Principles of Operating Systems

CS 446/646

5. Input/Output

a. Overview of the O/S Role in I/O

b. Principles of I/O Hardware

c. **I/O Software Layers**
   - ✓ Overview of the I/O software
   - ✓ Interrupt handlers
   - ✓ Device drivers
   - ✓ Device-independent I/O software
   - ✓ User-level I/O system calls

d. **Disk Management**

5.c I/O Software Layers

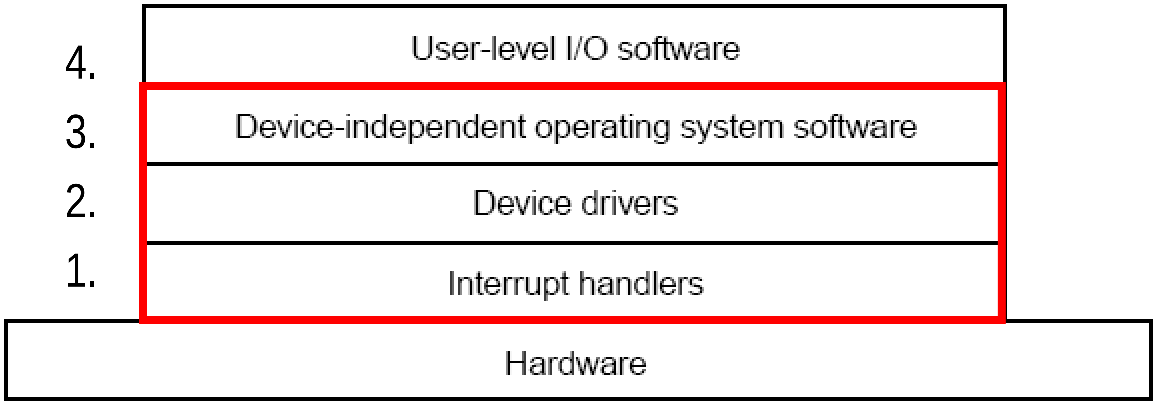
Overview of the I/O software

- Goals and services of the I/O software
  - ✓ device independence
    - access any new I/O device without rewriting the O/S
  - ✓ uniform naming
    - abstract naming space independent from physical device
  - ✓ error handling
    - lower layers try to handle the error before upper levels
  - ✓ asynchronous transfers
    - make interrupt-driven operations look blocking to processes
  - ✓ buffering
    - decouple transfer rates and insulate data from swapping

5.c I/O Software Layers

Overview of the I/O software

- The I/O component of the O/S is organized in layers
  1. interrupt handlers
  2. device drivers
  3. device-independent I/O
  4. user-level I/O system calls



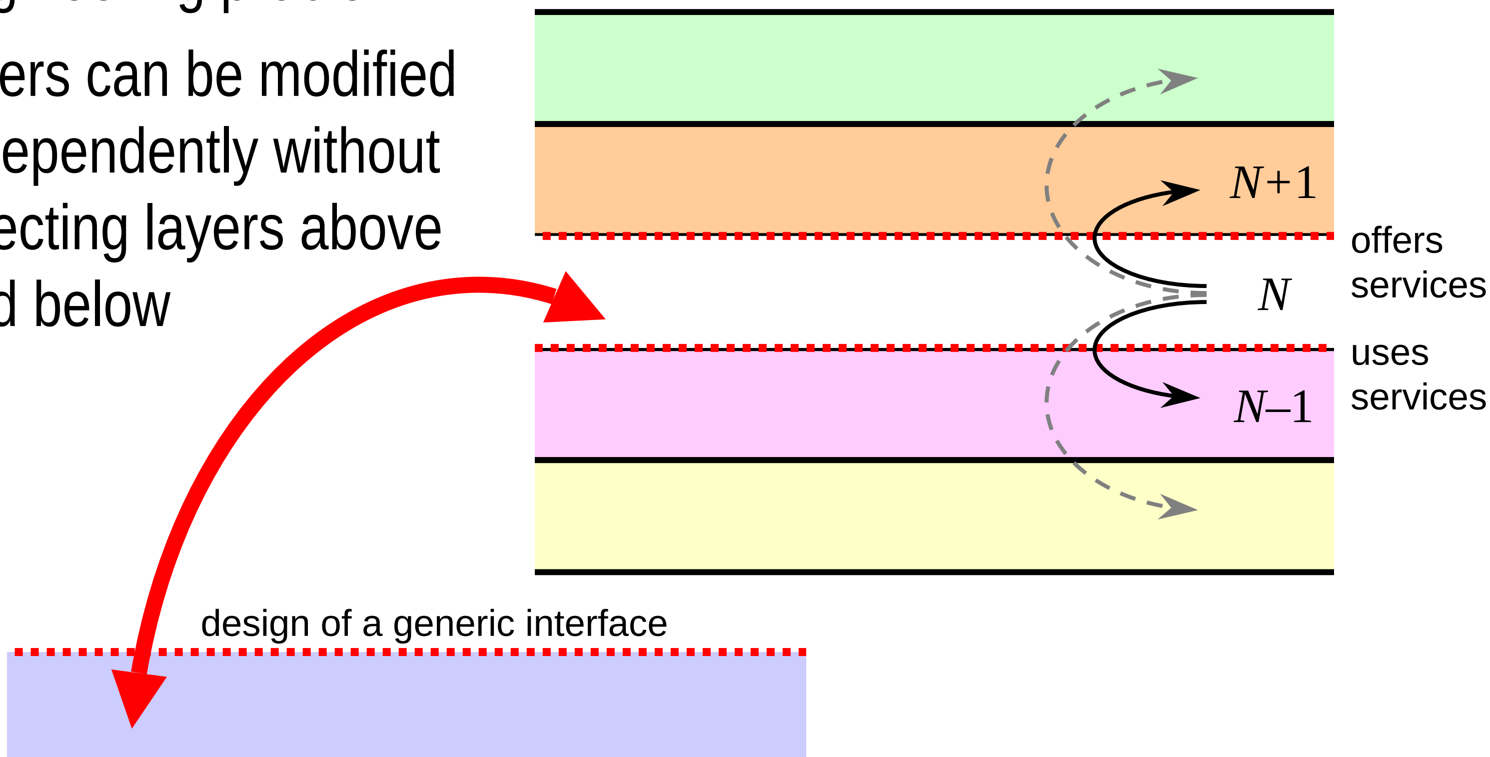
Tanenbaum, A. S. (2001)
Modern Operating Systems (2nd Edition).

Typical layers of the I/O software subsystem

5.c I/O Software Layers

Overview of the I/O software

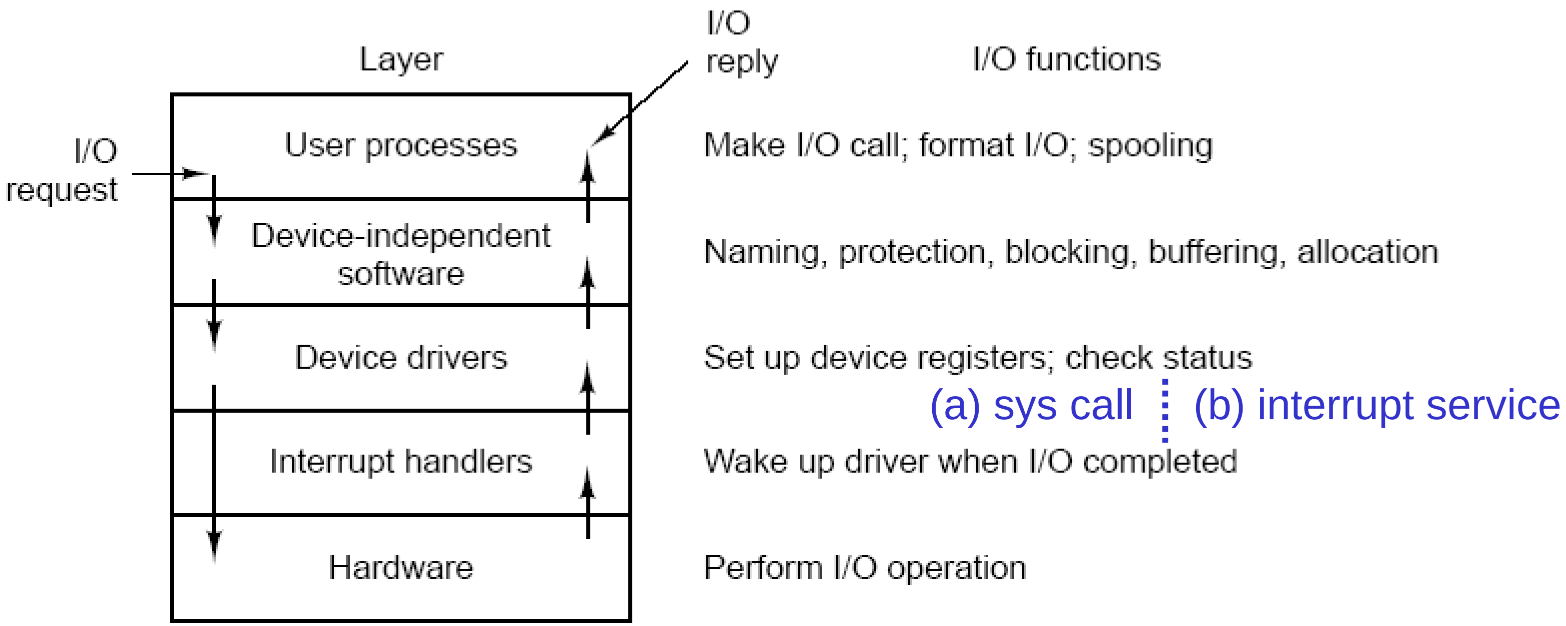
- Abstraction, encapsulation and layering
  - ✓ any complex software engineering problem
  - ✓ layers can be modified independently without affecting layers above and below



5.c I/O Software Layers

Overview of the I/O software

- Typical flow of control through the I/O layers upon an I/O request



Tanenbaum, A. S. (2001)
Modern Operating Systems (2nd Edition).

5.c I/O Software Layers

Interrupt handlers

1. Interrupt handler routines
   - ✓ interrupts basically use the same mechanism as exceptions and traps
   - ✓ when an interrupts happen, the CPU saves a small amount of state and jumps to an interrupt-handler routine at a fixed address in memory
   - ✓ the interrupt routine’s location is determined by an interrupt vector

5.c I/O Software Layers

Interrupt handlers

1. Interrupt handler routines (cont'd)

vector number	description
0	divide error
1	debug exception
2	null interrupt
3	breakpoint
4	INTO-detected overflow
5	bound range exception
6	invalid opcode
7	device not available
8	double fault
9	coprocessor segment overrun (reserved)
10	invalid task state segment
11	segment not present
12	stack fault
13	general protection
14	page fault
15	(Intel reserved, do not use)
16	floating-point error
17	alignment check
18	machine check
19Ð31	(Intel reserved, do not use)
32Ð255	maskable interrupts

nonmaskable, used for various error conditions

maskable, used for device-generated interrupts

Intel Pentium processor event-vector table

Silberschatz, A., Galvin, P. B. and Gagne. G. (2003) *Operating Systems Concepts with Java (6th Edition).*

5.c I/O Software Layers

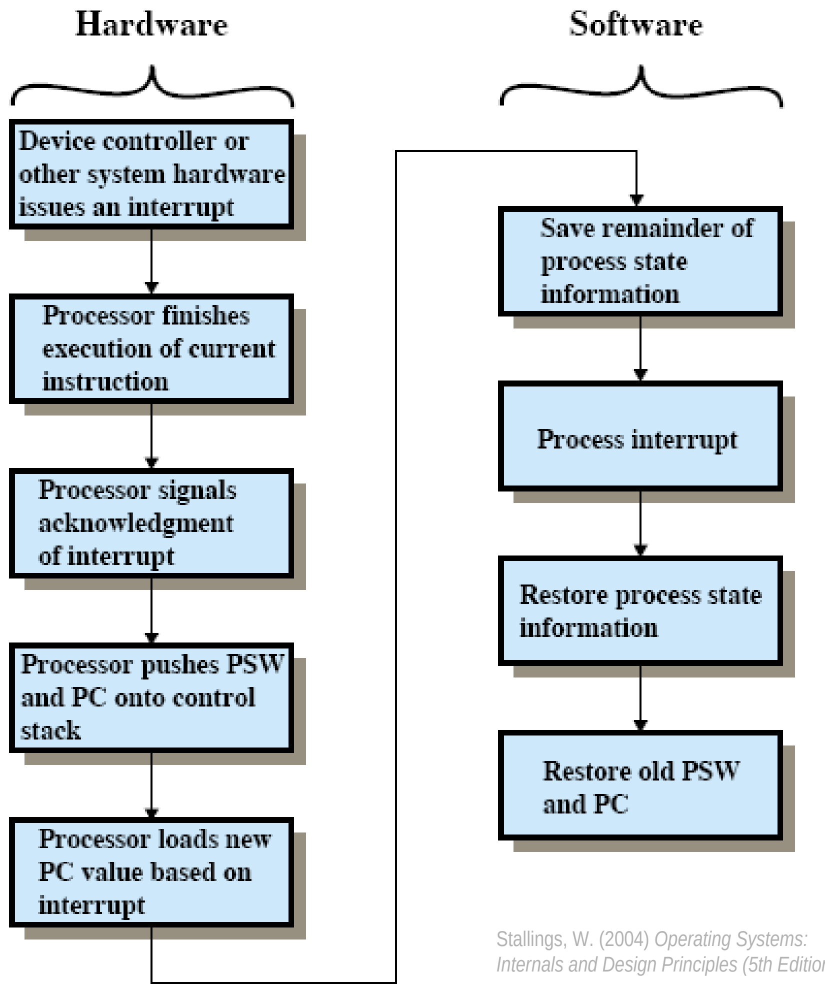
Interrupt handlers

1. Interrupt handler routines

- ✓ typical steps followed by an interrupt routine:
  a. save any registers not saved by the interrupt hardware
  b. set up a context (TLB, MMU, page table) for the routine
  c. set up a stack for the routine
  d. acknowledge the interrupt controller
  e. extract information from the I/O device controller's registers
  f. etc.
- ✓ interrupt processing is a complex operation that takes a great number of CPU cycles, especially with virtual memory

5.c I/O Software Layers

Interrupt handlers



Stallings, W. (2004) *Operating Systems: Internals and Design Principles (5th Edition).*

Simple interrupt processing

5.c I/O Software Layers

Device drivers

2. Device drivers

- ✓ each I/O device needs a **device-specific code** to control it
- ✓ device manufacturers supply drivers for several popular O/S
- ✓ a driver handles one type of device or one class (ex: SCSI)
- ✓ the driver logic is generally executed in kernel space (although microkernel architectures might push it in user space)
- ✓ drivers should “snap into place” in the kernel through device-independent interfaces (see next section)
- ✓ two main categories of drivers
  - block-device drivers: disks, etc.
  - character-device drivers: keyboards, printers, etc.

5.c I/O Software Layers

Device drivers

2. Device drivers (cont'd)

- ✓ a driver has several functions
  - accept abstract read/write requests from the device-independent software above and translate them into concrete I/O-module-specific commands
  - initialize the device, if needed
  - manage power requirements
  - log device events

5.c I/O Software Layers
Device drivers

2. Device drivers (cont'd)

✓ typical code organization of a device driver:

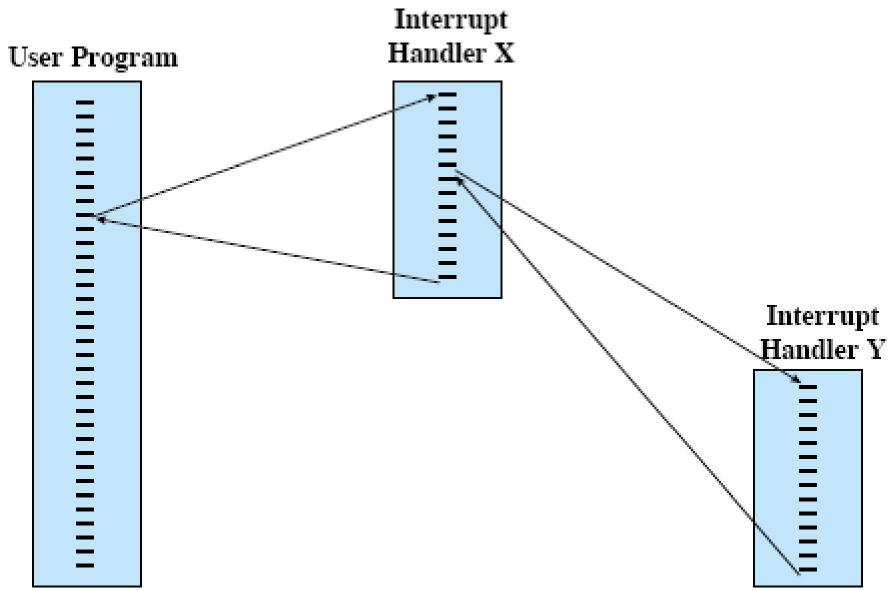
   a. check validity of input parameters coming from above
   b. if valid, translate to concrete commands, e.g., convert block number to head, track & sector in a disk's geometry
   c. check if device currently in use; if yes, queue request; if not, possibly switch device on, warm up, initialize, etc.
   d. issue appropriate sequence of commands to controller
   e. if needs to wait, block
   f. when interrupted, check for errors and pass data back
   g. process next queued request

5.c I/O Software Layers

Device drivers

2. Device drivers (cont'd)

- ✓ a driver code must be reentrant to allow for nested interrupts



(b) Nested interrupt processing

Stallings, W. (2004) *Operating Systems: Internals and Design Principles (5th Edition).*

5.c I/O Software Layers

Device-independent I/O software

3. Device-independent I/O software

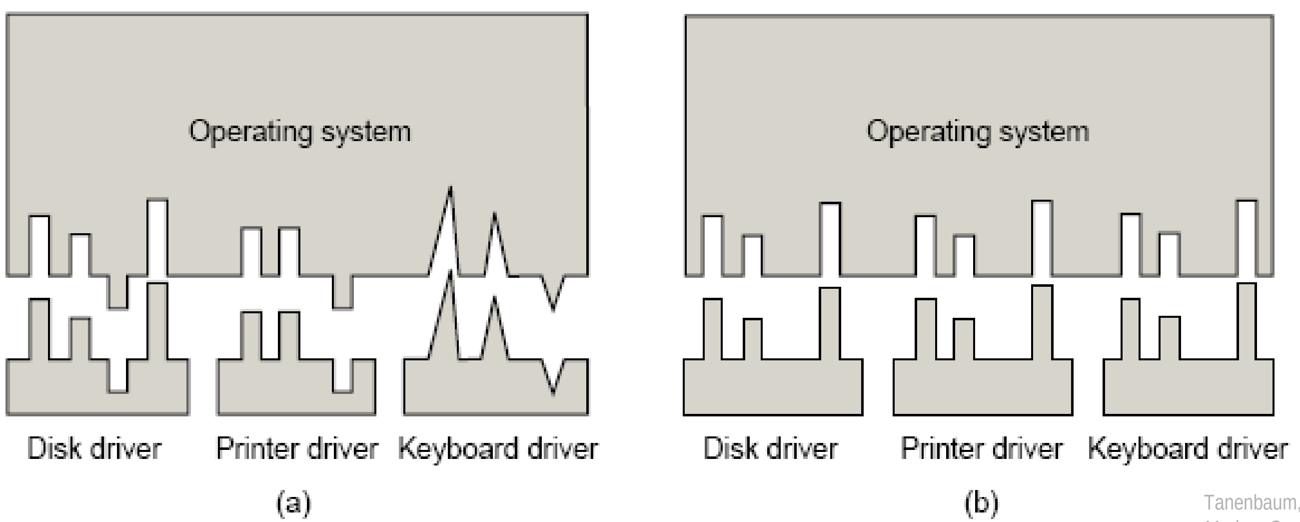
- ✓ generic functions provided by the kernel I/O subsystem:
  - uniform interfacing for device drivers
  - buffering
  - error reporting
  - providing a device-independent block size

5.c I/O Software Layers

Device-independent I/O software

3. Device-independent I/O software (cont'd)

- ✓ uniform interfacing
  - make all I/O devices look more or less the same, so that the O/S doesn't need to be hacked every time a new device comes along



Tanenbaum, A. S. (2001)
Modern Operating Systems (2nd Edition).

(a) Without and (b) with a standard driver interface

5.c I/O Software Layers

Device-independent I/O software

3. Device-independent I/O software (cont'd)

- ✓ uniform interfacing
  - therefore, generally one unified interface
  - possibly additional specialized extensions for the main device categories
  - block devices: **`read()`**, **`write()`**
  - random-access block devices: **`seek()`**
  - character-stream devices: **`get()`**, **`put()`**
  - network devices: network socket interface

5.c I/O Software Layers

Device-independent I/O software

3. Device-independent I/O software (cont'd)

✓ buffering = "decoupling"

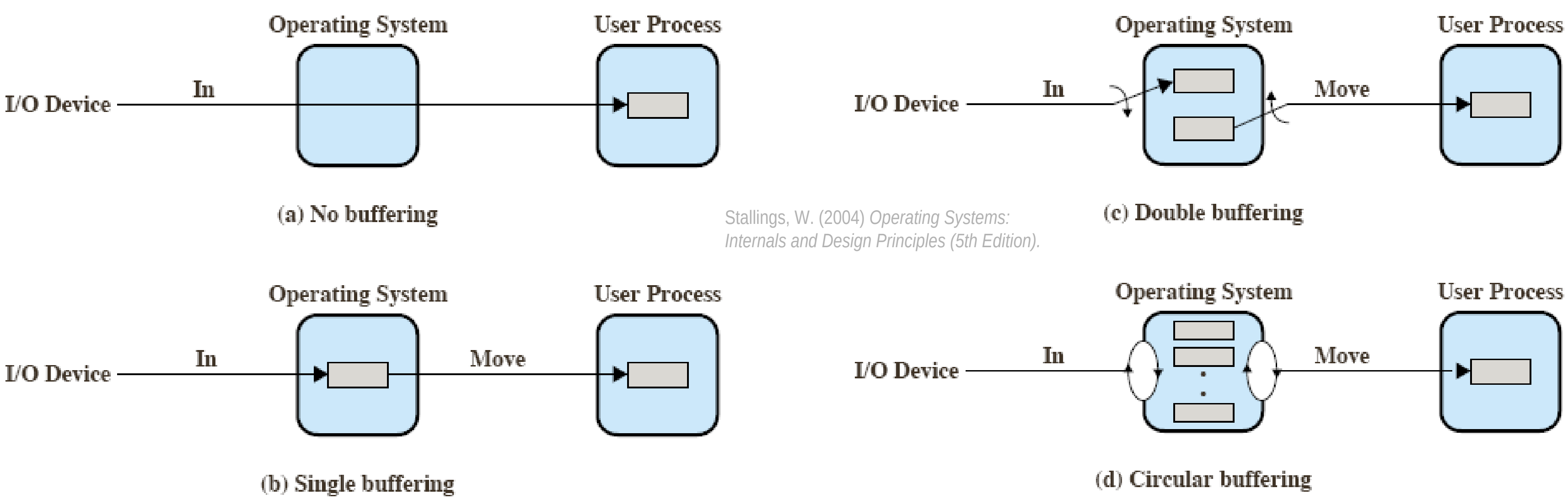
- memory area that stores data in kernel space while transferred between device and application
- cope with a speed mismatch between producer and consumer (ex: modem thousand times slower than disk)
- adapt between services with different data-transfer sizes (ex: fragmentation and reassembly of network packets)
- "copy semantics": cache data while transferred so it is not affected by changes from application or swapping
- read ahead (locality principle)

5.c I/O Software Layers

Device-independent I/O software

3. Device-independent I/O software (cont'd)

- ✓ buffering
  - double buffering: further decouples producer from consumer (ex: modem fills 2nd buffer while 1st buffer is written to disk)
  - circular buffering: extension suitable for rapid bursts of I/O



(a) No buffering

(b) Single buffering

(c) Double buffering

(d) Circular buffering

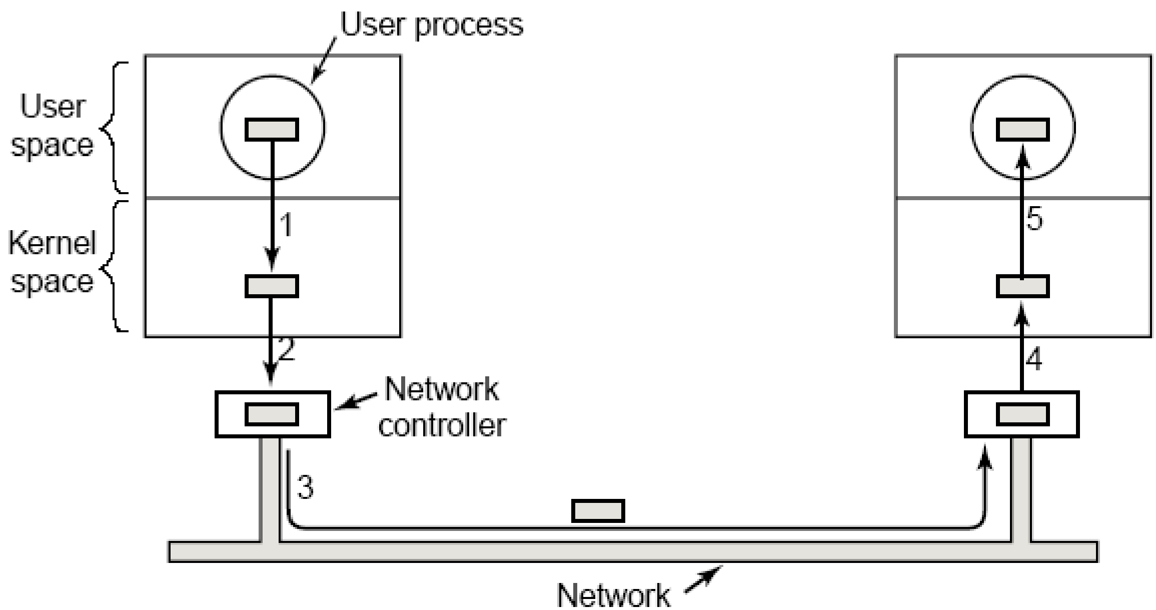
Stallings, W. (2004) *Operating Systems: Internals and Design Principles (5th Edition).*

5.c I/O Software Layers

Device-independent I/O software

3. Device-independent I/O software (cont'd)

- ✓ buffering in networking

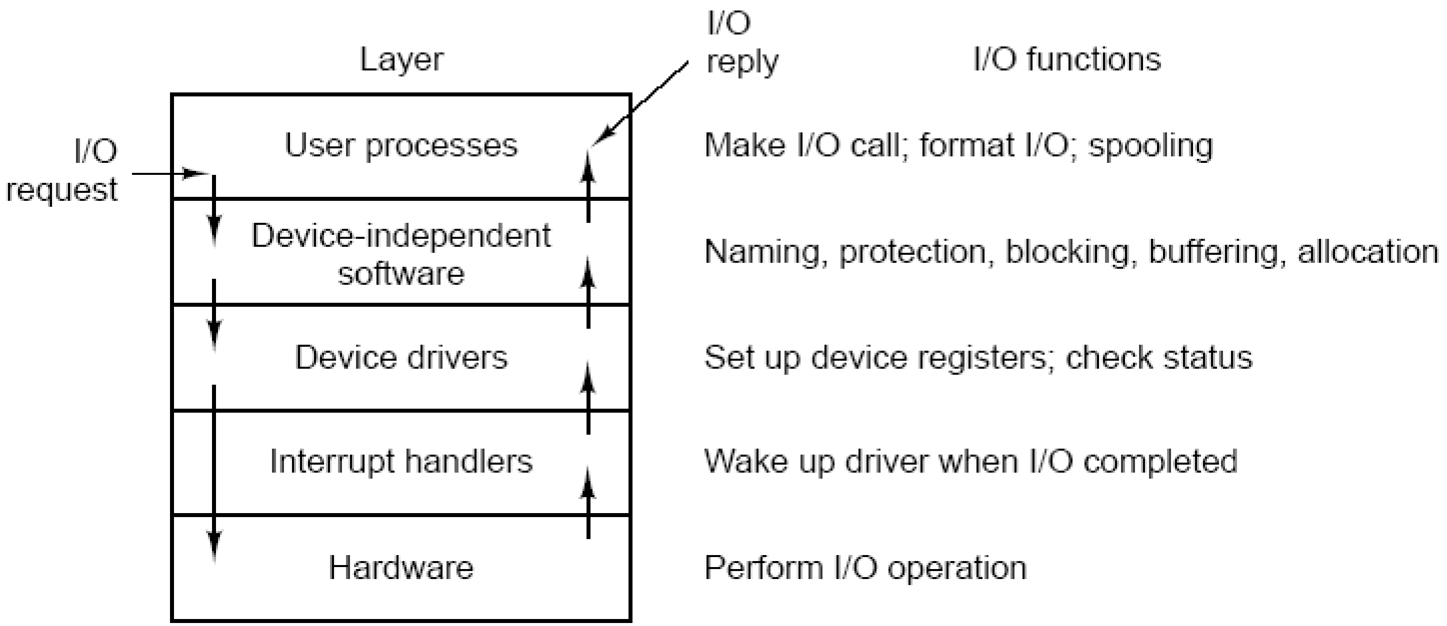


Tanenbaum, A. S. (2001)
Modern Operating Systems (2nd Edition).

5.c I/O Software Layers

User-level I/O system calls

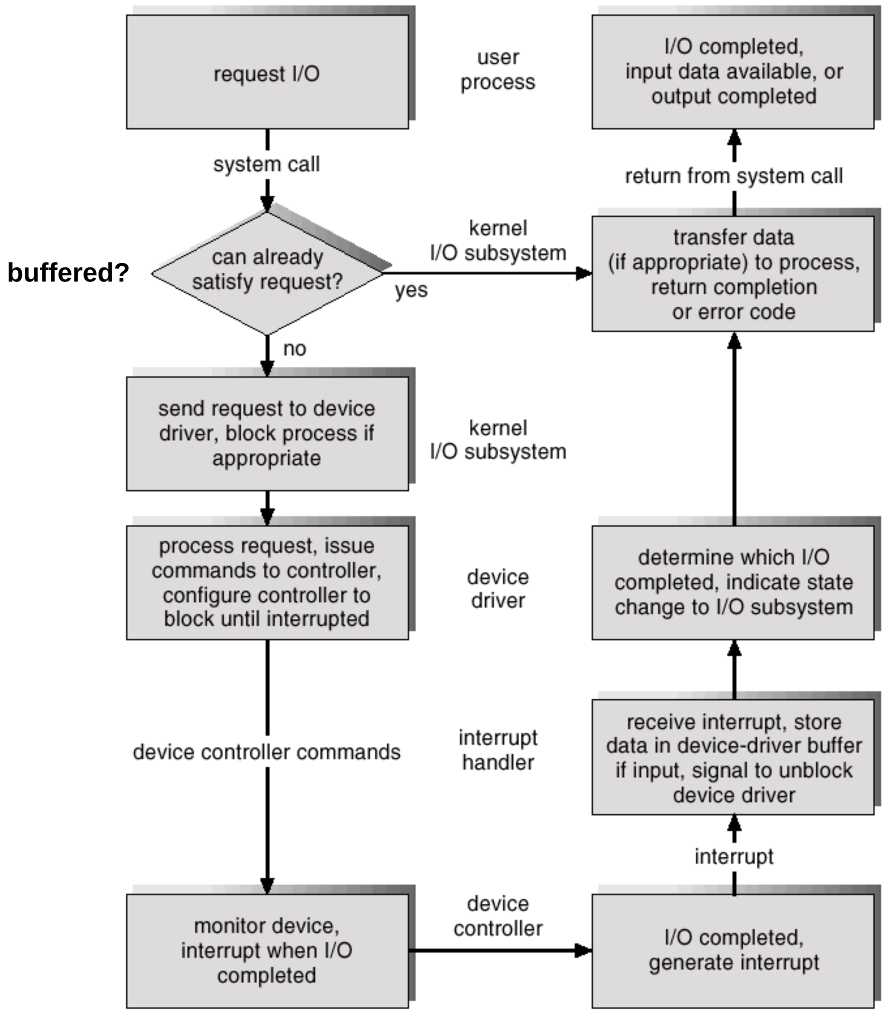
4. User-level I/O system calls



Tanenbaum, A. S. (2001)
Modern Operating Systems (2nd Edition).

5.c I/O Software Layers

User-level I/O system calls



Silberschatz, A., Galvin, P. B. and Gagne. G. (2003)
Operating Systems Concepts with Java (6th Edition).

The life-cycle of an I/O request

Principles of Operating Systems
CS 446/646

5. Input/Output

a. Overview of the O/S Role in I/O

b. Principles of I/O Hardware

c. **I/O Software Layers**
   - ✓ Overview of the I/O software
   - ✓ Interrupt handlers
   - ✓ Device drivers
   - ✓ Device-independent I/O software
   - ✓ User-level I/O system calls

d. **Disk Management**