MODULE *AtomicPtrV2*

EXTENDS TLC, $Naturals$, $Sequences$

CONSTANTS $Node$, nil

VARIABLES $pointer$, $counter$, $objects$, pc, $local\_addr$, $last\_counter$

$vars \triangleq \langle pointer, counter, objects, pc, local\_addr, last\_counter \rangle$

$Object \triangleq [ref : Nat, extra : Nat, added : \text{BOOLEAN}, destroyed : Nat]$

$NullAddr \triangleq (\text{DOMAIN } objects) \cup \{nil\}$

$State \triangleq \{$
 “Init”, “SwapPointer”, “IncreaseRefAgain”,
 “IncreaseRef”, “DecreaseLocalCounter”, “ClearExtraRef”,
 “UseObject”,
 “DecreaseRef”, “DestroyObject”, “Terminated”$\}$

$TypeOK \triangleq$
- $\land$ $objects \in Seq(Object)$
- $\land$ $pointer \in \text{DOMAIN } objects$
- $\land$ $counter \in Nat$
- $\land$ $pc \in [Node \rightarrow State]$
- $\land$ $local\_addr \in [Node \rightarrow NullAddr]$
- $\land$ $last\_counter \in [Node \rightarrow Nat]$

$Init \triangleq$
- $\land\ objects = \langle [ref \mapsto 1, extra \mapsto 0, added \mapsto \text{FALSE}, destroyed \mapsto 0] \rangle$
- $\land\ pointer = 1$
- $\land\ counter = 0$
- $\land\ pc = [n \in Node \mapsto \text{“Init”}]$
- $\land\ local\_addr = [n \in Node \mapsto nil]$
- $\land\ last\_counter = [n \in Node \mapsto 0]$

$goto(n, l) \triangleq$
- $\land\ pc' = [pc \text{ EXCEPT } ![n] = l]$

$newObject \triangleq [ref \mapsto 1, extra \mapsto 0, added \mapsto \text{FALSE}, destroyed \mapsto 0]$

$allocNew(n) \triangleq$
- $\land\ objects' = Append(objects, newObject)$
- $\land\ local\_addr' = [local\_addr \text{ EXCEPT } ![n] = Len(objects')]$

```
$reuseObject(n) \triangleq$
    $\exists\, addr \in \text{DOMAIN } objects :$
        $\land\ objects[addr].destroyed = 1$
        $\land\ objects' = [objects \text{ EXCEPT } ![addr] = newObject]$
        $\land\ local\_addr' = [local\_addr \text{ EXCEPT } ![n] = addr]$

$AllocateNewObject(n) \triangleq$
    $\land\ pc[n] = \text{“Init”}$
    $\land\ goto(n, \text{“SwapPointer”})$
    $\land\ \lor\ allocNew(n)$
        $\lor\ reuseObject(n)$
    $\land\ \text{UNCHANGED } \langle counter, pointer \rangle$
    $\land\ \text{UNCHANGED } last\_counter$

$SwapPointer(n) \triangleq$
    $\land\ pc[n] = \text{“SwapPointer”}$
    $\land\ pointer' = local\_addr[n]$
    $\land\ local\_addr' = [local\_addr \text{ EXCEPT } ![n] = pointer]$
    $\land\ \text{IF } counter = 0$
        $\text{THEN}$
            $\land\ goto(n, \text{“DecreaseRef”})$
            $\land\ \text{UNCHANGED } counter$
        $\text{ELSE}$
            $\land\ goto(n, \text{“IncreaseRefAgain”})$
            $\land\ counter' = 0$
    $\land\ last\_counter' = [last\_counter \text{ EXCEPT } ![n] = counter]$
    $\land\ \text{UNCHANGED } objects$

$IncreaseRefAgain(n) \triangleq$
    $\text{LET}$
        $addr \triangleq local\_addr[n]$
        $diff \ \ \triangleq last\_counter[n] - objects[addr].extra$
    $\text{IN}$
        $\land\ pc[n] = \text{“IncreaseRefAgain”}$
        $\land\ goto(n, \text{“DecreaseRef”})$
        $\land\ objects' = [$
            $objects \text{ EXCEPT } ![addr].ref = @ + diff,\ ![addr].added = \text{TRUE}]$
        $\land\ \text{UNCHANGED } counter$
        $\land\ \text{UNCHANGED } pointer$
        $\land\ \text{UNCHANGED } local\_addr$
        $\land\ \text{UNCHANGED } last\_counter$

$LoadPointer(n) \triangleq$
```

```
    $\land\ pc[n] = \text{"Init"}$
    $\land\ counter' = counter + 1$
    $\land\ local\_addr' = [local\_addr\ \text{EXCEPT}\ ![n] = pointer]$
    $\land\ goto(n, \text{"IncreaseRef"})$
    $\land$ UNCHANGED $objects$
    $\land$ UNCHANGED $pointer$
    $\land$ UNCHANGED $last\_counter$

$IncreaseRef(n) \triangleq$
    LET
        $addr \triangleq local\_addr[n]$
    IN
        $\land\ pc[n] = \text{"IncreaseRef"}$
        $\land\ objects' = [objects\ \text{EXCEPT}\ ![addr].ref = @ + 1]$
        $\land\ goto(n, \text{"DecreaseLocalCounter"})$
        $\land$ UNCHANGED $local\_addr$
        $\land$ UNCHANGED $counter$
        $\land$ UNCHANGED $pointer$
        $\land$ UNCHANGED $last\_counter$

$DecreaseLocalCounter(n) \triangleq$
    $\land\ pc[n] = \text{"DecreaseLocalCounter"}$
    $\land$ IF $pointer = local\_addr[n]$
        THEN
            $\land\ counter' = counter - 1$
            $\land\ goto(n, \text{"UseObject"})$
        ELSE
            $\land$ UNCHANGED $counter$
            $\land\ goto(n, \text{"ClearExtraRef"})$
    $\land$ UNCHANGED $local\_addr$
    $\land$ UNCHANGED $objects$
    $\land$ UNCHANGED $pointer$
    $\land$ UNCHANGED $last\_counter$

$ClearExtraRef(n) \triangleq$
    LET
        $addr \triangleq local\_addr[n]$
    IN
        $\land\ pc[n] = \text{"ClearExtraRef"}$
        $\land$ IF $objects[addr].added$
            THEN $objects' = [$
                $objects\ \text{EXCEPT}\ ![addr].ref = @ - 1]$
            ELSE $objects' = [$
                $objects\ \text{EXCEPT}\ ![addr].extra = @ + 1]$
        $\land$ $goto(n,$ "UseObject"$)$
        $\land$ UNCHANGED $local\_addr$
        $\land$ UNCHANGED $counter$
        $\land$ UNCHANGED $pointer$
        $\land$ UNCHANGED $last\_counter$

$UseObject(n) \triangleq$
    $\land$ $pc[n] =$ "UseObject"
    $\land$ $goto(n,$ "DecreaseRef"$)$
    $\land$ UNCHANGED $objects$
    $\land$ UNCHANGED $counter$
    $\land$ UNCHANGED $pointer$
    $\land$ UNCHANGED $local\_addr$
    $\land$ UNCHANGED $last\_counter$

$DecreaseRef(n) \triangleq$
    LET
        $addr \triangleq local\_addr[n]$
    IN
        $\land$ $pc[n] =$ "DecreaseRef"
        $\land$ $objects' = [objects$ EXCEPT $![addr].ref = @ - 1]$
        $\land$ IF $objects'[addr].ref = 0$
            THEN $goto(n,$ "DestroyObject"$)$
            ELSE $goto(n,$ "Terminated"$)$
        $\land$ UNCHANGED $local\_addr$
        $\land$ UNCHANGED $counter$
        $\land$ UNCHANGED $pointer$
        $\land$ UNCHANGED $last\_counter$

$DestroyObject(n) \triangleq$
    LET
        $addr \triangleq local\_addr[n]$
    IN
        $\land$ $pc[n] =$ "DestroyObject"
        $\land$ $goto(n,$ "Terminated"$)$
        $\land$ $objects' = [objects$ EXCEPT $![addr].destroyed = @ + 1]$
        $\land$ UNCHANGED $local\_addr$
        $\land$ UNCHANGED $counter$
        $\land$ UNCHANGED $pointer$
        $\land$ UNCHANGED $last\_counter$

$TerminateCond \triangleq$
    $\land$ $\forall n \in Node : pc[n] =$ "Terminated"
```

```
$Terminated \triangleq$
    $\land\ TerminateCond$
    $\land$ UNCHANGED $vars$

$Next \triangleq$
    $\lor\ \exists\, n \in Node :$
        $\lor\ AllocateNewObject(n)$
        $\lor\ SwapPointer(n)$
        $\lor\ IncreaseRefAgain(n)$
        $\lor\ LoadPointer(n)$
        $\lor\ IncreaseRef(n)$
        $\lor\ DecreaseLocalCounter(n)$
        $\lor\ ClearExtraRef(n)$
        $\lor\ UseObject(n)$
        $\lor\ DecreaseRef(n)$
        $\lor\ DestroyObject(n)$
    $\lor\ Terminated$

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

$FairSpec \triangleq Spec \land \mathrm{WF}_{vars}(Next)$

$FullyDestroyed \triangleq$
   LET
        $destroyedExceptLast(addr) \triangleq$
            $addr \neq pointer \Rightarrow objects[addr].destroyed = 1 \land objects[addr].ref = 0$

        $allDestroyed \triangleq$
            $\forall\, addr \in$ DOMAIN $objects : destroyedExceptLast(addr)$
   IN
        $TerminateCond \Rightarrow allDestroyed$

$UseObjectAlwaysValid \triangleq$
   LET
        $getObj(n) \triangleq objects[local\_addr[n]]$
        $notUseAfterFree(n) \triangleq$
            $\land\ getObj(n).destroyed = 0$
            $\land\ getObj(n).ref > 0$
   IN
        $\forall\, n \in Node : pc[n] = \text{“UseObject”} \Rightarrow notUseAfterFree(n)$

$IncreaseRefMustNotDestroyed \triangleq$
   LET
        $accessStates(n) \triangleq pc[n] = \text{“IncreaseRef”}$
        $getObj(n) \triangleq objects[local\_addr[n]]$
    IN
        $\forall n \in Node : accessStates(n) \Rightarrow getObj(n).destroyed = 0$

$AccessStateMustNotDestroyed \triangleq$
    LET
        $accessStates(n) \triangleq$
            $\lor pc[n] =$ "IncreaseRef"
            $\lor pc[n] =$ "IncreaseRefAgain"
            $\lor pc[n] =$ "DecreaseLocalCounter"
            $\lor pc[n] =$ "ClearExtraRef"
            $\lor pc[n] =$ "UseObject"
            $\lor pc[n] =$ "DecreaseRef"
            $\lor pc[n] =$ "DestroyObject"

        $getObj(n) \triangleq objects[local\_addr[n]]$
    IN
        $\forall n \in Node : accessStates(n) \Rightarrow getObj(n).destroyed = 0$

$AlwaysTerminate \triangleq \Diamond TerminateCond$

$IncreaseRefLeadToUseObject \triangleq$
    $\forall n \in Node :$
        $pc[n] =$ "IncreaseRef" $\leadsto pc[n] =$ "UseObject"

$Sym \triangleq Permutations(Node)$
```