
MODULE *Messages*

EXTENDS *Naturals, Sequences*

Stream states

CONSTANTS *Open, Closed*

Master arbitration message types

CONSTANTS *MasterArbitrationUpdate*

Write message types

CONSTANTS *WriteRequest, WriteResponse*

Response status constants

CONSTANTS *Ok, AlreadyExists, PermissionDenied*

An empty value

CONSTANT *Nil*

The state of all streams and their requests and responses

VARIABLE *requestStream, requests, responseStream, responses*

Message related variables

$messageVars \triangleq \langle requests, responses \rangle$

Stream related variables

$streamVars \triangleq \langle requestStream, responseStream \rangle$

This section models the messaging between controller nodes and the device. Messaging is modelled on *TCP*, providing strict ordering between controller and device via sequences. The 'requests' sequence represents the messages from controller to device for each node, and the 'responses' sequence represents the messages from device to each node. Requests and responses are always received from the head of the queue and are never duplicated or reordered.

Returns a sequence with the head removed

$Pop(q) \triangleq SubSeq(q, 2, Len(q))$

Sends request 'm' on the stream for node 'n'

$SendRequest(n, m) \triangleq requests' = [requests \text{ EXCEPT } ![n] = Append(requests[n], m)]$

Indicates whether a request of type 't' is at the head of the queue for node 'n'

$HasRequest(n, t) \triangleq Len(requests[n]) > 0 \wedge requests[n][1].type = t$

Returns the next request in the queue for node 'n'

$NextRequest(n) \triangleq requests[n][1]$

Discards the request at the head of the queue for node 'n'

$DiscardRequest(n) \triangleq requests' = [requests \text{ EXCEPT } ![n] = Pop(requests[n])]$

Sends response 'm' on the stream for node 'n'

$$SendResponse(n, m) \triangleq responses' = [responses \text{ EXCEPT } ![n] = Append(responses[n], m)]$$

Indicates whether a response of type 't' is at the head of the queue for node 'n'

$$HasResponse(n, t) \triangleq Len(responses[n]) > 0 \wedge responses[n][1].type = t$$

Returns the next response in the queue for node 'n'

$$NextResponse(n) \triangleq responses[n][1]$$

Discards the response at the head of the queue for node 'n'

$$DiscardResponse(n) \triangleq responses' = [responses \text{ EXCEPT } ![n] = Pop(responses[n])]$$

\ * Modification History
\ * Last modified *Thu Feb 21 15:08:51 PST 2019* by *jordanhalterman*
\ * Created *Wed Feb 20 23:49:28 PST 2019* by *jordanhalterman*