VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

ASIC DESIGN LAB – MVLD505P

SLOT: L33+L34

DIGITAL ASSIGNMENT -1

ELECTRONIC DICE GAME

SUBMITTED BY:

VIVEK VALLAMPATLA – 24MVD0034

POORNA SAI PRASAD – 24MVD0049

GOGULA SIVA SANKAR – 24MVD0035

AIM:

To design, implement, and verify the architecture of the given specification for a dice-based game module using Verilog and validate the functionality through simulation.

TOOLS USED:

- ModelSim
- Intel Quartus prime

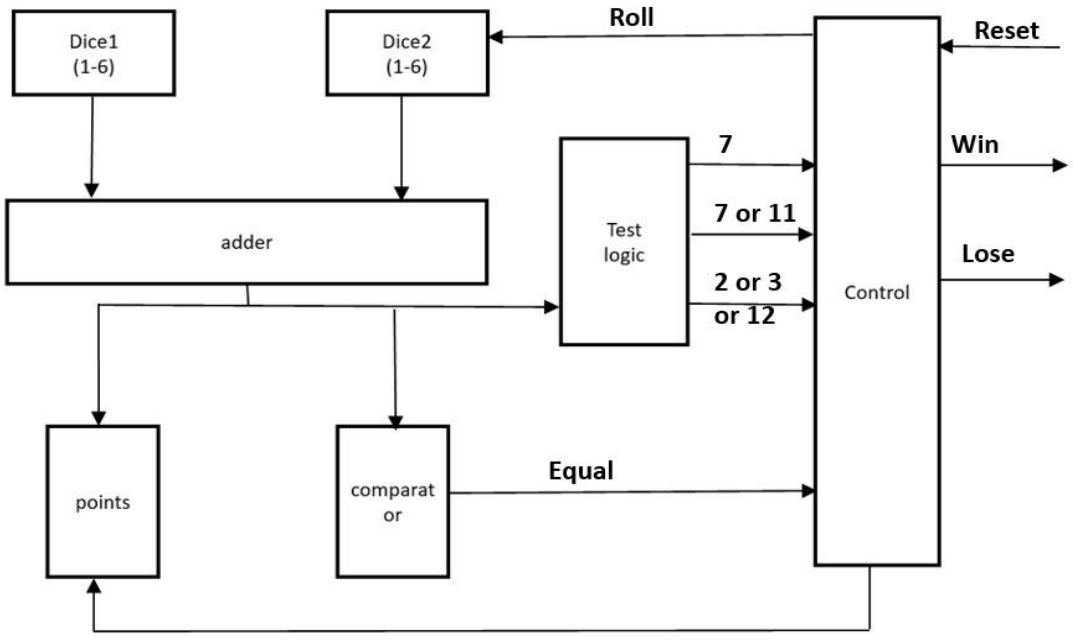
ALGORITHM:

State Transition Logic for dice_game:

1. **Initial State (s0):**
    - Reset win, lose, and point signals to 0.
    - Wait for the roll signal.
    - If roll is high, compute the sum of dice1 and dice2.
    - Transition to state s1.
2. **Win/Lose Check State (s1):**
    - If sum = 7 or sum = 11, set win to 1 and transition to s0.
    - If sum = 2, 3, or 12, set lose to 1 and transition to s0.
    - Otherwise, set point to sum and transition to s2.
3. **Point Check State (s2):**
    - Wait for the next roll signal.
    - If sum = point, set win to 1 and transition to s0.
    - If sum = 7, set lose to 1 and transition to s0.

BLOCK LEVEL - ARCHITECTURE:

- **Inputs:**
  - clock (Clock signal)
  - rst (Reset signal)
  - roll (Trigger to roll the dice)
  - dice1, dice2 (4-bit inputs representing dice values)
- **Outputs:**
  - win (Indicates a win)
  - lose (Indicates a loss)
  - point (Stores the point value for subsequent rolls)



1. **Dice1 (1-6) and Dice2 (1-6)**

- These modules represent the two dice in the game.
- Each dice generates a random number between 1 and 6 when rolled.

2. **Adder**

- This module takes the outputs of Dice1 and Dice2 as inputs.
- It adds the two numbers to calculate the total score from the dice roll.

3. **Test Logic**

- This module evaluates the total score produced by the Adder.
- Based on the rules:
    - If the sum is 7 or 11, it marks a Win condition.
    - If the sum is 2, 3, or 12, it marks a Lose condition.
- Other sums are passed on to determine further game states.

4. **Points**

- This module stores the total score from the first roll if it doesn't result in an immediate win or loss.
- The stored score is referred to as the Point.

5. **Comparator**

- This module compares the sum of subsequent rolls with the stored Point.
- If the new roll equals the Point, it results in a Win.
- If the roll results in a 7, it results in a Lose.

6. **Control**

- This is the main decision-making module that evaluates the results from the Test Logic and Comparator.
- It decides whether the player Wins, Loses, or continues playing.
- It also controls the flow of the game by resetting or allowing further rolls.

7. **Reset**

- This module resets the game to its initial state for a new round of play.

VERILOG CODE:

```
module dice_game(clock, rst, roll, dice1, dice2, win, lose, point);
input clock, rst, roll;
input [3:0] dice1, dice2;
output reg win, lose;
output reg [3:0] point;
parameter s0 = 2'b00, s1 = 2'b01, s2 = 2'b10;
reg [3:0] sum;
reg [1:0] ps, ns;

always @(posedge clock)
begin
if (rst)
ps <= s0;
else
ps <= ns;
end
always @(ps, roll, dice1, dice2, sum, point)
begin
win = 0;
lose = 0;
ns = ps;
case (ps)
s0: begin
win = 0;
lose = 0;
point = 0;
if (roll)
begin
```

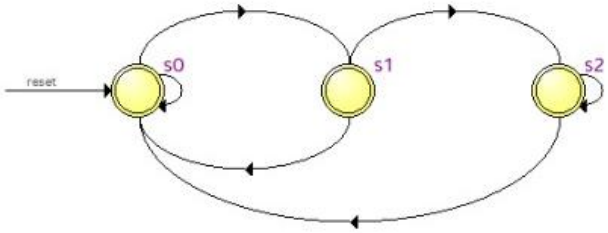
```
sum = dice1 + dice2;
ns = s1;
end
end
s1: begin
if (sum == 4'b0111 || sum == 4'b1011)
begin
win = 1;
ns = s0;
end
else if (sum == 4'b0010 || sum == 4'b0011 || sum == 4'b1100)
begin
lose = 1;
ns = s0;
end
else
begin
point = sum;
ns = s2;
end
end
s2: begin
if (roll)
begin
sum = dice1 + dice2;
if (sum == point)
begin
win = 1;
ns = s0;
end
```

```
else if (sum == 4'b0111)
begin
lose = 1;
ns = s0;
end
end
end
endcase
end
endmodule
```

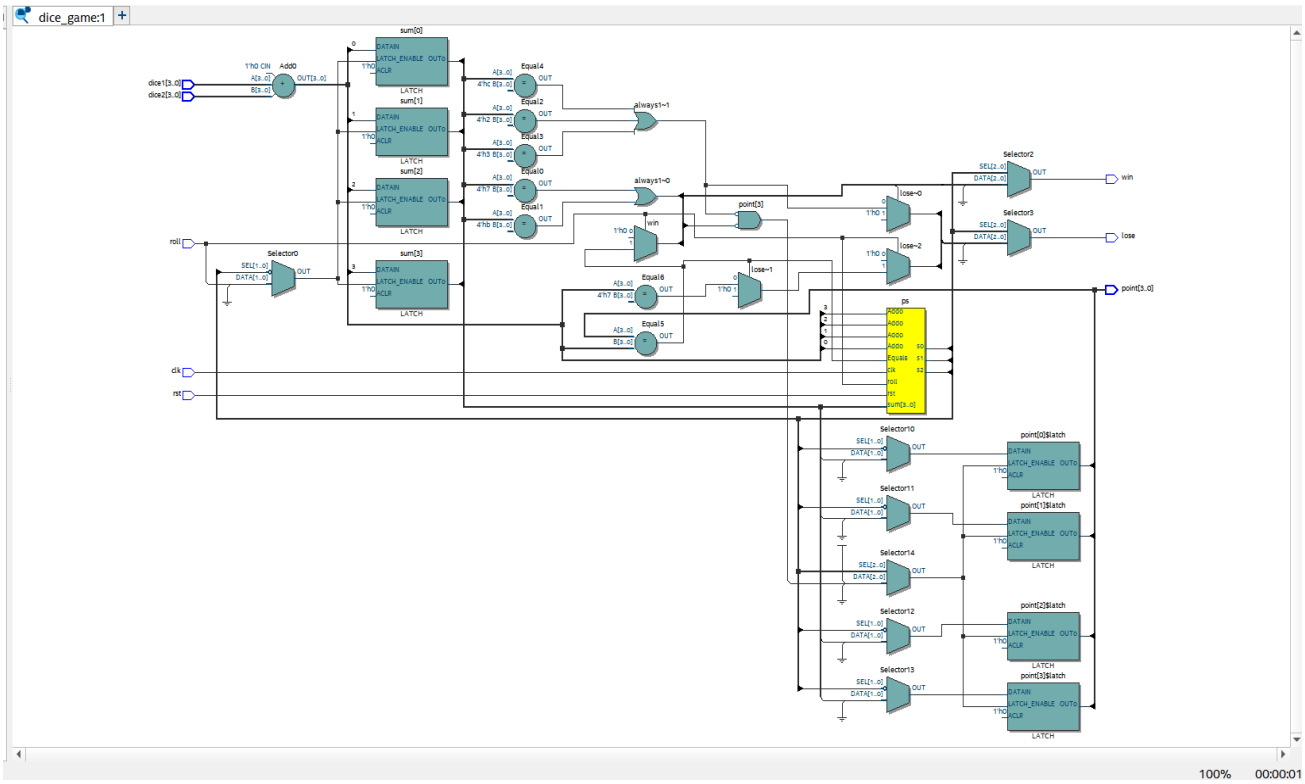
TESTBENCH:

```
module dg_tb();

reg clock, rst, roll;

reg [3:0] dice1, dice2;

wire win, lose;

wire [3:0] point;

dice_game d1
(.clock(clock),.rst(rst),.roll(roll),.dice1(dice1),.dice2(dice2),.win(win),.lose(lose),.point(point));

initial

begin

clock=1'b0;

rst=1'b1;

dice1 = 0;

dice2 = 0;

#10 rst = 0;

$monitor ($time, "the values are
clock=%b,rst=%b,roll=%b,dice1=%d,dice2=%d,win=%b,lose=%b,point=%b",clock,rst,roll,dice1,dic
e2,win,lose,point);

end

always #5 clock = ~clock;

initial

begin

#10 roll = 1; dice1 = 4; dice2 = 3;

#10 roll = 1; dice1 = 1; dice2 = 1;

#10 roll = 1; dice1 = 3; dice2 = 3;

#10 roll = 1; dice1 = 3; dice2 = 3;

#10 roll = 1; dice1 = 5; dice2 = 4;

#10 roll = 1; dice1 = 3; dice2 = 4;

#10 roll = 1; dice1 = 6; dice2 = 5;

end

endmodule
```

FSM:



ARCHITECTURAL VIEW:

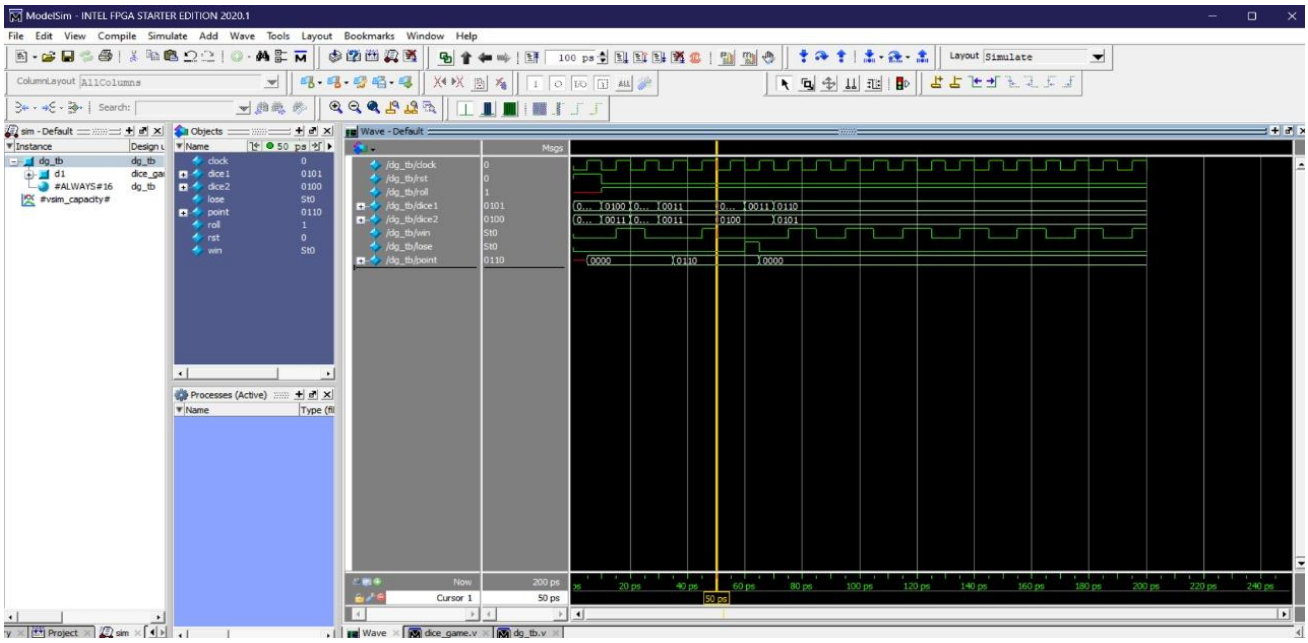


RESULTS:

TRANSCRIPT:

```
VSIM 4> run
#               10the values are clock=0,rst=0,roll=1,dice1= 4,dice2= 3,win=0,lose=0,point=0000
#               15the values are clock=1,rst=0,roll=1,dice1= 4,dice2= 3,win=1,lose=0,point=0000
#               20the values are clock=0,rst=0,roll=1,dice1= 1,dice2= 1,win=1,lose=0,point=0000
#               25the values are clock=1,rst=0,roll=1,dice1= 1,dice2= 1,win=0,lose=0,point=0000
#               30the values are clock=0,rst=0,roll=1,dice1= 3,dice2= 3,win=0,lose=0,point=0000
#               35the values are clock=1,rst=0,roll=1,dice1= 3,dice2= 3,win=0,lose=0,point=0110
#               40the values are clock=0,rst=0,roll=1,dice1= 3,dice2= 3,win=0,lose=0,point=0110
#               45the values are clock=1,rst=0,roll=1,dice1= 3,dice2= 3,win=1,lose=0,point=0110
#               50the values are clock=0,rst=0,roll=1,dice1= 5,dice2= 4,win=0,lose=0,point=0110
#               55the values are clock=1,rst=0,roll=1,dice1= 5,dice2= 4,win=0,lose=0,point=0110
#               60the values are clock=0,rst=0,roll=1,dice1= 3,dice2= 4,win=0,lose=1,point=0110
#               65the values are clock=1,rst=0,roll=1,dice1= 3,dice2= 4,win=0,lose=0,point=0000
#               70the values are clock=0,rst=0,roll=1,dice1= 6,dice2= 5,win=0,lose=0,point=0000
#               75the values are clock=1,rst=0,roll=1,dice1= 6,dice2= 5,win=1,lose=0,point=0000
#               80the values are clock=0,rst=0,roll=1,dice1= 6,dice2= 5,win=1,lose=0,point=0000
#               85the values are clock=1,rst=0,roll=1,dice1= 6,dice2= 5,win=0,lose=0,point=0000
#               90the values are clock=0,rst=0,roll=1,dice1= 6,dice2= 5,win=0,lose=0,point=0000
#               95the values are clock=1,rst=0,roll=1,dice1= 6,dice2= 5,win=1,lose=0,point=0000
```

WAVEFORM:



INFERENCE:

- The functionality of the dice_game module has been verified to meet the given specification.
- The module transitions correctly between states (s0, s1, s2) based on the inputs.
- The simulation results matched the expected behaviour.