Unit: III
Lecture: 2
(Memory Management)
The Address Spaces

Relocation:

Operating systems can manage where each process is stored in memory using a technique called relocation. The operating system is often stored in the highest memory addresses.

When the program compiles and executes, it starts with zero (0) physical address. The maximum address is equal to the total memory size minus the operating system size. The process is loaded and allocated a contiguous segment of memory. The first (smallest) physical address of the process is the base address and the largest physical address the process can access is the limit address.

There are two methods of relocation:

- **Static Relocation:**
    - the operating system adjusts the memory address of a process to reflect its starting position in memory.
    - once a process is assigned a starting position in memory, it executes within the space it has been allocated.
    - Once the static relocation process has completed, the operating system can no longer relocate the process until it terminates.
    - used by the IBM 360 as a stop-gap solution.
    - It is a slow and complicated process.
- **Dynamic Relocation:**
    - In this the hardware adds relocation register (base value) to the virtual address generated by the compiler.
    - The relocation register allows translation to a physical memory address.

Two problems have to be solved to allow multiple applications to be in memory at the same time without interfering with each other:

- **protection**
- **relocation**

Address Space → a new abstraction for a memory.

Address Space:

An address space is the set of addresses that a process can use to address memory.

- Each process has its own address space, independent of those belonging to other processes.

Giving Processes address space:

- **Using Base and Limit Registers (a simple version of dynamic relocation):**

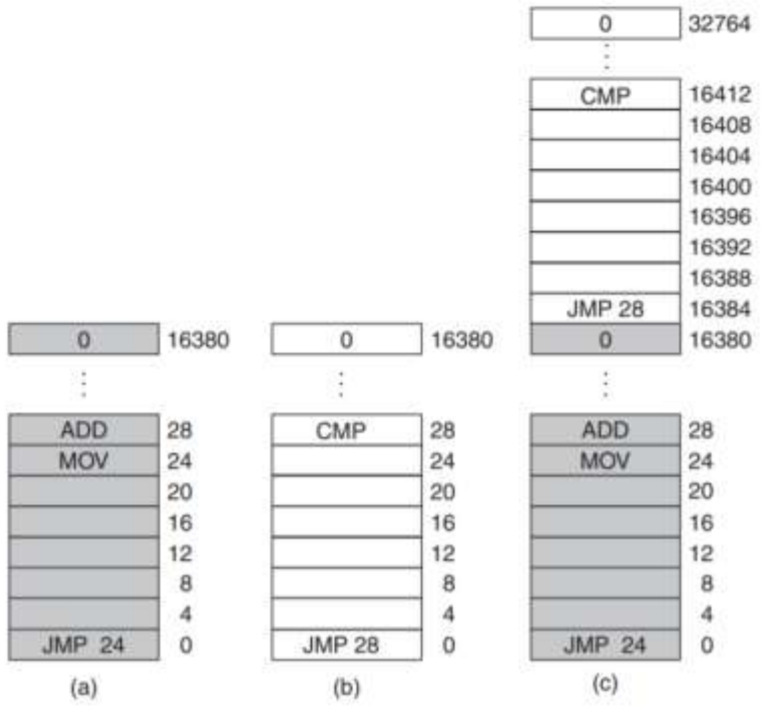
Using base and limit registers each process can have its own private address space:

Each CPU is equipped with two special hardware registers usually called base and limit registers.

When these registers are used, programs are loaded into consecutive memory locations whenever there is room and without relocation during loading.

- When a process is run, the base register is loaded with the physical address where its program begins in memory and the limit register is loaded with the length of the program.

Consider the example of IBM 360 from Lecture: 1 (Fig c)



In fig (c), the base and limit values that would be loaded into these hardware registers when the first program is run:

Base Register= 0

Limit Register = 16384

The values used when the second program is run are:

Base Register= 16384

Limit Register = 32768

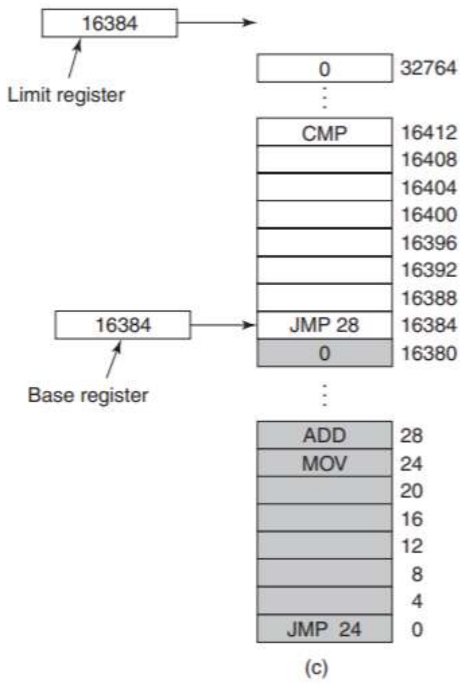


Fig: Base and limit registers used to give each process a separate address space

Every time a process references memory, either to fetch an instruction or read or write a data word, the CPU hardware automatically adds the base value to the address generated by the process before sending the address out on the memory bus.

Simultaneously, it checks whether the address offered is equal to or greater than the value in the limit register; in which case a fault is generated and the access is aborted.

So, in the example of Fig (c), the second program executes a:

JMP 28

instruction, but the hardware treats it as though it were JMP 16412

➔ it lands on the CMP instruction as expected.

In many implementations, the base and limit registers are protected in such a way that only the operating system can modify them.

This scheme allows the operating system to change the value of the registers but prevents user programs from changing the registers contents.

A disadvantage of relocation using base and limit registers is the need to perform an addition and a comparison on every memory reference. Comparisons can be done fast, but additions are slow due to carry-propagation time unless special addition circuits are used.