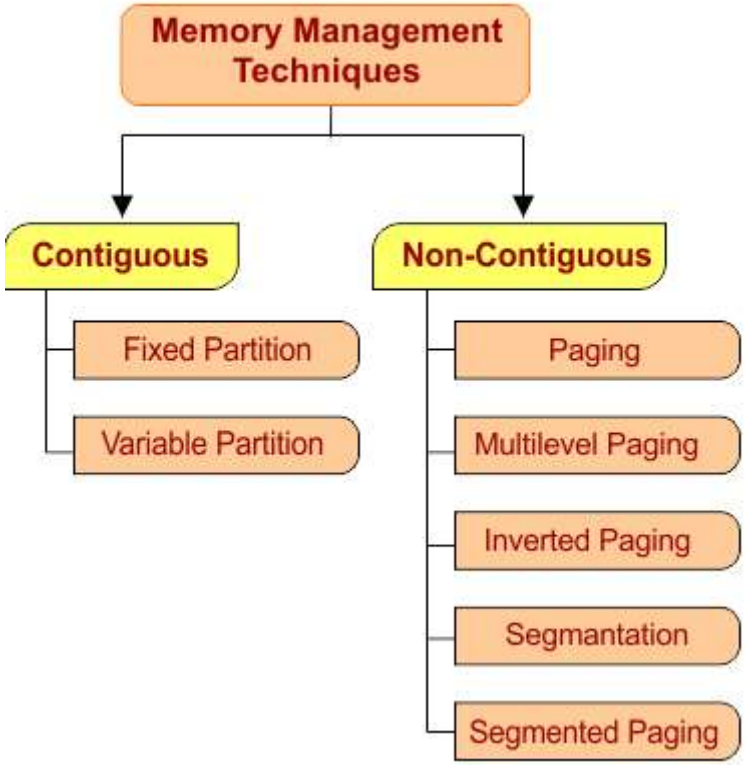






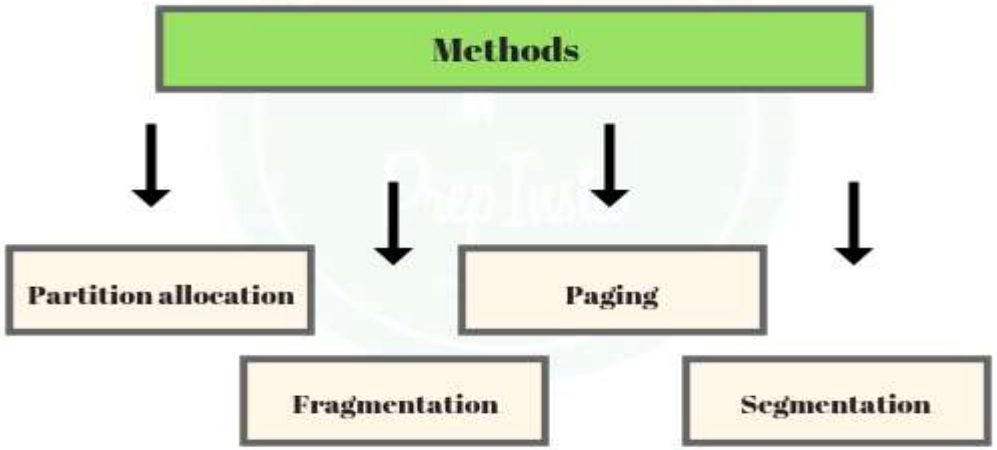
Memory Management Background

- Allocation
- Deallocation
- Memory Mapping
- Paging
- Swapping
- Memory Protection
- Memory Fragmentation
- Virtual Memory



Memory Management-Basic Concepts

- To rum a program, it must be brought into memory.
- Input queue – collection of processes on the disk that are waiting to be brought into memory to run the program.
- User programs go through several steps before being run

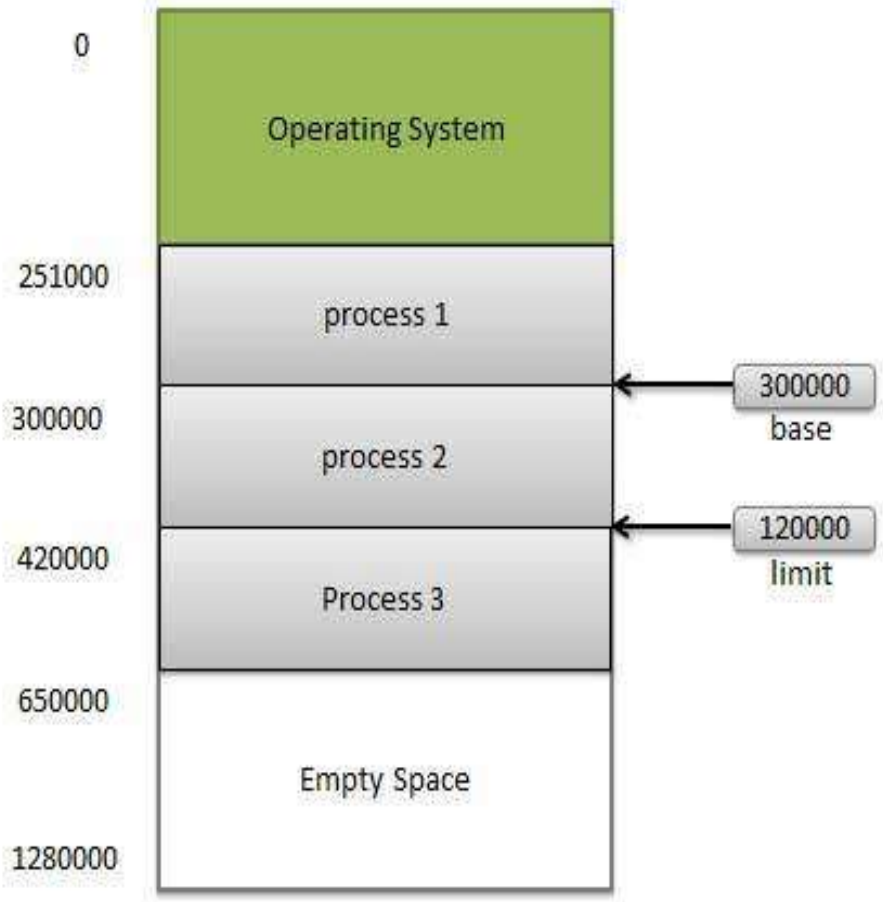


Background -Base and Limit Registers

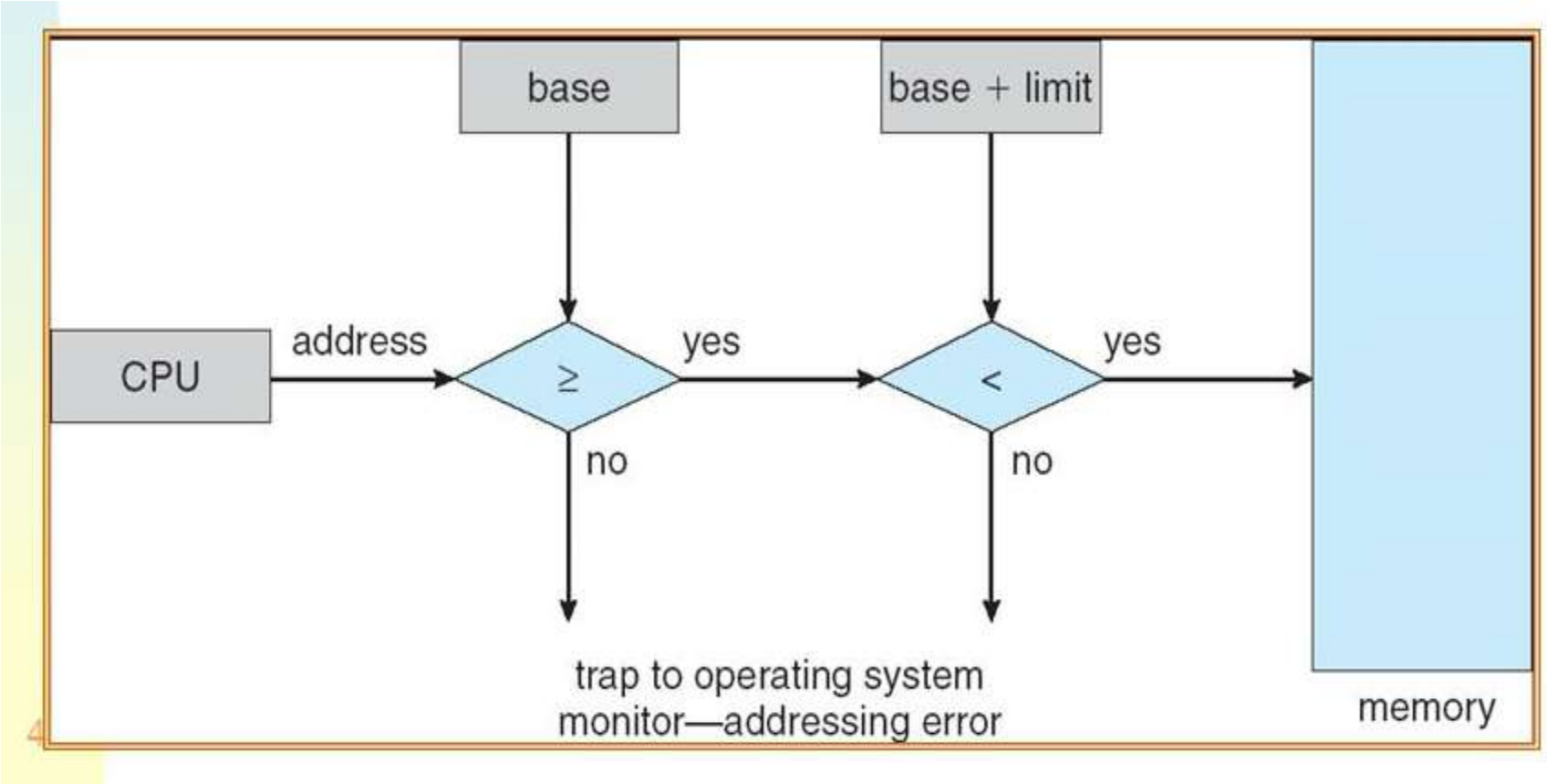
- Each process has a separate memory space.
- Protection can be achieved by using a **base register** and a **limit register**.
- Only the operating system (in kernel mode) can set the base and limit registers.
- If the CPU requests an absolute memory location *M*, the hardware verifies that

  $$\text{base} \leq M < \text{base} + \text{limit}.$$

  If this is not the case, the hardware generates a trap to the operating system.



Hardware Address Protection with Base and Limit Registers



Address Binding

Address binding: Mapping of instructions and data from one address to another address in memory.

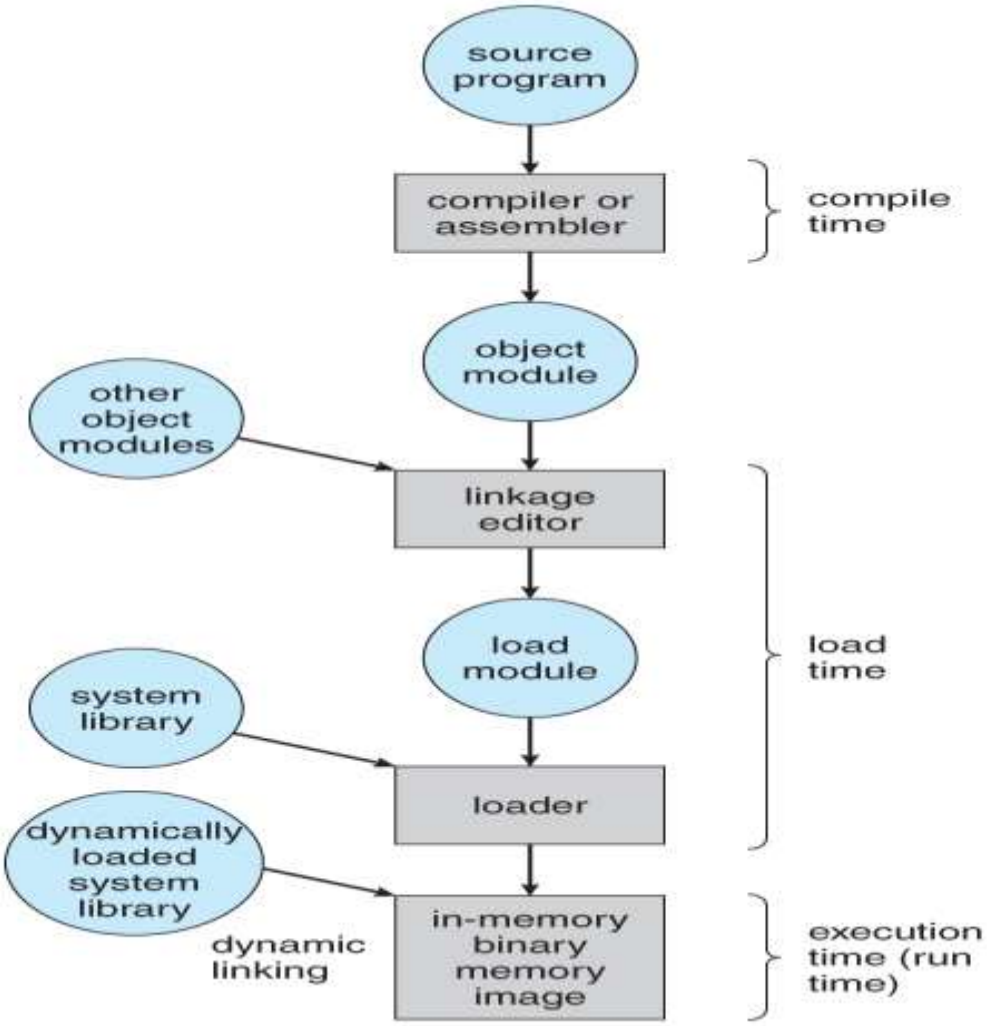
Three different stages of binding:

1 **Compile time:** Must generate absolute code if memory location is known in prior.

2 **Load time:** Must generate relocatable code if memory location is not known at compile time

3 **Execution time:** Binding delayed until run time if the process can be moved during its execution from one memory segment to another. Need hardware support for address maps (e.g., base and limit registers).

Address Binding



Logical Vs Physical Address Space

The concept of a logical *address space* that is bound to a separate *physical address space* is central to proper memory management.

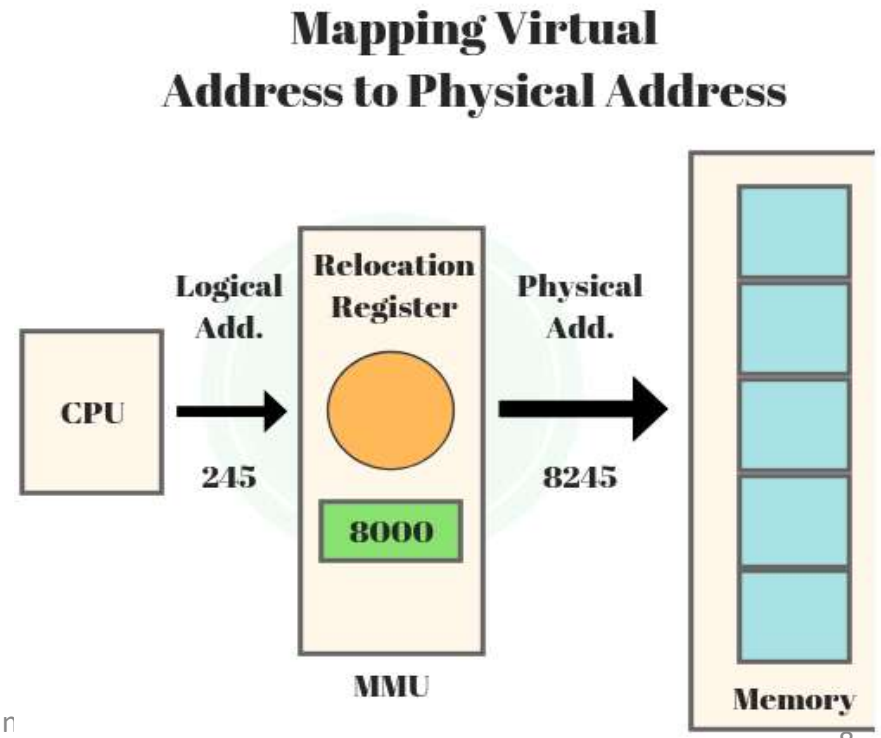
Logical address – generated by the CPU; also referred to as **virtual address**

Physical address – address seen by the memory unit

Logical and physical addresses are the same in **compile-time** and **load-time address-binding schemes**;
logical (virtual) and physical addresses differ in **execution-time address-binding scheme**

Logical address space is the set of all logical addresses generated by a program

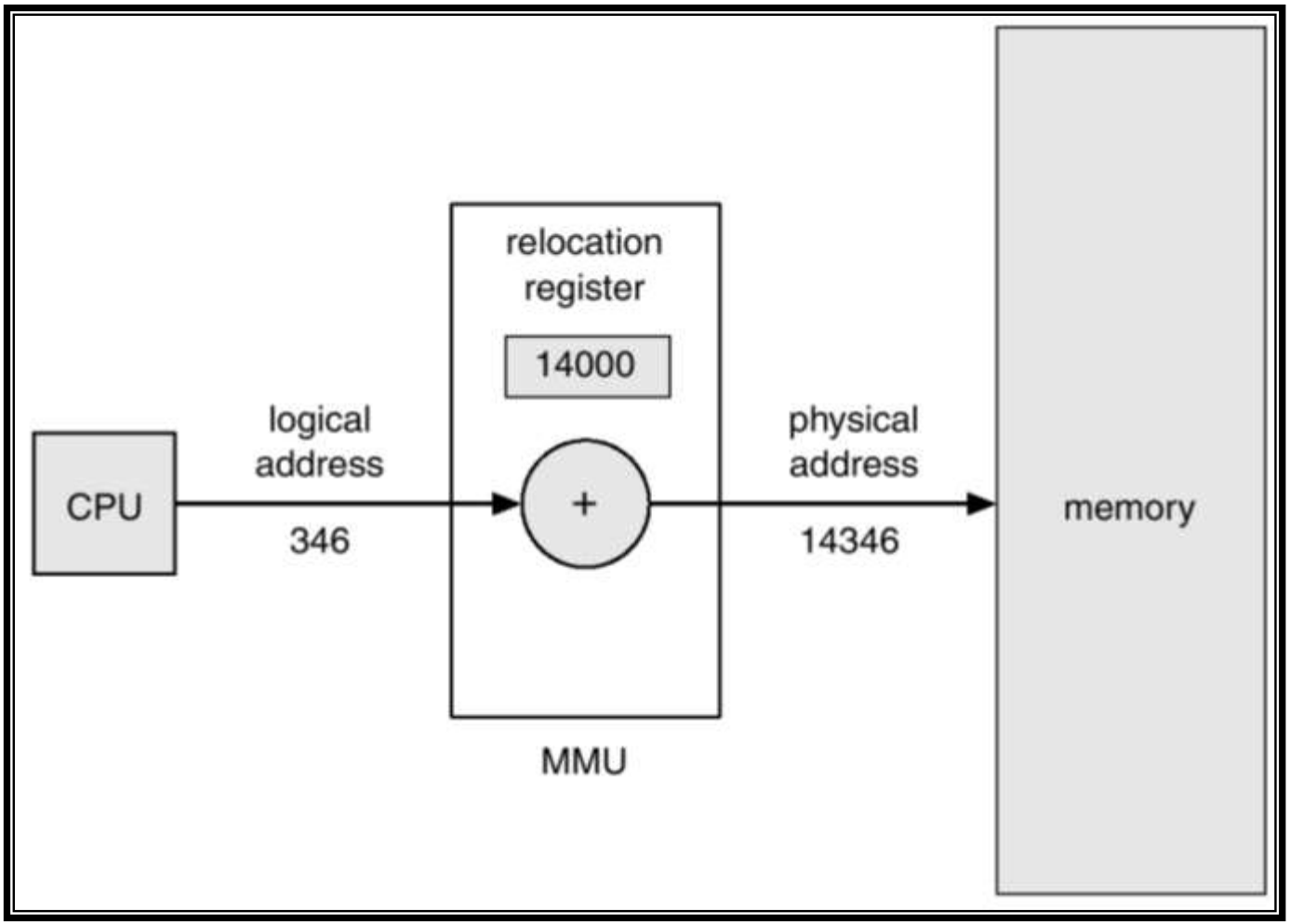
Physical address space is the set of all physical addresses generated by a program



Memory-Management Unit (MMU)

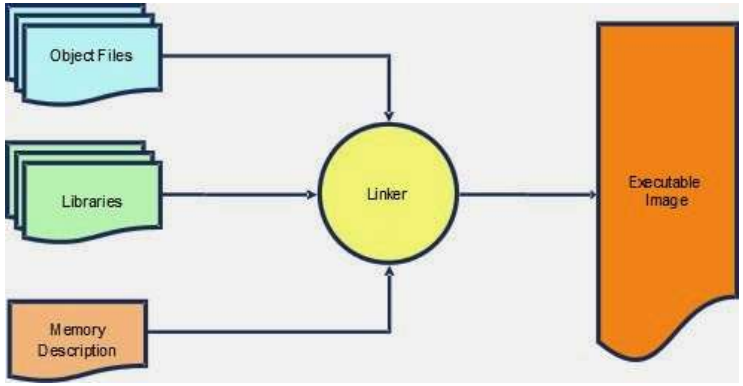
- Hardware device that maps virtual to physical address.
- In MMU scheme, the value in the relocation register is added to every address generated by a user process at the time it is sent to memory.
- The user program deals with *logical* addresses; it never sees the *real* physical addresses.

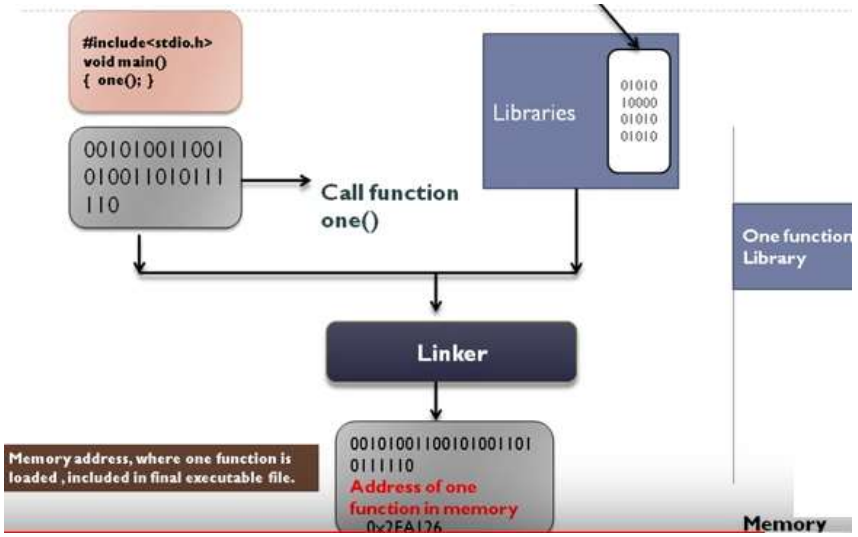
Dynamic relocation using a relocation register



Dynamic Linking

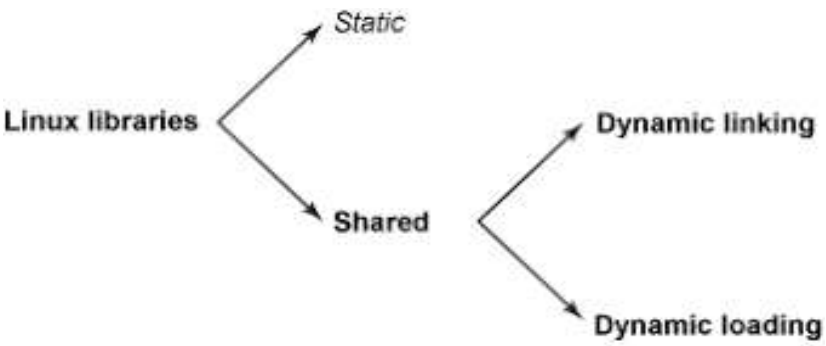
- Linking postponed until execution time.
- Process of linking external shared libraries into the program and then bind those shared libraries dynamically to the program
- Operating system needed to check if routine is in processes' memory address.

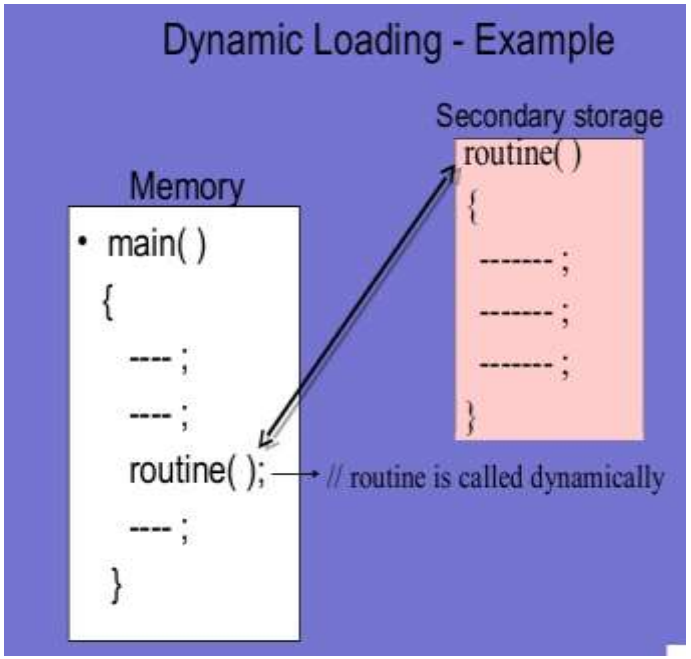




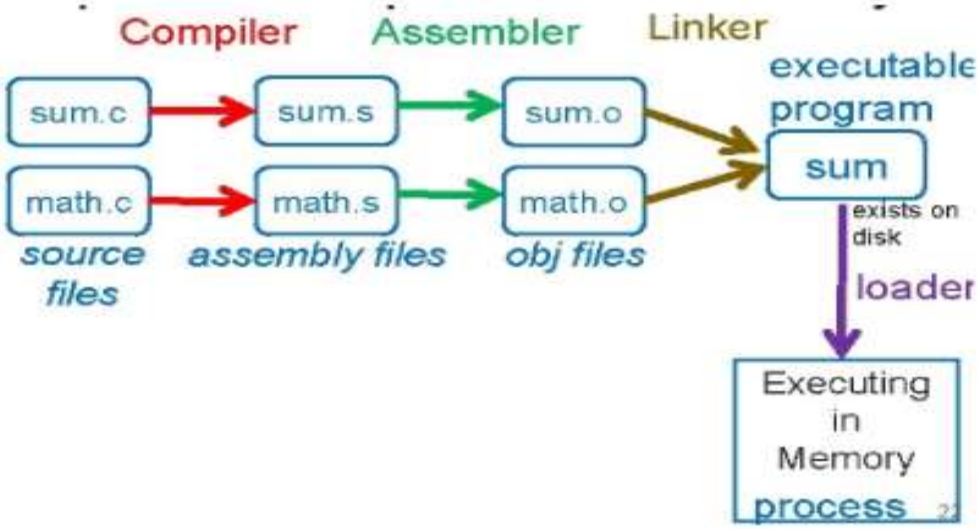
Dynamic Loading

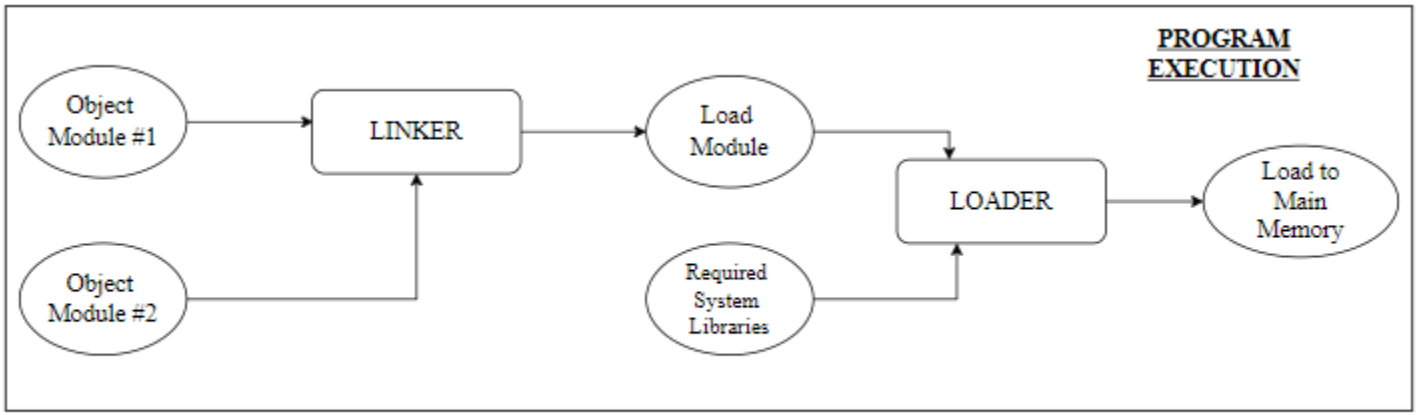
- Routine is not loaded until it is called
- Better memory-space utilization; unused routine is never loaded.
- Useful when large amounts of code are needed to handle infrequently occurring cases.
- No special support from the operating system is required implemented through program design.





Dynamic Linking &Loading

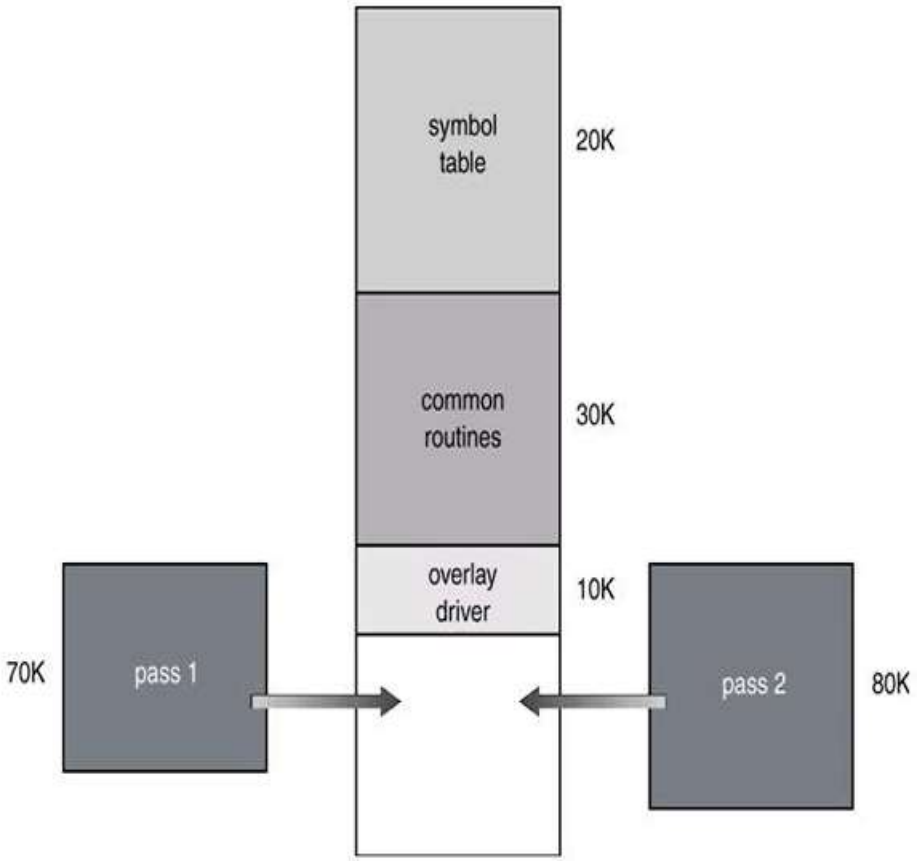




Working of loading and linking

Over Lays

- Keep in memory only those instructions and data that are needed at any given time.
- Needed when process is larger than amount of memory allocated to it.
- Implemented by user, no special support needed from operating system, programming design of overlay structure is complex

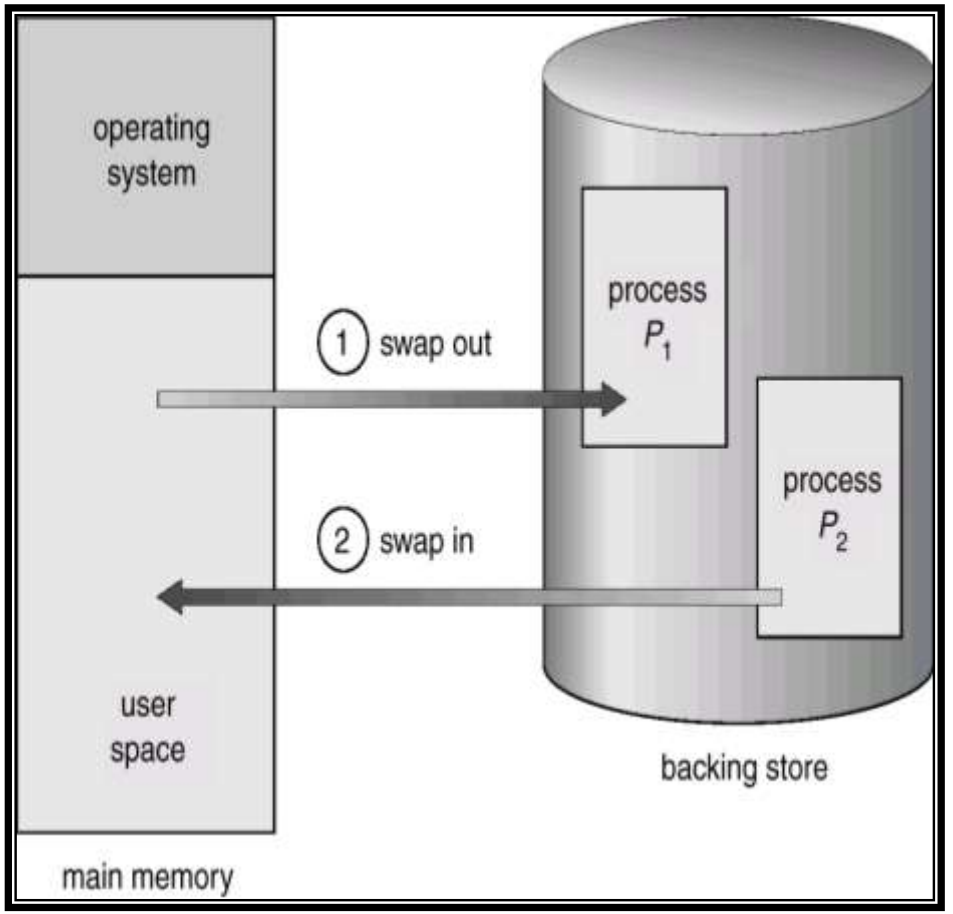


Swapping

Swapping is a mechanism in which a process can be swapped temporarily out of memory to a backing store (Swap Out) and then brought back into memory for continued execution. (Swap In).

Benefits:

- Allows higher degree multiprogramming
- Allows dynamic relocation
- Better memory utilization
- Less wastage of CPU time on compaction
- Can easily be applied on priority-based scheduling algorithms to improve performance



References

1. Silberschatz, Galvin, and Gagne, “Operating System Concepts”, Ninth Edition, Wiley India Pvt Ltd, 2009.
2 . Andrew S. Tanenbaum, “Modern Operating Systems”, Fourth Edition, Pearson Education, 2010.

Summarization