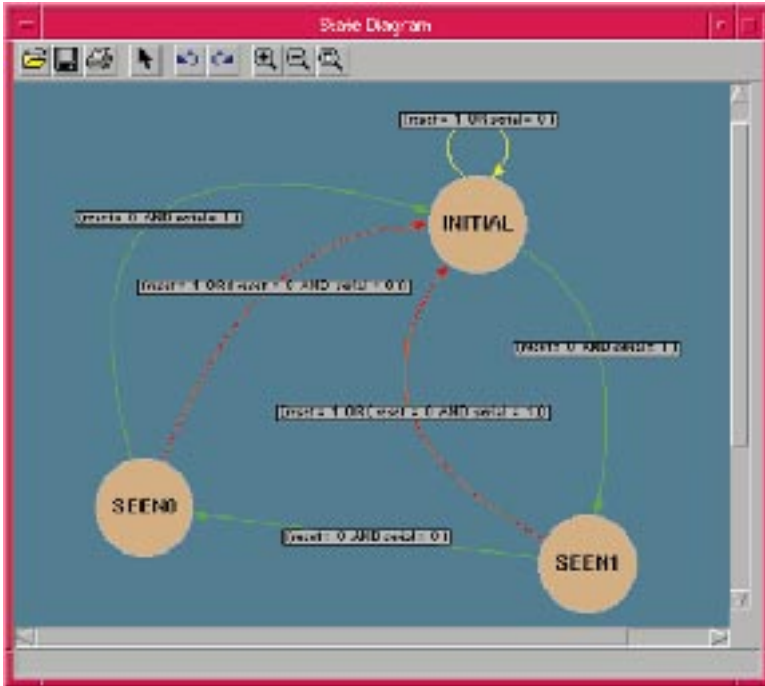


Figure 3

a)



b)

```
Code View
12 TYPE states IS (INITIAL, SEEN1, SEEN0);
13 SIGNAL state : states := INITIAL;
14 SIGNAL next_state : states := INITIAL;
15 BEGIN
16
17 -- process to latch state value
18 clk: PROCESS (clk, reset)
19 BEGIN
20   IF (clk'EVENT AND clk = '1' AND clk'LAST_VALUE = '0') THEN
21     state => next_state;
22   END IF;
23 END PROCESS clk;
24
25 -- process to determine next state
26 state_trans: PROCESS (state, reset, serial)
27 BEGIN
28   CASE state IS
29     WHEN INITIAL => IF (reset = '1' OR serial = '0') THEN
30       next_state => INITIAL;
31     ELSIF (reset = '0' AND serial = '1') THEN
32       next_state => SEEN0;
33     END IF;
34     WHEN SEEN1 => IF (reset = '1' OR (reset = '0' AND serial = '1')) THEN
35       next_state => INITIAL;
36     ELSIF (reset = '0' AND serial = '0') THEN
```

StateSure's code-coverage analysis of state machines lets you see untested transitions and expressions either in a state diagram (a) or as annotated Verilog or VHDL code (b).