Digital Logic

Lab14 Counters & review

Lab14 part1

- Counter
  - Binary counter
  - Johnson counter

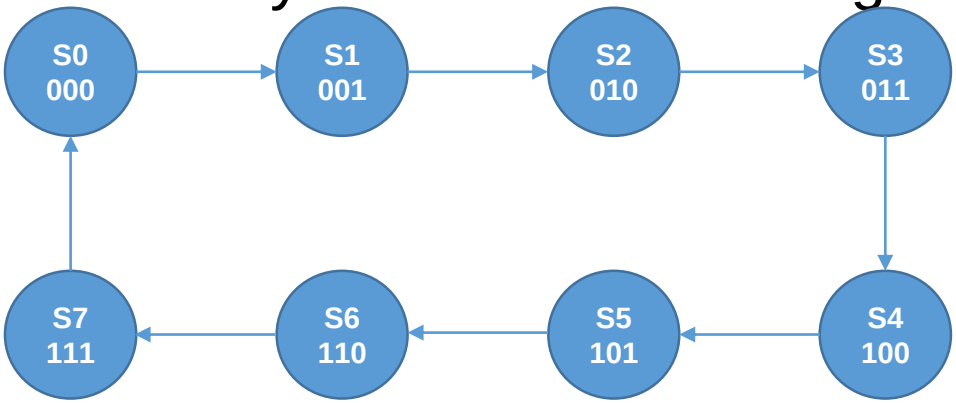
baiyh@sustech.edu.cn

Counters

- A counter is essentially a register that goes through a predetermined sequence of binary states. The gates in the counter are connected in such a way as to produce the prescribed sequence of states. Although counters are a special type of register, it is common to differentiate them by giving them a different name.

Binary Counter (1)

- Synchronous binary counters have a regular pattern

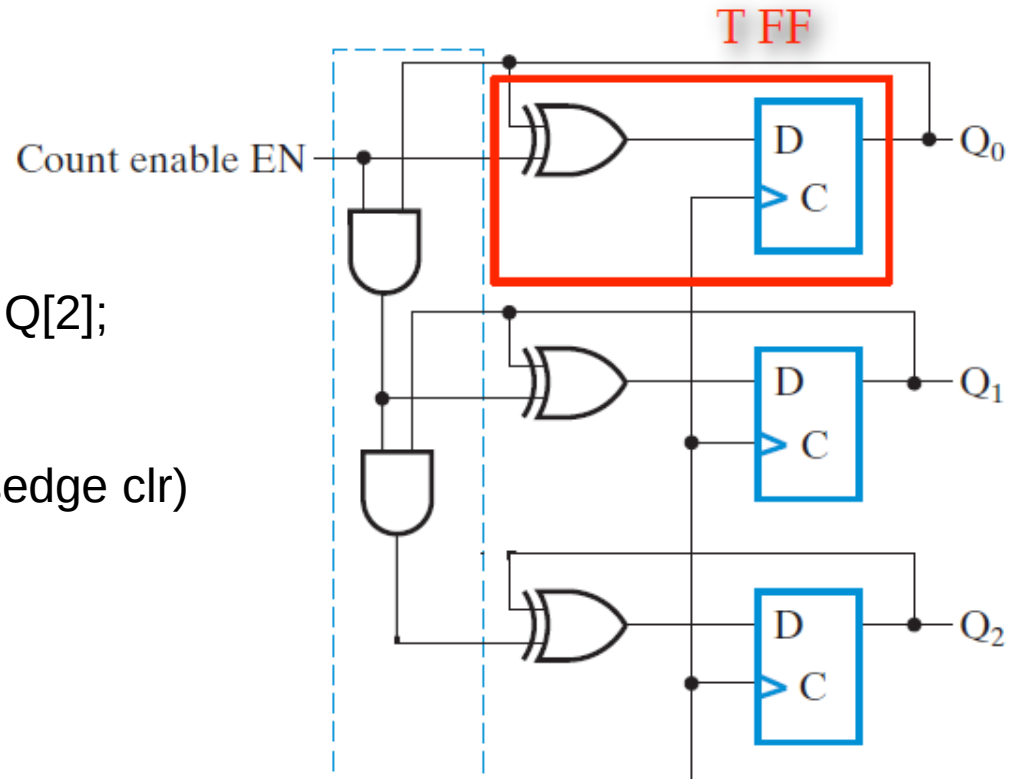


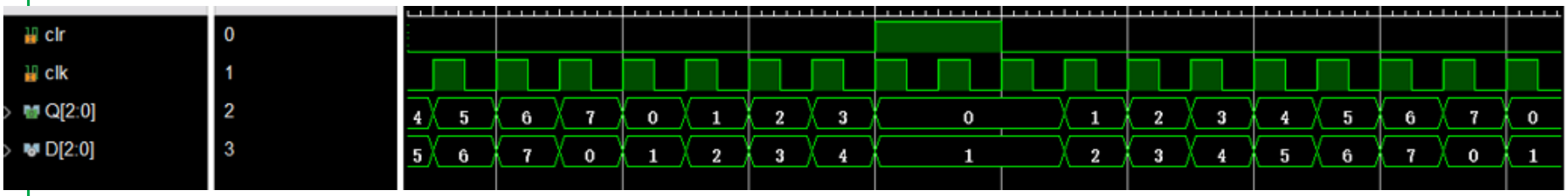
	Present State			Next State		
	Q2	Q1	Q0	D2	D1	D0
s0	0	0	0	0	0	1
s1	0	0	1	0	1	0
s2	0	1	0	0	1	1
s3	0	1	1	1	0	0
s4	1	0	0	1	0	1
s5	1	0	1	1	1	0
s6	1	1	0	1	1	1
s7	1	1	1	0	0	0

Binary Counter (2)

- Design a 3-bits binary counter using algebraic method

```
module count3a(
input wire clr,
input wire clk,
output reg[2:0]Q );
  wire [2:0] D;
  assign D[2] = (Q[1]&Q[0]) ^ Q[2];
  assign D[1] = Q[1]^Q[0];
  assign D[0] = ~Q[0];
always@(posedge clk or posedge clr)
   if(clr == 1)
    Q<=0;
   else
    Q<=D;
endmodule
```

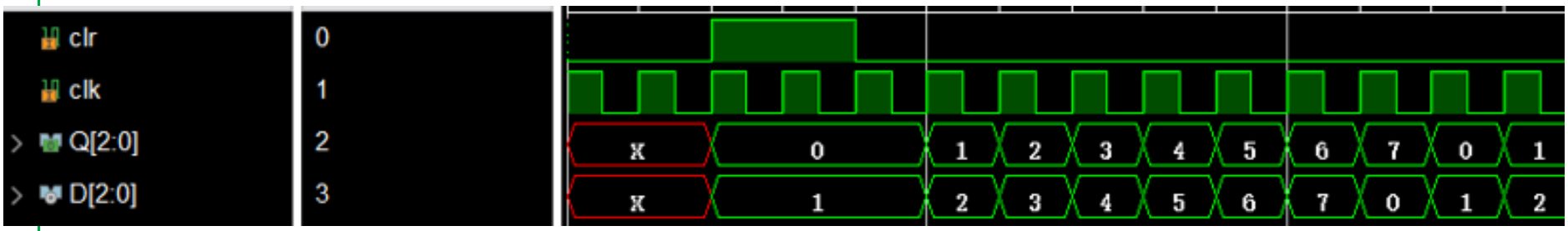




Binary Counter (3)

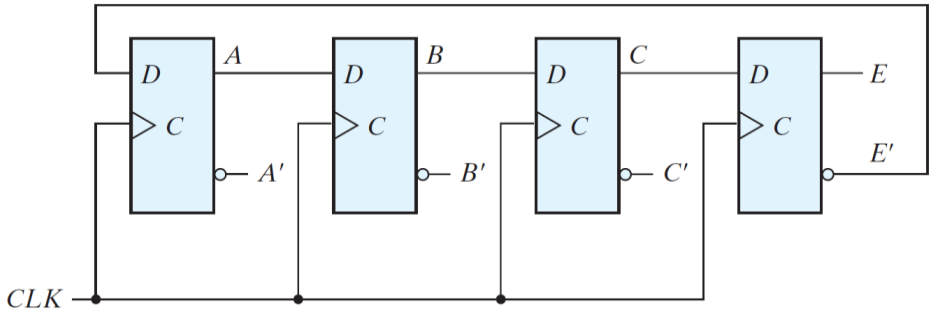
- Design a 3-bits binary counter using arithmetic method

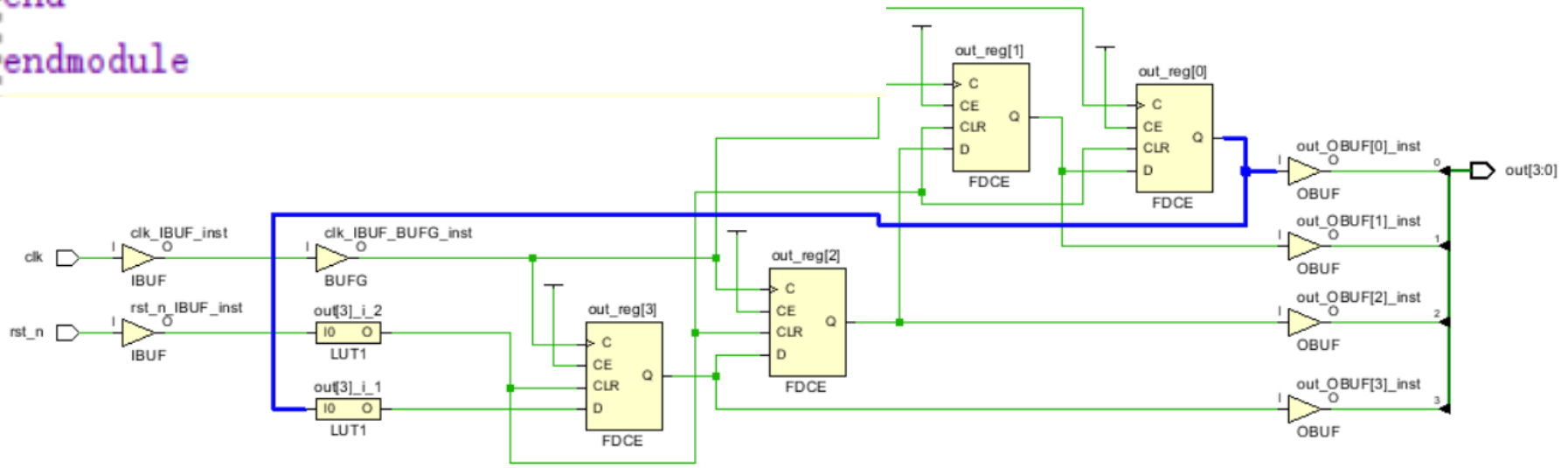
```
module count3b(
input wire clr,
input wire clk,
output reg[2:0] Q
    );
always@(posedge clk or posedge clr)
    begin
     if(clr == 1)
       Q<=0;
     else
       Q<=Q+1;
     end
endmodule
```



Johnson-counter(1)

```
module johoson_counter(
input clk,rst_n,output reg [3:0] out);
always @(posedge clk,negedge rst_n) begin
    if(~rst_n)
        out<=4'b0;
    else
        out<={~out[0],out[3:1]};
end
endmodule
```





Johnson-counter(2)

```
module johoson_counter(
input clk,rst_n,output reg [3:0] out);
always @(posedge clk,negedge rst_n) begin
    if(~rst_n)
        out<=4'b0;
    else
        out<={~out[0],out[3:1]};
end
endmodule
```

```
module johnsonCounterTb( );
reg clk,rst_n;
wire [3:0] out;
johoson_counter jc1(clk,rst_n,out);
initial begin
    clk = 1'b0;
    rst_n = 1'b0;
    #3 rst_n = 1'b1;
    forever #5 clk=~clk;
    #160 $finish;
end
endmodule
```