106010006 黃詩瑜 電機 21

Pre-Lab 3: Counters and Shift Registers I

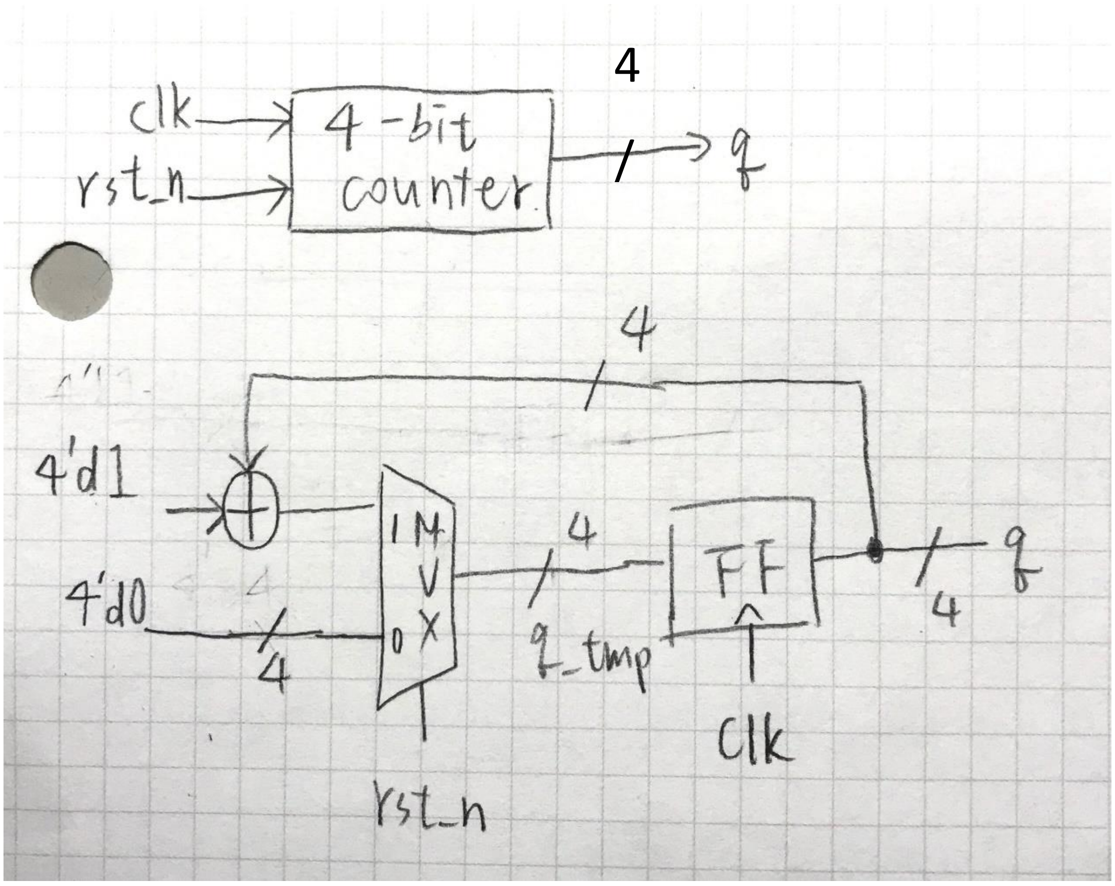
1. Consider a 4-bit synchronous binary up counter.

Design Specification:

Input: rst_n, clk

Output: [3:0]q

1.1 Draw the logic diagram



1.2 Construct Verilog RTL representation for the logics with verification.

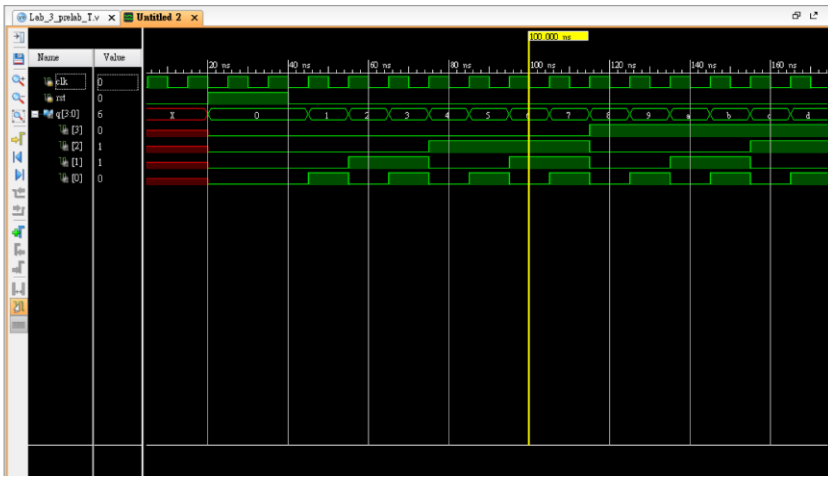
Module code:

```
`define BCD_BIT_WIDTH 4
`define BCD_ZERO 4'd0
`define BCD_ONE 4'd1
```

```
module Lab_3_prelab_M(
	q, // output
	clk, // global clock
	rst // active high reset
);
output [`BCD_BIT_WIDTH-1:0] q; // output
input clk; // global clock
input rst; // active high reset
reg [`BCD_BIT_WIDTH-1:0] q; // output (in always block)
reg [`BCD_BIT_WIDTH-1:0] q_tmp; // input to dff (in always block)
// Combinational logics
always @(q)
	q_tmp = q + `BCD_ONE;
// Sequential logics: Flip flops
always @(posedge clk or posedge rst)
	if (rst) q<=`BCD_BIT_WIDTH'd0;
	else q<=q_tmp;

endmodule
```

波形圖:

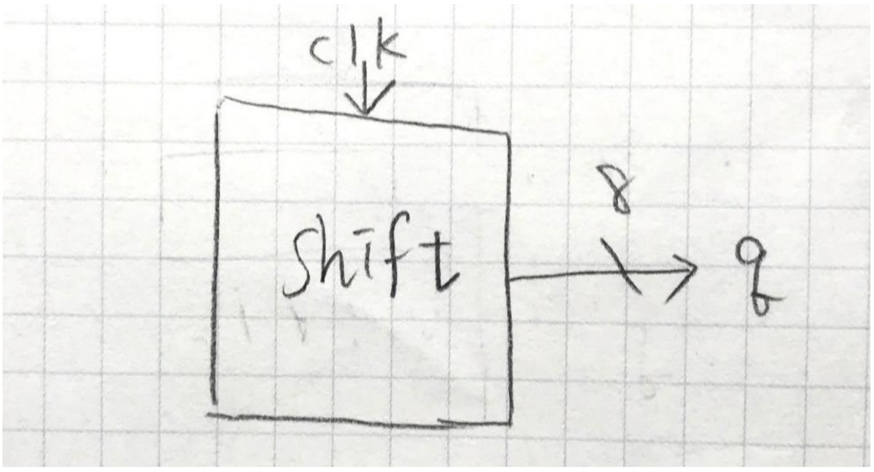


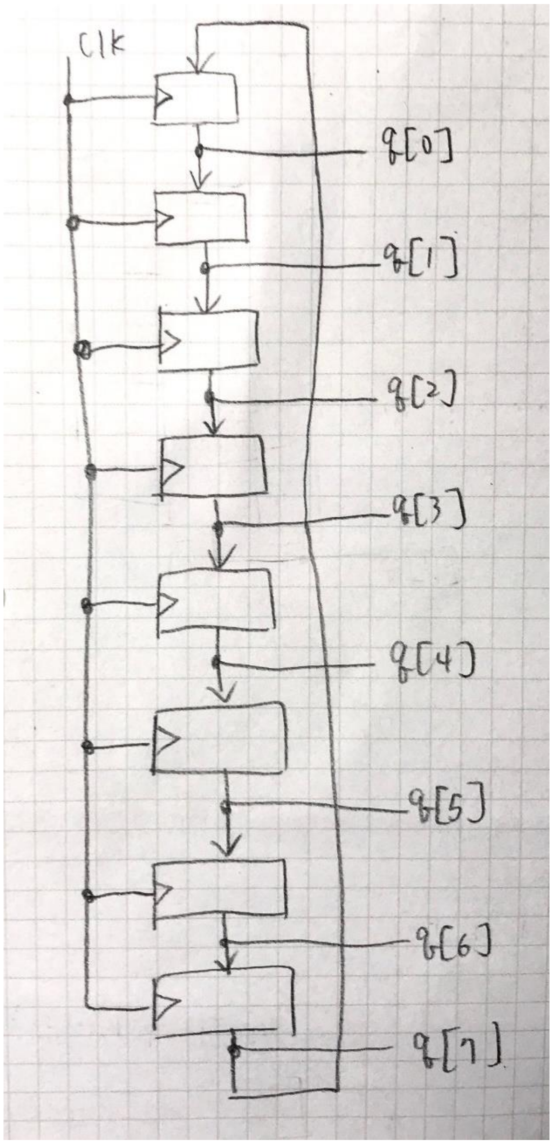
2. Cascade eight DFFs together as a shift register. Connect the output of the last DFF to the input of the first DFF as a ringer counter. Let the initial value of DFF output after reset be 01010101. Construct the Verilog RTL representation for the logics with verification.

Design Specification:

Input: clk , rst_n

Output: [7:0]q

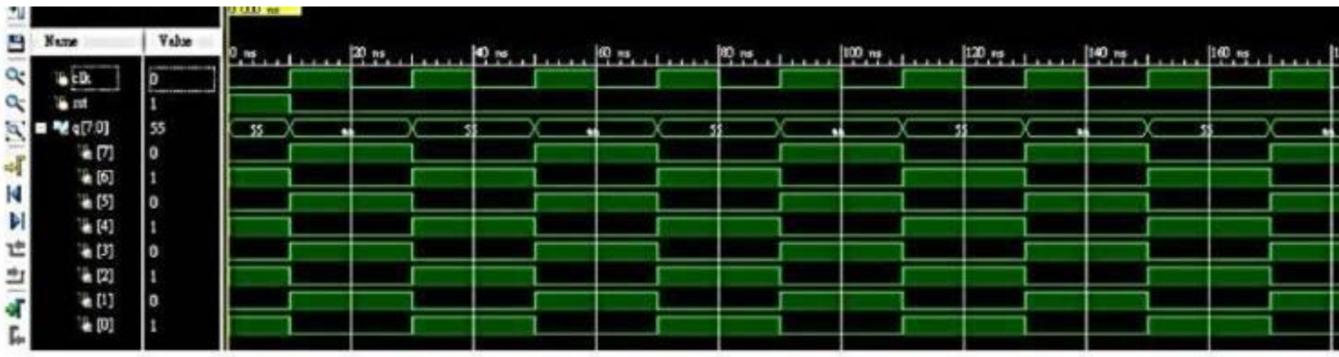




Module code:

```
module prelab3_2(
q, // shifter output
clk, // global clock
rst_n // active low reset
);
output [7:0] q; // output
input clk; // global clock
input rst_n; // active low reset
reg [7:0] q; // output
// Sequential logics: Flip flops
always @(posedge clk or negedge rst_n)
if (~rst_n)
begin
q<=8'b01010101;
end
else
begin
q[0]<=q[7];
q[1]<=q[0];
q[2]<=q[1];
q[3]<=q[2];
q[4]<=q[3];
q[5]<=q[4];
q[6]<=q[5];
q[7]<=q[6];
end
endmodule
```

波形圖:


Name
Value
clk 0
rst 1
q[7:0] 55
[7] 0
[6] 1
[5] 0
[4] 1
[3] 0
[2] 1
[1] 0
[0] 1
0 ns
20 ns
40 ns
60 ns
80 ns
100 ns
120 ns
140 ns
160 ns
55
aa
55
aa
55
aa
55
aa
55
aa