Lecture/Laboratory #10

- Review Lab
- Review lecture #9

- SR Flip – Flop
- Clocked FFs
- Edge – Triggered FFs (D – FF)
- JK – FF and T – FF
- Divider by 2^n
- Monostable Multivibrators (“one – shot”)
- Oscillators and Timer Chip (555)

Lab : Build frequency divider by 2^n

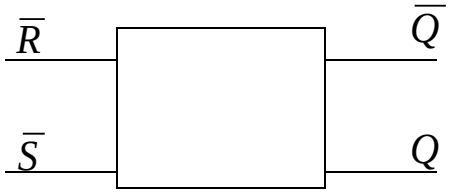
Sequential Logic

Devices with Memory

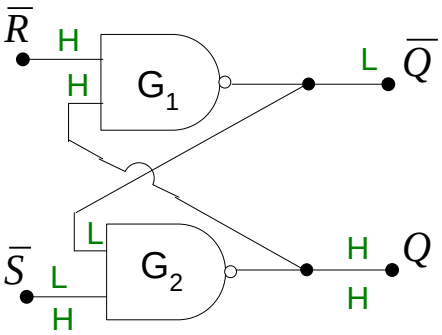
FLIP – FLOP → memory cell

$\overline{R}$ / $\overline{S}$ FF ← Reset / set Flip - Flop

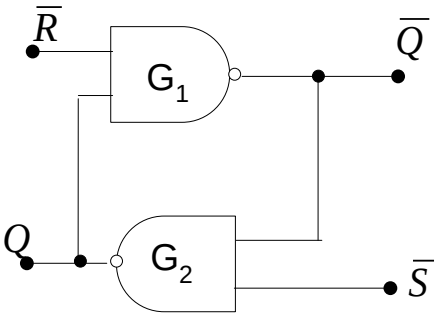
$\overline{R}$	$\overline{S}$	Q	$\overline{Q}$	
Inputs		Outputs		
1	0	1	0	stable
0	1	0	1	
1	1	Quiesent (no change)		
0	0	Not allowed		



Used as switch debouncer



or

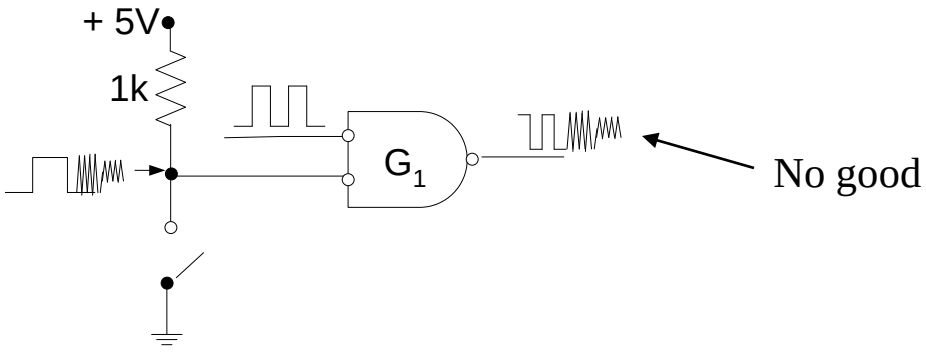


Equivalent presentation of ckts FF has two stable states (“bistable”).

Which state it is in depends on past history (memory)

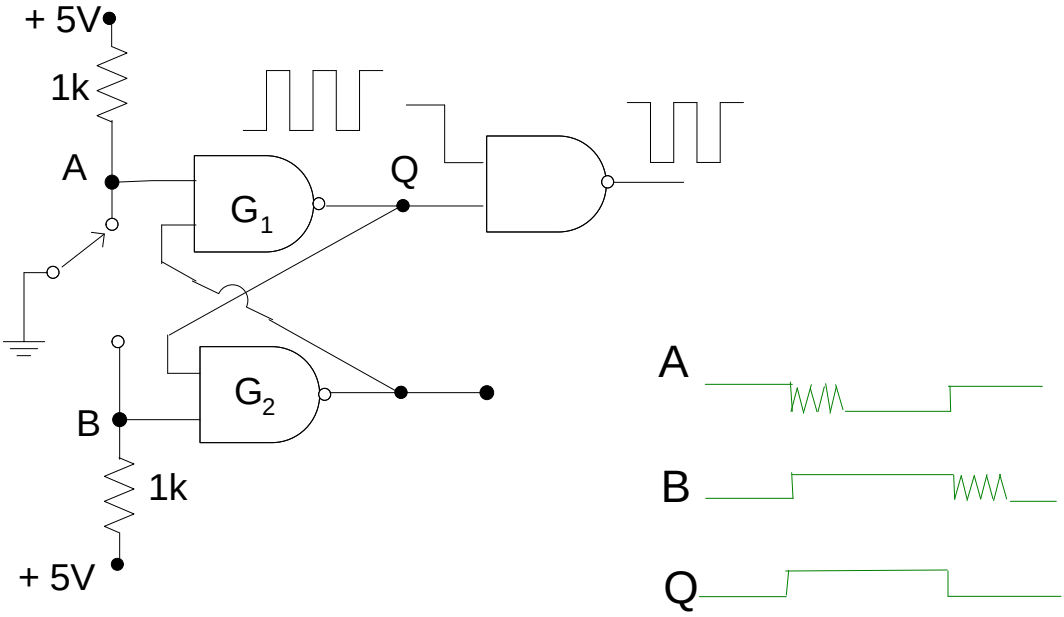
RS Flip – Flop as Switch Debouncer

Switch "bounce":



cure

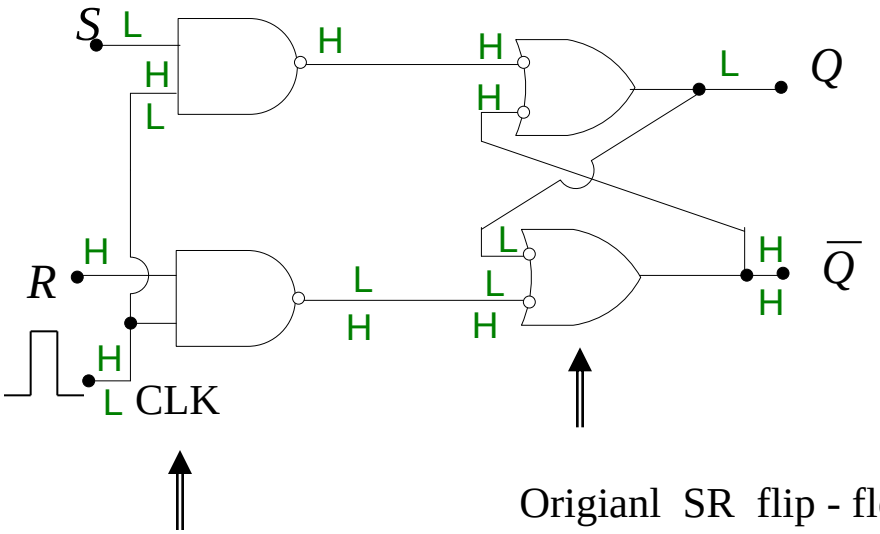
Switch debouncer:



Clocked Flip - Flops

SR (set-reset) flip – flops can be forced into one state or the other by just putting the right input signal. They are useful for number of applications (e.g. switch debouncing). However, the most popular form of such flip – flops has a single "clock" input and one or two "data" inputs. The outputs can change state or stay the same ⇒ it depends on data during the clock pulse arrival.

The simplest clocked flip-flop:



Origianl SR flip - flop

Par of gates controlled by clock
for SET and RESET inputs

Clocked Flip – Flops (cont.)

S	R	Q_{n+1}
0	0	Q_n (no change)
0	1	0
1	0	1
1	1	Indeterminate ("no allowed" in previous)

Q_{n+1} is output (Q) after the clock pulse, and

Q_n is output before clk pulse

The main difference between this and previous SR flip-flop is that S & R now behave as data inputs. Output depends on value of S and R when clock pulse comes.

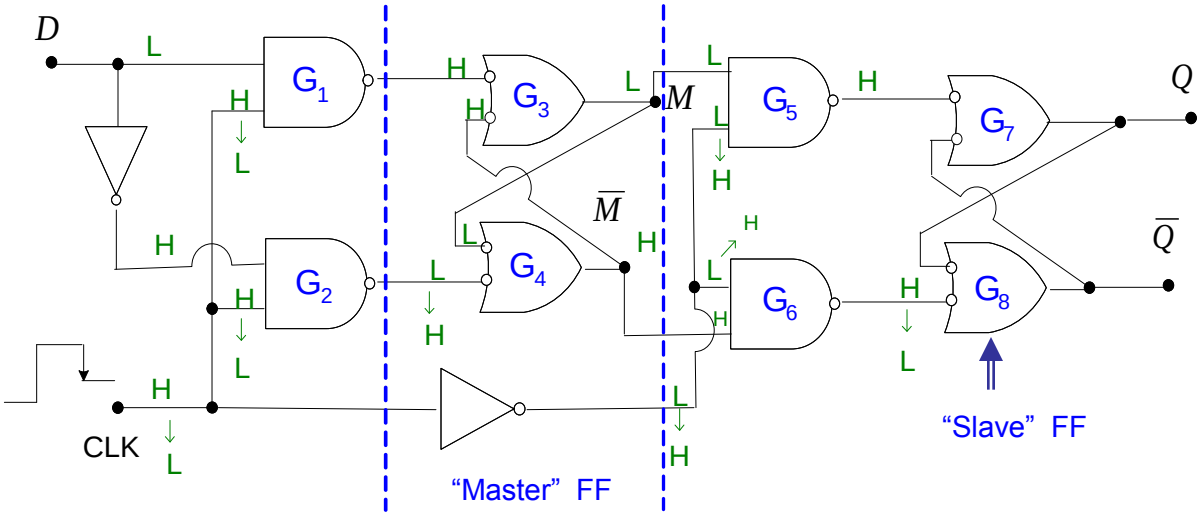
However, this flip-flop has one awkward property. Namely, the outputs can change during the time the clock is HIGH (1) if inputs change $\Rightarrow$ similarly as regular SR flip-flop.

Such property can be avoided by using edge-triggered flip-flops (D-type flip-flops), often used in "master – slave" configurations.

Edge-Triggered Flip-Flops (-D Flip-Flops)

These are popular, inexpensive flip – flops, packed as IC. In these FFs the data on inputs just before a clock transition ("edge") determines the output state after clock has changed.

Example of D – FF in "master-slave" configuration:



Analysis:

(a) CLK HIGH $\Rightarrow$ G_1 and G_2 are forcing "master" FF (G_3 and G_4) to the same state as input D $\Rightarrow$ M = D and $\overline{M} = \overline{D}$. Output from G_5 and G_6 is HIGH. If outputs from G_5 and G_6 are HIGH, then outputs from G_7 and G_8 will be the same as before (no change)

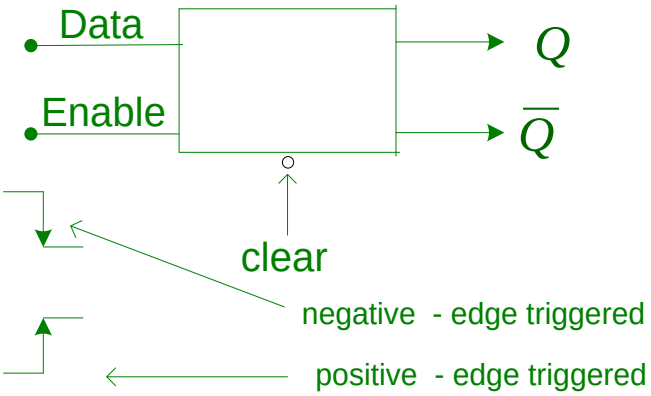
D Flip – Flops (cont.)

(b) CLK goes LOW $\Rightarrow$ outputs from G_1 and G_2 are HIGH independent of input D.

$\Rightarrow$ outputs from G_3 and G_4 are independent of D, therefore "master" FF outputs are transferred to "slave's" inputs and no further changes can occur until next rising edge of the clock.

(c) At the next rising edge of the clock (CLK goes HIGH) gates G_5 and G_6 decouple the "slave" from the "master" and G_1 and G_2 connect "master" with inputs $\Rightarrow$ M = D.

Schematically D – FF:



Negative-edge-triggered:

Q = D for E goes HIGH to LOW

Positive-edge-triggered:

Q = D for E goes LOW to HIGH

J - K Flip – Flops

J – K FF works similarly as D – FF, but it has two data inputs.

$J = \overline{K} \Rightarrow Q_{n+1} = J$

at next clock edge

$J = K = 0 \Rightarrow Q_{n+1} = Q_n$

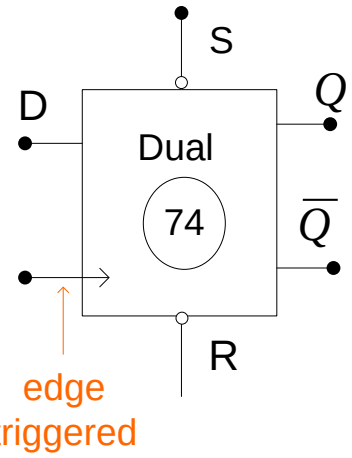
no change

Truth Table:

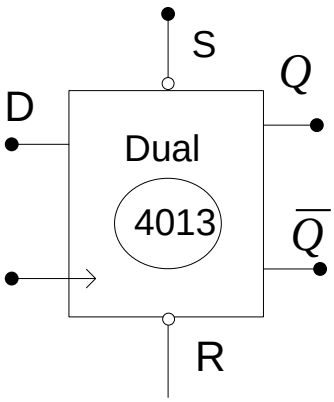
J	K	Q_{n+1}
0	0	Q_n
0	1	0
1	0	1
1	1	$\overline{Q}_n$

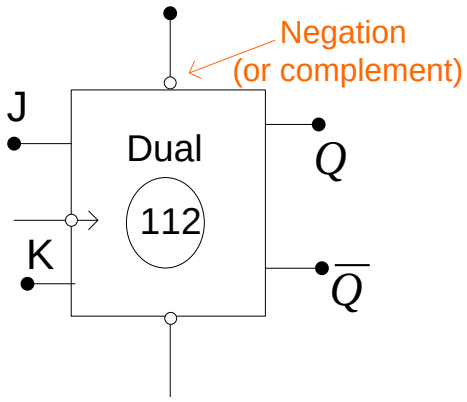
$J = K = 1 \Rightarrow$ output will reverse its state after each clock pulse – output will "toggle"

Examples of D – FF and JK – FF:



D – type FF (positive edge triggered)

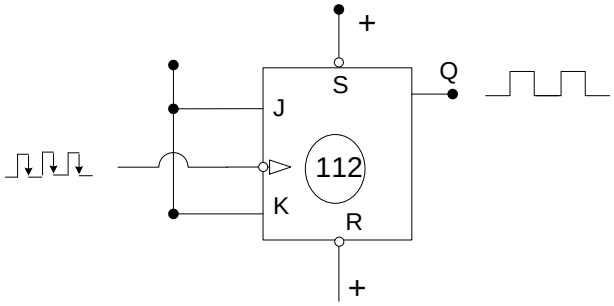




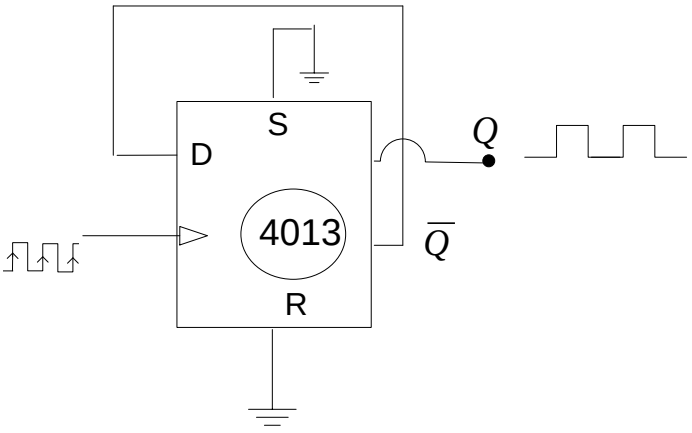
JK – type FF ("master" – "slave") (negative edge triggered)

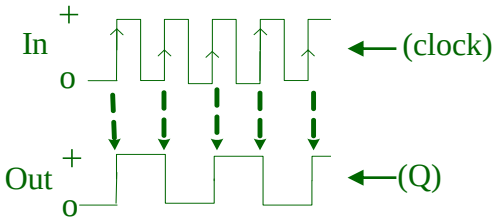
T Flip – Flop and Toggling

JK FF will toggle for both inputs HIGH (J = K = 1), as we have discussed. Practical use can be for making circuit, which divides by 2:



More common method of toggling is by using modification of D – FF, called T – FF:



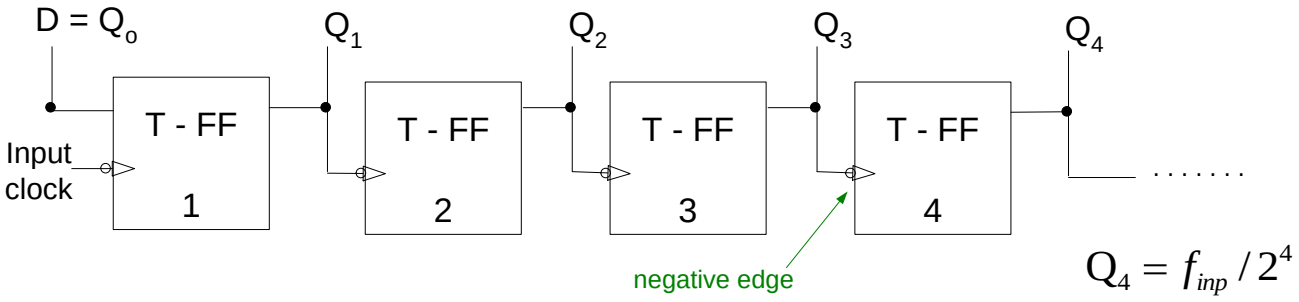


In T- FF input D is connected with its own complemented output $\overline{Q}$.

Therefore at the time of the clock pulse D = $\overline{Q}$ $\Rightarrow$ output at ½ frequency of the clock input.

Divider by 2^n

T- FFs can be used for creation divider by 2^n. Because one T – FF can provide divider by 2, therefore n T – FFs can $\Rightarrow$ by 2^n by connecting output of each one to next clock input:



Timing diagram: