UNIT 1: MEMORY MANAGEMENT

Unit Structure

1.0 Learning Objectives

After learning this unit, you will be able to understand:

- Memory Management Unit
- External and internal fragmentation
- Virtual page number
- Paging address Translation
- Virtual and physical memory
- Importance of Catch memory
- Associative memory

1.1 Introduction

Memory management is a type of subsystem which is an important part of an operating system. During the computing period there was continuously need of more memory in computer systems. Strategies have been developed to overcome this limitation and the most successful of these is virtual memory. Virtual memory makes the system appear to have more memory than it actually has by sharing it between competing processes as they need it. With earlier computing it is found that:

- Program should be carryout into the memory and is kept inside a process for further working.
- Input queue which is collecting of processes information on the disk are brought into the memory for implementation.
- The single process implementation will do from input queue which is loaded inside the memory for implementation.
- Once the implementation is done, then the implementation memory space will become free.
- In computers, the address space starts with 00000, which was first address of the user process that cannot be all 0.

1.2 Logical and Physical Address Protection

In the meantime, the computer acquires inter communicated through logical as well as physical addressing in order to map its memory. The logical approach is developed by the processor which is additionally designated as virtual address. The program observes this address space. Whereas the physical address exists the real address which continues assumed by the computer hardware like as memory unit. It is determined that the logical to physical address analysis occurrence acted or conveyed off is experienced by the Operating System. Here, virtual memory designates to the absence of estrangement of logical memory which is glanced nearby the process from physical memory which is inspected nearby the processor. On approximate of this separation, the computer programmer desires to be careful of about logical memory space while the operating system affirms two or more levels of physical memory space.

It is found that during the compile time and load time, the address binding schemes makes these two tend similar but they differ in execution time address

binding scheme and MMU (Memory Management Unit) that caters the translation of such addresses.

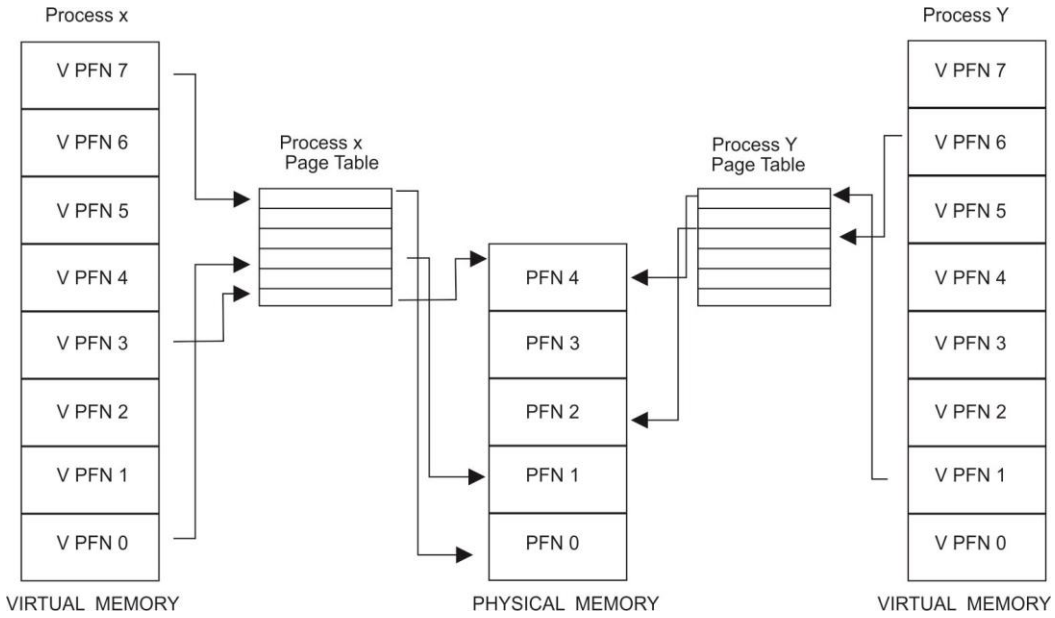

Fig 1.1 Virtual to Physical address mapping

In fig 1.1, it is looked that every procedure in the system comprises its own virtual address space. Similarly virtual address spaces are entirely alienated from each other furthermore on account of a process bounding one application cannot authorize another. Additionally, the hardware virtual memory approaches assign regions of memory to be maintained across writing. This conserves code as well as data from being overwritten by miscreant applications.

In the virtual to physical address mapping demonstrated in fig 1.1, as the processor deliver out the program it understands an instruction from memory furthermore analyse it. While the processor interprets the instruction, it desires to fetch or store the contents of a position in the memory. Following that, the processor will deliver out the instruction furthermore actuate onto the following instruction in the program. With this concept, the processor is consecutively approaching memory to deliver out the instructions or to acquire as well as accumulate the data.

In rack relevantly virtual memory system, complete similar addresses are virtual addresses in addition are not physical addresses. The virtual addresses acquires altered into physical addresses with the support of a processor based on

information that had been conveyed in a set of tables to preserve safe the operating system.

To simpler this, virtual as well as physical memory are allotted as handy sized chunks recognized as pages. Similarly chunks are of identical size where it serves difficult for the system to monitor as Linux on Alpha AXP systems utilizes 8 KB pages as well as on Intel x86 systems it applies 4 KB pages. Every chunk of pages are allotted with a unique number as page frame number (PFN).

Under such model, a virtual address is made of two parts:

- Offset
- Virtual page frame number

If the page size is 4 KB, then the bit ratio will be 11:0 of virtual address which has offset and bits number 12 and over this are virtual page frame number. Every time the processor comes across a virtual address which takes the offset and the virtual page frame number. For this, the processor should translate virtual page frame number into physical frame number and contact the location for correct offset into physical page by using page tables.

It is found that the virtual memory will allow process to be of virtual address spaces, so that there are times when you need processes to share such memory. Now the processor uses virtual page frame number as an index into the processes page table to get back its page table entry. If the entry is valid, then the processor will carry out the physical page frame number from such entry. If the entry is not a valid entry, then the process will try to come out with a non-existing area of its virtual memory. Under such conditions, the processor cannot solve the address and will pass the control to the operating system which is a permanent process.

The concept of 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

Check your progress 1

1. In logical address protection, the logical address is generated by the _____.

   a. memory c. memory address

   b. processor d. none of these

2. Input queue is a collection of process _____on the disk

   a. data c. both a and b

   b. information d. neither a nor b

3. Linux on Alpha AXP systems utilizes __KB pages

   a. 8 c. 32

   b. 16 d. 64

1.3 Paging and Segmentation

Paging

Paging is a process that will help in solving the problem that was seen in case of variable sized partitions such as external fragmentation. In paged system, the logical memory is sliced into number of constant sizes chunks called as pages. Further, the physical memory is pre-divided in certain constant sized blocks which are known as page frames. The page sizes or the frame sizes will be of power 2, and fluctuates between 512 bytes to 8192 bytes per page. They have certain bytes per page because of the implementation of paging mechanism with page number and page offset.

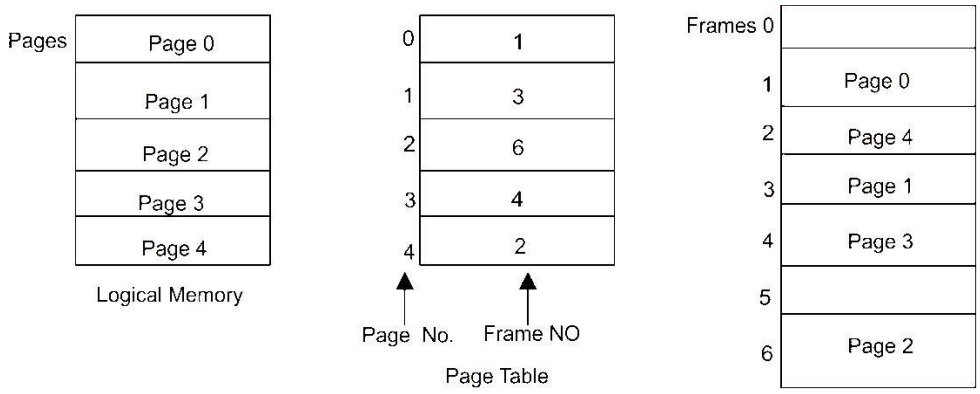

Fig 1.2 Paging operation

In fig 1.2, the process page gets loaded to particular memory frame. Such pages will further be loaded into neighboring frames or in non-neighboring memory as highlighted in the figure 1.2 It is seen that the outside fragmentation gets improved because the processes gets inside in a separate holes.

Page Allocation

With variable sized partitioning of memory, it is seen that every time a process of n size is loaded which is the best location from the list of available/free holes. Such type of dynamic storage allocation is required as it increases the efficiency and throughput of system. This type of selection can be done by using:

1) Best-fit Policy: It allocates the hole where the process is tight as the difference between whole size and process size is lowest.

2) First-fit Policy: This will allocates the initial found hole that can be big enough to fit in the new process.

3) Worst-fit Policy: It allocates the maximum size hole which leaves the full amount of unused space.

From the above three listed strategies it seems that the strategy best-fit and first-fit are better as compared to the worst-fit. Both best-fit and first-fit strategies are efficient in terms of time and storage capacity. In case of best-fit strategy, minimum leftover space is seen which will create the smallest hole which are not used frequently. In case of first-fit strategy, it uses least overheads in order to work because it is the simplest strategy to work upon. Possibly worst-fit also sometimes leaves large holes that could further be used to accommodate other processes. Thus all these policies have their own merits and demerits.

Hardware Support for Paging

It is seen that all logical page in paging scheme is further divided as:

- Page number (p) in logical address space
- Displacement in page pat which item resides

Such arrangement is known as Address Translation scheme as it shows that in case of a 16-bit address, we can divide the address as:

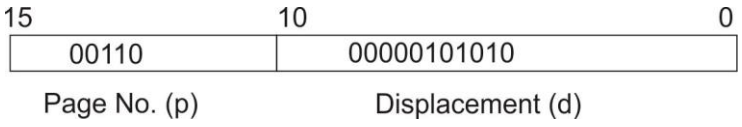

Fig 1.3 dividing the address

From the figure 1.3, it is seen that a page number will take 5 bits with its range starts from value 0 to 31 that can be 2^5-1. Likewise, if we consider an offset value of having 11-bits, then the range will be from 0 to 2023 which is 2^{11}–1. Totally, we see that the paging scheme uses 32 pages, each having 2024 locations. Also, the table that keeps virtual address to physical address translations is further classified as page table. It is found that as the displacement is fixed, the translation of virtual page number to physical page exists which can be seen in the figure 1.4.

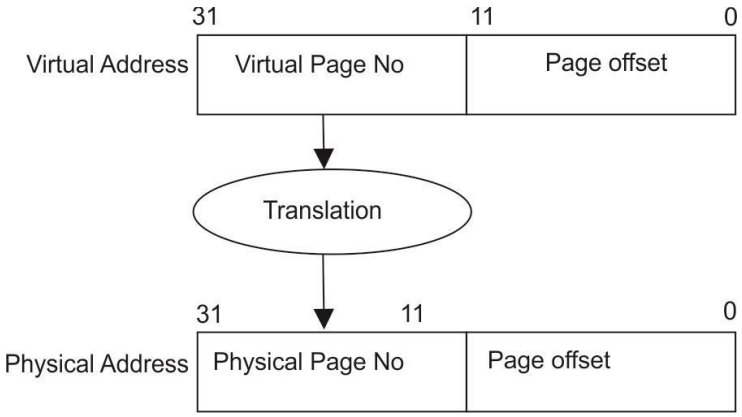

Fig 1.4 Address Translation scheme

It is seen that the page number is required in shape of an index which is into the page table containing base address for every corresponding physical memory page number. This arrangement will lowers the dynamic relocation efforts which are shown by the paging hardware support as in figure 1.5.

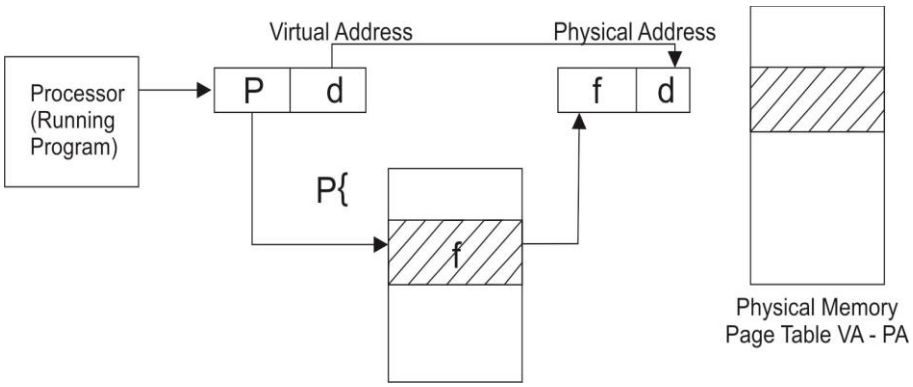

Fig 1.5 Direct Mapping

Paging address Translation by direct mapping

Consider a case of direct mapping as shown in fig 1.5 where a page table sends directly to physical memory page. In this, the drawback is that the speed of translation decreases because the page table is put in primary storage place having a considerably larger in size that increases the instruction execution time and led to lowering of system speed. In order to conquer such situation, the use of extra hardware such as registers and buffers are used.

Paging Address Translation with Associative Mapping

It is based on the utilization of fixed registers that has high speed and efficiency. Such small, fast-lookup Catch will help to put the whole page table in content-addresses associative storage place thereby making the speed to improve and further to care for the lookup problem of the Catch. These are known as associative registers or Translation Look-aside Buffers (TLB's). It is found that each register consists of two entries:

1) Key, which is matched with logical page.

2) Value which returns page frame number corresponding top.

Such arrangement is same as direct mapping scheme but only difference is that we have associative registers having few page table entries that made the search fast. It is quite expensive due to the presence of register support. Hence it is found that both direct and associative mapping schemes will combine to result in more benefits. In this, that page number is coordinated with associative registers at the same time. Also the percentage of number of times the page is found in TLB's is further termed as hit ratio. If it is not found, it is seek out in page table and added into TLB. In case if the TLB is full, then the page replacement policies will come into effect. It is found that the entry in TLB is limited only. Such type of combined scheme is shown in Figure 1.6.

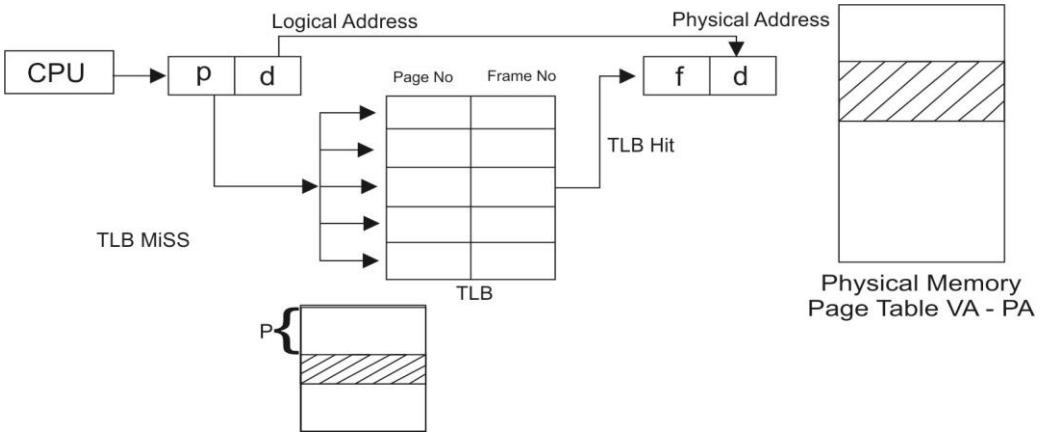

Fig 1.6 Type of combined scheme

It is seen that in paging hardware, there is a presence of some protection mechanism. Inside the page table there exists corresponding frame where a protection bit is linked. Such type of bit will show whether the page is read-only or read-write. In this, sharing code and data will takes place only when two pages table entries in different process shows the similar physical page where every process shares the memory. It is seen that if one process writes the data, then the second process will locate for the changes. Such type of an arrangement is quiet efficient while in communicating. Sharing is required to control in order to protect modification and admission of data in a single process with the help of second process. Such type of programs is kept independent as procedures and data where procedures and data that are pure/reentrant code get shared. Re-entrant code will not be able to change itself and should make sure that it contains a separate copy of per-process global variables. It is predicted that modifiable data and procedures will not share without the intervention of concurrency controls. Such type of non-modifiable procedures sometimes are called as pure procedures or reentrant codes. In case of an example, it is illustrated that in such system only single copy of editor or compiler code be kept in the memory, and all editor or compiler processes and executes sit with the help of single copy of code which will help in memory utilization.

Advantages

There are certain advantages of paging scheme such as:

1. Virtual address space must be greater than main memory size. i.e., can execute program with large logical address space as compared with physical address space.
2. Avoid external fragmentation and hence storage compaction.
3. Full utilization of available main storage.

Disadvantages

The disadvantages of paging scheme include:

1. Internal fragmentation problem led to wastage inside the allocated page
2. Extra resource consumption
3. Overheads for paging hardware
4. Virtual address to physical address translation takes place

Segmentation

In generic, a consumer or a programmer likes to observe system memory as an assembly of variable-sized allocations rather than as a linear arrangement of words. Segmentation occurs as a memory management arrangement that accepts this glance of memory.

Principles of Operation

Segmentation demonstrates an exchange arrangement for memory management. This arrangement bisects the logical address space into variable length allocations, named segments, with no appropriate sequencing among them. Each allotment has a name and a length. For clarity, segments are acknowledged by a segment number, rather than by a name. Hence, the logical addresses are acknowledged as a pair of segment number as well as offset within segment. It empowers a program to be broken down into feasible parts according to the user opinion of the memory, which is that time mapped into physical memory. Furthermore logical addresses are two-dimensional although actual physical addresses are still one-dimensional arrangement of bytes only.

Address Translation

This mapping between two is done by segment table, which contains segment base and its limit. The segment base has starting physical address of segment, and segment limit provides the length of segment. This scheme is depicted in Figure 1.7.

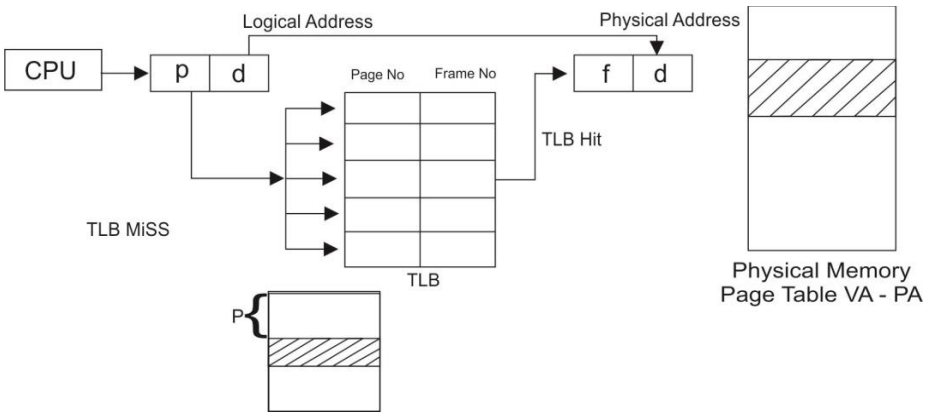

Fig 1.7 Address translation

The offset d must range between 0 and segment limit/length, otherwise it will generate address error. For example, consider situation shown in Figure 1.8.

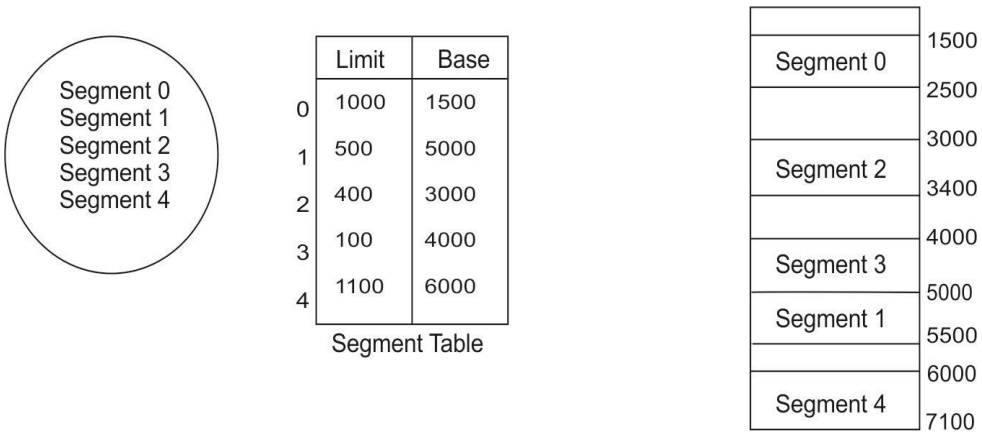

Fig 1.8 Principle of operation of representation

This approximation is comparable to adaptable partition allocation method with advancement that the process is bifurcated into parts. For quick retrieval we can utilize registers as in paged approach. This is comprehended as a segment-table length register (STLR). The segments in a segmentation mechanism dispatch to logical divisions of the process additionally are described by program names. Extract the segment number along with offset from logical address originally so that time the use of segment number as index into segment table gets capture segment base address along with its limit /length. Additionally, contemplate that the offset is not greater than allocated limit in segment table. Today, normally physical address is acquired by adding the offset to the base address.

Protection and Sharing

This approach in addition enables segments that are read-only to be allotted, so that two approaches can utilize shared code for advance memory efficiency. The intervention is comparable that no program can read from or write to chunks belonging to another program, except the allocations that have been set up to be apportioned. With each segment-table entry safety bit differentiating segment as read-only or execute unique can be employed. So fallacious attempt to write into a read-only segment can easily be preserved.

Sharing of segments can be accomplished by constructing common /same entries in segment tables of two asymmetric processes which point to equivalent physical location. Segmentation may continue from external fragmentation i.e., when blocks of released memory are not sufficient to adjust a segment. Storage compression as well as coalescing can shorten this barrier.

Check your progress 2

1.The page sizes or frame sizes is in the range of-

a. 512 bytes to 8192 bytes per page

b. 128 bytes to 512 bytes per page

c. 1024 bytes to 2048 bytes per page

d. 2048 bytes to 4098 bytes per page

2. In Best-fit Policy, the difference between hole size and process size is-

a. maximum

b. lowest

c. half

d. none of these

1.4 Virtual Memory

Virtual memory is an approach that empowers the accomplishment of processes which are not comprehensively obtainable in memory. The core clear benefit of this approach is that programs can be extended than physical memory. Virtual memory is the division of user logical memory from physical memory.

This division authorizes an intensely large virtual memory to be delivered for programmers when only a smaller physical memory is obtainable. Following are the circumstances, when complete program is not essential to be loaded completely in main memory.

- User written error handling routines are used only when an error occurred in the data or computation.
- Certain options and features of a program may be used rarely.
- Many tables are assigned a fixed amount of address space even though only a small amount of the table is actually used.
- The ability to execute a program that is only partially in memory would counter many benefits.
- Less number of I/O would be needed to load or swap each user program into memory.

- A program would no longer be constrained by the amount of physical memory that is available.
- Each user program could take less physical memory; more programs could be run the same time, with a corresponding increase in CPU utilization and throughput.

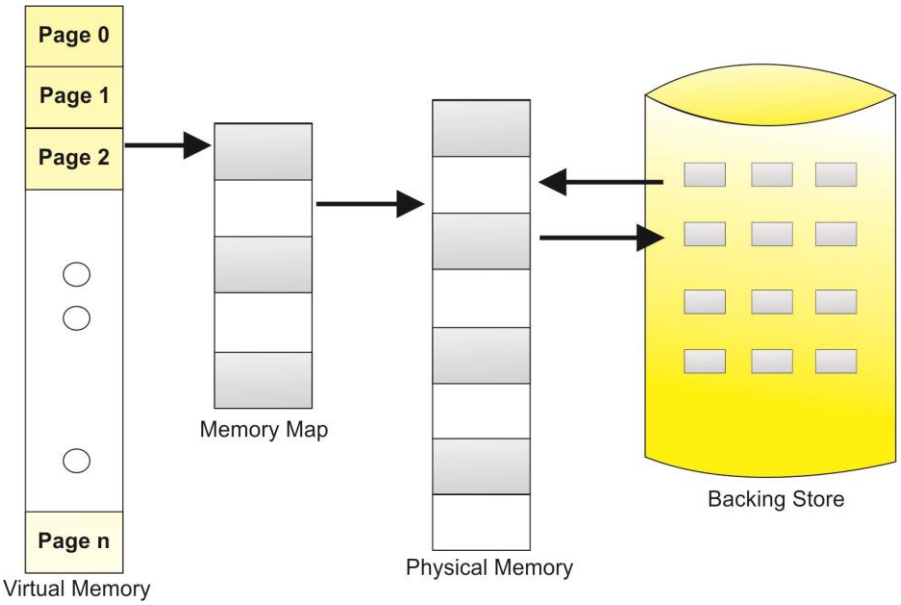

Fig 1.9 Virtual memory

Virtual memory is frequently exercised by demand paging. It can additionally be exercised in a segmentation system. Demand segmentation can further be utilized to supply virtual memory.

A demand paging mechanism is quite comparable to a paging system with exchanging. When we expect to achieve a process, we exchange it into memory. Rather than exchanging the complete process into memory, furthermore, we facilitate a lazy swapper called pager.

When a process is to be exchanged in, the pager conceives which pages will be facilitated before the process is exchanged out again. Instead of exchanging in a whole process, the pager carries only those essential pages into memory. So, it bypasses reading into memory pages that will not be used in anyway, shortening the swap time as well as the amount of physical memory expected.

Hardware support is essential to discriminate between those pages that are in memory as well as those pages that are on the disk employing the valid-invalid character scheme where correct as well as defective pages can be examined by checking the bit. Marking a page will hold no effect if the process never

approaches to approach the page. While the process achieves as well as accesses pages that are memory resident, execution approaches predominantly.

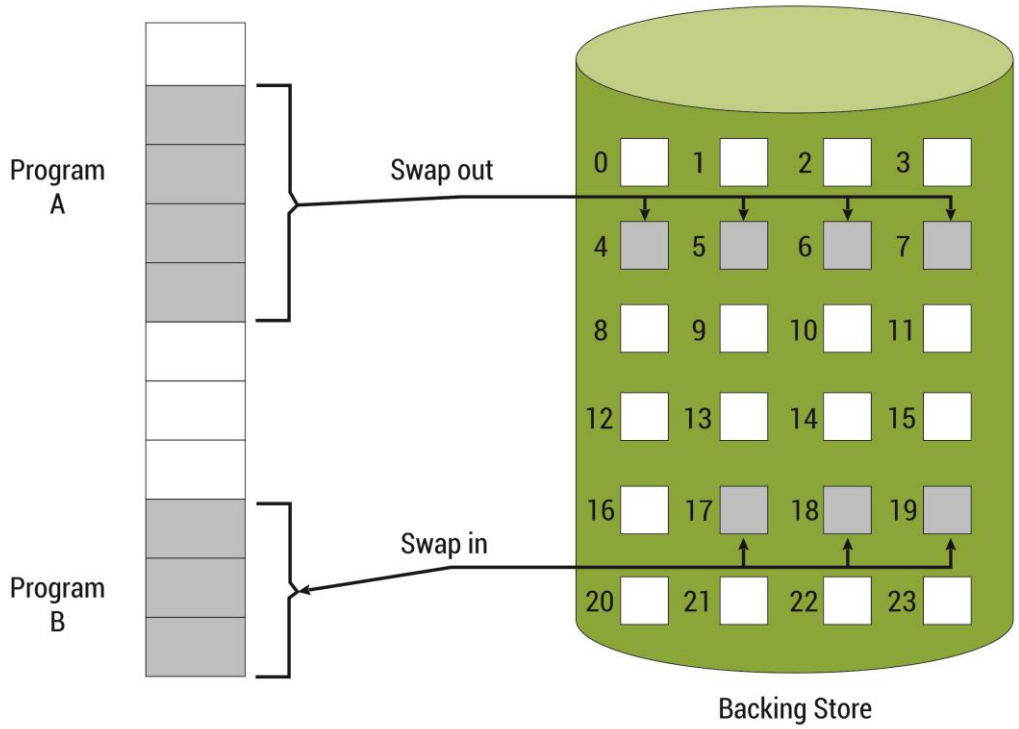

Fig 1.10 Demand paging system

Check your progress 3

1. The presence of virtual memory helps to share the memory among them.

a. processes
c. instructions
b. threads
d. none of the mentioned

2. _____ is the concept where a process is copied into main memory from secondary memory as required.

a. Paging
c. Segmentation
b. Demand paging
d. Swapping

3. Swap space is present in:

a. primary memory
c. CPU
b. secondary memory
d. none of the mentioned

1.5 Page Replacement Algorithms

Page replacement algorithms are the mechanisms exercising which Operating System determines which memory pages to exchange out, write to disk when a page of memory expects to be assigned. Paging occurs whenever a page fault arises also a free page cannot be facilitated for allocation approach accounting to analysis that pages are not obtainable or the number of free pages is shorten than necessary pages.

When the page that was chosen for exchanged and was paged out, is referenced again that time it has to read in from disk, furthermore this stipulates for I/O completion. This approach considers the quality of the page replacement algorithm: the lesser the time subsiding for page-ins, the better is the algorithm. A page replacement algorithm beholds at the edged information about approaching the pages delivered by hardware, additionally tries to choose which pages should be replaced to shorten the total number of page lacks, while balancing it with the costs of primary storage along with processor time of the algorithm itself. There are abundant different page replacement algorithms. We calculate an algorithm by bounding it on a definite string of memory reference as well as assessing the number of page faults.

RAND (Random)

- Determine some page to change at random.
- Affirms the following page to be referenced exists random.
- Can check breach algorithms across random page exchanged.

MIN (minimum) or OPT (optimal)

- Belady's optimal algorithm for the minimal number of page defects.
- Change the page that will be referenced best in the future or not at all.
- Problem: we cannot apply it, since we cannot forecast the future.
- This is the better case.
- Can exercise it to match external algorithms against

FIFO (First In, First Out)

- Choose the page that has been in main memory the longest.
- Exercise a chain (data structure).

- Problem: however a page has been residing for a long time, it may be absolutely useful.
- Windows NT as well as Windows 2000 utilize this algorithm, as a local page replacement algorithm (explained distinctly), with the pool approach (described in more detail separately).
- Construct a bay of the pages that have been labeled for removal.
- Manage the pool in the identical way as the rest of the pages.
- If a latest page is expected, take a page from the pool.
- If a page in the bay is referenced again foregoing being replaced in memory, it is clearly reactivated.
- This is relatively efficient.

LRU (Least Recently Used)

- Select the page that was final referenced the longest time ago.
- Affirms current behavior is a good guru of the immediate future.
- Can control LRU with a list identify the LRU stack or the paging stack (data structure).
- In the LRU stack, the initial entry explains the page referenced least recently, the last entry describes to the last page referenced.
- If a page is referenced, proceed it to the end of the list.
- Problem: stipulates updating on every page referenced.
- Too slow to be used in practice for controlling the page table, however many systems use assessments to LRU.

NRU (Not Recently Used)

- As an appraisal to LRU, choose one of the pages that has not been exercised currently (as opposed to identifying exactly which one has not been employed for the longest amount of time).
- Save one bit identified the "used bit" or "reference bit", where 1 => used recently and 0 => not used recently.
- Variants of this scheme are exercised in numerous operating systems, involving UNIX along with Macintosh.

- Most variations facilitate a scan pointer and pass through the page frames one by one, in some order, inspecting for a page that has not been used currently.

Working Set (WS)

- Clearly address the problem of thrashing.
- Thrashing: when the computer system is compulsive with paging, i.e., CPU has little to do but there is heavy disk traffic moving pages to and from memory with little use of those pages.
- Working set: the pages that a process has used in the last w time intervals.
- Choose any page that is not in the working set.
- Global: not in the working set of much ready process.
- If no such page exists, swap out some process.
- The medium-term scheduler places a process in the waiting-for-memory queue.

Working Set Policy

- Block the number of processes in the develop list so that whole can have their functioning set of pages in memory.
- Before beginning a process, make sure it's working set is in main memory.
- Too costly in practice, however there are some good approximations.

Page Fault Frequency algorithm (PFF) -- dissimilarity of Working Set

- When a page fault exists, if the last page fault for that process was fresh, that time increase the size of its working set (up to a maximum).
- All processes begin with a default ws dimension.
- Load original code pages, original data pages, original stack pages.
- If a process holds not faulted currently, ease the size of its ws i.e. exclude all pages not used currently ("used bit").
- Exercised in Windows NT as well as Windows 2000 as a complete page replacement algorithm (described separately).
- They assign to it as automatic working-set trimming.
- Additionally, in WinNT, can call the process object service to alter working-set min as well as max for a process, up to a defined max as well as min.

Check your progress 4

1. Page replacement algorithms decides:

   a. which memory pages to be exchanged

   b. which segment pages to be exchanged

   c. which data pages to be exchanged

   d. all

2. In FIFO page replacement algorithm, the page to be replaced_____.

   a. with oldest page selected

   b. with new page selected

   c. random page selected

   d. none

3. Which algorithm select page that was not used for long period whenever a page is replaced?

   a. first in first out algorithm

   b. additional reference bit algorithm

   c. least recently used algorithm

   d. counting based page replacement algorithm

1.6 Catch Memory

The Catch Memory exists the Memory which is very nearest to the CPU, complete the current Instructions are saved into the Catch Memory. The Catch Memory is connected for storing the input which is allotted by the user additionally which is essential for the CPU to play a work. But the size of the Catch Memory is additionally low in contrast to Memory as well as Hard Disk.

Importance of Catch memory

The Catch memory lies in the direction between the processor as well as the memory. The Catch memory hence, has underling access time than memory further is faster than the main memory. A Catch memory acquires an access time of 100ns, while the main memory may acquire an access time of 700ns.

The Catch memory is very costly moreover owing to subsists limited in capacity. Earlier Catch memories were practicable individually but the microprocessors include the Catch memory on the chip itself.

Expectation for the Catch memory is just to the mismatch between the speeds of the main memory as well as the CPU. The CPU clock as lectured earlier is very fast, whereas the main memory access time is contrastingly slower. Therefore, no matter how fast the processor is, the processing speed depends additional on the speed of the main memory (the energy of a chain is the energy of its weakest link). It is on account of this analysis that a Catch memory acquires access time closer to the processor speed that is created.

The Catch memory stores the program (or its part) currently being executed or which may be executed within a short period of time. The Catch memory additionally accumulates temporary data that the CPU may commonly stipulate for manipulation.

The Catch memory performs according to diversified algorithms, which determine what information it acquires to store. These algorithms work out the chance to adopt which data would be most repeatedly expected. This probability is worked out on the basis of past attestations.

It appears as a high speed buffer between CPU along with main memory additionally is used to temporary store very energetic data and action all along processing because the Catch memory is faster than main memory, the processing speed is elevated by making the data as well as instructions desired in current processing available in Catch. The Catch memory is very costly and therefore is bordered in capacity.

Check your progress 5

1.The closest memory to the CPU is:

a. RAM c. Catch

b. ROM d. all

2. A Catch memory acquires an access time of:

a. 100ns c. 350ns

b. 700ns d. 500ns

1.7 Hierarchy of Memory Types

Memory hierarchy is employed in computer architecture when chattering behaviour events in computer architectural idea, algorithm predictions, as well as the compact level programming composes such as confounding locality of reference. A "memory hierarchy" in computer storage discriminates each level in the "hierarchy" by response time. There are physically various brands of memory acquiring significant asymmetries in the time to read or write the contents of a peculiar position in memory, the measure of information that is read or written on an allotted condition, the complete volume of information that can be stored, along with the unit amounts of storing an assigned amount of information. To optimize its operation as well as to capture greater efficiency along with economy, memory is arranged in a hierarchy with the greatest performance as well as in universal the best high-priced devices at the top, as well as with progressively lessen performance additionally less costly devices in following layers as shown in fig 1.11 The contents of a certain memory hierarchy, along with the way in which data flows between immediate layers, might be arranged as results.

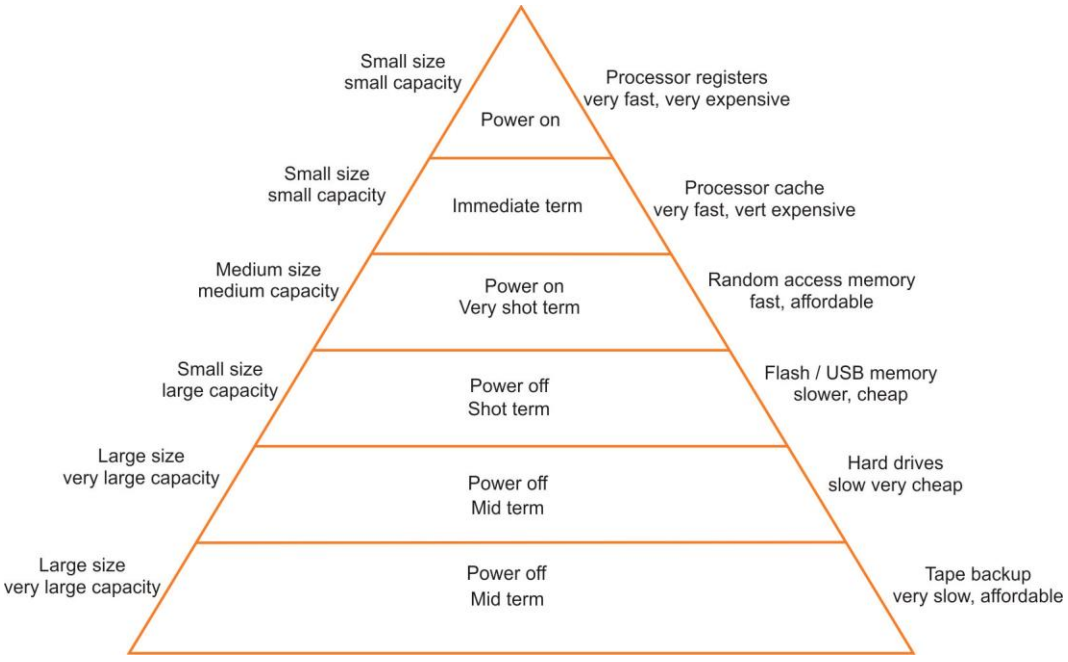

Fig 1.11 Memory hierarchy

Register

A single word confined in each register of the processor; definitely a word includes 4 bytes. This is sometimes not considered of as chunk of the hierarchy.

Catch

These are bunches of words within the Catch; definitely an individual group in the Catch will gather 64 words (say 256 bytes), along with there will be, say, 1024 alike groups, assigning a complete Catch of 256 KBs. Individual word flows between the Catch as well as registers within the processor. All transfers into further out of the Catch are controlled completely by hardware.

Main memory

Words within the main (random-access) memory. On a very high performance system, groups of words acknowledging to a group within the Catch are conveyed between the Catch as well as the main memory in a unique cycle of main memory. On lower-performance systems the dimension of the group of words in the Catch is more than the width of the memory bus, along with the transfer takes the category of a chain of memory cycles. The algorithm that administers this movement is exercised completely in hardware. Main memory measurements are very variable – from as little as 1 GB on a compact system up to numerous GB on a high-performance system.

Online backing store

Blocks of words confined on permanently connected backing store. There may be bilateral somewhat abnormal forms of activity here:(a) swapping device – pages (of say 4 KBs) or segments (up to many GBs) of memory acquired on a swapping device are carried as comprehensive units between their backing-store home along with a page frame or segment domain in main memory, underneath the control of an algorithm applied by the software of the operating system furthermore with hardware assistance to denote when pages or segments are to be actuated;(b) backing store – comprehensive files, or apparently identifiable subsections of big files, are coursed between the backing-store device along with the main memory in acknowledgment to accurate actions by the programmer, normally by a supervisor call to the operating system.

Demountable storage

Comprehensive files, assisted up onto removable disks or magnetic tape within the file archive system along with the archiving system. Accomplished files are preceded in both directions. The development of backup copies along

with the improvement of a backed-up file may be automatic, or may stipulate direct facilitation by the end user. For additional systems the backup agency is definitely a changed feature of a video or audio cassette system, perchance elevated in some structure of computer-controlled cassette-handling robot assessment. Smaller systems may conduct a cassette system or floppy disks.

Read-only library

Accomplished files, as well as collections of affiliated files connecting to an individual application, contained on read-only devices alike as CD-ROM, or on a machine with numerous appearance of write-protection discipline. Complete measures of files are read into the mechanism from the read-only device, however for distinct reasons there are never any transits from the system to the device.

Check your progress 6

1. Register is of:

   a. 5 bytes c. 7 bytes

   b. 4 bytes d. 16 bytes

2. Catch memory contains _____bytes.

   a. 128 bytes c. 256 bytes

   b. 512 bytes d. 1024 bytes

1.8 Associative Memory

Memory that is approached by content rather than by address; content addressable is for the time being applied synonymously. An Associative Memory authorizes its users to discriminate part of a pattern or key as well as acquire the values affiliated with that model.

An associative memory is a content-addressable architecture that maps a portion of input prototypes to a set of output prototypes. There are two categories of associative memory:

- auto associative
- hetero associative

An auto associative memory accumulates a formerly stored prototype that most immediately looks like the today's prototype. In a hetero associative memory, the accumulated prototype is, in general, distinct from the input prototype not only in content yet perhaps furthermore in type as well as format.

In 1988, Kosko enlarged the Hopfield example by encompassing an incremented layer to act ceaseless auto associations as favourably as hetero associations on the Catch memories. The network architecture of the bi-directional associative memory (BAM) example resembles to that of the linear associate although the connections are bi-directional, i.e. BAM allocates forward as well as backward transfer of information between the layers. The BAM example can conduct either auto associative as well as hetero associative remembers of stored information.

Check your progress 7

1. In auto associative memory, the stored prototype looks like:

a. current prototype c. accumulated prototype

b. defined prototype d. all

1.9 Let Us Sum Up

In this unit, we have learned:

- That memory management is a type of subsystem which is an important part of an operating system.
- Input queue which is collecting of processes information on the disk.
- The page sizes or the frame sizes will be of power 2, and fluctuates between 512 bytes to 8192 bytes per page.
- Segmentation occurs as a memory management arrangement that accepts this glance of memory.
- Virtual memory is frequently exercised by demand paging.
- Page replacement algorithms are the mechanisms exercising which Operating System determines which memory pages to exchange out.
- The Catch Memory exists as Memory which is very nearest to the CPU.

1.10 Answers for Check Your Progress

Check your progress 1

Answers: (1-b), (2-c), (3-a)

Check your progress 2

Answers: (1-a), (2-b)

Check your progress 3

Answers: (1-a), (2-b), (3-c)

Check your progress 4

Answers: (1-a), (2-a), (3-c)

Check your progress 5

Answers: (1-c), (2-a)

Check your progress 6

Answers: (1-b), (2-c)

Check your progress 7

Answers: (1-c)

1.11 Glossary

1. **Memory hierarchy** - Refers to different types of memory.
2. **Catch Memory** - It is the closest memory available for the CPU.

1.12 Assignment

What are the four important tasks of a memory manager?

1.13 Activities

What are the three tricks used to resolve absolute addresses?

1.14 Case Study

What are the problems that arise with absolute addresses in terms of swapping?

1.15 Further Readings

1. The Operating system by Andrew Tannenbaum.
2. Operating System by Mach.