

CS420: Operating Systems

Kernel I/O Subsystem

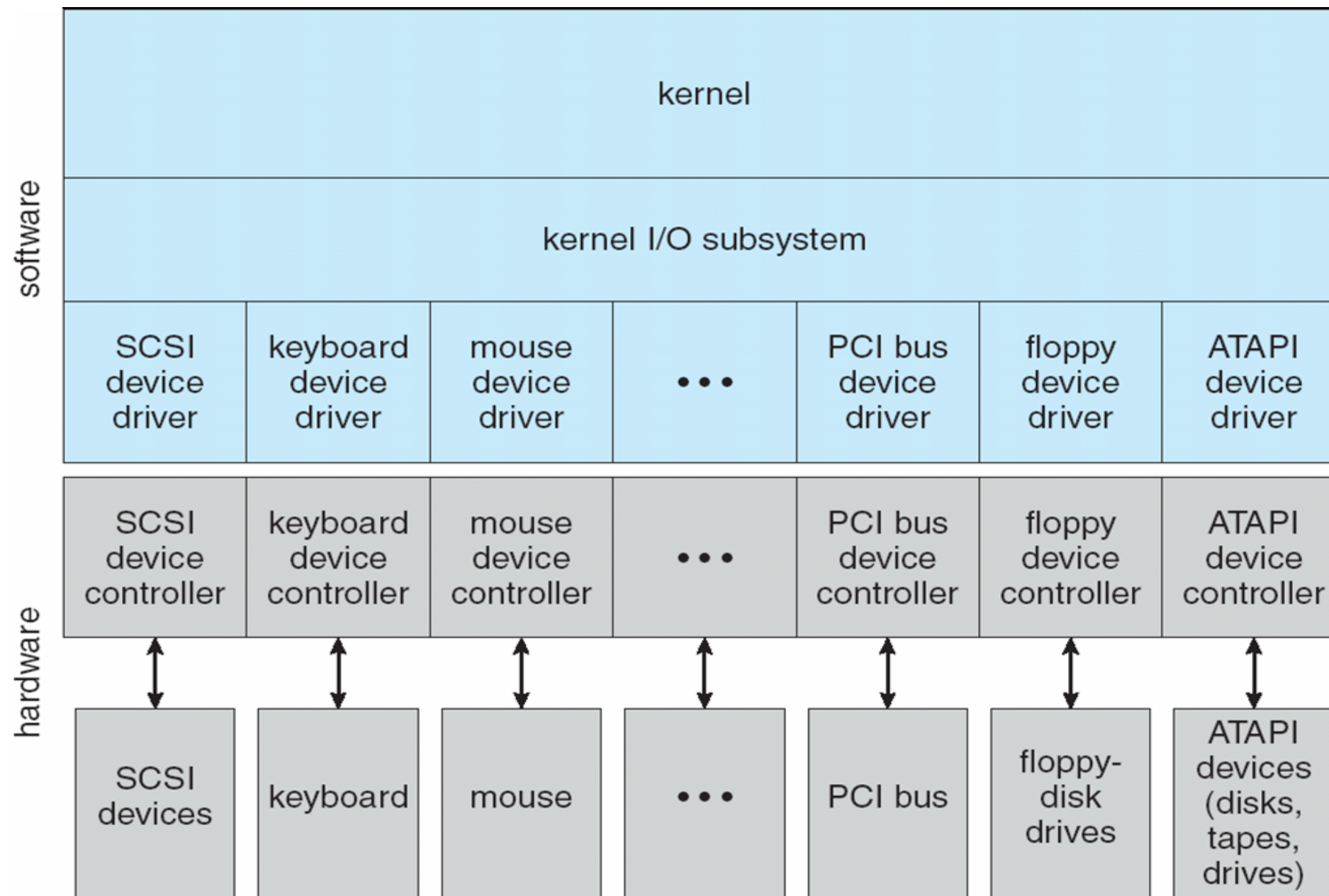
James Moscola

Department of Engineering & Computer Science

York College of Pennsylvania



A Kernel I/O Structure



Kernel I/O Subsystem

- **The kernel provides many services related to I/O**
 - Scheduling
 - Buffering
 - Caching
 - Spooling
 - Device reservation
 - Error handling

Kernel I/O Subsystem

- **Scheduling**

- Some I/O request ordering via per-device queue
 - Attempt to use devices optimally while still providing priority
- Some implement Quality of Service (i.e. IPQOS)

- **Buffering - store data in memory while transferring between devices**

- Helps cope with device speed mismatch
- Helps cope with device transfer size mismatch
- To maintain “copy semantics”
 - Data is first copied from user application memory into kernel memory
 - Data from kernel memory is then written to device
 - Prevents application from changing contents of a buffer before it is done being written

Kernel I/O Subsystem

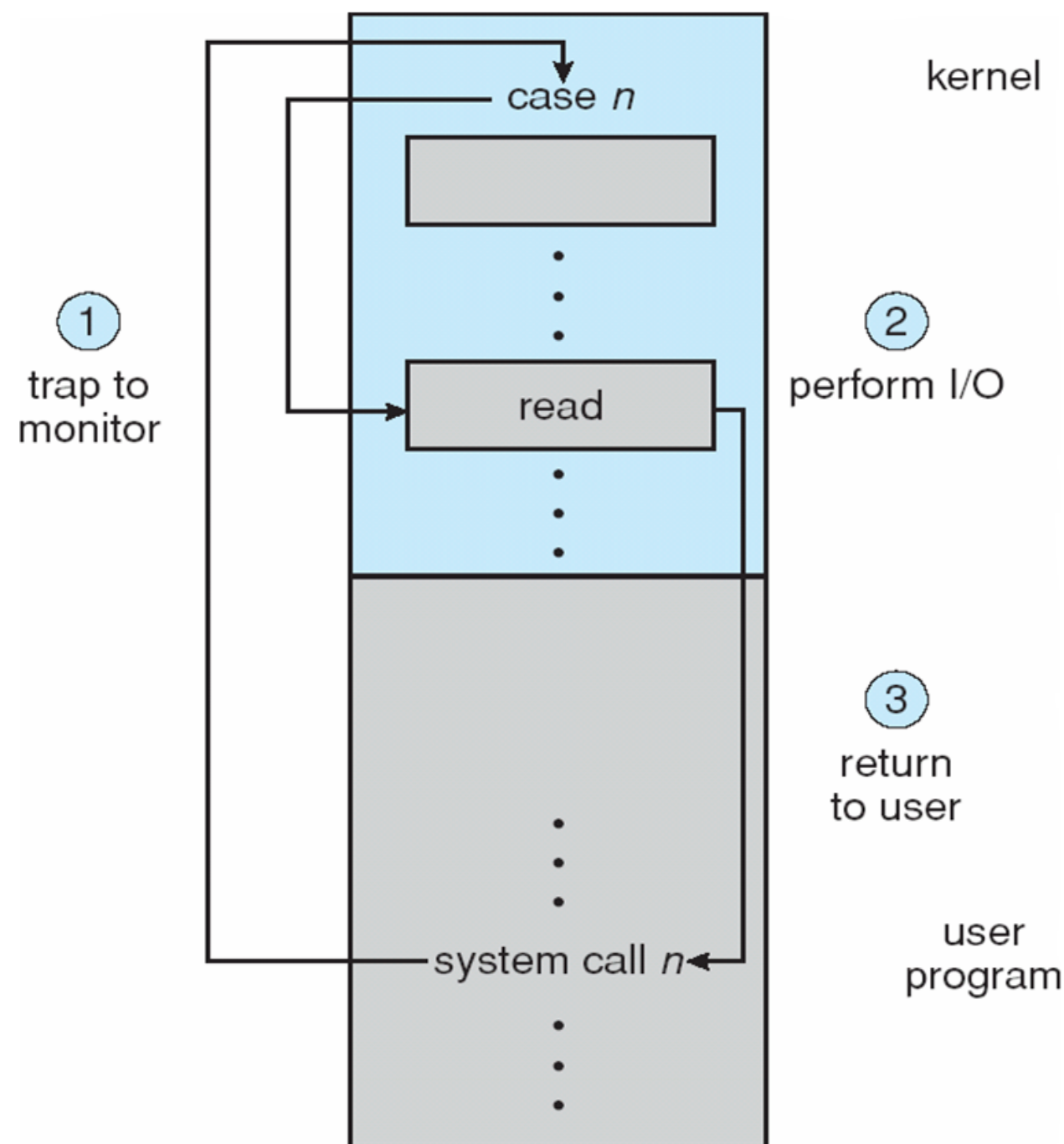
- **Caching** - possible to reuse buffers in main memory for caching
 - Cache files that are read/written frequently
 - Increases performance
- **Spooling** - a buffer that holds output for a device that cannot accept interleaved data streams
 - If device can serve only one request at a time
 - Allows a user to see individual data streams and remove streams if desired
 - i.e., Printing
- **Device reservation** - provides exclusive access to a device
 - System calls for allocation and de-allocation
 - Watch out for deadlock

Error Handling

- **OS can recover from disk read, device unavailable, transient write failures**
 - Retry a read or write, for example
 - Some systems more advanced – Solaris FMA, AIX
 - Track error frequencies, stop using device with increasing frequency of retry-able errors
- **Most operating systems return an error number or code when I/O request fails**
 - `errno` in UNIX/Linux environments
- **System error logs hold problem reports**

I/O Protection

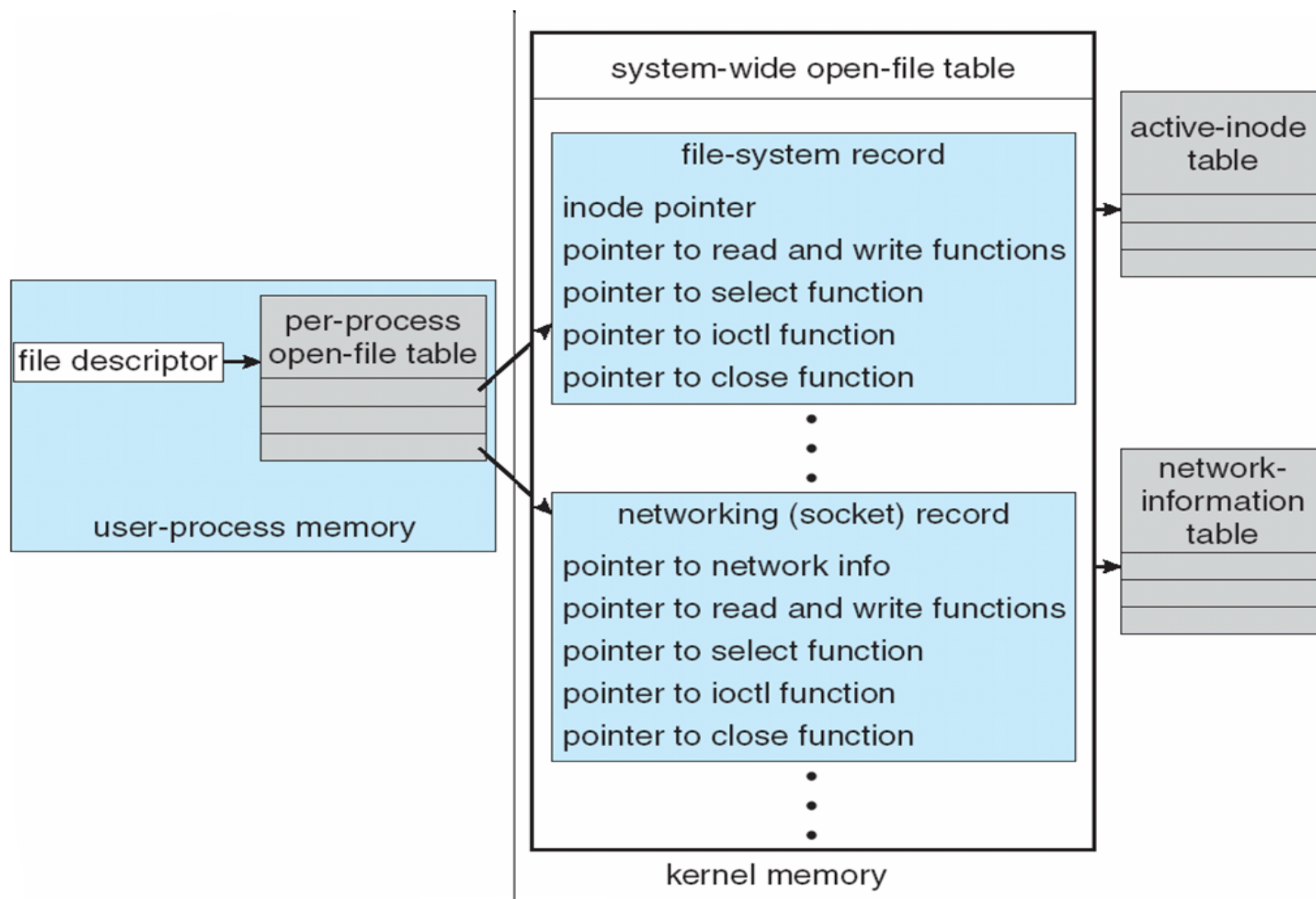
- **User process may accidentally or purposefully attempt to disrupt normal operation via illegal I/O instructions**
 - All I/O instructions defined to be privileged (i.e. kernel must perform I/O)
 - I/O must be performed via system calls
 - Memory-mapped and I/O port memory locations must be protected too



Kernel Data Structures

- **Kernel keeps state info for I/O components, including open file tables, network connections, etc.**
- **Many complex data structures to track buffers, memory allocation, “dirty” blocks**
- **Some operating systems use object-oriented methods and message passing to implement I/O**
 - Windows uses message passing
 - Message with I/O information passed from user mode into kernel
 - Message modified as it flows through to device driver and back to process

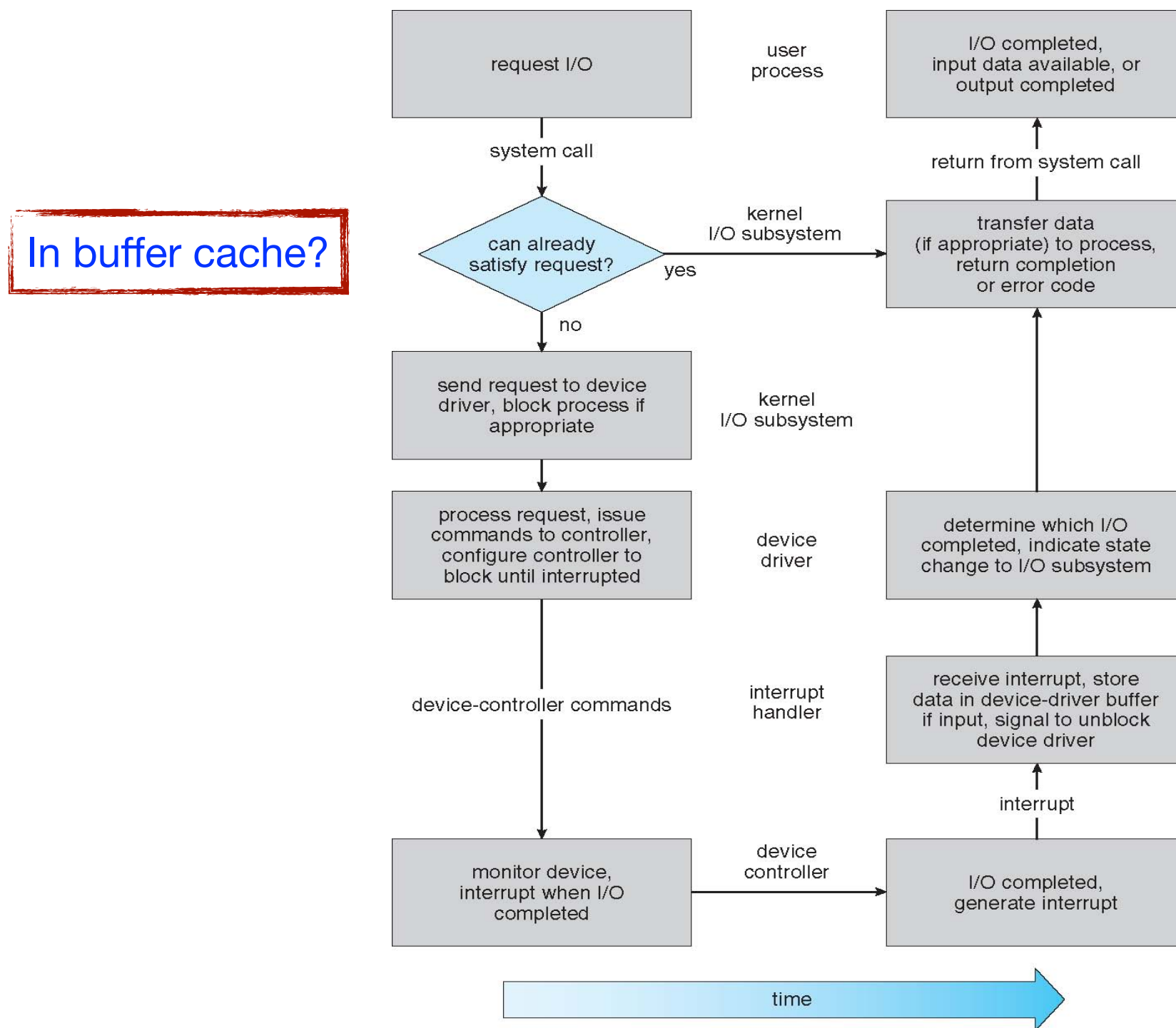
UNIX I/O Kernel Structure



I/O Requests to Hardware Operations

- **Consider reading a file from disk for a process:**
 - Determine device holding file
 - Translate name to device representation
 - Physically read data from disk into buffer
 - Make data available to requesting process
 - Return control to process

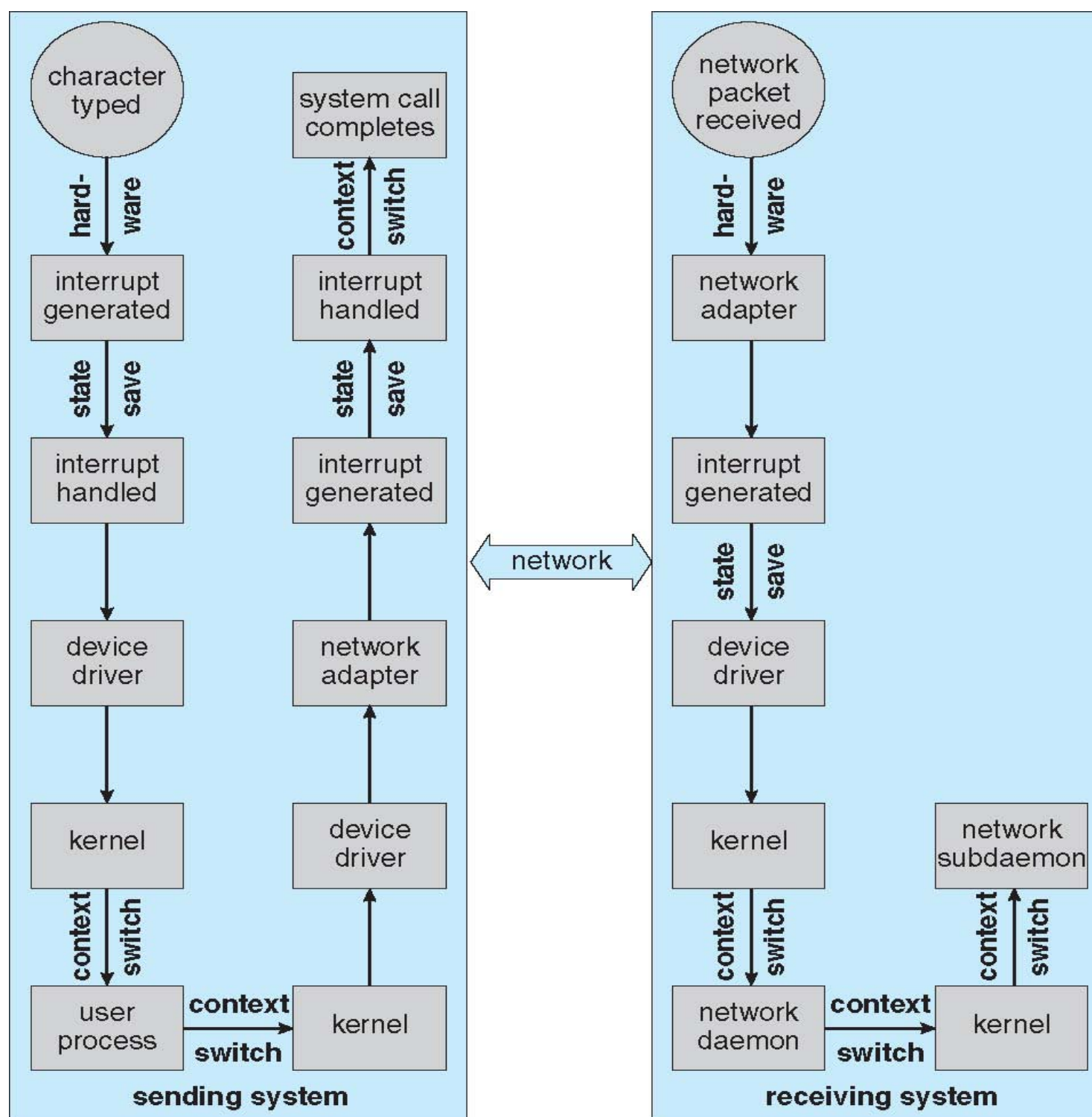
Life Cycle of An I/O Request



Performance

- **I/O a major factor in system performance:**
 - Demands CPU to execute device driver, kernel I/O code
 - Context switches due to interrupts
 - Data copying
 - Network traffic especially stressful

Intercomputer Communications



Methods to Improve System Performance

- **Reduce number of context switches**
- **Reduce data copying when passing data between device and application**
- **Reduce frequency of interrupts by using large transfers, smart controllers, polling**
- **Use DMA controllers to offload simple data copying from the CPU**
- **Use smarter hardware devices that can do more of the work**
- **Move user-mode processes / daemons to kernel threads**

Device-Functionality Progression

