Consider checking:

\[ \text{Count} = (++\text{count}) \mod \text{limit} \]

"Assembly" code:

```
load count
load 1
add
load limit
remainder
store count SC
```
Context switch: execution transfer from $t_1$ to $t_2$ or $t_1$ to $t_1$
This is called race condition.

Java solves this problem through mutual exclusion, the policy of not allowing multiple threads into critical operations simultaneously.

The keyword synchronized accomplishes this:

```java
synchronized void critical()
{
}
```