SEQUENTIAL CIRCUITS - II

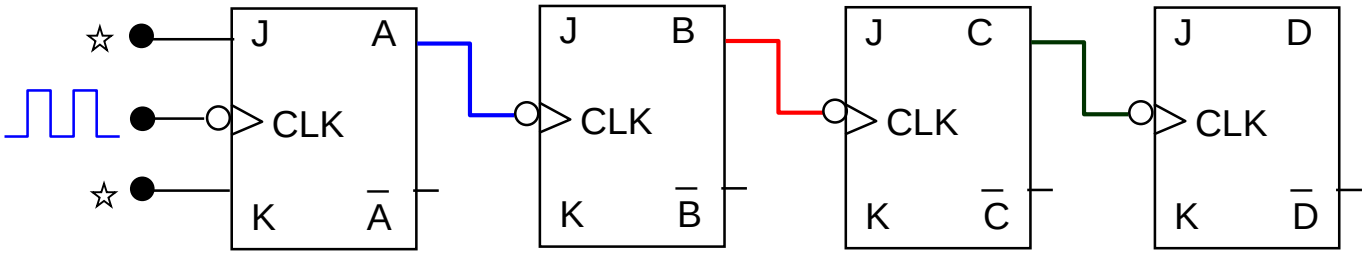
Counters

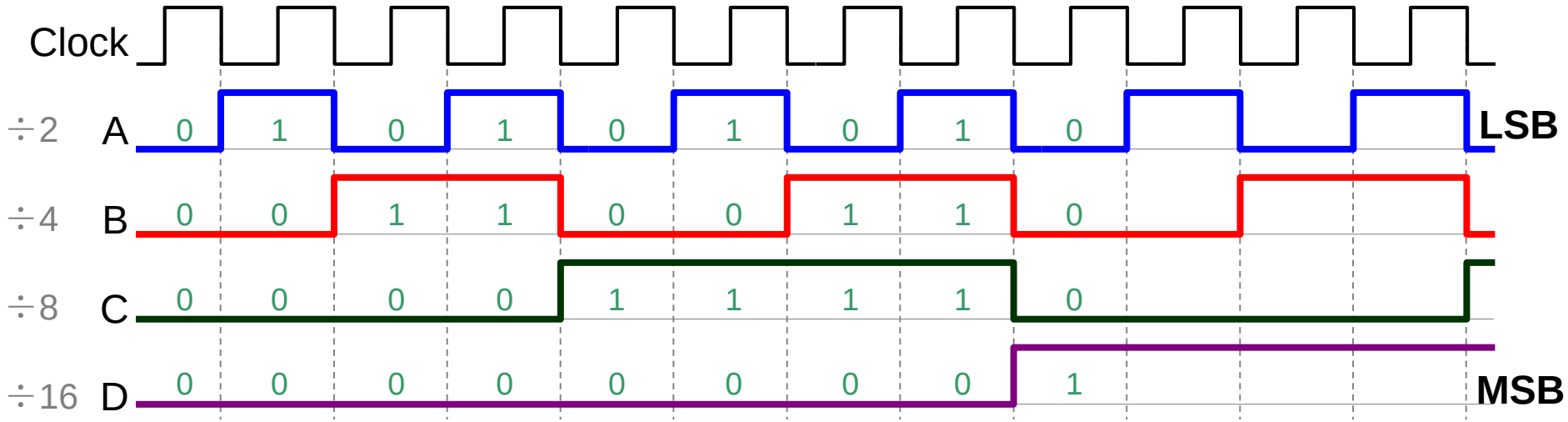
Asynchronous : Circuit elements do not get the clock input simultaneously

Synchronous Counters: Circuit elements get the clock input simultaneously

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

Ripple Counter (Asynchronous):





Counters : Mod - X

- Each FF successively halves the input clock frequency
- The 4-bit counter counts from $0000\ (0) \rightarrow 1111(15)$

➔ 16 distinct count states ⇒ called a mod-16 counter

- *N* x FFs connected this way will have 2^N states ⇒ mod-2^N

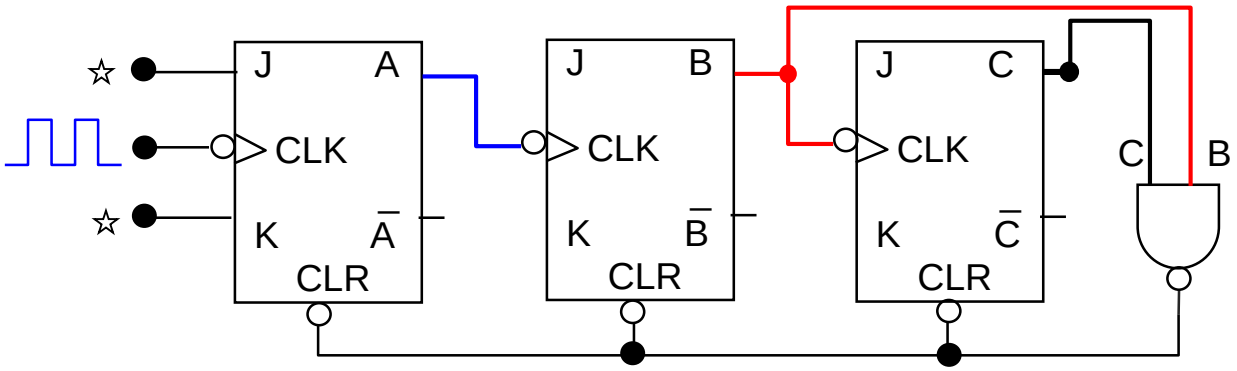
How to obtain a counter with mod-X < 2^N?

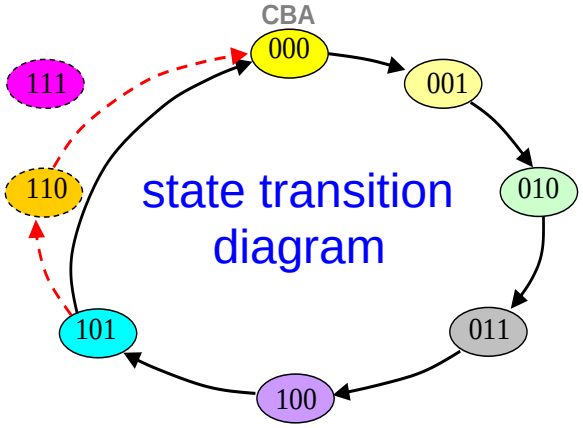
1. Assume counter starts from 0
2. Identify FFs that will be in HIGH state when count = X
3. Feed those FFs outputs to a NAND gate
4. Connect NAND gate output to ***asynchronous*** CLR

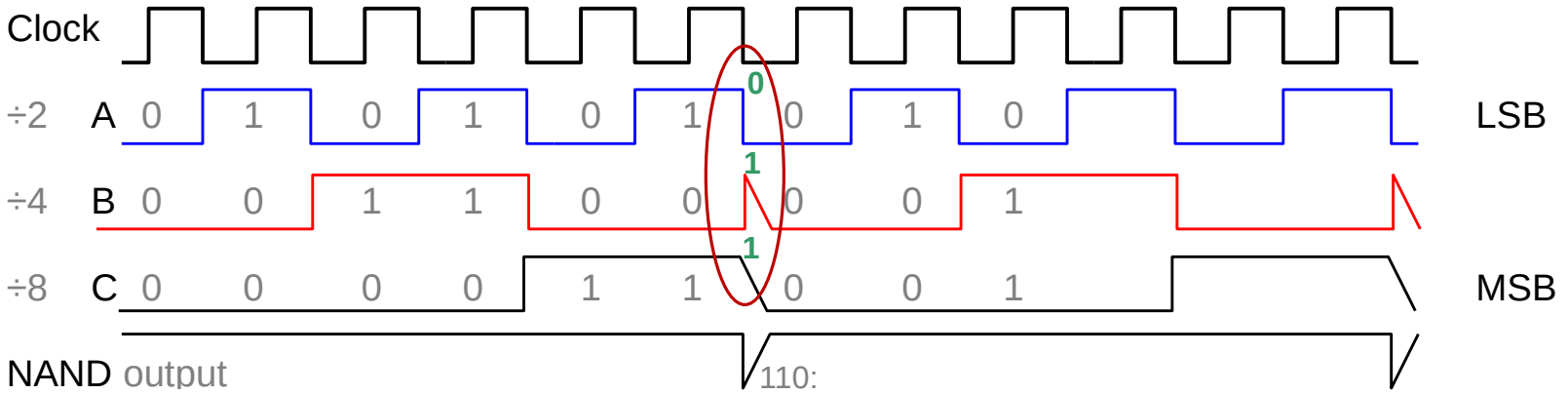
CLK	CLR	J	K	Q^+
X	L	X	X	L
↓	H	L	L	Q
↓	H	L	H	L
↓	H	H	L	H
↓	H	H	H	$\overline{Q}$

Mod-6 Asynchronous Counter

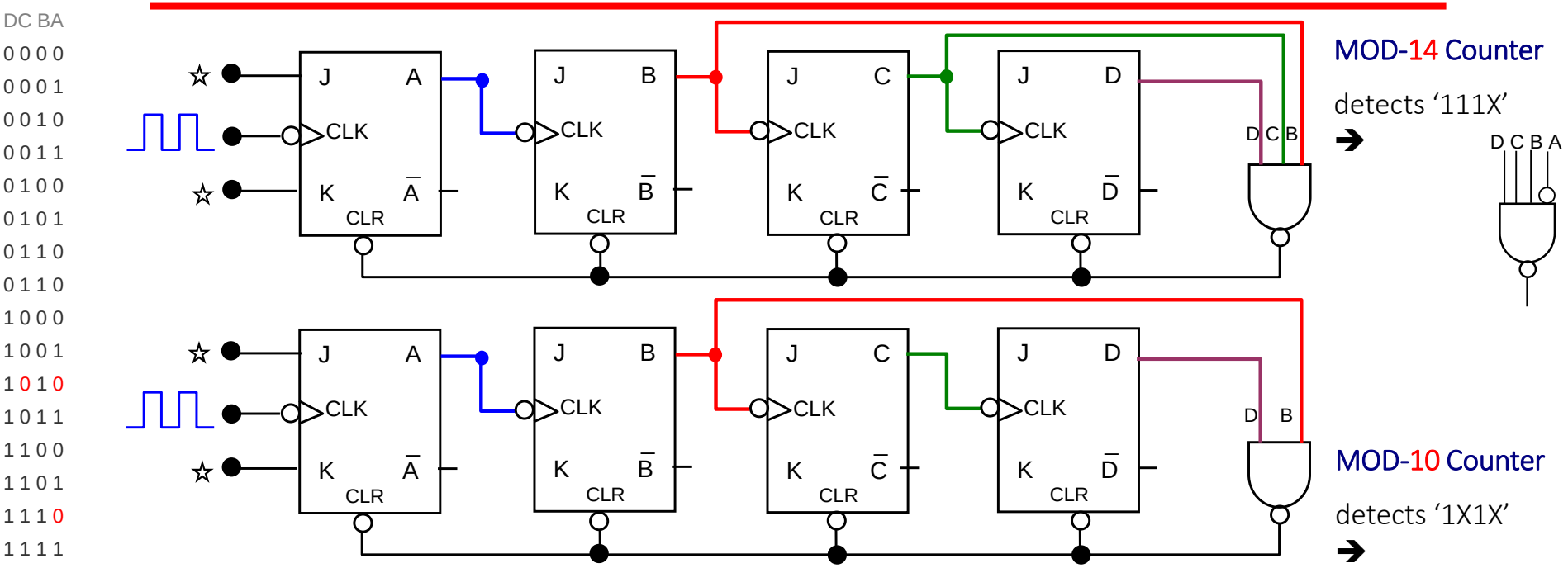
1) How many FFs ? _3_ **2) Clear at output = ? _6_**







Mod-? Counters

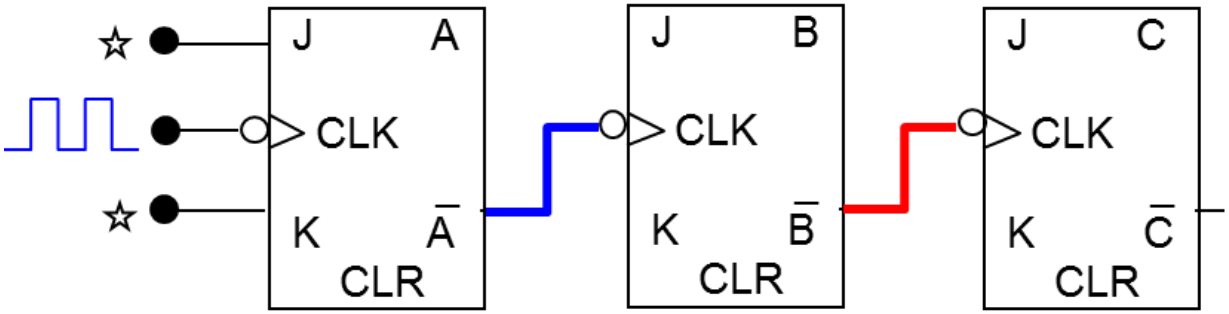


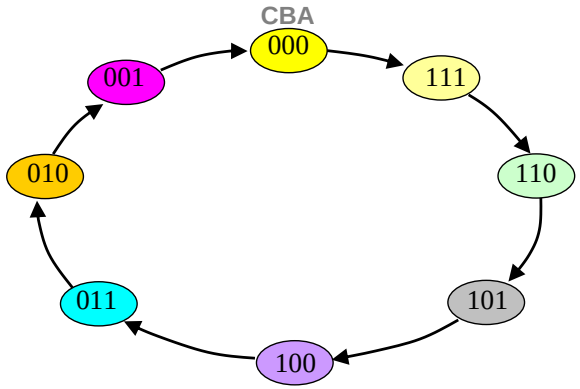
Decade / BCD counter: counts up in ordinary binary sequence
0000 → 1001

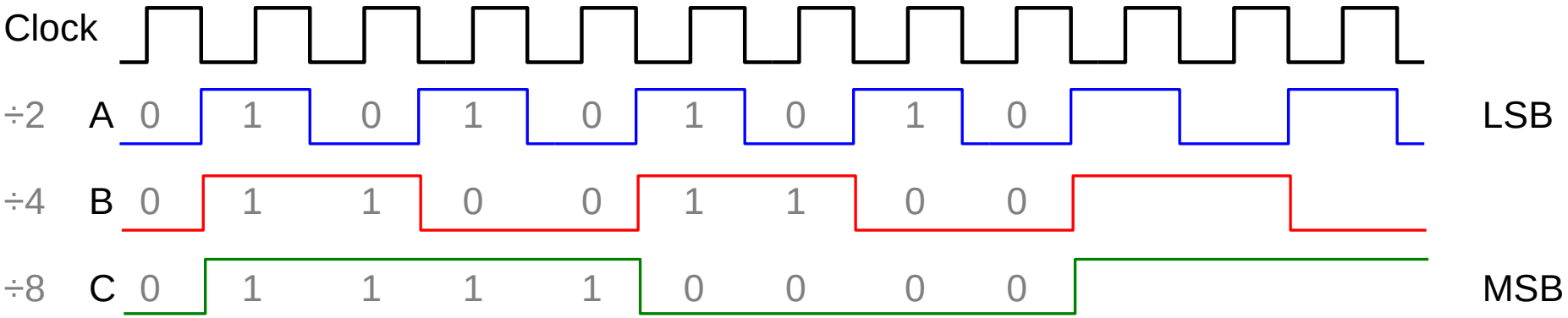
Ripple counters considered so far, counted up↑.
How to make them count down↓?

Count Down Ripple Counter…

To count down : connect complements of FF outputs to clock inputs of succeeding FFs

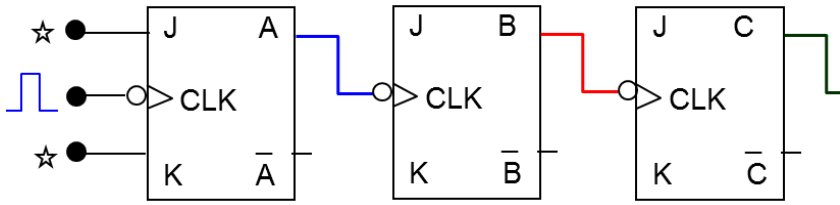


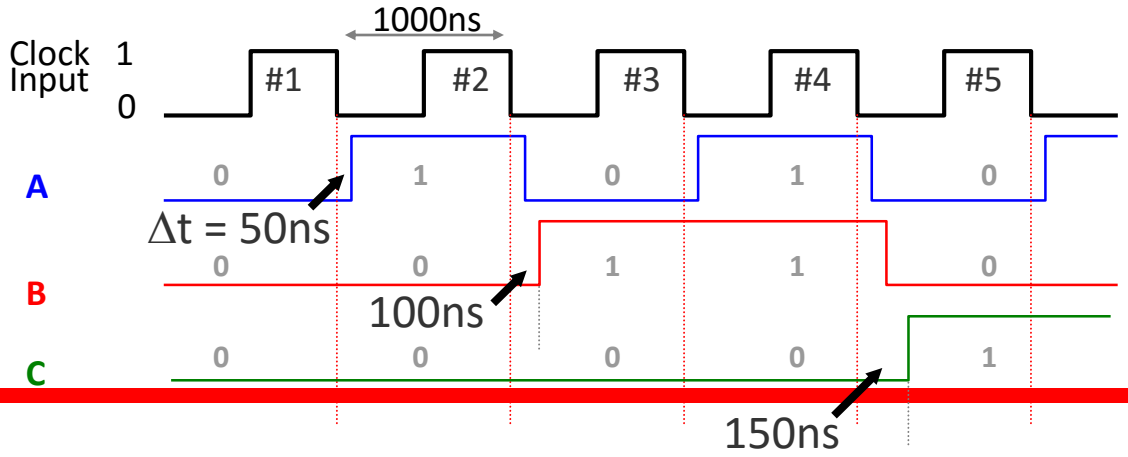




t_{pd} ➔ limiting frequency

- 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 $n* \Delta tpd$ after t_0

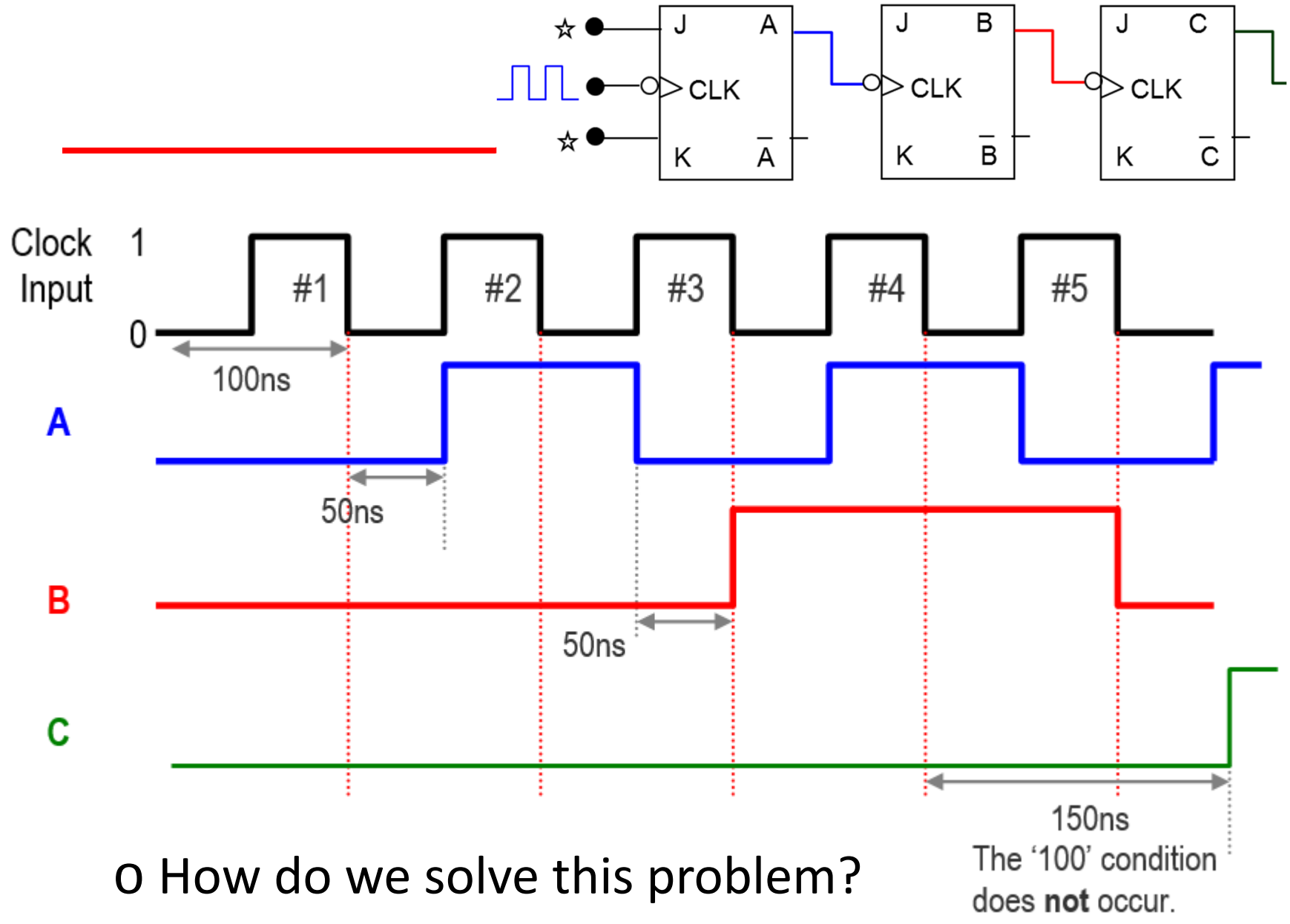




For proper operation:

$Tclock \geq n*\Delta tpd$

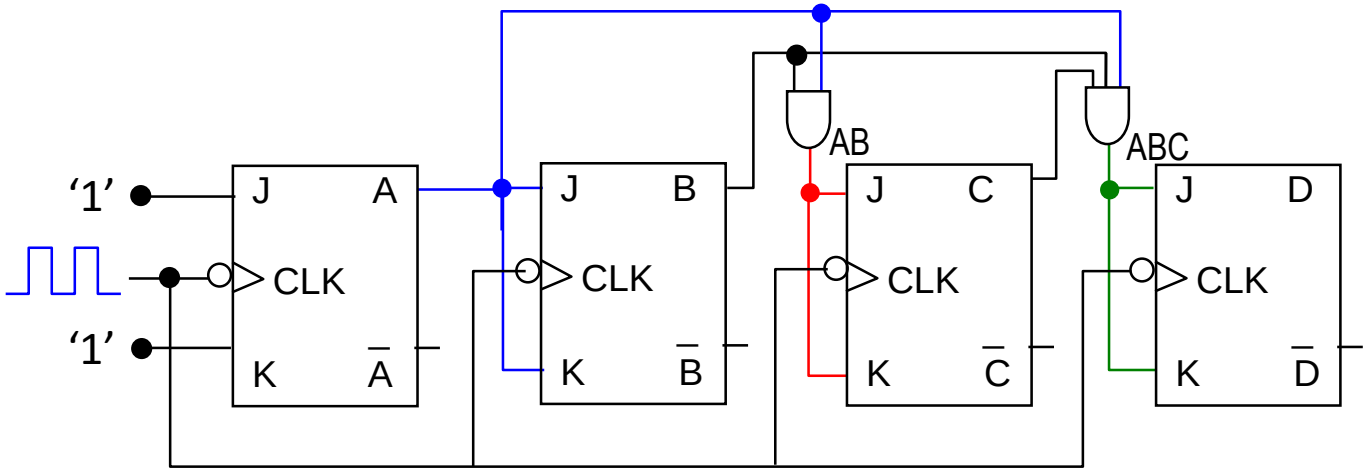
$fmax \leq 1 / n*\Delta tpd$



- How do we solve this problem?

Synchronous (Parallel) Counters

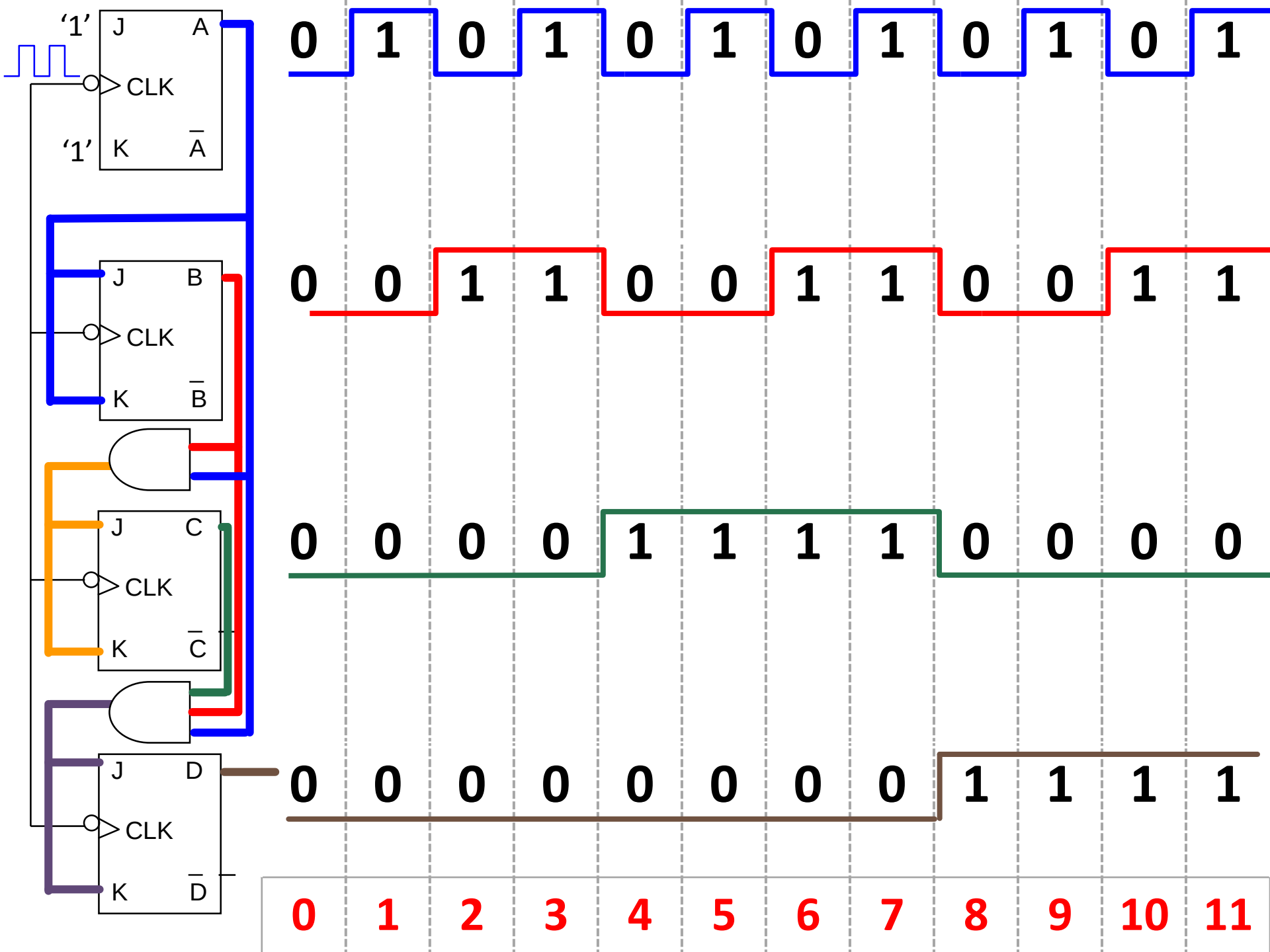
Mod-16 Synchronous Parallel Counter :

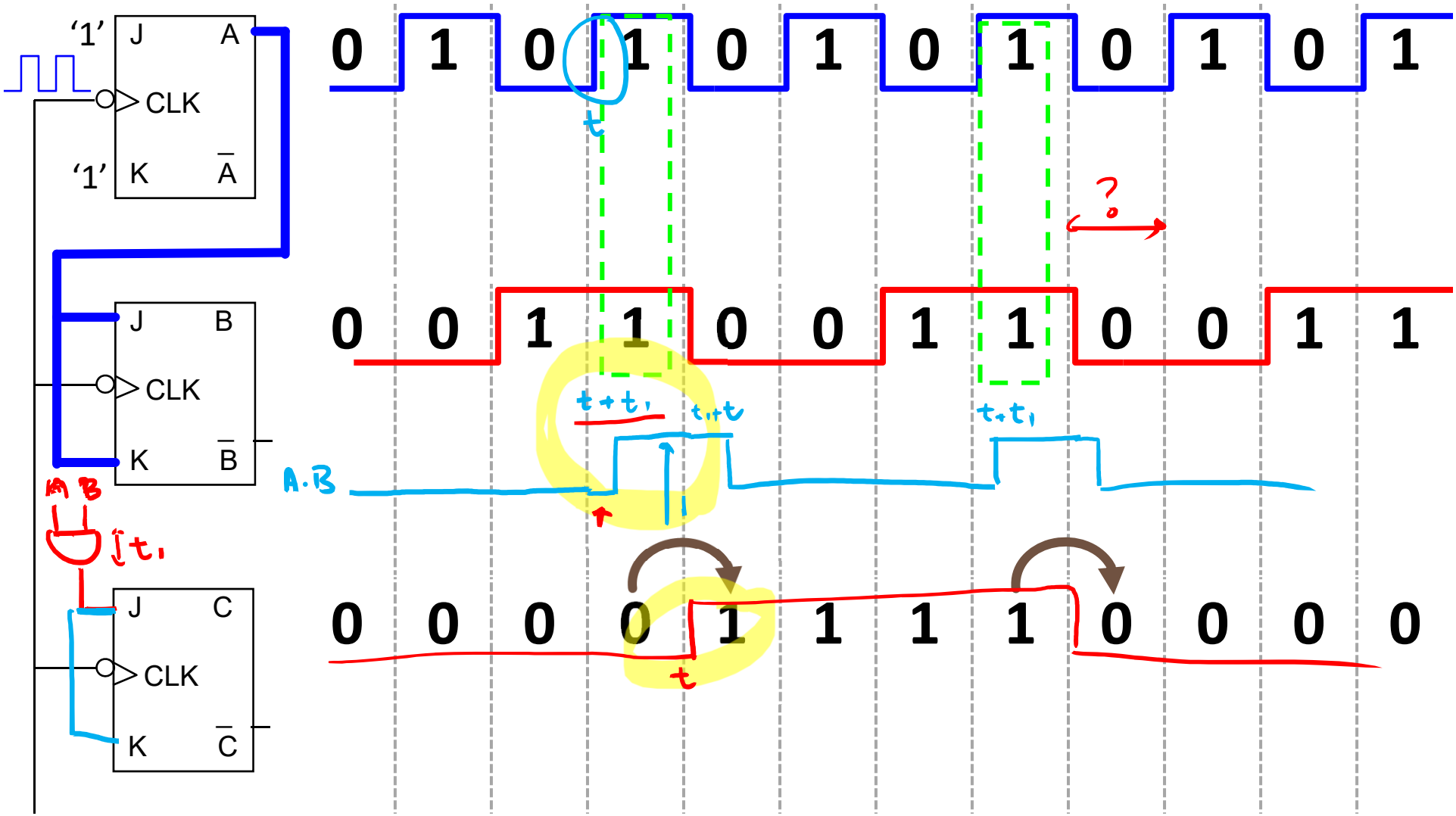


B toggles when A = 1
C toggles when AB = 1
D toggles when ABC = 1

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

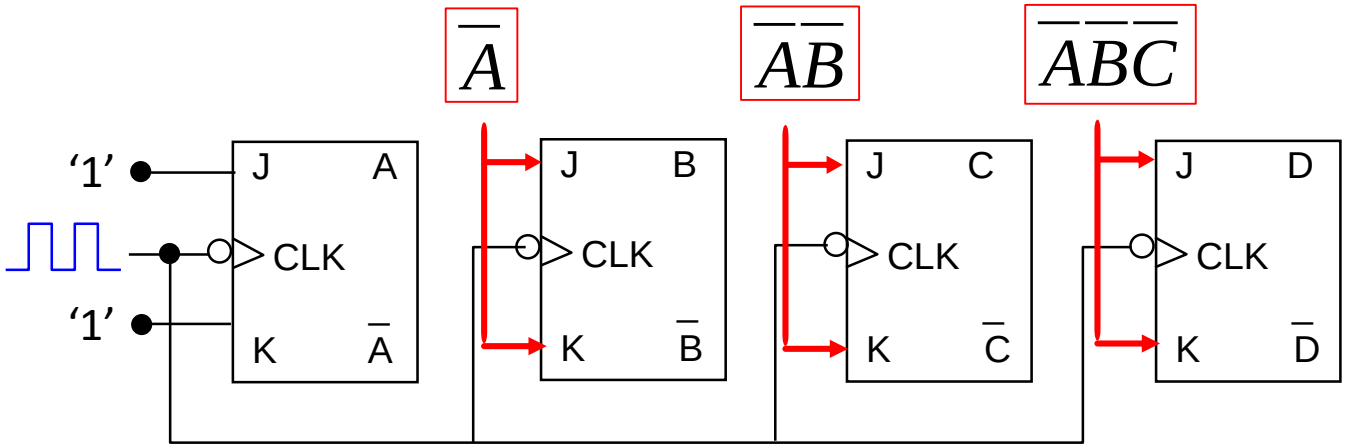
- Since all FFs are triggered simultaneously by the clock, this counter can operate at higher frequencies.
- Total delay = Δtpd (FF) + Δtpd (AND)
- Operating frequency is irrespective of the number of FFs
- Disadvantages? – more gates!


‘1’
J
A
CLK
‘1’
K
$\overline{A}$
J
B
CLK
K
$\overline{B}$
J
C
CLK
K
$\overline{C}$
J
D
CLK
K
$\overline{D}$
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


'1'
J
A
CLK
'1'
K
$\overline{A}$
J
B
CLK
K
$\overline{B}$
A·B
J
C
CLK
K
$\overline{C}$
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
t
?
t+t1
t1+t
t+t1
t

Counting Down…

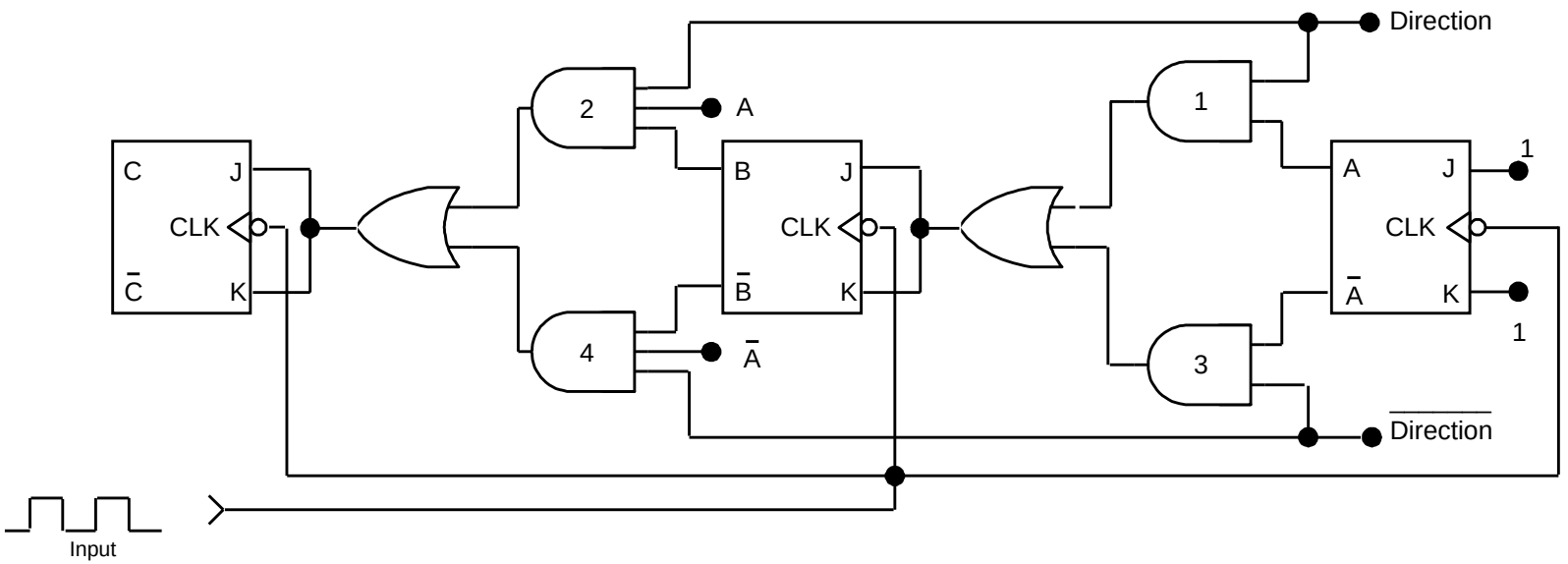
What about a count down mod-16 counter ?



B toggles when A=0
C toggles when AB=0
D toggles when ABC=0

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

  $J,K_{FFB} = A$

  $J,K_{FFC} = A.B$

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

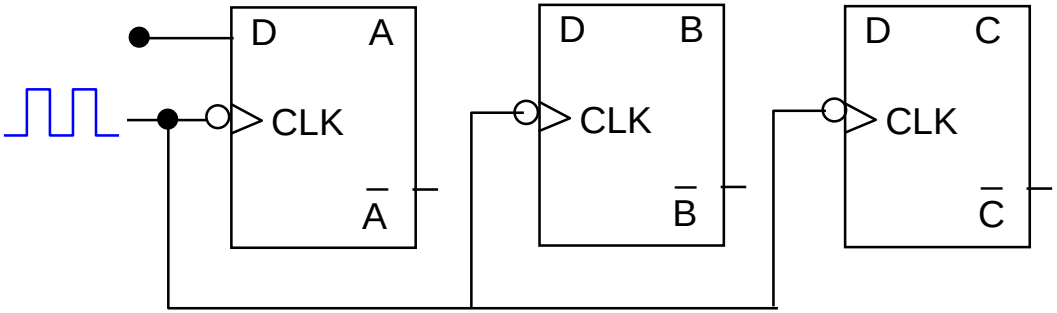
  $J,K_{FFB} = \bar{A}$

  $J,K_{FFC} = \bar{A}.\bar{B}$

Example (D Flip-Flops)

CLK	D	Q^+
↑	0	0
↑	1	1

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

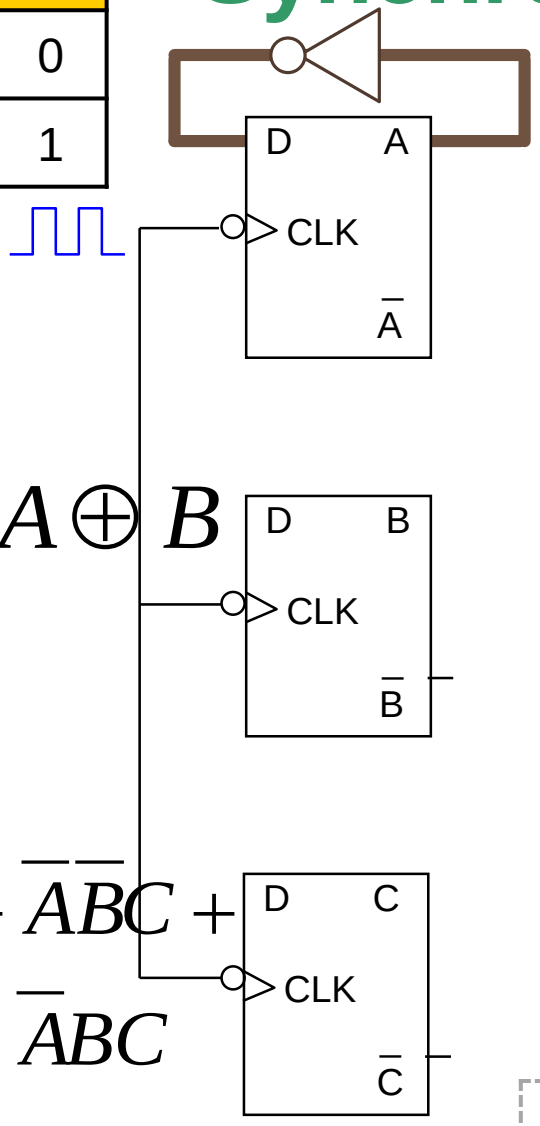
$$D_A = \overline{A}$$

$$D_B = A \oplus B$$

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

Synchronous Counters (DFF)

CLK	D	Q^+
↑	0	0
↑	1	1



0	1	0	1	0	1	0	1	0
	$A\overline{B}$	$\overline{A}B$						
0	0	1	1	0	0	1	1	0
			$AB\overline{C}$		$A\overline{B}C$			
0	0	0	0	1	1	1	1	0
				$\overline{A}\overline{B}C$		$\overline{A}BC$		
0	1	2	3	4	5	6	7	0

Verilog!

- 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 ?