```
1 ---------- MODULE Counter ----------
  TLA+ module for Op-based Counter. See its implementation in paper Burckhardt@POPL'2014.
  We check that the Op-based Counter satisfies the strong eventual convergence property (SEC)

10 EXTENDS Naturals, Sequences, Bags, TLC

12 CONSTANTS
13     Replica,    the set of replicas
14     Max         Max[r]: the maximum number of the Inc() event replica r ∈ Replica can issue

16 VARIABLES
       Protocol variables.
20     counter,    counter[r]: the current value of the counter at replica r ∈ Replica
21     acc,        acc[r]: the number of increments performed since the last broadcast at replica r ∈ Replica
22     incoming,   incoming[r]: incoming messages at replica r ∈ Replica
       Auxiliary variables for model checking.
26     inc         inc[r]: the number of Inc() events issued by the replica r ∈ Replica; for finite-state model checking

  The type correctness predicate.
31 TypeOK ≜ ∧ counter ∈ [Replica → Nat]
32          ∧ acc ∈ [Replica → Nat]
33          ∧ incoming ∈ [Replica → SubBag(SetToBag(Nat))] \* message ordering is not important; using bag (i.e., mul
34          ∧ inc ∈ [Replica → Nat]

36 ----------

  The initial state predicate.
40 Init ≜ ∧ counter = [r ∈ Replica ↦ 0]
41        ∧ acc = [r   ∈ Replica ↦ 0]
42        ∧ incoming = [r ∈ Replica ↦ EmptyBag]
43        ∧ inc = [r   ∈ Replica ↦ 0]

45 ----------

  Replica r ∈ Replica issues an Inc() event.
49 Inc(r) ≜ ∧ TRUE    no pre-condition
50          ∧ counter' = [counter EXCEPT ![r] = @ + 1]    current counter + 1
51          ∧ acc' = [acc EXCEPT ![r] = @ + 1]    # of accumulated increments + 1
52          ∧ inc' = [inc EXCEPT ![r] = @ + 1]    # of increments + 1
53          ∧ UNCHANGED ⟨incoming⟩

  Broadcast a message m to all replicas except the sender s.
58 Broadcast(s, m) ≜ [r ∈ Replica ↦
59                      IF s = r
60                       THEN incoming[s]
61                       ELSE incoming[r] ⊕ SetToBag({m})]
```

Replica r issues a *Send*() event, sending an update message.

$Send(r) \triangleq \land acc[r] \neq 0$ there are accumulated increments
$\land acc' = [acc \text{ EXCEPT } ![r] = 0]$ reset $acc[r]$
$\land incoming' = Broadcast(r,\ acc[r])$ broadcast $acc[r]$ to other replicas
$\land \text{UNCHANGED } \langle counter,\ inc \rangle$

Replica r issues a *Receive*() event, receiving an update message.

$Receive(r) \triangleq \land incoming[r] \neq EmptyBag$ there are accumulated increments from other replicas
$\land \exists\, m \in BagToSet(incoming[r]) :$ message reordering can be tolerant
$(\land counter' = [counter \text{ EXCEPT } ![r] = @ + m]$
$\land incoming' = [incoming \text{ EXCEPT } ![r] = @ \ominus SetToBag(\{m\})])$ each message is delivered exa
$\land \text{UNCHANGED } \langle acc,\ inc \rangle$

The Next-state relation.

$Next \triangleq \land \exists\, r \in Replica : Inc(r) \lor Send(r) \lor Receive(r)$

The specification.

$vars \triangleq \langle counter,\ acc,\ incoming,\ inc \rangle$
$Spec \triangleq Init \land \Box[Next]_{vars} \land \text{WF}_{vars}(Next)$

A state constraint that is useful for validating the specification using finite-state model checking: each replica $r \in Replica$ can issue at most $Max[r]Inc()$ events.

$IncConstraint \triangleq \forall\, r \in Replica : inc[r] \leq Max[r]$

The correctness of counter: eventual convergence (EC) and strong eventual convergence (SEC).

$Convergence \triangleq \forall\, r,\ s \in Replica : (counter[r] = counter[s] \land counter[r] \neq 0)$ $counter[r] \neq 0$: excluding the initial s
$EC \triangleq \Diamond Convergence$

$AccBroadcast \triangleq \forall\, r \in Replica : acc[r] = 0$ all accumulated increments have been broadcast
$MessageDelivery \triangleq \forall\, r \in Replica : incoming[r] = \langle\rangle$ all messages have been delivered
$SConvergence \triangleq \forall\, r,\ s \in Replica : counter[r] = counter[s]$ no $counter[r] \neq 0$

$SEC \triangleq \Box((AccBroadcast \land MessageDelivery) \Rightarrow SConvergence)$

---

* Modification History
* Last modified *Mon Jun* 04 19:52:43 *CST* 2018 by *hengxin*
* Created Sun *Jun* 03 20:08:57 *CST* 2018 by *hengxin*