Report HW 3 ECE-111

Name: Jasper Huang

PID: A17796149

Homework 3a (Johnson Counter)

1. **SystemVerilog design code**
    a. **johnson_counter**

```
module johnson_counter (
  input logic clk, clear, preset,
  input logic[3:0] load_cnt,
  output logic[3:0] count
);
  always @(posedge clk or negedge clear) begin
    // Student to add code for Johnson Counter
    if (!clear) begin
      count = 4'b0000;
    end else if (!preset) begin
        count = load_cnt;
    end else begin
      count = {~count[0], count[3:1]};
    end
  end
endmodule: johnson_counter
```

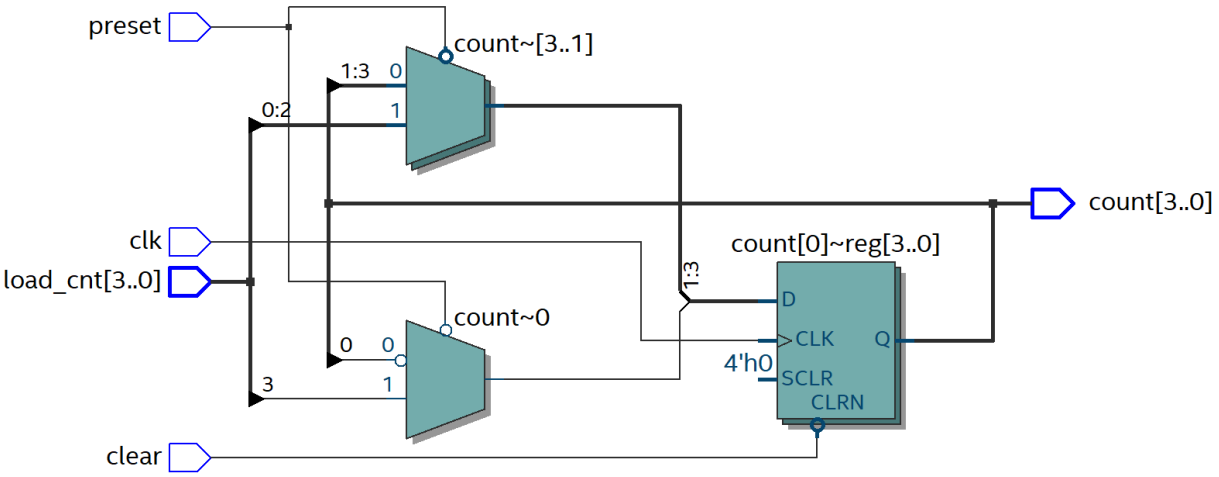
2. **Synthesis Resource usage and schematic generated from RTL netlist viewer**
   a. **Resource Usage**

Analysis & Synthesis Resource Usage Summary

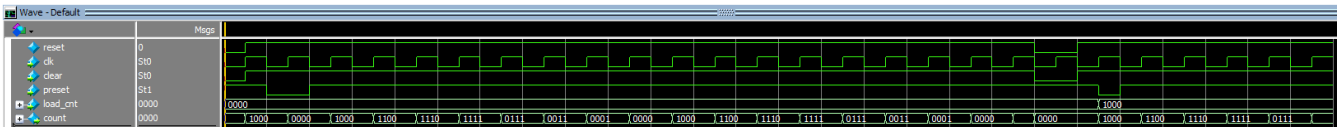
<<Filter>>

	Resource	Usage
1	Estimate of Logic utilization (ALMs needed)	2
2		
3	Combinational ALUT usage for logic	4
1	-- 7 input functions	0
2	-- 6 input functions	0
3	-- 5 input functions	0
4	-- 4 input functions	0
5	-- <=3 input functions	4
4		
5	Dedicated logic registers	4
6		
7	I/O pins	11
8		
9	Total DSP Blocks	0
10		
11	Maximum fan-out node	preset~input
12	Maximum fan-out	4
13	Total fan-out	39
14	Average fan-out	1.30

   b. **RTL netlist viewer**

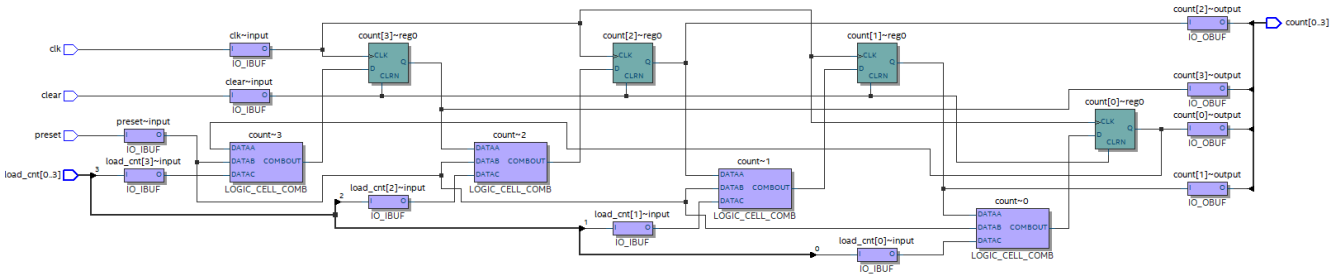


3. **Simulation snapshot and explain simulation result**



When reset/clear flag is low, the count is set to 0000, and when preset is low, the count is set to load_cnt which is 0000 in the first low preset flag. After that, when the count and preset are both high, the count change from 0000 to 1000 to 1100 to 1110 to 1111 to… back to 0000, which is the expected behavior of johnson counter. When the second reset and preset flag lowered, both flag perform the same as previous, with preset to 1000.

4. Post-Mapping schematic



Homework 3b (Universal Shift Register)

1. **SystemVerilog code snapshot**
    a. **universal_shift_register (student fill part)**

```
// student to fill code for SIPO-L, SIPO-R, PISO-L, PISO-R, SISO-L, SISO-R operation mode of shift register
// Note : Use non-blocking assignment statements within case item expressions
3'b001 : begin
  shift_reg <= {shift_reg[2:0], sin};
  sout <= 1'b0;
end
3'b010 : begin
  shift_reg <= {sin, shift_reg[3:1]};
  sout <= 1'b0;
end
3'b011 : begin
  shift_reg <= {shift_reg[2:0], 1'b0};
  sout <= shift_reg[3];
end
3'b100 : begin
  shift_reg <= {1'b0, shift_reg[3:1]};
  sout <= shift_reg[0];
end
3'b101: begin
  shift_reg <= {shift_reg[2:0], sin};
  sout <= shift_reg[3];
end
3'b110: begin
  shift_reg <= {sin, shift_reg[3:1]};
  sout <= shift_reg[0];
end
default : begin
  shift_reg <= 4'b0000;
  sout <= 1'b0;
end
```

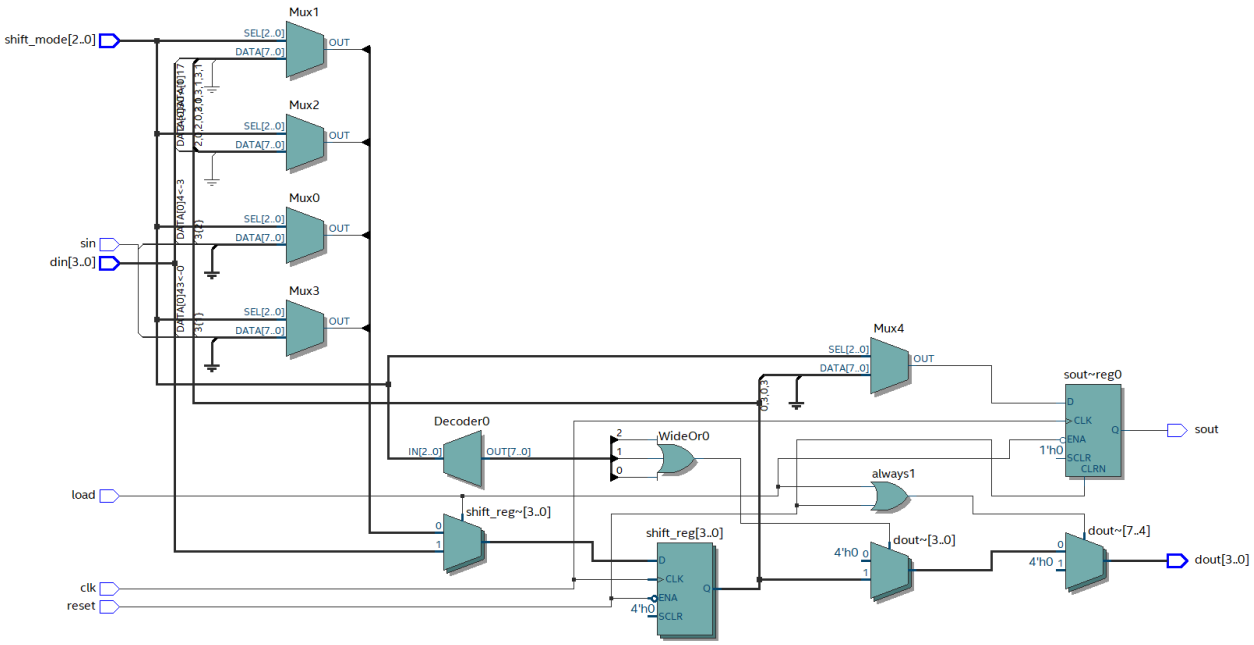
2. **Synthesis Resource usage and schematic generated from RTL netlist viewer**
   a. **Synthesis Resource usage**

Analysis & Synthesis Resource Usage Summary

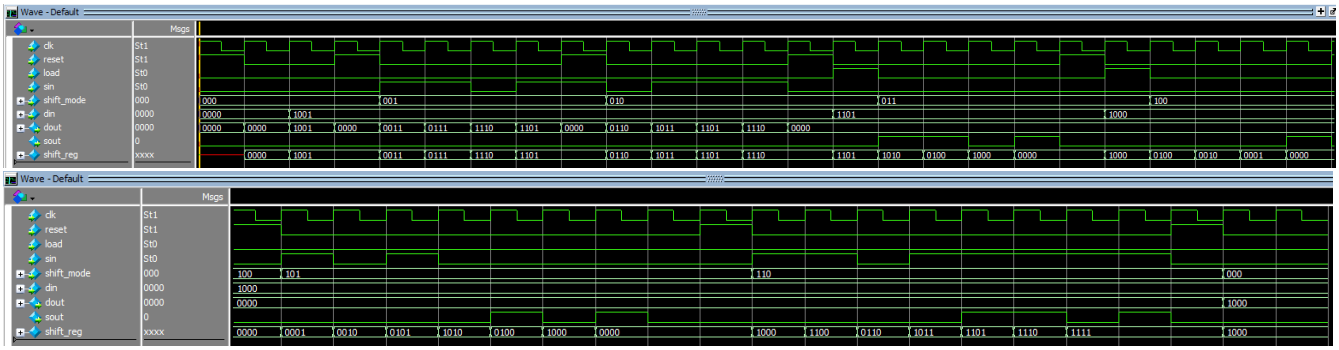
<<Filter>>

	Resource	Usage
1	Estimate of Logic utilization (ALMs needed)	9
2		
3	Combinational ALUT usage for logic	11
1	-- 7 input functions	3
2	-- 6 input functions	4
3	-- 5 input functions	2
4	-- 4 input functions	1
5	-- <=3 input functions	1
4		
5	Dedicated logic registers	5
6		
7	I/O pins	16
8		
9	Total DSP Blocks	0
10		
11	Maximum fan-out node	reset~input
12	Maximum fan-out	9
13	Total fan-out	97
14	Average fan-out	2.02

   b. **RTL netlist viewer**



3. **Simulation snapshot and explain simulation result**



For different shift modes, the operations perform as expected, with each of the following:

- **000: PIPO, which gives a dout 1001, output of din = 1001**
- **001: SIPO-L, which gives a dout of 1101, output of sin = 1,1,0,1 from left to right**
- **010: SIPO-R, which gives dout of 1110, output of sin = 0,1,1,1 from right to left**
- **011: PISO-L, which gives a sout of 1,1,0,1, output of din = 1101 from left to right**
- **100: PISO-R, which gives a sout of 0,0,0,1, output of din = 1000 from right to left**
- **101: SISO-L, which gives a sout of 1,0,1,0, output of sin = 1,0,1,0 with a different order in shift reg**
- **110: SISO-R, which gives a sout of 1,1,0,1, output of sin = 1,1,0,1 with a different order in shift reg**

All waveform matches the expected calculation (including serial and parallel out) for each operation.

4. Post-Mapping schematic