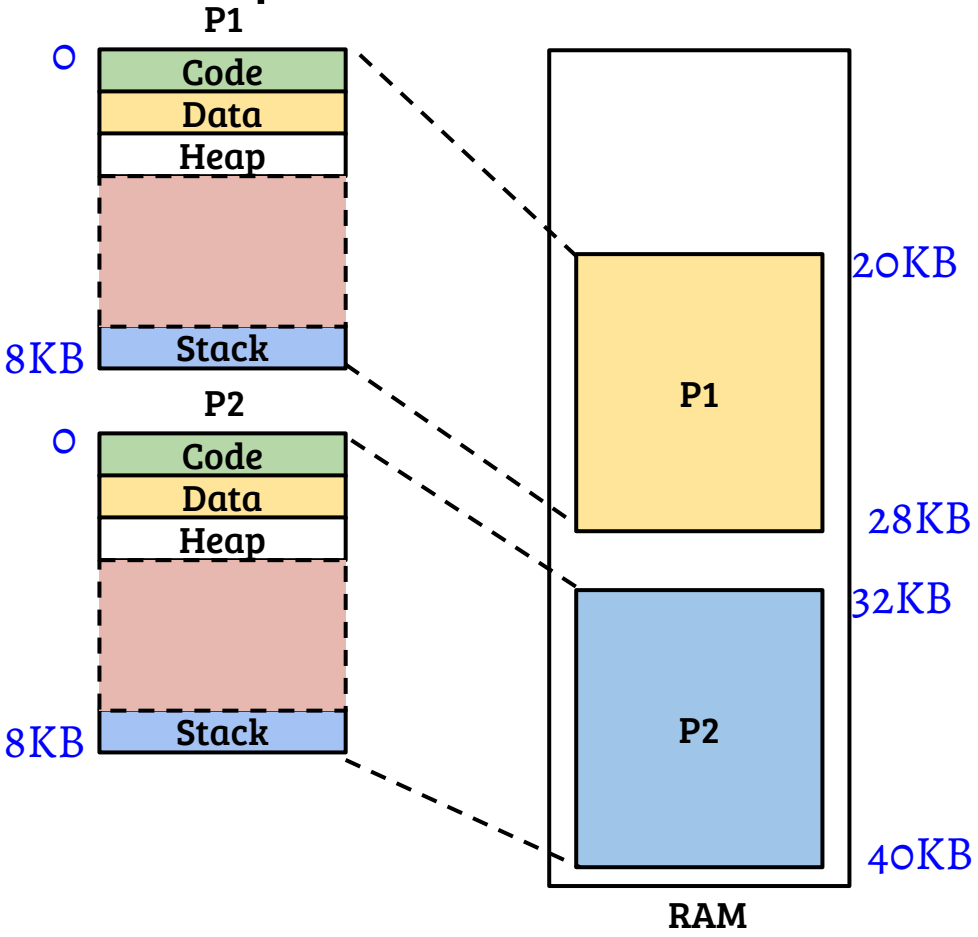
CS330: Operating Systems

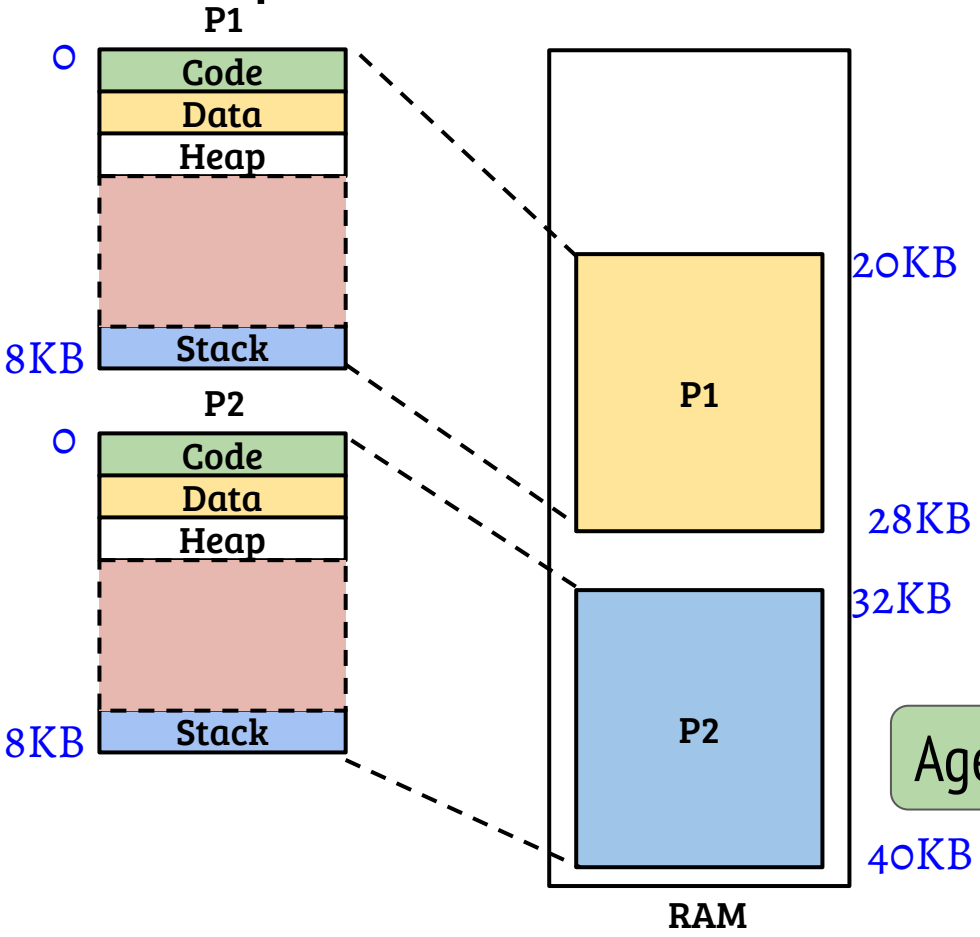
Virtual memory: Segmentation

Recap: Translation at address space granularity



- Physical memory of same size as the address space size is allocated to each process
- Issues: Memory inefficient, inflexible

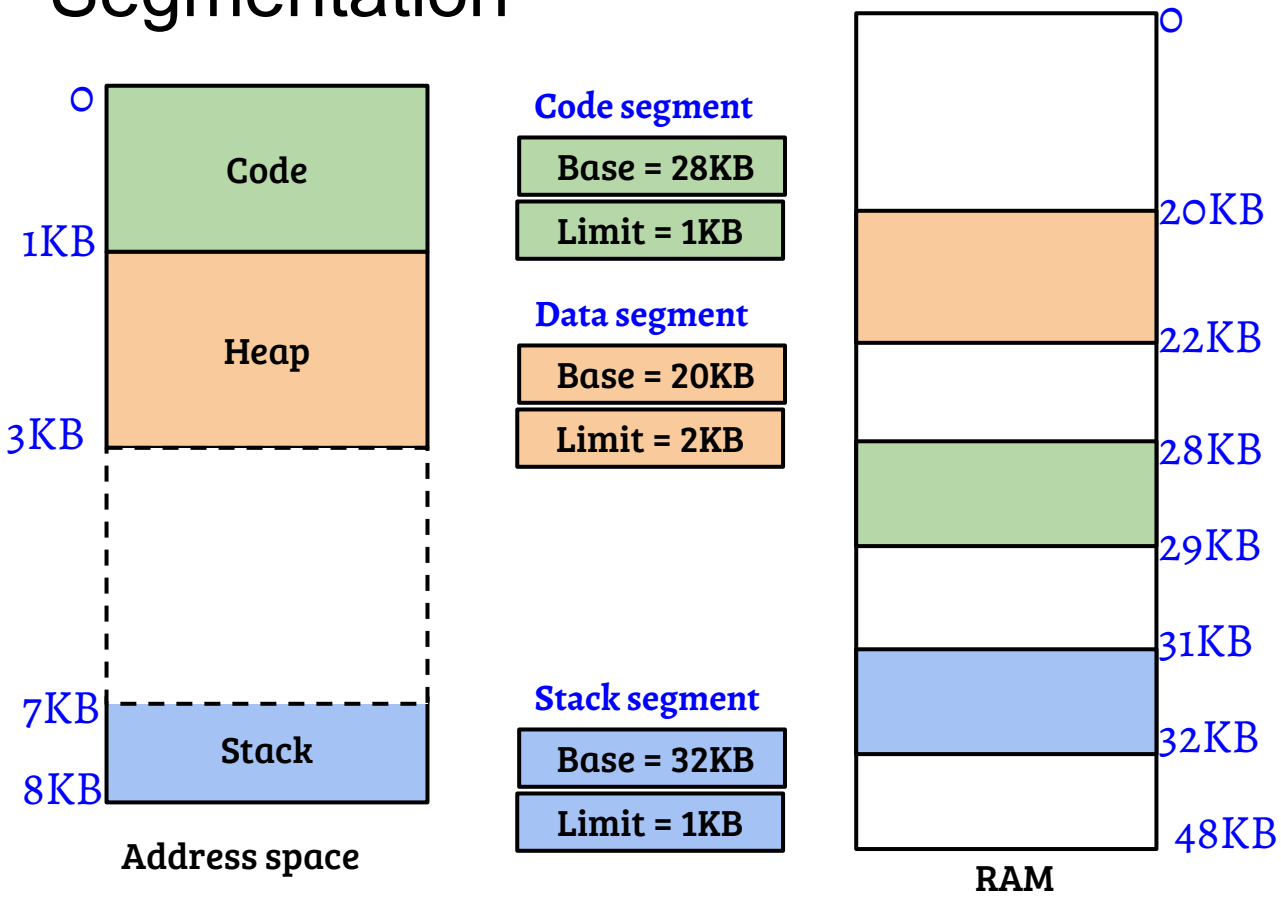
Recap: Translation at address space granularity



- Physical memory of same size as the address space size is allocated to each process
- Issues: Memory inefficient, inflexible

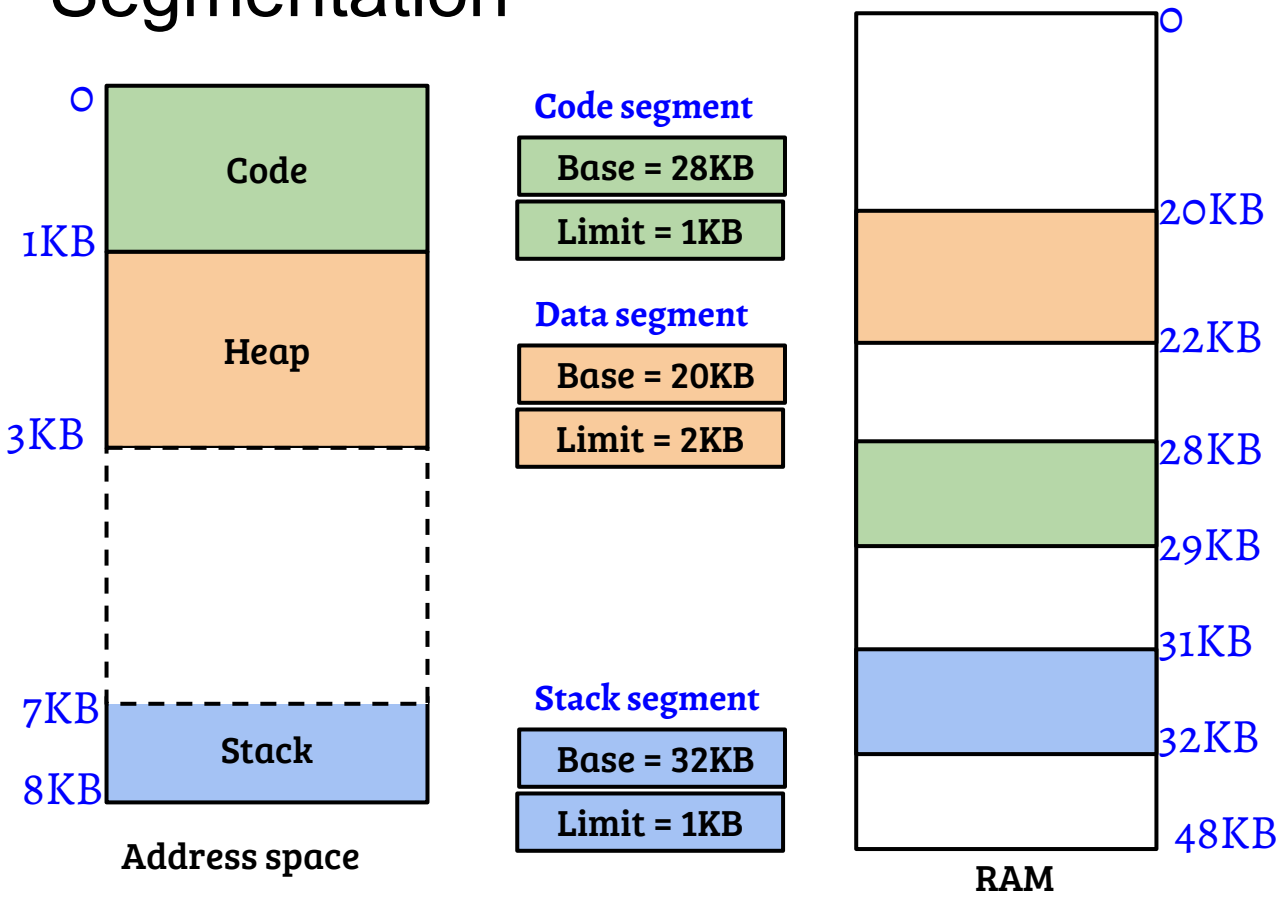
Agenda: Translation at segment granularity

Segmentation



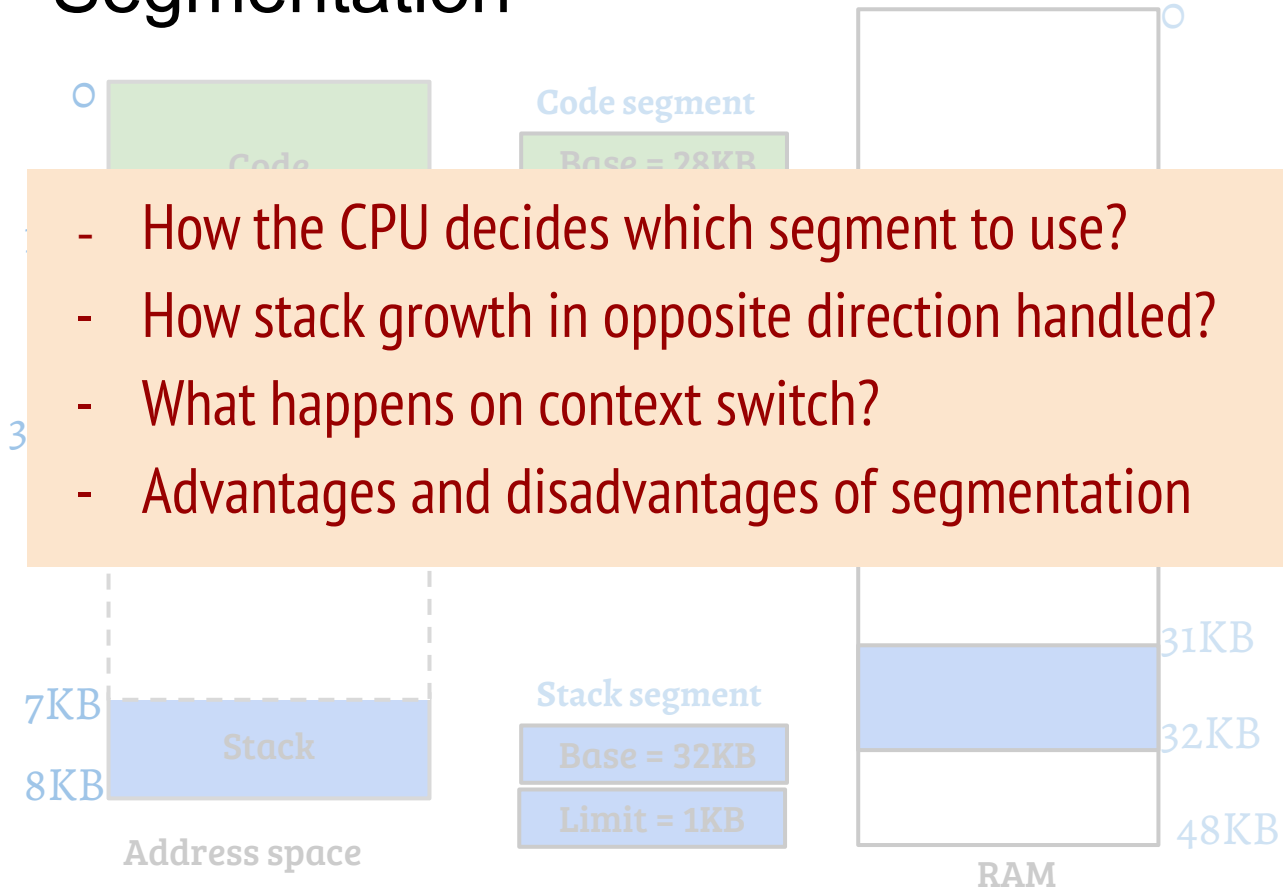
- Extension of the basic scheme with more base-limit register pairs

Segmentation



- Example
  - Code address
  - Data address

Segmentation



- How the CPU decides which segment to use?
- How stack growth in opposite direction handled?
- What happens on context switch?
- Advantages and disadvantages of segmentation

Segmentation: Explicit addressing

- Part of the address is used to explicitly specify segments
- In our example,
  - virtual address space = 8KB, address length = 13 bits and there are three segments
  - Two MSB bits used to specify the segment: “00” for code, “01” for data and “11” for stack
  - The hardware selects the segment register based on the value of two MSB bits and rest of the bits are used as the offset
  - Max. size of each segment = 2KB

Issues with explicit addressing

- Inflexible
  - Data and stack can not be sized dynamically
- Wastage of virtual address space
  - In our example, 2KB virtual address is unusable
- Note: Physical allocation is still done in an on-demand basis

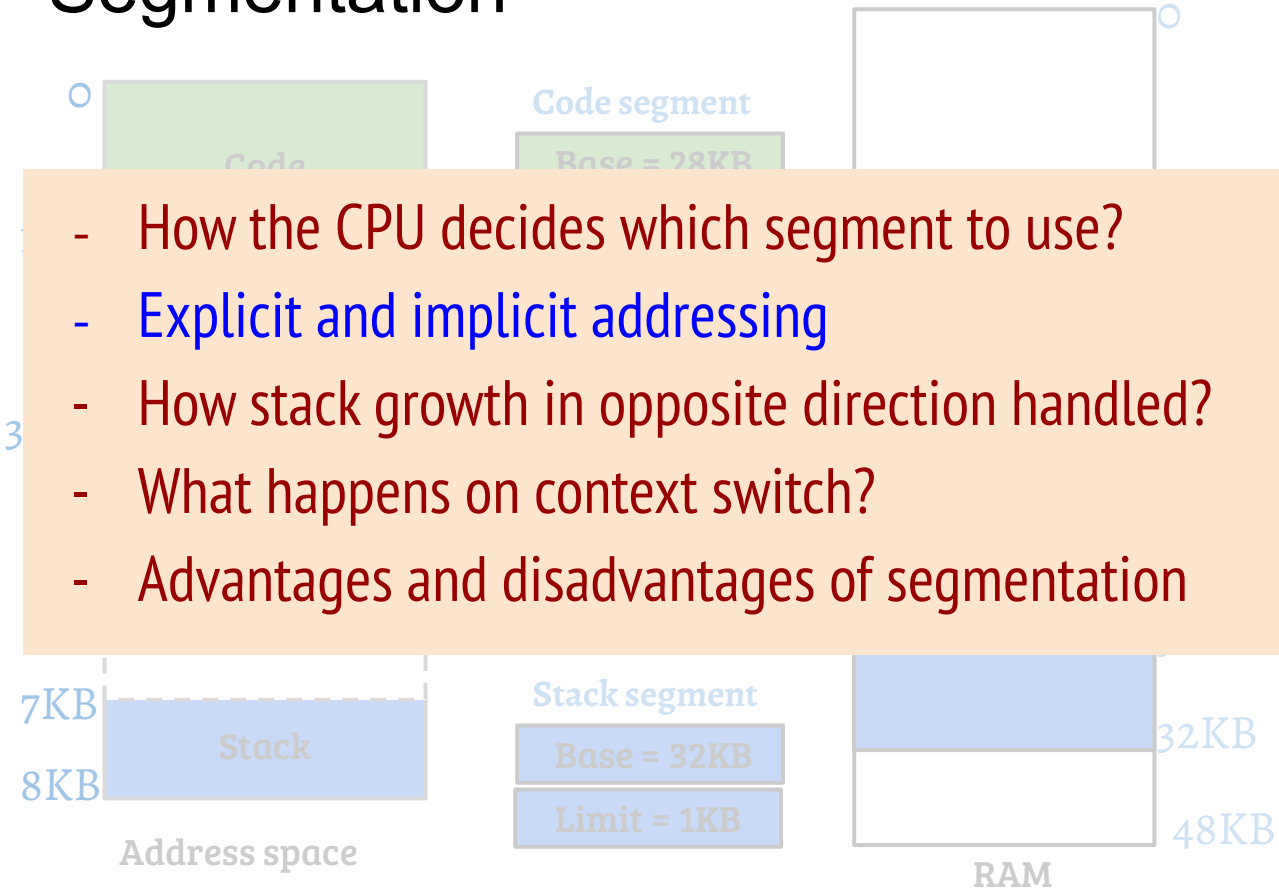
Segmentation: Implicit addressing

- The hardware selects the segment register based on the operation
- Code segment for instruction access
  - Fetch address, jump target, call address

Segmentation: Implicit addressing

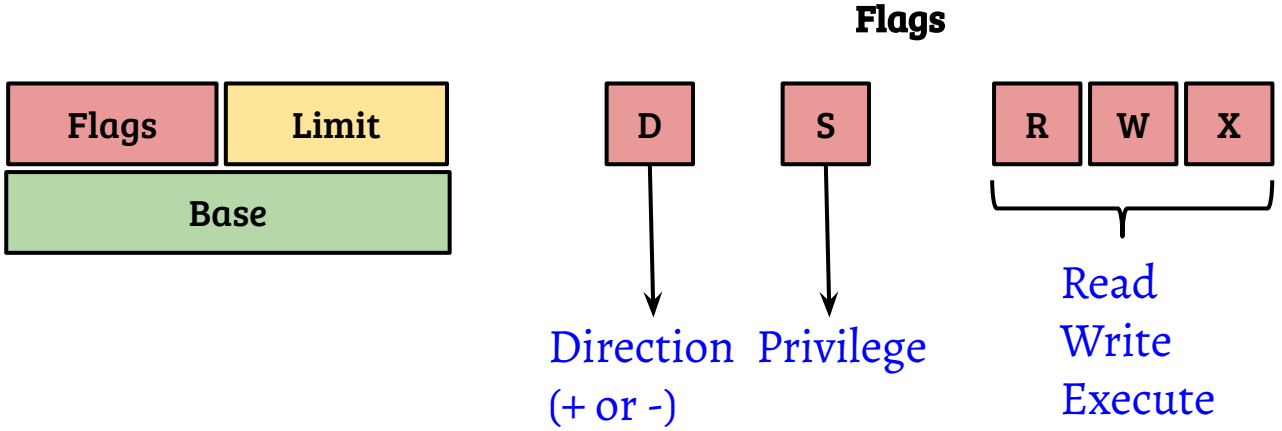
- The hardware selects the segment register based on the operation
- Code segment for instruction access
  - Fetch address, jump target, call address
- Stack segment for stack operations
  - Arguments for push and pop, indirect addressing with SP, BP
- Data segment for other addresses

Segmentation



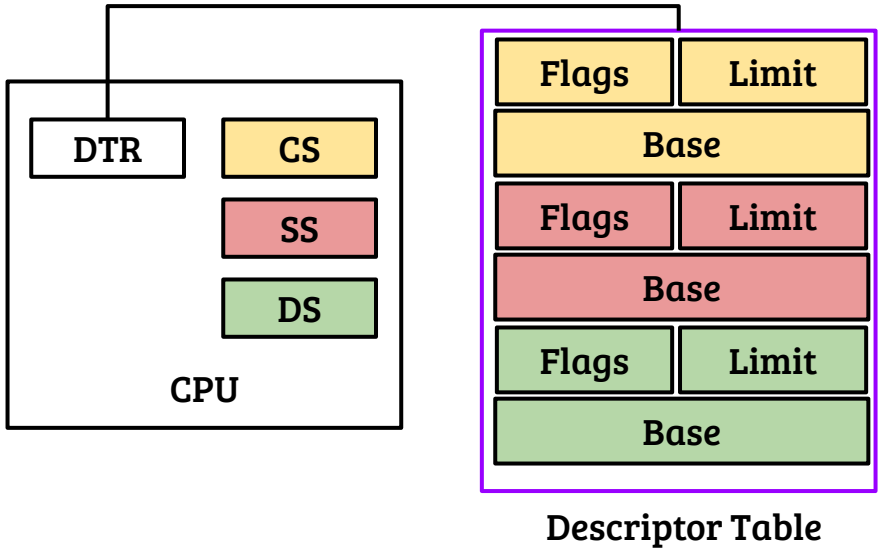
- How the CPU decides which segment to use?
- Explicit and implicit addressing
- How stack growth in opposite direction handled?
- What happens on context switch?
- Advantages and disadvantages of segmentation

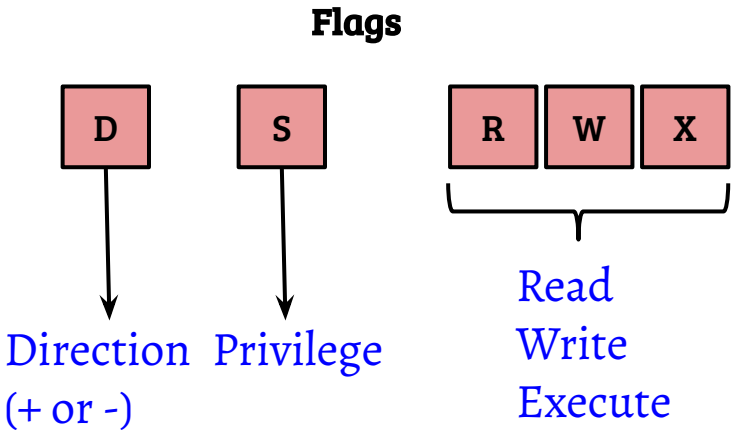
Segmentation (protection and direction)



- For stack, direction is -ve, used by hardware to calculate physical address
- “S” bit can be used to specify privilege, specifically useful in code segment
- R, W and X can be used to enforce isolation and sharing

Segmentation in reality

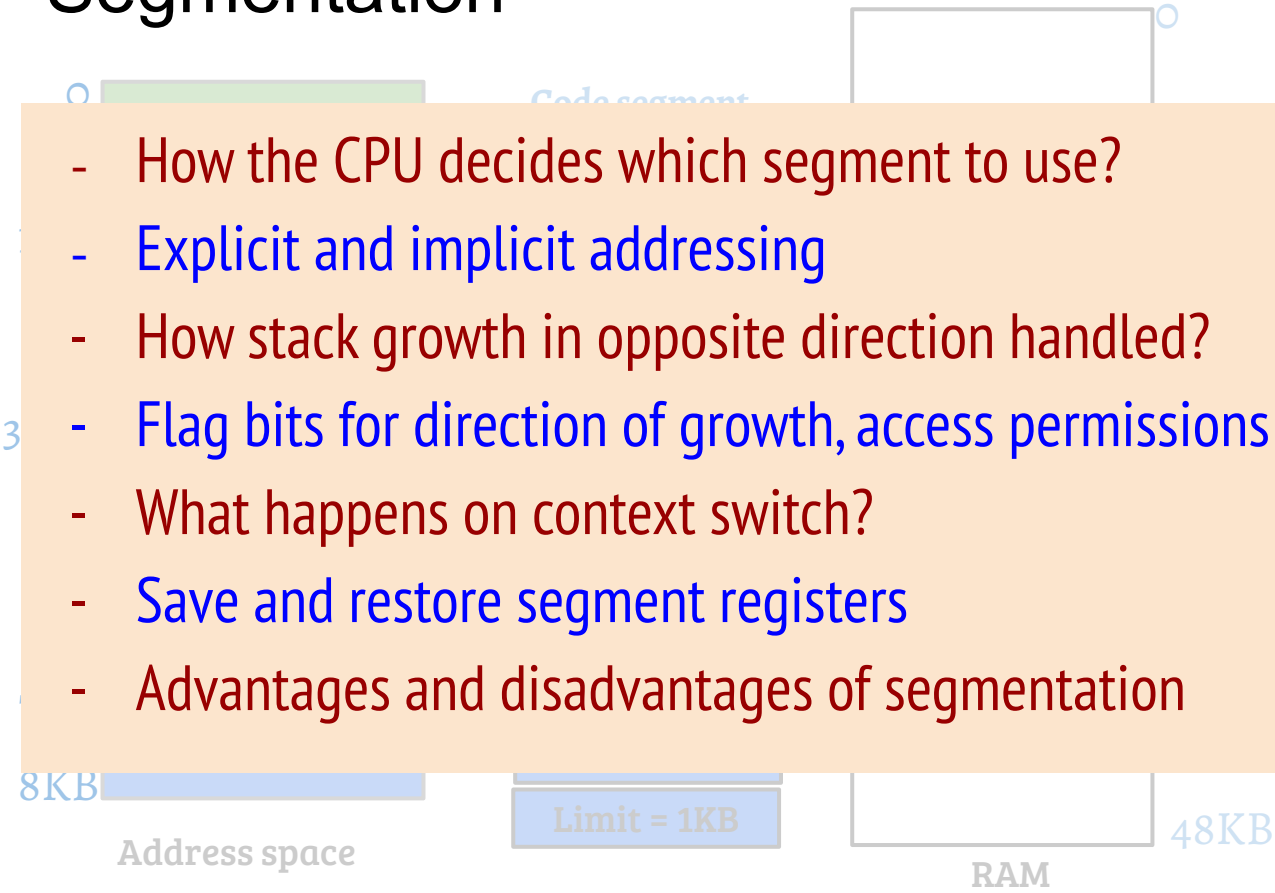




- Descriptor table register (DTR) is used to access the descriptor table
- # of descriptors depends on architecture
- Separate descriptors used for user and kernel mode

Segmentation

- How the CPU decides which segment to use?
- Explicit and implicit addressing
- How stack growth in opposite direction handled?
- Flag bits for direction of growth, access permissions
- What happens on context switch?
- Save and restore segment registers
- Advantages and disadvantages of segmentation



Advantages and disadvantages of segmentation

- Advantages
  - Easy and efficient address translation
  - Save memory wastage for unused addresses
- Disadvantages
  - External fragmentation
  - Can not support discontiguous sparse mapping