Principles of Operating Systems

CS 446/646

5. Input/Output

René Doursat

Department of Computer Science & Engineering
University of Nevada, Reno

Fall 2005

Principles of Operating Systems

CS 446/646

0. Course Presentation
1. Introduction to Operating Systems
2. Processes
3. Memory Management
4. CPU Scheduling
5. **Input/Output**
6. **File System**
7. **Case Studies**

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**

d. **Disk Management**

Principles of Operating Systems

CS 446/646

5. Input/Output

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

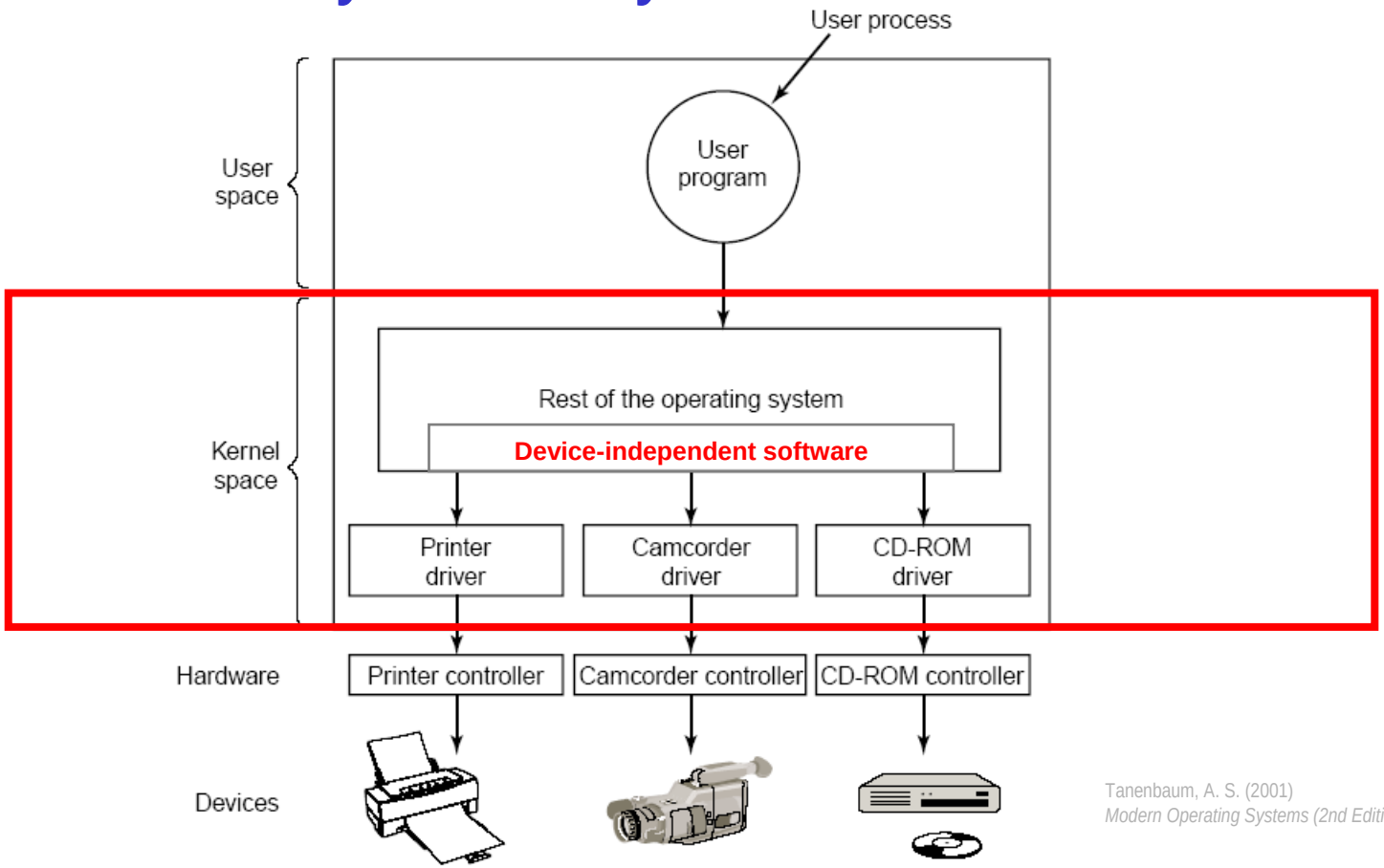
b. **Principles of I/O Hardware**

c. **I/O Software Layers**

d. **Disk Management**

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

- The I/O subsystem is layered

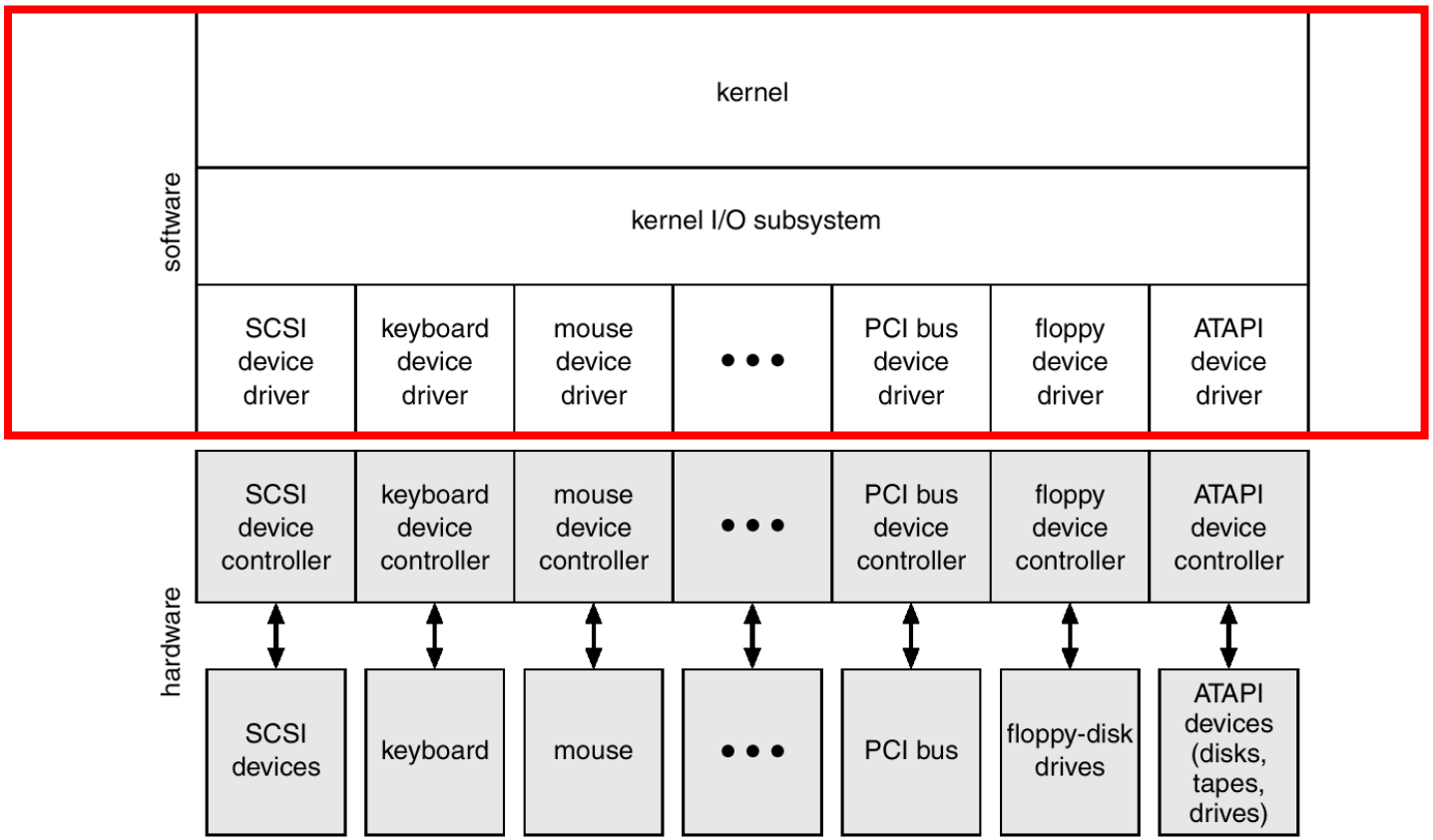


Tanenbaum, A. S. (2001)
Modern Operating Systems (2nd Edition).

Layers of the I/O subsystem

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

- The I/O subsystem is layered



A kernel I/O structure

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

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

- Chart of operating system responsibilities in I/O

§D – The O/S is responsible for controlling access to all the I/O devices

- ✓ the O/S hides the peculiarities of specific hardware devices from the user
- ✓ the O/S issues the low-level commands to the devices, catches interrupts and handles errors
- ✓ the O/S relies on software modules called “device drivers”
- ✓ the O/S provides a device-independent API to the user programs, which includes buffering

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**

d. **Disk Management**

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

- ✓ The diversity of I/O devices
- ✓ I/O bus architecture
- ✓ I/O devices & modules
- ✓ CPU-I/O communication

c. I/O Software Layers

d. Disk Management

5.b Principles of I/O Hardware

The diversity of I/O devices

- Great variety of I/O devices
  - ✓ storage devices
    - disks
    - tapes
  - ✓ transmission devices
    - network cards
    - modems
  - ✓ human-interface devices
    - screen
    - keyboard
    - mouse

5.b Principles of I/O Hardware

The diversity of I/O devices

- I/O devices vary in many dimensions (fuzzy boundaries)
  - ✓ character-stream vs. block
    - character devices transfer bytes one by one
    - block devices transfer blocks of bytes as units
  - ✓ sequential vs. random-access
    - sequential devices transfer in a fixed order they determine
    - random-access devices can be “seeked” at any storage location
  - ✓ synchronous vs. asynchronous
    - synchronous devices have predictable transfer times
    - asynchronous devices are irregular

5.b Principles of I/O Hardware

The diversity of I/O devices

- I/O devices vary in many dimensions (cont'd)
  - ✓ sharable vs. dedicated
    - sharable devices may be used concurrently
    - dedicated devices
  - ✓ speed of operation
    - devices speed range from a few bytes to a few GB per second
  - ✓ read-write, read only, or write only
    - some devices are both input/output, others only one-way

5.b Principles of I/O Hardware

The diversity of I/O devices

- I/O devices vary in many dimensions

aspect	variation	example
data-transfer mode	character block	terminal disk
access method	sequential random	modem CD-ROM
transfer schedule	synchronous asynchronous	tape keyboard
sharing	dedicated sharable	tape keyboard
device speed	latency seek time transfer rate delay between operations	
I/O direction	read only write only readÐwrite	CD-ROM graphics controller disk

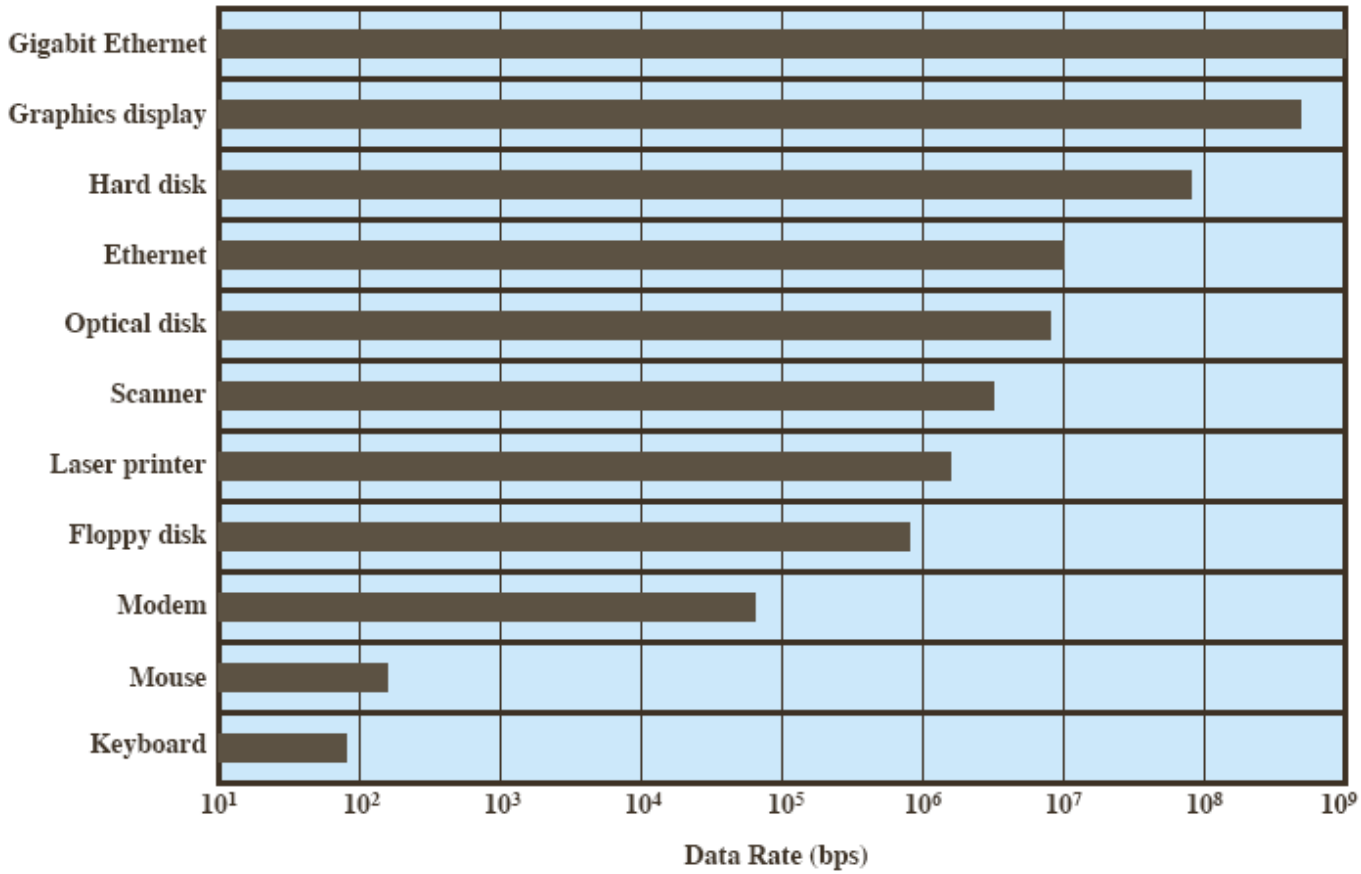
Characteristics of I/O devices

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

5.b Principles of I/O Hardware

The diversity of I/O devices

- I/O devices vary hugely in data transfer speed



Typical I/O device data rates

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

5.b Principles of I/O Hardware

The diversity of I/O devices

Device	Data rate
Keyboard	10 bytes/sec
Mouse	100 bytes/sec
56K modem	7 KB/sec
Telephone channel	8 KB/sec
Dual ISDN lines	16 KB/sec
Laser printer	100 KB/sec
Scanner	400 KB/sec
Classic Ethernet	1.25 MB/sec
USB (Universal Serial Bus)	1.5 MB/sec
Digital camcorder	4 MB/sec
IDE disk	5 MB/sec
40x CD-ROM	6 MB/sec
Fast Ethernet	12.5 MB/sec
ISA bus	16.7 MB/sec
EIDE (ATA-2) disk	16.7 MB/sec
FireWire (IEEE 1394)	50 MB/sec
XGA Monitor	60 MB/sec
SONET OC-12 network	78 MB/sec
SCSI Ultra 2 disk	80 MB/sec
Gigabit Ethernet	125 MB/sec
Ultrium tape	320 MB/sec
PCI bus	528 MB/sec
Sun Gigaplane XB backplane	20 GB/sec

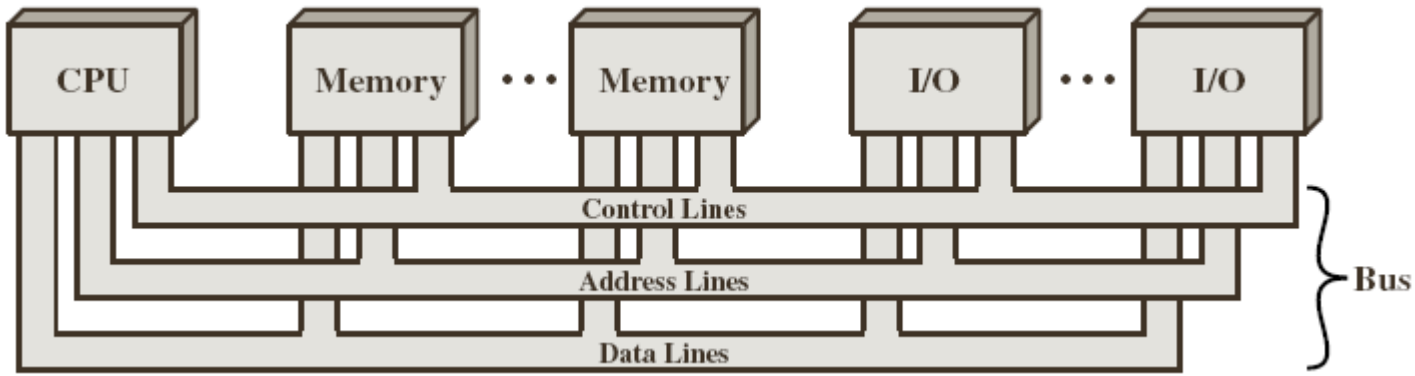
Tanenbaum, A. S. (2001)
Modern Operating Systems (2nd Edition).

Some typical device, network, and bus data rates

5.b Principles of I/O Hardware

I/O bus architecture

- CPU, memory and I/O devices communicate via buses
  - ✓ a system bus typically consists of 50 to hundreds of data lines, address lines, and control lines
  - ✓ each line carries only 1 bit at a time, therefore the bus width *and* frequency are key factors in performance



Stallings, W. (2006) *Computer Organization & Architecture: Designing for Performance (7th Edition).*

Bus interconnection scheme

5.b Principles of I/O Hardware

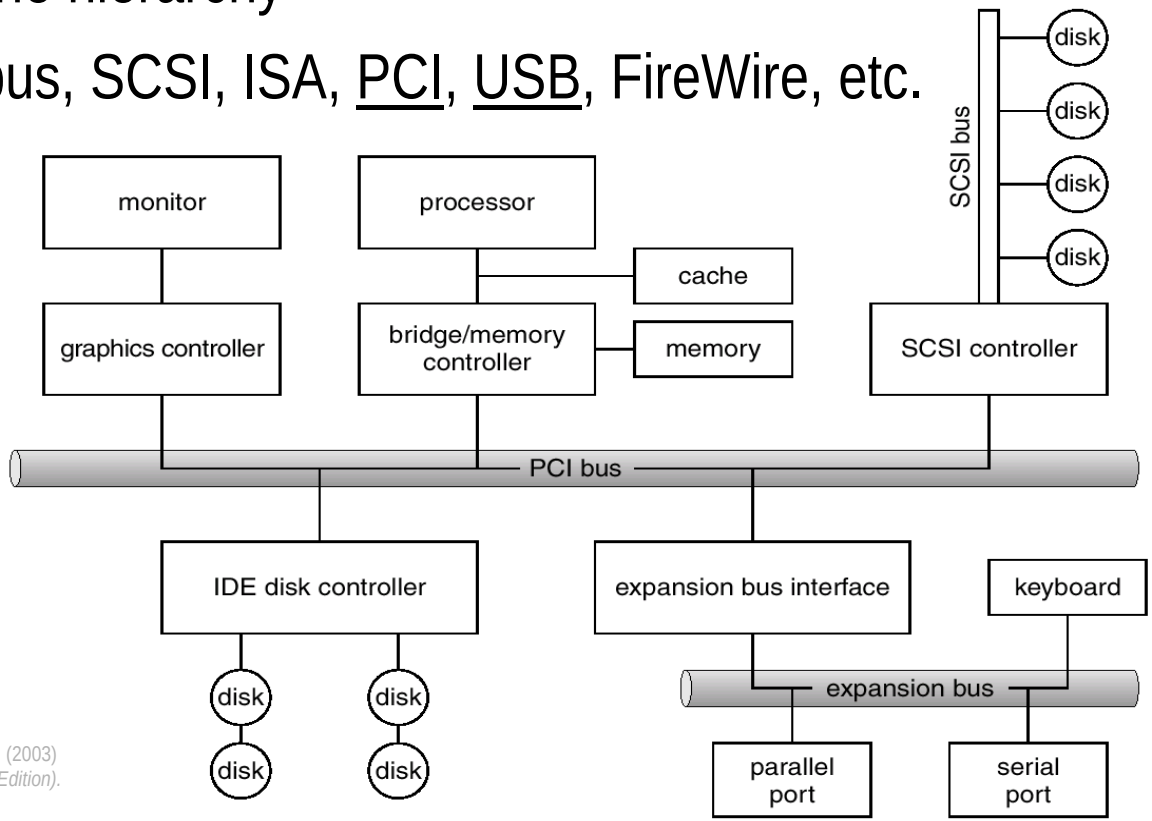
I/O devices

- Typical bus structure
  - ✓ data lines
    - provide a path for moving data between system modules
  - ✓ address lines
    - used to designate the source or destination of the data
  - ✓ control lines
    - transmit commands and timing information between modules
    - memory read/write, I/O read/write, bus request/grant, etc.

5.b Principles of I/O Hardware

I/O bus architecture

- Typical bus interconnection layout
  - ✓ computer systems contain multiple types of buses at different levels of the hierarchy
  - ✓ memory bus, SCSI, ISA, PCI, USB, FireWire, etc.

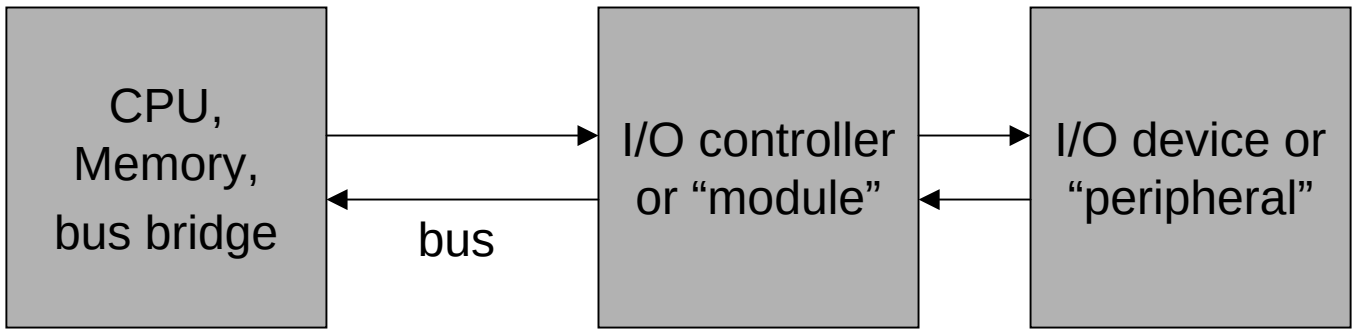


Silberschatz, A., Galvin, P. B. and Gagne. G. (2003) *Operating Systems Concepts with Java (6th Edition).*

5.b Principles of I/O Hardware

I/O bus architecture

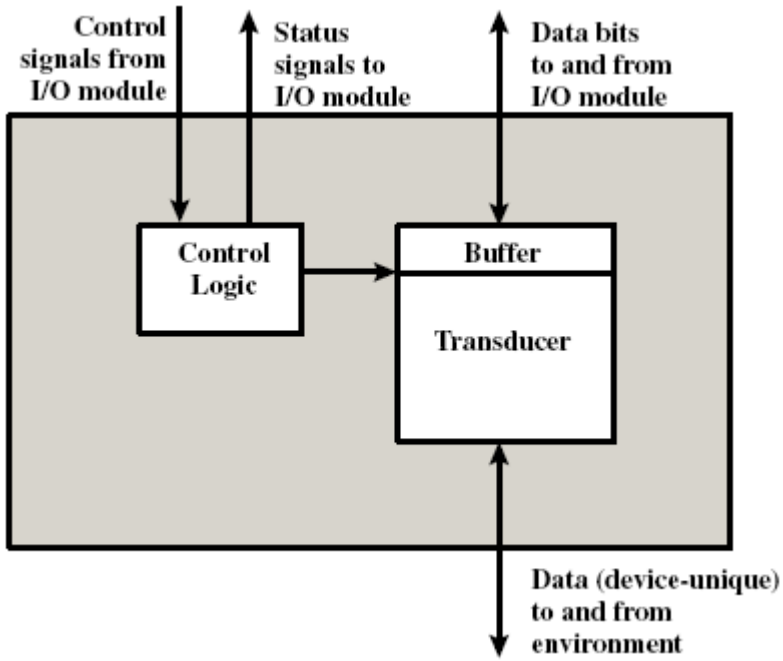
- Basic hardware I/O communication architecture



5.b Principles of I/O Hardware

I/O devices & modules

- Schematic structure of an I/O device
  - ✓ interface to the I/O module
  - ✓ interface with the physical/electrical apparatus



Stallings, W. (2006) *Computer Organization & Architecture: Designing for Performance (7th Edition).*

Block diagram of an I/O device

5.b Principles of I/O Hardware

I/O devices & modules

- Typical I/O interface with the host (via the I/O module)
  - ✓ control registers
    - can be *written* by the host to start a command or change the mode of the device
  - ✓ status registers
    - contain bits *read* by the host that indicate whether a command has completed, a byte is available to be read from the data-in register, or there has been a device error
  - ✓ data registers (buffer)
    - data-in registers are read by the host to get input
    - data-out registers are written by the host to send output

5.b Principles of I/O Hardware

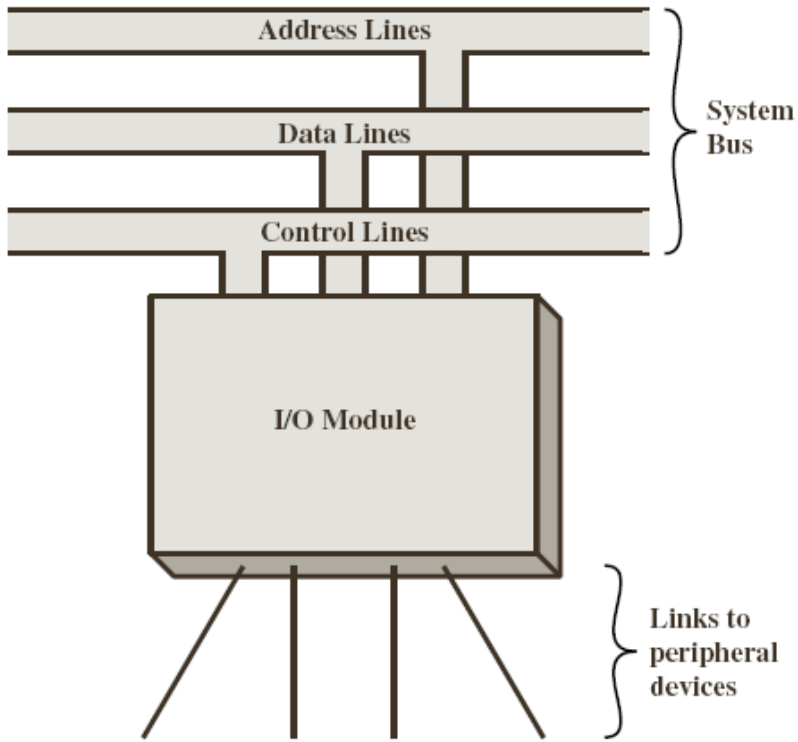
I/O devices & modules

- I/O interface with the physical/electrical apparatus
  - ✓ transducer
    - converts analog electro-mechanical events (specific to the device) into binary data

5.b Principles of I/O Hardware

I/O devices & modules

- I/O controllers or "modules"
  - ✓ intermediate between the I/O device (peripheral) and CPU or memory



Generic model of an I/O module

Stallings, W. (2006) *Computer Organization & Architecture: Designing for Performance (7th Edition).*

5.b Principles of I/O Hardware

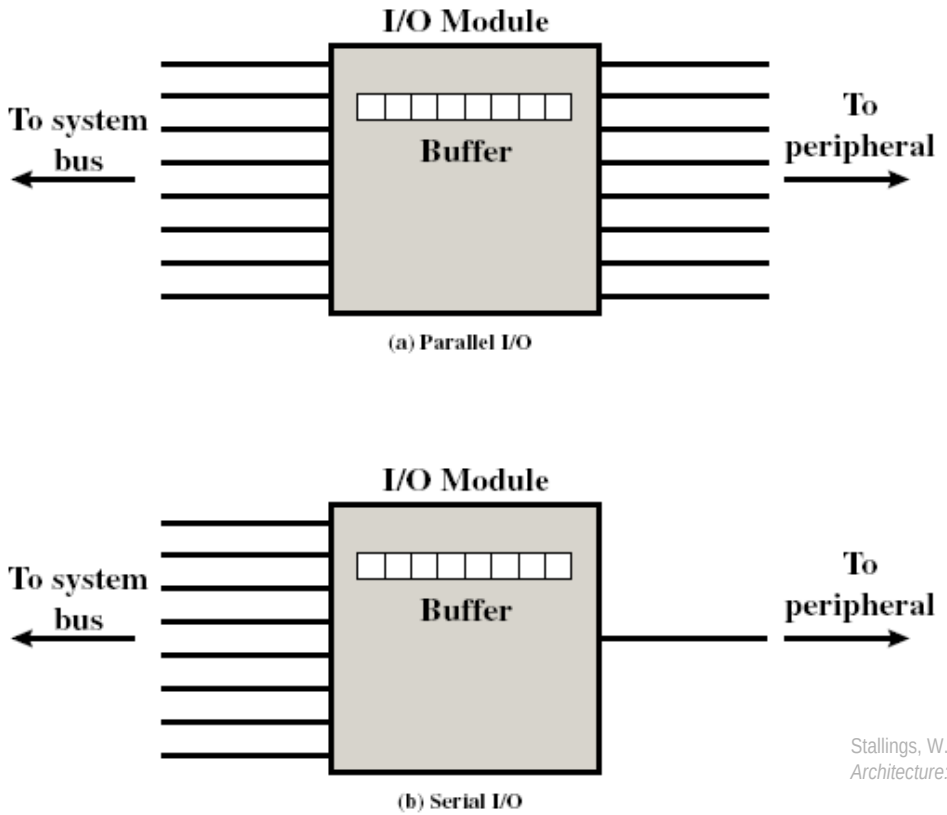
I/O devices & modules

- Why I/O modules? Why not connecting the devices directly to the bus?
  - ✓ wide variety of peripherals with various operation methods: don't want to incorporate heterogeneous logic into CPU
  - ✓ modules offer a more unified hardware command interface
  - ✓ data transfer rate slower or faster than memory or CPU
  - ✓ different data and word lengths
  - ✓ multiplexing
- Functions of an I/O module
  - ✓ interface to CPU and memory via system bus
  - ✓ interface to one *or more* peripherals by custom data links

5.b Principles of I/O Hardware

I/O devices & modules

- Example of I/O module-device interface



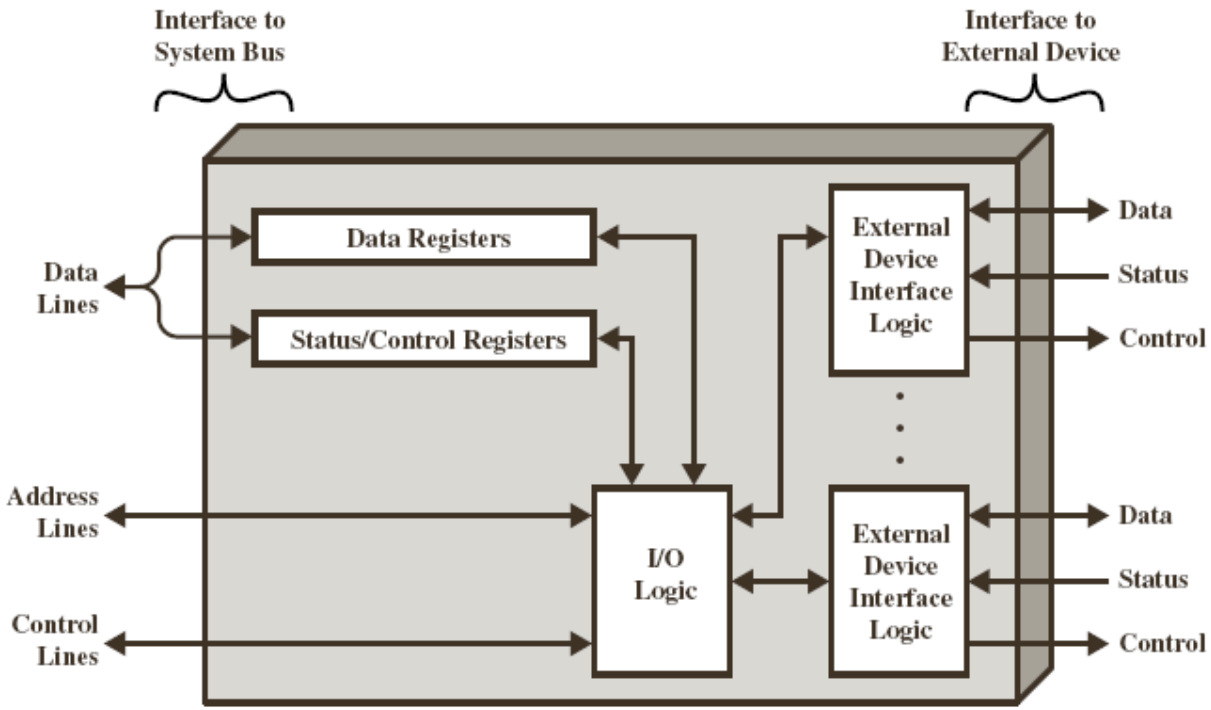
Stallings, W. (2006) *Computer Organization & Architecture: Designing for Performance (7th Edition).*

Parallel and serial I/O

5.b Principles of I/O Hardware

I/O devices & modules

- Schematic structure of an I/O module
  - ✓ interface also based on control, status and data lines
  - ✓ basically an adapter/multiplexer



Block diagram of an I/O module

Stallings, W. (2006) *Computer Organization & Architecture: Designing for Performance (7th Edition).*

5.b Principles of I/O Hardware

CPU-I/O communication

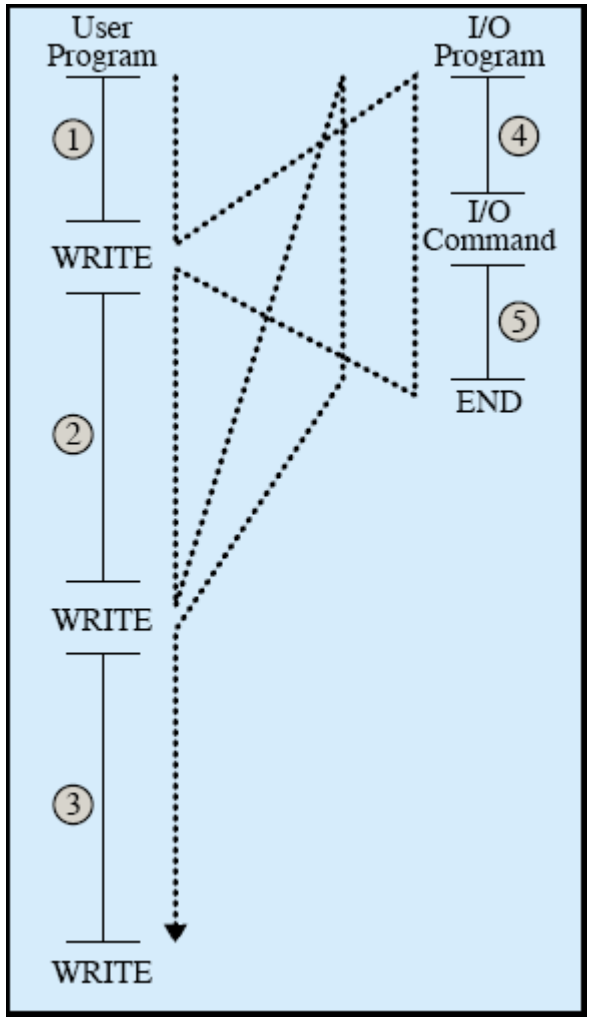
- Three communication protocols between CPU and I/O
  1. Programmed I/O ("busy waiting")
     - the CPU must repeatedly poll the device to check if the I/O request completed
  2. Interrupt-driven I/O
     - the CPU can switch to other tasks and is (frequently) interrupted by the I/O device
  3. Direct Memory Access (DMA)
     - the CPU is involved only at the start and the end of the whole transfer; it delegates control to the I/O controller that accesses memory directly without bothering the CPU

5.b Principles of I/O Hardware

CPU-I/O communication

1. Programmed I/O

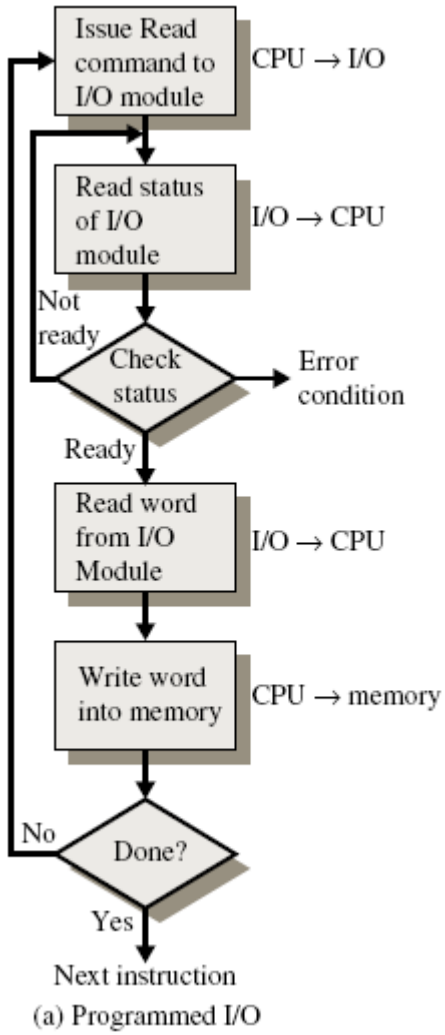
- ✓ the CPU issues an I/O command (on behalf of a process) to an I/O module
- ✓ the CPU (that process) then busy waits for completion before proceeding
- ✓ also called “busy waiting” or “polling”



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

5.b Principles of I/O Hardware

CPU-I/O communication



(a) Programmed I/O

Stallings, W. (2006) *Computer Organization & Architecture: Designing for Performance (7th Edition).*

5.b Principles of I/O Hardware

CPU-I/O communication

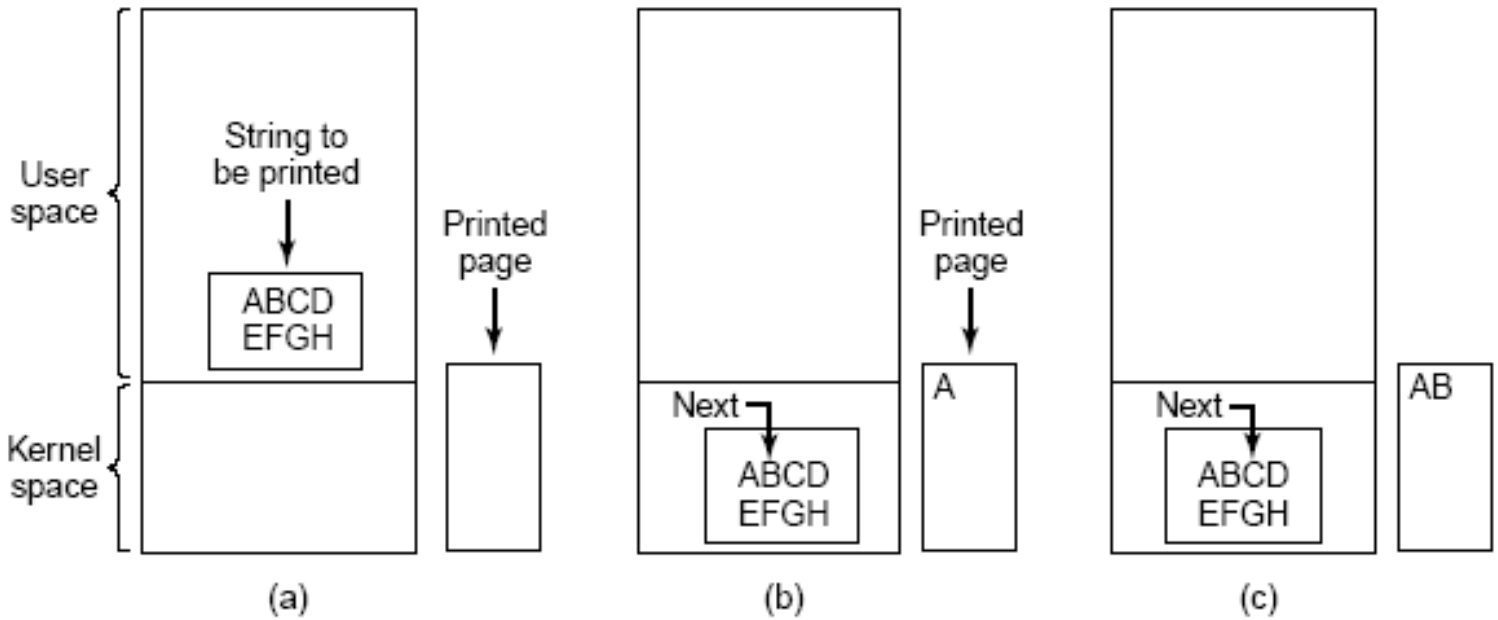
1. Programmed I/O

- ✓ basic handshake protocol between CPU and I/O module
  a. host repeatedly polls *occupied* bit of I/O module until cleared (this is the busy waiting part)
  b. host sets *write* bit in *command* register and writes byte into *data-out* register
  c. host sets *command-ready* bit
  d. module notices *command-ready* bit and sets *occupied* bit
  e. module reads *command* and *data-out* registers and orders device to perform I/O
  f. when suceeded, module clears *command-ready* bit, *error* bit and *occupied* bit

5.b Principles of I/O Hardware

CPU-I/O communication

- Example: writing a string to the printer



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

Steps in printing a string

5.b Principles of I/O Hardware

CPU-I/O communication

- Example: writing a string to the printer

1. . . . using programmed I/O:

```
copy_from_user(buffer, p, count);          /* p is the kernel bufer */
for (i = 0; i < count; i++) {              /* loop on every character */
      while (*printer_status_reg != READY) ;/* loop until ready */
      *printer_data_register = p[i];       /* output one character */
}
return_to_user( );
```

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

Programmed I/O code

5.b Principles of I/O Hardware

CPU-I/O communication

1. Programmed I/O problems

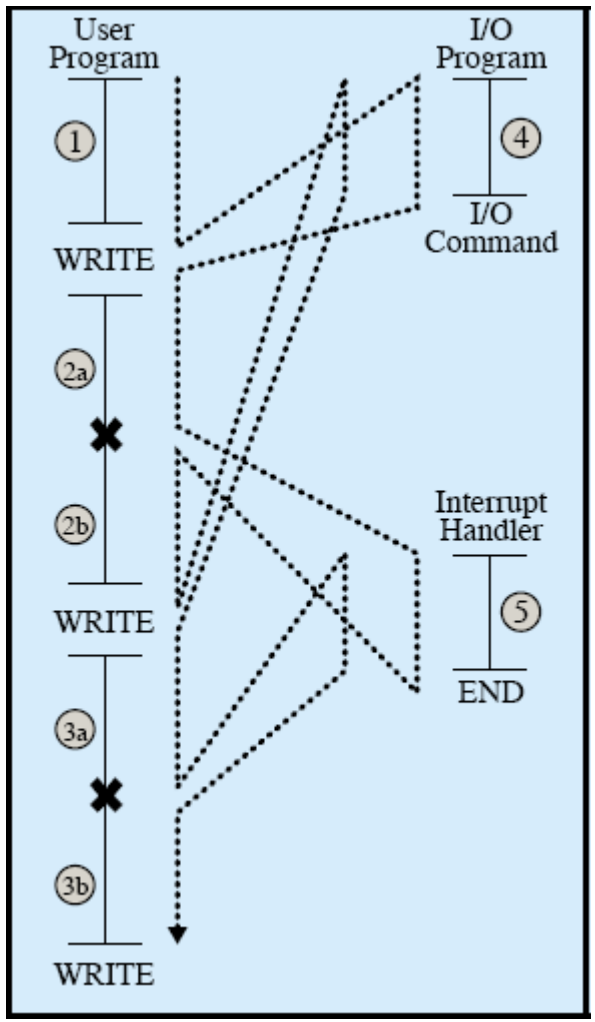
- ✓ the I/O device (module) is passive and needy
- ✓ the CPU needs to continually check the I/O status
  - to minimize the CPU waiting time
  - but also to avoid overflow in the small buffer of the controller: needs to be regularly cleared
- ✓ naturally this is a waste of CPU time if the I/O transfer is slower. . . which it always is
- ✓ no longer an option today

5.b Principles of I/O Hardware

CPU-I/O communication

2. Interrupt-driven I/O

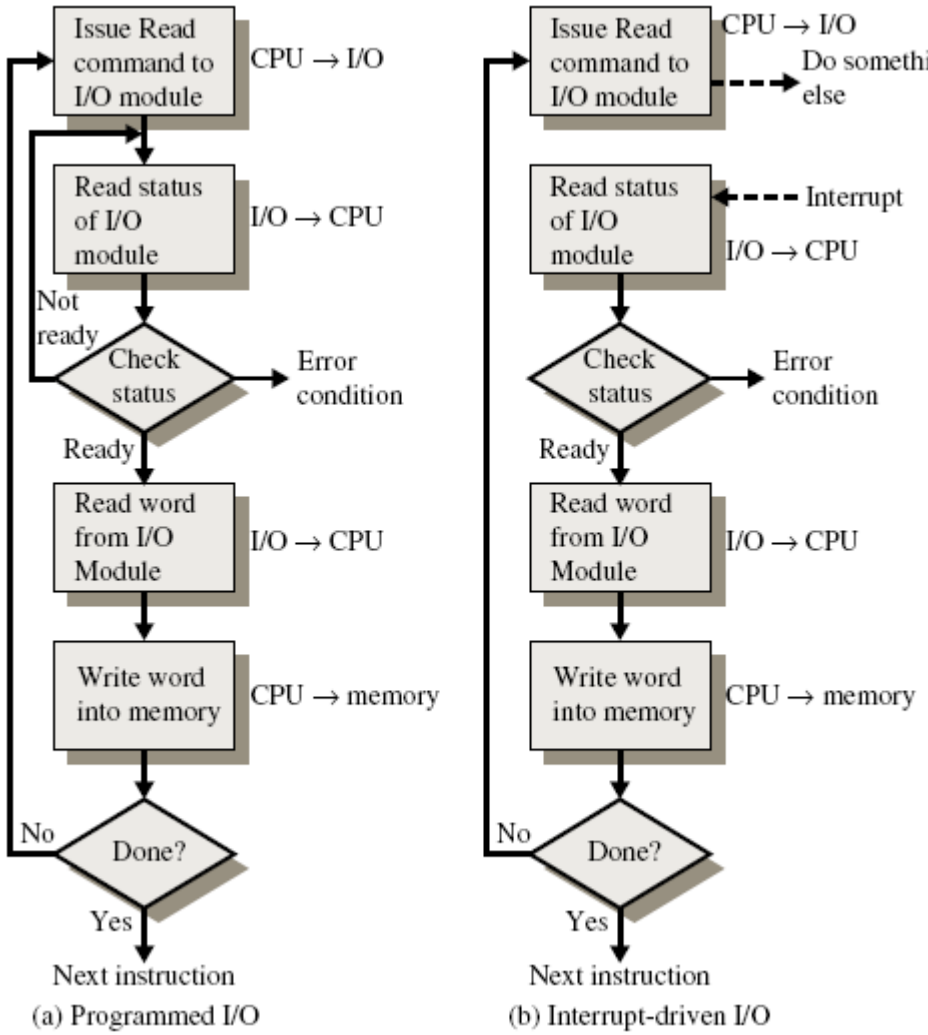
- ✓ the CPU issues an I/O command (on behalf of a process) to an I/O module
- ✓ . . . but does not wait for completion; instead, it continues executing subsequent instructions
- ✓ then, later, it is interrupted by the I/O module when work is complete
- ✓ note: the subsequent instructions may be in the same process or not, depending whether I/O was requested asynchronously or not: *process wait ≠ CPU wait!*



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

5.b Principles of I/O Hardware

CPU-I/O communication



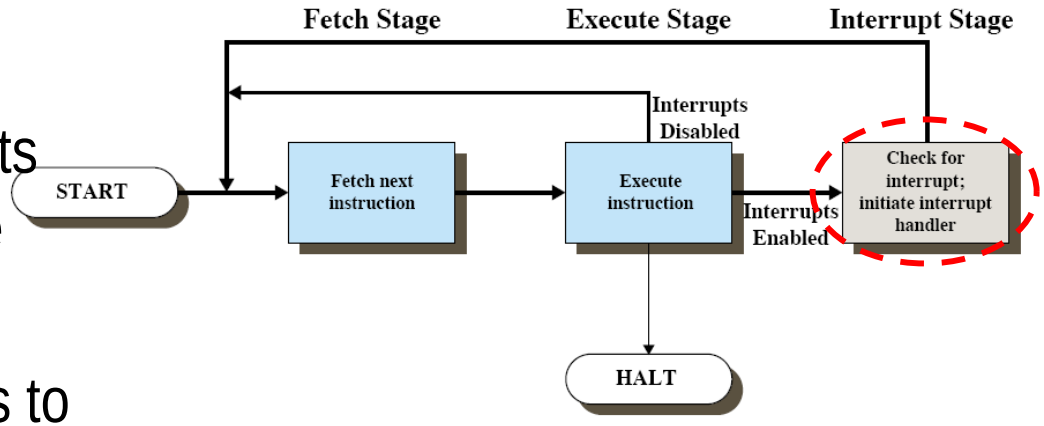
Stallings, W. (2006) *Computer Organization & Architecture: Designing for Performance (7th Edition).*

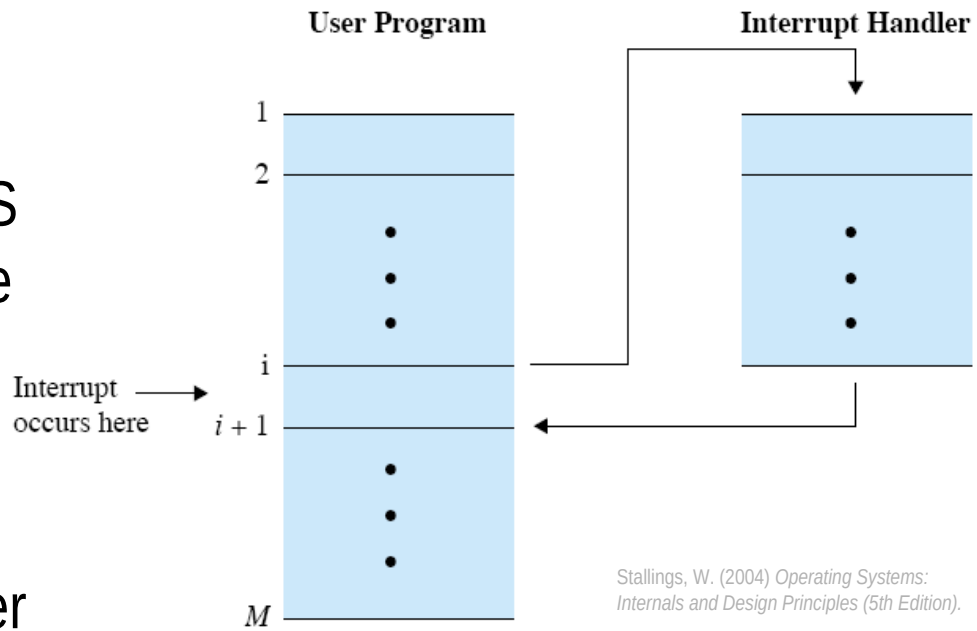
5.b Principles of I/O Hardware

CPU-I/O communication

2. Interrupt-driven I/O

- ✓ CPU senses interrupts in a third stage of the fetch/execute cycle
- ✓ control (PC) transfers to an interrupt handler in kernel space,
- ✓ which branches to O/S routines specific to the type of interrupt;
- ✓ the CPU is eventually returned to this user program . . . or another





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

5.b Principles of I/O Hardware

CPU-I/O communication

- Example: writing a string to the printer

2. . . . using interrupts:

only for the first character

```
copy_from_user(buffer, p, count);
enable_interrupts( );
while (*printer_status_reg != READY) ;
*printer_data_register = p[0];
scheduler( );
```

(a)

```
if (count == 0) {
    unblock_user( );
} else {
    *printer_data_register = p[i];
    count = count – 1;
    i = i + 1;
}
acknowledge_interrupt( );
return_from_interrupt( );
```

(b)

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

Interrupt-driven I/O code: (a) system call and (b) interrupt service procedure

5.b Principles of I/O Hardware

CPU-I/O communication

2. Interrupt-driven I/O

- ✓ relies on an efficient hardware mechanism that saves a small amount of CPU state, then calls a privileged kernel routine
- ✓ note that this hardware mechanism is put to good use by the O/S for other events:
  - in virtual memory paging, a page fault is an exception that raises an interrupt
  - system calls execute a special instruction (TRAP), which is a software interrupt

5.b Principles of I/O Hardware

CPU-I/O communication

2. Interrupt-driven I/O problems

- ✓ the I/O device (module) is more active but still very needy
- ✓ wasteful to use an expensive general-purpose CPU to feed a controller 1 byte at a time

5.b Principles of I/O Hardware

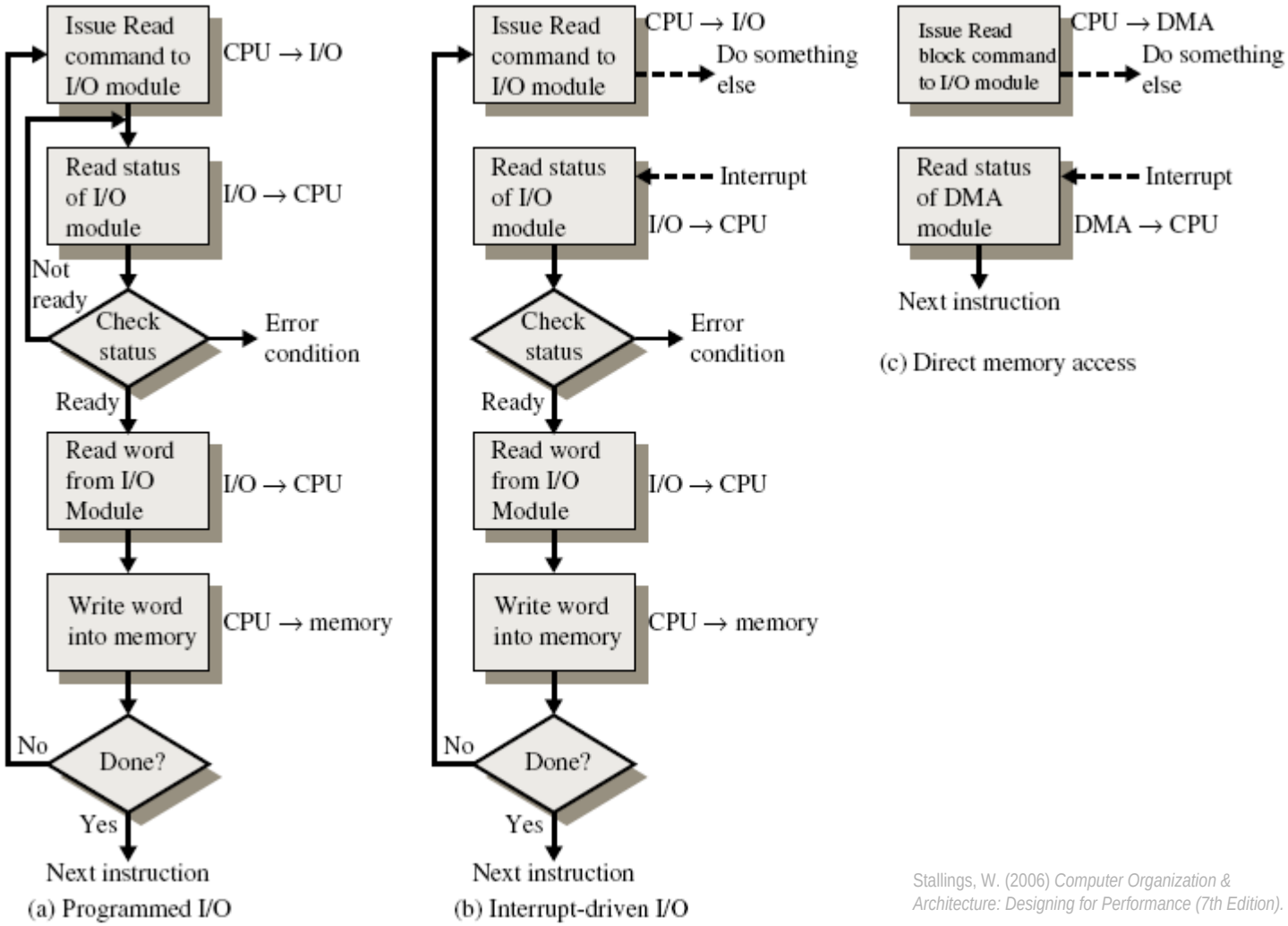
CPU-I/O communication

3. Direct Memory Access (DMA)

- ✓ avoids programmed/interrupted I/O for large data movement
- ✓ requires a special-purpose processor called DMA controller
- ✓ bypasses CPU to transfer data directly between I/O device and memory
- ✓ the handshaking is performed between the DMA controller and the I/O module
- ✓ only when the entire transfer is finished does the DMA controller interrupt the CPU

5.b Principles of I/O Hardware

CPU-I/O communication

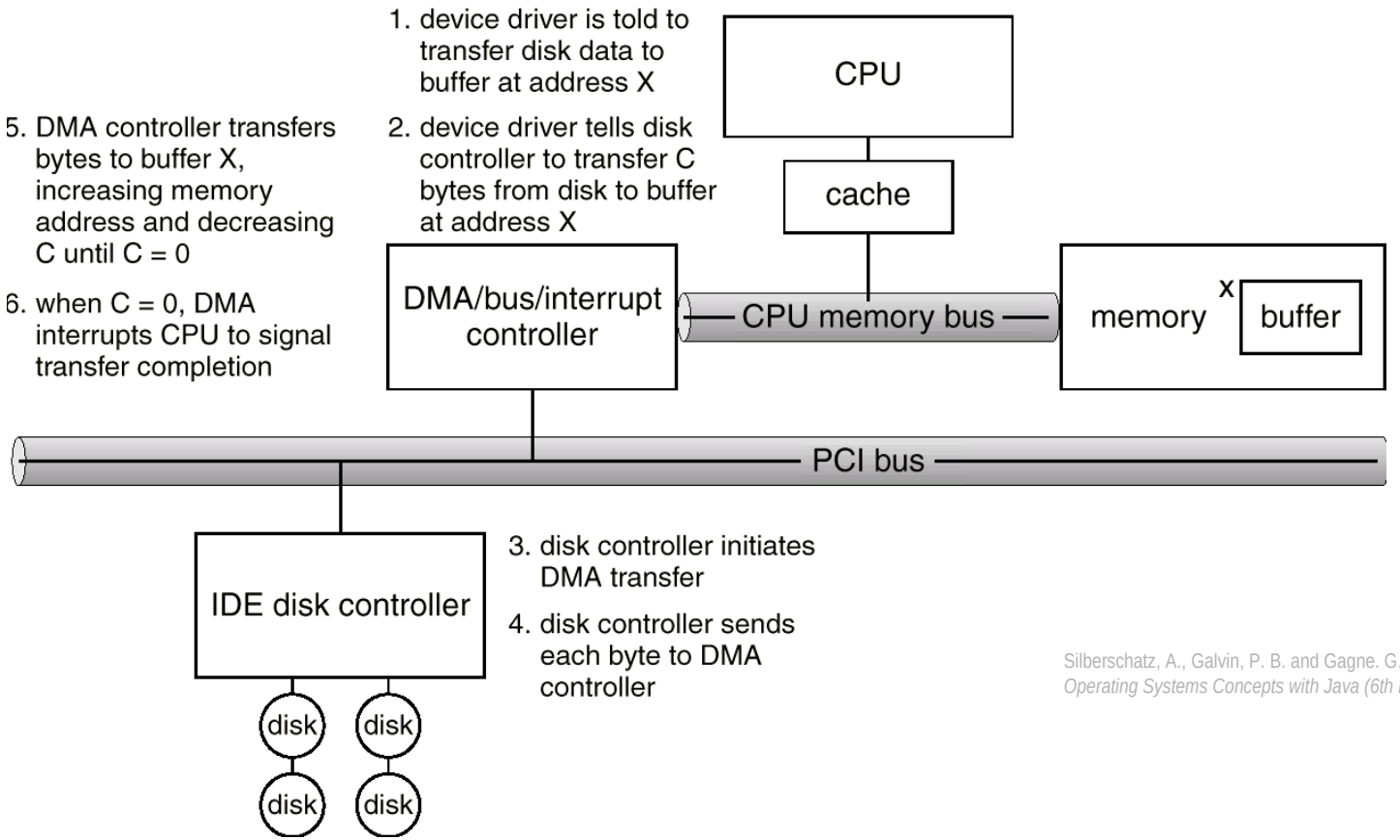


Stallings, W. (2006) *Computer Organization & Architecture: Designing for Performance (7th Edition).*

5.b Principles of I/O Hardware

CPU-I/O communication

3. Direct Memory Access (DMA)



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

Steps in a DMA transfer

5.b Principles of I/O Hardware

CPU-I/O communication

- Example: writing a string to the printer

3. . . . using DMA:

```
copy_from_user(buffer, p, count);
set_up_DMA_controller( );
scheduler( );
```

(a)

```
acknowledge_interrupt( );
unblock_user( );
return_from_interrupt( );
```

(b)

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

DMA-supported I/O code: (a) system call and (b) interrupt service procedure

5.b Principles of I/O Hardware

CPU-I/O communication

- Summary

	No Interrupts	Use of Interrupts
I/O-to-memory transfer through processor	Programmed I/O	Interrupt-driven I/O
Direct I/O-to-memory transfer		Direct memory access (DMA)

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

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

- ✓ The diversity of I/O devices
- ✓ I/O bus architecture
- ✓ I/O devices & modules
- ✓ CPU-I/O communication

c. I/O Software Layers

d. Disk Management