MODULE *Exokernel*

EXTENDS *Naturals*, *Sequences*

Parameters and constants

CONSTANT *TaskCount*, *ResourceCount*

VARIABLE *tasks*, *resources*, *allocations*

Tasks are modeled as a sequence of task *IDs*, where each task has an index.

$Tasks \triangleq 1 \,..\, TaskCount$

Resources are modeled as a sequence of resource *IDs*, where each resource has an index.

$Resources \triangleq 1 \,..\, ResourceCount$

$TaskState \triangleq \{\text{“waiting”}, \text{“running”}\}$

$Availability \triangleq \{\text{TRUE}, \text{FALSE}\}$

Type invariant of the specification

tasks: function mapping from *Tasks* to *TaskState*
resources: function mapping from *Resources* to *Availability*
resources: function mapping from *Resources* to *Availability*

$$
\begin{aligned}
&TypeInvariant \triangleq \\
&\quad \land\ tasks \in [Tasks \to TaskState] \\
&\quad \land\ resources \in [Resources \to Availability] \\
&\quad \land\ allocations \in [Tasks \to [Resources \to \{\text{TRUE}, \text{FALSE}\}]]
\end{aligned}
$$

The state of the system is represented by the current allocation of resources to tasks.
$allocations[i][j] = \text{TRUE}$ means task i is using resource j.

$$
\begin{aligned}
&Init \triangleq \\
&\quad \land\ tasks = [t \in Tasks \mapsto \text{“waiting”}] \\
&\quad \land\ resources = [r \in Resources \mapsto \text{TRUE}] \quad \text{All resources are available} \\
&\quad \land\ allocations = [t \in Tasks \mapsto [r \in Resources \mapsto \text{FALSE}]]
\end{aligned}
$$

A task can request access to a resource if it's not already in use.

$$
\begin{aligned}
&RequestResource(t,\ r) \triangleq \\
&\quad \land\ resources[r] = \text{TRUE} \\
&\quad \land\ allocations[t][r] = \text{FALSE} \\
&\quad \land\ tasks[t] = \text{“waiting”} \\
&\quad \land\ tasks' = [tasks \ \text{EXCEPT}\ ![t] = \text{“running”}]
\end{aligned}
$$

$\land\ resources' = [resources \text{ EXCEPT } ![r] = \text{FALSE}]$
$\land\ allocations' = [allocations \text{ EXCEPT } ![t][r] = \text{TRUE}]$

A task can release a resource when it has finished using it.

$ReleaseResource(t,\ r) \triangleq$
$\land\ allocations[t][r] = \text{TRUE}$
$\land\ tasks[t] = \text{“running”}$
$\land\ tasks' = [tasks \text{ EXCEPT } ![t] = \text{“waiting”}]$
$\land\ resources' = [resources \text{ EXCEPT } ![r] = \text{TRUE}]$
$\land\ allocations' = [allocations \text{ EXCEPT } ![t][r] = \text{FALSE}]$

Task behavior: tasks can either request or release resources

$TaskAction \triangleq$
$\exists\, t \in Tasks,\ r \in Resources :$
$(RequestResource(t,\ r) \lor ReleaseResource(t,\ r))$

$DoNothing \triangleq \text{UNCHANGED } \langle tasks,\ resources,\ allocations \rangle$

The next-state relation defines valid state transitions

$Next \triangleq TaskAction \lor DoNothing$

Specification of the overall system behavior

$Spec \triangleq Init \land \Box[Next]_{\langle tasks,\ resources,\ allocations \rangle}$

$vars \triangleq \langle tasks,\ resources \rangle$

Fairness $\triangleq$ WF_$vars(TaskAction)$

Invariant: No two tasks can hold the same resource simultaneously

$ResourceExclusivity \triangleq$
$\forall\, r \in Resources :$
$\forall\, t1,\ t2 \in Tasks : (t1 \#\ t2) \Rightarrow \neg(allocations[t1][r] \land allocations[t2][r])$

ASSUME Assumption $\triangleq tasks \in [Tasks \to TaskState]$

THEOREM $Init \Rightarrow$ Fairness
BY DEF $Init$, Fairness
