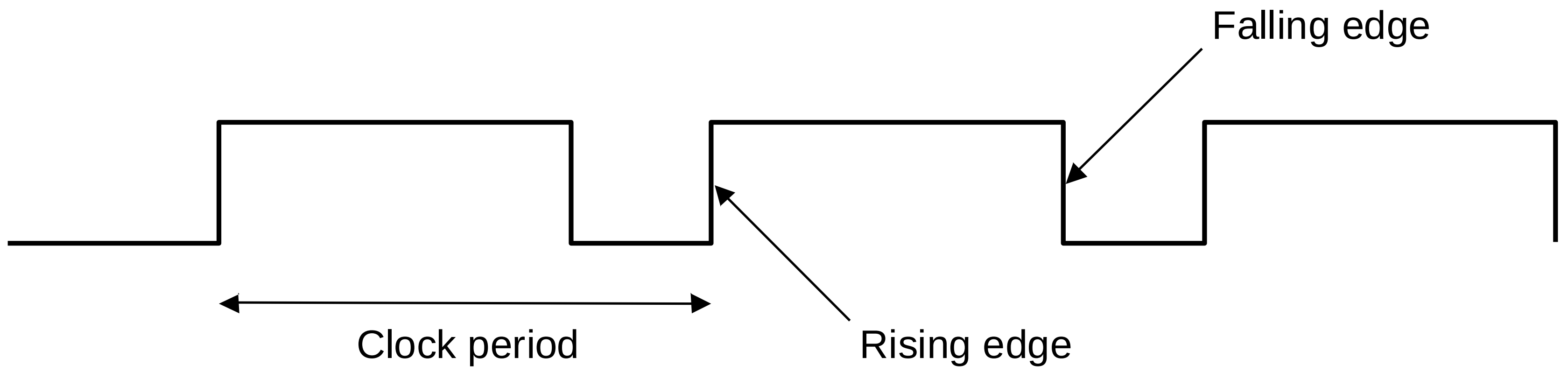
TUTORIAL 2 SEQUENTIAL LOGIC CIRCUIT

Overview

- **We will review the following concept in this tutorial:**
- **Sequential logic circuits**
  - Have memory
  - Output depends on the current input(s) and value stored in the memory (called *state*)
- **Clock**
- **Timing diagram**
  - To describe the behavior of the circuit along time
- **Work with a few practical examples**
  - S-R latch and D latch with NAND gates
  - Edge-triggered D flip-flop
  - Register
  - Binary counter

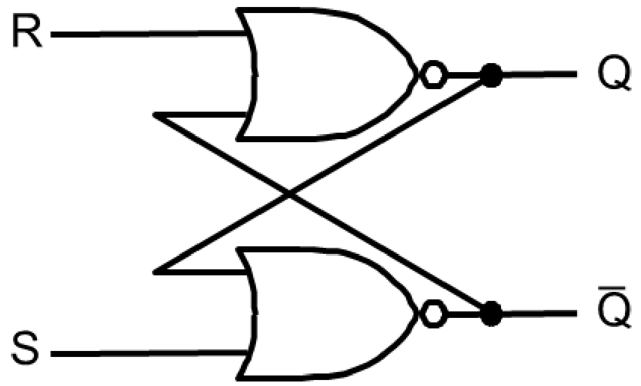
Clock

- A clock acts as a global signal that gives all the components in the system an indication of time.
- Clock is used in sequential logic to decide and co-ordinate state updates.
- A clock signal has three important key words that you need to know:



S-R Latch: the Simplest Memory Element

- A **S-R latch** (Set-Reset latch) using **NOR** gates shown below is the simplest memory element.
- Un-clocked memory element (Asynchronous device)

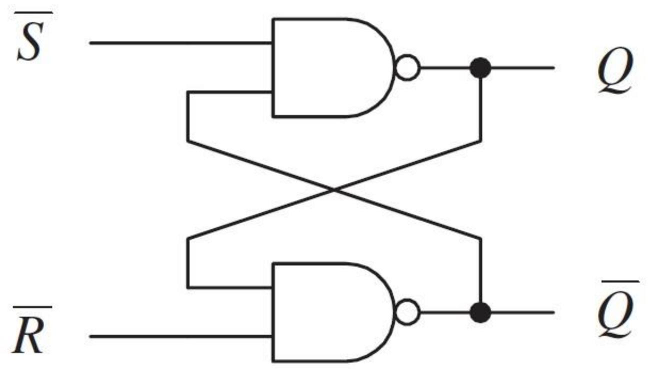


S	*R*	Action on Q
0	0	latched
0	1	Q = 0
1	0	Q = 1
1	1	forbidden

- **Question**: How does this circuit "stores" information? What is the size of the information being stored?

S-R Latch with NAND Gates

- **S-R latch** (Set-Reset latch) can also be implemented by **NAND** gates
- **Un-clocked** memory element (**Asynchronous** device)

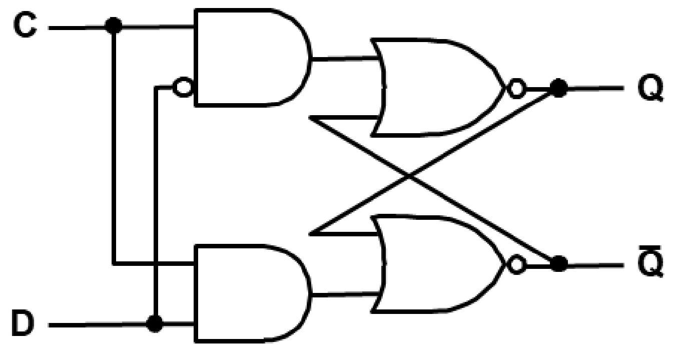


$\overline{S}$	$\overline{R}$	Action on Q
0	0	forbidden
0	1	Q = 1
1	0	Q = 0
1	1	latched

- **Note**
  - S-R NAND latch is an inverted version of S-R NOR latch. Thus, the truth table is also inverted.
  - The input signal is effective when it is de-asserted.

A Clocked Memory Element: D-latch

- The value stored in a D-latch can be updated iff the clock is asserted (i.e. C=1).
- An implementation using NOR gates is shown below. S-R latch (Set-Reset latch) can also be implemented NAND gates

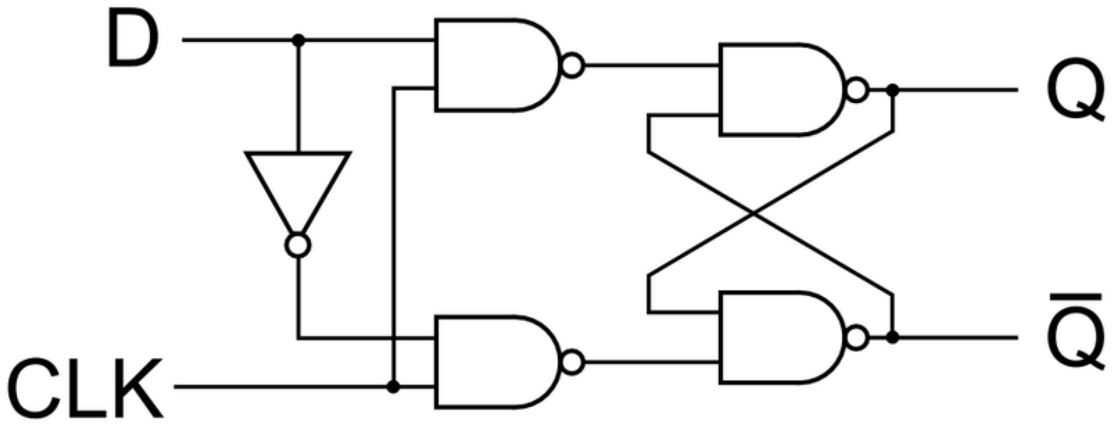


C (Clock)	D	Action on Q
0	0	Nothing changed
0	1	Nothing changed
1	0	Q = 0
1	1	Q = 1

- **Note**
  - the S-R latch on the right part of the D-latch circuit.
  - From the circuit, argue whether it is possible to do update when the clock is not asserted?

D-latch with NAND Gates

- **D latch can also be implemented with NAND gates as shown in below figure.**
  - Clocked memory element (Synchronous device)

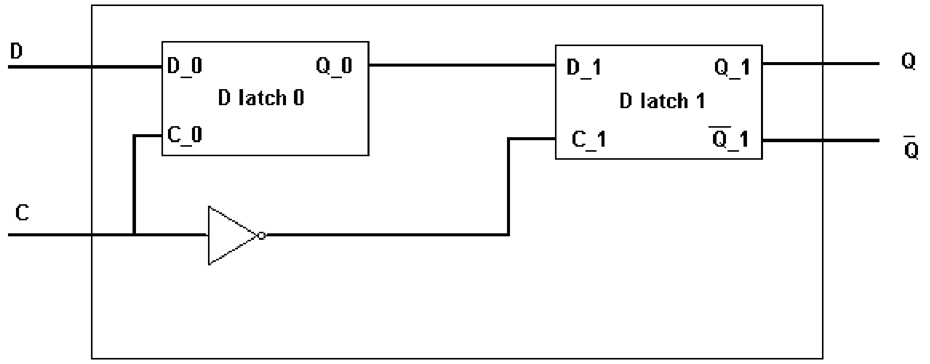


C (Clock)	D	Action on Q
0	0	Nothing changed
0	1	Nothing changed
1	0	Q = 0
1	1	Q = 1

$$Q = (D \; nand \; C) \; nand \; \bar{Q}, \qquad \bar{Q} = (\bar{D} \; nand \; C) \; nand \; Q.$$

D Flip-flop

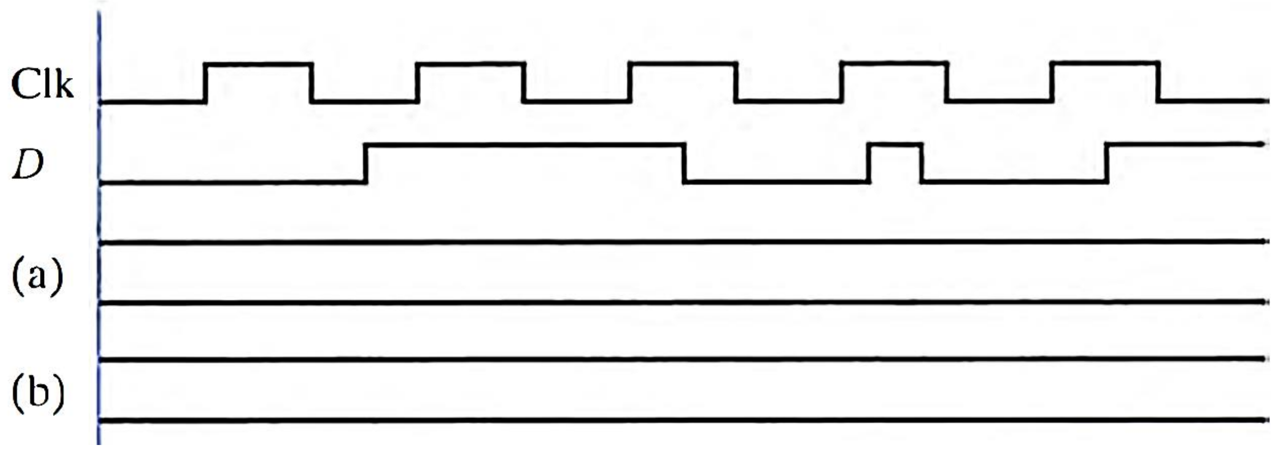
- **A D Flip-flop can be updated only on a falling/rising clock edge.**
  - There are many ways to create a D flip flop, the figure below (from the lecture notes) shows a D flip flop created from two D latches.



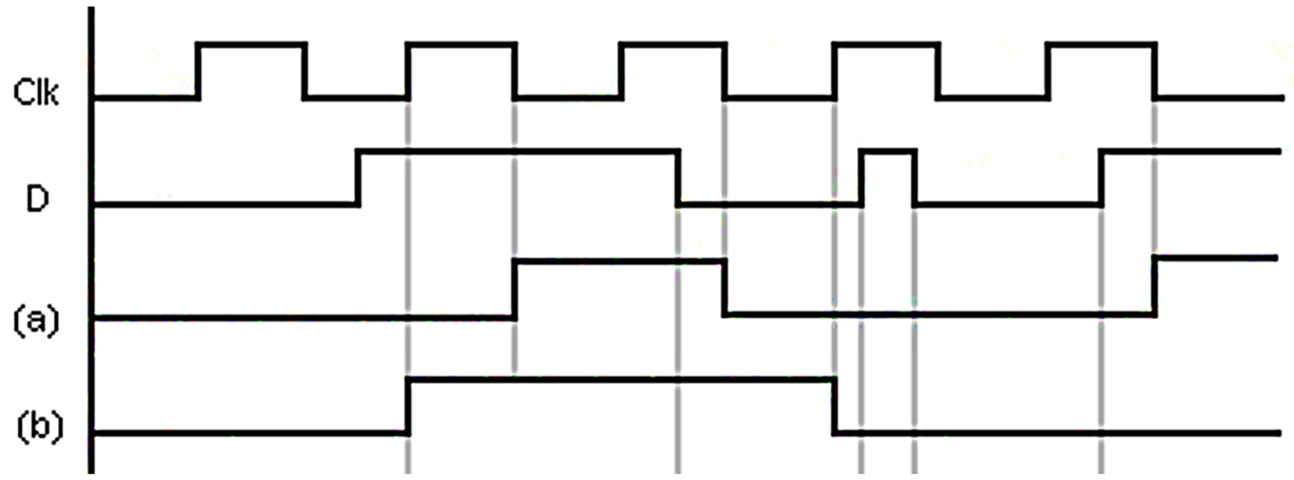
- **Question: Can this flip-flop be updated on a rising or falling clock edge?**
- **Question: Without adding new hardware (wires are OK), how to modify the device so that it can be updated on the other clock edge?**

Exercise

- Given the input (D) and clock transitions (Clk) of the Figure below, determine the output of the device if it is a:
  - (a) Falling-edge triggered D flip flop.
  - (b) Rising-edge-triggered D flip-flop.

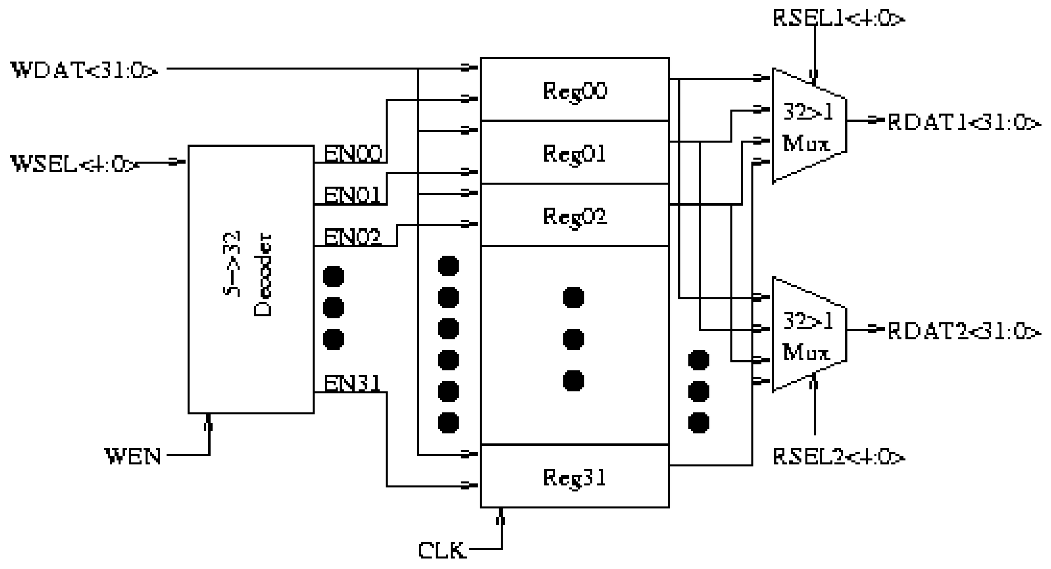


Solution



Register File

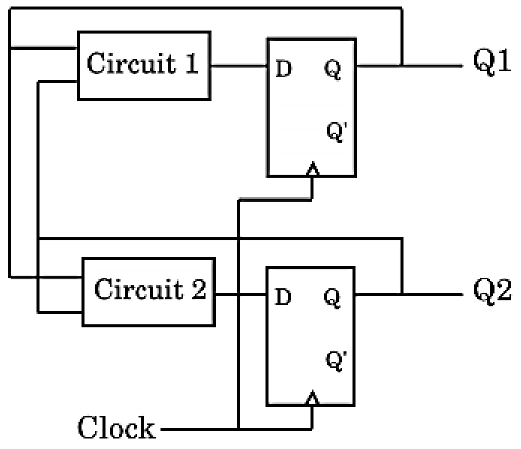
- A **register file** is a piece of hardware that allows reading from and writing to the desired registers.
- It is usually implemented by way of fast static RAMs with multiple ports.

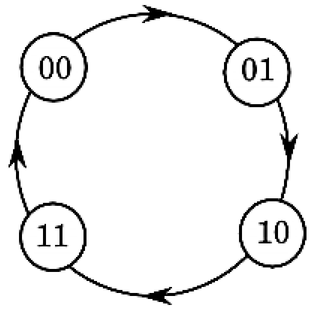


- ❑ How do you read from a register (what are the inputs)?
- ❑ How do you write to a register (what are the inputs)?

Synchronous Binary Counters

- **Synchronous digital circuit – Binary counter with 2-bit memory**
  - Only has clock signal as input
  - At each clock pulse, the counter takes up a new state and thus goes through a specific count sequence.
  - The block diagram, structure and state transition diagram of a two-bit binary counter is of the following form.
  - Design the state sequencing logic which consists of the two combinational circuits labelled Circuit 1 and Circuit 2.



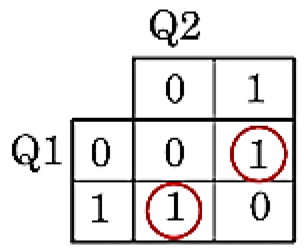


Truth Table and K-map

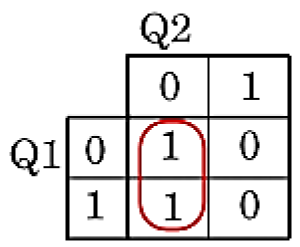
- Step 1: Construct the truth table (or transition table)

Inputs (t_n)		Outputs (t_{n+1})	
Q_1	Q_2	D_1	D_2
0	0	0	1
0	1	1	0
1	0	1	1
1	1	0	0

- Step 2: Construct the corresponding K-map



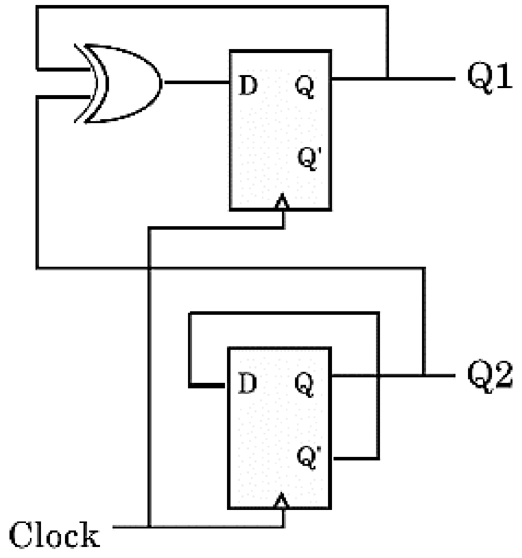
D1



D2

Truth Table and K-map

- Step 3: Simplified logic function
  - ❑ $D_1 = Q_1\overline{Q_2} + \overline{Q_1}Q_2 = Q_1 \oplus Q_2$
  - ❑ $D_2 = \overline{Q_2}$
- Step 4: Circuit design


D
Q
Q'
Q1
D
Q
Q'
Q2
Clock