临界区问题的硬件指令解决方案（Synchronization Hardware）

硬件同步机制

- CPU 制造者为临界区问题提供硬件支持
- 单处理器架构 – 利用“关中”途径

  关中后，当前执行的代码段（临界区）不会被强占

  “关中”法在多处理器架构中太低效，不解决问题

硬件同步机制（续）

- 提供硬件指令，体现原子操作 (atomic) 特征
  - Atomic = 不可中断

一种是“测试并赋值”
(test memory word and set value)

一种是“交换内存内容”
(swap contents of two memory words)

TestAndSet 指令

定义：

```
boolean TestAndSet (boolean *target)
{
    boolean rv = *target;
    *target = TRUE;
    return rv:
}
```

利用 TestAndSet 指令，设计解决方案

- 共享一个布尔变量 lock，初始化为 FALSE
- 算法：

```
while (true) {
        while ( TestAndSet (&lock ))
                ;   /* do nothing
                //    critical section
        lock = FALSE;
                //      remainder section
}
```

Swap 指令

定义：

```
void Swap (boolean *a, boolean *b)
{
        boolean temp = *a;
        *a = *b;
        *b = temp:
}
```

利用 Swap 指令，设计解决方案

- 共享一个布尔变量 lock，初始化为 FALSE
- 每个进程独有一个布尔变量 key

```
while (true) {
        key = TRUE;
        while ( key == TRUE)
                Swap (&lock, &key );
                  //    critical section
         lock = FALSE;
                  //      remainder section
 }
```

利用硬件指令的 N 进程临界区方案

- 无论运用 TestAndSet 指令，还是 Swap 指令，似乎都不能满足 Bounded Waiting 条件


End