Chapter 3 Memory Management

—— *Virtual Memory System*

Yuan Cangzhou
yuancangzhou@buaa.edu.cn

Outline

- **Introduction to Virtual Memory System**
- Modular Implementation
- Virtual Address Spaces
- Segment Driver
- Page Fault

Why Have a Virtual Memory System?

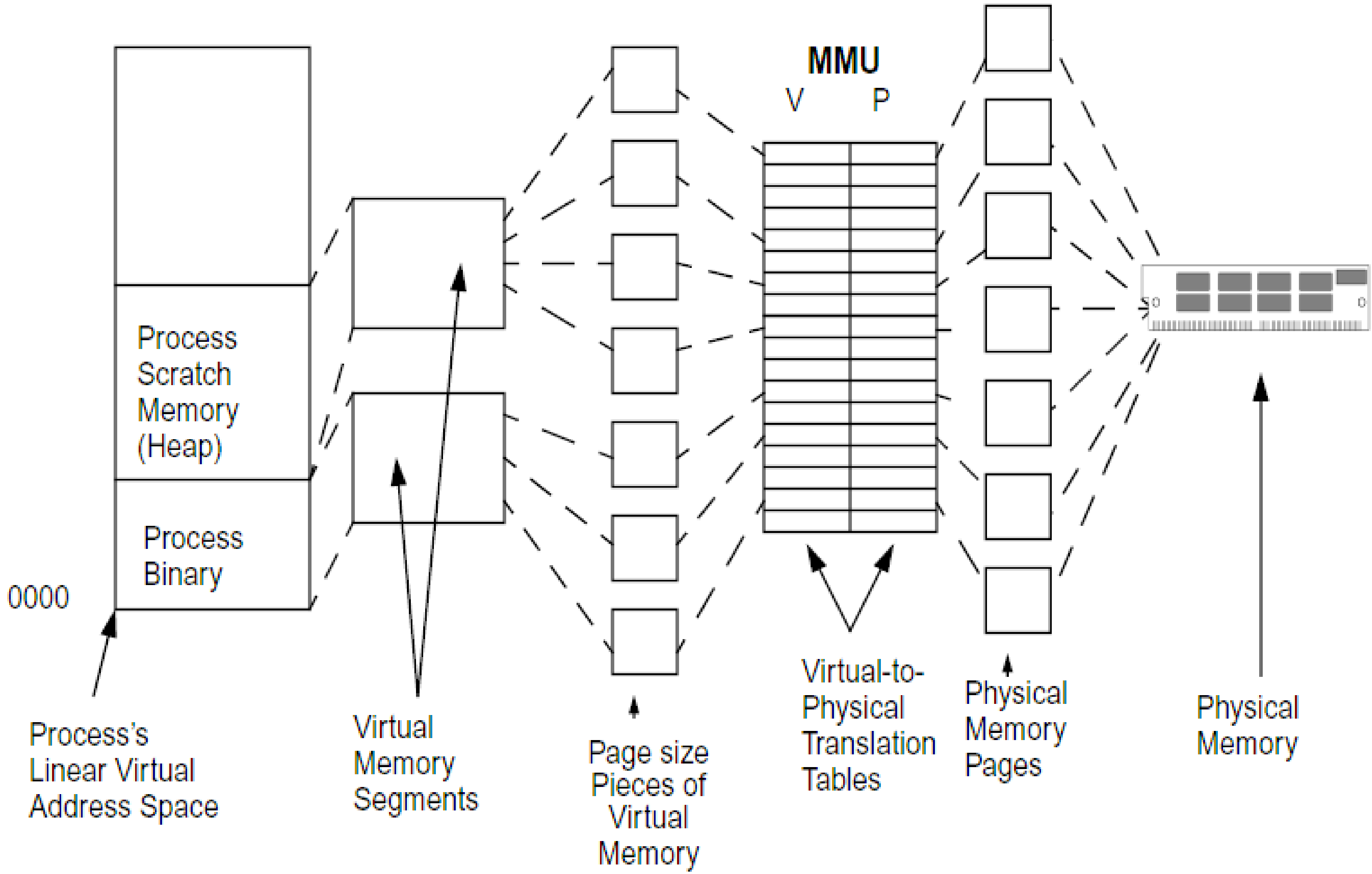
- A virtual memory system offers the following benefits:
  - It presents a simple memory programming model to applications so that application developers need not know how the underlying memory hardware is arranged.
  - It allows processes to see linear ranges of bytes in their address space, regardless of the physical layout or fragmentation of the real memory.

- It gives us a programming model with a larger memory size than available physical storage (e.g., RAM) and enables us to use slower but larger secondary storage (e.g., disk) as a backing store to hold the pieces of memory that don’t fit in physical memory.

The Major Functions of a VM System

- It manages virtual-to-physical mapping of memory
- It manages the swapping of memory between primary and secondary storage to optimize performance
- It handles requirements of shared images between multiple users and processes

Solaris Virtual-to-Physical Memory Management



Swapping and Demand Paging

- The Solaris kernel uses a combined demand-paged and swapping model
  - Demand paging is used under normal circumstances
  - Swapping is used only as a last resort when the system is desperate for memory

Memory Sharing and Protection

- Multiple users' processes can share memory
  - Multiple processes can sharing program binaries and application data
  - The Solaris kernel introduced dynamically linked libraries
- Memory Protection
  - A user's process must not be able access the memory of another process
  - A program fault in one program could cause another program (or the entire operating system) to fail
  - Using hardware facilities in the memory management unit

Other Functions of Solaris VM System

- Other than management of application memory, the Solaris VM system is responsible for managing:
  - the kernel
  - user applications
  - shared libraries
  - file systems
- One of the major advantages of using the VM system to manage file system buffering is that
  - Providing significant performance improvements for applications that use the file system
  - Removing the need for tuning the size of the buffer cache

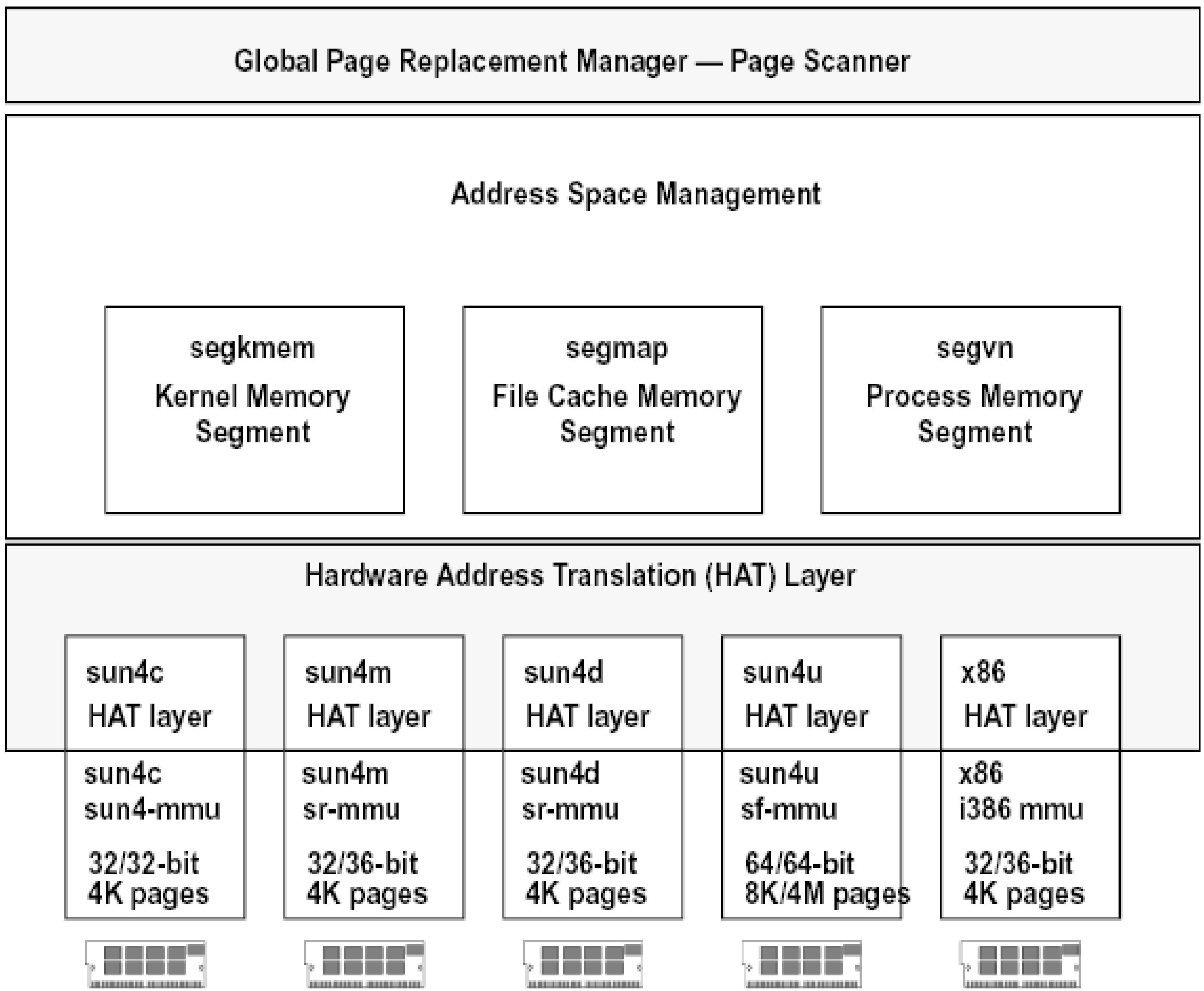
Outline

- Introduction to Virtual Memory System
- **Modular Implementation**
- Virtual Address Spaces
- Segment Driver
- Page Fault

The Memory System Objects

- The Solaris VM system provides an open framework that now supports many different memory objects
- The most important objects of the memory system are
  - Segments
  - Vnodes
  - Pages

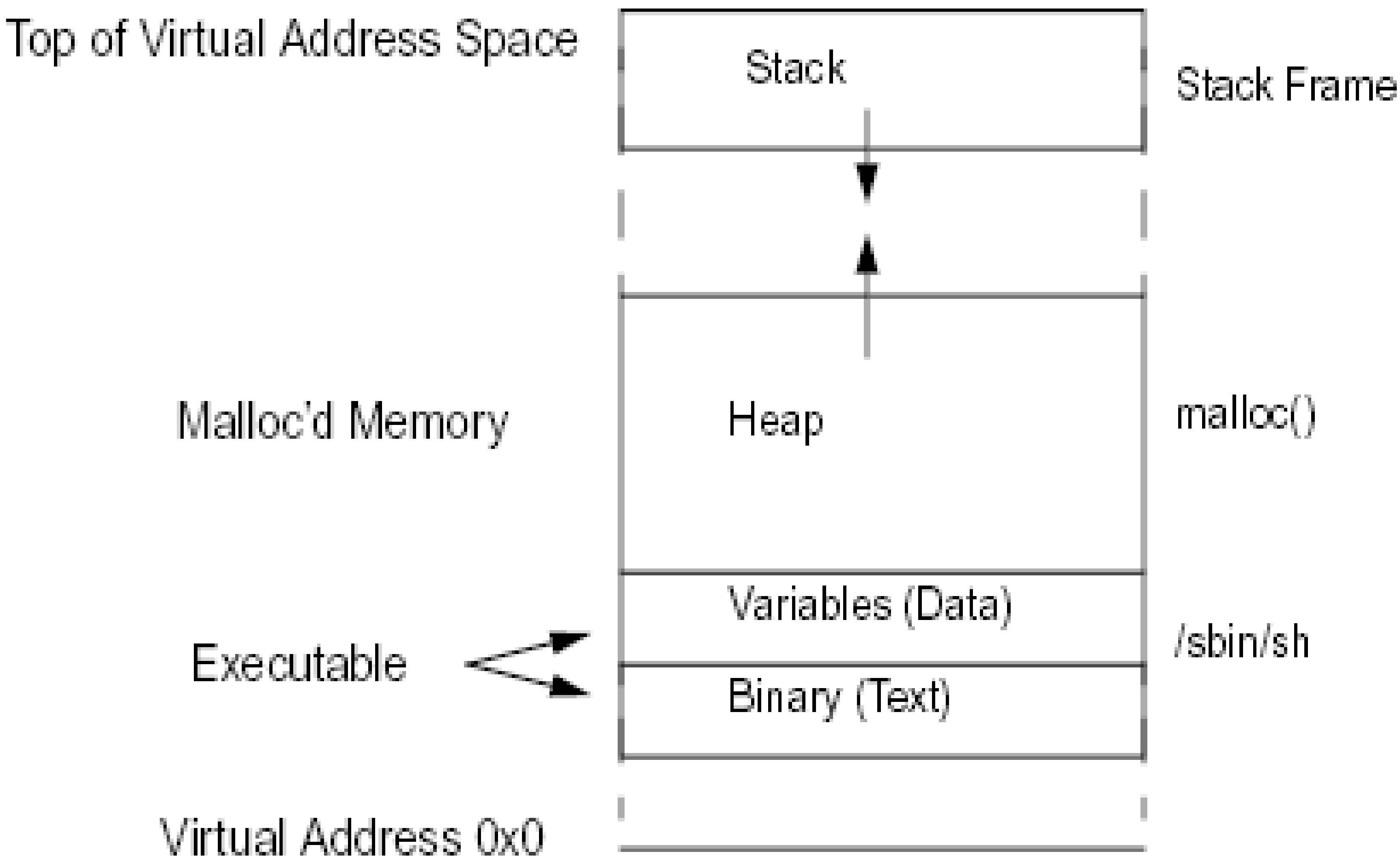
Solaris Virtual Memory Layers



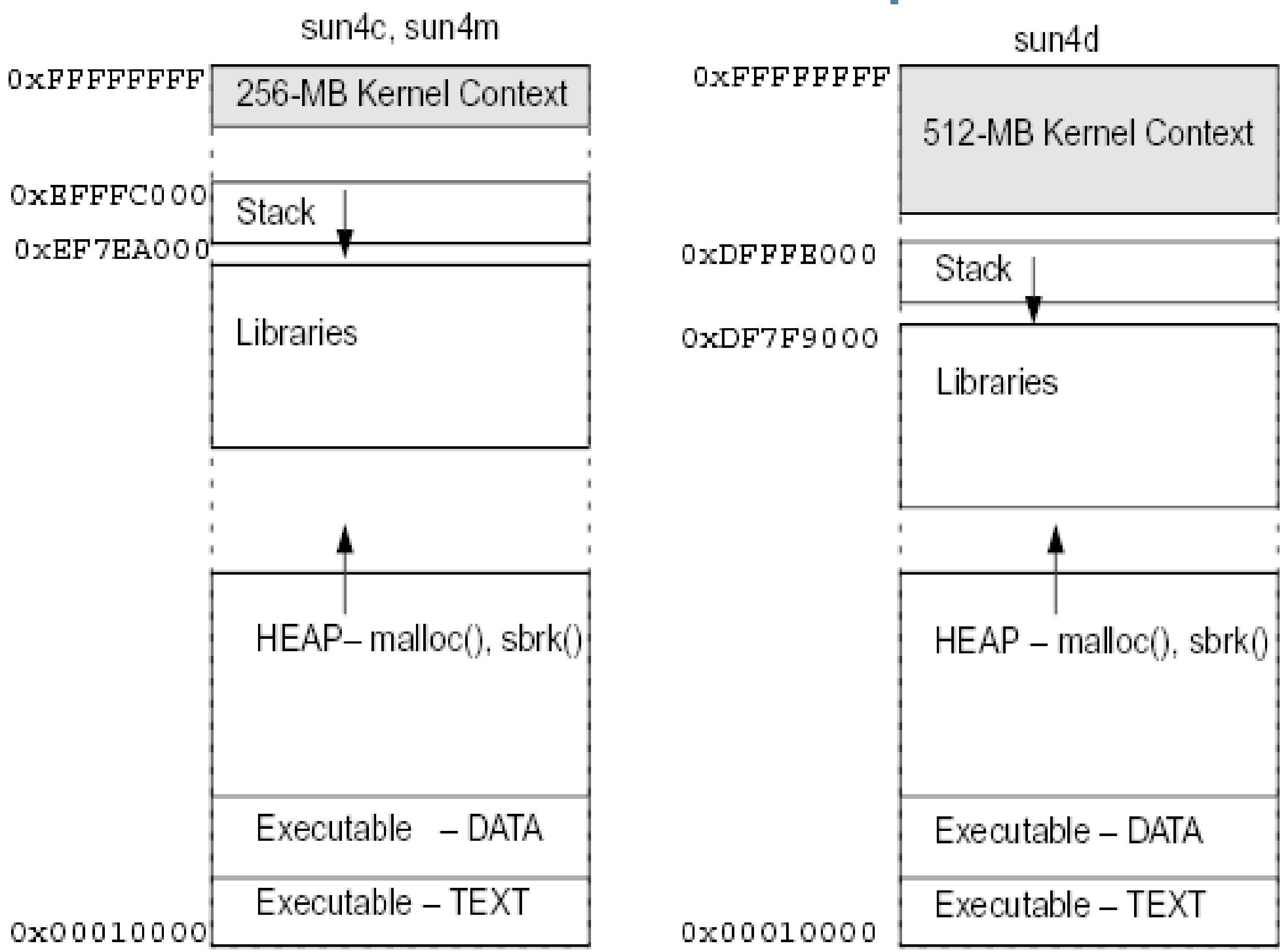
Outline

- Introduction to Virtual Memory System
- Modular Implementation
- **Virtual Address Spaces**
- Segment Driver
- Page Fault

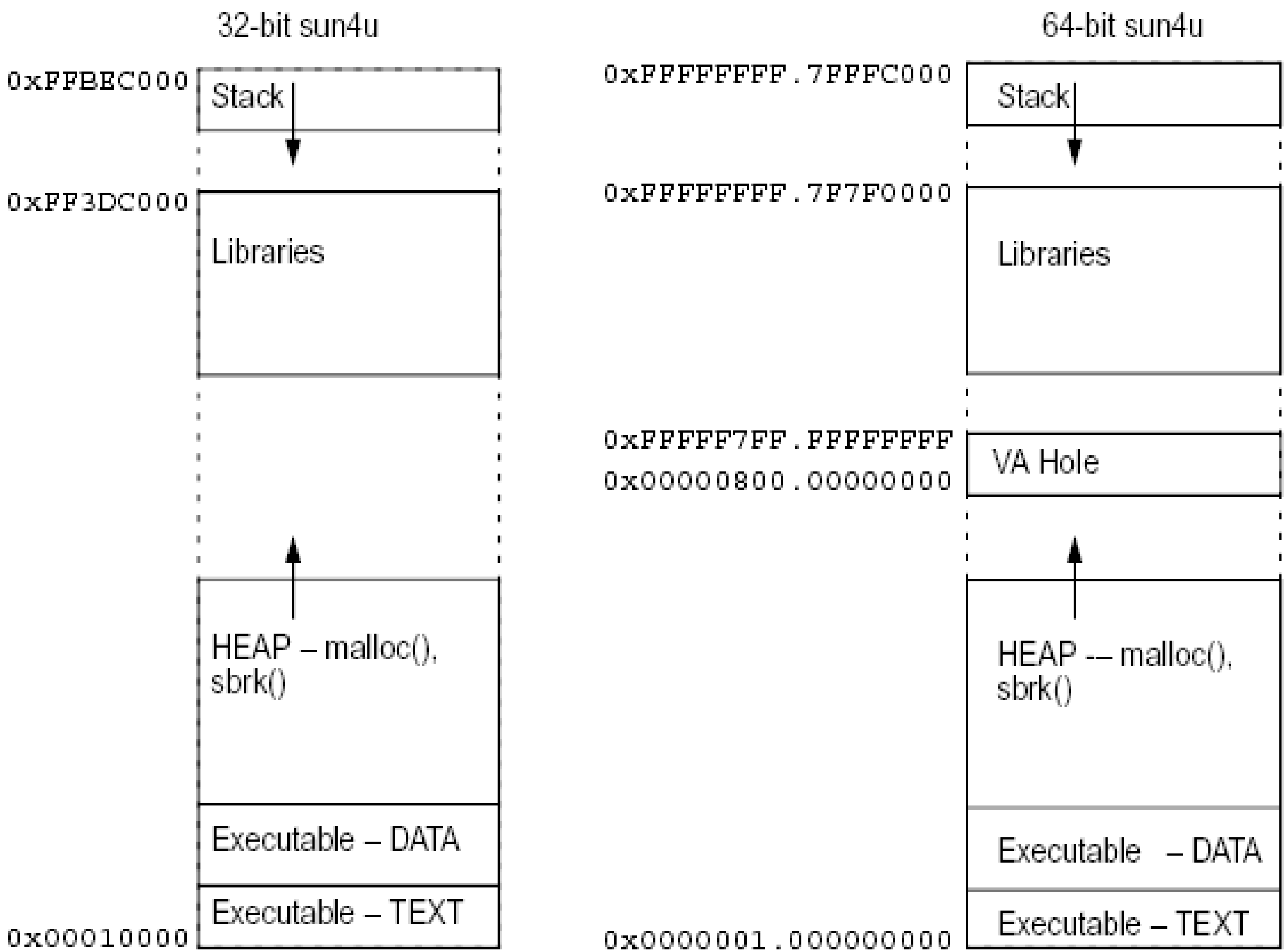
Process Virtual Address Space



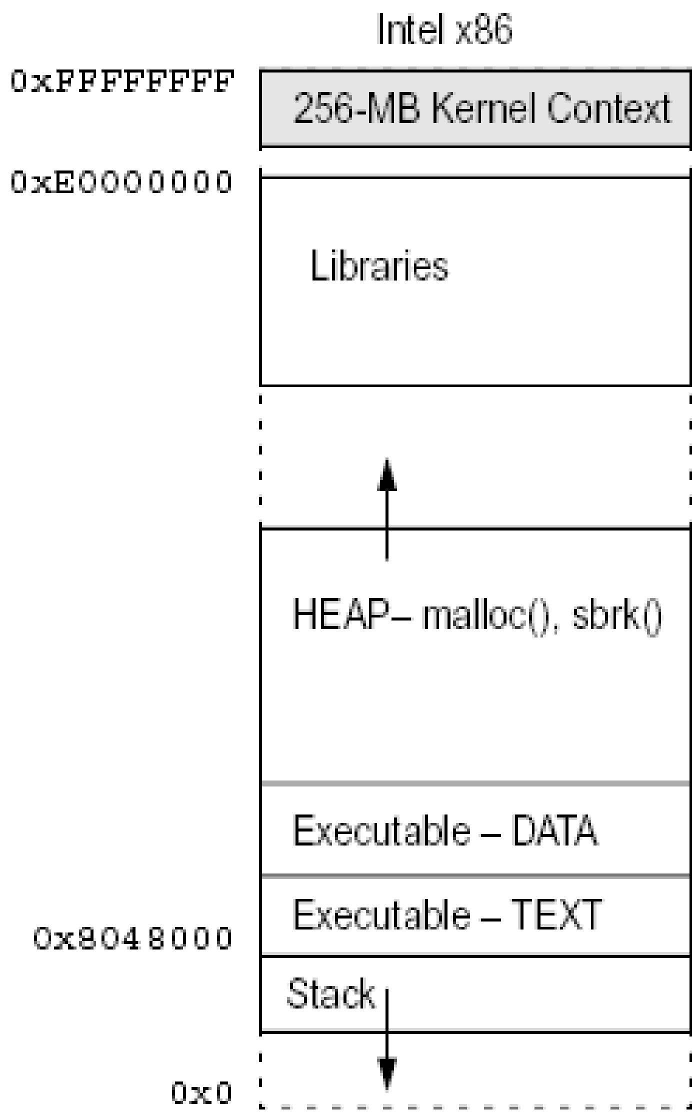
SPARC 32-Bit Shared Kernel/Process Address Space



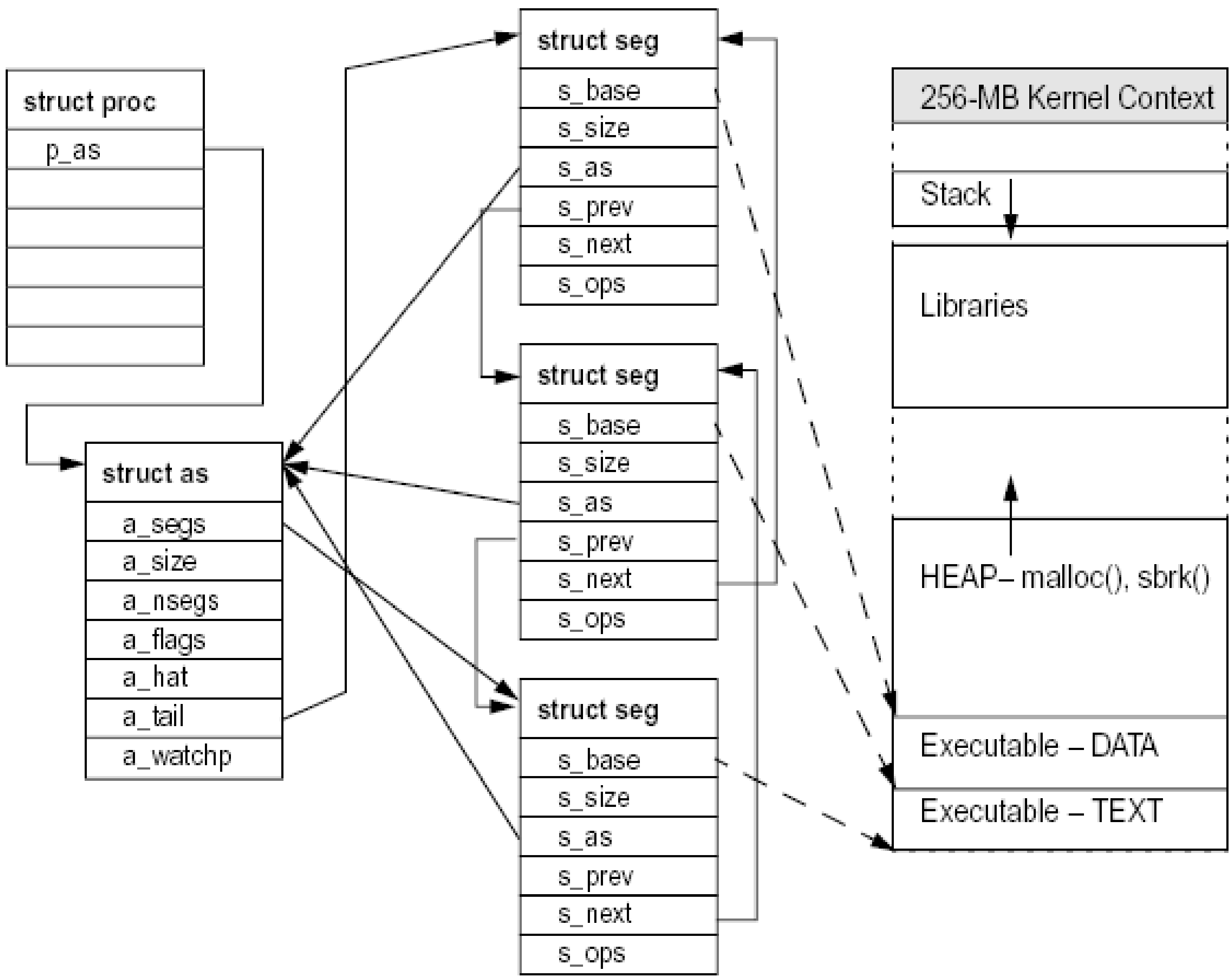
SPARC sun4u 32- and 64-Bit Process Address Space



Intel x86 Process Address Space



The Address Space



The Functions of Address Space Subsystem

- Duplication of address spaces, for fork()
- Destruction of address spaces, for exit()
- Creation of new segments within an address space
- Removal of segments from an address space
- Setting and management of page protection for an address space
- Page fault routing for an address space
- Page locking and advice for an address space
- Management of watchpoints for an address space

fork() and vfork()

- The fork() system call
  - Duplicating the address space of current process
  - Duplicating the entire address space configuration
- The vfork() system call
  - Borrowing the parent's existing address space
  - Calling exec() system call

Address Space Fault Handling

- Some of the faults are handled by the common address space code
  - If the fault does not lie in any of the address space's segments
- Others are redirected to the segment handlers
  - If the fault does lie within one of the segments

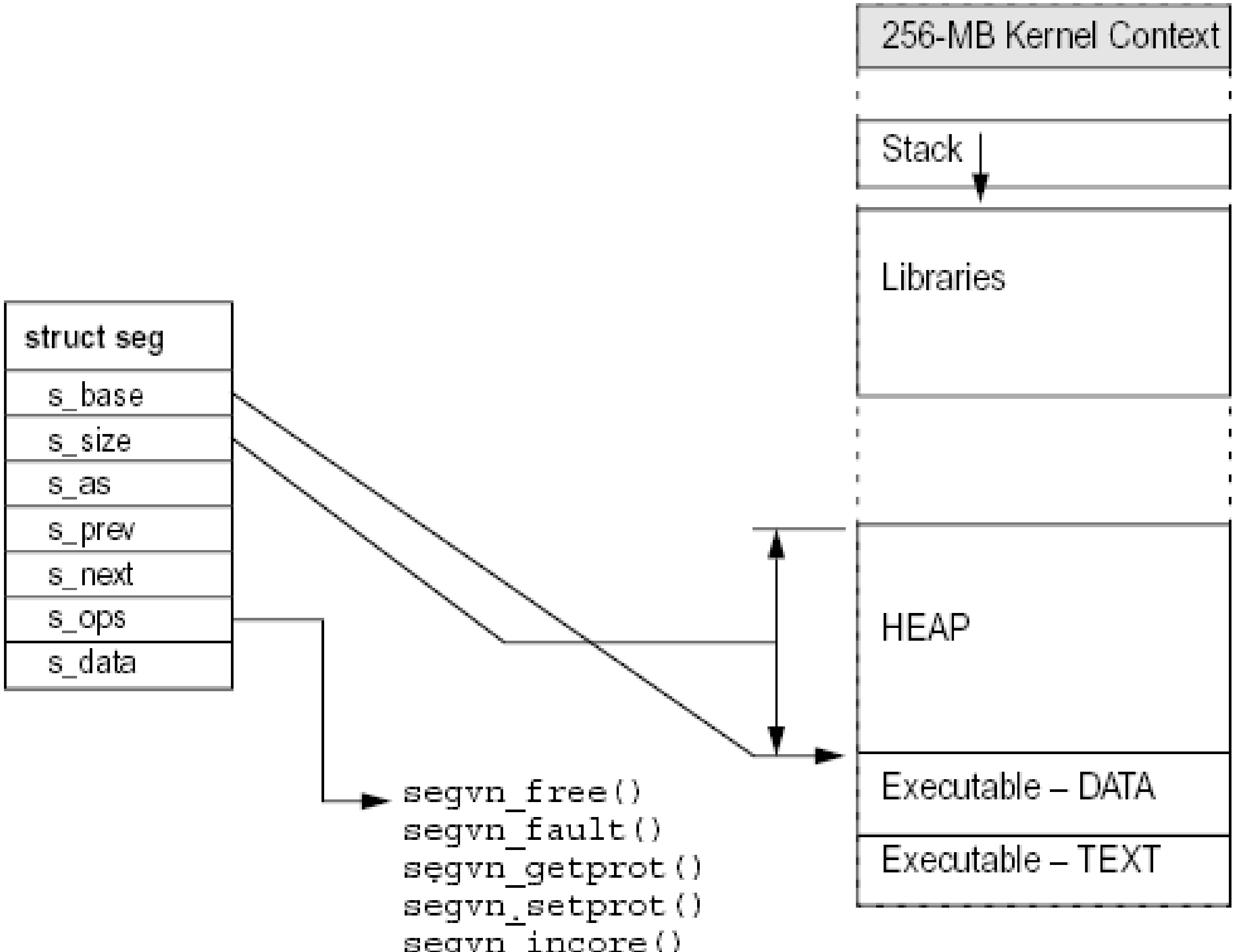
Outline

- Introduction to Virtual Memory System
- Modular Implementation
- Virtual Address Spaces
- **Segment Driver**
- Page Fault

Memory Segments

- Memory segments manage the mapping of a linear range of virtual memory into an address space
- The objective of the memory segment is to allow both memory and devices to be mapped into an address space

Segment Interface



Segment Driver

- The segment driver provides a similar view of linear address space
- To implement an address space, a segment driver implementation is required to provide at least the following:
  - functions to create a mapping for a linear address range
  - page fault handling routines to deal with machine exceptions within that linear address range
  - a function to destroy the mapping

Solaris 7 Segment Driver Methods

- Solaris 7 Segment Driver Methods
  - advise()、checkprot()、dump()、dup()、fault()、faulta()、free()、getmemid()、getoffset()
  - getprot()、gettype()、getvp()、incore()、kluster()、lockop()、pagelock()、setprot()
  - swapout()、sync()、unmap()
- A segment driver implements a subset of the above methods

Outline

- Introduction to Virtual Memory System
- Modular Implementation
- Virtual Address Spaces
- Segment Driver
- **Page Fault**

Page Faults

- When do Page Faults occur
  - MMU-generated exceptions (trap) tell the operating system when a memory access cannot continue without the kernel's intervention
- Three major types of memory-related hardware exceptions can occur:
  - major page faults
  - minor page faults
  - protection faults

Major Page Faults

- When does a major page fault occur
  - when an attempt to access a virtual memory location that is mapped by a segment does not have a physical page of memory mapped to it and the page does not exist in physical memory.
- How to arrange the new page
  - Create a new page for that address, in the case of the first access
  - Get copies in the page from the swap device

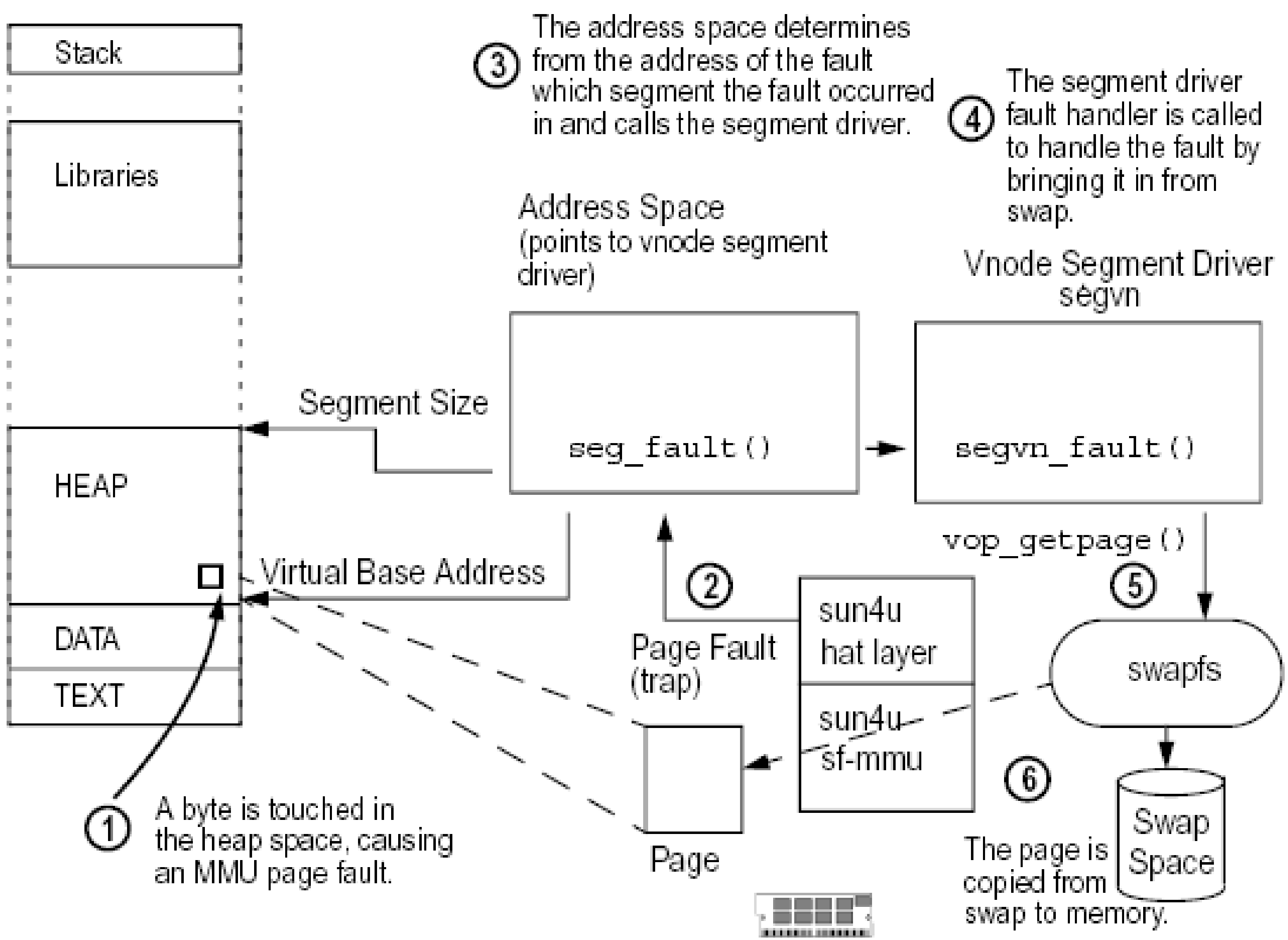
Minor Page Faults

- When does a minor page fault occur
  - When an attempt is made to access a virtual memory location that resides within a segment and the page is in physical memory
  - But no current MMU translation is established from the physical page to the address space that caused the fault
- A page fault occurs, but the physical page of memory is already present and the process simply needs to establish a mapping to the existing physical page

Protection Faults

- When does a page protection fault occur
  - When a program attempts to access a memory address in a manner that violates the preconfigured access protection for a memory segment
- Protection modes can enable any of read, write, or execute acces
- The memory protection fault is also initiated by the hardware MMU as a trap that is then handled by the segment page fault handling routine

Page Fault Example



Reference

- Jim Mauro, Richard McDougall, Solaris Internals-Core Kernel Components, Sun Microsystems Press, 2000
- Sun, Multithreading in the Solaris Operating Environment, A Technical White Paper,2002
- Max Bruning, Threading Model In Solaris, Training lectures,2005
- Solaris internals and performance management, Richard McDougall, 2002

End

- Last.first@Sun.COM