EES4725
Digital Circuits and FPGA Design

Chua Dingjuan
elechuad@nus.edu.sg

Lecture 5

SEQUENTIAL LOGIC - COUNTERS AND STATE DIAGRAMS

VERILOG FOR SYNCHRONOUS COUNTERS

Quiz

Date : Monday 29 December

Time : 1400 - 1530

Venue : Room 322

Format : Paper-Pen

MCQ/MRQ/Open-ended Questions

Open-Book (hardcopy materials only)

Synchronous (Parallel) Counters

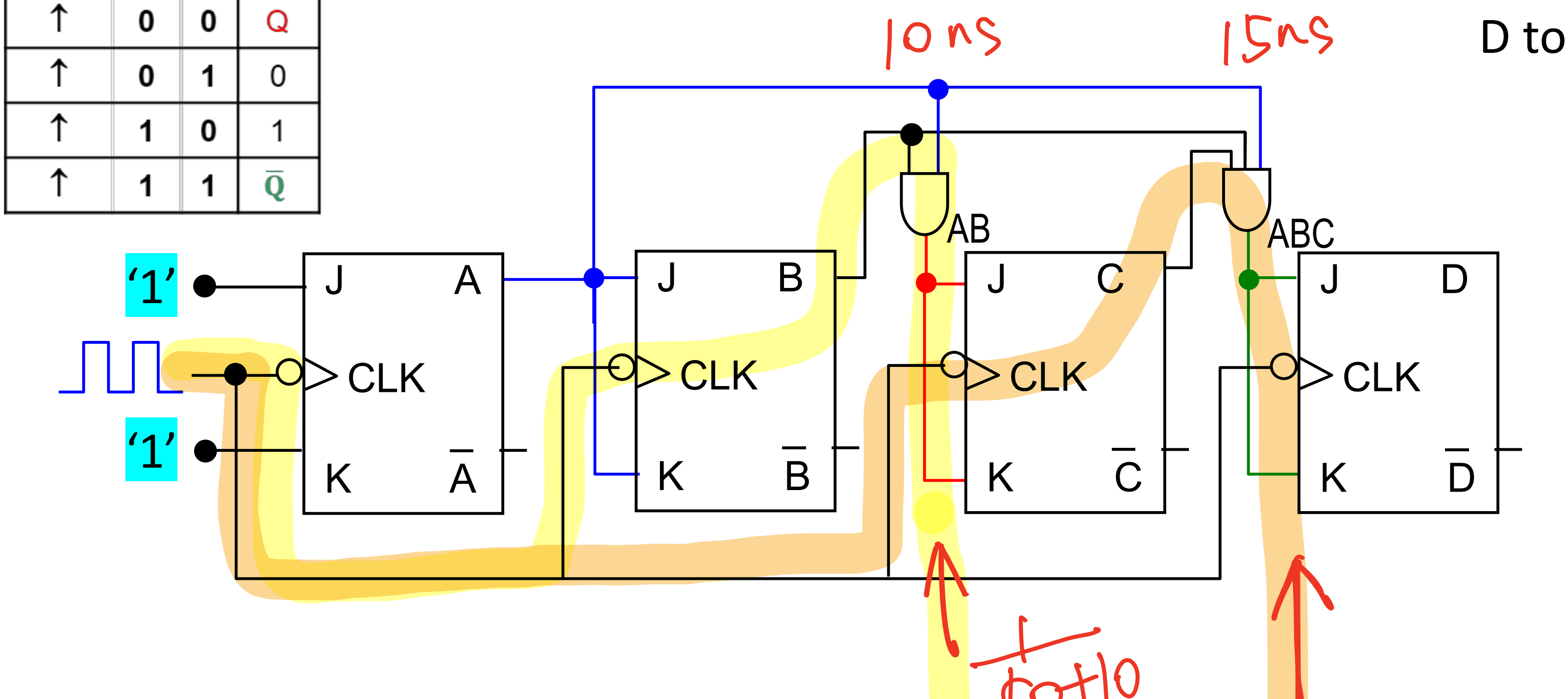
Mod-16 Synchronous Parallel Counter :

CLK	J	K	Q+
↑	0	0	Q
↑	0	1	0
↑	1	0	1
↑	1	1	$\bar{Q}$

B toggles when A = 1

C toggles when AB = 1

D toggles when ABC = 1



Clock	D	C	B	A
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10	1	0	1	0
11	1	0	1	1
12	1	1	0	0
13	1	1	0	1
14	1	1	1	0
15	1	1	1	1
0	0	0	0	
	c	o	n	t

- **Synchronous Counters:** all circuit elements get the clock input simultaneously

50ns

50+15ns

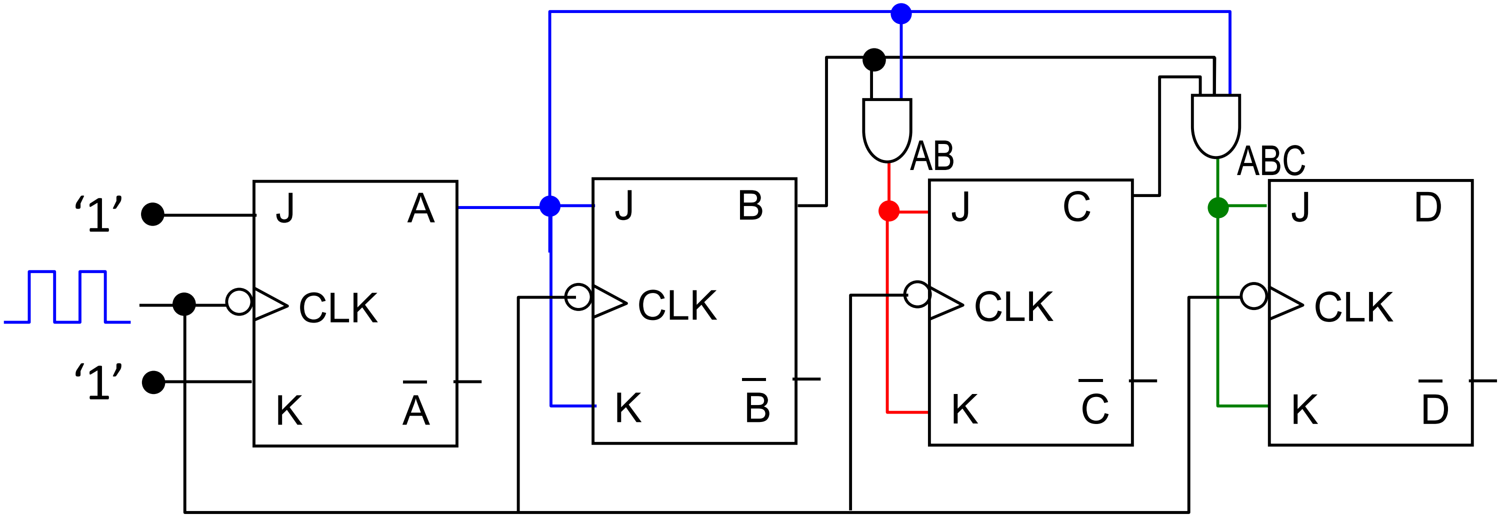
- Which is the critical path?

- Critical Path Delay = Δtpd (FF) + Δtpd (AND)

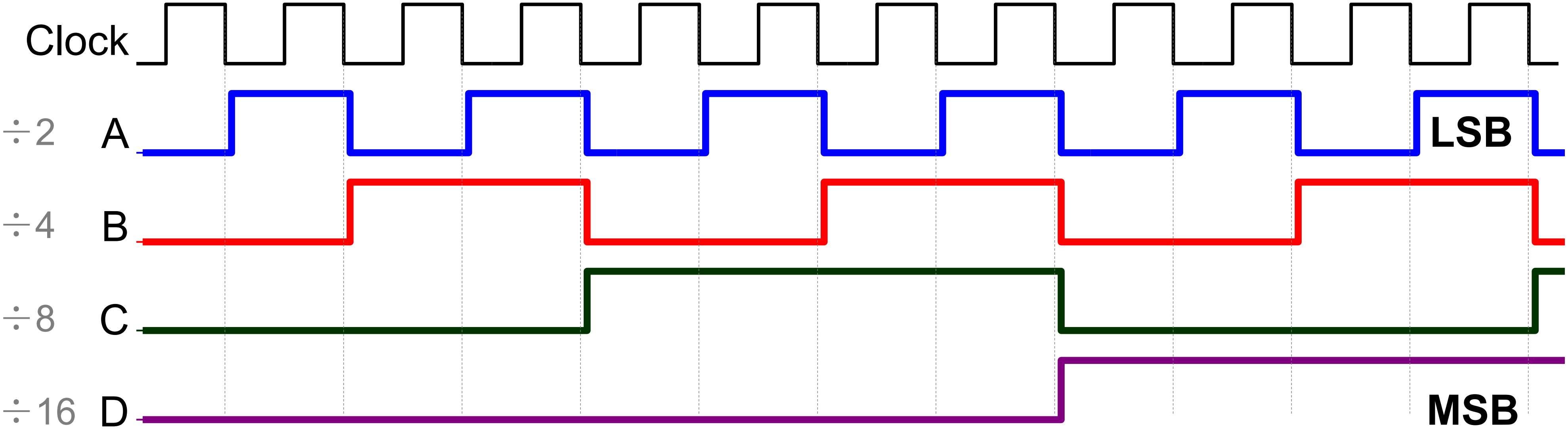
$f_{clk,max} = 1/(\Delta tpd(FF) + \Delta tpd(AND)$

Synchronous (Parallel) Counters

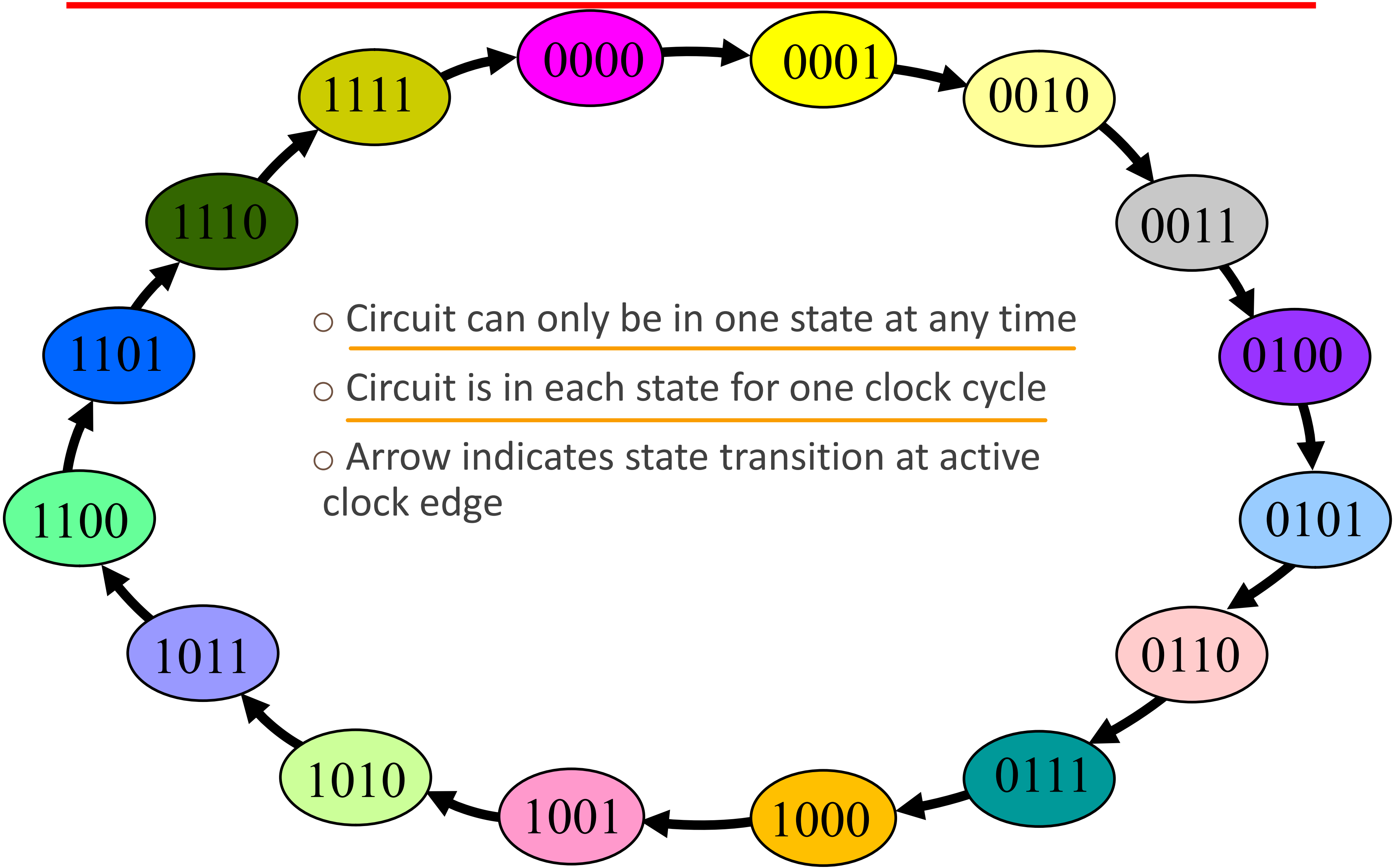
Mod-16 Synchronous Parallel Counter :



```
always @
(negedge clk)

begin

    q <= q + 1;

end
```



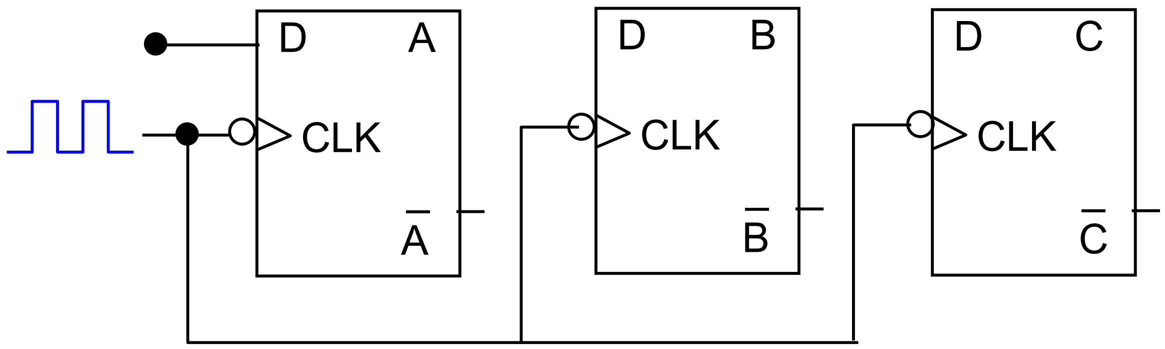
State Transition Diagram



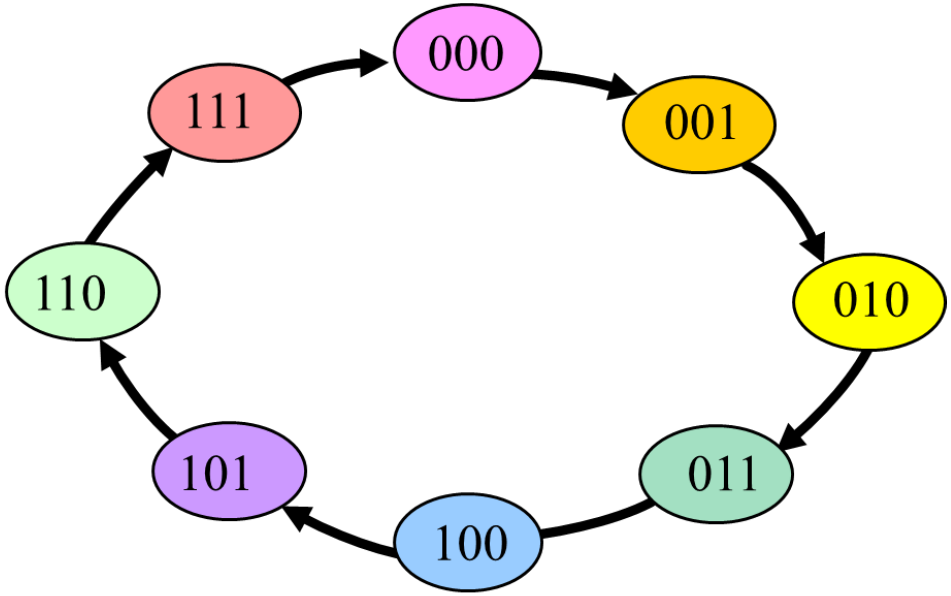
- ○ Circuit can only be in one state at any time
- ○ Circuit is in each state for one clock cycle
- ○ Arrow indicates state transition at active clock edge

Example (D Flip-Flops)

Mod-8 Count-Up Counter Using D-FFs:

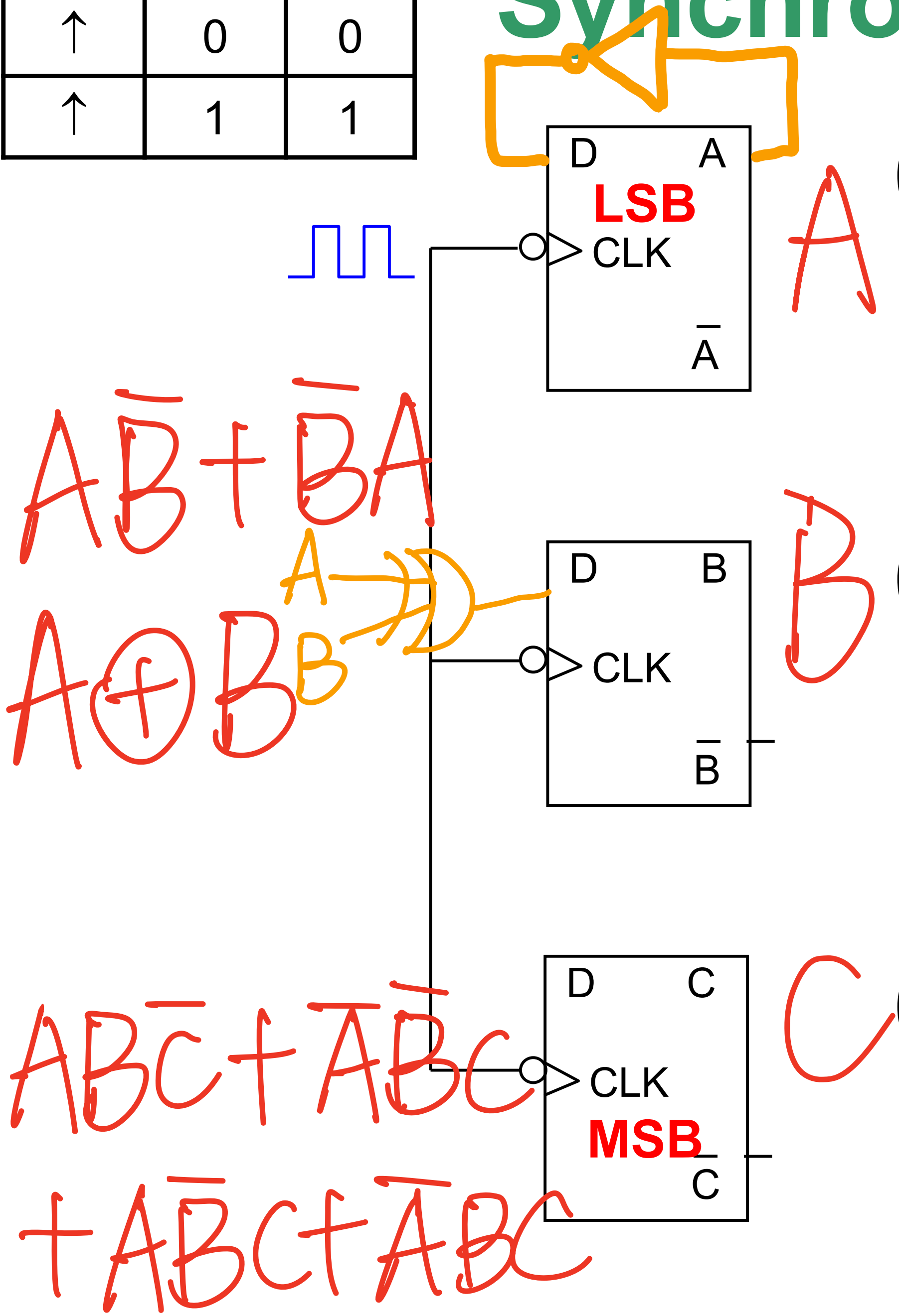


Count	C	B	A
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1
0	0	0	0
c	o	n	t



CLK	D	Q^+
↑	0	0
↑	1	1

Synchronous Counters (DFF)



$A\bar{B}+\bar{B}A$

$A \oplus B$

$AB\bar{C}+\bar{A}\bar{B}C$
$+A\bar{B}C+\bar{A}BC$

A	0	1	0	1	0	1	0	1	0
B	0	0	1	1	0	0	1	1	0
C	0	0	0	0	1	1	1	1	0
Count	0	1	2	3	4	5	6	7	0

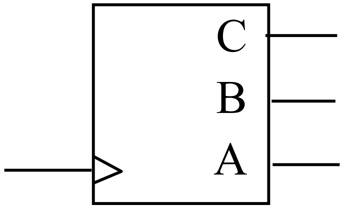
SEQUENTIAL CIRCUITS - III

DESIGN METHOD FOR SYNCHRONOUS COUNTERS

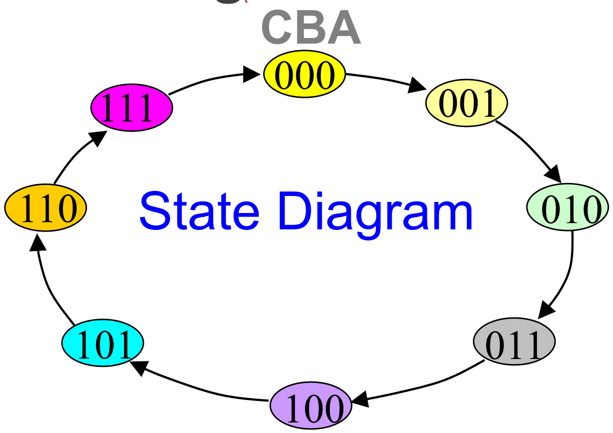
Design Method - Synchronous Counters

Goal : Given the state diagram of a counter realize it using common FFs (and combinational logic).

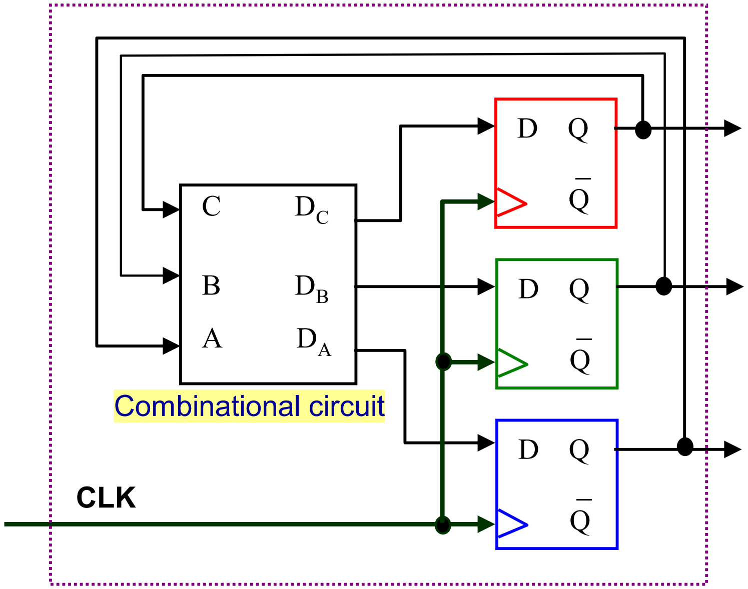
Example : **Design a *3-bit counter* having the following state diagram. Use D FFs.**



Functional block diagram of 3-bit counter.



3-bit synchronous counter



FF outputs are **fed back** to combinational circuit inputs.

Combinational circuit outputs D_A, D_B, & D_C are connected to D FF inputs and will be transferred to the output at next active clock edge.

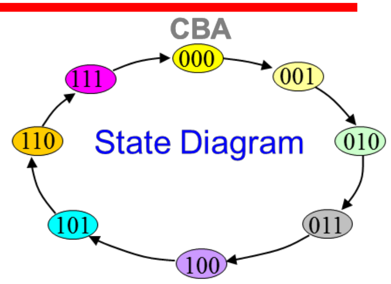
Key: Design combinational circuit to **take previous counter outputs** & **produce the next state**.

Systematic design method is similar to that used for FF conversion considered before.

Design Method : Steps

Step 1

- Draw a State Diagram for the desired Count Sequence



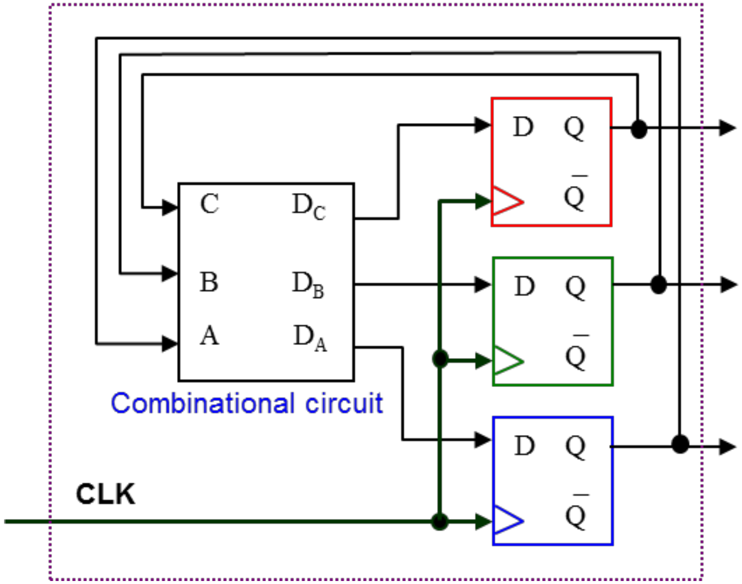
Step 2

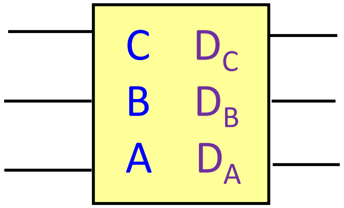
- Determine the Functional Block Diagram of the N-bit Counter.

1) Number of flip-flops?

2) Inputs and Outputs of combinational circuit?

3-bit synchronous counter



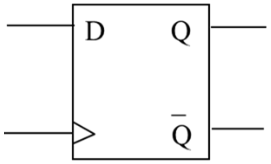


Combinational Circuit

Inputs: Present-state counter outputs (A,B,C).

Outputs: Next-state counter outputs to connect to FF inputs. (D_A, D_B, D_C)

Design Method – Cont'd



D Flip Flop
Excitation Table

Q	Q^+	D
0	0	0
0	1	1
1	0	0
1	1	1

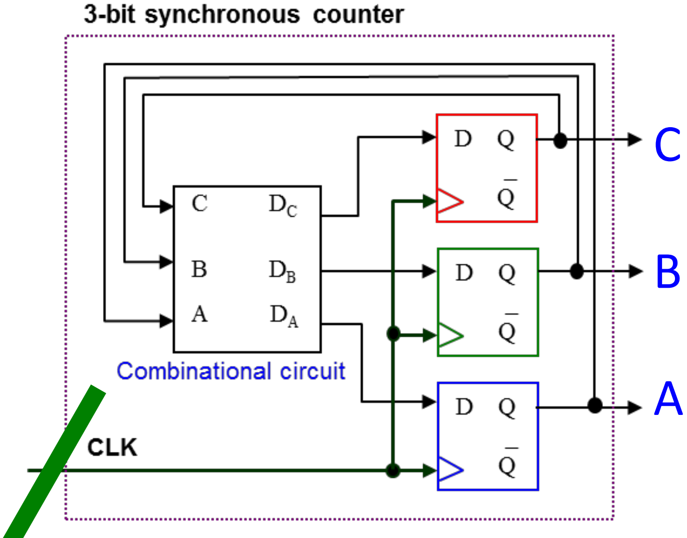
Step 3A

- **Truth table of the combinational circuit.**

A. Determine **next state table** for the counter.

Present-state outputs			Next-state outputs		
C	B	A	C^+ / D_C	B^+ / D_B	A^+ / D_A
0	0	0	0	0	1
0	0	1	0	1	0
0	1	0	0	1	1
0	1	1	1	0	0
1	0	0	1	0	1
1	0	1	1	1	0
1	1	0	1	1	1
1	1	1	0	0	0

Next-State Table



We now have a truth table for the combinational circuit!

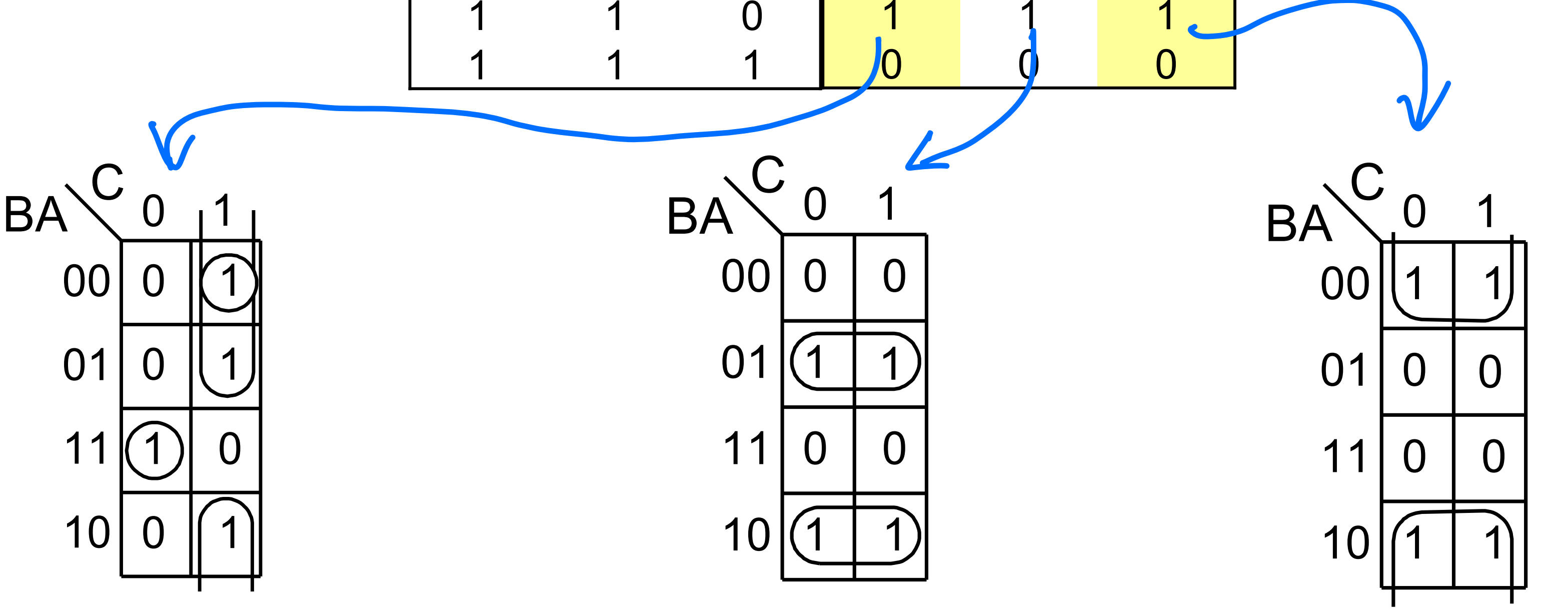
A synchronous counter can be realized with D FFs or with any other FF

Design Method – Cont’d

Step 4

- **Realize the circuit.**

Present-state outputs			Required FF input		
C	B	A	D_C	D_B	D_A
0	0	0	0	0	1
0	0	1	0	1	0
0	1	0	0	1	1
0	1	1	1	0	0
1	0	0	1	0	1
1	0	1	1	1	0
1	1	0	1	1	1
1	1	1	0	0	0



BA \ C	0	1
00	0	1
01	0	1
11	1	0
10	0	1

$D_C = AB\overline{C} + \overline{B}C + \overline{A}C$

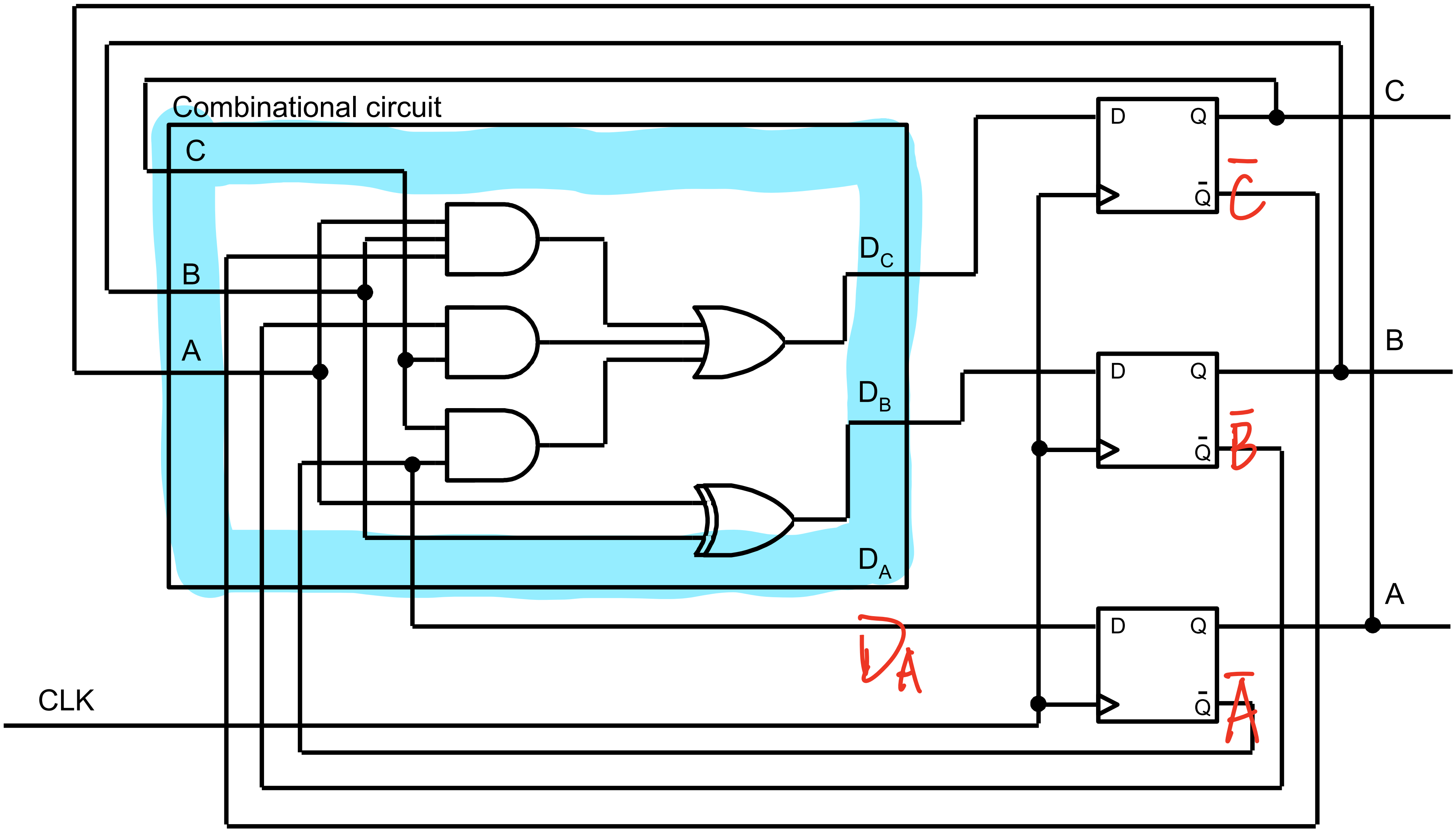
BA \ C	0	1
00	0	0
01	1	1
11	0	0
10	1	1

$D_B = A\overline{B} + \overline{A}B = A \oplus B$

BA \ C	0	1
00	1	1
01	0	0
11	0	0
10	1	1

$D_A = \overline{A}$

Synchronous 3-bit counter



Synchronous Counter Example 2

Design a synchronous counter with count sequence using DFFs:

101,001,000,010,110,100,101,… **(mod-6).**

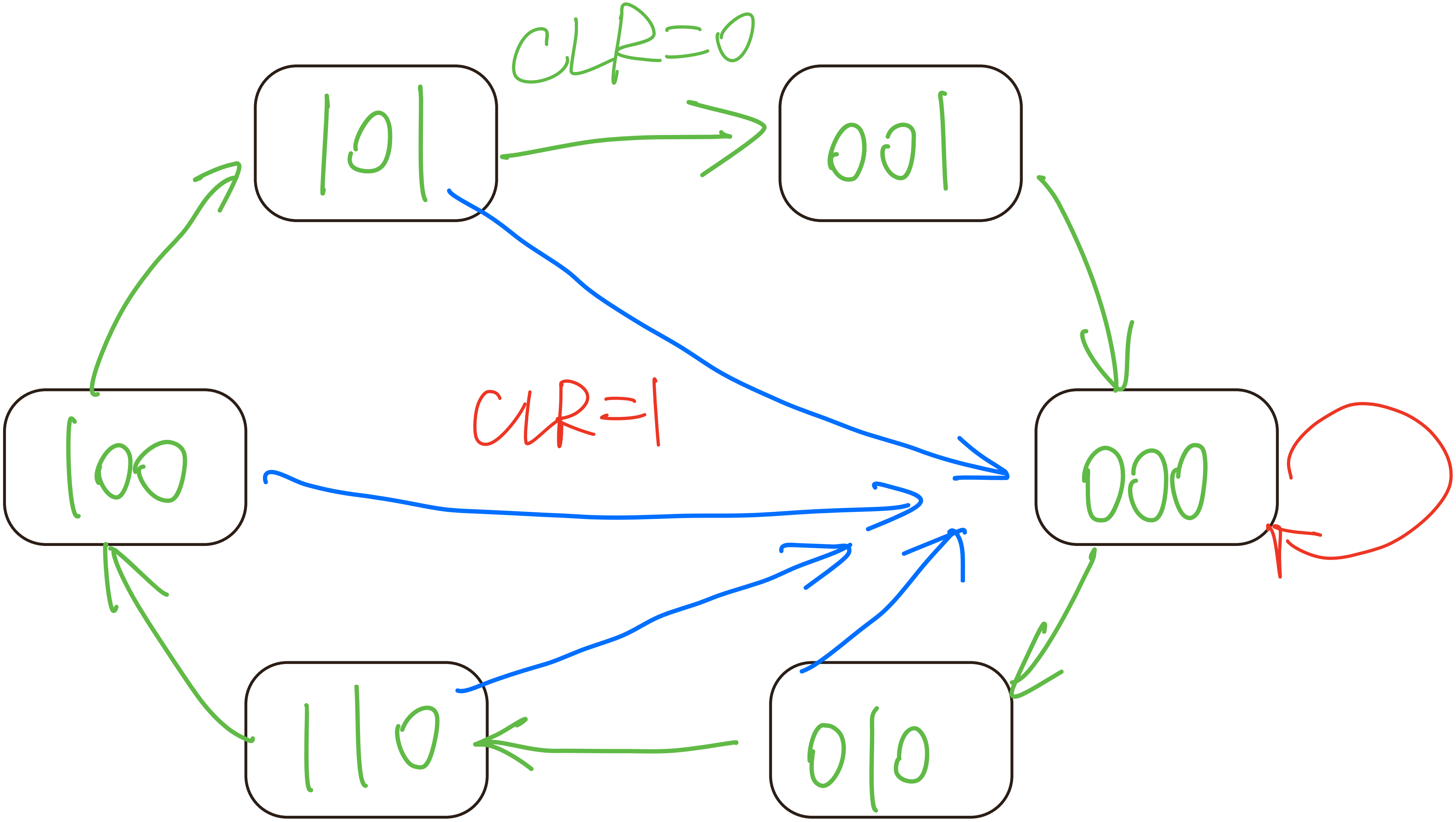
CBA

The counter has an external active high synchronous CLEAR input. When CLEAR = 1, clear the counter to 000 at next active clock edge.

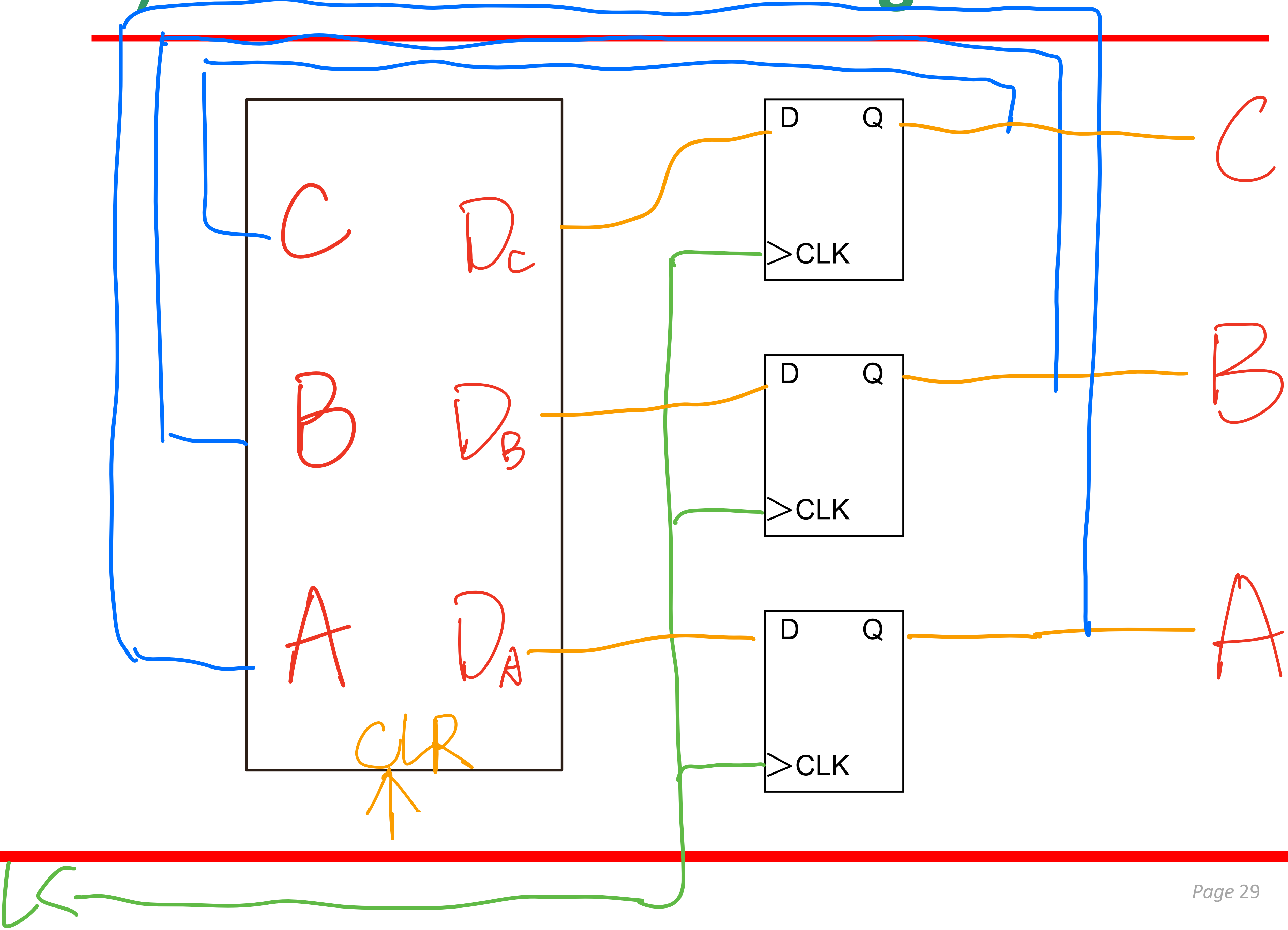
1) Draw the State Transition Diagram
2) Draw the Functional Block Diagram (optional)
3) Derive the output of the next state table using Kmaps.

1) State Diagram

101,001,000,010,110,100,101 ,… (mod-6).



2) Functional Block Diagram



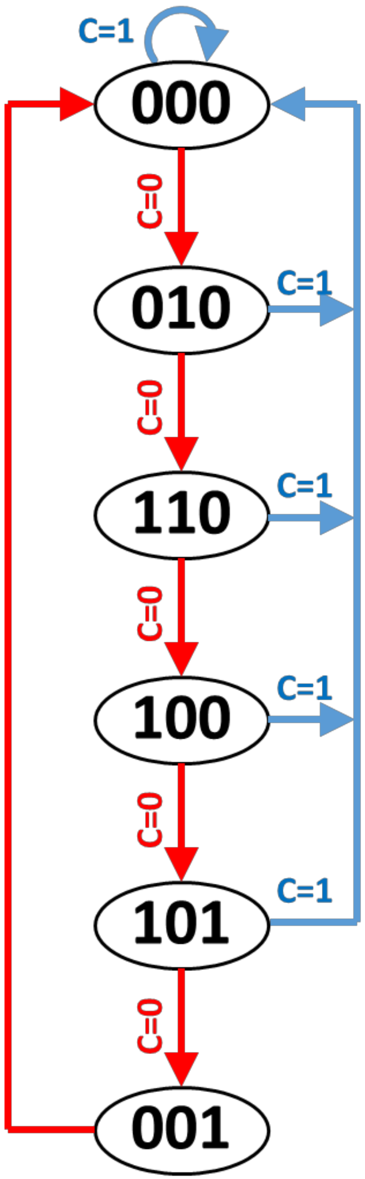
3) Next State Table / Truth table of C.C.

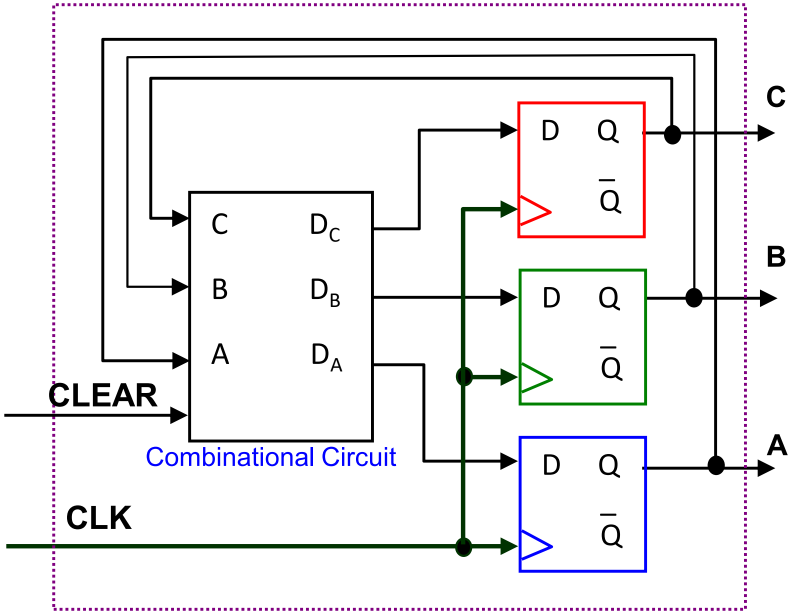
CLR	C	B	A	C+	B+	A+
0	1	0	1	0	0	1
0	0	0	1	0	0	0
0	0	0	0	0	1	0
0	0	1	0	1	1	0
0	1	1	0	1	0	0
0	1	0	0	1	0	1
1	1	0	1	0	0	0
1	0	0	1	0	0	0
1	0	0	0	0	0	0
1	0	1	0	0	0	0
1	1	1	0	0	0	0
1	1	0	0	0	0	0

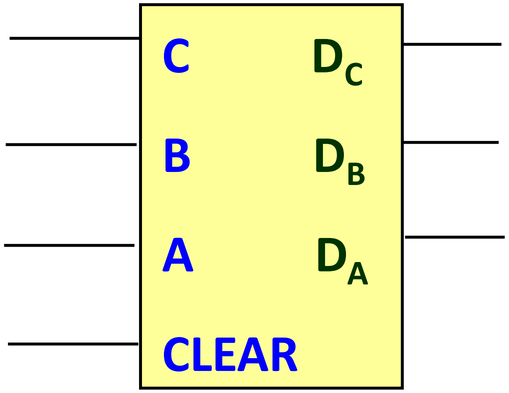
Step1: Write state diagram.

Step2: Determine functional block diagram of counter.

Step3 : Functional block diagram of combinational circuit.







Step4 : Get TT of combinational circuit using FF excitation table.

Step5 : Realize circuit.

CLEAR	C	B	A	C^+	B^+	A^+	D_C	D_B	D_A
0	0	0	0	0	1	0	0	1	0
0	0	0	1	0	0	0	0	0	0
0	0	1	0	1	1	0	1	1	0
0	0	1	1	X	X	X	X	X	X
0	1	0	0	1	0	1	1	0	1
0	1	0	1	0	0	1	0	0	1
0	1	1	0	1	0	0	1	0	0
0	1	1	1	X	X	X	X	X	X
1	X	X	X	0	0	0	0	0	0

SOP for flip-flop inputs

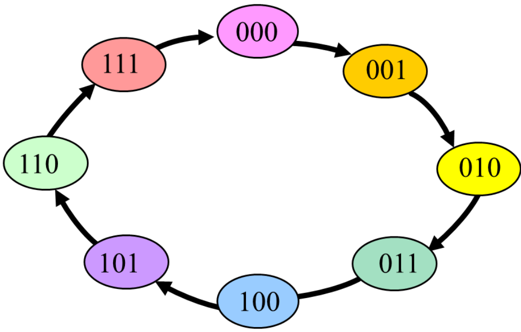
$$D_C = \overline{CLEAR} \bullet B + \overline{CLEAR} \bullet C \bullet \overline{A}$$

$$D_B = \overline{CLEAR} \bullet \overline{C} \bullet \overline{A}$$

$$D_A = \overline{CLEAR} \bullet C \bullet \overline{B}$$

Verilog for Synchronous Counters

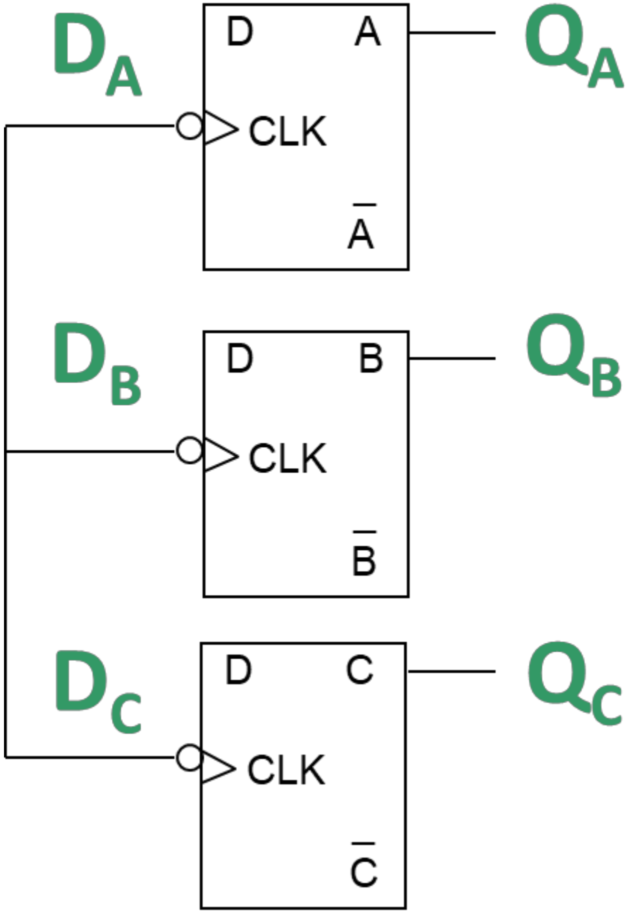
Mod-8 Counter in Verilog



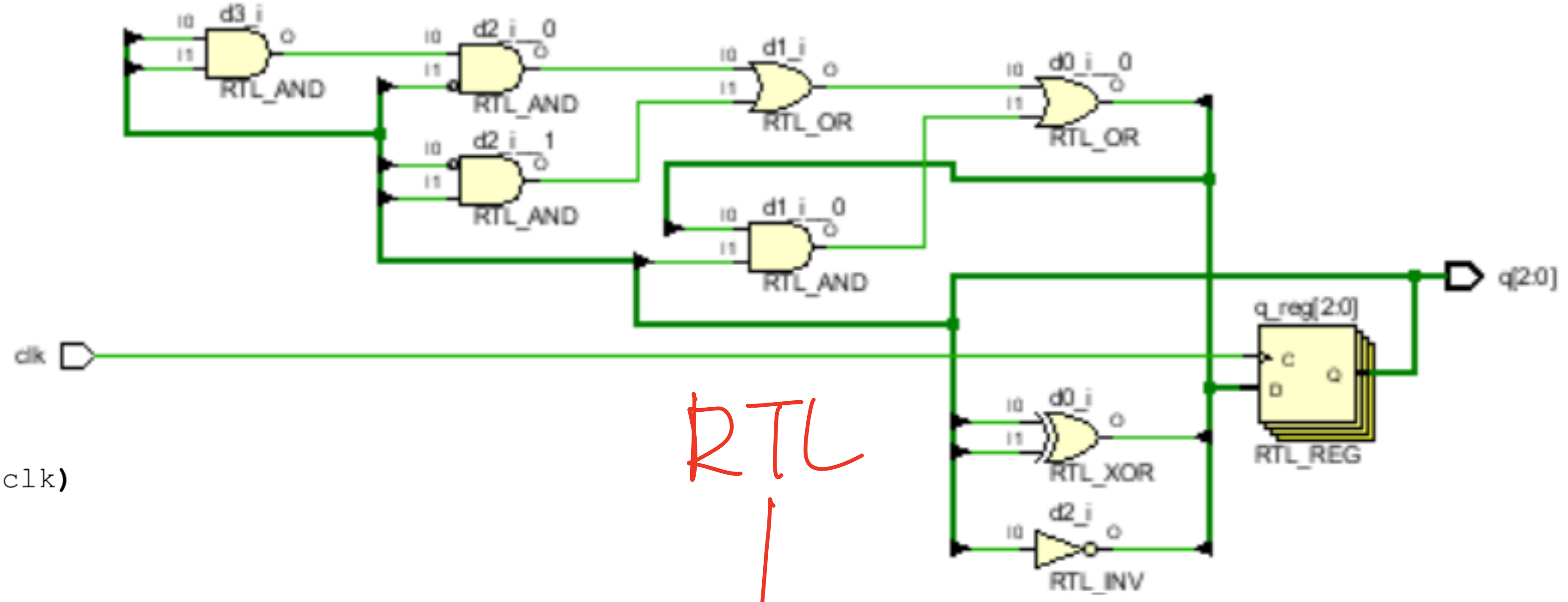
$$D_C = AB\overline{C} + \overline{B}C + \overline{A}C$$

$$D_B = A\overline{B} + \overline{A}B = A \oplus B$$

$$D_A = \overline{A}$$



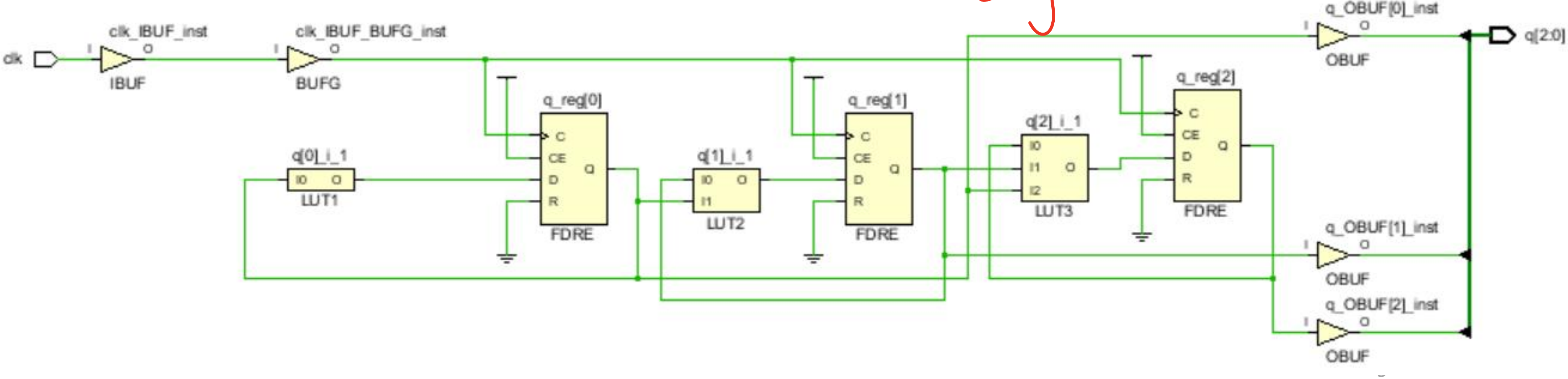
```
module mod8( input clk,
             output reg [2:0] q);

wire [2:0] d;
reg [2:0] q = 0;

always @ (posedge clk)
begin
    q <= d;
end

assign d[0] =  ~q[0]

assign d[1] = q[0]^q[1]

assign d[2] =

endmodule
```



```
always @ (posedge clk)

begin
    q <= d;
end

assign d[0] = ~q[0];

assign d[1] = q[0] ^ q[1];

assign d[2] = q[0]&q[1]&~q[2] |
             ~q[1]&q[2] | ~q[0]&q[2];
```

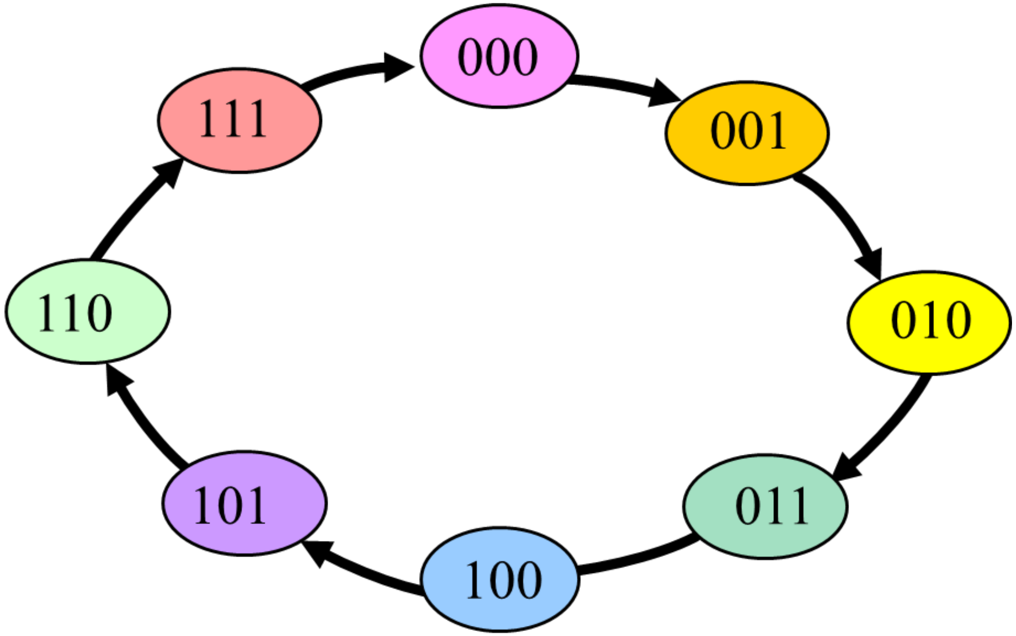
RTL

Sythesis

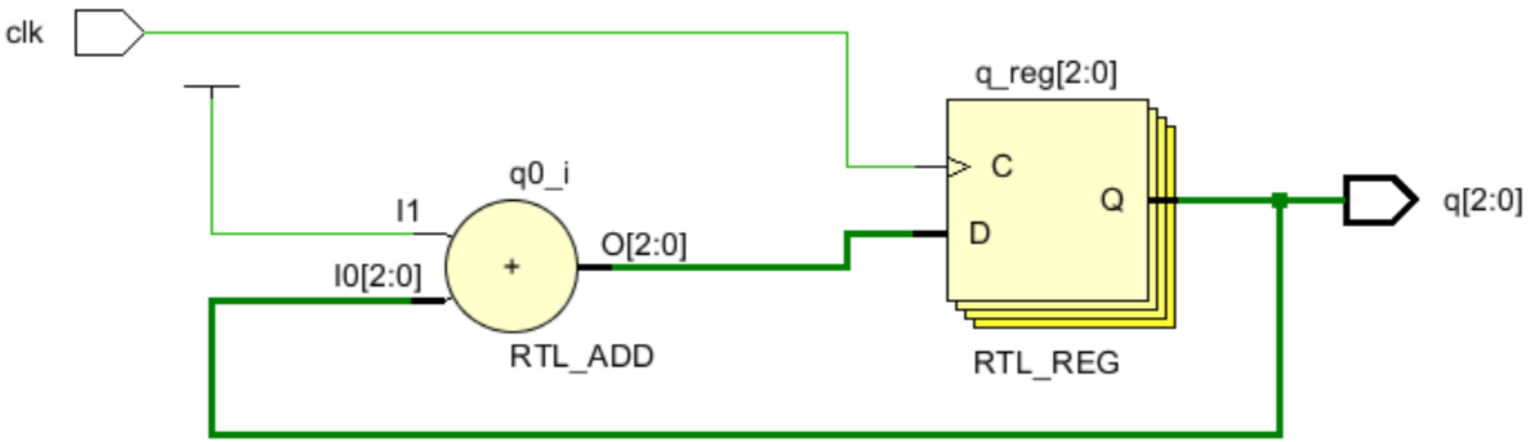


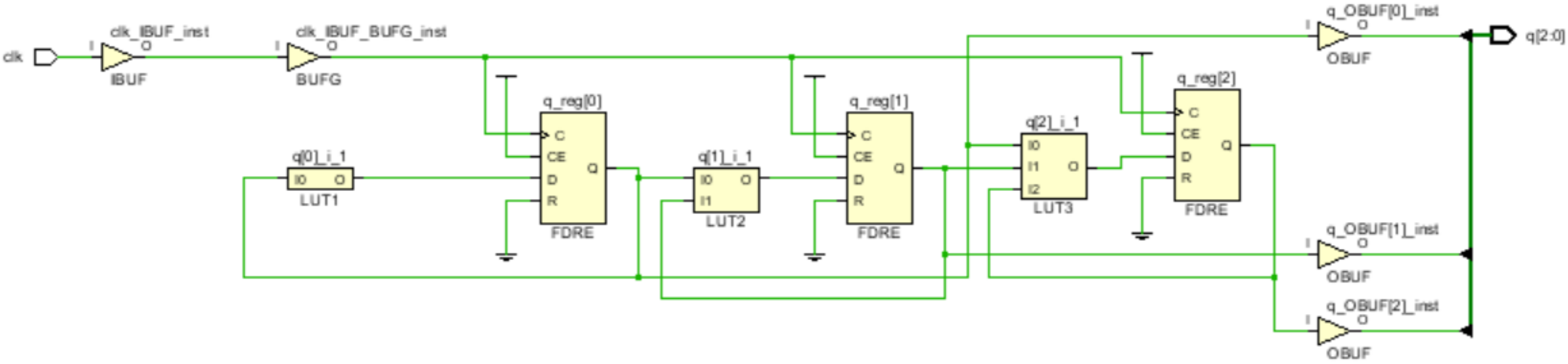
Mod-8 Counter in Verilog

```
module mod8( input clk,
             output reg [2:0] q);

initial begin
   q = 3'b000;
end

always @ (posedge clk)

begin

   q <= q + 1;

end

endmodule
```



```
initial begin
   q = 3’b000;
end

always @ (posedge clk)

begin

    q <= q + 1;

end

endmodule
```

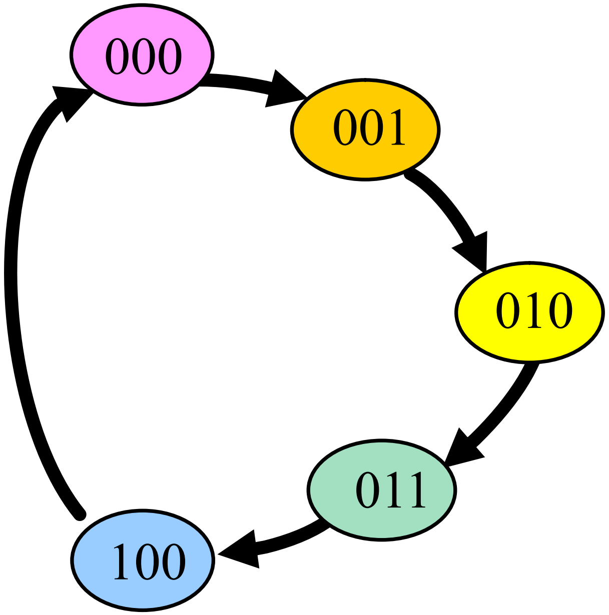




Precise Clock Generation in Verilog?

```
module mod6(  input clk,
              output reg [2:0] q,
              output reg led);

initial begin
   q = 3'b000;
   led = 0;
end

always @ (posedge clk)

begin

   q <= (q == 3'b100) ?  0 : q + 1;
   led <= (q == 3'b000) ? ~led : led;

end
endmodule
```

0 to 4



What is frequency of led? $\frac{f_{clk}}{10}$