MODULE *BlockingQueue*

EXTENDS *Naturals*, *Sequences*, *TLAPS*

CONSTANTS *Producers*, *Consumers*, *BufCapacity*

ASSUME $Assumptions \triangleq \land Producers \neq \{\} \land Consumers \neq \{\}$
$\land (Consumers \cap Producers) = \{\}$
$\land BufCapacity \in (Nat \setminus \{0\})$

$data \triangleq$ CHOOSE $d : d$ Some data.

VARIABLES *buffer*, *waitC*, *waitP*

$vars \triangleq \langle buffer, waitC, waitP \rangle$

$TypeOK \triangleq \land Len(buffer) \in 0 \,..\, BufCapacity$
$\land waitP \in$ SUBSET $Producers$
$\land waitC \in$ SUBSET $Consumers$

$NoDeadlock \triangleq (waitC \cup waitP) \neq (Producers \cup Consumers)$

$Notify(ws) \triangleq$ IF $ws \neq \{\}$
THEN $\exists\, x \in ws : ws' = ws \setminus \{x\}$
ELSE UNCHANGED ws

$Wait(ws,\, t) \triangleq \land ws' = ws \cup \{t\}$
$\land$ UNCHANGED $buffer$

$Put(t,\, d) \triangleq \lor \land Len(buffer) < BufCapacity$
$\land buffer' = Append(buffer,\, d)$
$\land Notify(waitC) \land$ UNCHANGED $waitP$
$\lor \land Len(buffer) = BufCapacity$
$\land Wait(waitP,\, t) \land$ UNCHANGED $waitC$

$Get(t) \triangleq \lor \land buffer \neq \langle\rangle$
$\land buffer' = Tail(buffer)$
$\land Notify(waitP) \land$ UNCHANGED $waitC$
$\lor \land buffer = \langle\rangle$
$\land Wait(waitC,\, t) \land$ UNCHANGED $waitP$

$Init \triangleq buffer = \langle\rangle \land waitC = \{\} \land waitP = \{\}$

$Next \triangleq \lor \exists\, t \in (Producers \setminus waitP) : Put(t,\, data)$
$\lor \exists\, t \in (Consumers \setminus waitC) : Get(t)$

$Spec \triangleq Init \land \Box[Next]_{vars}$

Scaffolding: Establish that *TypeOK* is inductive.

LEMMA $ITypeInv \triangleq Spec \Rightarrow \Box TypeOK$

$\langle 1 \rangle$ USE $Assumptions$ DEF $TypeOK$

$\langle 1 \rangle 1$. $Init \Rightarrow TypeOK$

BY DEF $Init$

$\langle 1 \rangle 2$. $TypeOK \wedge [Next]_{vars} \Rightarrow TypeOK'$

BY DEF $Next$, $vars$, Put, Get, $Notify$, $Wait$

$\langle 1 \rangle$.QED BY $\langle 1 \rangle 1$, $\langle 1 \rangle 2$, PTL DEF $Spec$

An inductive invariant that implies *NoDeadlock*.

$$\begin{aligned} IInv \triangleq\ & \wedge TypeOK \\ & \wedge NoDeadlock \\ & \text{This is the meat!} \\ & \wedge buffer = \langle\rangle \Rightarrow (Producers \setminus waitP) \neq \{\} \\ & \wedge Len(buffer) = BufCapacity \Rightarrow (Consumers \setminus waitC) \neq \{\} \end{aligned}$$

Proof that *Spec* is deadlock-free.

THEOREM $DeadlockFreedom \triangleq Spec \Rightarrow \Box IInv$

$\langle 1 \rangle$ USE $Assumptions$ DEF $IInv$, $NoDeadlock$, $TypeOK$

$\langle 1 \rangle 1$. $Init \Rightarrow IInv$

BY DEF $Init$

$\langle 1 \rangle 2$. $IInv \wedge [Next]_{vars} \Rightarrow IInv'$

BY DEF $Next$, $vars$, Put, Get, $Notify$, $Wait$

$\langle 1 \rangle 3$. $IInv \Rightarrow NoDeadlock$

BY DEF $IInv$

$\langle 1 \rangle 4$. QED

BY $\langle 1 \rangle 1$, $\langle 1 \rangle 2$, $\langle 1 \rangle 3$, PTL DEF $Spec$