* Paging and Segmentation are the non-contiguous memory allocation techniques.
* Paging divides the process into equal size partitions called as pages.
* Segmentation divides the process into unequal size partitions called as segments.

**Segmented Paging-**

|  |
| --- |
| Segmented paging is a scheme that implements the combination of segmentation and paging. |

**Working-**

In segmented paging,

* Process is first divided into segments and then each segment is divided into pages.
* These pages are then stored in the frames of main memory.
* A page table exists for each segment that keeps track of the frames storing the pages of that segment.
* Each page table occupies one frame in the main memory.
* Number of entries in the page table of a segment = Number of pages that segment is divided.
* A segment table exists that keeps track of the frames storing the page tables of segments.
* Number of entries in the segment table of a process = Number of segments that process is divided.
* The base address of the segment table is stored in the segment table base register.

**Translating Logical Address into Physical Address-**

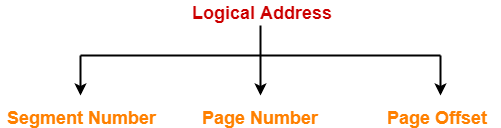
* CPU always generates a logical address.
* A physical address is needed to access the main memory.

Following steps are followed to translate logical address into physical address-

**Step-01:**

CPU generates a logical address consisting of three parts-

1. Segment Number
2. Page Number
3. Page Offset



* Segment Number specifies the specific segment from which CPU wants to reads the data.
* Page Number specifies the specific page of that segment from which CPU wants to read the data.
* Page Offset specifies the specific word on that page that CPU wants to read.

**Step-02:**

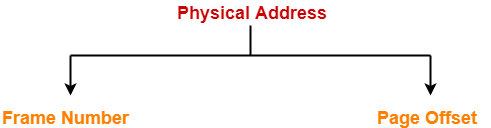
* For the generated segment number, corresponding entry is located in the segment table.
* Segment table provides the frame number of the frame storing the page table of the referred segment.
* The frame containing the page table is located.

**Step-03:**

* For the generated page number, corresponding entry is located in the page table.
* Page table provides the frame number of the frame storing the required page of the referred segment.
* The frame containing the required page is located.

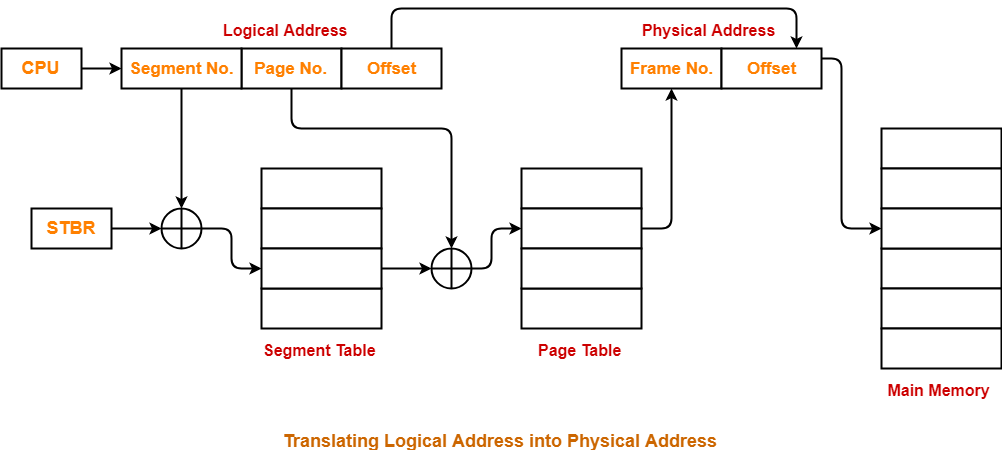
**Step-04:**

* The frame number combined with the page offset forms the required physical address.
* For the generated page offset, corresponding word is located in the page and read.



**Diagram-**

The following diagram illustrates the above steps of translating logical address into physical address-



**Advantages-**

The advantages of segmented paging are-

* Segment table contains only one entry corresponding to each segment.
* It reduces memory usage.
* The size of [**Page Table**](https://www.gatevidyalay.com/page-table-paging-in-operating-system/) is limited by the segment size.
* It solves the problem of external fragmentation.

**Disadvantages-**

The disadvantages of segmented paging are-

* Segmented paging suffers from internal fragmentation.
* The complexity level is much higher as compared to paging.

## ****PRACTICE PROBLEMS BASED ON SEGMENTED PAGING-****

## ****Problem-01:****

A certain computer system has the segmented paging architecture for virtual memory. The memory is byte addressable. Both virtual and physical address spaces contain 216 bytes each. The virtual address space is divided into 8 non-overlapping equal size segments. The memory management unit (MMU) has a hardware segment table, each entry of which contains the physical address of the page table for the segment. Page tables are stored in the main memory and consists of 2 byte page table entries. What is the minimum page size in bytes so that the page table for a segment requires at most one page to store it?

## ****Solution-****

Given-

* Virtual Address Space = Process size = 216 bytes
* Physical Address Space = Main Memory size = 216 bytes
* Process is divided into 8 equal size segments
* Page table entry size = 2 bytes

Let page size = n bytes.

Now, since page table has to be stored into a single page, so we must have-

Size of page table <= Page size

### ****Size of Each Segment-****

Size of each segment

= Process size / Number of segments

= 216 bytes / 8

= 216 bytes / 23

= 213 bytes

= 8 KB

### ****Number of Pages Of Each Segment-****

Number of pages each segment is divided

= Size of segment / Page size

= 8 KB / n bytes

= (8K / n) pages

### ****Size of Each Page Table-****

Size of each page table

= Number of entries in page table x Page table entry size

= Number of pages the segment is divided x 2 bytes

= (8K / n) x 2 bytes

= (16K / n) bytes

### ****Page Size-****

Substituting values in the above condition, we get-

(16K / n) bytes <= n bytes

(16K / n) <= n

n2 >= 16K

n2 >= 214

n >= 27

Thus, minimum page size possible = 27 bytes = 128 bytes.

## ****Problem-02:****

Considering problem-01, give the division of virtual address.

## ****Solution-****

### ****Number of Bits Required For Segment Number-****

Number of segments the process is divided

= 8

= 23

Thus, Number of bits required to identify a particular segment in segment table = 3 bits.

### ****Number of Bits Required For Page Number-****

Number of pages a segment is divided

= Segment size / Page size

= 8KB / 128 bytes

= 213 bytes / 27 bytes

= 26 pages

Thus, Number of bits required to identify a particular page in table = 6 bits.

### ****Number of Bits Required For Page Offset-****

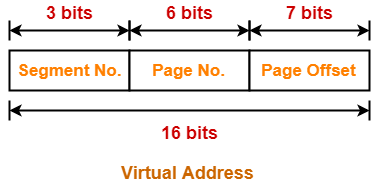
Page size

= 128 bytes

= 27 bytes

Thus, Number of bits required for page offset = 7 bits.

Thus, virtual address is divided as-



## ****Problem-03:****

A certain computer system has the segmented paging architecture for virtual memory. The memory is byte addressable. Both virtual and physical address spaces contain 216 bytes each. The virtual address space is divided into 8 non-overlapping equal size segments. The memory management unit (MMU) has a hardware segment table, each entry of which contains the physical address of the page table for the segment. Page tables are stored in the main memory and consists of 2 byte page table entries.. Assume that each page table entry contains (besides other information) 1 valid bit, 3 bits for page protection and 1 dirty bit. How many bits are available in page table entry for storing the aging information for the page? Assume that page size is 512 bytes.

## ****Solution-****

Given-

* Virtual Address Space = Process size = 216 bytes
* Physical Address Space = Main Memory size = 216 bytes
* Process is divided into 8 equal size segments
* Page table entry size = 2 bytes = 16 bits
* Page table entry besides other information contains 1 valid bit, 3 protection bits, 1 dirty bit
* Page size = 512 bytes

### ****Number of Frames in Main Memory-****

Number of frames in main memory

= Size of main memory / Page size

= 216 bytes / 512 bytes

= 216 bytes / 29 bytes

= 27 frames

Thus, Number of bits required for frame identification in page table entry = 7 bits

### ****Number Of Bits Available For Storing Aging Information-****

Number of bits available for storing aging information

= Number of bits in page table entry – ( Number of bits required for frame identification + 1 valid bit + 3 protection bits + 1 dirty bit)

= 16 bits – (7 + 1 + 3 + 1) bits

= 16 bits – 12 bits

= 4 bits