Hardware Solutions to Critical Section Problem

- Peterson solution is not guranteed to work on modern computers with multiple cores.

- Use hardware instructions provided by modern Computers

- TestAndSet Instruction
- Swap Instruction

Important characterstics of these instructions :—

1) Two TestAndSet() instructions cannot be run in parallel on two different cpu's.

T1 T2
Tes Test

2) TestAndSet() instruction is executed atomically

→ 1) Parallel execution ←

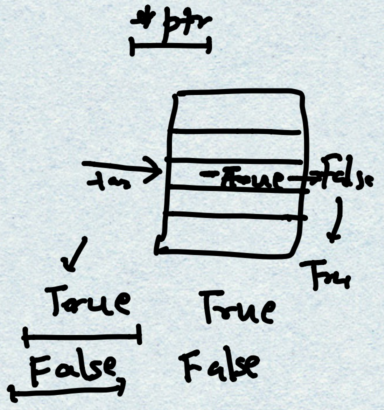
→ 2) Interrupts ←

```
lock = False
do {start
    while (testAndSet(&lock));
P1
  P2  // critical Section   P1
    lock = False
} while (True);
```

P1 P2 P1, P2

1) Mutual Exclusion ✓
2) Progress ←
3) Bounded wait

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



Fair — Many process can ~~stra~~ starve.

Wasting CPU cycles ←

Swap() Instruction For Critical Section Problem

1) Two process cannot run swap() instruction in parallel on two different cpu's.

2) swap() instruction is executed atomically.

i) Parallel execution

ii) Interrupts

```
lock = False                         P1
do {
    key = True;
    while(key == True) {
        swap(&lock, &key);
    }
    //Critical Section
    lock = False;
} while(True);              Re sleep()
```

P2   P2   1st   2nd

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

$\boxed{key == False}$ lock = F, key = T

$\boxed{Lock = False}$ lock = T, key = F

ME

Progress

Bounded wait