第21讲 进程的并发控制
硬件实现方法

电子科技大学
University of Electronic Science and Technology of China

Approaches of Mutual Exclusion

- *Software Approaches*
- ***Hardware Support***
- Semaphores
- Monitors
- Message Passing

Mutual Exclusion: Hardware Support

- Interrupt Disabling
  - A process runs until it invokes an operating-system service or until it is interrupted.
  - Disabling interrupts guarantees mutual exclusion.
- The price of this approach is high.
- Multiprocessing
  - Disabling interrupts on one processor will not guarantee mutual exclusion.

Mutual Exclusion by Interrupt Disabling

$$\begin{tikzcd} {} \arrow[d] \\ \text{Disable interrupts} \arrow[d] \\ \text{critical section} \arrow[d] \\ \text{Enable interrupts} \end{tikzcd}$$

Mutual Exclusion: Hardware Support

- Special Machine Instructions
  - Performed in a single instruction cycle.
  - Not subject to interference from other instructions.(避免冲突)
  - Reading and writing
  - Reading and testing

Mutual Exclusion: Hardware Support

- Test and Set Instruction

```
function testset ( var i:integer ) : boolean ;
   begin
   if i = 0 then
     begin
       i := 1;
       testset := true;
     end
   else testset :=false;
  end.
```

Mutual Exclusion: Hardware Support

```
program mutualexclusion;
const n=…; /* 进程数 */
var bolt:integer;
procedure P(i:integer);
begin
  repeat
repeat  {nothing}  until  testset
     (bolt);
    < 临界区 >;
    bolt :=0;
    < 其余部分 >
  forever
end;
```

```
begin /*  主程序 */
  bolt :=0;
  parbegin
    P(1);
    P(2);
    …
    P(n)
  parend
end.
```

Mutual Exclusion: Hardware Support

Exchange Instruction

```
procedure exchange ( var r :register; var m :memory );
var temp;
begin
  temp := m;
  m := r;
  r := temp;
end.
```

Mutual Exclusion: Hardware Support

```
program mutualexclusion;
const n=…; /* 进程数 */
var bolt:integer;
procedure P(i:integer);
var key:integer;
begin
  repeat
     key:=1;
       repeat exchange(key,bolt) until
     key=0;
    < 临界区 >;
    bolt := 0;
    < 其余部分 >
  forever
 end;
```

```
begin /*  主程序 */
  bolt :=0;
  parbegin
    P(1);
    P(2);
    …
    P(n)
  parend
end.
```

Mutual Exclusion Machine Instructions

- **Advantages**
  - Applicable to any number of processes on either a single processor or multiple processors sharing main memory.
  - It is simple and therefore easy to verify.
  - It can be used to support multiple critical sections.

Mutual Exclusion Machine Instructions

- **Disadvantages**
  - **Busy-waiting is employed.**
  - **Starvation is possible.**
    - **when a process leaves a critical section and more than one process is waiting.**
  - **Deadlock is possible.**
    - **If a low priority process has the critical region and a higher priority process needs, the higher priority process will obtain the processor to wait for the critical region.**