CA Lab: Homework 5

student: Dimitri Tabatadze

April 7, 2023

Task Description

Create the model of a 3-bit register with the signals: reset, set and load. Develop a testbench and simulate the design.

1. Draw the circuits of 3-bit register *10 points*
2. Create next state table for 3-bit register *10 points*
3. Write the equations for the next states *10 points*
4. Write the code of 3-bit register in Verilog *60 points*

Please, solve the problems (write the comments) *5 points*, take the clear screenshots and combine all your solutions in one pdf file, then upload in teams. *5 points*

Discussion

The task is to build a 3 bit parallel register which has 6 inputs:

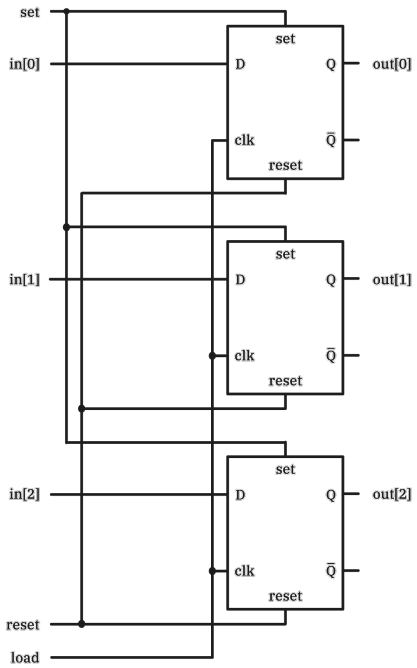
- 3 data,
- 1 set,
- 1 reset,
- 1 load

bits.

1. when the `reset` bit goes high, all bits of the register should be set to `0`.
2. whne the `set` bit goes high, all the bits of the register should be set to `1`.
3. when the `load` bit goes high, the bits of the register should be set to the respective bits of the `data` input.
4. if none of the above are set, the bits of the register should remain unchanged.

Solution

1. Here is the circuit diagram:



2.

set	reset	load	data	out	out$^+$
0	0	1	`xxx`	`yyy`	`xxx`
0	1	0	`xxx`	`yyy`	000
1	0	0	`xxx`	`yyy`	111

3. When the `set` signal goes high:

$$\text{out}^+[0] = 1$$
$$\text{out}^+[1] = 1$$
$$\text{out}^+[2] = 1$$

When the `reset` signal goes high:

$$\text{out}^+[0] = 0$$
$$\text{out}^+[1] = 0$$
$$\text{out}^+[2] = 0$$

When the `load` signal goes high:

$$\begin{aligned}\text{out}^+[0] &= in[0]\\ \text{out}^+[1] &= in[1]\\ \text{out}^+[2] &= in[2]\end{aligned}$$

When any of the signals goes low:

$$\begin{aligned}\text{out}^+[0] &= out[0]\\ \text{out}^+[1] &= out[1]\\ \text{out}^+[2] &= out[2]\end{aligned}$$

4. The Verilog code for a register described above:

```
module register (
    input reset,
    input load,
    input set,
    input [2:0] data,
    output reg [2:0] out = 0
);

always @(posedge load) begin
    out = data;
end

always @(posedge set) begin
    out = 3'b111;
end

always @(posedge reset) begin
    out = 3'b000;
end

endmodule
```

Conclusion

Although the given task was not so clear with what should have been done, I did what I interpreted the task as. I very much hope that this is correct.

Reference

- me