CS420: Operating Systems

I/O Systems

James Moscola
Department of Physical Sciences
York College of Pennsylvania

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

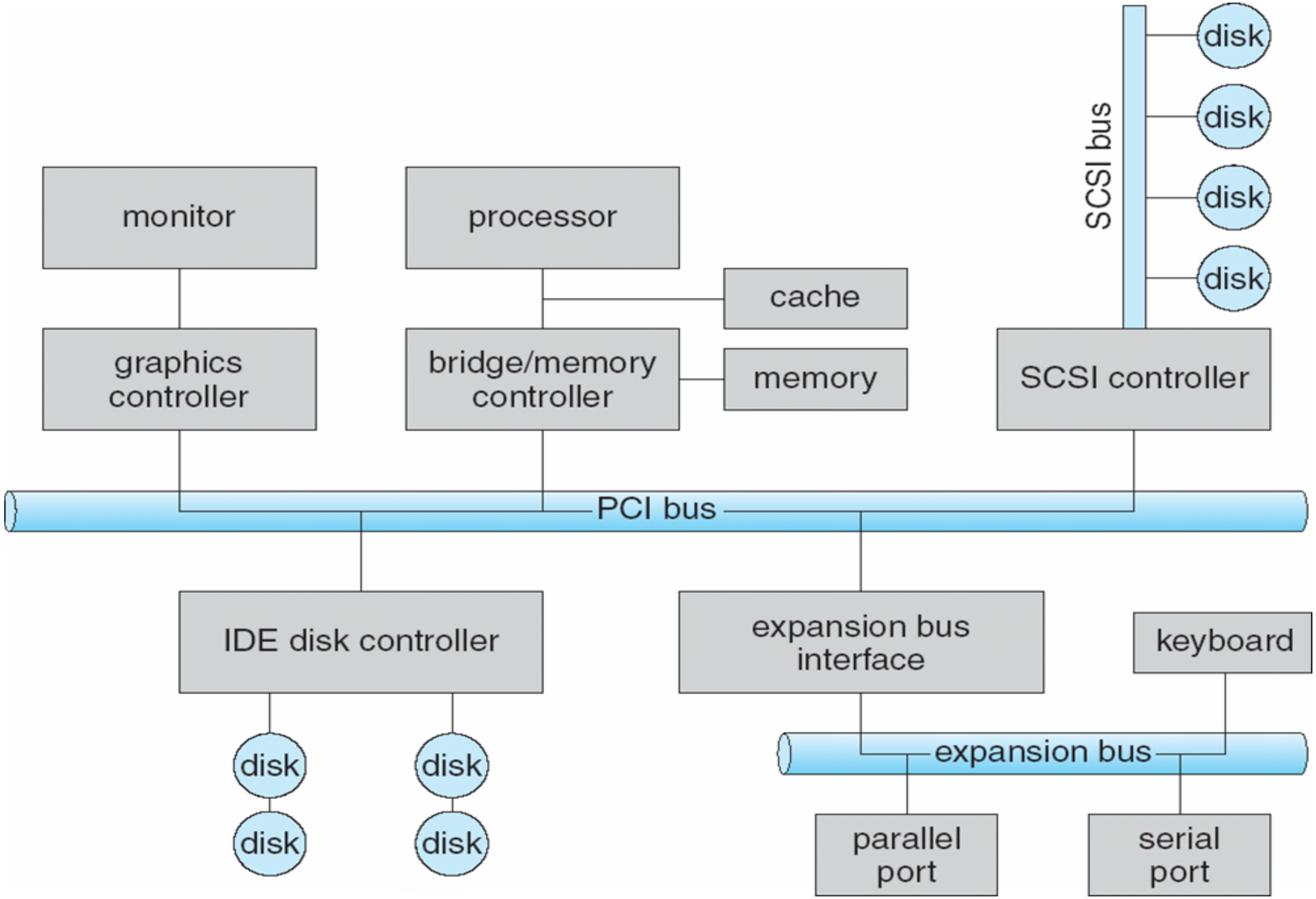
- **Ports, buses, device controllers connect to various devices**

- **Device drivers encapsulate device details**
  - Present uniform device-access interface to I/O subsystem

I/O Hardware

- **Incredible variety of I/O devices**
  - Storage (hard disk drives, flash drives, CDs, DVD, etc.)
  - Transmission (wireless network cards, Ethernet cards, modems ;-)
  - Human-interface (mouse, keyboard, trackpad, etc.)
- **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 into device
    - Sometimes separate circuit board
    - Contains processor, microcode, private memory, bus controller, etc

A Common PC Bus Structure



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
  - Registers are typically 1-4 bytes, or FIFO buffer
- **Devices have addresses, used by**
  - Direct I/O instructions
  - Memory-mapped I/O
    - Device data and command registers mapped to processor address space
    - Especially useful for large address spaces (graphics)

Device I/O Port Locations on PCs (partial list)

I/O address range (hexadecimal)	device
000–00F	DMA controller
020–021	interrupt controller
040–043	timer
200–20F	game controller
2F8–2FF	serial port (secondary)
320–32F	hard-disk controller
378–37F	parallel port
3D0–3DF	graphics controller
3F0–3F7	diskette-drive controller
3F8–3FF	serial port (primary)

Polling

- **One option when communicating between the host and a controller is polling**
  - For each byte of I/O
    - Read busy bit from status register until 0 (busy-waiting)
    - Host sets read or write bit and if write copies data into data-out register
    - Host sets command-ready bit
    - Controller sets busy bit, executes transfer
    - 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
    - If CPU switches to another task, data may be lost (buffer overflows)

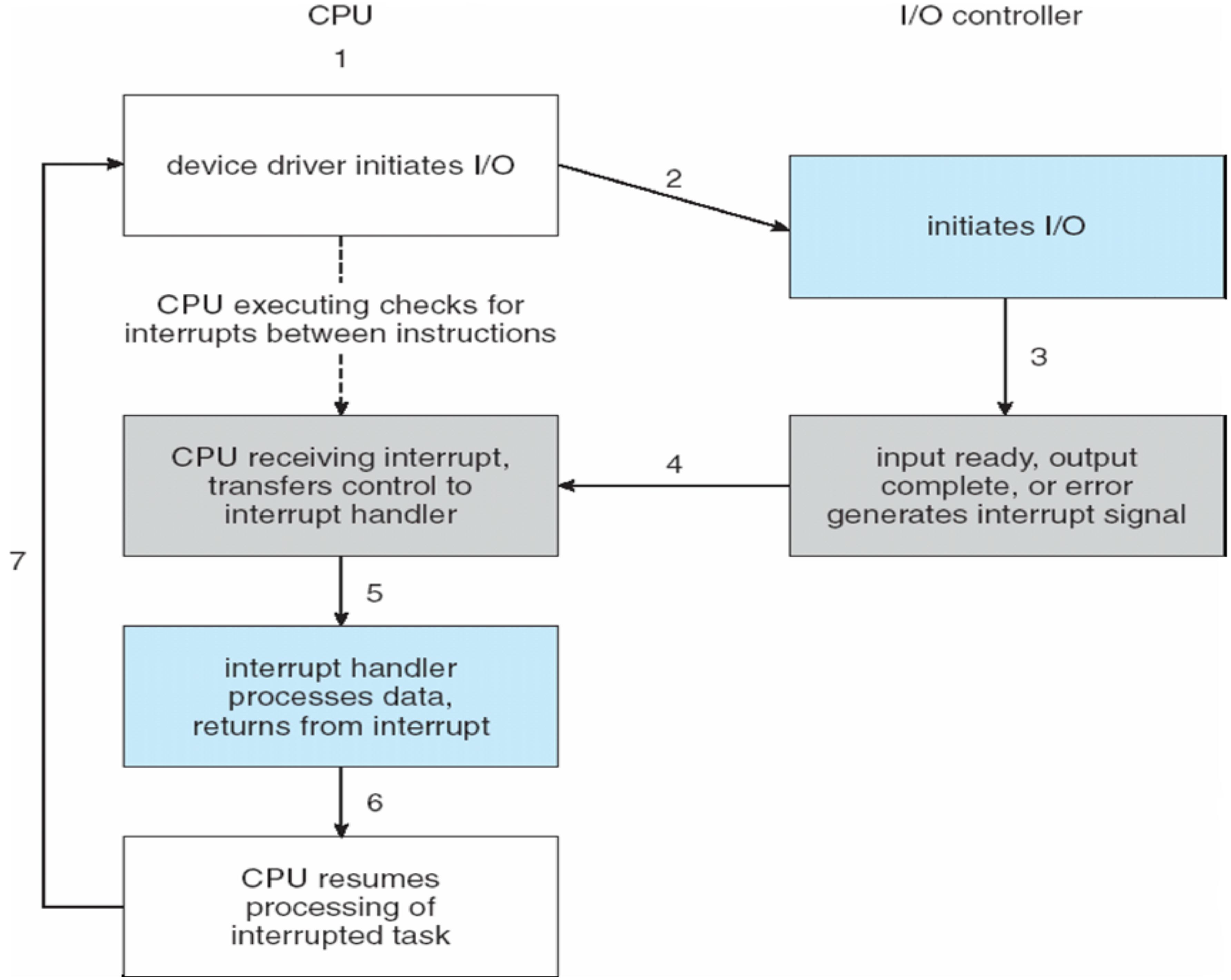
Interrupts

- **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 handler receives interrupts**
  - Maskable to ignore or delay some interrupts -- important when kernel is modifying shared data structures
- **Interrupt vector is used to dispatch interrupt to correct interrupt handler**
  - Context switch at start and end
  - Priority based
  - Some interrupts in vector are non-maskable
  - May need to chain interrupts if more than one device at same interrupt number

Intel Pentium Processor Event-Vector Table

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
19–31	(Intel reserved, do not use)
32–255	maskable interrupts

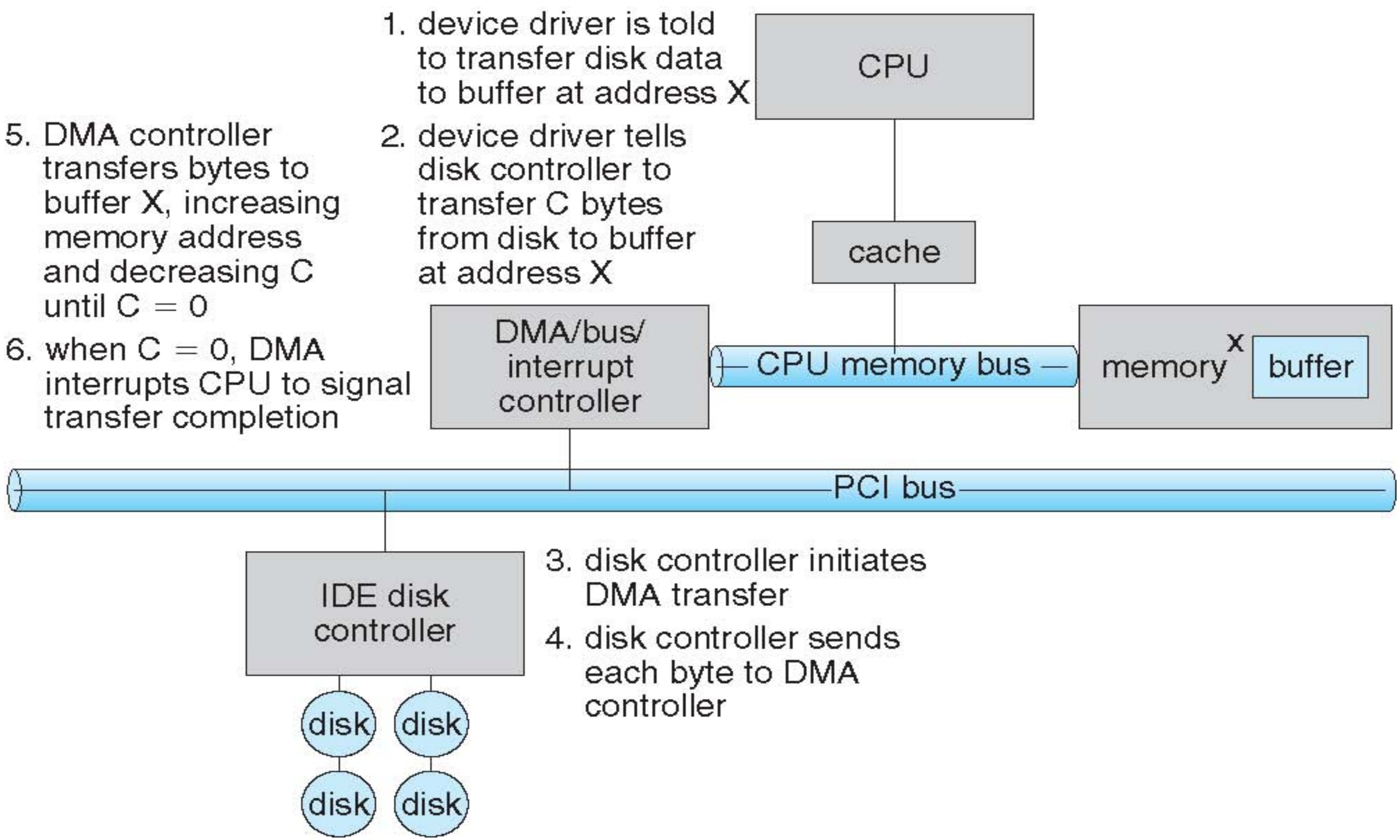
Interrupt-Driven I/O Cycle



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**
  - Don't want to waste CPU cycles to do something as simple as transferring data
- **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 memory bus from CPU (CPU CANNOT use memory bus during this time)
  - When done, interrupts to signal completion

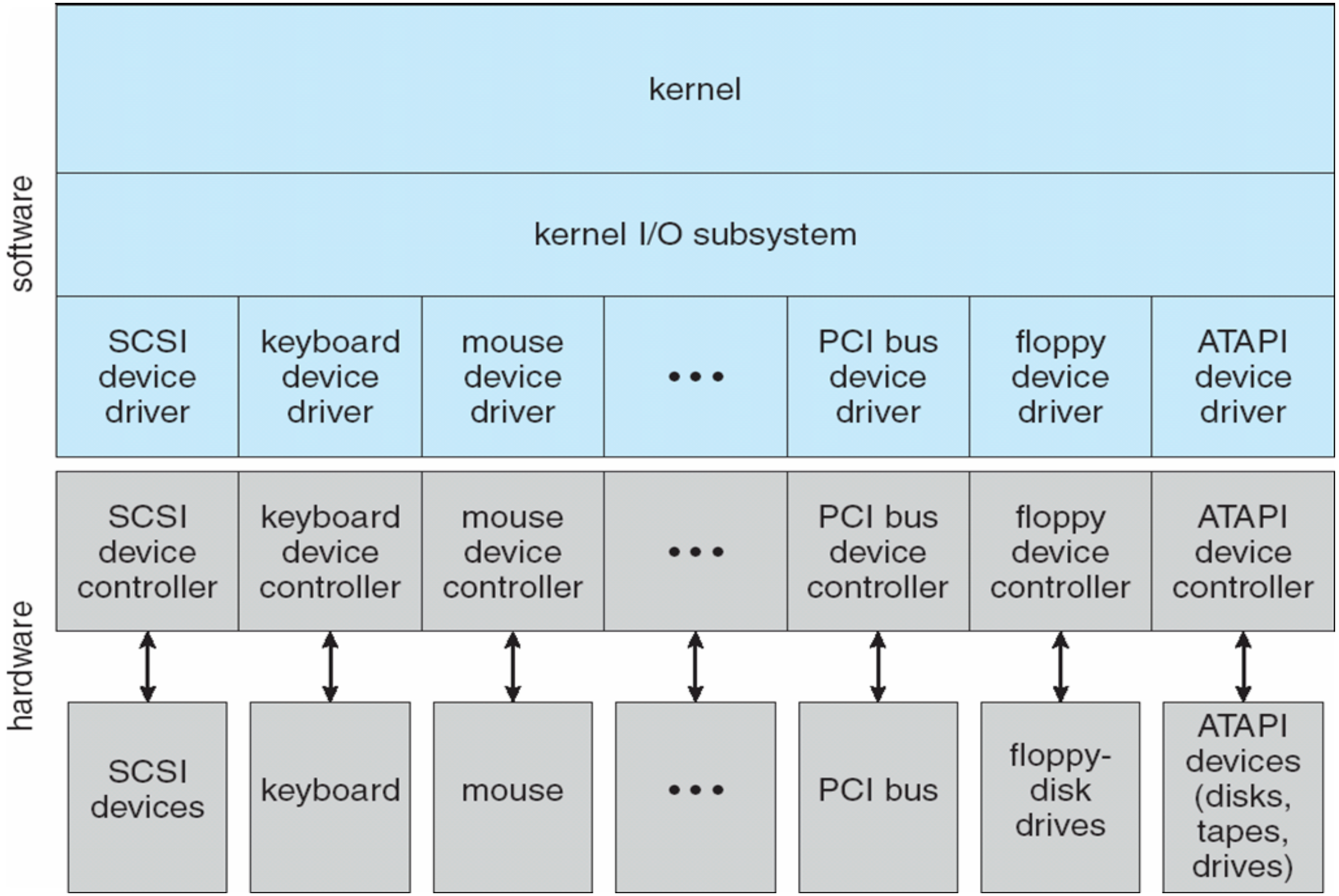
Six Step Process to Perform DMA Transfer



Application I/O Interface

- **It is useful for the operating system to be able to treat devices in a standard, uniform way**
  - Too many different types of devices for OS designers to handle each individually
  - Devices types are grouped and the OS provides a standard interface to those groups
- **Device drivers specific to each device implement one of the standard interfaces provided by the OS**
- **Device-driver layer hides differences among I/O controllers from kernel**
- **New devices using already-implemented protocols to communicate need no extra work**
- **Each OS has its own I/O subsystem structures and device driver frameworks**
- **Devices vary in many dimensions**
  - Character-stream or block transfer
  - Sequential or random-access
  - Synchronous or asynchronous transfers (or both)
  - Sharable or dedicated (can it be used concurrently by multiples threads/processes?)
  - Speed of operation
  - read-write, read only, or write only

A Kernel I/O Structure



Characteristics of I/O Devices

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 read–write	CD-ROM graphics controller disk

Characteristics of I/O Devices (Cont.)

- **Subtleties of devices handled by device drivers**

- **Broadly I/O devices can be grouped by the OS into:**
  - Block I/O
  - Character I/O (Stream)
  - Memory-mapped file access
  - Network sockets

Block and Character Devices

- **Block devices include disk drives**
  - Commands include `read`, `write`, `seek`
  - Raw I/O, direct I/O, or file-system access
  - Memory-mapped file access possible
  - DMA

- **Character devices include keyboards, mice, serial ports**
  - Commands include `get`, `put`
  - Libraries layered on top allow additional functionality

Network Devices

- **Different enough from block and character devices to have its own interface**

- **Unix and Windows NT/9x/2000 include socket interface**
  - Commands include `open`, `close`, `send`, `receive`
  - Separates network protocol from network operation

Clocks and Timers

- **Provide current time, elapsed time, timer**

- **Programmable interval timer used for timings, periodic interrupts**
  - Determining when a time-slice for a thread is complete
  - Periodically flushing disk buffers