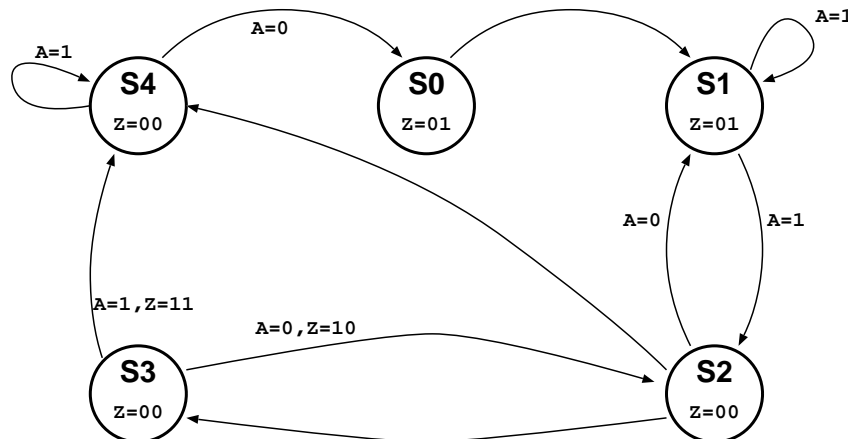


2 Finite State Machines

This question has three parts.

- (a) [20 points] An engineer has designed a deterministic finite state machine with a one-bit input (A) and a two-bit output (Z). He started the design by drawing the following state transition diagram:



Although the exact functionality of the FSM is not known to you, there are **at least three mistakes** in this diagram. Please list **all** the mistakes.

There are four problems with this diagram

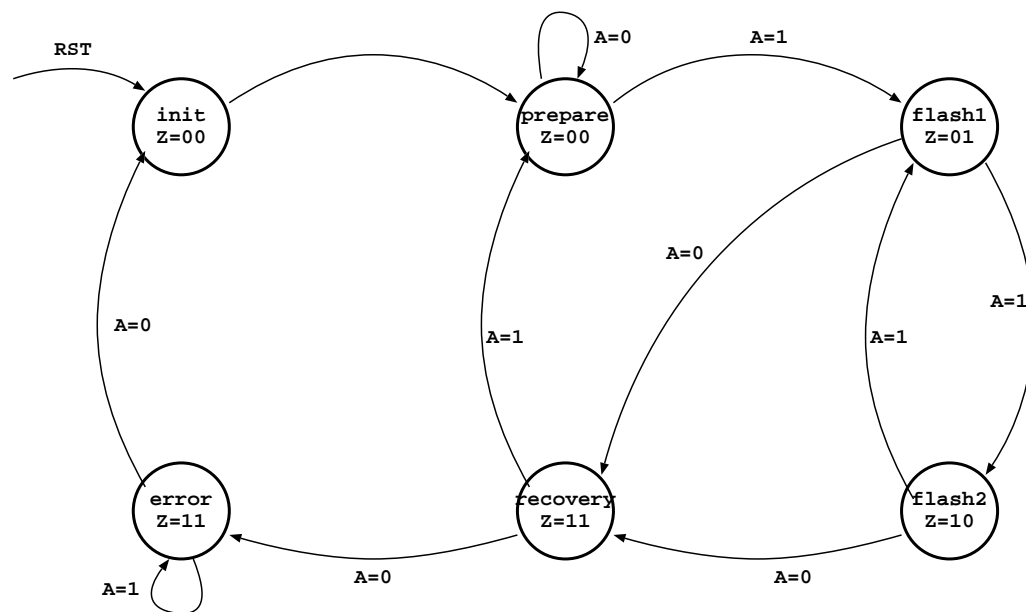
- Most states have a Moore labelling (output state in the bubble), one has a Mealy type labelling (output given with input transitions) (5 points)
- There are two different transitions both with $A = 1$ from state $S1$. What will happen with $A = 0$ is missing (5 points)
- There are two different transitions from state $S2$, without labeling which input triggers them (5 points)
- There is no reset state (5 points)

- (b) [25 points] After learning from his mistakes, your colleague has proceeded to write the following Verilog code for a much better (and **different**) FSM. The code has been verified for syntax errors and found to be OK.

```
1 module fsm (input CLK, RST, A, output [1:0] Z);
2
3     reg [2:0] nextState, presentState;
4
5     parameter start    = 3'b000;
6     parameter flash1   = 3'b010;
7     parameter flash2   = 3'b011;
8     parameter prepare   = 3'b100;
9     parameter recovery  = 3'b110;
10    parameter error     = 3'b111;
11
12    always @ (posedge CLK, posedge RST)
13        if (RST) presentState <= start;
14        else     presentState <= nextState;
15
16    assign Z = (presentState == recovery) ? 2'b11 :
17              (presentState == error)     ? 2'b11 :
18              (presentState == flash1)    ? 2'b01 :
19              (presentState == flash2)    ? 2'b10 : 2'b00;
20
21    always @ (presentState, A)
22        case (presentState)
23            start      : nextState <= prepare;
24            prepare    : if (A) nextState <= flash1;
25            flash1     : if (A) nextState <= flash2;
26                      : else nextState <= recovery;
27            flash2     : if (A) nextState <= flash1;
28                      : else nextState <= recovery;
29            recovery   : if (A) nextState <= prepare;
30                      : else nextState <= error;
31            error      : if (~A) nextState <= start;
32            default    : nextState <= presentState;
33        endcase
34
35 endmodule
```

Draw a proper state transition diagram that corresponds to the FSM described in this Verilog code.

(25 points should be given only if the diagram is absolutely correct)



- (c) [5 points] Is the FSM described by the previous Verilog code a Moore or a Mealy FSM? Why?

Moore, the output Z only depends on the state (*presentState*) and not on the input (A).