VLSI LAB Experiment-4

Harshavardhan Alimi

18EC10021

Design of a circuit that takes 8 bit binary input and checks divisibility of the aggregate binary input by 3.

Aim :-

The aim of this experiment is to design a circuit that takes 8 bit binary input and gives output “1” if the aggregate binary input is divisible by 3.

Schematic diagrams and labels :-

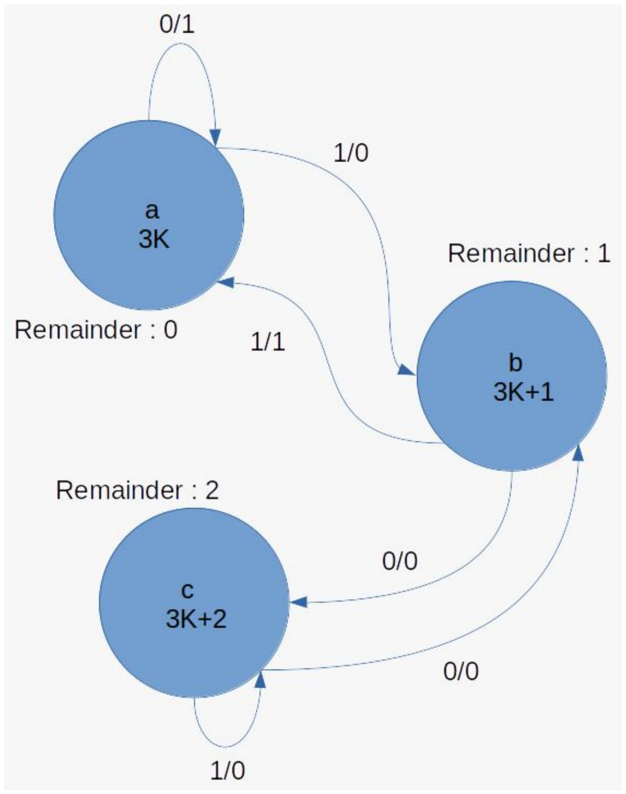


Fig.1:- State diagram for the above implementation using FSM

Transition table for FSM			
Prev Remainder	Input	Remainder	output
0(state:a)	0	0(state:a)	1
0(state:a)	1	1(state:b)	0
1(state:b)	0	2(state:c)	0
1(state:b)	1	0(state:a)	1
2(state:c)	0	1(state:b)	0
2(state:c)	1	2(state:c)	0

Fig.2 :-State table representation of the FSM

Description :-

- The model is implemented using the concept of a finite state machine.
- As we will be given 8 bit input we have to determine the 8-bit output sequentially i.eThe first bit(MSB) of output is determined by the first bit(MSB) of input and the second bit(next to MSB) of output is determined by first(MSB) 2 bits of input and similarly the 8-bit output is determined
- For example,if input is 110 then output bits will be(from MSB)
  - 1st bit: from input only 1 bit=1=1,not divisible 3 so out[2]=0
  - 2nd bit: from input 1st 2 bits=11=3,divisible by 3 so out[1]=1
  - 3rd bit: from input 1st 3 bits=110=6,divisible by 3 so out[0]=1
- Using the concept of FSM, we can build this by implementing 3 states to look over divisibility by 3,because any integer can be displayed in any one of the form 3K or 3K+1 or 3K+2(i.e.,when divided by 3 there can be 0 or 1 or 2 as remainders only).
- The 3 states are :
  - State a: initial state or state having remainder 0(i.e., the number will be having the form of 3K).
  - State b: state having remainder 1(i.e., the number will be having the form of 3K+1).
  - State c: state having remainder 2(i.e., the number will be having the form of 3K+2).
- If ‘0’ is the next bit in the sequence this implies multiplication by 2(of the previous sequence) in its decimal equivalent. Similarly if ‘1’ is the next bit in the

sequence this implies multiplication by 2(of the previous sequence) and then addition with 1 in its binary form.

- **If ‘0’ is the next bit in sequence :-**
  - If the number is in state a (3k form), on placing a 0 beside it in binary form it will change to 6K form which is still divisible by 3. Therefore it will remain at the same state a and output should be 1.
  - If the number is in state b (3k +1 form), on placing a 0 beside it in binary form it will change to 6K +2 form whose remainder is 2. Therefore it will go to state c and output should be 0.
  - If the number is in state c (3k+2 form) on placing a 0 beside it in binary form it will change to 6K + 4 = 3k’ + 1 form whose remainder is 1 and it will go to state b and output should be 0.
- **If ‘1’ is the next bit in sequence :-**
  - If the number is in state a (3K form), on placing a 1 beside it in binary form it will change to 6K +1 form whose remainder is 1 and it will go to state B and output should be 0.
  - If the number is in state b (3k +1 form), on placing a 1 beside it in binary form it will change to 6K + 3 = 3K’ form which is still divisible by 3. Therefore it will go to state a and output should be 1.
  - If the number is in state c (3k+2 form) on placing a 1 beside it in binary form it will change to 6K + 5 = 3k’ + 2 form whose remainder is 2. Therefore it remains at the state c and output should be 0.

Verilog codes :-

```
`define a 3'd0   // remainder is 0

`define b 3'd1   // remainder is 1

`define c 3'd2   // remainder i 2


module div_by_3_fsm(in,out);

   input [7:0] in;

   output reg [7:0] out;


   integer i;
```

```
reg [2:0] state,nstate;  //state and next state


always @(in)
begin
    state=`a;  // intial state
    for(i=7;i>=0;i=i-1)  // looping to assign for all seq output
    begin
        case(state)
            `a: if(in[i])
                    begin
                        nstate = `b;  // next state
                        out[i] = 1'b0;  // ouput single bit
                    end
                else
                    begin
                        nstate = `a;
                        out[i] = 1'b1;
                    end
            `b: if(in[i])
                    begin
                        nstate = `a;
                        out[i] = 1'b1;
                    end
                else
                    begin
                        nstate = `c;
                        out[i] = 1'b0;
```

```
                       end
               `c: if(in[i])
                       begin
                           nstate = `c;
                           out[i] = 1'b0;
                       end
                   else
                       begin
                           nstate = `b;
                           out[i] = 1'b0;
                       end
           endcase
           state=nstate;
       end
   end
endmodule

module div_by_3_fsm_tb();
   // input
   reg [7:0] in;
   // output
   wire [7:0] out;
   // instantion
   div_by_3_fsm uut(in,out);

   initial
   begin
```

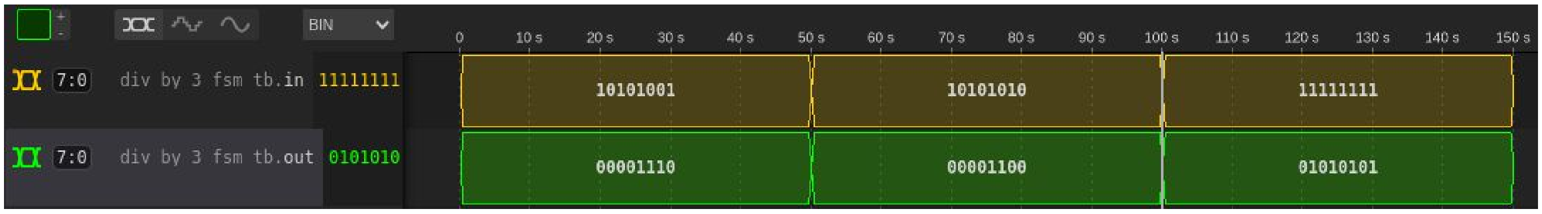
```
        $monitor("time=%5d, in = %b, out = %b",$time,in,out);
        in = 8'b10101001;
        #50 in=8'b10101010;
        #50 in=8'b11111111;
        #50 $finish;
    end

    initial begin
        $dumpfile("test.vcd");
        $dumpvars;
    end
endmodule
```

Runtime log :-

```
VCD info: dumpfile test.vcd opened for output.
time=   0, in = 10101001, out = 00001110
time=   50, in = 10101010, out = 00001100
time=  100, in = 11111111, out = 01010101
```

Results :-



Discussion/Conclusion :-

- Divisibility by three is checked for the given binary sequence.
- Conditional statements have been used to check to assign the next state and output based on previous state and input bit sequentially.
- For loop has been used to store the output as each bit is attached to the input.
- The output is plotted and displayed using GTKWAVE.

Drive links for verilog codes :-

CODE