Operating Systems

Jinghui Zhong （钟竞辉）
Office： B3-515
Email： jinghuizhong@scut.edu.cn

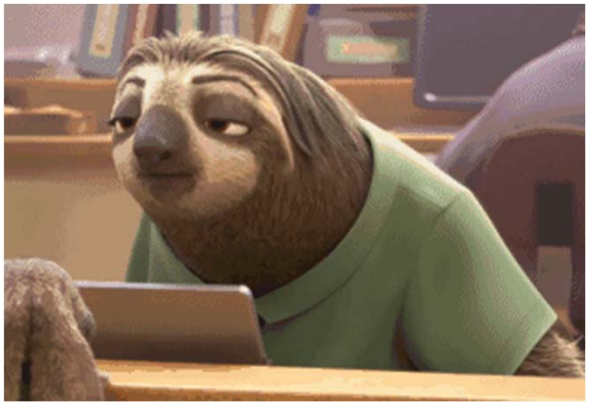
I/O Device

- **Two common kinds of I/O devices:**
  - ① **Block device:** stores information in fixed-size blocks.
  - ② **Character device：** delivers or accepts a stream of characters, without regard to any block structure.
- Special device: e.g., clocks.

Principles of I/O Hardware

- I/O devices cover a huge range in speeds

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

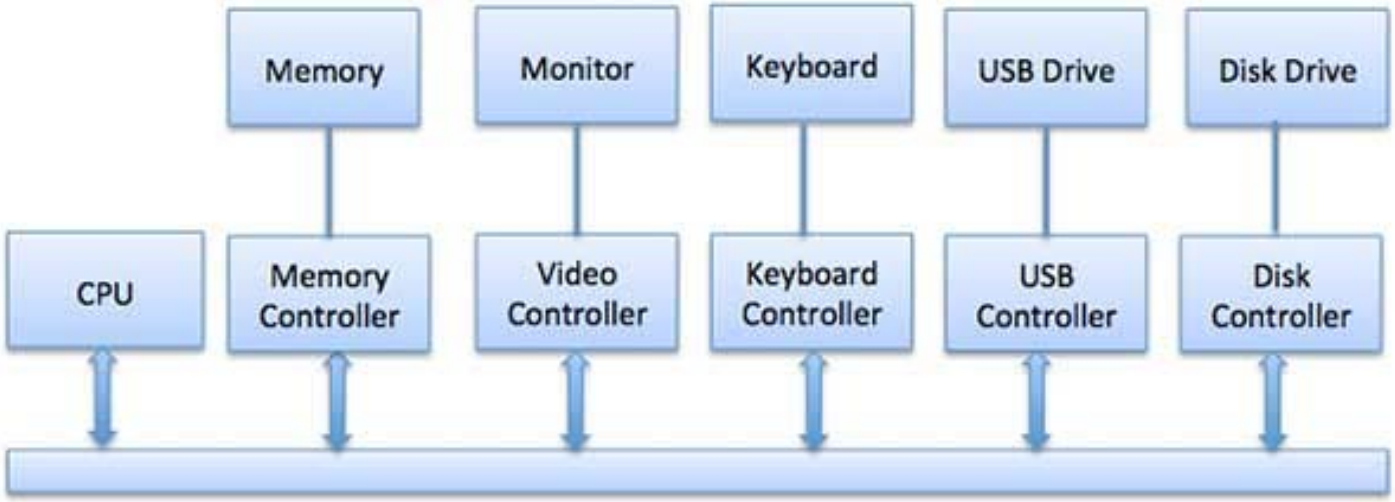


Device Controllers

- **Components of I/O devices:**

① Mechanical component ;

② Electronic component： i.e., device controller



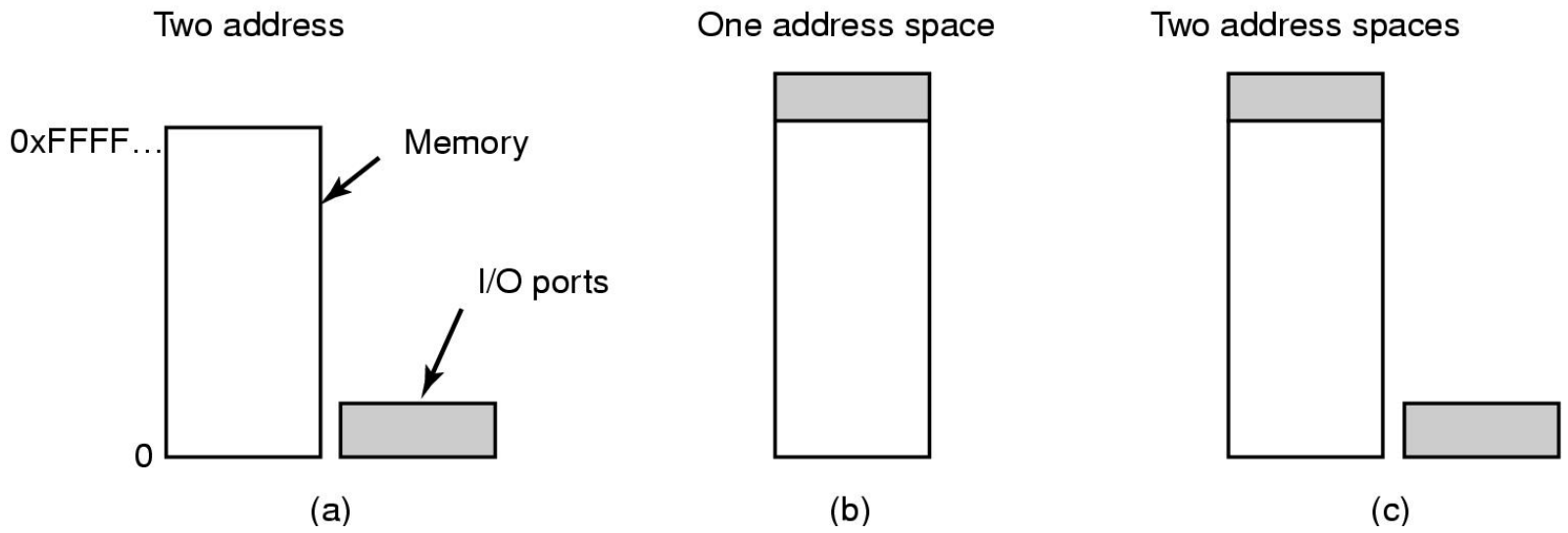
Device Controllers

- A device controller is a part of a computer system that makes sense of the signals going to, and coming from the CPU.

- Each device controller has a **local buffer** and some **registers**. It communicates with the CPU by interrupts. A device's controller plays as a bridge between the device and the operating system.

Memory-Mapped I/O

- Three approaches:
  ① Each control register is assigned an **I/O port** number.
  ② All the control registers are mapped into the memory space. This is called **memory-mapped I/O**.
  ③ Mapping I/O data buffers into memory space but separating I/O ports from memory



Memory-Mapped I/O

IN REG, PORT,
OUT PORT,REG

e.g.,

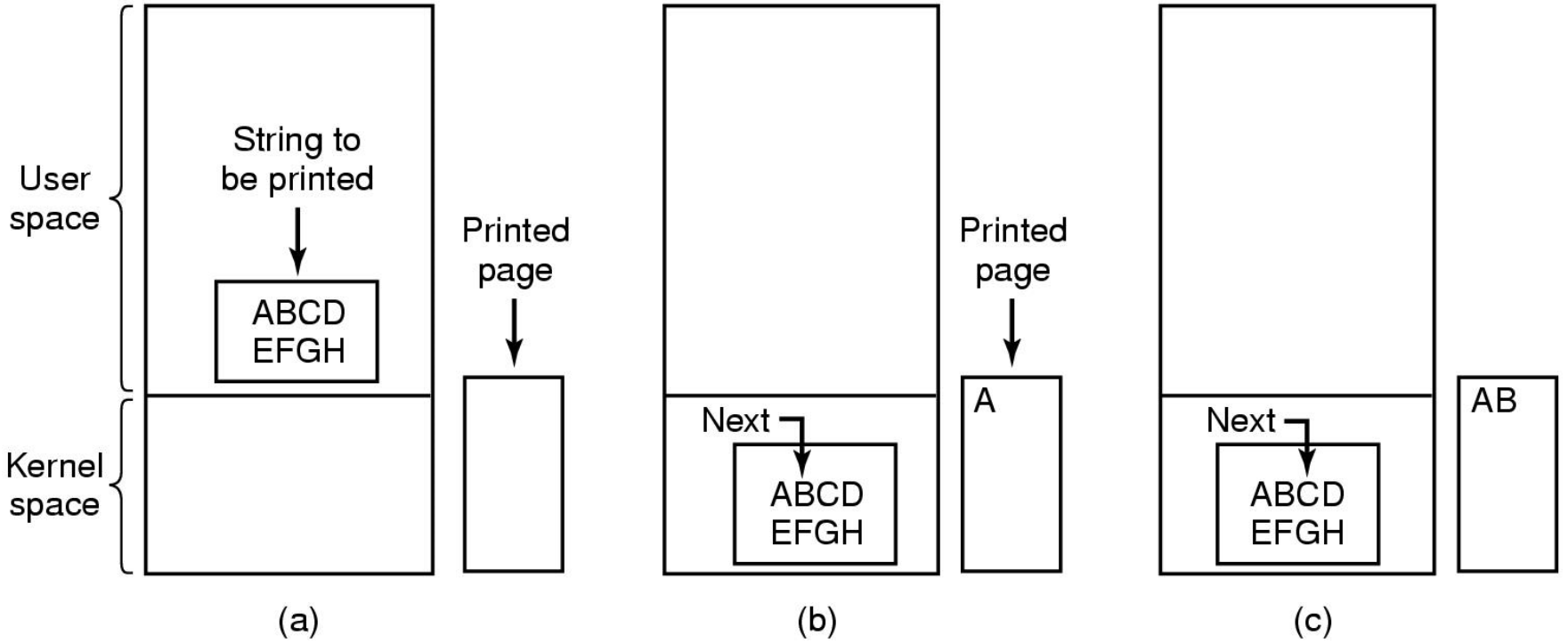
IN R0, 4
and
MOV R0, 4

Programmed I/O

- **Programmed input/output (PIO)**

① A method of transferring data between the **CPU** and a peripheral.

② Software running on the CPU uses instructions to perform data transfers to or from an I/O device.



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( );
```

Writing a string to the printer using programmed I/O

Interrupt-Driven I/O

- Writing a string to the printer using interrupt-driven I/O
  ① Code executed when print system call is made
  ② Interrupt service procedure

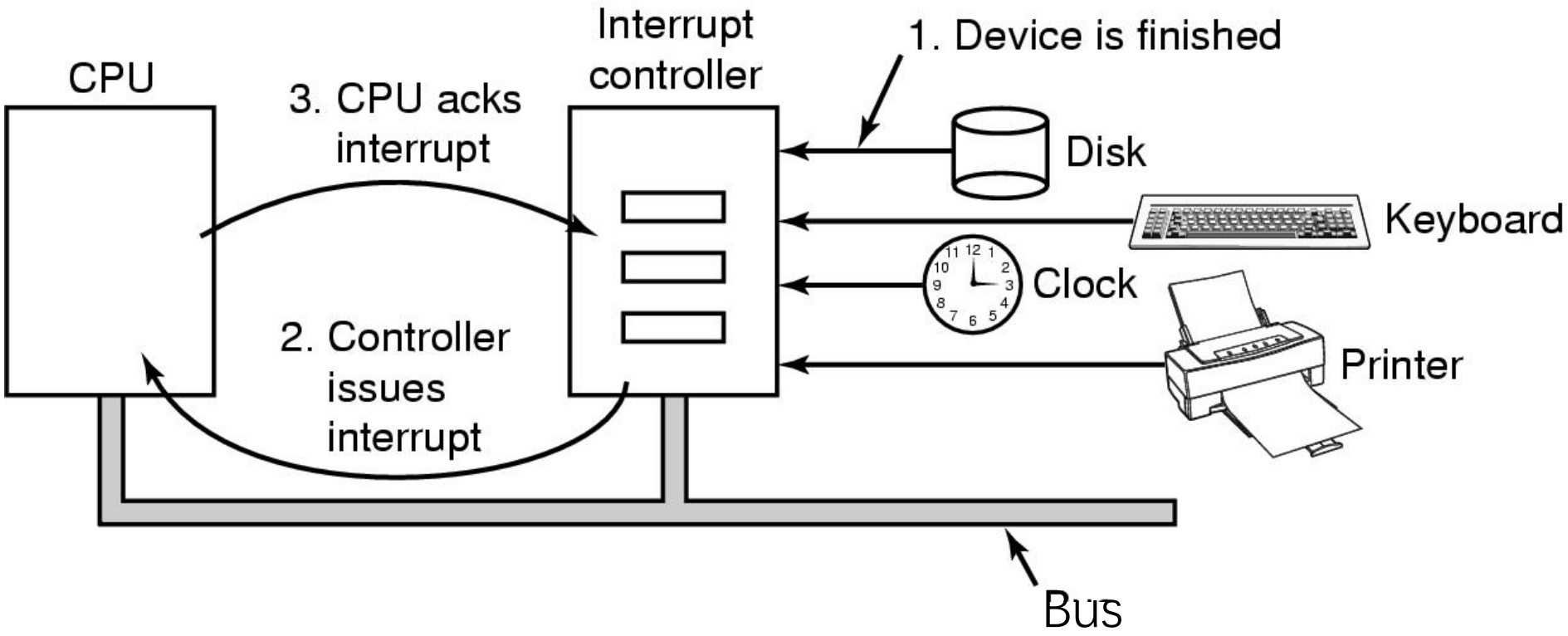
```
copy_from_user(buffer, p, count);
enable_interrupts( );
while (*printer_status_reg != READY) ;
*printer_data_register = p[0];
scheduler( );
```

(a)

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

(b)

Interrupt



How interrupts happens?

Connections between devices and interrupt controller actually use interrupt lines on the bus rather than dedicated wires

Interrupt Handlers

- Interrupt handlers are best hidden, have driver starting an I/O operation block until interrupt notifies of completion

- Interrupt procedure does its task, then unblocks driver that started it.

PP.430, Problem 8

Suppose that a computer can read or write a memory word in 5 nsec. Also suppose that when an interrupt occurs, all 32 CPU registers, plus the program counter and PSW are pushed onto the stack. What is the maximum number of interrupts per second this machine can process?

I/O Using DMA

- Printing a string using DMA
  - (a) code executed when the print system call is made
  - (b) interrupt service procedure

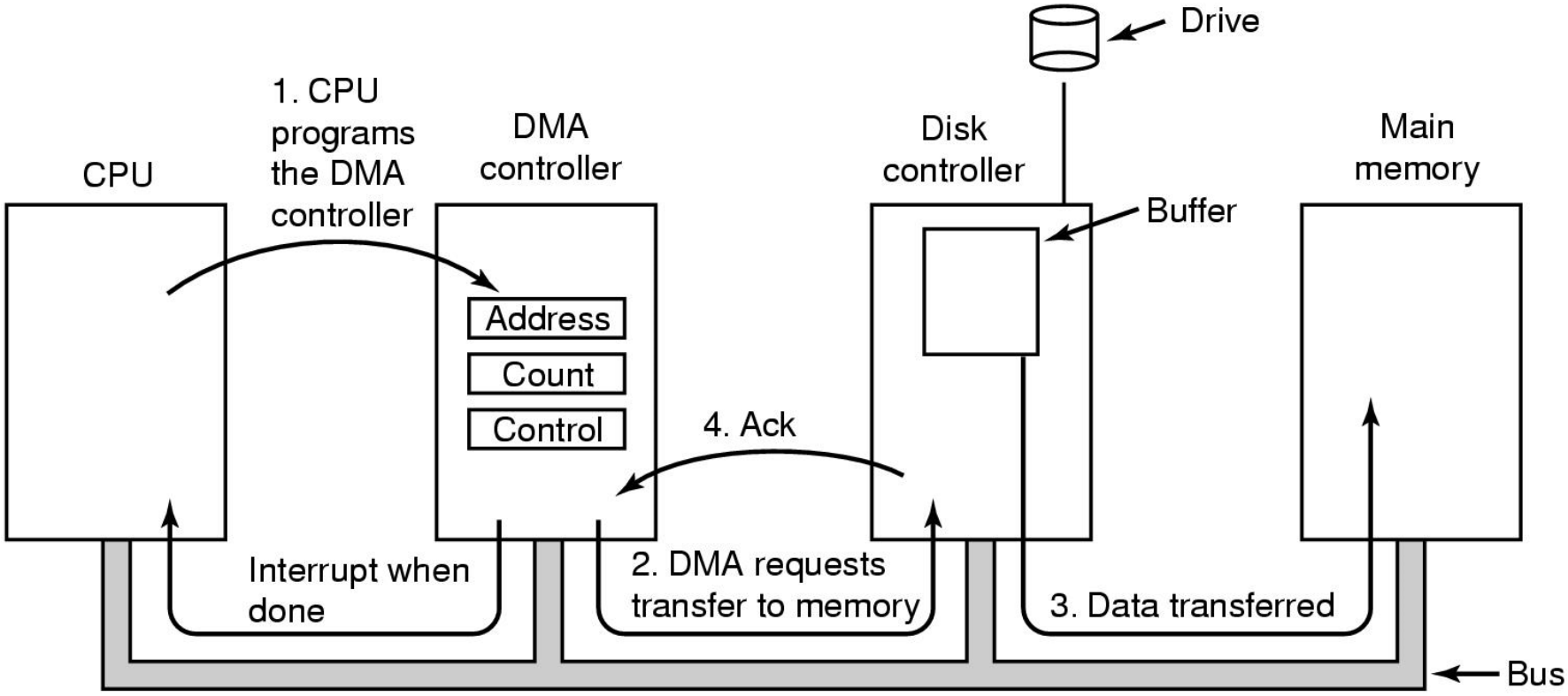
```
copy_from_user(buffer, p, count);
set_up_DMA_controller( );
scheduler( );
```

(a)

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

(b)

Direct Memory Access (DMA)



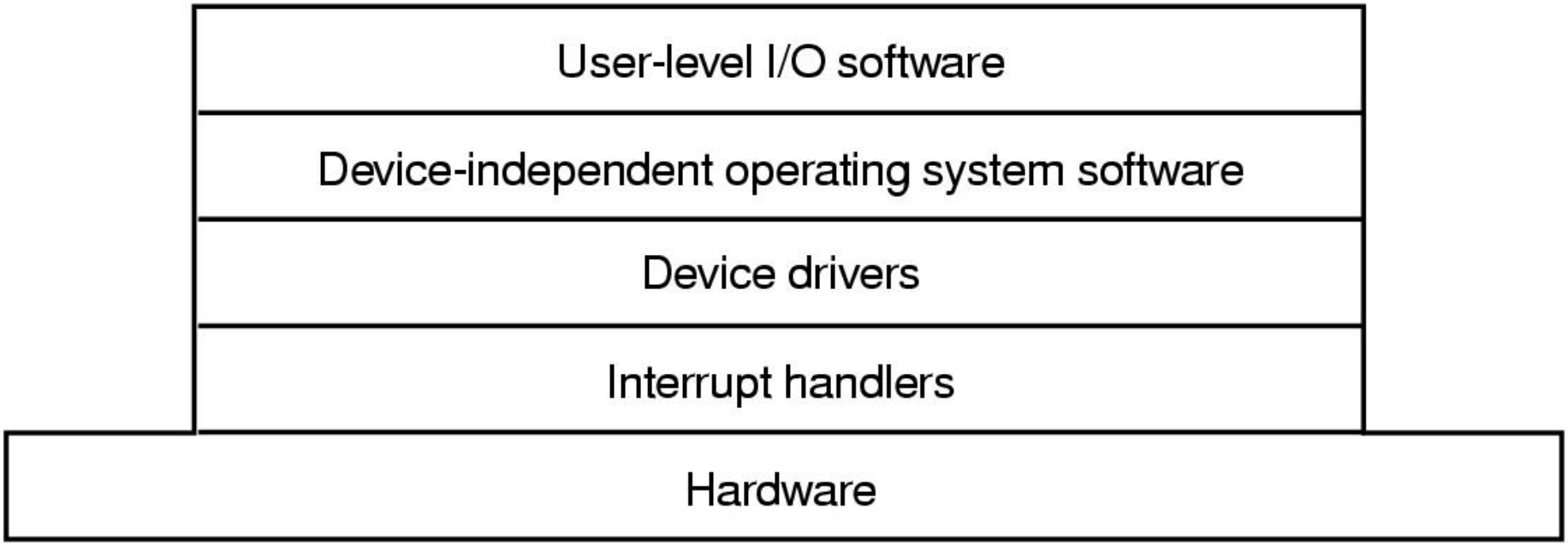
Operation of a DMA transfer

PP.429, Problem 5,6

5. A DMA controller has five channels. The controller is capable of requesting a 32-bit word every 40 nsec. A response takes equally long. How fast does the bus have to be to avoid being a bottleneck?

6. Suppose that a system uses DMA for data transfer from disk controller to main memory. Further assume that it takes t_1 nsec on average to acquire the bus and t_2 nsec to transfer one word over the bus ($t_1 \gg t_2$). After the CPU has programmed the DMA controller, how long will it take to transfer 1000 words from the disk controller to main memory, if (a) word-at-a-time mode is used, (b) burst mode is used? Assume that commanding the disk controller requires acquiring the bus to send one word and acknowledging a transfer also requires acquiring the bus to send one word.

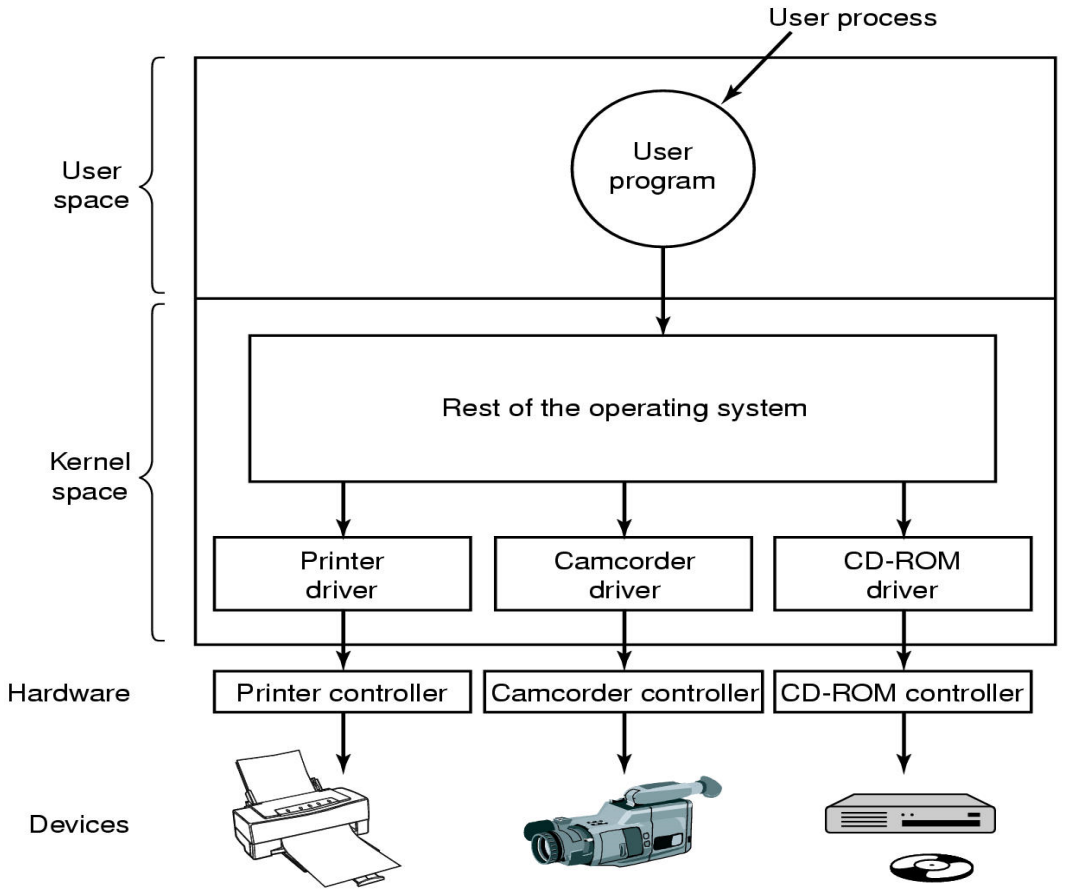
I/O Software Layers



Layers of the I/O Software System

Device Drivers

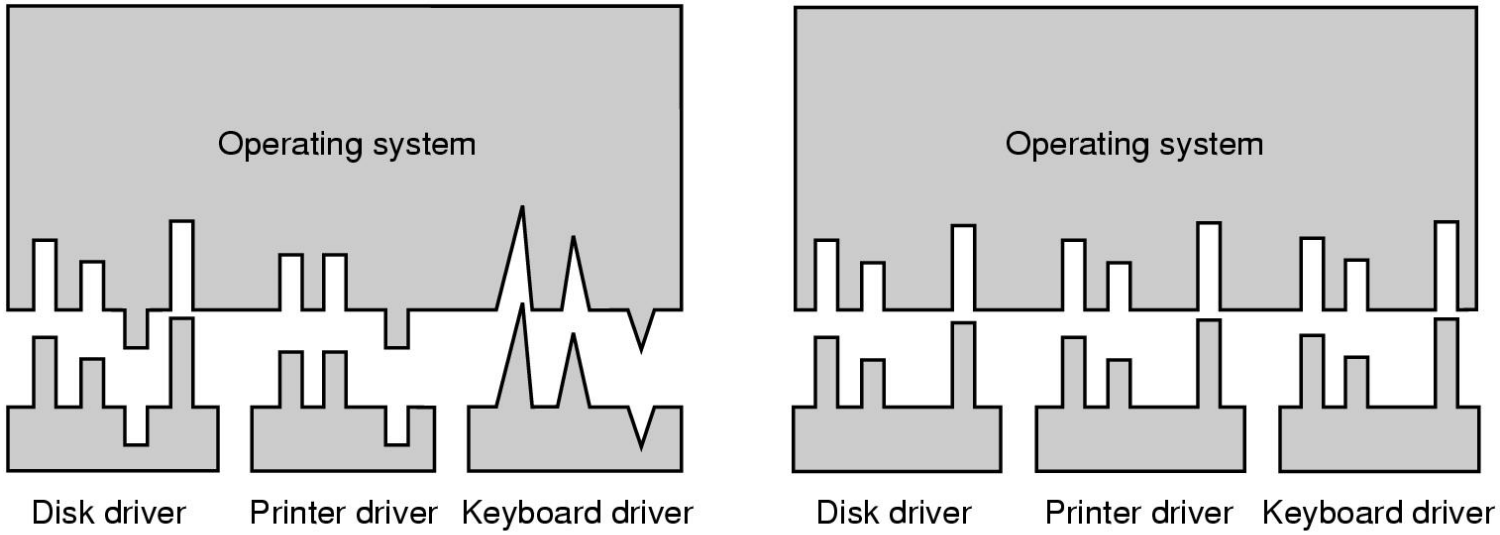
- Communications between drivers and device controllers goes over the bus; Logical position of device drivers is shown in the following figure.



Several issues of Designing I/O Software

- Device independence

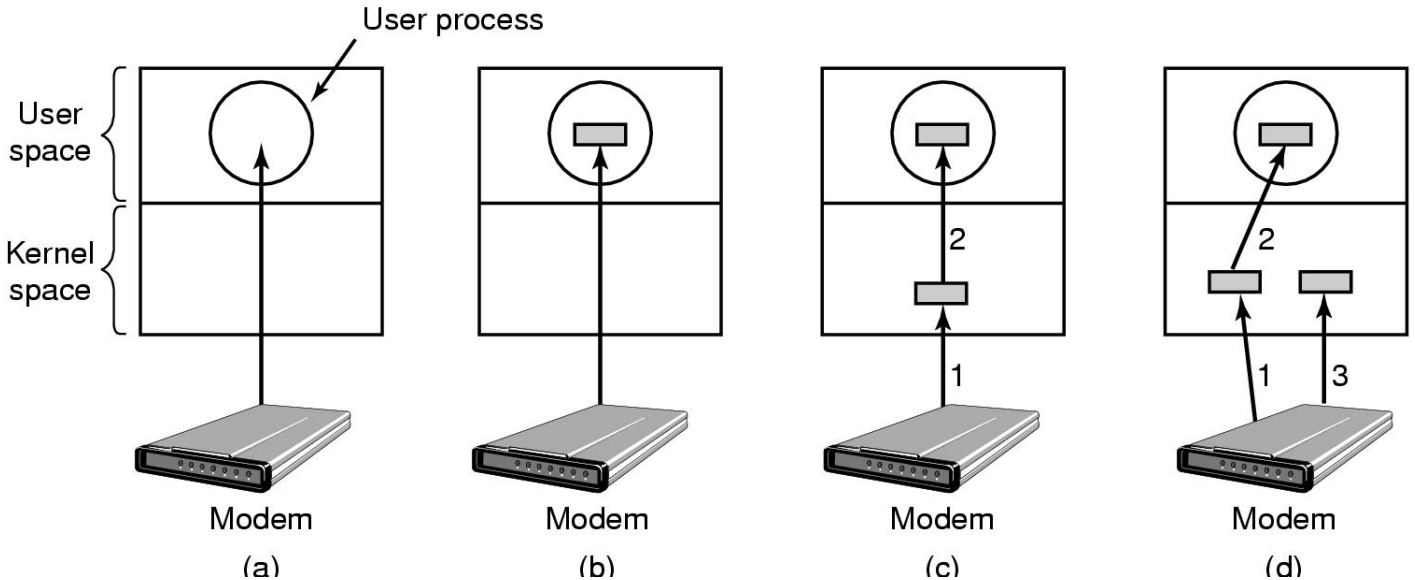
  e.g., programs can access any I/O device without specifying device in advance (e.g., floppy, hard drive, or CD-ROM)



(a) Without a standard driver interface (b) With a standard driver interface

Several issues of Designing I/O Software

- Buffering



(a) Unbuffered input
(b) Buffering in user space
(c) Buffering in the kernel followed by copying to user space
(d) Double buffering in the kernel

Check Points

① What are the two kinds of IO devices.

② What are the two components of IO devices.

③ What is programmed IO?

④ What is Interrupt-Driven IO?

⑤ What is IO using DMA?

⑥ What are the five layers of the IO software system

⑦ What are the difference between driver and interrupt handler?