

EXTENDS *Integers, Sequences*

CONSTANT *Data*

We first define *Remove*(*i, seq*) to be the sequence obtained by removing element number *i* from sequence *seq*.

$$\begin{aligned} \text{Remove}(i, seq) &\triangleq \\ &[j \in 1 \dots (\text{Len}(seq) - 1) \mapsto \text{IF } j < i \text{ THEN } seq[j] \\ &\quad \text{ELSE } seq[j + 1]] \end{aligned}$$

VARIABLES *AVar, BVar,* The same as in module *ABSpec*
AtoB, The sequence of data messages in transit from sender to receiver.
BtoA The sequence of ack messages in transit from receiver to sender.
 Messages are sent by appending them to the end of the sequence.
 and received by removing them from the head of the sequence.

$$\text{vars} \triangleq \langle AVar, BVar, AtoB, BtoA \rangle$$

$$\begin{aligned} \text{TypeOK} &\triangleq \wedge AVar \in Data \times \{0, 1\} \\ &\quad \wedge BVar \in Data \times \{0, 1\} \\ &\quad \wedge AtoB \in Seq(Data \times \{0, 1\}) \\ &\quad \wedge BtoA \in Seq(\{0, 1\}) \end{aligned}$$

$$\begin{aligned} \text{Init} &\triangleq \wedge AVar \in Data \times \{1\} \\ &\quad \wedge BVar = AVar \\ &\quad \wedge AtoB = \langle \rangle \\ &\quad \wedge BtoA = \langle \rangle \end{aligned}$$

The action of the sender sending a data message by appending *AVar* to the end of the message queue *AtoB*. It will keep sending the same message until it receives an acknowledgment for it from the receiver.

$$\begin{aligned} \text{ASnd} &\triangleq \wedge AtoB' = \text{Append}(AtoB, AVar) \\ &\quad \wedge \text{UNCHANGED } \langle AVar, BtoA, BVar \rangle \end{aligned}$$

The action of the sender receiving an ack message. If that ack is for the value it is sending, then it chooses another message to send and sets *AVar* to that message. If the ack is for the previous value it sent, it ignores the message. In either case, it removes the message from *BtoA*.

$$\begin{aligned} \text{ARcv} &\triangleq \wedge BtoA \neq \langle \rangle \\ &\quad \wedge \text{IF } \text{Head}(BtoA) = AVar[2] \\ &\quad \quad \text{THEN } \exists d \in Data : AVar' = \langle d, 1 - AVar[2] \rangle \\ &\quad \quad \text{ELSE } AVar' = AVar \\ &\quad \wedge BtoA' = \text{Tail}(BtoA) \\ &\quad \wedge \text{UNCHANGED } \langle AtoB, BVar \rangle \end{aligned}$$

The action of the receiver sending an acknowledgment message for the last data item it received.

$$\begin{aligned} \text{BSnd} &\triangleq \wedge BtoA' = \text{Append}(BtoA, BVar[2]) \\ &\quad \wedge \text{UNCHANGED } \langle AVar, BVar, AtoB \rangle \end{aligned}$$

The action of the receiver receiving a data message. It sets $BVar$ to that message if it's not for the data item it has already received.

$$\begin{aligned} BRcv \triangleq & \wedge AtoB \neq \langle \rangle \\ & \wedge \text{IF } Head(AtoB)[2] \neq BVar[2] \\ & \quad \text{THEN } BVar' = Head(AtoB) \\ & \quad \text{ELSE } BVar' = BVar \\ & \wedge AtoB' = Tail(AtoB) \\ & \wedge \text{UNCHANGED } \langle AVar, BtoA \rangle \end{aligned}$$

$LoseMsg$ is the action that removes an arbitrary message from queue $AtoB$ or $BtoA$.

$$\begin{aligned} LoseMsg \triangleq & \wedge \vee \wedge \exists i \in 1 \dots Len(AtoB) : \\ & \quad AtoB' = Remove(i, AtoB) \\ & \quad \wedge BtoA' = BtoA \\ & \vee \wedge \exists i \in 1 \dots Len(BtoA) : \\ & \quad \quad BtoA' = Remove(i, BtoA) \\ & \quad \wedge AtoB' = AtoB \\ & \wedge \text{UNCHANGED } \langle AVar, BVar \rangle \end{aligned}$$

$$Next \triangleq ASnd \vee ARcv \vee BSnd \vee BRcv \vee LoseMsg$$

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

$$ABS \triangleq \text{INSTANCE } ABSpec$$

$$\text{THEOREM } Spec \Rightarrow ABS!Spec$$

$FairSpec$ is $Spec$ with fairness conditions added.

$$\begin{aligned} FairSpec \triangleq & Spec \wedge SF_{vars}(ARcv) \wedge SF_{vars}(BRcv) \wedge \\ & WF_{vars}(ASnd) \wedge WF_{vars}(BSnd) \end{aligned}$$

\ * Modification History
\ * Last modified Sat Jun 11 19:31:09 CST 2022 by wengjialin
\ * Last modified Wed Dec 27 13:29:51 PST 2017 by lamport
\ * Created Wed Mar 25 11:53:40 PDT 2015 by lamport