

Operating System Principles

操作系统原理

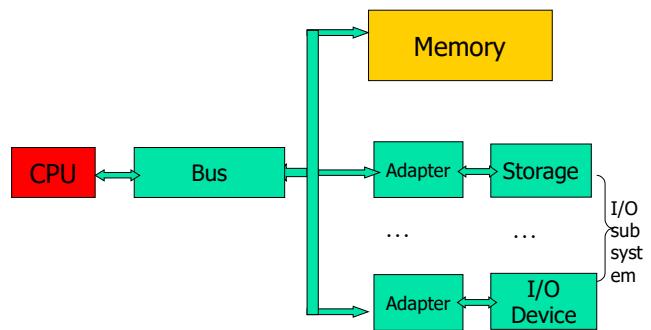
Input/Output

李旭东

leexudong@nankai.edu.cn

Nankai University

Computer System:Hardware



OSP © LeeXudong@nankai.edu.cn

I/O System Target

- Basic
 - Issue commands to the device
 - catch interrupts
 - Handle errors
- Easy to use
- Extensibility
 - Device independence

leexudong@nankai.edu.cn

3

Objectives

- Principles of I/O Hardware
- Principles of I/O Software
- I/O Software Layers
- Disks
- Clocks
- User Interfaces
- Power Management

leexudong@nankai.edu.cn

4

Principles of I/O Hardware

- ### I/O Devices
- Block devices
 - Block addressable
 - Character devices
 - A stream of characters
 - Other devices
 - Clocks

leexudong@nankai.edu.cn

6



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

7

Device Controllers

I/O Unit

- A mechanical component
- An electronic component
 - Device controller: (i.e.) Adapter 适配器
 - Preamble 前置码
 - ECC: Error-Correcting Code 差错校验码

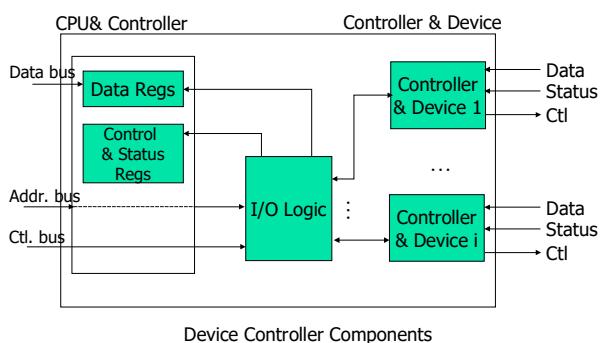
Device Controller

- Convert the serial bit stream into a block of bytes and perform any error correction necessary
- Can handle one, two, four ... identical devices
- Interface between the controller and device
- ANSI, IEEE, ISO standard, A de facto standard
- IDE, SCSI, SATA, USB, Firewire(IEEE 1394)

leexudong@nankai.edu.cn

8

Device Controllers



OSP © LeeXudong@nankai.edu.cn

9

Memory-Mapped I/O

Device controller

- Control registers
- A data buffer

I/O port

I/O port space: set of I/O ports

Special I/O instruction

- IN REG, PORT
- OUT PORT, REG

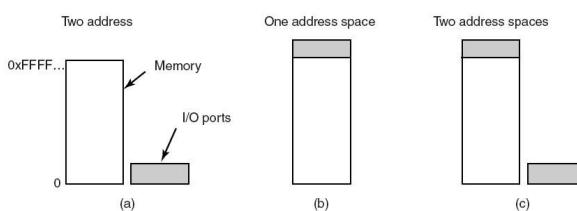
Memory-Mapped I/O

- MOV R0, 4

leexudong@nankai.edu.cn

10

Memory-Mapped I/O

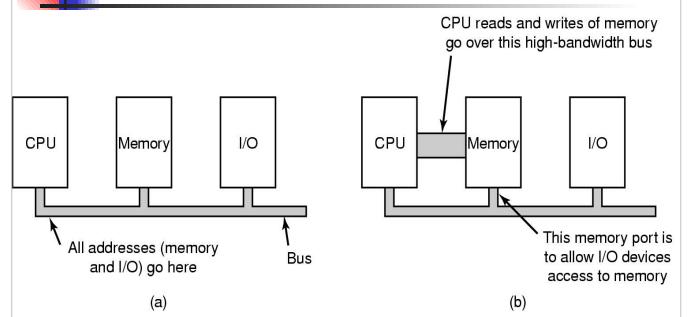


- a) Separate I/O and memory space
 b) Memory-mapped I/O
 c) Hybrid

leexudong@nankai.edu.cn

11

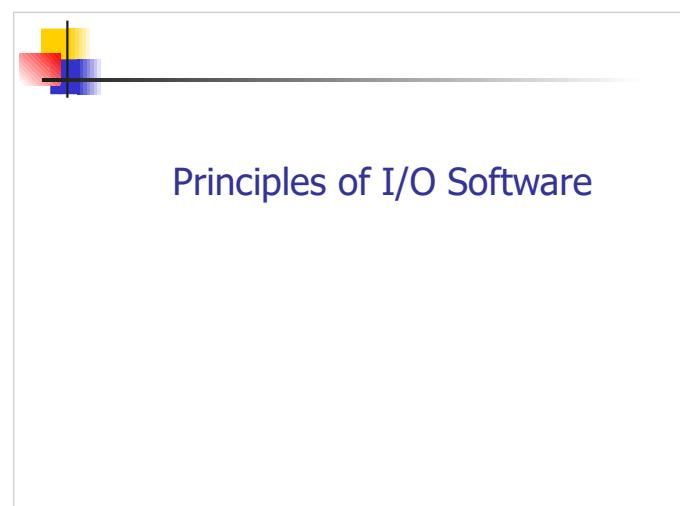
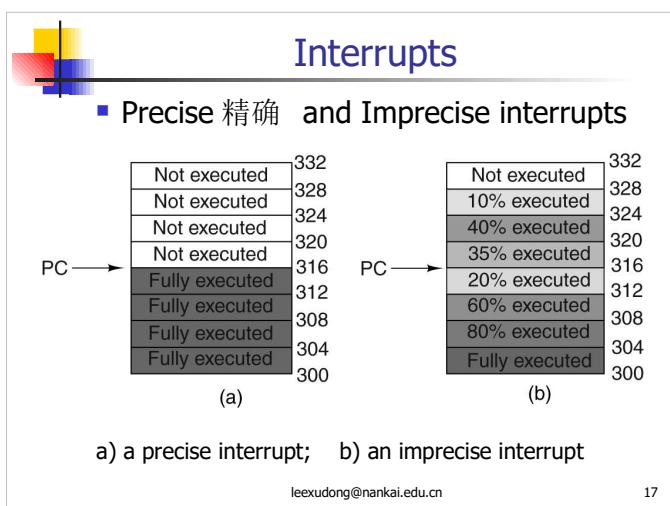
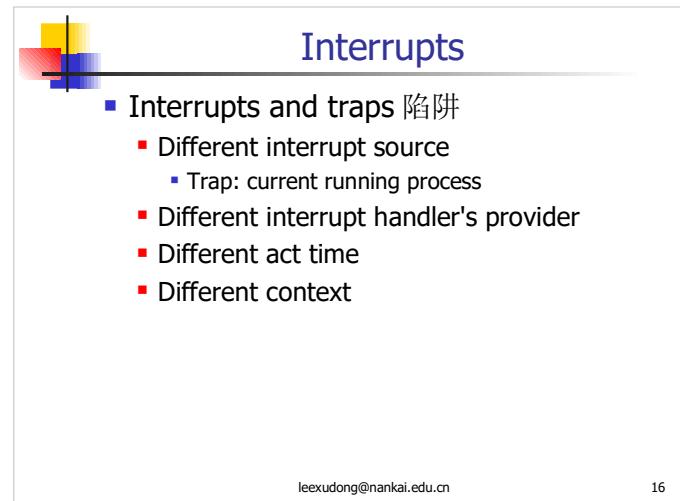
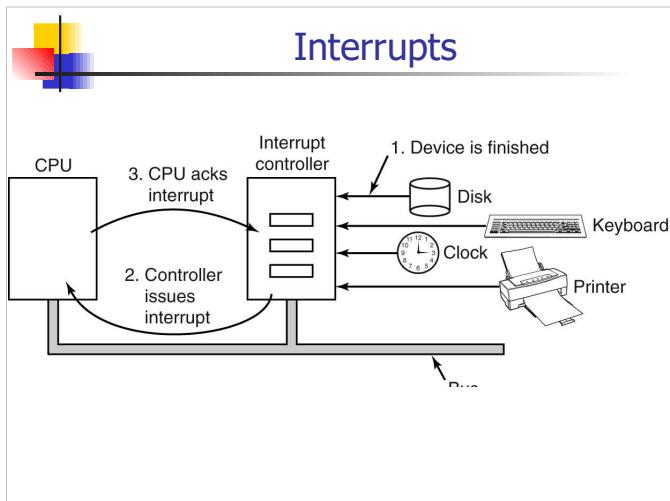
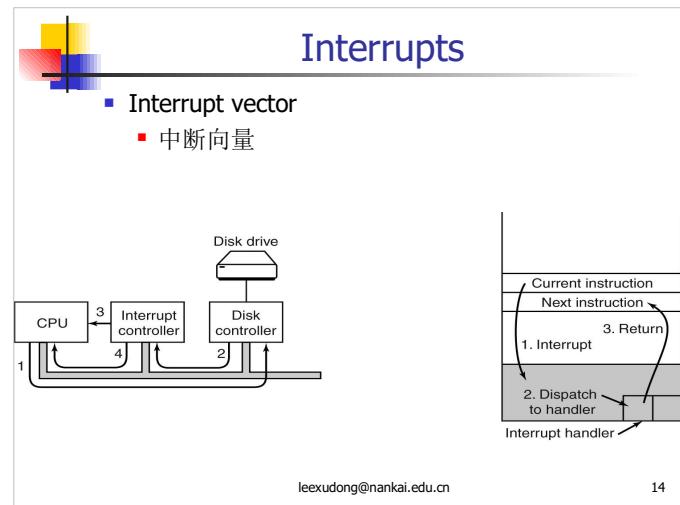
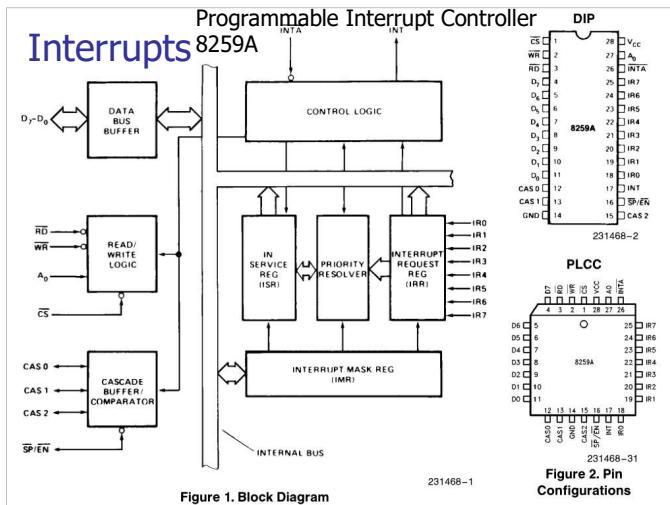
Memory-Mapped I/O



- (a) A single-bus architecture
 (b) A dual-bus memory architecture

leexudong@nankai.edu.cn

12



Goals of I/O Software 1/2

- device independence 设备独立性
 - programs can access any I/O device
 - without specifying device in advance
 - (floppy, hard drive, or CD-ROM)
- uniform naming 统一命名
 - name of a file or device a string or an integer
 - not depending on which machine
- error handling 错误处理
 - handle as close to the hardware as possible

OSP © LeeXudong@nankai.edu.cn

第 19 页

Goals of I/O Software 2/2

- synchronous(同步) V.S. asynchronous(异步)
 - blocked transfers vs. interrupt-driven
- buffering 缓冲
 - data coming off a device cannot be stored in final destination
- Shareable vs. dedicated devices
 - disks are shareable
 - tape drives would not be

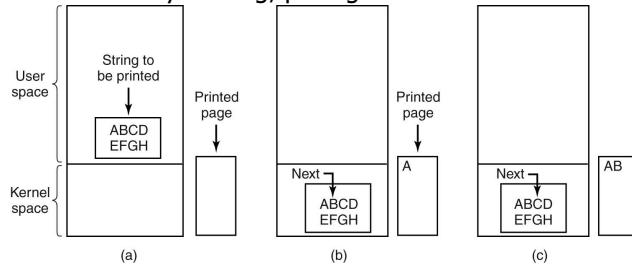
OSP © LeeXudong@nankai.edu.cn

第 20 页

Programmed I/O

▪ Programmed I/O 程序控制 I/O

▪ Busy waiting, polling



Steps in printing a string

leexudong@nankai.edu.cn

21

Programmed I/O

▪ Busy waiting

```
copy_from_user(buffer, p, count);           /* p is the kernel buffer */
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();
```

leexudong@nankai.edu.cn

22

Interrupt-Driven I/O

▪ 中断驱动式 I/O

```
copy_from_user(buffer, p, count);
enable_interrupts();
while (*printer_status_reg != READY) ;
*printer_data_register = p[0];
scheduler();
if (count == 0) {
    unblock_user();
} else {
    *printer_data_register = p[i];
    count = count - 1;
    i = i + 1;
}
acknowledge_interrupt();
return_from_interrupt();
```

(a)

(b)

Writing a string to the printer using interrupt-driven I/O
a) Code executed at the time the print system call is made
b) Interrupt service procedure for the printer

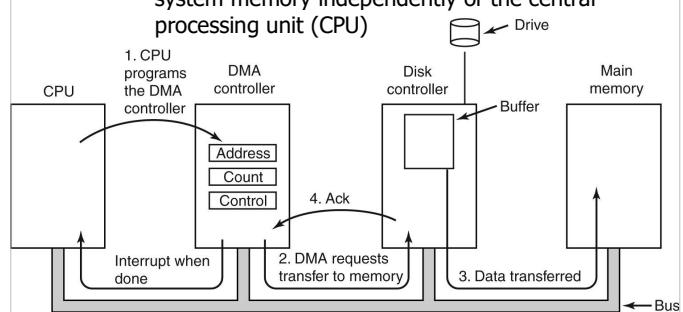
leexudong@nankai.edu.cn

23

I/O Using DMA

▪ Direct Memory Access, DMA

- a feature of computerized systems that allows certain hardware subsystems to access main system memory independently of the central processing unit (CPU)



I/O Using DMA

▪ 使用 DMA 的 I/O

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

(a)

```
acknowledge_interrupt();  
unlock_user();  
return_from_interrupt();
```

(b)

Printing a string using DMA.

- (a) Code executed when the print system call is made.
- (b) Interrupt service procedure.

leexudong@nankai.edu.cn

25

DMA

▪ Bus modes

- Word-at-a-time mode
- Block mode

▪ DMA modes

- **Burst mode:** 突发模式
 - DMA controller requests for the transfer of one word and gets it
 - Cycle stealing mode: block the CPU
- **Fly-by mode:** 飞越模式
 - DMA controller tell the device controller to transfer the data directly to main memory

leexudong@nankai.edu.cn

26

DMA

- Physical memory addresses
- Virtual memory addresses
 - DMA controller must use the MMU to have the virtual-to-physical translation done
- Disk ↔ Disk Controller buffer ↔ mem

leexudong@nankai.edu.cn

27

I/O Using Channel

- Channel 通道
- A high-performance input/output (I/O) architecture that is implemented in various forms on a number of computer architectures, especially on mainframe computers
- Channel architecture uses a separate, independent, low-cost processor
- Channel processors are simple, but self-contained, with minimal logic and sufficient on-board scratchpad memory (working storage) to handle I/O tasks
- Each channel may support one or more controllers and/or devices, but each channel program may only be directed at one of those connected devices

leexudong@nankai.edu.cn

28

Channel I/O

- Channel program
 - a sequence of channel command words (CCWs) which are executed by the I/O channel subsystem.

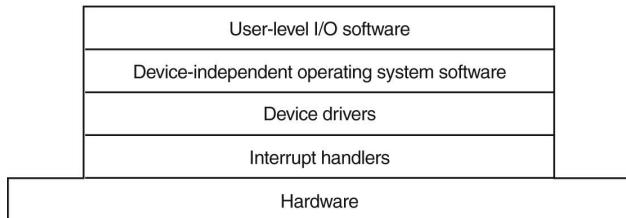
OP	P	R	Bytes	Mem Addr.
WRITE	0	0	80	813
WRITE	0	0	140	1034
WRITE	0	1	60	5830
WRITE	0	1	300	2000
WRITE	0	0	250	1850
WRITE	1	1	250	720

leexudong@nankai.edu.cn

29

I/O Software Layers

I/O Software Layers



leexudong@nankai.edu.cn

31

Interrupt Handlers

- Save registers not already been saved by interrupt hardware.
- Set up a context for the interrupt service procedure.
- Set up a stack for the interrupt service procedure.
- Acknowledge the interrupt controller. If there is no centralized interrupt controller, reenable interrupts.
- Copy the registers from where they were saved to the process table.

leexudong@nankai.edu.cn

32

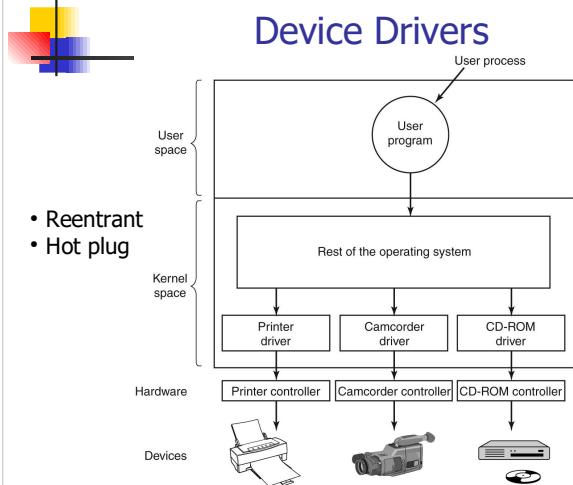
Interrupt Handlers

- Run the interrupt service procedure.
- Choose which process to run next.
- Set up the MMU context for the process to run next.
- Load the new process' registers, including its PSW.
- Start running the new process.

leexudong@nankai.edu.cn

33

Device Drivers



34

Device Drivers

- Device driver
 - Each I/O device attached to a computer needs some device-specific code for controlling it
 - Each device driver normally handles one device type, or at most, one class of closely related devices
 - In order to access the device's hardware, meaning the controller's registers, the device driver normally has to be part of the os kernel

leexudong@nankai.edu.cn

35

Device-Independent I/O Software

Uniform interfacing for device drivers
Buffering
Error reporting
Allocating and releasing dedicated devices
Providing a device-independent block size

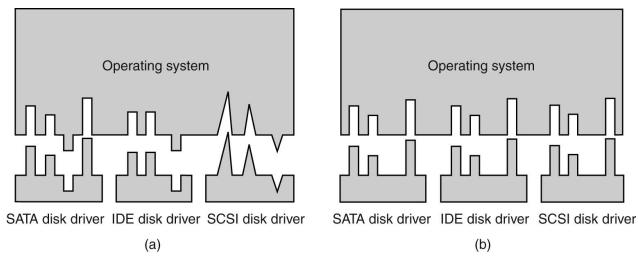
Functions of the device-independent I/O software

leexudong@nankai.edu.cn

36

Uniform Interfacing for Device Drivers

- Major device number, minor device number

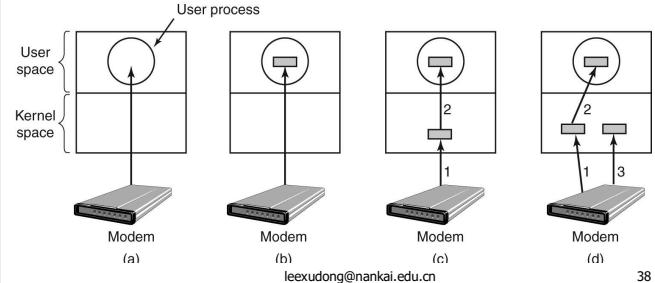


leexudong@nankai.edu.cn

37

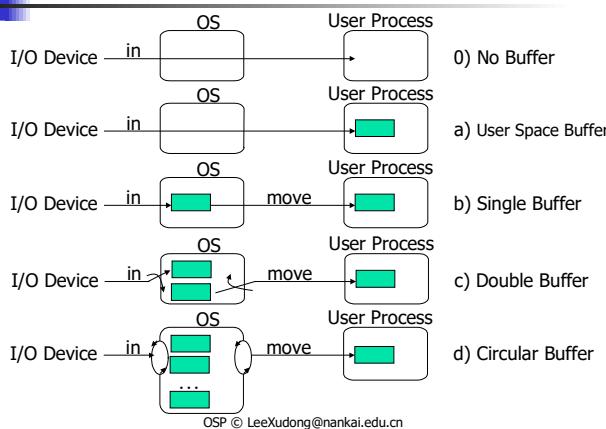
Buffering

- Buffers can increase application **performance**
 - by allowing synchronous operations such as file reads or writes to complete quickly instead of blocking while waiting for hardware interrupts to access a physical disk subsystem



38

Buffering



Device-Independent Block Size

- Different disks may have different sector sizes
- The device-independent software hides this fact and provides a uniform block size to higher layers

leexudong@nankai.edu.cn

41

Error Reporting

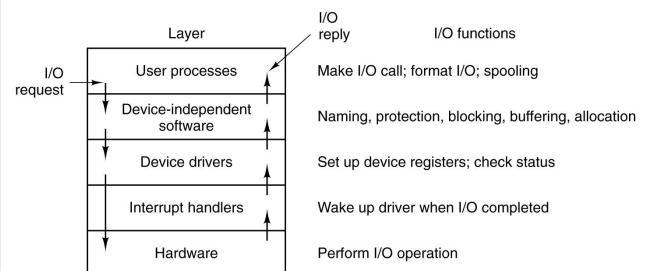
- Errors are far more common in the context of I/O than in other contexts
- Classes of I/O errors
 - Programming errors
 - Actual I/O errors
- How to handle I/O errors
 - ...

leexudong@nankai.edu.cn

40

User-Space I/O Software

- Applications
 - Open, Read, Write, ..., Close



leexudong@nankai.edu.cn

42

Disks

- Magnetic Disks
 - IDE: Integrated Drive Electronics
 - SATA: Serial ATA
- overlapped seeks
 - A controller can do seeks on two or more drivers at the same time
- Logical block addressing
 - X Cylinders, Y heads, Z sectors
 - (x, y, z)
 - IBM PC: (65536, 16, 63)

leexudong@nankai.edu.cn

43

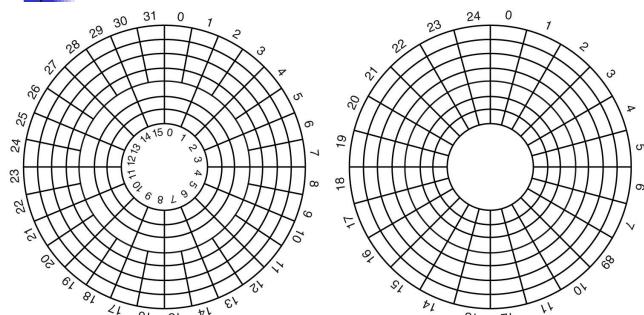
Disks

Parameter	IBM 360-KB floppy disk	WD 18300 hard disk
Number of cylinders	40	10601
Tracks per cylinder	2	12
Sectors per track	9	281 (avg)
Sectors per disk	720	35742000
Bytes per sector	512	512
Disk capacity	360 KB	18.3 GB
Seek time (adjacent cylinders)	6 msec	0.8 msec
Seek time (average case)	77 msec	6.9 msec
Rotation time	200 msec	8.33 msec
Motor stop/start time	250 msec	20 sec
Time to transfer 1 sector	22 msec	17 μ sec

leexudong@nankai.edu.cn

44

Disks



a) Physical geometry of a disk with two zones
b) A possible virtual geometry for this disk

leexudong@nankai.edu.cn

45

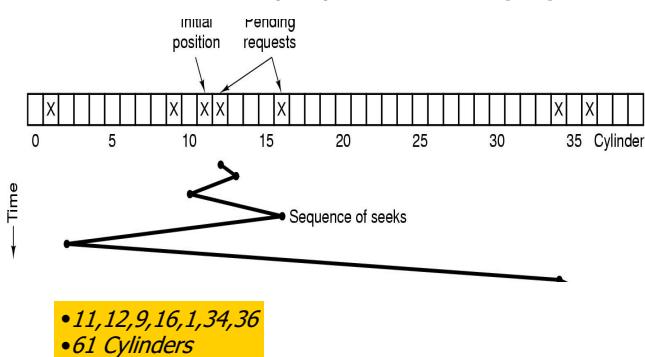
Disk Arm Scheduling Algorithms

- Time required to read or write a disk block determined by 3 factors
 - Seek time
 - Rotational delay
 - Actual transfer time
- **Seek time dominates**
- Error checking is done by controllers
- Scheduling Algorithms
 - FCFS , SSF , Elevator
 - Example: 11, 1, 36, 16, 34, 9, 12
 - Moved 111 cylinders

OSP © LeeXudong@nankai.edu.cn

SSF disk scheduling algorithm

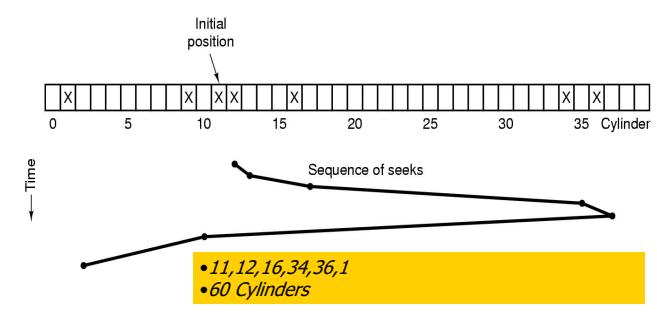
- Shortest Seek First (SSF) disk scheduling algorithm



OSP © LeeXudong@nankai.edu.cn

Elevator disk scheduling algorithm

- Elevator(电梯) disk scheduling algorithm
 - SCAN algorithm(扫瞄)



OSP © LeeXudong@nankai.edu.cn

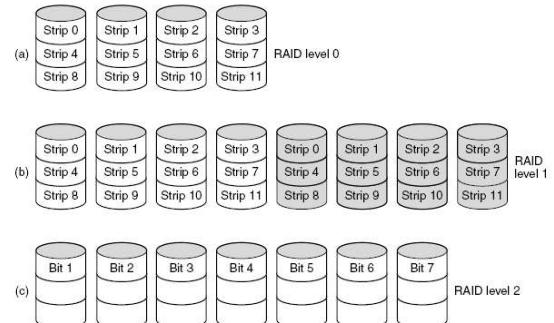
RAID

- RAID
 - Redundant Array of **Inexpensive** Disks
 - Redundant Array of **Independent** Disks
- Goals of RAID
 - Improve disk performance
 - Improve disk storage safe

leexudong@nankai.edu.cn

49

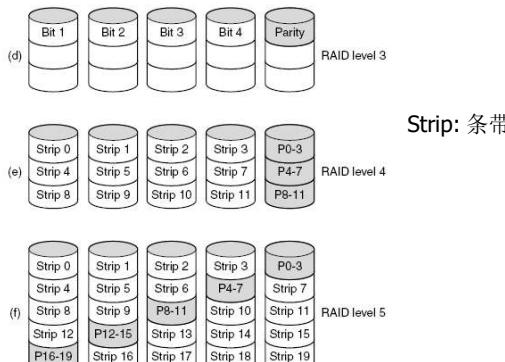
RAID



leexudong@nankai.edu.cn

50

RAID

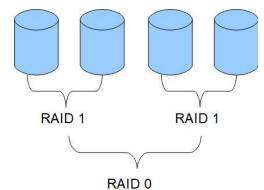
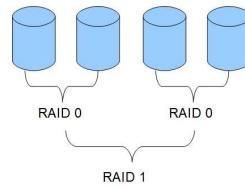


leexudong@nankai.edu.cn

51

RAID

- ? Which one is better?



Others Storage Devices

- CD-ROM:
 - Compact Disc – Read Only Memory
- CD-Recordables
- CD-Rewritables
- DVD
 - Digital Video Disk
- ...

leexudong@nankai.edu.cn

53

Stable Storage

- Stable Storage
 - Different from RAID
 - Logical Error: keep the disk consistent
 - crashes during writes corrupting the original data without replacing them by newer data
- ECC
 - Error-correcting code
 - 16 bytes: ecc(512 bytes)
- Stable Storage
 - Stable Writes
 - Stable Reads
 - ...

leexudong@nankai.edu.cn

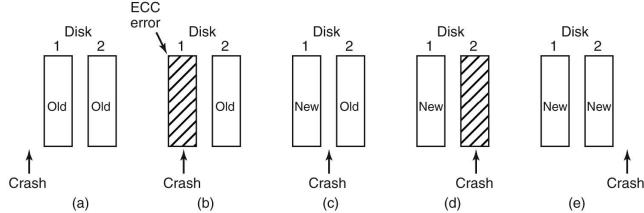
54

Stable Storage

■ Stable Storage

▪ Crash Recovery

- CPU crash :five cases



leexudong@nankai.edu.cn

55

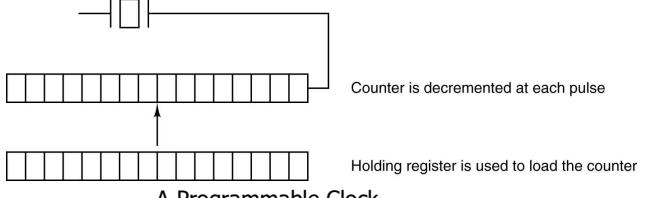
Clocks(Timers)

Clock Hardware

- Crystal oscillator 晶振体 , a counter, a holding register
- CPU interrupt: clock tick
- 8284/82C284

▪ Two modes: One-shot mode, Square-wave mode

Crystal oscillator



A Programmable Clock

leexudong@nankai.edu.cn

56

Clocks(Timers)

- Frequency Multiplier 倍频器
- UTC: Universal Coordinated Time
 - Greenwich Mean Time
 - UNIX: Jan.1, 1970
 - Windows: Jan.1, 1980



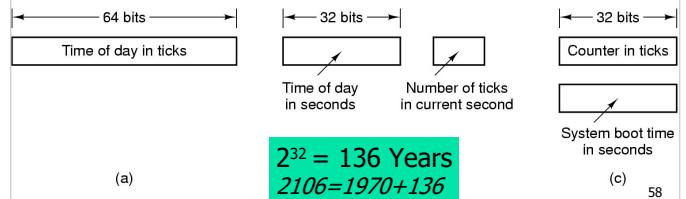
Location: Shanghai

Set the time: Manually Automatically from the Internet

57

Clock

- Soft Timers
- Duties of Clock Software
 - Maintaining the time of day
 - Preventing processes from running longer than they are allowed to
 - Accounting for CPU usage
 - Handling the alarm system call made by user processes
 - Providing watchdog timers for parts of the system itself
 - Doing profiling, monitoring, and statistics gathering



(a)

(c)

58

User Interface

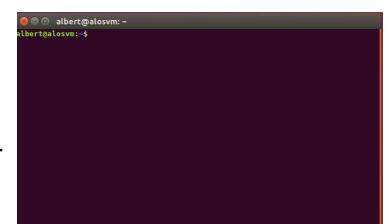
- Terminal Devices
 - Input Devices
 - Keyboard
 - Mouse
 - ...
 - Output Devices
 - Monitor
 - ...

leexudong@nankai.edu.cn

59

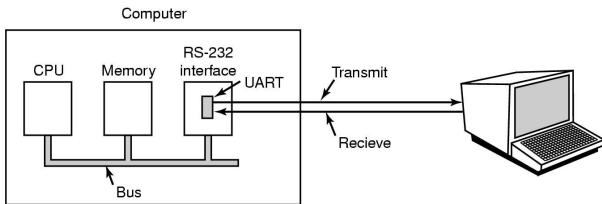
Terminal 终端

- Text Terminal
 - RS-232*
- GUI Terminal
 - Graphical User Interface
 - Windows
 - X Window*
 - ...
- Net Terminal*
 - SLIM: Stateless Low-level Interface Machine



OSP © LeeXudong@nankai.edu.cn

Text Terminal: RS-232



- An RS-232 terminal communicates with computer 1 bit at a time
- Called a serial line – bits go out in series, 1 bit at a time
- Windows uses COM1 and COM2 ports, first to serial lines
- unix: /dev/tty1 /dev/tty2
- Computer and terminal are completely independent

OSP © LeeXudong@nankai.edu.cn

Text Terminal: RS-232 Input 1/2

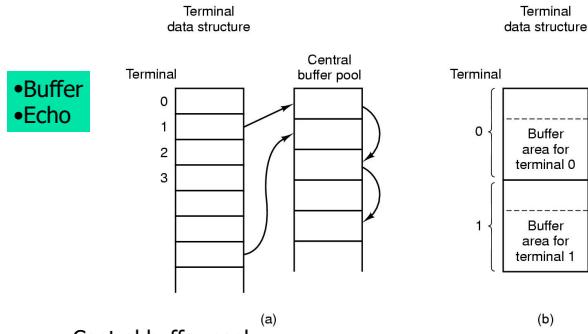
Character | POSIX name | Comment

Character	POSIX name	Comment
CTRL-H	ERASE	Backspace one character
CTRL-U	KILL	Erase entire line being typed
CTRL-V	LNEXT	Interpret next character literally
CTRL-S	STOP	Stop output
CTRL-Q	START	Start output
DEL	INTR	Interrupt process (SIGINT)
CTRL-\	QUIT	Force core dump (SIGQUIT)
CTRL-D	EOF	End of file
CTRL-M	CR	Carriage return (unchangeable)
CTRL-J	NL	Linefeed (unchangeable)

Characters handled specially in canonical mode

OSP © LeeXudong@nankai.edu.cn

Text Terminal: RS-232 Input 2/2



- Central buffer pool
 - 10 chars per buffer
- Dedicated buffer for each terminal

OSP © LeeXudong@nankai.edu.cn

Text Terminal: RS-232 Output

Escape sequence	Meaning
ESC [nA	Move up n lines
ESC [nB	Move down n lines
ESC [nC	Move right n spaces
ESC [nD	Move left n spaces
ESC [m;:nH	Move cursor to (m,n)
ESC [sJ	Clear screen from cursor (0 to end, 1 from start, 2 all)
ESC [sK	Clear line from cursor (0 to end, 1 from start, 2 all)
ESC [nL	Insert n lines at cursor
ESC [nM	Delete n lines at cursor
ESC [nP	Delete n chars at cursor
ESC [n@	Insert n chars at cursor
ESC [n/m	Enable rendition n (0=normal, 4=bold, 5=blinking, 7=reverse)
ESC M	Scroll the screen backward if the cursor is on the top line

▪ The ANSI escape sequences 转义序列

- accepted by terminal driver on output
- ESC is ASCII character (0x1B)
- n, m, and s are optional numeric parameters

▪ Termcap 终端数据库

OSP © LeeXudong@nankai.edu.cn

GUI

▪ Graphical User Interface

- Douglas Engelbart:
 - Stanford Research Institute
 - human-computer interaction
- Xerox PARC: copied
- Steve Jobs(Apple Corp.): Production
 - Apple Lisa、Macintosh
- Microsoft Windows: From Apple

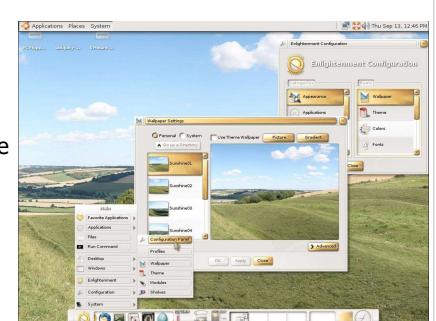


OSP © LeeXudong@nankai.edu.cn

GUI

▪ GUI “WIMP”

- Window
- Icon
- Menu
- Pointing device



OSP © LeeXudong@nankai.edu.cn

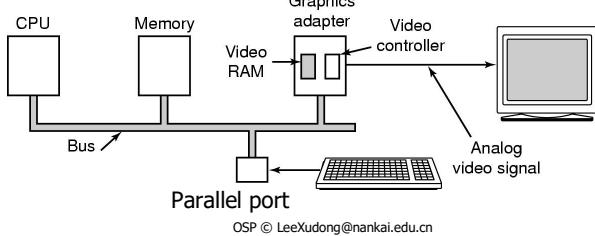
GUI: Display Hardware 1/2

Display Hardware

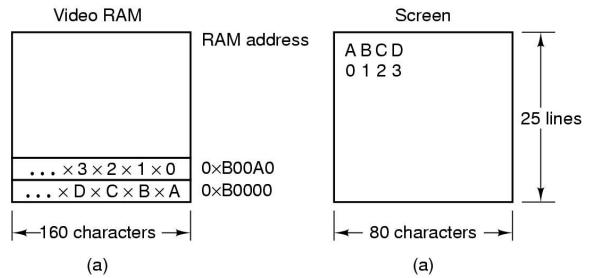
- Vector graphics 矢量设备
- Raster graphics 栅格设备
 - Char mode 、 Bit mode (pixel 像素 , bitmap 位图)

Memory-mapped displays

- driver writes directly into display's video RAM



GUI: Display Hardware 2/2



- A) A video RAM image
 - simple monochrome display
 - character mode
 - the Xs are attribute bytes
- B) Corresponding screen

OSP © LeeXudong@nankai.edu.cn

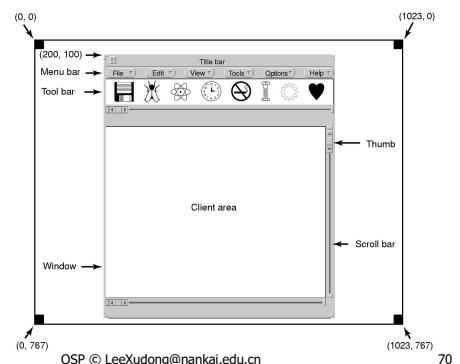
GUI: Input Software

- Keyboard driver delivers a number
 - 键盘扫描码 , ASCII 码
 - driver converts to characters
 - uses a ASCII table
- Exceptions, adaptations needed for other languages
 - 键盘映射 keymap or 代码页 code page
 - many OS provide for loadable keymaps or code pages

OSP © LeeXudong@nankai.edu.cn

GUI: Output Software

- Sample window located at (200,100) on XGA display



GUI: MS Windows 1/5

```
#include <windows.h>

int WINAPI WinMain(HINSTANCE h, HINSTANCE hprev, char *szCmd, int iCmdShow)
{
    WNDCLASS wndclass;           /* class object for this window */
    MSG msg;                    /* incoming messages are stored here */
    HWND hwnd;                  /* handle (pointer) to the window object */

    /* Initialize wndclass */
    wndclass.lpfnWndProc = WndProc;          /* tells which procedure to call */
    wndclass.lpszClassName = "Program name"; /* Text for title bar */
    wndclass.hIcon = LoadIcon(NULL, IDI_APPLICATION); /* load program icon */
    wndclass.hCursor = LoadCursor(NULL, IDC_ARROW); /* load mouse cursor */

    RegisterClass(&wndclass); /* tell Windows about wndclass */
    hwnd = CreateWindow( ... ); /* allocate storage for the window */
    ShowWindow(hwnd, iCmdShow); /* display the window on the screen */
    UpdateWindow(hwnd); /* tell the window to paint itself */
}
```

Skeleton of a Windows main program (part 1)

OSP © LeeXudong@nankai.edu.cn

GUI: MS Windows 2/5

```
while (GetMessage(&msg, NULL, 0, 0)) { /* get message from queue */
    TranslateMessage(&msg); /* translate the message */
    DispatchMessage(&msg); /* send msg to the appropriate procedure */
}
return(msg.wParam);
}

long CALLBACK WndProc(HWND hwnd, UINT message, UINT wParam, long lParam)
{
    /* Declarations go here. */

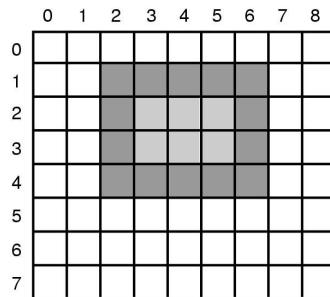
    switch (message) {
        case WM_CREATE: ...; return ...; /* create window */
        case WM_PAINT: ...; return ...; /* repaint contents of window */
        case WM_DESTROY: ...; return ...; /* destroy window */
    }
    return(DefWindowProc(hwnd, message, wParam, lParam));/* default */
}
```

Skeleton of a Windows main program (part 2)

OSP © LeeXudong@nankai.edu.cn

GUI: MS Windows 3/5

- Drawing
 - Lines, curves
 - Filled Areas
 - bitmaps
 - Displaying text
- An example rectangle drawn using *Rectangle*
 - Rectangle(hdc, xleft, ytop, xright, ybottom)*
 - Echo box represents one pixel



OSP © LeeXudong@nankai.edu.cn

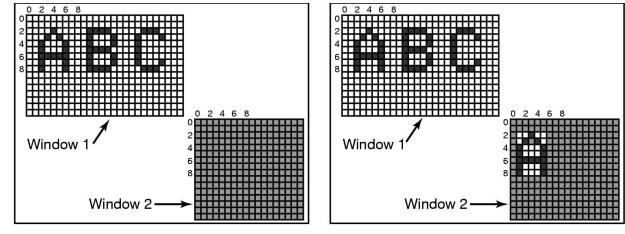
GUI: MS Windows 4/5

Bitmap 位图

- BitBlt(dst(hdc), dx, dy, width, height, src(hdc), sx, sy, rasteroperator)*

Copying bitmaps using *BitBlt*

- A) Before, B) After



(a) OSP © LeeXudong@nankai.edu.cn (b)

GUI: MS Windows 5/5

Fonts

- *BitBlit*: bitmap
- *TrueType*: outlines of the characters

20 pt: abcdefgh

53 pt: abcdefgh

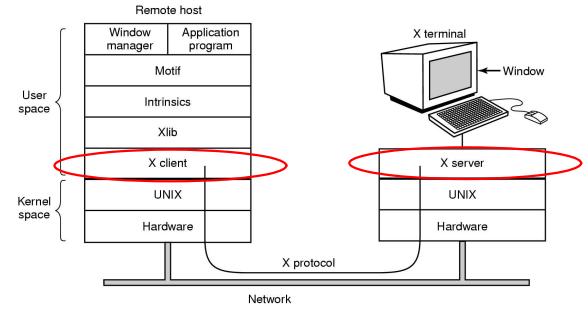
81 pt: abcdefgh

Examples of character outlines at different point sizes

OSP © LeeXudong@nankai.edu.cn

75

Network Terminals: X Window 1/3



Clients and servers in the M.I.T. X Window System

OSP © LeeXudong@nankai.edu.cn

```
#include <X11/Xlib.h>
#include <X11/Xutil.h>

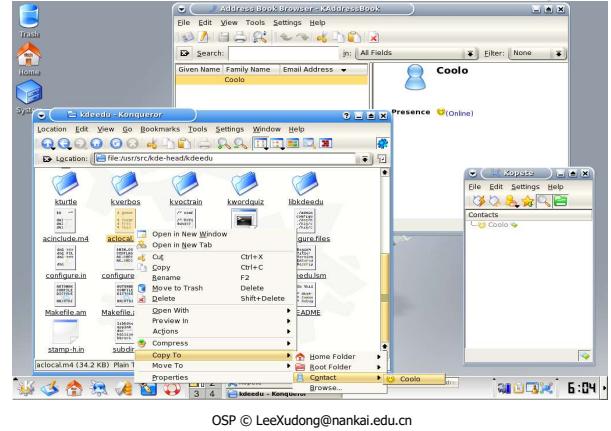
main(int argc, char *argv[])
{
    Display *disp; /* server identifier */
    Window win; /* window identifier */
    GC gc; /* graphic context identifier */
    XEvent event; /* storage for one event */
    int running = 1;

    disp = XOpenDisplay("display_name"); /* connect to the X server */
    win = XCreateSimpleWindow(disp, ...); /* allocate memory for new window */
    XSetStandardProperties(disp, ...); /* announces window to window mgr */
    gc = XCreateGC(disp, win, 0, 0); /* create graphic context */
    XSelectInput(disp, win, ButtonPressMask | KeyPressMask);
    XMapRaised(disp, win); /* display window; send Expose event */

    while (running) {
        XNextEvent(disp, &event); /* get next event */
        switch (event.type) {
            case Expose: ...; break; /* repaint window */
            case ButtonPress: ...; break; /* process mouse click */
            case KeyPress: ...; break; /* process keyboard input */
        }
    }

    XFreeGC(disp, gc); /* release graphic context */
    XDestroyWindow(disp, win); /* deallocate window's memory space */
    XCloseDisplay(disp); /* tear down network connection */
}
```

Network Terminals: X Window 3/3

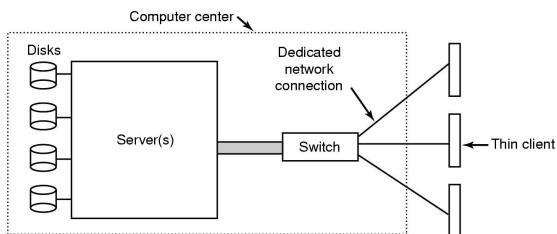


OSP © LeeXudong@nankai.edu.cn

Network Terminals: SLIM

- SLIM

- Stateless Low-level Interface Machine



OSP © LeeXudong@nankai.edu.cn

Network Terminals: SLIM

- SLIM Protocols

- SET
- FILL
- BITMAP
- COPY
- CSCS

OSP © LeeXudong@nankai.edu.cn

Power Management

- ENIAC

- 180000 vacuum tubes, 140000 watts of power

- PC

- 200-watt power supply, 85% efficient
- If 100 million of PC are turned on at once worldwide
 - 20000 megawatts: 20 nuclear power plants

- Battery-powered Computers

- Heart of the problem: the batteries cannot hold enough charge to last very long, a few hours at most
- No process: More Law

OSP © LeeXudong@nankai.edu.cn

Power: Reducing Energy Consumption

- Way 1:

- Turn off parts of the computer (mostly I/O devices) when they are not in use

- Way 2:

- The application program uses less energy
 - Possibly degrades the quality of the user experience, in order to stretch out battery time

OSP © LeeXudong@nankai.edu.cn

Power: Hardware Issues 1/2

- Batteries

- disposable 一次性使用的
- rechargeable 可再充电的

- Modes

- Working
- Sleeping
- Hibernating
- Off

OSP © LeeXudong@nankai.edu.cn

Power: Hardware Issues 2/2

- Power consumption of various parts of a laptop computer

Device	Li et al. (1994)	Lorch and Smith (1998)
Display	68%	39%
CPU	12%	18%
Hard disk	20%	12%
Modem		6%
Sound		2%
Memory	0.5%	1%
Other		22%

Top 3

OSP © LeeXudong@nankai.edu.cn

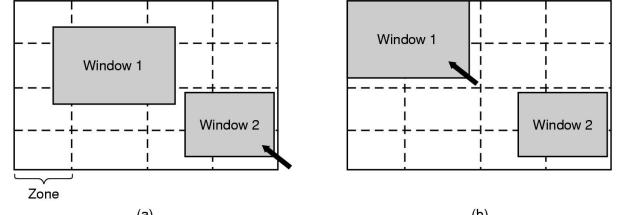
Problems of Power Management

- Which devices can be controlled?
- Are they on/off, or do they have intermediate states?
- How much power is saved in the low-power states?
- Is energy expended to restart the device?
- Must some context be saved when going to a low-power state?
- How long does it take to go back to full power?

OSP © LeeXudong@nankai.edu.cn

Power:Display

- To get a bright sharp image
 - The screen must be backlit and that takes substantial energy
- Way1:turn off the screen
- Way2:divide the screen into many zones, turn off part of zones
 - J. Flinn and M. Satyanarayanan, "Managing battery lifetime with energy-aware adaptation.,," ACM Trans. Comput. Syst., vol. 22, no. 2, pp. 137–179, 2004.



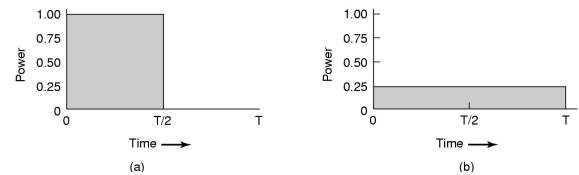
Power: hard disk

- Hard disk
 - It takes substantial energy to keep it spinning at high speed, even if there are no accesses
- Way1:Spin the disk down after a certain number of seconds or minutes of inactivity
- Way2:Have a substantial disk cache in RAM
- Way3:OS informs running programs about the disk state by sending it messages or signals

OSP © LeeXudong@nankai.edu.cn

Power:CPU

- CPU
 - Sleep
 - Cut voltage



- (a)Running at full clock speed
- (b)Cutting voltage by two: cuts clock speed by two, cuts power by four

OSP © LeeXudong@nankai.edu.cn

Power:Memory

- Way1:switch off the cache
- Way2:switch off the main memory
- Reloading
 - ...

OSP © LeeXudong@nankai.edu.cn

Power:Wireless Communication

- Wireless communication
 - Radio transmitter
 - Radio receiver
- Turn off the radio receiver
 - Base station,
 - Buffer?
- Turn off the radio transmitter
 - Buffer
- When should the radio be switched off?

OSP © LeeXudong@nankai.edu.cn

Power: Thermal Management

- Thermal 热量
 - Modern CPUs get extremely hot due to their high speed
- Electric Fan
 - Power of Electric Fan?
 - When should the fan be switched off/on?

OSP © LeeXudong@nankai.edu.cn

Power: Battery Management

- Smart Battery
 - Can communicate with the OS
 - Report states of the battery:
 - maximum voltage, current voltage
 - maximum charge(负荷), current charge
 - maximum drain(消耗) rate, current drain rate
- Multiple Batteries
- ACPI
 - Advanced Configuration and Power Interface

OSP © LeeXudong@nankai.edu.cn

Power: Degrade

- Degrade Operation 退化运行
 - OS tells the programs to use less energy, even if this means providing a poorer user experience
 - Better a poorer experience than no experience when the battery dies and the lights go out
- How a program can degrade its performance to save energy?
 - tradeoff

OSP © LeeXudong@nankai.edu.cn

Summary

- Principles of I/O Hardware
- Principles of I/O Software
- I/O Software Layers
- Disks
- Clocks
- User Interfaces
- Power Management

leexudong@nankai.edu.cn

Q&A?

