MODULE *Mutex*

EXTENDS *Naturals*, *Sequences*, *FiniteSets*

CONSTANT N Number of processes

VARIABLES

pc,	Program counter for each process (noncritical, trying, critical)
$lock$,	Lock state (FREE or process id that holds it)
$queue$	Sequence of processes waiting for the lock

$vars \triangleq \langle pc, lock, queue \rangle$

$States \triangleq \{\text{“noncritical”}, \text{“trying”}, \text{“critical”}\}$
$Procs \triangleq 1 \,..\, N$

Helper operator to check if a process is in the queue
$InQueue(p) \triangleq \exists\, i \in 1 \,..\, Len(queue) : queue[i] = p$

$TypeOK \triangleq$
- $\land \quad pc \in [Procs \to States]$
- $\land \quad lock \in \{0\} \cup Procs$ 0 means FREE
- $\land \quad queue \in Seq(Procs)$ Queue is a sequence of process *IDs*

$Init \triangleq$
- $\land\, pc = [p \in Procs \mapsto \text{“noncritical”}]$
- $\land\, lock = 0$
- $\land\, queue = \langle\rangle$ Empty sequence

Try to acquire lock
$Try(p) \triangleq$
- $\land\, pc[p] = \text{“noncritical”}$
- $\land\, \neg InQueue(p)$ Not already in queue
- $\land\, pc' = [pc \text{ EXCEPT } ![p] = \text{“trying”}]$
- $\land\, queue' = Append(queue, p)$ Add to end of queue
- $\land$ UNCHANGED $lock$

Enter critical section if lock is free and first in queue
$Enter(p) \triangleq$
- $\land\, pc[p] = \text{“trying”}$
- $\land\, lock = 0$
- $\land\, Len(queue) > 0$ Queue not empty
- $\land\, Head(queue) = p$ Must be first in queue
- $\land\, lock' = p$
- $\land\, queue' = Tail(queue)$ Remove from front of queue
- $\land\, pc' = [pc \text{ EXCEPT } ![p] = \text{“critical”}]$

Exit critical section
$Exit(p) \triangleq$

$\land\ pc[p] = \text{“critical”}$
$\land\ lock = p$
$\land\ lock' = 0$
$\land\ pc' = [pc \text{ EXCEPT } ![p] = \text{“noncritical”}]$
$\land$ UNCHANGED $queue$

System transitions
$Next \triangleq \exists\, p \in Procs : Try(p) \lor Enter(p) \lor Exit(p)$

Fairness conditions
$Fairness \triangleq \forall\, p \in Procs :$
$\land\ \mathrm{WF}_{vars}(Try(p))$
$\land\ \mathrm{WF}_{vars}(Enter(p))$
$\land\ \mathrm{WF}_{vars}(Exit(p))$

Safety: mutual exclusion
$MutualExclusion \triangleq$
$\forall\, p1,\ p2 \in Procs :$
$(p1 \neq p2) \Rightarrow \neg(pc[p1] = \text{“critical”} \land pc[p2] = \text{“critical”})$

Liveness: if a process is trying, it eventually enters
$Liveness \triangleq$
$\forall\, p \in Procs : (pc[p] = \text{“trying”}) \leadsto (pc[p] = \text{“critical”})$

No starvation: if a process is in the queue, it eventually enters
$NoStarvation \triangleq$
$\forall\, p \in Procs : InQueue(p) \leadsto (pc[p] = \text{“critical”})$

Complete spec
$Spec \triangleq Init \land \quad \Box[Next]_{vars} \land Fairness$

THEOREM $Spec \Rightarrow \Box TypeOK$