Counters

(9)

Sequential circuit that follows a sequence of distinct states under the control of input clock pulses.

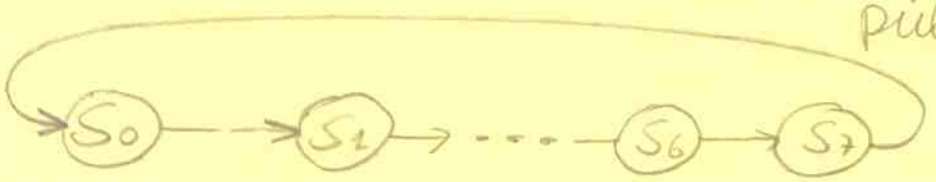
Ripple counters Vs. Synchronous counters

up counter, down counter

Modulo-M counter counts up to M pulses

Natural modulus counter counts $M = 2^N$ pulses and requires $N = \log_2 M$ FF's for implementation, known as N-bit counter.

Example - Mod-8 counter - Synchronous since pulses are entered to CLK inputs of all FF's



States can be stored in negative edge-triggered T FF's

in Negative edge TFF clock is entered with negation

Q(t)	Q(t+1)	T
0	0	0
0	1	1
1	0	1
1	1	0

Excitation Table

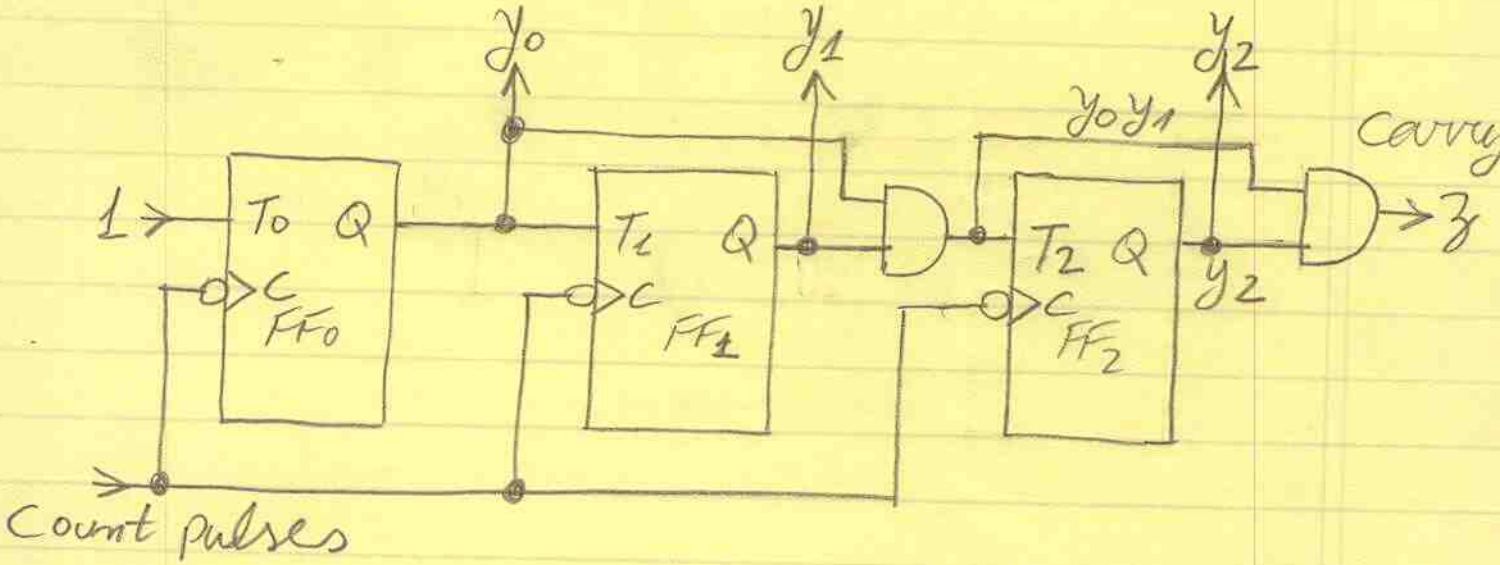
PS	y_2	y_1	y_0	NS	Y_2	Y_1	Y_0	T_2	T_1	T_0
S_0	0	0	0	S_1	0	0	1	0	0	1
S_1	0	0	1	S_2	0	1	0	0	1	1
S_2	0	1	0	S_3	0	1	1	0	0	1
S_3	0	1	1	S_4	1	0	0	1	1	1
S_4	1	0	0	S_5	1	0	1	0	0	1
S_5	1	0	1	S_6	1	1	0	0	1	1
S_6	1	1	0	S_7	1	1	1	0	0	1
S_7	1	1	1	S_0	0	0	0	1	1	1

With the aid of K-map ($y_2 y_1 y_0$) we obtain

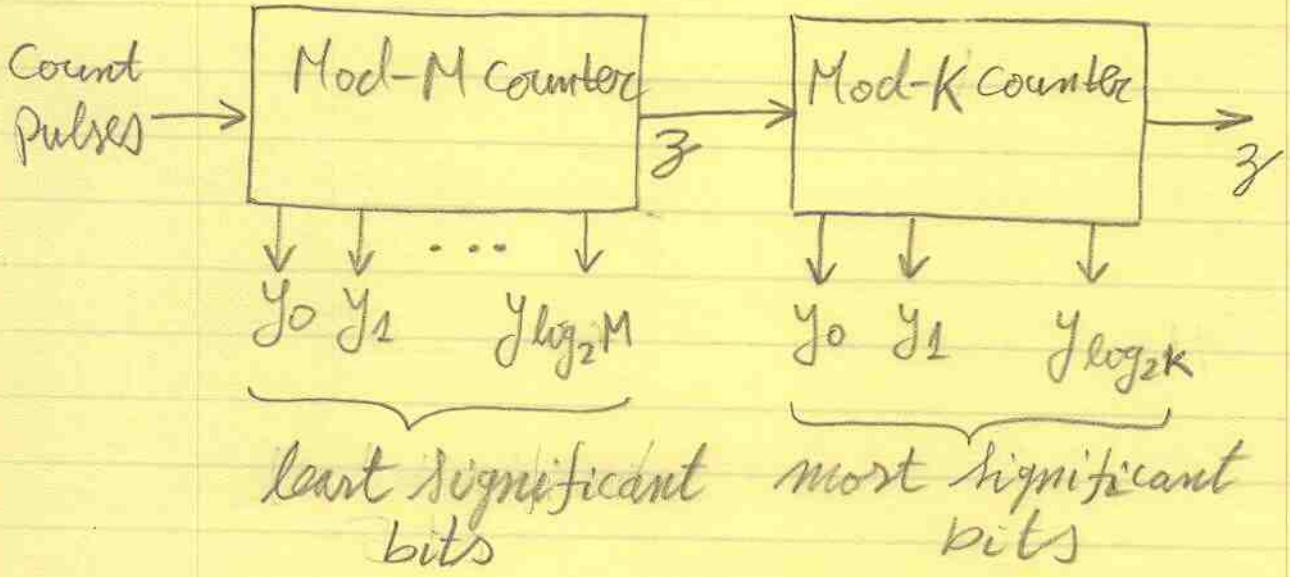
$T_2 = Y_1 Y_0 \quad T_1 = Y_0 \quad T_0 = 1$

count output carries

$Z = y_1 y_2 y_3$ - Indication on 8 counts



Implementing Mod-MK counter



Example: decimal counter with JK FF's

- We use BCD state assignment

Q(t)	Q(t+1)	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

excitation table

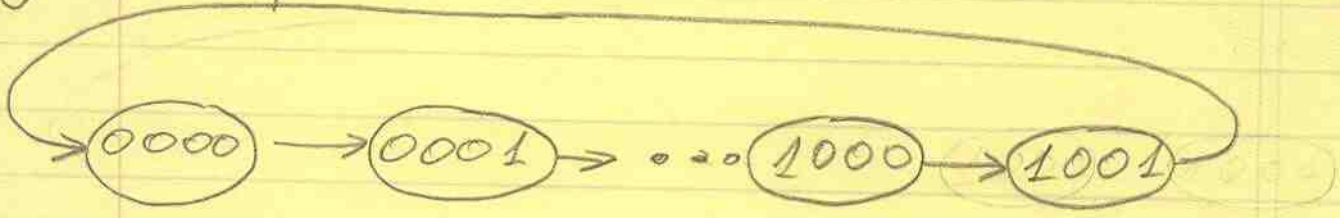
PS				NS				
y_3	y_2	y_1	y_0	Y_3	Y_2	Y_1	Y_0	$J_3 K_3 \cdots J_0 K_0$
0	0	0	0	0	0	0	1	
0	0	0	1	0	0	1	0	
0	0	1	0	0	0	1	1	
0	0	1	1	0	1	0	0	
0	1	0	0	0	1	0	1	
0	1	0	1	0	1	1	0	
0	1	1	0	0	1	1	1	
0	1	1	1	1	0	0	0	
1	0	0	0	1	0	0	1	
1	0	0	1	0	0	0	0	

We create now a table of J_i K_i by replacing the Y_i with their corresponding implied JK excitation values.

For instance, $(y_0, Y_0) = (0,1)$ results $(J_0, K_0) = (1, X)$, or $(y_3, Y_3) = (1, 0)$ results $(J_3, K_3) = (X, 1)$. We then obtain the table for all J_i and K_i

y_3	y_2	y_1	y_0	J_3	K_3	J_2	K_2	J_1	K_1	J_0	K_0
0	0	0	0	0	X	0	X	0	X	1	X
⋮							⋮				
1	0	0	1	X	1	0	X	0	X	X	1

And then every J_i and K_i equation is obtained by K-map



$J_0 = K_0 = 1$

$J_1 = y_3' y_0$, $K_1 = y_0$

$J_2 = K_2 = y_1 y_0$ $\qquad Z = y_3 y_0$

$J_3 = y_2 y_1 y_0$, $K_3 = y_0$

Important: Need to check whether this is self-starting (code) state assignment.

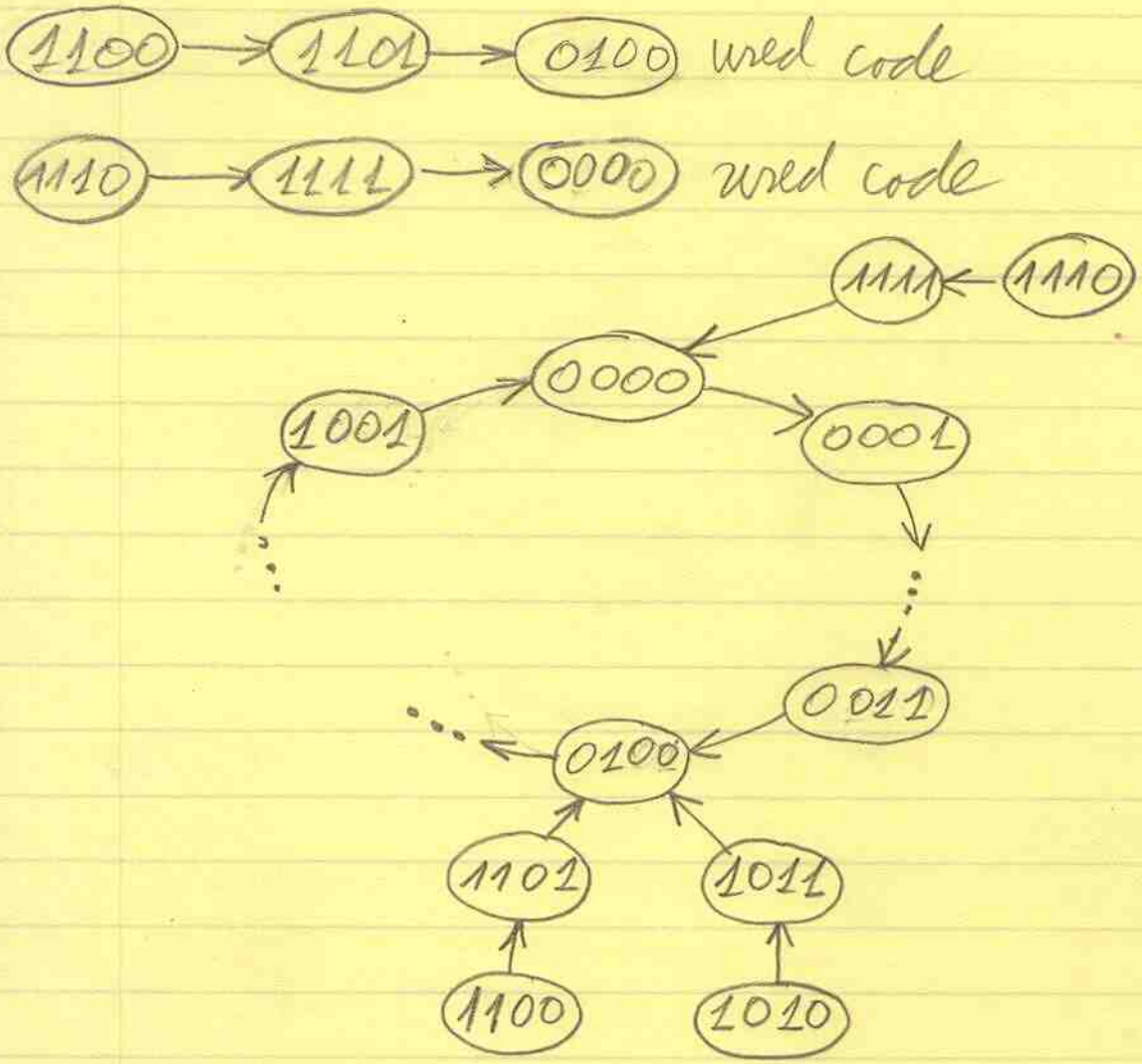
① We need to set $y_3 y_2 y_1 y_0$ all unused codes (circuit can start with arbitrary values of y_i's)

② For each unused code calculate all J_i and K_i to find Y_i $i=0,1,2,3$ and then derive $Y_3 Y_2 Y_1 Y_0$

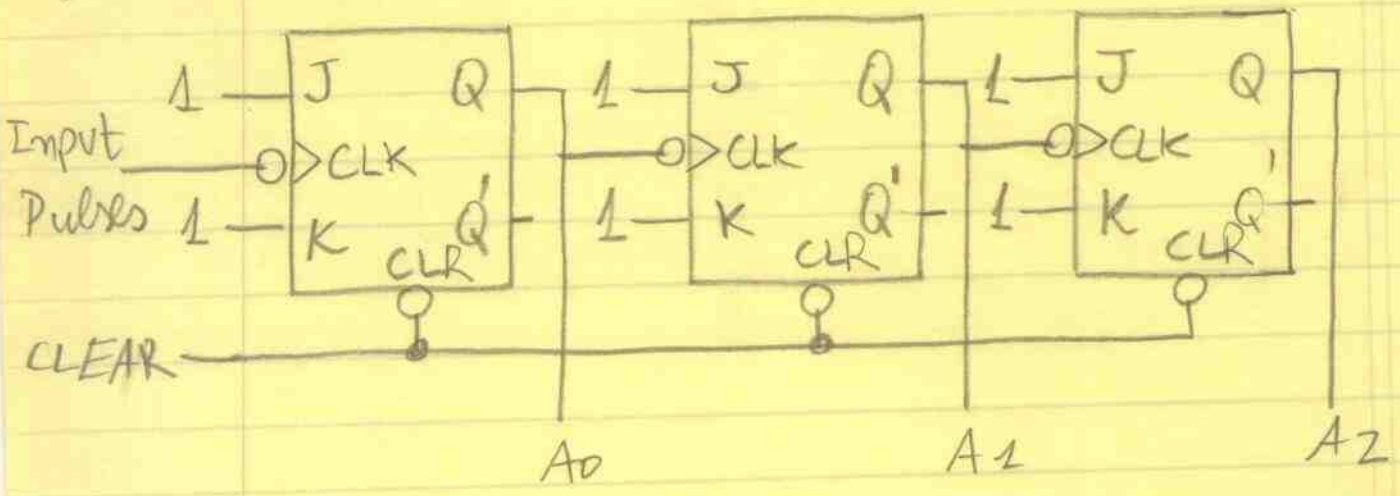
③ If $Y_3 Y_2 Y_1 Y_0$ correspond to any assigned code we are done. Otherwise process repeats. It must reach to a state code then state assignment needs take all codes and create transition to defined states.

(1010) → use J,K equations → (1011) → (0100)

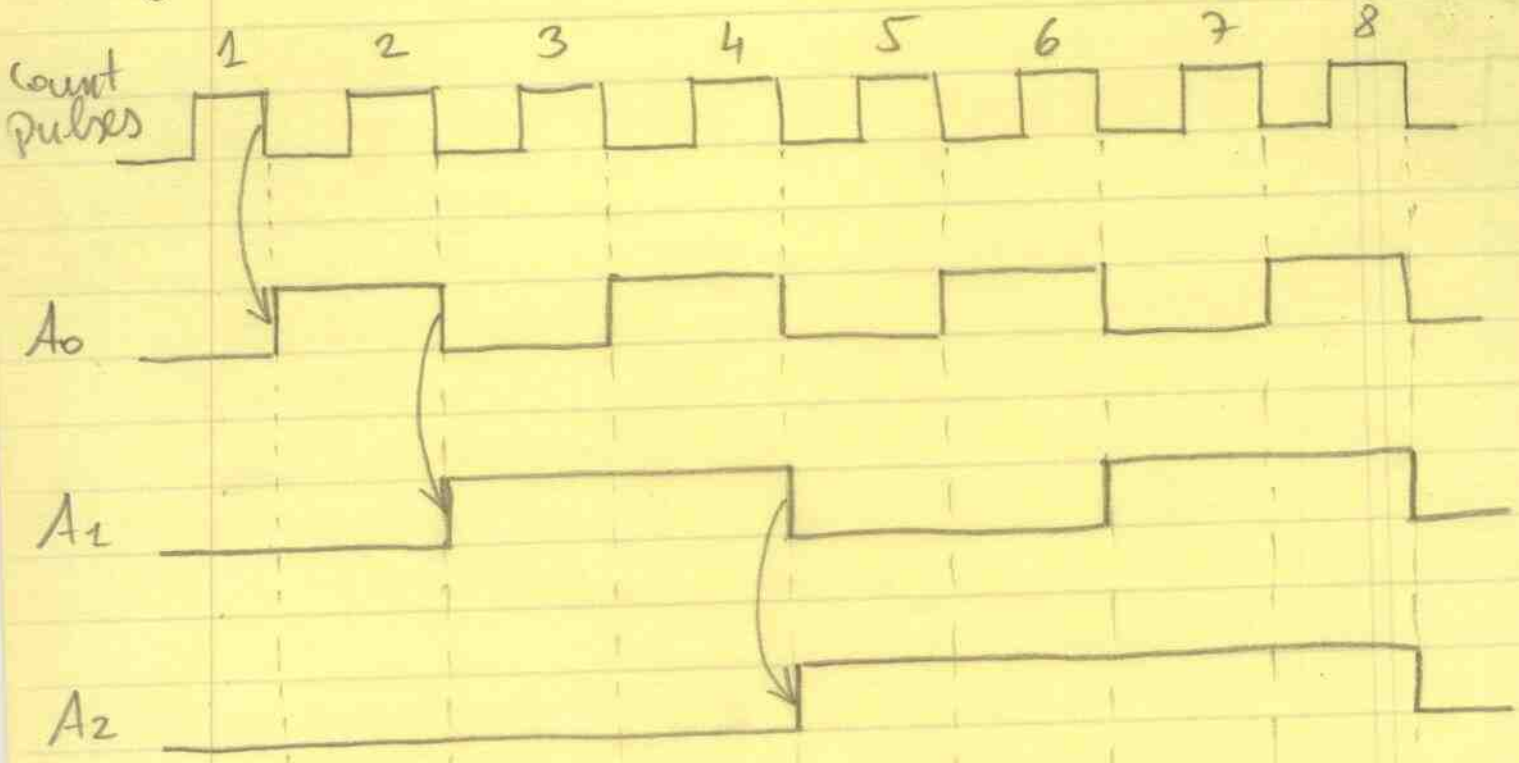
unused code — unused code — used code


1100
1101
0100
used code
1110
1111
0000
used code
1111
1110
0000
1001
0001
0011
0100
1101
1011
1100
1010

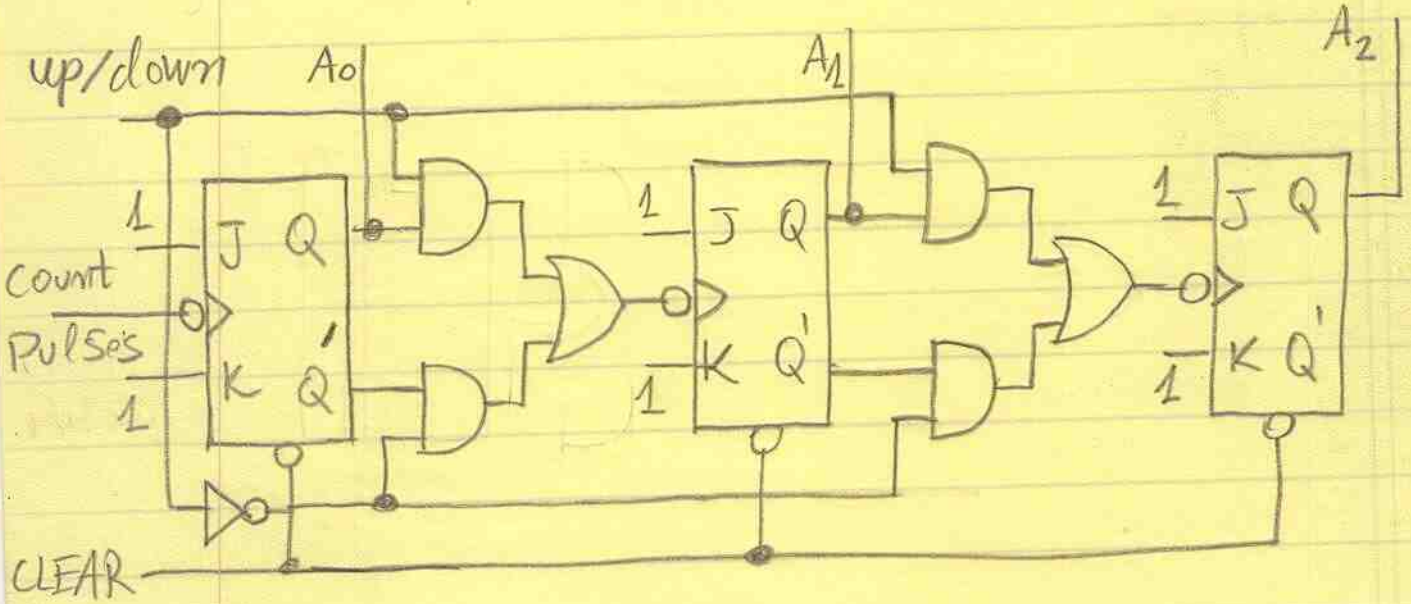
Ripple Counters



Counter is cleared (asynchronously), all A's 0
all J's and K's are permanently 1, so any
negative edge of clock will complement Q



- This is called **mod-M** ($M=8$) counter and also **divide-by-M** counter
- This is an **up-counter**, counting from 0 to 7. Looking at Q' it is a **down-counter**
- Here is a ripple **up/down** counter



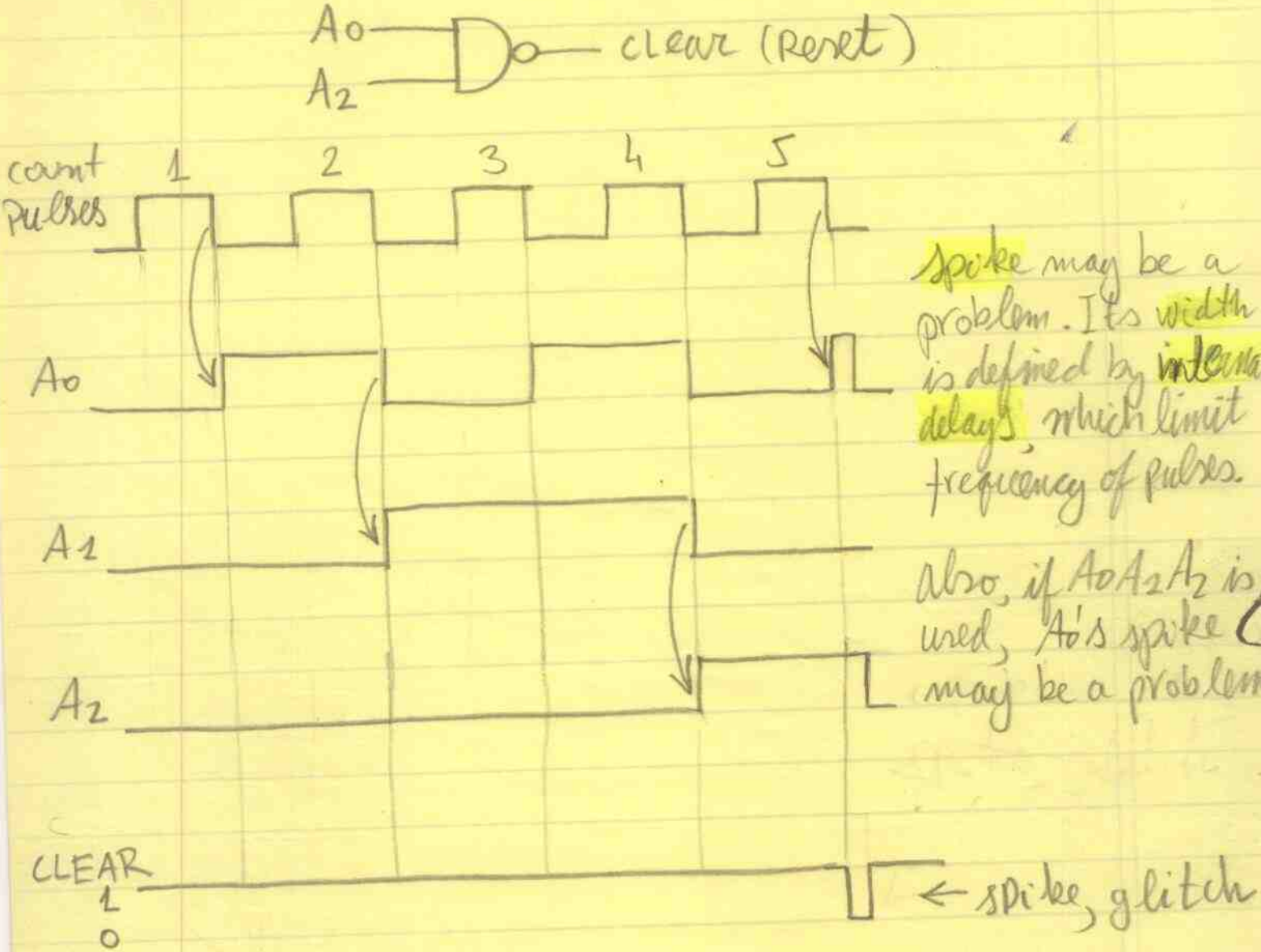
when $U/D=1$ it is up-counter

when $U/D=0$ it is down-counter

Ripple Counters with arbitrary Modulus

The idea is to generate a **clear (reset)** pulse upon the occurance of M at $A_0 A_1 \cdots$

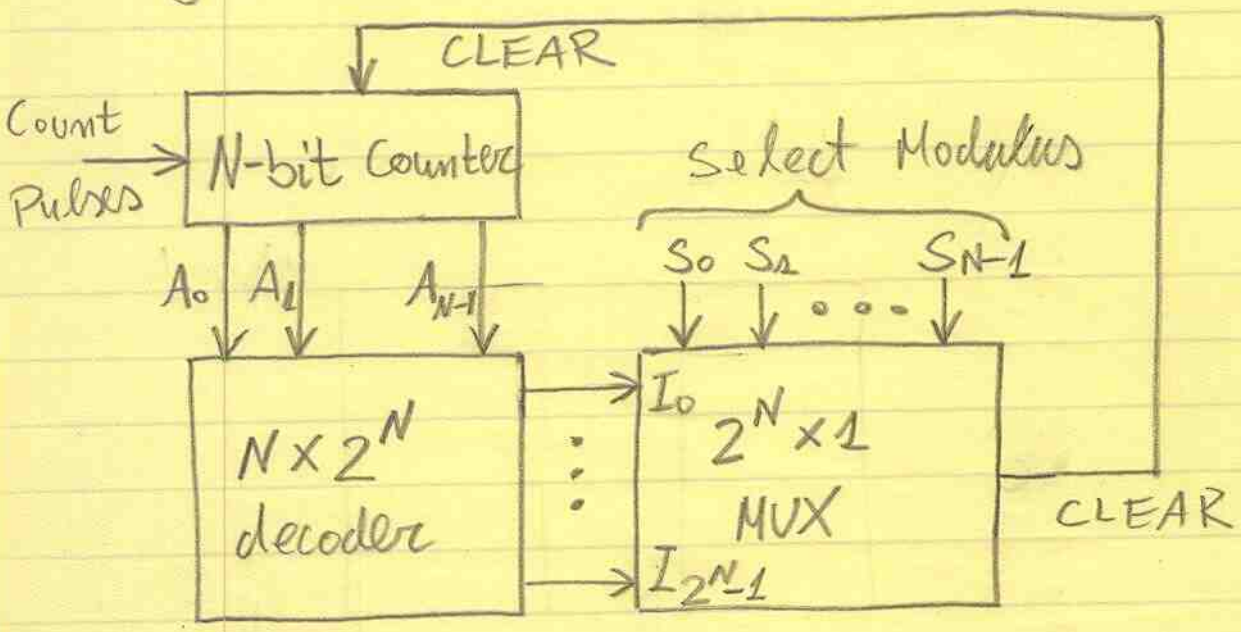
which will then reset the counter



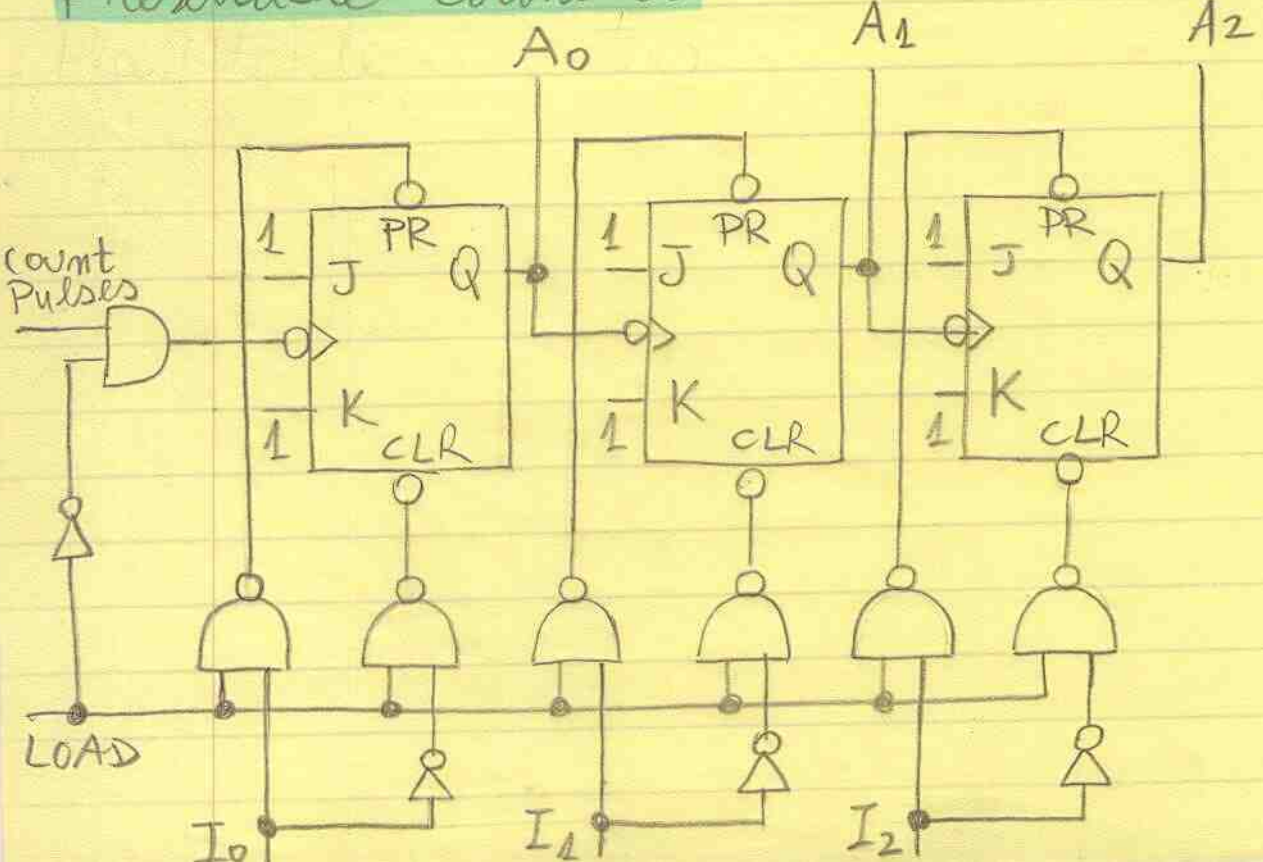
Conditions of clear to change from 1 to 0 occur at 5th pulse falling edge, where $A_0 A_1 A_2 = 101$, CLEAR=0 which then resets $A_0 A_1 A_2 = 000$ AND CLEAR=1

Selectable Modulo Counter

An N-bit counter can be "programmed" to any $M \leq 2^N$ modulus as follows



Presettable Counters



Sometimes we'd like an M-counter to count up from K to M or down from M to K, $0 \leq K \leq M-1$. This is accomplished by parallel load of K to the preset asynchronous input of the FF. Let $K = I_2 I_1 I_0$, $I_i = 0,1$. Once LOAD=1, Preset inputs will receive 0, while clear input = 1, hence $Q_i = 1$. If $I_i = 0$, preset = 1 and clear = 0, hence $Q_i = 0$ at presetting time count pulses are inhibited when Load = 0, counter is operating, starting count from $I_2 I_1 I_0$.

Propagation delay in ripple counters

Ripple counters are simple to construct but suffer from propagation delay proportional

to their sink. Recall that most significant bit switches on negative edge of all other FF's, but FFi switches only after FFi-1. Therefore, the minimum time interval between successive pulses is

$$T_C \geq N \cdot t_p$$

In today's VLSI technologies $t_p = 100\,p\$ec = 0.1\,n\$ec = 10^{-10}\,Sec$. Assume 4-bit counter

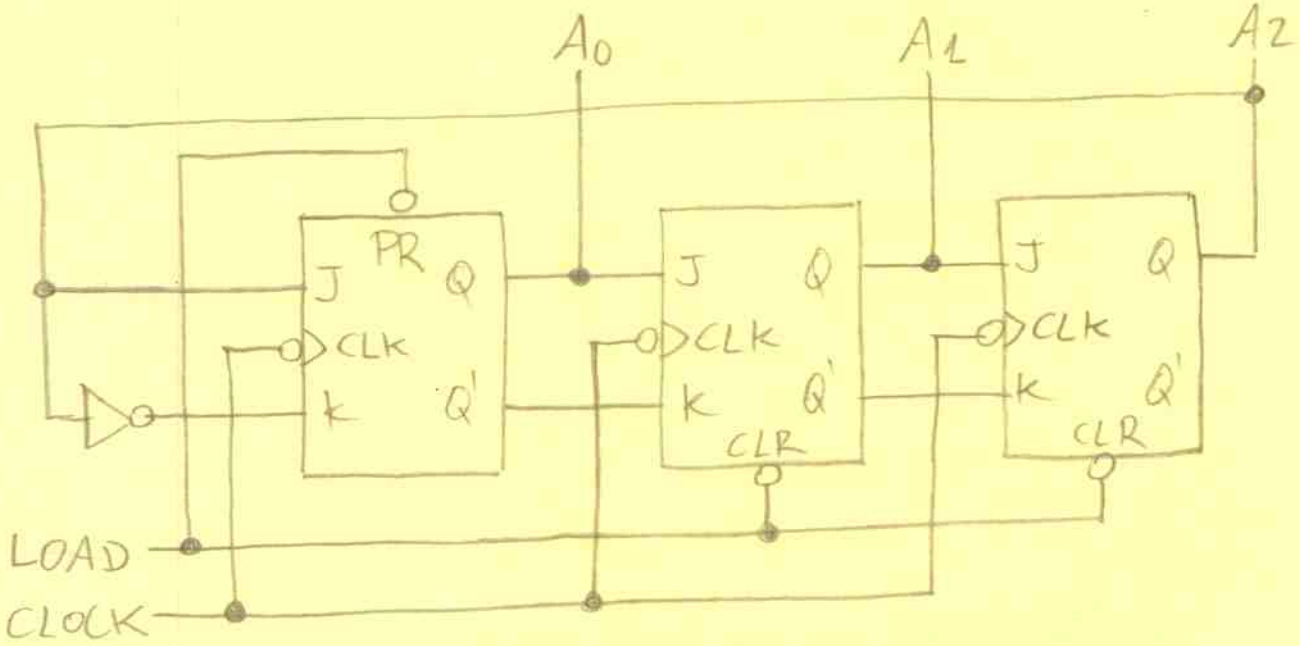
$$f_{max} = \frac{1}{T_C} = \frac{1}{10^{-10} \times 4} = 2.5\ GHz$$

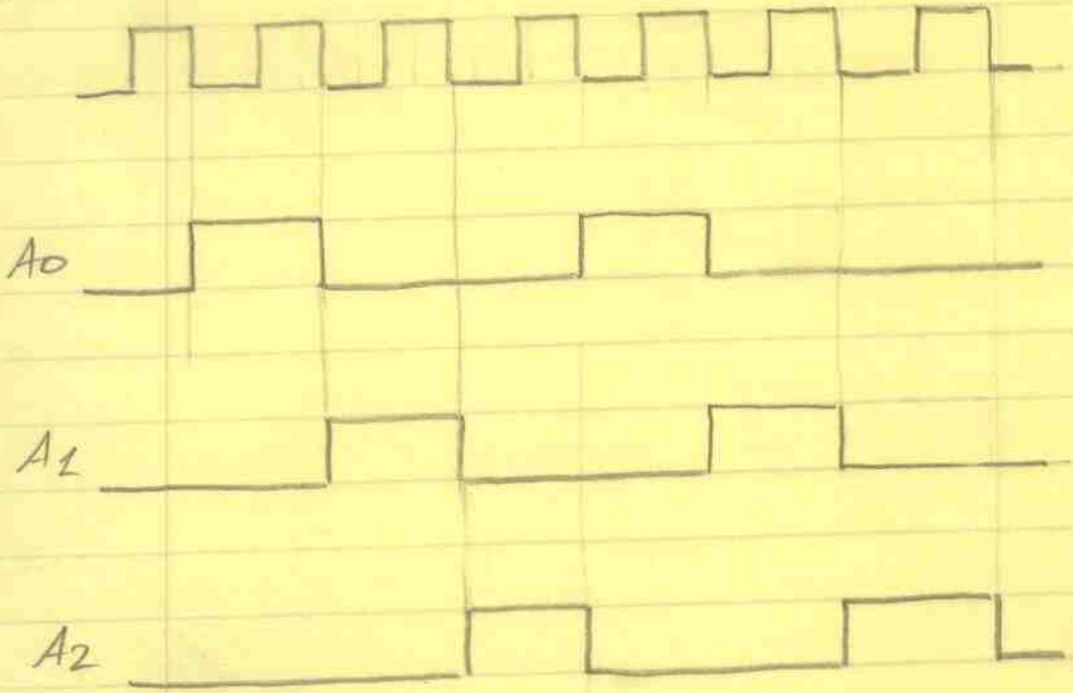
Synchronous Counters

Ripple counters suffer from speed of operation (propagation delay) and forced binary count sequence resulting from its construction. Any different counting sequence requires code conversion

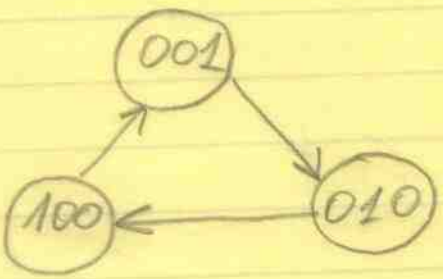
In synchronous counters pulses are connected to clock inputs of all FF's, allowing much faster operation. They are built similarily to ripple counters and can be configured similarily.

Special type are Ring Counters





FF0 is initially loaded with 1 (by preset) and $A_1A_2 = 00$ (cleared). Hence initial state of this ring counter is 001. It will go then to 010 and 100



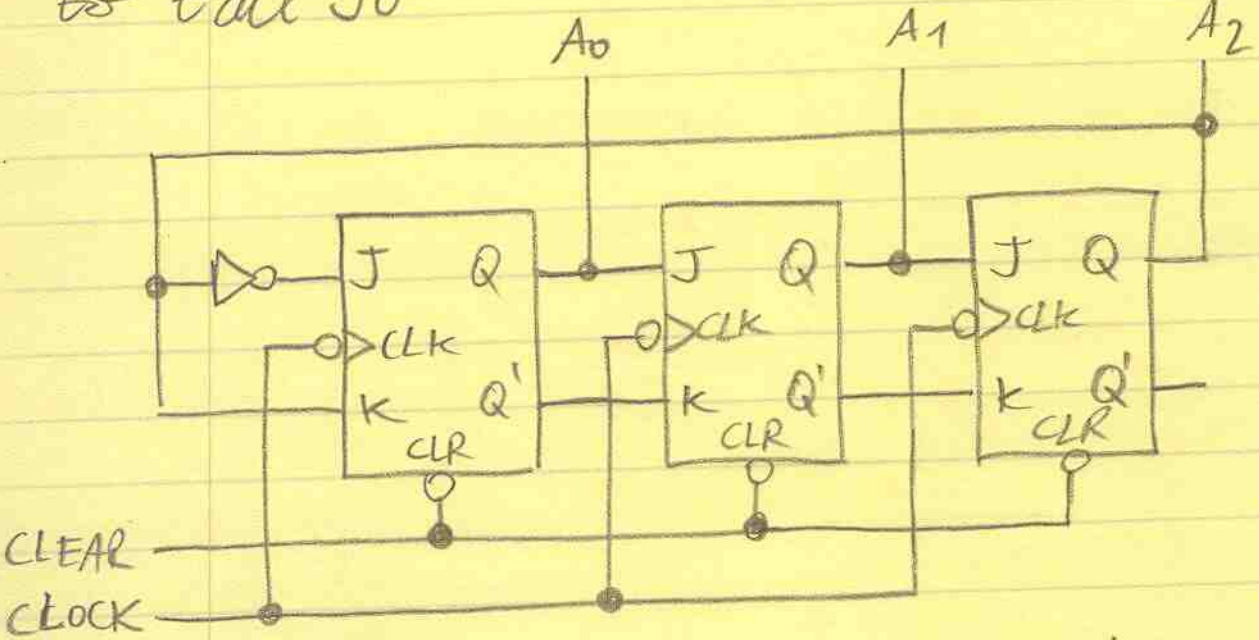
Not all initial FF states are valid, for instance

Presetting all FF's to 1 will make 111 permanent. clearing all to 0 will make 000 permanent. An N-bit ring counter has N valid states (incorporating single 1 and N-1 0's) and $2^N - N$ invalid states.

In Order to make a ring-counter self starting. The idea is to ensure that as long as there's at least one 1's in the pipe up to A_{N-1} A_0 sets to 0 until all pipe is 0 and then the A_0 goes to 1. This is accomplished by connecting J_0 to $(A_0 + A_1 + \cdots + A_{N-1})$

Twisted Ring counter (Johnson, switch tail)

The number of valid states can be dubled by connecting A_{N-1}' (head of ring counter - complement) to tail J0



Initially CLEAR resets all FF's then every negative CLK edge makes A0 1 and pushes all 1's to next FF's untill the pipe is Filled. Once A_{N-1} is 1, A0 turns to 0 and same 0 propargtion occurs for next N clock's negative edges.

1	CLEAR	0	0	0
2		1	0	0
3		1	1	0
4		1	1	1
5		0	1	1
6		0	0	1
0		0	0	0
		⋮		

There are 2N valid states and $2^N - 2N$ invalid ones.

Self starting is possible.

2N is always even, but modifying to odd is possible.

Ripple counter requires decoding, Ring doesn't.

Application - Digital clock (frequency division)

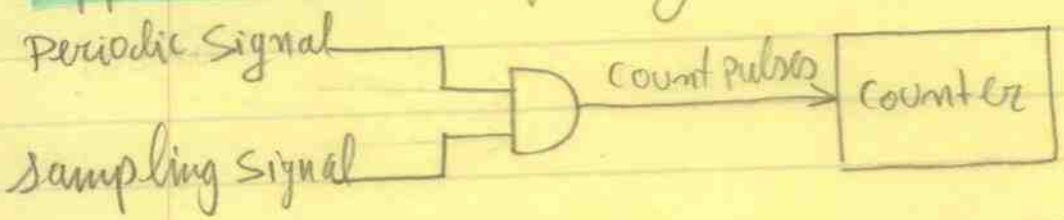
- Use the 50 Hz line voltage to generate 50Hz pulses
- Divide by 50 with a counter to generate

1 Hz Pulses (seconds)

- Divide by 60 to generate minute pulses
- " " 60 " hours "
- " " 24 to generate a day pulses

Use the output of 24-hour counter to display hours and similarly for minutes and seconds.

Application - Frequency measurement



$$f = \frac{\text{count pulses}}{\text{sampling time}}$$

longer sampling - more accurate

Application - Generation of timing signals

The operation of digital systems requires various timing signals.

With N-bit binary counter we can generate any $L \leq 2^N$ timing signals as follows