MODULE *IPCProtocol*

In *Linux* versions before 2.6.11, the capacity of a pipe was the same as the system page size (*e.g.*, 4096 bytes on *i*386). Since *Linux* 2.6.11, the pipe capacity is 16 pages (*i.e.*, 65,536 bytes in a system with a page size of 4096 bytes).

EXTENDS *Naturals*, *Sequences*

VARIABLES *inQueueIn*, *inQueueOut*, *inQueue*,
outQueueIn, *outQueueOut*, *outQueue*

CONSTANT *Message*, *MessagePairs*, *N*
ASSUME $(N \in Nat) \land (N > 0)$ Both queues have the same number of messages
ASSUME $(MessagePairs \in [msgIn : Message,\ msgOut : Message])$

A simple type invariant

$$\begin{aligned} TypeOK \triangleq\ & \land\ MessagePairs \in [msgIn : Message,\ msgOut : Message] \\ & \land\ \forall\, msgPair \in MessagePairs : msgPair.msgIn \neq msgPair.msgOut \\ & \land\ inQueue \in Seq(Message) \\ & \land\ outQueue \in Seq(Message) \end{aligned}$$

Util function

$$Last(s) \triangleq s[Len(s)]$$

$InQueue \triangleq$ INSTANCE *BoundedFIFO* WITH $in \leftarrow inQueueIn,\ out \leftarrow inQueueOut,\ q \leftarrow inQueue$
$OutQueue \triangleq$ INSTANCE *BoundedFIFO* WITH $in \leftarrow outQueueIn,\ out \leftarrow outQueueOut,\ q \leftarrow outQueue$

Make sure that if the in queue is non-empty, given some length in queue of x,
out queue will eventually reach a point where it will be at least that size, if not greater

$$\begin{aligned} MsgTrans \triangleq\ & \forall\, x \in Nat : \\ & (Len(inQueue) > 0) \Rightarrow Len(inQueue) = x \rightsquigarrow Len(outQueue) \geq x \end{aligned}$$

Make sure that once the message goes in, it must go out as it's pair

$$\begin{aligned} MsgIncl \triangleq\ & \forall\, msgPair \in MessagePairs : \\ & (Len(inQueue) > 0) \Rightarrow Last(inQueue) = msgPair.msgIn \rightsquigarrow Head(outQueue) = msgPair.msgOut \end{aligned}$$

$$\begin{aligned} Init \triangleq\ & \land\ InQueue!Init \\ & \land\ OutQueue!Init \end{aligned}$$

$$\begin{aligned} BNext \triangleq\ & \land\ InQueue!BNext \\ & \land\ OutQueue!BNext \end{aligned}$$

$$\begin{aligned} Spec \triangleq\ & \land\ MsgTrans \\ & \land\ MsgIncl \end{aligned}$$

$\land\ Init$
$\land\ \Box[BNext]_{\langle inQueueIn,\ inQueueOut,\ inQueue,\ outQueueIn,\ outQueueOut,\ outQueue\rangle}$

THEOREM $Spec \Rightarrow \Box\, TypeOK$