Chapter # 5 (Memory Management)

Memory Management:

Memory Management is the functionality of OS which handles or manage primary memory by moving the process back and forth between main memory and disk during execution

Function:

1. Keep track of all memory location. Weather is memory is allocated or free.
2. Keep track of how many memory is allocated
3. Take decision of which process get how many memory and from which section of memory
4. When the process will get the memory

Memory Management Requirement:

1. Relocation
2. Protection
3. Sharing
4. Logical Organization
5. Physical Organization

1. **Relocation:**
    a. Programmer does not know where the program will be placed in memory when it is executed.
    b. While the program is executing, it may be swapped to disk and returned to main memory at a different location (relocated).
2. **Protection**:
    a. Processes should not be able to reference memory locations in another process without permission
    b. Impossible to check absolute addresses at compile time
    c. Must be checked at run time
    d. Memory protection requirement must be satisfied by the processor (hardware) rather than the operating system (software)

3. Sharing:
    a. Allow several processes to access the same portion of memory
    b. Better to allow each process access to the same copy of the program rather than have their own separate copy
4. Logical Organization:
    a. Programs are written in modules
    b. Modules can be written and compiled independently
    c. Different degrees of protection given to modules (read-only, execute-only)
    d. Share modules among processes

5. Physical Organization:
    a. Memory available for a program plus its data may be insufficient
    b. Programmer does not know how much space will be available

Why Memory Management is required?

- Allocate and de-allocate memory before and after process execution.
- To keep track of used memory space by processes.
- To minimize fragmentation issues.
- To proper utilization of main memory.
- To maintain data integrity while executing of process.

Address Spaces:

Logical Address Space:

The set of all logical addresses generated by a program is referred to as a **logical address space**

Physical Address Space:

The set of all physical addresses corresponding to these logical addresses is referred to as a **physical address space**

Memory Allocation:

- It's of two type

1. Contiguous Memory Allocation
2. Non-Contiguous Memory Allocation

Syed Bakhtawar Fahim
Bakhtawarfahim10@gmail.com

Contiguous Memory Allocation

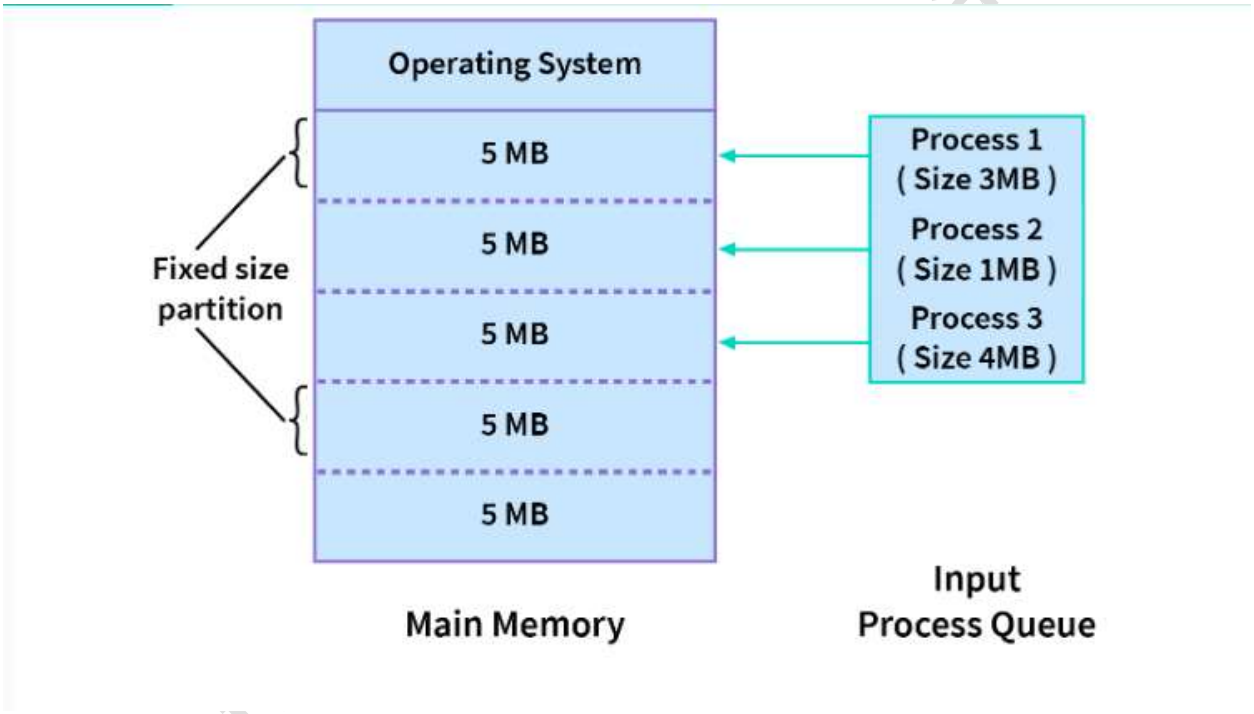
Contiguous Memory Allocation is a type of memory allocation technique where processes are allotted a continuous block of space in memory.

This allocation can be done in two ways:

1. Fixed-size Partition Scheme
2. Variable-size Partition Scheme

Fixed-size Partition Scheme

In this type of contiguous memory allocation technique, each process is allotted a fixed size continuous block in the main memory. Also called "Static Partitioning"



Advantages

1. Because all of the blocks are the same size, this scheme is simple to implement.
2. As at a time multiple processes can be kept in the memory, this scheme can be implemented in a system that needs multiprogramming.
3. It is easy to keep track of how many blocks of memory are left, from which we know that how many process can get the memory spaces

Syed Bakhtawar Fahim
Bakhtawarfahim10@gmail.com

Disadvantages

1. As the size of the blocks is fixed, we will not be able to allot space to a process that has a greater size than the block.
2. The size of the blocks decides the degree of multiprogramming

Note: The number of processes that can stay in the memory at once is called the degree of multiprogramming. Hence, the degree of multiprogramming of the system is decided by the number of blocks created in the memory.

Variable Size Partitioning:

Each process is allotted space depending upon its requirements. There is no defined fixed-size block. As the blocks are variable-sized, which is decided as processes arrive, this scheme is also called Dynamic Partitioning.

Advantages

1. There is no internal fragmentation. Hence, there is no memory wastage in this scheme.
2. As there are no blocks that are of fixed size, even a process of big size can be allotted space

Disadvantages

1. Because this approach is dynamic, a variable-size partition scheme is difficult to implement.
2. External Fragmentation

Non-Contiguous Memory Allocation

Paging

Paging is a memory management technique in which the operating system fetches the processes from the secondary storage into the main memory in the form of fixed size memory blocks. These fixed size blocks are called pages. Paging is a logical concept and it is used for quick access to data.

In this technique, the main memory is split into small blocks of physical address (memory). These small blocks are called frames. In paging, the frame size is fixed. The frame size must be equal to the page size in order to utilize the main memory completely and to avoid external fragmentation.

Syed Bakhtawar Fahim
Bakhtawarfahim10@gmail.com

Advantages and Disadvantages of Paging in OS

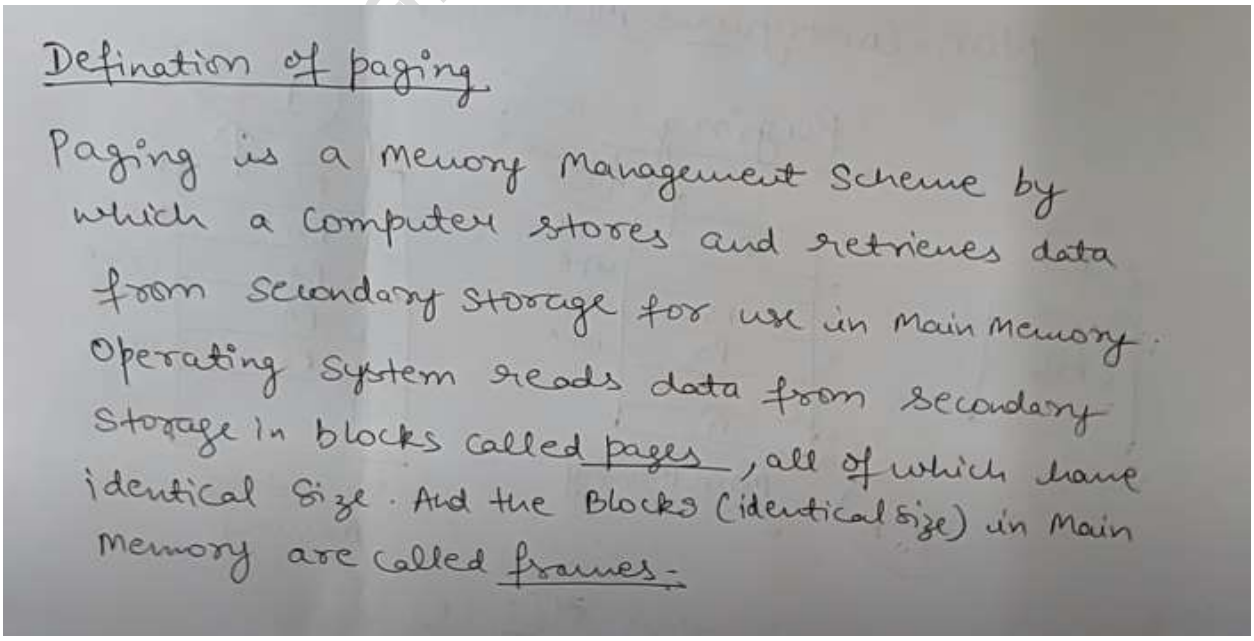
Paging is based on breaking physical memory into fixed-sized blocks called frames and logical memory in to fixed-sized blocks called pages. Let's discuss some advantages and disadvantages associated with Paging.

Advantages of Paging in OS

1. Paging allows the data to be stored in a non-contiguous manner.
2. Paging helps to solve the external fragmentation issue
3. Swapping of pages is easy due to the same size.
4. Paging is also a very simple algorithm for memory management.
5. With the help of TLB cache, Paging is even faster.

Disadvantages of Paging in OS

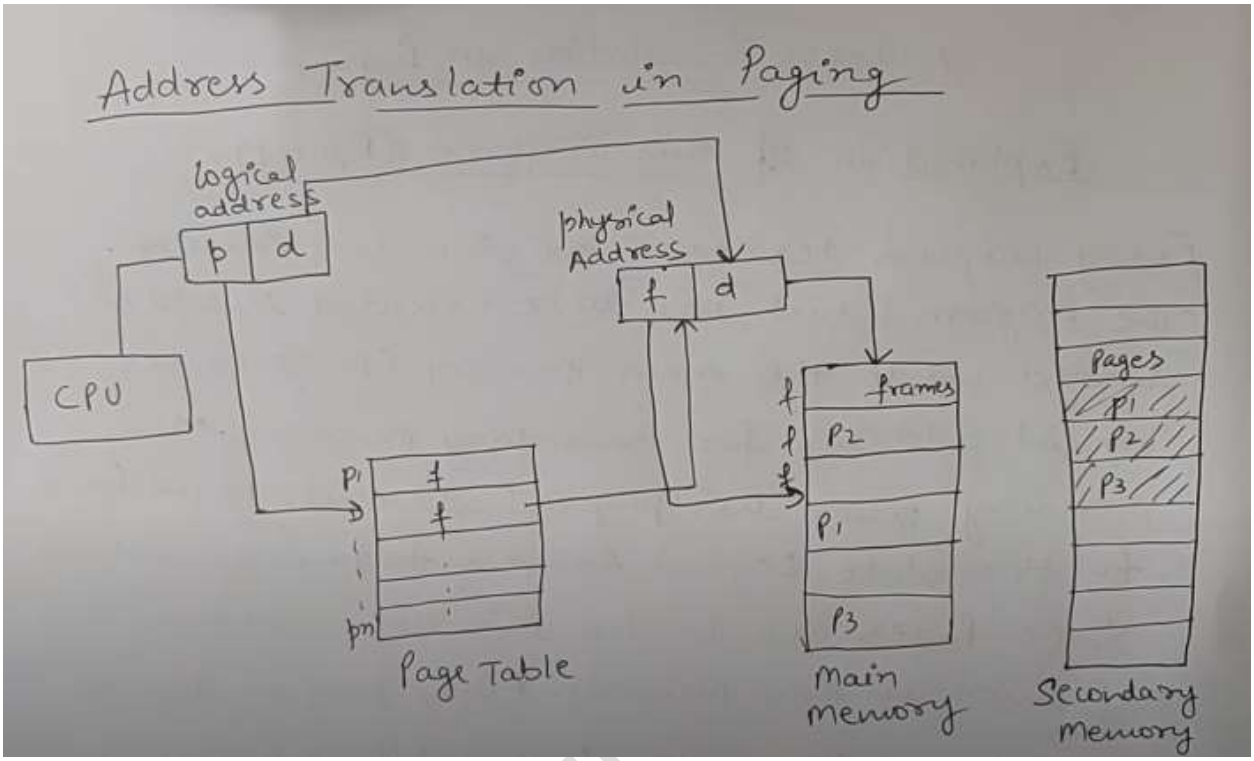
1. Paging requires a large memory space as the page table is also stored in the main memory.
2. There is still the issue of internal fragmentation because the page has a fixed size, but a process may request more or less space.
3. Without the use of TLB or in case of TLB miss, the page access time (time to translate address) is large.



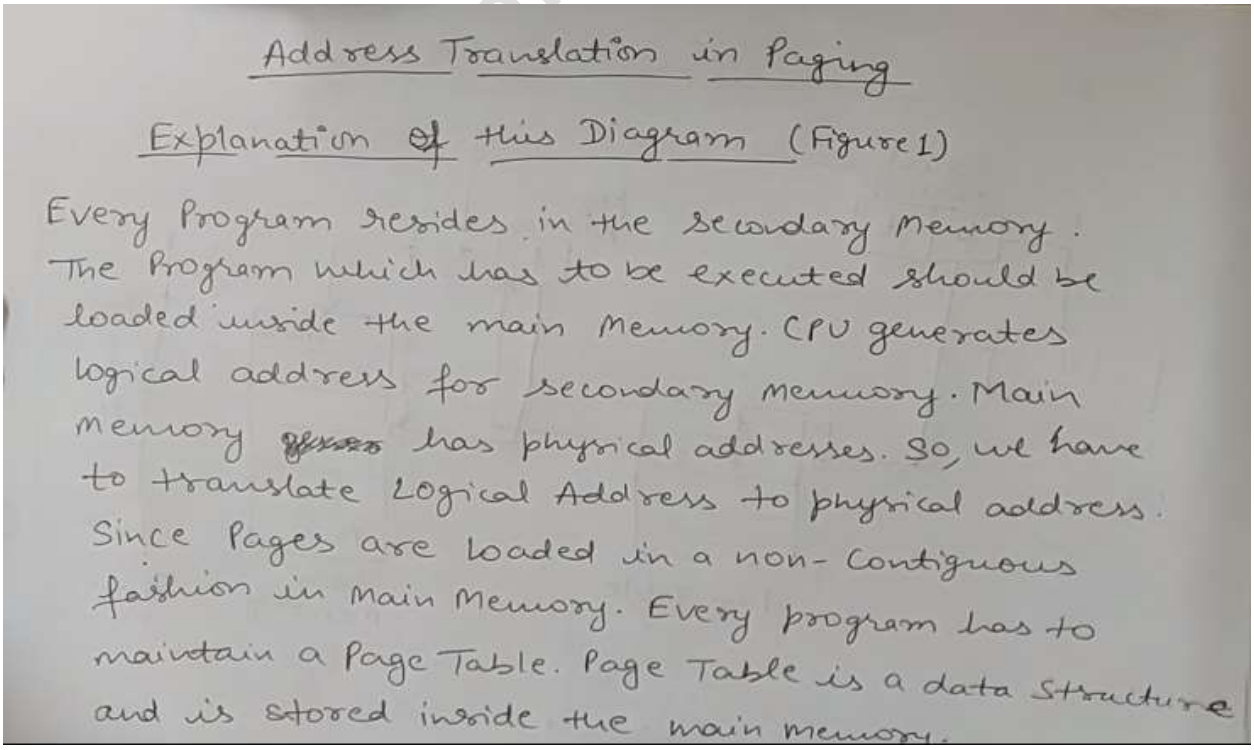
Syed Bakhtawar Fahim
Bakhtawarfahim10@gmail.com

Address Translation in Paging

Figure



Explanation



Address Translation in Paging

Explanation of this Diagram (Figure 1)

Every Program resides in the secondary Memory. The Program which has to be executed should be loaded inside the main Memory. CPU generates logical address for secondary memory. Main memory has physical addresses. So, we have to translate Logical Address to physical address. Since Pages are loaded in a non-Contiguous fashion in Main Memory. Every program has to maintain a Page Table. Page Table is a data Structure and is stored inside the main memory.

Syed Bakhtawar Fahim
Bakhtawarfahim10@gmail.com

Page Table stores the page no's and frame no's
Foreg:- Page no. 1 is stored at frame no. 4 and so on.

Defination of Page Table – It is the data structure stored in Main Memory that stores the mapping between Virtual addresses and Physical addresses.

Now CPU generates Logical address. The logical address consists of 2 parts — ① Page no. ② Instruction offset then CPU refers the Page Table and finds out which page no. is present at which frame no. and hence translate the Physical address. Physical

address consists of two parts — ① frame no. ② Instruction Offset. foreg:- CPU wants to execute Page no. 1 and inside Page no. 1 instruction no. 5. i.e

P1	5

L.A

P1 is page no. and 5 is the instruction offset. now CPU refers Main memory and finds out (say P1 is present at frame no. 4 (f4).

f4	5

P.A

The instruction offset is same 5. CPU goes to frame no. 4 and picks 5th instruction and executes it and gives the output. This is how we translate L.A to P.A in paging.

Syed Bakhtawar Fahim
Bakhtawarfahim10@gmail.com