MODULE *SimpleStore_Regular*

The model represents the model of a simple storage system that provides *regular* register semantics. It works as follows: - It has a queue of incoming *pending_rd* and a queue of incoming *pending_wr* (*pending_rd*, resp *pending_wr*). These queues are very flacky meaning that they may drop requests when they like. - It has a reliable store (oviously single-copy), which takes out *pending_rd* and *pending_wr* from the queues one at a time and executes them. Once a write is executed by the store, there's no turning back. The write cannot be dropped anymore. - Writes: are appended to the *pending_wr* and are later dropped or executed by the store. - Reads: are appended to the *pending_rd* and are later dropped or executed by the store in the following way. A read can return EITHER the content of the store (so the last committed value) OR the value of one of the *pending_wr* that are still pending in *pending_wr* (so value of some overlapping write).

Thus, the model corresponds to "regular registers" in *Lamport* terms [On Inter-Process Communication]. Reads that don't overlap any write return the last committed value and *pending_rd* that overlap some *pending_wr* return either the alst committed value or the value of any overlapping write.

EXTENDS *simplestore_quickrd* extends the linearizable simple store

$SSR\_TypeInvariant \triangleq SS1\_TypeInvariant$

$SSR\_Init \triangleq SS1\_Init$

STORE operations:

Reads: not serialized w/ *pending_wr*. A read can return either the value of the last committed wr, or the value of one of the pending (overlapping) *pending_wr*.

```
$SSR\_HdlRead \triangleq$   handle one read requests
          EITHER get the store value (value of last committed write),
    $\land$  $\lor$ $last\_read\_val' = store$

          OR the value of one of the overlapping pending write requests
       $\lor$ $\exists\, idx \in 1 \,..\, Len(pending\_wrreq)$ :
           $\land$ $last\_read\_val' = pending\_wrreq[idx]$
       $\lor$ $\exists\, v \in Val$ :
           $\land$ $pending\_wrresp[v] > 0$   there is a write pending for this value
           $\land$ $last\_read\_val' = v$        read this value

       $\lor$ $\exists\, v \in Val$ :
          $\land$ $failed\_wr[v] > 0$ there's a failed write for this valye
          $\land$ $last\_read\_val' = v$   read this value

    $\land$ UNCHANGED $\langle pending\_rd \rangle$

$SSR\_RunStore \triangleq$
    $\lor$ $\land$ $SSR\_HdlRead$
       $\land$ UNCHANGED $\langle pending\_wrresp, pending\_wrreq, failed\_wr, store \rangle$
    $\lor$
       $\land$ $SS\_CommitWrite$
       $\land$ UNCHANGED $\langle pending\_rd, last\_read\_val \rangle$
```

Full specification.

$SSR\_Next \triangleq$

$\lor SS1\_Client$ a client submits a request (query or update)

$\lor SSR\_RunStore$ the store deals w/ the updates

$\lor SS\_ChannelActions$ the incoming channel for the store drops some requests

$ssrvars \triangleq ss1vars$

$SSR\_Spec \triangleq SSR\_Init \land \Box[SSR\_Next]_{ssrvars}$

Invariants

$SSR\_AllInvariants \triangleq$

$\land SSR\_TypeInvariant$

Theorem

THEOREM $SSR\_Spec \Rightarrow \Box SSR\_AllInvariants$