Announcements

- Project #3
  - Is out
  - Deadline is before midterm #2 (start early)

Project #3

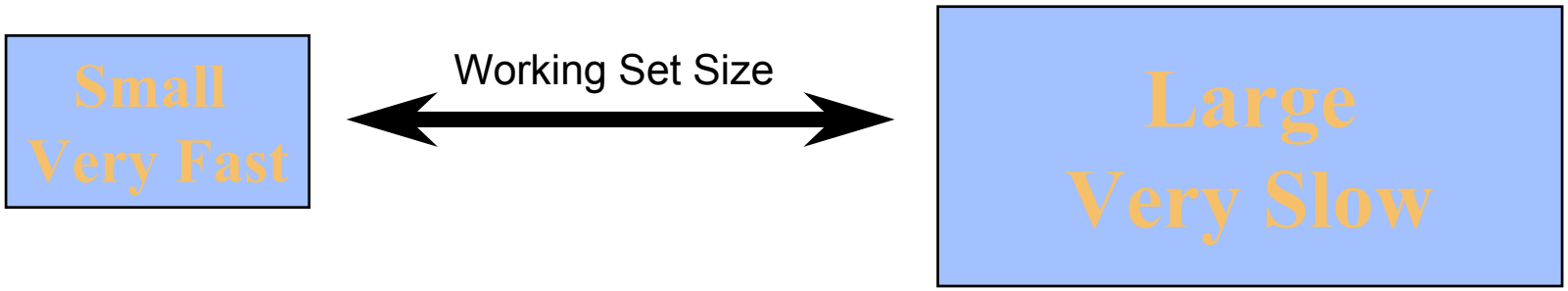
- What is pageable?
  - User memory including text, data, and stack
- Memory model
  - Kernel memory in low memory
  - User memory in high memory
- Paging Bits
  - cr3 – Page Table Base Register (PTBR)
  - cr0:31 – Enable Paging bit
  - cr2 – Address causing page fault
- Page Faults
  - Look in errorCode fields of interrupt

Steps of Project

- Enable Paging
  - Map all of physical memory
- Get separate page table for User Process
  - Map user pages at 2GB
  - Update Segment Info
  - Context switch PTBR
- Get page faults working

Working Sets and Page Replacement

- Programs usually display reference locality
  - temporal locality
    - repeated access to the same memory location
  - spatial locality
    - consecutive memory locations access nearby memory locations
  - memory hierarchy design relies heavily on locality reference
    - sequence of nested storage media
- Working set
  - set of pages referenced in the last delta references



Preventing Threashing

- Need to ensure that we can keep the working set in memory
  - if the working sets of the processes in memory exceed total page frames, then we need to swap a process out
- How do we compute the working set?
  - can approximate it using a reference bit

Implementation Issues

- How big should a page be?
  - want to trade cost of fault vs. fragmentation
    - cost of fault is: trap + seek + latency + transfer
  - Does the OS page size have to equal the HW page size?
    - no, just needs to be a multiple of it
- How does I/O relate to paging
  - if we request I/O for a process, need to lock the page
    - if not, the I/O device can overwrite the page
- Can the kernel be paged?
  - most of it can be.
  - what about the code for the page fault handler?

Segmentation

- Segmentation is used to give each program several independent protected address spaces
  - each segment is an independent protected address space
  - access to segments is controlled by data which describes size, privilege level required to access, protection (whether segment is read-only etc)
  - segments may or may not overlap
    - disjoint segments can be used to protect against programming errors
    - separate code, data stack segments

- Disjoint Segments can be used to exploit expanded address space
  - In 16 bit architectures e.g. (8086 and 80x86 in V86 mode) each segment has only 16 bits of address space
  - In distributed networks consisting of multiple 32 bit machines, segmentation can be used to support single huge address space
- Segments can span identical regions of address space - *flat model*
  - Windows NT and Windows ‘95 use 4 Gbyte code segments, stack segments, data segments

X86 Segmentation + Paging