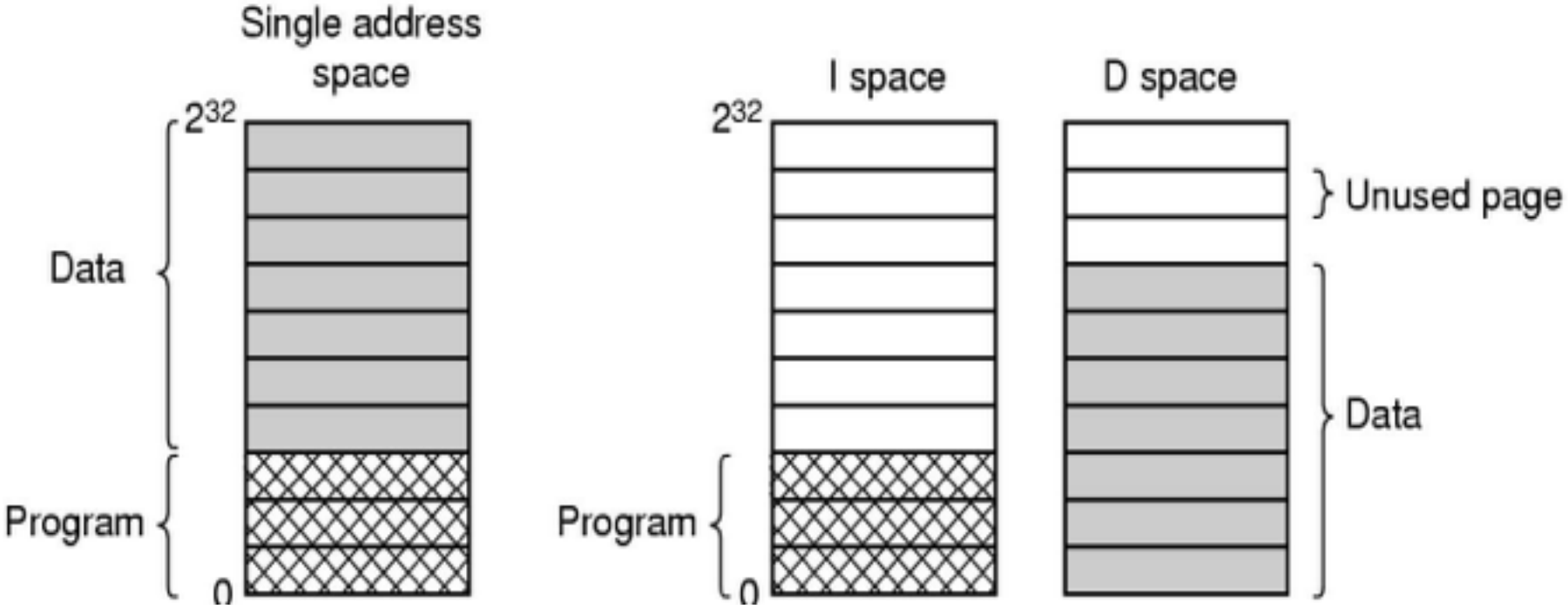
# Memory Management - Segmentation

Kartik Gopalan

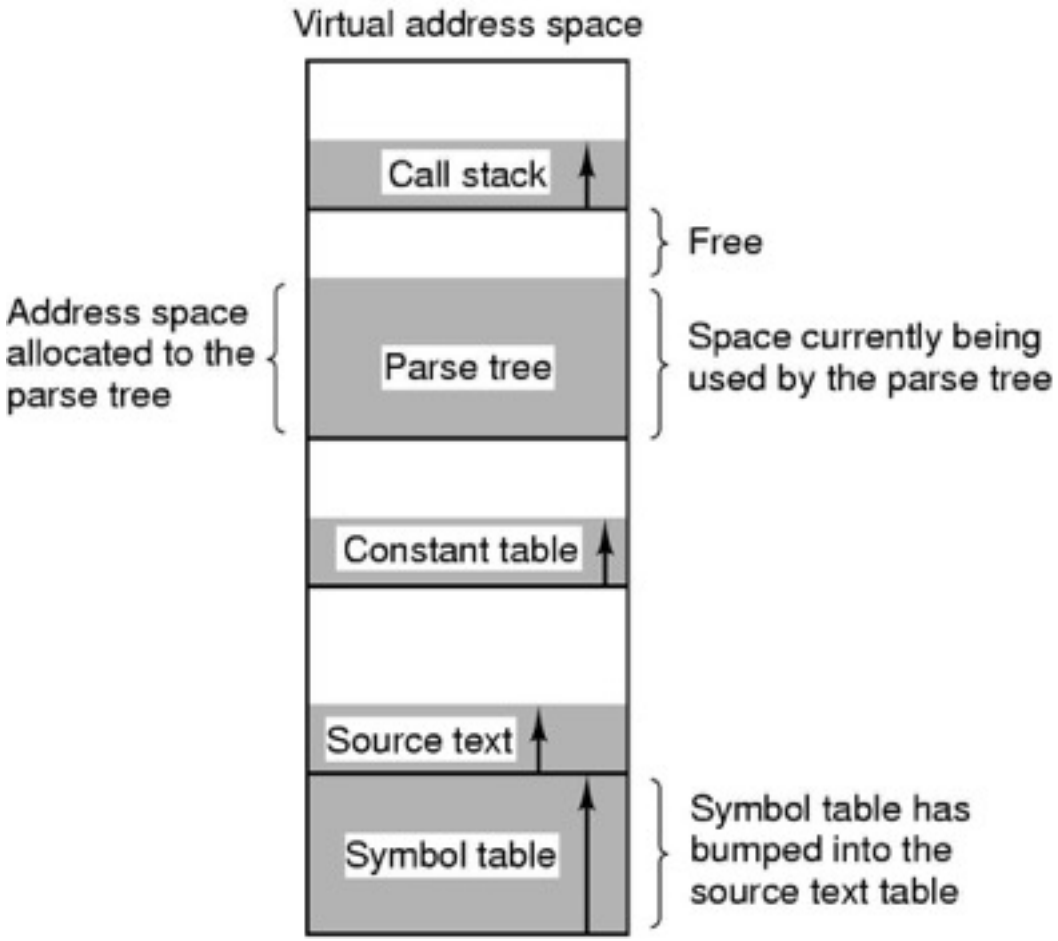
Chapter 3 Tanenbaum's book

# Separate Instruction and Data Spaces



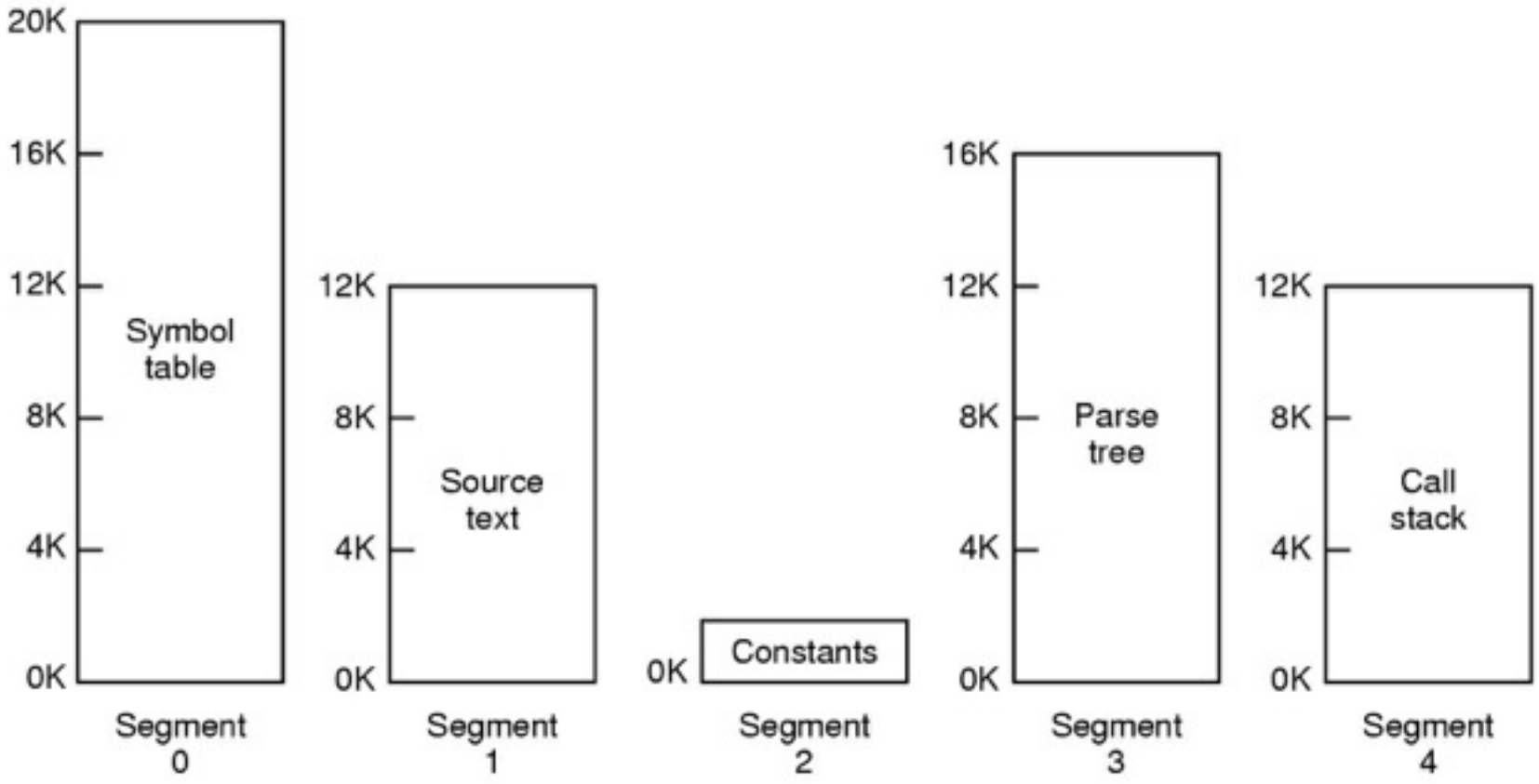

- One address space
- Separate I and D spaces

# Example: Compiler program without segmentation




- One-dimensional address space with growing tables

# Example: Compiler program with segmentation

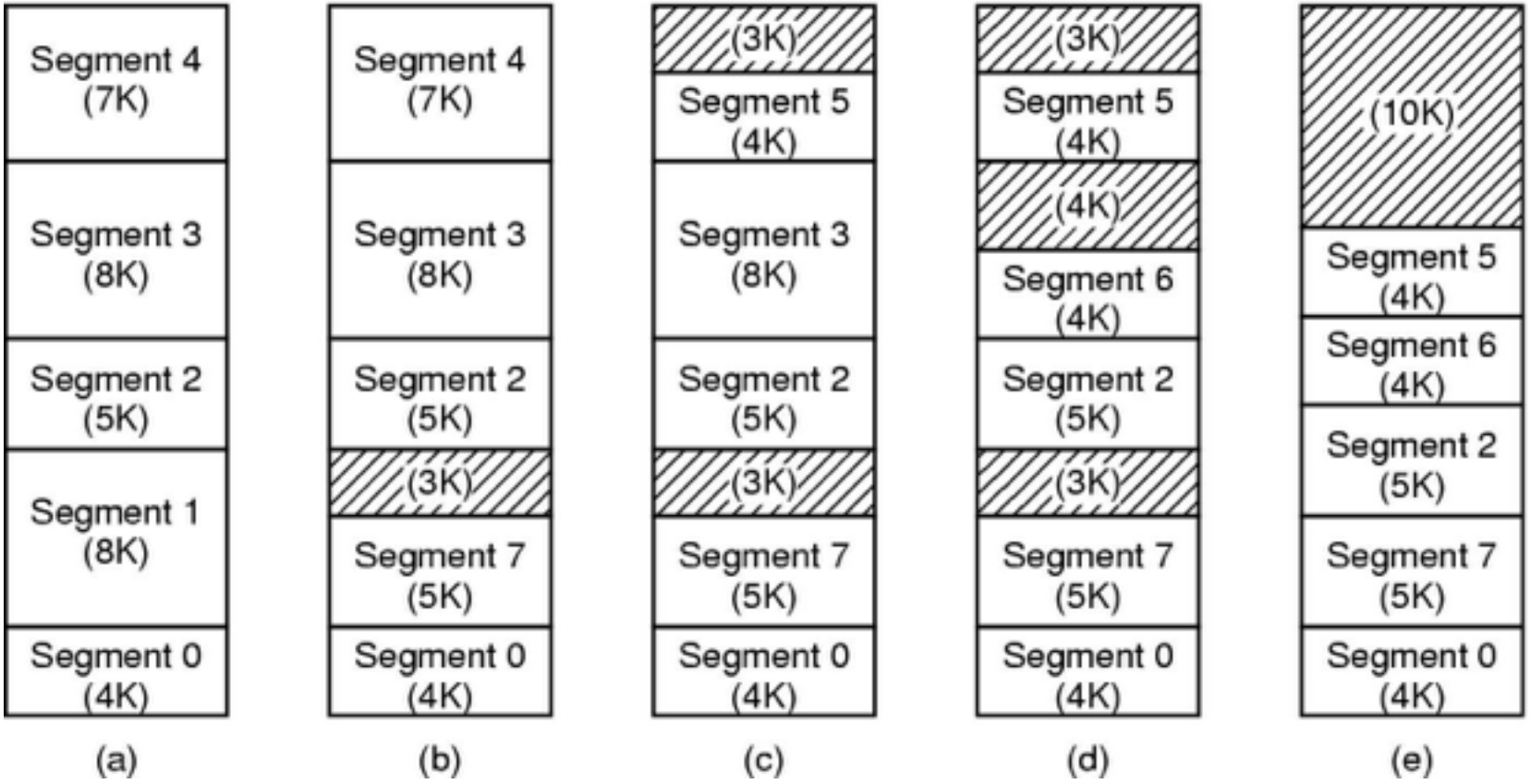



- One process has multiple address spaces.
- Each address space grows or shrink, independently

# Comparison of paging and segmentation

| Consideration | Paging | Segmentation |
|---|---|---|
| Need the programmer be aware that this technique is being used? | No | Yes |
| How many linear address spaces are there? | 1 | Many |
| Can the total address space exceed the size of physical memory? | Yes | Yes |
| Can procedures and data be distinguished and separately protected? | No | Yes |
| Can tables whose size fluctuates be accommodated easily? | No | Yes |
| Is sharing of procedures between users facilitated? | No | Yes |
| Why was this technique invented? | To get a large linear address space without having to buy more physical memory | To allow programs and data to be broken up into logically independent address spaces and to aid sharing and protection |

# Implementation of Pure Segmentation

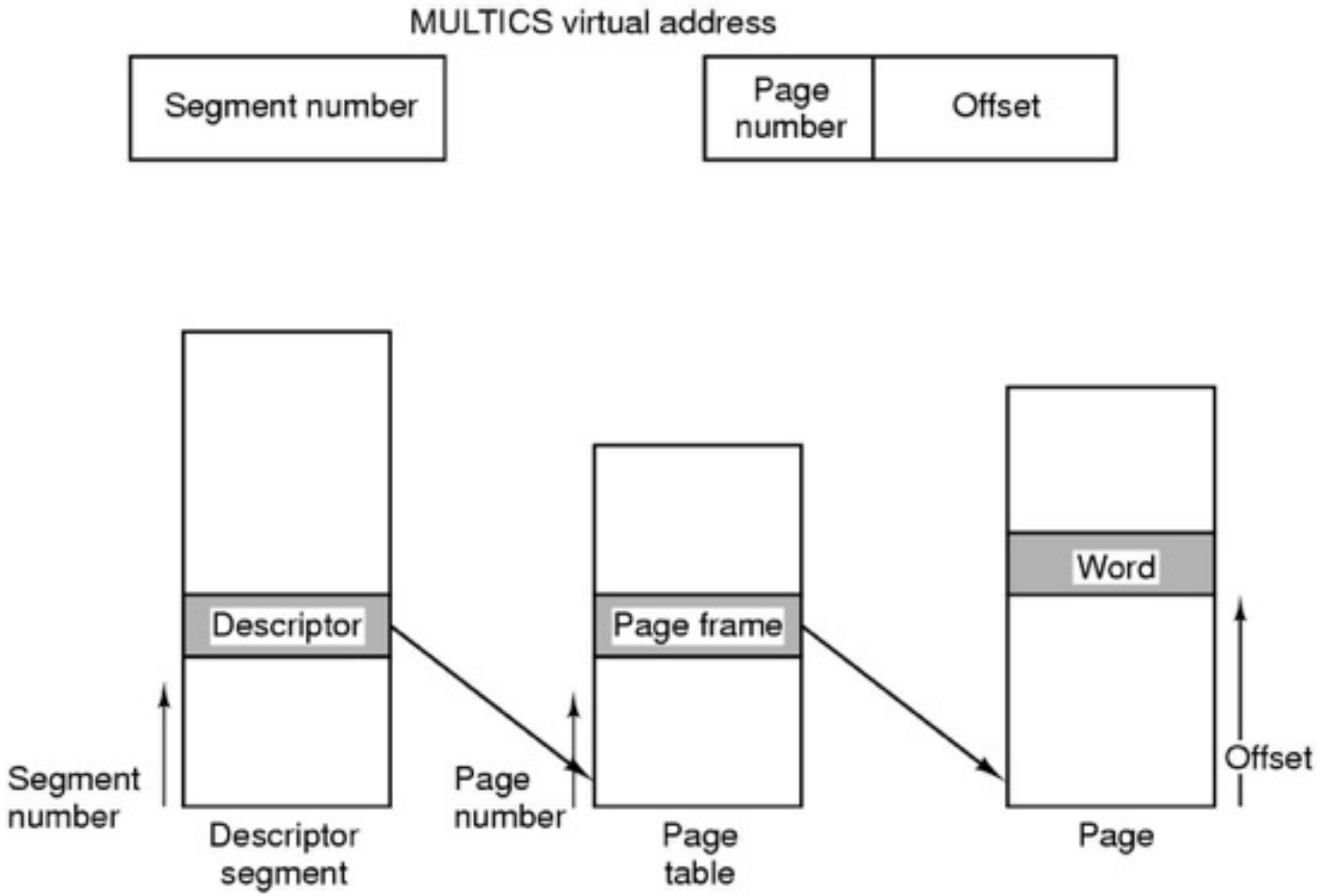



(a)-(d) Development of checkerboarding (external fragmentation)
(e) Removal of the checkerboarding by compaction

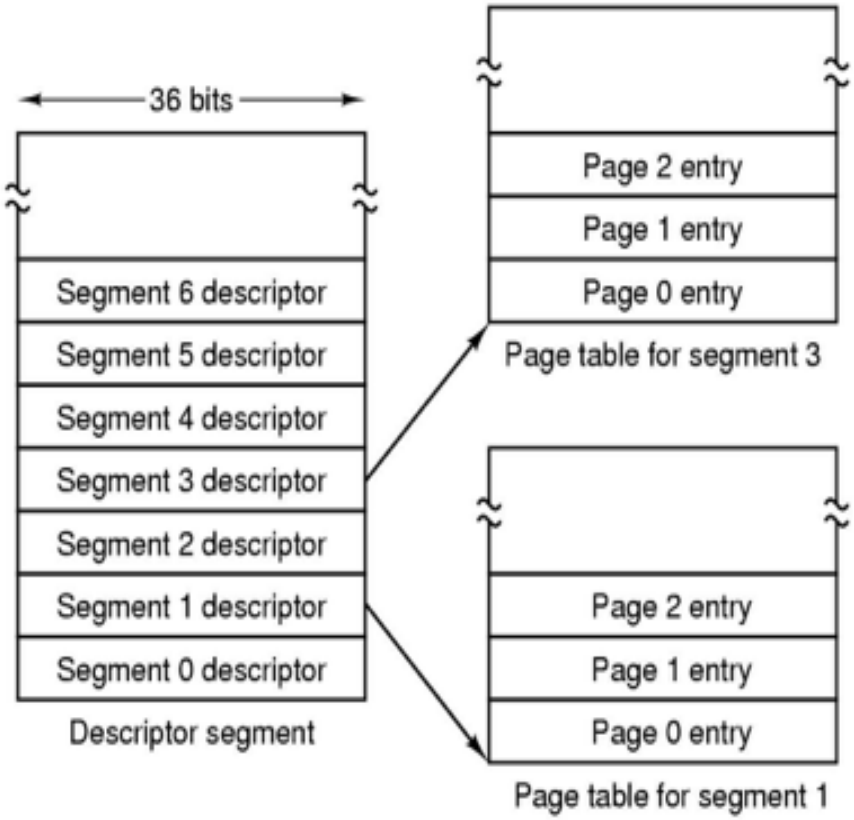
# MULTICS — Paged Segmentation

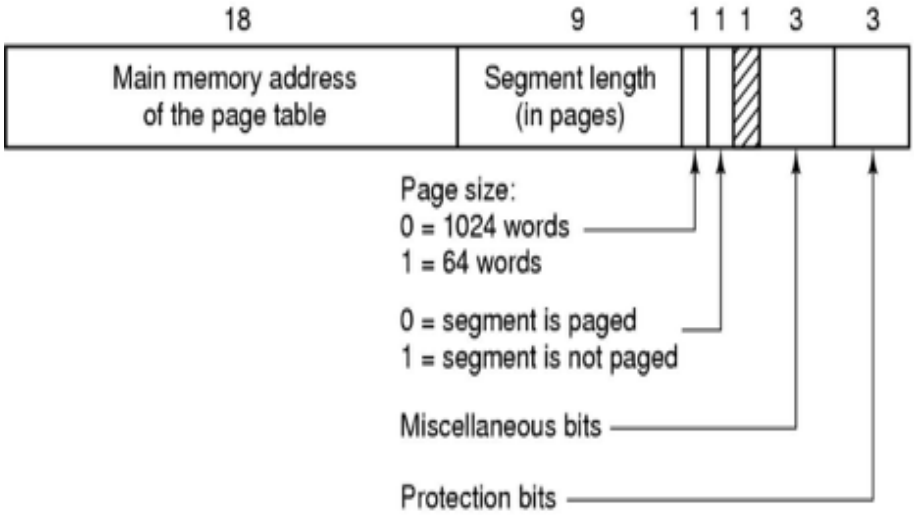
- Every process can have multiple virtual address spaces (or segments)
- Each Segment has its own page table
- Advantage
  - Each segment can have the full virtual address space allowed by number of address bits
- Disadvantage
  - Switching from one segment to another has a high context switch penalty, even within the same process.

# Translation of a Multics Virtual Address



# Segmentation with Paging: MULTICS

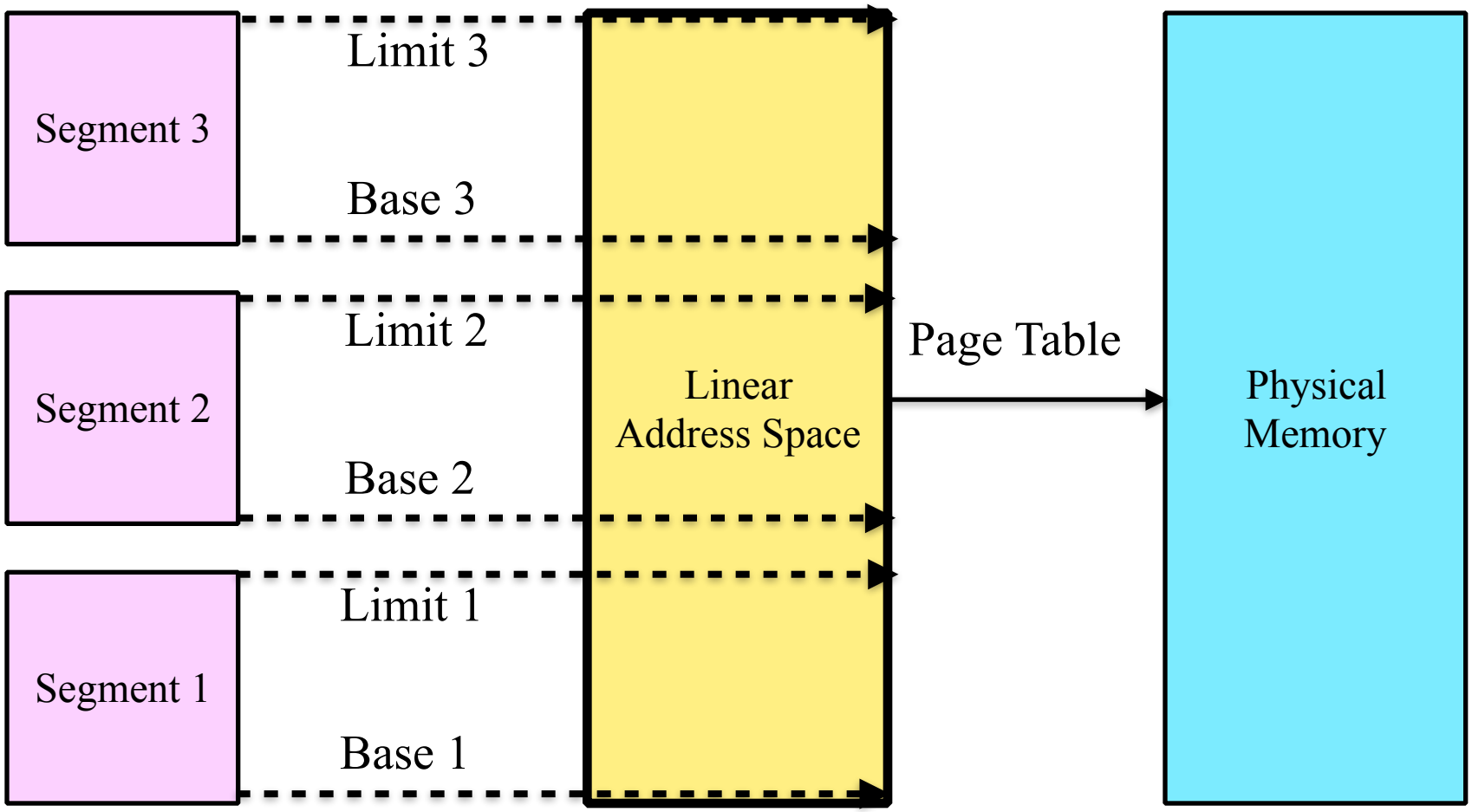


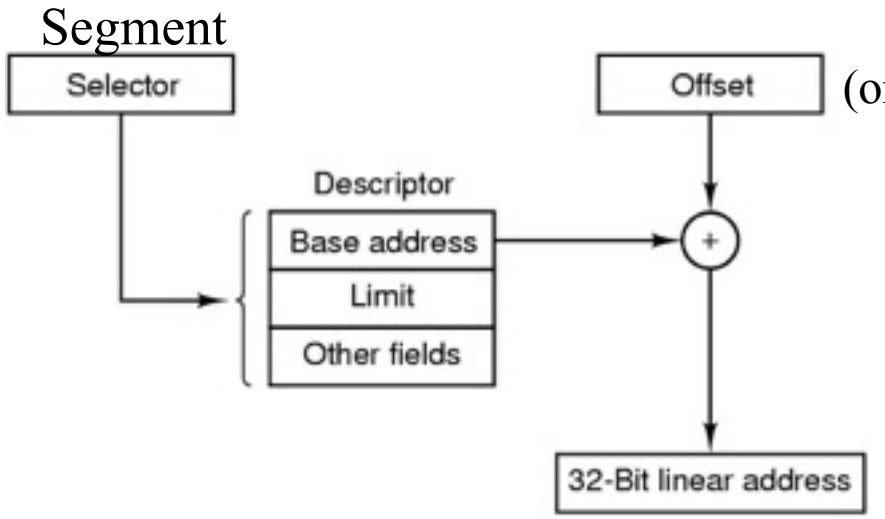

- Segment descriptor table has one descriptor for each segment
- Each segment descriptor points to a page table

# Pentium — Paged Segmentation

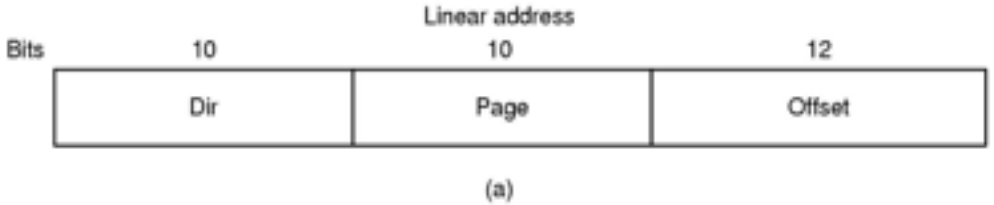
- Each process can have multiple segments
- Multiple segments map to one linear address space
- Linear address space has one page table



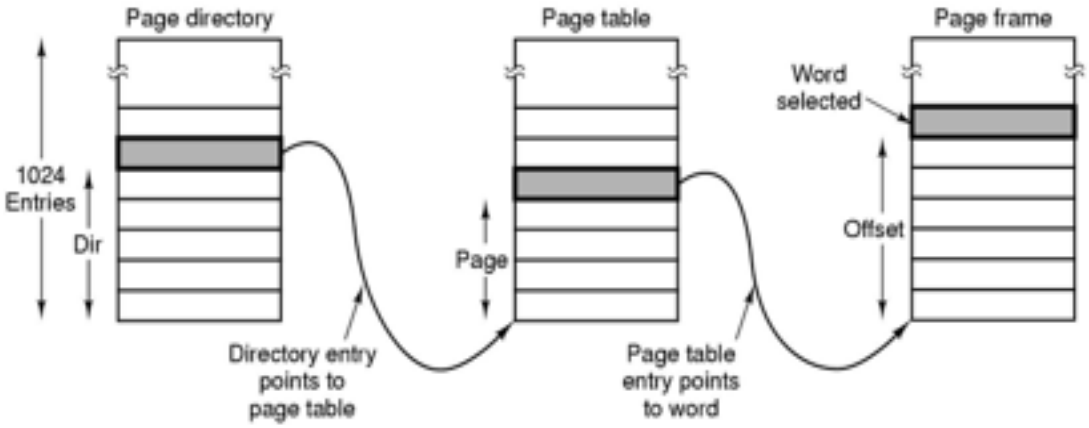
# Translation of a Pentium Virtual Address




Step 1: Convert (selector, offset) pair to a linear address using segment descriptor




(a)

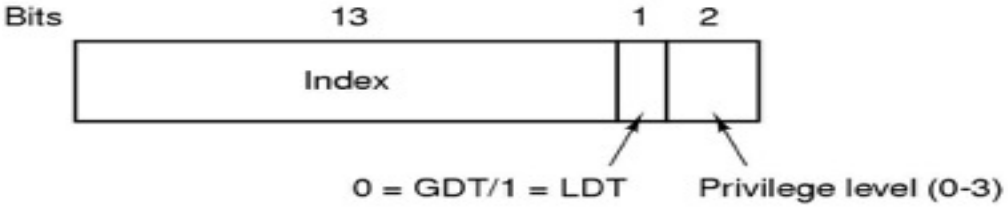



(b)

Step 2: Convert linear address onto a physical address using page table entry

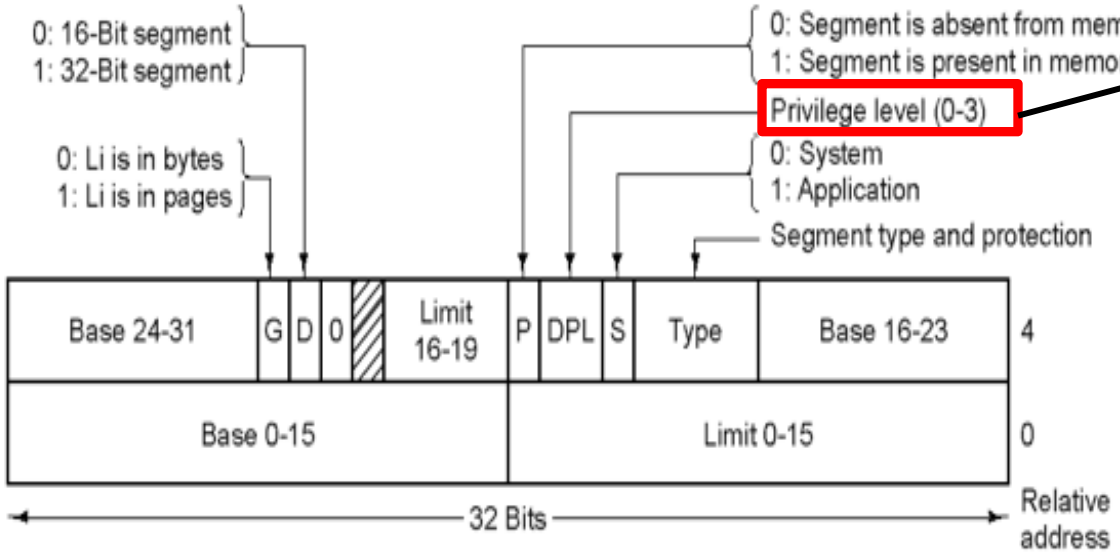
# Segmentation with Paging: Pentium

Similar to MULTICS, but addresses a number of different design goals




## A Pentium selector

(equivalent to Segment Number in Multics)
GDT = Global Descriptor Table
LDT = Local Descriptor Table




Four privilege levels in x86

- Code segment descriptor
  - Data segment descriptors are slightly different

# References


- Chapter 3: Modern Operating Systems, Andrew S. Tanenbaum
- Segmentation
  - http://en.wikipedia.org/wiki/Memory_segment
- x86
  - http://en.wikipedia.org/wiki/X86
- Intel Memory model
  - http://en.wikipedia.org/wiki/Intel_Memory_Model