

```
1  library ieee;
2  use ieee.std_logic_1164.all;
3  use work.all;
4
5  entity state_machine3 is
6      port(x0, x1: in std_logic;
7           clock : in std_logic;
8           reset: in std_logic;
9           y0, y1: out std_logic);
10 end state_machine3;
11
12 architecture behave of state_machine3 is
13
14     type states is (st1, st2, st3, st4);
15     signal present_state : states;
16     signal next_state: states;
17
18 begin
19
20     clkd: process(clock, reset)
21     begin
22         if (reset = '1') then
23             present_state <= st1;
24         elsif (clock'event and clock = '1') then
25             present_state <= next_state;
26         end if;
27     end process clkd;
28
29     state_trans: process(present_state, x0, x1)
30     begin
31         case present_state is
32             when st1 =>
33                 if (x0 = x1) then
34                     next_state <= st2;
35                     y0 <= x0; y1 <= x1;
36                 elsif (x0 = '0') then
37                     next_state <= st1;
38                     y0 <= '0'; y1 <= '1';
39                 else
40                     next_state <= st1;
41                     y0 <= '1'; y1 <= '1';
42                 end if;
43             when st2 =>
44                 if (x0='0' and x1='1') then
45                     next_state <= st3;
46                     y0 <= '0'; y1 <= '0';
47                 else
48                     next_state <= st4;
49                     y0 <= '1'; y1 <= '0';
50                 end if ;
51             when st3 =>
```

```
52         if (x0='0' and x1='0') then
53             next_state <= st3;
54             y0<='0'; y1<='0';
55         elsif (x0='1' and x1='0') then
56             next_state <= st4;
57             y0<='0'; y1<='1';
58         else
59             next_state <= st4;
60             y0<='0'; y1<='0';
61         end if ;
62     when st4 =>
63         if (x0='0' and x1='1') then
64             next_state <= st4;
65             y0<='1'; y1<='1';
66         elsif (x0='1' and x1='0') then
67             next_state <= st1;
68             y0<='1'; y1<='1';
69         else
70             next_state <= st2;
71             y0<='0'; y1<='1';
72         end if ;
73     end case;
74 end process state_trans;
75 end behave;
76
```