ECE 2300
Digital Logic & Computer Organization

Spring 2025

Sequential Logic:
Clocks
Latches
Flip-Flops

Cornell University

Announcements

- HW 2 due tomorrow
- HW 3 will be posted today

CMOS Logic: True or False?

1. NMOS is an active low switch

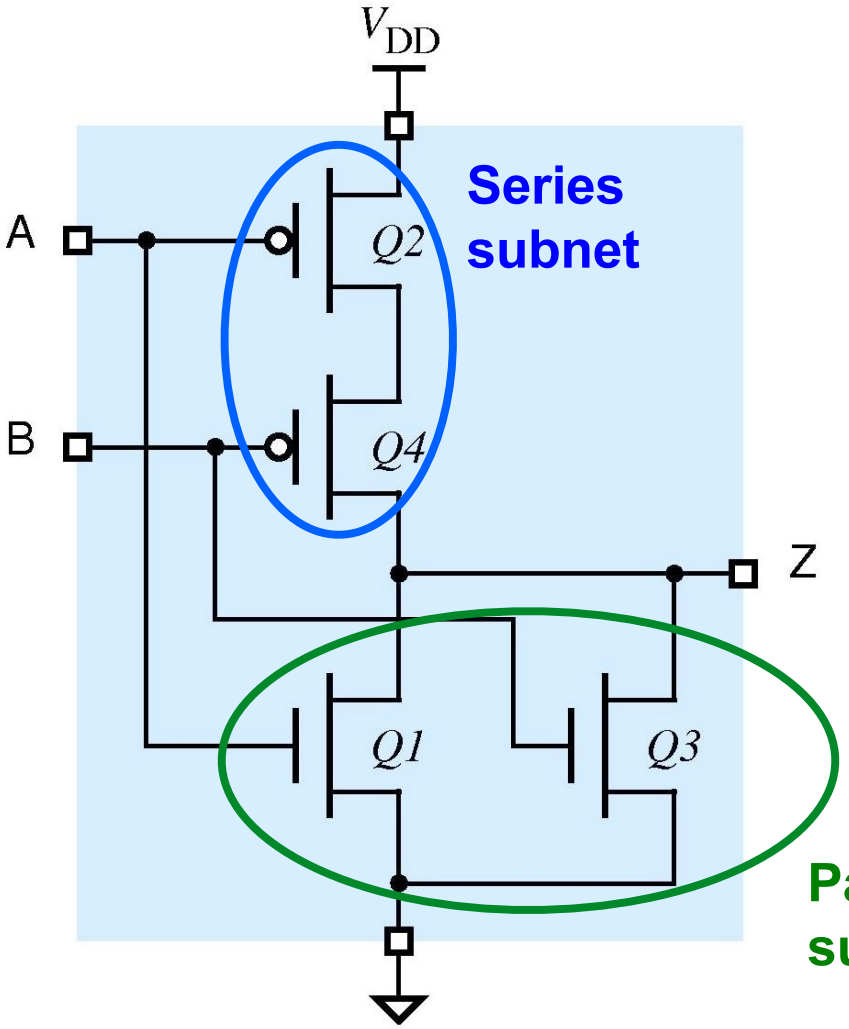
2. PMOS (pull-up) network passes a poor zero

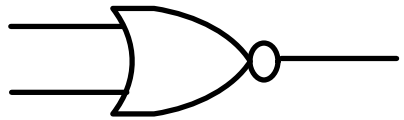
3. NMOS and PMOS networks can be ON at the same time

4. A CMOS gate consists of an even number of transistors

5. A 2-input NOR requires 2 transistors

2-Input NOR Gate in CMOS





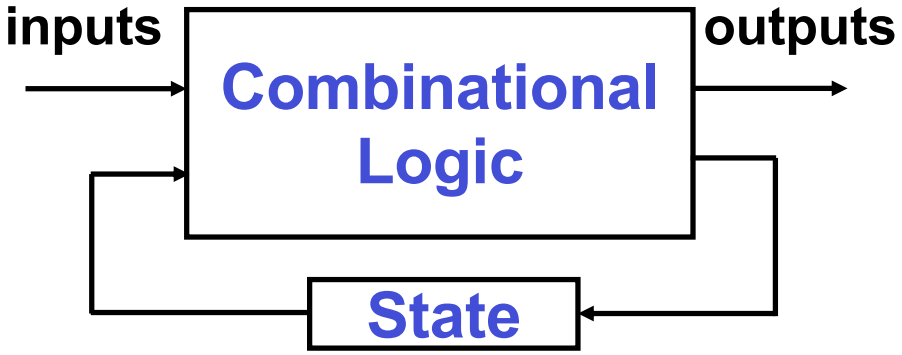
Pull-up: A'•B'

Pull-down: (A+B)'

Combinational vs. Sequential Circuits

Combinational Circuit

Sequential Circuit



- **Combinational**
  - Output depends only on current inputs

- **Sequential**
  - Output depends on current inputs plus *state* (past history)
  - Requires memory

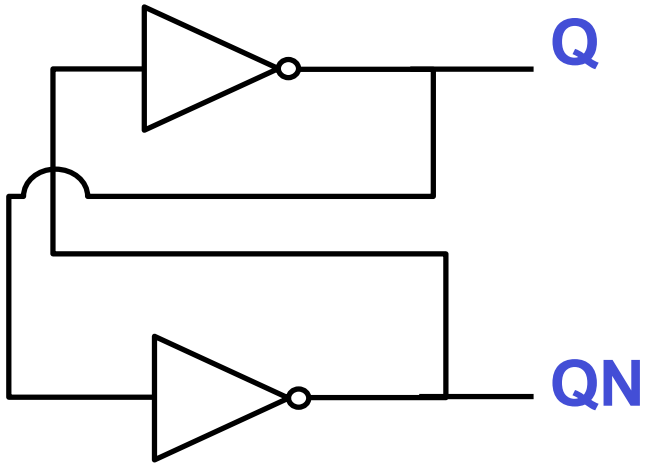
Sequential Circuits

- Outputs depend on inputs and state variables
- The state variables embody the past history of the circuit
  - **Storage elements hold the state variables**
- A clock periodically advances the circuit

Basic Storage Element

Bistable Element

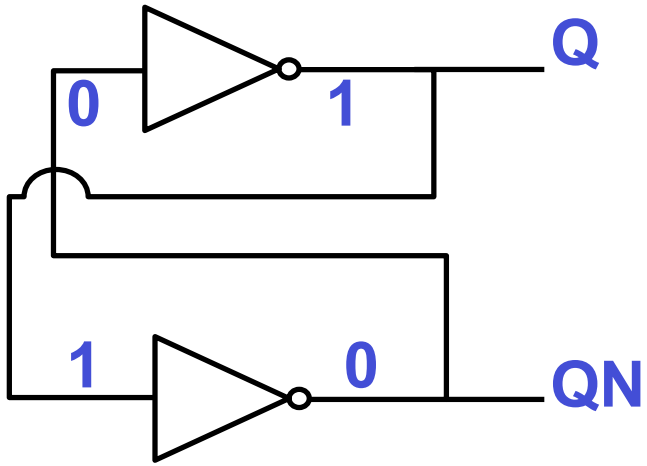
- Basic storage element
- Inverters with outputs connected to inputs

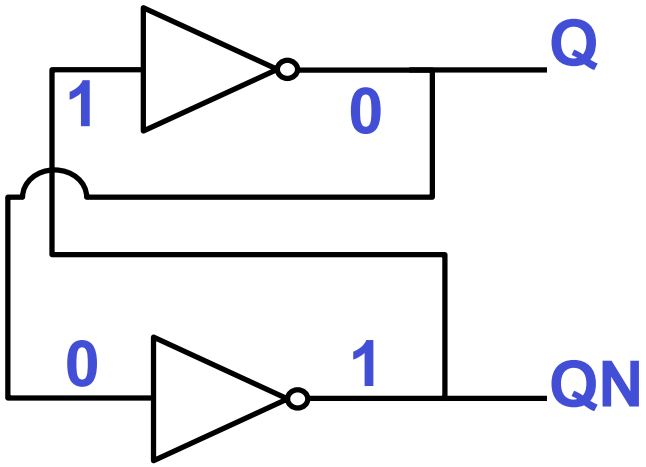


- What does the circuit do?

Bistable = Two Stable States

- Two inverters with outputs connected to inputs
- Bistable element stores a "given" value indefinitely (as long as powered)



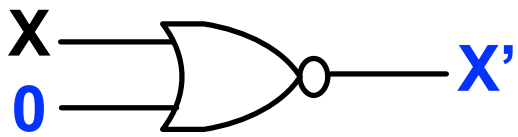


How can we change the stored value?

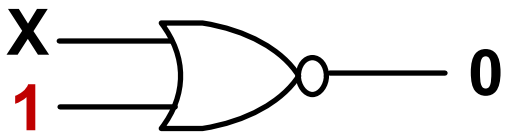
(Extra inputs are required)

Revisit 2-input NOR

- NOR can implement an inverter

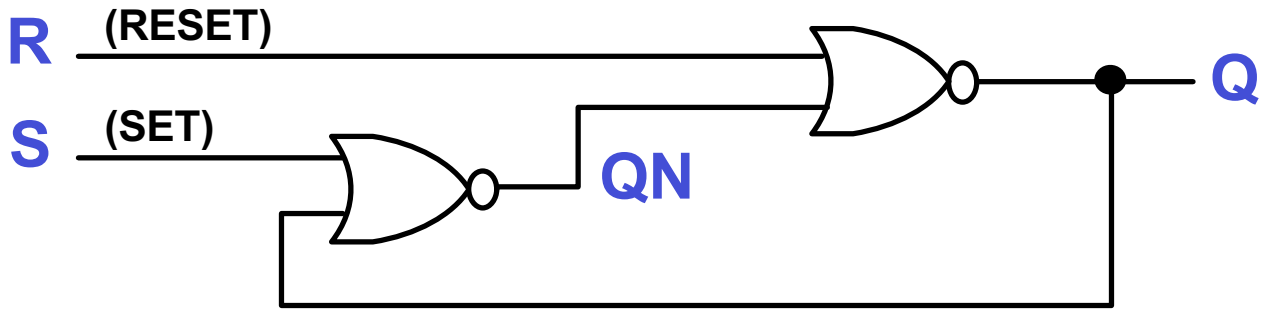


- NOR has a null element



A NOR gate outputs 0 when one of the inputs is 1, regardless of the value of the other input

S-R Latch (Set-Reset Latch)



Set (S) and Reset (R) inputs allow (re)setting stored value

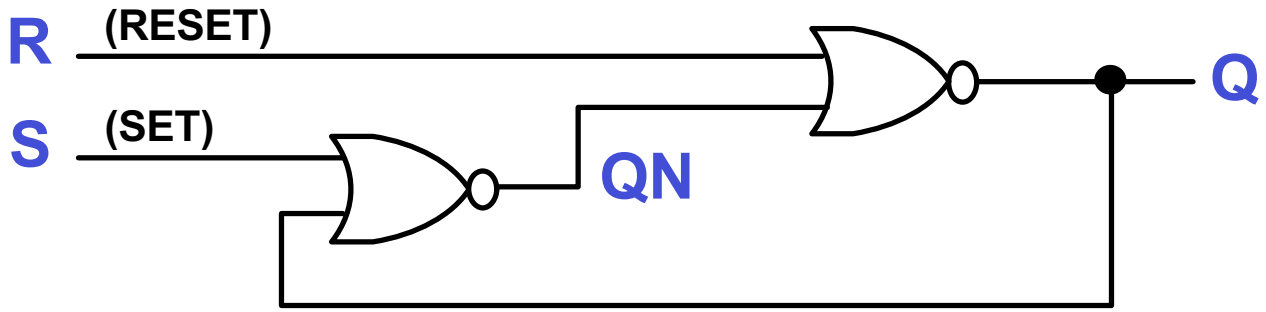
Hold: both NORs act as inverters, forming a bi-stable element

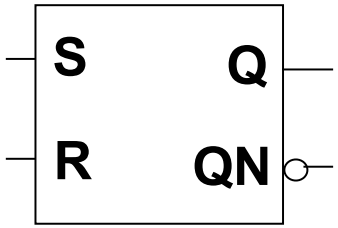
Reset/Set: the null element of NOR ("1") is in effect

Q_{next} is new state of Q when R or S changes

	S	R	Q_{next}
Hold state	0	0	
Reset	0	1	
Set	1	0	
	1	1	

S-R Latch (Set-Reset Latch)





Boolean expression for Q_{next} in terms of R, S, Q:

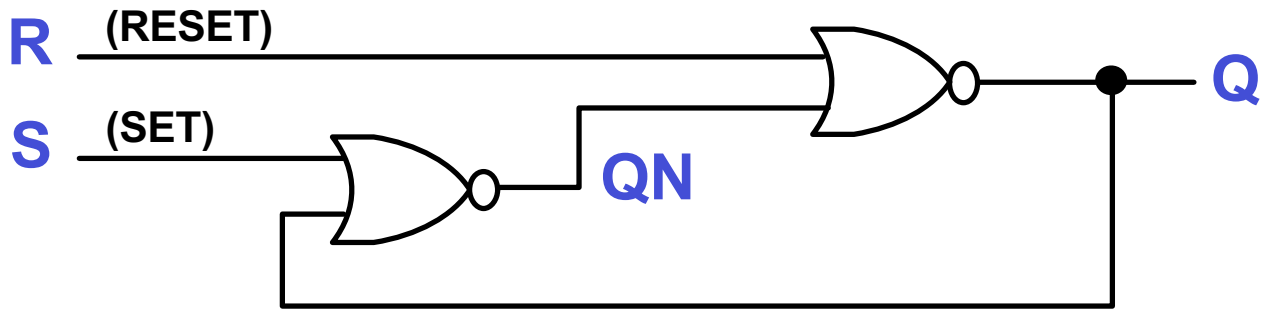
$$Q_{next} = (R + QN)'$$
$$= (R + (S + Q)')'$$
$$= R' \cdot (S + Q)$$
$$= R' \cdot S + R' \cdot Q$$

S	R	R'•S	R'•Q	Q_{next}
0	0	0	Q	(Last) Q
0	1	0	0	0
1	0	1	Q	1
1	1	0	0	0

When SR=00, it holds (*latches*) the previous state

Avoid SR = 11

Exercise: S-R Latch



- **Suppose the S-R latch is initialized with Q=0, write down the next state given the following input sequence**

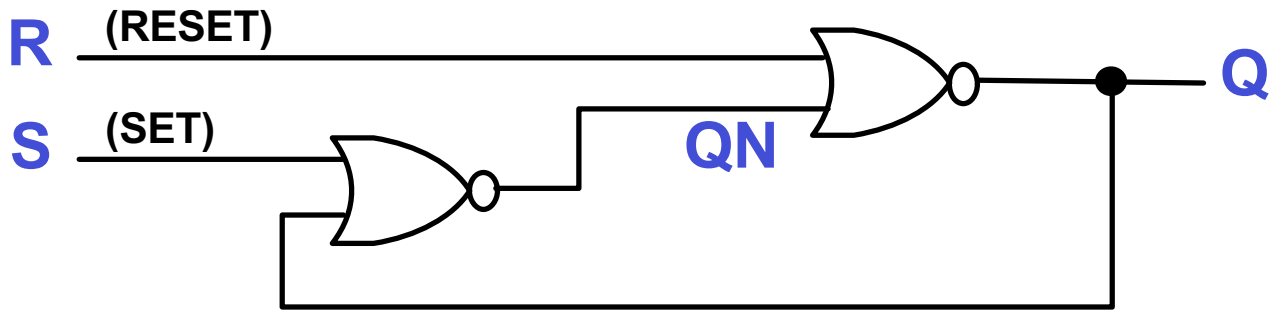
  S=1, R=0 → Q = ?

  S=0, R=0 → Q = ?

  S=0, R=1 → Q = ?

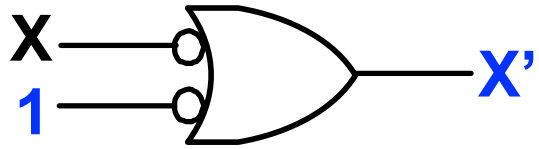
Instability (Avoid SR=11)

- **SR=11 may cause instability of the internal state**
  1. **When SR=11 → Q=QN=0**
  2. **If inputs subsequently change to SR=00, output may be undefined**

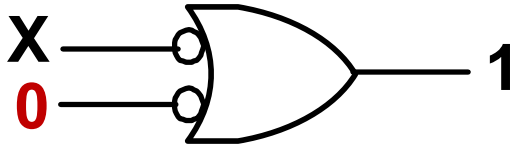


Revisit 2-input NAND

- **NAND can implement an inverter**



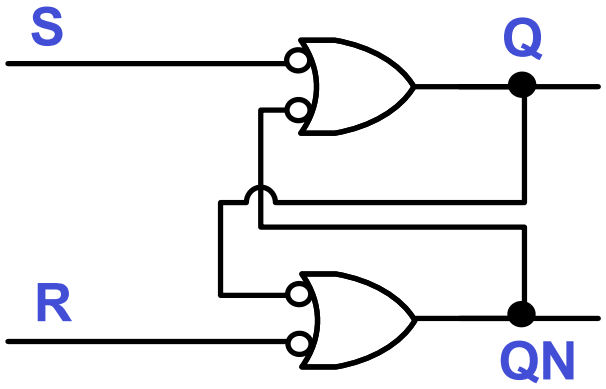
- **NAND has a null element**



A NAND gate outputs 1 when one of the inputs is 0, regardless of the value of the other input

$\overline{S}$-$\overline{R}$ Latch

- *S-bar-R-bar* latch: **Built from NAND gates**

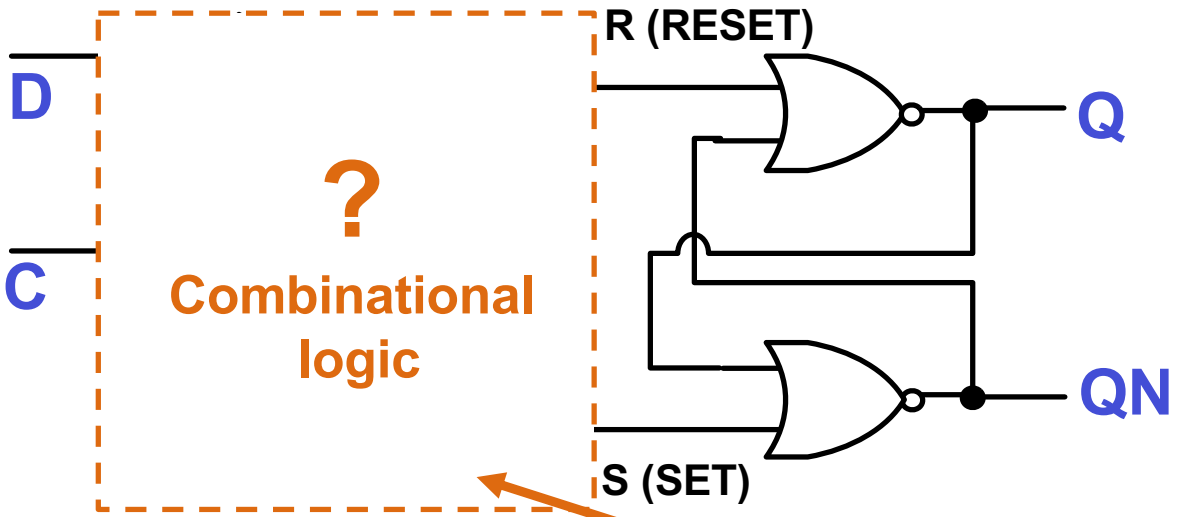


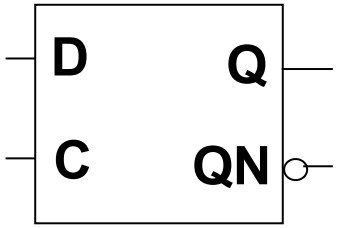
D Latch

- D latch: builds on S-R latch where S and R cannot be both 1
  - Output "follows" input
- D latch captures input data (what to set) when certain condition holds (when to set)
- Operates in two modes
  - Open (or enabled): input flows through to output
  - Closed (or disabled): output does not change

D Latch

- When C is enabled, Q output follows D input
- When C is disabled, Q output retains last state



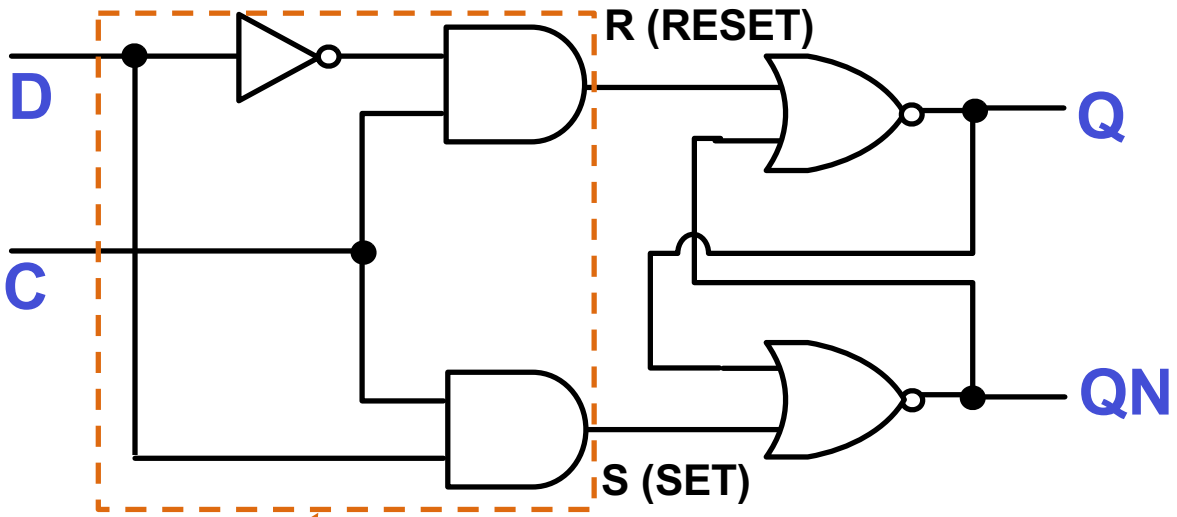


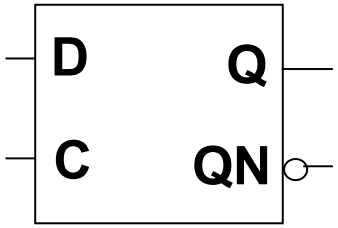
Can we build a D latch using an SR latch as a building block?

C	D	R	S	Q_{next}
1	0			0
1	1			1
0	X			(Last) Q

D Latch

- When C is enabled, Q output follows D input
- When C is disabled, Q output retains last state



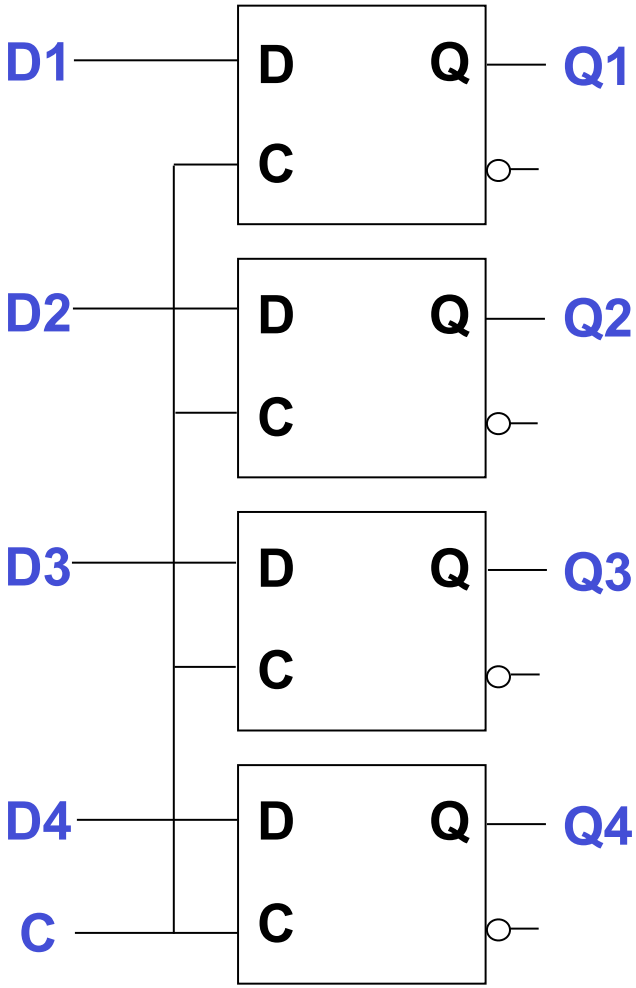


R = C•D'
S = C•D

C	D	R	S	Q_{next}
1	0	1	0	0
1	1	0	1	1
0	X	0	0	(Last) Q

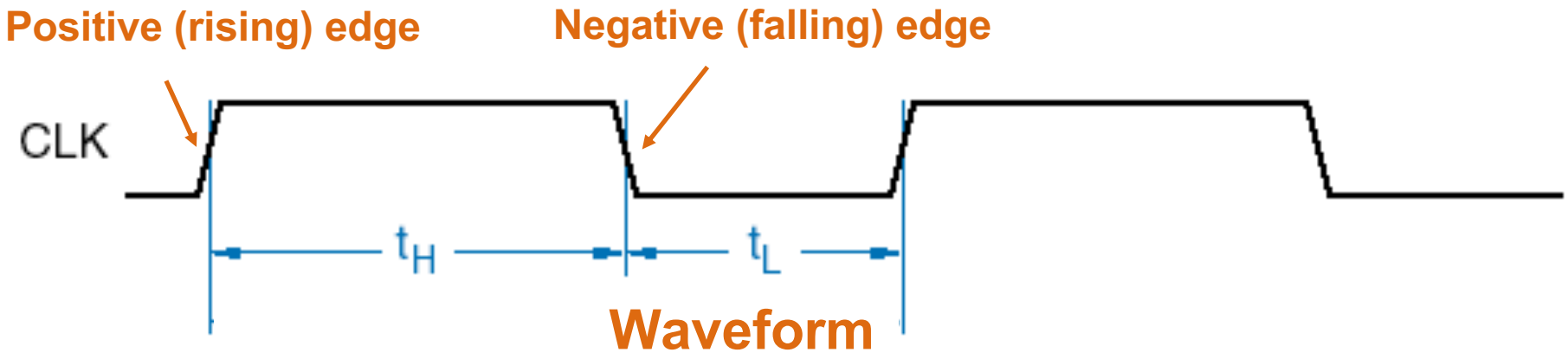
Multi-Bit Latch

- Simultaneously latches multiple bits
- "Latch" may refer to 1 bit latch or multi-bit one



Clock

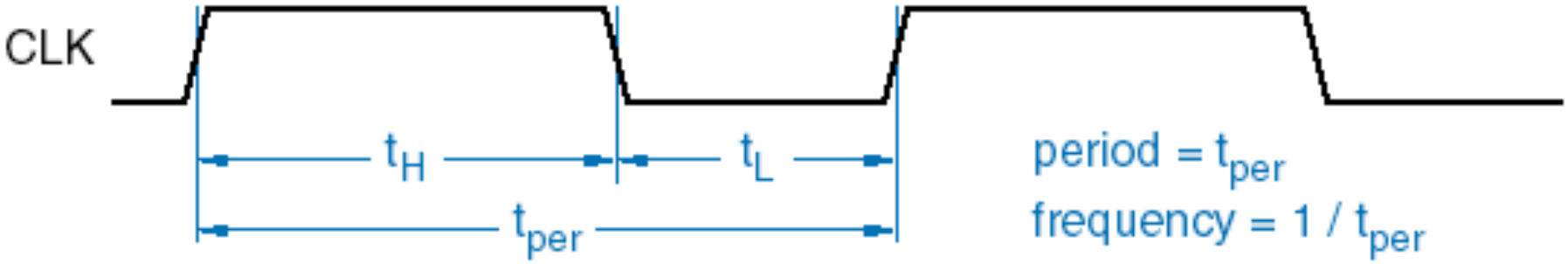
- An input to a sequential circuit that changes output and state values at a predetermined rate



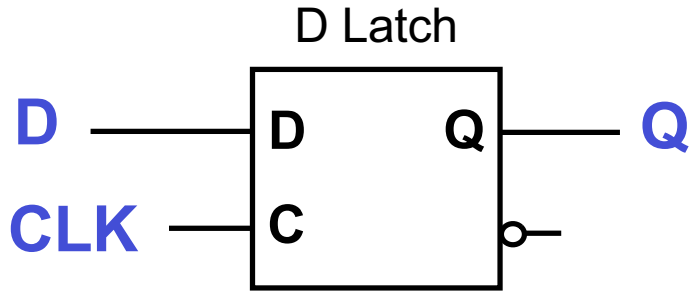
- Triggering edge: Transition of the clock that captures input data
  - By default, we use positive rising edge (L→H) as the triggering edge in this course
  - Clock tick: Occurrence of a triggering edge

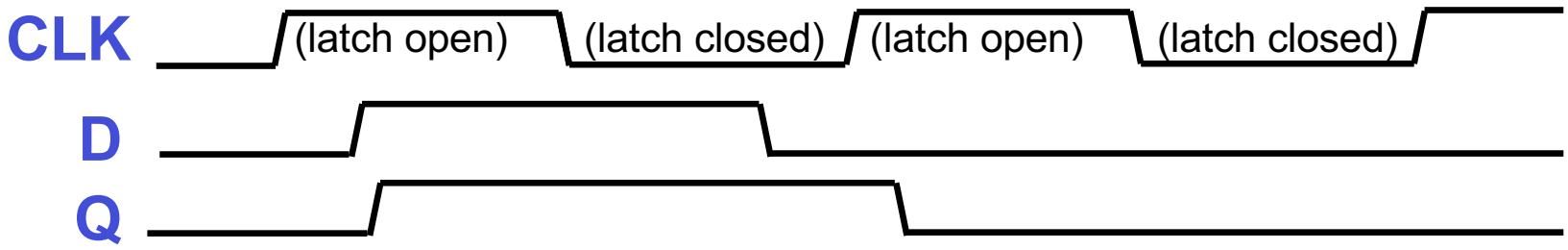
Clock Period and Frequency

- **Clock Period (cycle time): Time between successive transitions in the same direction (e.g., L→H)**
  - **e.g., 1ms, 2ns, 500ps**
- **Clock Frequency: 1/period**
  - **e.g., 1kHz, 500MHz, 2GHz**



D Latch Timing



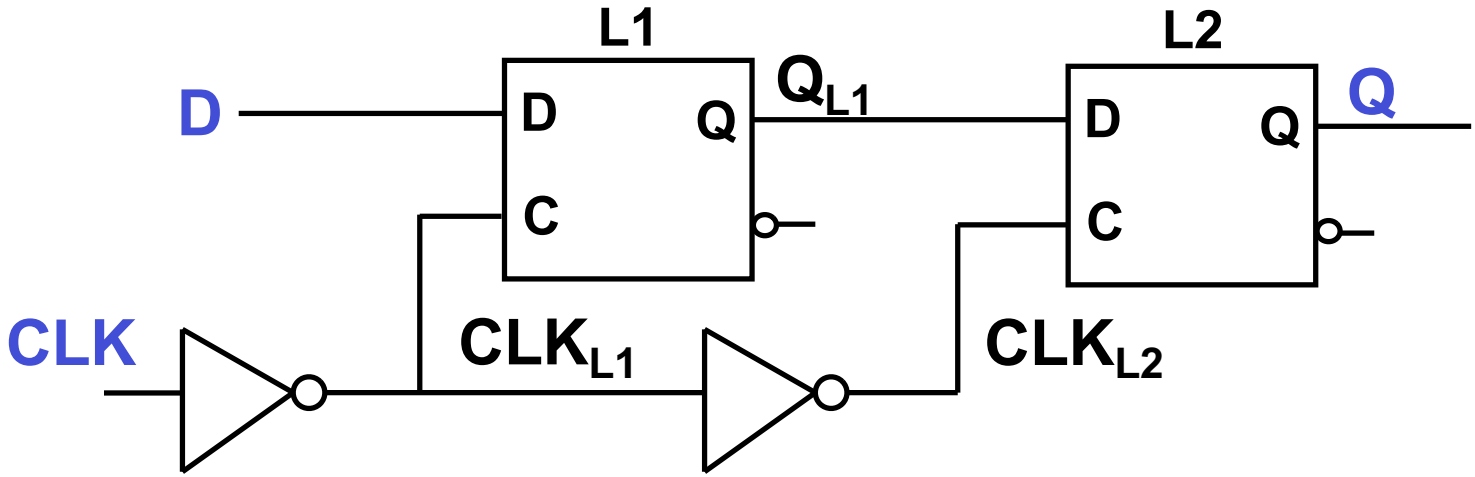


- **D latch is level sensitive**
  - **Captures input D when the latch is enabled (i.e., latch open when clock is high)**
  - **Holds the previous state otherwise (i.e., latch closed when clock is low)**

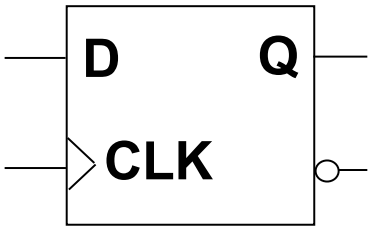
Flip-Flop

- An edge-sensitive storage element
- D flip-flop (FF): Two D latches back-to-back
  - Inputs: D (Data) and Clock
  - Output: Q
- Captures input (copy D to Q) on *triggering edge* of clock
  - Otherwise (when clock steady at 0 or 1) it retains its value

D Flip-Flop

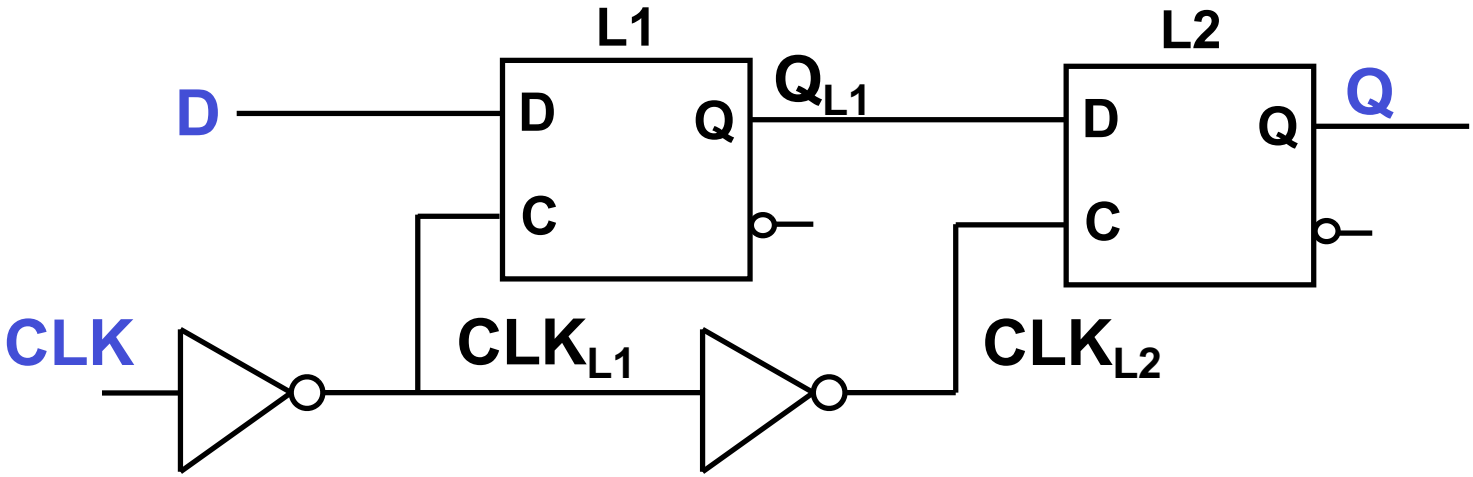


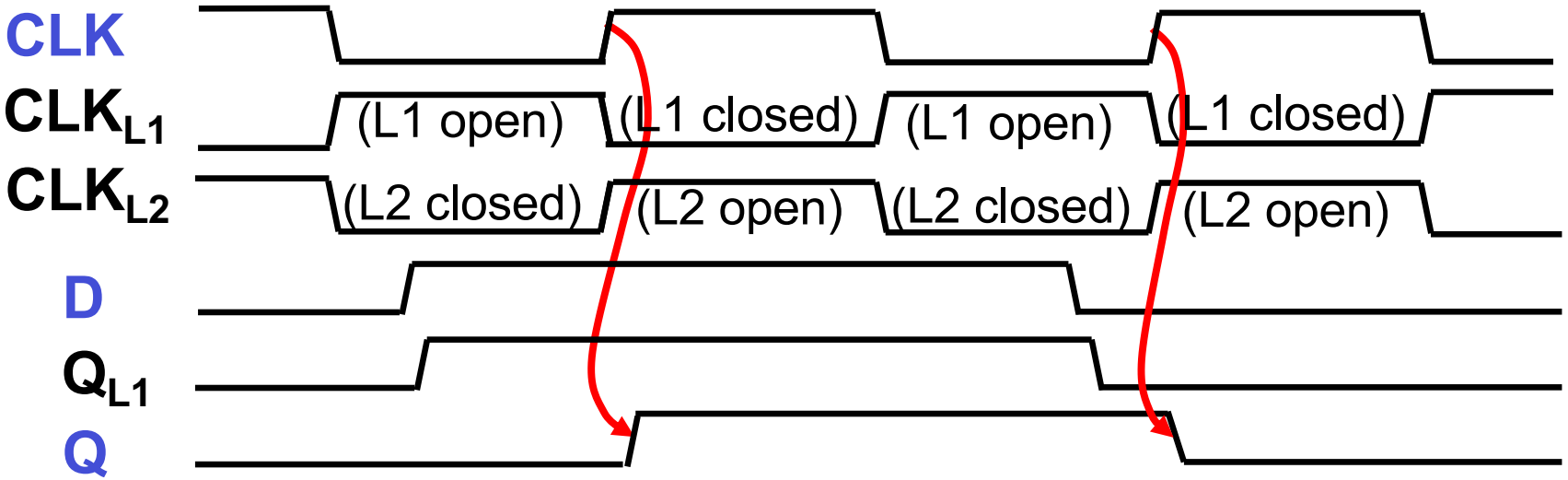
D	CLK	Q_{next}
0	↑	0
1	↑	1
X	0	(Last) Q
X	1	(Last) Q



Q_{next} is new state of Q after a triggering clock edge occurs (rising edge in this case)

D Flip-Flop Timing

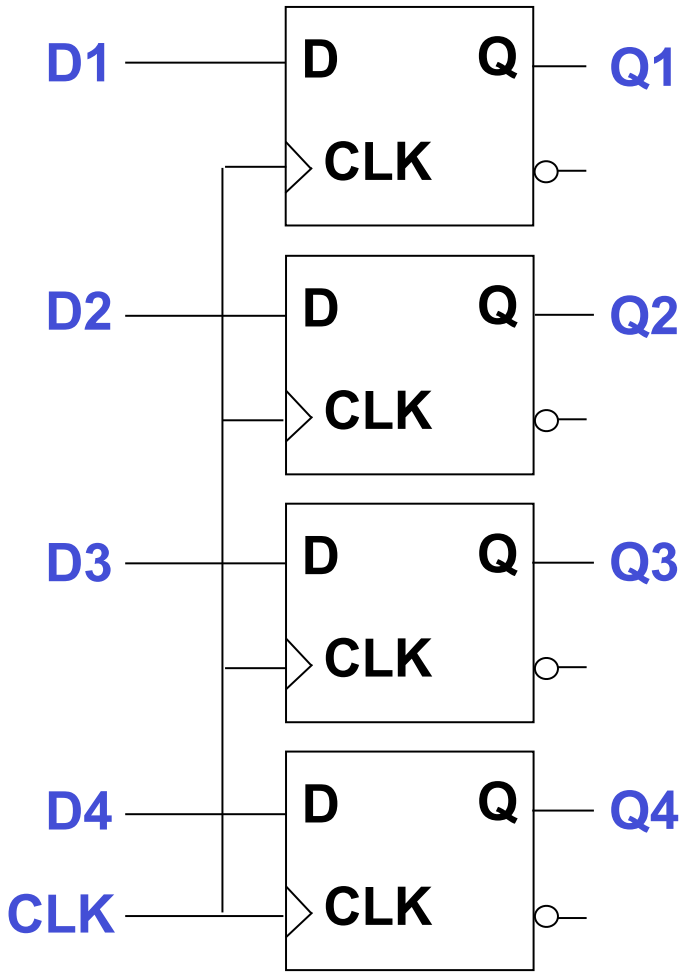




Input D copied to Q on the rising edge of the clock

Register

- Collection of FFs operating off common clock
- A single D flip-flop is a 1-bit register



Next Class

More Sequential Logic
Verilog
H&H 4.1-4.5 (skip VHDL parts), 5.4