SEQUENTIAL CIRCUITS - II

Ask Week 6 Questions here…

You can ask questions during the week using slido here :

https://app.sli.do/event/eaDnCNnRsdRpc7vviSMziF

Or at slido.com + #2026 002

Or the tiny little QR :

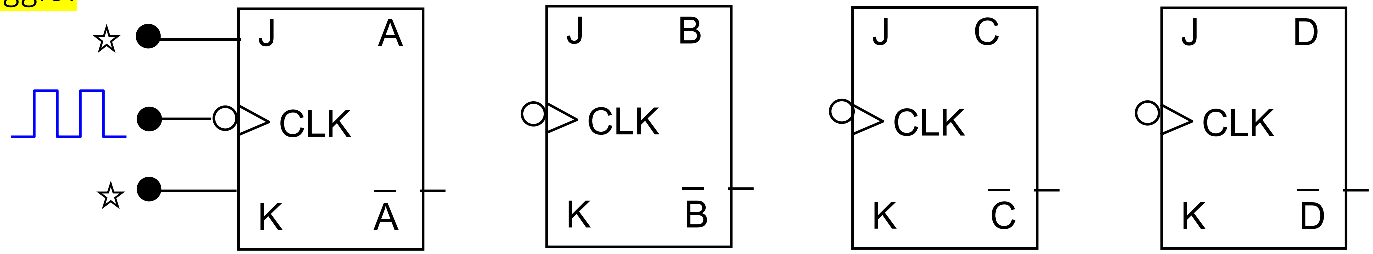
Counters

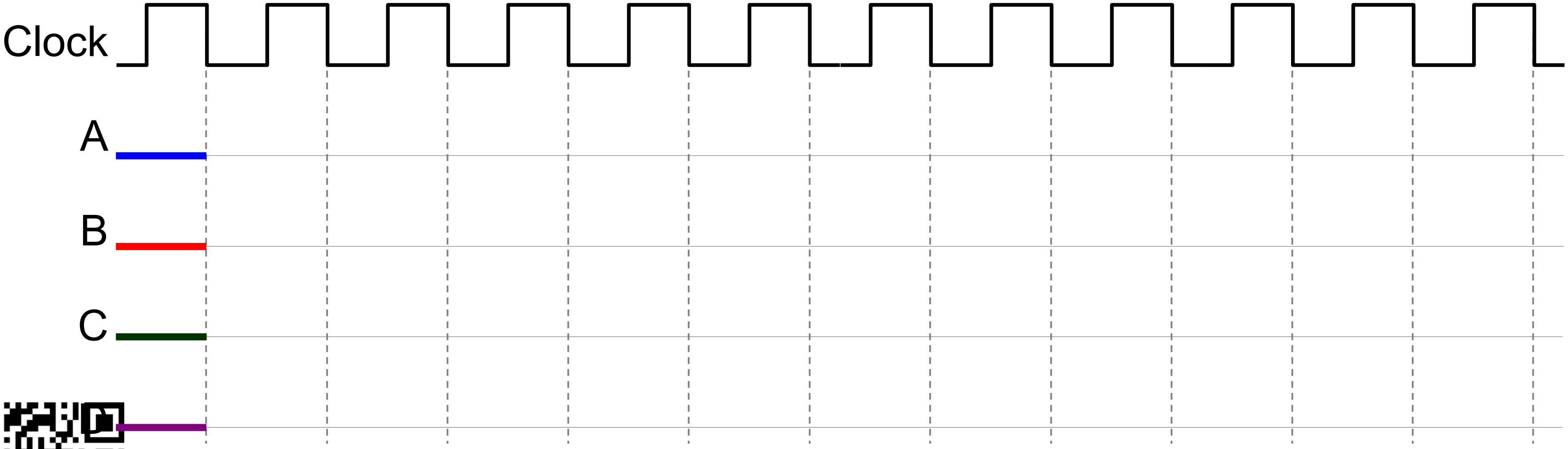
Asynchronous : Circuit elements do not get the clock input simultaneously

Synchronous Counters: Circuit elements get the clock input simultaneously

☆ All J & K inputs **HIGH**
⇒ FF outputs toggle!

Ripple Counter (Asynchronous):



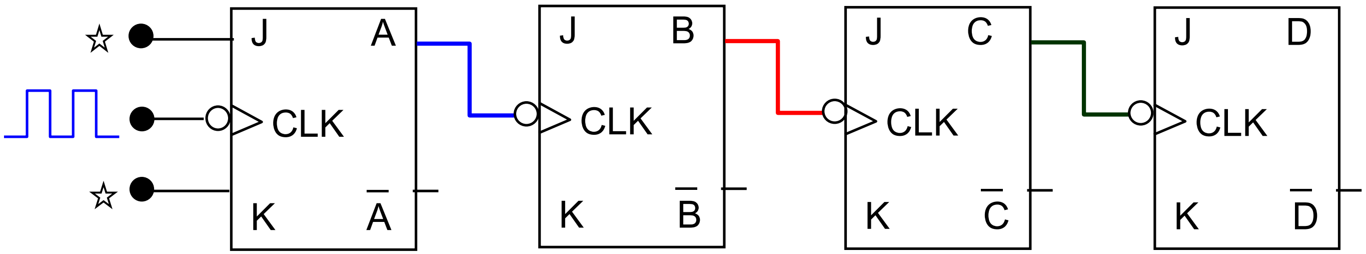


Counters

Asynchronous : Circuit elements do not get the clock input simultaneously

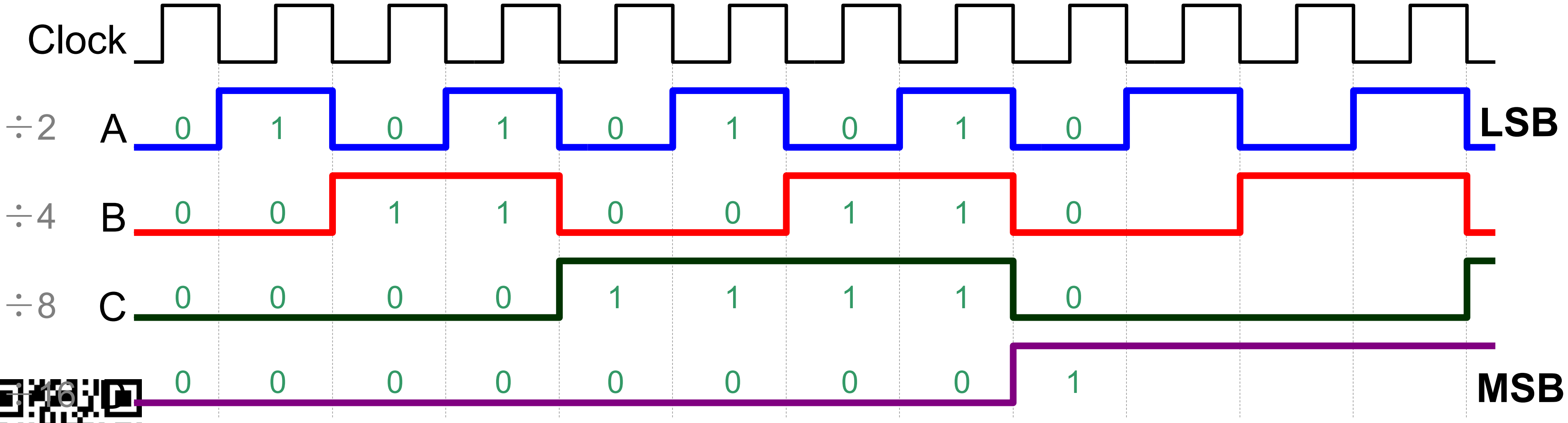
Synchronous Counters: Circuit elements get the clock input simultaneously

Ripple Counter (Asynchronous):

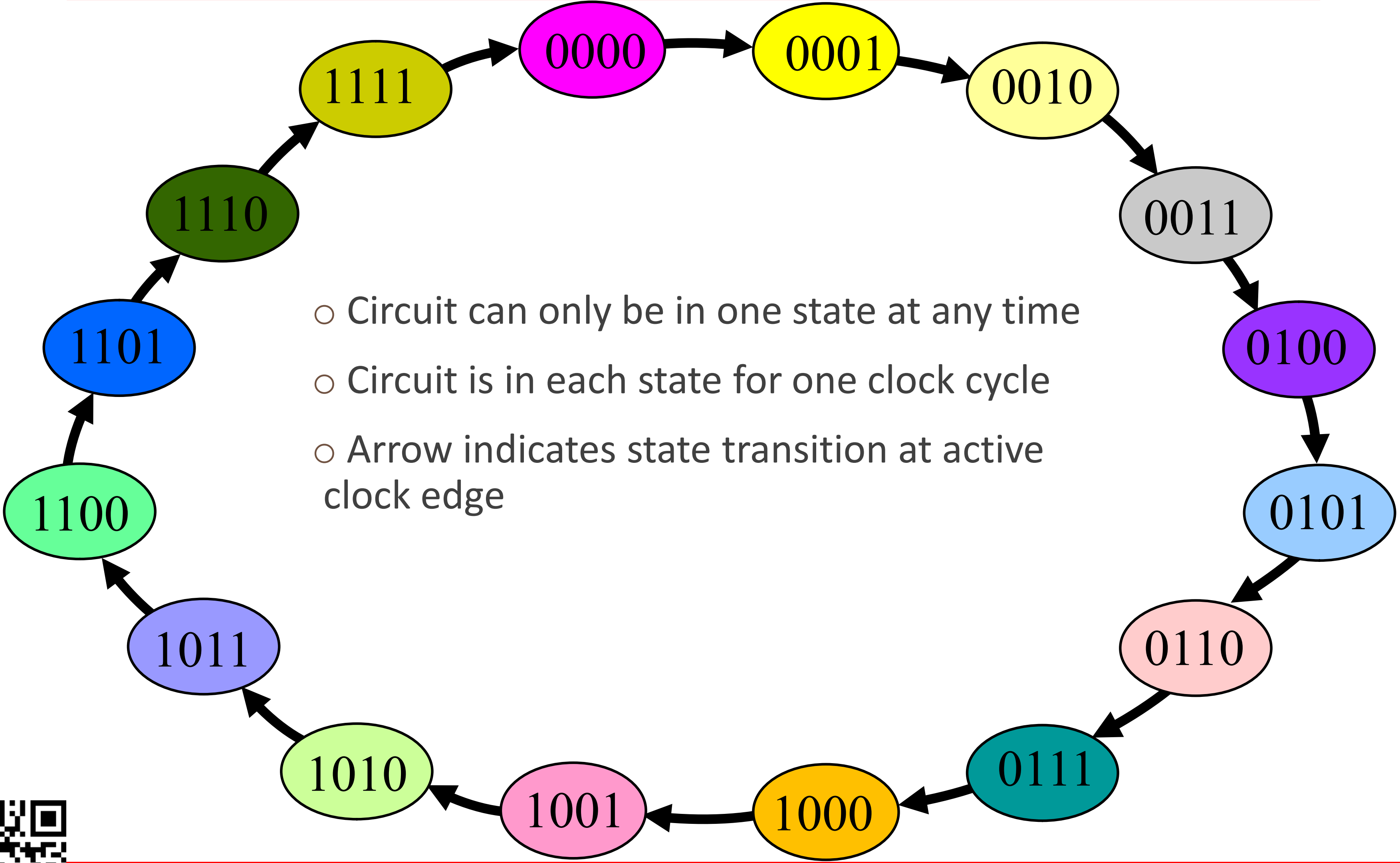


☆ All J & K inputs HIGH

⇒ FF outputs toggle!



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

Counters : Mod - X

- Each FF successively halves the input clock frequency
- The 4-bit counter counts from 0000 (0) → 1111(15)
  ➔ 16 distinct count states ⇒ called a mod-16 counter
- *N* x FFs connected this way will have ___ states ⇒ mod-

How to obtain a counter with mod-X < 2^N?

1. Use FFs with CLR functions
2. Assume counter starts from 0
3. Identify FFs that will be in HIGH state when count = X
4. Feed those FFs outputs to a NAND gate
5. Connect NAND gate output to ***asynchronous*** CLR

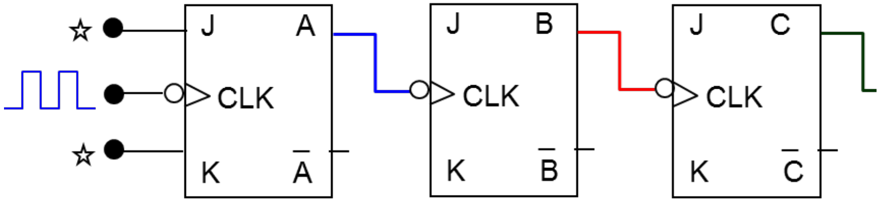
CLK	CLR	J	K	Q^+
X	L	X	X	L
↓	H	L	L	Q
↓	H	L	H	L
↓	H	H	L	H
↓	H	H	H	$\bar{Q}$

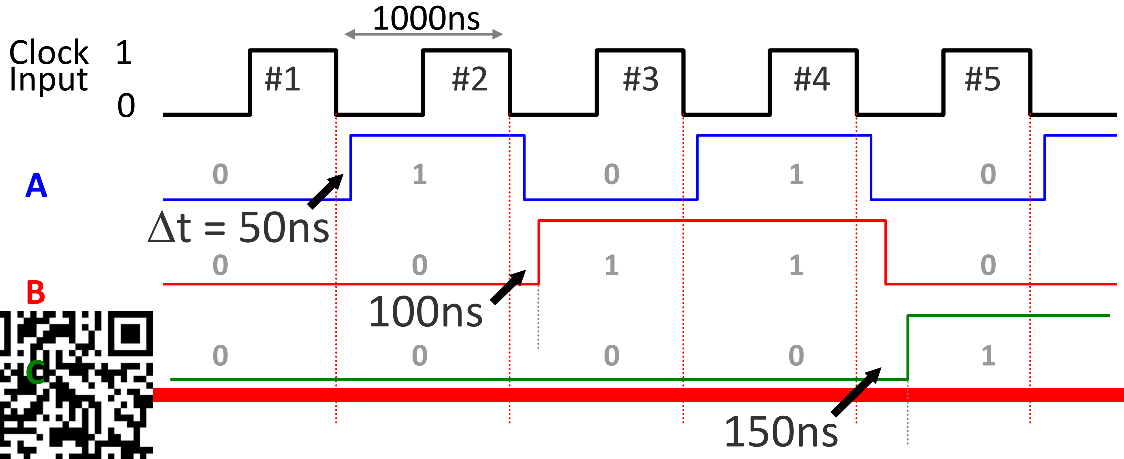
t_{pd} ➔ limiting frequency

- Ripple counters are very easy to implement
- But they have a major drawback ➔ they cannot operate beyond a limiting frequency
- Due to *propagation delays* of the FFs in the chain adding up:

1. Clock input FF_1: t_0
2. Clock input FF_2: $t_0 + \Delta tpd$
3. Clock input FF_3: $t_0 + 2*\Delta tpd$
4. Clock input FF_N: $t_0 + (n - 1) * \Delta tpd$

➔ the nth FF changes state at n* Δt after t_0





For proper operation:

Tclock ≥ (n*Δtpd)

fmax ≤ 1 / (n*Δtpd)

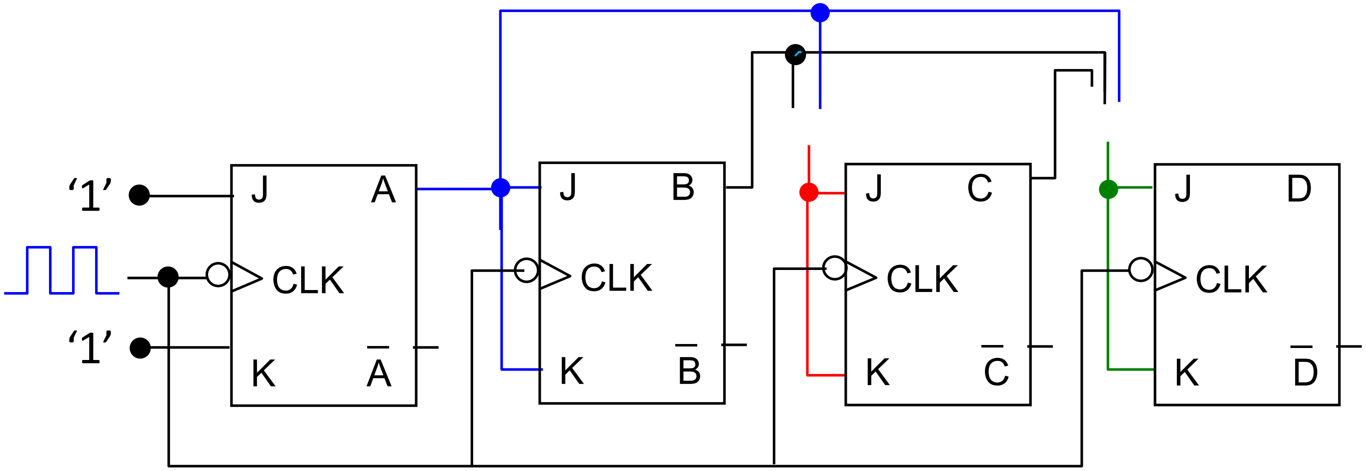
Synchronous (Parallel) Counters

Mod-16 Synchronous Parallel Counter :

B toggles when _____

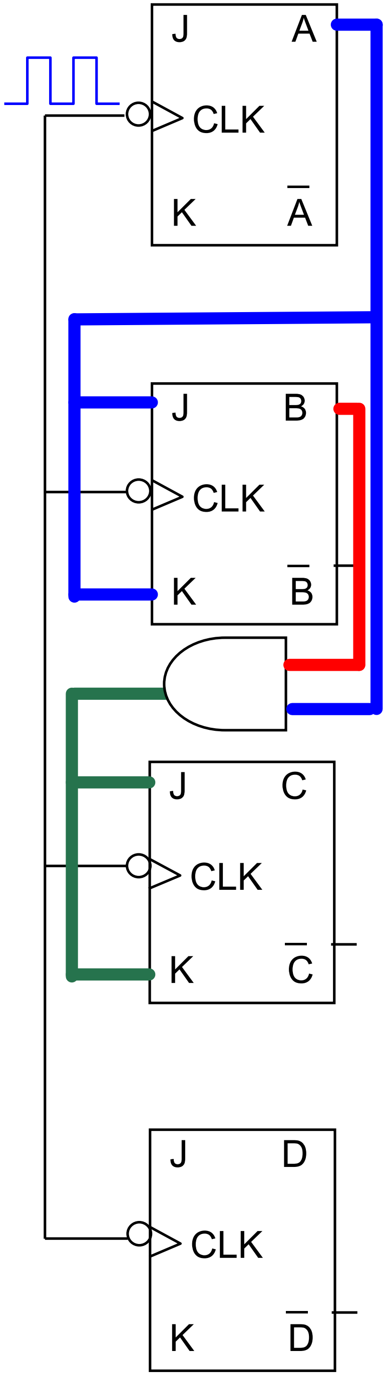
C toggles when _____

D toggles when _____



Count	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	0
	c	o	n	t

- ○ Since all FFs are triggered simultaneously by the clock, this counter can operate at higher frequencies.
- ○ Critical Path Delay = Δtpd (FF) + Δtpd (AND)
- ○ Operating frequency is irrespective of the number of FFs
- ○ Disadvantages?

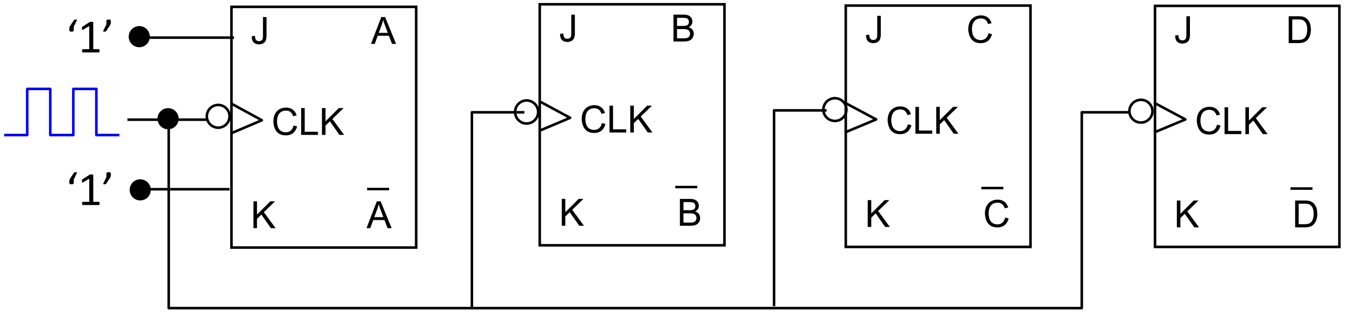


0	1	0	1	0	1	0	1	0	1	0	1
0	0	1	1	0	0	1	1	0	0	1	1
0	0	0	0	1	1	1	1	0	0	0	0
0	0	0	0	0	0	0	0	1	1	1	1
0	1	2	3	4	5	6	7	8	9	10	11

Counting Down…

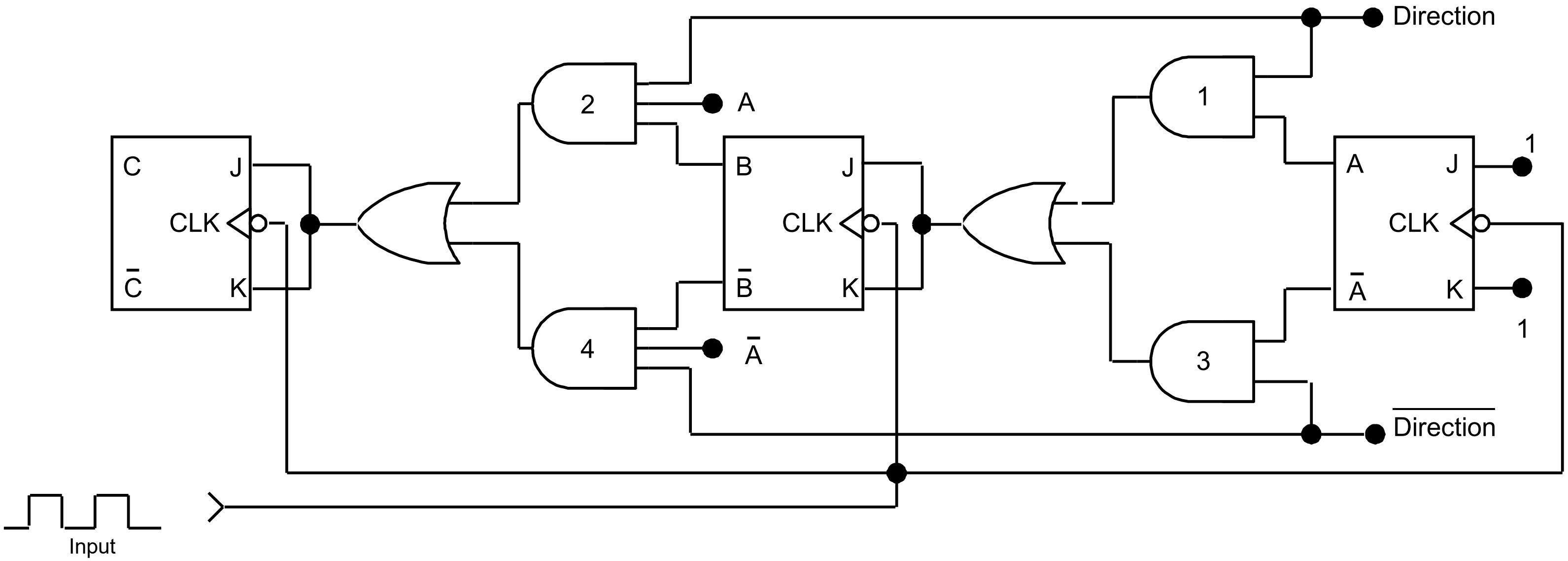
What about a count down mod-16 counter ?

B toggles when
C toggles when
D toggles when



Count	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	0
	c	o	n	t

Up/Down Synchronous Counters



- Counting Up : $Direction = 1, \overline{Direction} = 0$

  J,K_{FFB} =

  J,K_{FFC} =

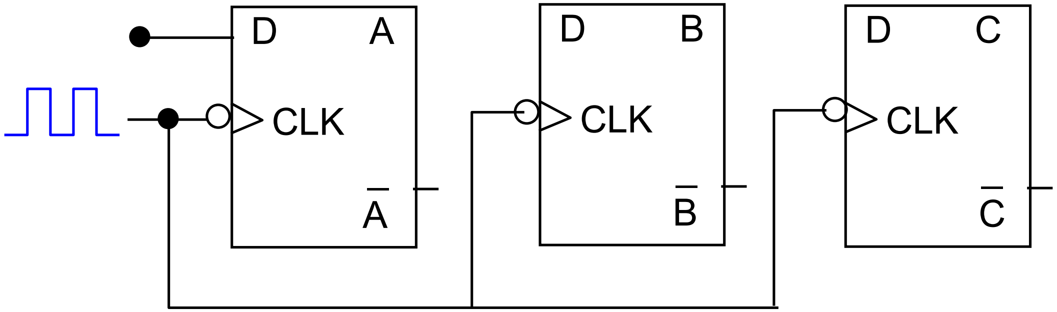
- Counting Down : $Direction = 0, \overline{Direction} = 1$

  J,K_{FFB} =

  J,K_{FFC} =

Example (D Flip-Flops)

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

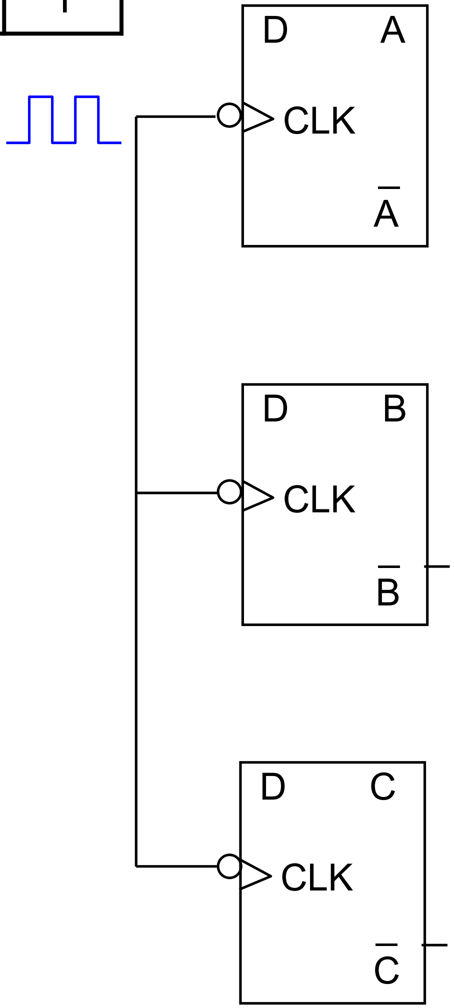


Count	C	B	A
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1
0	0	0	0
c	o	n	t

CLK	D	Q^+

CLK	D	Q^+
↑	0	0
↑	1	1

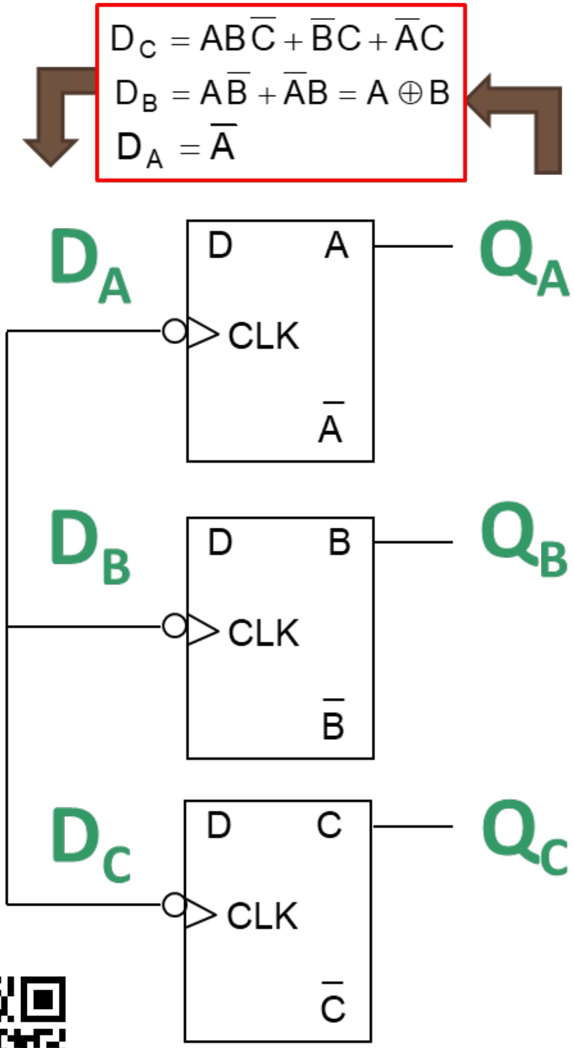
Synchronous Counters (DFF)



0	1	0	1	0	1	0	1	0
0	0	1	1	0	0	1	1	0
0	0	0	0	1	1	1	1	0
0	1	2	3	4	5	6	7	0

Verilog for
Synchronous Counters

Mod-8 Counter in Verilog



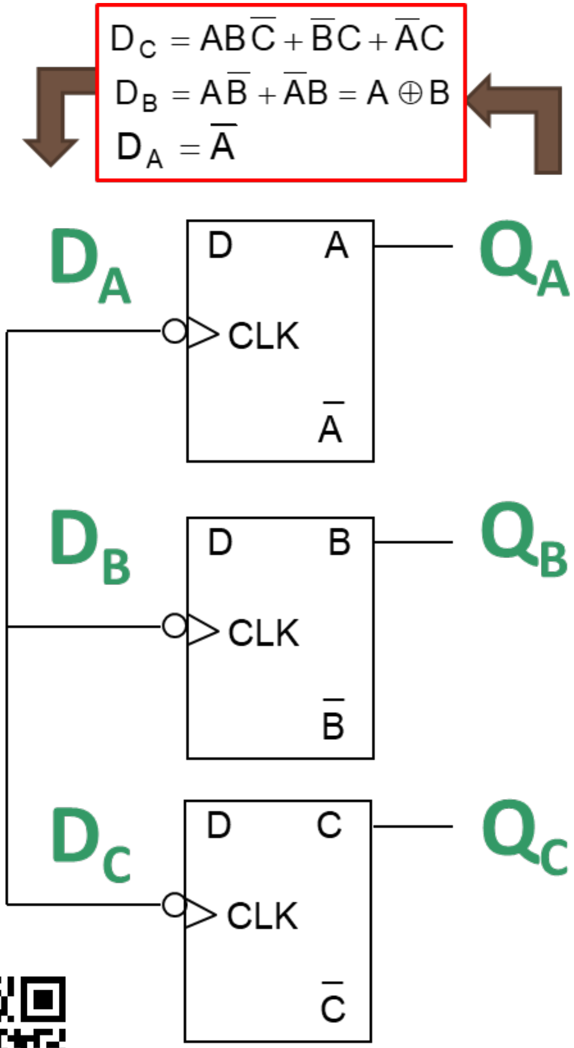
$$D_C = AB\overline{C} + \overline{B}C + \overline{A}C$$

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

$$D_A = \overline{A}$$

```
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] =

assign d[1] =

assign d[2] =

endmodule
```

Mod-8 Counter in Verilog



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

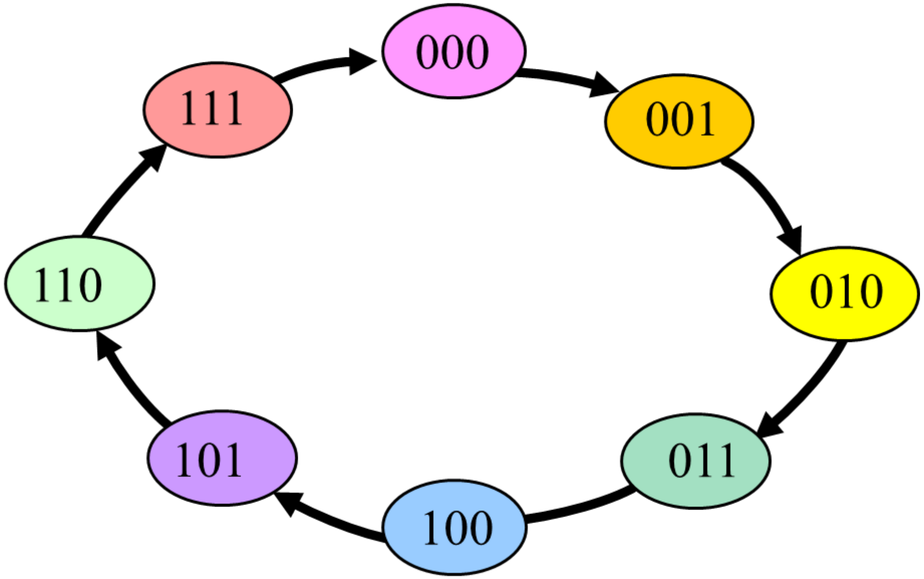
$$D_B = A\overline{B} + \overline{A}B = A \oplus B$$

$$D_A = \overline{A}$$

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

endmodule
```

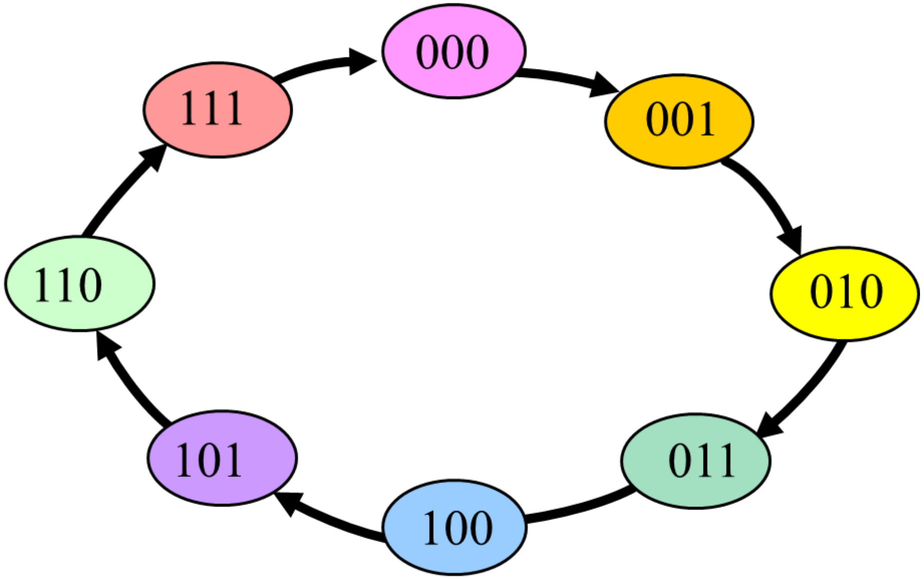
Mod-8 Counter in Verilog

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

initial begin
   q = 3’b000;
end

always @ (posedge clk)

begin

   _________________________

end

endmodule
```



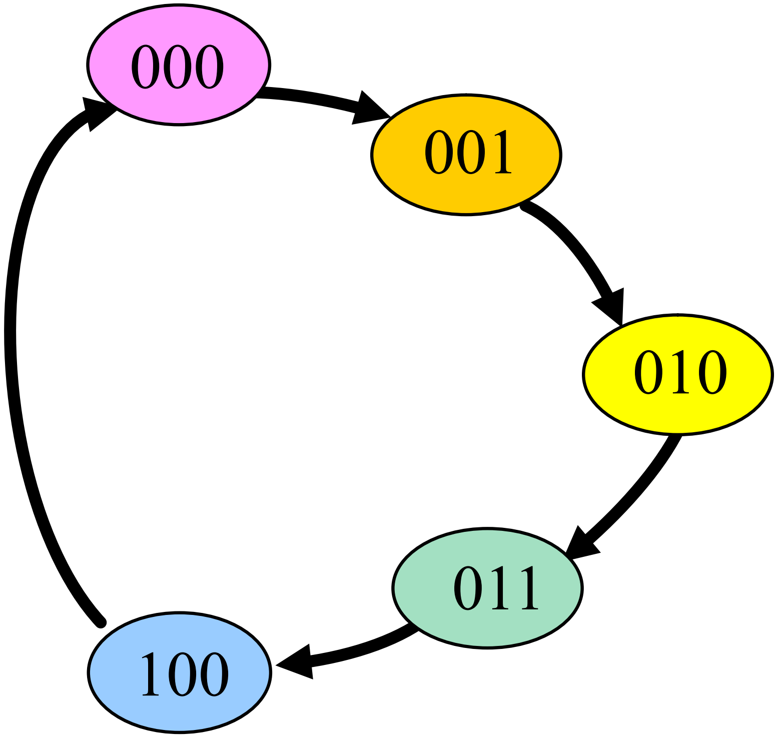
Mod-8 Counter in Verilog

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

initial begin
    q = 3’b000;
end

always @  (posedge clk)

begin

    q <= q + 1;

end

endmodule
```



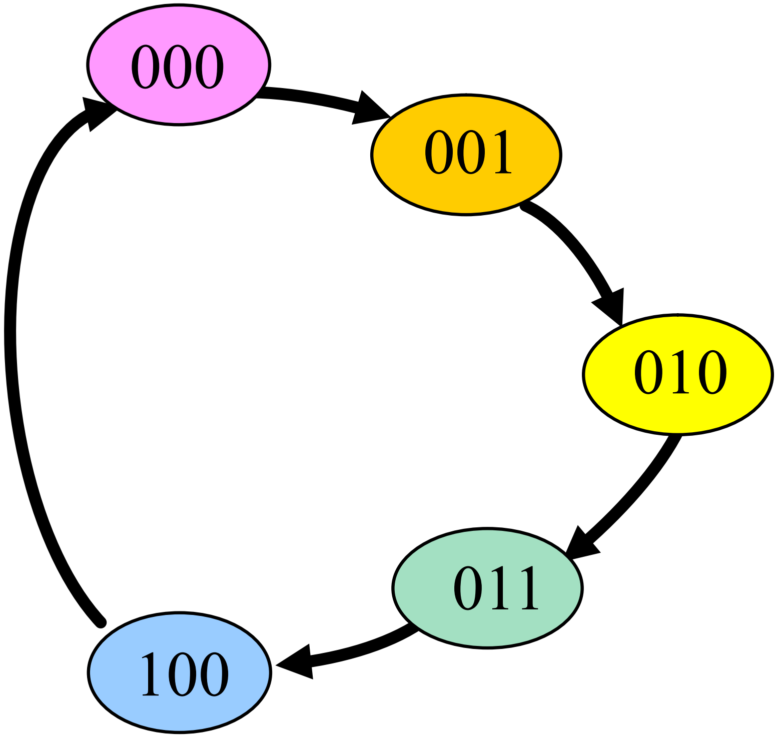
Mod-5 Counter in Verilog

```
module mod5( input clk, clr,
             output reg [2:0] q);

initial begin
   q = 3’b000;
end


always @ (posedge clk, posedge clr)

begin

    q <= ____________________

end

endmodule
```



Mod-5 Counter in Verilog

```
module mod5( input clk, clr,
             output reg [2:0] q);

initial begin
   q = 3'b000;
end


always @ (posedge clk, posedge clr)

begin

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

end

endmodule
```



Last Slide!

- Incrementing / Counting is easy in Verilog! ➔ COUNT <= COUNT + 1;
- What about the following features?
  - Positive / Negative clock edge triggered
  - Counting Up / Counting Down
  - mod-X Counters
  - Synchronous / Asynchronous Resets
  - Synchronous / Asynchronous Presets

```
module counter(input clear, clk, output reg [3:0] q);

always @ (posedge clk) begin

    q <= clear ? (q - 1) : 4’b0000;

end
endmodule
```

- What counter does this code describe?

1. Positive/Negative Edge clock triggered?
2. Asynchronous / Synchronous Clear?
3. Count Up / Down Counter ?

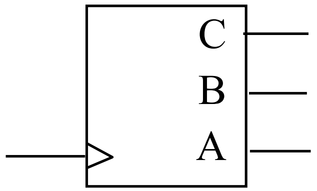
SEQUENTIAL CIRCUITS - III

DESIGN METHOD FOR SYNCHRONOUS COUNTERS

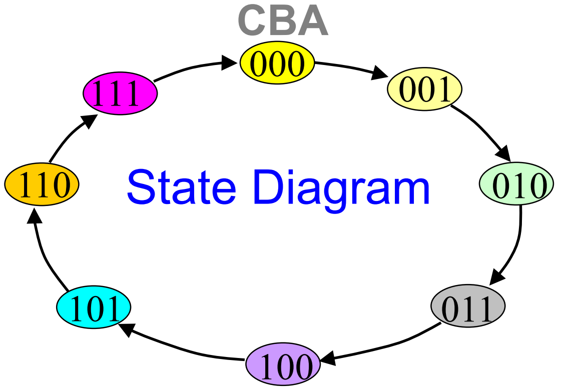
Design Method - Synchronous Counters

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

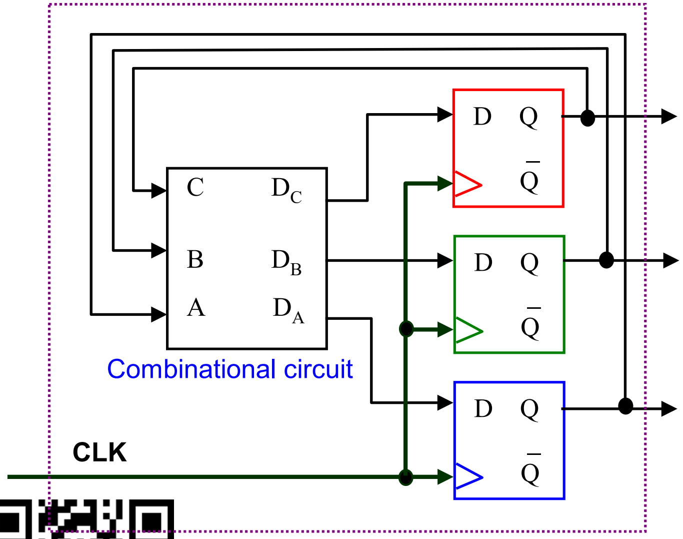
***Example* : Design a *3-bit counter* having the following state diagram. Use D FFs.**



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.

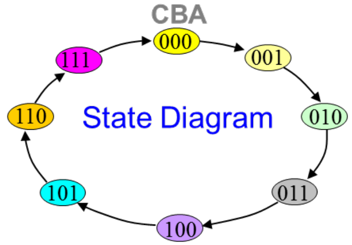
Key: Design combinational circuit to **take previous counter outputs** & **produce the next state**.

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

Design Method : Steps

Step 1

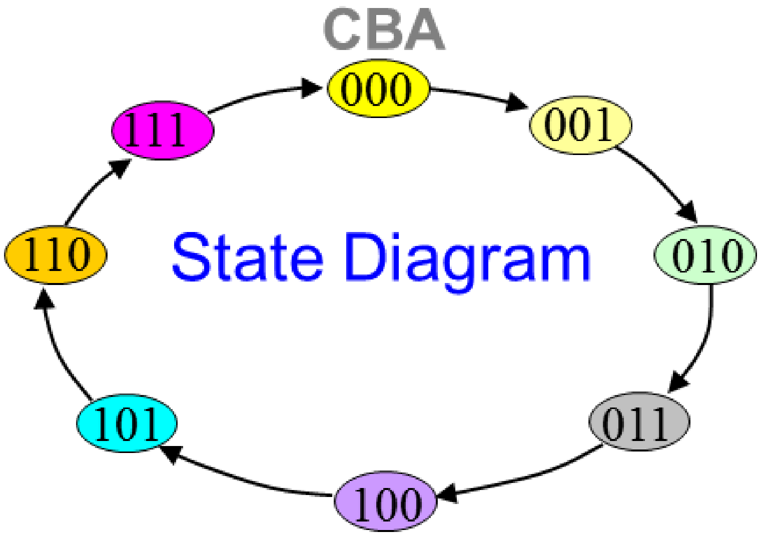
- Draw a State Diagram for the desired Count Sequence



Step 2

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

1) Number of flip-flops?
2) Inputs and Outputs of combinational circuit?

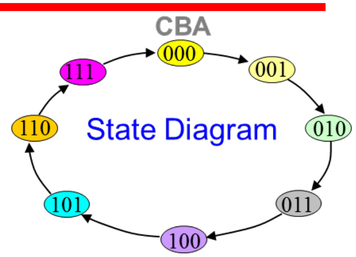


1) Number of flip-flops?

Design Method : Steps

Step 1

- Draw a State Diagram for the desired Count Sequence



Step 2

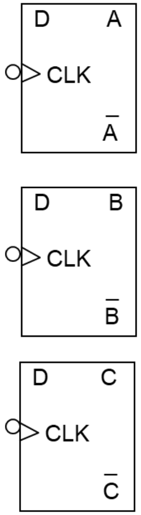
- Determine the Functional Block Diagram of the N-bit Counter.

1) Number of flip-flops?

2) Inputs and Outputs of combinational circuit?

1) Number of flip-flops?

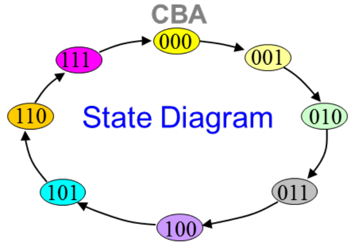
2) Inputs and Outputs of combinational circuit?



Design Method : Steps

Step 1

- Draw a State Diagram for the desired Count Sequence



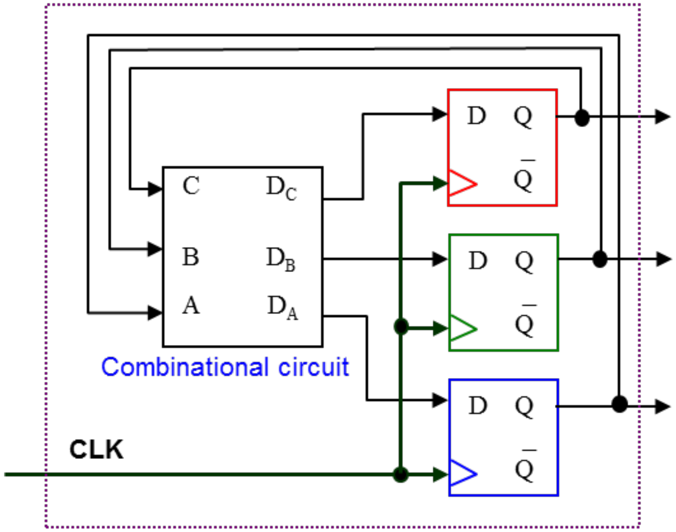
Step 2

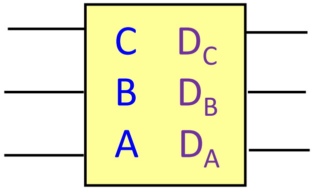
- Determine the Functional Block Diagram of the N-bit Counter.

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)

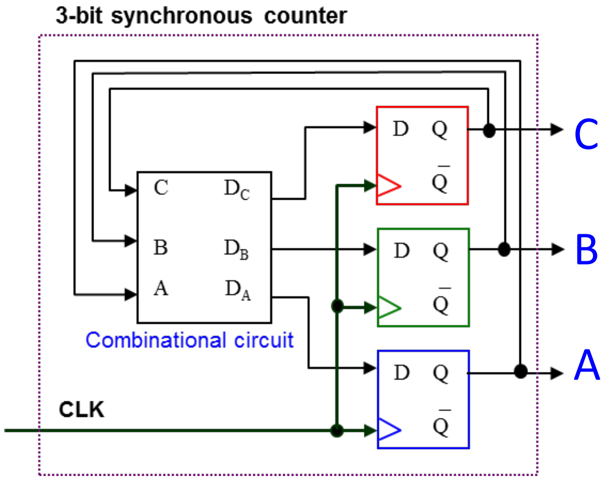
Design Method – Cont’d

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^+	B^+	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



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

Design Method – Cont'd

***Excitation Table :** Specifies what the **FF** inputs should be for a specific Q → Q^+ transition to occur.

Step 3B

- **Truth table of the combinational circuit.**
  B. Using the excitation table, determine the output values of the combinational circuit.

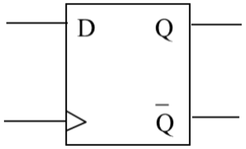
Present-state outputs

Next-state outputs

Required FF input

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

Next-State Table

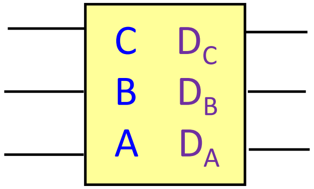


D Flip Flop Excitation Table

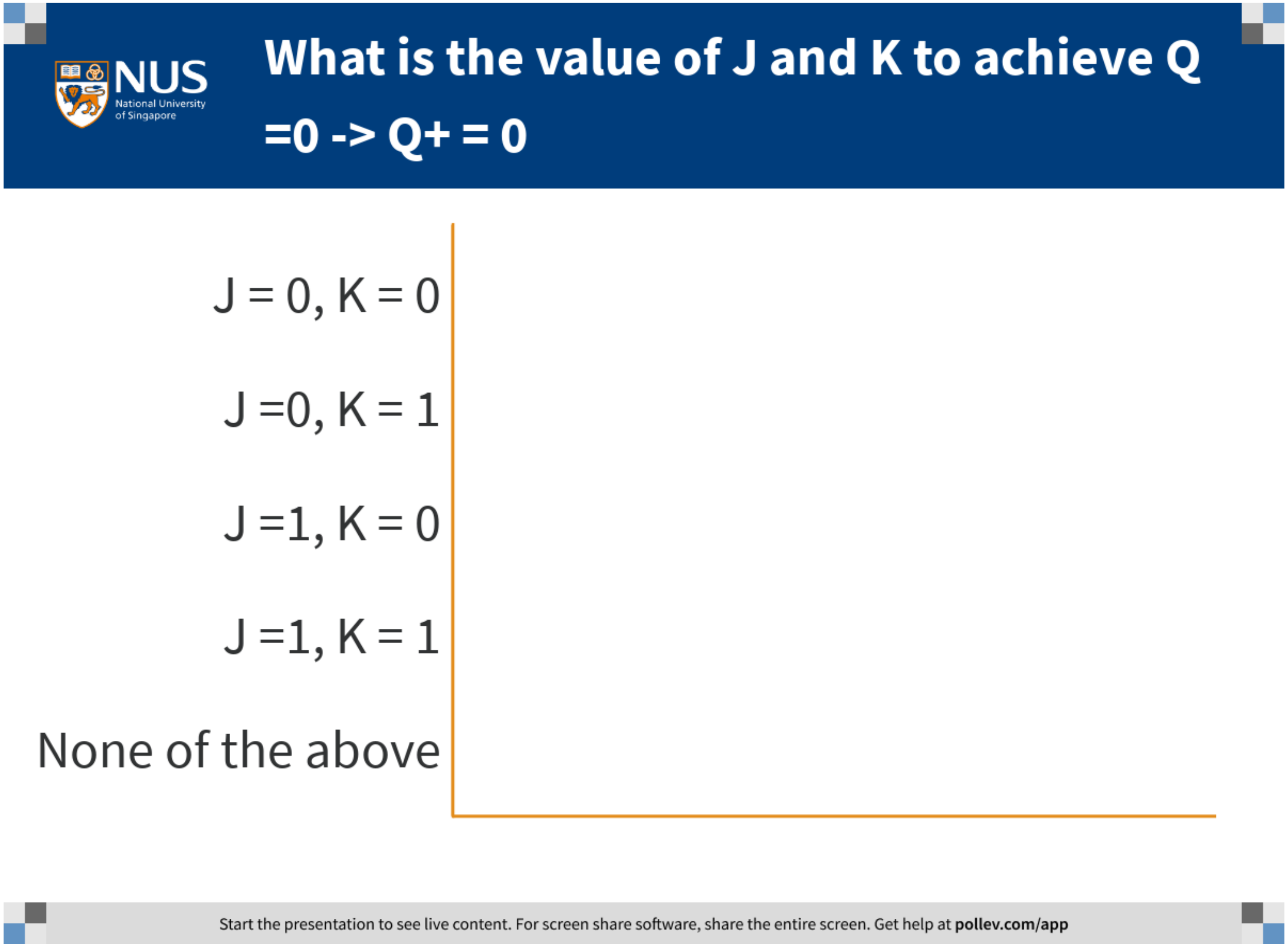
Q	Q^+	D
0	0	0
0	1	1
1	0	0
1	1	1

D ⇔ Q^+

We now have a truth table for the combinational circuit!



Excitation Table



NUS National University of Singapore

What is the value of J and K to achieve Q =0 -> Q+ = 0

J = 0, K = 0

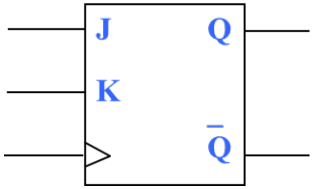
J =0, K = 1

J =1, K = 0

J =1, K = 1

None of the above

Start the presentation to see live content. For screen share software, share the entire screen. Get help at **pollev.com/app**



JK Flip Flop
Excitation Table

Q	Q^+	J	K
0	0	?	?
0	1		
1	0		
1	1		

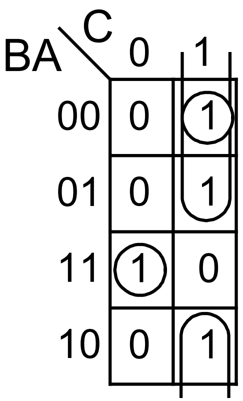
***Excitation Table :** Specifies what the **FF** inputs should be for a specific Q → Q^+ transition to occur.

Design Method – Cont’d

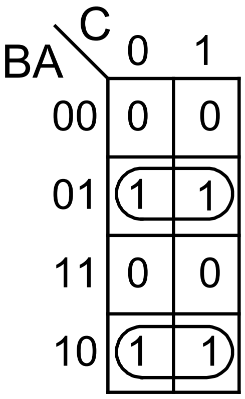
Step 4

- **Realize the circuit.**

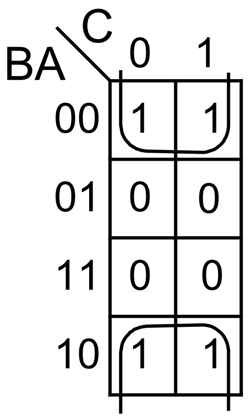
Present-state outputs			Required FF input		
C	B	A	D_C	D_B	D_A
0	0	0	0	0	1
0	0	1	0	1	0
0	1	0	0	1	1
0	1	1	1	0	0
1	0	0	1	0	1
1	0	1	1	1	0
1	1	0	1	1	1
1	1	1	0	0	0



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



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



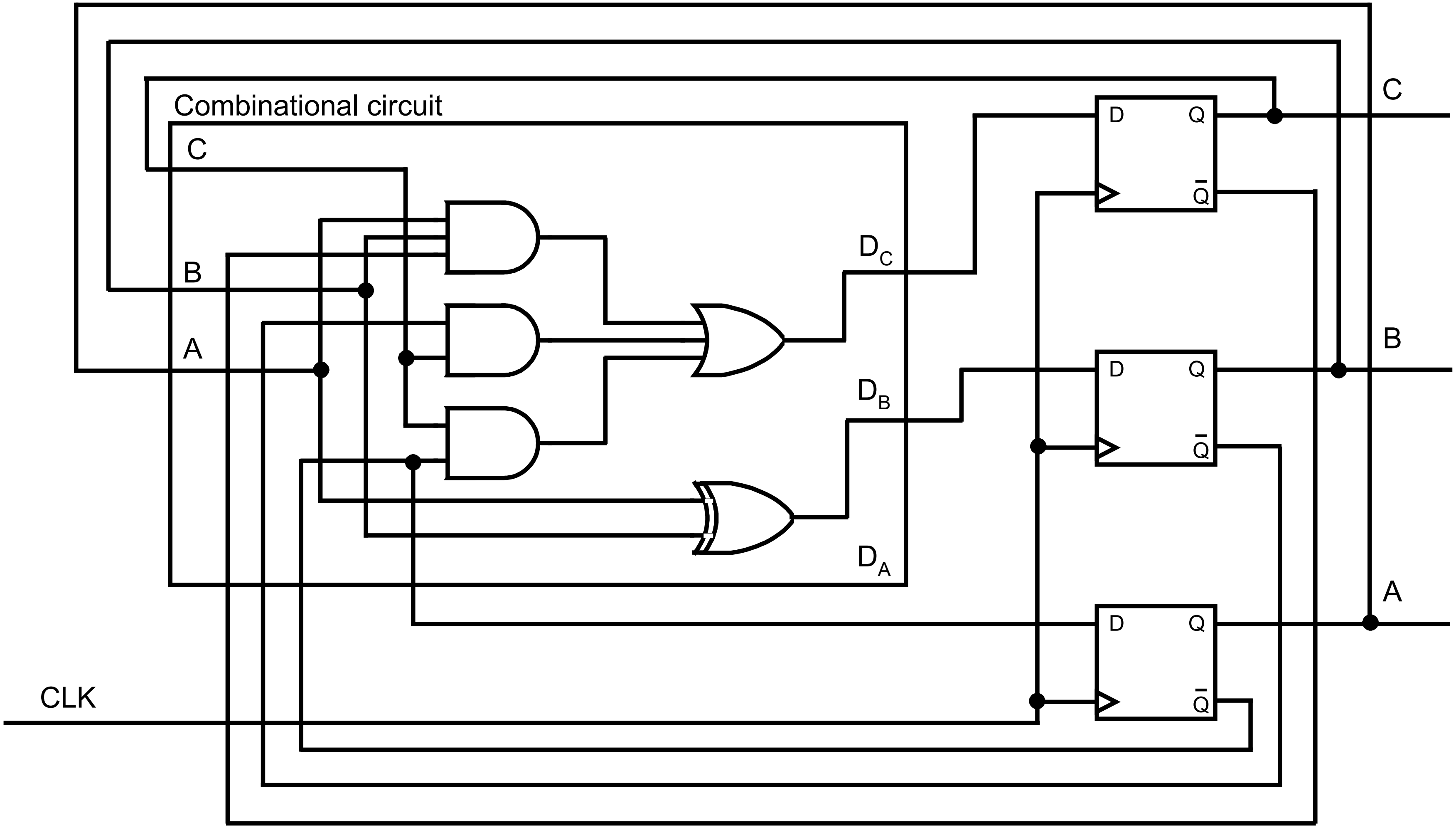
BA \ C	0	1
00	1	1
01	0	0
11	0	0
10	1	1

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

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

$D_A = \overline{A}$

Synchronous 3-bit counter



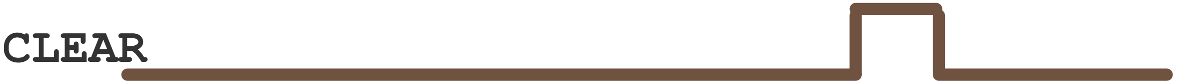
Synchronous Counter Example 2

Design a synchronous counter with count sequence using DFFs:

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

The counter also has an external **active high** synchronous CLEAR input which will clear the counter to 000 at next active clock edge when set to ‘1’.

e.g. 101,001,000,010,110,100,000,010..



1) State Diagram

2) Functional Block Diagram

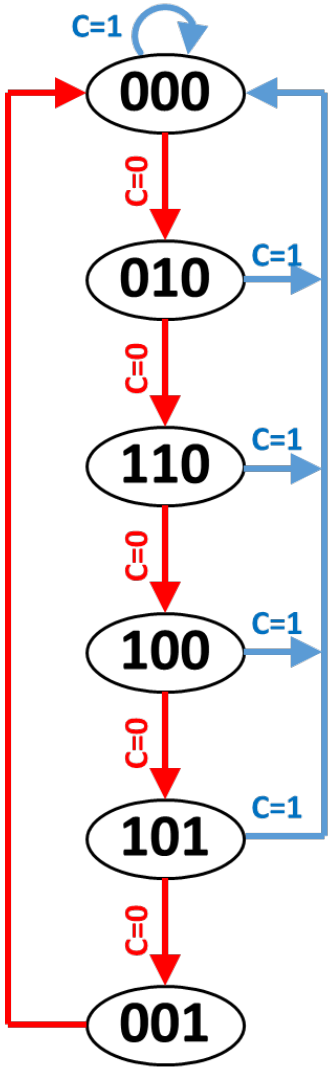
3) Next State Table / Truth table of C.C.

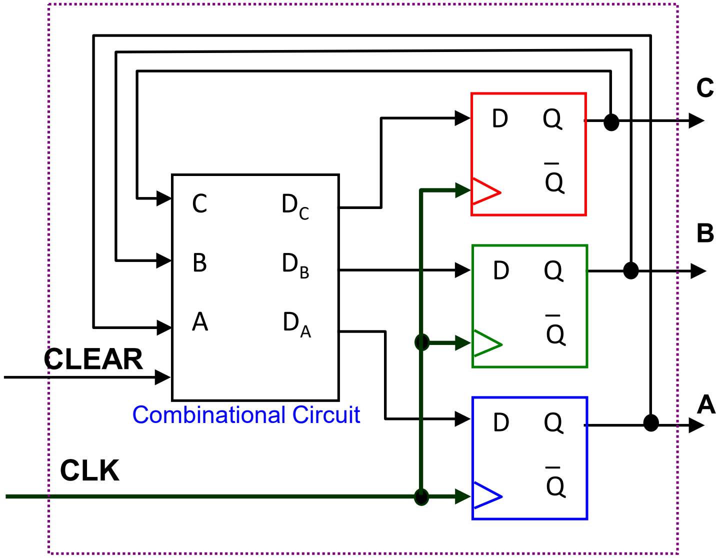
4) Final Implementation

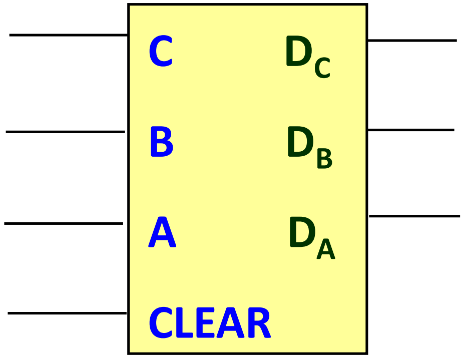
Step1: Write state diagram.

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

MSOP for flip-flop inputs

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

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

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