西安交通大学
软件学院

操作系统原理

Operating System Principle

田丽华

2-2 硬件保护

Hardware Protection
硬件保护

single-user programmer-operated system, complete control of the computer system, no protection单用户系统，程序员可完全控制系统

batch system, need some protection

- e.g. an indefinite loop reading cards不断读取卡片

multiprogramming, a program may accidentally or deliberately/maliciously modify the code or data of another program

Hardware Protection
硬件保护

multi-user environment, need protection for files, data on disk/tape
多用户环境，需要保护磁盘上的文件、数据

hardware trap to the OS, when
硬件陷入到OS，当发生以下情况：

01 illegal instructions, or access memory not in the address space, etc.
非法指令，或访问不属于自己的地址空间的内存

02 process terminated/aborted, core dumped 进程终止

Dual-Mode Operation
两状态操作

Sharing system resources requires operating system to ensure that an incorrect program cannot cause other programs to execute incorrectly.
共享系统资源要求操作系统确保有误程序不会引起其他程序的运行错误

Provide hardware support to differentiate between at least two modes of operations.
至少在两个运行状态之间提供硬件支持

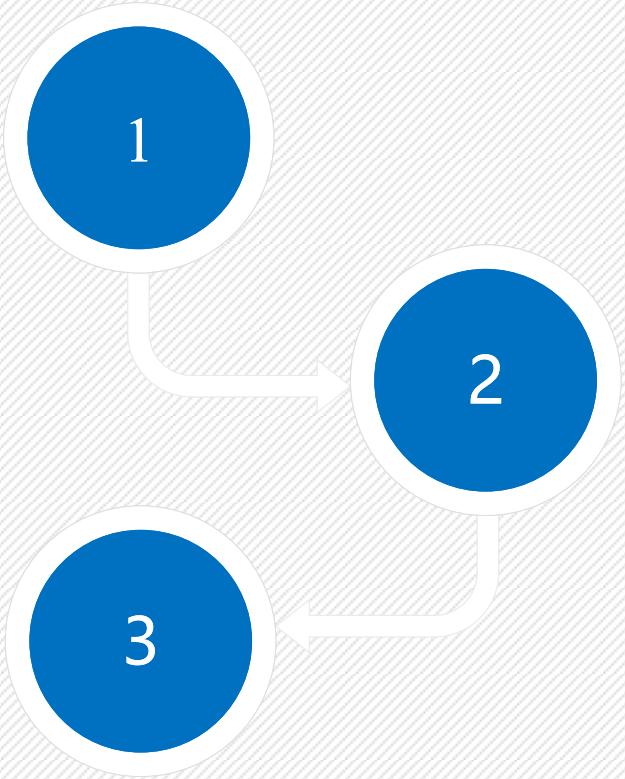
1. User mode – execution done on behalf of a user.
用户态-代表用户执行

2. Monitor mode (also supervisor mode or system mode) – execution done on behalf of operating system.
管态（特权模式或系统模式）-代表操作系统执行

Dual-Mode Operation
两状态操作

mode bit added to computer hardware to indicate the current mode: monitor (0) or user (1)
模式位添加到计算机硬件，表示当前模式

- mode bits in PSW



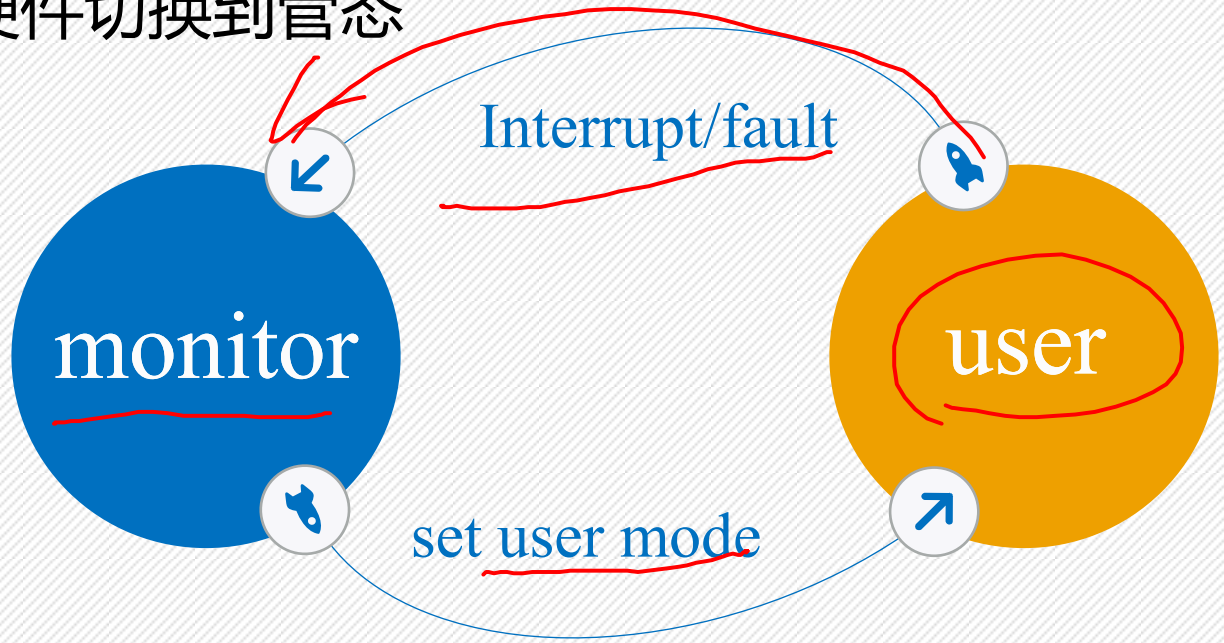
at boot time, the hardware starts in monitor mode
系统引导时，硬件处于管态

starts user processes in user mode
在用户模式下执行用户进程

Dual-Mode Operation
两状态操作

when an interrupt or fault occurs hardware switches to monitor mode

出现中断或陷阱，硬件切换到管态



dual-mode protect OS from errant users, and errant users from one another

- Privileged instructions can be issued only in monitor mode

Dual-Mode Operation
两状态操作

***privileged instructions*:特权指令**

in user mode:

always change mode to kernel

the way to implement *system call*

arguments passed via registers, stack, or memory

(with pointers in registers or stack)

in kernel mode

just do it!

I/O保护

I/O Protection

all I/O instructions are privileged instructions
所有的I/O指令都是特权指令

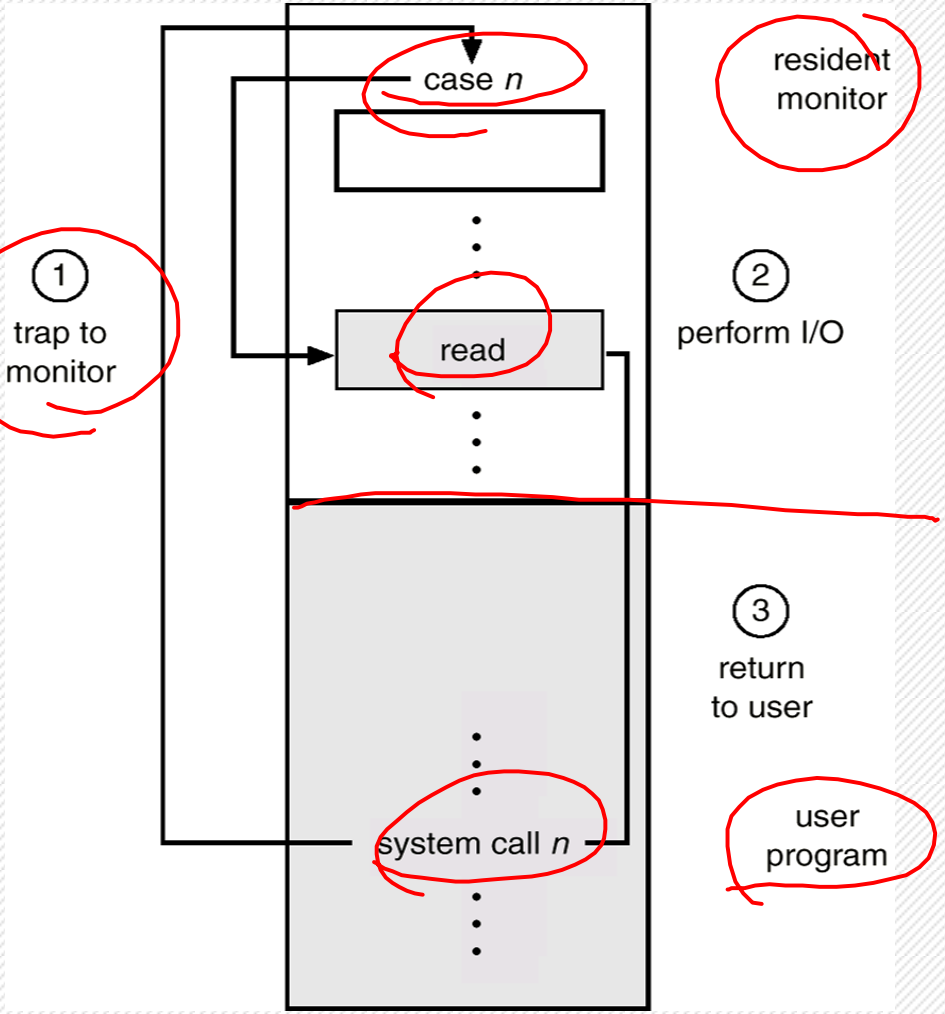
- memory mapped I/O, disk I/O (file access), display I/O, printer I/O, network I/O
- a user cannot issue I/O instructions directly, he must do it through system call
- 用户不能直接用i/o指令，必须通过系统调用

it must be ensured that a user program could never gain control of the computer in monitor mode
确保用户程序不能在管态下控制计算机

Dual-Mode Operation

I/O保护

Use of a system call to perform I/O



Dual-Mode Operation

内存保护

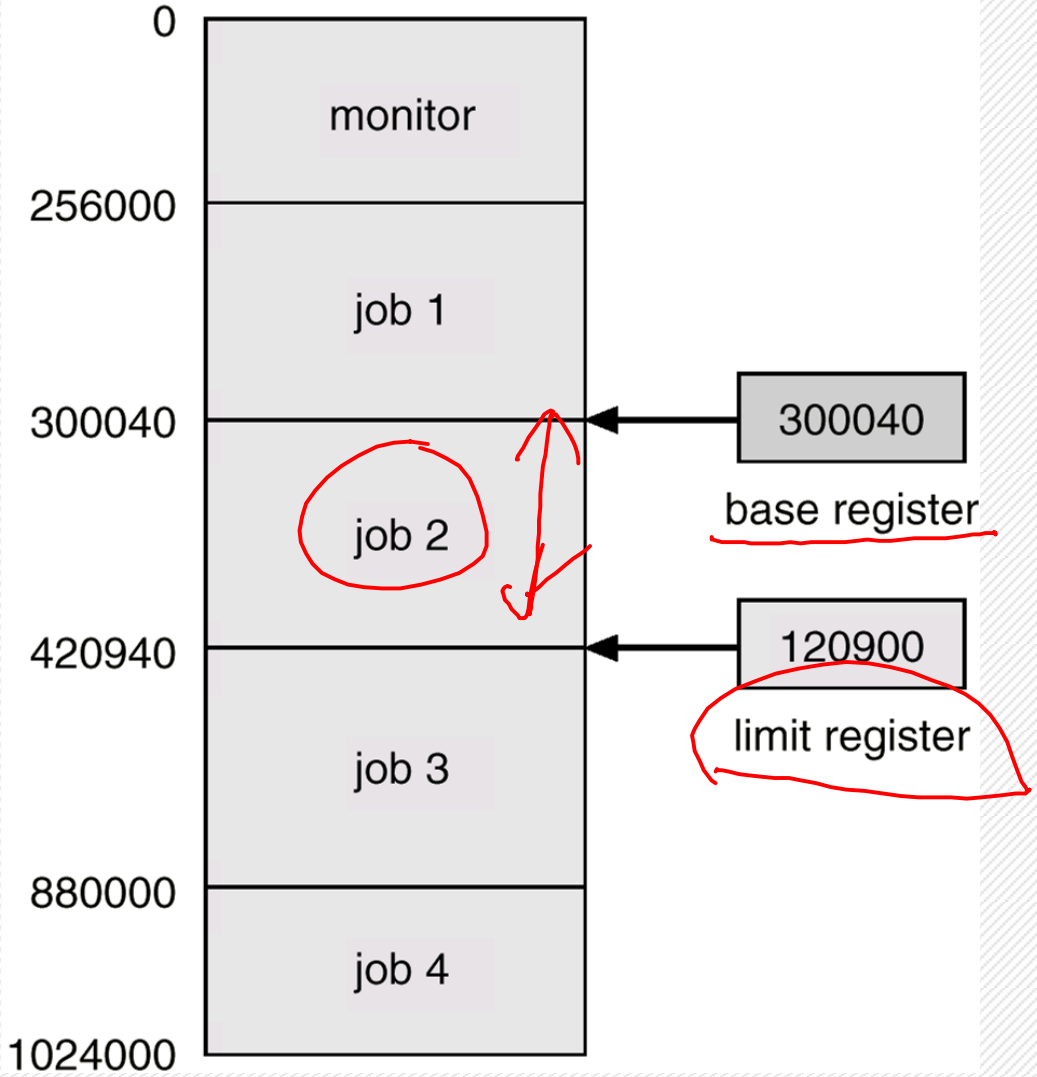
- must provide memory protection
  - at least for the interrupt vector and the interrupt service routines

    必须保护中断向量和中断服务程序
- in order to have memory protection, add two registers that determine the range of legal addresses a program may access

  确定进程能访问的合法空间
  - *base register* 基址寄存器 – holds the smallest legal physical memory address.
  - *limit register* 界限寄存器 – contains the size of the range
- memory outside the defined range is protected

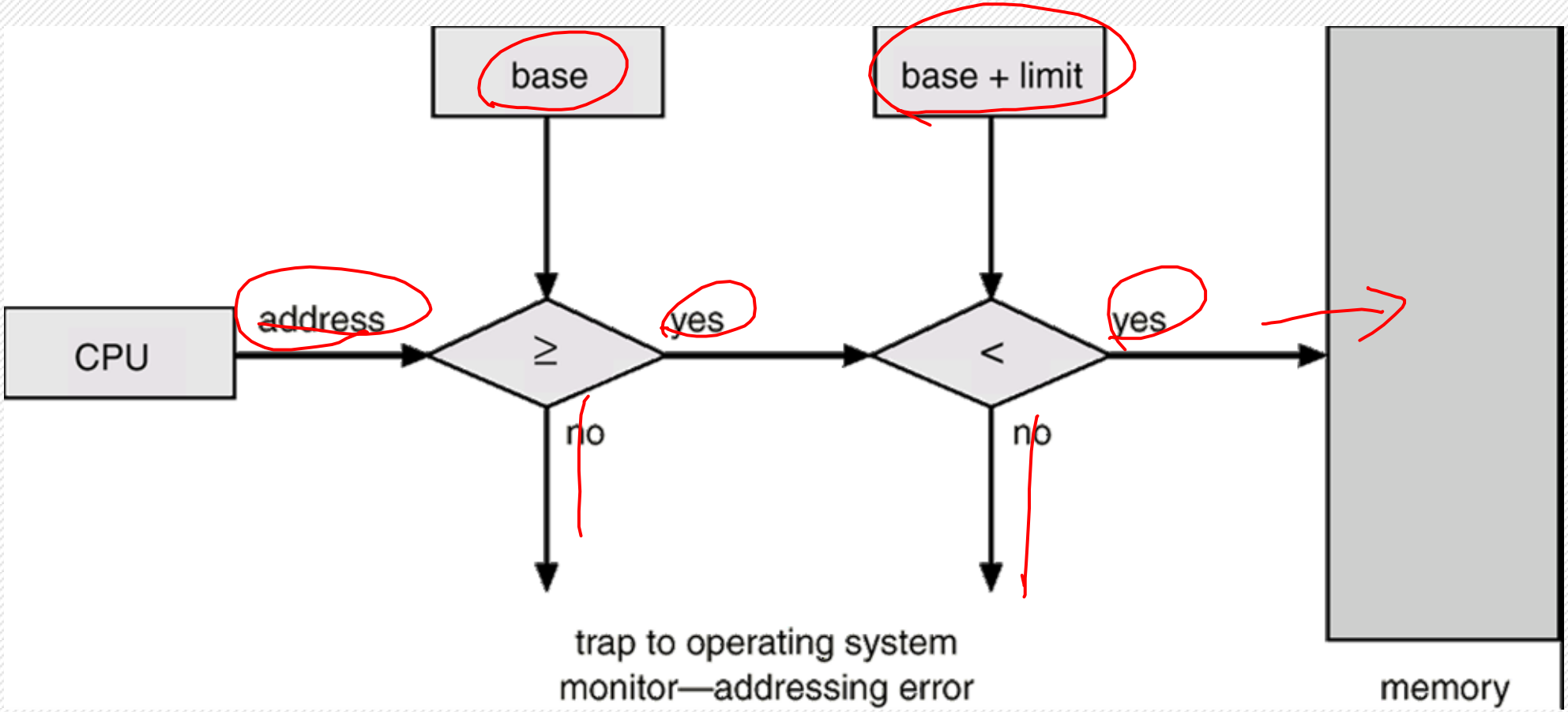
Dual-Mode Operation

内存保护



Dual-Mode Operation

内存保护



Memory Protection

- *every* address generated in every instruction in *user mode* is hardware checked
- 用户态下所生成的每个地址都要经过硬件检查
- base and limit registers can be loaded by only the OS by using a special privileged instruction
- when executing in monitor mode, the operating system has unrestricted access to both monitor and user's memory

Dual-Mode Operation

内存保护

Memory Protection

1. change the (base and limit) registers
2. load users' programs into users' memory
3. dump memory map of those programs in case of errors
4. access and modify parameters of system calls

Dual-Mode Operation

CPU保护

CPU protection

we must prevent a program from getting stuck in an infinite loop, or not calling OS services and never returning control to the OS

需防止用户程序陷入死循环或者不调用系统服务且不将控制权返回到OS

timer 定时器

interrupts computer after specified period to ensure operating system maintains control

Dual-Mode Operation

CPU保护

The timer

load-timer is a privileged instruction

before turning over control to the user's program, the OS ensures that the timer is set to interrupt

设定计时器以便产生中断

if the timer interrupts, control transfers to the OS, and the user program may be killed or given more time to run

定时器中断，控制权会返回给OS