



南京大学
NANJING UNIVERSITY

计算机操作系统

6 并发程序设计 – 6.2 临界区管理

6.2.3 临界区管理实现的硬件方式

掌握临界区管理实现的硬件方式

测试并建立指令

```
TS(x) {  
    if (x==false) { x=true; return true;  
    } else return false;  
}
```

```
Boolean lock;  
    lock = false;                                // 临界区可用  
process Pi {                                     // i = 1,2,...,n  
    Boolean pi;  
    repeat pi=TS(lock) until pi;                // 循环请求锁  
    临界区;  
    lock = false;                                // 解锁  
}
```

对换指令

```
swap (a,b) { temp=a; a=b; b=temp; }
```

```
Boolean lock;
```

```
lock = false;
```

```
//临界区可用
```

```
process Pi {
```

```
// i = 1,2,...,n
```

```
Boolean pi;
```

```
pi = true;
```

```
repeat swap(lock, pi) until !pi; //循环请求锁  
临界区;
```

```
lock = false;
```

```
//解锁
```

```
}
```

实现临界区管理的硬件设施

- TS和swap指令均是忙式等待，效率低
- 简单的解决办法是在进出临界区时开关中断，这样临界区执行就不会中断了，执行就有原子性

关中断；临界区；开中断

- 操作系统原语就采用这种实现思路
- 但是，临界区的指令长度应该短小精悍，这样才能保证系统效率
- 不建议用户程序使用，**滥用是可怕的！**