Segmentation

Module 14

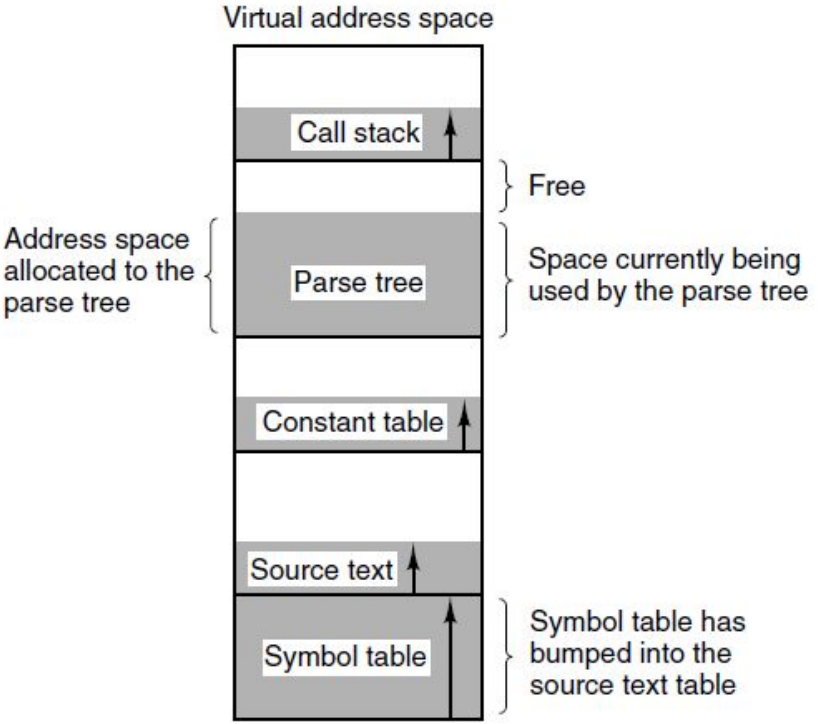
Pages vs. Segments

- Page sizes are not arbitrary - but they are not dictated by programmer requirements
  - Compilers will often work hard to put common things on the same page - but they can only do so much!
  - Where pages are a **machine-driven** construct - segments refer to “logical” chunks of related addresses

Segments

Notice that the starting points of each of the groups must be set by the **compiler.**

What if call stack becomes too large?

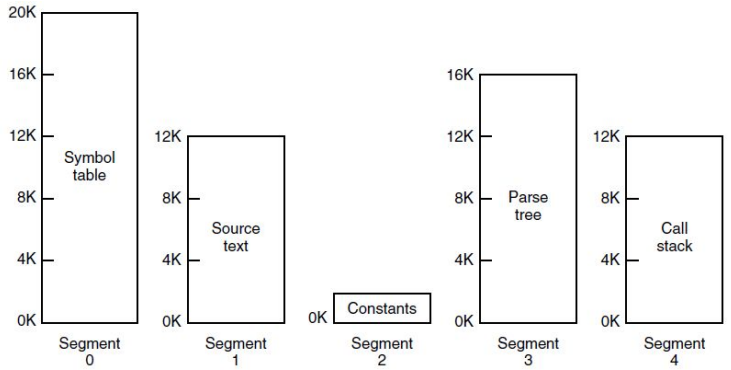


Segments = Address Spaces

- Segmentation breaks up a process into several address spaces:
  - Each space has 0-max addresses, where max is set by hardware.
  - Each process can have a maximum number of segments
  - In pure segmentation, each segment is allocated contiguously

Segments

- Segment size is NOT fixed - can grow.
- Segments will be managed with a segmentation table (base + limit)
- In practice, we still have higher order bits indicate segment #,
- lower order bits indicate offset



Pure Segmentation… same problems!

Segmentation allows for easier compilation, linking, and maximizes the advantages of locality.

Of course… if we allocate segments contiguously… we have our friend external fragmentation to worry about again!

Paging to the rescue again!

Example: MULTICS

Very influential OS (1965)

- Addresses were **34-bit**.
  - High order 18 bits represented segment #
  - … so 2^{18} segments per process was possible
  - Low order 16 bits were for offset within segment
    - Segments were 2^{16} large

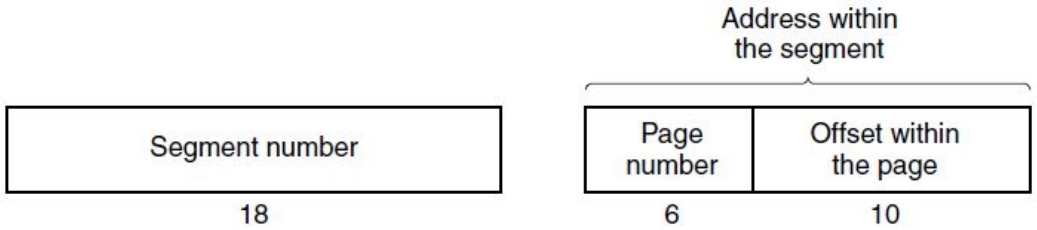
Ah… the segmentation fault!

When presented with a 34-bit address, a segmentation table lookup

If a particular segment was not in memory, a trap would occur… this term has **been reused**

Segments to Pages

The segment descriptor told the system where to find the *page table* for the segment



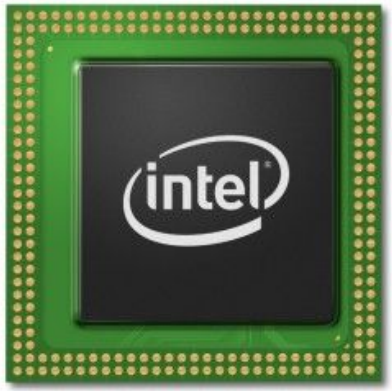
From there - things proceeded as normal - each page was of uniform size.

The Intel x86

- All 32-bit Intel x86 processes also supported segmentation
  - Segments were larger: 32-bits (1 billion+)
  - But had fewer segments: 16K

Ultimately this has become obsolete, and is not part of the x86-64 bit architecture.

Scope



- Conceptually - segmentation is fairly simple
- The complexity of segmentation is in its hardware implementation
  - It’s complicated because of the indirection needed to maintain addressability
- This is an OS course, not a hardware course - you’ll be expected to understand the concept - not implementation