



深圳大学
Shenzhen University

操作系统

第二讲 进程的描述与控制

谭舜泉

计算机与软件学院

第2章 进程的描述与控制

第一节 进程的描述

一、进程的引入

程序的顺序执行要求：

- **顺序性**：程序按次序逐步执行
- **封闭性**：独占全机资源，所有资源状态由程序完成
- **可再现性**：程序的执行结果不变，且与计算机的执行速度无关

第2章 进程的描述与控制

第一节 进程的描述

一、进程的引入

- 在多道程序系统、分时系统、实时系统等系统中，由于存在多个运行的程序，使程序的执行具有**并发性**、**共享性**、**异步性**等特征；

第2章 进程的描述与控制

第一节 进程的描述

一、进程的引入

程序的并发执行时的特征：

- **间断性**：多个程序相互制约（如输入程序与计算程序的先后关系）
- **失去封闭性**：多个程序共享资源（资源状态由多个程序改变）
- **不可再现性**：程序的执行结果与计算机的执行速度有关，也与其它程序的执行相关

第2章 进程的描述与控制

第一节 进程的描述

一、进程的引入

- 为了使程序在并发、共享、异步的环境下能正常运行，必须专门设置一个**控制数据区**，为程序保留运行的现场（进程标识、处理机状态、进程调度、进程控制等信息）
- **程序（纯代码段+数据段） + 程序专用控制数据区 = 进程**

第2章 进程的描述与控制

第一节 进程的描述

二、进程的定义与特征

1、进程的定义

- 进程是程序的一次执行
- 进程是进程实体(包括程序段、数据和PCB)的运行过程, 是系统进行资源分配和调度的一个独立单位

第2章 进程的描述与控制

第一节 进程的描述

二、进程的定义与特征

2、进程的特征

- **结构性**：由程序段、数据段和进程控制块组成
- **动态性**：进程可以被动态地创建、执行、撤消
- **并发性**：在同一时间内有多个进程运行
- **独立性**：是独立运行和获得资源的基本单位
- **异步性**：异步执行



第2章 进程的描述与控制

第一节 进程的描述

二、进程的定义与特征

3、进程与程序的区别

- 进程是**动态**的，程序是静态的（是指令的集合）
- 一个程序可以包含多个进程
- 进程可以描述并发活动，程序则不明显
- 进程执行需要**处理机**，程序存储需要介质
- 进程有**生命周期**，程序是永存的



第2章 进程的描述与控制

第一节 进程的描述

二、进程的定义与特征

4、进程的类型

(1)、系统进程

- 在管态（系统态、核心态）下运行的进程
- 它被分配一个初始的资源集合，这些资源可为它所独占，也可以有最高优先权优先使用
- 可以直接做显示、设备I/O操作
- 可以执行一切指令，访问所有寄存器和存储区

第2章 进程的描述与控制

第一节 进程的描述

二、进程的定义与特征

4、进程的类型

(2)、用户进程

- 在目态（用户态）下运行的进程
- 它通过系统服务请求的手段**竞争**系统资源
- 不能直接做I/O操作
- 只能执行规定的指令，访问指定的寄存器和存储区

第2章 进程的描述与控制

第一节 进程的描述

二、进程的定义与特征

4、进程的类型

(3)、**父进程**：由系统或用户首先创建的进程

(4)、**子进程**：由父进程创建的进程

第2章 进程的描述与控制

第一节 进程的描述

二、进程的定义与特征

4、进程的类型

(5)、父、子进程的关系

- **进程控制**：任何子进程只能由父进程创建及撤消，子进程不能对父进程实施控制
- **运行方式**：可同时执行，也可以等待子进程完成后，父进程再执行
- **资源共享**：子进程可全部或部分共享父进程的资源



第2章 进程的描述与控制

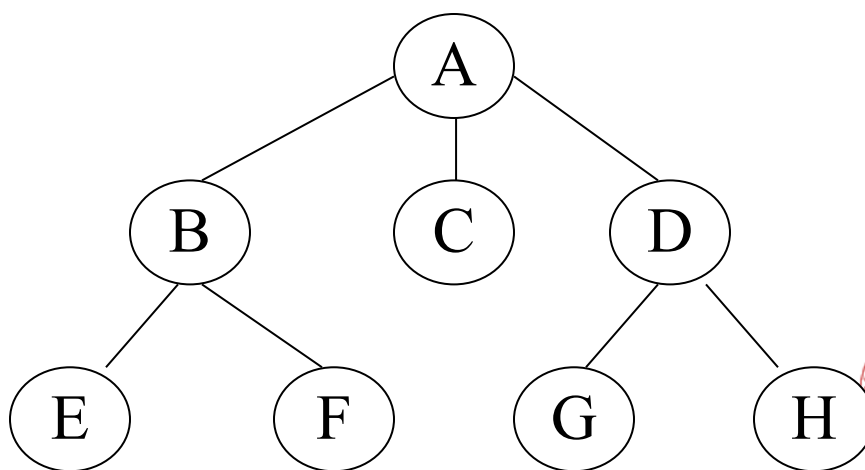
第一节 进程的描述

二、进程的定义与特征

4、进程的类型

(6)、进程关系图

- 父子（孙）进程之间构成一棵树形结构，即**进程树**



第2章 进程的描述与控制

第一节 进程的描述

二、进程的定义与特征

5、进程的结构

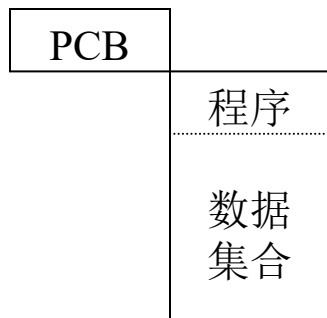
- 进程由**程序、数据和进程控制块**组成
- **程序**：描述进程所完成的功能（纯代码段）
- **数据集合**：程序运行时所需要的数据区
- **进程控制块（PCB）**：既能标识进程的存在，又能刻画进程瞬间特征的数据结构

第2章 进程的描述与控制

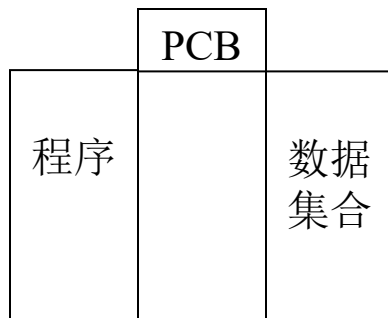
第一节 进程的描述

二、进程的定义与特征

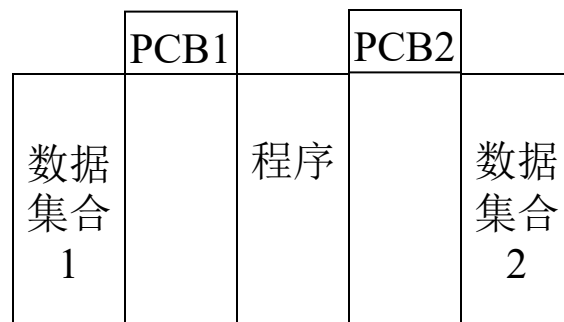
5、进程的结构



程序与数据集合放在连续内存中



程序与数据集合放在内存的不同区域



两个进程有各自的PCB和数据集合，但共享同一个程序



深圳大学
Shenzhen University

第2章 进程的描述与控制

第一节 进程的描述

三、进程的基本状态

1、三种基本状态

- **就绪状态**：进程获得了除处理机CPU之外的所有资源（可以有多个，排成**就绪队列**）
- **执行状态**：进程的程序正在处理机上执行（单CPU中只有一个**进程**处于该状态）
- **阻塞状态**：因发生某事件（如请求I/O，申请缓存空间等）而**暂停执行**的状态（也称为**睡眠状态**或**等待状态**）

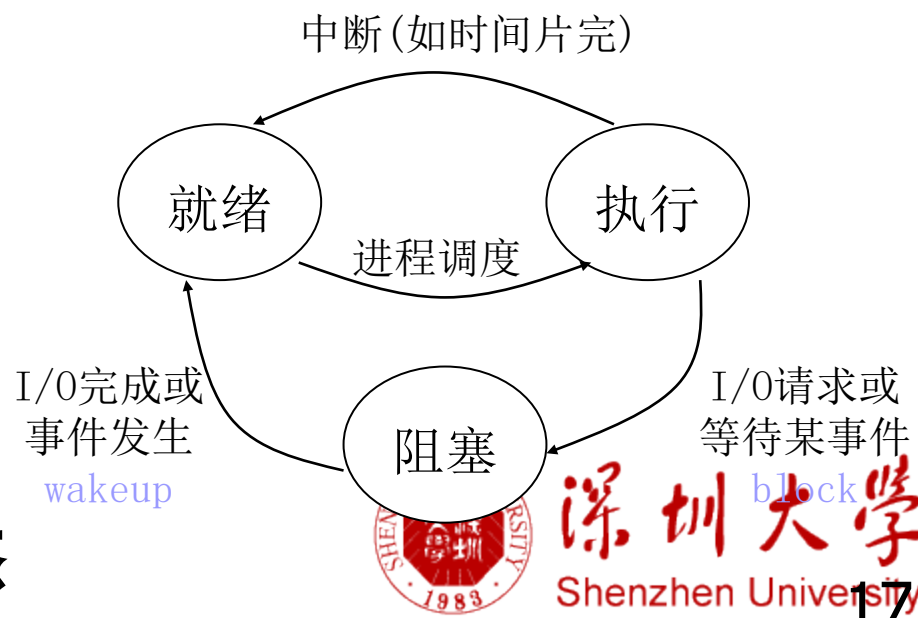
第2章 进程的描述与控制

第一节 进程的描述

三、进程的基本状态

2、进程基本状态的转换

- 处于执行状态的进程，可以主动用阻塞原语 (**block**) 将其变为阻塞状态
- 处于阻塞状态的进程，可以用唤醒 (**wakeup**) 原语将其变为就绪状态



第2章 进程的描述与控制

第一节 进程的描述

四、进程的挂起状态

1、挂起状态

- 挂起状态是一种静止状态，即不能马上投入运行的状态，包括静止就绪状态(Readys)和静止阻塞状态(Blockeds)
- 处于挂起状态的进程可以存放到外存上保留，而且可以回收这些挂起状态进程的内存、设备等部分资源

第2章 进程的描述与控制

第一节 进程的描述

四、进程的挂起状态

2、设置挂起状态的原因（p39）

- 终端用户的需要：（调试）
- 父进程的需求：（子进程同步）
- 负荷调节的需要：（减轻负荷）
- 操作系统的需要：（检查资源使用情况）

第2章 进程的描述与控制

第一节 进程的描述

四、进程的挂起状态

3、活动状态与挂起状态的转换

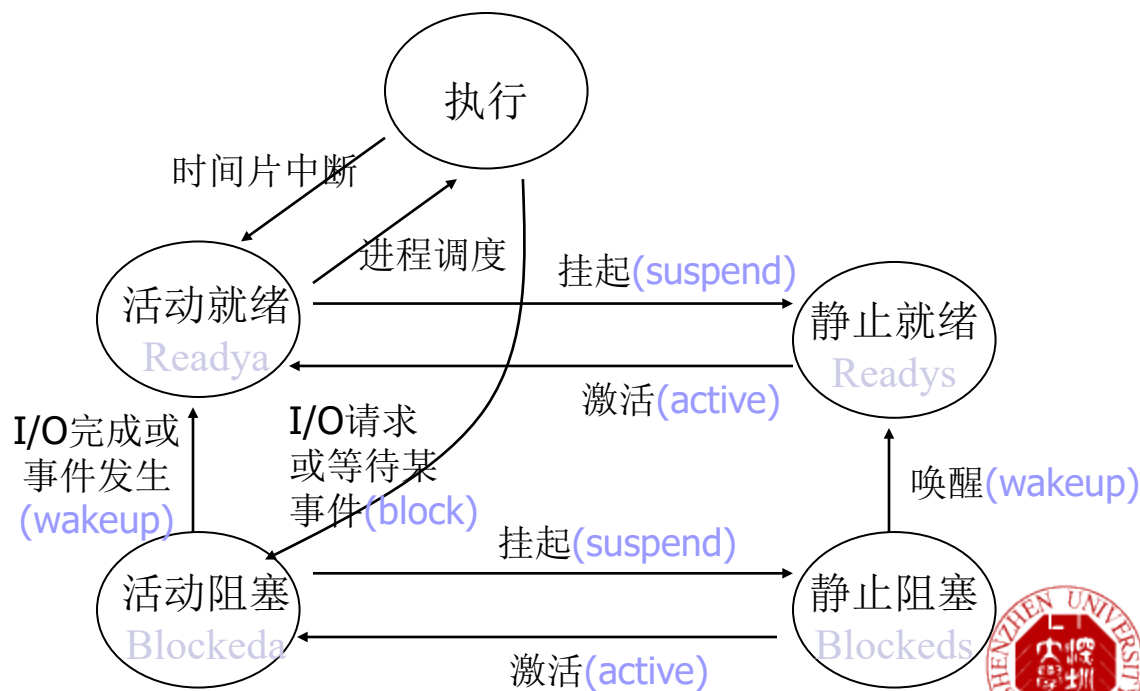
- 处于**活动状态**的进程 (Ready, Blocked) , 可以用挂起 (Suspend) 原语将其变为**挂起状态** (Readys, Blockededs)
- 处于挂起状态 (静止状态) 的进程, 可以用**激活 (Active)** 原语将其变为活动状态

第2章 进程的描述与控制

第一节 进程的描述

四、进程的挂起状态

5、具有挂起状态的进程状态转换



第2章 进程的描述与控制

第一节 进程的描述

五、进程控制块PCB

1、PCB的作用

- 描述进程的变化过程
- 记录进程的外部特征
- 记录进程与其他进程的联系
- 是进程存在的**唯一标志**
- 系统通过PCB控制和管理进程

第2章 进程的描述与控制

第一节 进程的描述

五、进程控制块PCB

2、PCB的内容

(1)、进程标识符信息

- 内部标识符信息（进程唯一序号）
- 外部标识符信息（由创建者提供的进程名字）

第2章 进程的描述与控制

第一节 进程的描述

五、进程控制块PCB

2、PCB的内容

(2)、处理机状态信息

- 处理机执行进程的现场信息
- 包括通用寄存器、指令计数器、程序状态字PSW、用户堆栈指针等信息

第2章 进程的描述与控制

第一节 进程的描述

五、进程控制块PCB

2、PCB的内容

(3)、进程调度信息

- 进程状态
- 进程优先级
- 进程调度所需的其它信息 (如进程已用CPU时间)
- 事件 (阻塞原因)

第2章 进程的描述与控制

第一节 进程的描述

五、进程控制块PCB

2、PCB的内容

(4)、进程控制信息 (p42)

- 程序和数据的地址
- 进程同步和通信机制（如消息队列指针等）
- 资源清单
- 链接指针（与其它PCB形成一个队列）

第2章 进程的描述与控制

第一节 进程的描述

五、进程控制块PCB

3、PCB的组织形式

(1)、线性表方式

- 将所有PCB不分状态，全部组织在一个连续表中
- 优点是简单
- 缺点是必须扫描整个PCB表
- 一般在进程较少的系统中使用

PCB1
PCB2
PCB3
PCB4
PCB5
PCB6
:
PCBn

第2章 进程的描述与控制

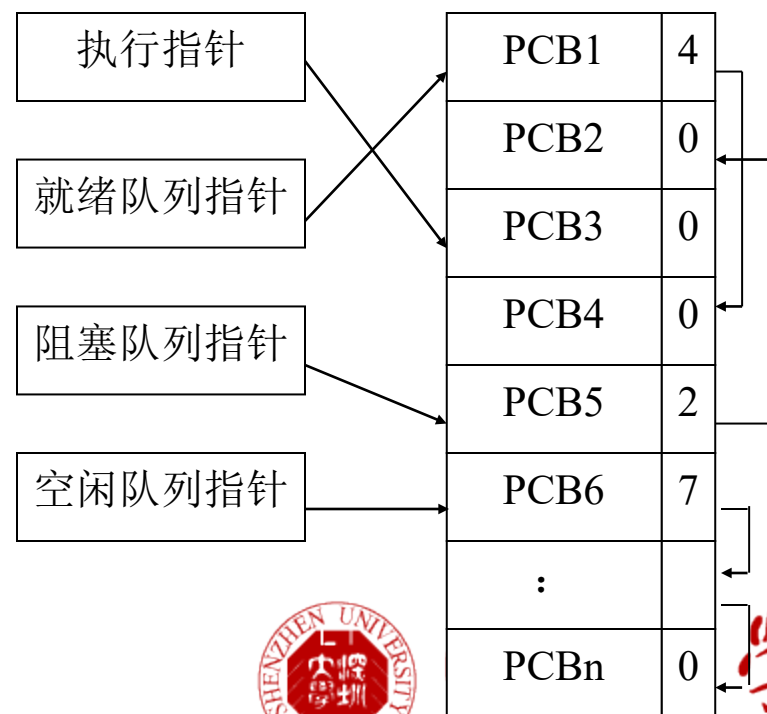
第一节 进程的描述

五、进程控制块PCB

3、PCB的组织形式

(2)、链接方式

- 把具有相同状态的PCB, 链接成一个队列
- 执行队列、就绪队列、阻塞队列、空闲队列



第2章 进程的描述与控制

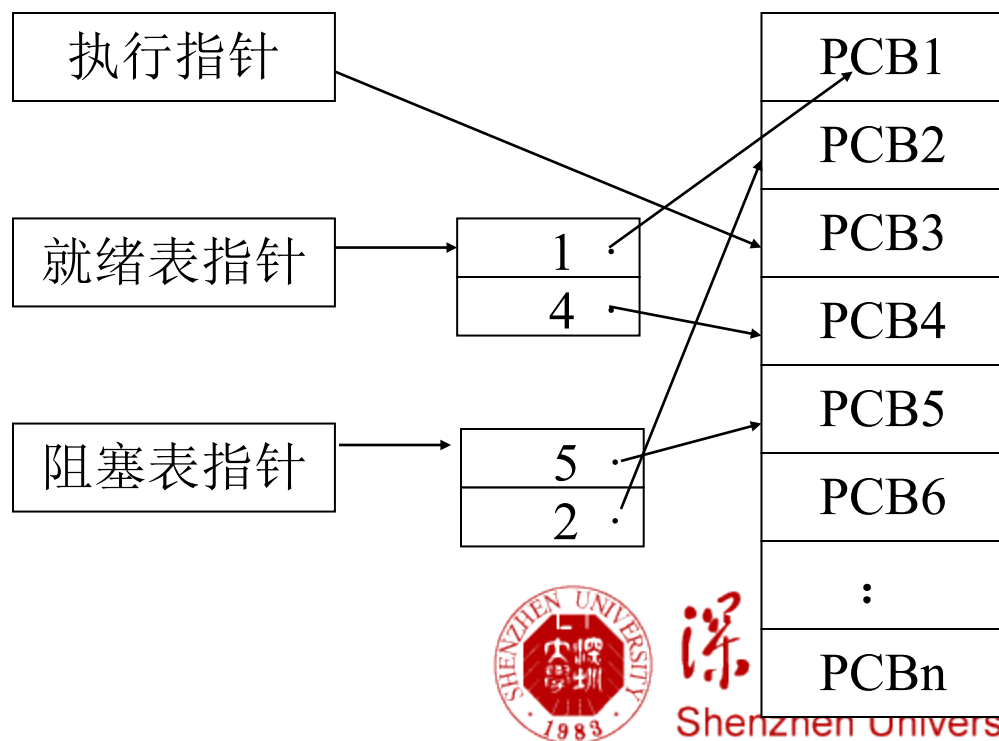
第一节 进程的描述

五、进程控制块PCB

3、PCB的组织形式

(3)、索引方式

- 把具有相同状态的PCB在PCB表中的首址，按顺序组成一个索引表（如就绪索引表、阻塞索引表）



第2章 进程的描述与控制

第二节 进程控制

一、原语操作

进程控制由原语操作实现。

- 原语是由若干条指令构成，用于完成一定功能的一个过程
- 原语操作是一种“原子操作”
- 原语操作中的所有动作，要么全做，要么全不做
- 原语操作是一种不可分割的操作
- 原语操作是OS内核执行基本操作的主要方法

第2章 进程的描述与控制

第二节 进程控制

二、进程的创建

1、引起进程创建的事件

- **用户登录**：用户合法登录后（分时系统）
- **作业调度**：某作业被调度后（批处理系统）
- **提供服务**：用户调用某些系统调用或命令后
- **应用请求**：由用户程序自己创建进程，如 Unix 中的 `fork()` 函数，Windows 中的 `CreateProcess()` 函数

第2章 进程的描述与控制

第二节 进程控制

二、进程的创建

2、进程创建原语

```
procedure create (pn, pri, res, fn, args);  
begin
```

```
    getfreepcb(i);  
    if i=NIL then return (NIL);  
    i.id := pn; i.priority := pri;  
    i.resources := res;  
    memallocate(datasetsize, add);  
    if add = NIL then  
        begin  
            pcbrelease (i);  
            return (NIL);  
        end;  
    i.dataadd := add; i.datasize:=  
    datasetsize;  
    datasetinit(i.dataadd, args);  
    filestate(fn, add, size);  
    if add = NIL then  
        begin
```

```
        memallocate(size, add);  
        if add = NIL then  
            begin  
                memrelease (i.dataadd,  
i.datasize);  
                pcbrelease(i); return(NIL);  
            end  
            read(fn, size, add);  
        end;  
        i.textadd := add; i.textsize := size;  
        i.prog := fn; i.pc := add;  
        i.children := 0; i.parent := EXE;  
        EXE.children := EXE.children+1;  
        i.state := "ready"; i.queue := RQ;  
        insert(RQ, i);  
        otherinit;  
        return(i);  
    End;
```

申请空白PCB

分配资源

初始化进程PCB

将新进程插入
就绪队列

原语说明:

Pn新进程的外部名, pri优先级, res初始资源, fn执行程序文件名, args是fn的参数表

add内存区地址, size内存区大小, RQ为就绪队列, EXE是执行态进程的PCB指针



深圳大学
Shenzhen University

第2章 进程的描述与控制

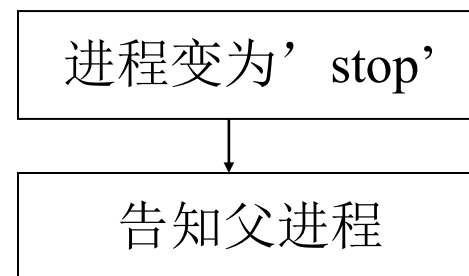
第二节 进程控制

三、进程的终止

1、进程正常结束

■ 通过停止原语实现

```
procedure halt(i);  
begin  
    i.state := 'stop';  
    send(i.parent, 'completed');  
end;
```



第2章 进程的描述与控制

第二节 进程控制

三、进程的终止

2、异常结束：

出现某些错误和故障而迫使进程终止（如越界、非法指令等）

3、外界干预：

用户或系统干预
父进程请求或终止

第2章 进程的描述与控制

第二节 进程控制

三、进程的终止

4、进程终止原语

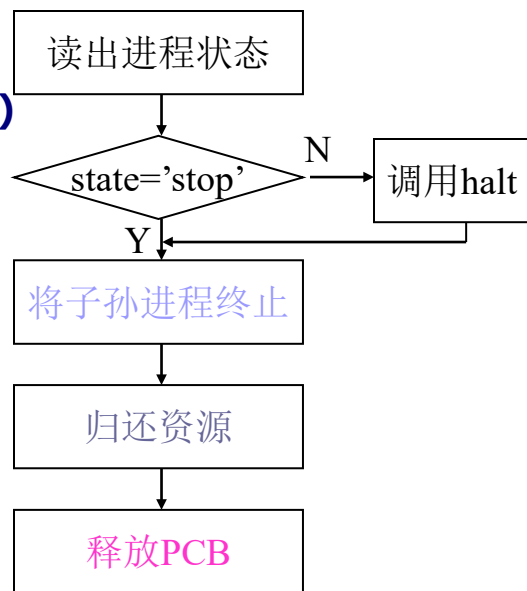
```
procedure destory(i);  
begin  
    if i.state <> 'stop' then  
        halt(i);  
    while i.children > 0 do  
    begin  
        i.children := i.children - 1;  
        findchild(i, child);  
        destory(child);  
    end;  
end;
```

```
memrelease(i.dataadd, i.datasize)  
close(i.prog, t);  
if t = true then  
    memrelease(i.textadd,  
i.textsize);  
resrelease(i);  
remove(i.queue, i);  
pcbrelease(i);
```

end; 原语说明:

remove将进程移出队列

findchild查找子进程



第2章 进程的描述与控制

第二节 进程控制

四、进程的阻塞

1、引起进程阻塞的事件

- 请求系统服务（如请求分配I/O）
- 启动某种操作（如启动I/O）
- 新数据尚未到达（如进程通信）
- 无新工作可做（系统进程）

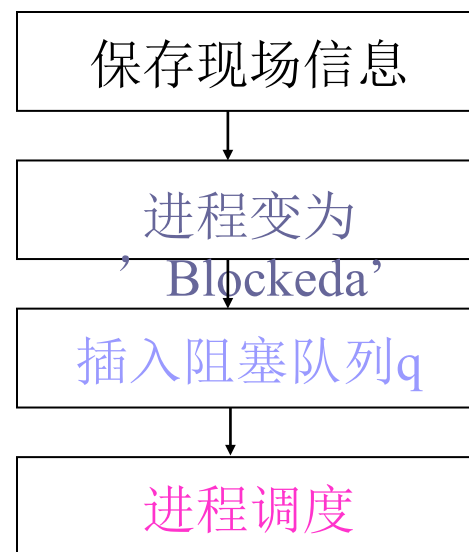
第2章 进程的描述与控制

第二节 进程控制

四、进程的阻塞

2、进程阻塞原语

```
procedure block(q);  
begin  
    save(EXE);  
    EXE.state := 'Blocked';  
    EXE.queue := q;  
    insert(q, EXE);  
    EXE := NIL;  
    scheduler;  
end;
```



原语说明:

insert将进程插入队列



深圳大学
Shenzhen University

第2章 进程的描述与控制

第二节 进程控制

五、进程的唤醒

1、引起进程唤醒的事件

- 系统请求实现（如获得I/O资源）
- 某种操作完成（如I/O操作完成）
- 新数据已经到达（如其它进程已将数据送达）
- 又有新工作可做（系统进程）

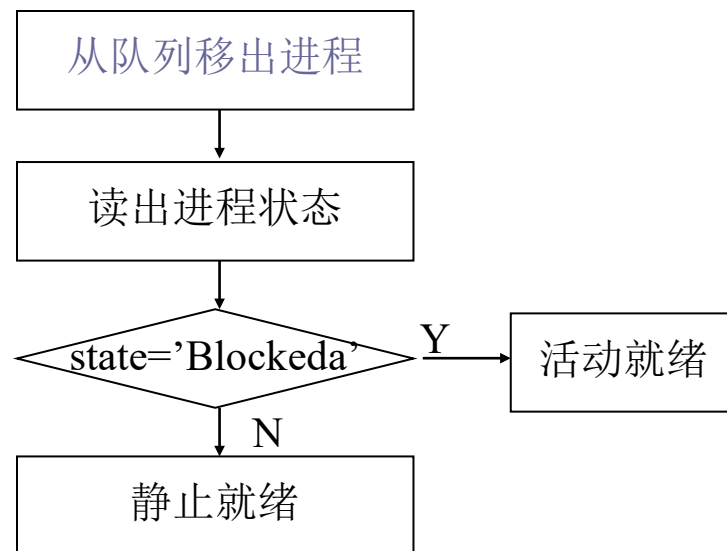
第2章 进程的描述与控制

第二节 进程控制

五、进程的唤醒

2、进程唤醒原语

```
procedure wakeup(q);  
begin  
    outqueue(q, i);  
    if i.state = 'Blocked' then  
        i.state = 'Ready';  
    else i.state = 'Ready';  
    i.queue := RQ;  
    insert(RQ, i);  
end;
```



原语说明:

outqueue从队列移出进程,
并返回该进程号

Readya活动就绪

Readys静止就绪



深圳大学
Shenzhen University

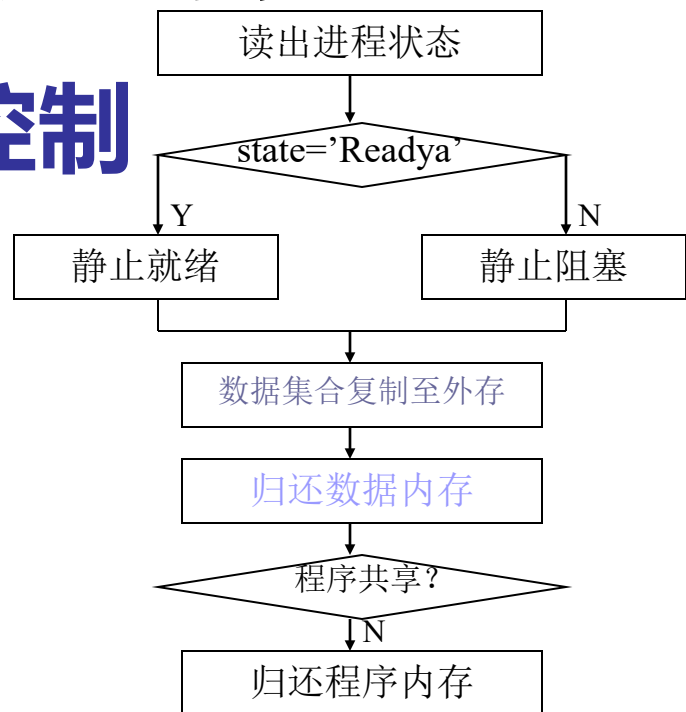
第2章 进程的描述与控制

第二节 进程控制

六、进程的挂起

■ 调用挂起原语suspend

```
procedure suspend(i);  
begin  
  if i.state = 'Readya'  
  then i.state='Readys' else i.state='Blocked';  
  swapout(i.dataadd, i.datasize, add)  
  i.swapadd := add;  
  memrelease(i.dataadd, i.datasize);  
  close(i.prog, t);  
  if t = true then memrelease(i.textadd,  
i.textsize);  
end;
```



原语说明:

swapout将数据集合复制到外存交换区, 并返回地址

Blocked_a活动阻塞

Blocked_s静止阻塞



深圳大学
Shenzhen University

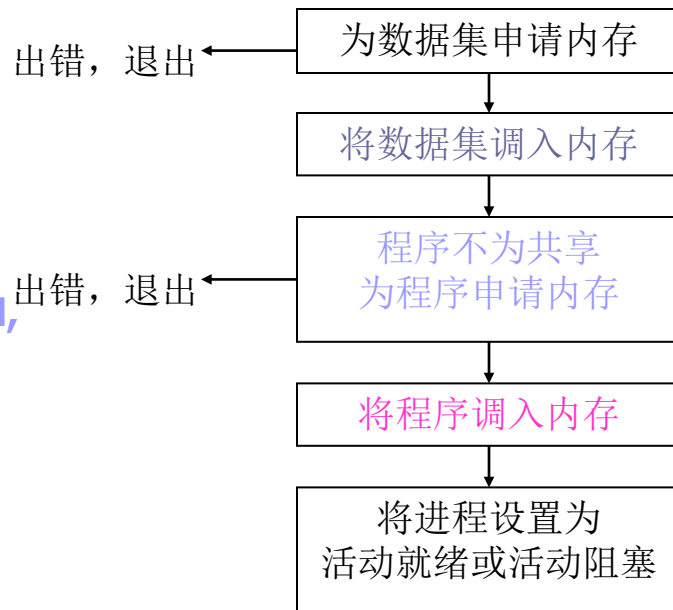
第2章 进程的描述与控制

第二节 进程控制

七、进程的激活

■ 调用激活原语active

```
procedure active(i);  
begin  
    memallocate(i.datasize, add);  
    if add = NIL then return(false);  
    swapin(i.swapadd, i.datasize, add);  
    i.dataadd := add;  
    filestate(i.prog, add, size);  
    if add = NIL then  
        begin  
            memallocate(size, add);  
            if add = NIL then  
                return(false);  
            else  
                return(true);  
        end;  
    end;  
    memrelease(i.dataadd, i.datasize);  
    return(false);  
end;  
read(i.prog, size, add);  
i.textadd := add;  
if i.state = 'Readys'  
then i.state := 'Readya'  
else i.state := 'Blocked'  
return(true);  
end;
```



原语说明:

swapin将数据集合从外存
交换区复制到内存



深圳大学
Shenzhen University