I/O systems

Operating Systems Design

References

- Silberschatz ***Operating System Concepts Ninth Edition*** 2012. Chapters 13 I/O systems

Objectives

- Explore the structure of an operating system's I/O subsystem
- Discuss the principles of I/O hardware and its complexity
- Provide details of the performance aspects of I/O hardware and software

Quiz

- What are I/O system goals?

a) To offer users a simplified logical view.

b) To optimize I/O.

c) To facilitate peripherals management.

d) All of them.

Contents

- I/O Hardware
- I/O Modules
- I/O Techniques

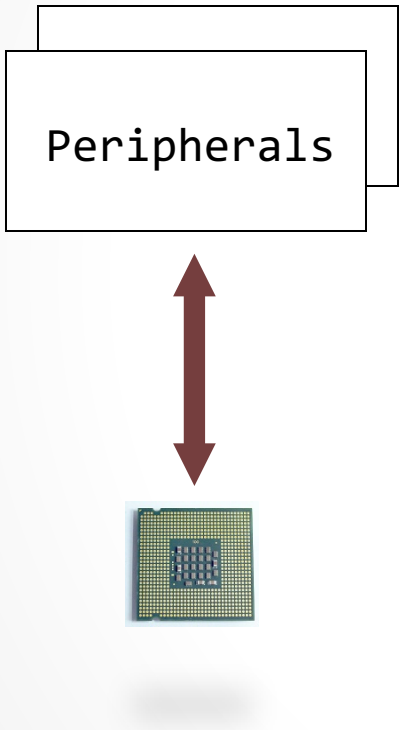
Overview

- I/O management is a major component of operating system design and operation
  - Important aspect of computer operation
  - I/O devices vary greatly
  - Various methods to control them
  - Performance management
  - New types of devices frequent
- Ports, buses, device controllers connect to various devices
- **Device drivers** encapsulate device details
  - Present uniform device-access interface to I/O subsystem

I/O Hardware

- High variety of I/O devices
  - Storage
  - Transmission
  - Human-interface
- Common concepts - signals from I/O devices interface with computer
  - **Port** - connection point for device
  - **Bus** - **daisy chain** or shared direct access
  - **Controller** (**host adapter**) - electronics that operate port, bus, device
    - Sometimes integrated
    - Sometimes separate circuit board (host adapter)
    - Contains processor, microcode, private memory, bus controller, etc

Peripheral concept



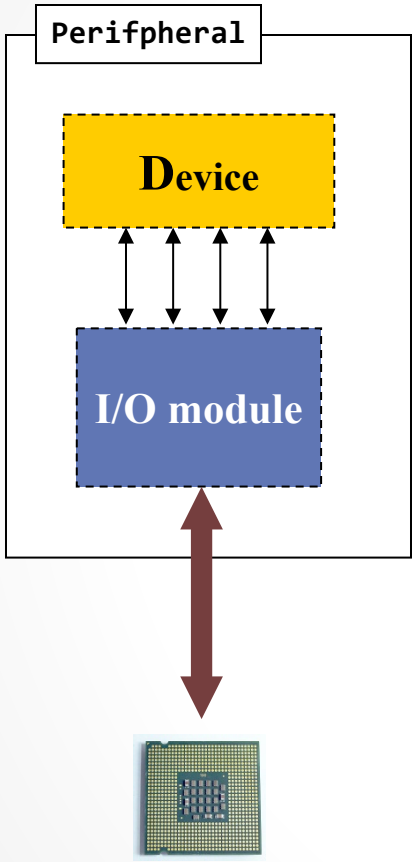
- **Peripheral**:
  - An external device connected to the CPU through I/O modules.
  - Functions: information storage or connecting the computer with the outer world.

Peripheral classification (by usage)

- **Comunication**:
  - Human - machine
    - (Terminal) keboard, mouse, …
    - (Print) plotter, scanner, …
  - Machine- machine(modem, …)
  - Physical environment- Machine
    - (Lectura/accionamiento) x (analógico/digital)

- **Storage**:
  - Direct access (Disks, DVD, …)
  - Sequential access (Tape, printer,..)

General structure of a peripheral



Composed of:

- Device
  - Hardware interacting with the environment
- I/O module
  - Controller
  - Interface between device and CPU
  - Hides device paticularities

Peripheral = Device + Controller

Quiz

- Which one is a block device.

a) Mouse

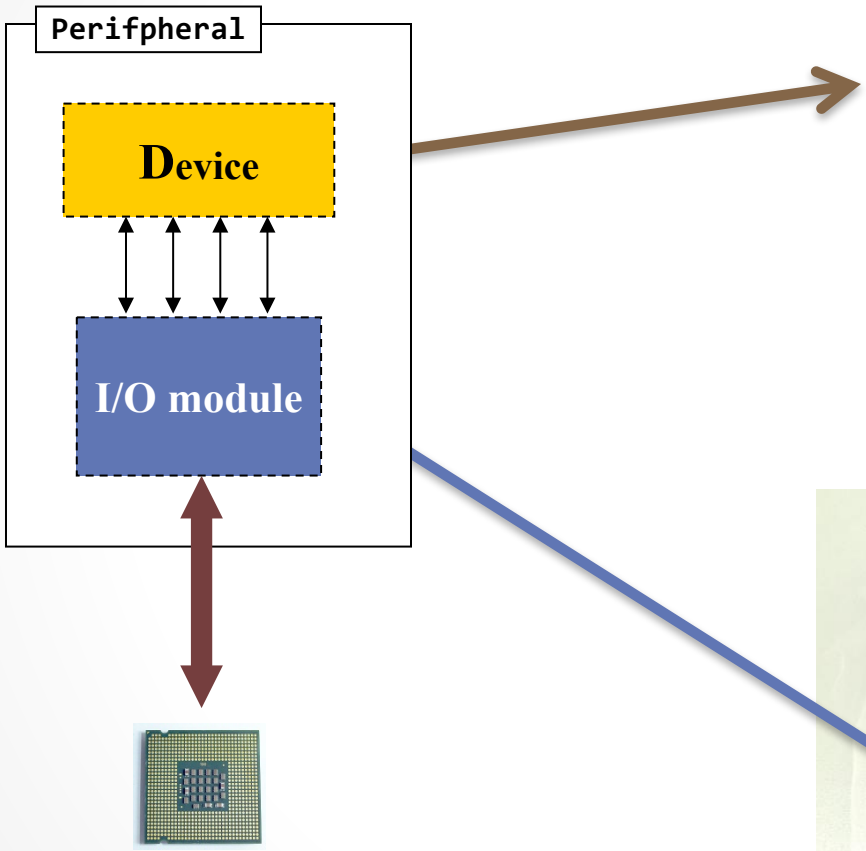
b) Disk

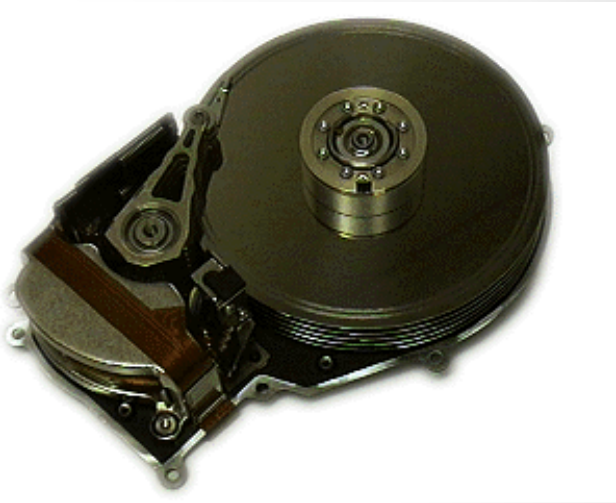
c) Keyboard

d) Display

Example

Hard disk





Quiz

- What are the characteristics of a disk?

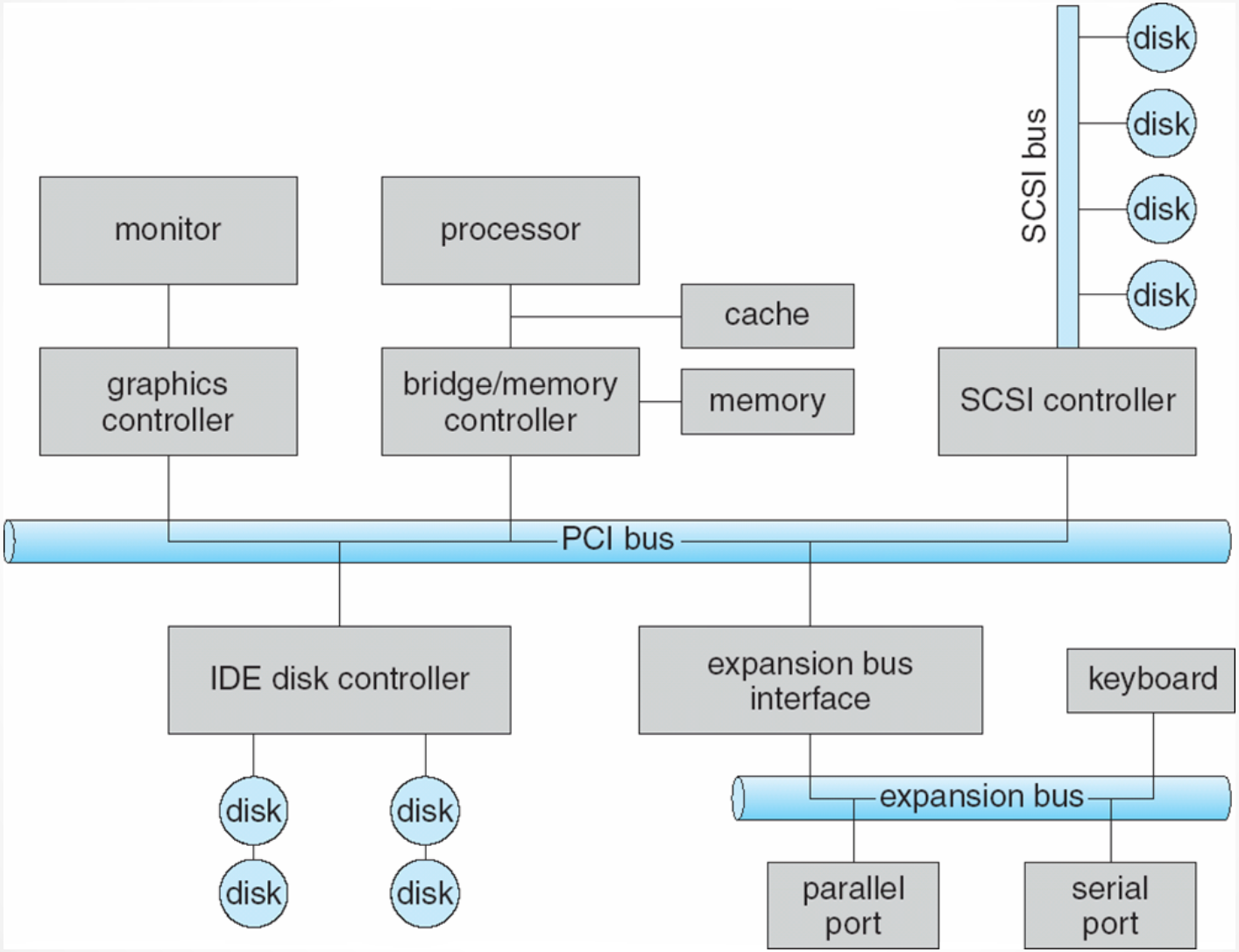
a) Block device, Sharable, Read-only

b) Char device, Sharable, Read-only

c) Block device, Random, Read-write

d) Block device, Dedicated, Read-write.

A Typical PC Bus Structure



Quiz

- A traditional PCI bus is:

a) A bus interconnecting CPU and memory

b) A bus for connecting I/O devices

c) A SAN bus

d) A bus interconnecting CPU with its caches.

I/O Hardware (Cont.)

- I/O instructions control devices
- Devices usually have registers where device driver places commands, addresses, and data to write, or read data from registers after command execution
  - Data-in register, data-out register, status register, control register
  - Data registers have typically 1-4 bytes, or FIFO buffer
- Devices have addresses, used by
  - Direct I/O instructions
    - Trigger bus lines to select proper devices
    - Move bits into and out of device registers
  - Memory-mapped I/O
    - Device data and command registers mapped to processor address space
    - Especially for large address spaces (graphics)

Quiz

- Memory-mapped I/O can be used for communicating between a device controller and a CPU.

a) True

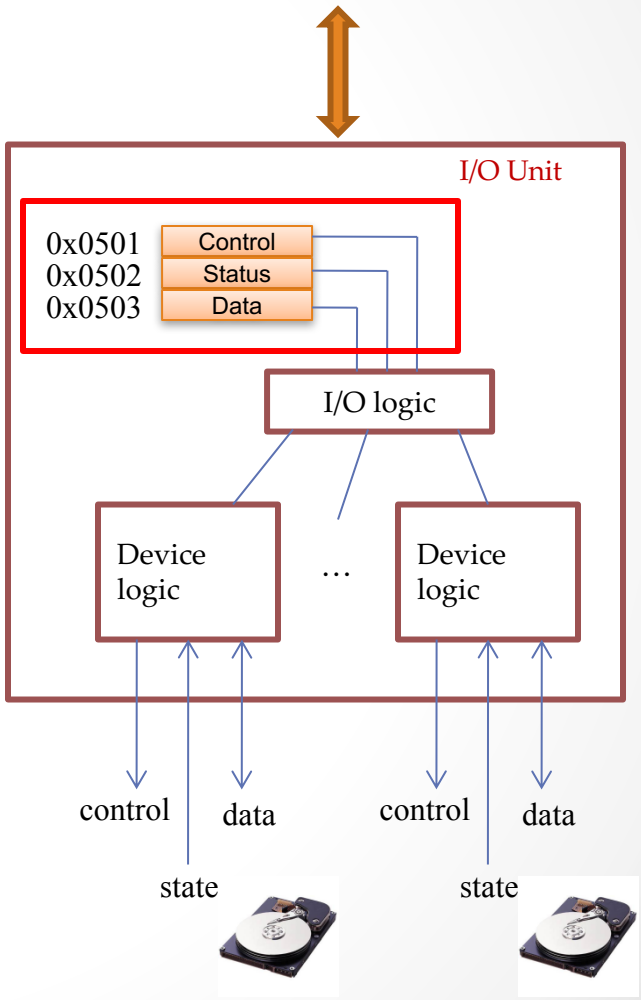
b) False

I/O Module

structure & operation

3 kind of registers:

- Control.
  - Commands for peripheral
- Status
  - Status since the last command
- data
  - Data exchange between CPU/peripheral

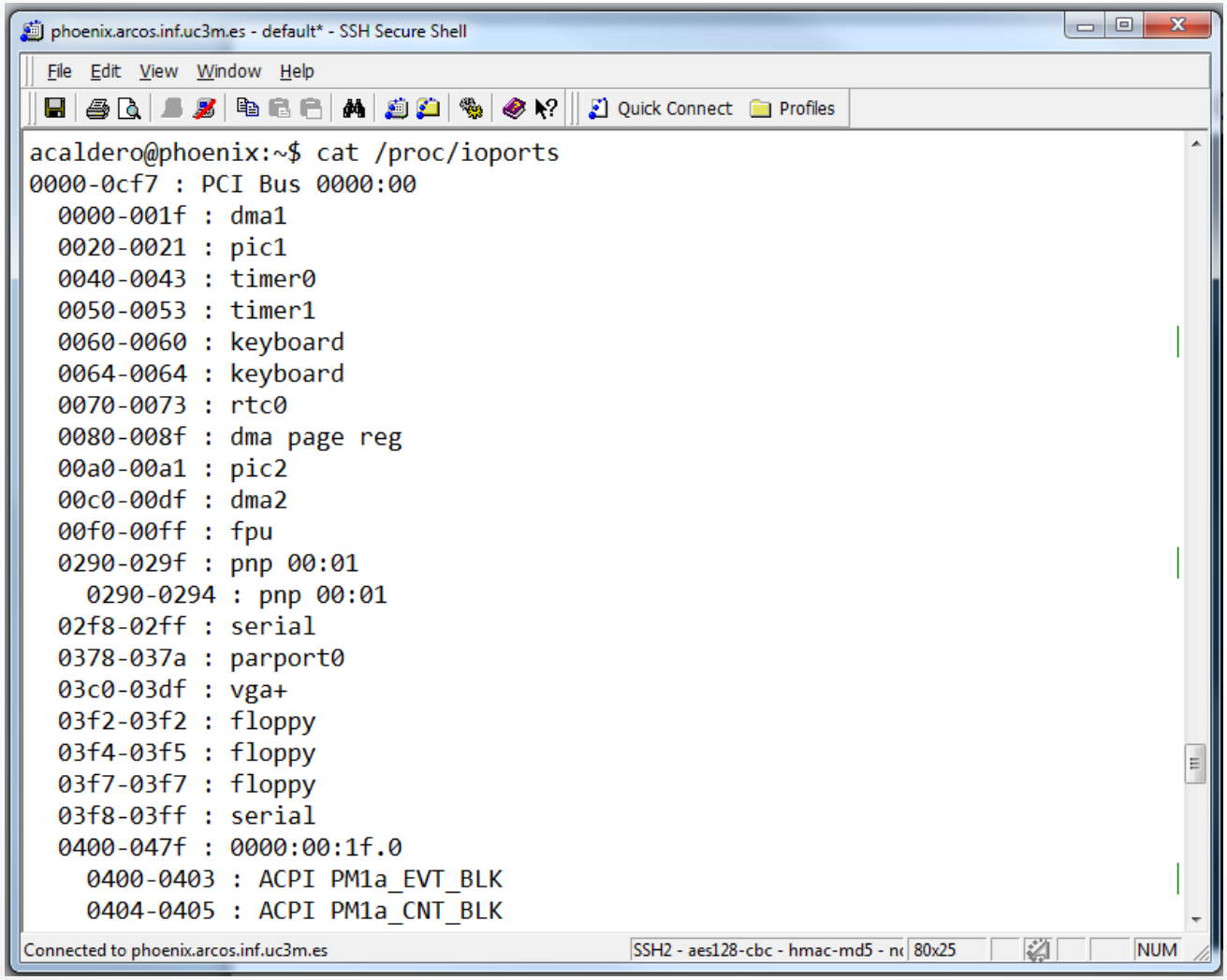


Major features:

- Transfer unit
- Adressing
- Interaction CPU-controller

I/O address

Linux



Quiz

- A port is

a) a connection point

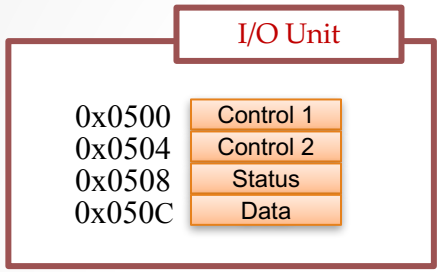
b) a socket

c) a pipe

d) a bus

Example

USB port



- control 1
  - 0: read
  - 1: write
- control 2
  - r/w offset
- Status
  - 0: busy
  - 1: ready
- Data
  - Device data

Control:

```
out(0x504,10) ;  // offset
out(0x500,0) ;    // read
```

Status:

```
do {
   in(0x508,&status) ;  // ¿ready?
} while (0 == status) ;
```

Data:

```
in(0x50C,&data[i]) ; // read data
```

Quiz

- A device controller is:

a) a driver

b) an operating system module

c) a hardware that manages a device

Interaction CPU-Controller

- **Polling (program I/O)**
  - CPU executes an I/O program: wait -> transfer
- **Interrupts**
  - CPU: only transfer, does not wait
- **I/O by DMA (direct memory access)**
  - CPU does not transfer, just notifies end of transfer
    - Sophisticated controller
    - DMA logic: counters, control signals, …

Quiz

- What is true about polling?

a) It means concurrently requesting data to multiple devices.

b) It is appropriate if the I/O device is slow.

c) It means busy waiting until the data are ready.

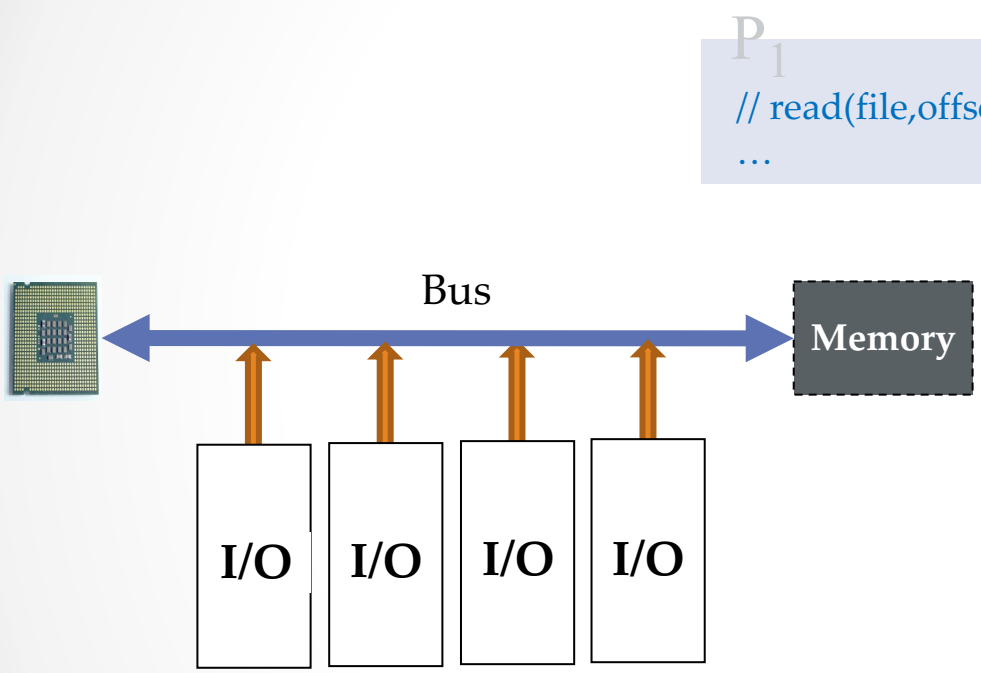
d) It means that an I/O process that directly writes data to the memory to avoid the CPU to miss the data, and then wait for them.

Polling

- For each byte of I/O
  1. Read busy bit from status register until 0
  2. Host sets read or write bit and if write copies data into data-out register
  3. Host sets command-ready bit
  4. Controller sets busy bit, executes transfer
  5. Controller clears busy bit, error bit, command-ready bit when transfer done
- Step 1 is **busy-wait** cycle to wait for I/O from device
  - Reasonable if device is fast
  - But inefficient if device slow
  - CPU switches to other tasks?
    - But if miss a cycle data overwritten / lost

Example

Polling




```
P1
// read(file,offset,100) ;
…
```

```
k
out(0x504,10) ;
out(0x500,0) ;
for (i=0; i<100;i++)  {
     do {
       in(0x508,&status) ;
     } while (0 == status) ;
     in(0x50C,&data[i]) ;
}
```

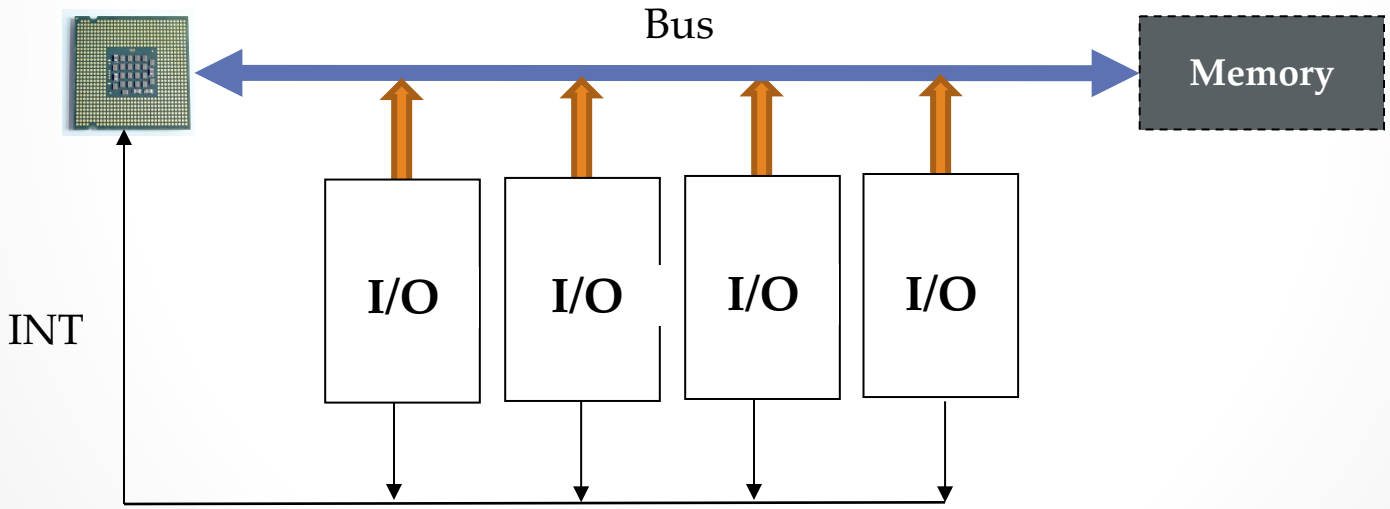
```
P1
// read(file,offset,100) ;
…
```

Interrupts

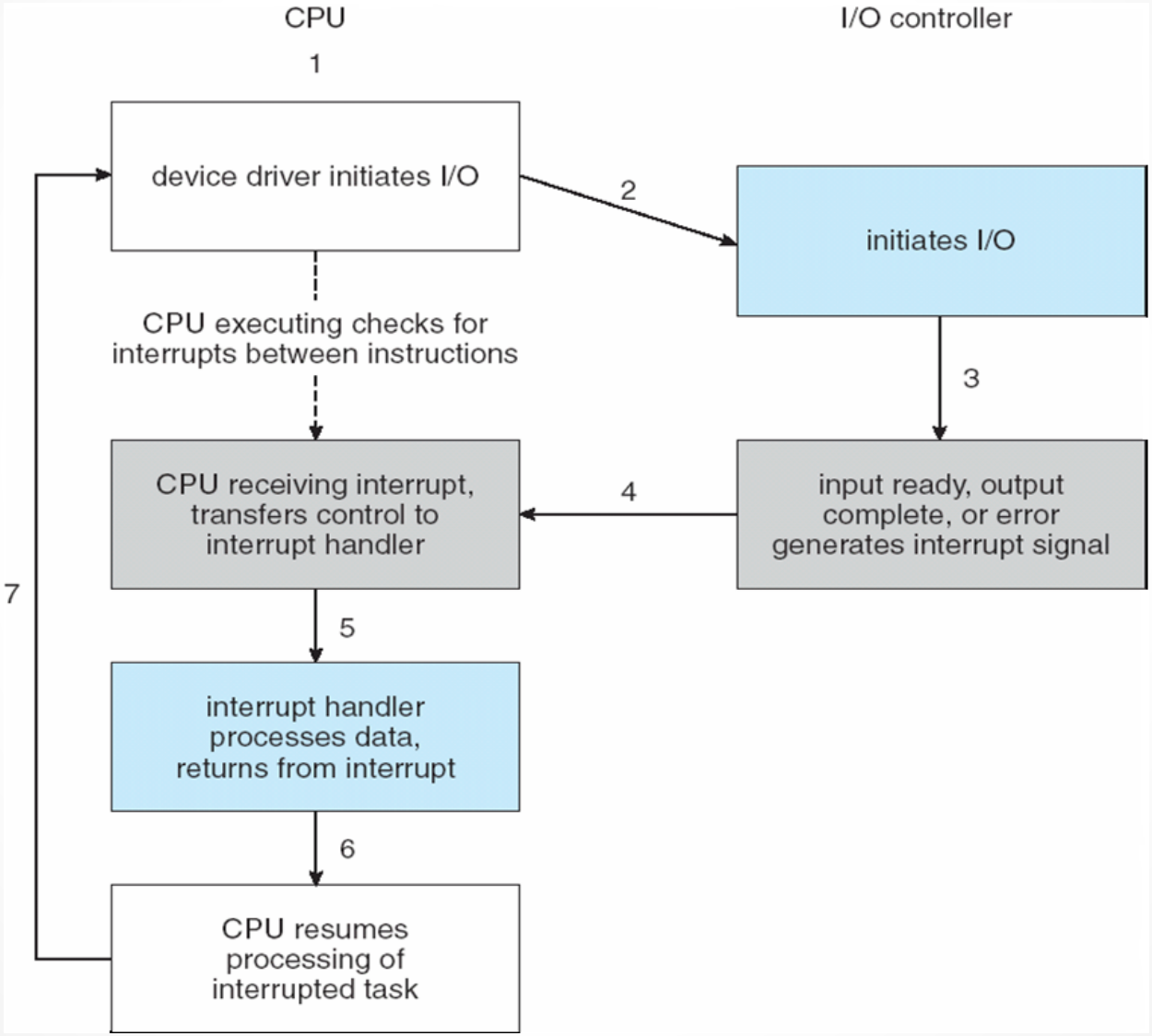
- Polling can happen in 3 instruction cycles
  - Read status, logical-and to extract status bit, branch if not zero
  - How to be more efficient if non-zero infrequently?
- CPU **interrupt-request line** triggered by I/O device
  - Checked by processor after each instruction
- Interrupt vector to dispatch interrupt to correct handler
  - Priority levels
  - Some **nonmaskable** (e.g., crytical memory errors)
  - Interrupt chaining if more than one device at same interrupt number

Example

Interrupts and I/O

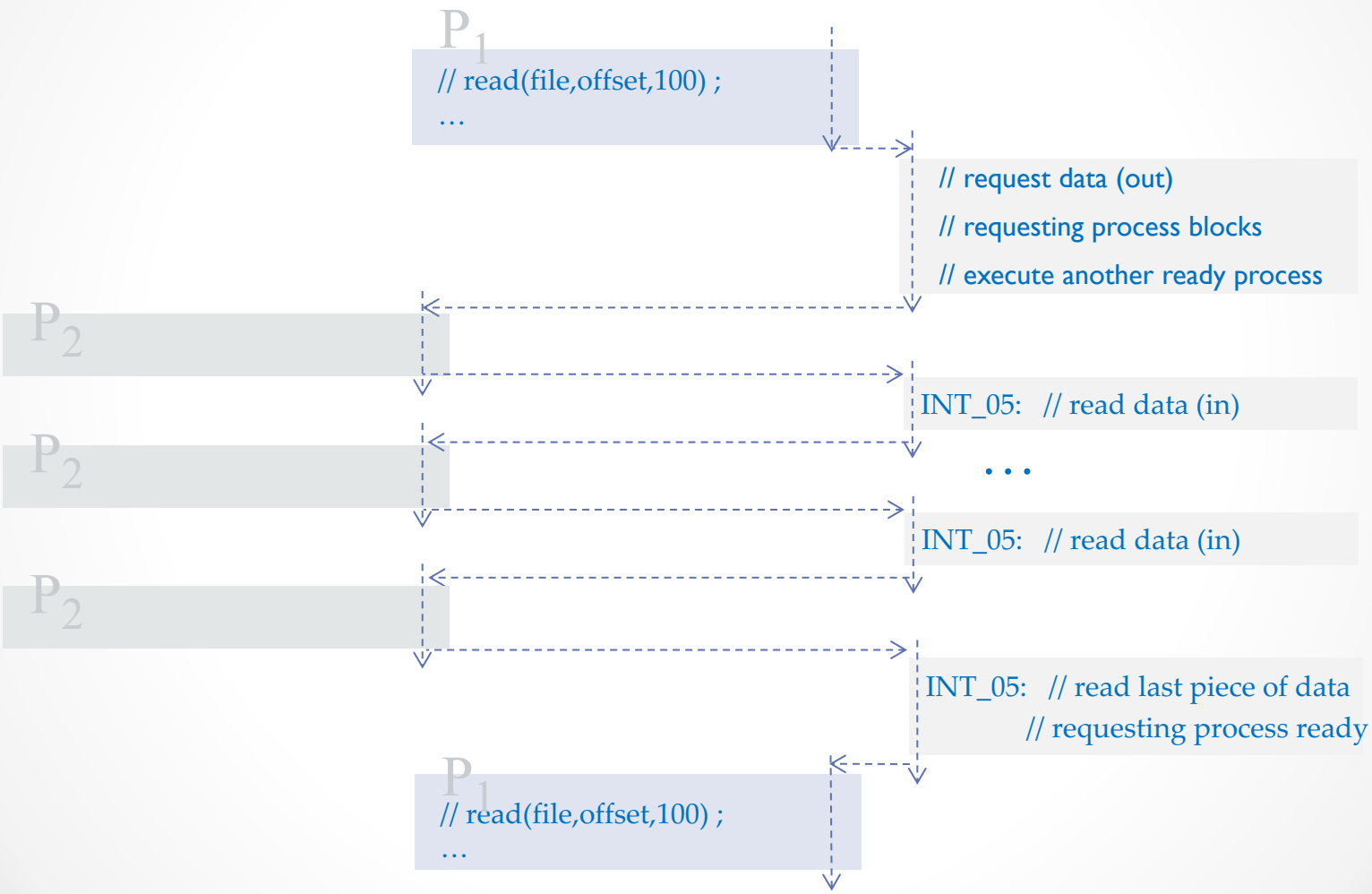


Interrupt-Driven I/O Cycle



Example

I/O with interrupts



Direct Memory Access

- Used to avoid **programmed I/O** (one byte at a time) for large data movement
- Requires **DMA** controller
- Bypasses CPU to transfer data directly between I/O device and memory
- OS writes DMA command block into memory
  - Source and destination addresses
  - Read or write mode
  - Count of bytes
  - Writes location of command block to DMA controller
  - Bus mastering of DMA controller – grabs bus from CPU
  - When done, interrupts to signal completion

Quiz

- What is the advantage of DMA?

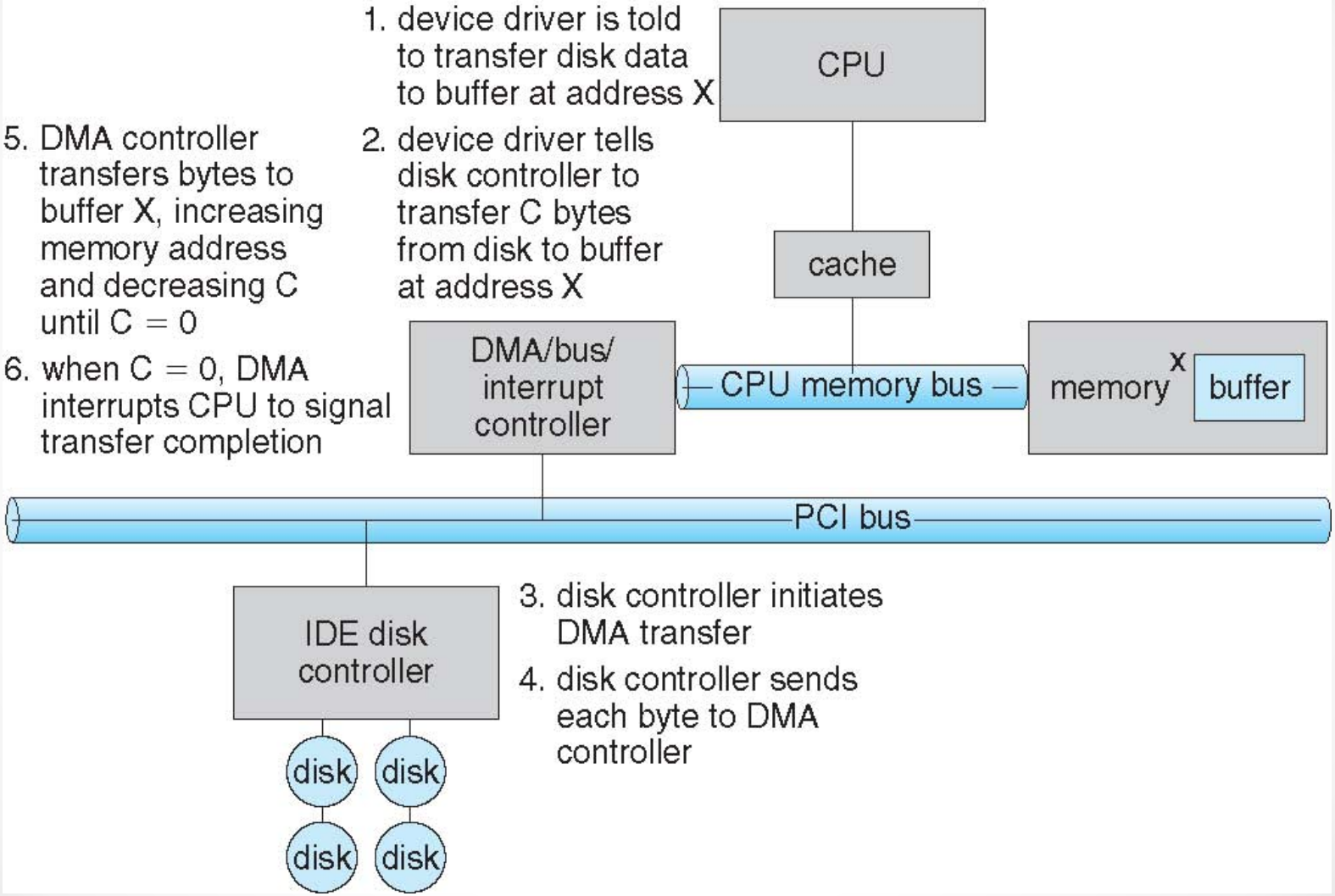
a) Less operations to memory.

b) Less interaction CPU-controller.

c) More cache in the device.

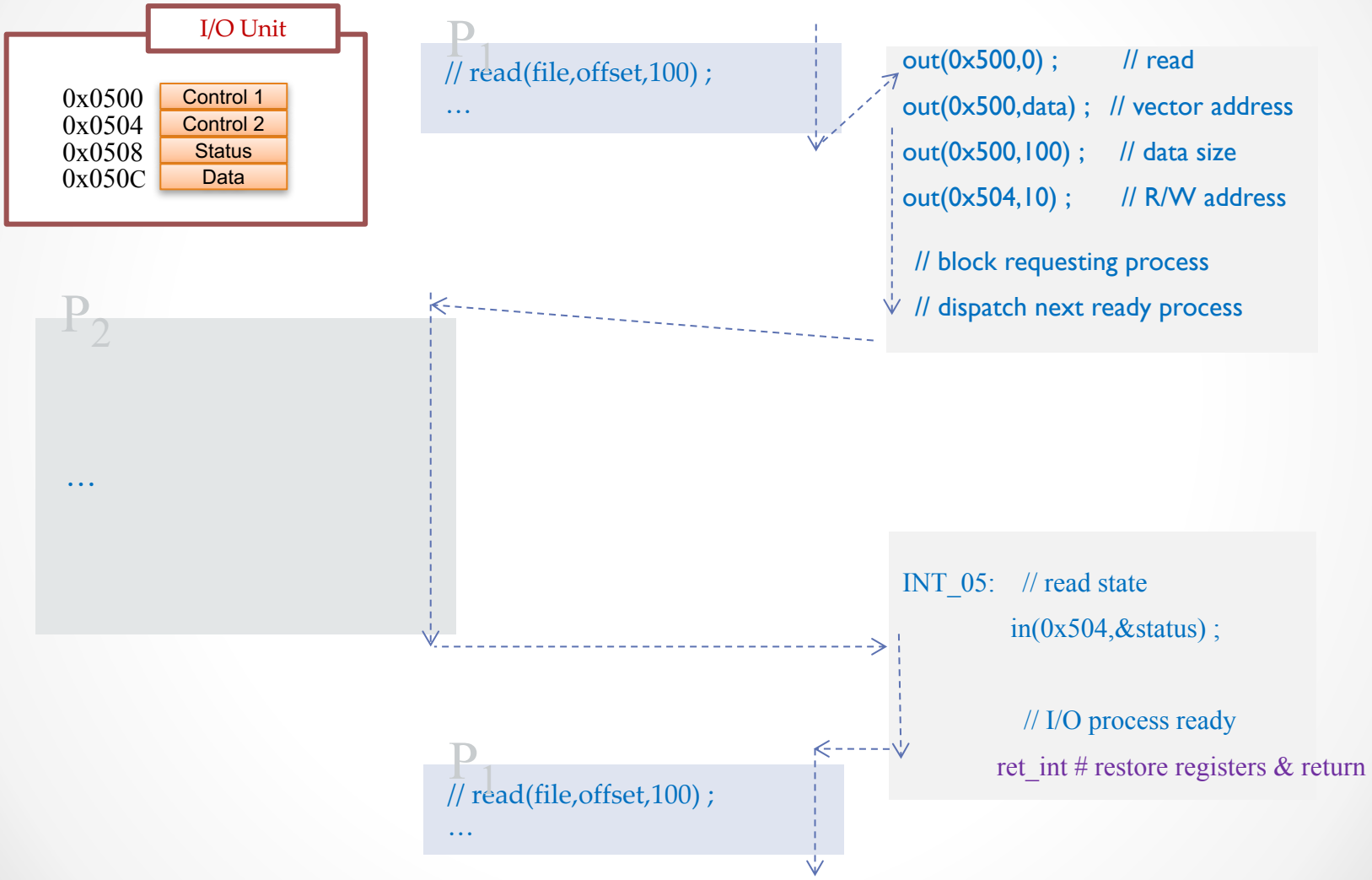
d) Makes use of programmed I/O

Six Step Process to Perform DMA Transfer



Example

I/O by DMA



Quiz

- Which is the preferred way of CPU-Controller interaction for a block device?

a) Polling.

b) Interrupts.

c) Direct Memory Access

d) I/O channel

Grupo ARCOS

Operating Systems Design
Grado en Ingeniería Informática

Universidad Carlos III de Madrid