MODULE *lock_free_stack_ABA*

For *ABA* problem, see: https://*en.wikipedia.org*/wiki/*ABA_problem*

NOTE: The label "pop2a" in *popStack*() below "causes" the bug to be exposed. Try deleting it. This would have the effect of atomically executing:
CAS(*HEAD*, *local_myhead*, *address*[*local_myhead*].*next*);
because it hides the internal assembly code that first computes the register value, *reg_next*, and then executes *CAS*().

Note that the field "*onStack*" is present only for debugging, and for assertions. Don't hesitate to add extra fields and variables to ease the job of model checking.

The bug is the *ABA* problem. When illustrating this bug, the errors in the "Model Checking Results" tab will have too many frames to easily read. You can then remove many of the unnecessary labels in other routines to easily see the cause of the bug: the *ABA* problem.

Always include these *PlusCal* modules for basic data types.

```
EXTENDS Naturals, Sequences, TLC, FiniteSets
```

These constants will have to be assigned a value within the generated module.

```
CONSTANTS MAX_ITER, MAX_STACK_SIZE
```

NUM_THREADS is a PRE-DEFINED constant. It cannot be changed later.

```
NUM_THREADS ≜ 2
```

NULL is a unique value (pre-defined constant)

```
NULL ≜ CHOOSE n : n ∉ 1 . . MAX_STACK_SIZE
```

```
--algorithm LockFreeStack{
variables retVal = [thread ∈ 1 . . NUM_THREADS ↦ NULL],
          address = [addr ∈ 1 . . MAX_STACK_SIZE ↦
                        [next ↦ NULL, onStack ↦ FALSE, data ↦ 0]],
          initialized = FALSE ;
          HEAD ;

  CAS: compare-and-swap
  CAS must be a macro. This reflects that it is an assembly instruction
    that modifies its arguments. It must not use the call-by-value of a procedure.
  CAS is atomic. So, there are no intermediate labels.
macro CAS( x, y, z )
{
  if ( x = y ) {
    x := z ;   swap y for z as value of x
    retVal[self] := TRUE ;
   } else {
     retVal[self] := FALSE ;
  } ;
 }

procedure pushStack( elt )
  variable local_myhead ;
{
  push1: address[elt].next := HEAD ;
  push2: assert address[elt].onStack = FALSE ;
  push3: address[elt].onStack := TRUE ;
  tryAgainPush: local_myhead := HEAD ;
        push5a:  address[elt].next := local_myhead ;
        push5b:  CAS(HEAD, local_myhead, elt) ;
        push5c:  if ( ¬retVal[self] ) {
        push5d:    goto tryAgainPush ;
                   } ;
  endPush: return ;
 }

procedure popStack( )
  variable local_myhead, reg_next, elt ;
{
  tryAgainPop:
     local_myhead := HEAD ;
  pop1: if ( local_myhead = 0 ) {   If I believe HEAD = 0, return now.
  pop1a:  retVal[self] := NULL ;
  pop1b:  return ;
          } ;

  pop2:  reg_next :=  address[local_myhead].next ;
  pop2a: CAS(HEAD, local_myhead, reg_next) ;
  pop2b: if ( ¬retVal[self] ) {
  pop2c:   goto tryAgainPop ;
           } ;
  pop3: elt := local_myhead ;
  pop4: assert address[elt].onStack = TRUE ;
  pop5: address[elt].onStack := FALSE ;
  pop6: retVal[self] := elt ;
  endPop: return ;
 }

process ( thread ∈ 1 .. NUM_THREADS )
  variable my_set = {}, myelt,
           init_thread, iterations = MAX_ITER ;
{
  init1: init_thread := CHOOSE thr ∈ 1 .. NUM_THREADS : TRUE ;
  init2: if ( self = init_thread ) {   // This thread will initialize global data
  init4:   HEAD := 0 ;
  init5: while ( HEAD < MAX_STACK_SIZE ) {
  init6:      address[HEAD    + 1] := [next ↦ HEAD, onStack ↦ TRUE, data ↦ 1] ;
  init7:      HEAD := HEAD + 1 ;
            } ;
  init8:   initialized := TRUE ;   //init_thread will set this global var
         } ;
  init9: await initialized ;   // all threads will wait for this

  th1: while ( iterations > 0 ) {
  th2:   either { if ( my_set ≠ {} ) {
                           // Set myelt at random (non-deterministically)
  th2a:                with ( tmp ∈ my_set ) { myelt := tmp } ;
  th2b:                my_set := my_set \ {myelt} ;
  th2c:                call pushStack(myelt) ;
                    } }
         or {
  th3a:       call popStack() ;
  th3b:       if ( retVal[self] ≠ NULL ) {
  th3c:         my_set := my_set ∪ {retVal[self]}  // Add the popped elt to my_set
               } } ;
  th4: iterations := iterations − 1 ;
  print iterations ;
   } ;   end while
} ;   end process

}   \* end algorithm
```

BEGIN TRANSLATION

Procedure variable *local_myhead* of procedure *pushStack* at line 62 col 12 changed to *local_myhead_*

Procedure variable *elt* of procedure *popStack* at line 77 col 36 changed to *elt_*

CONSTANT $defaultInitValue$

VARIABLES $retVal$, $address$, $initialized$, $HEAD$, pc, $stack$, elt, $local\_myhead\_$, $local\_myhead$, reg_next, $elt\_$, my_set, $myelt$, $init\_thread$, $iterations$

$vars \triangleq \langle retVal, address, initialized, HEAD, pc, stack, elt, local\_myhead\_, local\_myhead, reg\_next, elt\_, my\_set, myelt, init\_thread, iterations\rangle$

$ProcSet \triangleq (1 \,..\, NUM\_THREADS)$

$Init \triangleq$ Global variables

$\land\ retVal = [thread \in 1 \,..\, NUM\_THREADS \mapsto NULL]$

$\land\ address = [addr \in 1 \,..\, MAX\_STACK\_SIZE \mapsto [next \mapsto NULL, onStack \mapsto \text{FALSE}, data \mapsto 0]]$

$\land\ initialized = \text{FALSE}$

$\land\ HEAD = defaultInitValue$

Procedure *pushStack*

$\land\ elt = [self \in ProcSet \mapsto defaultInitValue]$

$\land\ local\_myhead\_ = [self \in ProcSet \mapsto defaultInitValue]$

Procedure *popStack*

$\land\ local\_myhead = [self \in ProcSet \mapsto defaultInitValue]$

$\land\ reg\_next = [self \in ProcSet \mapsto defaultInitValue]$

$\land\ elt\_ = [self \in ProcSet \mapsto defaultInitValue]$

Process thread

$\land\ my\_set = [self \in 1 \mathrel{..} NUM\_THREADS \mapsto \{\}]$

$\land\ myelt = [self \in 1 \mathrel{..} NUM\_THREADS \mapsto defaultInitValue]$

$\land\ init\_thread = [self \in 1 \mathrel{..} NUM\_THREADS \mapsto defaultInitValue]$

$\land\ iterations = [self \in 1 \mathrel{..} NUM\_THREADS \mapsto MAX\_ITER]$

$\land\ stack = [self \in ProcSet \mapsto \langle\rangle]$

$\land\ pc = [self \in ProcSet \mapsto \text{“init1”}]$

$$
\begin{aligned}
push1(self) \triangleq\ & \land\ pc[self] = \text{“push1”} \\
& \land\ address' = [address \text{ EXCEPT } ![elt[self]].next = HEAD] \\
& \land\ pc' = [pc \text{ EXCEPT } ![self] = \text{“push2”}] \\
& \land\ \text{UNCHANGED } \langle retVal, initialized, HEAD, stack, elt, \\
& \qquad local\_myhead\_, local\_myhead, reg\_next, elt\_, \\
& \qquad my\_set, myelt, init\_thread, iterations\rangle
\end{aligned}
$$

$$
\begin{aligned}
push2(self) \triangleq\ & \land\ pc[self] = \text{“push2”} \\
& \land\ Assert(address[elt[self]].onStack = \text{FALSE}, \\
& \qquad \text{“Failure of assertion at line 65, column 10.”}) \\
& \land\ pc' = [pc \text{ EXCEPT } ![self] = \text{“push3”}] \\
& \land\ \text{UNCHANGED } \langle retVal, address, initialized, HEAD, stack, elt, \\
& \qquad local\_myhead\_, local\_myhead, reg\_next, elt\_, \\
& \qquad my\_set, myelt, init\_thread, iterations\rangle
\end{aligned}
$$

$$
\begin{aligned}
push3(self) \triangleq\ & \land\ pc[self] = \text{“push3”} \\
& \land\ address' = [address \text{ EXCEPT } ![elt[self]].onStack = \text{TRUE}] \\
& \land\ pc' = [pc \text{ EXCEPT } ![self] = \text{“tryAgainPush”}] \\
& \land\ \text{UNCHANGED } \langle retVal, initialized, HEAD, stack, elt, \\
& \qquad local\_myhead\_, local\_myhead, reg\_next, elt\_, \\
& \qquad my\_set, myelt, init\_thread, iterations\rangle
\end{aligned}
$$

$$
\begin{aligned}
tryAgainPush(self) \triangleq\ & \land\ pc[self] = \text{“tryAgainPush”} \\
& \land\ local\_myhead\_' = [local\_myhead\_ \text{ EXCEPT } ![self] = HEAD] \\
& \land\ pc' = [pc \text{ EXCEPT } ![self] = \text{“push5a”}] \\
& \land\ \text{UNCHANGED } \langle retVal, address, initialized, HEAD, \\
& \qquad stack, elt, local\_myhead, reg\_next, elt\_, \\
& \qquad my\_set, myelt, init\_thread, iterations\rangle
\end{aligned}
$$

$$
push5a(self) \triangleq\ \land\ pc[self] = \text{“push5a”}
$$

```
                  $\land\ address' = [address \text{ EXCEPT } ![elt[self]].next = local\_myhead\_[self]]$
                  $\land\ pc' = [pc \text{ EXCEPT } ![self] = \text{“push5b”}]$
                  $\land\ \text{UNCHANGED } \langle retVal, initialized, HEAD, stack, elt,$
                                 $local\_myhead\_, local\_myhead, reg\_next, elt\_,$
                                 $my\_set, myelt, init\_thread, iterations \rangle$

$push5b(self) \triangleq$ $\land\ pc[self] = \text{“push5b”}$
                  $\land\ \text{IF } HEAD = local\_myhead\_[self]$
                        $\text{THEN } \land\ HEAD' = elt[self]$
                              $\land\ retVal' = [retVal \text{ EXCEPT } ![self] = \text{TRUE}]$
                        $\text{ELSE } \land\ retVal' = [retVal \text{ EXCEPT } ![self] = \text{FALSE}]$
                              $\land\ HEAD' = HEAD$
                  $\land\ pc' = [pc \text{ EXCEPT } ![self] = \text{“push5c”}]$
                  $\land\ \text{UNCHANGED } \langle address, initialized, stack, elt,$
                                 $local\_myhead\_, local\_myhead, reg\_next, elt\_,$
                                 $my\_set, myelt, init\_thread, iterations \rangle$

$push5c(self) \triangleq$ $\land\ pc[self] = \text{“push5c”}$
                  $\land\ \text{IF } \neg retVal[self]$
                        $\text{THEN } \land\ pc' = [pc \text{ EXCEPT } ![self] = \text{“push5d”}]$
                        $\text{ELSE } \land\ pc' = [pc \text{ EXCEPT } ![self] = \text{“endPush”}]$
                  $\land\ \text{UNCHANGED } \langle retVal, address, initialized, HEAD, stack, elt,$
                                 $local\_myhead\_, local\_myhead, reg\_next, elt\_,$
                                 $my\_set, myelt, init\_thread, iterations \rangle$

$push5d(self) \triangleq$ $\land\ pc[self] = \text{“push5d”}$
                  $\land\ pc' = [pc \text{ EXCEPT } ![self] = \text{“tryAgainPush”}]$
                  $\land\ \text{UNCHANGED } \langle retVal, address, initialized, HEAD, stack, elt,$
                                 $local\_myhead\_, local\_myhead, reg\_next, elt\_,$
                                 $my\_set, myelt, init\_thread, iterations \rangle$

$endPush(self) \triangleq$ $\land\ pc[self] = \text{“endPush”}$
                   $\land\ pc' = [pc \text{ EXCEPT } ![self] = Head(stack[self]).pc]$
                   $\land\ local\_myhead\_' = [local\_myhead\_ \text{ EXCEPT } ![self] = Head(stack[self]).local\_myhead\_]$
                   $\land\ elt' = [elt \text{ EXCEPT } ![self] = Head(stack[self]).elt]$
                   $\land\ stack' = [stack \text{ EXCEPT } ![self] = Tail(stack[self])]$
                   $\land\ \text{UNCHANGED } \langle retVal, address, initialized, HEAD,$
                                  $local\_myhead, reg\_next, elt\_, my\_set, myelt,$
                                  $init\_thread, iterations \rangle$

$pushStack(self) \triangleq$ $push1(self) \lor push2(self) \lor push3(self)$
                     $\lor\ tryAgainPush(self) \lor push5a(self) \lor push5b(self)$
                     $\lor\ push5c(self) \lor push5d(self) \lor endPush(self)$

$tryAgainPop(self) \triangleq$ $\land\ pc[self] = \text{“tryAgainPop”}$
                       $\land\ local\_myhead' = [local\_myhead \text{ EXCEPT } ![self] = HEAD]$
                       $\land\ pc' = [pc \text{ EXCEPT } ![self] = \text{“pop1”}]$
```

```
            ∧ UNCHANGED ⟨retVal, address, initialized, HEAD, stack,
                          elt, local_myhead_, reg_next, elt_,
                          my_set, myelt, init_thread, iterations⟩

pop1(self) ≜ ∧ pc[self] = "pop1"
             ∧ IF local_myhead[self] = 0
                  THEN ∧ pc' = [pc EXCEPT ![self] = "pop1a"]
                  ELSE ∧ pc' = [pc EXCEPT ![self] = "pop2"]
             ∧ UNCHANGED ⟨retVal, address, initialized, HEAD, stack, elt,
                          local_myhead_, local_myhead, reg_next, elt_,
                          my_set, myelt, init_thread, iterations⟩

pop1a(self) ≜ ∧ pc[self] = "pop1a"
              ∧ retVal' = [retVal EXCEPT ![self] = NULL]
              ∧ pc' = [pc EXCEPT ![self] = "pop1b"]
              ∧ UNCHANGED ⟨address, initialized, HEAD, stack, elt,
                           local_myhead_, local_myhead, reg_next, elt_,
                           my_set, myelt, init_thread, iterations⟩

pop1b(self) ≜ ∧ pc[self] = "pop1b"
              ∧ pc' = [pc EXCEPT ![self] = Head(stack[self]).pc]
              ∧ local_myhead' = [local_myhead EXCEPT ![self] = Head(stack[self]).local_myhead]
              ∧ reg_next' = [reg_next EXCEPT ![self] = Head(stack[self]).reg_next]
              ∧ elt_' = [elt_ EXCEPT ![self] = Head(stack[self]).elt_]
              ∧ stack' = [stack EXCEPT ![self] = Tail(stack[self])]
              ∧ UNCHANGED ⟨retVal, address, initialized, HEAD, elt,
                           local_myhead_, my_set, myelt, init_thread,
                           iterations⟩

pop2(self) ≜ ∧ pc[self] = "pop2"
             ∧ reg_next' = [reg_next EXCEPT ![self] = address[local_myhead[self]].next]
             ∧ pc' = [pc EXCEPT ![self] = "pop2a"]
             ∧ UNCHANGED ⟨retVal, address, initialized, HEAD, stack, elt,
                          local_myhead_, local_myhead, elt_, my_set, myelt,
                          init_thread, iterations⟩

pop2a(self) ≜ ∧ pc[self] = "pop2a"
              ∧ IF HEAD = local_myhead[self]
                   THEN ∧ HEAD' = reg_next[self]
                        ∧ retVal' = [retVal EXCEPT ![self] = TRUE]
                   ELSE ∧ retVal' = [retVal EXCEPT ![self] = FALSE]
                        ∧ HEAD' = HEAD
              ∧ pc' = [pc EXCEPT ![self] = "pop2b"]
              ∧ UNCHANGED ⟨address, initialized, stack, elt, local_myhead_,
                           local_myhead, reg_next, elt_, my_set, myelt,
                           init_thread, iterations⟩
```

$$
\begin{array}{lll}
\mathit{pop2b}(\mathit{self}) \triangleq & \land\ \mathit{pc}[\mathit{self}] = \text{“pop2b”} \\
& \land\ \text{IF } \neg \mathit{retVal}[\mathit{self}] \\
& \qquad \text{THEN } \land\ \mathit{pc}' = [\mathit{pc} \text{ EXCEPT } ![\mathit{self}] = \text{“pop2c”}] \\
& \qquad \text{ELSE } \land\ \mathit{pc}' = [\mathit{pc} \text{ EXCEPT } ![\mathit{self}] = \text{“pop3”}] \\
& \land\ \text{UNCHANGED } \langle \mathit{retVal}, \mathit{address}, \mathit{initialized}, \mathit{HEAD}, \mathit{stack}, \mathit{elt}, \\
& \qquad \mathit{local\_myhead\_}, \mathit{local\_myhead}, \mathit{reg\_next}, \mathit{elt\_}, \\
& \qquad \mathit{my\_set}, \mathit{myelt}, \mathit{init\_thread}, \mathit{iterations} \rangle
\end{array}
$$

$$
\begin{array}{lll}
\mathit{pop2c}(\mathit{self}) \triangleq & \land\ \mathit{pc}[\mathit{self}] = \text{“pop2c”} \\
& \land\ \mathit{pc}' = [\mathit{pc} \text{ EXCEPT } ![\mathit{self}] = \text{“tryAgainPop”}] \\
& \land\ \text{UNCHANGED } \langle \mathit{retVal}, \mathit{address}, \mathit{initialized}, \mathit{HEAD}, \mathit{stack}, \mathit{elt}, \\
& \qquad \mathit{local\_myhead\_}, \mathit{local\_myhead}, \mathit{reg\_next}, \mathit{elt\_}, \\
& \qquad \mathit{my\_set}, \mathit{myelt}, \mathit{init\_thread}, \mathit{iterations} \rangle
\end{array}
$$

$$
\begin{array}{lll}
\mathit{pop3}(\mathit{self}) \triangleq & \land\ \mathit{pc}[\mathit{self}] = \text{“pop3”} \\
& \land\ \mathit{elt\_}' = [\mathit{elt\_} \text{ EXCEPT } ![\mathit{self}] = \mathit{local\_myhead}[\mathit{self}]] \\
& \land\ \mathit{pc}' = [\mathit{pc} \text{ EXCEPT } ![\mathit{self}] = \text{“pop4”}] \\
& \land\ \text{UNCHANGED } \langle \mathit{retVal}, \mathit{address}, \mathit{initialized}, \mathit{HEAD}, \mathit{stack}, \mathit{elt}, \\
& \qquad \mathit{local\_myhead\_}, \mathit{local\_myhead}, \mathit{reg\_next}, \mathit{my\_set}, \\
& \qquad \mathit{myelt}, \mathit{init\_thread}, \mathit{iterations} \rangle
\end{array}
$$

$$
\begin{array}{lll}
\mathit{pop4}(\mathit{self}) \triangleq & \land\ \mathit{pc}[\mathit{self}] = \text{“pop4”} \\
& \land\ \mathit{Assert}(\mathit{address}[\mathit{elt\_}[\mathit{self}]].\mathit{onStack} = \text{TRUE}, \\
& \qquad \text{“Failure of assertion at line 92, column 9.”}) \\
& \land\ \mathit{pc}' = [\mathit{pc} \text{ EXCEPT } ![\mathit{self}] = \text{“pop5”}] \\
& \land\ \text{UNCHANGED } \langle \mathit{retVal}, \mathit{address}, \mathit{initialized}, \mathit{HEAD}, \mathit{stack}, \mathit{elt}, \\
& \qquad \mathit{local\_myhead\_}, \mathit{local\_myhead}, \mathit{reg\_next}, \mathit{elt\_}, \\
& \qquad \mathit{my\_set}, \mathit{myelt}, \mathit{init\_thread}, \mathit{iterations} \rangle
\end{array}
$$

$$
\begin{array}{lll}
\mathit{pop5}(\mathit{self}) \triangleq & \land\ \mathit{pc}[\mathit{self}] = \text{“pop5”} \\
& \land\ \mathit{address}' = [\mathit{address} \text{ EXCEPT } ![\mathit{elt\_}[\mathit{self}]].\mathit{onStack} = \text{FALSE}] \\
& \land\ \mathit{pc}' = [\mathit{pc} \text{ EXCEPT } ![\mathit{self}] = \text{“pop6”}] \\
& \land\ \text{UNCHANGED } \langle \mathit{retVal}, \mathit{initialized}, \mathit{HEAD}, \mathit{stack}, \mathit{elt}, \\
& \qquad \mathit{local\_myhead\_}, \mathit{local\_myhead}, \mathit{reg\_next}, \mathit{elt\_}, \\
& \qquad \mathit{my\_set}, \mathit{myelt}, \mathit{init\_thread}, \mathit{iterations} \rangle
\end{array}
$$

$$
\begin{array}{lll}
\mathit{pop6}(\mathit{self}) \triangleq & \land\ \mathit{pc}[\mathit{self}] = \text{“pop6”} \\
& \land\ \mathit{retVal}' = [\mathit{retVal} \text{ EXCEPT } ![\mathit{self}] = \mathit{elt\_}[\mathit{self}]] \\
& \land\ \mathit{pc}' = [\mathit{pc} \text{ EXCEPT } ![\mathit{self}] = \text{“endPop”}] \\
& \land\ \text{UNCHANGED } \langle \mathit{address}, \mathit{initialized}, \mathit{HEAD}, \mathit{stack}, \mathit{elt}, \\
& \qquad \mathit{local\_myhead\_}, \mathit{local\_myhead}, \mathit{reg\_next}, \mathit{elt\_}, \\
& \qquad \mathit{my\_set}, \mathit{myelt}, \mathit{init\_thread}, \mathit{iterations} \rangle
\end{array}
$$

$$
\begin{array}{lll}
\mathit{endPop}(\mathit{self}) \triangleq & \land\ \mathit{pc}[\mathit{self}] = \text{“endPop”} \\
& \land\ \mathit{pc}' = [\mathit{pc} \text{ EXCEPT } ![\mathit{self}] = \mathit{Head}(\mathit{stack}[\mathit{self}]).\mathit{pc}] \\
& \land\ \mathit{local\_myhead}' = [\mathit{local\_myhead} \text{ EXCEPT } ![\mathit{self}] = \mathit{Head}(\mathit{stack}[\mathit{self}]).\mathit{local\_myhead}] \\
& \land\ \mathit{reg\_next}' = [\mathit{reg\_next} \text{ EXCEPT } ![\mathit{self}] = \mathit{Head}(\mathit{stack}[\mathit{self}]).\mathit{reg\_next}]
\end{array}
$$

$$\begin{array}{l}
\land\ elt\_' = [elt\_\ \text{EXCEPT}\ ![self] = Head(stack[self]).elt\_]\\
\land\ stack' = [stack\ \text{EXCEPT}\ ![self] = Tail(stack[self])]\\
\land\ \text{UNCHANGED}\ \langle retVal,\ address,\ initialized,\ HEAD,\ elt,\\
\qquad\qquad local\_myhead\_,\ my\_set,\ myelt,\ init\_thread,\\
\qquad\qquad iterations\rangle
\end{array}$$

$$\begin{array}{l}
popStack(self) \triangleq\ tryAgainPop(self) \lor pop1(self) \lor pop1a(self)\\
\qquad\qquad \lor\ pop1b(self)\ \ \lor pop2(self) \lor pop2a(self)\\
\qquad\qquad \lor\ pop2b(self)\ \ \lor pop2c(self) \lor pop3(self)\\
\qquad\qquad \lor\ pop4(self) \lor pop5(self) \lor pop6(self)\\
\qquad\qquad \lor\ endPop(self)
\end{array}$$

$$\begin{array}{ll}
init1(self) \triangleq & \land\ pc[self] = \text{“init1”}\\
& \land\ init\_thread' = [init\_thread\ \text{EXCEPT}\ ![self] = \text{CHOOSE}\ thr \in 1\,..\,NUM\_THREADS : \text{TRUE}]\\
& \land\ pc' = [pc\ \text{EXCEPT}\ ![self] = \text{“init2”}]\\
& \land\ \text{UNCHANGED}\ \langle retVal,\ address,\ initialized,\ HEAD,\ stack,\ elt,\\
& \qquad\qquad local\_myhead\_,\ local\_myhead,\ reg\_next,\ elt\_,\\
& \qquad\qquad my\_set,\ myelt,\ iterations\rangle
\end{array}$$

$$\begin{array}{ll}
init2(self) \triangleq & \land\ pc[self] = \text{“init2”}\\
& \land\ \text{IF}\ self = init\_thread[self]\\
& \qquad \text{THEN}\ \ \land\ pc' = [pc\ \text{EXCEPT}\ ![self] = \text{“init4”}]\\
& \qquad \text{ELSE}\ \ \ \land\ pc' = [pc\ \text{EXCEPT}\ ![self] = \text{“init9”}]\\
& \land\ \text{UNCHANGED}\ \langle retVal,\ address,\ initialized,\ HEAD,\ stack,\ elt,\\
& \qquad\qquad local\_myhead\_,\ local\_myhead,\ reg\_next,\ elt\_,\\
& \qquad\qquad my\_set,\ myelt,\ init\_thread,\ iterations\rangle
\end{array}$$

$$\begin{array}{ll}
init4(self) \triangleq & \land\ pc[self] = \text{“init4”}\\
& \land\ HEAD' = 0\\
& \land\ pc' = [pc\ \text{EXCEPT}\ ![self] = \text{“init5”}]\\
& \land\ \text{UNCHANGED}\ \langle retVal,\ address,\ initialized,\ stack,\ elt,\\
& \qquad\qquad local\_myhead\_,\ local\_myhead,\ reg\_next,\ elt\_,\\
& \qquad\qquad my\_set,\ myelt,\ init\_thread,\ iterations\rangle
\end{array}$$

$$\begin{array}{ll}
init5(self) \triangleq & \land\ pc[self] = \text{“init5”}\\
& \land\ \text{IF}\ HEAD < MAX\_STACK\_SIZE\\
& \qquad \text{THEN}\ \ \land\ pc' = [pc\ \text{EXCEPT}\ ![self] = \text{“init6”}]\\
& \qquad \text{ELSE}\ \ \ \land\ pc' = [pc\ \text{EXCEPT}\ ![self] = \text{“init8”}]\\
& \land\ \text{UNCHANGED}\ \langle retVal,\ address,\ initialized,\ HEAD,\ stack,\ elt,\\
& \qquad\qquad local\_myhead\_,\ local\_myhead,\ reg\_next,\ elt\_,\\
& \qquad\qquad my\_set,\ myelt,\ init\_thread,\ iterations\rangle
\end{array}$$

$$\begin{array}{ll}
init6(self) \triangleq & \land\ pc[self]\ = \text{“init6”}\\
& \land\ address' = [address\ \text{EXCEPT}\ ![HEAD+1] = [next \mapsto HEAD,\ onStack \mapsto \text{TRUE},\ data \mapsto 1]]\\
& \land\ pc' = [pc\ \text{EXCEPT}\ ![self] = \text{“init7”}]\\
& \land\ \text{UNCHANGED}\ \langle retVal,\ initialized,\ HEAD,\ stack,\ elt,\\
& \qquad\qquad local\_myhead\_,\ local\_myhead,\ reg\_next,\ elt\_,
\end{array}$$

```
                      my_set, myelt, init_thread, iterations⟩

init7(self) ≜ ∧ pc[self] = "init7"
              ∧ HEAD' = HEAD + 1
              ∧ pc' = [pc EXCEPT ![self] = "init5"]
              ∧ UNCHANGED ⟨retVal, address, initialized, stack, elt,
                           local_myhead_, local_myhead, reg_next, elt_,
                           my_set, myelt, init_thread, iterations⟩

init8(self) ≜ ∧ pc[self] = "init8"
              ∧ initialized' = TRUE
              ∧ pc' = [pc EXCEPT ![self] = "init9"]
              ∧ UNCHANGED ⟨retVal, address, HEAD, stack, elt,
                           local_myhead_, local_myhead, reg_next, elt_,
                           my_set, myelt, init_thread, iterations⟩

init9(self) ≜ ∧ pc[self] = "init9"
              ∧ initialized
              ∧ pc' = [pc EXCEPT ![self] = "th1"]
              ∧ UNCHANGED ⟨retVal, address, initialized, HEAD, stack, elt,
                           local_myhead_, local_myhead, reg_next, elt_,
                           my_set, myelt, init_thread, iterations⟩

th1(self) ≜ ∧ pc[self] = "th1"
            ∧ IF iterations[self] > 0
                 THEN ∧ pc' = [pc EXCEPT ![self] = "th2"]
                 ELSE ∧ pc' = [pc EXCEPT ![self] = "Done"]
            ∧ UNCHANGED ⟨retVal, address, initialized, HEAD, stack, elt,
                         local_myhead_, local_myhead, reg_next, elt_,
                         my_set, myelt, init_thread, iterations⟩

th2(self) ≜ ∧ pc[self] = "th2"
            ∧ ∨ ∧ IF my_set[self] ≠ {}
                     THEN ∧ pc' = [pc EXCEPT ![self] = "th2a"]
                     ELSE ∧ pc' = [pc EXCEPT ![self] = "th4"]
              ∨ ∧ pc' = [pc EXCEPT ![self] = "th3a"]
            ∧ UNCHANGED ⟨retVal, address, initialized, HEAD, stack, elt,
                         local_myhead_, local_myhead, reg_next, elt_,
                         my_set, myelt, init_thread, iterations⟩

th2a(self) ≜ ∧ pc[self] = "th2a"
             ∧ ∃ tmp ∈ my_set[self] :
                  myelt' = [myelt EXCEPT ![self] = tmp]
             ∧ pc' = [pc EXCEPT ![self] = "th2b"]
             ∧ UNCHANGED ⟨retVal, address, initialized, HEAD, stack, elt,
                          local_myhead_, local_myhead, reg_next, elt_,
                          my_set, init_thread, iterations⟩
```

$$
\begin{array}{lll}
th2b(self) & \triangleq & \land\ pc[self] = \text{“th2b”} \\
 & & \land\ my\_set' = [my\_set\ \text{EXCEPT}\ ![self] = my\_set[self] \setminus \{myelt[self]\}] \\
 & & \land\ pc' = [pc\ \text{EXCEPT}\ ![self] = \text{“th2c”}] \\
 & & \land\ \text{UNCHANGED}\ \langle retVal, address, initialized, HEAD, stack, elt, \\
 & & \qquad\qquad local\_myhead\_, local\_myhead, reg\_next, elt\_, \\
 & & \qquad\qquad myelt, init\_thread, iterations\rangle
\end{array}
$$

$$
\begin{array}{lll}
th2c(self) & \triangleq & \land\ pc[self] = \text{“th2c”} \\
 & & \land\ \land\ elt' = [elt\ \text{EXCEPT}\ ![self] = myelt[self]] \\
 & & \quad \land\ stack' = [stack\ \text{EXCEPT}\ ![self] = \langle[\begin{array}{lll} procedure & \mapsto & \text{“pushStack”}, \\ pc & \mapsto & \text{“th4”}, \\ local\_myhead\_ & \mapsto & local\_myhead\_[self], \\ elt & \mapsto & elt[self]]\rangle \end{array} \\
 & & \qquad\qquad\qquad\qquad\qquad\qquad \circ\ stack[self]] \\
 & & \land\ local\_myhead\_' = [local\_myhead\_\ \text{EXCEPT}\ ![self] = defaultInitValue] \\
 & & \land\ pc' = [pc\ \text{EXCEPT}\ ![self] = \text{“push1”}] \\
 & & \land\ \text{UNCHANGED}\ \langle retVal, address, initialized, HEAD, local\_myhead, \\
 & & \qquad\qquad reg\_next, elt\_, my\_set, myelt, init\_thread, \\
 & & \qquad\qquad iterations\rangle
\end{array}
$$

$$
\begin{array}{lll}
th3a(self) & \triangleq & \land\ pc[self] = \text{“th3a”} \\
 & & \land\ stack' = [stack\ \text{EXCEPT}\ ![self] = \langle[\begin{array}{lll} procedure & \mapsto & \text{“popStack”}, \\ pc & \mapsto & \text{“th3b”}, \\ local\_myhead & \mapsto & local\_myhead[self], \\ reg\_next & \mapsto & reg\_next[self], \\ elt\_ & \mapsto & elt\_[self]]\rangle \end{array} \\
 & & \qquad\qquad\qquad\qquad\qquad\qquad \circ\ stack[self]] \\
 & & \land\ local\_myhead' = [local\_myhead\ \text{EXCEPT}\ ![self] = defaultInitValue] \\
 & & \land\ reg\_next' = [reg\_next\ \text{EXCEPT}\ ![self] = defaultInitValue] \\
 & & \land\ elt\_' = [elt\_\ \text{EXCEPT}\ ![self] = defaultInitValue] \\
 & & \land\ pc' = [pc\ \text{EXCEPT}\ ![self] = \text{“tryAgainPop”}] \\
 & & \land\ \text{UNCHANGED}\ \langle retVal, address, initialized, HEAD, elt, \\
 & & \qquad\qquad local\_myhead\_, my\_set, myelt, init\_thread, \\
 & & \qquad\qquad iterations\rangle
\end{array}
$$

$$
\begin{array}{lll}
th3b(self) & \triangleq & \land\ pc[self] = \text{“th3b”} \\
 & & \land\ \text{IF}\ retVal[self] \neq NULL \\
 & & \qquad \text{THEN}\ \land\ pc' = [pc\ \text{EXCEPT}\ ![self] = \text{“th3c”}] \\
 & & \qquad \text{ELSE}\ \ \land\ pc' = [pc\ \text{EXCEPT}\ ![self] = \text{“th4”}] \\
 & & \land\ \text{UNCHANGED}\ \langle retVal, address, initialized, HEAD, stack, elt, \\
 & & \qquad\qquad local\_myhead\_, local\_myhead, reg\_next, elt\_, \\
 & & \qquad\qquad my\_set, myelt, init\_thread, iterations\rangle
\end{array}
$$

$$
\begin{array}{lll}
th3c(self) & \triangleq & \land\ pc[self] = \text{“th3c”} \\
 & & \land\ my\_set' = [my\_set\ \text{EXCEPT}\ ![self] = my\_set[self] \cup \{retVal[self]\}] \\
 & & \land\ pc' = [pc\ \text{EXCEPT}\ ![self] = \text{“th4”}]
\end{array}
$$

$$\land \text{UNCHANGED } \langle retVal, address, initialized, HEAD, stack, elt, local\_myhead\_, local\_myhead, reg\_next, elt\_, myelt, init\_thread, iterations \rangle$$

$$
\begin{aligned}
th4(self) \triangleq\ & \land pc[self] = \text{“th4”} \\
& \land iterations' = [iterations \text{ EXCEPT } ![self] = iterations[self] - 1] \\
& \land PrintT(iterations'[self]) \\
& \land pc' = [pc \text{ EXCEPT } ![self] = \text{“th1”}] \\
& \land \text{UNCHANGED } \langle retVal, address, initialized, HEAD, stack, elt, \\
& \qquad local\_myhead\_, local\_myhead, reg\_next, elt\_, \\
& \qquad my\_set, myelt, init\_thread \rangle
\end{aligned}
$$

$$
\begin{aligned}
thread(self) \triangleq\ & init1(self) \lor init2(self) \lor init4(self) \lor init5(self) \\
& \lor init6(self) \lor init7(self) \lor init8(self) \\
& \lor init9(self) \lor th1(self) \lor th2(self) \lor th2a(self) \\
& \lor th2b(self) \lor th2c(self) \lor th3a(self) \lor th3b(self) \\
& \lor th3c(self) \lor th4(self)
\end{aligned}
$$

$$
\begin{aligned}
Next \triangleq\ & (\exists\, self \in ProcSet : pushStack(self) \lor popStack(self)) \\
& \lor (\exists\, self \in 1 \,..\, NUM\_THREADS : thread(self)) \\
& \lor \quad \text{Disjunct to prevent deadlock on termination} \\
& \quad ((\forall\, self \in ProcSet : pc[self] = \text{“Done”}) \land \text{UNCHANGED } vars)
\end{aligned}
$$

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

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

END TRANSLATION

* Modification History
* Last modified *Thu Apr* 11 15:49:50 *EDT* 2019 by gene
* Last modified Sun *Oct* 22 06:26:25 *EDT* 2017 by celestekostopulos-cooperman
* Created *Fri Oct* 20 22:08:53 *EDT* 2017 by celestekostopulos-cooperman