Paging Memory management

Noncontiguous Allocation

This means that memory is allocated in such a way that parts of a single logical object may be placed in noncontiguous areas of physical memory.

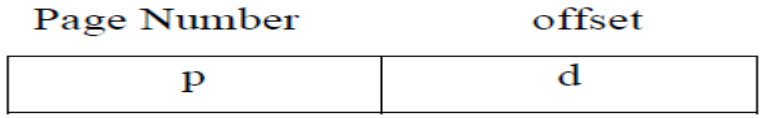
Address translation performed during the execution of instructions establishes the necessary correspondence between a contiguous virtual address space and the noncontiguous physical addresses of locations where object items reside in physical memory at run time.

Paging

- It is a memory management scheme that removes the requirement of contiguous allocation of physical memory.
- The physical memory is conceptually divided into a number of fixed size slots called page *frames*.
- The virtual address space of a process is also split into fixed size blocks of the same size called *pages*.
- The page sizes are usually an integer power of base 2. Allocation of memory consists of finding a sufficient number of unused page frames for loading the requesting process's page.
- An address translation mechanism is used to map virtual pages to their counterparts. Since each page is mapped separately, different page frames allocated to a single process need not occupy contiguous areas of physical memory.
- Paging solves the external fragmentation problem by using fixed sized units in both physical and virtual memory.

Allocation

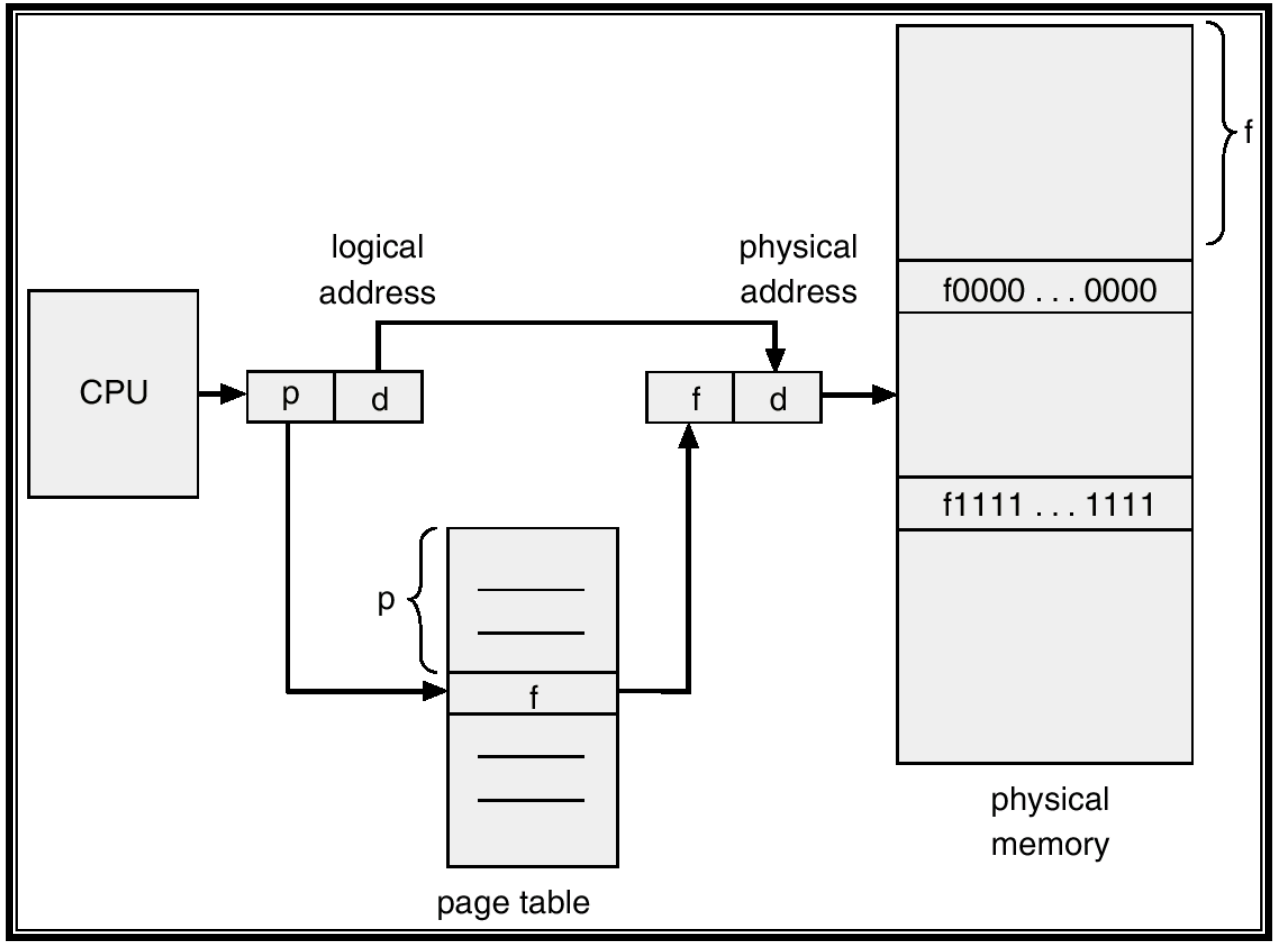
- When a process is to be executed, its corresponding pages are loaded into any available memory frames.
- The mapping of virtual address to physical addresses in paging systems is performed at the page level.
- Each virtual address is divided into two parts: Page Number (Virtual) and the offset within that page.



where p is the index of PT and d is the displacement within the page.

- In paging systems, address translation is performed with the aid of a mapping table called the Page Table (PT).
- There is one PT entry for each virtual page of a process and the higher order bits of the physical base address (Page frame number) is stored in a PT entry.
- The logic of the address translation process in paged system is as follows:
  - The virtual address is split by hardware into the page number and the offset within that page.
  - The page number is used to index the PT and to obtain the corresponding physical frame number.
  - This value is then concatenated with the offset to produce the physical address, which is used to refer the target item in memory.

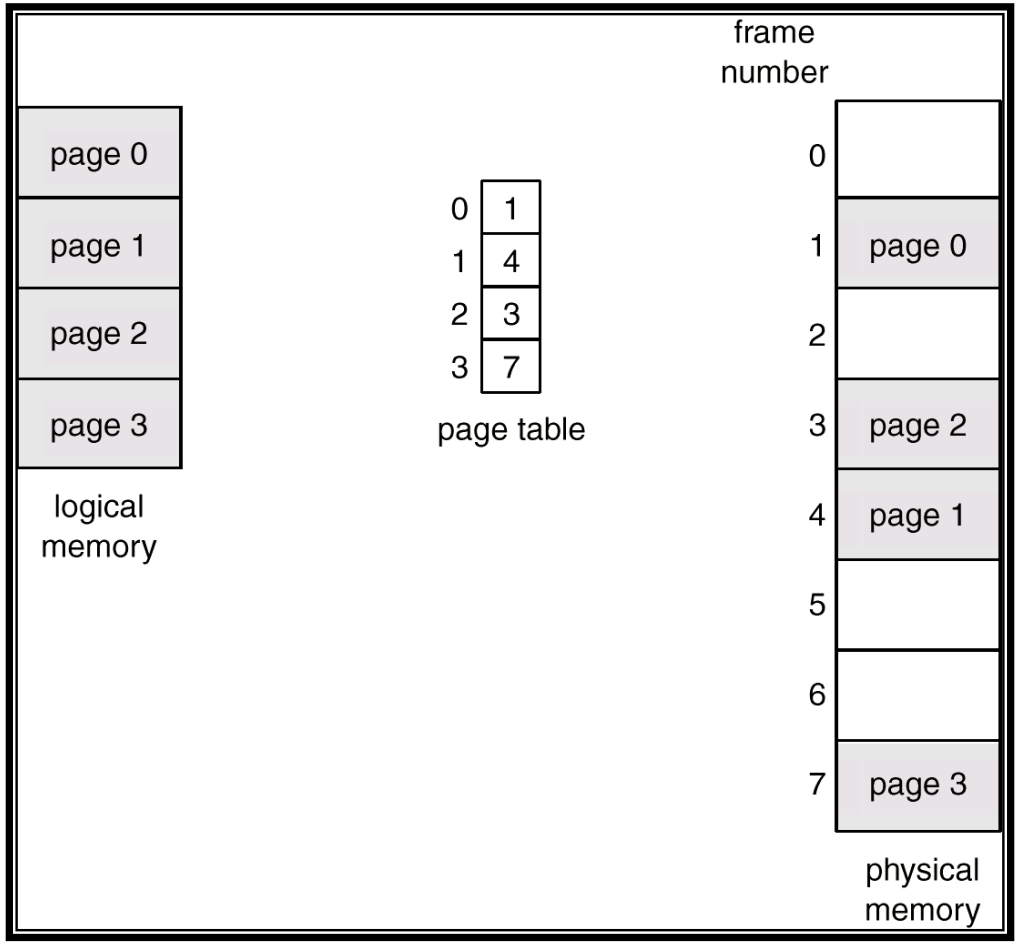
- OS keeps track of the status of each page frame by means of a physical memory map that may be structured as a static table referred as Memory Map Table (MMT). MMT has a fixed number of entries that is identical to the number of page frames in a given system.
- f = m/p where f is the number of page frames, m is the capacity of the installed physical memory and p is the page size.
- When a process of size s requests the OS for allocation, OS must allocate n free page frames so that n = ⌈(s/p)⌉ where p is the page size.
- If the size of a given process is not a multiple of the page size, then the last page frame may be partly unused which is called page fragmentation or page breakage.
- All frames fit all pages and any fit is as good as any other.


CPU
logical
address
p
d
physical
address
f
d
f
f0000 . . . 0000
f1111 . . . 1111
p
f
page table
physical
memory

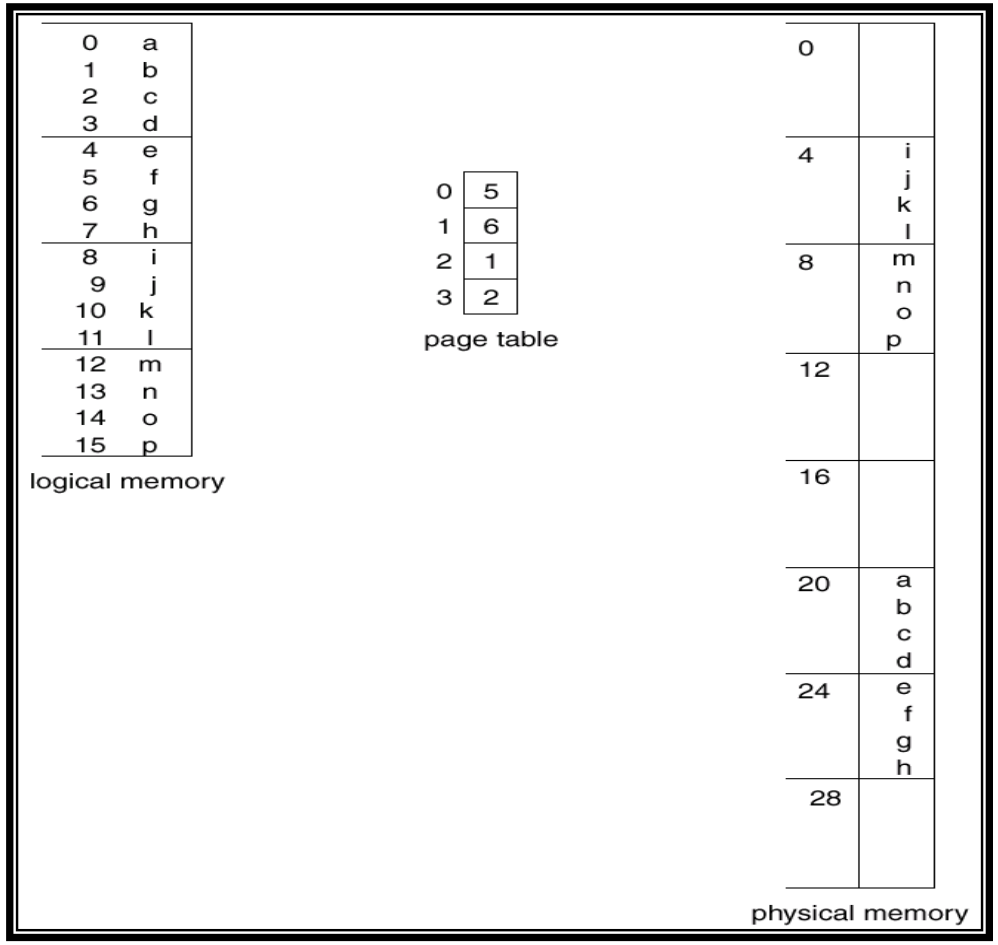
- The efficiency of the memory allocation algorithm depends on the speed with which it can locate free page frames.
- Assuming free frames are randomly distributed in memory, average number of MMT entries x, that needs to be examined in order to find n free frames may be expressed as

  $x = n/q$ where q is the probability that a given frame is available for allocation.
- The number of MMT entries searched x is directly proportional to kn, where $k = 1/q$ and $k \geq 1$.
- If static table form is used then inverse proportion of x to q implies that the number of MMT entries that must be examined in order to locate x frames, which increases with the amount of memory in use.
- An alternative solution is to link the page frames into a free list.

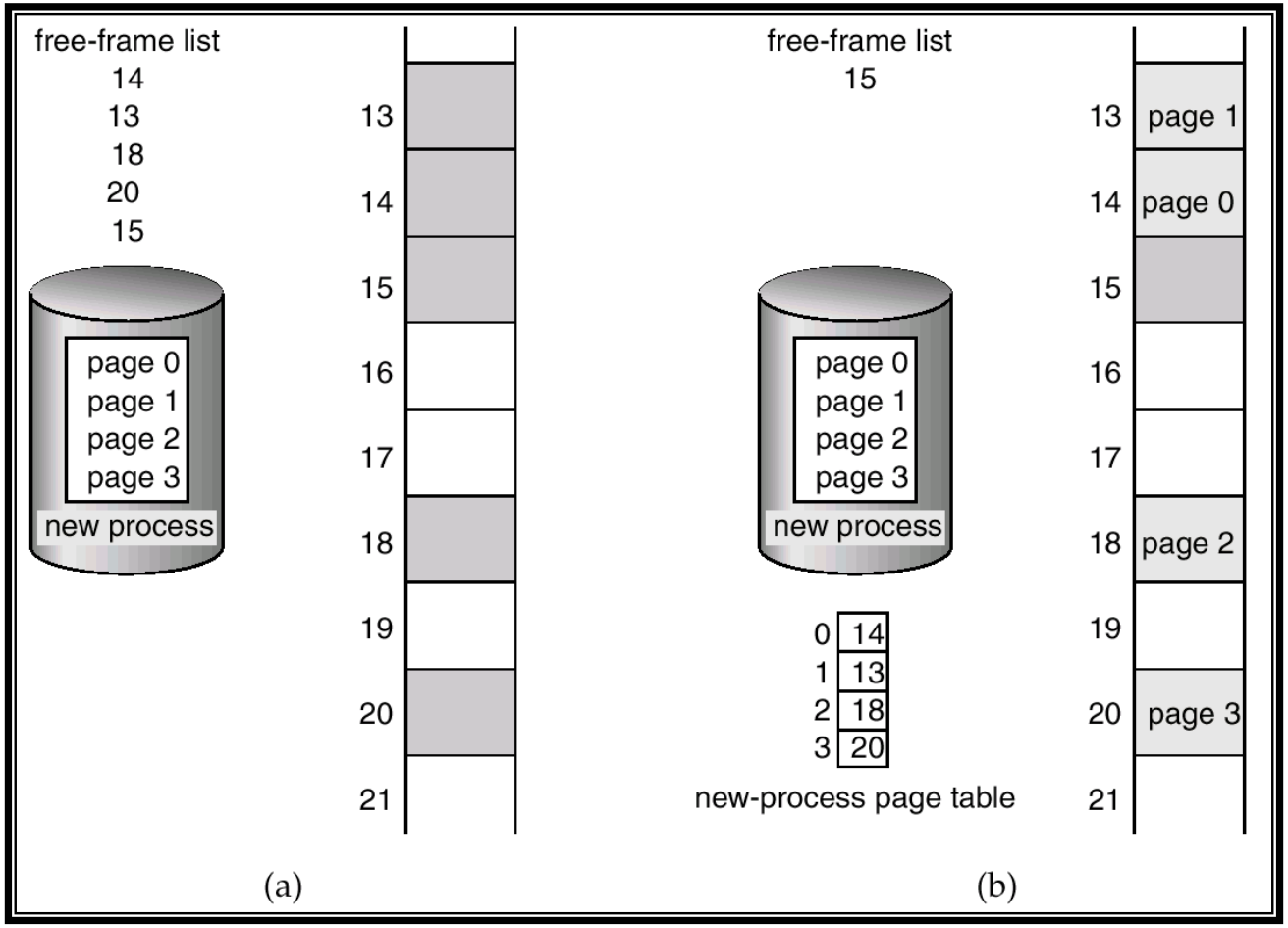
Example


frame
number
page 0
page 1
page 2
page 3
logical
memory
0 1
1 4
2 3
3 7
page table
0
1 page 0
2
3 page 2
4 page 1
5
6
7 page 3
physical
memory

Example


0 a
1 b
2 c
3 d
4 e
5 f
6 g
7 h
8 i
9 j
10 k
11 l
12 m
13 n
14 o
15 p
logical memory
0 5
1 6
2 1
3 2
page table
0
4 i j k l
8 m n o p
12
16
20 a b c d
24 e f g h
28
physical memory

Free Frames


free-frame list
14
13
18
20
15
page 0
page 1
page 2
page 3
new process
13
14
15
16
17
18
19
20
21
(a)
free-frame list
15
page 0
page 1
page 2
page 3
new process
0 14
1 13
2 18
3 20
new-process page table
13 page 1
14 page 0
15
16
17
18 page 2
19
20 page 3
21
(b)

Hardware support for paging

- It concentrates on conserving the memory necessary for storing of the mapping tables and on speeding up the mapping of virtual to physical address.
- Each PT is constructed with only as many entries as its related process has pages.
- Page Table Limit Register (PTLR) is set to the highest virtual page number defined in the PT of the running process.
- Accessing of the PT of the running process may be facilitated by means of the PT base register (PTBR) which points to the base address of the PT of the running process.

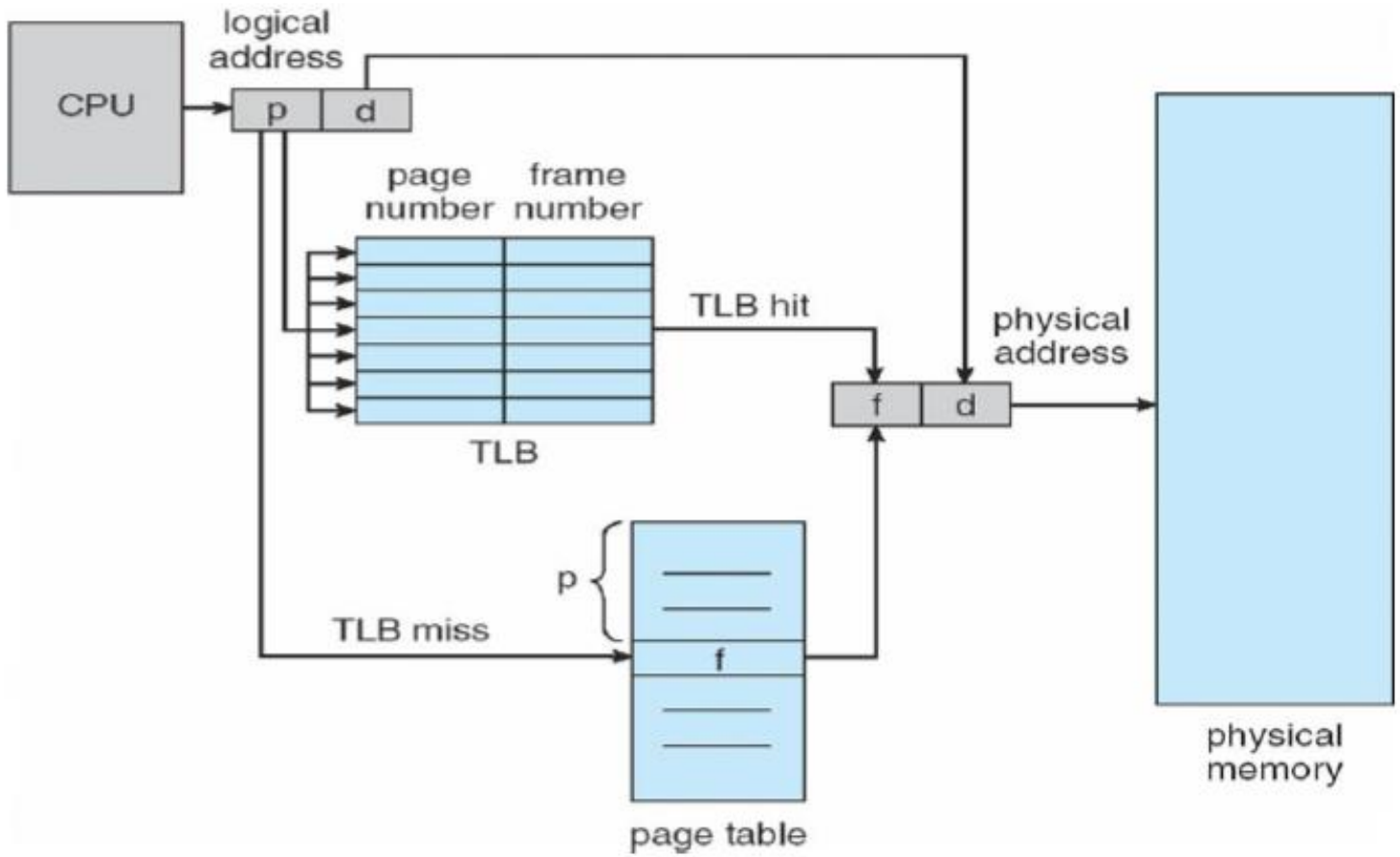
- The values loaded in the register are defined at process loading time and stored in the related PCBs.
- Address translation in paging systems also requires two memory references: One to access the PT for mapping and the other to refer the target item in physical memory.

- One popular approach is to use a high speed associative memory for storing a subset of often used PT entries called as Translation Lookaside Buffer (TLB).
- Address translation begins by presenting the page number portion of the virtual address to the TLB.
- If it is present then the page frame is combined with offset to produce the physical address. The target entry is searched in PT if it is not in TLB.

Associative Memory

Page #	Frame #

- Associative memory – parallel search
- Address translation (A´, A´´)
  - If A´ is in associative register, get frame # out.
  - Otherwise get frame # from page table in memory


CPU
logical address
p
d
page number
frame number
TLB hit
TLB
physical address
f
d
p
TLB miss
f
page table
physical memory

Effective Access Time

- Associative Lookup = ε time unit
- Assume memory cycle time is 1 microsecond
- Hit ratio – percentage of times that a page number is found in the associative registers; ratio related to number of associative registers.
- Hit ratio = α
- Effective Access Time (EAT)

$$EAT = (1 + \varepsilon)\,\alpha + (2 + \varepsilon)(1 - \alpha)$$

$$= 2 + \varepsilon - \alpha$$

Structure of PTs

Hierarchical Paging:

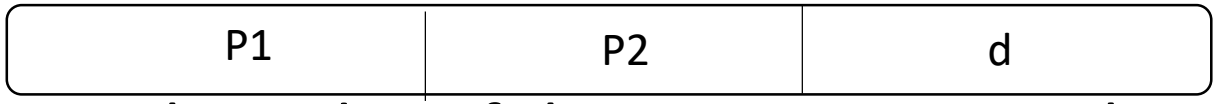
For systems with large logical/virtual address space, the size of PT needed will be in large size.

A PT of large size can be divided to smaller pieces to avoid allocating a large contiguous space.

This can be accomplished in several ways.

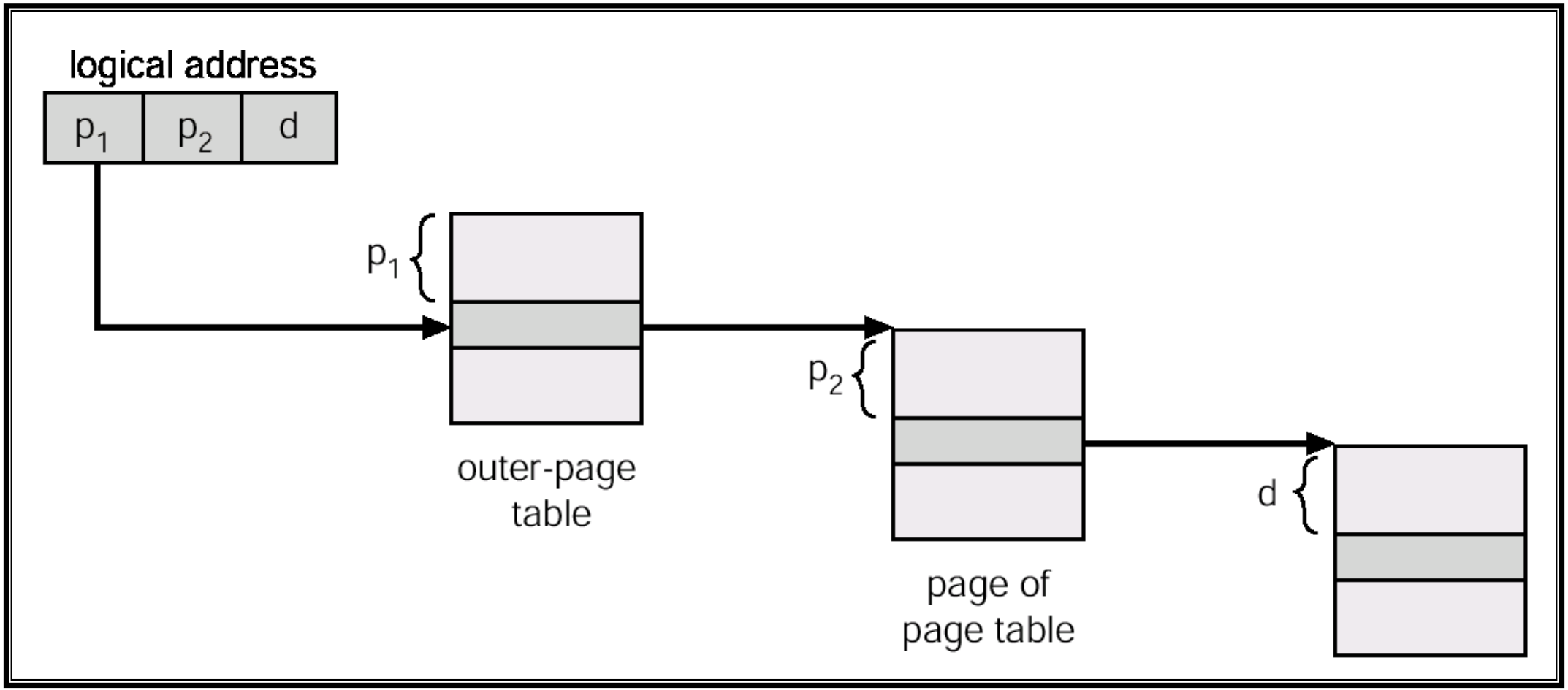
Two level paging

PT itself is treated as pages. The page number component of virtual address is split into two portions one is the index of the outer table and second is the offset for PT.

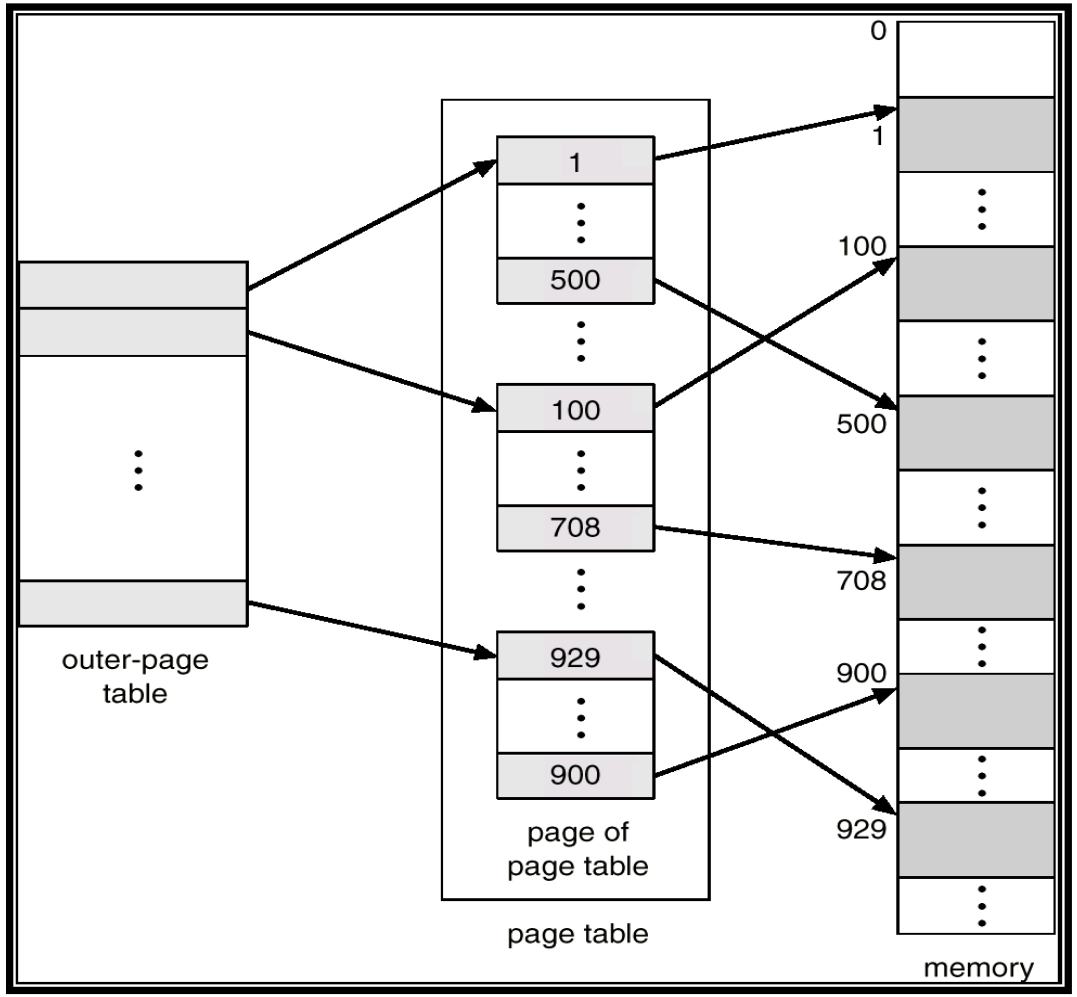


where P1 is the index of the outer PT, P2 is the displacement within the page of the outer page table and d is the displacement within the appropriate page.

Address-Translation Scheme



Two-Level Page-Table Scheme



Variation of Two level Paging:

- The virtual address space of a process is divided into equal sections. The virtual address consists of section number, index to the PT and offset.

For systems with very large virtual address space three level or four level paging can be used.

Hashed Page Tables

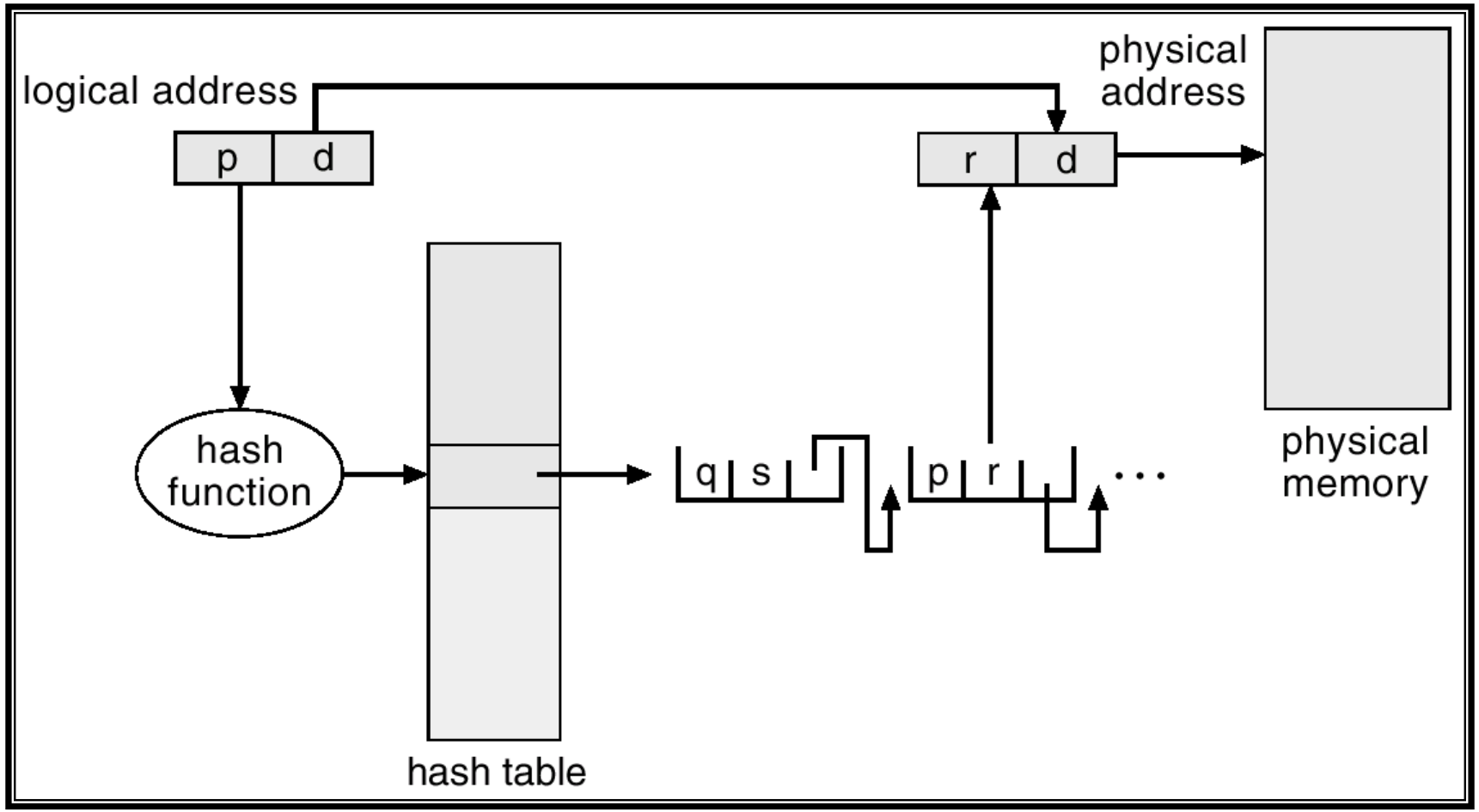
For handling address space larger than 32 bits, a hash page map table is used where the hash value is the virtual page number.

Each entry in the hash table contains a linked list of elements that hash to the same locations.

Each element consists of three fields:

- Virtual Page number
- Value of the mapped page frame
- A pointer to the next element in the linked list.

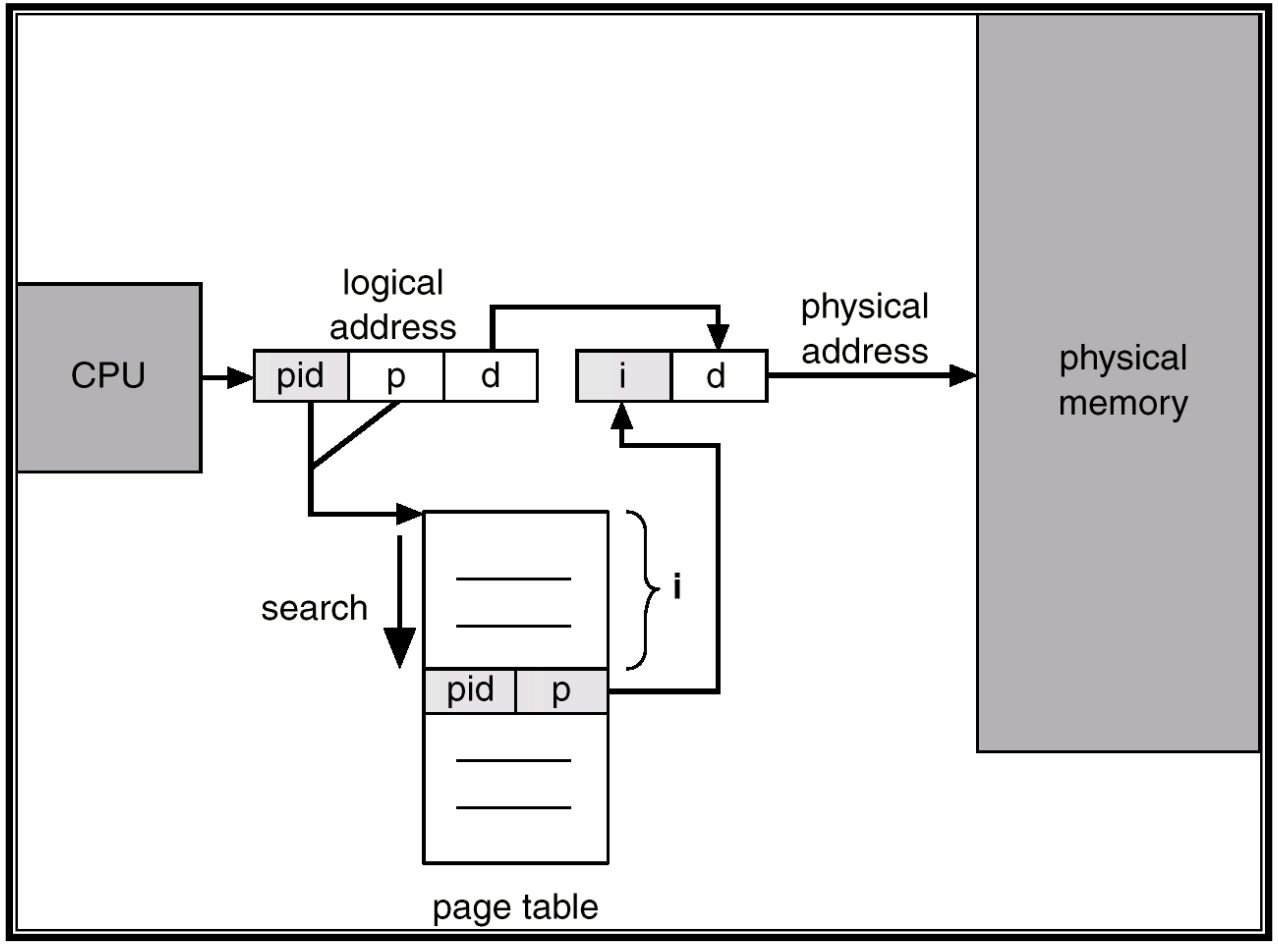
Hashed Page Table



Inverted Page table

- An inverted PT has one entry for each page frame of memory.
- Each entry consists of the virtual address of the page stored in that physical location with information about the process that owns the page.
- Virtual address consists of process id, page number and offset.

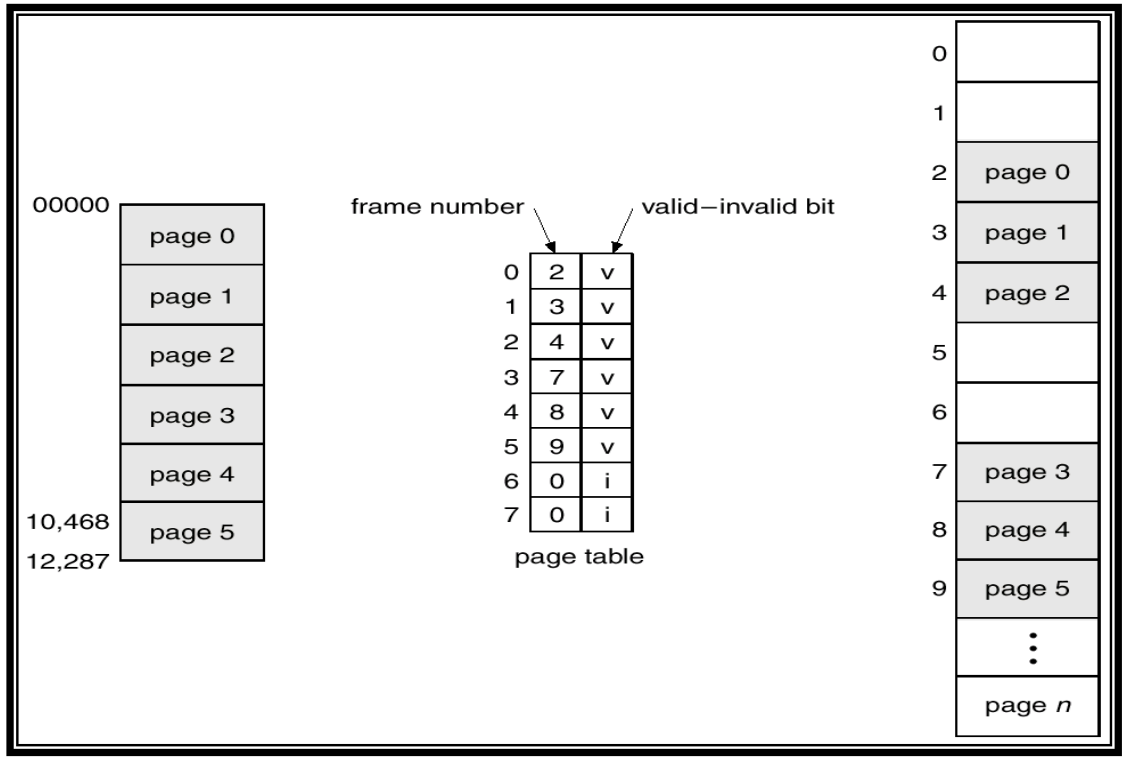
Inverted Page Table Architecture



Protection

- PTLR is used to detect and to abort attempts to access any memory beyond the legal boundaries of a process.
- Memory protection implemented by associating protection bit with each frame. Access bits can be added to PT entries and they can be made transparent to programmers.
- *Valid-invalid* bit attached to each entry in the page table:
  - “valid” indicates that the associated page is in the process’ logical address space, and is thus a legal page.
  - “invalid” indicates that the page is not in the process’ logical address space.

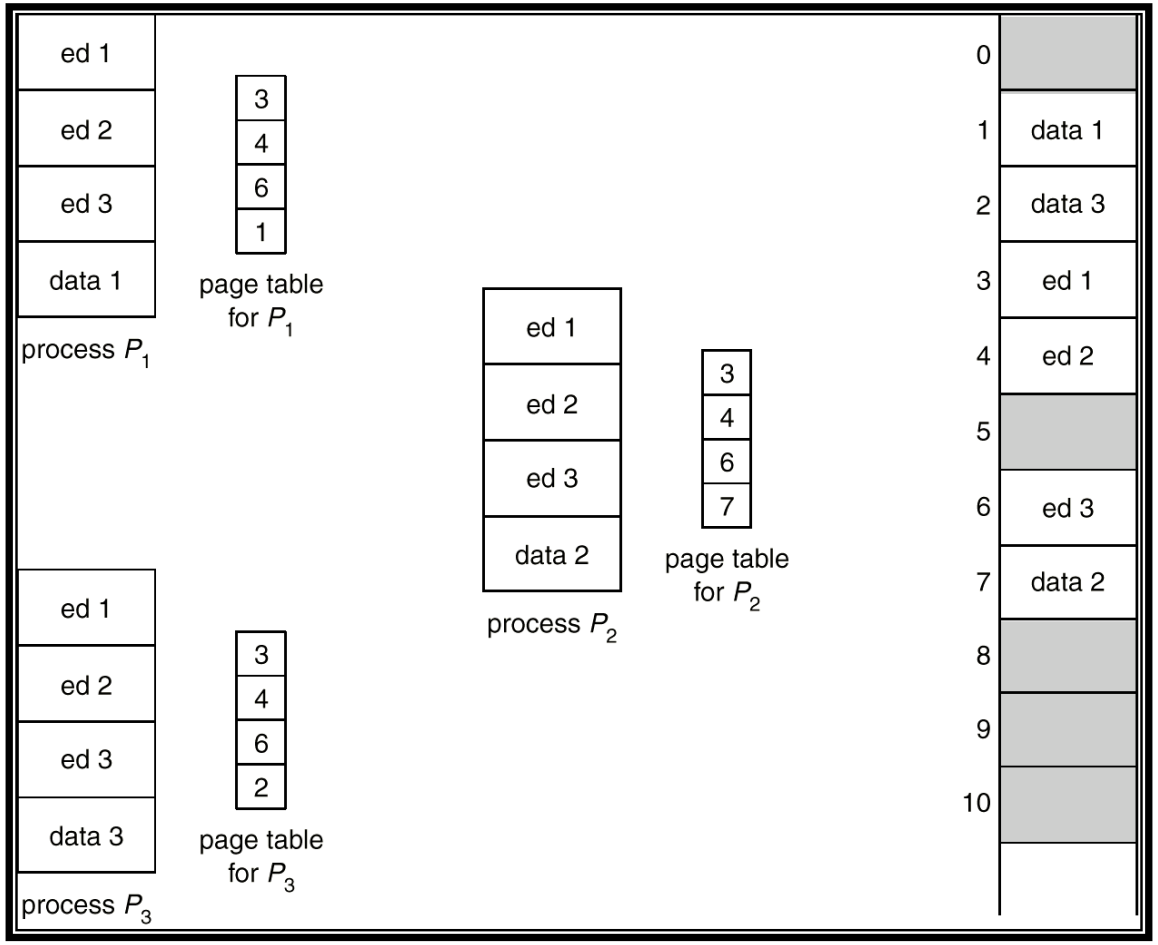
Valid (v) or Invalid (i) Bit In A Page Table



Sharing

- Sharing of pages is straightforward and a single physical copy of a shared page can be easily mapped into as many distinct address spaces as desired.
- Different processes may have different access rights to the shared page.

Shared Pages Example



Advantages:

- Managed entirely by OS and is transparent to programmers
- No compaction is needed thereby it eliminates external fragmentation
- Allocation and deallocation is simple and incurs less overhead
- Memory utilization is very high and is optimal

Disadvantages:

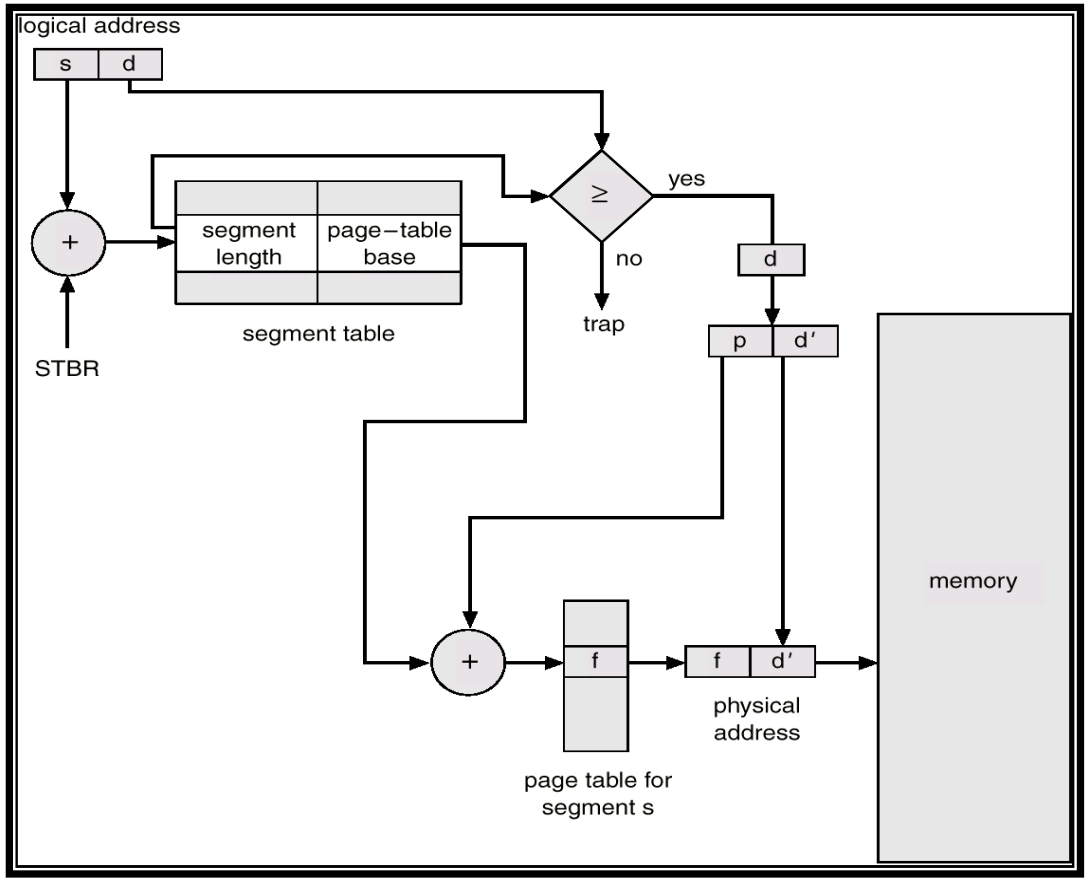
- Storage overhead
- PT per process and MMT lead to page fragmentation
- Table fragmentation is quite large when page size is small
- TLB is expensive and reduces effective memory bandwidth
- Sharing of pages is restrictive

Segmentation with Paging – MULTICS

- The MULTICS system solved problems of external fragmentation and lengthy search times by paging the segments.

- Solution differs from pure segmentation in that the segment-table entry contains not the base address of the segment, but rather the base address of a *page table* for this segment.

MULTICS Address Translation Scheme



Segmentation with Paging – Intel 386 Address Translation

segmentation with paging for memory management with a two-level paging scheme.