MODULE *hazptr*

EXTENDS *Naturals*, *TLC*, *FiniteSets*, *Sequences*
CONSTANTS *NumReaders*, *NumWrites*

$R \triangleq 2$

```
  --algorithm hazptr
variables cur = 0, hzdreclist = {}, timeline = ⟨⟩

process writer = 1
variables old = 0, rlist = {}, write = 0
begin
WriterLoop:
    while write ≠ NumWrites do
    Update:
        old := cur ;
        cur := cur + 1 ;
        timeline := Append(timeline, write) ;
    Retire:
        rlist := rlist ∪ {old} ;
    Scan:
        if Cardinality(rlist) ≥ R then
            free all ones without hazard pointers
          rlist := {x ∈ rlist : x ∈ hzdreclist} ;
          print "cleaned rlist" ;
        end if ;
        write := write + 1
end while ;
end process ;

process reader ∈ 2 .. NumReaders + 1
variables saved_read = 0, saved_read_ptr = 0
begin
Acquire:
    await 1 ∈ DOMAIN timeline ;
      wait until something can be read. This mimics a real function that would
      check for the data to not be null
    hzdreclist := hzdreclist ∪ {cur} ;
Read:
    saved_read_ptr := cur ;
    saved_read := timeline[cur] ;
Check:
      mimic a pointer dereference
    assert saved_read = timeline[saved_read_ptr] ;
Release:
    hzdreclist := hzdreclist \ {cur} ;
```

end process ;

end algorithm ;

BEGIN TRANSLATION ($chksum(pcal) =$ "$e300f735$" $\land\ chksum(tla) =$ "$456bb21d$")

VARIABLES cur, $hzdreclist$, $timeline$, pc, old, $rlist$, $write$, $saved\_read$, $saved\_read\_ptr$

$vars \triangleq \langle cur, hzdreclist, timeline, pc, old, rlist, write, saved\_read, saved\_read\_ptr \rangle$

$ProcSet \triangleq \{1\} \cup (2 \,..\, NumReaders + 1)$

$Init \triangleq$ Global variables
$\land\ cur = 0$
$\land\ hzdreclist = \{\}$
$\land\ timeline = \langle\rangle$
Process writer
$\land\ old = 0$
$\land\ rlist\ = \{\}$
$\land\ write = 0$
Process reader
$\land\ saved\_read = [self \in 2 \,..\, NumReaders + 1 \mapsto 0]$
$\land\ saved\_read\_ptr = [self \in 2 \,..\, NumReaders + 1 \mapsto 0]$
$\land\ pc = [self \in ProcSet \mapsto$ CASE $self = 1 \to$ "WriterLoop"
$\square\quad self \in 2 \,..\, NumReaders + 1 \to$ "Acquire"$]$

$WriterLoop \triangleq\ \land\ pc[1] =$ "WriterLoop"
$\land$ IF $write \neq NumWrites$
THEN $\land\ pc' = [pc$ EXCEPT $![1] =$ "Update"$]$
ELSE $\land\ pc' = [pc$ EXCEPT $![1] =$ "Done"$]$
$\land$ UNCHANGED $\langle cur, hzdreclist, timeline, old, rlist, write, saved\_read, saved\_read\_ptr \rangle$

$Update \triangleq\ \land\ pc[1] =$ "Update"
$\land\ old' = cur$
$\land\ cur' = cur + 1$
$\land\ timeline' = Append(timeline, write)$
$\land\ pc' = [pc$ EXCEPT $![1] =$ "Retire"$]$
$\land$ UNCHANGED $\langle hzdreclist, rlist, write, saved\_read, saved\_read\_ptr \rangle$

$Retire \triangleq\ \land\ pc[1] =$ "Retire"
$\land\ rlist' = (rlist \cup \{old\})$
$\land\ pc' = [pc$ EXCEPT $![1] =$ "Scan"$]$
$\land$ UNCHANGED $\langle cur, hzdreclist, timeline, old, write, saved\_read, saved\_read\_ptr \rangle$

$Scan \triangleq\ \land\ pc[1] =$ "Scan"

$\land$ IF $Cardinality(rlist) \geq R$
THEN $\land\ rlist' = \{x \in rlist : x \in hzdreclist\}$
$\land\ PrintT(\text{“cleaned rlist”})$
ELSE $\land$ TRUE
$\land\ rlist' = rlist$
$\land\ write' = write + 1$
$\land\ pc' = [pc \text{ EXCEPT } ![1] = \text{“WriterLoop”}]$
$\land$ UNCHANGED $\langle cur, hzdreclist, timeline, old, saved\_read, saved\_read\_ptr\rangle$

$writer \triangleq WriterLoop \lor Update \lor Retire \lor Scan$

$Acquire(self) \triangleq \land\ pc[self] = \text{“Acquire”}$
$\land\ 1 \in \text{DOMAIN } timeline$
$\land\ hzdreclist' = (hzdreclist \cup \{cur\})$
$\land\ pc' = [pc \text{ EXCEPT } ![self] = \text{“Read”}]$
$\land$ UNCHANGED $\langle cur, timeline, old, rlist, write, saved\_read, saved\_read\_ptr\rangle$

$Read(self) \triangleq \land\ pc[self] = \text{“Read”}$
$\land\ saved\_read\_ptr' = [saved\_read\_ptr \text{ EXCEPT } ![self] = cur]$
$\land\ saved\_read' = [saved\_read \text{ EXCEPT } ![self] = timeline[cur]]$
$\land\ pc' = [pc \text{ EXCEPT } ![self] = \text{“Check”}]$
$\land$ UNCHANGED $\langle cur, hzdreclist, timeline, old, rlist, write\rangle$

$Check(self) \triangleq \land\ pc[self] = \text{“Check”}$
$\land\ Assert(saved\_read[self] = timeline[saved\_read\_ptr[self]],$
$\text{“Failure of assertion at line 44, column 5.”})$
$\land\ pc' = [pc \text{ EXCEPT } ![self] = \text{“Release”}]$
$\land$ UNCHANGED $\langle cur, hzdreclist, timeline, old, rlist, write, saved\_read, saved\_read\_ptr\rangle$

$Release(self) \triangleq \land\ pc[self] = \text{“Release”}$
$\land\ hzdreclist' = hzdreclist \setminus \{cur\}$
$\land\ pc' = [pc \text{ EXCEPT } ![self] = \text{“Done”}]$
$\land$ UNCHANGED $\langle cur, timeline, old, rlist, write, saved\_read, saved\_read\_ptr\rangle$

$reader(self) \triangleq Acquire(self) \lor Read(self) \lor Check(self) \lor Release(self)$

Allow infinite stuttering to prevent deadlock on termination.
$Terminating \triangleq \land \forall\, self \in ProcSet : pc[self] = \text{“Done”}$
$\land$ UNCHANGED $vars$

$Next \triangleq writer$
$\lor (\exists\, self \in 2\,..\,NumReaders + 1 : reader(self))$
$\lor Terminating$

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

$Termination \triangleq \Diamond(\forall\, self \in ProcSet : pc[self] = \text{“Done”})$

END TRANSLATION