

```

1  ┌────────────────────────── MODULE lww_register ───────────────────────────┐
2  EXTENDS Integers, Sequences, TLC
3  CONSTANTS N
4
5  Procs  $\triangleq$  1 .. N
6
7  INITIAL  $\triangleq$  [t  $\mapsto$  0, val  $\mapsto$  ""] initial value [t  $\mapsto$  0, val  $\mapsto$  ""]
8
9
10 --algorithm lww_register
11 variables
12   msgs = [j  $\in$  Procs  $\mapsto$  INITIAL]; to send the messages
13
14 define
15    $\_Compare(p1, p2) \triangleq p1.t \leq p2.t$ 
16
17    $\_Merge(p1, p2) \triangleq$  IF  $\_Compare(p1, p2)$  THEN p1 ELSE p2
18 end define ;
19
20 assign a value and timestamp into local payload
21 macro  $\_Assign(v)$  begin
22   payload := [t  $\mapsto$  JavaTime, val  $\mapsto$  v];
23   print ToString(self)  $\circ$  " assigned "  $\circ$  ToString(payload);
24 end macro ;
25
26 send the payload 'p' to a random proc
27 macro  $\_Send(p)$  begin
28   if payload  $\neq$  INITIAL then
29     with j  $\in$  Procs  $\setminus$  {self} do
30       msgs[j] := payload;
31       print ToString(self)  $\circ$  " sent "  $\circ$  ToString(msgs[j])  $\circ$  " to "  $\circ$  ToString(j);
32     end with ;
33   end if ;
34 end macro ;
35
36 receive the payload
37 macro  $\_Receive()$  begin
38   if msgs[self]  $\neq$  INITIAL then
39     print ToString(self)  $\circ$  " received "  $\circ$  ToString(msgs[self]);
40     if payload = INITIAL then payload is empty, just receive the message
41       payload := msgs[self];
42     else merge the payload and received message
43       payload :=  $\_Merge(payload, msgs[self])$ ;
44     end if ;
45     msgs[self] := INITIAL;
46     print ToString(self)  $\circ$  " merged "  $\circ$  ToString(payload);
47   end if ;
48 end macro ;

```

```

50 fair process Register  $\in$  Procs
51 variables
52    $i = 0$ , count iterations
53   payload = INITIAL; local payload
54 begin Main:
55   while  $i < N$  do
56     either Assign:
57        $\_Assign(self)$ ;
58     or Send:
59        $\_Send(payload)$ ;
60     or Receive:
61        $\_Receive()$ ;
62     end either ;
63     Loop:
64        $i := i + 1$ ;
65   end while ;
66 end process ;

68 end algorithm ;
70 BEGIN TRANSLATION
71 VARIABLES msgs, pc

73 define statement
74  $\_Compare(p1, p2) \triangleq p1.t \leq p2.t$ 
76  $\_Merge(p1, p2) \triangleq$  IF  $\_Compare(p1, p2)$  THEN p1 ELSE p2
78 VARIABLES i, payload
80 vars  $\triangleq \langle msgs, pc, i, payload \rangle$ 
82 ProcSet  $\triangleq (Procs)$ 
84 Init  $\triangleq$  Global variables
85    $\wedge msgs = [j \in Procs \mapsto INITIAL]$ 
86   Process Register
87    $\wedge i = [self \in Procs \mapsto 0]$ 
88    $\wedge payload = [self \in Procs \mapsto INITIAL]$ 
89    $\wedge pc = [self \in ProcSet \mapsto "Main"]$ 
91 Main(self)  $\triangleq$   $\wedge pc[self] = "Main"$ 
92    $\wedge$  IF  $i[self] < N$ 
93     THEN  $\wedge \vee \wedge pc' = [pc \text{ EXCEPT } ![self] = "Assign"]$ 
94        $\vee \wedge pc' = [pc \text{ EXCEPT } ![self] = "Send"]$ 
95        $\vee \wedge pc' = [pc \text{ EXCEPT } ![self] = "Receive"]$ 
96     ELSE  $\wedge pc' = [pc \text{ EXCEPT } ![self] = "Done"]$ 
97    $\wedge$  UNCHANGED  $\langle msgs, i, payload \rangle$ 

```

```

99   $Loop(self) \triangleq \wedge pc[self] = \text{"Loop"}$ 
100     $\wedge i' = [i \text{ EXCEPT } ![self] = i[self] + 1]$ 
101     $\wedge pc' = [pc \text{ EXCEPT } ![self] = \text{"Main"}]$ 
102     $\wedge \text{UNCHANGED } \langle msgs, payload \rangle$ 

104   $Assign(self) \triangleq \wedge pc[self] = \text{"Assign"}$ 
105     $\wedge payload' = [payload \text{ EXCEPT } ![self] = [t \mapsto JavaTime, val \mapsto self]]$ 
106     $\wedge PrintT(ToString(self) \circ \text{" assigned " } \circ ToString(payload'[self]))$ 
107     $\wedge pc' = [pc \text{ EXCEPT } ![self] = \text{"Loop"}]$ 
108     $\wedge \text{UNCHANGED } \langle msgs, i \rangle$ 

110   $Send(self) \triangleq \wedge pc[self] = \text{"Send"}$ 
111     $\wedge \text{IF } payload[self] \neq INITIAL$ 
112       $\text{THEN } \wedge \exists j \in Procs \setminus \{self\} :$ 
113         $\wedge msgs' = [msgs \text{ EXCEPT } ![j] = payload[self]]$ 
114         $\wedge PrintT(ToString(self) \circ \text{" sent " } \circ ToString(msgs'[j]) \circ \text{" to " } \circ ToString(j))$ 
115       $\text{ELSE } \wedge \text{TRUE}$ 
116         $\wedge msgs' = msgs$ 
117     $\wedge pc' = [pc \text{ EXCEPT } ![self] = \text{"Loop"}]$ 
118     $\wedge \text{UNCHANGED } \langle i, payload \rangle$ 

120   $Receive(self) \triangleq \wedge pc[self] = \text{"Receive"}$ 
121     $\wedge \text{IF } msgs[self] \neq INITIAL$ 
122       $\text{THEN } \wedge PrintT(ToString(self) \circ \text{" received " } \circ ToString(msgs[self]))$ 
123       $\wedge \text{IF } payload[self] = INITIAL$ 
124         $\text{THEN } \wedge payload' = [payload \text{ EXCEPT } ![self] = msgs[self]]$ 
125         $\text{ELSE } \wedge payload' = [payload \text{ EXCEPT } ![self] = \_Merge(payload[self], msgs[self])]$ 
126         $\wedge msgs' = [msgs \text{ EXCEPT } ![self] = INITIAL]$ 
127         $\wedge PrintT(ToString(self) \circ \text{" merged " } \circ ToString(payload'[self]))$ 
128       $\text{ELSE } \wedge \text{TRUE}$ 
129         $\wedge \text{UNCHANGED } \langle msgs, payload \rangle$ 
130     $\wedge pc' = [pc \text{ EXCEPT } ![self] = \text{"Loop"}]$ 
131     $\wedge i' = i$ 

133   $Register(self) \triangleq Main(self) \vee Loop(self) \vee Assign(self) \vee Send(self)$ 
134     $\vee Receive(self)$ 

136   $Next \triangleq (\exists self \in Procs : Register(self))$ 
137     $\vee \text{Disjunct to prevent deadlock on termination}$ 
138     $((\forall self \in ProcSet : pc[self] = \text{"Done"}) \wedge \text{UNCHANGED } vars)$ 

140   $Spec \triangleq \wedge Init \wedge \Box [Next]_{vars}$ 
141     $\wedge \forall self \in Procs : WF_{vars}(Register(self))$ 

143   $Termination \triangleq \Diamond (\forall self \in ProcSet : pc[self] = \text{"Done"})$ 

145  END TRANSLATION

```

147 |
| * Modification History
| * Last modified *Fri Dec 14 17:44:05 PST 2018* by *ocosta*
| * Created *Fri Dec 14 16:18:39 PST 2018* by *ocosta*