```
──────────────────── MODULE cacheinvalidationv2 ────────────────────

  Cache incorporates the data
CacheCompleteFill(k) ≜
    ∧ cacheFillStates[k].state = "respondedto"
        Either the cache is empty for that key
    ∧ ∨ cache[k] ∈ CacheMiss
        or we are filling a newer version
      ∨ ∧ cache[k] ∉ CacheMiss
        ∧ cache[k].version < cacheFillStates[k].version
    ∧ cacheFillStates′ = [cacheFillStates EXCEPT   Reset to 0
                            ![k].state = "inactive",
                            ![k].version = 0
                          ]
    ∧ cache′ = [cache EXCEPT
                       ![k] = [
                          type ↦ "hit",
                          version ↦ cacheFillStates[k].version
                       ]
               ]
    ∧ UNCHANGED ⟨database, invalidationQueue⟩

CacheIgnoreFill(k) ≜
    ∧ cacheFillStates[k].state = "respondedto"
      If we have a newer version in cache, ignore fill
    ∧ ∧ cache[k] ∈ CacheHit
      ∧ cache[k].version ≥ cacheFillStates[k].version
    ∧ cacheFillStates′ = [cacheFillStates EXCEPT   Reset to 0
                            ![k].state = "inactive",
                            ![k].version = 0
                          ]
      Don't update cache
    ∧ UNCHANGED ⟨cache, database, invalidationQueue⟩


CacheHandleInvalidationMessage ≜
    ∧ ∃ message ∈ invalidationQueue :   Dequeue invalidation queue in any order
            Key must be in cache
        ∧ ∧ cache[message.key] ∈ CacheHit
            Message needs to be newer than the cache
          ∧ cache[message.key].version < message.version
          Update item in cache
        ∧ cache′ = [cache EXCEPT
                       ![message.key] = [
                          type ↦ "hit",
                            Update to version in invalidation message
```

```
                    $version \mapsto message.version$
                  $]]$
        Remove message from queue because handled
        $\land\ invalidationQueue' = invalidationQueue \setminus \{message\}$

    $\land$ UNCHANGED $\langle cacheFillStates,\ database \rangle$

$CacheIgnoreInvalidationMessage \triangleq$
    $\land\ \exists\, message \in invalidationQueue :$  Dequeue invalidation queue in any order
          Ignore invalidation messages for messages not in cache
        $\land\ \lor\ cache[message.key] \in CacheMiss$
          Or when the cache already has the same or larger version
          $\lor\ \land\ cache[message.key] \notin CacheMiss$
             $\land\ cache[message.key].version \geq message.version$
        Remove message from queue to ignore
        $\land\ invalidationQueue' = invalidationQueue \setminus \{message\}$
    Don't update cache
    $\land$ UNCHANGED $\langle cacheFillStates,\ database,\ cache \rangle$
```

* Modification History
* Last modified *Wed Jun* 15 13:58:25 *MST* 2022 by *elliotswart*
* Created *Wed Jun* 15 13:58:13 *MST* 2022 by *elliotswart*