

```

1  module seqdetector_top(
2  input clock, ← clock
3  input reset, ← reset
4  input X,
5  output reg Z
6  );
7
8  (reg [2:0] state;)
9  (reg [2:0] next_state;] = D
10
11 // Define states as constants
12 parameter [2:0] 3-bit cons compile
13 S0 = 3'b000, time
14 S1 = 3'b001,
15 S2 = 3'b010,
16 S3 = 3'b011, 3'b101 = 3'd5
17 S4 = 3'b100,
18 S5 = 3'b101, value =
19 S6 = 3'b110; 3-bit: binary encod
20
21
22 // Register Block
23 always_ff @(posedge clock or posedge reset) begin
24     if (reset)
25         state <= S0;
26     else
27         state <= next_state; // Q* = D
28 end
29
30 // Next state block
31 always_comb begin
32     // case statement is like if else but the condition is on a single variable
33     case (state)
34     if state = S0: D
35         next_state <= X ? S0 : S1;
36     else if state = S1:
37         next_state <= X ? S0 : S2;
38     S2:
39         next_state <= X ? S4 : S3;
40     S3:
41         next_state <= X ? S5 : S3;
42     S4:
43         next_state <= X ? S0 : S6;
44     S5:
45         next_state <= X ? S0 : S6;
46     S6:
47         next_state <= X ? S0 : S2;
48     endcase
49 end
50
51 // Output block (Moore) Mealy
52 always_comb begin
53     case (state)
54     S5: Z = 1'b1;
55     S6: Z = 1'b1;
56     default: Z = 1'b0;
57     endcase
58 end
59 endmodule
60

```

logic / wire / reg $\in \{0, 1\}$ high impedance
 $\in \{0, 1, x, z\}$
 \uparrow
 undefined

initial block

Quartus
 +
 FPGAs

3-bit cons compile time
 $3'b101 = 3'd5$
 \uparrow
 decimal

3-bit: binary encod

if state = S0: D
 non blocking
 else if state = S1:
 next_state <= X ? S0 : S2;
 S2:
 next_state <= X ? S4 : S3;
 S3:
 next_state <= X ? S5 : S3;
 S4:
 next_state <= X ? S0 : S6;
 S5:
 next_state <= X ? S0 : S6;
 S6:
 next_state <= X ? S0 : S2;

$Y \in (X) ? a : b;$
 true
 = 1
 false