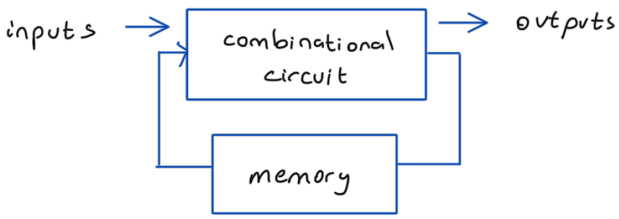
6 - synchronous sequential circuits

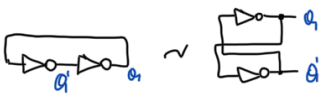
flip-flop / latch = circuit that has two stable states and can be used to store state information (memory)



- same inputs can produce different outputs, depending on memory

memory
1. holds a value
2. the value that was saved can be read
3. the value that was saved can be changed

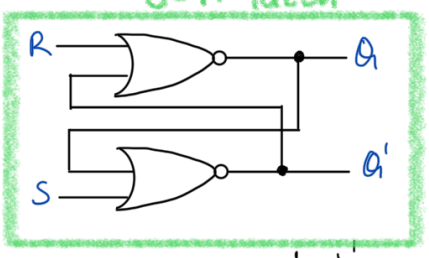
basic idea of storage in circuits:
make a loop (feedback), so the circuit outputs are inputs.



1. ✓
2. ✓
3. ✗ → we cannot change Q

using NAND gates - SR latch (set/reset) → memories and sequential
inputs → S, R
outputs → Q, Q' → they fed back into the circuit → also inputs

S-R latch



$Q_{next} = (R + Q'_{current})'$

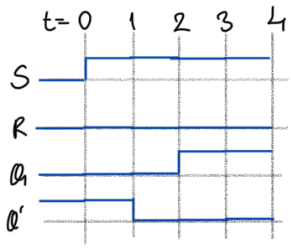
$Q'_{next} = (S + Q_{current})'$

	inputs S	inputs R	current Q	current Q'	next Q	next Q'	
SR=00	0	0	0	1	0	1	(depends on current Q) remain in the same state
	0	0	1	0	1	0	
SR=01	0	1	0	1	0	1	reset/clear to 0
	0	1	1	0	0	1	
SR=10	1	0	0	1	1	0	set to 1
	1	0	1	0	1	0	

S	R	Q
0	0	no change
0	1	0 (reset)
1	0	1 (set)
1	1	avoid!

☆ when SR = 11 → Q = Q' = 0 (contradiction) → infinite loop (don't set SR=11)

latch delays:

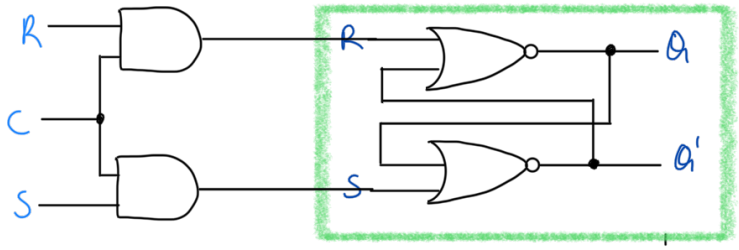


example

SR = 10
Q = 0 → Q' = 1

↳ $Q_{next} = 1$

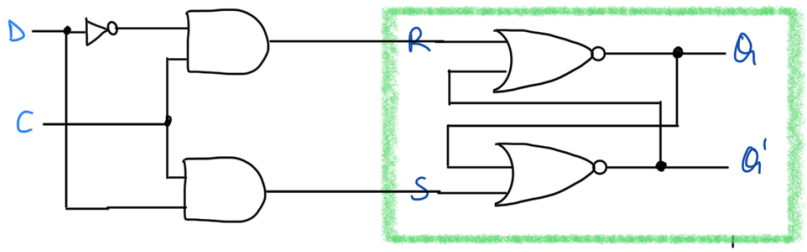
an S-R latch with a control input C



C control input = acts like an enable

C
0 → no change in the output
1 → acts as in the table

a data latch D based on an S-R latch



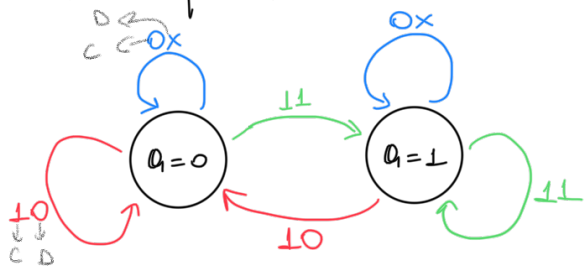
C	D	Q
0	x	no change
1	0	0
1	1	1

(C=1, Q=D)

no need for two inputs to set and reset

state diagrams: are used to describe sequential circuits

C	D	Q(t+1)
0	x	Q(t)
1	0	0
1	1	1



latches in real life

ALU: arithmetic logic unit: does arithmatic operations like addition, substraction, and logic operations like and, or

Latches: acting as a memory

example

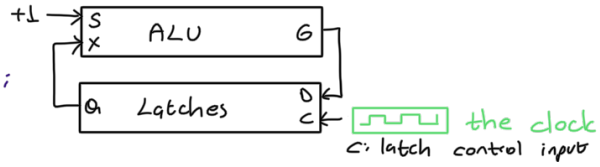
the operation that we want to implement G=X+1 → increment

when and how to stop cycle?

when: add another signal to circuit, when it is 1, the computation is completed

clocks:  → used to synchronize circuits (the period must be set appropriately for the ALU)

our example:



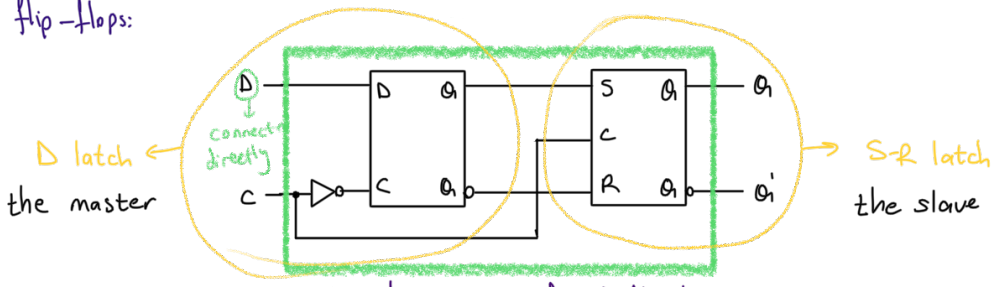
C=1

the latch will be enable for writing

☆ higher frequency does not always mean faster machine. (the work done at each period is also important)

how: by combining latches together in a special way to form flip-flops

flip-flops:

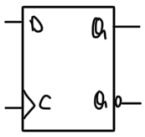


internal structure of a D flip-flop
(it is not a latch anymore, because we connected clock signal to C)

C = enables either the D latch or the SR latch, but not both

C	D latch (master)	SR latch (slave)
0	enabled → whenever D changes the masters output change too	disabled → has no affect on it maintains current state
1	disabled → output is the last D input value	enabled → its state changes to refl. the master's output → th D input value (when C=1)

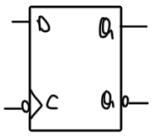
D flip-flop:



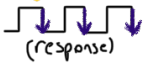
- positive edge triggered flip-flop
- the output changes only after the positive edge of C.

C	D	Q
0	x	no change
1	0	0 (reset)
1	1	1 (set)

same with D latch, but we have a clock



- negative edge

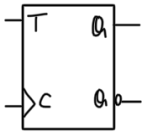


(responding to a clock)
- response to positive level

starting value of Q = flip-flops provide inputs that let us set or clear the state
↳ so we reset the circuit once to initialize

flip-flop variations

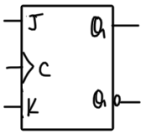
T flip-flop: can only maintain or complement its current state



C	T	Q next
0	x	no change
1	0	no change
1	1	Q' current

JK flip-flop: has inputs act like S and R, but 11 state is used here to complement current state

in S-R it is avoided



C	J	K	Q next
0	x	x	no change
1	0	0	no change
1	0	1	0 (reset)
1	1	0	1 (set)
1	1	1	Q' current

Charasteristic tables: the tables that we've made so far, but we do not add $C=0$ to simplify. ($C=1$ in the tables)

no chang

- the tables do not indicate the positive edge-triggered behaviour.
  - ↳ it does not matter which trigger to use, but we cannot refer it from the table

D flip-flop

D	$Q(t+1)$	Operation
0	0	reset
1	1	set

$Q(t+1) = D$

T flip-flop

T	$Q(t+1)$	Operation
0	$Q(t)$	no chage
1	$Q'(t)$	complement

$Q(t+1) = T'.Q(t) + T.Q'(t)$

$= T \oplus Q(t)$

J-K flip-flop

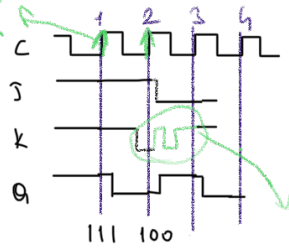
J	K	$Q(t+1)$	Operation
0	0	$Q(t)$	no change
0	1	0	reset
1	0	1	set
1	1	$Q'(t)$	complement

$Q(t+1) = K'.Q(t) + J.Q'(t)$

characteristic equations

positive-edge

flip-flop timing diagrams



$Q(1) = 1$ → initial

$Q(2) = 0$ → $JK = 11$

$Q(3) = 1$ → $JK = 10$

$Q(4) = 0$ → $JK = 01$

this change can happen, the value on the line (purple) is important

Summary

- to use memory in larger circuits, we need to:
  - ↳ keep latches disabled until new values are ready to be stored
  - ↳ enable the latches just long enough for the update to occur
- a clock signal is used to synchronize circuits. the cycle time reflects how long operations take
- flip-flops restrict the memory writing interval, to just the positive edge of the clock signal
  - ↳ ensures that memory is updated only once per clock cycle
  - ↳ they are several different kinds of flip-flops, but they all serve the same purpose → storing b