```
┌──────────────────────────────── MODULE Hermes ────────────────────────────────┐
EXTENDS    Integers

CONSTANTS   NODES,
            MAX_VERSION

VARIABLES  msgs,
           nodeTS,
           nodeState,
           nodeLastWriter,
           issuedWriteTS,
           aliveNodes,
           receivedAcks

 The consistent invariant: all alive nodes in valid state should have the same value / TS
HConsistent ≜
    ∀ k, s ∈ aliveNodes :  ∨ nodeState[k] ≠ "valid"
                           ∨ nodeState[s] ≠ "valid"
                           ∨ nodeTS[k] = nodeTS[s]

HMessage ≜
    [type : {"INV", "ACK"}, sender     : NODES,
                            version    : 0 . . MAX_VERSION,
                            tieBreaker : NODES]
        ∪
    [type : {"VAL"},        version    : 0 . . MAX_VERSION,
                            tieBreaker : NODES]

HTypeOK ≜    The type correctness invariant
    ∧     msgs            ⊆ HMessage
    ∧     aliveNodes      ⊆ NODES
    ∧ ∀ n ∈ NODES : receivedAcks[n] ⊆ (NODES \ {n})
    ∧  nodeLastWriter ∈ [NODES → NODES]
    ∧  issuedWriteTS  ∈ [NODES → [version    : 0 . . MAX_VERSION,
                                  tieBreaker : NODES           ]]
    ∧  nodeTS         ∈ [NODES → [version    : 0 . . MAX_VERSION,
                                  tieBreaker : NODES           ]]
    ∧  nodeState      ∈ [NODES → {"valid", "invalid", "invalid_write",
                                  "write", "replay"}]

HInit ≜   The initial predicate
    ∧  msgs            = {}
    ∧  aliveNodes      = NODES
    ∧  receivedAcks    = [n ∈ NODES ↦ {}]
    ∧  nodeState       = [n ∈ NODES ↦ "valid"]
    ∧  nodeLastWriter  = [n ∈ NODES ↦ CHOOSE k ∈ NODES :
                                       ∀ m ∈ NODES : k ≤ m]
```

```
   ∧ nodeTS            = [n ∈ NODES ↦ [version ↦ 0,
                                       tieBreaker ↦
                                        CHOOSE k ∈ NODES :
                                         ∀ m ∈ NODES : k ≤ m]]
   ∧ issuedWriteTS     = [n ∈ NODES ↦ [version ↦ 0,
                                       tieBreaker ↦
                                        CHOOSE k ∈ NODES :
                                         ∀ m ∈ NODES : k ≤ m]]
├──────────────────────────────────────────────────────────────────┤
send(m) ≜ msgs′ = msgs ∪ {m}

receivedAllAcks(n) ≜ receivedAcks[n] = NODES \ {n}

equalTS(v1, tb1, v2, tb2) ≜
   ∧  v1 = v2
   ∧  tb1 = tb2

greaterTS(v1, tb1, v2, tb2) ≜
   ∨ v1 > v2
   ∨ ∧  v1 = v2
     ∧ tb1 > tb2

isAlive(n) ≜ n ∈ aliveNodes

nodeFailure ≜
   ∧ aliveNodes′ = aliveNodes \ {CHOOSE k ∈ aliveNodes : ∀ m ∈ aliveNodes : k ≤ m}
   ∧ UNCHANGED ⟨msgs, nodeState, nodeTS, nodeLastWriter, issuedWriteTS, receivedAcks⟩
├──────────────────────────────────────────────────────────────────┤
HRead(n) ≜
   ∧ nodeState[n] = "valid"
   ∧ UNCHANGED ⟨msgs, nodeTS, nodeState, nodeLastWriter,
                 aliveNodes, issuedWriteTS, receivedAcks⟩

HWrite(n) ≜
   ∧ nodeState[n]         ∈ {"valid"}
   ∧ nodeTS[n].version < MAX_VERSION
   ∧ receivedAcks′        = [receivedAcks   EXCEPT ![n]  = {}]
   ∧ nodeLastWriter′      = [nodeLastWriter EXCEPT ![n] = n]
   ∧ nodeState′           = [nodeState      EXCEPT ![n]  = "write"]
   ∧ nodeTS′              = [nodeTS          EXCEPT ![n].version    =
                                                      nodeTS[n].version + 1,
                                                  ![n].tieBreaker = n]
   ∧ issuedWriteTS′       = [issuedWriteTS  EXCEPT ![n].version    =
                                                      nodeTS[n].version + 1,
                                                  ![n].tieBreaker = n]
   ∧ send([type       ↦ "INV",
           sender     ↦ n,
           version     ↦ nodeTS[n].version + 1,
           tieBreaker  ↦ n])
∧ UNCHANGED ⟨aliveNodes⟩

HReplayWrite(n) ≜
   ∧ nodeState[n] = "invalid"
   ∧ ¬isAlive(nodeLastWriter[n])
   ∧ nodeLastWriter′ = [nodeLastWriter EXCEPT ![n] = n]
   ∧ nodeState′      = [nodeState      EXCEPT ![n]  = "replay"]
   ∧ receivedAcks′   = [receivedAcks   EXCEPT ![n]  = {}]
   ∧ issuedWriteTS′  = [issuedWriteTS  EXCEPT ![n] = nodeTS[n]]
   ∧ send([type        ↦ "INV",
           sender      ↦ n,
           version     ↦ nodeTS[n].version,
           tieBreaker  ↦ nodeTS[n].tieBreaker])
   ∧ UNCHANGED ⟨nodeTS, aliveNodes⟩
────────────────────────────────────────────────────────────────────────
HRcvAck(n) ≜
    ∃ m ∈ msgs :
     ∧ m.type = "ACK"
     ∧ m.sender ≠ n
     ∧ m.sender ∉ receivedAcks[n]
     ∧ equalTS(m.version,
               m.tieBreaker,
               issuedWriteTS[n].version,
               issuedWriteTS[n].tieBreaker)
     ∧ nodeState[n] ∈ {"write", "invalid_write", "replay"}
     ∧ receivedAcks′ = [receivedAcks EXCEPT ![n] =
                                 receivedAcks[n] ∪ {m.sender}]
     ∧ UNCHANGED ⟨msgs, nodeLastWriter, issuedWriteTS,
                  aliveNodes, nodeTS, nodeState⟩

HSendVals(n) ≜
   ∧ nodeState[n] ∈ {"write", "replay"}
   ∧ receivedAllAcks(n)
   ∧ nodeState′       = [nodeState EXCEPT ![n] = "valid"]
   ∧ send([type        ↦ "VAL",
           version     ↦ nodeTS[n].version,
           tieBreaker  ↦ nodeTS[n].tieBreaker])
   ∧ UNCHANGED ⟨nodeTS, nodeLastWriter, issuedWriteTS,
                 aliveNodes, receivedAcks⟩

HCoordinatorActions(n) ≜
   ∨ HRead(n)
   ∨ HReplayWrite(n)  this is for failures
   ∨ HWrite(n)
    ∨ HRcvAck(n)
    ∨ HSendVals(n)
├──────────────────────────────────────────────────────────────────────┤
HRcvInv(n) ≜
    ∃ m ∈ msgs :
     ∧ m.type = "INV"
     ∧ m.sender ≠ n
     ∧ send([type       ↦ "ACK",
             sender     ↦ n,
             version    ↦ m.version,
             tieBreaker ↦ m.tieBreaker])
     ∧ ∨ ∧ greaterTS(m.version,
                     m.tieBreaker,
                     nodeTS[n].version,
                     nodeTS[n].tieBreaker)
         ∧ nodeLastWriter′ = [nodeLastWriter EXCEPT ![n] = m.sender]
         ∧ nodeTS′ = [nodeTS EXCEPT ![n].version    = m.version,
                                    ![n].tieBreaker = m.tieBreaker]
         ∧ ∨ ∧ nodeState[n] ∈ {"valid", "invalid", "replay"}
             ∧ nodeState′ = [nodeState EXCEPT ![n] = "invalid"]
           ∨ ∧ nodeState[n] ∈ {"write", "invalid_write"}
             ∧ nodeState′ = [nodeState EXCEPT ![n] = "invalid_write"]
       ∨ ∧ ¬greaterTS(m.version,
                      m.tieBreaker,
                      nodeTS[n].version,
                      nodeTS[n].tieBreaker)
         ∧ UNCHANGED ⟨nodeState, nodeTS, nodeLastWriter⟩
     ∧ UNCHANGED ⟨issuedWriteTS, aliveNodes, receivedAcks⟩

HRcvVal(n) ≜
    ∃ m ∈ msgs :
     ∧ nodeState[n] ≠ "valid"
     ∧ m.type = "VAL"
     ∧ equalTS(m.version,
                 m.tieBreaker,
                 nodeTS[n].version,
                 nodeTS[n].tieBreaker)
     ∧ nodeState′ = [nodeState EXCEPT ![n] = "valid"]
     ∧ UNCHANGED ⟨msgs, nodeTS, nodeLastWriter, issuedWriteTS,
                  aliveNodes, receivedAcks⟩

HFollowerActions(n) ≜
    ∨ HRcvInv(n)
    ∨ HRcvVal(n)
├──────────────────────────────────────────────────────────────────────┤
```

```
$HNext \triangleq$
    $\lor \exists n \in NODES :$
        $\lor HFollowerActions(n)$
        $\lor HCoordinatorActions(n)$
    $\lor nodeFailure$ this is for failures
```

* Modification History
* Last modified *Fri Jul* 13 15:20:40 *BST* 2018 by *akatsarakis*
* Created *Tue Jul* 10 09:43:12 *BST* 2018 by *akatsarakis*